Creating Passionate Users

By Kathy Sierra
with Eric Freeman and Dan Russell

Edited by Kevin Conroy

A Lulu Book

Text set in Garamond

Printed in the United States of America
A Lulu Book

www.lulu.com

The following comes from Kathy Sierra's blog, "Creating Passionate Users" at http://headrush.typepad.com/

This book had originally aimed to include every posting made, but since Lulu is unable to bind a book with 1,320 pages, we've had to "trim the fat" to get down to the 740 page maximum.

The posts in this book span from the blog's start in December 2004 thru April 2007.

Getting past the brain's crap filter.

By Kathy Sierra on December 22, 2004

Your brain didn't come with a manual. And that sucks. Before we started the Head First series, my partner Bert and I spent years studying ways to get learning into someone's brain, and the more we learned about the brain, the scarier it got. Because in so many ways, Your Brain Is Not Your Friend. It thinks you're still living in a cave, and it's sole job is survival of *you* as a human, and survival of the species. And what IT thinks is important and what YOU think are... really different.

Learning a programming language, it turns out, isn't high on the brain's list of Things To Keep You Alive. You know this, of course, because you remember the feeling -- you're in college, finals are tomorrow, and you're cramming to within an inch of your life. But you find yourself reading the same page, maybe the same paragraph, over and over and over and over barely able to stay awake. The illegal dose of caffeine isn't working. But then the hot babe from the next dorm walks by and suddenly you're alert, coherent, energetic even. Your brain was doing a, "Hmmmmm... calculus or survival of the species... damn... tough choice!".

So we've been spending a lot of time thinking about how important it is to get past The Gatekeeper (the brain's crap filter). If the brain is trying to save your life by keeping out the OBVIOUSLY unimportant thing like tomorrow's final, then how do you *trick* the brain into thinking the boring, dry thing is as important as that tiger that ate your ancestors?

All the studies seem to show that the center of everything is your amygdala--the almond-shaped organ (actually one on each side of your brain) that responds to things that might pose a threat or help you in some crucial way (and it does some of this

without your conscious awareness). If your amygdala were programmable, you'd tell it to PLEASE treat a grade less than C on tomorrow's exam as LIFE-THREATENING, and could you PLEASE pay attention and record this to long-term storage. But you can't. Or can you?

There *is* a way to program it, kind of. The inputs that tell your brain that something is important and worth recording are *feelings*. You pay attention, and record, that which you feel, because the brain is paying attention to the chemistry associated with emotions. When you see a tiger (in the wild, not a zoo), your brain recognizes the threat and chemicals surge. Your brain says, "This is REALLY important -- so remember EVERYTHING." If you've been in a car wreck, you might know the phenomenon where you remember *everything* including the background details like which song was playing. Because your brain did a complete snapshot of the whole damn scene, knowing that this was a Very Bad Thing, but not knowing which parts were important--so it said, "What the hell -- I'll just save it all."
(And I'll talk in a later blog about why your brain reacts differently to the tiger in the zoo than in the wild... it's another really cool thing the neurobiologists have learned).

So the question again is, "how do you get the brain to treat, say, learning Java as though it were potentially life-saving?" We use this in our books to try to help people learn more quickly and more deeply, and with a more lasting memory (because we write on difficult technical subjects, and some of our books are certification exam guides as well, where memory is crucial).

But then we started to realize that it applies to marketing as well...that the principles we use to increase attention and memory for the purposes of learning are the same principles you need to do what marketing guru Seth Godin says is essential today to break through--**Be Remarkable**. If you want people to talk about your product or service, it better be something really worth talking about. And today--with conventional advertising on its last legs--it's harder than ever to break through and be heard. Your users (or potential users) are so overwhelmed with messages (99% crap) trying to compete for their attention, that their brains are working overtime trying to keep those messages OUT. Remember, the brain wants to conserve bandwidth for the really important things... snakes,

spiders, the fact that fire is hot, that socially you need to do a little better so that you have a hope in hell of sleeping with... that sort of thing. Their brains are NOT scanning for an FAQ of how your product is technically superior or logically a better choice or... pretty much anything related to the features of whatever it is you're trying to sell.

So, that was the first thing we learned about the brain--how the crap filter really works and how to get past it. In later blogs, I'll go into a lot more detail about that. But we learned a lot more about how to get--and keep-someone's attention, some of which I taught at UCLA Extension in the mid 90's at the IBM New Media Lab (and used during my days as a game developer). We've been doing a lot of experimenting including during my time as a Java trainer/evangelist for Sun Microsystems, and later with the creation of the new series for O'Reilly. The books have all become overnight bestsellers in their category, and since we *know* we aren't very good writers, we're pretty sure it's because we spoke to the reader's BRAIN, not the reader himself. We believe that talking to your customer/client/user/prospect matters less than WHICH part of them you talk to.

Bert and I are working on a tutorial we're presenting at ETech on Creating Passionate Users based on a session we presented at the last two Foo Camps, and we've finally decided to work out the details in a blog. We'll use this space to work on our "Creating Passionate Users" tutorial (and we're also doing a book on this), as well as talk about new things in the Head First series and an interactive learning site we're working on. Our passion is the brain, but we'll talk about the core elements we believe you need to inspire customers/users including lessons learned from cognitive science, psychology, video/computer game design, entertainment (Hollywood), and yes, even advertising still has something to say (although advertising no longer works well, it still holds the key to some of the things that DO work... more later).

So... we don't know where this will go, but we'll do our best to give as much as we've been getting from the contributions of so many others on the web.

http://headrush.typepad.com/creating_passionate_users/2004/12/how_well_do_you.html

If some people don't HATE your product, it's mediocre.

By Kathy Sierra on December 23, 2004

That notion is shared by lots of folks including Don Norman, and we've taken it to heart, given how many people *hate* (with a passion) our books. But if you take the safe path, your chances of breaking through the market clutter is almost zero. There's simply too much competition for virtually *everything* today--and with so many choices of products, services, whatever, making a dent requires something dramatic. Dramatically different. No, not just different, but *different in the ways that matter to the user*.

Seth Godin has a comment (I think from the Purple Cow book) that goes something like this, "Today, being safe is risky, and being risky is safe." (I might have mangled that, but that's the idea.)

So we had a choice with our first book series (the Head First books): do we play it safe and write a good, solid, decent programming book that the widest range of readers can appreciate, or do we completely abandon the conventions in favor of something that many would HATE, but that would be a dramatically better experience for others? If we played it safe, we knew we'd have to kick and scream and claw (and the publisher would have to spend a fortune on marketing) to take even a tiny piece of the market already well-served by at least a dozen well-respected, technically excellent books. But if we took the unsafe path, we risked getting viciously BASHED in public (just *imagining* what Slashdot would do to us was made me ill) and losing whatever minor reputation we'd built with javaranch. But really, we had almost nothing to lose. It was O'Reilly that really took the Big Risk (more on *that* story in another post), since they DID have a reputation at stake... a reputation crafted over many years and with thousands of books.

So we thought about it for around five seconds and decided to go for it ;) We took Seth's advice and chose the risky-is-actually-safer road by questioning nearly every assumption about The Way Things Are Supposed To Be. Instead, we asked, "If there were no constraints other than the ones imposed by the 2D

page/book format, what could we do to help people learn better, faster, deeper?" We knew a lot about how to answer that question (from years of research and experience working on it), but we also knew that some people would hate it. REALLY hate it. We just crossed our fingers (as did O'Reilly, thanks mainly to Tim's personal pleasure at being disruptive) and hoped that just a few more people would LOVE it than hate it, and that the people who really loved it would care enough to spread the word.

And thankfully, that's what happened. Several things surprised us though:

1) More people loved it than we expected. Head First Java went immediately to the top of the Java bestseller list in the US (across both online and brick-and-mortar stores, according to Bookscan), was a finalist for a Jolt Cola/Software Development award, and was chosen a Top Ten Computer Books of 2003 by Amazon), and stayed on top for 18 months until it was replaced by our Head First Servlets book (which was selected as an Amazon Top Ten Computer Books of 2004). The other two Head First books became instant bestsellers in their categories as well. We could not possibly care more about what our learners have to go through to learn this stuff, and that caring and extra effort (these books are much more difficult and time-consuming to build) is making a difference.

2) Of the people who hate it, the most vocal have been other computer book authors. We chalked this up at first to a simple Who Moved My Cheese thing, but later realized it isn't that simple. We now believe that a lot of it has to do with defining what a "book" is... and that most of the computer book authors were *writers*, and many of them damn good ones, who saw our books as a degradation in writing, a kind of "pandering to the MTV generation". In many ways, that's *exactly* what our books are. But we don't consider ourselves *writers* and we don't consider our users to be *readers*. We consider them *learners*. And that means our job is not to *write* but to ***help them learn***. Another issue is that many folks believe that it is just unprofessional to put such "silly" things in a technical book, and that it shows disrespect for both the learner and the professional topic. We violently disagree, of course, because everything we do in the books has a very specific purpose based on reaching the brain, and we're very passionate ourselves about both the topics

and the act of delivering them, but that's a more involved topic I'll look at later.

3) Of those who started out hating it, some later found it to be "an acquired taste", and some of our initial vocal haters later became vocal supporters.

4) Of the folks who hate it, most (but certainly not all) are *not* in the target audience. In other words, they believe it's bad "for others", rather than evaluating it as someone actually trying to use it for its intended purpose. In other words, they believe they're speaking on behalf of the people who really ARE in the target audience. So we get a lot of comments like, "How can ANYONE learn from this crap?" from people who already know the topic.

5) A surprisingly vocal group hated it *not* because of its format, but because its very premise--making it easier to learn Java--was just BAD. Bad for the tech industry. Bad for the existing Java programmers. Bad because it would allow those who "don't even DESERVE to learn Java to start taking our jobs". We dismissed that as ridiculous at first, but then we heard that a few other authors of beginning Java books had experienced the same phenomenon. One well-known author of an excellent, but very advanced Java book, put it this way, "I guess it makes sense that your book would be successful now... all the SMART people already KNOW Java."

What did NOT surprise us was that the audience for the Head First books is skewed younger. People with brains wired up in the 60's and 70's are more likely to find our books [euphimism]*unpleasant*[/euphimism] than those wired up in the fast-cut visual sensibility world of Sesame Street/MTV/Video Games. This is not 100% (and let's just say that Bert and I grew up when Pac Man and Space Invaders were considered "stimulating media"), but we used the research that points to differences in brain wiring and visual perception between those raised on slower media and those raised on, well, *the faster stuff*. I'll talk more about those brain differences in other posts.

We couldn't possibly be more supportive now of the "be risky" and "embrace the hatred" model for launching a new product or service. Because let's face it--getting people to choose your excellent-but-mainstream product over all the other excellent-but-mainstream products that serve the user's needs is an uphill

(if not impossible) battle today, and even if it's possible... who has the marketing budget?

So take a chance, and be brave. My skin isn't as thick as it needs to be... when people trash us, for good or lame reasons, it hurts. But it's worth it. We offered this series to two other publishers (more on that in another post) and they didn't just turn it down, but turned it down with impunity...*laughing* as in, "Oh, like THAT is going to work! Ha-ha-ha...". I was virtually fired from Sun for some of the ideas that proved to be most-liked by customers (in other words, Sun said, "Customers will hate this... shut up about it OR ELSE."). But thankfully for us, O'Reilly was more than willing to take a chance (although rumors abound that it caused quite an internal battle at O'Reilly--with Tim and Mike Loukides and Kyle Hart on one side, and many, many others suggesting that Head First books would seriously damange their reputation).

As my partner Bert likes to quote, "If you aren't living on the edge... you're taking up too much room." (Of course, we were lucky enough to have O'Reilly footing most of the bills for this *experiment*, but still...the first book took six months of pouring our heart into it, day and night, for both Bert and I). To all of our early adopters and vocal supporters, THANK-YOU! We owe you so much, and when you take the time to tell others -- and even better, to tell US -- what the books have meant to you, that makes it so worth it. Computer books are not a way to make a good living today because the tech book market is down so far today, but the emails from happy user/learner folks keeps us going.

http://headrush.typepad.com/creating_passionate_users/2004/12/if_some_people_.html

Learning isn't a push model

By Kathy Sierra on December 26, 2004

Back in my AI days (when I used to have a clue, or thought I did anyway ;), the book Scripts, Plans, Goals, and Understanding was required reading for some of us. And I've been a fan of Roger Schank ever since. Of all the work that has influenced the direction of our learning principles, his has had the greatest impact. We were thrilled to see his work on intelligence move toward developing better theories (and tools for) learning.

We get letters from people who want to know more about the metacognitive topics we cover in the intro to the Head First books, and I'd suggest that anyone interested in learning theory put his e-Learning book high on your list. Maybe even at the top.

Some consider him an acquired taste, and he has a lot of detractors. He's one of the more outspoken critics of the education system in the US and slams just about everything from secondary schools to colleges to corporate training. A typical quote of his from the book, discussing corporate training:

"So what's wrong with training? Everything that's wrong with training can be stated in four words: *it's just like school*. The educational model in school does not work. That fact, however, hasn't deterred business from adopting this model, which is based on the belief that people learn through listening. Memorize the teacher's words; memorize the training book's policies and procedures. It's at this point in my public talks that audience members rise up in protest."

And one of my favorites:

"First and foremost: When learning isn't engaging, it's not learning. The movies, for all their faults, usually get this idea right. In the film *Dead Poet's Society*, Robin Williams plays a teacher who jumps on top of desks, makes the class laugh, tells great stories, and gets the class involved in what he's teaching. The educational establishment at the school hates the way Williams teaches, based on the premise that if students are having fun, something must be wrong. Listening to lectures and memorizing countless facts and figures aren't engaging activities for most people. Minds wander; real goals take over."

Another book that has some good research data is this one by long-time learning guru Ruth Clark.

Yes, both of these books happen to be focused on e-learning, but the principles apply whether you're doing classroom training, learning books, online training, or developing just-in-time performance support materials.

But regardless of differences among learning theories, one thing virtually *everyone* agrees on is this: **Knowledge cannot be pushed into someone's head while they sit passively reading or listening. Knowledge is a co-creation... the learner must construct the new knowledge in his own head.** And usually (or some say ALWAYS), the new knowledge must be mapped into something that's already IN the learner's head.

This notion of knowledge-as-co-creation is crucial for us. Which is one of the reasons we were horrified at the thought of creating learning books. Because for the most part, *reading* is just like *listening*. Worse, reading a fairly dry text book is like listening to a dry lecture-- pretty much THE weakest form of learning. So trying to make a book into a learning experience flies in the face of everything we believe in about learning (our backgrounds are in computer science, game development, entertainment, teaching, and looooong stints in artificial intelligence including the field of intelligent tutoring systems (AI meets CBT)).

So our mission was, given the constraints of a book format, and knowing that learning is far less likely to happen if the learner is just... *reading*, what can we do to help get them involved and start *flexing their brain cells*? So we tried a bunch of different things, figuring that the more we can throw at it, the more likely it is that *something* will work at least *some* of the time, for the people who try to learn from the book. From the feedback we've gotten now, 18 months in, we know that some of what we did to help make this happen is working, and some of the things just bombed. And some of the things we didn't plan--that are there simply as artifacts of trying to apply some *other* principle, turned out to be a key component to getting the learner involved.

Getting the learner to co-create knowledge isn't a simple task; lots of pieces have to come together. For example, if we provide the absolute best thought-provoking exercises, but can't get the reader to stay awake long enough to get to them, we lose. If we

provide ways in which the reader can get involved and really build some brain cells, but the challenge level is simply too high (or just as bad, too *low*), we lose. If the material simply isn't stimulating enough to hook the reader in, and he won't stay with us start to finish, we lose (since we're not a reference book). In other words, our whole premise and promise to the user--that they'll learn more quickly, more deeply, and with less pain--depends on them *really staying with it* and doing the work. And we're painfully aware that if we don't deliver on that promise, they'll have no reason to ever buy another book in this format. We're in this for the long haul, so we're deeply dedicated to really making this work for people who want to learn.

We looked at the whole system in which an environment for learning occurs, and that's why we drew on so many domains to help us. Learning theory says the learner must be motivated, but says almost nothing about how to *get* people motivated. Learning theory says the learner must be engaged and involved, but says almost nothing about how to really make that happen. On the other hand, Hollywood, Madison Avenue, and good sales people have something to teach us about providing motivation. They understand seduction and communicating *personally meaningful benefits*. (More on that topic in other posts.) Game designs have something to teach us about that as well. After all, kids and adults alike will spend hours and hours and hours immersed in thinking/planning/strategizing in the course of playing a video game.

We wondered, can we try to turn a technical book into something that will make people *want* to stay with it and keep turning the pages? And even if we *can*, will they be motivated to actually DO the exercises? What if they *don't* do the exercises? Can we provide OTHER ways to try to make learning happen? To get the user to think at a higher level... to process the new content in such a way that he constructs new knowledge in his head, rather than just passively reading?

Our answer was, "We don't know if we *can*, but yes, we can certainly try, and here's how..." Among the things we use to try to get people flexing brain cells are:

1) We use cliff-hangers, where the learner is drawn into the scenario only to be left hanging without the full answer, to help spur their curiosity into speculating on the solution or result.

2) We use debates/arguments/discussions between two characters (which could be people or even anthropomorphized parts of the system like the compiler vs. the virtual machine) where there isn't always a clear winner. Both sides might make compelling, convincing cases for their personal view, and this kind of forces the learner's brain into evaluating (one of those higher-level thinking tasks on Bloom's taxonomy), weighing the merits of each side, and drawing his own conclusions. Sometimes we have a definite side, but looking at the same scenario from more than one perspective is in an of itself a way to help inspire deeper thought processing of the concepts.

3) Knowing that most people claim to skip the lab exercises in a book that say, "Now go to your computer and type this in...", we have 40-50 in-book (workbook style) exercises you do with a pencil, right inside the pages of the book. We want readers/learners to have NO excuse for not doing the exercise when they're using the book the way they tell us they do--on the train, bus, plane, in the park, wherever they have a spare moment at lunch, etc. In other words, when they're not necessarily within easy reach of doing the real thing they're learning.

4) We make those exercises use a wide range of brain capabilities--so there are right-brained pattern-matching exercises, left-brained code troubleshooting and logic puzzles, draw this picture, answer this question, write this code, make this decision, etc.

5) We ask questions and provide exercises sometimes for things that we haven't fully explained, so that the learner must *apply* what he's already learned and extrapolate to figure out something he hasn't actually seen yet.

6) We provide "garden path" scenarios, where the learner is led down a road that looks so right, but turns out to be SO wrong. This is based on the theory (Roger Schank has a lot to say about this as well) that we learn a lot more from mistakes and surprises than from things that work as expected. Think about it... what are the things you're most likely to remember when you're working? When things go just as you expect, just as you were told, there's nothing memorable. But when you're humming along and suddenly the thing you expect fails, and you get just the opposite... you get that WTF?? feeling. And ***that*** is what you remember. So we try to provide at least a few of those

visceral, "I won't make THAT mistake again!" experiences when they happen. (And thanks to the wonderful Java APIs, those doesn't-work-the-way-you'd-think counterintuitive scenarios are plentiful in some of our books :)

Of course it's a little tricky using these techniques. It makes our books suck as reference books, of course, but we're 100% clear that our books are learning experiences, not references. Because you might flip back to a page and actually find yourself *reading something that's just wrong*. So we have to use a lot of other things in our books to try to get people to read the whole topic, at least the first time through, in order to get the whole "Here I am doing this stuff and BAM! It blew up..." experience of the story. And there will always be some people who hate the approach precisely because it DOES includes these tricks. They feel cheated that we deliberately led them down a garden path when we could have just *told them how it really works to begin with*. And while they have a valid point, that "telling not showing" approach (considered a really bad thing to do among novelists and filmmakers) is the weakest form of learning.

You might hate the approach, but it's closer to the messy, confusing, oh SHIT experience of what happens when you do these things in the real world, and it's guaranteed to be more memorable. If you can stand getting through it. That's part of why we have to do a lot of other things to try to make that "getting through it" more tolerable, or even interesting. (That's where the game design theory comes in; more on that later, but here's a hint: the flow state psychologists call optimal experience that game designers know as "make them want to get to the next level by getting the challenge/skill/seduction blend right."

Ok, that's it for now. My apologies for spelling errors and typos. Repeat to yourself: this is NOT an article; it's a blog. I'm trying to get better at this...

http://headrush.typepad.com/creating_passionate_users/2004/12/learning_isnt_a.html

Is your message memorable?

By Kathy Sierra on December 29, 2004

To be remembered, it must be *memorable*.

Yeah, that's a DUH thing. But we're constantly amazed by how many people ignore it.

(Us included, but we're trying.)

But focusing on that one simple concept can make a HUGE impact on whatever you're trying to do-- teach, sell, whatever.

After years of research and speculation, the neurobiologists are finally unlocking some of the deeper secrets of memory, led in large part by the work of Nobel winner Eric Kandel. Bert and I attended a presentation of his on the neurochemistry of memory, and it was... memorable :) They can actually take a neuron (not from a human), large enough to see with the naked eye, stick it in a petri dish and... *teach it*. By altering the chemicals, they can put it in a state where it will never learn, and they can put it in a state where it learns after just one trial. (Read once, remember always.)

And they now know the agent responsible for my, um, less-than-A college final exam grades: CREB-2. Your brain is constantly doing a balancing act between CREB-1 (enables long-term memory formation) and CREB-2 (prevents it). It's all connected to protein synthesis, needed for encoding memories to long-term storage.

So if you're a teacher, trainer, author, advertiser... and you want to increase recall and retention, you're in for the fight of your life against CREB-2. Why is CREB-2 there? To save your life. Or at least your sanity. You obviously don't want to remember *everything*.

The big problem, of course, is that *you aren't in control of your CREB-2*. Your brain is making the decisions about what's important and what isn't. I talked about this a lot in one of my first blogs: Getting past the brain's crap filter.

How, then, do you get past someone's CREB-2 (crap filter)? How do you make something *memorable*? Exploit what the brain is tuned to pay attention to. Exploit what the *brain* thinks is important.

The rough part is that even when people TRY to tell their brain "this is important, this is important, this is important", the brain says, "no it isn't, no it isn't, no it isn't." So if you're trying to get people to remember something, the sad part is that even when they WANT to remember, it's not guaranteed. You know this, of course, since you've all tried to remember things you read and study, but it just doesn't happen the way you'd like or even *need*.

So what *does* the brain remember? There are two main roads to memory--the slow painful one, or the much faster one. The slow painful one is through repetition. Repeated exposure (or what Kandel and others call "trials") eventually works. It's as if your brain says, "this sure doesn't FEEL very important, but he's read this damn paragraph 17 times, so I guess it is..." The quick one is to use the chemistry of emotions. Or as Roger Schank puts it:

"You remember that which you *feel*."

I'm really blending two things here--getting their *attention*, and getting them to *remember*. And they are closely related, because they're tied to triggering things the brain thinks is worth paying attention to. But I'm still mixing them more than is technically correct, because it is certainly possible to get someone's attention without getting them to remember, but for the most part, the distinctions don't matter. All I'm concerned about now is how to make the brain *care*.

And the key is to evoke *feelings*. The stronger the feeling, the more likely the brain is to pay attention and record what's happening. If you register a big flatline on the emotional richter scale (as you would during a dry lecture or reading a dull text book), your brain takes that as a perfect sign that "this is SO not life threatening."

There's a catch, though. Because intense feelings of *stress* also act as a memory suppresor. So it can't be just *any* feeling, but most will do the job. That's why putting people in a learning situation where they're feeling stressed, pressured, incapable, overwhelmed, stupid, etc. just compounds the problems they already have trying to memorize the stuff you're teaching.

OK, so what kind of feelings can I use? ANYTHING ELSE! Not everything is appropriate, of course, but anything I can get away with could work. Anything that causes a chemical shift, however slight, is an improvement.

That means humor, shock, horror, surprise, delight, joy, sex, thrill, etc. The problem today is that there's already so much of that, especially as advertisers try to break though the noise when the noise level today is already so high. It takes a LOT more to, say, *shock* someone than it did even ten years ago as people become desensitized. But context matters. In Colorado Springs, CO, I'd be shocked to see a billboard with a naked person on it. But when I worked in Hollywood, I wouldn't even *notice* the posters, billboards, store displays featuring naked people (often of uncertain gender) selling everything from shoes to software, because they were so common.

A racy scene in even a mainstream novel isn't too surprising (although often still memorable), but in a programming text book, even the slightest *hint* is unexpected. And sometimes *unexpected* is all you need.

The brain is highly tuned for novelty. It spent thousands and thousands of years scanning for the unusual, the moving, the changing, the doesn't-quite-pattern-match. USE THAT. The brain is tuned for sex... (like I had to actually tell you that :)), so USE THAT. I was about to add the requisite (where appropriate of course), but then... using where it is NOT quite appropriate works even better. Again, if you can get away with it. Please don't give me a morality lecture... I'll assume that everyone is using good judgement with respect to children, sexual harrassment, etc. I'm just talking about how the brain works, period.

The brain is tuned for things perceived as scary or threatening. USE THAT. (Although that one is a little riskier, because too much stress leads to the opposite effect). Shock and surprise are great, though. Again, anything you overuse will dimish its effectiveness, so the more variety of brain-triggering techniques the better.

In other blogs, I'll focus in on individual techniques. But of all the approaches to getting past CREB-2, the one that might be the best and easiest in most situations is simply "novelty". In other words, "don't do what is expected in that context." I think that'll probably be my next post...

Note to our authors: expect me, Bert, Eric, and Beth to be grilling you on what you're doing to get past the CREB-2. Even just a little cleverness, something just slightly off-center, something ordinary in one context but a little bizarre in

*this*context, or anything that elicits even the slightest head tilt or slight smile can be a big improvement in a technical text book, so it doesn't take a lot.

If you're an advertiser/marketer, on the other hand... wow. That's more of a challenge. On the other hand, people are so used to (and tuned out to) bullsh**, that simply being brutally honest (once they stop being cynical that you're just PRETENDING to be honest) is a major out-of-context experience that will work. If everyone finally gets on the Hughtrain, though, that'll only work until it's become the norm. (I doubt that'll ever happen, but the world would be a much better place if it does!).

If you want something to be remembered, CREB-2 is the moat you gotta get past. *Shock on.*

http://headrush.typepad.com/creating_passionate_users/2004/12/is_your_message.html

Getting what you expect is boring.

By Kathy Sierra on December 30, 2004

Otherwise known as the "Oh Shit!/Oh Cool!" technique.

Earlier I blogged about how the brain is tuned for novelty, but tunes *out* that which is common or expected.

Some of the areas where this matters include training, filmmaking, advertising, and I suppose *dating*. Director/writer David Mamet says that the prime objective of a director is to present a story that is "both surprising and inevitable at the same time." Kind of the "OH!!" followed by "Oh... *of course*..." feeling.

AI and learning guru Roger Schank puts it this way in his e-Learning book,

"A good course must enable failures that surprise the student. Failure is the key to learning. We have to work hard to recover when things don't work out the way we expected...For this natural learning process to work in a course, the course must surprise its students. But, more than that, it must put students in a situation where they are entertaining predictions in the first place."

And from an article titled Information is Surprises:

"Information is surprises. We all expect the world to work out in certain ways, but when it does, we're bored. What makes something worth knowing is organized around the concept of expectation failure."

At Sun, we used to have a lot of battles over the evaluation form that customers filled out at the end of a training course. Instructors *hated* the fact that customers ranked things based on "Meets Expectations" (including the two ends of the expectations scale, "below expectations" and "exceeds expectations"). The instructors wanted it to be based on something less subjective, or at least customer "satisfaction", believing that a measurement of the quality of their work should not be tied to the customer's expectations.

But the business folks like Tom Peters and Seth Godin tell us that when it comes to things like word of mouth, expectation is EVERYTHING. I don't have links handy, but there are plenty of

studies that show when someone's expectations are met, they won't talk about it... even if they believe that what they got was awesome! Even if expectations were high, everything is as it should be when they're met.

People talk about things that are surprising, or that really suck.

You talk about the waiter who went way above the call of duty. You talk about that movie that Ebert gave four stars but that you thought was one long and painful cliche.

So wake up the brain and do something surprising or at least unexpected *for a given context*. If you're a teacher, trainer, or authoring learning materials, for frick's sake ***don't have all your exercises, lessons, and stories simply confirm what the learner expects!***. If the learner spends a half-hour doing an exercise that does *exactly* what you said it would, that's valid practice, but not memorable.

Sometimes they *need* the practice and reinforcement, of course, so it doesn't mean you won't include the "confirmation" activities. But when you want them to really *learn* and remember something *new*, look HARD for opportunities where things don't work as expected. Places where something behaves counterintuitively, or radically different from something that appears (at least on the surface) similar are golden.

I've already talked about ways we try to use this in the books:

* Garden paths (things that look like sound approaches, but then blow up at the end).

* Counterintuitive examples.

* Using show-don't-tell on common mistakes.

* Examples that have a common framework, but often with a weird twist.

* Unusual visuals and metaphors.

I worked in the mid 90's for one of the coolest new media companies, AND Interactive (later sold to TCI and then brought down in a spectacular flame-out) co-founded by Hollywood creative Allen Debevoise. My projects were managed by another Hollywood producer, John Valenti (yes, Jack's son), and the thing I remember most about them both (and John especially) is that the WORST thing you could do is be "on the nose."

The classic example for me was when I was developing an interactive CD-ROM of Oracle's annual report. It had lots of splashy graphics and video, etc. but I committed pretty much THE worst possible offense when I chose, for the financial section splash screen, a graphic of a cash register. But then, why stop with just ONE cliche...I went ahead and added a really cool sound effect of a cash register ch-ching!

That John ever let me enter the building again is a mystery, and the warning to never EVER be "on the nose" ever again, still haunts me. And I've noticed that the phrase "on the nose" is now a popular way to criticize screenplay dialog that not only violates "show don't tell", but eliminates all possibility of subtext. (I think "thou shall not be on the nose" is one of McKee's ten commandments of story)

So just in case you needed one more reason to be surprising, unexpected, or simply weird--you can say it's just being brain-friendly. Yet another way to get past the brain's crap filter.

And if you're one of our authors, you can expect us to be looking for ways you've encouraged the "Oh Shit!" (yikes, can't believe it did *that*) experience, or the "Oh Cool!" (wow, that's amazing... I didn't know you could do *that*) feeling. :)

http://headrush.typepad.com/creating_passionate_users/2004/12/getting_what_yo.html

Users shouldn't think about YOU

By Kathy Sierra on January 3, 2005

Do you care what your users think of you?

STOP IT.

Our best advice for creating passionate users is:

Care ONLY about what your users think of themselves as a result of interacting with your creation.

That's a major shift for a lot of people, especially our tech book authors (and instructors). It's so natural to write with a critic sitting on your shoulder representing the person who isn't even in your target audience anyway, slamming you for leaving something out, or not being technical enough, or not *proving how smart you are*. I have a little story about this...

One of my jobs at Sun was to help raise the customer ratings of the Java instructors--to help instructors find more strategies for making every student/customer happier with the classes. A big mystery was why some of the most technically proficient instructors, who really knew their stuff and were good at delivering it, were getting average scores in after-class surveys. Meanwhile, the technically stronger instructors were pissed off that some of the less-competent instructors were getting fantastic scores.

The typical response was, "The instructors getting the good scores are just better entertainers. The post-class scores aren't a good reflection of what's REALLY important--*delivering technically correct and advanced material*." They'd complain that there was no line item on the survey that measured the things that mattered like, *"Does the instructor know the material?"* or *"Is the instructor competent with the technology?"*

See the disconnect? The instructors wanted the scores to be *all about them*. And that's the problem. It's the same one we have sometimes with tech book authors. What we tried to tell the instructors was this: "Most of the time students don't CARE how smart you are. They come in assuming that you're *supposed* to be here, so stop trying to prove how smart you are and get on with helping *them* get smarter."

The instructors didn't have an ego problem; they were absolutely convinced that students were coming to class to *hear from an expert*.

So I decided to conduct an experiment. One of the first things instructors were originally told to do was "establish credibility". Many believed that the longer and more detailed the instructor's resume, presented during introductions, the more receptive--and confident--the students would be. I'd audit classes, and sure enough, the instructors devoted a lot of time at the beginning of the class to introducing themselves. (Followed by a brief moment where the students each had a turn to say their name and a one-sentence self-intro.)

I was determined to prove that the "establishing credibility" thing was not just unnecessary, *it was a harmful misconception*. I had evidence that students come IN believing you're credible, and as long as you don't do something to screw that up, you don't need to convince them. In other words, you start the class with a pre-established credibility balance. Points will be deducted if you do or say something stupid, and most especially--if you get caught LYING by pretending to know something that you don't, or failing to admit when you're guessing. What I needed to prove was that by working hard during the class to make sure the students know how smart *you* are, you have a negative impact on the students' experience. They end up recognizing how smart you are, sure, but *that's not why they took the class!* They took the class so that THEY could be smarter, and with very few exceptions, they couldn't give a crap about you.

To prove this, I took it to the extreme in my own Java classes--*I stopped introducing myself completely*. At first, I simply cut down my own introduction, while simultaneously increasing the time devoted to *their* introductions. But eventually I went all the way and simply walked into class and started, without ever saying my name or anything at all. I just jumped into having them introduce themselves, and then we were off and running.

Somewhere, usually during the first day of class, some student would ask my name because they needed to ask me for help during a lab. When that happened, I would walk over to the board and say to everyone, "Oops -- sorry, guys--I forgot to tell you my name", and I'd write my email address on the board (which was then kathy.sierra@sun.com). So even when I *did*

give them my name, it was in the context of a way in which they could contact me for help. I made even my own introduction about *what it meant to them*.

The I-don't-matter-so-don't-introduce-myself plan was just the beginning of the "it's not about YOU" experiment. I would conduct the rest of the five day course with ALL of my energy devoted to making THEM smarter, rather than trying to make sure they knew how smart *I* was. (A clever and necessary strategy on my part, since I'm NOT all that smart ;)

The year-long experiment was a success, and I won a nice bonus from Sun for being one of only four instructors in north America to get the highest possible customer evaluations. But what was remarkable about this is that this happened *in spite of my not being a particularly good instructor or Java guru*. I proved that a very average instructor could get exceptional results by putting the focus ENTIRELY on the students. By paying no attention to whether they thought I "knew my stuff", etc.

And when I say that I was "average", that's really a stretch. I have almost no good "presentation skills", and when I first started at Sun I thought I was going to be fired because I refused to ever use the overhead slides and just relied on the whiteboard (where I drew largely unrecognizable objects and unreadable code). But...I say "average" when you evaluate me against a metric of traditional stand-up instructor presentation skills. Which I believe are largely bull**** anyway. Assuming you meet some very minimal threshold for teaching, all that matters is that you help the students become smarter. You help them learn... *by doing whatever it takes*. And that usually has nothing to do with what comes out of YOUR mouth, and has everything to do with what happens between their ears. Of course this means you, as the instructor, have to design and enable situations that cause things to happen. Exercises, labs, debates, discussions, heavy interaction. In other words, things that THEY do, not things that YOU do (except that you create the scenarios).

Remember that learning isn't a push model.

One important question our authors and instructors ask sometimes is, "how do you KNOW when you're successful at this?" That ones easy--*users will talk about themselves, instead of talking about YOU*. To check how we were doing with our Head First books, we did an analysis of just under 200 Amazon

reviews of our books, and compared them against the Amazon reviews of our closest competitors. We looked only at the positive reviews, and we looked only at books that had similar ratings of 4 to 5 stars.

What we were looking for, and found, was a simple reflection of the "it's not about YOU" concept. What we expected, and found, was this:

When compared to our competitors, fewer readers mention how smart we (the authors) are.

When compared to our competitors, far MORE readers use first-person language to talk about themselves in relation to the content. In other words, they talk about THEIR new understanding, etc.

When compared to our competitors, far MORE readers use our first names in the review.

This last one might seem irrelevant for this discussion, but to us it isn't. We assumed that when readers/learners are in awe of the author/instructor, it sets up a more traditional master-student feeling, where the "teacher" is granted a level of respect, and that this would be reflected in readers' use of the author's last name rather than first. And this is exactly what we found. By not being ABOUT us, readers have had a greater tendency to see us as simply *helpers* rather than professors or gurus. That more casual relationship shows up in reviews.

It's about them, not us.

How does that drive what goes into our books? In a ton of ways discussed in other blog entries, but perhaps the most dramatic is in what we leave OUT. When an author or instructor is worried about whether he'll come across as smart, he'll tend to include things that get in the way by adding cognitive overhead. It takes a certain amount of bravery to leave things out, but by ignoring what critics will do to us, and thinking only about what's good for the learner, the decision is easy. If it doesn't support the learner, cut it. And that goes not just for topics, but for the kind of language we use as well.

Too many learning experiences and books leave the learner feeling impressed as hell with the instructor/author, but... *stupid*. Next time you read a technical book or take a class that's daunting and difficult, and you're starting to get that sinking

feeling that you're not smart enough... remind yourself that it's not your fault. That the instructor or author is simply proving their own technical prowess (in what we believe is a misguided attempt to help you), but at your expense. [Disclaimer: what we're saying applies to LEARNING books, not reference books.]

And when you take a class or read a book that leaves YOU feeling smarter, letting the instructor or author know the ways in which YOU have improved is the most wonderful thing you can do. To know that we've helped you better understand and do something new is the most motivating thing for us. And to all those who HAVE let us know, we cannot thank you enough. You are why we do this, despite the drastic drop in the tech book market. And when you write a review, please PLEASE talk about yourself, and not us. We want to hear what cool thing YOU are doing.

http://headrush.typepad.com/creating_passionate_users/2005/01/users_shouldnt_.html

Your user's brain wants a conversation!

By Kathy Sierra on January 5, 2005

Which would you prefer to listen to--a dry formal lecture or a stimulating dinner party conversation?

Which would you prefer to read--a formal academic text book or an engaging novel?

When I pose this question to authors or instructors, I usually hear, "You *think* the obvious answer is the dinner party and the novel, but it isn't that simple."

Followed by, "It all depends on the *context*. I'd *much* rather hear a dry formal lecture on something I'm deeply interested in than listen to inane dinner party conversation about Ashlee's lip-syncing blunder."

But here's what's weird--your brain wants to pay more attention to the party conversation than the formal lecture *regardless* of your personal interest in the topic.

Because it's a *conversation*.

And when your brain thinks it's part of a conversation, it thinks it has to pay attention... *to hold up its end*. You've felt this, of course. How many times have you sat in a lecture you *really* needed and wanted to pay attention to, but still found it hard to stay focused? Or how about the book you can't seem to stay awake for... finding yourself reading the same paragraph over and over because you keep tuning out--*despite your best effort to stay with it*?

But here's the coolest (and for me, the most fascinating) part of all this:

When you lecture or write using conversational language, your user's brain thinks it's in a REAL conversation!

In other words, if you use conversational language, the listener/reader's brain still thinks it has to hold up its end, so it pays more attention. It really is that simple, and that powerful (at least if you really want to help users pay attention and remember your message).

At least that the conclusion some researchers have come to, and you can read more about this in this book and this book. (Warning, that last book is by the guys responsible for Microsoft Bob, so... acquiring research and *applying* that research in a useful way can be very different).

For a long time, there was a rule (although nobody remembers who came up with it) that said computer books (unless they're "for dummies") must be written in a formal style. Possibly THE worst example of this for me was the editorial group at what was then Sun Educational Services--where the reasons for using formal language included:

1) It's professional. Formal language == professional. Conversational tone == unprofessional.

2) It's easier to localize.

3) It's more appropriate (whatever "appropriate" meant... we never knew for sure.)

I railed against #1 (one of the things that did, eventually, cost me my job there).

I also railed against #2, especially since it meant that in order to better localize, we'd need to suck out all remaining life that hadn't already been destroyed by reason #1.

It got so ridiculous that for a while that we were told not to use contractions, because "they don't localize well." While I'm sure they had valid reasons for making that claim, wow. My naive thinking is that anyone doing translations that can't deal with the contractions is going to have much bigger issues... and if they're using a machine-translation, YIKES! Even worse.

And here's the problem I have with not using contractions... what do science fiction writers do when they want to make sure you recognize (without being told) that a character is either:

a) a robot/android

or

b) an alien

Think about it. *They don't use contractions*. Dead giveaway every time. One of the only computer/AI characters who DID use contractions was Hal, and, well, he was psychotic. So I

should qualify the rule as "non-psycho robots, androids, and aliens do not use contractions."

So here we were--in the interest of localization--stripping all remaining humanity out of the material.

But that's the extreme example, of course. It's quite easy to write in a formal, non-conversational tone and still use contractions. And it's that formal, non-conversational language that causes a reduction in comprehension and retention and recall. (There's good data on this in the "science of instruction" book I linked to earlier in this post).

If an author isn't forced by the editorial police to formalize the language, why, then, do so many still use it in their learning books? I've asked this question of a lot of authors, and it usually comes down to a violation of the"users shouldn't think about YOU" rule. In other words, the author is considering how he or she--the author--will be perceived. One author put it this way to me, "I want people to see me as the serious kind of person who says, 'listen up because I'm only going to say this once!', where YOU, on the other hand, want people to feel, "hey, let's have some fun!". When I said, "this is a problem... why?" He made the valid point that unlike us, he was using his books as a means to further his consulting career, so what people thought of HIM as the author really did matter.

It was kind of like, "I don't want people to see me as someone they'd like to have over for dinner. I want them to see me as someone they'd like to *hire*." The number of arguments I could make about that statement could go on for days, but that's a different topic.

Other arguments are that by making the language conversational, we're not showing the topic--and the readers--the proper respect. I'll let you consider that one.

Another argument is that using conversational language makes it sound like a "dummies" book. That's potentially valid, but only because the "dummies" series was the first to really make a format dedicated to NOT using formal language. As long as the perception is that "dummies/very beginning books use casual language but advanced books do not", then that's a problem. We're trying to change this in our part of the world by claiming that the opposite could be true--that the more advanced the topic, the more you NEED to pull out all the stops in trying to

make it more understandable. And in fact, that's how it is with our books--*because* of our format, we're able to cover far more ground and dig into more advanced topics than a similar book using formal language.

I know this is horribly overgeneralized, but as a high-level rule, we believe:

If you're using formal language in a lecture, learning book (or marketing message, for that matter), you're worrying about how people perceive YOU. If you're thinking only about the USERS, on the other hand, you're probably using more conversational language.

Now, there are limits--and context DOES matter. How far you go in "conversational language" is a matter of culture and your audience. We use words that some believe are inappropriate in a technical book, including "sucks", and we wrote "wtf?" in the margin of one page. So far, we haven't used any four-letter words (although we'd occasionally like to), but our books have found their way into high school courses, and we haven't felt that the content *needed* or would benefit from those words.

But the research supports that you don't have to take it that far to get the benefits of "the brain tunes into conversational language because it thinks it's IN a conversation." Just rearranging a few words to be more casual and applying a readability index can help.

The tip we give our authors is this: when you're writing, paste up a couple pictures of real people, and imagine you're talking to *them* as opposed to writing for some abstract notion of "reader". Most importantly, **ignore the advice your high school writing teacher gave you**--that you must never "write the way you talk." Because from the brain's point of view, it is far *better* to write the way you talk. In fact, while it doesn't make for *great* writing to, say, print a transcript of a real conversation, that would still give you *better* learning material than something you wrote using passive, third-person voice in a formal tone.

And never underestimate the power of using "you" in your writing!

http://headrush.typepad.com/creating_passionate_users/2005/01/your_users_brai.html

Keeping users engaged...

By Kathy Sierra on January 8, 2005

So you got past the brain's crap filter, but now how do you *keep* their attention? How do you keep them involved? Take a lesson from game developers...

The more time someone spends with your "message", the greater the chances that they'll understand, retain, and be able to recall that message. So whether you're trying to help someon learn or get users to become more involved with your [product/service/website/music/cause/whatever], keeping the user engaged is crucial. But how?

By looking to places where keeping users engaged is not a problem--places where people *want* to stay involved. And that means looking to hobbies, sports, games... places where people are passionately addicted! Wouldn't we all love to have users that felt as passionately about our stuff as people feel for their favorite activities? Look at the characteristics of people who are passionate about something:

*** They want to learn as much as they can about it.**

* They want to connect with other users (user groups, conferences, clubs, online forums, etc.)

* They're willing to spend money to get the latest and greatest.

and sometimes most importantly...

*** They talk about it to others. They can't help being envagelists.**

So, the solution is simple:

Just make sure your product or service is as engaging as, say, snowboarding. Problem solved.

So what happens if your product is something just a tad *less* compelling... something like soap? Hugh at Gaping Void got a similar question when we wrote about how your brand makes the customer a more powerful entity. In a comment, someone asked, *"How can garbage bags increase my personal power, or how could you even pitch them in such a way? Power over not having the bags rip and spill stuff all over me? Am I missing something? I do see where this could apply in the technical arena, or with cars, or suits, or whatever. I'm not seeing that the Hughtrain has universal marketing power."*

I thought Hugh nailed the answer (or at least pointed to where the answer lies) with this:

"I think it is possible to be evangelical about a garbage bag.

But you need imagination and a sense of adventure."

So yes, it's easy to get people excited and involved when the thing you make, write about, etc. is something people *can* become passionate about. But what *about* soap, or canned chile, or garbage bags? What if your product is a spreadsheet? Sure it's easy for the guys who write fun tools for 3D animation to have passionate users, but what about the rest of us?

I can't speak for what Hugh meant, but *we* believe the answer (the imagination and sense of adventure) lies in creating an environment *around* your product or service that uses what game designers know to create opportunities for *flow states*(Beth has a lot of great things to say about flow in this post). And there's a formula for that. It's not *easy* to do, but the formula itself is simple.

It's about getting the challenge level right, and creating opportunities for people to *want* to get to The Next Level. It's about giving people the "I Rule!" experience. But first, you have to get yourself out of the way, since it doesn't matter if users think YOU kick ass. It only matters that they think THEY kick ass.

Give users a way to kick ass.

And giving them brighter whites in their laundry doesn't count as kicking ass. Giving them slightly stronger garbage bags doesn't count. Tastier chile isn't enough. Nice-smelling soap doesn't do it either.

The true feeling of *kicking ass* comes from challenge. If you get the challenge level right, people enter that state of flow where they lose track of time because they're so fully engaged and involved. They feel good about what they were able to do and learn. It's a kind of natural high, and it's been directly linked with happiness. More flow==more happiness. And game developers (and the researchers who study flow states) know *exactly* what creates the right challenge level (although it isn't one size fits all--what's challenging to some will be too difficult or too easy for others, although there are ways around that with dynamically adjusting challenge levels... but that's a different topic).

Challenge depends on your skills and perception of the task. If you perceive the challenge is too difficult, the flow state vanishes because you become frustrated and ultimately give up. If you perceive the challenge as too easy, the payoffs aren't worth it, and you lose interest. You can't feel like you kick ass (I Rule!) if the thing you're doing is dead simple or meaningless. Games or activities (skiing, rock climbing, running, etc.) that keep people engaged have a challenge level that matches the user's skills and knowledge and most importantly--*keeps increasing*.

The key is to have a cycle where the user can keep building their skills to reach higher and higher levels! In other words, the challenge keeps building, but so do the user's skills and knowledge. The spiral is a continuous cycle of motivation/seduction followed by a period of intense activity toward a goal followed by REACHING that goal which then gives you *more* skills and knowledge (superpowers, tools, whatever) that let you achieve still higher levels... and on it goes. Five hours later you're at Level Eight, or skiing bigger moguls, or helping save the world.

Which brings us back to... *garbage bags*. Nobody in the history of the world has become passionate and engaged and challenged and in flow as a result of their choice of garbage bags. What's the Next Level of garbage bags? Twist ties instead of those built-in

handles? White instead of black? Would anyone want to become expert at learning how the damn things are manufactured?

So, those who're trying to do this with garbage bags DO have a big extra challenge of their own, and they have to take extra steps. But as Hugh said, it's possible:

If there's nothing inherently engaging about your product or service, you need to create something *around* your product or service that *is*.

In other words, you have to make the engaging challenge about something somehow *related* to your product, and then you have to provide users with both the challenge AND the tools to keep increasing their skills and knowledge around that challenge area. This related thing does NOT necessarily have to be deeply *spiritually fulfilling*. You don't have to make everything a Big Important Cause. You don't have to come up with some change-the-world benefit by, say, associating your product or service with a save-the-whatever campaign. Think about it... video games keep people engaged, involved, learning, passionate, and on and on, without ever suggesting that you're directly fulfilling a higher purpose other than feeling more personally powerful at *something*. *Achieving a flow state is fulfilling on a personal level because it creates an "I Rule/I Kick Ass" experience.* And that's a Good Thing, whether it's attached to an important cause or not. *Happiness is beneficial all by itself.*

(And one can argue that in a systems thinking way, the more experiences like that a person has, the happier they are and ultimately--the more likely they are to contribute in the world. To pursue more adventures and challenges and who knows where it could lead...but that's not necessarily our job.)

So what might that be? Let's take something easier first... like coffee and soap. Coffee is easy because you could approach it in many different ways, but here are a couple:

* Provide a means to make people the coffee equivalent of wine sommeliers. Give them the tools, education, and challenges. Hold contests. Award certificates. Make snobs.

or

* Help people understand and become involved with the issues around Fair Trade. Coldplay's Chris Martin reaches hundreds of thousands of fans at his concerts, and usually the camera zooms

in for a close-up of his hands on the keyboard, where he's written "Fair" on one and "Trade" on the other. Those thousands of fans leave the concert, go home, and google on "fair trade". A coffee producer who shows some fair trade awareness is good, but that's not enough to create passionate users. But a coffee roaster who, say, provides interesting and challenging ways for people to learn and more importantly--*become involved* in issues around fair trade might have a better chance at keeping users engaged.

What if your product is soap? Maybe something like:

* Teach people to make their own soaps, using natural ingredients. Hold contests where people can submit the most wild-looking soaps, or that use the most exotic and unusual ingredients. Give people a way to keep learning more and increasing their skills, and provide interesting payoffs (a reason to get to the Next Level).

* Similar to getting people involved in issues around fair trade, you could use hemp as a platform for skill, knowledge, and increasing challenge.

OK, back to garbage bags. I've been delaying this one because I really don't know what to do there. If it were my *job* to know, though, I'd spend a lot of effort on it. But I'll throw out some probably lame ideas as a starting point:

* The easiest way would be to make sure your garbage bags really ARE made in some environmentally supportive and interesting way, and provide the tools for people to increase their knowledge and skills *in some interesting and challenging way*. When I say interesting and challenging, I mean that you can't just put up a bunch of good info to read. They might read it, but then what? That's not enough for passion.

But let's assume you just don't have that luxury. Your bags are what they are, and you have no control over how they're made. Then what?

* If you can't make it about the product itself, make it about the *packaging*. Make the packaging a collectible work of art. Hire Hugh to put his cartoons on the inside of the boxes. Give people a way to learn just what the hell these things mean (or better yet--let people *speculate* on their own interpretations). Make sure you keep varying the cartoons with new boxes, so people

have to keep buying them to see what's next. Make them exclusives, so you won't get them anywhere else.

The Chocolove company makes great chocolate bars, but so do a ton of other companies. But their packaging is truly special. They even put poems inside, and sometimes the poem is continued... in another type of bar! So you *must* keep buying the additional bars for your sweetie or you've left her/him with an unfinished poem.

* Provide games on your site. Really good, fun, interactive, high-score publishing games. Put clues to the games inside the boxes. Even if there's nothing at all in the box, if you can keep people on your site longer, they'll at least feel *something* related to your product.

* Let users design the boxes or even the bags. Or be campy and let people design the most obnoxious colors. Let people submit and post digital videos about their most bizarre/creative use of a trash bag. Teach people digital editing skills.

* Make really cool designer trash bins. Hire the best graphic designers for your bags and boxes, and offer free industrial design appreciation classes on your web site. Invite people to submit design ideas.

[UPDATE: John Mitchell commented with what I think is a much better suggestion: ." ...For example, think about the coffee example... It would take a

bold garbage bag company to actually talk about something meaningful like

actually reducing the amount of garbage."]

Of course all of this costs extra money, but you were probably going to have to find something to do with that advertising budget anyway, as ads continue to asymptotically approach total uselessness.

So if you think of what used to be your ad budget as going to your "help users kick ass and have an 'I Rule!' experience" campaign, it's just shifting the dollars to something way more useful and interesting for *everyone*. DISCLAIMER: this is all assuming that you already have a great product. A product that's at least as good as most of your competitors in terms of features and meeting the user's basic need for the product (in this case, *hold trash*). The things in this post are about rising above the

noise when there are potentially many competing products that all do roughly the same thing for the user, and do it perfectly well. In other words, this is about what to do when there just *isn't* anything truly, deeply remarkable about the product itself.

None of this is easy, and of course I'm way oversimplifying everything here. That's why we're doing a whole book on it, and a three-hour tutorial about it at the upcoming ETech conference. Game designers work extraordinarily hard at getting the challenge levels right to keep people passionate about the games, and there's both an art and science to it. But that doesn't mean there aren't a bunch of practical, useful lessons we can learn from them.

Combine that with the lessons from cognitive scientists, psychologists, learning theorists, and entertainment (that's a whole different area I'll talk about in separate blogs), and there is a formula that almost anyone can apply. We've implemented some of this in our books, and our suggestion to new authors is to be extremely careful about the challenge level in your book.

A lot of first-time authors err dramatically in one direction or the other, either by trying to make it too difficult (so that they'll be perceived as smart and credible) or by making it too easy (in a misguided attempt to build the learner's self-esteem by making sure they have plenty of successes). Remember that too easy is just as unengaging as too difficult, in some cases more so. And simply being successful at something isn't enough to give you the I Kick Ass feeling either... I could be quite successful at a book of puzzles designed for third-graders, but I sure wouldn't reach a flow state and feel powerful.

Anyway, this is just a first pass at some of these topics I'll be looking at much more deeply over the next several months as the book evolves.

http://headrush.typepad.com/creating_passionate_users/2005/01/keeping_users_e.html

Passion is infectious.

By Kathy Sierra on January 14, 2005

Tens of thousands of Mac enthusiasts converged on San Francisco for MacWorld this week, and my health insurance should cover my trip there. Why?

Because being around passionate, enthusiastic people is good for your brain.

In his book The New Brain, neurologist Richard Restak points out that your brain has a built-in tendency toward modeling/mimicking those you are around. That's *scary*, when you think about it. It means that hanging out with the whiners and complainers at the water cooler (not to mention those overly critical, judgemental, negative neighbors, friends, and family members) *will tend to make you behave like them*.

His suggestions are that you seek out and spend more time around those whose brains you like (and want to BE like) and avoid those with attitudes and behaviors that you don't want for yourself. That sounds a bit harsh, but it's the result of, well, a looooooooong evolution of the brain. Our survival as a species was based on the ability of babies and children being able to emulate others, and your brain can't tell the difference between

those you don't want to be like (but are around) and those you do. You control it by getting the hell out of there.

You've probably experienced this--you sit around a group of people whining and complaining (without any attempt to problem-solve) and pretty soon you find two things:

1) You start feeling your energy slipping

2) You start subtly *acting* more like them.

You might find yourself saying something and immediately thinking, "I can't believe I just SAID that!"

It's just your brain doing what brains do.

But think about the people you know who make you feel... *energized*. Enthusiastic. Excited. ***Passionate.*** Think of the times you've let someone else's enthusiasm sweep you away. Of course you might later come to your senses and realize, for example, that you don't actually *need* the new iPod Shuffle, seeing as how you already own two *other* iPods.

The point is, *being swept away with enthusiasm is good for your brain and your health.*

The implication is this:

If you want to create passionate users, spend time around passionate users.

Even better, spend time around others who are also trying to inspire passion in others. There's plenty of brain research that explains why you should surround yourself with passionate, energetic people and stay away from the, "This job would be great if it weren't for the frickin' USERS" people. If you want to be more creative, spend time around more creative people. Better problem solving? Spend time with those who spend more time looking for solutions than complaining about problems.

Want to change the world? Spend time around people who *want to change the world*. That's exactly what Tuesday night was like when O'Reilly held a party for the new book Eric mentioned, Andy Hertzfeld's Revolution in the Valley. Most of the original Macintosh creators/developers were there including Andy, Bill Atkinson, and others... even Wozniak was there. Now, I've spent too much time in Hollywood to be impressed by the company of "celebrities", but these folks--they *did* change the world. And

more importantly, from the brain's perspective, *they knew it*. It was their goal.

I was there with Bert, the O'Reilly folks, and Mac gurus Tom Negrino and Dori Smith (of backupbrain blog fame, among other things), and we all agreed that we were hoping some of what was going on in that room, with all those change-the-world Mac creators would rub off on us.

Just being in the room was enough to get you high.

(Then again, it could have been the beer... the Thirsty Bear, where the party was held, makes an outrageously good "Golden Vanilla" brew. I don't drink alcohol more than about twice a year, so...it kicks my ass when I do!)

If you would excuse yourself from a setting where there was too much second-hand smoke, then you should do the same thing when there's too much of an attitude or behavior that you don't want your brain to slide into. And don't fall into the trap of thinking you have complete control over this--it's *extremely* difficult to prevent something your brain spent millions of years evolving to do.

And next time someone tries to strap you into the, say, *golf appreciation chair*, let them. They might not ever succeed, but just being around someone trying to evangelize for their favorite sport, game, drink, whatever... is usually good for you.

Of course nobody has yet *succeeded* in making ME a golf convert, but for the sake of being around more passion, I say bring it on. :)

http://headrush.typepad.com/creating_passionate_users/2005/01/passion_is_infe.html

The effect of sound on users

By Kathy Sierra on January 15, 2005

In Hollywood, some say that in a movie the visuals tell you *what* you're seeing, while the soundtrack tells you how to *feel* about it. I used to teach new media and interaction design at UCLA Extension's Entertainment Studies Department, and in the film scoring classes (which I didn't teach) students often started out with a classic exercise: Watch the shower scene in the movie Psycho, *without the sound*. Without that Ee-Ee-Ee-Knife-Slashing audio it just doesn't feel like the scene that scared me into being more of a bath than shower girl.

And audio has even been shown to affect the audience perception of the quality of a presentation more than the visuals. I don't have a link handy, but there's a study that showed that the quality of the audio causes people to change their evaluation of the quality of the visuals, but that it doesn't work in reverse. In other words, given a film clip or animation, raising the quality of the audio caused people to say, "Hey, the visuals are good!", and when the audio quality sucked, the audience rated the visuals as worse even when the visual quality was the same in both the bad and good audio versions. But... changing the quality of the visuals did NOT have that same power. If the sound sucked but the visuals were fabulous, it didn't cause viewers to say, "Wow... good sound too."

So, sound has the power to raise (or lower) audience perception of visuals, but visual doesn't have the power to change how the audience perceives the audio.

But sound is usually the second-class citizen in the non-professional multimedia world, while visuals take center stage in everything. (Unless the photographer, videographer, or animator happens to also be a musician). Everyone has a digital camera now--that makes everyone at least an amateur photographer. And everyone has some kind of digital editing software like Photoshop Elements that brings high-end photo manipulation to the home user. And why stop with *still* pictures when digital camcorders are so cheap now? With editing software like iMovie shipping with every Mac, anyone can become a video editor now.

But while the emphasis on developing visual sensibilities and skills has continued to build (almost everyone with a digital camera today knows design fundamentals like the "rule of thirds" or how not to cut people off at their joints), what about the poor stepchild audio? Sure we could all *listen* to music, but where were the tools that would bring music creation to non-musicians in the way that the visual tools (and books and references) brought graphic and photographic editing to non-photographers and non-graphic designers?

I gave a presention on this discrepancy ten years ago to a new media group in Los Angeles, and while I stood there ranting about how nobody had made the Photoshop equivalent of audio (or even the Kid Pix equivalent) one guy in the audience, Kevin Klinger, started thinking about it. He went home, thought some more, and decided to *start a company* to do just that. I was at Mac World many years later, and there was the Smart Sound booth, and Kevin said, "Hey, thanks for the idea." I couldn't believe it. *He actually did it.*

SmartSound's focus is on giving people a way to create sound tracks for videos, without being a musician. And the "smart" part comes from the cleverly-engineered ways in which the software fits itself to the movie. Because another problem with most home movies is that the music often doesn't *finish*, it often just fades out (or worse, cuts off), because the music was too long for the video. SmartSound is an amazing program and goes way past what I had in mind when I gave the talk. But it's main focus is on making sound tracks for, say, corporate videos, while

I was still waiting for that low-end, home-use music creation program for non (or very weak) musicians.

Something that would encourage "regular people" to start developing music and sound sensitivities in the way that we've developed our design and visual awareness and creative skills. In other words, when will "the rest of us" get to work on the "A" in "AV"?

Of course, Apple's done that now with the phenomenal Garage Band! A tool that threatens to turn people who have no business making music into musicians. (When I say "no business making music" I'm referring to the notion some have that there must be clear boundaries between those who *create* art and those who *appreciate* it. I think that's bullshit... we're all born creative even if many of us will never EVER hope to be professionals. Heck, we all spent our first nine months listening to a 24-7 dance beat.)

I blogged earlier about the way that teens and twenty-somethings today often treat turntabilism the way forty-year olds used to treat guitars. *They treat it like an instrument.* Something to use for creating music. So the audio world is definitely changing, and Garage Band is, I think, the single most important step in bringing music into the world formerly reserved only for graphics programs.

The four of us care a lot about sound here, because sound (and especially music) has a powerful effect on learning, in two ways. First, it manipulates emotions, and emotions play a huge role in memory formation. Second, the more senses you can involve in learning, the greater your chances of retaining and later recalling the knowledge. Think about it--if you file something in *two* places instead of one, you've doubled your chances of getting it back again, and when you remember something as, say, a sound, image, and text--you've just given yourself three potentially different ways in which that info is stored in the brain. Triple the chances of getting it out when you need it.

But... we're still doing books and books don't give us a way to do that. We *are* planning some multimedia formats for the fairly near future, but for now, we're doing what we can with things we expect you--the reader--to do to fully realize the power of audio. When you come across a limerick, poem, or song lyric in the book, for the love of god PLEASE say it out loud! Don't just read

it silently (although even then, you often will hear yourself saying it with a rhythm, and even *that* helps).

Anyway, I'll have a lot more to say about audio in the future, but for now, here's my really BAD version of the film school exercise for those who aren't musicians or sound designers. I have three videos, all with the exact same little scene, but using three different songs. (The songs are just simple sequences I put together in GarageBand, which means no copyright issues :) [Disclaimer: I'm not a musician (which will be painfully obvious if you play these). The point is the *effect* the music has, regardless of what you're looking at.]

Your assignment is to play the videos, in order, and with each one, write down the following info before moving on to the next video:

1) What kind of movie is this? Speculate on what the movie might be about, based on the *feeling* you get from the audio. Do NOT use your brain to try to think something up, just go with what pops into your head based on the feeling.

2) Speculate on what might be happening in this character's life. Make up what the next scene might be, based on the feeling you have from the sound.

Once you've done that with all three, play one of them without any sound at all and see how you feel about it now. (You can experiment by playing one of the others, and see if the last music you heard affects how you view it when there's no music at all.)

Finally, pick an emotion/feeling you want to evoke, and find some music you have that you think will create that feeling. Play that while the video is playing, and test it on someone else and see if your victim gets the feeling.

Here they are, and remember... watch them in this order (note: they are Quicktime movies, about 1.5 Mbs, sorry dial-up users : (

Version one movie, Version two movie, Version three movie

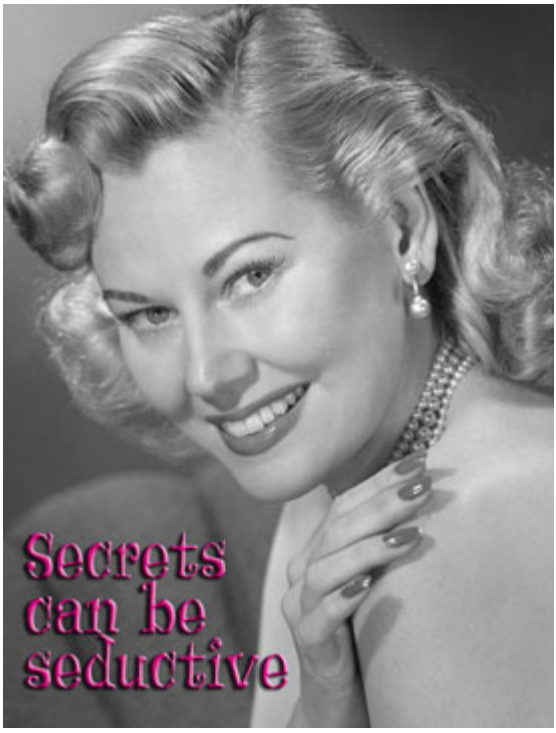
So, how are you using the power of sound in your teaching and learning and communicating?

http://headrush.typepad.com/creating_passionate_users/2005/01/the_effect_of_s.html

Transparency vs. Seduction

By Kathy Sierra on January 18, 2005

"Transparency" is a hot word in business blogs, but just how much transparency do we really want? At what point does transparency become Too Much Information?



I'm not talking about financial transparency or being honest over things like fair trade, sweat shop labor, animal testing, blog motivation and sponsorship, environmental issues, harmful/dangerous things, or even poor customer service and products that weren't ready to ship but... *did*. I assume honesty is becoming the "killer app" for businesses-be honest or be killed. But that's not the kind of transparency I'm talking about.

I'm thinking more of things like Scoble's Channel 9, and this notion of putting up, say, a blog devoted to a reality-tv style behind-the-scenes view into a company as a (in part) marketing tool. I'm talking about *ethical* businesses using transparency to help inspire passionate users... a fantastic idea, but how far do you take it?

What do you *risk* when you put up video of your meetings, project notes, discussion transcripts, product development process details, or even just photos or a webcam of your "team at work"?

Mystery.

Seduction.

Do I *really* want to know what's behind the curtain?

What if it sucks the suspense out of the whole thing? What if surprise and delight are intimately connected, and that removing all the surprise takes away much of the delight?

I was one of those at the MacWorld keynote in San Francisco who--knowing all the while that a low-end Mac was likely to be part of the show--gasped when Jobs held it. We'd all been describing what we thought it might be with uninspired words like "headless box" and "iCheap", but here we were-surprised, no SHOCKED by even the damn *box* it comes in. I was thrilled that I didn't know going in what I'd *really* see on that stage.

People love surprises.

The brain is turned on by mystery, curiosity, seduction.

I think it was the Dalai Lama's brother who said (in an NPR interview) something like, "If you shine a bright light into every corner of your apartment, it will become unliveable."

I'm thinking that sex isn't the *worst* model for thinking about this. Or if not sex, then at least *romance*. Honesty is crucial in a romantic relationship. *Brutal* honesty can kill it. If I ask a question, don't lie. But if you feel the urge to "share" every last detail, you might find me less interested. Does that make me shallow? No, it makes me *human*. Our brains are tuned for things that make us curious, because it saved us from being tiger snacks.

I won't open my presents before Christmas, and I keep my eyes closed while someone special "prepares the surprise". I like twist endings and shocking revelations. *I don't peek.* (OK, maybe just a little.)

Surprise is an aphrodisiac, so please do not tell me EVERY LAST THING about your product development and process. No, once I know you're up to *something*, I usually prefer to stay blissfully

in the dark, or at least in a blindfold, while you prepare to dazzle me.

It works for filmmakers and novelists and lovers.

And yes, markets *are* conversations. (The brain is all *about* conversations.) But what *kind* of conversation? Because I'll take a stimulating flirtatious dinner party to a laid-bare, tell-all, talk show any time. Shock me with your stories, your liveliness, and your creativity. Not by revealing what you look like when you haven't showered for a week. Yes, genuine (and I'll indulge in one more cliche--*authentic*) communication is more important than ever as people lose their last bit of tolerance for bullshit. But I'm talking about keeping a healthy, scintillating *balance*. Cold and distant won't work anymore, but don't rush too quickly to the opposite end. Thank-you, but unless I'm sleeping with you, I'd prefer NOT to see you quite so up close and personal. So tease me, drop hints, do the business equivalent of showing a little skin, but hold a little something in reserve. *Whet my appetite*.

And as Beth just reminded me, "this is not about *hiding the truth* from stockholders, auditors, customers. It's about ***keeping the next cool thing under wraps***."

Charm me. Delight me. Make me gasp and I might be yours forever. :)

(You can read more on transparency here, here, here, and here. And by the way, I'm not suggesting that Channel 9 has slipped over the edge at this point, only that it might be playing in the danger zone.)

http://headrush.typepad.com/creating_passionate_users/2005/01/transparency_vs.html

Crafting a user experience

By Kathy Sierra on January 20, 2005

There's a pretty simple formula for keeping users engaged; we call it the spiral experience model. It's based on four parts:

1) Get their attention (get past the brain's crap filter).

2) Give them challenging, engaging experiences. (Experiences designed to keep them in the flow state.) This part is a spiral, where the user gets a payoff for their interaction (getting to the "next level"), and the payoff, in turn, creates new interest (seduces them) to want to use their new knowledge/skill/superpower to keep going... and on it goes.

The keys are challenge, meaningful payoff, and creating new interest by giving them clear, cool new goals. ("Now that you reached this level (or now that you know this new tool, or understand this new issue), look how you can use that new knowledge/skill/superpower to do this even COOLER thing...").

This spiral is in some ways at the heart of game design, good learning experiences, pacing in many novels and films, sports that keep you in the flow state, and is the model we try to use in our books. But you can use it for just about anything you communicate-the idea is to inspire users to want to learn more (or at least *do* more), so that they want to keep progressing. The payoff/reward for their involvement should be a meaningful lead-in to yet another round of wanting more...

3) Leave them with the "I Rule!" feeling.

Remember, it doesn't matter what users think about YOU. All that matters is what they think about themselves as a result of interacting with your [whatever it is you make or do].

We're on a big deadline to finish the Tiger edition of the Head First Java book (supposed to be done tomorrow!), so this is the Cliff Notes version of the model. More later...

http://headrush.typepad.com/creating_passionate_users/2005/01/crafting_a_user.html

Most classroom learning sucks

By Kathy Sierra on January 22, 2005

The problem with most corporate/adult learning programs is that they're just like school. And the problem with school is that it sucks. It works against the way the brain wants to learn.

The best learning occurs in a stimulating, active, challenging, interesting, engaging environment. It's how the brain works. The best learning occurs when you *move* at least some part of your body. The best learning occurs when you're actively involved in co-constructing knowledge in your own head, not passively reading or listening. (Taking notes doesn't really count as being *actively involved*.)

People complain that their kids can't pay attention in school, then their kid comes home and spends two hours studying the elaborate world of Halo 2. Reading, absorbing, problem solving, using sophisticated mental maps, and on it goes.

When learning is "presented" in a push model, your brain says, "This is SO not important." You're in for the battle of your life when you try to compete against the brain's natural instinct to

scan for unusual, novel, possibly life-threatening or life-enhancing things.

Forcing people to sit in a chair and listen (or read) dry, formal words (with perhaps only a few token images thrown in) is the slowest, least effective, and most painful path to learning.

Yet it's the approach you see replicated in everything from K-12, to universities, to adult/corporate training.

Skyler (my switcher-daughter) was fortunate enough to go to a private school until 6th grade. In that school, there were no classrooms. There was no teacher-at-the-front rows of chairs thing. Kids sat where they wanted to do their work--on the floor, on the deck, at the kitchen table, *whatever worked for them*. There were no lectures, no formal lessons. When kids needed help on a "project", they asked, and one of the teachers helped them. If a few kids were dealing with the same thing, the teacher might take them into what looked like a little corporate conference room, for an ad-hoc session. Even then, the teacher was more like a mentor/guide, and not the "sage on the stage". Kids were allowed to work on whatever they wanted, as long as they were fulfilling, somehow, their goals to include geography, math, language, etc.

And each kid had his entire curriculum custom-made for his personal interests. *For the things that turned his brain on.* One kid was obsessed with dinosaurs, so with the help of his teacher, he designed his entire first year around dinosaurs. Everything he did was based on learning more about dinosaurs. Math was based on calculating sizes and dates, and making his own categorizations. Language was, well, he had to learn to read if he wanted to learn about his passion. Geography was based around researching the areas where different dinosaurs lived at different times, creating timelines, etc.

Another kid's father frequently traveled on business, and his son was fascinated with hearing the stories his father told about the places he went. So they built a program around the hotel brochures his father brought back. He learned to read the brochures, then to work out the distances between the different hotels, and even make little spreadsheets to calculate expenses and work out budgets, etc.

The important thing was that they took the time to discover what the kids were passionate about, and used that as a vehicle for motivation.

Kids aren't motivated about *geography*. They're motivated by where dinosaurs lived, or where their dad is today. They aren't motivated by *arithmetic*. They're motivated by how big dinosaurs are or calculating which hotel their dad should visit.

And that's just the *first* year. By the next year, they've done the dinosaur/hotel thing to death and they're ready for something completely different. The idea of weaving everything-math, science, language, history, geography, whatever-into a framework that capitalizes on the learner's passion was the most dramatic example of powerful education that I'd ever seen. Her school had no grades, and no homework. Ever. It was a leap of faith for most of the parents, that somehow your kids were keeping pace with their counterparts in the "normal" school system, especially since most of us knew that we couldn't afford this forever, and that our kids would all eventually make their way into public schools to finish out.

The school *did* give standardized tests, and the typical score for the kids in the high 80's to 90's percentile against the national average for their grade. Even more importantly, most kids left 6th grade scoring at least two years ahead of their public school (and every bit as intelligent) peers.

The most depressing result of Skyler's transition to public school was when she came home one day a few weeks into her 7th grade, and said, "In real school, they don't seem to like it when you question the teacher..." She was horrified to be labeled somewhat of a troublemaker, because she'd been treated for so many years as a thinking person, encouraged to challenge and question and not assume it was *her* fault if she didn't understand something. Suddenly dropped into the US public school system, she quickly learned that it's a very different world. She knew more about learning theory and the brain than most of her school's administration, and her tolerance for poor/weak educational experiences was pretty low.

She did have some fabulous teachers throughout the rest of her public school days, but wouldn't you know it--*they were always the teachers getting into trouble with the school administration or even parent's groups*. In a later post I'll tell you a shocking story about one of her teachers who made the national news,

twice, for encouraging students to think--and act-- for themselves. He was nearly fired during a witch hunt that both local and national media seized on (although most later offered apologies when it became obvious what was really going on).

One of the biggest mistakes adult learning programs and learners can make, in my opinion, is to use traditional school as the model. It doesn't work for kids, and it doesn't work for adults. *Because it doesn't work for the brain*. I know there are enormous challenges and pressures for delivering public school learning (that so many teachers don't have the option or power to change), but most adult education programs that follow the same poor model don't *have* those excuses. In many cases, adult classroom training looks like school just because that's how it always looks. There are a *lot* of interesting and wonderful exceptions in the adult learning world, of course, and a lot of novel things being done with everything from arrangement of chairs in the room to the role of the instructor as facilitator rather than "teacher", and I'll say more on that later.

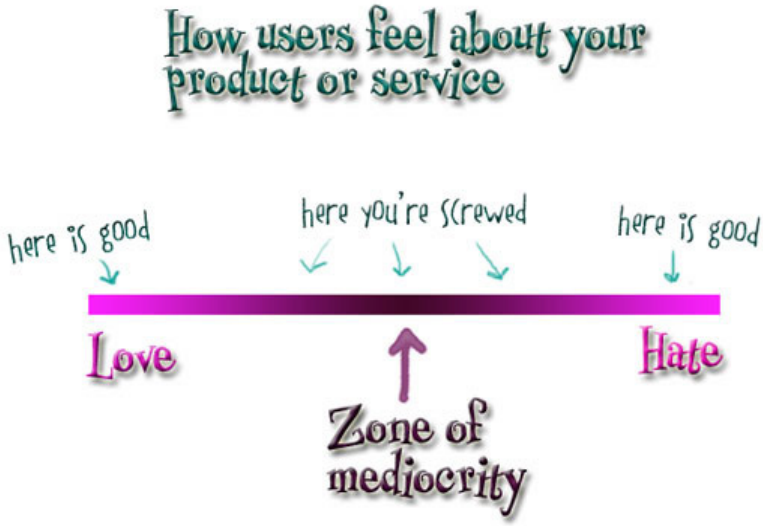
But for the most part, we're still using the same approach that, given the pace of information change today, is even LESS useful than it was in the past. We need a big change.

[Update: several people have asked about Skyler's school--it was Manhattan Academy in Manhattan Beach California. Be sure to read their philosophy section; when Skyler was there they really *meant* these things. Too many schools have a nice set of bullet points about their values, but putting them into practice is a different thing. Manhattan Academy *walked the walk*.]

http://headrush.typepad.com/creating_passionate_users/2005/01/most_classroom_.html

Be brave or go home

By Kathy Sierra on January 25, 2005



Seth Godin says that today, "being safe is risky, and being risky is safe." And if you're out there creating something on the edge, someone's going to hate it. Probably a *lot* of someones. One thing we noticed from our Amazon reviews was that we get mainly five-stars and one-stars, but not much in the middle. They either love it *a lot* or they hate it with a passion. Whenever I start to feel bad about a scathing review, I remind myself that Don Norman said, "If someone doesn't really hate your product, it's mediocre." And mediocre is where you SO do not want to go.

Ever since we started this crazy scheme (18 months ago with the release of the the first book in the series), we've been thinking that the extremeness of our reviews was a good thing, and now someone's confirmed it. A NYTimes article looks at a professor who analyzed Amazon book rankings for, among other things, a book's "controversiality index". From the article:

> *"But the most telling variable is the one star rating. Professor Gronas found that books high on what he called the "controversiality index" are given almost as many one-star as five-star ratings, creating a horseshoe-shaped curve. As it turns out, these books also tend to have high sales."*

You gotta be brave out there, now more than ever. That's one of the reasons we're big Tim O'Reilly fans, because he's definitely brave. He was more than willing to risk taking a chance on us when the other publishers turned us down or said they'd publish Head First if we scaled back to only 10% of what we wanted to do. That was like telling us, "You can be only 10% brain-friendly". No thanks.

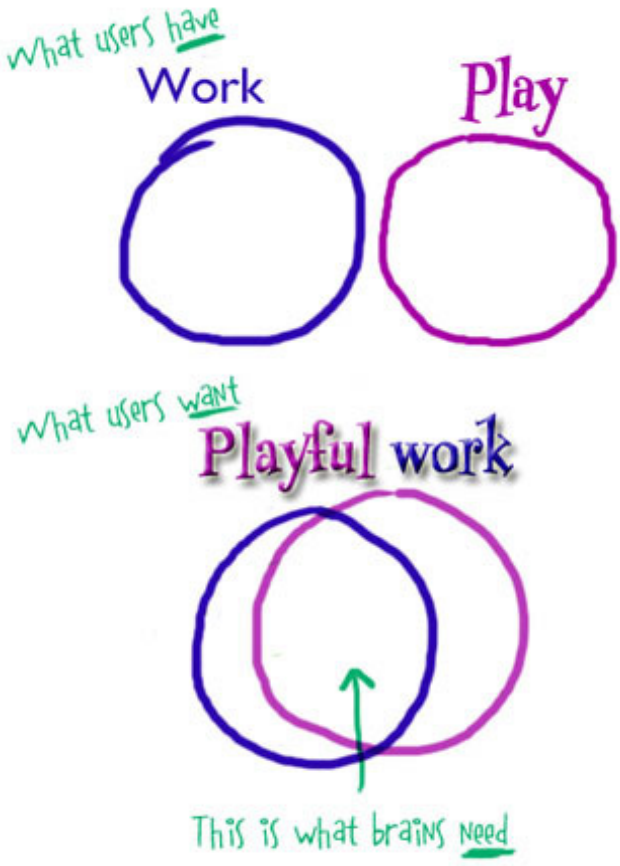
Making only *incremental* improvements won't work today, not with the gazillion competing products and services all fighting for attention and offering pretty much the same perceived benefits. Just keep being brave and most importantly--when you start to have doubts about how far out-there you should go, and you're imagining how the critics will burn you alive, just remember that the *worst* thing is being in the Zone of Mediocrity. *That's* what we should all be afraid of.

Creating passionate users is NOT about finding ways to make *everyone* like you. It's about finding ways to use your own passion to inspire passion in others, and anything with that much power is bound to piss off plenty of status-quo/who-moved-my-cheese people. *Bring it on*.

http://headrush.typepad.com/creating_passionate_users/2005/01/be_brave.html

Creating playful users...

By Kathy Sierra on January 26, 2005



If you're a game developer, the things you're building are all *about*play. But what if your product or service isn't inherently playful?

Brains love play. Find a way to bring more play (or at least a sense of *playfulness*) into someone's life, and you might just end up with a fan.

Brains evolved to play, and apparently the bigger the brain, the more likely it is to play. ***Play turns the brain on.***

So, OK, but what if the product itself is for an utterly non-playful task? You can still bring a sense of playfulness into the mix. One extremely difficult CAD program I heard about created a game

to teach people how to use the software. To "get to the next level", you had to learn more of the tools.

Steve Zehngut, from Zeek Interactive started his company in the mid-90's by specializing in building interactive games for business, designed either for marketing, training, or both. One really cool game was designed to teach people about photocopiers (I think it was for Toshiba), and you (the player) were being attacked by your office mates who were throwing wads of paper at you. You had to figure out *which* copier to hide behind and use as a weapon. The best weapon, of course, was the machine that fired staples, but... you had to *know* your copier models in order to pick the most effective "weapon".

I was very disappointed that I wasn't able to attend the Serious Games Summit last October.

But *playfulness* doesn't have to mean *games*.

Helping people feel just a little more playful, especially if it's connected to their *work*, or with anything they do that's more typically associated with words like painful, tedious, boring, stressful (as opposed to words like "fun"), doesn't have to mean giving them a game. Even something as simple as making your documentation more compelling (and even a little whimsical), can make a huge difference.

You're a musician, and on your web site you create Make-Work-Suck-Less playlists (which you also put on iTunes, of course) for people at work. You tell them what to listen to for ever possible bad work situation. Want to kill your boss? Pick this track. About to head into yet another dull, pointless, loaded with marketing-speak buzzwords meeting? Pick *this* track. Encourage users to make their *own* making-work-suck-less playlists.

You put easter eggs in your otherwise ludicrously dull accounts receivable software, and spread hints about them on the internet. Suddenly it's a little treasure hunt cleverly disguised as a boring business task. (I know, I KNOW programmers have been fired for doing that. I came quite close, and that was for putting an easter egg into a--wait for it--GAME. My easter egg wasn't on the approved list of "features"... incredible that even when you're technically in the business of fun, "management" can be so serious).

You're a realtor and you hold feng shui workshop/parties (hoping your sellers will take the hint and whip their homes into shape...)

You're a huge rental apartment complex and you host dog parties for your tenants.

You're writing a computer programming book, and you put in puzzles, games, fun pictures, and festive examples with unusual characters.

Surprises are one of the best things you can do--psychologists claim that intermittent rewards can be more engaging than consistent rewards. Remember, surprise=delight.

I worked for a guy who ran an exclusive, foofy, insanely expensive health club. He took 100% of what should have been (back then, when Ads were King) his advertising budget, and instead put ALL of it into a monthly "member surprise" budget. Nobody ever knew what was going to happen. You'd be in an aerobics class with 100 people (it was a big place), and as you walked out, suddenly there were carts loaded up with bowls of frozen yogurt and a toppings bar. You're in the weight room when the employees start walking through handing out exclusive t-shirts, always with his logo, and always with a fun quote, that you knew would never appear on a t-shirt again. Members collected these things like rare beanie babies. The late-night exercise classes were the hardest to fill, but he would take the worst time slot and make it interesting... the 9 PM folks might walk out of class only to be handed a wine cooler or even a relaxation CD.

It always felt like a party in there! And employees fought over the chance to be the one who got to hand out the cool stuff. And there was no hierarchy in deciding who got to do that...everyone from the janitors to the office bookeeper might be "picked" to be the hero. I had never before, and never since, seen the kind of loyalty among both staff and members that I saw in that place. His attrition rate for both members and employees was less than half the industry average for health clubs at the time. (I'll have more stories about him in other posts--his name is Cliff Coker, and his father was one of the founders/inventors of the very first selectorized exercise machines (the ones with the weight stacks, as opposed to free weights), Universal Gym Equipment.)

Spend some cycles cultivating your more *festive* side. Think ***party***. Think of that person you know who is so *fun* to be around. The one who manages to make a little adventure out of *everything*. If you can give your users even one moment *more* of that feeling, the world will be a better place. :) [cue cheesy, sappy pollyanna music, and insert cute kid-with-puppy picture]

Hugh ("He likes us! He really likes us!") got me thinking about this with a quite lively (be sure and read all the comments) gaping void post on how Microsoft should be more playful. While that's beyond *my* powers of imagination, it's certainly an interesting challenge...

So, what are YOU doing to help your users be a little more playful?

http://headrush.typepad.com/creating_passionate_users/2005/01/creating_playfu.html

Teaching and advertising

By Kathy Sierra on January 27, 2005

Teacher, meet ad guy. Ad guy, meet teacher. You two could learn a LOT from each other in this braver, grimmer, faster, more authentic world. But I can sum up my feelings as:

Teachers need to get better at *motivation*.

Advertisers need to get better at...*caring and honesty*. (Not to mention things like REAL retention and recall--something teachers know a little something about...)

Advertising (in its conventional, old-school form) may indeed be dying. Meanwhile teachers/instructors are struggling more than ever to get learners to pay attention and learn. But I believe both groups could improve their results if they took a lesson from the other. Advertisers need to care, and be honest--something teachers can be quite proud of. Teachers, on the other hand, need to work on their motivation--the domain that advertisers have (or *had*) down.

Advertisers have 30 seconds in which to convince someone that this [insert any lame product] will lead to more sex. And the weird part is how damn effective they've actually *been* at this, especially in the days when everybody read the same limited number of magazines and watched the same three TV networks.

Teachers, on the other hand, have been providing inspiration and changing the lives of kids. Almost any adult today can think back to at least one teacher who really made a difference in their life. Why? Because the teacher *cared*, and cared enough to be *honest*. They were authentic.

But teachers are finding themselves less effective today, when the competition for attention has become much more fierce, and the signal to noise ratio makes it harder than ever to get *anything* to stick. Students of all ages today would simply rather be doing something else than sitting in class learning... what exactly?

If I'm teaching, I want to remember that I need to offer "meaningful benefits". And by meaningful benefits, I don't mean, "...if you do this, then the enterprise component will stay synchronized with the underlying persistent store..." No, if an advertiser rewrote that, he might say, "Because if you do this with the enterprise component, you'll be a frickin' hero and... have more sex." Or, "because if you DON'T do this, you'll lose your job and nobody has sex with losers..."

What can I learn from that? I can take the motivation to its logical conclusion, then take one step back, and let the learner make the leap. So instead of, "... then the enterprise component will stay stay synchronized with the underlying persistent store..." I might say, "if you don't do it this way, you could be a victim of the dreaded Lost Update problem and... *that means you could lose the entire record of Suzy's last Victoria's Secret purchase."* Then I let *them* make the one final leap to, "the boss screams at me, it shows up on my performance eval, I don't get that raise, and that means... less sex." (And yes, there's a reason I said "Victoria's Secret" and not "lose the entire record of Bill's Office Supplies purchase...". It's almost biologically impossible to not have at least some tiny chemical reaction to the phrase "Victoria's Secret" that simply doesn't happen when you're talking about pencils and staplers. And remember, it's that chemical reaction that leads to attention and memory. *It's that chemical reaction that tells the brain that* ***this is important! Pay attention and record!***

And what can advertisers learn from teachers? To be honest. To find out what really IS good for people. No, not to find out, to *care*. Then they use their powers of motivation... *for good*. To help people learn faster, become more effective, *make better*

choices. Yes I really AM that naive and optimistic. But if the Cluetrain predictions are true, and I believe they are, and advertising is no longer going to work, then advertisers are going to have a lot more time on their hands. And they can *use* that time to, say, start a blog that teaches someone why they really *should* buy this product, and how this product really *can* make their life better.

Most importantly, if the product is crap, or it can't really do what they're claiming, I hope advertisers will do what teachers do...*be honest*. Care.

So, when someone asks me how to become a better instructor, I often tell them to study up a little on what advertisers are doing. When someone in marketing wants to do a better job, I tell them to learn a thing or two about learning.

http://headrush.typepad.com/creating_passionate_users/2005/01/teaching_and_ad.html

Giving a damn about customers...

By Kathy Sierra on January 29, 2005

The Performance Review

That's a true story. It happened to me, at Sun. While sitting in the hospital early on a Monday morning, waiting for my CAT scan (after a donkey kick to the head that sent me there unconscious the night before), I called in to explain why I wouldn't be showing up at the customer's site that day. I was told, "there's *nobody* in all of Sun's education division that can do this now, and we can't reschedule that customer's enterprise Java course for at least three months." Long Pause. "OK, I'll be there. But tell them I'll be late. Oh, and you better warn them I look like... well, I hope they aren't squeamish."

The customer's employees were horrified when they saw me--both shocked and incredibly grateful that I had actually done this.

And of course my mangers at Sun were deeply appreciative. Or so I imagined. Fast forward to my annual performance review a couple months later when I get my "Meets Expectations" rating.

I asked the obvious question, "So if [rattle off my list of do-anything-for-the-customer examples, of which the donkey incident which was just one] only MEETS expectations, then

what the hell does it take to EXCEED expectations?" For dramatic effect I added, "Because I have to tell you, another year like this and I'll be *dead*." I was only partly exaggerating.

The manager's answer sums up the problem nicely, "There's a quota for the eval ratings and, uh, we gave 'Exceed' to Fred because he had a higher number of 'on-platform' hours. His work accounted for more direct revenue."

I countered with, "But Fred (not his real name) *hates* customers; he shows open disdain for them when they ask a question. And because I'm on the Quality Reveiw Board, and have to field all the customer complaints, I know that YOU know this is no secret to the customers. They leave his courses vowing never to take a Sun course again."

"That's not the point," the manager says. "This is simply about numbers. My hands are tied."

(Within 24 hours, someone had posted a Dilbert cartoon on my cubicle where Dilbert had donated a kidney to their biggest customer, and still got a "Meets Expectations.")

From a systems thinking perspective, it's no great leap to say that while Fred might have been responsible for more revenue *that* year, his "I hate customers" attitude was responsible for a devastatingly low customer-retention rate. The next time those customers took an advanced Java course, it sure wasn't from Sun. (And we actually had numbers to prove this.)

Meanwhile, the management of that company I walked into with my smashed face never forgot what I did, and they saw that as a reflection of the value Sun put on meeting their customer commitments, *no matter what*. We continued to do business with them almost non-stop from that first week. I set the tone for their relationship with Sun. (Not that I recommend the whole donkey-kick thing as a viable *strategy*...)

I guess I have two points:

1) If you're a manager, for the love of god PLEASE make taking care of the customers a top value. Customers are living, breathing people--not just Six Sigma stats.

2) Never, ever let your head be in striking range of a donkey.

http://headrush.typepad.com/creating_passionate_users/2005/01/giving_a_damn_a.html

Cognitive bandwidth is like dial-up

By Kathy Sierra on January 29, 2005

A couple days ago I got an email from Steve Krug, author of the wonderful web usability book Don't Make Me Think, which is in my Top Ten Computer Books list on Bookpool.com.

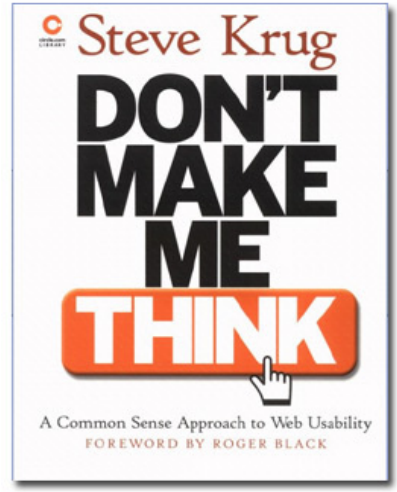
I thought about how *our* books could have been named just the opposite of his--*DO Make Me Think*, since much of our approach is about how to get learners to process new information more deeply. In other words, we work hard to *make* people think.

But then I realized that both his book *and* our approach could have been named:

Don't make me think about the *wrong* things.

I can't speak for Steve, but my interpretation of his message is something like:

When someone comes to your vintage vinyl store, they want to think ONLY about the records.



They do *not* want to think about whether that picture over there is the thing they're supposed to click. They do *not* want to think about where they are on your site, how they got there, and how the hell they get *back* to where they wanted to be. Worst of all (for the store, anyway), they do *not* want to think about whether your website actually *is* an online vinyl store.

If I'm digging for just the right record for my perfect remix, *that's* what my brain wants to focus on. I want your site to stay in character, and not take me out of the digging-for-vinyl experience by forcing me to think about your *user interface*. I want to be in flow, just as I would in, say, a real bricks and mortar record store, where the experience is intuitive.

Cognitive bandwidth is precious.

We try to reflect this in our learning books in two main ways:

1) Use a strict 80/20 approach with the material.

Rather than taking a topic, making a chapter out of it, and doing it *to death*, we try to focus on just the part that gives you the power you need to be creative, and leave off everything else. Because we assume you're not reading our book as an intellectual exercise or to skim every possible factoid about the topic. **We assume you actually want to *do* something.**

2) Don't use an example that comes with cognitive overhead.

We had a Java course at Sun where one of the early exercises was on the *looping* constructs of the language. But the exercise itself was a task that, among other things, involved converting newtons to kilograms. The scenario was some kind of package shipping system, or something like that.

Of course what happened is that when the students got to that exercise, they focused their brain on the whole newton-to-kilogram thing, and struggled with understanding the shipping domain. In other words, *they were thinking about the wrong things*. All we wanted them to do at that point in the course was understand the basics of looping. But the exercise added so much cognitive overhead that looping was the *last* thing they were thinking about. [Disclaimer: we don't always succeed at this... I've authored more than one chapter where *I* forgot the point. But we're trying. Hard.]

When someone has trouble *applying* knowledge, it's usually because they really never *had* knowledge. They had *information*, and that's not the same thing. You can get information just through listening or reading, but *knowledge* requires *thinking*... ***thinking about the RIGHT things***.

Our advice to our authors, teachers, and web/software developers is this:

Figure out what you *really* want users to think about. This is almost always *the cool thing they want to do (pick the right record, learn how loops work, etc.)*. Do whatever it takes to keep them from having to think about anything else!

Imagine your users all have thought bubbles over their heads that say, "Don't make me think about the wrong thing!" If a user has a confused look, it should be because she's struggling with whether the sea foam green bustier really works with the neon pink skirt (*it doesn't*), or whether the iPod Shuffle is better than therapy (*it is*).

http://headrush.typepad.com/creating_passionate_users/2005/01/when_cognitive_.html

Making content meaningful to users

By Kathy Sierra on January 31, 2005

At a trade show, you can almost always tell whether you're talking to an engineer, marketing, or sales person. (Yes, I'm stereotyping and generalizing to make a point). The *engineer* (that would be me) just starts telling you all the cool things the system does, rattling off the technical details as if you cared, let alone *understood*. The *marketing* person's speech is peppered with buzzwords that make the product as compelling as a tax form.

But the skilled and ethical *sales* person, now *they* know that a potential user doesn't care about *you* as much as he cares about what this means for *him*. The good sales person knows you don't care about technical details or even features. You care about *what those features mean to you*. The good sales person knows it isn't even about *benefits*, but about **the benefits *you* care about.** (And this applies to teachers/authors as well as people

trying to *sell* something. After all, as teachers we're trying to sell learners on why they should pay attention and flex a few neurons on the material...)

So the simplest solution when you want to get someone excited (or better yet--*passionate*) about what you do is... *ask*. Find out what they do, need, and want, and map what you offer into something meaningfully relevant for that person. And if you can't come up with one, then you're either working for the wrong cause (i.e. a product or service that sucks for pretty much everyone), OR what you have is simply not a good fit for this particular person or company, and you *tell them that*. I'm enormously impressed when a sales person refers me to a competitor, for example.

But what if you don't have that luxury? What if you're not at the trade show or on the sales floor or anywhere where you can have a one-on-one conversation? How can you make what you have seem *personally relevant*?

A lot's being written (and developed) around the notion of personalization today, and not everyone thinks it's a useful strategy. But there are some fairly simple ways to tailor a message in a way that makes it more relevant, and sometimes with surprisingly good results.

I worked as the programmer on an interactive marketing compaign for a large car company, and the model we wanted to use was The Good Salesperson. In other words, we wanted a system where the user/customer could walk up, answer a bunch of questions, and using a combination of artificial intelligence and a large content database, the system would deliver to the user a highly customized experience that matched what a Good Salesperson would have done... by asking questions and providing tailored answers. (sheesh, that last sentence came dangerously close to marketing-speak)

Just one problem--*no budget*. We didn't have the time or money to build that. So we did the least we could get away with; something we thought would have almost no effect, but turned out to be astonishingly effective! We saw some research (sorry, I can't dig it up right now... I just moved last week and I'm an organizational disaster), that suggested that even the most subtle shift in framing or positioning the way you offer information about your product can make a very large difference in the user's perception of how this relates to them personally.

So here's what we did:

* When the user walked up to the system, they had to answer just a single question--

What's most important to you in a car?

* Based on that one answer, we changed *only the headline/title* of the screens that followed.

For example, if the person said, "I care about safety more than I care about maintenance costs", then on the screen that talks about the engineering of the car, the headline would say something like, "Engineered with your safety in mind..." or something like that. And we might throw in a gratuitous picture of a kid in a car seat. (Yeah, I *know* that's manipulative, but it wasn't untrue.)

The main point of the system, though, was that *99% of the content was the same for every user*. We didn't have custom-tailored screens other than the banner at the top. But it turned out that by orienting the content--the same content *everyone* saw--to something meaningful for that individual, the information became more relevant.

Of course you don't want to do this dishonestly--as it would be if we said something like, "Your safety is our MOST IMPORTANT GOAL", and then if you chose "Resale value" we said, "Maintaining your resale value is our MOST IMPORTANT GOAL". But by putting a personally-tailored headline over non-custom content, we were able to connect the content to the user's individual desires. Honest, but personalized.

And according to the client, it was a huge success! People spent much more time on each screen then in the previously uncustomized version.

As teachers we use this same principle--at the beginning of class, for example, when I ask the students to introduce themselves, I try to learn as much as I can about their background and interest in the subject. Then if that person asks a question, I try to tailor my answer toward what it means to them personally, or better yet -- I try to get *them* to make the connection based on my answer, by asking them to tell *me* how that relates to what they're doing.

So how do we do this in a book? Not that well, but we try. First, we make sure that we talk to as many potential readers as we

can, to at least find out what the top two or three goals are for the majority of readers. Then we try to weave those in to the content. But we also try to include sections in each chapter where we talk about the *same* content from *multiple* perspectives, so that if the first way we frame it isn't the one that motivates you, perhaps one of the other ways will be closer to matching your personal interest and goals.

The real point is this:

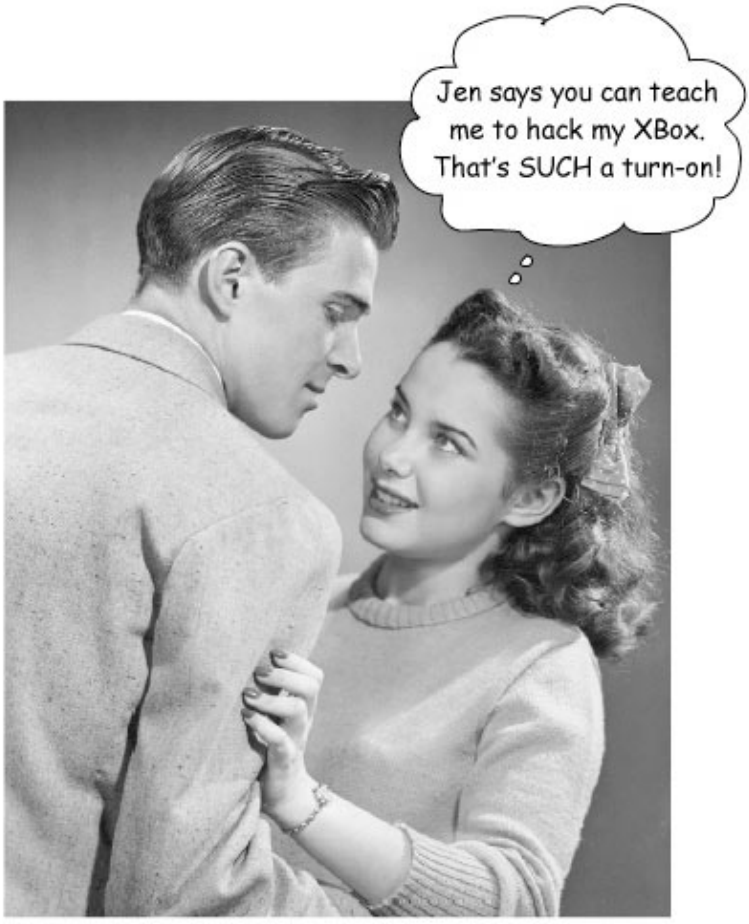
When it comes to your features and even benefits, one-size-does-not-fit-all. Try to find ways to connect what you do/have to what each individual finds personally meaningful. The good news is that it can take only the tiniest, subtlest shift in how you frame the information to help someone make that connection.

But you'll never know unless you *ask*.

http://headrush.typepad.com/creating_passionate_users/2005/01/we_should_all_t.html

Upgrade your users, not just your product

By Kathy Sierra on February 2, 2005



Learning is a drug. To the brain, learning new things is inherently pleasurable.

So if markets are conversations, why not use the conversation to help someone *learn*?

A lot of the marketing-folks-with-a-clue have begun talking about the need for brands (or whatever comes *after* brands) to offer something more *meaningful* to users. Just yesterday Hugh

talked about the marketing-spirituality thing, and Evelyn blogged on purpose-driven marketing.

The consensus seems to be that a user/customer today wants something to believe in. To be part of something *bigger than himself*. But if you're a customer looking for something to believe in, and you're looking Out There, why couldn't that bigger-than-you thing be... *a better YOU*.

What better way to give your users the "I Rule!" experience than to help them learn new things... maybe things that stretch them in ways they never dreamed possible. While you're upgrading your *product* to version 2.0, why not help upgrade the user's brain. Why not help build Person 2.0.

I bought a Nikon Coolpix 5700 because I wanted to get a little more serious about my photos--to do something a step beyond point-and-shoot. I wanted to learn more about photography. It's certainly in Nikon's best interest to help me get hooked on photography, because next thing you know... I'll be buying the extra lenses, and then pretty soon I'll have to get a better camera, and on it goes. IF they can get me to become passionate not about the camera, but about *photography*.

So they provide photography lessons on their site. Sure enough, I'm getting sucked in. I almost whipped out my credit card for a new lens just during the time I was researching this :)

And what you teach doesn't have to be about what you *sell*, if your product doesn't lend itself to something people could truly become passionate about doing. We talked about this with the garbage bag thing earlier. Yes you could teach them about issues around garbage, but perhaps it's more motivating to teach them how to make a *mockumentary* about the issues around garbage. Teach them something that might not be perceived as quite so cool, in the context of something that *is*. So maybe it's not so crazy for a company that makes garbage bags to teach video editing and movie-making, and help people have an outlet for those new skills. The Digital Garbage Film Festival.

Skyler learned to make her switcher parody on the Howard Dean site. Yes, the site was encouraging people to make "I switched to Dean" ads, that you could vote on, and the site included a complete set of instructions on how to make one. Storyboards, lighting techniques, everything.

Part of what we're trying to do on the Passionate Users blog is encourage people to use *learning* as a tool of choice in inspiring users, because it *works*. Learning is one of the fundamental reasons games are so engaging. For most games, the moment you have nothing left to learn is the moment you become bored and move on. Most teachers know that real self-esteem doesn't come from people *thinking* you're good at something... it comes from actually *being* good. Almost any activity gets better and better the more you improve, the improvement is nearly always a result of *learning*.

Musicians know this. Snowboarders know this. Programmers know this.

The more you learn, the better you are at something. The better you are, the more engaging it is. If you can help people have *more of that feeling*, they won't talk about how good *you* are-- they'll talk about how much *they* kick ass.

And that's a powerful formula for creating passionate users.

Helping someone become *more* than they were before is a wonderful gift to users and to the world. If your customers are older, they might not even realize they're still capable of learning so much, or that the new brain research on plasticity shows it's almost never to late to even become an *expert* at something new. You could change a life in a really cool way.

Now I want to see Microsoft help teach me to hack my XBox. Now THAT would be a turn-on... ;)

http://headrush.typepad.com/creating_passionate_users/2005/02/upgrade_your_us.html

Users aren't dangerous

By Kathy Sierra on February 3, 2005

Users aren't suffering from a highly contagious disease, but it sure looks that way given how hard some developers (and managers, and marketers) work to avoid coming into contact with a live one.

Bert was a software engineer for a company that sold software systems for managing broadcasting, so his users were radio and television station employees, and it was one of those dramatic examples of where the entire company revolved around the users. Everyone at that company--*everyone*-- had to do regular rotations through not just tech support but *customer training*.

Can you imagine that? Picture the programmer writing code *knowing* that at some point it'll be his butt in front of a room

full of confused users. And confusion leads to fear, and fear leads to anger, and anger leads...

The difference between having to come face-to-face with a user and *not* is staggering. Those of us who've been "on the front line" (or to use corporate speak -- *in the customer-facing positions*) know how stressful it can be, *especially* when your job is to support a product or service that basically sucks.

But when you're safely in your cubicle, where users are simply an abstract concept rather than real flesh and blood, what's the worst that can happen? You get a bug report, or maybe even a stern memo from upper management when the complaint or tech support calls get too high.

Those of us who've worked the line scoff at your little memo...*we're* the ones who get ripped a new one when the work built by the safe, protected people isn't right.

I gave a presentation to an all-hands meeting for a division of Sun, and I asked the group to raise their hands if they'd met a live customer in the last 30 days. Couple of hands went up. "The last 90 days?" One more. "The last *year*?" Another two. There were over 100 people in that room directly responsible for deliverables that went straight to users... in this case, Java training courses.

Without *really* talking to users the best you can hope for is to *meet* their expectations. You won't be able to craft that extra special *magic* that makes them passionate if you don't talk (and listen) to them.

This flies in the face of some software development models (and course development models) that believe if you've done your specifications right, there should be no need for the "workers" (programmers, writers, etc.) to ever come in contact with real users. That's just nonsense most of the time. Because even common sense tells us that what users are able to articulate *before* they have something is rarely a perfect match for what they say *after* they've actually experienced it. It's just like most market research... *people can't usually tell you in advance exactly how they'll react to something*. They just have to try it.

You just have to be there to watch. And listen. And learn. And then take what you learned and go back and refine, which is why the old waterfall model is pretty much the worst thing to happen to users.

What if you aren't in a situation where you can bring users to meet your employees? What if there's simply no way to get your developers out to watch and interact with real users in the field?

I found someone with a video camera and we grabbed customers coming out of a Java training course, where I was *hoping* to get some of them complaining on tape. Because the image and sound of someone yelling at you carries *way* more emotional weight than, say, reading a nasty letter sent by even the most irate customer.

And I did get a little of that. But I got something else, something I didn't expect, that I believe had a much bigger impact on all of us. Because what the customers wanted to express was how *important* this learning was to them. Course developers got to hear students explain what their courses *really* meant to these users--complete with fears ("Will I be able to learn it in one week? Will learning Java be enough to save my job?"), hopes ("I'm planning to take the exam next month and then transfer into a better department.), and dreams ("I'm going to use Java to program a game to teach kids the effect of ecological changes").

The developers got to hear how their work had a deep impact on real, living, humans. People with names, faces, and voices. From that moment on, the employees who watched that video lost the luxury of seeing customers as an abstract notion, and forever had the sound and image of real humans to haunt them when trying to decide if something with errors was still, "good enough to ship."

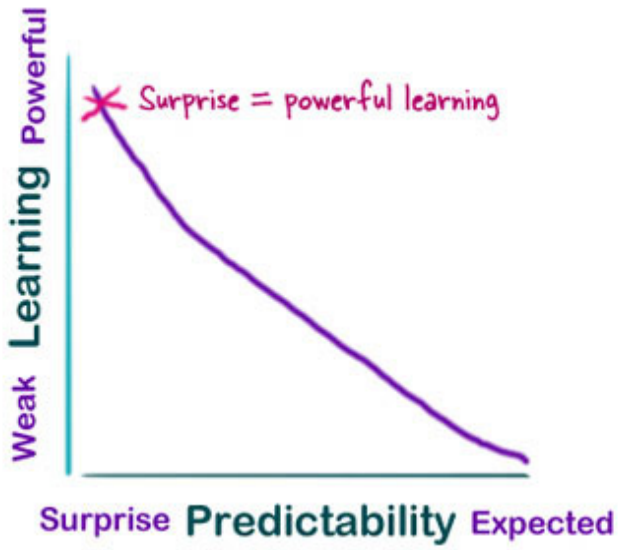
If you're a manager, I'll assume that *you* spend time with users. But if you don't make sure that your developers do, you're robbing them of the chance to learn *first-hand* just how important their work really is. It's so easy for so many of us to forget that the result of our work ultimately touches another person's life in some way.

Of course one of the downsides of creating passionate users is that when you *do* meet them in person, they might just want to hug you. :)

http://headrush.typepad.com/creating_passionate_users/2005/02/users_arent_dan.html

The WTF learning principle

By Kathy Sierra on February 4, 2005



The best learning happens when you're surprised... when you don't get what you expect. (I talked about this earlier in getting what you expect is boring.)

The brain is a prediction machine. It's constantly scanning to make sure that nothing fails to meet its predictions. And as long as everything is just as the brain expected, there's no need to wake up and pay attention.

Think about it... you come home from work, you throw your keys in the bowl on the little table beside the door (where you always put them) without looking. But then your keys fall straight to the floor! Someone moved the little table 6 inches to the right. NOW you notice the keys, bowl, table, and all your attention is on who moved your table and why. But had that not happened, you wouldn't have spent a synapse thinking about that table or your keys or the bowl. *Your brain would have gotten exactly what it expected.*

Think about the times you've done something that made intuitive or logical sense, but turned out to be SO wrong. The times where you've said, "Whoa--I'll never do *that* again." Those

are the memorable moments where you *really* learned something.

This is where so many teachers (and books) go wrong. In trying to make the learning smooth, and in a well-intentioned attempt to save the learner from having to learn the hard way, they simply *tell* you in advance what to do and what not to do. If there's a surprise lurking, they just tell you up front and spare you the trouble.

But they just robbed you of the chance to *remember*. To have that thing seared into your brain. What's worse, is that *after* they tell you how things really work, then they give you a lab exercise that simply demonstrates *exactly* what they told you. No surprises there, and your brain never really wakes up. At least not until someone really *hot* walks into the room. (Remember, at that point your brain is thinking... "UDP socket programming or survival of the species...")

If you're designing learning of any kind -- whether it's user documentation, training courses, or something to get your users excited... *be surprising*. If there's a gotcha, or anything that might be surprising in either a good OR bad way, for gosh sakes *don't just spit it out*. Give them a chance to experience it either for real, or at the least -- as we do in our books -- by weaving a story that leads them right into the trap, springing it on them when they least expect it.

Put a post-it note on your computer that says "Surprise!" Practice surprising your friends or co-workers. ***Do something unexpected every day*** until it becomes a habit to look for the opportunity to surprise. (This does not mean that everyone will appreciate your surprises, of course. After all, you really *did* look better as a blonde...) Valentine's day is coming, so you might as well start prepping now :)

http://headrush.typepad.com/creating_passionate_users/2005/02/the_wtf_learnin.html

F*** the rules!

By Kathy Sierra on February 7, 2005

How often do you question The Implicit Rules? How often do you challenge The Assumptions? How often do you make sure that you're not doing something a certain way simply because *that's the way it's done*?

How often do you recognize when others are judging, criticizing, or trying to guilt-trip you based on some unstated rule about How People Are Supposed To Be? How often do you call someone on it? When I hear someone say, *"Everybody* does it this way", or "*"Everybody* knows this is appropriate behavior under these circumstances...", I try to ask, "Who *wrote* that rule?"

I believe that The Rules are the leading cause of crap products, frustrated users, and unhappy relationships. I'm not talking about *all* rules and standards of course, just the largely-unstated-but-blindly-accepted ones that:

***Never* made sense.**

No longer make sense.

Make sense, but only in a *different* context.

One of the things that makes challenging the rules so damn hard is that other people have *so* much invested in... *keeping the status quo*. And of course the minute you question a rule, you're potentially threatening the people who've been following that rule... *even if those people don't understand the rule either!* Simply *hinting* that there could be a **better way** is enough to trigger someone's defenses.

Another problem with questioning the rules is that they're so pervasive, and we just come to accept them. Just as memes and urban legends and "talking points" become real simply because they're *repeated*, rules take on a life of their own even when nobody can remember why they exist in the first place!

And the rules exist in every last part of our life both personal and professional. (Did you wait too long to make that phone call? Did you dress appropriately? Are you using language appropriately? Did you file the necessary papers? Did you use your advertising budget for *advertising*? Did you charge your user for the tech support call? Are you being "a team player"? Did you cover all of the topics in the book?)

A huge chunk of the implicit professional rules today are damaging because they inhibit innovation. They stop the one thing businesses need the most--*breakthough ideas*.

Yes, one could get into trouble at work for asking too many questions about The Rules, but what have you got to lose? Your job? That might have been relevant a few years back, but it's becoming less a factor today. Sure, you definitely *could* lose your job (like, um, *me*) for asking too many "but, WHY?" questions, but if you *don't*, your entire *company*, or perhaps an entire *industry* could start slipping under the waves, anchored by inhibiting, outdated, or just plain stupid rules.

At Sun, I used to hear "Customers don't want that." Or "Customers need you to do it this way." And I'd always ask (nicely at first), for the evidence. "How exactly do we KNOW this?" Who decided that? If we ever did know that was true, is it STILL true? And on I went... you can see how annoying I was.

I told the rest of the team that we should put each rule on trial for its life. Make it sit in the middle of the room and *defend* itself. And if it couldn't come up with a good enough reason to live... out it went. At first, of course, it was a "cute" idea that everyone had fun with, but eventually I was shut down when one manager's response to one of my "But why are we doing it this way?" questions was, "Because that's what upper management said, and so that's what we're going to do! End of discussion!" Based on *my* response to that statement, I became known as "short-timer Kathy" from that day on...

If we'd followed The Rules when designing the original Head First book (Head First Java), we would have been simply one more of the 2,000 other currently-selling Java books, with virtually no way to break past the established, well-loved existing books. Even a spectacular marketing campaign would not have been enough to even earn back our advance. We simply looked at every rule and constraint and said, "let's pretend these don't exist... what can we do to implement the metacognitive learning principles so that people can learn more quickly, with better retention, than with most traditional approaches?" Because of their willingness to challenge the rules, it's O'Reilly (and not Sun Press) that has the top four selling Java books right now. :) The amazing part to me is that O'Reilly, having *invented* and established many of those rules, had a lot more at stake in breaking them, but that's Tim for you.

So... what implicit the-way-things-are rules are *you* accepting without question? What are you taking for granted and assuming? What rules can *you* put to the test? My favorite words for this:

"Why?"

"What happens if we *don't*?"

"When was the last time anyone verified that?"

"Is that *still* true?"

and the best one...

"Who *wrote* that rule?"

UPDATE: Oh wow -- just in case you didn't see this in the comments for the F*** the Rules post, Dave Wheeler (his blog) created a 4-page PDF on the The Business Lifecycle of Rules that really puts it all in clear perspective.

If I still *had* a real job, I'd make copies of this and sneak them under everyone's door or at least leave them scattered around the photocopier and break room. Excellent Dave!

http://headrush.typepad.com/creating_passionate_users/2005/02/f_the_rules.html

Spiral learning

By Kathy Sierra on February 10, 2005

Spirals show up everywhere from fractals to nautilus shells. Software developers know the spiral as iterative development--a model in stark (positive) contrast to the old linear waterfall model.

One huge problem with the waterfall model is that in its traditional form, it's *not based in reality*. It assumes that it's entirely possible for each stage to be done perfectly (and permanently) and then *thrown over the fence* (or cubicle wall) to the next group in the system. Nice theory, that. The guys doing the requirements finish their job and then, hey, they might as well all go on vacation. Their work is done. And so on

down the line until the product is delivered to the users. The name itself (waterfall) describes the key limiting characteristic of the waterfall model--*it's one way only. Water doesn't go back up.*

User experience designers (especially with *games*) often use a spiral model to keep cycling the user through stages of interest/motivation, engagement, and payoff (I described the user experience spiral here.

But where software developers and game designers use the spiral model, *learning* designers (teachers, instructional designers, tech book authors) often *don't*. Yet a spiral model most closely matches how learning *really* happens.

The typical training course or technical book takes a linear approach to the topic, teaching each topic completely before moving on to the next. Each topic/phase in the course depends on having mastered the previous topic/chapter. ("OK, *that's* done... now we can move to the next one.") This is usually wrong on so many levels...

By teaching a topic completely in one section/chapter, there's probably way too much cognitive overload. When learning a new programming language, for example, do I really need to learn *every possible way to write a loop* before I can move on to, say, object interaction? If you teach me *only* a ***for*** loop, for example, I can just move on to what I *really* want to do... repeat something (or iterate over something).

By taking the "now we're on loops, so let's look at ALL the details of EVERY kind of loop syntax..." you've just postponed (delayed gratification) what I really want to do-- *use* a loop to do something interesting.

A spiral model lets you do what our editor Mike Loukides refers to as:

Give them the minimum new knowledge and skills needed to be creative.

Learning should work *just like a game*. The spiral looks like this:

1) Get me interested (make the case for why I should be motivated to learn this).

2) Give me a challenging and engaging activity (learning this new thing).

3) Give me the payoff/reward for having learned this (let me *apply* what I just learned to something interesting and meaningful, or at least *fun*).

4) Repeat with new thing that builds on what I now know.

By taking an iterative--rather than linear--approach to each topic, the learner gets to do more interesting things more quickly. If you force me to do each topic to death before moving on, I might have to wait until the frickin' end of the book or course before I can actually do anything really cool. And that's a motivation killer for sure. And without motivation, learning suffers dramatically. How many of us have left a course knowing that we were *exposed* to a lot of content, but we still can't actually *do* anything?

Another benefit of the iterative/spiral model for learning is that the spiral approach is much more forgiving. If the linear model relys on "we're only going to do this topic once, so you better pay attention!" and assumes that I've completely learned that topic before moving on (made less likely by the fact that I'm given too many details about the topic), then if I really *didn't* nail it, I'm screwed moving forward.

But by iterating through the topic, I get another chance--potentially *many* more chances--to revisit the topic. So if I'm still a little fuzzy on the details the first time through that topic, then when it comes up again in a later iteration of the course/book, I get another chance to get or reinforce more clarity. Maybe I didn't quite get it the first time, even though I was able to *use* it, but perhaps the new things I've done since the last time I saw this topic have given me a better perspective. So the second time we come back to it, I'm in a better place to ask the right questions and see this topic in a broader context.

Learning should use the spiral experience model just as a game does. Each new thing I learn should be a chance to help me "get to the next level." Iterating through the topics means revisiting the *same* topic in multiple places (if needed). So each iteration through a topic gives me *just what I need and no more* to do something creative with what that new skill/knowledge. If I need to learn more before the course or book is done, then come back to it later... when it's needed for something new.

Obviously I'm not talking about a *reference* book, but a reference book and a *learning* experience are wildly different beasts. They have completely different goals. The problem is that most books aren't really *sure* whether they're for learning or reference, or worse--they try to be BOTH. Reference books should be designed in a linear model. Learning experiences should be spiral. That's a dramatic difference, and you can't shoehorn a spiral experience into a linear format without weakening both.

[Disclaimer: We (Head First authors) suffer from a little too much linearity in our certification study guides, because the exam tests people on details that go way past what they need to actually *use* the topic. So we've tended to do a much better job of topic iteration in our *non*-certification books than our exam prep books, but really, there's no good excuse for why we haven't done more to iterate even through the you-must-know-everything cert topics. We promise to do better with our cert books in the future.]

When you're communicating new knowledge to your users:

What's the minimum you can give them that'll let them be creative?

Iterating through topics lets them do more interesting things more quickly. If they need more on a topic, they'll either get it later--on another trip through the spiral--or you can simply point them to a reference where they can learn the rest of the details when they need them. The point is, **get them having fun and doing interesting things as quickly as possible!**

http://headrush.typepad.com/creating_passionate_users/2005/02/spiral_learning.html

Six Degrees of a User

By Kathy Sierra on February 14, 2005

How many people are between you and real users? Each person in the middle is another point-of-communication-failure, and by the time the message gets back to the real engineers, god only knows what's happened to it.

We had a phone call with Tim O'Reilly a couple days ago about some *communication* problems of our own, and his theory was that we were all suffering from "The Telephone Game", where each iteration of the message lost information. *Entropy sucks.*

I talked about this before in Users Aren't Dangerous, but it's tricky to do. In Los Angeles I once worked for what had been the coolest training company on the planet, Mind Over Macintosh. (There's no link, because it no longer exits.) The owner, Bruce Kaplan, was a brilliant marketer and creative force... he was largely responsible for bringing places like the LA Times into the

digital/desktop publishing world, and then repeated this again by introducing much of Hollywood to new media.

In some other blog I'll talk about more of the amazing things he did that made the place so special, but here I want to mention the one that struck me as the most obvious difference between his company vs. Sun Education--*talking to customers*. When I first came there to design and teach courses in interactive multimedia, Bruce would suggest that we have personal conversations with each student *before* they ever showed up to class. While most students were at first surprised that the teacher of their upcoming course was calling to chat, *everyone* agreed that it made a huge difference. I knew exactly who was coming, what they wanted and needed, and I could usually tailor the course around the students who would be there that particular week, based on what I'd learned. By the time they showed up on the first day, we'd already established a relationship.

In a few cases, we were able to stop someone from ending up in a course that wasn't right for them, and could steer them in a better direction (even if that meant they ended up with a different vendor). This practice of talking to every student before the course started became standard practice for me, and I couldn't imagine doing otherwise.

Until it was time to teach my first course as a Sun employee. I (silly me) asked my manager for the student phone list for my upcoming course. She looked at me like I had just asked for an AK-47. "You want to *what*?" she asked, as though the notion of the instructor phoning the students was bizarre and unthinkable. Clearly, only official Marketing or Customer Service employees had direct phone contact with students. "But these people are going to be spending 40 hours with me next week... so it's not like I won't be talking with them *then*." And while there wasn't exactly a *rule* that said instructors-don't-talk-to-students-prior-to-the-course, it was just beyond anyone's imagination why I'd want to do such a thing. I was of course horrified that they didn't have a policy *requiring* instructors to talk to their students!

And it showed. I was once asked to teach a custom advanced enterprise Java class at a customer location, where a previous instructor of ours had already taught it and the customer was upset with it. My job was to go in and try to give them what they

really wanted. When I spoke with the customer's representative, he described what went wrong:

"The instructor came in and started teaching. The students quickly realized that the level of the course was too introductory for these students. But the poor teacher was constrained by his slides, so he dutifully went through the course doing the best he could under the circumstances."

There's so much wrong there that I hardly know where to start. The idea of an instructor being "constrained by his slides" is insane, but that's a different (bad) issue--the notion of having the course materials completely drive the course! (I'll have a lot more to say about the use of slides/presentations in another entry, but the folks at Missing Link know a *lot* about presentations).

But the whole thing could have been avoided had the instructor spoken to the students *in advance*, so that he'd at least have known what they were *really* looking for. Instead, he was forced to rely on the message that came through three people before *he* heard it--the sales rep, the custom course developer, and then his manager who scheduled him for the course.

Anyway, I'll have more to say about Bruce later because contrasting what he did to the other training companies I have worked for couldn't be more dramatic. Just one of his insights was, for example, that the kind of coffee you served in the break area could actually be the deciding factor for a customer. His brochures (which were actually collectible posters) specifically *mentioned* the coffee. So you might wonder why Mind Over Macintosh no longer exists... given how fabulous it was? Because Bruce eventually sold it to a Big Corporate Training Company that sucked the soul out of it (I won't mention names), starting with the name change.

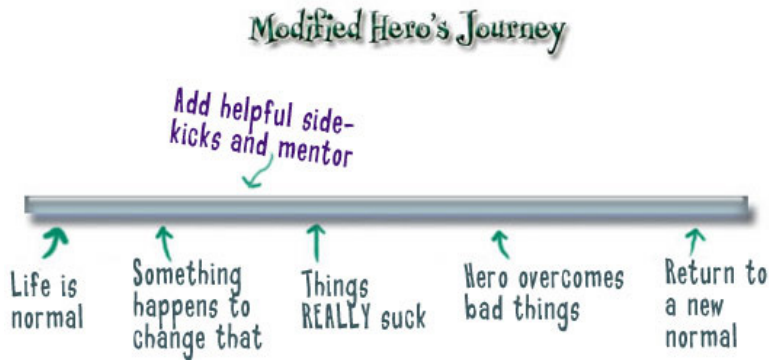
(But now he's living another creative life and dream as a musician, playing mandolin with his wife Claudia.)

So, if talking to customers/users can be such a simple thing, why do some companies find it so hard and strange to do? Why was it that what was *unthinkable* at one place (to not talk to customers), was the status quo at another?

http://headrush.typepad.com/creating_passionate_users/2005/02/six_degrees_of_.html

The User's Journey

By Kathy Sierra on February 15, 2005



Lord of the Rings. Starwars. NeverWhere.

A beer commercial. Linux. College.

Viagra ads. Learning Java. Starting a business.

What do they all have in common?

Things are normal. Things become challenging. Thanks to the help of friends and perhaps a mentor/wizard, you're able to overcome the challenges. You return to the *new and improved* normal. A hero.

What would happen if developers/marketers/teachers tried to help users experience a kind of a hero's journey, and offered a way to help them through each stage? Unfortunately, too many products or services don't give the user a chance to get past the initial crises ("Help! I can't make your product work!"), and the user never ends up... a hero. They end up failing. Quitting. The "I Suck" experience instead of "I Rule!" And since users are increasingly less likely to take all the blame, your company or product is Sauron. Sure, the user was defeated... but only because Your Company Is Evil. As a developer of learning experiences, I desperately don't want to be the enemy. (I always fancied the *trickster* role though...)

The opposite (and sometimes just as bad) experience is where your product or service offers nothing interesting or challenging, or it doesn't try to at least inspire the user to *do* something interesting or challenging with it. No Challenge = No Hero.

One of the most powerful aspects of the Hero's Journey is that the hero comes out the other side *better* than he was before. (Or as Michael puts it... *bigger*.) Are you supplying a reasonable challenge, and then offering a way to move through the stages of that challenge and ultimately come out changed for the better?

Obviously not all products and services are--or need to be--particularly inspiring and challenging. I'm thinking toilet paper doesn't need to, um, take me on a journey. But... that doesn't mean there isn't a way for a company with an utterly (and ideally) unchallenging product to be associated with something meaningful. *Something that upgrades the lives of their users.*

The user's journey doesn't have to be *about* the product. It can be about something related to the ingredients in the product, or the design, or the company, or the employees, or causes supported by the company or...

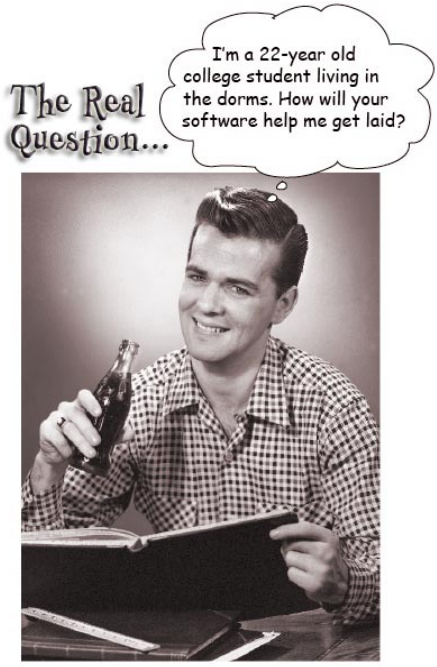
If your product or service is daunting for users, or what they *do* with the product or service is challenging, you can welcome that as a great opportunity to give users the "I Rule!" experience. It means you'll have a much easier time taking them on a little hero's journey. If your product or service (or what users might do with it) is *not* challenging, then you can still ask, "What can I do to inspire our users to take on a new challenge?", and then somehow craft a challenge (suggestion: teaching your users something cool and rewarding is often an easy answer).

So, what are *you* doing to help your users on a hero's journey? What can you do to associate what you do/make/sell/write/build with a hero's journey? What can you do to help your user through the "I Suck" phase and into the "I Rule!" phase?

http://headrush.typepad.com/creating_passionate_users/2005/02/the_users_journ.html

What Users Really Want

By Kathy Sierra on February 16, 2005



The quote in the picture is slightly paraphrased from a brilliant rant by jwz (owner of DNA Lounge). The original (but you have to read the whole thing for context ;):

"So I said, narrow the focus. Your "use case" should be, there's a 22 year old college student living in the dorms. How will this software get him laid?"

The piece centers on designing (and *spinning*) products for what the *user* wants, rather than... oh, never mind. I'll just put in another quote because I can't say it as well:

"If you want to do something that's going to change the world, build software that *people want to use* instead of software that *managers want to buy*."

(Thanks to Jed Cousin and Nathan Torkington for the link!)

http://headrush.typepad.com/creating_passionate_users/2005/02/what_users_irea.html

Creating passionate fans

By Kathy Sierra on February 17, 2005

Musicians know a lot about making and keeping fans. Last night the four Headrush bloggers (me, Eric, Beth, and Bert) went to a sold-out Finn brothers concert at the Boulder Theater, and two amazing things happened:

First, the show was nearly cancelled because Neil had a severe case of the flu. *He was in the hospital in San Francisco the day before*. But they pumped him full of drugs (and apparently a single-malt scotch) and he somehow managed to get there, arriving minutes before the show was to start. I hadn't even been a Finn fan prior to the show (Eric and Beth talked us into it), but I have a really soft spot for those who leave a hospital rather than disappoint the customers/fans. :)

But something more important went on throughout the night...

They played the songs they've been playing for *twenty years* as soulfully and passionately as though it was their first time. As though we were their first and most important audience.

Think about how damn many times over the years they must have played Neil's biggest hit, "Don't Dream It's Over". *Thousands*. But it was achingly beautiful last night despite what might have been, like, the 3,042nd time they've played it live.

But I'll come back to that in a minute.

It got me thinking about how good some artists and bands are at loving their fans. Or at least getting their fans to love *them*. I've seen Coldplay three times in the last couple years, in three different venues. (I love live shows). Each time was amazing, and Chris Martin was always inspiring. But the last time was incredible--it was at one of the most magical concert spots in the country, Red Rocks Amphitheatre. Red Rocks is outside, and the

concerts are held during the summer when it *should* be warm and gorgeous. But not this night. It was pouring rain, from start-to-end, and near freezing.

But the fans stuck it out, shivering and huddling under plastic tarps, blankets, and trash bags and everyone was drenched. Chris Martin kept mentioning how *grateful* he was that everyone was there putting up with this. He even apologized for the weather!? But at the end, when he should have been as anxious as anyone to just get the hell out of there, he said he was going to do something they never do... *an extra encore*. He told us that he felt so bad about what we'd been going through that he wanted to do something special for us, so they came back out *again* after their last encore, and then did something they'd never planned on... and started playing.

We felt like we were the most special audience they'd ever played for. Here we all were, completely miserable, and still thinking we were lucky to have been part of that show, and that we experienced something nobody else would.

Skyler is a fan of the UK indie band Travis. They don't tour the US much, so it was a big deal when they came to town when she was 14. We got there hours before the doors opened to get a good spot, and before lining up we went around the back of the building, and there stood the band's frontman/lead singer Fran Healy. What happened next was astonishing (keep in mind that while Travis is mostly unknown in the US, they're HUGE in the UK. This is not your local bar band):

Fran: "Hello there, I'm Fran." (as if she might not actually know that!)

Skyler: "Hi, I'm Skyler".

Fran: "Hey, you're from the message board!"

Skyler: (stunned) "Yes! Wow!"

Fran: "It's great to meet you in person Skyler. Is this where you live?"

(On it goes, with two of the other three band members coming out of the bus to say hi.)

Think about that... it means the band actually *reads* their message board posts, enough to have recognized Skyler as a frequent poster.

So you probably all have a ton of stories about a band or artist that really made you feel like they cared deeply about their fans, but I wanted to come back to that part about singing a song as though it were the first time. I've thought about how many times I've taken classes from teachers who you *knew* had been teaching this class *forever*. You knew because it showed. They were barely present. You might know it as the "phoning it in" effect. You've almost certainly been there.

So that's the question... how do you keep your work feeling inspired and passionate? Fresh? If you're a manager, what can you do to help your employees stop sliding into the phoning-it-in stage? Obviously putting them under constant stress isn't the right idea, but what about making sure they have chances to have variety in their work, or at least occasional chances to work on a different kind of project or role, at least temporarily, to step back and look at their work differently.

How can you keep your own work from suffering from phone-it-in? What can you do so that when you sing to that audience after twenty years, you leave them feeling as though this was your debut night, and they were the most special audience you'll ever play to?

http://headrush.typepad.com/creating_passionate_users/2005/02/creating_passio.html

The power of One

By Kathy Sierra on February 19, 2005

If you asked the head of a company like, oh I don't know... *Sun* for example, which employee he'd prefer: the perfect team player who doesn't rock the boat or the one who is brave enough to stand up and fight for something rather than accept the watered-down group think that maintains the status quo (or makes things worse), which would he choose?

In his book Re-imagine", Tom Peters says, "We will win this battle... and the larger war... only when our talent pool is both deep and broad. Only when our organizations are chock-a-block with obstreperous people who are determined to bend the rules at every turn..."

I'm guessing there aren't many CEOs who'd publicly disagree with Tom on that.

So yes, I'm thinking Mr. CEO of Very Large Company would say that their company should take the upstart whatever-it-takes person over the ever-compromising team player. "If that person shakes us up, gets us to rethink, creates a little tension, well that's a *Good Thing*", the CEO says. *riiiiiiiiight*. While I believe most CEOs *do* think this way, wow, that attitude reverses itself quite dramatically the futher you reach down the org chart.

There's a canyon-sized gap between what company heads *say* they want (brave, bold, innovative) and what their own middle *management* seems to prefer (yes-men, worker bees, team players). Oh, you won't actually *hear* any manager say that... but you see it over and over again in their choices. When the tech downturn hit, wouldn't you know it... the less-than-team-player folks were the first to go in layoffs. Yet, these were probably the folks the company most needed when it became painfully clear that *business as usual* was failing horribly.

Just one of the many problems with the whole team player thing is that you (the one accused of NOT being one) have almost no defense against it. In the business world (except at the top or in certain industries), team players are thought to be filled with inherent goodness. Those who challenge the status quo *against the team* are viewed as hurting the culture and productivity of the team. Mavericks, they call us. Cowboys. Lone wolfs. *Trouble makers*. That's not completely untrue. Teams where everyone is completely in sync with little disagreement *are* more productive.

But the question is... productive at *what*? Because team think usually promotes doing things exactly the way they've always done them. Not exactly a recipe for being totally f'in amazing.

Team thinking leads to incremental improvements, and prevents *revolutionary* ideas.

Revolutionary thoughts are, by definition, *thinking outside the team*.

Purple Cows just don't usually come from teams working together to reach a solution. Purple Cows come from the wild-ass idea one guy had in the shower. That doesn't mean he can't be part of a team, but the more unusual an idea is, the more resistance it will get from a group, and that's often enough to

suck the life out of an idea. Or it goes from being a *purple* cow to one that's merely a *slightly darker shade of brown*.

I'm not dissing teams--our books are all collaborative efforts, and far better because of it. And we consider ourselves to be on a team that includes our publisher O'Reilly. It's not teams that are the problem, it's the rabid insistence on *teamwork*. Group think. Committee decisions.

Most truly remarkable ideas did not come from teamwork. Most truly brave decisions were not made through teamwork. The team's role should be to act as a supportive environment for a collection of *individuals*. People with their own unique voice, ideas, thoughts, perspectives. A team should be there to encourage one another to pursue the wild ass ideas, not get in lock step to keep everything cheery and pleasant.

I simply don't buy into the "none of us is as good as all of us" as fact. While it's often true, it's just as often *not*. There are times when you can and *should* step back and say, "Not only am I *as* good as all of us, I'm actually *better* at this particular thing, because the entire team is headed in the wrong direction, and there's too much inertia to get the whole damn team to turn around at the same time." Obviously a manager doesn't want total anarchy and chaos from each individual thinking their idea rules and everybody else is an idiot, but somewhere there's a balance, and the heavy emphasis on teamwork/teamplayer-ness is tipped way too far in the non-individual direction.

I consider "There's no 'I' in Team" to be terribly depressing. It sounds, in fact, just like what the Borg said on Star Trek. There is most definitely an "I" in any team *I'm* on. *I* have value in, and out, of a team. I will not surrender my passion in order to be a team player. And any team who doesn't value that isn't a team I want to be part of. I do believe that a team can change the world, but it's still a team of individuals supporting each other in being brave, strong, innovative, and passionate.

There is an "I" in PASSION.

http://headrush.typepad.com/creating_passionate_users/2005/02/the_power_of_on.html

Point of view matters

By Kathy Sierra on February 21, 2005

My previous post on the power of One was from the perspective of the individual on a team. But then Eric Titcombe made a great comment that could reflect the manager's point of view. And that reminded me of how the POV of people in different job roles and departments can be so different. Marketing (and/or sales) folks have a point when they say that without them, the best product in the world won't have enough customers (although that picture is changing). The engineers have a point that without them, the marketers have...nothing. Who is right? Does it matter?

I developed a game for Virgin Sound & Vision (a part of Virgin which had still been owned by Richard Branson and was focused on younger games), called Terratopia, that had 300 people working on it in one form or another. And exactly one

programmer... *me*. I thought I was the center of the world for that game. I mean, come on, without the programmer, there WAS no game. But then when the credits were created (and rolled to look like movie credits), I came in around number 12. I didn't really care, but it still shocked me that I, center of the world, wasn't #1. But of course the producer assumed that HE was the center of that product, and the art director thought HE was, and the story's creator thought SHE was, and the lead designer, and I think even the sound designer/composer thought *he* ought to be above the *coder*.

So it was all a matter of perspective.

But what I think is far more important than recognizing that each part of the world in which the product or service exists has a different POV, is finding a way to make sure these different people *talk* to one another. I couldn't believe how few cross-departmental meetings we had when I was at Sun. Here I am complaining about people not ever talking to an external *customer*, when an even deeper issue is that so many people never talk to *anyone outside their own department*. And that's just crazy.

A couple of months ago I was at the Sun campus working on the new version of the Java programmer certification exam, when I bumped into the marketing guy to whom that exam belonged. He asked what I was doing there, and I told him how we were working on the new exam, and how cool it was. He looked at me strangely and said, "but there's really nothing new there, right? It's just yet another version of the same old exam." So I looked at *him* strangely and said, "Are you KIDDING me? This is a profoundly different exam in so many ways, and..." off I went, detailing all the reasons why I thought there was indeed a Big New Story here. I invited him down to the meeting room where he could meet the entire exam development team and interview us while we were right in the midst of it. He was seriously surprised, but in the end... delighted.

Whether the right people from other departments know about the exciting things you're working on should *not* depend on whether a contractor crashes into them accidentally in the cafeteria.

If you're producing a product, and you're the engineer, for frick's sake find a way to make sure the marketing and sales people hear how exciting it is. Don't wait for official department

channels, *just tell them*. If YOU show how excited you are, chances are you'll infect *them* and god knows, the marketing folks could probably use a little passionate enthusiasm from the ones who deliver the goods.

And vice-versa. If you're in marketing, why aren't you *really* spending time talking to the ones who do the work? If I ruled the world and were a manager, I'd absolutely insist on getting these groups to meet face-to-face (or at least on web-cam) on a regular basis, not just at the annual company picnic.

And if both sides spent more time learning what the other folks *really* did, they might *use* that new knowledge and appreciation in key ways. One of the worst things that can happen to an engineering team, for example, is when *marketing* suddenly schedules a "press opportunity", which means... an *impressive demo*. I was once given two weeks to come up with a version of a game (All Dogs Go To Heaven, for MGM) that was going into a box of Cheerios. [perky voice]"You can't miss on this one, Kathy,--this is the first CD-ROM to ever go into a box of cereal. But we just KNOW you can do it" :) [/perky voice]

I was horrified, yet not surprised, that marketing had yet again promised something that would kill me to deliver. And I couldn't help thinking that if these folks really *knew* what we did and how software development worked (especially on games), they wouldn't just sign us up for stuff with abandon. On the other hand, at that time I had zero appreciation for what their job or life was like in marketing, so I considered them just a big fat annoyance. People whose sole responsibility was to mess up our schedules and way over-promise the press and clients.

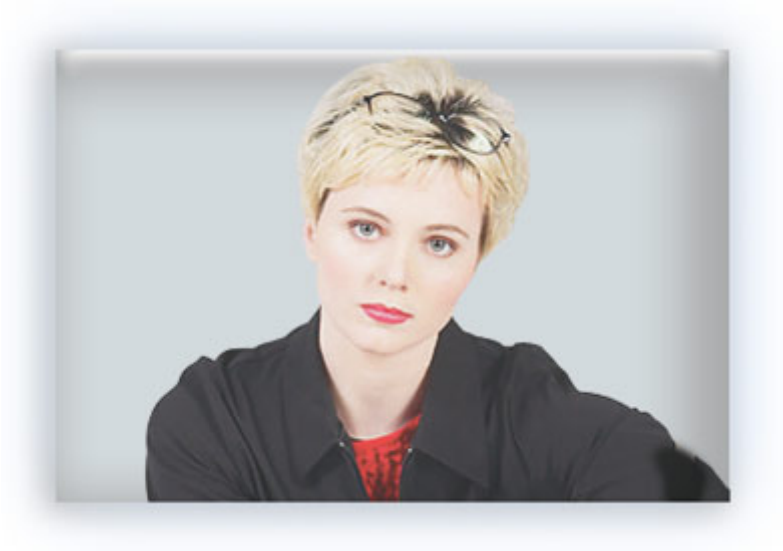
(Footnote: it turned out that I got *four* weeks instead of two, because just before they shipped it, someone realized that the protective sleeve around the CD-ROM might actually be harmful to the cereal. So I very nearly became the girl responsible for delivering *the game that killed kids*.)

So, what are you doing to see things from the POV of the other folks involved with your product or service? If you're a manager, what are you doing to encourage the conversation *within* the company?

http://headrush.typepad.com/creating_passionate_users/2005/02/point_of_view_m.html

Can you teach someone to care?

By Kathy Sierra on February 28, 2005



You usually can't create passionate users unless you deeply *care* about them. If you didn't, you probably wouldn't be reading this blog. But what about the other people on your team? How do you get *them* to care?

Obviously you can teach *customer service* skills. You can teach *active listening*. But can you teach them to *care*?

No.

But you can *infect* them.

Passion is infectious, and so is caring. The brain usually can't help sliding toward the behaviors of those that brain is around. So if you want people to care, make sure the culture of your environment has hit a critical mass of caring.

I worked for the Sports Club LA/Reebok for a few years, and one of my jobs was to write software that trained employees on customer service skills. Each of the several thousand employees in the entire company had to go through the same customer service training program. But we noticed that at some facilities, virtually 100% of the employees were nice-as-pie to the customers, while in a couple other facilities, you'd think it would

kill some of the front desk staff to even *smile* at a customer let alone help them out with anything. What was the difference?

Critical mass.

In the places where the service was awesome, the norm was to treat the customers like gems. If a new employee started working the front desk, for example, and didn't say goodbye to a customer as they walked out, *everyone noticed*. The rest of the people there would turn around with an odd look. Not a condescending or angry look, just... that it was *strange* to not hear someone say goodbye to a customer. The norm was to greet and say goodbye to customers, and anything that violated the norm was really noticeable.

But in places where the service sucked, that culture didn't exist. If a new employee started working the desk and didn't greet a customer, nobody noticed. Nothing out of the ordinary.

We fixed the situation in less than two weeks by taking the front desk employees who couldn't *imagine* not greeting the customers and sent enough of them out to the other facilities until we thought we had critical mass. *It worked*.

There's another question, of course, which is, "Yeah, but weren't they just being fake and going through the motions?" Just because they had the *behavior* of caring doesn't mean they actually did. That's true, although in many cases, it doesn't matter all that much as long as the behavior of the "faker" is indistinguishable from the Real Thing. But it would matter, in the end, because sooner or later that employee would be put to the test.

But that's where the brain kicks in... because the brain *can* get "infected" by an attitude of caring. It's not guaranteed, of course, but just as having a teacher or friend who is enthusiastic about something can eventually cause *you* to start genuinely liking that thing, you can be infected by being around enough people who really do make caring a top priority.

The tougher job is when you don't *have* critical mass and you somehow have to get there. And *that's* when you need to bring in The Big Guns... **real customers**. Most often, when people don't care about the users, it's because they don't see users as real people. They're just abstract concepts. But if forced to meet one face-to-face, or at least see some in a video talking about real needs, hopes, dreams, concerns, they'll have to start seeing

them as real humans. And unless you've got sociopaths on your team (and I have a former manager or two I might put in that category ;)), it'll be hard to keep them from feeling *something*.

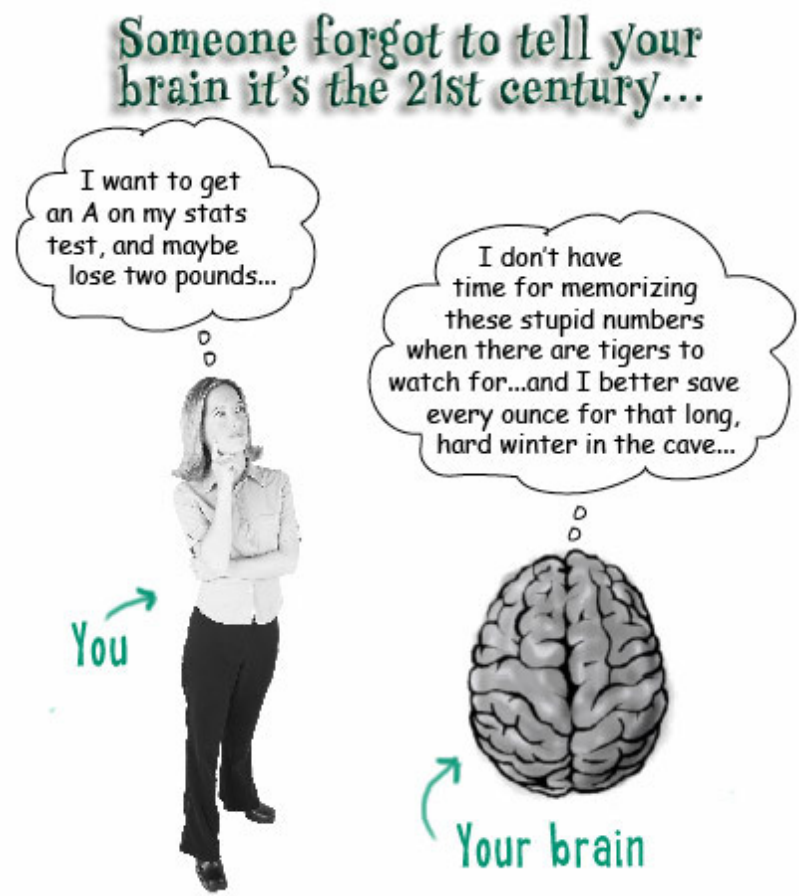
So you can't *teach* caring (although you can certainly teach ways to *demonstrate* caring to users), but you can use the brain's built-in tendency to model what it sees in others to *infect* the newcomers. And by finding ways to keep users "real people" instead of spreadsheet numbers, a critical caring mass shouldn't be that tough to hit.

I heart users :)

http://headrush.typepad.com/creating_passionate_users/2005/02/can_you_teach_s.html

Dealing with a legacy brain...

By Kathy Sierra on February 28, 2005



You thought dealing with legacy *code* was a challenge... the code in your *head* is thousands of years out of date. Plus the docs are really sketchy, and there's nobody alive who knows how to refactor it. But if that's what we're stuck with (at least until Ray Kurzweil's future gets here), we have to figure out ways to live a 21st centry life with a brain that thinks you're still living in a cave surviving on berries and mammoth meat.

A big part of our goal at passionate users is to find workarounds or ways to *trick* the brain into thinking that the content in your stats textbook is as life-threatening as a tiger, but it was a comment on my last post (from P-daddy) that reminded me

that's it's not just info attention/retention we're fighting for. A large part of why it's so frickin' hard to lose or even maintain weight is that your brain/body thinks it better save *everything* to prepare for that long winter.

(One of the reasons that most calorie-restricting diets make things *worse*... you're just cofirming what your brain was already worried about, and it says, "HELLO! We're *starving* here! If you thought I was hanging onto everything *before*, well now things are drastic, so I'm going into all-out protection mode." And you end up with an even bigger fight. You'll have to wait until we start a fitness blog to hear our thoughts on how to work around <i<that. Tip: weight training with *heavy* weights is the crucial part, because it sort of "tricks" your body into thinking that you're *growing*.)

But *knowing* what your *brain* is motivated by is half the battle. Because you can't change it, but you can work with it. The biggest challenge is that you can't simply consciously order your brain to care. You can't tell it, "OK, I know this *looks* really dull, but trust me--I'm screwed if I fail that exam on Tuesday..."

For learning, one of the best things you can do is *whatever it takes to convince your brain that what you're learning is life-threatening or life-saving*. What does your brain think is important? Novelty. Surprise. Sex. Danger. Shocking things. Stories. Human faces. Pleasure. Things that make you emotional. Things that move you, and things that cause you to move. Things that cause you to think deeply. Solving puzzles. Stories.

See the problem there?

Your stats textbook probably doesn't warrant a checkmark next to *any* of those. So, you'll have to retrofit it yourself. To trick your brain into thinking that what you're learning is important, find ways to add some of those things into what you're studying. But you can't do it by passively reading! Here are a few tips, though:

* Write notes, and read them out loud. Just *talking* helps your brain.

* Write notes as poems, haikus, limericks, songs, and say or *sing* them out loud. One guy we know quite literally writes songs on his guitar, and then records them as mp3's and shares them with others.

* Create a tiny little play, and have you and your study partners *act out* the parts of different components in the system. If you haven't done this, it seems weird. If you *have*, then you probably know just how amazingly effective this is. What might take fifteen repetitions when you're trying to read something and burn it in, might take just *one* little act. So, form the "Dorm Three Interpretive Dance Troupe", and start handing out scripts.

* No study partners? Teach your dog, or explain it to a rubber duck.

* Make pictures! Draw mind-maps. You can't possibly buy too many of those flip-chart-sized post-it notes, with some colorful Sharpie markers. If an illustration that the *author* creates is worth a thousand words, the picture that *you* draw is worth 10,000.

* For rote memorization, create your own mnemonics and flash cards you can carry around. (It's always best if you can use the "the more you understand, the less you have to memorize" approach, but there are always a few things you simply must burn in.)

* Use visualization.

* Use chunking and patterns -- (more on that in another post) to group the content into meaningful arrangements, so that you don't have to learn as many individual arbitrary bits, and can focus on bigger chunks.

* Involve more senses. Record your notes and listen to them, while walking around. There's even some evidence that having a strong smell, like freshly-popped popcorn or fresh-baked chocolate chip cookies can help you get the material in. (Or at least it's more fun.)

* Certain kinds of music *might* help, although this is a little controversial, there's some interesting research. (More in this book, including a music-for-learning CD.)

* And make sure you drink enough water. The brain works a lot better with fluid (and I don't mean *beer* up there).

* Make the hard thing you're studying the *last* hard thing you read before going to sleep (or before doing some long, brainless activity like a hike). A big part of your learning and memory encoding happens *after* you put the book down (or stop

listening). If you put something else into your brain before the other stuff has a chance to "gel", you'll weaken or completely inhibit that process. But you *can* mix things that use completely different parts of your brain. So you could learn Java, and then go work on your golf swing, without losing too much of the Java you were working on.

* If you're studying for an exam, and you wear the same shirt each time you study, there's some evidence that suggests you might have better recall if you wear that shirt into the exam room. Bummer about the smell, though... And after you pass the exam, you can have a sacrificial burning of the shirt along with your text book.

* The same principle that makes the shirt thing work can work against you if you always study in the exact same place, and then take the exam in a *different* location. So make sure that while you're wearing your "special shirt", you do your studying in different rooms, desks, cafes, etc.

* If you can find a way to link what you're studying to sex, go for it. Your brain won't forget, and your study partner may thank you. (Or, alternatively, slap you. Your brain won't forget *that* either.)

The most important thing is just to remember that your brain isn't *trying* to make it hard for you. Your brain is trying to save your life. You have to find a way to make your brain think it's helping you, by tricking it into seeing your stats homework as crucial to your survival. :)

http://headrush.typepad.com/creating_passionate_users/2005/02/dealing_with_a_.html

Creating Pissed Off Users

By Kathy Sierra on March 2, 2005

The second edition of Head First Java is finally out, and now shipping on Amazon (and apparently in stores, but I haven't seen it yet). But Amazon is spectacularly screwed up right now, most noticeably in how it handles new editions of a book.

So the big problem is--everyone is buying the *old* edition on Amazon, when they should be buying the *new* one! The trouble is, it's nearly impossible to *find* the newest edition on Amazon. It used to be that when you went to a book's catalog page, and there was a newer edition of that book, you'd see that. There would be a link to all editions.

Not right now, though. If you go to our first edition, there's NO way to know that it's the *older* edition. Then it gets worse. Imagine you do a search in Amazon on, say, "Head First Java". The exact title. The new one *still* doesn't appear! Although you'll get our old book, other books we did, and then a nice selection of other books that somewhere mention the words "head", "first", and "java" somewhere in their text. Which means that

you're actually *more* likely to find a book about having oral sex in a coffee shop than you are of actually finding a book that's *named* "Head First Java."

But wait... there's more! Let's say you find the old edition (the one that comes up on searches, etc.) but you already *know* there's a newer one. So you do what I'd do... click on the author's name to see the list of other titles by that author. But no, you'd be wrong there too.

Apparently all the publishers are mad as hell at Amazon, and we keep hearing, "They're working on it." One author put it bluntly, "Amazon is lying to their customers, by not telling people they're buying the older version of the books."

This is of course *not* a recipe for passionate users. I reckon a lot of people will get the old book only to realize at some point that there was a new one already shipping at the time they bought the old one!

So... be careful please, and if any of you have accidentally purchased the wrong Head First Java book on Amazon, we are so very sorry. Send it back to Amazon and demand they give you the new one. Trust me on this--you don't need both versions ;)

And for those of you not wanting to get burned by this on *any* book you purchase, right now it looks like the best thing to do is a normal search on the book you want, by title, and THEN do a sort by "publicatioin date". So far, that seems to be the best (and often *only*) way to bring up the newest editions. But Amazon has a history of changing their algorithms on a daily basis, so who knows what'll show up tomorrow. But this "don't show new editions correctly" thing has been going on for many weeks, and it's likely causing quite a lot of headaches for Amazon's customers.

http://headrush.typepad.com/creating_passionate_users/2005/03/creating_pissed.html

Creating a passionate...you

By Kathy Sierra on March 2, 2005

If you want to turn someone on, ask them about something they're passionate about and watch what happens. This is also a great recipe for cheering someone up. ***People love talking about their passions.*** (Which is one of the reasons that creating passionate users means you don't have to rely on traditional marketing... passionate people *talk*.)

But what about *you*? When someone asks you what you're passionate about, does it have anything to do with your work? Did it *ever*? If you used to be passionate about what you do, but now have trouble maintaining it, that's a problem. You can't expect to *inspire* passion if you're not feeling it yourself.

But it's tough to do. I remember sitting around at Virgin once, on a tight deadline, with several other members of the team, and we were all stressed, snapping at one another, just having a typical Bad Work Day. Finally one of the young QA guys who'd been testing a game in the corner stood up, looked at all of us, and said (with tons of attitude), "Are you HEARING yourselves? You guys are whining about making *games*?"

Okay, that shut us up. We were all doing exactly what we *loved* doing, but we had somehow stopped being mindful and slid into Work Attitude. From politics to policy, we were cranky about everything. Yet the artists were *doing art*. I was *programming* (which I love). Even the producer was doing what he loved--managing a large creative team and bringing a commercial product to market. But still, we were whining. "The deadline is too short." "The manager is an ass." "Marketing doesn't care about us." "The only time Richard Branson came to party with *us*, he was too drunk to notice." "We're the forgotten stepchild at Virgin... Virgin *Games* gets all the glory."

Now, that doesn't mean we should have just been *thrilled* with everything that was happening. But when we all took a step back, most of us knew that there was a time when we all would have killed for this job, and the chance to work at something we genuinely enjoyed, and most importantly--*were genuinely good at*. So why is it so hard to remember that sometimes? And what can you do about it?

I talked earlier about this notion of keeping your work fresh and inspired, as opposed to reaching the phone-it-in stage.

If you're working in a field you hate, at a job you hate... I don't know what to say except *get out as soon as you can*. But I'm really addressing this to those of us who actually *are* doing what we at least *once* really loved, but are having trouble keeping that early magic. I'd love to hear what other people do, but here I'll give you my own personal approach:

1) Find a way to be around others who are passionate about the work you do.

Passion is infectious. Even if it's just an online user group (although there are reasons why face-to-face is more effective).

Actually, just being around people who are passionate about *anything* is still good for you. (And conversely, stay away from

the people you don't want to turn into, or those who judge and criticize you.)

2) Attend conferences.

This automatically takes care of tip #1, but it goes beyond that. I try to go to at least three conferences a year, sometimes double that. I can usually live on the motivation I come home with for months, not to mention that virtually all of my better ideas have come from conferences and trade shows.

My two top favorites:

Game Developers Conference (bummer -- it's next week and I can't go this year because I'm preparing for eTech). Honestly, this conference is good for ANY developer, regardless of whether you are or ever plan to create games. Just being around that much enthsiasm and brain power... you'll be energized *and* full of ideas.

Siggraph I can't even explain this one if you haven't been there. Anyone remotely interested in art or technology will walk away changed. You don't even have to attend the actual conference sessions. Just pay for the exposition pass, and you'll be amazed. If you're looking for The Next Big Thing, there's a good chance it's lurking around Siggraph a few years ahead of time. And it's incredibly fun.

I encourage you to attend conferences both in--and out--of your field. Almost every year I try to attend at least one show that has little to do with what I'm working on... just to see if there are lessons learned in their domain that I can apply. I find that anything related to entertainment, advertising/marketing, or training can apply to virtually anything.

3) Ask yourself, "What did I used to really love about this?" Remind yourself why you wanted to do this!

It doesn't mean you don't change your mind, or outgrow it, or evolve, or whatever... but you won't know whether it's time to move on or whether you just lost your perspective unless you truly answer that question. Ask yourself, "assuming I do NOT win the lottery, what else would I rather be doing for work right now?"

You don't have to be passionate about the company you work for in order to be passionate about what you do. You don't have to love the company in order to love users.

4) Learn something new.

It doesn't even matter what it is, although if it's something new and cool related to what you do, that can help reinfuse your work. Remember, *learning turns the brain on*. If you ask people what they're passionate about, you'll almost *always* discover that this thing involves ongoing learning and improving.

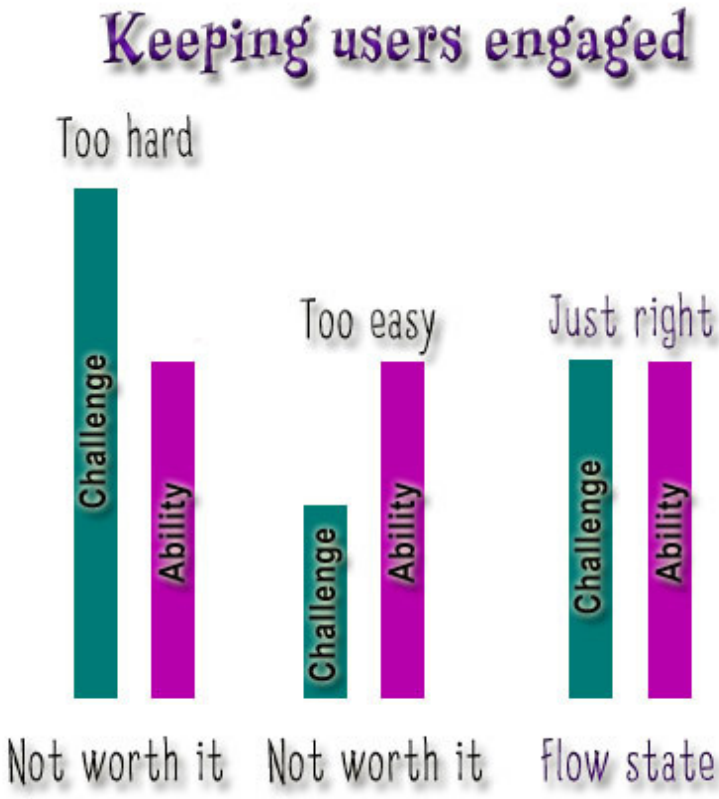
Of course if you *hate* the company you work for, and/or you *hate* what you're doing, you're not likely to create passionate users. In that case, I hope you're simply on short-timer mode, and you're reading this blog because you're planning for the time when you can do something you really *can* love :)

Most importantly, don't let anyone stand in your way. Passionate people are often threatening (although I have no idea why) to those who aren't so happy, and the threatened types can really f*** things up for you. It took me a long time to learn that lesson, especially because the threatened ones can often be the "wolf in sheep's clothing", offering you advice "out of concern" or trying to "not get your hopes up." I try to be realistic, and know there are no guarantees, but for gods sakes, I intend to keep my hopes *up*. I'm certainly going to do better that way than if I have my hopes *down*, and it makes the journey a lot more exciting.

http://headrush.typepad.com/creating_passionate_users/2005/03/creating_a_pass.html

Can you have too much ease-of-use?

By Kathy Sierra on March 7, 2005



We all talk about user-friendliness and usability, but is it possible to go *too* far? The answer really depends on the context, but yes, it *is* possible to make something so easy that it loses value. And the things people are passionate about *always* involve some level of continuous challenge. ***Something users can keep getting better at. Opportunities for growth.***

Think about it... skiing, dancing, chess, photography, flying, dressage, gardening, dog training, environmental activism, religion... when people are into any of those things *passionately* (as opposed to casually supportive), they keep wanting to get better! People who are passionate always have an opportunity (which they grab) to keep improving. To keep learning more. To improve their skills and knowledge about whatever it is they love so much. They read and they practice.

So if what you offer doesn't have any challenges associated with it, and things for which people can continually learn and improve, you'll have a harder time getting people passionate about it. Now, this doesn't mean you should make your *user interface* challenging. If you're writing software, it's usually because the user is going to use your software *to do something else*. And if that thing they do *using* your software is challenging, then you want your software to get the hell out of the way and let the user get on with what they *really* love--correcting the colors of old photos, creating three-dimensional images, writing the next great novel, finding real information in the noise of a signal they're analyzing, whatever.

And in that case, you want your software to be as easy as possible, and let the challenge lie in the thing they're passionate about. And anything you can do to make that activity a better experience for the user is one step toward helping them be passionate. Because the more time they spend in a state of flow, where they're completely focused on a challenging activity for which they have the right level of knowledge and skills (and without having to think about the interface they're using to *do* it), the more likely they are to stay engaged.

But if you're trying to create an environment in which people can be passionate, *something* (just not the interface) needs to be challenging, and there must be a way for users to build and grow their knowledge and skills in a way that keeps pace with the increasing challenge.

If the thing you want users to become passionate about is simply too easy, without enough opportunity for continuing challenge and growth, they'll get bored. It's not worth it. And if the thing you want them to be passionate about is too hard, they'll get frustrated. It's not worth it. This is tricky, because you have to find ways to balance that challenge level, while also providing opportunities for your users to *keep getting better*.

The key to inspiring passion is to have something worth learning, and a way for that learning to happen.

If you look at things that people are passionate about, there is always some way to tell that people have really become experts. They ski double-black diamonds. They have a black-belt. They are a grand master. They grow rare orchids. *They speak conversational Klingon*. So one of the ways to help people become more passionate is to figure out what it looks like when

people are *better* at that thing, and help find ways to make that happen for people. A ski resort with nothing but bunny slopes won't last long, even though everyone will have a wonderful happy and easy first three days, before they get bored and realize skiing isn't very fun. If there weren't those blue slopes beckoning (and all your friends already up there), there'd be little value in going back. And after blue, there's black, then back country, and...

Where there's *passion*, there's usually a *user kicking ass*.

Help give your users an "I kick ass" experience, and you'll greatly increase the chances that they'll become passionate.

[Update: you can get an interesting twist on this that we'll be talking more about in the future, in Dave Roger's UXCentric blog post on doing the Leonardo.]

http://headrush.typepad.com/creating_passionate_users/2005/03/can_you_have_to_1.html

What's in your wake?

By Kathy Sierra on March 8, 2005

Does your product or service support plug-ins and add-ons? Does it lend itself to follow-on products, accessories, support and training, etc.? *Does it inspire others to be part of your wake?*

Whoever competes with the iPod has to compete with a lot more than Apple's device--it has to compete with this great wall of stuff riding in the iPod wake. And these things all make the iPod a lot more appealing and flexible.

Inspiring a wake--where passionate people add value to *your* product or service with new things--is one of the fabulous side-effects of having passionate users. And your chances of creating passionate users just keeps going up the larger the wake gets. So it's a great big happy reinforcing feedback loop.

Some marketing folks have talked about user-created ads, but if you let users enhance what you *offer*, by adding more features or even just by *creating cool fan t-shirts*, you're much further up the passionate users curve.

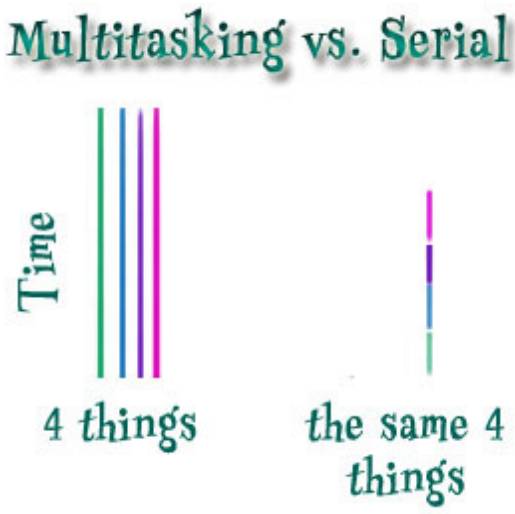
Are there ways in which you can encourage others to add value to your product? If it's software, do you have an API that supports plug-ins? Do you encourage others (even if it means no direct revenue for you) to provide training and support? Are people likely to write *books* about it? (More books on the shelf about your product=more visibility for your product, and more chances that someone will have a successful experience with it.) How many new businesses were started by users who liked something so much, the decided to start their *own* business around it. So what are you doing to help others build in your wake? Being closed, or trying to keep others from capitalizing on what you provide (in other words, trying to keep the wake for yourself), is a bad idea.

The more interesting and valuable your wake is, the more likely it is that you'll create more passionate users. And the more passionate users you have, the more likely it is that your wake will grow.

http://headrush.typepad.com/creating_passionate_users/2005/03/whats_in_your_w.html

Your brain on multitasking

By Kathy Sierra on March 9, 2005



If you're a programmer, you know that context-switching in a multi-threaded system isn't 100% free. There's overhead with tiny bits of time lost on each switch, as a new thread takes control. Well, it's the same way with your brain. Only a lot *slower*. And it doesn't look like

Brain 2.0, Now... with Multi-Processor Capability!

will be coming anytime soon.

And although there have been plenty of studies to show otherwise, the belief that multitasking will let us get more done continues. Think of how many times you've been on the phone with someone when you hear that little click-clack of their keyboard. (I hate that. I do it to other people, but I *hate* it when they do it to me.) And it makes me crazy when I'm trying to have a conversation with someone in the same room, while they're saying, "Uh-huh... yeah... I'm listening...sure, I can do this and talk at the same time...". *You know who you are ;)*

Our brains *can't* do even *two* independent things that require conscious thought, *especially* if those two things involve different goals. But that's OK, you might think, since multi-threaded systems on a single-processor aren't *technically* doing two things at the same time.. they're simply switching back and

forth so quickly that they just *appear* to be processing simultaneously. But that's the problem... the brain *isn't* a computer, and in many cases the brain works much more *slowly* than a modern processor.

With each context switch, say, from the phone conversation to the email, there's a hit. And it's not a subtle hit. One of the things I really like about stress-management expert Jon Kabat-Zinn is that he sometimes offers seminars and workshops on time-management, but when you get there, it turns out his approach isn't about how you manage your file folders, but about *mindfulness*. **Practicing mindfulness is like adding more hours to your day.** If you're mindful, time slows down. You get more done, enjoy things more, and feel less stress. These are big claims, but anyone who's practiced mindful meditation or, like me, mindfulness-hold-the-meditation-thanks, will swear it's true.

So if you're stressed for time, do everything you can to *resist the seemingly-intuitive notion that doing several things at once will save time*. I know how hard it is to let that go, but study after study proves this wrong (here's another article from CIO magazine). Obviously there are exceptions, especially if you're quite content to let the quality of the work go down, or to be *rude* to the person you're talking to.

But imagine what it would be like if every time your co-worker, friend, spouse, lover, child wanted to say something to you and you turned and gave that person *all* your attention. End of story. No television sucking you into the event horizon. No glancing at the computer. No talking on the phone or checking your watch or reading a report... just 100% mindful, totally there, perfect eye contact, YOU. If you already do this now, that's awesome. If not, then if you try it--and I mean *really* try it--your family might think something's wrong with you. (One of those, "Who are you and what have you done with my husband?" moments.)

One tip: the brain finds it almost impossible to *not* turn to look at a television that's on (more on that in another post). *So turn it off*. If you *must* have television, make it a destination event. Something you do consciously like choosing to go to the theater. One of the worst things you can do to your brain (and family) is just *have the TV on* when you're doing virtually anything else *but* sitting down to watch a specific show. In other words, have a damn good reason for turning it on, and I swear you'll get more

done (and have more energy... remember, television acts as somewhat of a temporary sedative to your brain. It literally sucks your energy, while simultaneously making you *feel* like it's helping you to relax. There's a great issue of Scientific American special edition on the Mind (volumne 14, number 1) that goes into a lot of technical detail about this).

If you want to get more done, *be mindful.*

If you want to have more time, *be mindful.*

Mindful means *one thing at a time.*

It's how the brain works, no matter how you try to convince yourself you can do it (although there *is* evidence that fast media/video-gamer kids *are* a little faster at switching. Not because they have a *younger* brain, but because their brains were more wired for this pace at a younger age).

As the Buddha might have said, when you're answering email, don't try to talk to someone at the same time. Be the emailing. ;)

http://headrush.typepad.com/creating_passionate_users/2005/03/your_brain_on_m.html

Motivated to learn?

By Kathy Sierra on March 23, 2005

Think of a time when you *wanted* to learn something because there was something you *needed* to do. It could be as simple as figuring out how to transfer a call on your new (and insanely complex) phone system at work, because your boss' wife somehow ended up at *your* extension and you SO don't want to hang up on her. Or it could be that you just realized that you're really tired of copy-and-pasting your contact info onto every one of your web pages, so you need to figure out how to dynamically include a snippet of HTML in every one of your JSP pages.

Now think back to most of what you learned in high school. How much biology do you remember? I mean, *really* remember? (Assuming you aren't a medical student or biologist today.) I am

100% certain that I'd fail some of the exams I took when I was 16... including some of the ones I aced at the time.

OK, so that was a pretty long time ago, and no matter how well you learned something, there's a little bit of a use-it-or-lose-it for a lot of topics. You might still have it all in your brain, but the mechanism for recalling it is too rusty to be useful.

But think about something more recent. Think about the last technical topic you learned from either a class or a book. *How much of the details do you remember?* The answer probably depends a *lot* on whether you knew that you *needed* to be able to do that particular thing you were learning. And that's huge. Because if even at a high level you know you *need* to learn PHP, if the parts your studying don't seem directly related to what you know you want to do, the learning will be weak.

And that's the problem with a huge chunk of learning today, from schools to colleges to corporate/IT training to books:

Just-in-*case* learning sucks compared to just-in-*time* learning.

That doesn't mean there aren't a lot of problems with just-in-time learning, too... usually just-in-time learning is also just-what-you-need to survive the current problem, and you might not even understand *why* the thing you're doing works. But there's a hybrid solution that we try (not always successfully) to do sometimes in our books or in the classroom, and it's this:

Give a compelling, personally motivating reason/benefit for the thing you're teaching, *before* you teach it!

In other words, try to make just-in-case learning *feel* more like just-in-time learning. In our Head First books, for example, you'll see a lot of things like, "Imagine you've just finished working on this project when suddenly the spec changes, and your boss says..." We try to give scenarios up-front, that at least provide a *tiny* bit of just-in-time motivation. That feeling of, "OK, I really need to be able to do this, so I need to figure out how..." vs. "I'm sure this is relevant or it probably wouldn't be in the book, but it's not something my brain needs to pay attention to right *now*..."

We try to get our authors and teachers to really work on this, but it's not always. I've had learners in a Java class who had no idea

if they would *ever* actually use Java in the real world. So I try to help them *imagine* what they might want to do, and I try to come up with things that might be inherently motivating, to make it more like a game. Almost *anything* can be made interesting and even compelling if the book/teacher doesn't suck the life and joy out of it by making it boring, academic, or too comprehensive and difficult (like when the book tries to be both a learning *and* reference book, so it covers absolutely everything about any given topic, including the stuff that even the author can't imagine actually using in the real world...)

I think I gave a few tips on doing this in a much earlier post on Show-dont-tell applied to learning.

A good goal: figure out ways to make just-in-*case* learning feel almost as motivating as just-in-*time* learning.

http://headrush.typepad.com/creating_passionate_users/2005/03/motivated_to_le.html

How do you thank your loyal users?

By Kathy Sierra on March 25, 2005

Wow -- I may have just had the best "customer love" experience of my life. Over the last five years, I've been spending way too much money buying prescription glasses from one of those foofy overpriced mall "eyewear" boutiques (The Eye Gallery in the Flatirons mall outside Boulder CO). I keep going there because the optometrist is amazing, helpful, nice to be around, treats you like a friend, and most importantly--spends a *lot* of time educating and motivating you about what's really going on with your eyes, how to take better care of them, what it means to your eyes to be living at such a high altitude, etc. And best of all (for me), the folks there take the time to help me find something at least half-way flattering (or at least they do a great job of making me *believe* that... by the time I leave they have me thinking I look like Heidi Klum. Of course that wears off completely once I'm in, say, the dressing room at Nordstroms).

So I just lost my last pair of glasses and went in all desperate, 20 minutes before closing. My normal doctor was out on maternity leave, but her new husband, who'd never seen me before was there and he decided to do the exam right then, after closing time. Then he and one of his assistants spent 45 minutes helping me while I agonized between the two "designer" frames I'd narrowed it down to. On one hand was the very fun, very french, very *expensive* pair of purple frames that I dearly wanted... and on the other were the tortoise shell ones that were still cool, but way more practical. I wimped out and went with the tortoise shell.

Now the good part...

I came back in the next day when they were ready, and the optometrist's husband pulled out the tortoise shell glasses with my new lenses, and did all the adjustments. Then just before I got up to leave, he said, "Oh, I talked to my wife last night about you, and you've been such a great customer that we decided you might want to have some fun... so we went ahead and made those purple ones for you as well. *They're on us*."

I was stunned. Those very festive, very french suckers cost over $300, and here they were saying, "Here, go have some fun!" Talk about endearing me for *life*--I'll never buy glasses

anywhere else as long as I still live anywhere in this *state*. And I'm dragging everyone else I know down there too, armed with all the knowledge they've given me about how important it is to have regular exams, the right UV protection, etc.

Yes, I spent a lot of money there over the last few years, but that's *nothing* compared to what I've spent on, say, my computers, stereo equipment, hell--I've spent more on *Amazon* just for books! But I've never had a personalized or even remotely *special* thank-you. Would it really kill most *big* companies to do that?

(I just remembered another fun example--two weeks after my father bought a new Honda, he got a huge shock when someone from the dealership showed up at his doorstop with a basket of fresh-baked cookies as a thank-you and follow-up.)

But the best part of the thank-you I got from the Eye Gallery is that they gave me an "I Rule!" experience. They weren't just creating a *loyal* customer, they were helping ME be more playful. ***They were helping me kick-ass.*** (Assuming you're willing to buy into my delusion that wearing those cool purple frames makes me smarter, more clever, and *definitely* more fun ;)

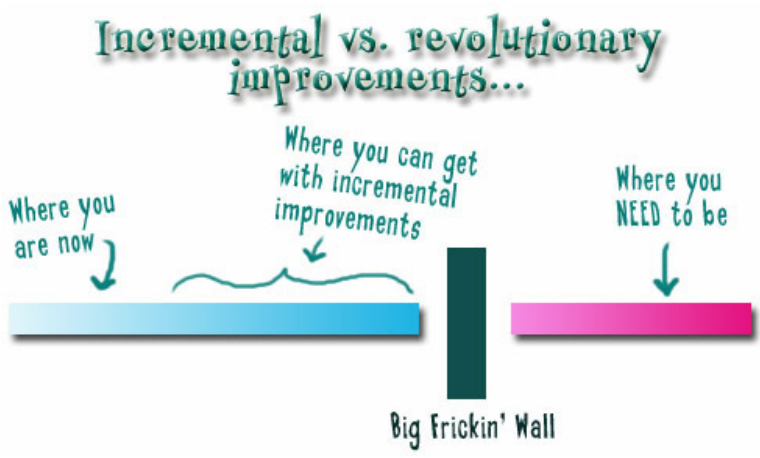
So, how are *you* thanking your users? How are the companies you do business with rewarding or at least acknowledging *you* for your loyalty? Next time you think about how to thank your users, see if there's a way to do something else for them, in the context of showing your appreciation. See if you what you do for them *makes them have more fun*.

They'll love you forever.

http://headrush.typepad.com/creating_passionate_users/2005/03/how_do_you_than.html

Incremental vs. revolutionary improvements

By Kathy Sierra on March 28, 2005



The true art of product or service development might come down to this:

Knowing when it's appropriate to make incremental improvements and knowing when you need a revolutionary leap.

Do you continue to patch, tweak, tune or do you throw away the old assumptions and start with a completely fresh approach?

This is obviously a complex issue, but the metric *we* use is this:

If you're competing for market share, with products or services that are hard to differentiate, *incremental improvements might be a waste of time and resources!*

That's how we looked at the computer technical book market. We set out to write just one book, Head First Java. We said, "There are over 2,000 currently-selling Java books on Amazon. We have very little name recognition, especially among people who don't yet *know* Java (the audience for our book), so what can we do?" And in fact most publishers, book reviewers, etc. were saying, "Does the industry even *need* yet another intro to Java book?!"

We saw that there was a big gap in usability/learnability for a lot of programming books, and knew from our backgrounds in

artificial intelligence, learning theory, and interaction design that there was an opportunity to make a *major* difference, if we could find a publisher willing to let us make the leap. So we proposed the idea to a major publisher (not O'Reilly). The publisher said, "That's way too radical and people won't accept it, but we'll let you do 10-20% of what you want and we can see how it goes..." In other words, they wanted us to make *incremental improvements* to the learning model used in the books we were going to be competing with. We declined, even though we were *really* anxious to have a book published, because we believed that without a *revolutionary* jump in the learning experience, we'd be in for a horrendously bloody fight, kicking and clawing for market share in an overcroweded field against successful competitors who were much better known.

A revolutionary improvement was taking the commonly-held assumptions and tossing out as many as we could if they stood in the way of a good learning experience. The thing is, there *isn't* anything revolutionary about Head First books if you view them in the context of *all* learning experiences, or even just the subset of "books designed for learning." You've all seen books that use visuals, surprise and novelty, strong metaphors, different learning styles applied to the same topic, etc. The difference is simply that you didn't see that *in programming books*.

So "revolutionary" often just means "revolutionary in THIS context." And that's also a way to think about where to *find* ideas for revolutionary improvements... look at what's being done in *other* domains, that might work in *yours*. In our case, we looked at trying to replicate as much as possible the things that make up a good *classroom experience*, and apply that to a *book*. In other words, instead of studying what's good about *books*, we looked at what's good about *classroom experiences*, and *then* we looked at books (largely *children's* books) for more ideas on the ways in which classroom learning could be mapped into a book. We didn't do a great job, either. But the leap was enough to make a significant difference to the majority of the market for those books.

In fact, when we look at it now, we realize that a lot of what we did in the Head First books was still mostly incremental improvements, and that we were *still* basing a lot of what we did on *the way it's usually done*. We chickened out in a lot of areas. (Which is why we designed another new series that you'll see the first books in near the end of this year.) But it's really hard not

to make only incremental improvements, because the natural tendency is to *focus on improving what's already there*. You ask the question, "How can we make this thing better?" Instead of, "What are we really trying to accomplish for our users?"

One of the hardest things to do is *throw away what you've worked so hard on*. That could mean throwing away real *stuff*--physical products or software code--or throwing away *ideas*. There's a certain amount of unlearning that usually has to happen, as well as letting go of things you might be especially proud of. ("Killing your babies" is the expression many writers use.)

The biggest problem with incremental improvements is that they often lead, eventually, to an impenetrable wall that stops you from *ever* ending up where you really need to be. **You can't get there from here.** In the good old days, you could solve that by simply out-advertising/out-marketing the competition. Well, *that's* out. (See Hugh, Seth, and other neo-marketing folks for thoughts on that.)

But today, there's more supply than demand of just about everything, and the competition is fiercer than ever. You certainly don't want to go down that road of competing solely on *price*, but if everyone in the game is trying to make incremental improvements, the liklihood of anyone breaking through in a significant (let alone *lasting* way) is slim.

FYI -- I'm reading a book somewhat related to this, based on the idea of creating Blue Ocean Strategies (the book link is on that site). It's premise is that competing for market share is the "bloody red ocean" and that what you *really* want is the "blue ocean" where the competition is simply irrelevant, because you've created "uncontested market space." I was quite skeptical of the book ("Oh, yeah, it's really simple -- just make the competition irrelevant") , but having gone halfway through the book, I'm starting to be a believer. Their approach really *does* offer a variety of different, concrete, do-able strategies for looking for ways to make this possible.

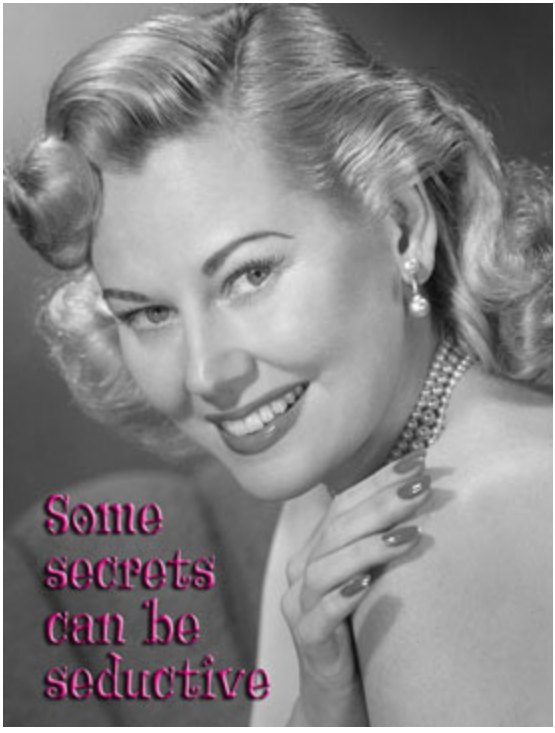
And while we're on the subject of incremental vs. revolutionary improvements, I wouldn't assume that this applies only to products or services (or, say, a legacy school system). ***It applies to your whole life.*** I've known marriages, for example, that were failing with patches, tweaks, and tunes--but who survived and eventually *thrived* by taking a *revolutionary*

step (moving away from the meddling in-laws, quitting the stressful job, choosing a simpler life, *throwing away the television*, etc.) And I've known people who gave up on incremental career improvements, made the revolutionary personal leap and changed their life in a dramatic way. Remember, thanks to what we now (and only very recently) know about the neuroplasticity of the brain, it's never to late to create You 2.0".

http://headrush.typepad.com/creating_passionate_users/2005/03/incremental_vs_.html

The importance of seduction and curiosity

By Kathy Sierra on March 30, 2005



Part of creating passionate users starts with building curiosity. Inspire them to want to learn, know, and do more. A comment from John Mitchell on my motivated to learn blog reminded me about this--he mentioned the importance of being **passionately curious** about the topic (and I couldn't agree more).

So can you inspire curiosity? Can you seduce the user into actively wanting more, even if that user didn't start out with their own intrinsic intellectual curiosity?

Sure. It won't work for everyone and every topic... but think about things that you know have worked for you in the past:

1) Be passionately curious *yourself* (good point John!)

The brain is tuned to mirror the behavior of others, so if your passionate curiosity is stronger than the other person's passive

disinterest, you have a chance to "infect" the other person. It's not just that you *know* what's exciting, wonderful, fascinating about a topic--it's that you genuinely *feel* it, and this is reflected in the way you talk about it, not just the actual content of your words. ***Passion breaks through.***

2) Be seductive

That means knowing when--and what--to hold back. Don't hand them *all* the answers... take them part way and tease and tantalize them into going the rest of the way. *The brain wants to find out what happens next*. It's what keeps you watching the movie until the end, staying up late at night with a page-turner, tuning in next week (especially if last week's episode was a cliff-hanger), and hoping for that second date... NPR refers to the phenomenon of wanting to hear the end a driveway moment--where you're listening to an engaging story (like on This American Life, or a radio diary) but arrive home before it's over. *You can't get out of the car*. You just have to hear how it all turns out.

3) Make them curious by doing something unusual, without an obvious explanation (a variation on #2)

In the Parelli natural horsemanship program, I learned a new way to "catch" a horse. I walk into the big pasture holding the halter and instead of walking straight toward my horse, I kind of meander around not even looking at her. Then when I come close enough for her to know I'm there, I stop and turn around so my back is to her... and I might even start walking away while fiddling with whatever I'm holding. Eventually, she can't stand it and has to know what I'm up to and why I *didn't* try to catch her. So she'll come over and "catch" me. (For all you pet people, I'll mention that we are not allowed to use treats as an enticement. They're coming because they're curious and it triggers their play--rather than fight or flight--instinct). In the book/movie "The Horse Whisperer", Robert Redford's character spends hours sitting in an open meadow until the terrified, escaping horse finally walks up to *him*. Curiosity can beat fear.

4) Offer a puzzle or interesting question... without giving them the solution.

It's almost impossible to turn away from a TV game show when a question has been asked but not yet answered. But it works for

almost anything that engages the brain's strong desire to find solutions.

Here's an example (many of you will know the answer to this already, so it won't work for you... but it works quite well on most people who don't know this problem):

==========

Kevin, a college student, walks into the cafe where he spots Reese, the gorgeous math whiz he's seen around campus. He works up the courage to walk over to her, "Hey Reese, I'm Kevin, and I heard you're the only one to ever get an A in Bozeman's Stats class... I'll buy you dinner if you help me study for the exam."

Reese look up skeptically then says, "I need to know if you're worth helping. Tell you what, I'll write my phone number on the back of one of these three business cards. I'll mark them on the front A, B, and C, but you won't know which card has my number on the back. If you pick the right card, you can call me and we'll schedule a study/dinner date."

Kevin's not happy, "But what does *that* prove about me? You're not even giving me a 50/50 chance... but OK, if that's the best I can do... I'll pick card 'B'".

Reese is left with cards 'A' and 'C', and says, "Before we look at your card, I'll give you another chance. I've just turned over card 'A', so you can see it doesn't have my number. That means my number is either on the card you picked, 'B', or the card I haven't turned over, 'C'. Do you want to switch your card 'B' for my card 'C'?"

Kevin cocks his head and thinks to himself, *Ah... she's trying to see if I recognize that the odds are the same for both cards (duh), since they both began with a 1 in 3 chance. She probably wants to see if I'm decisive and confident...* He looks at Reese and says, "No thanks; I'll stick with my original choice 'B'. It's just as likely to be the winner as your card 'C'."

Reese flips his card 'B' over and shows that it's blank. Her phone number was on card 'C'. Kevin laughs and says, "Well, you didn't give me a fighting chance. All the cards had a 1 in 3 chance of being the right one, and at least I didn't fall for your little swapping trick... you should still give me your number."

Reese rolls her eyes, shakes her head slowly, and says with a frown, "You know Kevin, if you would have agreed to swap cards when I gave you the chance, I would have given you my number even if it wasn't the card with my number. Switching cards would have shifted the odds in your favor, and I was really hoping you knew that. Sorry, but I don't want to waste my time trying to help you."

Unhappy and a little angry with Reese, Kevin leaves the cafe and tells the story to his roommate Manny. Kevin says, "Reese is just wrong... there's no way that switching my card for hers at the point would have made any difference. Both cards started with a 1 in 3 chance, and nothing changed that."

Manny looks at Kevin and says, "Dude... Reese is right. Switching cards would have changed your odds from 1 in 3 to 2 in 3. You blew it."

So... are Reese and Manny right?

Yes, they are. Swapping would have changed his odds of having the winning card.

Your job is to figure out why it works that way... and if you want to learn it here, you'll just have to wait for another blog entry (later this week, I promise) to find out.

[Note to those who recognize what the Kevin/Reese puzzle is about: don't reveal the true "name" of this problem, so we can make googling for the answer a little less easy for everyone else ;)]

==========

People who don't immediately understand the problem and don't believe it, will often set out trying to disprove it (they're curious to find proof that Reese, Manny, and *you* are wrong), or they believe it but they're so puzzled by it that they try to find out what's going on (curious to understand).

The brain wants it to make sense. :)

There are obviously lots of ways to get people curious, but it's been a highly underrated strategy for getting people engaged, hooked, motivated... all prereqs for *passion*. Think about ways you might use curiosity as a technique in everything from user documentation and tech writing (including books) to teaching to

dating to product development to marketing to horse training to parenting and even *delighting your significant other*.

Having a passionate curiosity is a true gift, and anything you can do to help give a little of that to another person is enhancing their life. If you want to truly delight someone, then seduce them (not the same as coerce or manipulate, if we make ethical distinctions--I realize some people don't like the word "seduce", but *we* love it) into being curious to learn more, grow more, stretch their mind, become more skilled, or just find out what it's like to be *better* at something.

http://headrush.typepad.com/creating_passionate_users/2005/03/the_importance_.html

The new geek speak / neo-marketing language

By Kathy Sierra on March 31, 2005

We mock the corporate b.s. speak, but have we listened to *ourselves* lately?

This latest Hugh cartoon I'm in love with reminded me how much the techies/geeks/neo-marketing folks (of which I'm a member) are doing just fine with our *own* brand (would that be a *hijacked* brand?) of buzzwords.

(And don't even get me started on the ones used in software architecture. I'll save those as a special subset.)

Not only are you supposed to know and *use* these terms, you're also not quite clued-in (or is it Hughed-in) if you don't also buy into their true meaning. That is, if you can figure out what that really is :)

There's no pot-calling-kettle-black thing here... I'm just as guilty (although I challenge you to scrutinize the archives for a single instance of my using the word "blogosphere"). I use "Hugh", and "Seth" with full assumption that their last names would be

redundant. I make jokes about "transparency" assuming you've heard the Cluetrain arguments. And I do *that* assuming you know what Cluetrain refers to.

I even use Scoble in my blog banner! (For Robert Scoble, whose blog I adore, despite his Microsoftness.) Here are a just a couple of recent Scoble quotes, "No RSS? Lame. That tells us you don't want connectors/sneezers/influentials to talk about you..." and "Be sensitive to the leading "connectors" -- they'll be the ones who'll really kick off your viral campaign." Of course none of those *words* are very new but what *is* new is for so many *geeks to be talking like marketers*.

Fortunately, there's hope. Like any problem, *acknowledging it* is the first step, and apparently there's even a drinking game around these words (much like the old business buzzword bingo, except more festive... *with alcohol*.

But... (and you knew there'd be a but) there's something really interesting in all this. The goal should be honesty, true. And all the new *emerging* technology and ideas, we *do* need new words. If a word or phrase describes something new, then it's not necessarily a b.s. buzzword used simply to obfuscate or to make ourselves sound like we have a clue. So, it might be completely appropriate to use these new words so casually, if they represent what we're trying to communicate.

A *bigger* question might be, should we use these words without defining them? Should we assume that our readers already know what (or who) we're talking about? Is this exclusionary or clique-ish? Yes, yes, and yes... if we're talking about passion.

For one thing, most of us using these words in a blog or other online doc have links. If someone doesn't know who Hugh is, they can click to find out. It stops me from interrupting regular readers with repeats and redefinitions, and Hugh's site does a far better job of trying to explain him (or not ;)) then I ever could. And thanks to Google, we can all get a definition along with the most recent conversations about just about any word I could possibly use.

But that's still not the most important reason to use some of these words and names without referencing them...

When people are passionate (or even just "into") something, they have a shared lexicon that helps dinstinuish them from those who aren't.

And this is not a bad thing. Professionals and hobbyists have had shared, specialized vocabularies for years. Among other things, it helps them get a message across more quickly than if they couldn't use those things. But it also helps build their devotion to their passion. Just figuring out the commonly-used phrases, words, names, stories, etc. are *part* of what gives people a sense of belonging. A sense of being a part of something special. A sense of having learned, and *earned* their way in. So in this case, exclusionary isn't necessarily a bad thing.

Becoming a part of something new usually *isn't* that simple. You have things to learn. Show me an area where people are passionate, and I'll show you how there is virtually *always* a learning curve that includes ideas, concepts, terminology that are specialized. Most people have an "I Rule" experience in part *because* they've "crossed the chasm" (reached the tipping point?) and learned what others are talking about. Of course between Google and wikipedia, it's almost too easy these days ;)

Obviously if you use way too much jargon, and the answers are *not* readily found, you *will* restrict your "tribe" (there's another one). *But that's not always a bad thing either!* You may decide that raising the barrier to entry adds value to those in the group who've taken the time and effort to come up the curve. You may decide that you can't even *be* true to who you are (you know, "your authentic voice") if you have to make the message clear and understandable to everyone, newcomers included. **Some passions are worth the trouble, and indeed *better* for having a certain amount of effort.**

Besides, something that gets you to go off and do a little research on your own is often much more powerful than if you're handed everything without having to think about it. So... what special words, concepts, stories, people are a part of what *you* are passionate about? Or a part of what you *want* people to be passionate about? Lowering the barrier to entry, especially when it comes to conversations, isn't always the best path when you want genuine passion.

(That said, if you EVER and I mean EVER catch me sounding anything like the couple in my cartoon here, slap your mouse around a few times to slap me out of it.)

http://headrush.typepad.com/creating_passionate_users/2005/03/the_new_geek_sp.html

You and your users: casual dating or marriage?

By Kathy Sierra on April 8, 2005

I was unexpectedly gone for a few days because I fell in love... *with my new skis*. What was supposed to be a late-season one-day trip became three of the best skiing days of my life. And all because I was on skis that made me feel like I kicked serious alpine ass. ;) No, this is more than love... this is *passion*.

I've always been in love with skiing, and it's been close to a passion. But despite my love for what you *do* with skis I was never seriously into *skis* (or any other equipment as long as it worked). If you'd asked me a week ago, I would have said that I "loved" K2 skis. My first pair of skis as a teenager were K2s, and every pair I've had since then have been K2s. If the company had tracked me, they'd have said I was a loyal, perhaps even passionate user. And I thought so too. But it turns out, I was merely a little *sentimental* about the company.

When the time came to make a new buying decision, I discovered just how *un*passionate I really was for K2. When I found an expert to help me (a ski guru at Boulder Ski Deals), we narrowed it down to two skis: a new K2 women-specific ski (very cool idea) called "One Luv", and a women-specific ski from Volkl, the 724 EXS Gamma.

But here's the thing -- although I favored the K2 because I had *thought* I was a fan of the company, when it turned out they didn't have them in my size and I'd have to wait a few days for them to come in, *I switched*. **My so-called love and loyalty evaporated.**

And I think a lot of businesses probably mistake customer retention (repeat buyers) for customer *love*, when it might be nothing more than the fact that humans tend to be habit-driven, especially in the face of so many choices. I tended to buy K2s because I knew it was a good company, and they'd always worked for me in the past. Although until this week, I probably never uttered the words, "I love my skis!" So when presented with a choice between something new I could have right then (and even at a slightly higher price) or waiting a couple days for my "love" brand, I dumped K2 without the tiniest flicker of emotion. ***I wasn't a passionate K2 user, and it turns out I wasn't really even in love.***

But now, with Volkl, it's a different story. Because of the combination of an awesomely-engineered pair of skis, and the expert thoughtfullness with which the sales guy at the store made his recommendations, I ended up with a pair of skis that took my skiing into an entirely new plane. After the last three days, I am not just passionate about *skiing*, I'm telling *everyone* I know to pay attention to Volkl, *especially women*. My passion for the *sport* of skiing has now been permanently bonded to a particular ski *vendor*. (And the *store where I got them* as well.)

Let's say these skis are stolen next month or next year. After I back away from the ledge, I'll do whatever it takes to get another pair of *these* skis. And if that model is discontinued, I'll buy a *different* pair of Volkl skis. I'm absolutely certain of this:

Ill wait as long as it takes to get them, and I'll pay a premium.

Remember, it doesn't matter how your users feel about YOU, all that matters is how they feel about themselves as a result of interacting with your product or service.

By making *me* kick ass, Volkl now has my undying loyalty and passion. Something K2 never managed to do. And yet, even if I'd had only good (but not fabulous) experiences on my K2s, the company *could* have inspired my loyalty and passion by giving me something more to believe in. But they didn't. At least not in advance. It turns out that K2 women donates a portion of their

sales to the Breast Cancer Research Fund, a cause dear to my heart (my mother died from breast cancer at the devastatingly young age of 40). But at the time of my purchase, I didn't know that. It might have tipped the balance, actually.

Could K2 have done something to turn my perceived-but-not-real loyalty into a long-term commitment? Probably, although I'm not sure how. Maybe they needed an aggressive ski registration system, so they could have known--and rewarded me--for being a K2 owner.

Hmmm... this gives me a lot to think about, but most important is the need to stop confusing loyalty with love, and love with true passion. Great customer service and a great product can earn you satisfaction, and often love, but until we get something close to *passion*, an attractive outsider can still turn a user's head. And the way to move toward passion, is to give your users the kind of experience I had this week... where I thought I was simply the hottest thing on Copper Mountain (I wasn't, of course, but that's not the point ;)

So, what are you doing to give your users the "I kick ass" experience? And what are you doing to help lock in your relationships with *regular*, but not yet *passionate* users?

http://headrush.typepad.com/creating_passionate_users/2005/04/you_and_your_us.html

Context matters

By Kathy Sierra on May 1, 2005

The Sun project that took us to Japan was the development of a new Java certification exam. It's meant to be a beginning level exam for entry-level employees or new graduates who haven't yet worked as Java programmers to at least demonstrate a basic level of knowledge. For more than a month we (the American team) argued with our Japanese counterparts over the objectives of the exam.

We (the Americans) figured it would be a scaled-down, easier version of the current programmer exam, with an emphasis on the fundamentals of the Java language. Simple.

They (the Japanese), on the other hand, felt that some of our objectives were too technically detailed, but then they included *all this other stuff* they wanted to test people on. Things like understanding the difference between the three Java "editions" (micro, standard, and enterprise), how each of these editions make sense given a design goal, problems/tradeoffs with deployment of these various editions, basic UML, and on and on...

In other words, **they wanted to test not just on the Java *language*, but also on the *context* in which Java is used.**

And there was no talking them out of it. Although at first we (Americans) complained, we finally had to agree that these

objectives make sense for *this* exam, where you want to know that the person holding the certification understands the Big Picture. And that seemed more valuable, for an entry-level person, than simply proving that they had memorized a basic level of facts about the language.

But then I remembered a book I read a few years back, The Geography of Thought, and it all started to click in for me. The book is amazing, and offers a ton of fascinating research and studies that prove that we DO think differently. My brain processes the world in ways different from that of my Japanese counterparts, and one of the those ways involves context.

To greatly oversimplify:

Context plays a more fundamental role for Asians than for westerners. Asians have a more difficult time thinking of an object as completely separate from its background.

Americans, on the other hand, focus on objects... *things* and categories more than relationships.

Asians think in *verbs* where we think in *nouns*. And these differences can have profound implications.

Continuing on from my post yesterday about Dan Pink's book, A Whole New Mind, a more holistic point of view is a perspective we're all going to need more of going forward. *Context matters.*

If you follow one of the trackbacks you'll land on this post on the Awasu blog that offers an insight on the whole thing-- that it's about *giving a damn*. That the Japanese are raised to give a damn about doing a job well, and that the aesthetic sensibility and attention to detail is simply one of the natural outcomes. Context matters.

And to respond to one of the comments, no, I don't think American design sucks. American Design is fabulous... the point is not comparing American to Japanese design, but rather comparing the context in which design exists in the two countries. American design is actually Design (capital "D"), done by Designers (and done well). In Japan, design is a stronger part of the culture whether its a tiny patch of grass an old woman crafts into a beautiful garden, or a city manhole cover, or a box lunch. It infuses *everything*. It's studied and practiced in a hundred different ways by a much greater range of the

population than occurs here. Here, in the US, Design is practiced by Designers. There, in Japan (and many other countries as well... Sweeden comes to mind), design is practiced by both Designers and... *designers*. Regular people conducting their work or pesonal lives with an appreciation that most of us did not get (unless we either pursued studies of Design/Art, or were, say, raised by a designer or architect).

And one last point on this that also came from Dan Pink's book--while I'm heaping praise on the design/creative sensitivities of the Japanese, ironically this aspect of Japanese culture has been supressed in their education system over the last many decades while they set out to kick our ass in cars and electronics. But things are changing... here's a quote from the book;

"Japan, which rose from the ashes of World War II thanks to its intense emphasis on L-Directed [left-brain directed] Thinking, is now reconsidering the source of its national strength. Although Japanese students lead the world in math and science scores, many in Japan suspect that the nation's unrelenting focus on schoolbook academics might be an outdated approach. So the country is remaking its vaunted education system to foster greater creativity, artistry, and play. Little wonder. Japan's most lucrative export these days isn't autos or electronics. It's pop culture. Meanwhile, in response to the mind-melting academic pressures on Japanese youth, the Education Ministry has been pushing students to reflect on the meaning and mission of their lives, encouraging what it calls, "education of the heart."

Wow... think about that for a minute or two.

Then be sure to read Dan's book and if you're interested in the Asian vs. Western thinking research (the studies are really fascinating!), check out The Geography of Thought. And meanwhile, I'm searching for ideas on how I can improve my own skills in Thinking In Context. I always fancied myself pretty good at that, but the fact that it took sheer force of will on the part of the Japanese Sun folks before I understood why the context questions belonged on the exam makes me question that...

http://headrush.typepad.com/creating_passionate_users/2005/05/context_matters.html

Users don't care if you are the best.

By Kathy Sierra on May 2, 2005

I know I'm preaching to the choir here, so this is directed at the people who aren't reading this but *should*:

Your users don't care about how fabulous *you* are. How fast your product is. How many awards you've won.

If we want to inspire our users, we have to care about how fabulous *they* are. How fast *they* are. How many awards *they* might win as a result of using our products or services. That's what sociologists, psychologists, and cognitive scientists tell us. It's what biologists and anthropologists tell us. *Self-interest is hard-wired into the brain*. That doesn't mean people aren't capable of thinking of others...but let's face it--when your user makes a list of the people he cares most about, *you're* not in the top ten.

We've talked about this in other blogs including Users shouldn't think about YOU, and How to create a non-fiction bestseller, but since it's my favorite theme, here I am again.

Because I just keep wondering why so many advertisers/marketers/companies/individuals keep promoting how great *they* are... how *they* are better than the competition, blah blah blah, rather than focusing on how important the *user* is, and better still... how this product or service will enhance the user's life.

And we're not talking some Big Important Deep Cosmic Spirtual Thing. Software developers, teachers, designers, car salespeople... we all have a chance to frame what we build and do in terms of how it helps the user kick ass. Here's what we wish employers and prospective clients would say to the person or company they're considering:

"Quit telling us how great *you* are, and start telling us how you plan to deliver something that helps the *user* become greater."

Or... "We care about the lives you *touch*. We want first-person testimonials. We want to hear from the guy who got a raise because of what he learned from your blog. We want to hear from the woman who laughed so hard coffee came out of her nose because of your game. We want to hear from the couple who found a shared interest because of your product."

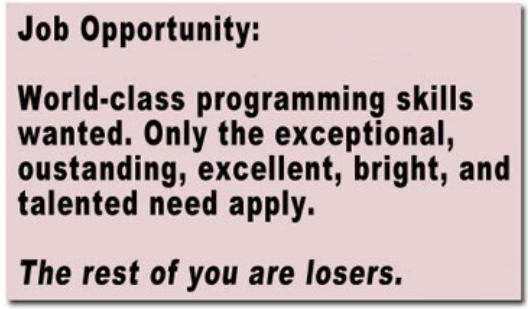
But again, it doesn't have to be anything earth-shattering. Think about the seemingly little things a company's product or service has done for you like... Made you smile. Made you feel--and *be*--a little smarter. Made you catch your breath over the beauty, quality, or sexiness of the product (or hell, even just the coolness of the *package*... anyone who's kept their iPod box beyond any possible reason knows what I'm talking about). Helped you take--or digitally alter--or display--a photograph that makes your child look as happy as you knew he was when you took the shot. Made you look like a million bucks (sorry Hugh, I meant *quid*). Helped you become just a shred more passionate about something you love. Better yet, helped you become passionate about something you didn't even know you *liked*.

So... who have you helped kick ass today?

Perhaps more importantly, for you passionate-user-creators, how are you making sure that you can hear about it? What can you do--or what can you ask your employer or clients to do--so that you can capture some of those testimonials? So many of those company feedback forms make me want to throw up because they're all about the company! The ideal feedback form would try anything possible to get the user/evaluator to talk about himself. So the next time an employer tells an employee what the users/customers think about the company or product or service, I'd love to see that employee respond with something like, "That means nothing to me. Tell me what the customer feels about *himself* as a result of our company..." Right. ;)

Hire Different

By Kathy Sierra on May 4, 2005



This isn't a real ad, but I pulled all of the attribute words (world-class programming skills, outstanding, excellent, bright, talented) from a Google job listing. *The last line about being a loser is all mine ;)* But it's not just Google that's looking for the best and brightest, of course.

I have to admit that this sounds exactly like the kind of developers *I'd* love to spend time working with. They'd be good for me. They'd raise *my* skills, and I'd probably get a little smarter just being *near* programmers who are world-class, exceptional, outstanding, excellent, bright, and talented. And there are plenty of people out there who meet that criteria.

The trouble is, those who meet that criteria often tend to be... *similar*. There's a reasonably good chance that they *got* to be world-class developers by having a somewhat similar background, from the C.S. degree at a top-notch school to work experience at a recognized company.

And in the US, that means they also tend to be under 45, white, and male.

So what?

According to James Surowiecki's The Wisdom of Crowds, that lack of diversity can hurt both innovation and decision-making. Sometimes with terrible consequences.

But he contends that it's not necessarily the lack of *demographic* diversity that's at the heart of the problems... it's *cognitive* diversity you need. If those doing the hiring are going after only world-class, exceptionally bright people with similar

skills, the *differences* between the Chosen Ones may not be that useful. He claims the company needs to hire *not just only the smartest people!*

And he cites lots of studies to back this up. Studies that demonstrate that a more diverse group, with fewer of the smartest people, under the right conditions, will consistently make *better* decisions than a group made of nothing but the smartest people. It's a pretty compelling argument when you look at the research he points to (although you might not always agree with his conclusions on some of it.)

One dramatic example involves what happened at NASA with the Columbia disaster. I won't go into his details (it's nearly a chapter long and includes other group dynamic factors besides lack of diversity), but here's one of his main points:

"What was missing most from the MMT, of course, was diversity, by which I mean not sociological diversity but rather cognitive diversity. James Oberg, a former Mission Control operator and now NBC News correspondent, has made the counterintuitive point that the NASA teams that presided over the Apollo missions were actually more diverse than the MMT. This seems hard to believe, since every engineer at Mission Control in the late 60's had the same crew cut and wore the same short-sleeved white shirt. But as Oberg points out, most of those men had worked outside of NASA in many differrent industries before coming to the agency. NASA employees today are far more likely to have come to the agency directly out of graduate school, which means they are also far less likely to have divergent opinions. That matters because, in small groups, diversity of opinion is the single best guarantee that the group will reap benefits from face-to-face discussion."

I'm not doing his arguments justice here, because it really does take the whole book to explain how--and why--all this works. But it made *me* think that Hire *Different* should be just as important as Hire *Smart*. I would hope that all hiring managers everywhere will read this book and perhaps get a new (counterintuitive) insight into why they might actually get a better result by, um, *lowering their standards*. Although I don't think of it as *lowering*, since candidate A who has this different perspective but isn't, say, as young, high-IQ, or classically-trained as candidate B, might bring something even more

valuable. In other words, what you lose in IQ points might be more than made up for by other things...

And in A Whole New Mind Dan Pink has a startling statistic: *IQ accounts for less than 15% of career sucess*. (Then he mentions research that suggests the most effective leaders are those who are *funny*--those who have their employees laughing much more than other managers do.)

I hope every hiring manager reads both of these books and at least *considers* some of their main points. And if you're *looking* for a job, the studies/research/stats in these books might give you a little more ammunition when you're up against the "we only want to hire people just like the world-class people we already have" attitude. It might help you learn to frame/position what you *do* bring, in ways that might not be immediately obvious.

http://headrush.typepad.com/creating_passionate_users/2005/05/hire_different.html

Fine-grained treats = user happiness

By Kathy Sierra on May 5, 2005

What makes your user's brain happy? What makes *your* brain happy? British novelist Iris Murdoch said it best:

> ***"One of the secrets of a happy life is continuous small treats."***

And the current issue of Scientific American Mind backs her up.

In an article called "Make Yourself Happy", author Maja Storch explains that personal happiness has two components: short-lived/immediate and long-term/habitual. "Short term pleasures create a stirring of emotions that psychologists refer to as positive affect", she says, "Most individuals underestimate the power this factor can have in both their private and professional lives." And my favorite:

> ***"One extravagant annual company picnic does not create a healthy working environment; it takes many immediate, smaller happy moments to achieve this atmosphere."***

So it looks like we're better off thinking about ways to delight our users and customers (and employees and family members!) with a steady stream of Good Things rather than, say, giving them one big reward.

I think most of us know this intuitively in our personal lives... most people seem to prefer a year's worth of repeat Small Special Moments to a year of nothing (or worse) followed by a fantastic birthday present (unless it's a 20" iMac G5 wrapped in

a 22 ft. AirStream CCD, in which case the entire previous year can pretty much suck and everything will be fine.)

But so many companies seem to feel like they can make up for a lot of user pain as long as they do something spectacular every once in a great while--like offer a huge discount on a related product, or when your frequent flyer miles finally pay off and earn you a trip.

And it's not just a matter of regularly delivering small treats that users (family members/employees, etc.) *expect*, or the effect loses its power. This is where animal clicker training has something to say:

Intermittent, *unexpected* treats are more powerful than regularly scheduled *expected* treats.

The question is... *how*? I talked about this earlier in Creating Playful Users, and it seems like the big keys are the things I've already mentioned:

Rewards/treats should be both fine-grained and surprising

What constitutes a "treat"? Obviously that depends on who your users are and what their relationship is to you, but here's a random list:

* Easter eggs in your software

* *Unexpectedly* and *uncommonly* good customer service or support experiences

* Something unexpected and special in the box your product ships in... (but in order to be unexpected it has to be *changed* on a regular basis).

* A special feature that doesn't get in the way but says...*we were really really really thinking about you here*. I'm finding a lot of these in the new Mac OS X Tiger release! (Like "mail PDF" that lets you go from viewing a web page to mailing it as a PDF email attachment in *one* step!)

* Sponsoring and supporting user groups with a variety of special treats... everything from study guides and posters to raffle t-shirts and other cool giveaways.

* Special surprises (extra downloads that only customers get, something fun in the mail, etc.) that show up at the user's mail/email unexpectedly. (I *always* stay for the entire credit roll

when I see a film in the theater (unless I hated the movie) out of respect, sure, but also because every once in a while you get an entire new scene (or really fun outtakes) that happens only *after* the credits are over... Napolean Dynamite and Constantine are two that come to mind).

Too many companies seem to give all the cool toys and treats to *prospective* customers--like trade show attendees, for example--but completely ignore you once you actually BUY the thing! That's just 180 degrees wrong. If they're pouring all this effort into enticing *new* customers, I can't help but think that if they channeled more of that budget to their *existing* customers (through both having a great product *and* continuing to surprise and delight them *after* the sale), then they'd increase their sales and marketing force by an order of magnitude as those customers go out and evangelize with way more credibility than the company reps or ads will ever have.

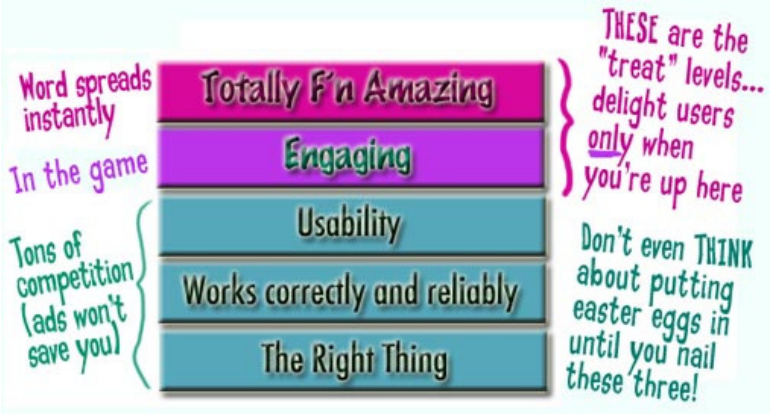
The message from the brain folks (and Iris Murdoch): spend less time thinking about The Big Reward and more time dreaming up and delivering the small treats. Write your significant other a funny message on a post-it and stick it somewhere surprising. That takes, what, 20 seconds? Slip a chocolate rabbit inside your employees' in baskets (you'll have to read the Scientific American Mind issue to understand *that* one). Don't punish your developers for putting an easter egg in the software...*encourage it*.

And I'm just following brain science when I run over to Ben & Jerry's as soon as I finish this post...

http://headrush.typepad.com/creating_passionate_users/2005/05/finegrained_tre.html

The case for easter eggs (and other clever user treats)

By Kathy Sierra on May 9, 2005



My previous post on user treats drew some arguments both for--and passionately *against* easter eggs in software. But in each of the arguments *against* easter eggs, the reason is virtually the same: "Why the hell are you spending your time creating these "surprises" when you haven't even bothered to fix the glaring *bugs*?"

So let's get this out of the way right now...
Until you've nailed the fundamentals--the things users want, need, and expect--don't bother trying to "surprise and delight" users. That just pisses 'em off.

That's why I updated the graphic I made for my earlier how to break through post to show the place a product or service should be *before* easter eggs come into the picture.

But this blog is about creating *passionate* users, not merely *satisfied* users. If you're still dealing with the bottom levels of the hierarchy, *you've got bigger issues to deal with first*. The scary thing is, in those lower levels you're kicking and clawing and trying to buy your way to market share amidst what is, for most of us, brutal competition. And that's *not* a fun place to be, financially *or* emotionally and spiritually.

So now that we're talking *only* about those who've satisfied user requirements and expectations...

A good easter egg is a playful, hidden or disguised feature that, when discovered, can offer surprise, delight, entertainment, humor, novelty, or an "I Rule" experience.

I'm not just talking about *software* easter eggs--you'll find them in movies, music, posters, logos (have you noticed the little extra image within the FedEx logo?), books (our Head First books have quite a few), games, magazines, *blogs*...

Brains thrive on the "OH!" moments of discovery.

They love to stumble on things, and even better--they love to *figure things out*. A lot of easter eggs are like puzzles waiting for you to find their second meaning, *especially* when that meaning requires some "insider" knowledge or skill.

If user engagement is a Good Thing (and for what most of us are creating, selling, writing it *is*), easter eggs can be a powerful ally in making that happen. Done right, easter eggs *can* add value that (unless you're doing a mission-critical app where undocumented code is a security or safety risk) is *worth it*.

Characteristics of good easter eggs:

* **They get users involved** as a participant rather than a passive specatator (people have told us they've through our books a second time *looking* for some of the easter eggs once they've found a few and realize that they're in there).

* **They get users to spend more time**

* **They're *remembered*** (you can use easter eggs to enhance learning)

* **They DO NO HARM!**

* **Their discovery is NOT a required part of the experience.** If it *is*, then it's not an easter egg, but simply a part of the product.

* **They give users the "I Rule" experience of being clever enough to "get it"**, especially if it's in plain sight, but requires "insider" info to recognize it. For a particular audience, for example, that might be "NCC-1701" as the license plate on a car. Just about every programming book (ours included) uses the number "42" in a statistically unlikely number of code examples. And there are more Monty Python references scattered in books and movies than any of us can possibly know.

Think about the number of movies that include lines made famous in *other* movies, or feature cameos from the director or an actor from the original movie, etc. but that only die-hard fans would discover. The person who "gets it" has the smug satisfaction of knowing that the line or joke would be lost on other mortals.

[In the new Hitchhiker's Guide movie, a key character from the original BBC TV show makes an appearance in the movie, as a *different* character. This recognition is not in any way required for you to enjoy the movie, but it's a wonderful delight and surprise to discover it, and those who do feel somewhat "special" for experiencing more in the movie than others will.]

* **They're entertaining in some way** (this is largely the point of making them)

* **They do NOT need to be funny, although they often are.** But they can be moving, inspirational (like discovering the developers were proud enough of their work to put in their personal signatures and sometimes photos), or thought-provoking.

Not everything that's entertaining and fun is *funny*... logic puzzles are fun, but not funny. Chess is fun, but not funny. Easter eggs can work the same way.

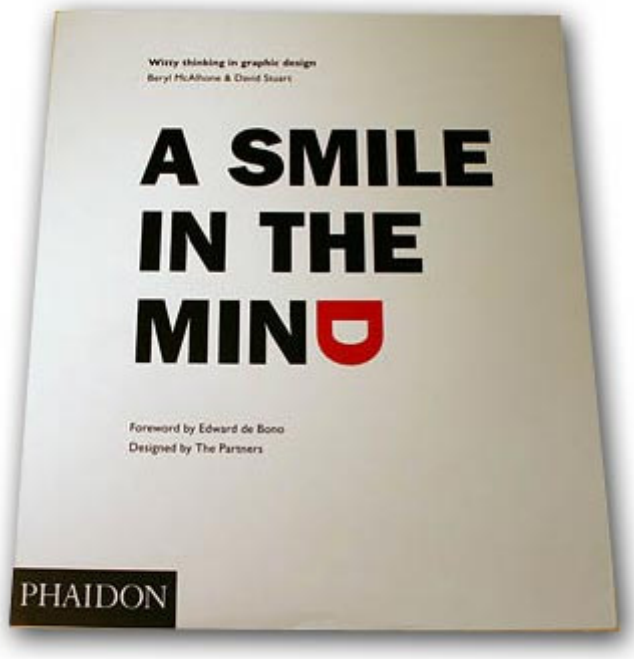
As a disclaimer, I'll state the obvious--we shouldn't be adding undocumented code to *anything* running a mission-critical application like, say, air traffic control or a nuclear power plant. Actually, we shouldn't be adding undocumented code to ANYTHING, but there's no reason a programmer can't document easter egg code, except for the whole getting fired thing... so there are obviously places you don't want to put these things. *But I'm not talking about those!*

So assume that we're all exercising common sense, and we're now trying to go beyond the basics in our products and services. We're trying to deliver more engaging, delightful, fine-grained, frequent treats to reward our users *and* the employees who get to make doing this **a part of their job description.**

To summarize: if you don't have the basics down, don't even *think* about adding special features like easter eggs. But if you're stopping at user satisfaction and meeting expectations, you're in for a bloody marketshare battle, because there's nothing stopping your competitors from doing the same thing. If you

want to compete for hearts and minds, you have to care about the higher notches of user experience, and easter eggs are *one* of the many tools you should have in your user-delight toolbox. Unfortunately, easter eggs have gotten a bad rap in many software shops, but some developers aren't so happy about that (like this Microsoft Longhorn evangelist).

And if you're looking for easter eggs in your own software, the best place to start is The Easter Egg Archive, which advertises "7884 easter eggs collected so far, 9 new in the last two weeks!"



Finally, you might want to check out this wonderful book on "witty thinking in graphic design", A Smile in the Mind. Although it's specific to design, it's a visually-rich (and very entertaining) book with a message that applies to virtually anything from teaching a class to writing a book to developing software to decorating your house. Leave it on your coffee table and watch what happens...

http://headrush.typepad.com/creating_passionate_users/2005/05/the_case_for_ea.html

Reverse-engineering passion: part 1

By Kathy Sierra on May 12, 2005

Part One: What it looks like (why we care)

Like all good geeks, I can't let something important remain unanalyzed. If we're talking about passion, we better know a little something about what that means. The best way to *create* passionate users is to figure out:

1) What it looks like when people are passionate about something

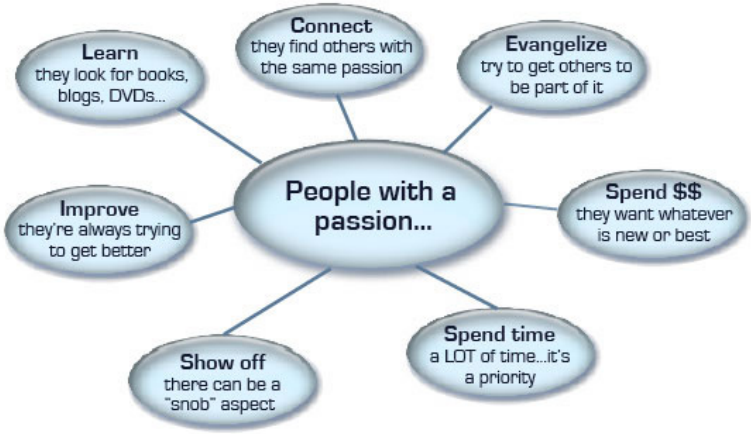
2) What kind of things people are passionate about

and finally...

3) The characteristics of the things people are passionate about

We're not going to leave it to chance or fads.

This post is about #1, What it looks like when people are passionate. This defines *why* we want it. It defines our goal! We hear people talking about wanting (or already having) "passionate users", but when they describe what it looks like, it's closer to "satisfied and happy" users. And since we're going for the full passion monty here, we can't stop with that.



If you're serious about creating *passionate* users, this is what it looks like. *This* is what we're trying to build. When making a decision about something, we have to ask the question, "Will this thing we're about to do support any of the things on this map?" In other words, are we doing something with our product, service, marketing, etc. that will help the user do any of the following:

Evangelize
Connect
Learn
Improve
Show Off
Spend Time
Spend Money

Too often, companies seem to focus only on the last one -- they're quite happy to find ways for the user to spend more money, but ignore the others. So let me add another question. Besides asking, "How is what we're about to do going to help us support one of the seven passionate characteristics?" we *should* be asking, "ARE we supporting all seven things?"

Do you help users connect with others who share that passion?

Do you have a way for users to learn more?

Do you give users a clear path for improvement, so that they're motivated to keep getting better? (Under the assumption that the better they are at it, the more they love it. Think about it...)

Do you give users a way to show off their expertise (the "more [insert here] than thou" thing)

Do you give users opportunities to spend more time on this passion?

Do you give users a way to spend more money around this passion?

Do you give users support for evangelizing to others?

Granted, *you* don't have to actually do all of these... you can support other third-parties in doing them for you.

For example, my co-authors and I are doing several of these things for *Sun*, without any direct support from Sun. I originally created javaranch.com, which is now the single largest Java community "fan" site on the internet. Between javaranch and/or our books, we support:

Users can **connect** with others.

Users can **learn** and *improve* through forums, articles, lessons, etc.

Users can **show off** either through answering questions, contributing articles, or--even better--by becoming "bartenders" (forum moderators).

Users can **spend time** on the site (to the great delight of their employers and family members ;)

Users can **evangelize** on the various discussion forums.

Users can **spend money** (which we sometimes hope will be on one of our books... hey, we have to eat too)

This support we provide is all part of Java's Passionate Wake, and yet Sun didn't do a damn thing to help us (well, other than create a wonderful programming language which we believe is passion-worthy).

Sun's job? Stay out of our way and let it happen! At one point a few years back, Sun's legal began sending threatening letters to javaranch (after I had turned the site over to it's current owner, Paul Wheaton), for using the word "Java" on the site including in the name of the site itself. They suggested some lovely changes. Paul wrote back saying, "Hey, we'll be happy to rename it .NetRanch or maybe C#Ranch..." and the whole thing was slashdotted making Sun out to be the big bad guys going after

their number one fan site. Rumor has it that James Gosling found out, and--virtually overnight--the whole "misunderstanding" was cleared up and Paul got a call from Sun marketing with a solution that would solve everyone's problems and make it possible for javaranch to carry on while still allowing Sun to protect it's trademark.

OK... back to the seven things. Again, you and your company don't need to personally *do* all seven things, but they are characteristics of passion, so if you want passion--they need to be somewhere in the equation. So if *you* don't support them, you need to help and encourage others to do it for you. Don't try to stop someone from making money off something *you* have built... because there's an opportunity cost for you if users don't get to do these seven things until YOU'RE ready to make them happen.

By giving up control--especially over the need to be the only one profiting from your creation--you greatly increase the chances that these seven things will happen more quickly, which in turn increases the chances that more and more people will become passionate. Truly passionate, not just satisfied or happy.

But if nobody is stepping up to support some of these things, then you better look for ways to kick-start the process. It could be as simple as starting a blog and providing instructions and materials for how to start a user group, or as complex as developing training programs, fan sites, and more.

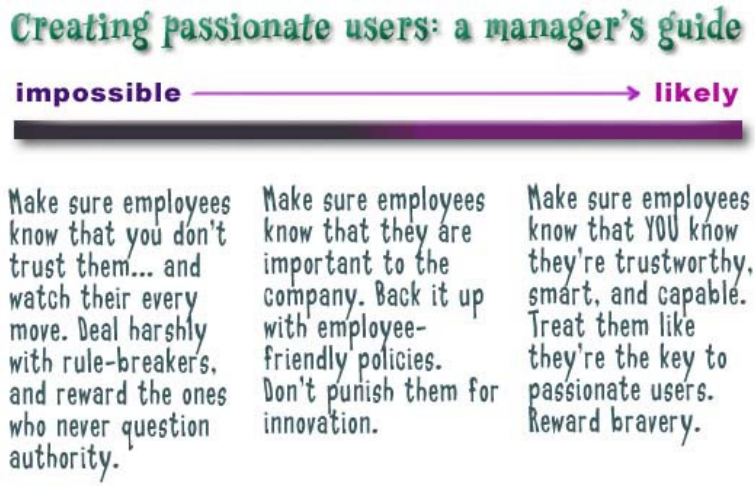
Next up: we'll look at the things people *are* passionate about, and see if we can extract some useful tips, tricks, and data from that.

And, oh yes, don't worry if you're thinking, "my product could NEVER have those things... I make trash bags." We're still going to answer that one too... but you'll just have to keep reading ;)

http://headrush.typepad.com/creating_passionate_users/2005/05/reverseengineer.html

Management's role in passionate users

By Kathy Sierra on May 23, 2005



A few months ago, Skyler started working part-time at a Mrs. Fields cookie store at a local mall here in Colorado. They treated her--no, *all* employees--like ex-convicts. The default company assumption was "Employees are NOT to be trusted!"

This came through in policy decisions, but never more obvious than the what-to-do-with-leftover-cookies-at-the-end-of-the-night policy:
Employees must throw away ALL unpurchased cookies at the end of the night. Employees are expressly forbidden from taking leftover cookies home.

Skyler is the kind of person who collects stray animals and, in some cases, stray *people*. She has a soft spot for the homeless. She'd be delighted to take a nightly walk down Pearl street (or one of the other places in the area where you might find street folks of various flavors) and hand out leftover cookies (which, as a somewhat-obnoxiously born-again vegeterian/health nut, she'd accompany with a lecture on nutrition...)

But no, those cookies are destined for the trash heap. If she wants to take them, she'll have to pay for them. Because the company policy of "you must throw the cookies away" is based on the assumption that Employees are Bad. They cannot be

trusted. If they're allowed to take leftover cookies home--so goes the company's conventional wisdom--you just KNOW what they'll do--they'll get closer to the end of the night and then go on a baking binge to generate as many leftover cookies as possible.

But hmmmm... if that's the *worst* that can happen... could it *possibly* be worth the bad will it creates between employees and The Management? And while it doesn't take a rocket scientist to recognize that treating employees this way is NOT the path to stellar customer service (let alone something like passionate users), I'm stunned that this kind of management practice still happens.

Let's say the cost of the "extra" cookie dough produced by the highly immoral college student with the cookie fetish is, oh, $60.00 per month. This adds up, sure. But what about the *cost* of the policy aimed to prevent it? Skyler couldn't *wait* to find another job, in large part because of this attitude of distrust. The cost of employee turnover probably averages in the *hundreds* of dollars per month, and it's no stretch to assume that the less you trust your employees, the higher the employee turnover.

So the company LOSES money on the policy because what they save in cookie dough they lose in the costs associated with poor employee retention.

And we haven't even touched on whether this ripples through to actual cookie revenue in the store. Do employees who aren't trusted behave as nicely to the customers as those who ARE trusted? Perhaps it's subtle--after all, Skyler isn't going to be rude to people regardless of the company's policies. But still... that little drain on her personal enthusiasm while at work infuses everything she does, and that includes every interaction with customers.

Of course, most of us are not entry-level employees at a fast-food mall store, but it's amazing how this attitude of mistrust exists in other companies for even the high-paid individual contributors from software developers to designers. Even if the company doesn't have these kinds of "we don't trust you" policies, their lack of trust still shows. Managers who question everything you do...who don't believe you're capable of working outside the strict procedures and rules set down by Those Who Know All Things And Make The Important Decisions.

I'm continually surprised by companies that hire someone for, say, a $100K a year position, and then treat them like they might do or say the wrong thing at any moment. They aren't allowed to talk to the press. They aren't allowed to *blog*. They aren't allowed to make critical decisions about the customers. *They aren't allowed to do what they THOUGHT they were hired to do!*

Every contemporary management book and philosophy (and just about every manager) says that the key to successful management is: "Hire good people and then get out of their way." But how many companies or individual managers actually *do* that?

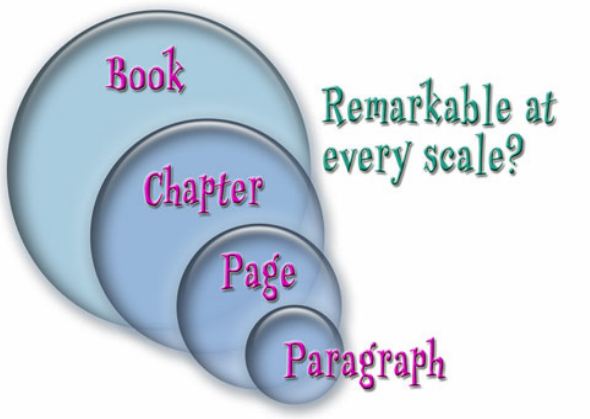
The footnote to Skyler's story is that she worked at Mrs. Fields only until the nanosecond that she found another job, which she did, at the Boulder Einstein's Bagels. And when I drop in for a latte or a bagel, I watch her in action interacting with the line of customers (the place is BUSY) and I notice the change. She's always *nice*, but there's something more. I now see in her the way people act when they know they're trusted and respected, and I *swear* the customers can feel it. And if even 2% of those customers decide to come back again that week simply because they had such a pleasant, energetic encounter with the bagel clerk...

[FYI: I've been out of commission for the last ten days, but I'm back :) Sorry about the missing blogs... and Beth couldn't jump in because Beth and Eric are in the midst of physically driving/moving from Santa Fe back to Bainbridge Island in Washinton. If you've emailed me in the last week, I'm still trying to catch up. Thanks for participating while we've been gone!]

http://headrush.typepad.com/creating_passionate_users/2005/05/managements_rol.html

Is your book, manual, or website remarkable (or recognizable) at every scale?

By Kathy Sierra on May 30, 2005



There's a game I used to play where you take a really small image from the painting of a famous artist and try to identify it. The trick is to see how small a sample you can use before you can no longer recognize either the painting or the artist. It's amazing just how identifiable a Van Gogh or a Monet or a Kandinsky or a Miro is, just from the tiniest slice. It's a wonderful game to teach yourself to really *see* the way the artist used color, texture, light, shapes, lines, etc.

Now, take the nearest computer book on your shelf and open it to a random interior page somewhere in the middle. Can you tell who the publisher is just by looking? Can you tell who the *author* is? Go a little further and start reading a paragraph. *Now* can you tell?

That's the problem.

The books might be easy to differentiate on a *larger* scale like, say, the level of a chapter or the whole book. A book from author "A" might cover the whole of the topic in a very different (and substantially better) way than author "B", but at smaller scales... can you tell the difference? Is there anything distinct about the look and feel? About the writing?

Why *shouldn't* a book be a reflection of the brand? Most publishers will tell you that they *are*. They enforce editorial standards and layout guidelines to help ensure *consistency*. **But consistency is not enough!** Not *nearly* enough to make a memorable impact. Not nearly enough to be even identifiably unique, let alone *remarkable*.

So why don't more publishers do more to ensure that their books are recognizable (and ideally remarkable) at every scale? Why don't more *authors* put their pages to the test... the "flip to a random page and see if there's anything different from the 30,000 other currently-shipping computer books on Amazon" test? A lot of authors don't because they're writing to strict formatting guidelines, and have no influence on the layout and style. And that's not always a bad thing... a lot of authors certainly don't pretend to be interior book designers. But they can still do it with their *writing* and *information* style. But I read so many paragraphs that could be so interachangeable with another book from a different publisher and author on the same topic.

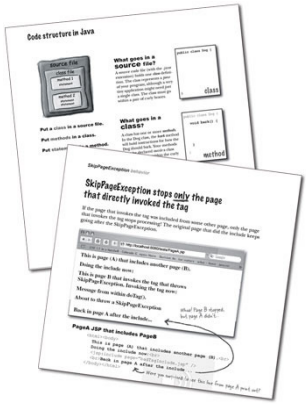
There are, of course, a ton of authors whose paragraphs you *can* recognize. Peter van der Linden, one of my favorites, immediately comes to mind along with my good friend Solveig Haugland. (Interesting twist -- Solveig now helps *edit* Peter's books...) And I can always tell (and enjoy) Bruce Eckel's books.

I'm not suggesting this recognizability is the most important thing -- you could certainly print each terrible paragraph in day-glow orange and it would be recognizable, even *remarkable*, but still a terrible book. But let's say we've crossed the threshold and we have good writing, good content, technical expertise... all the things a good computer book needs to have. Now what? How do you begin to differentiate yourself from all the other *equally good* books? We all know that you can do it simply by being the *first* out with a book on a particular topic, but that's not a sustainable and healthy strategy.

The *best* way, in our opinion, is to create the book for the *user*, using the approach I suggested in How to write a non-fiction bestseller. But we're talking about a different, smaller scale in this post...

So which brands/books *are* recognizable at every scale? Certainly the Dummies series does this. I believe the O'Reilly Missing Manuals series does this, as does their Hacks series.

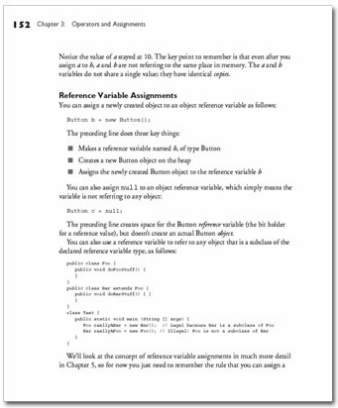
The Visual QuickStart guides from Peachpit are pretty easy to spot. Our Head First books pass the test pretty well:



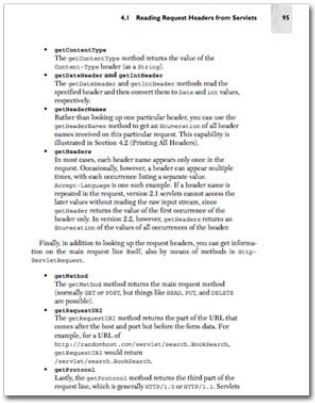
Our intention was for each page to look as though it was constructed by a somewhat strange instructor using a whiteboard and markers to draw things. It's supposed to have a kind of friendly hand-drawn classroom feel. That's not the feeling everyone *wants* from a technical book, but it works for our audience (shameless self-promotion: Head First Design Patterns was THE #1 bestselling computer book on Amazon for part of last week, according to their bestseller list--way to go Eric and Beth!), and they can spot it on virtually every page.

And it's not just in the look and feel (fonts, graphics, etc.), but in the actual writing style.

But just so you don't think I'm too full of myself... here's the *first* book Bert and I wrote, a couple months before we designed the Head First series:



Looks just like *every other computer book*. In fact, for comparison, here's another page from a *different* publisher. Can you tell who published either of these books? (Neither are from O'Reilly).



4.1 Reading Request Headers from Servlets 95

- **getContentType**
  The `getContentType` method returns the value of the `Content-Type` header (as a `String`).
- **getDateHeader and getIntHeader**
  The `getDateHeader` and `getIntHeader` methods read the specified header and then convert them to `Date` and `int` values, respectively.
- **getHeaderNames**
  Rather than looking up one particular header, you can use the `getHeaderNames` method to get an `Enumeration` of all header names received on this particular request. This capability is illustrated in Section 4.2 (Printing All Headers).
- **getHeaders**
  In most cases, each header name appears only once in the request. Occasionally, however, a header can appear multiple times, with each occurrence listing a separate value. `Accept-Language` is one such example. If a header name is repeated in the request, version 2.1 servlets cannot access the later values without reading the raw input stream, since `getHeader` returns the value of the first occurrence of the header only. In version 2.2, however, `getHeaders` returns an `Enumeration` of the values of all occurrences of the header.

Finally, in addition to looking up the request headers, you can get information on the main request line itself, also by means of methods in `HttpServletRequest`.

- **getMethod**
  The `getMethod` method returns the main request method (normally `GET` or `POST`, but things like `HEAD`, `PUT`, and `DELETE` are possible).
- **getRequestURI**
  The `getRequestURI` method returns the part of the URL that comes after the host and port but before the form data. For example, for a URL of `http://randomhost.com/servlet/search.BookSearch`, `getRequestURI` would return `/servlet/search.BookSearch`.
- **getProtocol**
  Lastly, the `getProtocol` method returns the third part of the request line, which is generally `HTTP/1.0` or `HTTP/1.1`. Servlets

Yeah, that's what we thought. Nothing identifiable. Nothing unique. Nothing recognizable. Nothing *remarkable*. At least not at the level of the page look and feel. The first one, from Bert and I, is our Osborne/McGraw-Hill Java certification book. The second page is from a great book -- Marty Hall's Core Servlets book published by Prentice-Hall.

But they look the same.

Is that *really* a problem? Don't virtually all novels look pretty much the same inside, and after all--this is about *writing* and words are, well, *words*? Does (or should) the typography and column grid make any difference?

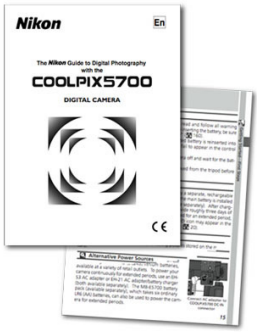
If you're writing *fiction*, I'd say no, it doesn't. Beyond basic readability. But then again, publishers have notoriously poor customer/market recognition. Almost *nobody* goes to the store believing their intention is to buy a book from a particular publisher. They go to buy a book on a particular topic, or from a particular author, or perhaps from a particular *series* (which is usually as close to brand recognition as a publisher ever gets).

But if you're writing *non*-fiction, I don't think it has to -- or SHOULD -- be that way. The problem with so many non-fiction books, especially books meant to be instructional, is that they're treated as "writing", when they should be treated as "experiences." Our goal is to change what's inside someone's head, and that might point to a very different approach than if

the goal is simply to "write a good book." So, I believe that publishers, authors, book interior designers, etc. should think more about the experience and less about the delivery of written words.

And if that experience is designed in a way that really works and is remarkable, then it *will* be recognizable at any scale, and will add to the power and memorability of the brand (and if all the other good things happen that we talk about on this blog, *may* even lead to *passion*).

And while we're on books, what about product *manuals*? I bought a Nikon Coolpix 5700 when it still cost over $1000. Nikon is a pretty cool company, and has some wonderful passion-inspiring things on their website (if they can make me a better photographer, they're going to train me to realize I need a more expensive model camera ;), but look at the manual that came with the camera:



For comparison, notice how it looks no more remarkable than the manual that came with my Canon digital video camera:

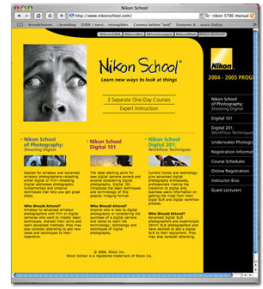
Absolutely nothing there to reinforce the brand. And although both manuals are decent, neither are particurlarly good. And neither go out of their way to try to make me *better* at using the

equipment, when if they DID, I'd be more inclined to buy the next thing they make -- including accessories and a more expensive and capable model.

In comparison, though, look at the manual that comes with a wonderful music software app, PropellerHead's Reason:

The manual has a nice look and feel that draws you in and makes you want to learn more about Reason. And the better you get, well, now I'm obviously going to have to upgrade to version 3.0...

And for one last contrast, here's what the part of the Nikon site looks like that includes online learning:



Now why can't the manual look at least a *little* bit like that?

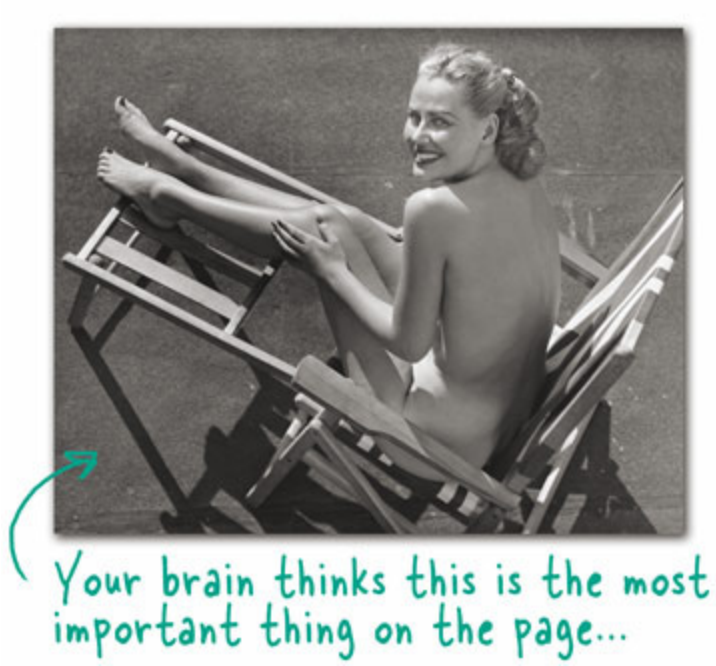
I think I'm going to do another blog on this topic of "remarkable at any scale", but in terms of things other than books and manuals. Maybe I'm just obsessed with mapping everything into fractals...

Thanks for being patient while I've been mostly offline folks. I think I'm really back this time :)

http://headrush.typepad.com/creating_passionate_users/2005/05/making_remarkab.html

Think Sexy

By Kathy Sierra on May 31, 2005



If you want to create passionate users, you need to understand *passion*. We do it in the geekiest of ways on this blog by reverse-engineering. But we can't just *study* it; we have to *feel* it.

Sure you can conjure up your own feelings of passion for skiing, dancing, golfing, coding, photography, etc. And we'll talk about that another time because it's crucial. But right now, let's think about... *sex*.

Call it neurobiological research. Call it marketing research. Call it... fun.

The brain cares deeply, profoundly, *passionately* about survival of the species. And that means sex.

But here I want to talk not about *sex*, but about the quality of *sexiness*. And for reasons we don't have to care about now, our brains seem to attribute sexiness to things that have nothing to do with a real breathing human.

A 45-year old programmer says, "Sure, this technology is ***sexier***, but we can't afford it now..."

A 29-year old attorney says, "That is the sexiest new sports car I've seen in the last five years."

A 17-year old student says, "That new iPod is really sexy."

I say, "I love this music... it's so damn sexy..."

A 32-year old graphic artist says, "That new package design is sexy."

A 65-year old architect says, "The curves of that new museum entrance are very sexy."

On it goes. And we're not talking about the *obvious* things like cologne in a bottle that's shaped like, well, you know. The unimaginative can simply use the shortest route to the brain's basic response to sex. They'll use the Coors Twins in an ad, for example, rather than come up with something more subtly clever.

But the rest of us can **Think Sexy** rather than relying on overt sex in our product design, marketing, adverstising, or in our case -- books (including covers).

Now, I'm guessing you spent more time looking at the picture at the top of this blog than the headline... *even if you are completely unaware of that extra time.* It just happens. Blame it on your chemistry. Blame it on your anatomy. But the more you personally respond to the notion of sexiness, the more likely you are to be able to conjure up the feeling when *you're* designing.

The iPod IS sexy. The Zen Micro is definitely *not*.

Given the overwhelming market share of the iPod, does that mean that most MP3-player buyers are simply shallow? Picking a product with as much sense as the 45-year old guy leaving his wife of twenty years to run with the cheerleader?

No.

We're not picking it because it's sexy. We're picking it because sexiness is part of what makes it a better product!

Better to hold. Better to use. Better to look at. ***Better to give you a good feeling.***

Don Norman talks about this in an essay Attractive Things Work Better, which Beth mentions in Why Cool is Good For Your Brain. (Side note: she's talking about attractive and cool qualities that aren't necessarily always *sexy*... sorry Beth and Eric, but however cool I think the Honda Element is, I don't think of it as *sexy* ;)

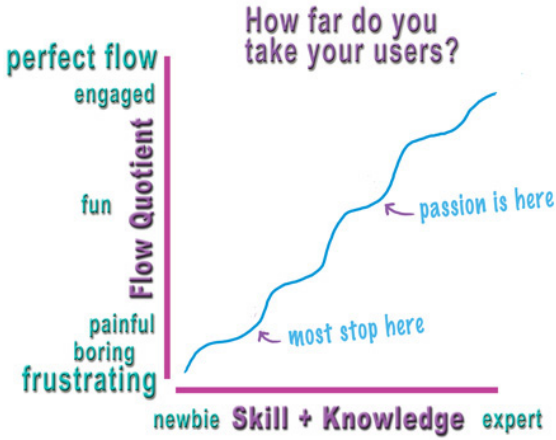
Whether you're designing a book, a software application, a piece of hardware, or a website... think sexy.

And have fun with the research.

http://headrush.typepad.com/creating_passionate_users/2005/05/think_sexy.html

Kicking ass is more fun

By Kathy Sierra on June 6, 2005



You know it's true. The better you get at something, the better it *feels*. Snowboarding. Programming. Writing. Learning Japanese. Chess. Painting. Building cars. Cooking. Designing a web page. Skateboarding. Teaching. Marketing. Being a parent. *Being in love.*

My running coach told me a few years ago, "It's just more fun when you're faster." I wasn't sure what he meant; I was just trying to get back in shape and do a decent 10K. But once I started training with much better runners, and began pushing myself and keeping my splits and timing my speed work... it *was* more fun. And it wasn't like I had any illusion of being competitive. Being better is just more fun.

The more we analyze and reverse-engineer passion, the more we see learning and growth as a key component. No, not *a* key--*the* key. **The more knowledge and skill someone has, the more passionate they become, and the more passionate they become, the more they try to improve their knowledge and skills.** (Much of it has to do with the flow state.)

Why are so many companies and causes doing virtually *nothing* to help users get better?

Assuming you have a good product or service or cause--just like everybody else out there we're all competing with:

It's not what you sell, it's what you teach that matters.

Or rather, what you help someone *learn*.

Too many books and businesses take users through the first steps and then leave them stranded and alone still in the frustrating and painful stage! How many readers claim they actually *finished* or even got halfway through a technical book? How many users ever learn anything but the most basic features of the software--even when they'd be *thrilled* if they could do more? But it just isn't worth it for them to struggle, so they stay with what they know, often using very inefficient steps to do something simply because that's the only "safe" way they feel comfortable with.

Kicking ass is more fun *regardless* of the task. It's more fun to *know* more. It's more fun to be able to *do* more. It's more fun to be able to *help others* do more.

I'll say more on this later, but I can think of a lot of wasted ad dollars that might be better spent *teaching*. Red Bull, for example, wants to be the drink of choice for late-night dancers. But rather than simply sponsoring raves and keeping popular DJ's well-stocked (like anyone else would in that business) ***they create new and better DJ's***. They offer the Red Bull Music Academy:

> *The Red Bull Music Academy is a unique environment where musical innovators shed light on the history, the motivations and the technology behind the tunes that we love. It's a place where ideas are expanded and friendships are forged in real time. It's where sonic theorists meet up with beat junkies and communicate the best way they know how - through music.*

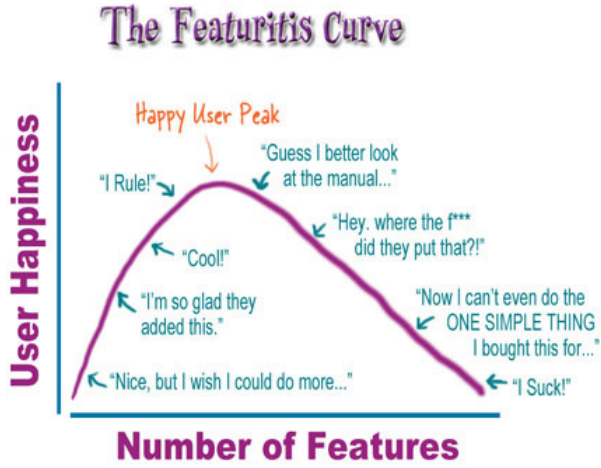
By helping more DJs (and wanna-be DJs) kick ass, they've done more to inspire *real* passion than any of their freebie promotions ever can.

So... how are you heping your users/customers/students/guests/visitors/clients/members/readers kick ass? What are you teaching them? How are you helping them get past the painful parts and into the better-than-drugs flow state?

http://headrush.typepad.com/creating_passionate_users/2005/06/kicking_ass_is_.html

Featuritis vs. the Happy User Peak

By Kathy Sierra on June 12, 2005



It's a gazillion degrees in my house right now, but I can't figure out the thermostat controls, so the heat's still on and the air conditioning unreachable. My new Denon receiver/tuner *sounds* amazing--good thing I'm using it mostly with my iPod; I have no clue how to tune in a radio station. When I bring up the newer versions of Microsoft Word, it looks so utterly foreign and overwhelming to me now that I give up and close it. And all I wanted to do was type a simple letter...

Most of you here know that Don Norman talked about this forever in the classic The Design of Everyday Things, but why didn't the designers and manufacturers *listen*?

My new Subaru factory-supplied car stereo uses that most evil of designs--*modes*. With so many features to support, they ran out of controls... so every control does multiple things depending on which *mode* you're in. None of it is intuitive or natural. Lose the manual and I'm screwed. Ten years ago, if you'd told me I'd one day need a manual to use my car radio, that would have been inconceivable. *All I want to do is find a frickin' radio station!*

Here's a little list of some of the things that seem to suffer the most from pushing too far past that "Happy User Peak":

* **Courses** that pack way too much content in. The learner is "exposed" to material that's "covered", but the learner hasn't

truly "learned" much and can't "do" much. Sun has a great 12-day Java course, except for one problem... *it's delivered as a five-day class*. The students leave on Friday with their heads exploding, unable to remember where they parked the car let alone how to compile their Java code.

* **Stereos** (or other consumer electronics and appliances) that use "modal" controls so that you cannot obviously figure out how to make it do the most BASIC FRICKIN' THINGS ; (

* **Software** that keeps adding feature upon feature until the simple things you used to do are no longer simple, and the whole thing feels overwhelming.

* **Technical books** that try to be "complete" but don't provide the focus and filtering and weighting the reader was hoping for. The more that's in the book, the longer it's going to take the learner (and the harder it'll be) to actually get through and *learn*. And the greater the chance that they'll stop reading before they become successful and have "I Rule" experiences. This seems to happen most when the publisher/editor/author didn't want to commit with both feet to being a learning book vs. a reference book, and tried to do both. When I see marketing copy for a learning book that says, "And you'll refer to it again and again after you finish..." or, "You'll want to keep it close even when you're done." red flags start flying. Reference books are for referring to (like the wonderful Nutshell series). Learning books are for reading once, maybe with some extra review, and a refresh if you don't use what you learned right away, but that's about it. (Note: our books *suck* as reference books.)

So again, why does this happen so often?

Our guess is *fear*. Fear of being perceived as having fewer features than your competitors. Fear that you won't be viewed as complete. Fear that people are making purchase decisions off of a checklist, and that he who has the most features wins (or at the least, that he who has the *fewest* features definitely *loses*). Fear of losing key clients who say, "If you don't add THIS... I'll have to go elsewhere."

Screw 'em. We believe that those providing the products and services that give the most "I Rule" experiences, without tipping too far over the Happy User Peak, will be the most successful. (Obviously there are a ton of exceptions, and yes of course I'm overgeneralizing.)

Push back. Of *course* you'll lose customers if you stop adding as many new features.

Or will you?

What if instead of adding new features, a company concentrated on making the service or product much easier to use? Or making it much easier to access the advanced features it already has, but that few can master? Maybe what they lose in market share in one area will be more than compensated for in another area. In a lot of markets, it's gotten so bad out there that simply being *usable* is enough to make a product truly remarkable.

We will resist the siren call of the market, because we believe the best path is:
Give users what they actually want, not what they say they want. And whatever you do, don't give them new features just because your competitors have them!

Each of our books, for example, covers fewer topics than its closest competitors. Yet we outsell *all* of them, and part of that is precisely *because* we cover less. Our readers learn fewer topics, but *nail* the important ones, and it turned out that for most people, *nailing it* was more important than *reading it*. Our readers put their trust in us to work hard at finding and focusing on what really matters, and brutally cutting the cognitive overload that comes with the rest, and we try not to let them down. (We definitely don't always get it right... I had to add a huge new chapter to the second edition of Head First Java, for example, because so many readers felt that collections/data structures were too important to have been relegated to an appendix.)

Be brave. And besides, continuing to pile on new features eventually leads to an endless downhill slide toward poor usability and maintenance. A negative spiral of incremental improvements. Fighting and clawing for market share by competing solely on features is an unhealthy, unsustainable, and un*fun* way to live.

Be the "I Rule" product, not the "This thing I bought does *everything*, but I suck!" product.

And I'll be your happy user :)

http://headrush.typepad.com/creating_passionate_users/2005/06/featuritis_vs_t.html

Building a successful online community

By Kathy Sierra on June 15, 2005

It was March 26, 2003, in the Santa Clara Convention Center in the heart of Silicon Valley. It was the ceremony for the closest

thing geeks have to an Oscar--the Jolt Cola/Software Development Magazine awards.

The last awards category was "Websites and Developer Networks".

First the finalists are announced, with all the usual suspects including Microsoft, IBM, BEA... and javaranch. WTF? *Javaranch?* It had no corporate sponsors. It was not a business. It was a quirky, no-budget all-volunteer community, run entirely by people who just wanted to be a part of it. It was simply a Java "fan" site--but a hugely successful one with numbers most sites would kill for--***over a half-million unique visitors a month.***

So how did Javaranch do it? (Oh yeah, they did win a 2003 award that night, and the next year as well, beating out Sun's java.net and Microsoft for a 2004 Jolt award.)

They did it by being passionately, single-mindedly, ferociously committed to enforcing one rule: "Be Friendly."

Not that you can't have a huge community without that rule... slashdot is the perfect example. But if you're trying to inspire ***passionate users***, I believe that enforcing a "Be Friendly" rule can be one of the best moves for long-term growth and retention of the community.

[Disclaimer: although I am the original founder of javaranch (in 1997), I'm not responsible for its real success. Most of the growth happened after I turned it over to Paul Wheaton. I gave javaranch its original heart and soul, but it is Paul and all the moderators (Sheriffs and Bartenders) who gave it a body and brain that could actually do something...]

Enforcing a "be nice" rule is a big commitment and a risk. People complain about the policy all the time, tossing out "censorship" and "no free speech" for starters. We see this as a *metaphor* mismatch. We view javaranch as a great big ***dinner party*** at the ranch, where everyone there is a ***guest***. The ones who complain about censorship believe it is a *public space*, and that all opinions should be allowed. In fact, nearly all opinions *are* allowed on javaranch. It's usually not about *what* you say there, it's *how* you say it.

And this isn't about being politically correct, either. It's a judgement call by the moderators, of course. It's fuzzy trying to

decide exactly what constitutes "not nice", and it's determined subjectively by the culture of the ranch. Sexy jokes are usually OK, racial jokes are not. Some perceive the *sexy* jokes as *sexist*, and therefore "not nice", but if we would laugh about it with our friends in a somewhat racy dinner party conversation, it stands. Javaranch censors for *meanness*, not to protect delicate sensibilities. To a lot of folks, that makes us "not nice", but we reckon these are the folks we wouldn't invite to our party, either. ;)

There is obviously no way to have a one-size-fits-all "be nice" rule; every culture will have its own. A church forum, for example, might draw the line much earlier.

I believe an online community can work with virtually *any* metaphor (I'll keep to myself what I think the slashdot metaphor is...), but that metaphor determines the kinds of people you attract and keep. The "frat party" metaphor supports one type of behavior, while the "public space" is another. The "professional business office" metaphor is different from the "passionate user group" model.

But the really good news is that if you have a strong and consistent culture, *whatever that culture is*, the community starts moderating itself. Kind of a hundredth-monkey effect... when enough people are behaving in a certain way, and that hits critical mass, it becomes not only accepted but *obvious* to everyone when it's being violated. (I talked about this earlier with respect to customer service in Can you teach someone to care?)

And for a wonderful article by someone who knows far more about online communities and social networks than I ever will, read Clay Shirky's speech from 2003 ETech, A Group Is Its Own Worst Enemy. Among other things, he talks about the challenges of balancing the idealistic goal of open and free speech with the atmosphere of the online community:

"And then, as time sets in, difficulties emerge. In this case, one of the difficulties was occasioned by the fact that one of the institutions that got hold of some modems was a high school. And who, in 1978, was hanging out in the room with the computer and the modems in it, but the boys of that high school. And the boys weren't terribly interested in sophisticated adult conversation. They were interested in fart jokes. They were interested in salacious talk. They were interested in

running amok and posting four-letter words and nyah-nyah-nyah, all over the bulletin board.

And the adults who had set up Communitree were horrified, and overrun by these students. The place that was founded on open access had too much open access, too much openness. They couldn't defend themselves against their own users. The place that was founded on free speech had too much freedom. They had no way of saying "No, that's not the kind of free speech we meant."

Pick your metaphor carefully. **Dinner Party** isn't for everyone, but it's usually my personal favorite for passionate user groups.

http://headrush.typepad.com/creating_passionate_users/2005/06/building_a_succ.html

T-shirt-first development

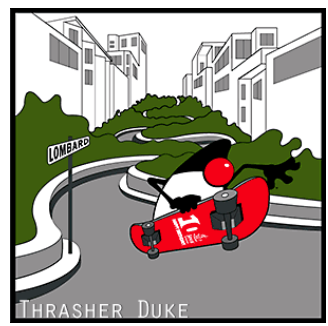
By Kathy Sierra on July 4, 2005

T-shirts *matter*.

This is what the merchandise store at the JavaOne conference looked like last week, after three days. Those bright green arrows are pointing to all the empty shelves. And the store was still open; that poor guy in the photo is choosing from among the two remaining t-shirt styles, one of which is *toddler-size only*.

Each attendee got a commemorative "Happy 10th Birthday Java" shirt just for registering, and vendors on the show floor gave out t-shirts like candy all week. So even though *everyone* had a pile of free t-shirts to take home, ***they couldn't wait to whip out their MasterCards for another one.*** Or maybe for a $50 fleece. Or a $300 leather Java jacket.

(I was really mad that they sold out of the "Skateboarding Duke" shirt before I had a chance to buy one! I would have paid *a lot*.)



Guy Kawasaki (the original Mac evangelist for Apple) said it in his 1992 book Selling the Dream: make the t-shirt *before* you make the product.

If you're a team lead, project manager, open source evangelist... make the t-shirt. If you're promoting a business, service, supporting a cause... make the t-shirt. And the more subversive, the better. If the t-shirt is for internal use only, see how far you can push before marketing or legal steps in. The more maverick the shirt, the more valued it becomes. At Sun, for example, there was always somebody trying to make an underground, unapproved shirt featuring the Java mascot Duke. If you were lucky enough to get one, that *meant* something.

I know...it's just a frickin' shirt. How can a t-shirt *mean* something? Think about it. Go look in your closet. Go look in your garage. How many special t-shirts are *you* holding onto for sentimental reasons? *Be honest*. How often have you lusted after someone *else's* limited edition shirt? If you're *really* honest, you'll remember the time you "borrowed" someone else's special t-shirt and "forgot" to give it back.

It's not just t-shirts, of course. It's bumper stickers. Window decals. Lapel pins. The back window of my car has decals for the two things I'm particularly passionate about--Apple and Parelli Natural Horsemanship".

A few years back, Wired online had a fun article on the marketing phenomenon of the Apple stickers. And I just saw a Jeep the other day with a window decal that said:

"It's a Jeep thing. You wouldn't understand."

I *believe* in these companies, despite whatever questionable things Apple (or Jobs) might do. I believe in what the Macintosh represents for the creative (and now, since OSX, geek) community. I feel that a small part of who I am is represented by the fact that I have--and *love*--Macs. And these aren't just shallow "coolness" values... but my sincere belief that because of the Macintosh, there are ways in which I kick ass that weren't possible before. Ways in which--through the books I create--I am helping others learn to love what they do and do what they love. [I think it's just as cool when people have a passionate anti-

Mac stance. Their rejection of all-things-Apple is something they're proud of.]

And I believe intensely in what Parelli has done for the state of horsemanship throughtout the world... helping hundreds of thousands of people move from a controlling, dominating relationship with horses to one of partnership and willingness and playfulness.

For me, the key intersection of these two companies is JOY. *Mine*. The real question is why I--and so many others--want to share (or show off) their relationship to a company, cause, product, idea, band, sport. We'll save that exploration for another time, but for now -- the main point is this:

Where there is passion, there are t-shirts.

Where there is passion, there are ways to *express* that passion to others, with t-shirts and bumper stickers and mugs as the primary vehicle. Does this mean that we want the t-shirts *because* we have passion for these things? Obviously, yes. But what if there's something even more interesting here... **what if some part of why we're passionate is *because* of the t-shirts?** And no I don't mean that we choose what to believe in simply because it's got a cool t-shirt (although, there's some shred of truth in that. I chose to run my first half-marathon, despite being in no way trained for it, because I HAD to have the t-shirt, and that was the only way to get one). What if the availability (and quality) of these "pride items" help to reinforce and build on the passion we have the potential for developing?

Remember, a big part of passion is connecting with others who share that passion. And showing your support/enthusiasm/belief is an element of what makes you a *member of the group*. By sporting the shirt, you *belong*.

So to those who see this as just one more terrible example of American consumerism -- worshipping the corporate logo gods -- I think that's missing the bigger point. It doesn't matter if it's a company, or a sport, or a cause. The "pride items" are about announcing some small piece of who you are to the world. Think of how much you can learn about a person just from those two things. What, for example, does it tell you about someone if they have a "Bush/Cheney" sticker on their car vs. a Peta decal? What does it tell you if they're wearing a Betty Rides snowboard shirt vs. a "No I won't fix your computer" shirt from Think Geek?

If you don't have a t-shirt for your product, service, or cause...*get busy*. And with Cafe Press, there's no excuse. It costs *nothing*. It's not the best quality, but it's a start.

And on that note, I'm horrified to realize that *I* haven't updated my cafepress site in years, and haven't put up a single thing on passionate users. Bad, bad Kathy. So... I'm going to spend the next day recrafting my cafe press store to have some of the artwork and cartoons from this blog.

Bonus dating tip: want to get to know someone? Don't just check out their bookshelf and iPod playlist. *Check their drawers*.

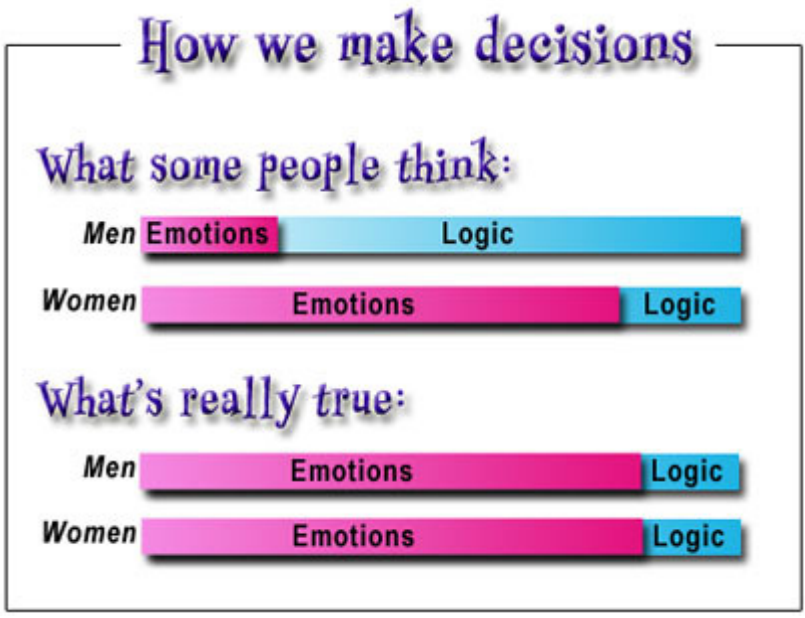
Boxers vs. briefs, cotton vs. silk, garters vs. no garters can only tell you so much. It's the *t-shirts* that reveal the soul. So, what are *you* wearing right now?

[Disclaimer for the cynical--this post is partly tongue-in-cheek. But you'll have to guess which part.]

http://headrush.typepad.com/creating_passionate_users/2005/07/tshirtfirst_dev.html

You're emotional. Deal with it.

By Kathy Sierra on July 5, 2005



No matter how enlightened and politically correct we've become, most people still tend to believe that when making decisions, men are less driven by emotions than women. Men use *left-brained* (metaphorically speaking) logical, rational thought. Men are persuaded to buy or act based on *thinking*, not *feeling*, while women do the opposite. You know, that whole Mars and Venus thing.

This wouldn't be so bad if those *left-brained* characteristics weren't seen as being more... *virtuous*.

Newsflash: emotions are sick and tired of being treated like second-class brain citizens! They're taking back their rightful place in the world, thanks to the work of brave neuroscientists like Joseph LeDoux and Antonio Damsio (author of Descarte's Error). These two, and a handful of others, withstood the mocking of their peers ("Wait... let me get this straight...you're basing your career on studying *emotions* [laughs hysterically, spits coffee out of nose]. That is hilarious! Oh, Antonio, you almost got me on that one... ha-ha-ha." But thanks to their strength of character, and scientific abilities, they've finally

started to offer honest-to-goodness, left-brain-compatible, bonafide *scientific evidence* of how crucial--and pervasive--emotions are in our lives.

You're *all* making decisions emotionally. You can deny it all you want, but you should be grateful for emotions. Without them, you'd remember almost nothing. Without them, you wouldn't learn much. Without them... *you'd probably be dead.* (And not much fun at parties or, for that matter, *in bed ;)*)

The key points for learning and marketing and creating passionate users is to keep this in mind:

People don't choose rationally to listen to your message and then have a feeling about it. They choose to listen to your message because they have a feeling about it.

If you're basing your communications solely on logical, rational, reasoned facts... the brain is not your friend. Emotions are the gatekeeper... if you want in, you gotta talk to the amygdala.

This doesn't mean that reason isn't crucial. In my little bar charts, logic is still there. You make a decision emotionally, but part of that decision is based on *using logic to figure out how you'll feel in the future about your decision.* In other words, you'll use logical thinking to predict whether you'll continue to feel good about the decision, or whether in the end... the guilt will be too much. Or that it's not worth the arguments you'll have with your spouse over it. You know the story.

And yeah, I've way over-exaggerated the bar chart to get your attention. In truth, when emotions or logic are not in balance, bad things happen. But we've spent the last several decades putting logic on a pedestal while poor emotions get kicked around and denigrated. In the end, guys, you're just as driven by emotions as women. Trust us... testosterone SO does not enhance your powers of reason. True, we women often show it differently... and certainly more freely than the average male. We don't have as much to prove there, and *we* always knew that emotions would one day gain the street cred they so richly deserve.

We've just been waiting for the neuroscience to catch up.

For more on emotions, check out the links above on LeDoux and Demasio, and don't forget Dan Pink's book A Whole New Mind, on why you must not only embrace your inner "right brain"

attributes, but work and learn to enhance them. Or face being outsourced, automated, or something else bad I can't remember.

Sooner or later, guys, you'll have to learn to cope with the knowledge that you're not nearly as rational as you thought. But I bet if you look back at the last big purchase you made, you'll know in your heart of hearts that no matter how good it looked on paper... you bought it because of how it made you feel. Deal with it. :)

http://headrush.typepad.com/creating_passionate_users/2005/07/youre_emotional.html

I know something you don't

By Kathy Sierra on July 6, 2005

Almost everybody loves to be the first to know something... or rather the first to *reveal* it to others. Whether it's a clever hack, a little-known easter egg, or a juicy bit of insider gossip. And nowhere is this more obvious than with passionate fans.

Last week I was talking to a store clerk with Pink Floyd playing in the background. Somehow we got to talking about the whole Dark Side of the Moon/Wizard of Oz sync thing, when he quite proudly (and a little conspiratorially) revealed the lesser known Echoes/2001 synchronization. For this guy bagging my carrots, it was a minor "I Rule!" moment.

He knew something interesting that I didn't. More importantly, it was something that promoted his Pink Floyd/rock fan status. He got whuffie for being the One Who Knew. (Don't know what "whuffie" is? Good. Because that means *I* get whuffie for being

"hip" enough to know it before you. ;) [I'm kidding --> note the winkie.]

In my t-shirt post I said:

Where there is passion, there are t-shirts.

Let me update that a little:
Where there is passion, there are t-shirts with sayings or symbols only a true insider understands.

I know guys who wear t-shirts with obscure references as a kind of "test" to see who belongs in their social circle. One male friend of mine said that if a woman ever recognized what his home-made t-shirt says, he'll know he's found the woman of his dreams. (It's some very subtle suggestion of an old Monty Python sketch). He's still single...

Look at your product, service, business, cause. When we reverse-engineer passion, we virtually *always* find secrets, legends, trivia, etc. that only insiders know. We virtually always find a custom and continually evolving lexicon that helps separate the newbies from the serious.

If you don't have anything like that... get started. Ideally, your passionate users/fans will take over creating and propogating some of this. But since we're reverse-engineering passion here, to try to jumpstart things--make sure you have memes worth spreading! If you're the owner, founder, designer, lead singer, whatever... surely there's something interesting in your background. If you're the marketer, *find* something.

If you're sure there's honestly *nothing* the least bit interesting, scandalous, clever, or funny, make something! (But please don't make s*** up! Not today, when truth isn't as highly-valued as one might hope). In other words, have something worth discovering. Worth hunting for. Something a guy (or gal) could get whuffie for being the first to reveal at a cocktail party or user group.

Obviously not all insider knowledge is equal. A sex scandal involving the previous CTO probably isn't worth as much long-term value as the story about the user who -- through your product -- saved the lives of seven baby dolphins. If you don't have legends in your business, try to find some. Try to help encourage them. Your users are your best source of fascinating,

memorable, amazing stories, but you'll never know unless you have a clear strategy for finding and capturing those stories.

Are you asking for user stories? Are you propogating stories? Are you embedding "secrets" that only the hard-core will discover? Easter eggs that *everyone* knows don't count for nearly as much as the stuff that's higher up the hard-core passionate users scale.

And if you don't know about the whole Scoble/cheerleader/Tom Cruise thing, then you're *obviously* not one of the true A-list insiders.

;)

http://headrush.typepad.com/creating_passionate_users/2005/07/i_know_somethin.html

Ten Tips for New Trainers/Teachers

By Kathy Sierra on July 11, 2005

Just because you've used lots of software doesn't mean you can write code. Just because you've been in lots of buildings doesn't mean you can be an architect. And just because you've logged a million frequent flyer miles doesn't mean you can fly a plane.

But if that's all ridiculously obvious, why do some people believe that just because they've *taken classes*, they can *teach?* (Or just because they've read lots of books, they can write one?) The problem isn't thinking that they can do it, the problem is thinking they can do it *without having to learn, study, or practice.*

I'm amazed (and more than a little disheartened) how many people believe that simply by virtue of their being skilled and knowledgeable in something, they're implicitly qualified to communicate, mentor, teach, or train that thing. It devalues the art of teaching to think that because you've been a student, you

can teach well. That because you've experienced learning, you can craft a learning experience.

But with that out of the way, nobody needs a PhD (or in most cases -- any degree at all) in education or learning theory to be a good teacher. Just as there are plenty of great software developers and programmers without a CompSci degree. People *can* be self-taught, and do a fabulous job, for a fraction of the cost of a formal education, but they have to be motivated and they have to appreciate why it's important. The irony is that most people with this attitude would themselves be insulted if the tables were turned--if *their* students didn't think they needed to learn anything from them... *that just going on instinct and winging it would be enough.*

So this is my starter list for new trainers and teachers (I won't debate any distinctions between "teaching" and "training"--we're talking about one who designs and/or delivers learning experiences, so I don't care what you call it, what your subject is, or even how old your learners are. The fundamentals of how humans learn are pretty constant, even if the application of those fundamentals can look quite different on the surface).

There are two different lists here--Eleven Things to Know, and Ten Tips for New Trainers. This is for newbies, so I'm sure I have nothing new to say for those of you who are already experienced teachers/trainers.

(A list of reference links is at the very bottom of the post. These aren't anything more than an off-the-top-of-my-head list, so please don't think of them as The Complete Story! And yes, I'm way overgeneralizing, or this would be book-length.)

Eleven Things to Know

1) Know the difference between "listening" and "learning".

Listening is passive. It is the lowest, least-efficient, least-effective form of learning. That means lectures are the lowest, least-efficient, least-effective form of learning. Listening alone requires very little brain effort on the learner's part (and that goes for reading lecture-like texts as well), so listening to learn is often like watching someone lift weights in order to get in shape.

2) Know how the brain makes decisions about what to pay attention to, and what to remember.

And here we are back to emotions again. Emotions provide the *metadata* for a memory. They're the tags that determine how important this memory is, whether it's worth saving, and the bit depth (metaphorically) of the memory. People remember what they *feel* far more than what they hear or see that's emotionally empty.

3) Know *how* to apply what you learned in #2. In other words, know how to get your learners to *feel*.

I'll look at this in the Ten Tips list.

4) Know the wide variety of learning styles, and how to incorporate as many as possible into your learning experience.

And no, we're not talking about sorting learners into separate categories like "He's a Visual Learner while Jim is an Auditory learner.", or "He learns best through examples." *Every* sighted person is a "visual learner", and *everyone* learns through examples. And through step-by-step instructions. And through high-level "forest" views. And through low-level "tree" views. Everyone learns top-down and bottom-up. Everyone learns from pictures, explanations, and examples. This doesn't mean that certain people don't have certain brain-style preferences, but the more styles you load into *any* learning experience, the better the learning is for everyone--*regardless* of their individual preferences.

(And while you're at it, know that *most* adults today do not truly *know* their own learning styles, or even how to learn. The word "metacognition" doesn't appear in most US educational institutions.)

5) Know the fundamentals of *current* learning theory!

(Check out the book links at the end of this post.)

6) Know why--and how--good *advertising* works.

It'll help you figure out #3. Be sure you recognize *why* this matters.

7) Know why--and how--good *stories* work.

Consider the learner to be on a kind of hero's journey. If Frodo is your student, and you're Gandalf... learn as much as you can about storytelling and entertainment. Learn what screenwriters and novelists learn. Know what "show don't tell" really means, and understand how to apply it to learning.

Humans spent thousands upon thousands of years developing/evolving the ability to learn through stories. Our brains are tuned for it. Our brains are *not* tuned for sitting in a classroom listening passively to a lecture of facts, or reading pages of text facts. Somehow we manage to learn in *spite* of the poor learning delivery most of us get in traditional schools and training programs (and books).

8) Know a little something about "the Socratic method". Know why it's far more important that you ask the good questions rather than supply all the answers.

9) Know why people often learn more from seeing the *wrong* thing than they do from seeing the *right* thing. Know why the brain spends far less time processing things that meet expectations, than it does on things that *don't*.

10) Know why it's just as important to study and keep up your *teaching* skills as it is to keep up your other professional skills. Yes there ARE professional organizations for trainers, with conferences, journals, and online discussions.

11) Know why using overhead slides to deliver a classroom learning experience can--sometimes (often)--be the *worst* thing you can do.

(Although yes, in many cases using slides for *some select pieces* of a course are important, beneficial, and crucial. What we're dissing is the practice where the entire class, start to finish, is driven around some kind of slides or presentation.)

12) Know how -- and why -- good games can keep people involved and engaged for *hours*. Learn how to develop activities that lead to a Flow State.

Ten Tips for New Trainers

1) Keep lecture to the absolute minimum.

There is nearly (but not always) something *better* than lecture, if learning is the goal. If your class involves a combination of lecture and labs, then if you're short on time--always cut the *lecture*, not the exercises! (Unfortunately, this is the opposite of what most trainers do.)

2) It is almost always far more important that your learners *nail* fewer subjects than be "exposed" to a wider range of subjects.

In most cases, it's far more important that your students leave able to DO something with their new knowledge and skills, than that they leave simply KNOWING more. **Most classroom-based instruction can be *dramatically* improved by reducing the amount of content!**. Give them the skills to be able to continue learning on their own, rather than trying to shove more content down their throats.

If your students leave feeling like they truly learned -- like they seriously kick ass because they can actually *do* something useful and interesting, they'll forgive you (and usually *thank* you) for not "covering all the material". The trainers that get cricism for not covering enough topics or "finishing the course topics" are the ones who didn't deliver a good experience with what they *did* cover.

3) For classroom trainers, the greatest challenge you have is managing multiple skill and knowledge levels in the same classroom! Be prepared to deal with it.

The worst thing you can do is simply pick a specific (and usually narrow) skill/knowledge level and teach to that, ignoring the unique needs of those who are slower or more advanced. And don't use the excuse that "if they don't have the prereqs, they shouldn't be here." Even among those who meet the formal prereq requirements, you can have drastically different levels. *Especially* if the teacher who delivered those prereq courses was in the "covering the material" mode. Sure, your students may have been "exposed" to the prereq material, but just because they heard it or read it does not mean they remember it now, or that they ever really "got it."

Techniques for dealing with multiple levels:

* Be sure you KNOW what you've got. Find out before the class, if you can, by speaking with the students or at least exchanging

emails. If you don't have access to students prior to the class, then learn as much as you can during introductions!

* Acknowledge the different levels right up front. The more advanced students are far more likely to get pissed off when they think you don't even *realize* or appreciate their level. By acknowledging it, you recognize their abilities and set the stage for having them act as mentors to the others.

* Have multiple versions of exercises! Have a "base" level of lab activities that everyone must complete, but have additional interesting, challenging options so that your advanced people aren't growing bored or frustrated waiting for the slower people to finish their exercises.

* For slower people, include *graduated hint sheets* for exercises. (More on that in the next point.)

4) Work hard to get *everyone* to complete the lab exercises, but NEVER give out the solutions in advance!

This is closely related to #3, because the most likely reason trainers don't have all students finishing labs is because there are some slower learners (and I don't mean "dumber", but simply less knowledgeable or experienced in the topic than the other students, or they just have a learning style that requires more time).

Be sure every students has been successful at the exercises! And if you give them the solution in advance, you've robbed them of the chance to seriously kick ass by working through it even when things get difficult. On the other hand, you don't want students to become completely stuck and frustrated, so use something like the technique below:

Using **graduated hints** can work wonders. Prepare three or more levels of hint sheets for the exercises, with each level more explicit than the last. The first level can offer vague suggestions, the second can be a little more focused, and the third can be fairly explicit. Students should be allowed to use these at their discretion, so it's best if you don't force the students to go to you for each new level. Make them available, but make it clear that it's important they turn to them only after [insert number of minutes relevant to your exercise].

After teaching literally thousands of programming and other courses, I can say with certainty that **the vast majority of your students will NOT simply go to the most explicit hints right off**. But this is conditional... I'm assuming that the exercise is relevant and interesting and challenging without being ridiculously advanced or clearly takes more time to complete than you're able or willing to allow for the exercise. If your exercises suck, for whatever reason, then hint sheets won't fix it.

5) Do group exercises whenever possible, no matter what you've heard.

I've heard every excuse, "Adults don't like to do group exercises." or "Professional developers don't like to do group exercises." or "People don't like to do group exercises when they're paying big bucks to be here." or "People from outside the US don't like to do group exercises... ". They're all bulls***. There is a huge social component to learning, regardless of how much we try to eliminate it in the classroom. There's a way to do interactive group exercises that works surprisingly well, and is usually quite easy.

A simple formula for group exercises

* Use groups of no more than 3 to 5. Try to go above 2, but after 5 you'll end up with some people hanging back. With 3-4 people, everyone feels more obligated to participate and be involved.

* When you assign an exercise (like, say, a two-page diagram of an enterprise architecture that they must label and explain), have each person START by working individually for a couple of minutes, THEN get them into their groups (be sure that they know who their group is BEFORE they start any work on the exercise).

* Eavesdrop on the groups and comment or just make sure they're on the right track. Drop hints or give pointers if they're veering into an unproductive approach.

* After a certain number of minutes, give a heads-up warning "60 seconds left..." so they can finish up.

* Be certain that someone in each group has the responsibility to record what the group comes up with. One person should be the designated spokesperson.

* After the exercise is done, keep the people in their groups and query each group about their answers, or any issues/thoughts they had while doing it.

Note: the first few times you do this in any new classroom, students might be quiet or skeptical about doing it, but after the first two or three, they'll have a hard time imagining how you could do it any other way.

6) Designing exercises

The best execises include an element of surprise and failure. The worst exercises are those where you spend 45 minutes explaining exactly how something works, and then have them duplicate everything you just said. Yes, that does provide *practice*, but it's weak. If you design an exercise that produces unexpected results... something that intuitively feels like it should work, but then does something different or wrong -- they'll remember that FAR more than they'll remember the, "yes, it did just what she said it would do" experience.

Note that paper and pencil exercises are GREAT. Even if your teaching programming or any other topic that involves *doing*. In our books, for example, we have simple "magnetic poetry" code exercises that don't involve everyone having to go to the computer. You can design even simple multiple-choice quizzes, although the more sophisticated the better. Be creative with creating workbook style exercises when you're teaching challenging subjects. In a programming class, for example, I'll have paper exercises (that they do both individually and in a group) that involve everything from, "fill in the rest of this class diagram with what you think should be there" to "fill in each empty method on this sheet with bullet points or pseudo code for what you think should happen there."

Depending on the classroom, you could even have an exercise that involves one group "teaching" something to another group. Assign group A to figure out the File API, for example, while group B has to research how and why the Serialization mechanism works the way it does in the lab you just did...

As hokey as they are, sometimes game-show style quizzes can still be fun. Especially when there's a set of topics that DO require boring, rote memorization. When they have to burn in certain key facts... you can liven it up and make it a little less painful.

The exercises in our Head First books (especially HF Java) are examples of paper execises we do in classrooms, that are separate from hands-on programming "lab" exercises.

The best form of longer lab exercises get learners in the flow state! This is where your game design studies can really come in handy. Remember, the flow state comes from activities that are both challenging but perceived as do-able. Get the challenge level right! Having multiple levels of hints means that a single exercise can work for a wider range of skill and knowledge levels without being too easy or too hard -- both of which will prevent the flow state.

Exercises should feel relevant! They should not feel like busy work or *strictly* practice (although for some kinds of learning, extra practice is exactly what you need, but in most cases -- you're looking to increase understanding and memory rather than simply practice a physical skill).

If students don't get the point of the exercise, you're screwed. It's up to you to either have an exercise where the point is dead-obvious, or that you can make a case for. The exercise does NOT need to be "real world" in the sense of the actual, complex world you live in. It should, however, reflect a simplified *virtual* world with its own set of rules. In a learning experience, you're usually trying to help them learn/get/remember only a single concept at a time. Way too many lab exercises that attempt to be "real world" have so much cognitive overhead that the real point you're trying to reinforce is lost.

7) Leave your ego at the door. This is not about you.

Your learners do NOT care about how much you know, how smart you are, or what you've done. Aside from a baseline level of credibility, it's far more important that you care about how smart THEY are, what THEY know (and will know, thanks to this learning experience) and what THEY have done. I'm amazed (and horrified) by how many instructors don't ever seem to get to know anything about their students. You should know far more about them than they know about you.

At the beginning of class, you do NOT need to establish credibility. You nearly always have a certain amount of credibility in the bank, even if they've never heard of you. You can LOSE that credibility by doing things like lying (answering a question that you really aren't certain about, without admitting

that you're not sure), or telling them you really DON'T know what you're doing. But you'll usually *hurt* the class if you spend time talking about how great YOU are.

The best way to let them know what you've done is in the context of a question someone asks, where you simply say, "Well here's how I solved that on an accounts database I was working on at...." But even better if you say something like, "Well here's how one of my clients/students/wo-workers solved it..."

8) Have a Quick Start and a Big Finish.

Get them doing something interesting -- even if it's just a group discussion -- very early. Don't bog them down with YOUR long introduction, the history of the topic, etc. The faster they're engaged, the better.

Don't let the class fizzle out at the end. Try to end on a high. It's like the movies... where they usually put the *best* song at the very end, during the closing credits... because this often determines the feeling you leave with. Ask yourself, "what were my students feeling when they left?" Too often, the answer to that is, "overwhelmed, and stupid for not keeping up". And usually, the fault is in a course that tried to do too much. That tried to *cover* (whatever the hell that means) too much.

9) Try never to talk more than 10-15 minutes without doing something interactive. And saying, "Any questions?" does *not* count as interaction!

Whether it's a group exercise, a lab, or at least an individual paper and pencil exercise of some sort... get them *doing* rather than *listening*. But be sure that the interaction isn't perceived as a waste of time, either.

10) Don't assume that just because you said it, they got it. And don't assume that just because you said it five minutes ago, they remember it now.

In other words, don't be afraid to be redundant. That doesn't mean *repeating* the same material over and over... but it often takes between 3 to 5 repeated exposures to something before the brain will remember it, so take the extra time to reinforce earlier topics in the context of the new things you're talking about. Great teachers know how to slip in the redundancy in an almost stealth way... where the thing is looked at again but from a different angle. It's up to you to keep it interesting and lively.

11) If you're not passionate, don't expect any energy from your learners.

That doesn't mean being an annoying cheerleader. Be honest, be authentic, but *be passionate*. It's your job as a trainer to find ways to keep yourself motivated. A lot of teachers/trainers feel it isn't their job to motivate the students. But that's ridiculous. Even the most motivated person in the world still finds it hard to stay motivated on each and every topic... especially when it gets tough. Think about how many technical books you've sat down to read on topics you were *extremely* interested in, but then couldn't find a way to keep yourself reading. Motivation for the overall topic and motivation for the individual *thing being learned* are completely different. You're there to supply the motivation for the individual things you're trying to help them learn.

Your passion will keep them awake. Your passion will be infectious. It's up to you to figure out how to stay passionate, or quit teaching until you get it back.

And finally, don't think of yourself as a teacher or trainer... since that puts the focus on what YOU do. Remember:

It's not about what YOU do... it's about how your learners feel about what THEY can do as a result of the learning experience you created and helped to deliver.

Rather than think of yourself as a teacher or trainer, try getting used to thinking of yourself as "a person who creates learning experiences... a person who helps others learn." In other words, put a lot *more* emphasis on the *learning* and a lot *less* emphasis on the *teaching*.

http://headrush.typepad.com/creating_passionate_users/2005/07/ten_tips_for_ne.html

What can software learn from kung fu?

By Kathy Sierra on July 19, 2005

What do Photoshop, martial arts, church, the military, accounting software, Star Trek, video games, digital video, web programming, online forums, chess, and cooking have in common? ***The Next Level.*** There's always something new to aim for and as you progress through each level, the motivation to go higher keeps growing. How many of you have felt the seduction--where you go into something thinking you'll never care about anything beyond the bare minimum entry-level, only to find yourself sucked in?

Next thing you know, it turns out you *did* want to learn CSS. Because once you know CSS, *then* you can do... (and on it goes). Turns out you *did* need something beyond what iMovie could do, so you just *had* to get Final Cut Express. Turns out you *did* want to earn the rank of "bartender"-- full forum moderator status on javaranch. Turns out you *did* decide to go for your SCWCD certification in Java. And why *not* get a brown belt?

Where there is passion, there is always the idea of a "next level".

The next level doesn't have to be explicit, like belt levels in martial arts, the specifically numbered levels in a video game, or a military rank. Sometimes the next level is simply a new, more advanced capability. The key point, though, is that even if the next level is implicit, *everyone recognizes it*. Or at least everyone involved in that activity. If you're at a Star Trek convention and the guy behind you in line starts speaking conversational Klingon, that *says* something. For that audience--the hard core trekkies--this guy has achieved an implicit high level of trekness. (Not that I'd know ;)

Even with something as seemingly mundane at work, you see it. The one woman in the office who truly "gets" tables in MS Word. Although she might have reached table mastery status simply because she was *forced* to, more often it was because she started down that path and found herself hooked on learning just a little more.

No matter what the job task, the feeling is something like this: "If I could just do [insert some capability just slightly beyond what you know now], then I'd be able to do this one cool thing." And just as with any video game, once you've got *that* new "superpower", the next natural desire is to learn the *next* thing... If you can find a way to give your users something to reach for... that next level... in terms of new capabilities that allow them to do still cooler things, you have a much greater chance of inspiring passion. Because reaching for that next level is what leads to greater engagement, and improves the chance of having users stay in flow (FYI: the August issue of Fast Company has a nice little article on Flow! It's not online yet; they still show July as the current issue.)

It's all about kicking ass.

Of course, some companies do exactly the wrong thing by making what should be, say, a level ***2*** task feel like a level ***8***. In other words, you shouldn't have to feel like you must "get to the next level" to do the most basic thing. The point of the next level concept is that users should feel like it's *worth the effort to get there*. That it's challenging, but for all the right reasons. That the **new cool thing** they'll be able to do justifies the time and energy spent learning, researching, practicing. So the featuritis vs. the happy user peak plays a role here.

Remember, learning is like a drug to the brain (actually, it *is* a drug). The best user experiences--combined with a clear path to greater expertise and the promise of more time in flow--are like a healthier, happier form of crack. One of the best examples of this drug-dealer model in software is Apple.

With iMovie, for example, *the first one is free*. But once you're hooked, you find yourself wanting capabilities found only in the $299 Final Cut Express. You find yourself wanting, no *needing* to do things you never even imagined before you started playing around with iMovie. And for a certain percentage of users, even Final Cut Express will have limitations. Now you need the $999 Final Cut Pro or--for just a few dollars more, what the heck--might as well go for the whole Final Cut Studio. They've managed to teach you to want the most expensive versions of their products. Then they do the same thing with sound (Garage Band --> Logic Express --> Logic Pro). It seems Apple has figured out the optimum price points for their "next levels", in order of FREE, $299, then $999.

But even if the goal is not to teach or inspire users to appreciate your higher-end products, just having goals to strive for is what matters. Whether the promise is that you can become a first-level moderator, a church usher, one who can use the RAW features of Photoshop, a CSS guru, a Sun Certified Business Component Developer, a double black diamond snowboarder, or a 3-dan go player... never forget that where there is passion, there is *always* a next level.

Software--or any product--can learn a lot from the martial arts, and I suppose the idea of rankings/belts/levels is probably the least of it. But it's a great place to start.

So what's your next level? Do your users know what the levels are? Too often, users *could* get excited and motivated if only they knew more. If you hear a user say something like, "But I never you could do that!", consider that a problem. How many more people would have stuck around if they'd known? With your software, product, service, club, subject you teach, whatever... is there a steady series of new possibilities out there worth reaching for, and more importantly, are you doing something to help users get there?

http://headrush.typepad.com/creating_passionate_users/2005/07/what_can_softwa.html

When process goes bad

By Kathy Sierra on August 8, 2005

Imagine this scenario... you've discovered a way to add one thing to your product that will double its usability. Just like that. And it's not a big deal to add. Or so you thought...

You bring it to your manager who discusses it with other people (people you're not allowed to talk to directly... you know, chain-of-command and all), and the answer comes back, "No". Why? "It just won't work with Our Process."

The systems, policies, procedures aren't set up to incorporate your proposed change, and nobody's willing to think about changing things. It would just be too disruptive. **It would make too many people *uncomfortable*.**

And we wouldn't want that.

Obviously there are times it doesn't make sense to shake up your hard-fought, well-tuned systems. Where the tradeoff doesn't justify it. But you all know what I'm talking about here... the times when NOT changing makes no sense. Or at least it makes no sense to those *outside*. Those with a perspective not colored by inertia, bruised from past experience, or threatened by new (and potentially better) views.

Too many times I've heard "upper management" assume that when employees (or users) insist that what the company is doing makes no sense (e.g. a policy that punishes customers or pisses off employees), it must be because the employee *just doesn't get it*. The employee doesn't have all the facts and doesn't see things from the "higher" perspective of management. The employee doesn't see the Big Picture.

Sometimes... sometimes that's bullshit.

Sometimes the employee or user is the only one who DOES "get it". Sometimes it's the lower-level (or at least more user-facing) employee who *really* knows how damaging a company's policies can be, or where the points of leverage really are. Sometimes it's the user who has a basis of comparison -- who hasn't bought into the company's worldview so long that they can't see any other reality.

I don't want to be too specific with names, but here are some examples I've experienced recently:

* A particular mechanism for annotating code in a book or manual would dramatically improve usability for the reader/learner. *But the documentation department can't do it because their ancient desktop publishing system won't support two "layers" intersecting*. Usability (and even innovation) takes a back seat to an old production system for which *many* cost-effective alternatives exist, but... it would still mean change and learning curves and *discomfort* amongst the production staff.

* A large software company decides to go after one of its *biggest* outside evangelists because the big software company has a policy that says "if anyone is going to make money from training on our products, it must be us." Stupid. Stupid. Stupid. (And from a systems persepctive, pretty much the worst thing you can do.)

* A bestselling author has found a way to reduce technical errors in first printings by as much as 70%, at *no* cost to the publisher

using a peer review system that involves volunteers. *But the publisher's copy editing process cannot cope with the shift in timing and granularity of copy edits* (despite the fact that the average technical book reader considers tech errors about 10x worse than grammatical errors). [FYI - I'm not talking about O'Reilly or Osborne]

* A fast food company demands that all clerks MUST "upsell" the customer, regardless of the customer's order. That means even when the customer *does* order a drink and fries to go with their meal, the order clerk is *required* to ask the customer if they'd like some other [insert specific thing] to go with it. That demans the employee and pisses off the customer, but The Policies leave no room for judgement calls about when it is or is not appropriate to upsell. It's always, end of story. Besides, it's not like a fast-food clerk has enough of a brain to make a good decision about that *anyway*. Right?

* A large software company insists that the documentation team not use contractions in their writing because "they don't translate well." So, they suck the life out of the user documentation to compensate for the poor quality of translators they use.

* A word processor that insists on capitalizing the word boolean as Boolean. Or that insists the code line:

```
public static void main (String[] args) { }
```

is actually:

```
Public static void main (String[] args) { }
```

* A software company's editorial staff that has all the control with none of the technical knowledge... who takes a book on web technologies and manages to turn HTTP POST into HTTP Power On Self Test. And who thinks that a database that's *not normalized* is actually... a *somewhat unusual* database". (Yes, that's a true story.)

* Don't get me started on all the stupid customer service related policies companies have that make no sense (well, at least not if they ever want those customers to come back) for things like returns, refunds, repairs, etc.

Gosh, I guess I decided to return to blog world with a little rant :)

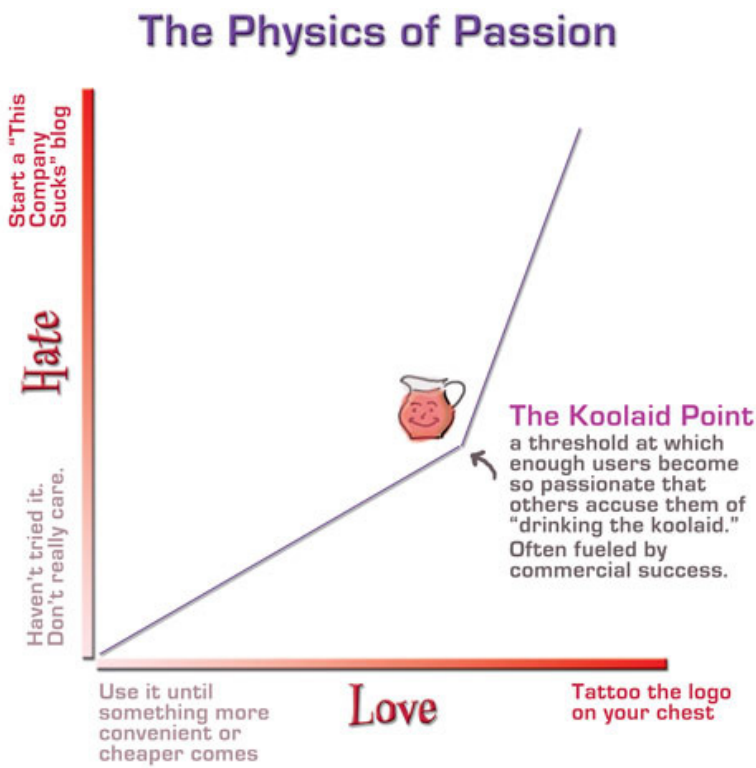
So... think about your policies and systems and procedures and process. I have this terrible fear that I'm going to be doing the same thing -- justifying staying with a production process -- even when it would be better for the user (or the author) to allow for more creativity, flexibility, and *change*. If today's business mantra is "change or die", we should all be looking for ways to make sure we don't fall asleep in the comfort of our working systems. And boy do I know how seductive those comfort zones can be...

As Jayne Howard (and others) have said,***"If you are not living on the edge, you are taking up too much room."***

http://headrush.typepad.com/creating_passionate_users/2005/08/when_process_go.html

Physics of Passion: The Koolaid Point

By Kathy Sierra on August 10, 2005



You don't *really* have passionate users until someone starts accusing them of "drinking the koolaid." You might have happy users, even *loyal* users, but it's the truly passionate that piss off others enough to motivate them to say something. Where there is passion, there is *always* anti-passion... or rather *passion in the hate dimension*.

If you create passionate users, you have to *expect* passionate detractors. You should *welcome* their appearance in blogs, forums, and user groups. It means you've arrived. Forget the *tipping* point--if you want to measure passion, look for the *koolaid* point.

And it would appear that 37 Signals has hit it. Within 48 hours of one another, independently, three groups reviewed the company: this blog, Salon, and Paul Scrivens' blog. Two of the reviews glowed. The other... provided balance in the universe.

Remember folks, we aren't going for user *satisfaction*. We aren't going for *happy*. We're going for all-out **passion**. And that comes with a price tag. Detractors. Lots of them. And they *talk*. For every passionate user out evangelizing you to everyone they meet, a koolaid-hunter will do his (or her) best to make sure everyone knows that your passionate users have lost their minds. That they're victim of marketing hype. *Sheep*.

But consider this...

The most popular and well-loved companies, products, and causes have the strongest opponents.

You'll know when you get there, because the buzz goes from pleasant to polarized. Moderate, reasoned reviews and comments are replaced with stronger language and more colorful adjectives on both sides. Those who speak out against you will be referred to as "brave" or "having the balls" (see the comments on Scriven's review) for daring to criticize. They're hailed as the smart ones who finally call the emporer on his buck-nakedness.

Should you ignore the detractors? Diss them as nothing *but* evidence of your success? Should you just wave them off with a "just jealous" remark? Absolutely not. Somewhere in their complaints there are probably some good clues for things you can work on. But if you start trying to please them all or even worse, *turn them into fans*, that could mean death. Death by mediocrity, as you cater to everybody and inspire nobody.

It is physically impossible to have everyone love what you do. And the more people do love it, the more likely it is that you'll have an equal and opposite negative reaction. X = -Y the physics of passion.

Would you want to be in 37 Signals' shoes right now, taking all this heat? You bet. Look who's been there before:

* Apple (see the wonderful Cult of Mac blog)

* Extreme Programming (see Matt Stephen's Software Reality blog)

* The Sierra Club

* The Red Sox (see the Yankees Suck site)

* NASCAR (read instanpundit's notes)

* The Hummer (read the official F*** You and your H2 site)

* Britney Spears (see the I Hate Britney Spears site)

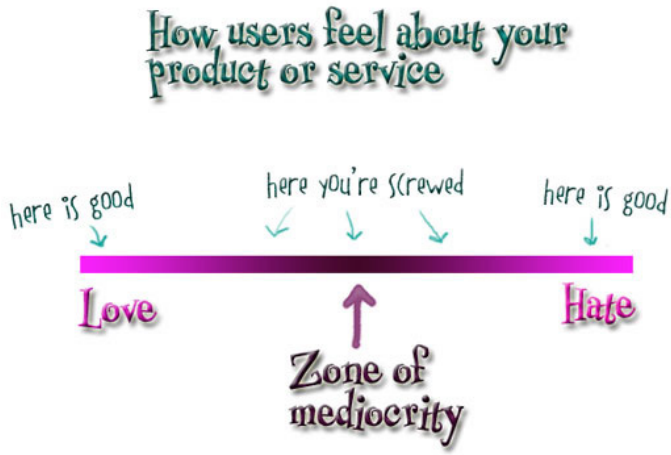
* Java (see the delightful No-one-cares-about-my-language-and-therefore-I-hate-Java note, or my special Java fan site, javaranch)

And on it goes.

Oh yeah, besides the "koolaid" word -- another word the detractors will use to marginalize something: "fad". As in, "Oh, that's just a fad. It'll be over soon." I remember hearing that in 1998 about Java, now the leading programming language. The iPod is a fad. Our Head First Java book was just a fad (yesterday on Amazon, out of *all 32,000* computer books, there were two Head First books in the top ten). Hip hop music was just a fad. Skateboarding. Snowboarding. *The web*.

So we'll see. But remember during those dark days when you're fending off the detractors (*especially* when they have legitmate complaints), that -- as Seth Godin tells us-- "Safe is risky and risky is safe."

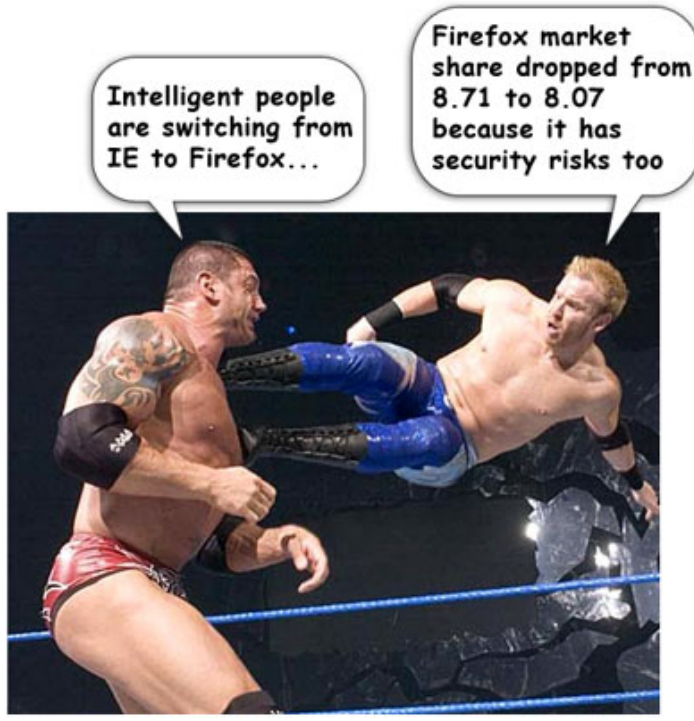
You'll never be perfect. Apple isn't perfect. Java isn't perfect. Our books are far from perfect. 37 Signals isn't perfect. **But you can be brave.**



http://headrush.typepad.com/creating_passionate_users/2005/08/physics_of_pass.html

The Smackdown Learning Model

By Kathy Sierra on August 15, 2005



What happens to your brain when you're forced to choose between two different--and potentially conficting--points of view? *Learning*. That's what makes the smackdown model such an effective approach to teaching, training, and most other forms of communication.

Whether you're writing user instructions, teaching a class, writing a non-fiction book, or giving a conference presentation, consider including at least some aspect of the smackdown model. It's one of the most engaging ways to cause people's brains to both feel *and* think -- the two elements you need for attention, understanding, retention, and recall.

How does it work?

By presenting different perspectives or views of the topic, the learner's brain is *forced* into making a decision about which one they most agree with. And as long as the learner is paying attention, you won't even have to *ask*. In other words, it doesn't

have to be a formal exercise where the learner must physically make a choice between multiple things; simply by giving their brain the conflicting message, *their brain has no choice*. Brains cannot simply leave the conflicts out there without at least *trying* to make an evaluation.

And making an evaluation puts it at the most advanced end of Bloom's Taxonomy. (The further along the hierarchy you go, the more cognitive brainpower is harnessed).

Why is this better than a single consistent message?

More brain flexing = more learning. (Yes, there's a big assumption here that the learner already understands the fundamentals behind the different viewpoints.)

When the learner is given a single message, and led through the topic step-by-step with no apparent alternatives, the learner's brain doesn't have to *think* as much. And since a single message is often less interesting, the material is less engaging and the learner isn't paying as much attention.

And the more *intense* the smackdown (i.e. the heat/fight of the opposing views) the more likely it is that the learner will *feel* something. And remember, we learn and remember that which we *feel*, not that which we merely *hear* or *read*.

But this is stupid... what about things for which there is only ONE right way?

Ah yes. Multiple points of view works great when it's a browser or web framework war, but what about something like the the speed of light? Or multiplication tables? 4 x 6 is 24. End of story. A fight over that would just be distracting and get in the way.

Right?

Maybe not. It's true that there are subjects for which there is no alternative point of view that makes any sense at all... so nothing to evaluate. But in that case, you can *still* use a smackdown approach by having the information *taught* from multiple perspectives. For example, if one teacher uses a rote approach, and another thinks an *understanding* approach is better, then one of the most powerful learning experiences would offer the learner *both*, with perhaps discussions amongst the learners over the relative tradeoffs of the two approaches. That means the smackdown isn't about the actual content being learned, but about *the way in which it's learned*. The more you get the

learners thinking (about the right things), the better the learning outcome.

How do you use it?

There are an infinite number of ways in which you could implement a smackdown, but here are a few favorites:

1) Presentation Smackdown

One of my favorite sessions at OSCON was Matt Raible's Spring vs. WebWork Smackdown. Two presenters, two frameworks, one guy with the big bell. The room was packed and *everybody* was paying attention. The presenters kept taking turns, and when a comment was deemed "below the belt" (a cheap shot), the bell guy kicked in.

Bonus benefit: this approach means you can get away with far fewer PowerPoint slides ;)

and you get a lot more audience participation.

Two other examples of a conference presentation smackdown are the Web Standards Smackdown and the JavaOne '04 Web Framework Smackdown. Another Web Frameworks Smackdown was held at JavaOne 2005, and discussed on this server side thread.

And Rick Ross considered a Java IDE smackdown in his Javalobby blog.

2) Head-to-head Review Smackdown

One example of a written comparison review is Ed Bott's TiVo vs. Windows Media Center smackdown. Another written example is a write-up that actually *captured* a conference panel smackdown on (this is old) J2EE vs. .NET.

Almost any decent and detailed multi-product or multi-perspective review can be considered a smackdown candidate, but *really*, if there's no heat and controversy and, well, *fighting*, then I wouldn't call it a smackdown.

What makes it an actual emotion-inducing learning experience is when the learner is at least *wondering* whether things could get a little rough. And in many cases, the rougher the better.

3) Anthropomorphized Debates

Head First readers might recognize this from some of our books. The implication of a smackdown is that two or more people are in a fight. A fight to win (even if no clear winner emerges). We make our smackdowns a little more personal in our books by breathing life into whatever it is being debated, and let that thing speak for itself. In Head First Java, for example, the compiler and the JVM argue over which of them is more important. Arrays go one-on-one with ArrayLists. And so on... with each "character" attacking or defending itself according to that character's personality and attributes.

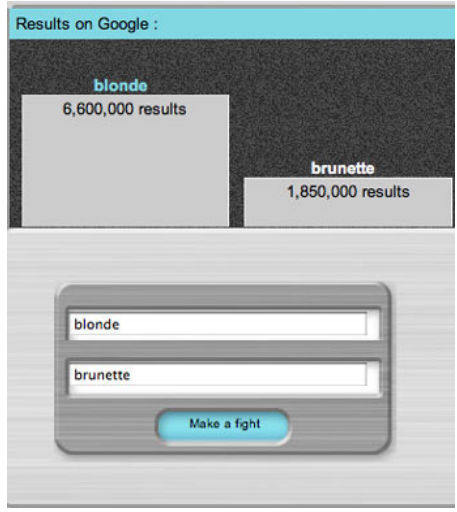
Another benefit of anthropomorhpizing the objects of the debate is that the learner can look at things from the perspective of that entity. We want learners, for example, to know *what the compiler's motivation is* (no actor jokes here ;). Who better to describe that than the compiler?

Variants

The Celebrity Death Match is quite popular. You could do your own version of this with a little Flash work.

And you'll just have to evaluate this one for yourself: Modified Living Sorority Smackdown.

Finally, if you haven't yet spent some time with Googlefight, here's a fun way to kill your productivity.



http://headrush.typepad.com/creating_passionate_users/2005/08/the_smackdown_l.html

Build something cool in 24 hours

By Kathy Sierra on August 24, 2005

The highlight of Foo Camp for me was hearing game development guru Squirrel Eiserloh talk on total immersion / ultra-rapid game development. I'm dying to try it for everything from creative writing to learning Flash to composing music to video/podcasting and of course game development. I cannot imagine a better, faster path to creativity, innovation, and most importantly *getting something done!*

The notion is this: stick people in a house for 48 hours, with a goal to have something created at the end. Depending on the nature of the goal, participants may be collaborating (like building a game together) or working alone (musicians composing, writers writing, etc.). The key is the process--a process that forces you to supress the "inner judges" that stifle creativity, and gives you not just permission but an *order* to create as much as possible, as fast as possible... even if what you end up with is 97% crap.

The point is to *learn* something valuable from the experience... something you'd likely never get to in your day job, even when--as it is for Squirrel and his game developer cohorts -- what you do in the jam *is* what you do in your day job. In other words, by

working under the ad-hoc/jam constraints, you're able to "improve your craft" and discover things about yourself and the work that you might never find in your traditional work environment. It takes the idea of rapid iterations to a completely different (dramatically compressed) time scale. What could take weeks, months, or years to evolve suddenly happens in hours. And the work never leaves your personal brain RAM! No more cost of switching contexts as you go from personal life to meetings to actual work to commuting to whatever... this is 100% being in the zone, where each hour spent in one of these jams is worth perhaps 10 or more hours at work in your usual environment.

The idea can be mapped to virtually anything for which you want to encourage maximum creativity, innovation, and most importantly... *getting something done*. While it may be a Big Deal to start your own Foo/Bar-style self-organizing conference, the total immersion "ad-lib jam" model is something we can all start in our home town, wherever that may be. All you need is a handful of participants (maybe 4-8), some delivery/take-out menus for chinese food and pizza (revise to reflect what goes for "fast delivered food" in your culture), maybe a few pillows and blankets, a whiteboard and some markers, and whatever other tools of the trade your participants need to *make* things.

(Sidebar: out of the 15 or so people at Squirrel's informal session, the most engaged participant was Amazon founder Jeff Bezos.)

For writers, that could mean laptops or even just paper and pencils. For programmers, that might mean the programming tools (game engine, compilers, source control if multiple participants are collaborating on the same app) or art tools (Photoshop, etc.). For music composition, that might mean real and virtual instruments (guitars, midi keyboards, synth guitars, mics) and sequencing software like GarageBand, Reason, or Logic. And for pure idea brainstorming, whiteboards, post-its, and big flip-chart pages to put stuff up on the walls as things progress.

I won't bother explaining *how* to run one of these things, because Squirrel and friends have already done this at:

The Ad-Lib Game Society site, which encourages others to start their own "lodges", like their founding chapter in Dallas, Zero Lodge. (He mentions that they've taken inspiration from the

earlier Immersion Composition Society, as well as the Indie Game Jam).

But does it have to be face-to-face?

This was a natural question. And the answer was... *probably*. A big part of what makes this work is that you are not in your normal environment. No kids, no chores, no I-should-be-doing-something-else. More importantly, ***it's the energy of the other participants that makes this so effective.*** You know exactly what I'm talking about if you've ever been in a highly engaged group where everyone's really cranking and you can almost feel the brain power and creativity rippling out of each person's head like Wi Fi.

Squirrel said that while they had tried a virtual jam, it wasn't that successful. One example he gave was that while at home you might hit the wall and give up (or get tired and go to sleep or do something else), when you start to hit that point during a live jam, all it takes is one guy walking by playing air guitar with his head phones on and you're suddenly hit with another wave of energy (or at least that little bit of competitiveness and pressure because you know you've got to demo something in three hours!)

The total immersion part of this is crucial, and until someone figures out a good way to make this happen remotely/virtually, face-to-face is probably going to be a lot more effective. (I have no doubt that there are ways to make this work remotely/virtually, but it would take some real effort and creativity to pull it off.)

Here are just a few of the ways in which I'd love to use this approach in my own life:

Storyboard Jam

We develop our books (Head First books, and the not-yet-announced new series we're working on) from storyboards, as opposed to outlines and TOCs. The storyboard is by far the most important part of the creative process for the books, and it's often the most difficult for authors... including those with tons of previous "traditional" writing experience. Having everybody go off and spend hours with their storyboards (either alone or in collaboration with another person), then periodically getting

back together for a show-and-tell with feedback would be amazing. In fact, we *did* do something like this once -- we called it a "Head First Bootcamp" -- that brought together a half-dozen prospective authors plus our O'Reilly editor, for five days in Colorado, all staying in one house, and with food brought in most of the time. One of the outcomes of this intensive week were Eric and Beth's storyboards for the Head First Design Patterns book, currently one of the top five bestselling computer books.

There's no doubt they would have produced these storyboards back in their own home, but this total immersion week *did* kick-start things in a big way, and gave them the opportunity for vital real-time feedback.

Learn Something New Jam

I've been trying to squeeze in some time to learn Flash, but each time I never get past the first few tutorials. *There's always something higher on the to-do list*. But if, say, 4-8 people got together, and we all had a sole task--to learn something new and then *create a demo* of what we learned at the end (with a checkpoint at the halfway mark), then it would give me the permission to just get in there and have Flash loaded into my brain, with the goal of creating a prototype of something I've been wanting to build. I honestly believe that if I don't do it this way, I simply may never get to it. I need someone to say, "You aren't allowed to do anything for the next 36 hours... no email, no going out to eat, no working on anything else."

Music Composition Jam

This one doesn't need explaining.

Write a [screenplay/article/chapter] Jam

Neither does this one.

Game Development Jam

I did a several year stint as a game developer, but have done virtually nothing since leaving that world to work at Sun (which, sadly, involved lots of enterprise development but NO games). I was thinking that the only way to get to work on games again was to work in the field, but that's ridiculous. There's no reason that me and six of my friends -- including coders and designers and maybe someone who understands audio -- couldn't get together and build a game. As Squirrel points out, we'd probably

learn more valuable lessons we could take back to our *real* work than with just about anything else we could do in that amount of time (including attending formal "training" classes).

Let's do it!

So... if you're in Colorado, anywhere around the Denver/Boulder area, please email me at headrush[at]wickedlysmart[dot]com, and let's start a new chapter/lodge of the Ad-Lib Game Development Society!

(And thanks Squirrel for such an inspiring lesson, and for putting up such great info on your site.)

And for everyone else, I urge you to study the info at the ALGDS site and consider starting your own in *your* area. And who knows... maybe we can attend jam sessions held by one another's lodges. I'd love to crash one of Squirrel's jams (I'll make the coffee!), and perhaps someone from out of town who wants to do a book could come to one of our book jams.

I'll say more over the next few days about lessons learned at Foo Camp, but this was by far my favorite, and the one I'm most likely to implement soon.

http://headrush.typepad.com/creating_passionate_users/2005/08/build_something.html

You ARE a marketer. Deal with it.

By Kathy Sierra on August 27, 2005

It's so trendy to diss marketing. Especially if you're in engineering, product design, or virtually anything *but* marketing. A comment for me by pinhut on my "You're emotional..." blog entry reads:

"this started out being so interesting. then you reveal yourself as a marketer. please terminate yourself."

The late (and brilliant) comedian Bill Hicks was an early adopter of the "all marketing is evil" meme:

"By the way, if anyone here is in advertising or marketing, kill yourself. No, this is not a joke: kill yourself . . . I know what the marketing people are thinking now too: 'Oh. He's going for that anti-marketing dollar. That's a good market.' Oh man, I

am not doing that, you fing evil scumbags."*** (asterisks are mine)

I was about to protest, "Dammit Jim, *I'm* a *programmer*, not a marketer!"

But that would be a lie. In this new open-source/cluetrain world, I *am* a marketer. And so are you. If you're interested in creating passionate users, or keeping your job, or breathing life into a startup, or getting others to contribute to your open source project, or getting your significant other to agree to the vacation *you* want to go on... congratulations. *You're in marketing*. Now go kill yourself.

The word "marketing" (and by extension, "marketers") has a bad rep for sure, as does "advertising" and "PR". But they all share a common goal--*connecting buyers and sellers*. Isn't that what we're doing?

Except with a **Find and Replace:**

"Buyers" becomes--> "readers" or "users" or "community participants"

"Sellers" becomes--> "authors" or "developers" or "organizations"

As Guy Kawasaki puts it, we're selling the dream.

But the difference between what we now consider "old-school marketing" (otherwise known as The Four P's -- product, price, promotion, and placement -- heavy on advertising and "branding") and the "neo-marketing" we're doing here is frickin' huge.

Here are a few ideas on some of the differences:

Old-school marketing	Neo-marketing
marketers/advertisers do it	*everyone* does it
focused on how the *company* kicks ass	focused on how the *user* kicks ass
marketers have the power	*users* have the power
advertising	evangelizing
tightly-controlled "brand message"	brand hijacked* by users
one-way broadcast	two-way conversation
company-created content	user-created content
he who out**spends**, wins	he who out**teaches**, wins
mass markets	selective, focused users
one-size-fits-all	personalized, custom-tailored
focus groups	user feedback & contributions... *betas*
deception	transparency
bulls***	authenticity
development often independent from marketing	impossible to separate development and marketing
the story must be compelling, but can be fiction ("buy this and you'll have more sex")	the story must be compelling, and must be real** ("buy this and you'll take better photos")
30-second spots are king	word-of-mouth is king
focus on branding	focus on passionate users
get the customer to believe in it	**YOU** believe in it

*See this Brandautopsy blog post on a brand hijack, or check out the book.

***Real* is relative to the desires and perceptions of the user. And who's to say that taking better photos won't in fact lead to more sex?

***rhymes with "hit"

But even if we feel OK about *doing* some of these marketingish things, there's still the problem of the word "marketing". We need a word that distinguishes the kinds of things we (developers/programmers, ministers, realtors, authors) do from old-school traditional marketing. I just don't know if the marketing-averse among us can rehabilitate that word... it's been too heavily associated (framed) with old-fashioned, negative, sleazy and inauthentic practices (even if much of that was a misconception... doesn't matter).

My "neo-marketing" label is just lame. Open Solaris' Laura Ramsey and I were talking about it this weekend, and she came up with an alternative that might be a good contender: Modern Attraction. We're not marketers, we're *attractors*. I don't know if that's the right phrase, but it still sounds better to me than "marketing". (Personally, I was voting for "cheerleader", but for some reason I just couldn't get the other programmers to go along with that...cute t-shirt ideas, though... ;)

Others have come up with replacement phrases as well, but none seem to have truly taken hold, and the word "conversation" isn't enough. What do *you* think? If we believe in something, and we want others to share what we know can be a fun/meaningful experience, whether it's getting involved in our open source project, or joining our cause, or--yes--buying our book or software--we need to get past our "go kill yourself now" thing. If framing it with a new word/phrase helps, perhaps that's a better approach than trying to give the word "marketing" a massive makeover.

Remember -- when people are passionate about something, and in a state of flow--and you have contributed to that by helping users/members learn and grow and kick ass--these are some of the happiest moments in their lives. Trying to promote more of that is something we should feel *wonderful* about, not guilty.

http://headrush.typepad.com/creating_passionate_users/2005/08/you_are_a_marke.html

Blow your own mind

By Kathy Sierra on August 30, 2005

If you want your brain to stay sharp, you have to work it. "I write complex code every day of the week. That's all the brain exercise I need." you say. *But you're wrong*. If you want to keep your brain alive, you have to do things your brain doesn't *expect*. The cortex forms new patterns... new synaptic connections in response to *novel* activity, and PET scans show that far fewer pathways are activated when the brain processes a routine task... even a complex one.

Imagine playing an electric guitar, for example, but *hearing* a saxaphone. That's what you get with a synth guitar (click on the video button to see and hear the guitar player). At the Telluride Bluegrass Festival, the Sam Bush band was playing a vintage rock song, complete with a vintage rock organ. But... *no keyboard player*. Then I realized it was one of the guitar players, on a synth guitar.

The cognitive disconnect between *watching* someone play the guitar but *hearing* an absolutely real smokey sax is... mind blowing. (FYI musicians, Roland and others have done a LOT of

work on the latency, and the latest generations of guitar synth/midi pickups is pretty damn good now.) Apparently *playing* a synth guitar is even stranger than watching/hearing someone else. I play a midi keyboard, so this shouldn't have been so surprising, but we *expect* keyboard synthesizers, so watching someone play a piano that sounds like a violin doesn't tweak neurons the same way anymore.

So what else can you do? The point is to do something *different*.

If you're into extreme sports, try a meditation retreat.

If you're someone who is *not* deaf, you might try attending a silent weekend, a total immersion "sign language jam" intended to give non-deaf people a brain-changing (and some say *life-changing*) experience. I've had friends attend one of these (they hold them in lots of different places) after taking some sign-language classes just for the unique experience--not because they *needed* to learn sign-language. They said it's extremely challenging... and very, very rewarding.

Or try Dark Dining, by visiting a restaurant virtually blind. They're popping up everywhere--London, Germany, Australia, and Switzerland, and Paris (google on "dining in the dark").

Or let's say you already tend toward being the quiet, chess-is-my-sport type. Then maybe you need something adventurous like, say, trek through Patagonia.

But you don't need to wait for some Big New Event to make your brian happy. It's more important to try to incorporate brain workouts into your everyday life. There's a fantastic book on all this by Lawrence C Katz, called Keep Your Brain Alive, be sure to check out this page on "neurobic exercises". But here are a few simple exercises from the book that you could do right now:

Turn your world upside down

"Turn pictures of your family, your deskclock, or an illustrated calendar upside down. Your brain is quite literally of two minds when it comes to processing visual information. The analytical "verbal" part of your brain (sometimes called "the left brain") tries to label on object after just a brief glance: "table", "chair", "child". The "right brain", in contrast, perceives spatial realtionships and uses nonverbal cues. When you look at a familiar picture right side up, your left brain quickly labels it an diverts your attention to other things. When the picture is

upside down, the quick labeling strategy doesn't work--and your right-brain networks kick in, trying to interpret the shapes, colrs, and relationships of a puzzling picture."

See things in a new light

"Place different-color geletain filters over your desk lamp. Colors evoke strong emotional associations that can create completely different feelings about ordinary objects and events. In addition, the occasionally odd effects of color (a purple styrofoam coffee cup) jars your brain's expectations and lights up more blips on your attentional "radar screen."

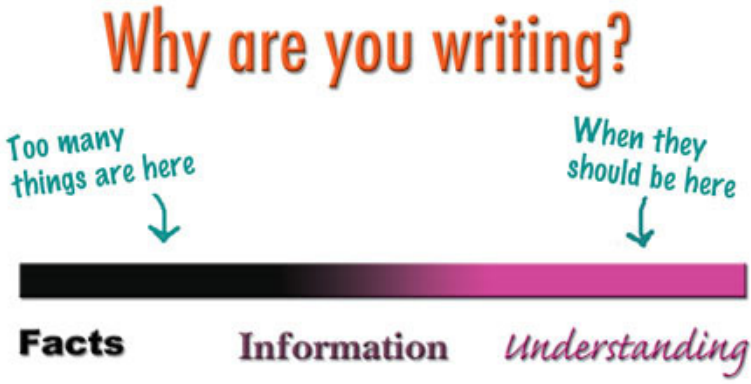
But the book has more than 80 others -- including some more ambitious activities to help blow your own mind. I tend to think that I'm doing a good job on this because I have a good mix of activities... programming, skateboarding, writing, reading weird science, skiing. But then I realized, these things aren't unique *for me*... they don't blow my mind (except for the weird science stuff). I've been doing them for a long time, so they don't tweak my brain. There's a lot of good stuff there, but no Whoa/WTF experience. I need to keep incorporating *new* things at every scale--from big macro things like adventure vacations to where I keep my paper clips.

So what are *you* doing to blow your own mind on a regular basis?

http://headrush.typepad.com/creating_passionate_users/2005/08/blow_your_own_m.html

Reference vs. Learning: pick ONE

By Kathy Sierra on September 4, 2005



The biggest problem with many tech books (and tech *training*) is indecision. When the author can't decide between creating a *reference* or *learning* book, they often try to do both. That rarely works. So the second most important question to answer when you start to write a tech book (or a book *proposal*), instructional manual, or training course is ***what's my goal?***

And the *most* important question is, "what's the *reader's* goal?"

When a non-fiction book (especially a tech book) doesn't sell, or a training course isn't successful, it's often because the reader was on one end of the graph while the writer/course developer was on the other. Or because the author/teacher believed that giving information *was* the way to communicate knowledge and understanding.

The difference between facts and information is straightforward: information *organizes* facts into a meaningful pattern. Without information, data and facts can be arbitrary and useless. There's a crucial place for reference information, and information architecture is art + science. (Two info architect bloggers are Louis Rosenfield and Jesse James Garrett, both who've written books.) Turning facts and data into meaningful information is--for a lot of books, websites, and manuals--often the destination. *The thing the users want.*

The big problems happen when the user wants and needs *knowledge and understanding* but gets only *information*. If *information* is a meaningful, useful organization of fact and

data, *understanding* is about knowing how--and more importantly *why*--to apply that information to do something creative.

Look at the tech books on your shelf. Chances are, more than a few of them exemplify what Prentice-Hall editor Greg Doench refers to as one of the **seven deadly sins of computer book authors**: *greed*. He calls it greed because he reckons it's the author's way of trying to capture the largest possible market for the book. He spots this most often in book proposals by first-time authors, he claims, when they make statements like "this book is for everyone from beginners to advanced users" and "after they learn from it, they'll use it again and again as a reference."

I don't think it's actually greed, of course. Often the authors don't have enough user data on what readers *do* want, so they're trying to be safe and do both. Or they're trying to make a book that simply *is* more helpful because it *does* offer both information *and* understanding. It's so tempting to try to offer something to readers where they need buy only *one* book that both teaches *and* is a reference-- it sounds so user-friendly. But it's nearly impossible to do well.

Our advice to our authors is: "You MUST choose one, and you must commit body and soul and keyboard to doing that one single thing--either **reference** (data and information) *or* **learning** (knowledge and understanding), while letting go of the other. Accept that you can't meet both goals, and that most of your readers don't *have* both goals, and figure out the best way to satisfy that one goal."

How do you decide which to choose from? Only your users can tell you. There is an interesting trend that might help, though--early adopters tend to need the least amount of hand-holding, and not only can but *want to* jump in armed only with a reference and a few tips. You just point them in the right direction, give them the information, and they're running. It's these early adopters that will help *define* the kinds of user stories that those of who write about more mature technologies will use to teach for knowledge and understanding.

Tim O'Reilly has been giving a talk on What Book Sales Tell Us About the State of the Tech Industry, and O'Reilly's extensive data crunching (for *all* publishers, not just O'Reilly Media) has consistently found that when a technology is fairly new, the

bestselling books on the topic tend to be more advanced, less step-by-step learning. But as the technology evolves, the balance shifts and the bestsellers are the beginning books while the advanced books start to drop off. This isn't completely intuitive unless you think about who the early adopters are. So for example, the bestselling Java book today is a beginner book, and the best selling C++ books are also focused more on beginners, where a few years ago, the bestselling Java books were advanced reference books, not "learning experiences."

This table shows a few of the key differences between Reference and Learning, and explains why we (my co-authors and I) believe so strongly in picking only one side of the table.

Data and Information	Knowledge and Understanding
impersonal, objective	**personal**, subjective
order-driven	**story**-driven
formal, precise language	**conversational** language
random, **non-linear** access	**linear**, read start-to-finish
organization is key	**understanding** is key
must be easy to **reference**	must be **memorable**
focused on **"what"** and **"where"**	focused on **"why"** and **"how"**
can be very **effective online**	usually **NOT effective online**
can work for **many user levels**	focused at a **specific user level**
author skill: **information architecture**	author skill: **teaching**
biggest task: choosing **structure**	biggest task: choosing **story**
common pitfall: **not enough information**	common pitfall: **too much information**
typically used just-in-time, **when needed**	often read away from work, **before needed**

Actively trying to do both means you'll probably do both *mediocre* at best, and today there's not enough tech book business (down to 1/10th of what it was in the bubble) to support anything but the books that know and meet their goal. Obviously there are places on the venn diagram that overlap; I can't conceive of a learning book that does *not* offer facts and information, and a great reference book *does* provide learning by using an information architecture that makes the knowledge and understanding explicit. Some of the best reference books organize the data in a way that offers not just meaning but a revelation... a higher pattern I wouldn't otherwise have seen without that organization. And those higher level patterns and revelations are memorable, just as a well-crafted learning experience.

The other big thing I'm not addressing here is that there are a *lot* of subgenres in each of those categories. It's not just a matter of a straight dictionary-style reference book vs. a fist-time-beginner learning book. You have tutorials, cookbooks, hacks, hands-on expert walk-throughs, nutshell books, and "missing manuals". Many of these have at least some element of each side of the learning vs. refererence table. But the point isn't about *avoiding* the other side of the table--it's about having only one thing as your ultimate goal, and then putting in only what supports that goal. It's about choosing NOT to include things if those things are there simply to try to satisfy both sides of the table (i.e. to be "greedy").

Footnote: when I mention the "greed" thing, it's from a great talk Greg gave at JavaOne on how to make a bestselling computer book. His slides aren't online anywhere, but he's given me permission to reproduce some of it on this blog, so I'll be putting that up soon. It's especially helpful to those who want to propose a book to a tech publisher (today's tip: do NOT put "this book is for both beginning and advanced users" in your proposal) ;)

http://headrush.typepad.com/creating_passionate_users/2005/09/writing_for_non.html

Conversational writing kicks formal writing's ass

By Kathy Sierra on September 6, 2005

If you want people to learn and remember what you write, say it conversationally. This isn't just for short informal blog entries and articles, either. We're talking books. Assuming they're meant for learning, and not reference, books written in a conversational style are more likely to be retained and recalled than a book on the same topics written in a more formal tone. Most of us know this intuitively, but there are some studies to prove it.

Your sixth grade English teacher warned you against writing the way you talk, but she was wrong. Partly wrong, anyway. Then again, we aren't talking about writing the way you talked when you were 12. Or even the way you talk when you're rambling. What most people mean when they say "write the way you talk"

is something like, "the way you talk when you're explaining something to a friend, filtering out the 'um', 'you know', and 'er' parts, and editing for the way you *wish* you'd said it."

So why aren't more technical books or articles written this way? One computer book author (who hates my books) sent me an email saying, "With your books, you want people to have *fun*" (he said it like that was a *bad* thing, but that's a different issue). "But with *my* books, I have a reputation as a consultant to think about, and I want people to have the impression of, 'listen carefully, because I'm only going to say this once.'" Whatever. I've talked about the danger of writing a book from the perspective of what it will do for *you* vs. what it means for the user in How to write a non-fiction bestseller.

Unless the book is a reference book, where precision matters over understanding, and the writing is meant to be *referred to* not *read and learned from*, there are almost NO good reasons for a tech book to be written in a formal (i.e. non-conversational) style. Much of the time, it's an indication that the author is thinking way too much about himself, and how *he* will be perceived. (Or *she*, of course, but to be perfectly sexist here--this *does* seem to be more of a guy thing--the "I'm more technically serious than thou" phenomenon.)

Sometimes it's simply because so many technical books *are* written that way, and it's just conventional inertia ("if the other books are written like that, and they sell, this must be the way it's done"). Other times, it's the author's of way of showing respect for both the topic and the reader--a valid goal, but an ineffective (and unneeded) approach.

And now we know that it's usually wrong, and users/readers are starting to fight back against painfully dry books, no matter how technically pure the content.

A study from the Journal of Educational Psychology, issue 93 (from 2000), looked at the difference in effectiveness between formal vs. informal style in learning. In their studies, the researchers (Roxana Moreno and Richard Mayer) looked at computer-based education on botany and lightning formation and "compared versions in which the words were in formal style with versions in which the words were in conversational style."

Their conclusion was:

"In five out of five studies, students who learned with personalized text performed better on subsequent transfer tests than students who learned with formal text. Overall, participants in the personalized group produced between 20 to 46 percent more solutions to transfer problems than the formal group."

They mention other related, complimentary studies including:

"... people read a story differently and remember different elements when the author writes in the first person (from the "I/we" point of view) than when the author writes in the third person (he, she, it, or they). (Graesser, Bowers, Olde, and Pomeroy, 1999). Research summarized by Reeves and Nass (1996) shows that, under the right circumstances, people "treat computers like real people."

So one of the theories on why speaking directly *to* the user is more effective than a more formal lecture tone is that the user's brain *thinks it's in a conversation, and therefore has to pay more attention to hold up its end!* Sure, your brain intellectually knows it isn't having a face-to-face conversation, but at some level, your brain wakes up when its being talked *with* as opposed to talked *at*. **And the word "you" can sometimes make all the difference.**

One striking part of the Moreno/Mayer study is how similar the actual content was. Here's the before and after example from the beginning of the lesson they studied:

Formal

"This program is about what type of plants survive on different planets. For each planet, a plant will be designed. The goal is to learn what type of roots, stem, and leaves allow the plant to survive in each environment. Some hints are provided throughout the program."

Conversational

"You are about to start a journey where you will be visiting different planets. For each planet, you will need to design a plant. Your mission is to learn what type of roots, stem, and leaves will allow your plant to survive in each environment. I will be guiding you through by giving out some hints."

And from another perspective, consider what former Wired editor Constance Hale wrote in Sin and Syntax:

"The second-person pronoun (you) lets the author hook the reader as if in conversation. Call it cozy. Call it confiding. *You* is a favorite of the Plain English folks, who view it as an antidote to the stiff impersonality of legalese and urge bureaucrats to write as if speaking to the public... " She goes on to give a pile of great examples.

We believe one of the biggest mistakes is to dismiss the things that work in teaching *younger* people by saying that they somehow don't work for adults. That's wrong. At the highest level, anyway. Obviously the implementation of a kid's learning book and one for adults will be different, and different subjects often require dramatically different approaches, but at the core, virtually all brains learn the same way--through emotional response (which in turn triggers the brain to pay more attention and possibly record to long-term storage). And engaging in a conversation has the potential to turn up the emotional gain much more than a dry, lifeless text or lecture.

If your brain had a bumper sticker, it would say:

I *heart* conversation.

http://headrush.typepad.com/creating_passionate_users/2005/09/conversational_.html

You can out-spend or out-teach

By Kathy Sierra on September 7, 2005

Imagine you're trying to launch a new software product, book, web service, church, small business, social cause, consulting practice, school, podcast channel, rock band, whatever. The most important skill you need today is not fund-raising, financial management, or marketing. It's not knowledge management, IT, or human resources. It's not product design, usability, or just-in-time inventory.

The most important skill today is... *teaching*.

Whatever it is you're launching is probably not in short supply, and there's always someone who's doing it better, faster, and cheaper (or will be within weeks). Most of us authors, non-profit evangelists, indie software developers, small start-ups (the soon-to-be Fortune 5,000,000) can barely afford broadband let alone a "marketing/ad campaign". We can't hire a publicist. We aren't going to be on Oprah.

But you're not interested in using deception and bulls*** to manipulate someone into buying a product, membership, or idea that you don't believe in yourself. And that's your big advantage over even the biggest and best-funded competitors: *your belief.*

Because what you *believe* in, you can *teach*. And teaching is the "killer app" for a newer, more ethical approach to marketing. While in the past, those who out-*spent* (on ads, and big promotions) would often win, that's becoming less and less true

today for a lot of things--*especially* the things designed for a younger, more-likely-to-be-online user community.

Kind of a markets-are-classrooms notion. Those who teach stand the best chance of getting people to become passionate. And those with the most passionate users don't *need* an ad campaign when they've got user evangelists doing what evangelists do... talking about their passion.

But passion requires real learning. Nobody is passionate about skiing on their first day. Nobody is passionate about programming in Java on their first day. Or *week*. It's virtually impossible to become passionate about something until you're somewhere up the skill/knowledge curve, where there are challenges that you believe are worth it, and that you perceive you can do.

Nobody becomes passionate until they've reached the stage where they want to grow in a way <i<>they deem meaningful. Whether it's getting better at a game or helping to save the world, there must be a goal (ideally, a continuously progressive goal) and a clear path to getting there. It's our job, if we're trying to encourage others to become passionate, to enable it. And the only way to do that is by *teaching*.

I've talked about all this before, but I wanted to consolidate the links and the "story" in one place:

1) The importance of learning/teaching your users:

Upgrade your users, not just your product

Kicking ass is more fun

(The better your users are at something, the more likely they are to become passionate.)

What software can learn from kung fu

(the Next Level is extremely motivating)

2) Teaching techniques:

Crafting a User Experience

(It's all about flow... balancing challenge and skill)

Keeping users engaged

Learning doesn't happen in the middle

(Have lots of beginnings and endings)

Just-in-time vs. just-in-case learning

(If you don't provide the "why", they may not listen to the "what" and "how")

Is your message memorable?"

(You have to get past the brain's crap filter)

Getting what you expect is boring.

(The "oh shit/oh cool" technique)

The users's journey

(take your user on a modified hero's journey)

The case for easter eggs (and other clever user treats)

(let the user discover "surprises")

Many of us would be better off if we ditched our marketing budget (hah! like we have one...) and put it all toward something that helps the user kick ass, have more fun, and want to learn more. And to be honest with myself here, part of the point is that **people who *want* to learn more are more likely to *want* more of your tools, services, community, and "tribe/pride items" around whatever it is they're learning.**(So make sure you and your wake can support that.)

There's no way I can ski as well on my $100 skis as I can on my $600 skis. That's a fact, not a marketing manipulation or my imagination. That I wouldn't have known the difference (or needed the difference) had I not learned to ski better is an important point, but even if the ski maker had been responsible for teaching me to improve to the point where I needed their more expensive skis, *it makes me happy to ski better*. I'm *grateful* that I've improved enough to benefit from better skis (and thankful I was able to get them). To use the lamest cliche--it really is a win/win.

I can process graphics and video much more quickly on my iMac G5 than I could on my old iBook G4. Thanks to Nikon's free online training, I now can take much more interesting photgraphs with my Nikon 5700 than I could with my old point-and-shoot digital Nikon. Nikon *taught* me to appreciate aperture control, something the clueless recreational snapshot taker I was before wouldn't have wanted and wouldn't have paid

for until Nikon gave me a reason. It's not a b.s. reason. It's not a fluffy "coolness" reason. It's about me taking better pictures--something I don't need, but really really enjoy. (And no, it certainly didn't hurt Nikon either ;)

I'll say it again -- if you're marketing-through-teaching, and helping your users kick ass, and in the process teaching them to appreciate your higher-end products or services, *this is not a bad thing*. I do respect that old-school marketing has done plenty of evil and horrifically damaging things to people and communities (even whole countries). But we are *not* those who pushed products without a conscience. We will be mindful, and we will not promote that which we don't believe in. This is about creating passionate users, and that can happen only if we help our users learn and grow and spend more time in flow.

These moments of flow you can help enable are some of the happiest moments in a person's life. And yes, this applies not just to hobbies and games and sports but even to work. After all, a big part of the success and passion around Getting Things Done, 43 folders, and 37 Signals software is about people being in flow... just getting their daily work done.

So, who can you help find flow today?

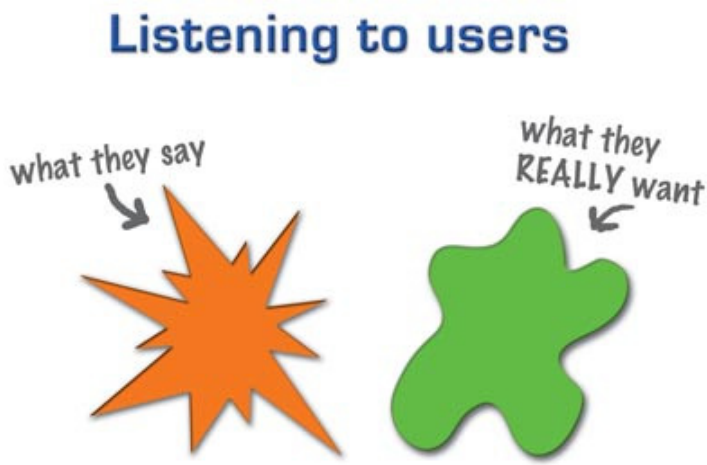
[Footnote: I'm leaving for the Parelli conference tomorrow morning, and internet access will be very limited (it's basically a cowboy ranch). I won't be back until Tuesday night, so if there aren't any more posts until next week, that's why. Matt Galloway and Shaded, you're in charge of comments while I'm gone (I'll make it up to you, I promise ;) No food fights.]

[Update: gulliver left a wonderful, important comment for this post, and as a result I added a few more links into this post. But you really must read the whole comment. Thanks gulliver!]

http://headrush.typepad.com/creating_passionate_users/2005/09/you_can_outspen.html

Listening to users considered harmful?

By Kathy Sierra on September 15, 2005



Is "listening to users" really the most important way to keep them happy and -- if we're lucky -- *passionate*? Is giving users what they *ask* for the best way to help them kick ass? Or should you create or modify a product based solely on what *you* believe in... even if it doesn't match what users *tell* you?

Last weekend I attended the sold-out Parelli Natural Horsemanship conference. I was surrounded by 2000 passionate fans (at least 75% of the people were wearing at least *one* Parelli-branded shirt, jacket, or hat). The conference was amazing (more on that in another post), but the real reason I went was to interview the founder/visionary Pat Parelli, for the Creating Passionate Users book. His hugely successful, multi-million dollar company is one of the few we've found that does virtually everything on our "reverse-engineering passion" checklists, without having first waited for the *fans* to do it themselves.

In the equestrian world (total annual impact of the horse industry on the US economy is $112 billion [yes, that's with a "b"]), Pat Parelli has so greatly outstripped the "horsemanship" competition that it doesn't even make *sense* to talk about competition. Software engineers will appreciate that horse training doesn't scale. So Parelli decided to teach *others* to do

what he does, and of course sell those folks a ton of high-end equipment and training products to help them do it. Nobody -- absolutely no other individual "horse whisperer" or company -- comes anywhere *near* Parelli in size and scope of the business - Parelli has two training centers, one in Colorado and one in Florida (combined over 700 acres for the facilities), and hundreds of thousands of participants in the home-study programs, clinics, and club membership. Their Parelli Horseman's University is one of the only state-accredited "natural horsemanship" programs in the US.

So that's the backstory. I have weeks' worth of posts to make on what I learned from Pat about the ways in which they've become such a passionate user success story, but today's post is about something I had completely wrong when I interviewed him:

Me: "So, you've recently made *drastic* changes to your program--a program that was already extremely successful. It's obvious that you've been really listening to your members and taking their feedback and using that to make these sweeping changes."

Pat: "No, listening to our members was maybe 20% of it, but the other 80% was something else."

And then he said it:

"We changed our entire program because *WE* knew we could do better. Because *WE* were still frustrated that people weren't learning quickly enough or progressing through the higher levels as well as we thought they could. People still weren't having the kind of relationship with their horse that *we* knew they could have, even though our students were delighted with the progress they were making. So we changed it all."

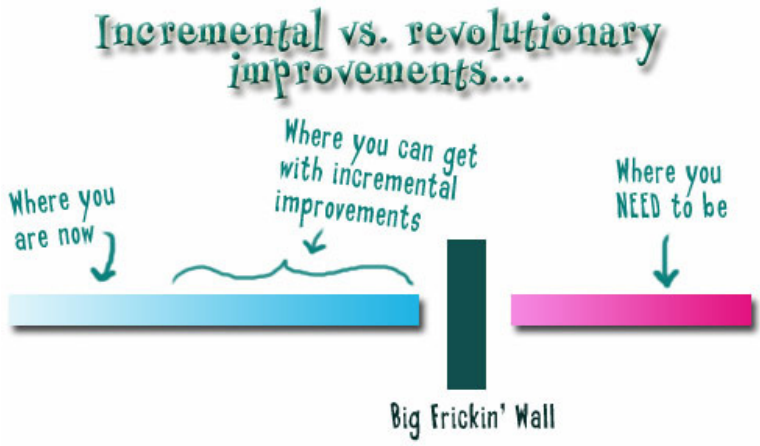
It turned out that most of the major changes they made to their program came *not* from user requests and suggestions, but from the Parelli team's own innovations. He went on to explain that their members/students/users had no idea *what* was needed to make better, faster, deeper breakthroughs. In fact, many of the changes went *against* what their user feedback seemed to suggest. In other words, in many ways the Parelli team deliberately did *not* listen to users.

They trusted themselves, and did what *they* believed was right for their users, even if it meant doing things that on the surface seemed even *less* user-friendly.

Most of us realize that focus groups are notoriously ineffective for many things, but we still assume that listening to real feedback from real users is the *best* way to drive new products and services, as well as improve on what we have. But there's a huge problem with that -- *people don't necessarily know how to ask for something they've never conceived of!* Most people make suggestions based entirely around *incremental* improvements, looking at what exists and thinking about how it could be better. But that's quite different from having a vision for something profoundly new.

True innovation will rarely come from what users say *directly*.

This doesn't mean that you don't *listen* to users--because the truth *is* embedded in what they say...but you have to look for the deeper meaning *behind* what they ask for, rather than always taking them at their word. If they ask for "D", as an improvement to "C", you might have to dig deeper to find out what it is about "D" that they want. And in *that* answer, you might find the nugget that leads you--and only you--to come up with "S" as a solution. And the "S" solution looks nothing at all like "D", but gets to the heart of what users really wanted and needed when they asked for "D".



In the end, you might have to trust yourself, even in the face of users who either want *more* than you know would be good or something less or *different* than you know you can offer if you keep innovating in revolutionary--not just incremental--ways. Our Head First books are among the top-selling computer books today, virtually all of them occupy the #1 slot for their topic category. But not only did *nobody* ask for such a bizarre format

for a technical book, *we were warned that it would never work*. We were told that people would hate these books. That they were too different, too pictorial, too... *tacky* to be taken seriously. But we knew the brain science and learning theory behind the format, and trusted that the principles worked. That for most (not all) readers, this format really *did* lead to faster, deeper learning. We trusted that people would look beyond the surface aspects of the implementation, and that if they got real results from the book, they'd tell others.

Two other publishers turned us down for the series before O'Reilly took the chance. And I was nearly fired from Sun for trying to sneak 5% of what's in Head First into Sun courseware.

Are users/readers too *clueless* to know what to ask for? Of course not. But it's not a potential Java programmer's job to be a learning theory expert, anymore than I could have helped conceive of the iPod. I could make incremental suggestions about most of the tools I use, sure, but I don't have the background, skills, or vision to suggest the kind of revolutionary changes that create breakthrough products and services outside of my own very narrow domain.

What sparked this post was a somewhat contentious (and bold) 37Signals post, but I also remembered this post by Wiley editor Joe Wilcox.

This is tricky, of course, because it's not always obvious which user complaints/suggestions are based on real problems with your product, vs. naive feature requests that would do more harm than good. (Don't forget the Happy User Curve)

And this is NOT about giving them simply what we know is good for them but that they really *don't* want, because they probably won't stick around. This is about giving them what they really DO want... but simply don't realize it because they had no way to *imagine* it.

So maybe the key is to listen not only to what users *say*, but more importantly to what is *motivating* what they say. The rest is up to us. If we really care about our users, they'll just have to trust us... but more crucially--we have to trust *ourselves*.

http://headrush.typepad.com/creating_passionate_users/2005/09/listening_to_us.html

The worst way to calm someone down

By Kathy Sierra on September 18, 2005

Think back a time when you were really angry, frustrated, freaked out, and someone told you to, "Relax. It's gonna be fine. Take a deep breath. Chill." Did this advice make you want to:

A) take a deep breath and relax

OR

B) Take a crowbar to that person's head and THEN relax

If you're like most people, and you're being honest, the answer is "B".

But it's the most intuitive thing we usually do -- either out of an honest attempt to calm them down, or because we think they're being irrational, ridiculous, over dramatic, type-A, or immature. In other words, *we* don't think their state is justified.

One of the most fascinating things I saw last week at the Parelli conference was a demonstration of taking three different extremely nervous (what they refer to as "right brain") horses--fearful, pacing, tense--and bring them to a relaxed state. What I *expected* was what we're all taught to do (or do instinctively with both pets and people)... a process of trying to be as calm and reassuring as possible. After all, becoming excitable ourselves can't possibly do anything but add more feul... right?

But what I *saw* was just the opposite.

The trainer, Linda Parelli, walked near the first horse and started pacing with him. When he turned, she turned. When he stopped to look at something he was afraid of, she stopped to look. When he started to run, she started to run. When he was tight with h is head up, she tighted her body as well. She just kept mirroring him like that for quite a few minutes, and then ever so slowly she started to "lead" just a little by getting to the point where he would normally turn around and taking just one step past it. The horse would follow, but then that was his limit and he'd turn, and she'd turn.

Over the course of 15-20 minutes, she eventually got him to a point where he was paying attention to her and letting her help him go past his earlier limits. Most importantly, whenever *he* relaxed--even if just for a split second--*she* would relax as well. But the instant he tensed up, she'd tense her body as well. Soon you could see a dramatic transformation--where the horse was eventually trying to figure out how to get *her* to calm down... and learning that if *he* relaxed, then she would. So the horse was believing that it was his job to "get this crazy human to relax."

Linda did variations of this with three different horses, all dramatic examples of how this seemingly counterintuitive approach could work a small miracle.

Obviously horses don't think like people. They have prey animal brains, and operate largely on the instincts of life-preservation. But still, I couldn't help but think how much more pissed off I get when I'm really upset and someone tells me to calm down. How completely *unhelpful* it is when I'm nervous and worried about something and someone tells me to "chill".

The foundation of many customer service training programs is to give an angry customer your full attention but remain as rational and cool and calm as possible. We're taught that if we match the customer's right-brain emotion with an emotional response, we'll make things worse. And that's true... at least if we respond *defensively* and especially if we get angry in response.

But still... maybe instead of always being the one who is "more rational than thou" when the other person is upset, maybe *sometimes* in *some scenarios* it would help to at first be a little less calm in response. (Not *angry* at the person who's complaining--that definitely WOULD make things much worse.)

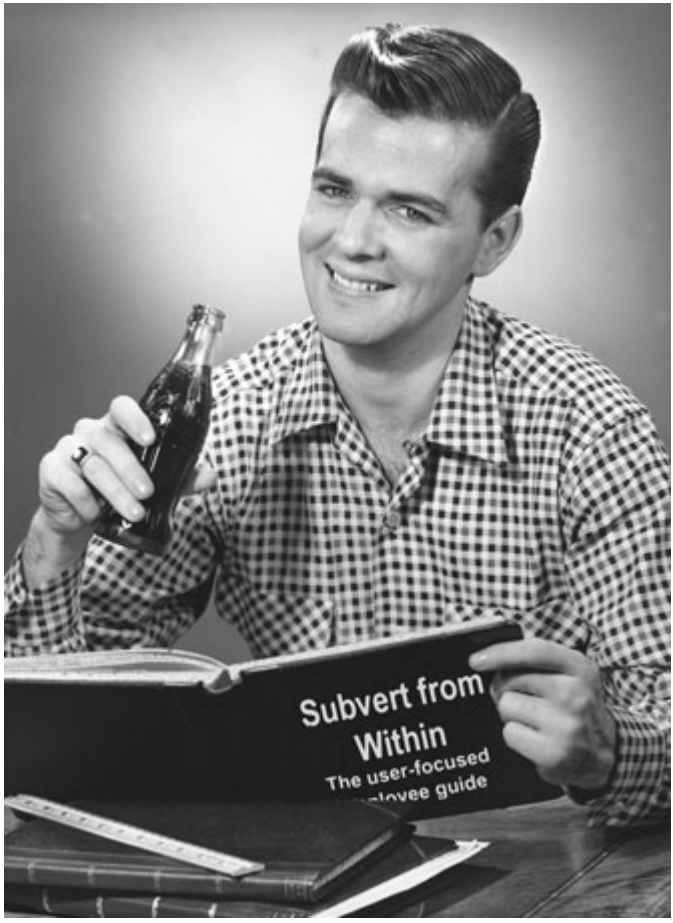
But there is *another* aspect of this that Linda also uses at times, and it goes beyond rapport and into something a little stranger (and deliciously tempting). I am NOT suggesting that this is a good, useful, ethical idea for people, but I'll mention it anyway because I think it's both funny and--with horses--seems to work. The idea is that you not only match the horse in "craziness", but even *exceed* him in some cases by just acting even MORE crazy (not angry or aggressive, just *nuts*)... so that the horse thinks, "Geez... I was scared but THIS human is *crazy*. I'm going to back away slowly and..." So with this approach, the horse calms *himself* down because you gave him something new to think about ("how can I get HER to stop being so crazy?") and that breaks his emotional pattern.

Just think about it... imagine what would happen if someone "goes off on you" and rather than reacting in a purely calm and rational way (or getting angry), you just suddenly act completely nuts. ;)

http://headrush.typepad.com/creating_passionate_users/2005/09/the_worst_way_t.html

Subvert from Within: a user-focused employee guide

By Kathy Sierra on September 23, 2005



It's one thing to talk about--and *execute*--a user-focused approach when you're a small company or an independent contractor. But what if you are, in fact, a fish in a sea as vast as, say, Microsoft? Can you hope to make a difference? Or does working at the "DarkStar" suck the soul from any employee with a passionate users bent?

I spent yesterday at Microsoft. And yes, it was on a "passionate users" mission -- something even my teenage daughter found hilarious given the Microsoft we all know and love to hate. But the day was a string of surprises and challenged assumptions

(starting with finding Liz Lawley in my workshop (someone I'd never met but long admired), and ending with meeting some amazing MS guys including Furrygoat's Steve Mafosky, Shawn Morrissey, and Lou (whose-last-name-I-forgot)).

It's so tempting to say that anyone who really cares that much about users ought to get the hell out of the big company. I know, having done my time at Sun. But I'd forgotten how to see Microsoft as something other than a Big Company. I'd forgotten (or never recognized) that it's a collection of individual *people*, and no matter how entrenched the company's views, policies, practices, values, bureaucracy, etc. are, there are motivated, smart, caring, creative *people* who work there.

And *these* folks have a chance to make a Difference (capital "D") on a scale that most of us will never touch. When Ward Cunningham (inventor of the Wiki, key player in extreme programming, etc.) went to work for Microsoft, much of the software engineering world was horrified that he'd even *consider* it. But he kept insisting that where *better* to produce positive change than going straight into the heart of one of the biggest sources of trouble for both users and developers in the software ecosystem?

But let's say you're *not* a Ward Cunningham or any other *famous*, visible, already influential industry player. You're an engineer, or maybe a program manager. In that case, you do what many of us did at Sun... **subvert from within**.

Here's my little unofficial guide to creating passionate users for those working in Big Companies. Most is from things a maverick (but cleverly disguised as *compliant*) group of us did at Sun, while we could. Only *one* of our original disruption team remains a badged Sun employee, but our legacy persists today in areas that won't make us famous, but *do* make a substantial difference in the experience that users get within the sphere we influenced.

In no particular order, here's a collection of tools used by our formerly underground User Liberation Army:

Language *matters*. Frame *everything* in terms of the user's experience.

In meetings, phrase *everything* in terms of the user's personal experience rather than the product. Keep asking, no matter *what*, "So, how does this help the user kick ass?" and "How does this help the user do what he really wants to do?" Don't focus on what the user will think about the *product*, focus everyone around you on what the user will think about *himself* as a result of interacting with it. Study George Lakoff for tips on using language to shift perceptions.

Be annoyingly persistent.

If you're relentless in the previous step--always asking the question, "how does this help the user kick ass?", it won't take that long before the people you interact with will anticipate that you're going to ask it, and that at least forces them to *think* about it for a moment. Over time, and over a large number of people, those moments can start to add up.

Capture user stories.

Keep a notebook or hipster PDA with you *always* and whenever another employee, blogger, (or user) tells you something good or bad about a *real* user's experience, write it down. Build up a collection, and make sure these stories are spread. Be the user's advocate in your group and keep putting *real* users in front of employees (especially managers). Imagine that you are the designated representative (like the public defender) of specific users, and represent them. Speak for them.

Speak for *real* users... not fake abstract "profiles".

Represent *real* people, not the abstract notion of "users". Rather than saying, "what users really want is...", refer to your collection of specific user stories and talk about *real* people. When you bring up users, talk about specific people with real names and experiences. Too many companies use fake "profile" characters as a way to think about real users (e.g. "The typical user is a thirty-five year old sales manager with a four-year degree and two kids who uses a computer for..."). While that's better than not thinking of users at *all*, it still puts both a physical and emotional distance between the company and *real* users. After all, it's impossible to truly care about pissing off the

"fake" 35-year old sales manager (even if you give the profile character a name, like "John"), but almost everyone starts to squirm when they think about a *real* person becoming upset with them.

When those around you talk about the abstract concept of "users" or "customers", try to bring up specific real people whenever possible.

Be afraid of Six Sigma. Be very afraid. Ditto for most other "quality programs".

Just as using fake user profiles creates and maintains a separation between company and users, anything that treats users as statistics and abstract numbers on graphs is a problem. To treat a poor user experience as some kind of "defect per million" is just crazy. This doesn't mean Six Sigma and other quality programs aren't important and effective... but people are not widgets. When widget A does not fit properly in widget B, that's a defect. When user Barry Porter cannot figure out how to do the basic thing he bought the software for, and he's *frustrated* and his job is at risk, that should provoke a more visceral reaction. Again, people aren't widgets. Make sure those around you keep being reminded of that.

Never underestimate the power of paper.

Print out little signs that say things like, "How does this help the user kick ass?" and leave them lying on the copier, or the fax machine, or taped on a bulletin board and your cube/office wall. Keep changing them! (Remember, once your brain expects to see it, it stops being effective.)

Get your hands on a video camera, and record some users.

This is one of the single best things we ever did at Sun... recording *real* users talking about the bad--and *good*--things they experience as a result of using the product or service. They don't need slick editing. Just simple videos that you can send around the intranet and show at meetings. Having the *user* advocate for himself -- in his own words -- is more powerful

than when *you* speak on his behalf. It's *very* hard for people to think of users as abstract numbers and line items when they have to actually see a *real* living breathing one with a face and a name and an eye color.

Start a subversive club. Right there on campus, recruit and organize your fellow ULA guerillas.

But... just don't call it that. At Sun, we called it a "Knowledge Design Book Study Group", and held meetings where we picked a particular book and then met to brainstorm on "what are the implications of that book for what we do with our users?" Our first book for our study group was Richard Saul Wurman's Information Anxiety (second edition). I don't care *what* your product is or who your users are, if they're human, they're almost certainly dealing with Information Anxiety.

Put pictures of real users on your walls. Act like they're as important to you as pictures of family members and pets.

YOU create the culture of caring about individual user experiences by demonstrating that it matters *this* much to you.

When product features are discussed *without* taking into account how it helps (or hinders) the user kicking ass, adopt a slightly confused, mildly annoyed look...

Act like it's really weird and inappropriate that the person never brought up the user. As though they left for work without putting on a clean shirt or brushing their teeth. It's just something you *do*. Over time, those around you should start to become uncomfortable when products are discussed without the concept of the user at the center. This is *especially* effective when there is more than one of you, so that you can -- as a group -- ALL act confused and annoyed. You want it to appear that EVERYONE thinks the way you do, and that *not* speaking up about the user is just...weird and wrong.

Blog about it

People are listening.

Challenge user-unfriendly assumptions every day.

When someone says, "We can't do that" or "We must do it *this* way" question it. Every time. Don't let anything go unchallenged. And when the answer is "because customers don't like it that way" or "customers want..." or something like that, always ask, "How do we know this?" (just act curious). It might be that the data on which that assumption is based is too old or was never well formed in the first place. You'll never know until you dig deep into the thinking that's driving the assumption.

Gather facts. Build a rational, logical case that maps a user-centric approach to real business issues.

You don't want to get into an opinion war. You want facts and stats on your side. If you can point to a specific plan for a feature change, for example, and say, "Well, when we did something similar over here in *this* area, we had a complaint ratio of..." The more "emotional" and touchy-feely someone perceives the emphasis on users to be, the less likely they are to take it seriously as a business case. There are always going to be a lot of people in the company who refuse to *care* about the real people, but they *will* care about numbers, so you should always be trying to prove that the user-kicks-ass approach has a compelling benefit for the business (beyond the obvious one that *you* and any other system thinker would see). We learned the hard way that we should never take it for granted that other people in the company will even *think* about this idea of the user being passionate and in flow.

Look for first-person language from users about their own experience. Challenge others to solicit first-person, user-as-subject language.

Do everything you can to get user feedback phrased in first-person terms. Rather than feedback that talks about what the user thinks should be in the product, try to solicit feedback that gets the user talking about *himself*. Users tend to want to tell

you what you should add/subtract from the product, but what you need is feedback where the user tells you about *himself* in relation to the product, even if it's negative.

Useful: "I tried to use the XYZ feature, and I couldn't figure out how to make it work."

Not useful: "The XYZ feature doesn't work properly."

Useful: "I was able to make a really cool image as a result of your app."

Not useful: "The app does a great job of image processing."

Set it up as a challenge for yourself and others you work with to figure out ways to generate first-person feedback where users talk about themselves. Make it a game or a contest to see who can get the user to use the "I" word the most often. What kind of questions could you ask that would lead to the user talking about himself rather than YOU or your PRODUCT?

Don't give up.

If you do, then quit at the earliest possible moment. But if you're relentless and you slowly recruit others to your cause, you *can* change a culture... one small group at a time. If you succeed, even in a small way, and help shift the supertanker just one degree... that one degree eventually means a profoundly different trajectory down the road. Even if your chance to make a difference is slimmer than for those of us in smaller groups (or lone wolf operations), you have a chance to make a WAY bigger impact, touching far more people's lives.

I must say that I won't ever feel the same way about Microsoft now that I've interacted with these folks. And while you might not have heard much about Brady Forrest (the guy responsible for bringing me in to do the workshop at Microsoft), that's going to be changing. I have friends at Sun, and now I have friends at Microsoft. It's hard to refer to something your friends belong to as "evil". And even if corporate Microsoft WERE truly evil, I reckon if my friends are there fighting the good fight from within to produce change, that's something I can feel good about.

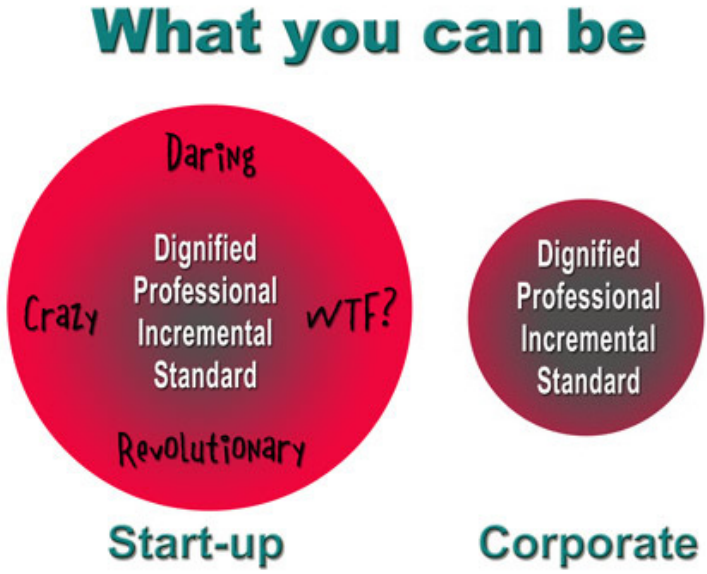
[Be warned, though, that I was asked or rather *urged* to leave Sun as a result of some of what's in here so... I wouldn't be

taking advice from me if I were you ;) I finally got the "you're not a team player" warning and put on probation (and eventually asked to leave), but my response was, "Oh, I AM a team player. It's just that I'm on the *user's* team." (I left out the part about, "Since clearly nobody ELSE around here is...")]

http://headrush.typepad.com/creating_passionate_users/2005/09/subvert_from_wi.html

"Dignity is deadly." - Paul Graham

By Kathy Sierra on September 26, 2005



What goes away when a company moves past the start-up phase? Living only on take-out and caffeine. Working in a [small] living room. Crazy, stupid, unprofessional behavior. *Wearing nothing but shorts and ripped t-shirts.*

Is this a good thing?

Hacker-turned-start-up-investor Paul Graham doesn't think so. In his keynote at the internal Amazon developer's conference in Seattle (that I was speaking at last week), he had a list of 40 bullet points of things Big Companies could learn from start-ups. He doesn't have an essay up for this, but he has a wonderful, somewhat related essay that I'm *hoping* you've all read by this time on What Business Can Learn From Open Source. (If you're new to Paul Graham, he can be an "acquired taste". Very smart, often controversial, rarely politically correct. Almost *always* thought provoking--or at least hurl-your-mouse-across-the-room provoking.)

My head was already spinning by bullet point six, but the one simple thing that stuck in my head was "dignity is deadly." Specifically this thought (I'm paraphrasing):

When you evolve out of start-up mode and start worrying about being professional and dignified, you only *lose* capabilities. You don't *add* anything... you only take away. Dignity is deadly.

At one point, Sun wasn't much more than creative genuis Bill Joy ("Oh, I think I'll just whip up BSD Unix on my own..."), and troublemaker Scott McNealy. Yet by the time I got to Sun, using the word "cool" in a customer training document was enough to warrant an entry in your annual performance eval. *And not in a good way*.

I cannot count the times I heard the word "professionalism" used as justification for why we couldn't do something. But I *can* count the few times I heard the word "passion" used in a meeting where the goal was to get developers to adopt our newest Java technologies. What *changed*? More importantly, was it a *positive* change? Was it a completely *necessary* change?

Why do we go from the business equivalent of the unruly-but-creative teenager to a stuffy parent? Can't we be something in-between? Why not the motivated, fun, creative 30-year old? (I'm not being ageist here -- this is a metaphor). If we're forced into becoming the "parent", why can't we at least be the *cool* parent from down the street? And by "cool", I mean the *truly* cool, not cool simply because they supplied the beer. (The 37 Signals folks always have a lot to say on this "stay small and act like a start-up" approach as well)

Some argue that by maintaining strict professionalism, we can get the more conservative, professional clients and thus *grow the business*. Is this true? Do we really *need* these clients? Isn't it possible that we might even grow *more* if we became braver? Seth Godin cautions that today, "Safe is risky, and risky is safe."

I'm somewhere in the middle of this. I'll use the word "ass" as in "kick-ass". But when I use the "F-word", well, there you have it. It's the "F-word", not the *actual spelled-out word*. hugh macleod, on the other hand, has a take-no-prisoners view. He'll do whatever the hell he pleases, always being 100% true to who he is, and when someone warned him that if he didn't cut back he'd never get the Big Clients, his response was: "Do you honestly think I'd have a good working relationship with clients who are offended that I used the word 'penis' in a cartoon?" He doesn't *want* those clients, and apparently... he hasn't done too

badly recently finding clients who like him just the way he is--pure authentic hugh--thank-you.

Yes there *is* a "Business Case" for maintaining certain levels of professionalism, dignity, and political correctness. And that's cool... as long as we're all recognizing at every turn that in some ways we are *losing* the tools we have available to us. That this need to meet professional expectations *restricts* us... perhaps even more than it *enables* a higher level of... what? Profits? Business? Clients? Respect?

The Head First book series was an attempt to use virtually everything brain-friendly that we were *not* allowed to do at Sun. And when Head First Java first came out, it immediately became the number one selling Java book, and still is today, just over two years later. I'm not at *all* suggesting that some of what's in Head First would have been appropriate for an official Sun course document, but could they have incorporated 20% without sacrificing dignity? Maybe.

By the time we ran things through the deadly *professionalism filters*, the life, passion, joy, and in this case--brain-friendliness--had been sucked out.

When "we just can't DO that here" takes away more than it adds, we should reconsider. But, people scream, "we can't afford to say f*** 'em to some of our biggest potential clients!" And I wonder... can we afford *not* to?

http://headrush.typepad.com/creating_passionate_users/2005/09/dignity_is_dead.html

Think Young

By Kathy Sierra on September 29, 2005

Is there something you loved to do when you were younger but that you stopped doing? Did you stop doing it because you truly outgrew it... or because you got *older*? If you want to keep your brain sharp and--just as importantly--*get to know your next generation of users*, you might want to dust off the legos and slot cars, buy a PSP, get out your skateboard, wear something from Urban Outfitters, and start going to live shows by bands you've never heard of.

Granted, half of you reading this are young enough to still be doing these things, and most geeks tend to *play* more than non-geeks (the average cubicle of the typical geek looks like a Toys 'R' Us kiosk), so some of you will have to work a little harder to come up with things you did when you were younger but *don't* do now. Or even better, things you did *not* do when you were younger, but always *wanted* to. (Data point: The fastest-growing group of first-time horse owners today are 40-year old women.)

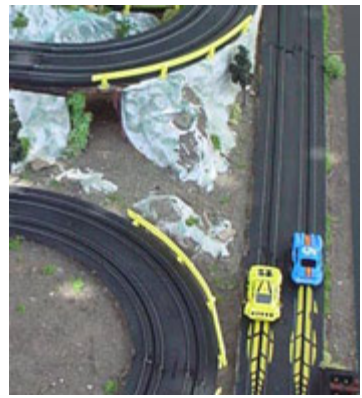
The Death by Dignity topic brought up some great comments about this including:

Michael Turyn: "Narrow-minded and humorless" is often mistaken for "mature"....

Tom Biggs referenced the Oscar Wilde quote: "Life is too important to take seriously."

But this post on college admissions from Julie Leung prompted my post here, especially with her last line:

"As our kids play Kick the Can, can we play with them? Even if that means kicking away our expectations?"



I've talked about the importance of knowing your users' brains, and staying on top of your *next* generation of users before in:

If you're over 35, do you have a clue? and when I realized that pissed a bunch of people off, I followed it with this one. But this is as much about keeping your *own* brain in tune as it is about keeping in touch with your users' brains. Doing things not typically done by people "your age" (whatever age that is... 25-year olds aren't doing the same things they did at 17) is a variation on blow your own mind.

Here are a few more tips:

1) Shuffle your music

Ryan Rawson, who was in my session at the Amazon conference, said that putting his iPod Shuffle in shuffle mode has completely changed the way he listened to music, and sort of "forced" him to stop listening to the same things over and over. Think about that--how many of you load your MP3 player with 5,000 songs, but still end up playing the same five playlists?

2) Have kids

If you don't have kids, rent some. Virtually *any* of your friends with children will be ecstatic to lend you theirs. I'm deathly afraid that once Skyler has completely moved out of the house, my appreciation for indie music will plummet, and I'll revert back to the 80's. (And not the *good, interesting, fashionably retro* 80's.)

3) Go to a toy store

Bring your credit card.

4) Make something

In atoms, not just bits.

5) Go to a live show

Yes, the parking is a pain, the second-hand smoke will kill you, and your high-frequency hearing is already shot from the concerts you went to in high school. Those were my reasons when I went for about five years without attending a *real* concert (the symphony doesn't count).

6) Attend a high-school talent show

Phone the local high schools and find out when their next talent show is. I guarantee it'll be entertaining. In a cringing sort of way.

7) Have--and play with--at least one remote control thing

Slot cars, RC hang gliders, boats, whatever.

8) Do a cartwheel at least once a month

My friend Solveig swears this is the secret to staying young. I hadn't done one for a decade when she forced me--under the influence of some microbrew--to do one in the middle of a San Francisco street after a JavaOne party. It nearly killed me, but now I make a practice of it.

9) Try to play that instrument you haven't touched for years

Guitar. Piano. Trombone?

10) Run

Virtually *everyone* runs when they're younger. Put an animal in a cage all day, and the first thing they want to do when you let them out is run, run, run. We should learn from that.

11) Watch movies for which you are not the target audience

12) Visit stores for which you are not the target audience. Buy something. *Wear it*.

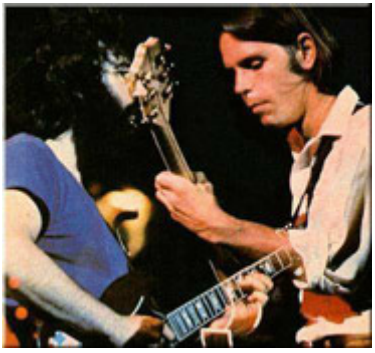
13) Be in a parade... or something just as ridiculous that you would never have considered before

It's something I did a few times as a kid, and did it for the first time as an adult a few weeks' ago. The stable where I board my horse is about 75% kids, and the stable owner decided to take 20 of them to be in the parade. My horse trainer said, "you're going to come to. It'll be good for you and your horse." After I stopped laughing, I realized he was serious. I thought that was the lamest thing I could imagine -- me with the 20 kids. And to make it worse, the theme of the parade was "the beach", so we all had to wear hawaiian shirts or bathing suits with leis and beach towels. One girl even had her horse in dreadlocks. But, I did it.

14) Do something with art -- paint, sculpt, whatever it is you used to do as a kid that you haven't done in a long time

15) Play games. Monopoly. Simpson's clue. Werewolf.

16) Read a mystery/thriller. Or whatever genre you *used* to read but don't any longer.

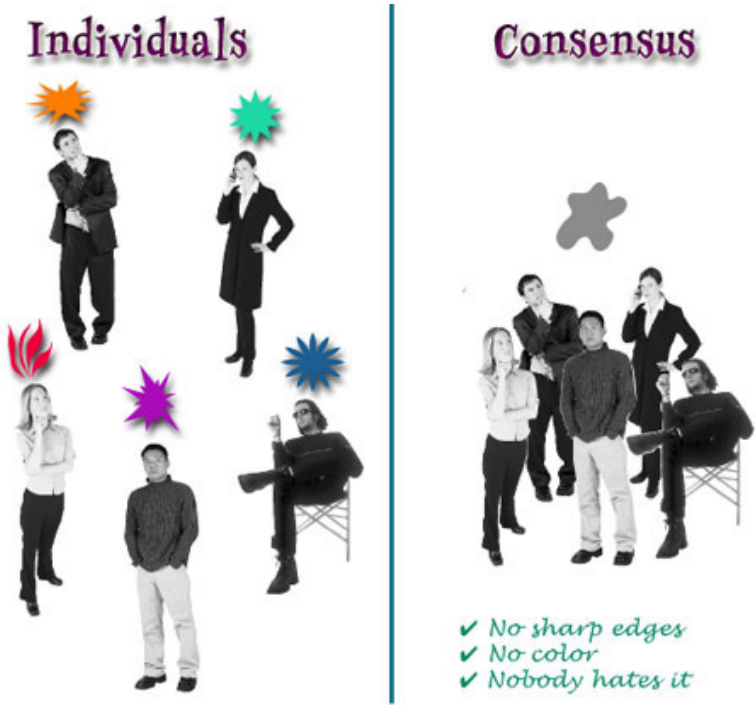


Your turn. What did you really LIKE doing when you were younger, but haven't done in quite a while? This doesn't mean that you're going to get back *into* it... but what can you at least *try*?

http://headrush.typepad.com/creating_passionate_users/2005/09/think_young.html

Keep the sharp edges!

By Kathy Sierra on October 2, 2005



"Great software isn't created by committee." That quote came from James Gosling, at the developer "fireside chat" at the last JavaOne. And from Applied Minds tech wizard Bran Ferren, "Art isn't the product of a team." Is this true?

First, I don't believe James was necessarily talking about the functionality and code when he said "great software". Clearly, teams of great programmers can produce great code. I *think* he means the kind of breakthrough apps that people can become passionate about, and I also think it's less about the programming and more about the *design* and spec.

And we can all have our own interpretation of the word "team"--at what point does a reasonably small, synergistic group building and adding to one another's strengths turn into an idea-crushing, groupthink team? That depends... very few good novels are written by more than *one* person. Perhaps for novels, *two* is the maximum, and even that's pretty rare. And we all recognize that indie films today tend to be of much higher

storytelling quality than the watered-down major studio films where there's often a huge gap between "the director's cut" and the final release edit.

But what about software or other products? What about the team responsible for decisions that affect the context in which users interact with your product, service, or company? How big can *those* teams be before they become completely dysfunctional? Obviously there's no absolute number... *two* people can cancel out each other's good ideas just as effectively as a dozen. If it's not simply about the absolute number, then what *is* it about?

It's about how hard the team/group works to exploit the smartest aspects of the team while maintaining the distance and diversity so artfully (and scientifically) suggested in James Surowiecki's Wisdom of Crowds book. It's about aggregating the intelligence of the individuals rather than having the group make decisions as a whole. And those are two profoundly different things. If you haven't read the book, I made an earlier summary of one of the key premises here.

Most importantly, it's about working to *keep* the sharp edges instead of smoothing them all over. It's about avoiding the dreaded "morph".

You've seen the morph phenomenon, where products end up looking like a morph of all competing products until there's virtually no major distinction. Nothing remarkable. Nothing we *love*. Think of all the new cars you see today that look soooooo much like every other car. With a few exceptions (like the Honda Element, the MINI), most look like they've been run through a morphing program that found the perfect *average*. I was about to add the ScionxB to my list of examples, but then I realized that it's looking dangerously close to the Element... and if the car designers aren't careful, we'll just be exchanging a road full of lookalike rounded cards for a road full of box-like cars with very little difference between them.

This is the car I've wanted all my life:

The '64 Mustang. My dream car.

And though some of the newer Mustangs over the years have been nice-looking cars, the designs today now look like they've been morphed with that of many *other* cars:

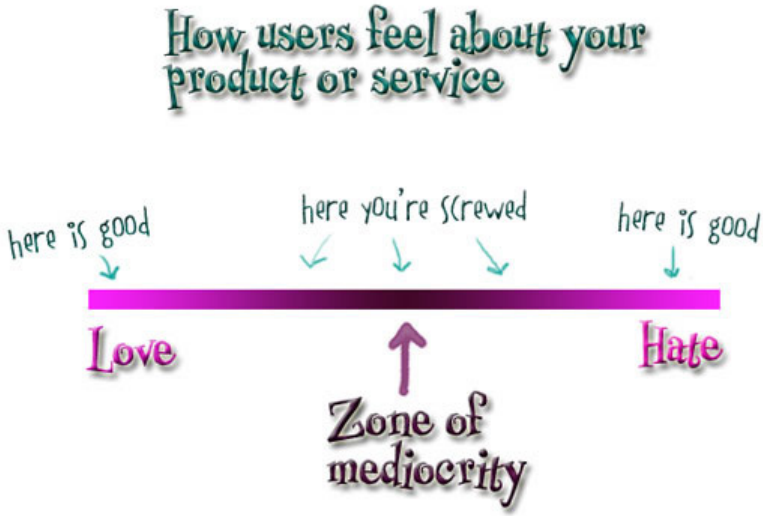
Metaphorically speaking, ***where are the sharp edges?***

Where are the strong ideas that come from either an individual or the product of *true* brainstorming? (Not the kind of meetings that *pretend* to be brainstorming, but where someone always plays "devil's advocate" and kills innovation at the roots, or where we all know that if we don't go along with "the group", we'll be in trouble.)

When people aren't brave enough for one reason or another, ideas are morphed and the sharp edges are worn away until there's little left but a completely palatable, utterly unlovable lump. (Again, I don't mean "sharp edges" literally--I happen to love my iPod precisely *because* it has no sharp edges... very sexy indeed.)

But then what *do* you do when the sharp-edged ideas of individuals are all different? You pick *one*. Or, applying the wisdom-of-crowds model, you take the best of several. But rather than morphing, you *aggregate* the ideas in whatever way is meaningful to this kind of product, service, process, idea. And you read Surweicki's book to find out why forcing out anyone who doesn't "fit" with the group can be not just *unproductive*, but in some cases *deadly* (read his discussion about the Space Shuttle).

And I'm going to keep posting this picture, way past the point when you're sick of seeing it:



If we allow groupthink/consensus to win, smoothing over all the pointed edges, we'll indeed have something that nobody hates. How many of us can afford to be there today?

http://headrush.typepad.com/creating_passionate_users/2005/10/keep_the_sharp_.html

Death by Devil's Advocate

By Kathy Sierra on October 6, 2005

Tom Kelley--general manager of IDEO--believes that "devil's advocate may be the biggest innovation killer in America today." We've all been in a meeting where a passionate idea is put forth but someone plays devil's advocate and drains the life out of the room. Invoking "the awesome protective power" lets the devil's advocate be incredibly negative and slash your idea to shreds, all while appearing not only innocent but reasoned, balanced, intelligent... all attributes loaded with business "goodness". Whew! Thank GOD for the devil's advocate, or we'd all be off blundering with our stupid ideas, oblivious to the insurmountable problems we were too clueless to see.

And it's that attitude--that notion that people can use "playing devil's advocate" with impunity--that Kelley believes is so damaging. In the October edition of Fast Company magazine, there's an excerpt from Kelley's upcoming book The Ten Faces of Innovation which is all about ways to *defeat* the devil's advocate to keep innovation alive. From the excerpt:

"What makes this negative persona so dangerous is that it is such a subtle threat. Every day, thousands of great new ideas, concepts, and plans are nipped in the bud by devil's advocates.

Why is this persona so damning? Because a devil's advocate encourages idea wreckers to assume the most negative possible perspective, one that sees only the downside, the problems, the disasters-in-waiting. Once those floodgates open, they can drown a new initiative in negativity."

Part of the problem is simply the *timing* of the devil's advocate invocation; if the devil jumps in at the earliest stage, the idea never has a hope in hell, or ends up being having all of its sharp edges smoothed over. And there's a big difference between someone crushing an idea based on spinning out *possible* negative scenarios, vs. someone who voices a genuine concern backed with real facts.

But this is tricky and subtle... I've been known to be the one to "voice a genuine concern backed with real facts" without stopping to consider whether those "facts" were *still* valid. The old, "We tried that before and it didn't work." is probably the fastest way to stop an idea, but someone always needs to ask, "Are we sure we tried EXACTLY that?" and "Has something changed in a way that invalidates what we tried earlier?" Or even just this response when someone says, "We tried that...", "You tried what?" Maybe the thing that was tried before was different in some non-obvious but profound way.

The other tricky thing is that if you try to shut a devil's advocate down, then you're perceived as being "unwilling to hear criticism" or "can't handle any disagreement". And of course, for however dangerous the devil's advocate is, there's the equally-dangerous "angel of optimism". The "angel of optimism" is one who answers every genuine criticism with a cheerful and dismissive, "Oh, there's always someone thinking the sky is falling." This doesn't mean that being cheerful and positive is a *bad* thing (as one who is all too often accused of playing this role), but *both* the angel of optimism and devil's advocate can do damage when they shut down other solutions.

I have no good answers to this, but Kelley offers some in his book. His main tool to fight against the devil's advocate is to simply have *other* personas. In other words, if Fred can play devil's advocate, then Danese can play a *different* role, one of the "ten faces" in the title of the book. These include "the

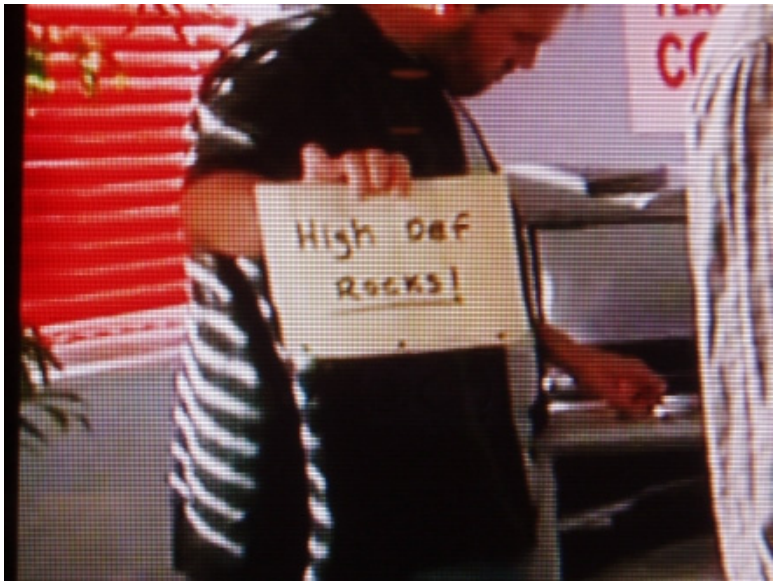
anthropologist", "the experimenter", "the experience architect", "the storyteller", and "the cross-pollinator".

One thing I know for sure, whether playing devil's advocate, angel of optimism, or any other persona, I believe the emphasis should be on offering *solutions*, not just criticism. Yes it's true that one can know something is wrong *without* knowing how to fix it, but if people tried to adopt the perspective that "I'm going to try to always include possible alternatives and solutions when I critcize", it might make meetings a little more bearable.

http://headrush.typepad.com/creating_passionate_users/2005/10/death_by_devils.html

Give users something to talk about

By Kathy Sierra on October 9, 2005



If you want people to talk, ***give them something to talk about.*** We all know that, but I love to see new examples. The picture above is from a scene in the television show My Name is Earl. But not *everybody* saw it. Only those with an HD TV had a picture wide enough (let alone <i.clear enough) to see this image of the guy holding the sign by the copy machine that says, "High def rocks!"

Apparently other shows are now including content that is either intended *only* for those with HD, or simply isn't available to those watching television at non-HD resolution. While this isn't rewarding the majority of their viewers, it's certainly giving the group of passionate television watchers (someone willing to spend several thousand dollars to watch TV is... never mind) something worth talking about.

Be sure to read the comments at HD Beat. My favorite is this one:

"So to summarize, you guys spent 1 to 2 thousand dollars for the privilege of seeing a guy hold up a sign?"

But that one little effort from the producers made those viewers who *did* spend WAY too much for a television feel... special.

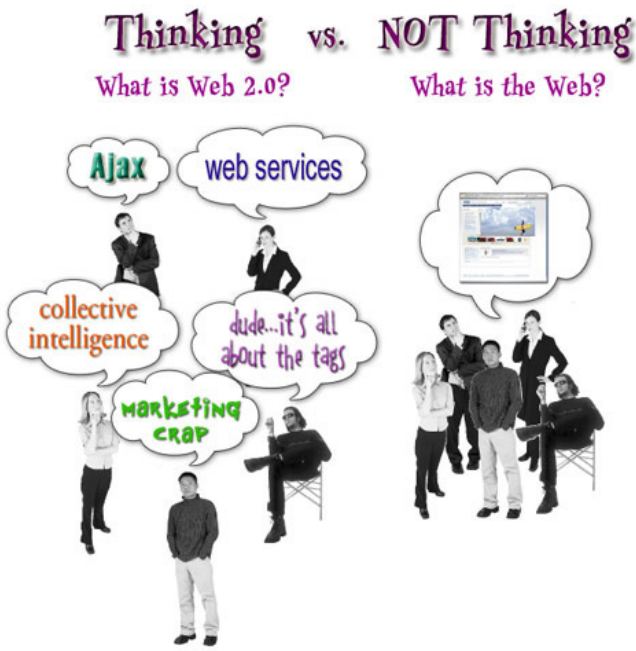
Rewarded. There's a lot of power in the feeling of I know something you don't. If you have a chance to offer that to a group of users, do it.

So how are *we* doing that? Not very well, I'm afraid... but we try -- in our books, for example, there are little bits of continuing storyline and character interaction and easter eggs that make sense *only* if you've read *several* of our books. So only people who have three of our books would ever realize those surprises and "inside references" exist. These surprises aren't any more special than a guy holding up a sign -- they don't add real content *value*, but it's our way of giving some of our most dedicated readers a tiny potential treat (assuming they notice -- if it's *too* obvious, remember, then it might not have as much value).

http://headrush.typepad.com/creating_passionate_users/2005/10/give_users_some.html

The best thing about Web 2.0

By Kathy Sierra on October 11, 2005



The best thing about Web 2.0 is that: *nobody knows what the hell it really means*. Even the ones who coined the term are still struggling to find a compact definition. And this is the true beauty and power of Web 2.0-**it makes people think**.

Not only does virtually nobody know what it really means, but we don't even know what it does NOT mean-Jason Fried blogged about The Top 10 Things that aren't Web 2.0 and as of today, that post has 88 comments. Yes, 88 comments arguing and debating almost Hot or Not style about whether something is or is NOT Web 2.0.

And anything that gets this many people talking, arguing, debating, and most importantly-*thinking*-is a really good thing. An amazing thing. Because each time someone fires a single neuron deciding whether there even *is* such a thing as Web 2.0 or whether it's just all marketing hype, is a moment in which that person gains knowledge and understanding. Not because someone shoved a perfect, high-resolution definition down their

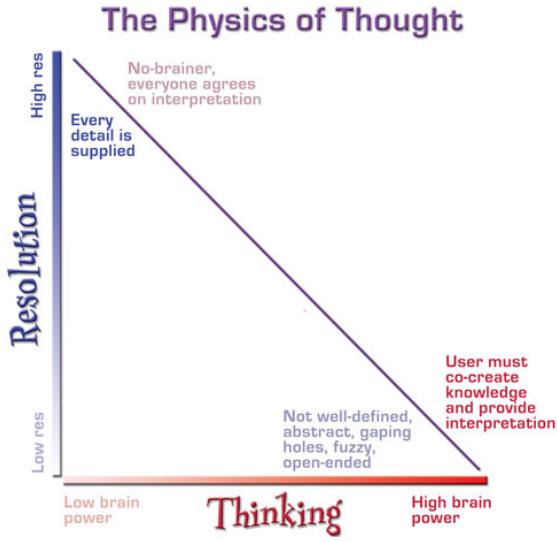
throat, but because the person was virtually forced into figuring it out for themselves.

Real knowledge and understanding is the product of a co-creation.

A joint effort between a brain and... something else. That *something else* can come in many forms-experimentation, conversation, studying, reading, synthesizing, etc. And some of the deepest, richest opportunties for new knowledge co-creation are those *forced* on our brains by low-resolution but compelling ideas, pictures, and concepts. And Web 2.0 has to be one of the most thought-provoking memes in recent history. Thinking has an absolute value all by itself, even if the *thought* provoked is simply "That's complete and utter crap!"

I have no idea what Web 2.0 really means. But the metacognitive effect of the Web 2.0 meme is one we can all learn from. After all, many of us would kill to get this many people thinking and *talking*. 88 comments on a short list of what something is *not*? Think about that...

If you're trying to help someone learn, inspire them, motivate them, engage them, involve them, or just get *some* kind of a reaction beyond mental and emotional flatline, **turn down the gain in strategic places.** Good teachers, filmmakers, novelists, advertisers, and storytellers know this. It is part of what makes cartoons so compelling.



In the classic (must must must read) Understanding Comics, Scott McCloud suggests that the more abstract (as opposed to photorealistic) nature of cartoons allows the viewer to identify with the character. An abstract, iconic face could be... almost anyone. But as photorealism *increases*, the likelihood of the user seeing himself in the character *decreases*. A cartoon happy face could be me. A photoreal image of a 25-year old male with cropped hair, a beard, and a pierced nose is clearly *not*.

But it's not just about whether you can imagine *yourself* as the character. In novels, for example, even with fairly explicit descriptions of the characters, our brains can't *help* but supply the details. We literally *create* the characters in our minds, and that's a big part of what keeps us engaged.

Advertisers use this notion of low-resolution with tricks as simple as using black and white (or very desaturated colors) rather than full vibrant for a sensual print or television ad. When the ad is full-color, high res, our brains can just kick back. But when the image is missing information, such as color, our brains can more easily become sucked into the image, supplying the pieces. *Filling in the blanks.*

Filmmakers use this in everything from cinematography to whether the ending is fully resolved. I saw A History of Violence last week, and walked out with the rest of the audience talking about what the ending meant, and speculating on what happened next. Clearly *nothing* happened next:the story was over! But our brains couldn't help spinning out scenarios and filling in the things that weren't said at the ending. Had they given the movie a nice Hollywood style ending, where everything is wrapped up complete with a bow, we would have left the theater satisfied, but with nothing left to think about (unless the film left *other* holes).

Good teachers use this-they leave holes. They ask learners to fill in the blanks. They use a smackdown learning model that forces learners to choose between multiple and potentially conflicting points of view. They don't lead users step-by-step down a carefully crafted, everything is supplied path. They send them out to explore, possibly even nudging them down a garden path that will lead to surprises (including failures) they never expected.

Many of us tend to think that more is more. That the more detail we provide, the better it is for our users and learners. Sometimes that's true, especially with reference material. But if you're trying to get people to learn, think, remember, engage, understand, grow... less is better. *Strategically removed, hidden, or temporarily withheld* content can mean the difference between passive, surface learning and involvement and deep, lasting, understanding.

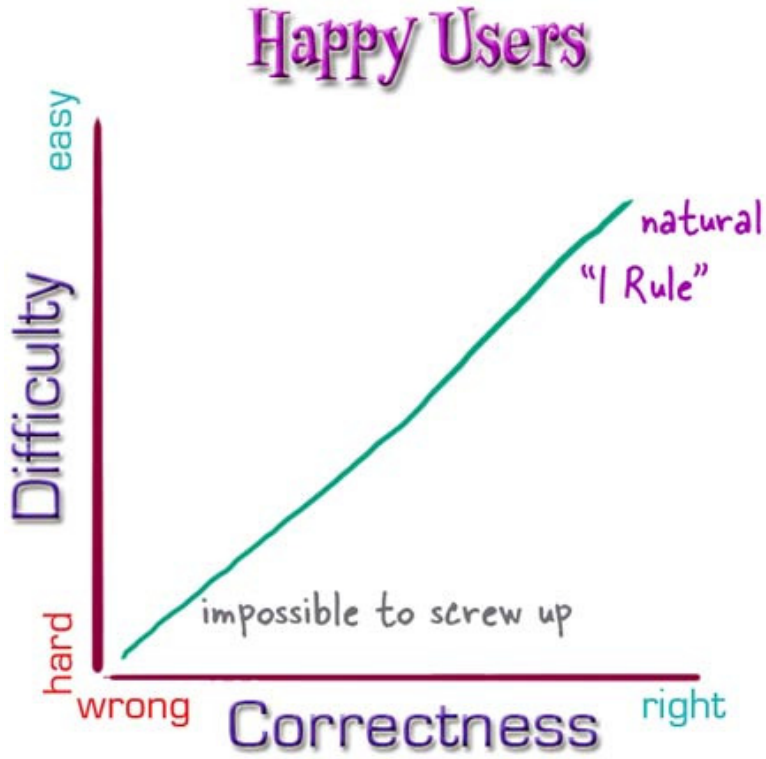
Our brains are wired to fill s*** in. That's what they *do*, and they can scarcely help it. Mind Hacks is loaded with examples of ways in which our brains supply missing information, often without our conscious awareness. But you can use this to your advantage when you're trying to get someone involved, and especially when you want them to learn.

Yes there's a huge danger that if you're not careful and strategic you'll just piss people off. Witholding content to help draw the user's brain in is certainly risky, and doesn't belong in places like, say, technical specs or some aspects of an interface. And Tim O'Reilly has certainly taken the heat for the whole Web 2.0 thing. But he's gotten more people talking, thinking, and even *creating* new things as a result. Had the Web 2.0 meme come down as a perfectly defined, high-resolution description with no room for user interpretation, it would have been far weaker. As it stands, Web 2.0 can never be said to mean *absolutely nothing*, because the brain power it takes someone to reach that conclusion had value.

http://headrush.typepad.com/creating_passionate_users/2005/10/the_best_thing_.html

Making happy users

By Kathy Sierra on October 19, 2005



"Make the right thing easy and the wrong thing hard." If designers followed that one clear principle, there'd be a lot more happy users. I'd get a lot more work done instead of struggling with a counterintuitive interface. *Writing* software would be easier because APIs would simply *make sense*, with less chance of blowing up at runtime. *I could use my car stereo.*

That mantra is one I hear every day, from my horse training coach, as the foundation for "natural horsemanship" principles. But I can think of a certain programming language and certain software and hardware vendors that could stand to spend some time hearing my trainer repeat that over and over and over...

Notice that the chart does NOT say, "Make the EASY things easy." It says, "Make the RIGHT things easy." And those things

might indeed be quite complex. "I Rule" experiences don't come from doing brainless trivial tasks. (But you can certainly have an "I Suck" experience when trivial tasks are made *hard*.)

The goal is to make it easy for the user to do *the thing he really wants to do*, while simultaneously making it difficult or impossible to screw things up. Every screw up, road block, confusion takes the user out of the flow state. It stops him from the thing he cares about, which is NOT how to use whatever tool, device, software it takes to do it.

Yes, I know this goal is dead obvious and "duh." But why are we drowning in products that seem to be made by those who have forgotten this? Or at the least, by those who were *unable* to do it...

Some examples of "make the right things easy and the wrong things hard" are:

1) A strongly-typed language that stops you from assigning a String of characters (like, "cheese") to a variable that you *said* was supposed to hold a number (like 42). Or a language like Java with an "exception" mechanism that forces you to acknowledge that bad things (like, the network is down) can happen.

[Yes, you give up other things in exchange for this "protection", so I'm not saying strongly-typed languages are right for everything...]

2) A product whose physical design makes its use obvious and natural, like a jack that fits into only one kind of port, and in only one, obvious orientation.

One variant of this is the concept of affordances, an example of which is a cup with a handle. The handle is said to *afford* grabbing it--which is the right thing. But a car dashboard with a nice flat surface affords the *wrong* thing--setting things on it (putting light papers on the dash can reflect on the windshield and make it impossible to see, not to mention what it does to your driving when things go sliding off the dash).

It's still possible to make products whose "correct" use is easy, but which *also* invite incorrect or even dangerous use. If designers follow only *half* of the principle ("make the right thing easy") but *don't* "make the wrong thing hard", then you might

have a 50/50 chance that a user will, say, *blow up* if they he plugs the X into the Y.

3) An API design which exposes the highest-level interface rather than a huge pile of lower-level calls (which could make it way too easy, for example, to call the right things in the wrong order), and whose methods/operations are *named* well! Half the reason our books sell so well is simply because some of the Java API designers used names that practically *beg* you to do the wrong thing.

4) A school program that relys on interesting group projects rather than dull, rote memorzation homework. And make those projects something you do mostly in class!

5) And speaking of kids... I try to follow this with the teenagers as much as possible, and one of the simples ways is to have a reduced rule set. The fewer the rules, the harder it is to break them (i.e. the "wrong" thing), and the easier it is to adhere to the ones that are there.

Important note: remember that this isn't simply about making everything easy or dumbing everything down! If I'm working on a video edit, for example, the video edit is where I want my brain bandwidth to go, *not* how to tell the *software* that the edit should go *here*. The point is to make the thing I want to do... the "right" thing... easy, but keeping that "right" thing as complex and sophisticated as it should be. I want my video editing software to give me enormous power, but I want to focus all my brain energy on deciding where--and how--the edit should *happen*, and have the act of causing the edit to take place as natural as possible.

Every moment I spend trying to figure out the interface--or worse, trying to recover from a terrible mistake the software allowed or even *invited*--is a moment *not* spent creating something. Doing my *real* work.

Games, for example, should not be *easy*, but the interface in which you play the game should be. The game should allow me to *stay in character* and not break the flow by forcing me to deal with a user error (as opposed to a "character" error) or by forcing me to stop and look at the manual again...

Also, this does principle/mantra does NOT totally apply to *learning experiences*, with the exception of tools used to *deliver* the learning experience. Much of the most memorable learning

comes explictly *from* failures and mistakes--things that did not match your expectations. And I sure don't want my next airline pilot to have had training that supported only the right thing (although many industrial disasters have been linked to cockpits and controls that made the wrong thing easy).

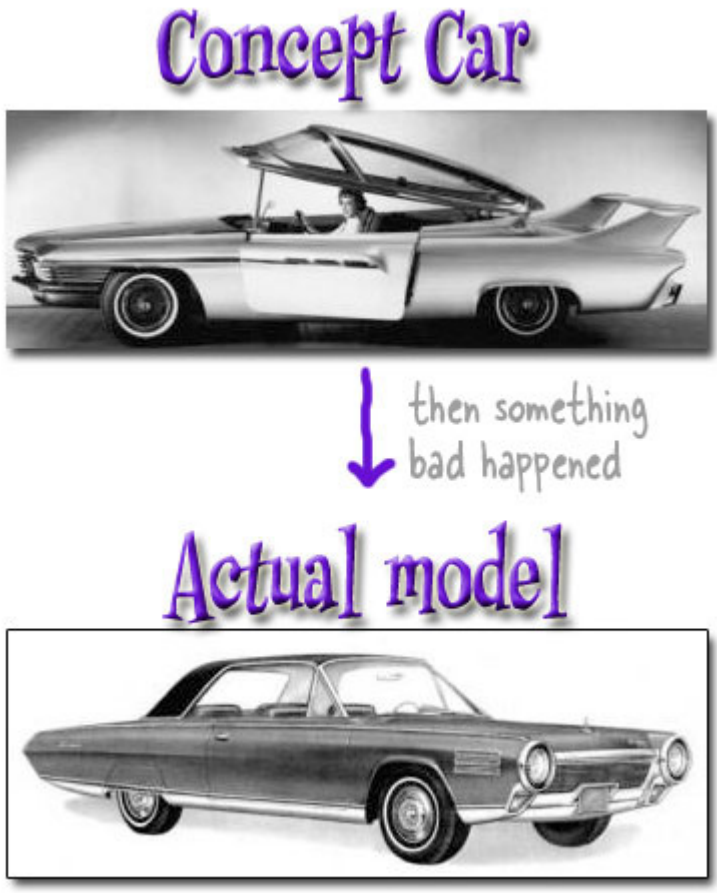
Sometimes we learn through struggle, but for the love of Smurfs, please think long and hard about *which* things the user *should* struggle with, and which things should get the hell out of his way.

I'm not saying that I know how to do this well either, but I can sure think of a zillion things I interact with where I think, "why on earth did they name that method in a way that suggests the thing you want but... does the opposite?" or "if they'd only flipped the direction of the switches, they'd be mapped perfectly to the direction of the thing they control (like the "up" switch moves things forward, and the "down" switch moves things back) or "if they didn't want you to sit on this thing, why'd they make it look and feel like a bench?"

http://headrush.typepad.com/creating_passionate_users/2005/10/making_happy_us.html

The Concept Carification effect

By Kathy Sierra on October 21, 2005



In the cover story in this week's Time magazine, Steve Jobs talks about "How Apple Does It." One of my favorite parts was this:

> *"Here's what you see at a lot of companies; you know how you see a show car and it's really cool, and then four years later you see the production car, and it sucks? And you go, What happened? They had it! They had it in the palm of their hands! They grabbed defeat from the jaws of victory!*
>
> *"What happened was, the designers came up with this really great idea. Then they take it to the engineers, and*

> *the engineers go, 'Nah, we can't do that. That's impossible,' And so it gets a lot worse. Then they take it to the manufacturing people, and they go, 'We can't build that!' And it gets a lot worse."*

Those car pictures show the before and after of the Chrysler "Turboflite" concept car. It's rather obvious that the "after" car, from 1965, looks nothing like the 1961 concept car. What happened?

And the same thing happens everywhere. There is a major computer book publisher (not O'Reilly, as will be obvious), where this guy (author/editor) had a wonderful concept for a new kind of computer book. Not like Head First, but every bit as unique and engaging. He had a vision, a manifesto even (I don't want to link to it or mention his name because I don't want to get anyone in trouble here). Authors were excited, people were on board, and the first book began production. You know how the story turns out, since it's the same story that plays out all too often... the very thing Jobs described. The production people started saying, "Oh, we can't do THAT..." and the resistance piled up until the book was released looking virtually like every other book, save a few fonts and a very weak theme. The guy with the original vision was disheartened. One of the other original champions of the project left the company, partly as a result of watching this concept have the life and sharp edges sucked out of it.

One of the things we loved about O'Reilly is that they said, "Yes, do it ALL." The Head First format is virtually identical to the concept Bert and I built in the original proposal. No edges were smoothed. Nobody said "we can't do that."

Obviously there are a zillion reasons why wild-ass concepts can't (and shouldn't) find their way into final production, but how many of those reasons are truly *valid*? When people say, "We can't afford to do it that way..." we should always ask, "Can't... or Don't Want To?" followed by, "Can we afford *not* to?"

If being remarkable is one of the only ways we can hope to compete in a world where *everything* has a ton of competition...

Of course, the article goes on to talk about how Bill Gates has "kicked the bits" out of Apple, proving that there IS another way, and that this way can be more successful. Which leads to my REAL favorite part of the article:

> *"Jobs doesn't care just about winning. He's willing to lose... He's just not willing to be lame, and that may, increasingly, be the winning approach."*

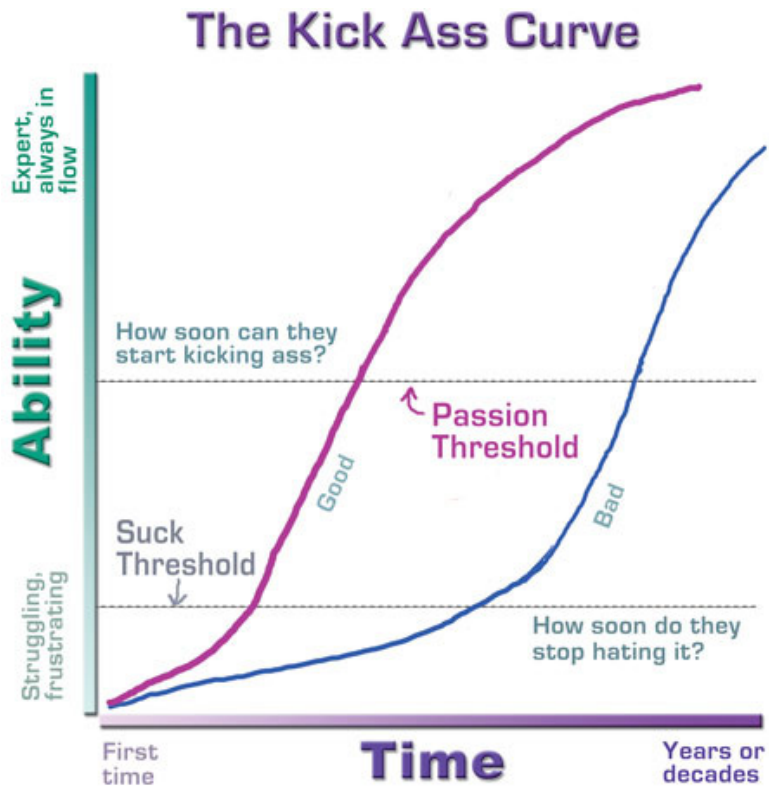
We have to keep fighting the Concept Carification effect, to keep at least some of our ideas alive, sharp edges intact. This is not an easy battle, since it involves separating the crap ideas from the brilliant concepts, with NO evidence. After all, most revolutionary concepts do NOT come directly from what users ask for. That's where we need to have faith. Yes, there are a ton of crap things out there that *should've* stayed in the concept stage, but if that's the price to pay for a world in which not *everything* is morphed into a nice safe incremental release, it's so worth it.

So have faith. When you're really really on to something magical, you can guarantee there will be devil's advocates, naysayers, and viscious critics every step of the way. Yes, sometimes those critics will be right, but if we aren't brave enough to fight through it when nobody knows *for certain*, then *everything* good will be stuck in the concept stage, and we'll be left with... all of the boring, undifferentiated, or lame products we have now.

http://headrush.typepad.com/creating_passionate_users/2005/10/the_concept_car.html

Attenuation and the suck threshold

By Kathy Sierra on October 25, 2005



How long do your users spend in the "I suck" (or "this product sucks") zone? Once they've crossed the *suck threshold*, how long does it take before they start to feel like they kick ass? Both of those thresholds are key milestones on a users path to passion, and it's often the case that he-who-gets-his-users-there-first wins.

Our O'Reilly editor Mike Loukides says our goal -- whether it's for product design or writing a tech book -- should be to focus on answering this question:

What is the minimum threshold at which the user can be creative?

Followed by:

Do whatever it takes to help them get there quickly.

And by "creative", he doesn't mean "be artistic". He means, "be able to apply the tool or knowledge or skill to do something useful or fun that *they* find meaningful or interesting." A long learning curve before true mastery is achieved is not the problem. The *real* problem is when there's a long learning curve just to get past the "I suck" (or, "this product sucks") zone, and a long curve before crossing the "Hey, I'm actually starting to kick ass at this!" threshold.

For most of us, our user wants to use our tools (software, books, sermons, screwdrivers, saddle, music) to *do* something *else* (collaborate electronically, learn, find inspiration, build a deck, ride a horse, dance). So we try to think about the *thing* they want to do, and how quickly we can get them through those two thresholds:

1) The *suck* threshold

The point at which they stop hating you (your company), the activity itself, or their complete inability to do anything useful.

2) The *passion* threshold

The point at which they start feeling like they kick ass. While passion is not a *guarantee* at this point, the chances of someone becoming passionate *before* this are slim.

And it's not *always* about the product--sometimes it's all about framing, documentation, and learning. It's about [straps self into buzzword appreciation chair] ***attenuation***. Turning *down* the gain. Narrowing. *Focusing*.

Or as O'Reilly's Rael Dornfest puts it:

"...bandwidth continues to broaden, cycles are going spare, storage grows ever larger and cheaper, and content keeps pouring from the fire hose. No longer constrained by any virtual limits, we're feeling the effects of this flood of digital assets."

It's no longer about generating digital data--we have more than enough already. The challenge is now: How do we visualize the data, filter it, remix it, and access it in ways meaningful to us?

In many subtle and not-so-subtle ways we're seeing user experience and design returning to software.

What developments in UI and HCI design promise to empower users rather than confuse and overwhelm them?"

There are so many opportunities. Raise your hand if you've been feeling overwhelmed with the pressure to keep up. Nod knowingly if you've ever said or thought anything like:

"They released a new rev *again*? Oh. Great. I guess I know how *I'm* spending my next few weekends..."

"Is there NO FRICKIN' LIMIT to what they'll add to these APIs?"

"Don't you DARE throw out that stack of journals, magazine articles, web printouts, partly-read books, and blogs. I really *am* going to get to them."

"All I did was take a single wifi-free week's vacation, and now I have 19,343 emails and at least 600 posts in my RSS reader I have to catch up on..."

"Why oh why didn't I become a plumber? Not *scalable*, sure, but also not *outsourceable*. And the domain knowledge is fairly stable... unlike my CS degree... [begins to laugh hysterically and inappropriately]".

"I realize this product went through beta, but seriously, did they watch any *real* humans to try to use this interface?"

Yes, there are so many opportunities. Anyone who can help *attentuate* the firehose in some way is a hero to those who are drowning.

And we can do it in so many different ways.

We can do it with "less is less" products (championed valiantly by the 37 Signals folks).

We can do it with better tutorials, reference materials, and learning experiences.

We can do it with better design.

We can do it with filters. Or maybe *lenses*.

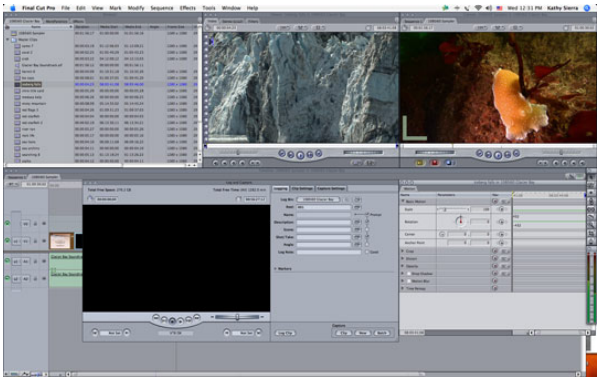
Remember, this is *not* about how long it takes to truly become an expert. In fact, where there is real passion there is *always* continuous learning and challenges in whatever it is the person is passionate about whether it's conversational Klingon or digital video editing or snowboarding or meditation or being a

wine snob/expert. This is not about *dumbing down* to give users a nice (albeit false) sense of *self-esteem*. This is about getting them to where they can actually *do something*.

Here are a few possibilities, but of course it depends greatly on the context of the tool (including expertise and expectations of the user):

1) Consider making different user profiles within the product itself, and allowing the user to choose a configuration for the interface that matches the user's goal and current level of skill and knowledge. Yes, that *could* mean having things like "advanced modes", and while that's a somewhat controversial usability practice, it definitely has a place, and can be done brilliantly for many (not all) products. But yes, it's about *attenuating* what a particular user is exposed to in the interface -- not *hiding* capabilities from them without their knowledge.

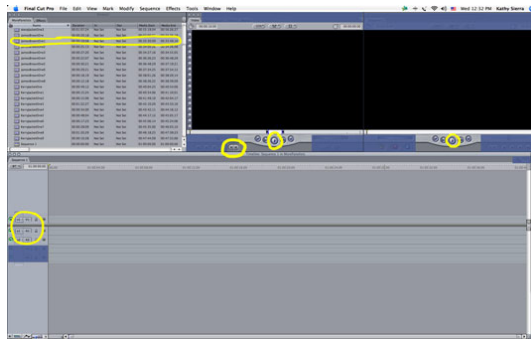
2) If you can't change the product, change the *documentation*. I've been working on and off on an intro to movie-making book to teach Final Cut Express and Final Cut Pro to mortals. The Final Cut interface is beyond overwhelming:



We could spend the first three chapters describing what each component of the interface is for. But that just keeps them in the suck zone longer, produces cognitive overload, and completely violates the "give them the minimum needed to start being creative." In other words, trying to explain the Final Cut interface only delays their ability to start doing the cool thing--editing video!

But we can attenuate the interface by postponing the "here's what every single one of the 230 things in the interface does..." (and that's just the part of the interface you can *see*...) and

instead focus their attention just on the six or less things they need to get in there and start editing video.



3) Use a spiral user experience model:

4) Create context-dependent FAQs and/or context-dependent "FDTs" (Frequently Done Things). At any given point in the use of a tool, what the user is most likely to do next is rarely random. By having some kind of reference or learning or embedded help that focuses on those can be a big help. Too many reference or training materials are organized by topic, when the user often has no idea what the topic IS. They want to *do* something, but they have no idea which part of the interface they're supposed to be looking up in the help file, because they don't know what comes next...

5) In training materials for the product, focus on getting the user *doing* something cool as early as possible! Don't bog them down with tons of theory before letting them apply what they've learned in some meaningful, interesting, and/or useful way. I've seen Java instructors make their students wait---*forever* before they students can actually write code, because the instructor believed they shouldn't be constructing code until they have a complete understanding.

That's not how humans work, and no, this is not a matter of "learning preferences" either. There may be some people who believe they are more *comfortable* learning the theory first, but that doesn't make it better learning -- even for those who believe they prefer it.

God knows that if we had to understand physics before we could ever start to walk... most of us would still not be walking.

6) Make sure there's a way for the user to *know* when they've crossed the thresholds. Sometimes the user is capable of doing more than they realize. Find a way to prove to them that they really *can* kick ass (or at least that they no longer suck). This must not be faked! This must be real, and again--not some attempt to dumb it down to make the user feel good. It may be that the user is doing something meaningful, that applies directly to what they *really* want to do, but the materials/instructor/app haven't made it clear enough how this seemingly simple thing relates or bridges to something that matters.

So remember...

The "time to stop sucking" and "time to first kick-ass" quotients are among the biggest advantages we have in a world where the competition is both fierce and plentiful. (And that's both market competition as well as competition for our scarce and precious brain/cognitive/attention bandwidth.) More importantly, it's a way in which we can make a positive impact on the lives of users.

And for more motivation, don't forget to read Information Anxiety.

Now where the hell did I put my GTD next action list...

http://headrush.typepad.com/creating_passionate_users/2005/10/getting_users_p.html

How to spend your marketing and ad budget

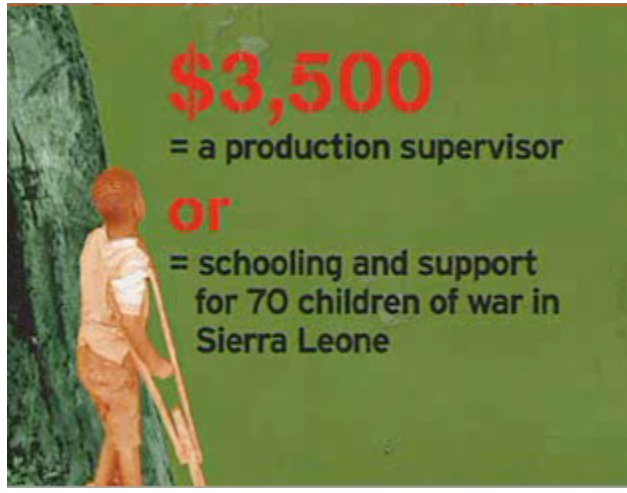
By Kathy Sierra on October 29, 2005

I've worked for companies that spent their entire ad and marketing budget on making their existing users deliriously happy. Let's say your marketing and/or ad budget doesn't have the same legs it used to, or that you've just decided to make a change. Or maybe you don't even *have* a marketing budget. Is there something you can do that might be more creative and, in many cases today, at least--if not more--effective?

These are off the top of my head and my usual disclaimers apply (doesn't work for everything, etc.), and I hope others will add *better* ideas.

Old Way	New Way
Hire a creative, award-winning *advertising* designer	Hire a creative, user-focused *product* designer
Focus groups	Field trips for employees (take them to where real customers are doing real work)
Print ads	Training articles
Hire a PR firm	Buy Typepad accounts for every employee in your company, and maybe some users too
Ads that talk about how you're *better* than the competition	Articles that talk about what you've *learned* from the competition
Buy the rights to a top-40 song for your commercials	Sponsor local musicians
Slick product brochures	Online case studies from real users that talk about how those users kick butt, not how YOU made it all possible...
Conference *sponsorships*	Conference *scholarships*
One 30-second commercial	*200* dairy cows for poverty-stricken families through Heifer International
Ads that imply you'll get laid if you drink/use this product...	Develop and support a socially-oriented online community and/or local user groups where people might get laid for *real*
Promotional newsletters	Online learning for users
Big ad campaign	Stop outsourcing your tech support and customer service
Product placements in a "fake" (TV, movie) world	Product placements in the "real" world, by donating samples to those who could benefit
Hire a word of mouth marketing firm	Create something worth talking about
Hire a "branding" expert	Hire, no--*promote from within*-- a "user happiness" expert
Run ads featuring "hot babes" in bikinis	Sponsor online fitness articles to help *users* get in bikini shape (or, your know, *swim trunk* shape)
Hire someone who knows how to create and spread compelling--but fake-- stories	Encourage employees and users to tell the *real* story, and spend the money internally to make sure the *true* story is a *good* one

The Sarah McLachlan music video for World on Fire puts a different spin on alternative uses for promotional budgets-they took nearly all of the $150,000 *production* budget for the music video and spent it on other things. I'm sure you've all seen it by now, but here are a few sample screens that come on (in between a few home-movie quality shots of Sarah playing her guitar):



Bonus: by putting the information into the video, they're also teaching fans/users the true costs of both the production of the video and things like the cost of educating a girl in Afghanistan.

Most of my suggestions aren't nearly as "worthy" as what they did with the World On Fire budget, but to a real user... having a better experience using the product or better yet--getting to the kick ass threshold more quickly--is still a pretty damn worthy cause.

And hey -- if you can help them get *laid* (by creating/supporting user groups and online communities where people often find meaningful and lasting relationships), then you've got something more powerful than all the "twins" ads money can buy. ;)

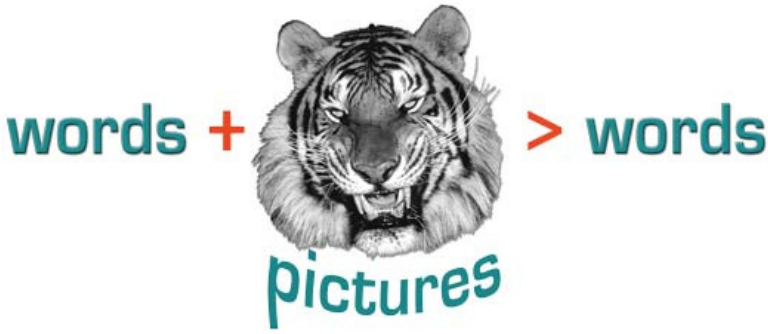
Please, add your ideas.

[Relevant links: Heifer International, World Changing, and Gaping Void)]

http://headrush.typepad.com/creating_passionate_users/2005/10/how_to_spend_yo.html

Words + pictures > words alone

By Kathy Sierra on October 31, 2005



How many appliances are visible in your kitchen? Don't read on until you have your answer.

If you're like most people, you took a mental visual walk through your kitchen, "looking for" appliances. "OK, next to the refrigerator on the right side there's the toaster... next to the coffee maker... the microwave is up *there*..."

We're visual creatures.

According to memory expert Kenneth Higbee, "The saying that a picture is worth a thousand words is usually applied to the effectiveness of a picture in *understanding* what was communicated; it may also apply to the effectiveness of a picture in remembering *what* was communicated."

One reason for this effect is that visual images are processed in two parts of the brain rather than just one. A pile of evidence supports that people learn more deeply from words with pictures than from words alone (Mayer, 1989b, Mayer and Gallini, 1990; Mayer, Bove, and others, 1996.), and overall, several studies combined have shown a median percentage gain of 89% effectiveness. *Pretty dramatic*. Some of the theory behind the gain you get when words and pictures are combined is that we use our brains more fully, processing the content more deeply, because we actively connect the words to the pictures. In other words, our brains work to make sense of the combined pictures and text, and that processing leads to more meaningful and memorable learning. That's the theory, anyway.

Perhaps more importantly, our target audience-the Sesame Street-->MTV-->XBox generation-has a highly developed visual sensitivity earlier generations lacked. In his book Digital Game-Based Learning, Marc Prensky claims, “In previous generations, graphics were generally illustrations, accompanying the text and providing elucidation. For today’s Games Generation, the relationship is almost completely reversed: the role of text is to elucidate something that was first experienced as an image.” He goes on to say, “They find it much more natural than their predecessors to begin with visuals and to mix text and graphics in a richly meaningful way.”

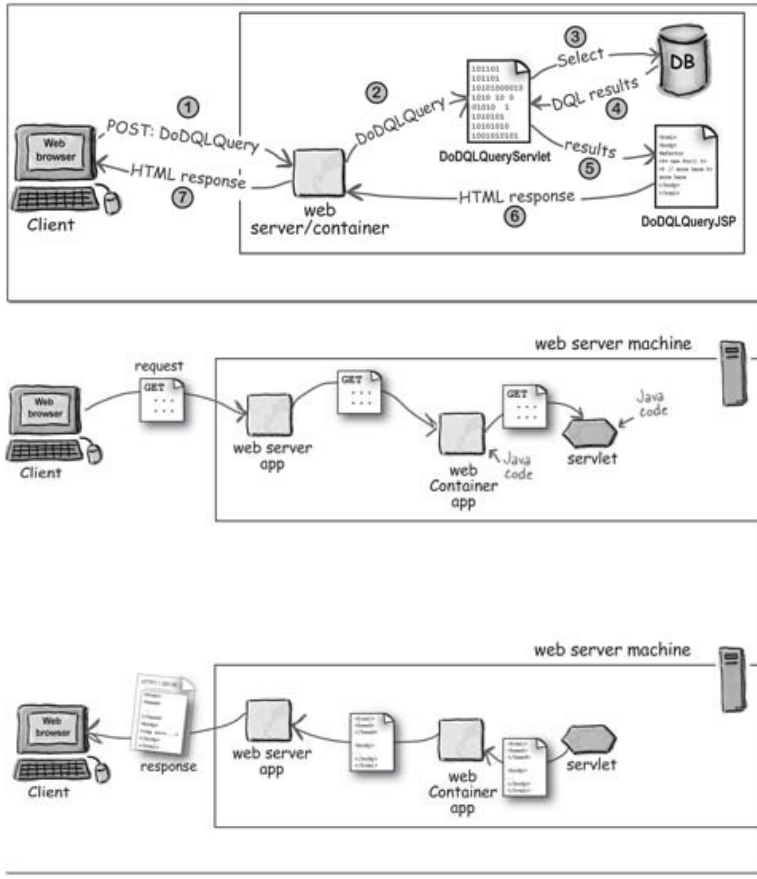
And when there are images, the text that goes with the images should be integrated *with* the pictures. In five different tests, one group was exposed to text placed *below* the illustration, while the second group was exposed to text placed *near* the illustration. Although both groups saw identical text and graphics (with the only difference being placement of the text), in all five studies the second group performed better on subsequent tests. When a reader has to keep switching between the graphic and its description, he has to work harder... on the wrong things. There’s only so much mental bandwidth in a reader’s brain, and [broken record and dead-obvious here] that bandwidth should be used for making sense of the actual topic, not for making sense of the *way the topic is presented*.

Tech/education publishers--pay attention here--the one thing that could make a huge difference is to switch from captions-*under*-pictures to captions-*within*-pictures. Yes, I've heard all the arguments for why this is difficult for production. But the potential gain is HUGE.

I've talked about this a lot before, but I've noticed some of my co-authors slipping a little on the graphics so this is a little reminder ;)

One of the main reason my cohorts and I are using graphics is so that the picture in the user's head more closely matches the picture we're trying to convey. If you use words alone, you have to be a *damn* good writer--much better than I am. Those who write with crystal clarity can describe something complex with a higher *chance* that the intended meaning makes it into the user's head, but there's still no guarantee -- AND -- using words alone isn't as effective for a *lot* of topics.

And even seemingly simple ideas can take a lot more *time* to convey if you don't use pictures. We value our reader's time tremendously, and that's a big part of why we are so graphic-heavy. I look at these two simple graphics and imagine how many paragraphs of words it would take to make sure the user "read" it the same way:



Given the potential for such dramatic gains, my co-authors and I keep wondering why the vast majority of adult technical materials have so few visuals. The arguments I hear are usually misconceptions, and fall into one of these:

1) Adults don't need pictures

2) Adults don't *want* pictures

3) Only "visual" learners need pictures

4) It takes a lot more work

For many, many, many topics, and many, many, many audiences--these notions are just *wrong*. Generating graphics can be more work, but you make it up in other ways. When I can generate a two-page spread describing a complicated server process, I just saved myself five or more pages of writing! (And the stress associated with trying to be certain my words describe the story in a way that causes the reader to form an accurate, vivid mental picture.)

All it takes is a little getting used to. I'm always amazed when teachers do eleborate white board drawings, but never put them in their books or articles. Or when engineers can do fabulous napkin drawings to explain things to colleagues, but never put them in their books or articles.

The one thing that makes a big difference for me in being able to create pictures: my wacom. I'd give up my iPod before my tablet. There, I said it.

(Of course, I have an emergency backup iPod)

http://headrush.typepad.com/creating_passionate_users/2005/10/words_pictures_.html

If your software was on a date...

By Kathy Sierra on November 4, 2005

How would your software (or product, service, book, cause, etc.) behave on a date? Perhaps the best model for software developers is the singles scene, so let's see how this time-tested dating advice for men might be applied to software:

What we want:

What we all too often get:

Dating Rules For Software

Look your best

You don't have to be the Brad Pitt of apps, but you should still make the *effort* to be pleasant looking. At the least, you should be *clean*. That whole "it's what's inside that counts" thing? It's true, but chemistry matters too, and we're genetically programmed to be *attracted* to attractive things. If nothing else, wearing your good shirt and combing your hair sends the signal that you *care*. That you bothered to take a shower before you showed up at our door, says something meaningful.

Be clean, be simple, keep the bling to an absolute minimum, and don't forget your mom's advice--"you never get a second chance to make a first impression."

Be fun. Don't be negative. Be the one others want to be around.

How do people feel when they're around you? Do they light up a little? Or do they feel inexplicably darker and less energetic when they spend time with you...

Hint: make a list of the apps, products, APIs, frameworks, etc. that make YOU happy. The ones that make you think, "this is awesome." Or better yet, the ones where you never think about them at all... because you're too busy *being* awesome doing the thing that led you to that tool in the first place.

Focus your energy on putting yourself on someone else's "makes me happy" list.

Be trustworthy and consistent.

There's a time and place for spontaneity, but we need to know we can count on you, no matter what. Make sure we can trust that when we click button A, thing B will happen. Every... single... time. And that it doesn't matter *when* we push it, or what you did before. Please, no unpredictable mood (or *mode*) swings.

If you use a particular pair of methods in your API, and then reuse those same names in another part of the API, make certain that they all behave in exactly the same way -- or at least exactly as you'd expect in that different context (terrible API violation of this: the ejbCreate() and ejbRemove() methods for entity vs. session beans in EJB).

Don't be fake.

Don't pretend to be something you're not. If part of your interface looks like it should do X, but does only Y (or worse, does X *plus* the recklessly dangerous Z), we may never trust you again. Don't try to be more than you are, and don't trick us into thinking you do one thing, when you actually do something completely different. Being simple and clean and *real* is far better than being a flashy fake.

Be polite, be helpful.

Don't dash off in the middle of dinner to run an errand, but if you *must*, at LEAST tell us how long you expect to be gone, so we'll have some idea of when to become concerned. An application that doesn't tell you what's going on is just *rude*. It's OK to offer tips... if we don't speak French, then by all means help us interpret the menu at that French restaurant.

Be forgiving.

We're not perfect. Sometimes we say or do stupid, wrong, or even dangerous things. Make it easy for us to recover and "save face", and we'll love you all the more. And the more you assume it was *your* fault, the better. Chances are, it *was*.

Be sensitive, be a good listener.

But not *over*-sensitive. Pay close attention to the subtle things; don't make us have to yell at you in order to get a reaction. Try to anticipate our needs, but don't make assumptions! We never said this would be easy... and yes, we're a bit high-maintenance, but worth it ;)

Don't assume I'm an expert.

You wouldn't expect that everyone you date will have studied human psychology, so you shouldn't expect a user to have read your manual cover to cover. Don't take us extreme helicopter skiing on our first date.

Be fun.

Not *funny*. Be fun in the way that a great game of chess is fun (but not funny). Life is too short (or too damn long? I can never remember which way that works) to spend time doing boring, tedious, frustrating work. The best dates of all are with those who can make even the most trivial, mundane things seem... engaging and interesting. Find out what part of this experience really *can* be interesting, and enhance that.

Don't assume there's no competition.

"There are plenty of fish in the sea" our mothers tell us when we're heartbroken at 15. NEVER take the attention you're getting now for granted. Even if you think you have a vendor-lock. Even if you think they'll stay with you simply because the cost of switching to someone else is too great... There is *always* someone potentially better, and *real* loyalty can't be bought. "Frequent Buyer" points might make it *look* like we're loyal, but underneath we're just waiting for the right opportunity to dump you. Don't mistake current participation for long-term loyalty.

Check your ego with the valet parking attendant.

You might be the best at what you do... for now (reread the previous tip)... but that's no excuse for treating those you date like idiots. And we really don't appreciate hearing you diss the competition, either. A little humility goes a very long way.

Married people really DO have more sex.

No matter how fun the one-night stands appear, they're ultimately empty and unsatisfying. Go for the long-term commitment. Be in this for a lasting relationship. If you really really care, we'll know, and we'll be willing to forgive *you* when you screw up--as you always will.

Any other dating tips for software or other product developers? Or examples of those who'd score on a second date as opposed to... those who'll never *get* that second date? If products were a potential mate, which one would you give *your* phone number to? Me? I'd sleep with Adobe InDesign in a *heartbeat*.

http://headrush.typepad.com/creating_passionate_users/2005/11/if_your_softwar.html

When clients (and bosses) go bad...

By Kathy Sierra on November 7, 2005

What's it like to work at your company? Is anything beyond 8 hours a Big Exception, or does leaving at 5 PM evoke the "working a half-day again?" crack: In all my various jobs, from independent contractor to start-up employee to one of the thousands at the big monolithic tech company, I've worked in every conceivable tech scenario. But the *worst* are the ones that become slaves to their clients--often driven by the fear of losing one.

And fear leads to underbidding. And underbidding leads to: pulling all-nighters to make an impossible deadline on too few

resources. (And the dark side is in there somewhere.) I love users, but as they say on the plane: **you must put on your own oxygen mask first.** You can't take take care of *users* if you or your employees are exhausted and stressed. On Maslow's hierarchy, empathizing with users comes *after* sleep.

I've seen too many startups begin with the promise of freedom, passion, and good intentions--only to end up exchanging one kind of "prison" (working for demanding *bosses*) for another--working for overly demanding *clients*. I've seen some companies become slaves to the client's whims because we had "too many eggs in one basket", allowing clients to exploit the fact that you need THEM much more than they need YOU. And then there's the company that's looking to sell or go public, promising everyone that "if you just work really hard for the next three years, we'll all make a ton of money." (Assuming you lived through the process.)

So why is it that some companies have such an unhealthy relationship with their clients (which means an equally unhealthy relationship with their employees)? It's not like we won't work our butts off for the right reasons, but when it becomes standard to put in 10 hour days and work at least a part of every weekend, simply to keep up with the insane deadlines, our creative energy drops to zero. You're getting *labor* but no passion. *Productivity* with no creativity. And we'll switch jobs in a nanosecond if we get the chance. The worst is when you're expected to work like a dog and the culture discourages complaining or even *questioning*.

There is at least one industry that has the right idea about the times when you have to put in long hours... *Hollywood.* Or at least the parts I worked in, which were post-production, games, advertising, and marketing. In the motion picture world, where your project manager is called a "producer", when they asked you to work the long hours, at least you were treated like a tempermental star who needed to be pampered ;)

In the Hollywood model, even the programmers usually got the diva treatment when the company really needed you to stay late. At most tech companies, on the other hand, when you have to work late, your manager springs for pizza, vending machine soft drinks, and maybe take-out burritos. But the Hollywood firms I worked for usually passed around the gourmet restaurant menus, if you worked beyond 7 PM. More surprising, they would

also do whatever was necessary to make my *daughter* happy about my working late (as a single parent). They'd "send a car" to her school, bring her in to the facility where we'd have dinner together (whatever she wanted), take her to the ultra-luxury private screening theater (telling her how "Steven Spielberg sat right in that same chair yesterday...") and let her watch previews and early cuts from movies that *nobody* outside the studio had ever seen. (One of my all-time favorite employers then was BLT, a motion picture advertising agency, but I also loved working for the now-defunct kid's game division of Virgin--Virgin Sound and Vision, under the best creative producer/manager I've ever worked for, Tom Mott)

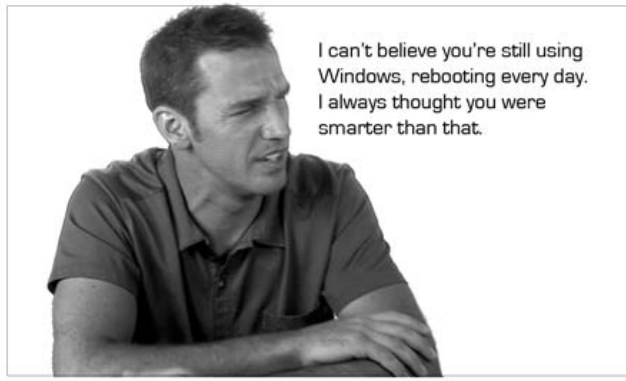
Ideally, a better way to treat employees is to try to avoid putting them in that spot *ever*. But things happen, and we understand that there will be times when we (the workers) just can't get it done during normal business hours, and the schedule just can't slip. Still, the difference between being *expected* to put in the long hours and being *worshipped* for doing it cannot be overstated. If we want to make happy users, *we* have to be happy. Our employers/managers/clients need to accept that, and act accordingly. If you're making us work late all the time because of lousy management, that's inexcusable. If you're making us work late because you're *greedy* and just want as much business as you can (im)possibly handle, that's inexcusable. But if you need us to work late because things happened that nobody predicted, or because this demo means something drastically important to the company, for which we will also be rewarded... then sure, we'll be willing to pitch in. But spend the extra few bucks to treat *us* as well as your clients. You should be wining and dining *us*, not them, when you're asking so much from us.

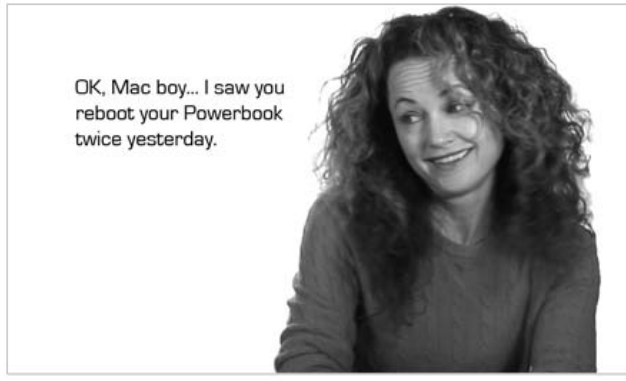
And as the tech employment market starts to tick up ever so slightly, it's becoming less and less of an "employer's market" again. I don't care about the Aeron chair, but I *do* care about having a life beyond work. If you can't make your business model work without promising your clients a miracle (which *we're* expected to pull off), change your business model! And when you DO ask us to go our ass off again, a little worshipping goes a long way ;)

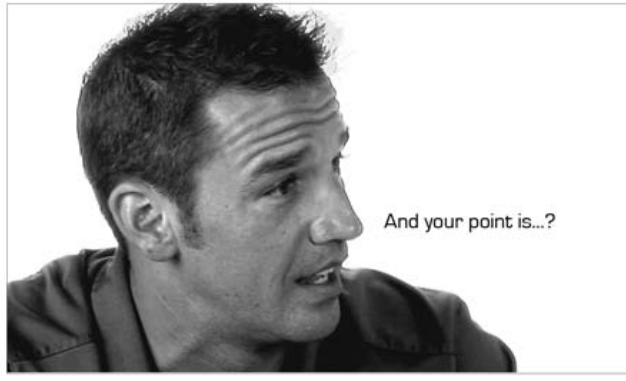
http://headrush.typepad.com/creating_passionate_users/2005/11/whos_in_control.html

Passion is blind

By Kathy Sierra on November 11, 2005







Forgiveness is relative. We diss *Windows* with impunity, but when our *Mac* does the same thing, well, *geeez* nobody's perfect. When Clinton lied, US conservatives were morally outraged. When one of their *own* lies, "This is just a partisan stunt... perjury is a technicality." When Office crashes, I *swear* at it. When InDesign crashes, I *empathize* with it.

Having passionate users is almost like a get-out-of-jail-free card.

I say *almost*, because if you *abuse* your position as the object of someone's passion, they'll eventually figure it out, and the sense of betrayal will make them angrier about your transgression (crashing, lying, running slow, being incomplete, etc.) than if they'd never loved you at all.

But there's no getting around it--we *all* have double standards. We are *all* cutting *one* side some slack while holding the *other* to our ruthless, concrete expectations. And of course we *will* all screw up. We *aren't* perfect. Neither is our software, our hardware, our service, our support, our employees, our policies, our products and services and ideas. But that's the beauty of passion--if you can inspire it, by helping your user kick ass--they WILL cut you some slack. *They'll forgive you when you screw up.*

And even their very definition of "screw up" is fluid. Like I said, when I reboot my Mac, it isn't Apple that screwed up. It's either *me* (what did I expect trying to hook those three things up together?) or just the nature of doing anything so sophisticated, superior, and cutting edge. A small, small price to pay. When I have to reboot my *Windows* machine, come on... rebooting is such a perfect metaphor for everything that's just so *wrong* with Microsoft.

True Apple fans know that the Nano screen only *appears* to have a problem with scratches because:

A) The screen scratches... DUH! The new users are just too stupid to take proper care of it.

and

B) The normal to-be-expected scratches are simply more *noticeable* now because of the increased screen resolution. The perceived "scratch problem" is actually an artifact of the Nano's superiority. A feature, not a bug.

Most of us stroll happily along never acknowledging the double standards we apply. We don't recognize that our specific level of forgiveness (and indeed, what we even decide *needs* forgiveness) is based almost entirely on whether we *love* something (we'll forgive almost *anything*), *hate* something (we'll forgive *nothing*), or don't care about it at all (we'll forgive based on whatever seems "reasonable").

But sometimes our double-standards bite us in the ass and we're forced to face it, as Phil Ringnalda did a few months back. When O'Reilly appeared to have search-engine-gaming ads, Phil slammed him in this blog entry. But when his friend Shelley Powers does it, the conversation got very interesting. It was fun (and impressive) to see Phil acknowledge and wrestle with the ambiguity of it all. A couple quotes from the comments:

"Unfortunately, I can't extend that absolution to you, and deny it to Tim O'Reilly...I don't like this answer either. Isn't there one where I can get back up on my high horse, and take a nice absolute moral position?"

I love the discussions that force us into grey, fuzzy, squirming positions where we must "hold two opposing thoughts simultaneously." But the point of my post is this -- wouldn't you rather be the one most likely to be forgiven than the one who can never "catch a break"? And again, I'm not talking about areas where you really *do* have serious problems that you'd rather sweet-talk your way out of than fix. I'm talking about the inevitable problems you just can't avoid. The "stuff just happens" events.

So, we have to ask ourselves... what can we do to put ourselves on the side of forgiveness? What can we do to help protect us from the times when we *will* screw up? What would it take in our product, company, service, whatever -- to get users to have a glass-half-full attitude about whatever it is we do? If "rebooting" is a metaphor, I'd rather be Apple than Microsoft.

(And that's another question to ponder... why *are* we so willing to diss Microsoft yet give Apple a break for some of the same things? More importantly, what--if anything--could Microsoft do to turn that around?)

http://headrush.typepad.com/creating_passionate_users/2005/11/passion_is_blin.html

How to come up with Breakthrough Ideas

By Kathy Sierra on November 30, 2005

Brilliant, wildly creative people can pull breakaway ideas from thin air. The rest of us need tools. EQing is the tool we used to design the Head First series, and we've been using it ever since.

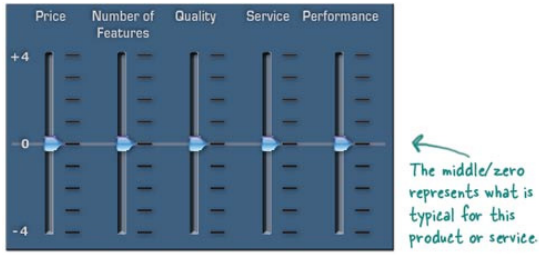
Bert, Eric, and I are all audio freaks -- we lust after the giant mixing boards at live shows, confident (and delusional) that we could do it *so* much better, if only we could get our hands on those sliders. So, an audio equalizer was a natural metaphor for us, and this is my first attempt to explain how we use the concept of EQing to brainstorm new designs.

(If you aren't familiar with how audio equalizers work, click here for a nano review.)

In our EQ model, when all the sliders are in the zero/middle position, this represents "the norm" for whatever that product, service, industry typically does. In other words, a slider turned down to -4 means that it's way below the norm, and a slider at +4 means way above the norm. But the slider says nothing about the actual absolute value of whatever that slider is for. (The number "4" means nothing -- it's just an arbitrary number that matches the graphic -- I could just as easily used "1" or "10" or "42".)I'll start with a simple example to give you a feel for it, before working our way to the good stuff. You'll have to read to the end to get to the "breakthrough" part. ;)

Example One: Typical, non-breakthrough EQing

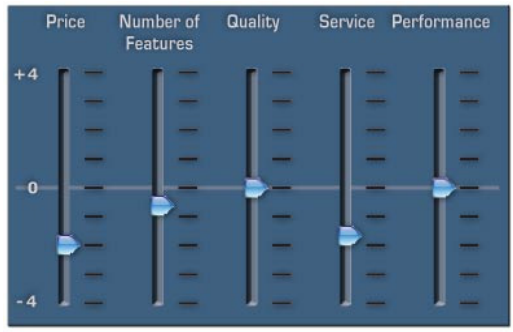
Equalizer for a typical computer hardware product:



In the graphic above, the zero represents what is average/typical for that kind of product. Assume there's more than one source for the product... The problem with computer hardware (as with

so many other things) is that there's *too much competition*. Too many companies clawing and biting for their slice of the market for that product.

How do they compete with one another? The typical--*and usually worst*--way to gain an edge is by tweaking one or more of the standard sliders. For example, a low-cost computer's EQ might look like this, relative to the norm:



This product sacrifices features and service in order to lower the price, but this works only while:

A) The features and service aren't essential to a large enough part of the market

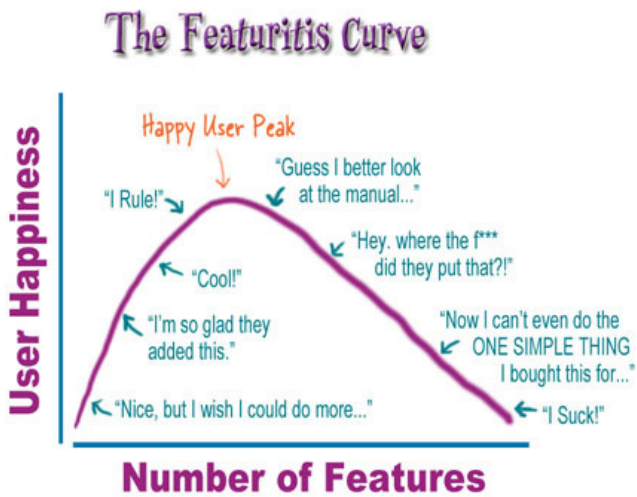
B) They are the only vendor doing it this way (i.e. the only low-cost vendor with this kind of EQ)

But if this particular EQ is successful, other companies (new and existing) will eventually make the same tweaks until the lower prices simply become The New Normal. In other words, the zero EQ point for the price and feature sliders now represent a lower *absolute* number, and now there's a viscious downward spiral of competition at the low end...

Another company may take the price slider down, but keep all the features at the middle/zero point (i.e. the norm). But then they're probably cutting somewhere else -- either by using cheaper or less-skilled employees, cutting employee benefits, and/or cutting customer service. A company might do just fine for a time, but quality will slip eventually, and customer happiness will drop. From a systems thinking perspective, this is not a sustainable strategy.

The big point is this: trying to compete with existing products, services, or ideas by tuning the SAME set of sliders everyone *else* uses is a painful path. The breathrough ideas usually come from adding ***new*** sliders! There are exceptions, though -- if you tune an existing slider in a dramatic or counterintuitive way, you might end up with a breakthrough edge, at least temporarily.

37signals, the folks behind the wildly popular Basecamp and Backpack, did that when they tuned the features slider way, way, way down. Their art is in knowing *which* features to leave out, of course. But turning down the features slider wasn't really their goal. **The goal (I think) was User Bliss**, an entirely *new* slider, and one that was partially tied (inversely) to the Num Of Features slider. Turning down the features turned out to be one of the most important ways to achieve User Bliss.

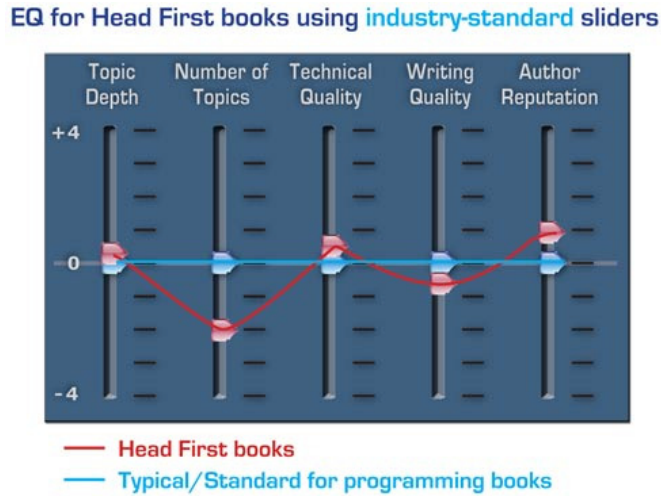


Knowing *which* sliders to include is just as important as knowing how to EQ them. When we say "typical sliders for this product type", it means that these sliders reflect the areas of focus for companies who make those products. And that's the problem. ***Breakthrough ideas come from "thinking outside the sliders."*** ;)

Example Two: Beginning Breakthrough EQing: adding *new* sliders

When we set out to design a new computer book series, we looked at the typical industry-standard sliders for a tech book. Most computer book formats make adjustments within these main sliders, although some topic categories (like digital photography or web design books) add a slider for color, another might use a slider for including a CD-ROM, etc.

When we thought about our first book--on Java--we wondered how the hell could we compete with 2000+ Java books still on the market? No amount of EQing the typical sliders would give us a breakthrough book. In fact, when we look at the EQ for our book against the typical sliders, the Head First format doesn't look good at all:



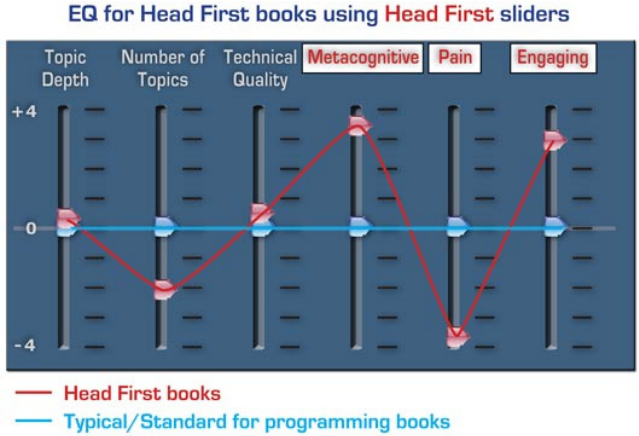
We needed to add new sliders.

Our first new slider was "Pain". We wanted to reduce the pain associated and assumed with learning a tough tech topic. Some books do this by lowering the number of topic and topic depth. To reduce pain, you *could* simply make an easier book. But that

was not our goal. We didn't want to simplify or "dumb down" the topic--we wanted to make a tough topic *less painful*. (A big distinction for us.)

We added two other related sliders as well ("metacognitive" and "engaging"), and the combination of the new sliders plus the tweaks we made to the standard sliders gave us our edge. Again, we certainly aren't the only ones to care about these things--but it DOES mean that we considered them far more important to *users* than had been previously assumed by most of the books in our category.

And here's where we started to use the EQ-it to model our ideas... the graphic below shows our our book compared to the norm. You can see that when we added the new sliders, we developed a very different EQ pattern from the other books. On the *new* sliders, we went way above the norm in "metacognitive" and "engaging". We reduced the "pain" slider. We also reduced the number of topics, but did not reduce topic depth.



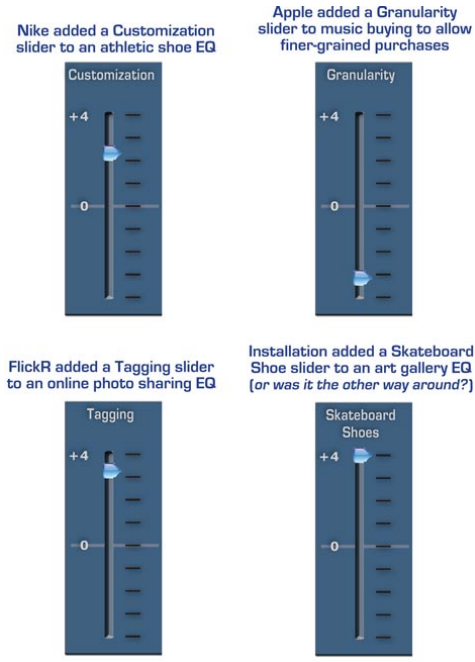
Adding *new* sliders--especially the "engaging" slider--was the key. But the sliders we added were not very innovative, they were simply **NOT TYPICAL** for a tech book. You can often make a breakthrough product simply by changing the weighting (i.e. adjusting the slider) of things that competitors have taken for granted. Look at things your competitors don't consider important enough to warrant a slider, and imagine what would happen if you promoted some of those things to first-class slider citizens and tuned them dramatically up or down. If--*and this is a big if*--these new sliders reflect previously unsatisfied user desires, you might have your edge.

But... adding sliders for things that the competition did not consider important (or mutable) enough to warrant a slider is just a warm-up. Rather than simply adding sliders for things the competition is *already* doing (but hadn't considered tweaking), why not add sliders for things the competition never dreamed of? This is where the biggest breakthroughs happen--when you add sliders that make others say, "WTF?"

For example:

Netflix added several new sliders not previously associated with video rental. Nike added customization, a slider not previously associated with athletic shoes. FlickR added tagging, a slider not previously associated with online photo sharing. (And they quickly ended up with a community slider as well.) Apple's iTunes added an interesting slider--"granularity"--to the purchase of music. Before iTunes, the "atomic unit" of music was usually a CD. You had to buy the whole thing even if half the songs sucked. By adding a slider for granularity--and tuning it way down below the norm--iTunes added true user value.

Here are some examples of new sliders companies have added. Could you add any of these to what *you* do?



Coming up with ideas using EQ modeling

Least effective way:

- Figure out what the existing sliders are for this product or service, and change the value of one or more sliders. This is how most companies compete, and it's usually the most painful--the constant struggle to reduce price, add features, whatever it takes to stay one step ahead of the competition.

More effective:

- Tune one or more of the typical sliders in an extremely dramatic way. For example, instead of *cutting* the price, make the product *free*. But this usually means you end up creating one or more *new* sliders for whatever business model allows you to make this drastic change.

***Much* more effective:**

- Add new sliders for things that competitors have taken for granted, and haven't been competing on. In other words, dramatically change the weighting of things the competition had not considered changing. Example: our books.

***Most* effective** (for breakthrough ideas, not always the *best* ideas ;)

-Add wildly new sliders for things nobody in that industry had considered.

Note that what's "wildly new" for one type of product or service might be standard/typical for another. A Customization slider, for example, would not be unusual for a wedding cake bakery, but was very unusual for athletic shoes.

Tips for finding NEW sliders

1) Borrow sliders from an entirely different product or service type. Customization, Subscription, Home Delivery, Entertainment, etc. -- things that make sense in *some* domains but have never been used on your product or service.

2) Look at the conventional wisdom--things everybody offering that product or service takes for granted--and see if you can tune a slider the others consider immutable (or unimportant).

3) Randomly add sliders and play what-if brainstorming games.

4) Ask *users* to come up with sliders. This is *more* effective (and similar to #3) if you ask users from a different domain! Remember, directly asking users what they want rarely leads to breakthrough ideas. Breakthrough ideas are, by definition, things nobody has yet imagined, but which users find compellling.

One final example:

Imagine an art gallery. Now imagine a skateboarding shoe store. Now smush those together into an art gallery/skateboard shoe store. That describes Installation, an imaginative and uber-cool store here in Boulder. While nearly ALL skateboard shops are pretty damn cool, the idea of "Gallery" as a slider was a unique idea. Or you could flip it -- the idea of adding "Shoes" (let alone skateboard shoes) as a slider to an Art Gallery equalizer is pretty strange.

I know I don't need to say it, but for disclaimer purposes I will--adding weird sliders *just to add sliders and be novel* isn't the point. The goal is to add sliders that turn out to be **really important to users**. And I say "turn out to be", because the most daring breakthrough products and ideas are rarely driven by user requests.

Typical art gallery goers weren't saying, "yeah, but what we REALLY want is for you to let us come here to buy high-end skate shoes." And skateboard shoe buyers (Skyler and I are both in that category) weren't asking for a gallery setting. But it turns out that the ambience and sheer creativity of the place IS compelling. It's worth the non-discount price of the cool shoes just to come away from your shoe-shopping experience inspired as you would be from, well, an art gallery visit.

It's too early to tell, though, if Installation's unique combination of sliders will be successful. Once the novelty wears off, will people still go back there for their shoes? I will, but I'm not their demographic, being about, oh, twice the age of today's young skaters. Installation is doing a lot more than just throwing art on the wall--but I'll say more about *that* in my next little post.

So, as an exercise, I'd like to challenge you to think about ways you can add sliders. One of the best exercises is to reverse-engineer other breakthrough products and draw out their equalizer -- showing how they differ from what is typical and standard. At the bottom of this post, I've included the blank

equalizer JPEG, plus the sliders, if you'd like to try making your own. Or just draw one on a whiteboard and scan it. If you create a blog entry with an equalizer of an existing product or service (or a wild-ass idea for something new), I'll link to it--as long as you have comments open ;)

Keep in mind that there is no one correct equalizer description of a particular product, service, or idea. It all depends on your perspective, and you might have an equalizer, for example, devoted solely to the customer service aspects of a product or service. And there's a fractally component here as well. With our "metacognitive" slider, for example, I have an entirely separate equalizer JUST for EQing the various metacognitive techniques we use. Our newest book design, for example, is brain-friendly, but in a profoundly different way than the Head First books, because we made a completely different EQ of the brain-friendly elements-- turning some way down, bumping some way up, and adding a few new sliders.

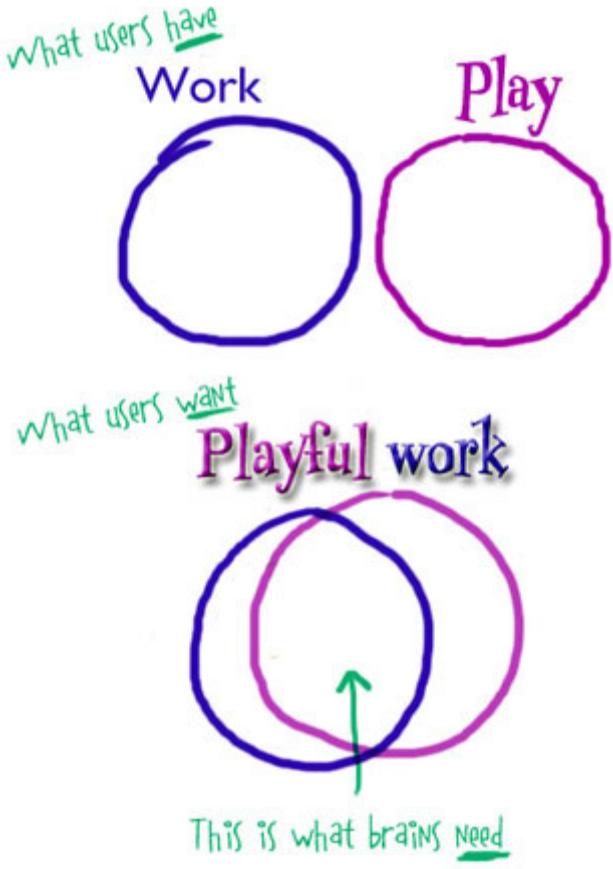
Above all, have fun!

FYI: This EQ modeling is fairly similar to a technique known as The Blue Ocean Strategy, a book I recommend (although it's a bit on the business school/academic side for me, but it's got a lot of great info). Their subtitle says it all: "How to create uncontested market space and make competition irrelevant"

http://headrush.typepad.com/creating_passionate_users/2005/11/how_to_come_up_.html

Never Underestimate the Power of Fun

By Kathy Sierra on December 1, 2005



There's been a *very* active discussion among Sun's "Java Champions" (Sun's program for external Java developer/evangelists, of which I'm a member) talking about why the Java programming language has lost some luster and Ruby is getting all the coder love.

I commented to the group, "Never underestimate the power of fun." We can talk all day about how much more powerful Java is (true), how it has orders of magnitude more resources, APIs, frameworks, etc. (true), and how it's been used to solve some of

the most complex, highly-scaled systems imaginable (true, think *Orbitz*). But in the end, even *programmers* are still human.

And human (*mammal*) brains are tuned for *play*. Evolution favored those with a high play drive, because play=learning, play=practice, and learning/practice=survival. Play--and laughter--sends a signal to the brain that "this is good, and it *matters*", which is why we're often more likely to remember especially funny things than neutral or annoying things.

But this isn't a post about programming--it's a follow-on to my previous post on brainstorming with EQ sliders. It's about **adding a slider for humor or fun**, where it isn't necessarily expected.

One innovative city government did just that, and they're earning a reputation as a city that does things governments "just don't DO." If you can imagine a city government trying to help its citizens "kick ass", well, that's what Bryan, Texas appears to be doing.

Jay G. Socol, the city's Public Information Officer, explained it to me:

> *"All cities with populations of 3,000 or more are required by law to publish and distribute (yawn) drinking water quality reports. Cities hate doing them, and residents throw them away, often without a glance. Bo-ring.*
>
> *This year, we developed a mid-year calendar (who sends out a calendar in July???) that featured fun photos of out Water Services employees -- guys who never, ever get the spotlight or recognition. Yet they perform a mega-essential service. So imagine what people thought when they opened their mailboxes and found a calendar that has a cover photo of water guys synchronized swimming in a pool?*

> *You have to open something like that and see what's inside. And that's exactly what people did. They loved the concept, the fun photos and...they even read the boring legal information. I move around town and quite often see the calendars hanging in businesses or in homes. We've had tons of requests for this thing -- even from other cities. And I believe it will change the way cities approach publications like this because they never knew that they could show a sense of humor."*

(You can see some sample pages of the calendar on the Bryan government page.)

Seeing this made me think about ways to add a "fun" slider to places where fun has diminished over time (like with Java), or into places it's never been (city government). (I recommend the A Smile in the Mind book for inspiration.)

Way to go Jay Socol! I hope the next city I live in has a government with this much spirit.

http://headrush.typepad.com/creating_passionate_users/2005/12/never_understi.html

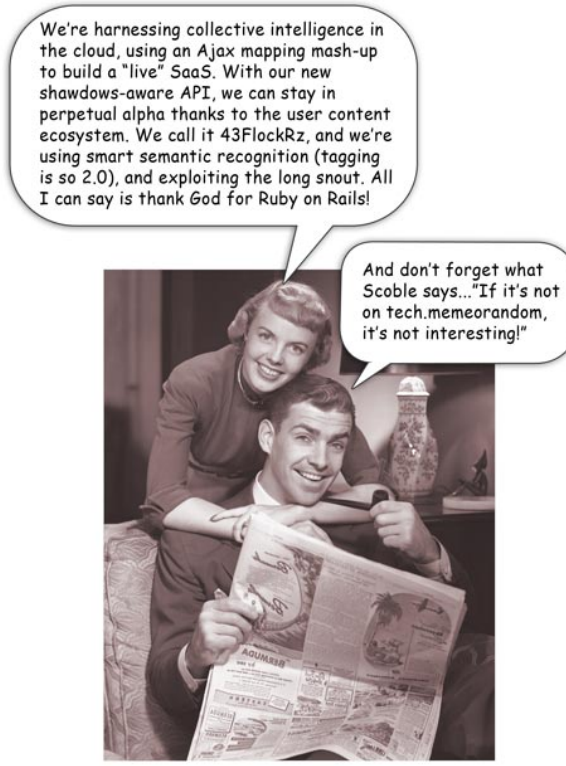
Have you updated your buzzwords?

By Kathy Sierra on December 4, 2005

You may *think* you're appropriately buzzword-compliant, but this is internet time, baby, and last month's 2.0 buzzwords are outdated (but not in that retro-cool way). If you're doing a VC pitch and building to flip, it's crucial that you sound as current as possible. Listen and learn:

Web 2.0

Web 2.01



Bonus points: mentioning "yellow fade" and "attenuation" in the same sentence could be worth an extra million. *This* month, anyway.

Grand prize: just as the hipster PDA made low-tech cool again, feigning ignorance of Web 2.x is the new black.

My take: Some of the *coolest* people have no frickin' clue what these buzzwords mean, and don't *care*. They aren't building to flip, they're building to *engage and inspire*.

That doesn't mean throwing the user-driven baby out with the Web 2.x bathwater--there's some really useful stuff in there (people genuinely LOVE FlickR and del.icio.us, for example). But these Web 2.x buzzwords are more *technology* and *business-model* focused than *user* focused, and that's a recipe for building things that meet the checklist but fail the users.

Where there is passion, there are users kicking ass. If we want to build Web 2.x and beyond, we should be thinking less about

how to upgrade our *technology*, and more about how to help upgrade our *users' brains, bodies, and spirits*. Yes, I realize that many--even most--of the Web 2.x buzzwords can *lead* to meaningful (in some cases profound) benefits for the users. But the *emphasis* still feels technology-driven, not user-kicking-ass-driven.

Why not rewrite these buzzwords in terms of what they mean for users?

For example, we know that "harnessing collective intelligence" is good... but why? I don't necessarily want you "harnessing" my anything, unless... unless it means I benefit from the result. And of course, that's the point-- that end-users can benefit from all that group wisdom, like Amazon reviews or delicious/popular tags, to help reduce the flood of data. So why not *say* it like that? Instead of calling it "harnessing collective intelligence", why not call it "helping users make smarter choices, more quickly, by accessing the knowledge, experience, and wisdom of a larger group?" If the focus is on the "harnessing" and not on the "so users can access..." the chances of building something that nobody actually wants goes up. (Of course, my variation of the buzzword is awkward, ungainly, and not likely to look good on a slide. But I'm sure other more creative people can make them snappier :)

We know that Ajax is good, but why? If it's about richer user experiences, then say so. And why stop there? What's the benefit to the user of that richer experience? And is that *always* a benefit?

And what does the *cloud* do for users? What does *attenuation* do for me? If it's about helping users reduce stress, or spend more time in flow, or have more fun, etc.... whatever it is, why not *say* that?

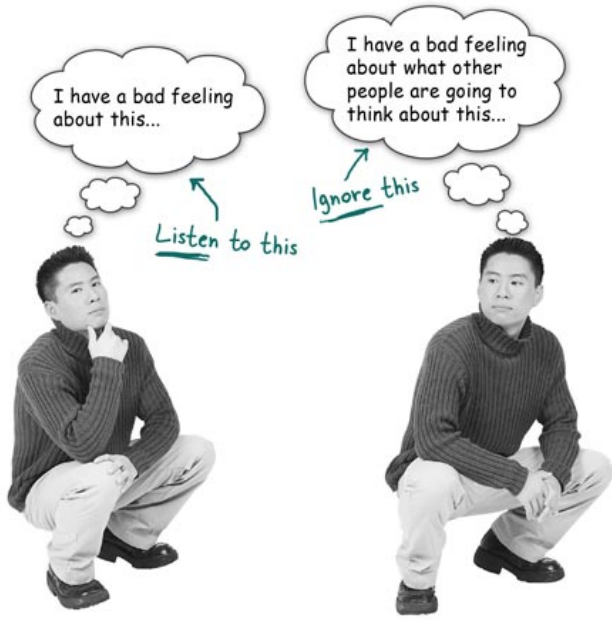
A buzz-phrase should explictly state how it directly benefits *the user*.

If I were a VC, the "elevator pitch" I'd ask for would be simply: "Tell me how this thing helps the user kick ass?" If you can't answer that, don't bother launching your power point.

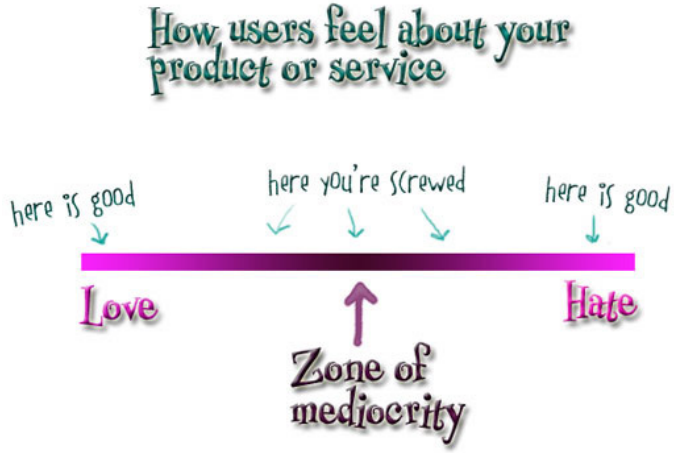
http://headrush.typepad.com/creating_passionate_users/2005/12/have_you_update.html

Being Brave is Tricky

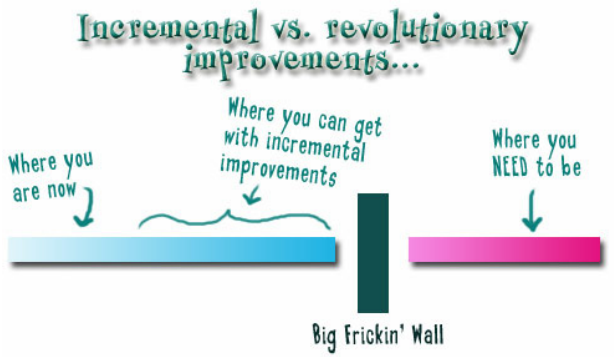
By Kathy Sierra on December 5, 2005



We've talked a lot about being brave, and the idea that if *nobody* hates your product, it's probably mediocre:



And we've talked about the difference between incremental improvements and revolutionary leaps:



But being brave is tricky.

If your new Big Idea doesn't scare the hell out of you, it's probably not a new Big Idea. If it doesn't scare *other* people, it might be because you allowed the consensus (or what you *imagined* as the consensus) to smooth the pointy bits, buffing and polishing the idea into a nice safe state that displeases nobody and *delights* nobody.

But what if your idea really does suck? We can never know for certain--some of the most important ideas have always defied logic, evidence, testing, focus groups, conventional wisdom, etc. But if you don't try, *then* where are we? Even if a Big Idea fails, that puts us one attempt closer to something that *will* work. The Big Ideas that *succeed* are often the ones that would never have happened if something *else* hadn't been tried and failed.

I don't have a good answer for this other than one tiny piece of advice--**try to determine the source of your fear.** If it's over what *other* people will say--or *are* saying--then you might want to acknowledge and ultimately ignore it. *Any* good new idea has critics. Many of those critics are smart, reasonable, sincere, not-afraid-of-change people who simply do not see and feel what you see and feel. Should you listen to them? Of course... with one ear, anyway. They might have truly useful info you didn't have--info that can help alter your course, change your decision, or at the least--prepare you for more criticism to come.

But--if we let the critics (or fear of criticism) talk us out of an idea we still believe in, the world will be more homogeneous. Smoother. *Less interesting*. Imagine where we'd be if people throughout history had *always* given in to the critics (or fear of critics). Imagine the ideas that would have been lost if others

hadn't been brave enough to stand up against smart people who disagreed. *Nature* needs change and diversity, but *humans* tend to favor the status quo.

Think about the times *you've* said, "What a terrible idea -- that will NEVER work!" about something that later succeeded and proved your perception wrong. If you can view your critics (I'm not talking about the "who moved my cheese" nay sayers who fear *anything* new) as people who are just as sincere as you are when *you* are wrong about someone's idea, it might help. They aren't trying to trash your idea -- they're trying to help. They might be saving your butt. But they might be dead wrong.

If your fears are coming from that nagging feeling within--your OWN little voice-in-the-head-or-gut that says, "something's not right here...", pay attention! But if your fear is over what *others* will say (or are saying), sometimes you just have to say, "screw 'em." ;)

Or as Apple says,

Here's to the crazy ones

"...you can praise them, disagree with them, quote them,

disbelieve them, glorify or vilify them.

About the only thing you can't do is ignore them.

Because they change things..."

Every time I make a post like this, I'm slammed somewhere for "glorifying the troublemakers and encouraging people to push their crazy, bad, stupid, dangerous ideas."

I don't *glorify* the "crazy ones". I thank God for them.

Yes, they might make mistakes. But is that better than the alternative?

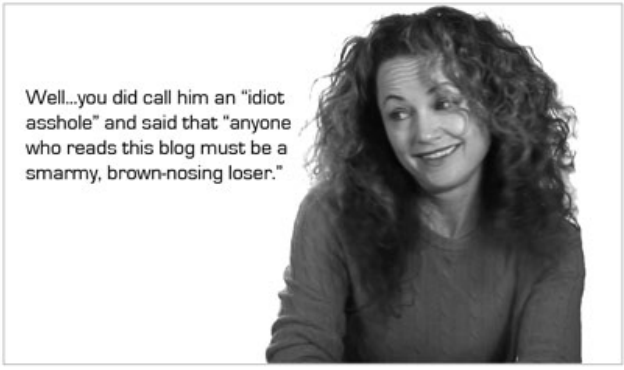
[Disclaimer: I'm obviously not talking about the kinds of ideas where lives are at stake. Different set of rules there...]

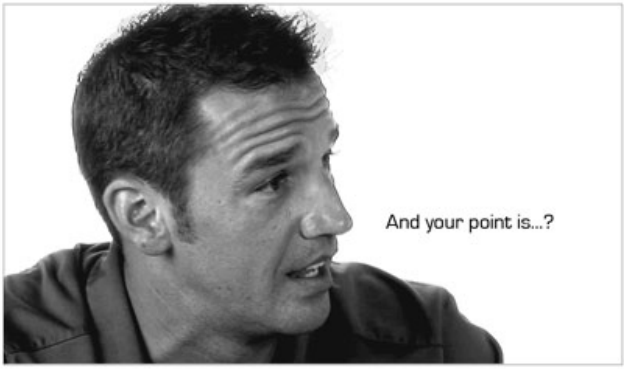
Think of new ideas and progress like photography -- you have to take a bunch of photos to get one good one. The more we have a culture that discourages and punishes all failures, the more we're just shooting our future in the foot.

http://headrush.typepad.com/creating_passionate_users/2005/12/being_brave_is_.html

Are "nice" and "honest" mutually exclusive?

By Kathy Sierra on December 8, 2005





No, "niceness" is NOT incompatible with "honesty." Although that seems to be quite a popular meme these days -- that you can be nice, or you can be truthful, but not both.

UPDATE: I think I better define what *I* mean by nice... I don't mean "sweet" or "complimentary" or anything more than the absence of abuse. And yes, I do consider the word "asshole" an abusive comment because it attacks the person, not the topic/idea/statement/whatever. This doesn't mean I wouldn't call someone that... but if I do, I have no business trying to claim that I'm actually interested in having a discussion. What's ironic about the Ben/Mean thing is that it was actually Mena who called Ben the A-word, after calling for more civility. I reckon I would have done the same.

I haven't been able to put it nearly as well as Just Kidding's TQ White did in a comment on Rogers Cadenhead's blog:

"...you can always be nice, even when you are being honest. This idea that somehow rudeness or unkindness is intrinsic to an honest discussion is completely wrong. It also, I believe, is an attitude that is destroying our ability to have public discourse.

Manners, politeness, respect, cutting a person some slack, even overlooking some of one's own more petty points are all things that are perfectly consistent with honesty. Honesty requires not contradicting things you know to be true. It requires advancing viewpoints that you believe. It says nothing about the linguistic tactics.

...represent the typical, juvenile attitude of people that simply don't care who they hurt. That use 'honesty' as a sleight of hand to deflect attention from willingness to brutalize people in pursuit of their own goals - often that cannot be advanced in a reasonable way."

Something to think about...

UPDATE: Mena responds to the controversy over her "let's be civil" thing, and her take makes a lot of sense in this:

"It's not about nice--it's about accountability. ...it's about taking as much responsibility for what we write online -- whether that's on a blog, in an email message, or on IRC -- as we would in a face-to-face, private conversation."

It seems as though one of her big concerns is that there are so many people in the world that aren't blogging, and what works on slashdot might scare a lot of others away. And I also don't believe that the price for blogging -- or speaking at a tech conference -- it that you must be wearing an asbestos suit 24-7. Requiring ultra-thick skin to participate more fully in the tech (or any) community is a barrier to entry that might just push away some of those who might not necessarily be *afraid* of the brutality, but simply don't *like* it. I don't agree with this "if you can't take the heat stay out of the kitchen" notion here. (Or is it, "if you can't run with the big dogs, stay on the porch..." I get those confused.)

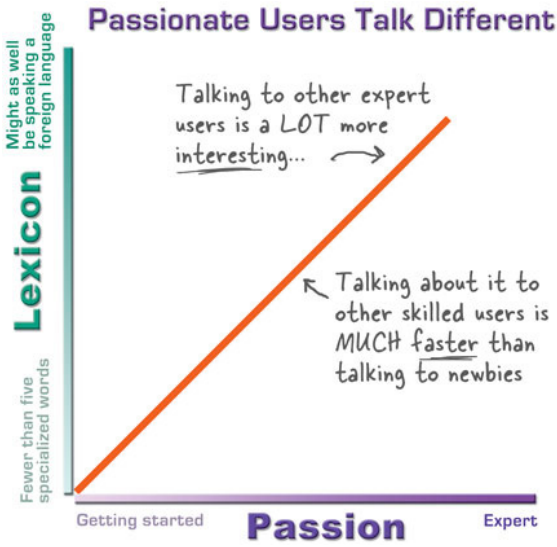
UPDATE: Rose made a good point in my comments -- my picture/post isn't actually describing the actual Ben/Mena thing--but the topic had been on my mind for some time, and this was simply the spark for a little post. The conversation continues all over the place, but mainly here, and includes this recent comment (made by an Englishman, no less:)

"The real problem with the abusive big-guns style of commenting is that it doesn't work (except as intimidation, but perhaps that's your real purpose?) Tell somebody that he is an asshole, that his work is bullshit, and you guarantee that he will (a) stop listening, and (b) reply in kind. Tell somebody that "this is wrong because X,Y,Z? and you might just change his mind."

http://headrush.typepad.com/creating_passionate_users/2005/12/are_nice_and_ho.html

Passionate Users Talk Different

By Kathy Sierra on December 11, 2005



Listen in on a conversation between three airplane pilots, and--assuming you aren't a pilot--you might understand 50% at best. Listen in on a conversation between three software architects, and even a new programmer might not have a clue. Snowboarders have their own terms. So do plumbers, photographers, librarians, ministers, dancers, realtors, musicians, graphic designers, and filmmakers (best boy? gaffer?).

But there's a world of difference between a specialized *lexicon* of domain-specific terms and *buzzwords*.

Domain-specific terms *compress* information, while buzzwords often *masquerade* as information.

Buzzwords are often (not always) semantically *empty* while specialized domain lexicons are semantically *dense*.

Domain-specific terms are usually associated with *passion*, or at least *expertise*, while buzzwords are often associated with those who might be *faking* expertise, or who are using them simply to impress others.

ZYZephyr wrote a great post taking me to task for my buzzwords post ranting (half tongue-in-cheek) about the 2.0 buzzwords. He

makes all the right points, and I agree with just about everything he said. Which tells me I didn't make my point, or that he didn't read my previous post about this topic ;)

My problem is not with the use of specialized language. On the contrary, in my earlier post on this I said (paraphrasing myself):

"When people are passionate (or even just "into") something, they have a shared lexicon that helps dinstinuish them from those who aren't.

Among other things, a shared vocabulary helps experts and professionals get a message across more quickly. But it also helps build their passion. Just figuring out the commonly-used phrases, words, names, stories, etc. are part of what gives people a sense of belonging. A sense of being a part of something special. A sense of having learned--and earned--their way in. So in this case, exclusionary isn't necessarily a bad thing.

Becoming a part of something new usually isn't that simple, especially if that new thing has real value. Pick an area where people are truly passionate, and there is virtually always a learning curve that includes new ideas, concepts, skills, knowledge and specialized terms. Most people have an "I Rule" experience in part because they've "crossed the chasm" and learned what the hell the experts are talking about."

Where this whole thing gets interesting is that many of the Web 2.0 buzzwords actually DO--for some people--compress and convey rich information. In other words, while I make a distinction between empty *buzzwords* and *domain-specific terms*, sometimes there's no clear line between the two. One guy's Web 2.0 *empty buzzword* is another one's *meaningful addition to the emerging technology lexicon*.

And that brings up the *other* thing I *like* about Web 2.0--that it has engaged so many people's minds in actively creating/defining/interpreting the meaning of the ideas, words, and concepts. Web 2.0 is both ambiguous *and* meaningful... but not for *everyone*. For many, the words are just useless marketing speak with no *there* there.

My problem with the Web 2.0 terms is not that they are meaningless. And my problem is not that they are too complex and should be dumbed down. *My problem is that they are focused on the technology and the business model, rather than*

focusing on what those things will mean to the end-user. And when I say "mean to the end-user", I don't mean that the end-user cares about the *words*. The end-user cares about what WE--the developers/implementors of Web 2.0-ish products or services--are creating for them.

When I say that the Web 2.0 words aren't user-driven, I don't mean that the users should be driving or even understanding the *words*. But if a deep concern for users isn't driving the *meaning* of these words, we're in for a flock of crap products and services that implement 2.0 goodness but do nothing to inspire or engage users. Again, my problem with 2.0 words is not about what they mean, or how consice or confusing they are, as much as about what they're *focused* on.

So, back to the "specialized words" thing... in helping support or build a community of passionate users, I would not discourage specialized lexicons--even (or *especially*) if that specialized lexicon means separating the newbies from the experts. That's as it *should* be and is part of what adds value to *becoming* an expert in the first place--you get to have this rich, complex, efficient communication with others and, yes, you might also consider that a way to *show off*. And I am not about to moralize on this one and suggest that wanting to "show off" is a bad thing. It's a part of human nature to take pride in how hard we've worked to learn this much and get this *good* at something. It's human nature to feel good about, well, *kicking ass*. Being recognized as an expert is certainly not the main benefit (or driving motivation) for becoming passionate about something, but for many people--it's a nice little side benefit.

Think about it... come on, really *think* about it. Somewhere in your past (maybe even within the last 48 hours), you've felt that little ever-so-slightly-I'm-better-at-this-than-you feeling that came from being able to keep up with a book, speech, or conversation that had words and phrases not known to "the rest of us." ;)

OK maybe you didn't feel all superior, but you at *at least* have felt the energy that comes from engaging and communicating at that higher level of complexity.

http://headrush.typepad.com/creating_passionate_users/2005/12/passionate_user.html

...but is it interesting?

By Kathy Sierra on December 13, 2005

Is your product *interesting*? Don't think--just answer. You probably said "yes." What about your *documentation*? Your *training* or *support*? What about your *blog*?

My friend Solveig Haugland (aka OpenOffice blog goddess) and I both did a stint in Sun's course development group, and were looking for ways to raise the quality without pissing off the entire department. So, with our manager's blessing, we created **The Checklist.** The Sun courseware already had elaborate "style guides" and strict technical requirements, but we didn't seem to be asking the simple questions that could make all the difference. If the course was technically correct, properly formatted, grammatically correct, satified the localization police, and all the deliverables were in the proper file formats and directory structure--then it was ready for beta.

The Checklist we made included all the *other*, less technical but equally (or more) important attributes like, "Does it manage

cognitive overload?" "Do the exercises reinforce the key points?" "Does each chapter include a 'if you remember only one thing from this module...' at the end?" "Does it include opportunities to learn from mistakes?" "Does it include redundancy to support memory?" "Does it include exercises to support the higher levels of Bloom's Taxonomy?" and on it went.

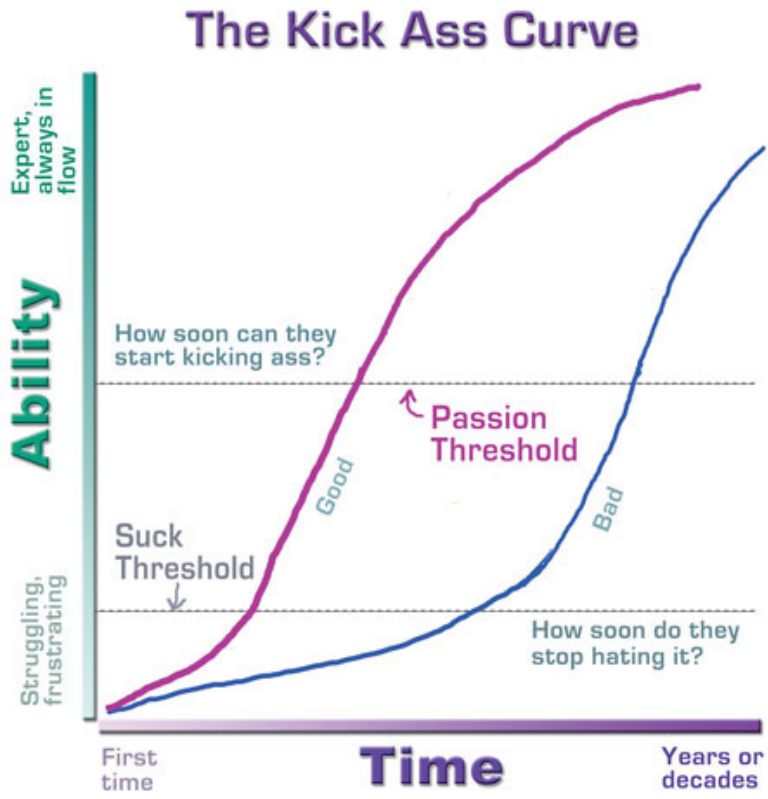
No big deal... just making sure the bases were covered.

But when we presented it to some key players in course development, one of the upper managers said, nice idea, but we don't need all of these. We nearly fell out of our chairs to see that he crossed out the simple question: ***Is it interesting?*** He said, "Whether the content is 'interesting' is completely irrelevant. If we get the topics right, and it's technically accurate, the content will be inherently interesting to programmers."

And the word "interesting" doesn't even set the bar very high--it's the word we use when we can't think of anything complimentary to say. "He is...well...*interesting*" or "Hmmm... *interesting* perspective." The words we *actually* wanted to use in the checklist were *compelling and engaging*, but we thought *interesting* would be an easier sell.

But even if he'd left "Is it interesting?" in, I now realize that many people would automatically check it off without really stopping to consider whether something really *is* interesting. Or that people would assume that given a certain context, "interesting" is irrelevant. Think about it. Even if your actual *product* is interesting (but still, stop and ask yourself if *that* is really true), do you have docs, FAQs, specs, articles, learning/support blogs, etc. that are NOT *interesting*? *Should* they be?

Obviously my opinion is YES YES YES. Because regardless of what the product is, whether it has passionate users may depend almost entirely on how quickly users can get past the Suck Threshold and the Passion Threshold. You may have a product that doesn't *require* a manual or support docs, but for most complex and sophisticated activities, docs or articles or books are needed as the user starts to explore more advanced uses. And it's those more advanced uses that lead to improving skills and knowledge and meeting challenges -- the whole "kick ass" thing that is a prereq for truly passionate users. [We believe that *nobody* is passionate about something they suck at.]



Why does "interesting" matter in getting past the suck/passion thresholds?

Isn't "technically accurate" or "high quality" enough? Well, how many technically accurate, high quality documents or training courses have *you* been exposed to that you dearly wished were a little better at holding your attention? The simple answer is:

The brain pays attention to--and remembers--that which it feels.

We've talked about this before...even if the reader/learner *wants* to pay attention and is interested in the topic, if the content itself is not offered in a reasonably interesting and engaging way, the brain keeps looking for something that *will* matter. Taking the time and care to make something *interesting* is simply being brain-friendly.

Your users won't learn and get better at whatever it is they're passionate about (or that you're hoping to *help* them become passionate about) unless their brains pay attention. And brains

pay attention to what *brains* care about, not necessarily what the conscious *mind* cares about. And to the brain, "interesting" is just the most basic prereq. The entry fee.

So, how *do* you make things *interesting?*

If you were a brain, and you'd been evolving for a very, very long time... what would *you* find interesting?

*** Surprise, novelty, the unexpected**

* Beauty

* Stories

* Conversation

* Emotionally touching (the whole kids and puppies thing)

* Counterintuitive failures or mistakes

* Fun, playfulness, humor

* Varying visuals

* Faces of people, especially with strong expressions

* Sounds, music

* Shock, creepy things

and of course...

* Sexiness

One fairly straightforward way to make documentation/training/articles interesting is to crank up four sliders **Conversation, Variety, Visuals, and Story**. I've talked before about conversational writing, and visuals, so in a post very soon I'll look at story and variety.

If you're *really* interested in story, you might want to look at Robert McKee's Story (if you saw the movie Adaptation, you'll recognize it). And if you haven't read Dan Pink's book, A Whole New Mind, there's some good story stuff in there (as well as a lot of other good things--I loved this book).

And for variety, well, *just do it* in whatever way makes sense. It's a lot more powerful than many people believe, because it's what your brain is *tuned* for. When the brain sees what it expects, it knows it can happily leave that thing behind and start hunting for something *else* to pay attention to.

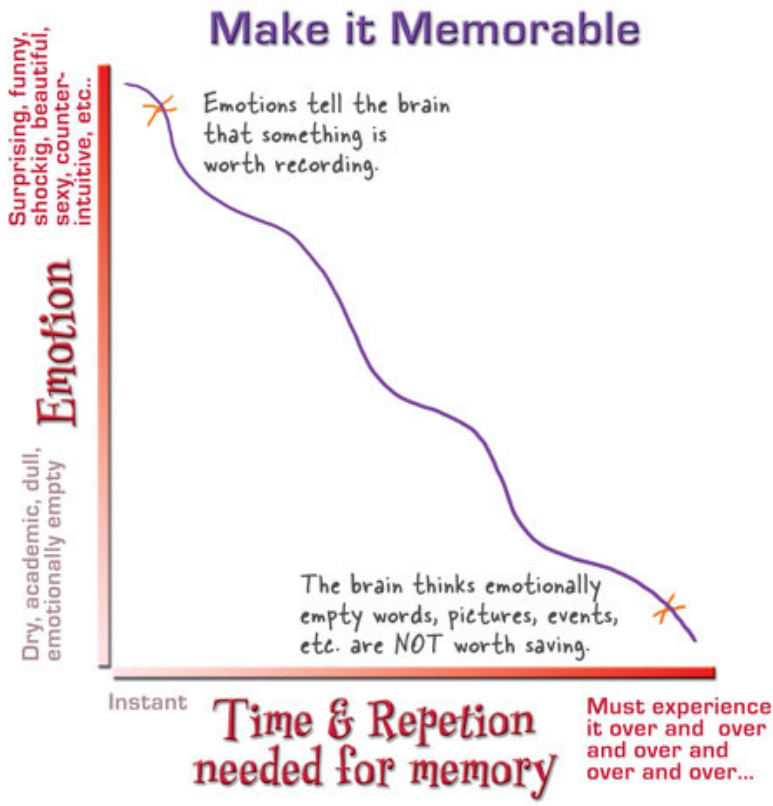
So, when you're making that checklist for your product, blog, article, book, documentation, training courseware, podcast--don't forget to include, "yes, it's all these wonderful things, but is it *interesting?"*

[Yes of course there's the big disclaimer that what is "interesting" to one is not interesting to another, but I assume we're all factoring that in :)]

http://headrush.typepad.com/creating_passionate_users/2005/12/but_is_it_interer.html

... but is it memorable?

By Kathy Sierra on December 19, 2005



So your product, training, documentation, presentation, blog, whatever is *interesting*, but is it *memorable*? Do you want it to be?

Where were you when you heard the news about 9/11? Chances are, you remember. What did you eat for dinner last Tuesday? Chances are, you do NOT remember (unless dinner involved a hot date, your birthday, a fist fight with the waiter, or some other emotionally-charged event). Just as emotions can tell the brain that something is worth *attention*, emotions also tell the brain that something is worth *recording*.

According to neurobiologist (and Nobel prize winner for his work on memory) Eric Kandel, a "switch must be thrown" to convert a memory from short-term to long-term storage. But a

neurochemical *smackdown* is happening inside your brain--two competing agents fighting for control of that switch. In one corner we have CREB-1, the essential component for throwing the switch that starts converting short-term memories to long-term storage. But in the other corner, we have CREB-2--CREB-1's arch rival. CREB-1's big goal in life is to throw the switch, but CREB-2 guards the switch saying, "Not so fast. If you want to throw that switch you'll have to get past me." CREB-2 is the gatekeeper!

If they gave you a drug that suppressed CREB-2, you'd remember *everything* the first time. While I would have killed for this the night before college exams, those for whom CREB-2 doesn't do its job are *not* having a good time. Think of all the things you're exposed to each moment, and imagine how awful it would if you remembered them all...

[Disclaimer: I'm playing fast and loose with the metaphors and science here]

If CREB-2 *inhibits* memory, then how do you inhibit CREB-2? How do you stop it from protecting the switch? There's the slow, painful (or at least *boring*) way we all used in college to get through some of our exams. We just kept rereading the same damn chapter over and over. With enough time and repetition, just about anything can be saved to long-term memory.

But there's a more efficient way--EMOTIONS. Scientists have confirmed (and you know it from experience) that emotions play a major role in memory. And it's thought that the chemicals of emotion must be telling CREB-2 to back off and let CREB-1 do it's work.

Just as the brain pays *attention* to that which it feels, the brain *remembers* that which it feels. If you can help your users trick their brains into thinking that something is important enough to store, you can help your users learn more quickly. Learning = getting past the suck threshold faster. And learning also means gaining the kind of skill and expertise that can meet the challenges needed to reach the flow state. And *that's* where you hit the passion threshold.

Remember--your users don't have to be passionate about your *product* in order to be *passionate users*. Sometimes--*often*--users are passionate about what they *do* with your product. And it's that thing they *do* where you can help them kick ass. Users

who "kick ass" are those who get good enough to reach a state of "optimal experience" doing whatever it is you're helping them do (through your product, service, support, learning, whatever). And that can happen with almost *anything*. It's the reason that the GTD system has become so popular--*it helps us spend more time in flow!*

If you want them to remember, make it memorable.

The number eight is arbitrary, but the numeral "8" overlaid on a picture of the spider (which brains are preprogrammed to react to), helps "burn in" the link between spiders and the number 8.

Emotions aren't the only things that improve the memorability of something--pictures, patterns, chunking, and all sorts of "memory tricks" can make a huge difference in whether something is recorded or--sometimes more importantly--whether it can be easily *recalled*. But I'll save those tricks for another post.

For now, think about how you can use the brain's built-in memory "tagging" system to help users learn/remember more quickly. Link the thing you want remembered with something likely to evoke at least the tiniest chemical reaction. And what are those things? **The same things that the brain finds *interesting*:**

*** Surprise, novelty, the unexpected**

* Beauty

* Stories

* Conversation (including conversational writing)

* Emotionally touching (the whole kids and puppies thing)

* Counterintuitive failures or mistakes

* Fun, playfulness, humor

* Varying visuals

* Faces of people, especially with strong expressions

* Sounds, music

* Shock, creepy things

and of course...

* Sexiness

The difference between whether you use these things to help focus attention or to support long-term memory is in how (and for how *long*) you use them. A picture of a spider will get your brain's attention, but by *linking* that spider to something (like the number "8"), you greatly increase the chance that the link between spiders and 8 legs is remembered. A fact is more likely to be remembered if that fact is being "stated" by the face of a person with a strong facial expression. Getting what you expect is not nearly as memorable as when something you thought would work *fails*. On it goes...

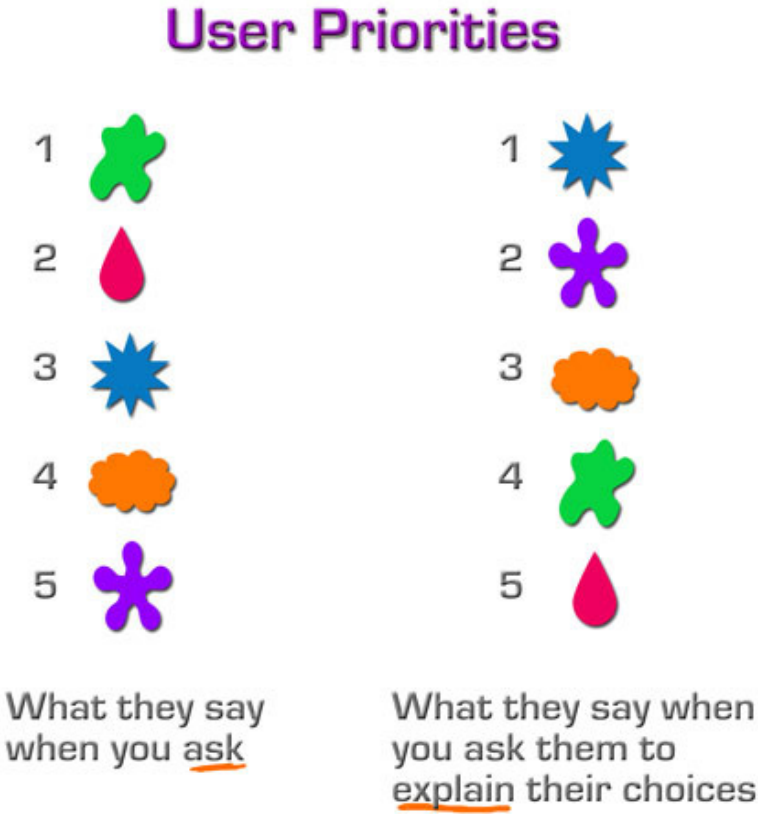
Oh yes, there *is* one "emotion" that has the opposite effect on memory. The chemistry of anxiety (the stress of worry) is the one feeling that works *against* memory. So whatever you can do to make users/learners feel *comfortable* about the learning experience goes a long way toward supporting memory. If people are made to feel *stupid* for "not getting it", the chances they'll learn it (let alone *remember* it) drop. And unfortunately, way too many technical manuals, tech support FAQs, books, and poorly designed product interfaces DO make us feel stupid.

So, "interesting" gets your foot in the door, but "memorable" is what helps build and support passionate users.

http://headrush.typepad.com/creating_passionate_users/2005/12/_but_is_it_memo.html

The Quantum Mechanics of Users

By Kathy Sierra on December 21, 2005



People have commented that "creating passionate users" means nothing more than "listening to users like we always have--DUH!" But if it *were* that simple, we'd all be producing--and using--products and services that people love. That meet real needs. That fulfill real desires. *That help people kick-ass.*

How, then, to explain the Grand Canyon-sized gap between what users *really* want and what we so often produce as a direct result of our sincere listening? Maybe the physics is wrong...

Light can behave as a wave, until you ask it to explain how it got from point A to point B, in which case it can behave as a particle. In other words, asking light to *explain* itself can change the very nature of how we perceive it. And this notion that sometimes

"*observing* an event *changes* the event" comes up in many areas of quantum physics.

But it's not just the subatomic world that gets weird when you look too closely--in some cases, asking a *user* to explain his choices changes his choices! In his book Blink, Malcolm Gladwell (author of The Tipping Point) gives an example where students were asked to rank order 44 different kinds of strawberry jams. When compared with the rankings of experts, the students did fairly well -- "even those of us who aren't jam experts known good jam when we taste it." But--and here's where it gets weird--when the students were asked in advance to provide not just the rankings but a written *explanation* of their choices, the student rankings lost virtually all correlation with that of the experts. As Gladwell puts it, "By making people think about jam, [the researchers] turned them into jam idiots."

Think about that when you're asking for user feedback whether as focus groups, user questionnaires, or even usability testing (although the implications are different for each of these things).

So how can we hope to learn anything about what our users want and need if the very act of answering a question could change their answer? We have to get better at making inferences from what we observe *without intervention*. We have to get to the spirit of what we observe, rather than focusing on the specific details. We have to reconize that what they *do* says much more than what they *say*, especially when they're not saying anything at all.

Readers here left some great comments about this on my earlier Listening to users post:

Tim said:

The comments about listening to what the users are saying, what they're not saying, and how it's being said reminds me of the quote by Claude Debussy, "Music is the silence between the notes."

And Matthew Moran said:

It is not that we should not listen to clients/users but we should not let their limited understanding of what is possible, limit where the solution/software/project can go. It is important to listen and draw additional information into the open. In this

way, we can discover what is truly desired but never contemplated from the client's perspective.

Paulo Eduardo Neves said:

Gilberto Gil, a great brazilian musician and the country current minister of culture, has a verse that says: "The people knows what they want, but the people also wants what they don't know".

Eric Stephens offered this link great post from Mark Hurst on Customer Research that includes:

"In our non-directed listening labs, we ask customers to use the Internet in the way they normally use it at home or work. While we do have a goal for the research, we try to let the customers lead us to the answer, rather than the other way around."

And Stu Max made it simple:

I guess that's how I'd wish you would reframe your point: You've always got to listen to your users, but sometimes you've got to listen beyond the words.

In addition to *listening* to users, we should *observe* them as a wildlife photographer or naturalist would--in the users native habitat, from a distance, with as little intervention as possible. We have to look for the whys based on the whats of their behavior. And when we do ask questions, the questions should be not just on specific behaviors ("why did you do it that way?") but also (perhaps more importantly) about what they value at an abstract level ("what does it mean to you to be using [whatever] in your [work/life]? How does it help you in (or prevent you from) kicking ass?")

This doesn't mean we shouldn't sweat the details--down to the last interface pixel, book font, metal finish, or drum beat. It all matters. And much of it *can* come from questioning users directly. The trouble is that this is where we tend to spend nearly *all* of our "listen to users" effort. We field complaints, solicit feedback, and accept customer requests. In other words, we focus on the trees and miss the forest.

Why not become "user naturalists" and find out what *really* makes our users inspired/frustrated/motivated/hopeless/passionate? Maybe the best way to find out what they need and want from our product

(or from a future product we hope to develop) is by asking about something *other* than the product. Maybe they say they're satisfied with our product (or the category of products in which ours belongs), but we need to ask if they're satisfied with the very nature of what they're using our product *for*. Maybe asking about their favorite hobbies--the things they *are* passionate about--can help shine a light toward a new feature or capability (a new slider) we hadn't previously imagined. One that nobody ever associated with this type of product or service.

While we can still ask why they chose the blue button, we must understand that if we tell them in advance that they'll need to explain their choice, that knowledge could change the outcome. It might cause them to click the blue button when they *would* have clicked the green one! When you collapse the wave function, make sure that what you get is not simply what you *caused* by looking. :)

http://headrush.typepad.com/creating_passionate_users/2005/12/the_quantum_mec.html

BrainDeath by Micromanagement: The Zombie Function

By Kathy Sierra on December 26, 2005

The most important function for a manager is X = -Y, where X is employee brain use and Y is degree of management. To use the horse whisperer's advice, The more you use your *reins*, the less they'll use their *brains*."

If you asked 100 managers which they'd prefer--employees who ***think***, or mindless ***zombies*** who respond only (and exactly) as ordered, you'd get 100 responses of, "What a ridiculous question. We hire smart people and stay out of their way so they can do their jobs." And if you asked 100 managers to define their management style, *none* would claim to be micromanagers. Probe deeper, though, and the truth begins to emerge.

Ask managers if their direct reports can make decisions as well as the manager can, and they hesitate. Ask if the manager could step in at a moment's notice and perform the employee's job, and too many managers would say--with *pride*--"yes."

Do you have a micromanager? *Are* you a micromanager? Are all micromanagers clueless or and/or evil? Of course not. Most micromanagers I've known (or had) were driven by one or both of the following:

1) Not enough time

Taking the time to give employees the same data, knowledge, and skills needed to *do things right* can be a luxury many managers just can't afford. Or so they think. While it's oh so tempting to just step in and DO IT, micromanagement doesn't scale. Better to:

Take the time it takes [now] so it takes less time [later]."

2) Concern for quality

Micromanagers often believe that they *know* more, and more importantly -- ***care*** more. Often they're right. But it's a downward spiral--

Micromanagement creates zombies.

Of course micromanagers don't actually *create* zombies--they simply inspire (or force) *zombieism* on the job. Follow those work zombies home, and their zombiness vanishes. Thier eyes light up, their brain kicks in, and their passion for playing with their kids, championing a cause, or just playing their favorite after-work hobby emerges. You see the side of them that micromanagement crushes.

Do you have a micromanager?

Or are *you* a micromanager? If you demonstrate *any* of these seemingly admirable qualities, there's a big clue that you might be making zombies.

1) Do you pride yourself on being "on top of" the projects or your direct reports? Do you have a solid grasp of the details of every project?

2) Do you believe that you could perform most of the tasks of your direct reports, and potentially do a better job?

3) Do you pride yourself on frequent communication with your employees? Does that communication include asking them for detailed status reports and updates?

3) Do you believe that being a manager means that you have more knowledge and skills than your employees, and thus are better equipped to make decisions?

4) Do you believe that you *care* about things (quality, deadlines, etc.) more than your employees?

Answering even a *weak* "yes" to any *one* of these might mean you either are--or are in danger of becoming--a micromanager. And once you go down that road, it's tough to return. A quote from Dune (can't remember exactly) applies here, and goes something like:

"Be careful of every order you give. Once you give an order on a particular topic, you are responsible for *always* giving orders on that topic."

What can you do if you have--or are--a micromanager?

Admit it, and deal with the two driving forces: concern for quality, and need for speed. Take the time it takes *today*. Invest in the time and training to give your employees whatever they need to make the decisions or complete the tasks you find yourself needing (or wanting) to do. And if *caring* is the big concern, well, you get what you create. If you treat employees like zombies, then zombies is exactly what you'll get. Sometimes all it takes is giving people a chance to develop more skill and knowledge, the space to use their brains, and a worthwhile challenge.

"But, but, but--they *don't* care as much as I do -- that's why *I'm* the manager and they're not." Bulls***. You might be the manager simply because you wanted to be a manager. Nothing wrong with that, but it doesn't necessarily mean you're better at the job than those you manage. It might even mean you're simply better at the details and support work than the *actual* work.

The companies I love to hate are those that allow only a single career path--the "management track". One of the things I liked about Sun was that Scott McNealy made a clear distinction between "Individual Contributors" and "Managers", and didn't penalize those who wanted to be--and stay--kick ass individual

contributors. Sun knew the value of not taking their brightest engineers and forcing them to choose between doing what they love vs. moving up the pay scale. Both tracks were recognized and rewarded. (Of course, when the bubble burst, all bets were off...)

Doing everything right doesn't guarantee passionate users, but if we--or those we manage--don't have passion, how can we expect to inspire our users?

And here's a parting thought... this obviously doesn't apply only to employees. What about parents who micromanage their kids? Teachers who micromanage their students? Ministers who micromanage their memebers? Political leaders who micromanage their, well, *us*? Or what about developers who micromanage their users? Hmmm....

http://headrush.typepad.com/creating_passionate_users/2005/12/braindeath_by_m.html

The hi-res user experience

By Kathy Sierra on December 30, 2005

Learning music *changes* music. Learning about wine changes wine. Learning about Buddhism changes Buddhism. And learning Excel changes Excel. If we want passionate users, we might not have to change our products--we have to change how our users *experience* them. And that change does not necessarily come from product design, development, and especially *marketing*. **It comes from helping users learn.**

Learning adds resolution to what you offer. And the change happens not within the product, but between the user's ears. The more you help your users learn and improve, the greater the chance that they'll become passionate.

What does it *mean* to say that someone is passionate about something? It's a lot like discussing porn--there's no clear

definition, but you know it when you see it. Nobody refers to the guy who knows just two types of wine--red or white--as "passionate about wine." But the movie Sideways was about people who were passionate about wine. The point was not that they *drank* a lot of wine (although in the movie, they definitely *did*), but that they *knew* so much about it. ***They knew enough to appreciate and enjoy subtleties that are virtually inaccessible to everyone else.***

It's the same way with classical or jazz music--learning about the music *changes* the music. What the music expert hears has more notes, more instruments, more syncopation... than what I hear when I listen to the same piece. Of course I don't mean the music technically changes, but if the way we *experience* it shifts, it is AS IF the music itself shifts.

And it's not just for hobbies. Think about a spreadsheet, for example. Joe Excel User can do the basics--calculations, pie charts, bar graphs, some reports. To Joe Excel User, the software is a tool for *doing spreadsheets*. But imagine Joe were to learn the deeper power and subtleties of not just the app itself, but the way in which the app could be used as, say, a modeling and simulation tool. For Joe, now, the software itself has transformed from a spreadsheet tool to a modeling and simulation tool. More importantly, **the way Joe *thinks* as he uses the software also changes.** Rather than approaching a session with Excel as a way to crunch some numbers, he sees it as a way to do predictions, forecasts, and systems thinking.

People are not passionate about things they know nothing about. They may be *interested*. They may *spend money*. But without the enhanced skill and knowledge that adds resolution, there is no real passion. At least not the kind we talk about (and aim for) here--the kind of passion we talk about when we say, "He is passionate about photography" or "She is passionate about animal rights" or even, "He is passionate about his Mac."

And a passion for *one* thing can spill over into a passion for life itself. And for many people, the loss of passion/desire for once-loved things is a clear symptom of clinical depression. For writer Larry McMurtry, the loss of passion for books (he's an antiquarian book collector when he isn't writing novels and screenplays) was one of the worst parts of the post-heart-surgery depression he experienced a decade ago. He simply stopped feeling that *feeling*. Books changed *back*--back to that

state the non-book-passionate experience--and were simply *old books*. Fortunately, McMurtry recovered and regained his passion for books.

So, what can you *change* for people? Or rather, what can you help others change for themselves? How can you increase the resolution of the products and services you offer--*without touching the products?* That doesn't mean you can take any old piece of crap and by teaching people to become expert, magically transform it into a work of art. But if there's potential for a richer experience--an experience the non-passionate don't see, taste, hear, feel, smell, touch, or ever recognize...why not see if there's a way to help more people experience that?

And since I believe that passion requires learning, and that means we all have to become better "learning experience designers", I'm working on a big "crash course in the latest learning theory" post that summarizes most of the key principles, in one place (with pictures :)

2005 may be the year HD finally arrived for TV and video, I hope 2006 is the year of HD User Experiences. And it's up to us to make that happen.

http://headrush.typepad.com/creating_passionate_users/2005/12/the_hires_user_.html

If pets could design user experiences...

By Kathy Sierra on January 2, 2006

This is a photo I took at Foo Camp of Caterina Fake's dog, Dos Pesos, while I watched him playing with blades of grass, imaginary objects, and Caterina. A few days ago, Caterina (whom many of you know as a co-founder of FlickR) posted a quote from Johann Huizinga, from *Homo Ludens* that included:

> *"We have only to watch young dogs to see that all the essentials of human play are present in their merry gambols. They invite one another to play by a certain ceremoniousness of attitude and gesture. They keep to the rule that you shall not bite, or not bite hard, your brother's ear...*
>
> *Here we have at once a very important point: even in its simplest forms on the animal level, play is more than a mere pysiological phenomenon or a psychological reflex. It goes beyond the confines of purely physical or purely biological activity. It is a significant function--that is to say, there is some sense to it. In play there is something "at play" which transcends the immediate needs of life and imparts meaning to the action. All play means something."*

When I saw this, I realized I left out perhaps the most important goal for 2006--*upping the FUN slider!*

We talked about this recently in never underestimate the power of fun, but I think it's a good thought for starting the new year on a more playful tone--and I mean playful *for our users*. And maybe the best teachers are our pets.

For most animals, it's almost impossible to separate play from learning. *They're virtually the same thing*. It's not just about having fun. Animals use play to develop physical and social skills, but they *continue* to play throughout their lives. Remember, the experience of "fun" floods the brain with *good drugs*. The neurochemistry of fun (and "fun" doesn't have to mean "funny", in the way that chess is fun but not funny) tells the brain to pay attention, engage, and remember. We're all hard-wired for this.

What can we do to bring more joy, fun, and flow into the lives of our users? Obviously the answer depends heavily on the kind of product or service or cause you support, but there will always be *something* we can add, subtract, or change to make our user's experience feel a little less like *work* and a little more like *play*. (For some of us, it could be as simple as a few fine-grained user treats).

http://headrush.typepad.com/creating_passionate_users/2006/01/if_pets_could_d.html

Crash course in learning theory

By Kathy Sierra on January 3, 2006

One formula (of many) for a successful blog is to create a "learning blog". A blog that shares what you know, to help others. Even--or *especially*--if that means giving away your "secrets". Teaching people to do what *you* do is one of the best ways we know to grow an audience--an audience of users you want to help.

It's what I try to do here because--let's face it--*you're just not that into me* ;) But I assume (since you're reading this blog) that you ARE into helping your users kick ass. So to make content that's worth your time and attention, I try to make this a learning blog. I reckon y'all could not care less what I had for dinner, who I ate with, or what I think about the latest headlines.

So, as promised in an earlier post, here's a crash course on some of our favorite learning techniques gleaned from cognitive science, learning theory, neuroscience, psychology, and entertainment (including game design). Much of it is based around courses I designed and taught at UCLA Extension's New Media/Entertainment Studies department. This is the long version, and my next post will be just the bullet points with the pictures--as a kind of quick visual summary.

This is *not* a comprehensive look at the state of learning theory today, but it does include almost everything we think about in creating our books. And although it's geared toward blogs/writing virtually everything in here applies regardless of how you deliver the learning--you can easily adapt it to prentations, user documentation, or classroom learning. And remember, this is a BLOG, so don't expect academic rigor ;) but I do have references, so leave a comment if there's something in particular you want.

Crash Course in Learning Theory

The long version...

Talk to the brain first, mind second.

Even if a learner is personally motivated to learn a topic, if the learning content itself isn't motivating, the learner's brain will do everything possible to look for something more interesting.

This applies to both *getting* and *keeping* attention, as well as memory. Remember, you can't do anything until you get past the brain's crap filter! And to the brain, a dry, dull, academic explanation is definitely CRAP (regardless of how much your *mind* cares about the topic).

Learning is not a one-way "push" model.

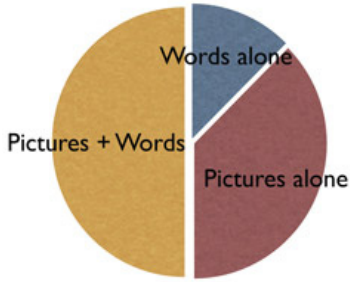
Learners are not "empty vessels" waiting to be filled with content pushed into it by an expert, blogger, author, etc. Learning is something that happens between the learner's ears--it's a form of co-creation between the learner and the learning experience. You can't *create* new pathways in someone's head... your job is to create an environment where the chances of the learner "getting it" in the way that you intend are as high as possible.

Provide a meaningful benefit for each topic, in the form of "why you should care about this" scenario.

Learning is much more effective if the learner's brain knows why what you're about to talk about matters. The benefit and/or reason why you should learn something needs to come *before* the actual content. Otherwise, the learner's brain gets to the end of what you're telling them and says, "Oh, NOW you tell me. If you'd said that earlier, I would have paid more attention..." This process of not-paying-attention is not completely within the learner's conscious control so, like I said, even if the person is motivated to learn this thing, their brain can still tune out during specific parts that don't start with a compelling benefit.

To find a "meaningful benefit", play the "Why? Who Cares? So What?" game with someone else. Describe the thing you're trying to explain, to which the other person asks, "Why?" Provide an answer, to which the person then asks, "Who cares?". Provide an answer, to which the person asks, "So?" At this point, when you're nearly ready to kill them for *not getting it*, you probably have the thing you should have said *instead* of whatever you said first (and second). The most compelling and motivating reason/benefit is almost always the thing you say only after you've answered at least three "Yeah, but WHY do I care?" questions.

Use visuals!



We are all visual creatures, and the brian can process visual information far more efficiently than words. These pictures can come in many forms:

* Info graphic or diagram

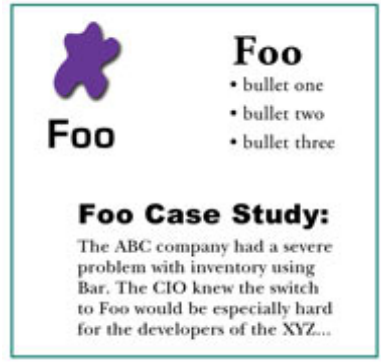
* Visual metaphor

* Picture of the thing being described, with annotations

* Picture of the end state

* Picture designed to create attention and recall

Use redundancy to increase understanding and retention.



Redundancy doesn't mean *repetition*--it means "say the same thing again, but differently." And "differently" can mean:

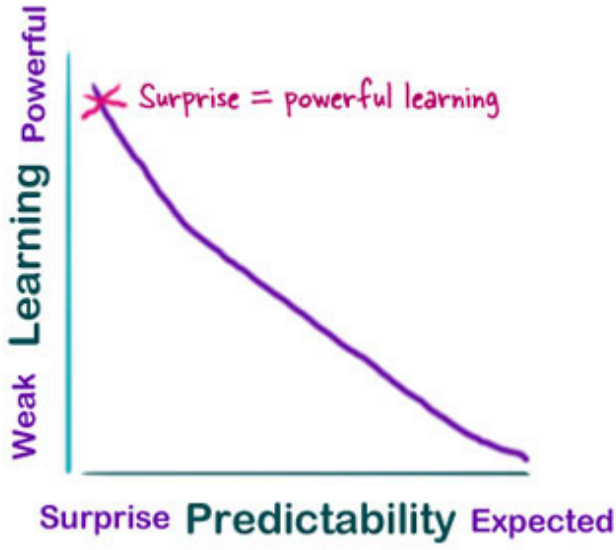
* From a different *perspective*.

* Using a different information channel (channels include things like Graphics, Examples, Prose explanations, step-by-step instruction/tutorial, case studies, exercises, summaries, bullet points, commentary, devil's advocate, Q & A, personal POV, etc.)

Also, **the more senses you engage, the greater the potential for retention and recall.** Even having a bowl of just-popped popcorn or the smell of freshly-baked cookies while learning, can make a difference. Bummer about web-delivered content, though...

Being terse is good for a reference document, but deadly in learning content. The best learning experience considers the way you'd learn that particular thing in real life -- but offers it in a safe, simulated, compressed form. Real-life learning is never terse; it's choas and confusion punctuated with moments of insight ("Ah-ha!") and clarity. It's a wave, not a straight line. A learning blog, book, or classroom shouldn't try to straighten it out!

Maintain interest with variety and surprise.



Use conversational language.

The brain pays more attention when it thinks it's in a conversation and must "hold up its end." And there's evidence that suggests your brain behaves this way even if the "conversation" is between a human (you) and a book or computer screen (or lecture).

Use mistakes, failures, and counter-intuitive WTF?

People usually learn much more from failures than from being shown everything working correctly or as expected.

The most memorable learning experiences are usually those where things are going along fine, making sense, etc. when you suddenly slam into something that goes terribly wrong. Describing the things that do NOT work is often more effective than showing how things DO work. (We call this the "WTF learning principle").

But *showing* is even better than *describing*. And even better than *showing* is *letting the learner experience*. Take the learner down a garden path where everything makes perfect sense until it explodes. They are far more likely to remember than if you simply say, "Oh, and be sure you do it such and such a way."

It's tempting to want to *protect* the learners from the bumps and scrapes experienced in the real world, but in many cases (with many topics) you aren't doing the learner any favors.

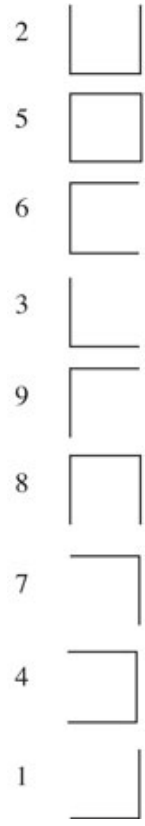
Use the filmaker (and novelist) principle of SHOW-don't-TELL.

Rather than lecture about the details of how something works, let them experience how it works by walking them through a story or scenario, where they can feel the bumps along the way.

Use "chunking" to reduce cognitive overhead.

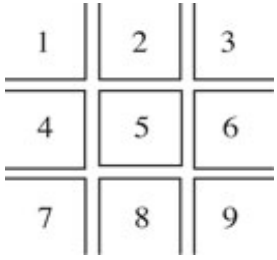
Remember, we have very little short-term memory (RAM) in our heads. The standard rule is that we can hold roughly 7 things before we must either commit some of it to long-term storage or toss it out to take in something new. And the things you hold in short-term memory vanish as soon as there is an interruption. You look up a phone number, and as long as you repeat it to yourself and *nobody asks you a question*, you can remember it--usually just long enough to dial the number. By the time you finish talking to the person on the other end of the line, the number is long gone.

Chunking takes fine-grained data/facts/knowledge and puts them into meaningful or at least *memorable* chunks to help reduce the number of things you have to hold in short-term memory, and increase the chance of retention and recall. For example, imagine you were asked to take 30 seconds to memorize the following "code symbols" for the numbers 1-10:



you'd be lucky to get 60% correct in a follow-up quiz given immediately after those 30 seconds. There are simply too many symbols to memorize in such a short time, and there's no instantly obvious way to relate them to one another.

But... with one simple change to the way in which the symbols are presented--and without changing the symbols:



30 seconds gets most people to 100% accuracy in the follow-up quiz. In other words, by grouping the symbols into a meaningful, memorable pattern, we reduce the number of

individual (and potentially arbitrary) things you have to memorize, and increase the chances.

Since stress/anxiety can *reduce* focus and memory, do everything possible to make the learner feel relaxed and confident.

That does not mean dumbing-down the material, but rather letting the learner know that -- "This IS confusing -- so don't worry if it's still a little fuzzy at this point. It will start to come together once you've worked through the rest of the examples." In other words, let them know that they aren't stupid for not getting it at this point. For especially difficult and complex topics, let the learner know where they should be at each stage, and help them decide whether they need to go back and repeat something. Make sure they know that this repetition is part of the normal learning process, not something they must do because they failed.

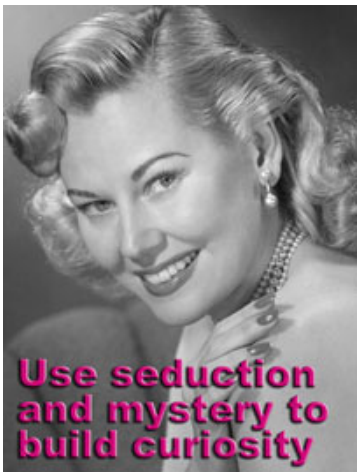
If you're worried about being patronizing, then *don't patronize*. Just be honest about what it takes for people to learn that content. But you can't do that unless you *know* how hard it is for a beginner to learn it. As experts, we have a tough time remembering what it was like NOT TO KNOW, so if you're not sure, do the research. One of the best ways to find out what newcomers struggle with is to visit online discussion forums for beginners in your topic. This is also a great way to come up with a table-of-contents or topic list, because what you THINK should be a no-brainer might be the one thing everyone gets stuck on, and what you think would be confusing could turn out to be easy for most people.

The point is, YOU are not necessarily the best judge of how your audience will learn the topic. And empathy rarely helps -- you cannot truly put yourself in someone else's shoes unless their brain and background are a very close match for yours. You have

to find out what your learners are struggling with, and suspend any judgement about "This *should* be a no-brainer."

Those who have *taught* a topic have a big advantage writing about it--they've fielded the questions and watched people struggle. **They know how things should be "weighted" according to how difficult they are.** But you can learn almost as much simply by lurking on beginner discussion forums (or attending user group sessions for newbies).

Use seduction, charm, mystery to build *curiosity*.



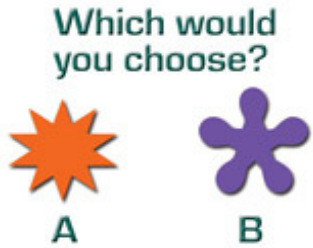
We're hard-wired to pay attention and pursue things we're attracted to. This isn't about selling them on an idea--it's about helping them stay engaged and *learning*. Knowing what--and when--to *withold* is one of the most powerful tools you have. If you're writing *reference* material (like this post), witholding will just piss people off. But in a learning experience, you want a page-turner. And don't even *think* about suggesting that "page-turner" doesn't apply to, say, technical material. If the purpose is learning, the learner has to stay engaged. It's up to you to craft an experience that keeps them hooked. This engagement might be within a *single* post, or you might offer little cliffhangers or teasers to keep them engaged across multiple posts, if that's what it takes to cover a topic.

Use a spiral model to keep users engaged.

Game developers know the importance of "The Next Level", and learning experiences must do the same. Each iteration through the spiral should start with a meaningful, motivating goal, followed by the interaction/activity/reading that moves you toward that goal, followed by a meaningful payoff. Ideally, the "meaningful payoff" leads right into the next motivating goal.

For example, in a game the payoff for completing a level might be "You Get A New Weapon". But now that you *have* that new weapon, here's the cool new thing you can do that you couldn't do before. Learning doesn't need to be any different. "Imagine you want to do X on your website..." is the goal that starts the topic, but when the topic is complete, the learning content can say, "Now that you have THAT new [superpower capability], wouldn't it be cool if you could do Y?" And off they go into the next round of learning.

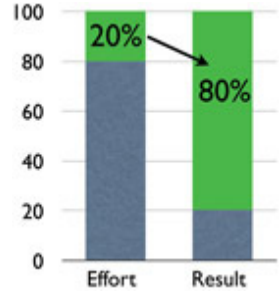
Don't rob the learner of the opportunity to think!



Rather than simply spelling everything out step by step, ask questions, pose multiple and potentially conflicting viewpoints, show the topic from different perspectives, and set up scenarios (and possibly exercises) that allow the learner to use deeper brain processing. Things that encourage deeper thinking are those that cause the learner to categorize, organize, apply, infer, evaluate, etc. Don't be afraid to pose questions that you don't answer right away.

Think back to those teachers you had who would ask a question then immediately answer it, as opposed to those who would answer a question then just sit there... waiting...

Use the 80/20 principle to reduce cognitive overload.



It's far more important that they *nail* the *key* things than be *exposed* to *everything*. Be brutal, be brave, be relentless in what you leave out. Knowing what NOT to include is more important in learning design than knowing what TO include.

Context matters.

Try to place facts, concepts, procedures, examples in a bigger context. Even if you've already discussed the context, don't be afraid to repeat that context again. For example, instead of always showing code snippets, show the code within the larger context of where it usually appears. Highlight the code you're focused on by bolding it, putting it in a box, etc., so that the learner is not overwhelmed by the amount of code, and can focus on just the part you're talking about, but still be able to see how that new code relates to the rest of the code. Our rule of thumb in our books is to show the same code context two or three times before switching to just the snippets (although this rule varies greatly with the type of code).

Emotion matters!

People learn and remember that which they FEEL. Look back at what you've written and if it's dry and lifeless, try to inject some energy. Dry, academic, formal, lecture-style writing is usually the WORST form of learning content.

One of the many ways to help tap into emotions (and increase attention and memory) is to use the brain's reaction to *faces*. Almost any kind of face with a strong expression evokes a part of the brain reserved *just* for processing faces. The ability to

accurately recognize faces and read facial expressions is a key element of survival for the brain...

Never underestimate the power of FUN to keep people engaged.

The act of having fun is also an emotion, so anything associated with fun has a greater chance of being remembered.

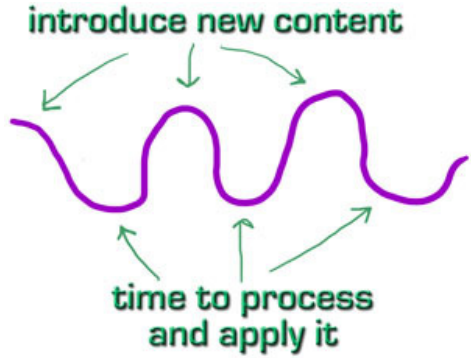
Use stories.

Humans have been learning from stories for, well, *a really really really long time*. Millenia longer than we've been learning from lectures on just the data and information. When we say "stories", we don't necessarily mean actual fictional "John's network went down just as he was plugging in the...", although those do work. But a "story" can simply mean that you're asking the learner to imagine herself wanting to do a particular thing, and then offering an experience of what that would be like if she were actually trying to accomplish it, with all the ups, downs, false leads, etc. (but again, with less of the actual pain she might experience in real life). A flight simulator, for example, is a kind

of story.You aren't just up there learning the controls; you're actually *flying* in a particular storyline.

If you're a software developer, another way to think about story-driven learning is to map use-cases to learning stories. Base your learning content around individual use-cases, and put the learner in the center of the use-case. One easy trick for designing story-driven learning is to start each topic with something like, "Imagine you want to do..." and then walk though that experience. It makes the learning organic and real, and helps make sure you get rid of the stuff that doesn't need to be there. If it doesn't show up in a use-case/story, are you so sure you should be teaching it?

Use pacing and vary the parts of the brain you're exercising.



Learning--and especially *memorization*--doesn't happen at an even pace. Brains--or especially *parts* of brains--get tired and lose focus. By varying the pace--and type--of learning content, you give a user's brain the chance to let one part rest while the other part takes over. For example, follow a heavy left-brain technical procedure with a big-picture example/story that covers the same topic. This helps the learner's memory in two different ways--the redundancy means two different chances to save the information, and the fact that you gave one part of the brain a break while shifting to a different part keeps their brain working longer without fatigue.

Think about it--if you hopped up and down on your right foot repeatedly, that right leg would give up after fewer repetitions than if you kept switching from right to left. Pacing--by frequently switching which parts of your body (or in this case, brain) you're using--lets you stay fresher for a longer period.

Also, recording something to long-term memory is rarely *instant* (although the stronger the associated emotion, the faster (and more likely) your brain is to record it). Memory is a physical/chemical process that happens *after* you've been exposed to something, and if anything interrupts the process, the memory is not stored. That's why people with serious head injuries often cannot remember what took place just prior to the injury--the process of recording those things to long-term memory was stopped.

If you want someone to *remember* something, you must give them a chance to process that memory. Relentlessly presenting new, tough information (like tons of code and complex concepts) without also including chances to reflect, process, think, apply, review, etc. virtually guarantees that much of the learning will be forgotten.

Remember, it's never about *you*. It's about how *the learner feels about himself* as a result of the learning experience.

Don't use learning content as a chance to show off your knowledge--that virtually guarantees your content won't be user-friendly. Use it as a chance to help someone's life a little.

A successful learning blog is about helping the readers learn and grown and kick ass! Make that happen, and your stats will take care of themselves. In contrast, the best way to ensure a *low* readership is to assume that readers are into *you*. Offering users nothing but your opinions, however well-reasoned, might not be enough to make it worth their scarce time and attention.

"If you teach it, they will come."

http://headrush.typepad.com/creating_passionate_users/2006/01/crash_course_in.html

REAL motivation posters

By Kathy Sierra on January 14, 2006

"The customer is always right." "Employees are our greatest asset." "The customer is why we're here." You see those lame posters in businesses across the world. They mean little, and they motivate *nobody*.

Yesterday, Bert and I were in Miami after giving a "creating passionate users" talk on Marco Island. Bert had a printing emergency, and the Hilton's business center printer was glacial. Bert mentioned it to the front desk clerk who was a *hero* and had Bert come behind the desk to the office and use the *way faster* printer back there.

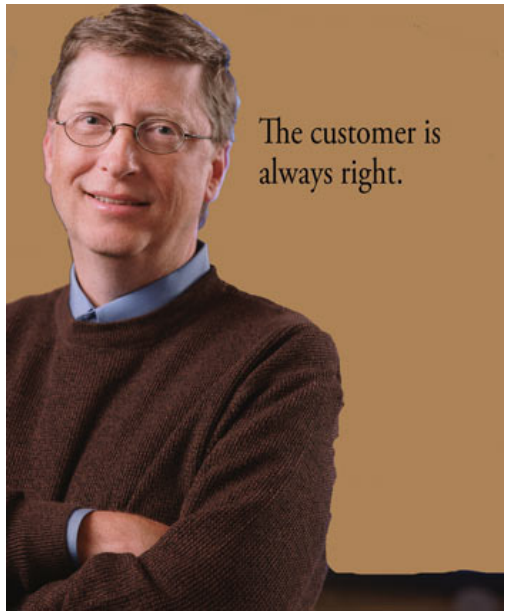
Fast forward 15 minutes, when the manager-on-duty walks behind the desk, sees that A Customer has Been Allowed Behind The Desk, and proceeds to rail on the clerk--telling her that customers are NEVER supposed to be back there, and what did she think she was doing, and never do that again, etc.

Ironically, the one thing Bert "saw" behind the desk was a for-employee-eyes-only poster that said in 72-point bold type, "The Customer is why we're here." and "The Customer is NEVER an inconvenience."

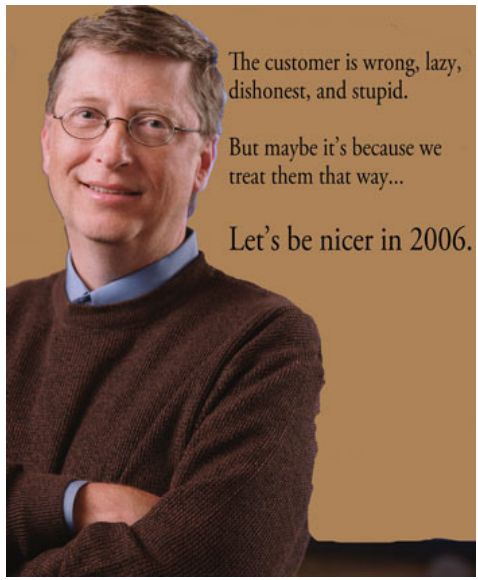
Then I recalled the time I worked on the interactive version of Oracle's Annual Report. We did a video shoot of Larry Ellison saying openings for each chapter of the report. If only I'd kept the outtakes for the one that had my team on the floor--Larry was reading a script for the opening of the "People" chapter and had just completed the phrase, "Our employees are our greatest asset," when something off-camera pissed him off. He threw the script down and began ranting and swearing, including the words, "What is this crap? I want somebody fired for this s***!"

So, just how useful *are* those cliched slogans? (Including my personal worst--"None of us is as smart as all of us.") Maybe we should replace them with something a little more *real* for 2006. Here are three of my before-and-after posters...

Bill Gates *fake:*

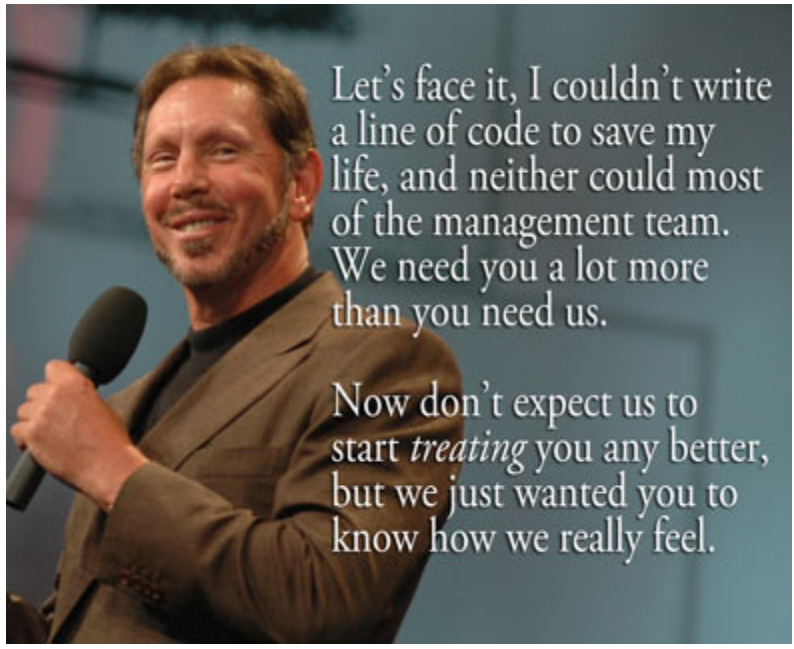


Bill Gates *real:*



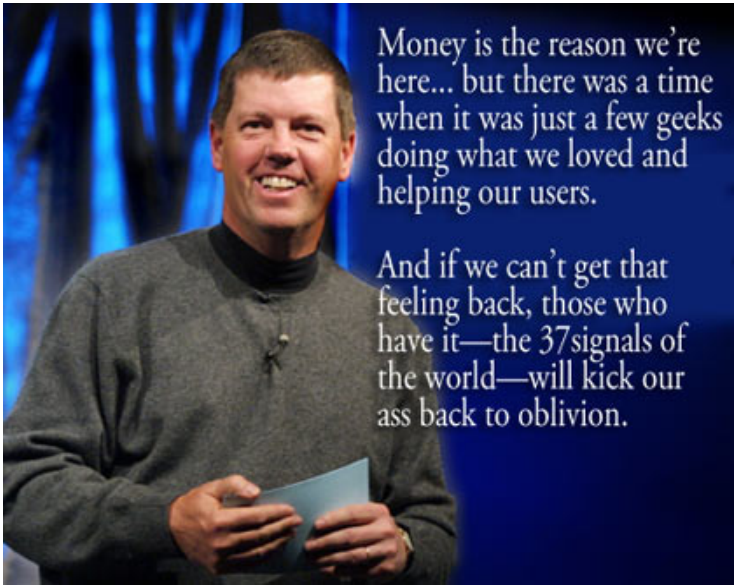
Larry Ellison *fake:*

Larry Ellison *real:*



Scott McNealy *fake:*

Scott McNealy *real:*



Here's to a more real 2006 ;)

[And don't forget to visit Despair Inc., for a look at the world's best *demotivational* posters.]

So, what meaningless slogans can *you* replace (or at least destroy) today? Hmmm... I wonder what would happen if you changed some at work... would anyone even notice? That's the problem. It's not like anyone *reads* these things, let alone *applies* the message. Motivation--especially when it comes to a deep concern for the users--does not come from *saying* it. It comes from a culture of *meaning* it.

http://headrush.typepad.com/creating_passionate_users/2006/01/real_motivation.html

Death by risk-aversion

By Kathy Sierra on January 30, 2006

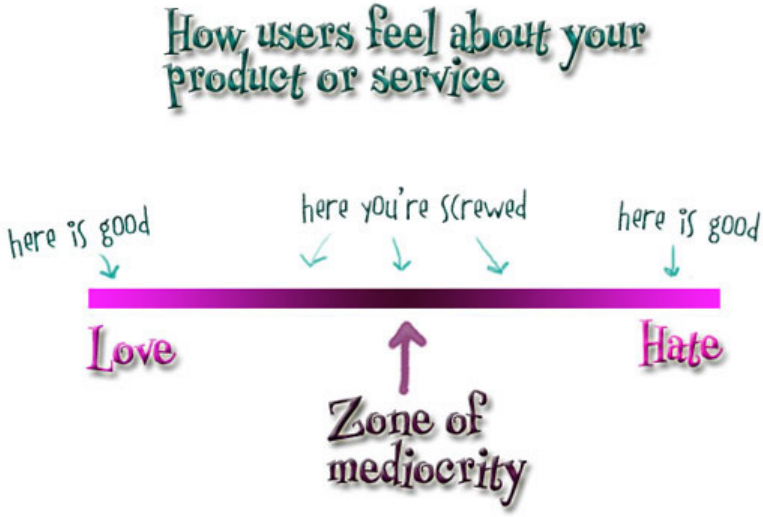
Memo to Microsoft: you've got people doing some *amazing* things over there. If you could just ***get the hell out of the way***, the world might change for the better.

Risk-aversion is the single biggest innovation killer, and of course it's not just Microsoft that's been infected. Taking risks is... risky. But if *not* taking risks is even *riskier*, then WTF?

Sure the big companies have it *bad* and may fall the hardest if they don't get a clue and a cure, but none of us is immune. You see the safe path *everywhere*. Today at lunch I had one of those conversations with a co-author about the cover of the next Head First book, and there I was suggesting a "safer" cover model than the one he wanted (complete with all the logical reasons

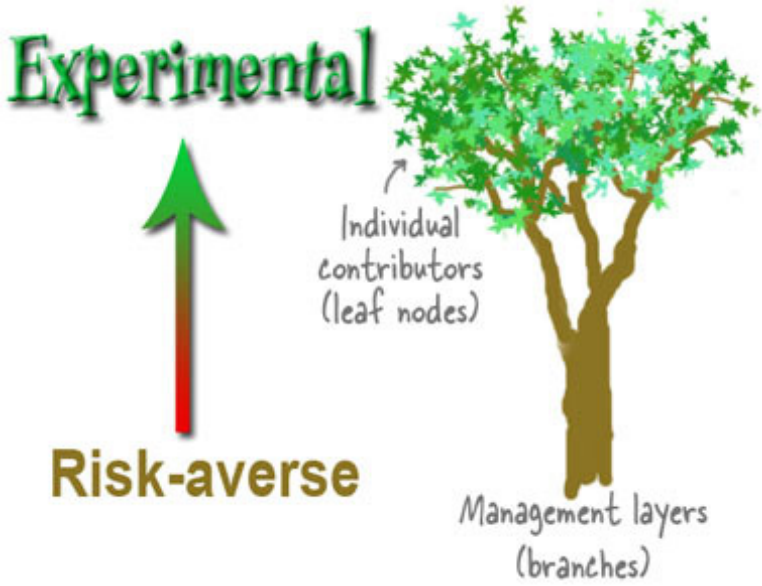
why people could complain about his choice). I still can't believe the words that were coming out of my mouth.

Blogging has not made this easier... if anything, the idea that a gazillion bloggers and commenters (or even ONE loud one) will seize any opportunity to find fault with your ideas and attempts can dampen one's willingness to be brave. So here's my quarterly reminder to all (me included) that if you're not doing something that someone hates, it's probably mediocre.



But back to Microsoft... as I said in my previous post, Robert Scoble kept using the phrase "risk-averse" when defining some of Microsoft's problems. And I heard the same thing from Liz Lawley, who has been fascinated by the disconnect between the wonderful ideas MS employees have for products and services, and the final products and services released to the public. Somehow, according to Liz, fear steps in between those two points.

But *whose* fear? The metaphor Liz used (she got from someone else) was that many of the "leaf nodes" (what Microsoft and Sun and others refer to as "individual contributors") tend to be innovative and brave, but many of the "branches" (i.e. *layers of management*) can't stomach the risks. In their (admirable) desire to be strong and stable, the "branches" put safety above all else.

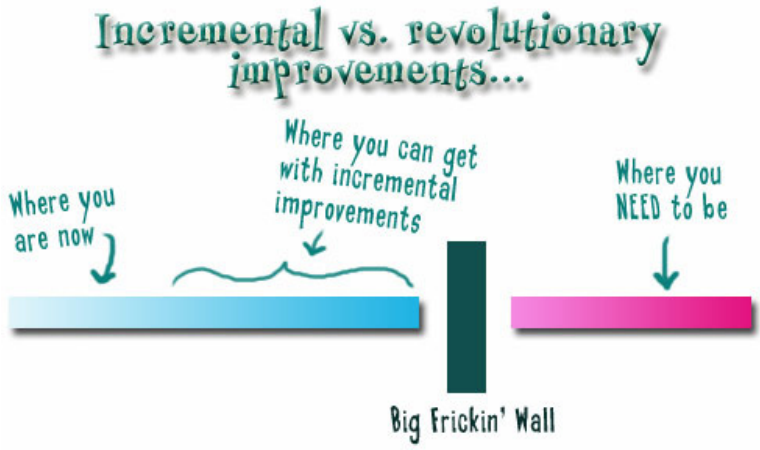


What *kind* of safety? Sometimes managers are putting the best interests of the company first. That's great--they're often more experienced and have a better grasp of the bigger context. But (and it's a really big but) sometimes they're just worried about their own damn job. In other words, the leaf node/individual contributors often think about the effect of their work on *users*, while the mid-level managers often think about the effect of their work on *their job*. And whose fault is that? All those layers of bosses. Even *one* risk-averse boss in the chain-of-command can do major damage to innovation, spirt, motivation, etc.

So add one more skill to our career advice for young people: **be willing to take risks!** Perhaps more importantly, be willing to tolerate (and perhaps even *encourage*) risk-taking in those who are *managed* by you. Of course I realize that this is much easier said than done. I was a "leaf node" at Sun, and a zillion other places before that. I've even done a little time as a "branch" (and I sucked at it).

But can anything be *done* about all the spirit-squashing risk-aversion? *Recognition is the first step*. Unfortunately, those who *recognize* it tend to be the leaf nodes--the ones with the power to *create* and *implement* the ideas, but very little power to *authorize* them. Those with the most potential to create change are the *branches*. The Managers With a Clue.

I had one of those at Sun--Jari Paukku, now a "Senior Change Management Specialist" for Nokia in Finland. He knew how to walk that fiber-thin line between keeping those above him happy while keeping those below him from losing passion. He knew how to pick his battles, and believed in doing the right thing for both customers and his team. But he also knew that getting himself *fired* wouldn't do his "leaf nodes" (like me) any good, so he didn't let us run wild with *every* cool idea that popped into our heads. If the idea had value, though, and was the right thing for customers, he would help us figure out a way (even if it was through a sneaky back door) to make it happen, or at least plant the seeds of possibility.



I do have a *few* general tips for dealing with risk-aversion:

Regularly review your sacred cows

Regularly review the assumptions behind all your decisions

Are those assumptions still valid?

Practice LETTING GO

Here's where the Buddhists have an edge. Too many of us hold on to practices or ideas (including sacred cows) long past their sell-by date. If it doesn't serve us any longer, it's time to give it up no matter how well it served us in the past.

Of course, "letting go" means temporarily experiencing that painful, awkward, "I suck" stage again. But pro athletes do it if they want to break through plateus. Go players do it to move up in ranks. Musicians let go of habits and styles. Programmers do

it (waterfall anyone?). Writers do it. Anyone who has switched from skiing to snowboarding (or switched from regular to "goofy foot") has learned to let go.

Stewart and Caterina did it when they let go of their product being strictly a *game*, and allowed it become Flickr. O'Reilly did it when they let go of technical books being strictly about *text* (or for that matter, that books were strictly about *print*.)

Easy and familiar is safe, but often comes with built-in, unscalable walls. You can't get there from here.

Push the boundaries *strategically*, one-by-one

Whether you're a leaf or a branch, pick your battles carefully, one poke at a time. Better to live another day to keep fighting the good fight then, say, being fired for trying to do it all at once.

Use blogs to build support within the company

The collective power of all those Microsoft employee bloggers is helping (one small step at a time) build support for the leaf nodes, and the braver branches who manage them. (And yes, this particular tip was Robert Scoble's advice to other Microsofters fighting risk-aversion).

If all else fails and the culture of risk-aversion is stealing your soul, consider going into "short-timer" mode.

This was my path at Sun, and the one I recommend when you're dangerously close to losing heart. You'll probably be fired, but at least you can do some good on the way out by making a *lot* of noise. And you never know what will happen later... today, just a few years after my most ungraceful Sun exit, I not only contract with Sun in several key areas (including leading the development on most of their programmer/developer certification exams), but I'm also a founding member of Sun's Java Champions (I know, lame name, what's with the "champs/champions" thing and tech companies?) program. I find myself in conference calls with departments that still house a few folks who desperately wish I'd switched to .NET ;)

Keep reminding yourself that life is short!

One of the benefits of having a scary illness or major loss is that it reminds you of just how much time is ticking away, and that you always have options to make changes. If you have a great idea, what do you risk by *not* persuing it? Will you have more regrets if you try and fail than if you don't try at all? Some of the best and biggest ideas happen within the scope of large companies, but some of the most world-changing happen... *elsewhere*.

http://headrush.typepad.com/creating_passionate_users/2006/01/death_by_riskav.html

It's the [?], stupid!

By Kathy Sierra on February 1, 2006

Do you *know* what the "comma-stupid" phrase is for your product or service? In other words, do you know what is most meaningful for your users? Because whatever that word or phrase is (i.e. the part that comes before the ", stupid!"), it should be driving *everything* from product development to documentation to support and marketing.

The "comma-stupid" phrase popped into american culture in 1992, with the political message, "It's the economy, stupid." But the one that got me is from the game industry. In the early 90's,

a company was founded that most agreed was a dreamteam combination of Hollywood and Silicon Valley ("sili-wood", as it was called then). Made up of some of the best and brightest from both the entertainment and high-tech worlds, "Rocket Science Games" was a cover story on Wired before their first product was released.

And they got more than $10 million initial financing. Those of us struggling to develop CD-Rom games on $100k to 300k budgets were envious and a little resentful. Sure enough, Rocket Science's first games were technologically spectacular. But...*they were complete and utter failures in the market*. And then a rumour started that one of the founders allegedly said something like, "Oh, we get it now. It's about the *game play*." I have no idea if anyone ever said that, but the meme spread throughout our little entertainment/tech game world that went "Well... DUH! It's the game play, stupid!" Of course, defining what *game play* meant in that context is an entirely different subject for another day... ;)

Meanwhile back at Microsoft... in the Search Champs program they talked and demo'ed and listened and demo'ed and questioned and talked and demo'ed. Throughout most of it, I saw and heard about features. Cool features. Innovative features. Sometimes *jaw-dropping* features. But I almost never heard any over-arching context for this collection of features. I almost never heard discussions of the ***meaningful benefit*** this collection of stuff would give the user. In fact, most discussions about the user's perspective were around usability.

Usability schmusability... where's the part where we talk about how this helps the user *kick-ass?* So *what* if the feature is brilliantly implemented and dead simple to use? Where's the part where we ask, "use it do WHAT?" Where's the part where you say, "We've decided that our mission and message is, "It's the [something-goes-here], stupid!"

Framing it this way might not change the product one bit. Or it might change it profoundly. It might mean a deep change in the way we *talk* about and *teach* and *support* users in our products. And yes, of course, it might mean a deep change in how we *market* that product or service. But if we don't figure it out and stay focused and clear, we risk heading in directions that don't serve the user's ultimate purpose. (The most obvious result of not staying "on message" is *featuritis*.)

We all have to figure out what the user cares most about, and drive *everything* from that "comma-stupid" phrase. For our books, we use, "It's the ***learning experience***, stupid!" Not the *depth of technical expertise*, not the *breadth of coverage*, and *definitely* not the *writing quality*.

Yet, even *that* still isn't getting to the heart of what matters to the user. Because somewhere behind "learning experience", there is a *reason* why the user *wants* that learning experience! And it is really *that* reason that matters. In other words, "learning" is not really the user's ultimate destination/reason for picking up one of our books. It's simply a means toward some *other* goal. But what is that *ultimate* goal? That's what we all have to ask, uncover, and focus on for whatever we're creating/offering/teaching/evangelizing.

And that's why we're so fond of the phrase, "kick ass", because it serves as a placeholder for what the user ultimately wants. That "I Rule!" experience should drive what most of you are trying to build or promote. [Note for clarification: we mean "I Rule!" like that "YES!" feeling you get when you do something tricky, *successfully*.]

Some of the most common high-level answers to the "comma-stupid" phrase (and which are forms of "kicking ass") are:

It's the [user spending more time in flow], stupid!

It's the [user feeling a sense of belonging], stupid!

It's the [user having more sex], stupid!

It's the [user experiencing peace of mind], stupid!

It's the [user having more fun], stupid!

An example of a NON useful answer might be:

It's about the [user getting more work done], stupid!

While this may be the key benefit of your product, it's not the user's ultimate goal. You must ask, "how does getting more work done help the user kick ass?" The answer may be, to spend more time in flow.

But how do you know when you've arrived at the ultimate answer? After all, "spending more time in flow" isn't necessarily the end state, right? Well... given that the flow state has been linked to human happiness, we consider it an end state. Ditto

with sex, belonging, peace of mind, fun, and other meaningful states that speak to deeper human needs and/or desires. But really, the way you phrase your "comma-stupid" depends entirely on what keeps you and your team motivated toward the right things.

I believe we all should spend time--a lot of time--figuring out exactly what should be in our "comma-stupid" phrase. We can start by asking, "What does the user care about?" Followed by, "OK, but WHY does he care about that?" Follwed by, "And why does he care about *that*? until we get to the heart of it. Then we pick a phrase... a message that expresses this in a way that everyone on the team can understand. Then from that point forward, *every* decision should include two questions:

1) How will this [thing we're about to do] support, enable, or amplify what the user cares most about?

2) How will this [thing we're about to do] potentially *hurt* or stand in the way of what the user cares most about?

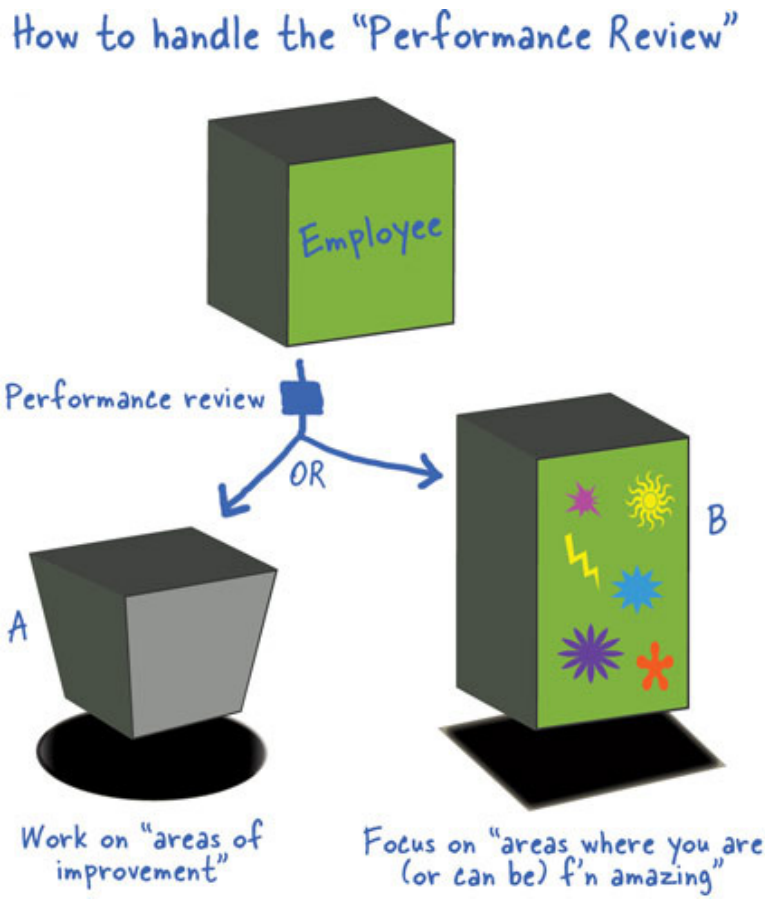
And I actually believe that for 90% of us (my work included) the answer to the "comma-stupid" question is "the user kicking ass", but of course it's up to us to define exactly what "kicking ass" means for our particular context. So that's my challenge to you--ask yourself if you have a clear, "It's the [something], stupid!" Then ask yourself if it gets to the real heart of what is most meaningful to the user. In other words, if you say, "It's the usability, stupid", you aren't there. You could have a highly usable tool that doesn't help the user in something they can kick ass at. And once you have that clear message, take a hard look at your product or service and see how much of what you have supports, enables, or amplifies that user goal, and see how much--if any--stands in the way.

Most importantly, keep asking yourself, "How can I help my users kick ass?" And to answer *that*, you'll have to know the context in which users interact with your product or service. Chances are, whatever you provide is NOT their ultimate goal. It's just a tool to get to something that *is* meaningful.

http://headrush.typepad.com/creating_passionate_users/2006/02/its_the_stupid.html

Mediocrity by "areas of improvement"

By Kathy Sierra on February 6, 2006



How many times in your life (school, career, relationships) have you been told about your "areas of improvement"? How much time and energy have you spent *working* on those areas? If you're a manager, how much emphasis do you put on those areas during a performance review?

Maybe instead of working on our *weaknesses*, we should be enhancing and exploiting our *strengths*? What if the price for working on weakness (and who even *decides* what is and isn't a "weakness"?) is *less chance to be f'n amazing?*

There are several books out about this, although I haven't read them -- but the idea gets my attention:

Teach With Your Strengths, which says on its Amazon page,

"Defying the orthodoxy that teachers, to be more well rounded, should work to strengthen their weaknesses, this book, drawing on research by the Gallup Organization, maintains that great teachers are those who teach with their greatest talents and abilities."

That book is an expansion of the ultra best-selling Now, Discover Your Strengths by Marcus Buckingham. I don't know if the books are actually good, but again, it's the idea I enthusiastically support.

Too many companies (and managers, spouses, etc.) focus too much on bringing everyone up to some level of competency in a laundry-list of attributes including time-management, communication skills, writing ability, filling out TPS reports, teamwork/teamplayer, attitude, organization, sensitivity, adhering to corporate goals and policies, etc. Clearly, there is some minimum threshold for each attribute beneath which a person might be *impossible* to work with no matter what the situation. But too often those minimum thresholds are set MUCH TOO HIGH and not specifically tailored enough to the individual.

By focusing on "areas of improvement", we're putting a square peg in a round hole. What do we end up with? A crappy, rounded off peg who meets the minimum thresholds *at the expense of their most kick-ass attributes*. What if let ourselves (and those we manage) spend a lot more energy in the areas where we are--or could be--amazing? I suggest taking a very hard look at the "areas of improvement" list and see if we can rearrange the context so that those things become less important. In other words, why don't we try to make a **square hole**?

I know that everything I've said here can be abused and used as an excuse for poor performance in *every* area. But remember, this is about tradeoffs -- so I'm assuming that we're cutting some "areas of improvement" slack to those who demonstrate that they HAVE areas in which they are--or could be--amazing. And I'm also assuming that those areas have some real potential use/benefit. But really, do your best programmers need to be

filling out their TPS reports? How many of us have lived through the cliched scenario where the time-sheets we fill out need an entry for "time spent filling out time sheets"?

OK, I admit I have a thing against performance reviews in general, but if we must have them, I'd love to see some big changes to the typical form. I'd like to see a teeny, tiny space reserved for "areas of improvement", which lists only those things deemed absolutely critical that are below the minimum threshold, and I'd like to see a BIG space titled "Areas where you are (or can be) f'n amazing." Then a plan can be custom-tailored for removing not the areas of weakness, but the things which make those weaknesses a problem (and which get in the way of using their strengths).

And this isn't just for employees--many of us need to think about this in our startups (something I'm just beginning to deal with now)... are we trying to exploit our strengths, or are we in a position where we're forced to spend too much precious effort improving our weak areas? To use the business cliche, are we trying to do business in areas that aren't our "core competency"? Agile companies are those who can turn on a dime and recognize when an area might be profitable but is slowly leading them in a direction away from their unique strengths.

If we have everyone working on their weaknesses, we do smooth out the attribute curve. But then we get mediocrity in a wide range of areas, and less f'n amazing work in narrow ranges. For many of us, we just can't *afford* mediocrity. There's too much competition there.

So, what can we do to make more square holes?

http://headrush.typepad.com/creating_passionate_users/2006/02/mediocrity_by_a.html

Re-igniting passion

By Kathy Sierra on February 9, 2006

I assume that most--if not all--of us reading this blog *chose* the professions we're in (or studying for), whether it's programming, design, teaching, marketing, music, running a business, heading a church, writing, whatever. But the tough problem is holding on to that initial feeling of energy, enthusiasm, optimism, *passion* we have at the beginning.

I recently saw the most dramatic difference between that initial passion and the just-a-job attitude, when I went to the CUSEC conference (Canadian University Software Engineering Conference) in Montreal. What happened at that conference was something I never could have expected.

[UPDATE: article about the conference in the Concordia Journal]

The first strange thing was that this well-organized, well-run conference was put on entirely by... students. From the moment I got there, I kept looking for, well, the *adults*. Here's the conference chair/founder (long-term student John Kopanas, in the front center) and the rest of the commitee:

The second strange thing was that these young student organizers, and ALL several hundred of the attendees, expressed a passion for software engineering that I had *never* seen or heard! Not in my 20 years in technology. It was impossible to talk about anything meaningful in the tech world without hearing about a fresh approach, new perspective, or a sincere and thoughtful concern. Their energy was shocking. The conversations hopped from professional ethics to the cultural impact of their work as software engineers. Ethics? Cultural impact of their work? Half these guys weren't even old enough to drink beer in the US.

I don't know what they put in the water up there, but we could use some down here. Because too many of us (me included) tend to let that early enthusiasm slide... we forget why this thing we do used to matter to us, and we might start wondering why we ever got into it the first place. ***We start phoning it in.***

Forget how that ultimately affects the end user, what about *us*? What would my days be like, for example, if I could always remember (and re-experience) the "I Rule!" feeling I got when my first program compiled. What would it be like if I remembered how excited I was to get my very first email from a reader telling us that the book *made a difference in his life*? What would it be like if I remembered how lucky I am to be doing something that--at some point, anyway--I really REALLY wanted to do?

I'm reading a book right now that is probably the BEST book on teaching/learning I've ever seen--What the best college teachers do, by Ken Bain. It involves a long-term study of the *best* teachers (and "best" is measured in extremely important and meaningful ways, not simply by test scores, grades, or student evaluations). Apparently ALL of the best teachers kept their teaching fresh and inspiring no matter how many times they had to teach the same damn topic, day after day, year after year. From the book:

> *"Jeanette Norden told us that before she begins the first class in any semester, she thinks about the awe and excitement she felt the first time anyone explained the brain to her, and she considers how she can help her students achieve that same feeling."*

> *"Teaching is not acting, yet good teachers do expect to affect their audience when they talk: to capture their attention, to inspire, to provoke thoughts and questions...this practice has all the power of careful analysis, but it also entails thet energy of feelings and attitudes that no induction and deduction can achieve."*

We can't expect passionate users, if we ourselves can't hold (or rediscover) the passion we felt for the work we chose. That doesn't mean we have to love our actual *job* -- I've had plenty of JOBS that could suck the life out of the most inspiring work on the planet. But I'm not talking about the *job*, which you can change, this is about the actual *thing*--teaching, writing, programming, delivering a sermon, playing with your kids, training your dog, giving a presentation, managing a team, evangelizing a cause, whatever it is.

One year ago, I wrote about the same topic--wondering how some musicians can play the same song a thousand times as though it were the first time. So consider this my annual reminder (to myself as much as anyone else) to NOT PHONE IT IN.

How do we do that? If you've got some good tips, I'd love to hear 'em. But perhaps the simplest (and yet hardest to do) answer is to sit your ass down and force yourself to *remember*.

http://headrush.typepad.com/creating_passionate_users/2006/02/reigniting_pass.html

Rethinking testimonials

By Kathy Sierra on February 12, 2006

Do you have user testimonials on your website? Are they all about *you*? Chances are, they're letters or quotes or reviews that talk about how fabulous your product or service is, how impressed they are with you, *how much you rock*. If they're anything like some of the ones we have in our books, they suck.

They suck because the focus is all on *you*, when they should be first-person accounts of how the user kicked ass (or at least *improved* in some way) as a result of your product or service. Even better would be testimonials that described exactly *how* they used your thing to become better. In other words, testimonials-as-tutorials. Something that provided value to both existing and potential new users.

Four types of testimonials

1) Completely made-up. I'm assuming that none of you reading this blog would do something that stupid or unethical.

2) Real, but without last name and company name. You know, the ones modeled after the quotes in Cosmo or Maxxim, from *Jennifer S., copywriter,* or *Fred G., IT Director*.

These are usually worse than having no testimonials at all. Even if they *are* real, they *look* fake. With a few exceptions (for, say, hair replacement or "increase your manhood" products), there's little reason a user wouldn't give you their full name. It's

possible that a company doesn't want their competitors to know that they're using your product, but if that's the case, don't use their testimonial!

My first thought when I see these no-last-name-no-company-name quotes is that either they're made up, or the person giving the testimonial didn't like you enough to publicly endorse you. Either way, it looks bad.

3) Real, with name and company, raving about the *company*.

4) Real, with name and company, talking about *themselves*. These are the most compelling, and potentially the most *helpful* to other users (another example of marketing-by-teaching).

The one important distinction I didn't make in those categories is that there are *some* testimonials whose value is in lending credibility. These are the endorsements from people who other people trust, and who may not be actual users themselves.

These credibility endorsements were crucial to our Design Patterns book, for example, because we took an extremely important and serious topic and made it appear... less serious. That key people in the patterns community (including Erich Gamma and Ward Cunningham, inventor of the "wiki") endorsed the book lent a valuable credibility to it.

But when I look at the most prominent quotes on our back cover and inside pages, way too many fall into the "The book is great" or "You guys are great" category, and only a few talk about the user. But

here's a quote (from an Amazon review of the patterns book) that we *should* be using:

"Sitting in a developer's meeting yesterday I was really surprised that, while I clearly didn't have the years of experience the other coders had, I had no problem keeping up and was even able to contribute. I'm now moving in to the new assignment fairly well and am confident that I'll be able to pick up the details of this language now that I've got such a good grounding from this book."

And I'd love to go even further--to start using some category 4 testimonials that would be more helpful to others. For example, people in online forums often share study tips with others on how they used the book, or--even better--they talk about how

they *applied* what they learned to improve their real-world work. Those tips make for more useful testimonials.

DoubleClick heading in the right direction

Noah Brier sent me a link to the DoubleClick main page, which recently moved to a more user-focused marketing approach. It showcases real users right up front at the top of the main page, which I think is *great* (much better than what I've done so far), and the quotes are a combination of category 3 and 4. The quotes aren't big enough to have useful tips, but I applaud them for moving in the right direction.

37signals testimonials... need improvement

I checked out the 37signals main page, and it's full of testimonials from category 3--all real, but talking mainly about how great the product is. Many of these are important for credibility (if BusinessWeek or the Wall Street Journal said something about my product, I'd definitely be showing it), and having lots of users say how much they loved it is important to prospective users. But of all the main page testimonials, only a few are what I'd call category 4.

Here are some of the best ones...

> *"Basecamp has already been key to winning a project, being the main thing that differentiated us from a very close competitor, and it's had a massive, positive impact on our working practices, even after just a couple of weeks."*

(Describes how the user kicked ass)

and

> *"I've spent the last 3 days working out strategies to completely revamp and streamline my record-keeping and information-storage strategies for my work, personal interests, and scholarly projects. As a librarian, someone in the information organization and retrieval business, this is particularly exciting!"*

Not so fast... 37signals has case studies!

OK, so they get about the same grade for their main-page testimonials as I'd give us for our books... C. But, the 37signals folks have something wonderful--category 5, we'll call it, on their case studies page! And it was this page, in fact, that helped sell *me* on using Basecamp.

So, A+ for the case studies, guys. If you could find a way to pull a few useful pieces out of the case studies, and include them as part of your testimonials, that would be even better. That's something we could all do--get some case studies, and some helpful testimonials from that.

The main point: users don't care about *you* as much as they care about themselves.

Make the users the heroes--the stars--instead of the company, product, or service. Once you've covered your credibility base, wouldn't users rather hear how other *users*--people just like them--have used it to kick ass in some way? Anything we can do to elicit first-person language from our users, that talks about what they themselves have done, is far more valuable than a glowing report about you.

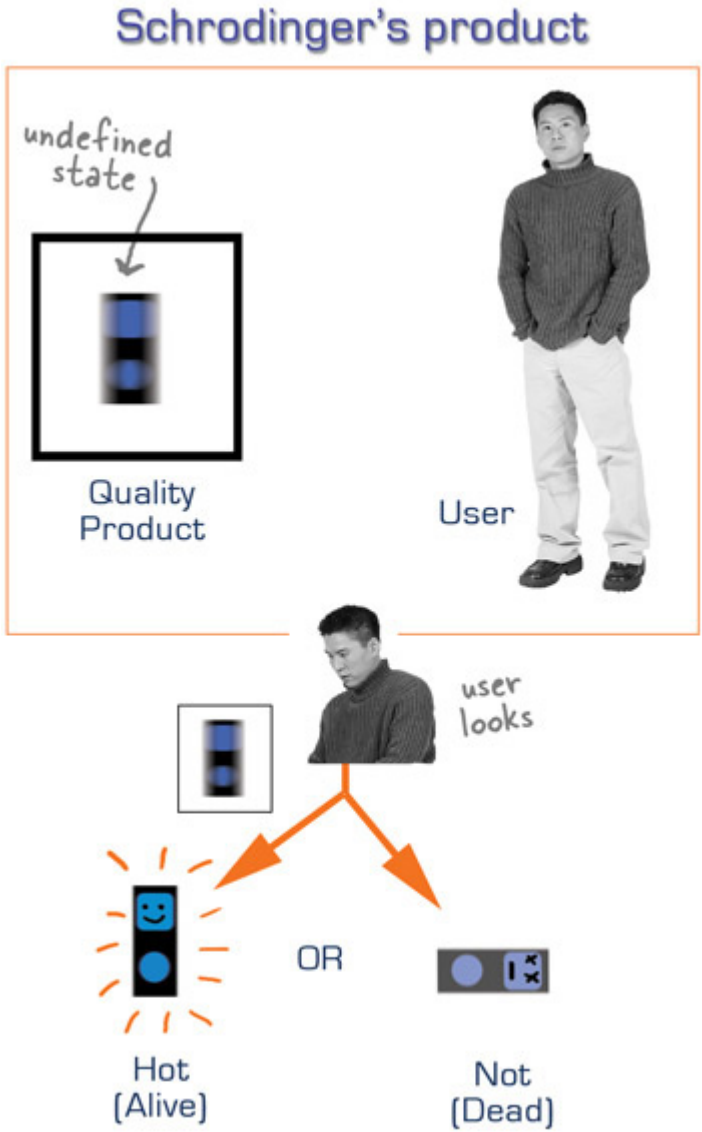
If any of you have examples of good testimonials, I'd love to hear about it.

[bonus link: Author, speaker, and branding consultant Tom Asacker has a wonderful PDF (be warned-17 MBs) that's full of great info presented in a visually compelling way. I highly recommend it. It's not about testimonials, but the overlap between what Tom talks about and what we talk about here is heavy. I love this guy, despite our disagreements on a few thing ;)]

http://headrush.typepad.com/creating_passionate_users/2006/02/rethinking_test.html

Schrodinger's Products (ten ways to be desirable)

By Kathy Sierra on February 15, 2006



If a company makes a high-quality product, but user's don't find it sexy or appealing, does that product exist? Continuing our quantum physics theme, we did our own little "thought

experiment" about that. One conclusion could be, "You have nothing until a user *wants* it."

(Shrodinger's Cat review)

[Update: trying to force this idea into anything remotely resembling real physics wasn't working, so I took it out. Consider it a very *loose* metaphor for this idea. A real stretch...]

So maybe that's a model for what our products are like. We make a high-quality product, but it isn't really alive/dead (hot/not, successful/unsuccessful, hit/flop) *until a user finds it desirable*. Unlike earlier days when there weren't so many choices and one could compete on features, product quality alone guarantees nothing. We can make a solid, bug-free, easy-to-use, feature-rich product, but we can't know it's true state until a user looks, thus collapsing the wave function. Is it desirable? Is it appealing? *Is it sexy*?

How do you change your odds? How do you reduce the chance of that "kill" trigger firing? It's tricky, since we already know that listening to users isn't the most reliable way to know we're on the right track. Perhaps the best we can do is stay more focused on the user's perception and experience than on the actual product itself. It's so easy to get caught up in feature lists, implementation quality, performance metrics, etc. and then miss the whole point. We focus on the trees, not the forest. We create a product that flawlessly meets a checklist, but that nobody lusts after. It's like the blind date your friends keep describing as, "Yeah, but he's got a great personality and he's smart and funny and..."

There's no denying the basic human fact that *chemistry matters*. *Looks* matter. *Sexiness* matters. Fortunately, we can define "sexy" quite broadly. The iPod is *sexy*. To a programmer, a slick, elegant framework can be described as "sexy". Some cars are sexy.

And if we're talking about *desirability*, sexiness doesn't always have to be in the equation. Things which evoke "good feelings" can be intensely desirable, even if those feelings are about having fun. Something that makes you say, "God, that's the cutest thing I have ever seen can be desirable. And you know I'm going to say it (Wally, cover your ears) -- *something that helps you kick ass* can be desirable.

So, back to the real question... what can we do so that when the user "opens the box", the wave form collapses in our favor? I don't know, but I'll throw some ideas out there and I'm hoping you will add more:

Ten ways to make your product *desirable*

1) Pay attention to style.

Aesthetics mean more today than they did even fifteen years ago. And don't be thinking that this does not apply to *your* product. Remember, this is like dating... it's not "selling out" to wear your good shirt on that first date, and first impressions matter deeply. For that cat, remember, that first look was life or death. (yeah, yeah, yeah, that was different -- radioactive decay and all that -- but I'm taking metaphoric license)

2) Pay attention to the emotional appeal.

Besides the product itself, this might include packaging, your website, documentation, anything that the user might see before making a decision.

3) Show it in action... with real people.

People are drawn to people. Brains pay attention to people. Seeing another person using the product or enjoying the service, whatever, is more powerful than just showing the product sitting there (exceptions are made, of course, if your product is inherently sexy and compelling all by itself. That's a little harder, though, for a software screen...)

4) Don't use pictures of generic shiny happy people that have become cliches.

I said, "real people." And use your most compelling testimonials. But... real doesn't have to mean *unattractive* or unappealing. Yes, we wish that people didn't have such a shallow perspective, but this is simple neurochemistry, and part of what makes us human is our brain's ability to seek out and respond to things *it finds attractive*. And many of the things that attract the brain (not necessarily the mind) are things it believes look "healthy." No, I'm not saying put a naked girl on the product page--that's way too unimaginative. But that doesn't mean it wouldn't work, unfortunately.

5) Make sure it's clear to prospective users how this helps them kick ass

The more obvious this is, the more compelling the product or service. Be sure the user can see a clear path to getting up the curve (if there *is* a learning curve), and demonstrate exactly how you will help the user get there. In this lifetime. This is where training and support really matter. But again, it's not enough to *have* good training, documentation, etc. -- you have to make sure this is clear to the prospective user.

6) Appeal to as many senses as possible.

Even if your product exists solely in software, use colors, shapes, and potentially sounds (audio is tricky, and a whole separate topic) to give users a sensation of touching or hearing something (heck, pictures of food may make them smell and taste something). Consider podcasts and video, or even song lyrics or poems. Think about rhythym.

7) Make it meaningful.

Give them something to believe in. Something *real*. I don't need to lecture any of you on ethics, so I won't. Something to believe in could be an approach to development, like the 37signals Manifesto, or it could be the charitable causes you support (and encourage your users to support), or it could be a philosophy that resonates with the user. (Watch the Sarah McLachlan Worlds on Fire video for inspiration.)

8) Make it justifiable, so the user doesn't have to feel guilty

We all make decisions emotionally, and rationalize them later. Helping the user with that after-the-fact rationalization makes it that much easier. Almost nobody makes a decision based *solely* on the hard, rational facts, but they are comforted by knowing that they "made a smart decision." Remember, even if your product does nothing more than help someone have a more enjoyable time, that's potentially a mental health/stress management benefit.

9) Support a *community* of users

We all want to belong. Products and services with affinity groups are a HUGE added value for users, whether it's the confidence of knowing you'll get technical help/support, or the joy of being part of a "tribe" we can be proud to belong to. (Make sure you give users a way to "show off" their affiliation--practice T-shirt First Development)

10) Never underestimate the power of fun.

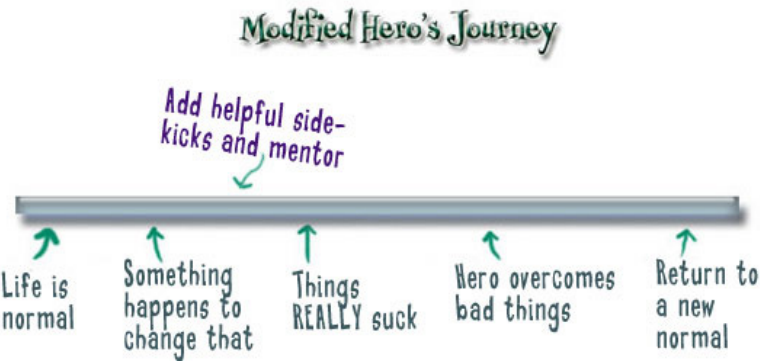
Humans--all mammals--have a very strong play drive; it's crucial to our survival. Show someone how you can help them have a little more fun in their life, and you might be irresistable.

Remember, desirability does not necessarily mean we'll have passionate users, but it's a crucial first step. The more desirable the product, the more likely the user is to want to spend more time with it *getting better*. And it's the *getting better and better* part--the learning and growing--that is the foundation of passionate users.

http://headrush.typepad.com/creating_passionate_users/2006/02/schrodingers_pr.html

Where there's passion, there are stories

By Kathy Sierra on February 15, 2006



Anyone who is passionate about golf knows the backstory about Tiger Woods. Those passionate about film can tell a story or two about Fellini, Eisenstein, and Kubric. Those passionate about open source know the story of Linus, Stallman, and the stories from Revolution OS. Most serious Mac lovers know a good bit of lore from the Revolution in the Valley. Heck, those who are *really* into Web 2.0, or FlickR, know the backstory of Caterina and Stewart's it-was-supposed-to-be-part-of-a-game creation.

From "creation mythology" to gossip to heroic against-all-odds tales, one of the ways we judge whether someone is truly *passionate* (as opposed to *just* enthusiastic) is if they know the key people and their stories. Do your users know your story? If not, you might consider writing it down and making it public.

But what if it's not interesting? Look again. Are you sure there isn't *something* worth telling (and more importantly, that others will enjoy *re*-telling)? No? If your founder story is just...too... dull... then find a compelling *user* story. After all, Tiger Woods didn't *invent* golf. Ask your users if they have an interesting, "hero's journey" story that somehow involved your product. It may not be dramatic enough to count as actual "lore", but it's a start.

Look at your website. Do you have a backstory there?

We don't, so we're writing one -- although many of the bits and pieces of the Head First story is somewhere in this blog. Short version -- Sun tells me that my ideas about learning are *bad*. I

tell *them* that *their* ideas are bad. That didn't go over well. I'm kicked out and vow to put EVERYTHING I wasn't allowed to do there in a book series, to prove that the learning theory was sound, just for the satisfaciton of saying, "Told you so!" I get Bert to help me, and we create and submit an unsolicited proposal, cold, to O'Reilly, who had never heard of us (we had never written a book at the time we sent the Head First proposal). Tim O'Reilly loves it, most of O'Reilly hates it. Most other tech book authors hate it too. The only bright spot was shortly after it was released when author Dori Smith, of whom I was a fan, sent me an email saying, "I saw Head First Java in the store and told my husband that 'this is the book I wish I'd written'" She had no idea (until now) how much of a turning point that was, after we'd been taking such a beating from other authors. (Some day I'll say more about the details of that first year, and why so many people hated the book.)

Oh yeah, before O'Reilly, we submitted it to two other major publishers. They turned us down. One of the editors who turned it down got into trouble when the book-he-turned-down went to #1 on the Amazon computer bestseller list. Tim has been known to "congratulate them on their fine judgement" ;)

Today it's one of the most successful new computer book series (in unit sales and revenue) since the bubble, has been on the Amazon Top Ten Computer Books each year since we started (2003), won the Jolt Cola / Software Development Award for Best Computer Book, and has survived two slashdot reviews. The whole "creating passionate users" thing grew out of our desire to teach the brain-friendly principles we use to others, and discuss how they can be applied to other things. It started as a talk to other authors and editors, then became this blog, and a book is in progress. This time, we got lucky and we were *right*. I'll spare you all of the other things we did that *weren't* ;)

Your turn.

http://headrush.typepad.com/creating_passionate_users/2006/02/where_theres_pa.html

Brain death by dull cubicle

By Kathy Sierra on February 20, 2006

You always knew that dull, boring cubicles could suck the joy out of work, but now there's evidence that they can change your *brain*. Not mentally or emotionally, no, we're talking *physical structural changes*. You could *almost* say, "Dull, lifeless work environments cause brain damage."

I said "almost", because it depends on your definition of brain damage. What the research suggests is that in unstimulating,

unenriched, stressful environments, the brain STOPS producing new neurons (more on that later). But it's only been the last few years that scientists have finally realized that the human brain *can* build new neurons. For most of the previous century, it was believed that we were born with all the neurons we'd ever have.

Scientists who believed in and studied the idea of "neurogenesis" were dismissed, criticized, ignored. But Princeton's Elizabeth Gould has picked up the neurogenesis ball and run with it. She is almost single-handedly changing the face of neuroscience and psychology.

From a fascinating article in the new print issue of Seed Magazine (my new favorite):

"Eight years after Gould defied the dogma of her field and proved that the primate brain creates new cells, she has gone on to demonstrate that the structure of the brain is incredibly influenced by one's surroundings."

One of the most interesting (and, in hindsight, "doh!") discoveries was that one of the main reasons researchers kept finding NO evidence of new neuron development in their test primates is because *they kept them in an environment which shut that process down*. In other words, it was the caged-living that stopped the neurogenesis process. By giving her animals a rich, natural enviornment, Gould "flipped the switch" back on, allowing their brains to work normally, and sure enough--the happier, more stimulated animals showed a DRAMATIC increase in neurogenesis as well as dendrite density.

One summary:
"Complex surroundings create a complex brain."

[One interesting and beautiful back story--researcher Fernando Nottebohm had showed earlier that neurogenesis was necessary for bird songs. "To sing their complex melodies, male birds needed new brain cells. In fact, up to 1% of the neurons in the bird's song center were created anew, every day." Of course, his work was dissed as irrelevant. I mean, come on, these are *bird brains*. "Avian neurogenesis was explained away as an exotic adaptation..."]

So, back to cubicles. The key to Gould's demonstration of neurogenesis (where virtually all other primate studies had failed) was the stimulating environment. Cages stopped

neurgenesis, which she describes as "The neurons stop investing in themselves." She links caged environments with stress, and stimulating natural environments as less stressful, so there is a big assumption here that a dull, boring, unstimulating cube life is also stressful (for the brain, anyway--it doesn't mean the work itself is stressful).

But she didn't just throw them in a natural environment... she also made sure they had a lot of opportunities for play. And perhaps very importantly--*frequent rotation and introduction of new toys*. I've always wondered why in every game company I worked for (or anyplace with "creatives"), it was assumed and encouraged that people made elaborate virtual worlds out of their workspace, but in non-game/creative workplaces, not so much. While this is often *allowed* in the cubes of non-game programmers, for the most part it's only the young hipster startups that consider this a primary, essential element of their corporate culture. [Apparently those ping-pong tables and games in those web startups were more than just examples of bubble/VC excess.]

With Gould's work, it would seem, we should not only be *allowing* employees to, say, decorate their cube, but we should be encouraging it at every turn AND take steps to make frequent changes to the area. (And by "changes", I don't mean rotating demotivation posters). For too many places I've worked, a new "official policy poster" or some new HR thing is about all the stimulating change we got. (That, and the increasingly emphatic signs posted in the coffee/kitchen area about "your mother doesn't work here, it's up to US to keep it clean!!!")

It would appear that blowing your own mind on a regular basis is not just a good idea, it's a key part of neurogenesis. One of the conclusions she came to is that "learning heals the brain." And again, we aren't talking emotionally or psychologically, we're talking physical structures. She believes that even those who have *been* in a stressful environment can undo much of the damage by not just removing the stress, but actively introducing enriching and stimulating things.

Experiencing and learning new things is *literally* exercise for the brain!

That's so cool.

The implications of her work are of course much deeper and more significant than just "dull and/or stressful work environments with low stimulation suppress neurogenesis, which means less or no new brain cells." There are all kinds of social implications as well, although she points out that like most scientists, she does not want to see her work "twisted for political purposes".

But it does mean that the work we are all doing to help our users learn and grow and develop (and kick ass ;) is a lot more meaningful than just good customer support. Remember, when WE say "passionate users", we mean the kind of passion that inspires people to spend time learning and getting better at whatever it is they're passionate about. So it would seem that it might not be a huge stretch to say:

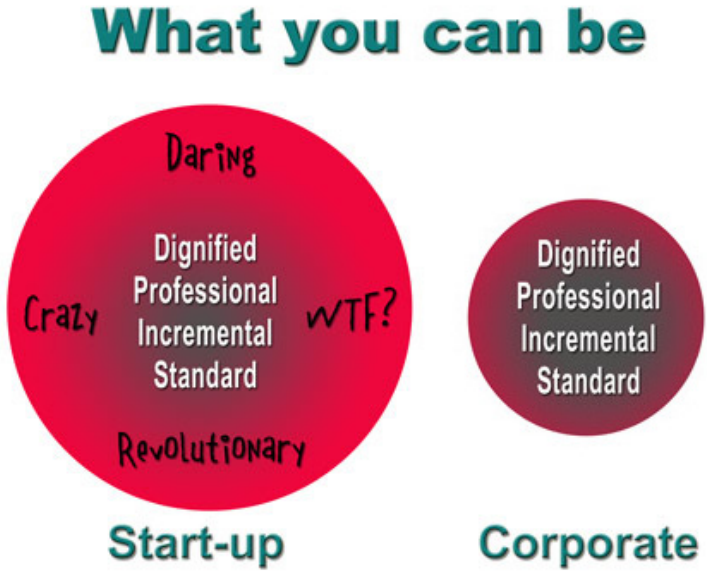
***Passionate* users grow more brain cells!**

Apparently all work and no play makes Jack not just *dull*, but *dumber*. So don't forget to have fun...

http://headrush.typepad.com/creating_passionate_users/2006/02/brain_death_by_.html

Dignity is Deadly, Part Two

By Kathy Sierra on February 28, 2006



In an earlier post, I talked about Paul Graham's talk at Amazon, on what the corporate world can learn from start-ups. The point that stuck with me the most is this:

When you evolve out of start-up mode and start worrying about being professional and dignified, you only lose capabilities. You don't add anything... you only take away. Dignity is deadly.

So I made a list of what I've found in start-ups versus what I've lived in the corporate world. I painted this in the extreme on both sides, of course.

Or did I?

	Start-up	Corporate
Culture	Disruptive	Risk-averse
Focus	Users	Profit and productivity
PowerPoint style	Presentation Zen	Bullet points, pie charts
Innovation	Revolutionary leaps	Incremental
Dress code	You must	"Business Casual"
Workspace	Rich, stimulating, personalized	Professional, efficient
Voice	Conversational language, human to human	Business-speak, corporate entity to consumer
Public image	Read our blog...	Carefully crafted by marketing
Technology	Mac, Linux box, TabletPC	Windows 2000
Driving force	Passion (with profit)	Profit
Users are...	Community, tribes, the people we have to take care of	Demographics, market segments, market opportunities
Burning question	How does this help the user kick ass?	How does this align with our core competencies?
Sensibility	Urban, real	Suburban, fake
Borrowed mantra	"Think Different"	"We're Number One!"
Daily reads	Gaping Void	Wall Street Journal
Magazines	Seed, Mental Floss, Wired, FastCompany, Make	Forbes, Harvard Business Review
Employee reading	"A Whole New Mind", "Flow", "Wisdom of Crowds", "Purple Cow"	"Negotiate to Win", "Lean Six Sigma", "The Effective Executive"
Competition	Irrelevant	Competing for market share on price and/or features
Fun?	Yes (what a lame-ass question)	Don't understand the question...
Business plan	Changes more often than our software specs	Five-year strategic plan
Corporate office	Starbucks, the W hotel lobby, somebody's garage, doesn't matter!	A nice office in a nice part of town, with nice parking
You'll be fired for...	doing crap work, back-stabbing, treating users badly	Putting "WTF???" in an email to the CEO
Mission statement	We love what we're doing. We need to make a profit so we can keep doing it.	Our mission is to deliver expert solutions to customers while maximizing shareholder value.

I'm not saying start-ups couldn't learn a thing or two from Big Business, but keep in mind the name of this blog. We're not doing Best Business Practices For Maximixing Profit. This is about creating passionate users. The good news is the big(ish) companies that "get it" are working hard to incorporate the best of both cultures. There's no REAL reason why a big, established company cannot keep the start-up spirit alive, while still running a business and yes, *maximizing shareholder value*. And as each day goes by, the chances that your users/customers/clients are from the gamer generation goes up. That's the best news of all... because the post-boomers don't give a damn about your "professionalism". They want to know how you can help them kick ass, and why you don't have a blog.

[apologies for the huge graphic there, but I wanted to make it easy for people to grab it]

Links from the table:

Gaping Void

Presentation Zen

http://headrush.typepad.com/creating_passionate_users/2006/02/dignity_is_dead.html

How to be an expert

By Kathy Sierra on March 3, 2006

The only thing standing between you-as-amateur and you-as-expert is ***dedication.*** All that talk about *prodigies*? We could *all* be prodigies (or nearly so) if we just put in the time and focused. At least that's what the brain guys are saying. Best of all--*it's almost never too late.*

Seriously. How many people think they've missed their opportunity to be a musician, or an expert golfer, or even a chess grand master because they didn't start when they were young? Or because they simply *lacked natural talent*? Those people are (mostly) wrong. According to some brain scientists, almost anyone can develop world-class (or at least top expertise) abilities in things for which they aren't physically impaired. Apparently God-given talent, natural "gifts", and genetic predispositions just aren't all they're cracked up to be. Or at least not in the way most of us always imagined. It turns out that

rather than being naturally gifted at music or math or chess or whatever, a superior performer most likely has a gift for concentration, dedication, and a simple desire to keep getting better. In theory, again, *anyone* willing to do what's required to keep getting better WILL get better.

Maybe the "naaturally talented artist" was simply the one who practiced a hell of a lot more. Or rather, a hell of a lot more *deliberately*. Dr. K. Anders Ericsson, professor of psychology at Florida State University, has spent most of his 20+ year career on the study of genuises, prodigies, and superior performers. In the book The New Brain (it was on my coffee table) Richard Restak quotes Ericsson as concluding:

> *"For the superior performer the goal isn't just repeating the same thing again and again but achieving higher levels of control over every aspect of their performance. That's why they don't find practice boring. Each practice session they are working on doing something better than they did the last time."*

So it's not just how *long* they practice, it's *how* they practice. Basically, it comes down to something like this:

Most of us want to practice the things we're already good at, and avoid the things we suck at. We stay average or intermediate amateurs forever.

Yet the research says that if we were willing to put in more hours, and to use those hours to practice the things that *aren't* so fun, we could become good. Great. *Potentially brilliant*. We need, as Restak refers to it, "a rage to master." That dedication to mastery drives the potential expert to focus on the most subtle aspects of performance, and to never be satisfied. There is always more to improve on, and they're willing to work on the less fun stuff. Restak quotes Sam Snead, considered one of the top five golfers of the twentieth century, as saying:

"I know it's a lot more fun to stand on the practice tee and rip your driver than it is to chip and ptch, or practice sand shots with sand flying back in your face, but it all comes back to the question of how much you're willing to pay for success."

There's much more to the brain science around this topic, of course--I'm just doing the highlights. And a lot of the research is new, made possible today by how easy it is for researchers to get

time with an fMRI or PET scan. And I stretched just a little... there *is* some thought that to be, literally, THE best in the world at chess, or the violin, or math, or programming, or golf, etc. you might indeed need that genetic special something. But... that's to be *THE* best. The research does suggest that whatever that special sauce is, it accounts for only that last little 1% that pushes someone into the world champion status. The rest of us--even without the special sauce--could still become world (or at least national) class experts, if we do the time, and do it the right way.

Where this ties into passionate users is with the suck threshold and kick-ass (aka "passion") threshold. Your users will typically fall into one of the three categories in the graphic: expert, amateur, or drop-out. The drop-outs decide that during that "I suck at this" phase, it isn't worth continuing. They give up. Is that something you can work on? Do you *know* what your attrition rate is?

But the most troubling--and where we have the most leverage--is with the amateur who *is satisfied with where they are*. These are the folks who you overhear saying, "Yes, I know there's a better way to do this thing, but I already know how to do it *this* [less efficient, less powerful] way and it's easy for me to just keep doing it like that." In other words, they made it past the suck threshold, but now they don't want to push for new skills and capabilities. *They don't want to suck again*. But that means they'll never get past the kick-ass threshold where there's a much greater chance they'll become passionate about it. The further up that capability curve they are, the higher-res the user experience is!

Can we help make it easier for them to continue on the path to becoming expert? Remember, *being better is better*. Whatever you're better at becomes more fun, more satisfying, a richer experience, and it leads to more flow. This is what we're trying to do for our users.

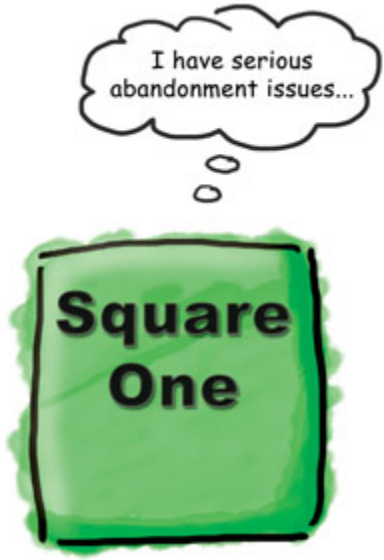
Oh yes, about that never too late thing... most of us can kiss that Olympic ice skating medal good-bye. And at 5' 4", my basketball career is probably hopeless. But think about this... actress Geena Davis nearly qualified for the US Olympic archery team *in a sport she took up at the age of 40, less than three years before the Olympic tryouts*.

And if the neuroscientists are right, you can create new brain cells--by learning (and not being stuck in a dull cubicle)--at virtually *any* age. Think about it... if you're 30 today, if you take up the guitar tomorrow, you'll have been playing for TWENTY years by the time you're 50. You'll be kicking some serious guitar butt. And if you're 50 today, there's no reason you can't be kicking guitar butt at 70. What are you waiting for?

http://headrush.typepad.com/creating_passionate_users/2006/03/how_to_be_an_ex.html

Don't forget square one...

By Kathy Sierra on March 4, 2006



"When you're done with square one, pick it up and take it with you."

Horse trainer Linda Parelli says that, and her take on amateurs-vs.-experts is that the amateurs forget the fundamentals. Her husband Pat, founder of Parelli Natural Horsemanship (the most successful example of passionate users I've ever seen), says the same thing. (In the Parelli training system, the fundamentals are called the Seven Games.)

My trainer Darren went to an expert-level workshop with Pat and a dozen other top horsemen. Pat asked each in turn what they were hoping to improve on. Each one gave an elaborate description of some very advanced, elite thing they were struggling with. Pat listened intently and when they had all finished, he shrugged and said to all of them, "Get your games better."

The problem the Parelli's see in those trying to transition from skilled amateur to expert virtually always comes down to something from the fundamentals that they either never quite mastered, or that they forgot over time. So, perhaps that's one more thing the superior performers do better than the rest of us-

-they keep practicing the fundamentals. This fits with the notion that experts practice things that aren't necessarily *fun*, which can include both the things they still don't do well, AND the non-exciting basics.

Bert Bates (my co-author) is a blackbelt level go player, one of the best amateur players in the state. But when a visiting expert--four belt levels above Bert--showed up at the local go tournament, Bert was surprised to see the guy reading a book on fundamental go problems that Bert had read much earlier in his learning. The expert said, "I must have read this at least a hundred times. My goal each time is to see how much more quickly I can solve all the problems in the book than I did the last time."

Some of the best athletes never forget the fundamentals--whether it's Tiger Woods practicing the basics, or a pro basketball player working on free throws. A good musician might still practice arpeggios. A programmer might... I don't know, actually. What would be the fundamentals that a good programmer might forget? I'll have to think about that one.

But the Parelli's have another piece of advice that I think is equally important--that you shouldn't get *stuck* trying to perfect the fundamentals before moving on. There's a girl at my barn who has been taking dressage lessons on and off for the last ten years. Both her and her horse are bored out of their minds because the trainer won't let them progress to anything interesting until they are virtually perfect on the basics. The Parelli approach is, "Keep moving forward, because you'll gain new tools that you can use to go back and perfect the fundamentals." But this is where the "don't forget square one" message comes in--the problem is with the people who do NOT use their new "superpowers" to fix what might be lacking in the basics.

I've been struggling with the same thing in skiing. I've been an intermediate/advanced skier for frickin' ever. I have a ton of fun, I can waltz down the steepest blues and the occasional light black. But... there has always been a flaw in my basic form/technique that has slammed me into a brick wall. The only way I can ever go further is by going back to correct and redo a part of the fundamentals... the way I weight shift. So all of last year, I pretty much sucked. I had to go through what Keith Ray mentioned in the comments: conscious incompetence.

And while we're here... Cleve brought up a great point in the comments to my previous how to be an expert post: how does "work on the things that suck" fit in with the whole "play to your strengths" thing? There are some great comments discussing it, but here's my take--I think the "work on the things you suck at" is *within* the domain you've already decided is something you WANT to get better at, as opposed to a weakness that just doesn't seem like *you*. In other words, I suck at about a million things which I am weak in and have no interest in pursuing. But... I love skiing, so if I want to get better, I have to be willing to work on my weak spots so that I can have more of what I want (fun, flow, etc.).

So choosing the thing you would *like* to be expert in is probably not going to be in an area where your nature/personality/interests are "weak". But within the thing you choose to pursue, you have to work on the less fun things, which include both the things you're not as good at, AND the basic fundamentals.

Are you helping your users take square one with them? Are there areas where your users may be missing some fundamentals they'll need? I guess we all have to figure out how to make the fundamentals less boring--apparently that's what the experts know how to do. And if you see me struggling and swearing down an intermediate slope at Copper Mountain, yell out "lift your inside ski!" as you fly by.

http://headrush.typepad.com/creating_passionate_users/2006/03/dont_forget_squ.html

User Enchantment...

By Kathy Sierra on March 15, 2006

The best user experiences are *enchanting*. They help the user enter an alternate reality, whether it's the world of making music, writing, sharing photos, coding, or managing a project. Even a spreadsheet has the potential to be as engaging as a game.

Until the interface comes crashing into your virtual world, throwing you back to the *real* one. That intense feeling of being engaged--the flow state--is interrupted. ***The spell is broken.***

Work or play comes to a grinding halt while you fiddle with controls, interact with the man behind the curtain, or--worst of all--refer to a manual. Every moment spent d***ing around with the interface, equipment, or bureaucracy is a moment spent *outside* the thing that interface, equipment, or system is meant to support.

One of the themes I heard over and over at ETech and SXSW (Jason Fried, Craig Newmark, and others) was the developer mantra of "get out of the way." In other words, build the thing so that it stays the hell out of the way and lets the user get on with what they *really* want to do.

Getting out of the way means not breaking the spell.

If your product is like a movie in a theater, ask yourself what parts of it force the user to be aware of the *interface* rather than *the story*?

What breaks the spell?

UI means User Interface, and UE means User Experience, but there's nothing in the UE label that suggests the experience is *engaging*. We tell our co-authors to always be thinking of ways

to charm and enchant the reader/learner, so we're recasting the "E" in UE to "Enchantment."

Sometimes...we even use the word ***seduce.***

Is this unethical and manipulative? Not if the user is a co-conspirator. When I go to the theater, I *want* to be swept up in the world of the film. Anything that drags me back to the fact that I'm in a physical theater is an unwelcome interruption.

We go into a theater ready to suspend disbelief, but as filmmakers know--if at any point the movie "breaks character", the spell is broken. Disbelief now puts the rest of the story at risk.

Our readers pick up one of our books because they want to learn Java, Design Patterns, HTML, AJAX, etc. We know it, they know it, and we're not trying to trick them into something they haven't acknowledged they want. They are complicit in this "hero's journey." So no, we see enchantment and charm as a way to make the experience as engaging and flow-inducing as possible.

Whatever your product or service, your users/members/guests are interacting with what you provide for a *reason*. And remember, that reason may be VERY different from, say, "using your tool." I don't use a camera to "use a camera". *I use it to take photos*. I want your tool (camera) to stay out of my way so that I can focus on the flow of composing and capturing shots, not working out how the hell to change the shutter speed. I don't use Basecamp to "use a project management app." *I use it to manage my book review*. I want the camera and the Basecamp app to fade into the background so I can focus on what I care about most.

Most people claim that the times they spend truly in flow are among the happiest times of their life. I ended my SXSW session with, "As developers, we are so fortunate to have the opportunity to bring more joy into people's lives."

Go kick some user ~~experience~~ enchantment butt :)

(And god help us if Microsoft takes over Hollywood)

[Bonus link: fun video of a real-world interaction with the paperclip. NOT SAFE FOR WORK (uses the "F" word)]

http://headrush.typepad.com/creating_passionate_users/2006/03/user_enchantmen.html

Ultra-fast release cycles and the new plane

By Kathy Sierra on March 16, 2006

I just came back from dinner with my daughter Skyler (that's her in the picture). She's an *extremely* passionate myspace user. In her words, "If you're not on myspace, you don't exist." So I asked what made myspace so compelling... why didn't she fall in love with LiveJournal? Her answer is a lesson for software developers (especially Web 2.0-ers), and was a theme of SXSW:

"myspace keeps doing what everybody really wants, and it happens instantly."

She said they respond to feedback, "As soon as you *think* of something, it's in there."

She said, "It's always evolving. It changes constantly. There's always something new."

I asked if these changes were disruptive or made it harder to use when nothing stays the same, and she gave me that teenage-attitude-eye-rolling-what-a-lame-question look.

Then she said the weirdest thing of all: "myspace is like a whole new plane of existence."

She wasn't kidding.

And I thought of two things I heard at ETech and SXSW that were really thought-provoking...

Danah Boyd's astonishing talk at ETech and the talk by the guys from SkinnyCorp (founders of Threadless).

On the culture of myspace:

If you have kids of any age, or customers under the age of 40, or support an online community, I urge you to read Danah's transcript (the link above)--this woman knows as much about the culture of myspace as anyone, and she has a ton of insight and knowledge about online communities. At a deep level. (I consider her blog in my top 20 for sure.)

On lightning release cycles:

Skyler's comment about how myspace keeps changing and growing organically, *almost every day*, is a passionate user's view of what the developer's call quick release cycles. Where software developers are typically on release cycles of 6 months to a year, the Threadless guys said that even **two weeks** was a little long. In fact, virtually all of the web 2.0-ish folks at the conference mentioned these quick release cycles as crucial.

There are a ton of issues, obviously, like what happens when a new release breaks something that previously worked. The Threadless guys said that happens, but only rarely, and they just do a rollback. Skyler said she's seen things break on myspace, but nobody seems to care much since they know it'll probably be fixed *tomorrow*.

I wonder if quick release cycles become almost *addictive* to the end users... we're so used to thinking of how upset they'll be when we change things, but clearly this is a different (and frickin' HUGE) group of users who not only don't *mind* the change--they THRIVE on it. Perhaps those quick releases are a little like quick *fixes*. Code Crack.

The Threadless guys (and the 37signals guys) have said pretty much the same thing as Skyler did -- that some people may complain when you change something, and occasionally you might even lose someone from the community, but that it's very rare for someone to *stay* upset. One day the users are threatening to revolt if XYZ isn't put back the way it was, and the next day they've all but forgotten.

So, quick release cycles and a new plane of existence. I have to think about this some more... I'd love to hear your thoughts about any of this!

[UPDATE: there's a lively discussion about how this relates to game development, on Raph Koster's blog (an excellent regular read for those interested in game design/dev)]

[FYI -- I'm still not current with my emails. I'm working on it, and I'm home now for the next six weeks, so I WILL catch up.]

http://headrush.typepad.com/creating_passionate_users/2006/03/ultrafast_relea.html

Web 2.0 is like Group Therapy

By Kathy Sierra on March 19, 2006

When did Web 2.0 become synonomous with **Sharing**? And I mean "Sharing" with a capital "S". Sharing knowledge, lessons learned, product reviews, tips and tricks, links/bookmarks, and even photos makes sense to me. The ability to harness collective intelligence (the whole wisdom of crowds thing) is something we all benefit from. But that's "sharing" with a small "s".

I believe the "small s" version is what Tim O'Reilly is referring to in his Web 2.0 Compact Definition . He doesn't even *use* the word "sharing". He uses "architecture of participation." And participation is not the same as Sharing. But the definitions of Web 2.0 have morphed and made Sharing=Goodness a central theme. And sometimes, Sharing means Too Much Information.

In America, we're all rightfully outraged by invasions of privacy, while our appetite for the private lives of others grows unchecked:

Reality TV

Celebrity gossip

(and now even blog and tech celebrities!)

Personal secrets

(granted--I LOVE this site, but at least here it's more about the art... and anonymous)

Dooce, the top 100 blog whose official description is, "I talk a lot about poop, boobs, my dog, and my daughter." Outside of the dog, and an occasional kid picture, the rest makes me cringe. The odd thing is, Dooce author Heather Armstrong is such an amazing writer that she could make renewing her car registration worth reading. I don't consider the whole bowel-movement-thing to be an essential part of what makes her so compelling. (hope I'm not wrong on that one...)

More and more, the Web 2.0 and Blog world feels like a highly-scaleable, web-enabled way to peek into more medicine cabinets. And it's even sucking the slightly elicit fun out of *that* now that we're all encouraged to *Share*. Where's the mystery? Where's the excitement that comes from *not knowing everything*? Is the (metaphorical) allure of the strip-tease gone forever?

I don't want to know what's in my present before I open it. I look away during movie trailers. I love the air-tight secrecy around Apple's product announcements.

Please, please Web 2.0 folks -- don't let "harnessing collective intelligence" become group therapy. Feeding our lust for personal, public revelations (it's just a matter of time before nearly everyone has been trashed online by an angry ex) isn't helping *raise* our collective intelligence.

Just look at all the scary similarities:

Group therapy	Web 2.0
It's all about "Sharing"	It's all about "Sharing"
Personal validation	XHTML validation
Improve social skills	Improve social networks
Rounded pillows	Rounded corners
Equality and inclusion highly *valued*	Equality and inclusion highly *touted*
Nothing is too personal for a session	Nothing is too personal for a blog
Therapist as facilitator	Moderator as facilitator
Big fears	Big fonts
Makes you feel better as long as you keep coming back	Makes you feel better as long as you keep coming back
"My father ignores me"	"The A-list ignores me"
Paranoid	Delusional
Special lexicon: *enabling, co-dependent, stuffing*	Special lexicon: aggregate, folksonomy, AJAX
Soft, pleasant colors in the office	Soft, pleasant colors in the interface
There's always an a**hole in your group	There's always an a**hole in your comments
Big fights over use of the word "a**hole"	Big fights over use of the word "a**hole"
Self-esteem issues	Technorati rank issues
In denial about the quality of your relationships	In denial about the quality of your content

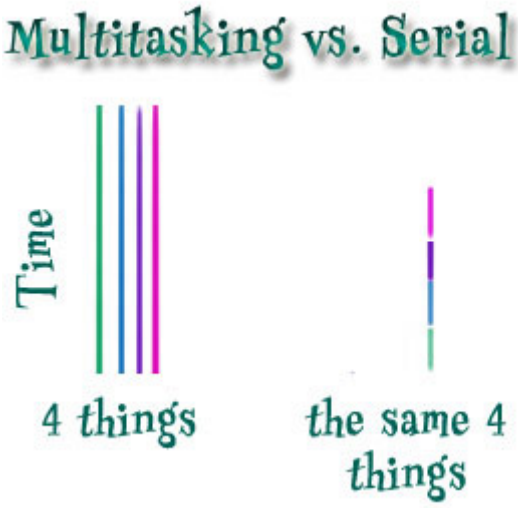
Bonus question: be honest, how many of you clicked on the gossip link? How long did you stay there? ;)

[Disclaimer: this post was meant, more or less, as a joke]

http://headrush.typepad.com/creating_passionate_users/2006/03/web_20_is_like_.html

Multitasking makes us stupid?

By Kathy Sierra on March 22, 2006



I'm typing this while talking on my cell phone to one person and IMing another. Am I fooling myself that I can actually *do* these three things without a loss of quality? No... because I *know* I can't. I understand that what most of us call multitasking comes with a steep price tag.

But where I once believed that the myth of multitasking was about *time* (that doing four things simultaneously takes much longer than to do those same four things in sequence), scientists now know it's also about *quality*. And it gets worse... it's not just that the quality of those four things in parallel will suffer, it's that your ability to think and learn may suffer. Some researchers believe that all this constant, warpspeed, always-on multitasking is causing young people, especially, to become less able to follow any topic *deeply*. (more on that in another post)

Perhaps the biggest problem of all, though, is that the majority of people doing the most *media* multitasking have a big-ass blind spot on just how much they suck at it.

We *believe* we can e-mail and talk on the phone at the same time, with little or no degradation of either communication.

We *believe* we can do homework while watching a movie.

We *believe* we can surf the web while talking to our kids/spouse/lover/co-worker.

But we *can't*! (Not without a hit on every level--time, quality, and the ability to think deeply)

From the current cover story in Time magazine:

"Decades of research (not to mention common sense) indicate that the quality of one's output and depth of thought deteriorate as one attends to ever more tasks."

And according to Jordan Grafman, chief of the cognitive neuroscience section at the National Institute of Neurological Disorders and Stroke:

"Kids that are instant messaging while doing homework, playing games online and watching TV, I predict, aren't going to do well in the long run."

And from this study on young people and media use:

"Nearly one-third (30%) of young people say they either talk on the phone, instant message, watch TV, listen to music, or surf the Web for fun "most of the time" they're doing homework."

The news is not all bad, of course -- from the Time article:

"The breadth of their knowledge and their ability to find answers has just burgeoned...but my impression is that their ability to write clear, focused and extended narratives has eroded somewhat."

And yes, we are all able to do some form of multitasking--some can even win an Olympic gold medal listening to an iPod. But the brain science helps explain this--we can do two things at once as long as one of them is something we've practiced so much that it doesn't require any sort of cognitive *planning* (there's a lot about this in the Time article).

The main problem today is that cognitive overload--provoked by so much media to attend to--is happening at a pace our poor little hunter/gatherer brains never evolved to deal with, and there's only so much that neural rewiring can do. And of course this is all very recent. When I was in high school, there were no iPods. There were no cell phones. No web, email, or IMing. No blackberrys. No PSP. (How did we ever *survive*? asks my daughter.) Multitasking for me in high school meant a ripping

game of 1-bit Pong while simultaneously flirting with the geek from my history class.

Whenever I talk about the big myth of multitasking, people *always* come up to tell me how they themselves just "have the kind of brain that can do this." Riiiiiight. They don't. I don't. You don't. And maybe you'd realize it if you turn off your cell phone, disable IM, mute the little "ding" alarm that says you've got email, and just *sit there* for a few moments.

The big problem for most young people, it seems, is that they don't know *how* to "just sit there." They get the shakes after just a few minutes without media stimulation. But that is also a whole separate topic I'll get to very soon...

One of the most interesting things discussed in the Time article is that neuroscientists have established the specific area of the brain responsible for context switching. And unfortunately for some of us, it appears that this part of the brain performs less well as our brain ages. In a nutshell, the older we get, the less quickly and effectively we can multitask. But... most parents of teenagers already know that we have no frickin' idea how our kids manage to do what they do simultaneously. The key issue, though, is that while we now know they're better at it than we (the parents) are, they aren't half as good at it as they think they are.

And chances are, *you* aren't as good at it as you think you are. ;)

http://headrush.typepad.com/creating_passionate_users/2006/03/multitasking_ma.html

Manager 2.0

By Kathy Sierra on March 27, 2006

You can't very well have a Web 2.0 company run by version 1.0 managers, right? Yes, I'm making fun of the 2.0ness of it all, but if we're throwing version numbers around with impunity, might as well take it to the absurd.

One dramatic difference between mature tech companies and the Web 2.0 startups is the way employees are managed. Or rather, the fact that they are *not* "managed." Most Web 1.0 companies (like, say, my former employer Sun... they put the dot in dotcom, remember?) are not only too *big*, but their management practices are just too old school (and not in a retro hip way) to foster a company culture that matches the culture of the new community/user-centric Web 2.0.

[Note: I'm talking mostly about the non-VC, non built-to-flip, non whee-its-another-bubble! startups]

My favorite example of the difference between Web 1.0 and Web 2.0 management is that Web 2.0 "geek community values" infuse many of these startups at the cellular level. Never has the notion of "community" meant so much to business, so it's no surprise a Web 2.0 manager would think of employees as a community.

And *that* may change everything.

Yes, there has always been a startup vs. corporate culture comparison, but this is different. "Community" did not play such a central role for the bubble/Web 1.0 startups.

So, all you Web 2.0 startups -- please keep the best of the Web 2.0 spirt alive by NOT adopting the awful practices of management 1.0. There's nothing new here, of course--Tom Peters has been talking about this for frickin' ever. (And plenty of others before and after him.) What *is* new is that while it's always been a *good idea* to manage this way, this time it's virtually a *given*.

Yes, this is ridiculously oversimplified, does not work out of context, and you can't take things in the 2.0 column ala carte. I still have absolutely NO idea what Web 2.0 even means, but whatever it is, people are in the equation (both users and

employees) in a new and more meaningful way. As my friend Nat Torkington says, "It's no longer aspergers and emacs."

Manager 1.0	Manager 2.0
Policies dictated by management and imposed on employees	All employees are asked to help with policy decisions and solutions
Pay for performance (merit-based) and tied to promotions	Pay is generous and fair, profit-sharing is the norm
Overall secrecy--less is revealed to employees at each level down the hierarchy	Employees are given as much information about the company as possible, including financial
"Healthy competition" among teams and individuals. Only the "best-performing" get bonuses and promotions	Employees and teams challenge themselves. If one person or team succeeds, everyone wins
Formal job description created by management. Changes to the description must be officially granted	Informal job role created by employee, tailored to their strengths and interests, and changes all the time
Emphasis on "teamwork" (but competitive internal practices often work against it)	Emphasis on "community"
Employees tightly controlled--responsibility without authority	Employees have autonomy, responsibility, and authority
External rewards and incentives as motivation	Intrinsic motivation to do really good work
Customer is king (but internal policies and process often hinder this)	Users are king, but not at the expense of employees
Performance appraisals conducted by supervisors	Continuous peer review... official "appraisals" irrelevant due to constant communication
Hierarchical structure	A Hollywood model
Hiring based on past performance and current skill set	Hiring based on curiosity, ability to learn, and passion
Deadlines imposed on those who do the work by those who don't	Deadlines agreed on and set by those doing the work
Employees often forced to learn new things and "change"	Employees create opportunities to learn and be challenged

[FYI--I am having a fantastic time going through your introductions! Thank you so much for letting me learn a little bit about who you are. It helps a lot more than you know.]

http://headrush.typepad.com/creating_passionate_users/2006/03/manager_20.html

Are you a passionate tech user?

By Kathy Sierra on March 28, 2006

Are you a passionate user of a tech product? I'd really like to know about it for the book!

In the next couple of weeks I'm going to make a few requests for people who have a story to tell about something they're passionate about, and also to ask if you can point me to products/services you think fit the criteria I'm looking for. It's for the "Creating Passionate Users" book (from O'Reilly) that I'm now so behind on I'm surprised the publisher isn't holding my horse hostage. [And a big thanks to my review team that's come back online]

Remember, the definition of "passionate" we're using means that you are always learning, growing, improving in some way related to the tool/product/service OR (more likely) something that the tool lets you do. (For example, you're passionate about your Canon camera because it lets you take great photographs, and you're always trying to get better with your photos--tweaking the manual settings, etc.)

What I'm looking for today is:

1) Anyone who is truly passionate about their **Wacom** tablet.

Whether you're passionate about the device itself (programming the buttons just so, learning how to push the limits, maybe lusting after (or owning) the Cinteq, etc.) OR (again, more likely) passionate about the kinds of work you're able to do with your tablet that you know you couldn't do otherwise.

2) Anyone who is passionate about **any tech product--hardware or software--that is not a game**. I'm especially interested in things more associated with productivity or any kind of business work. For example, an Excel user who is constantly pushing to use the tool to do more interesting things with modeling, etc.

3) Anyone who is passionate about a particular **non-profit cause or service**. Again, I'm looking for things where you are **actively involved** and always trying to learn and/or do more related to it. Simply being a very strong believer/supporter is not enough... it must be something for which you are

continuously learning more and potentially getting more involved.

(p.s. this could include a church, although I'm NOT looking for stories about passion for a religion, but rather a passion for a specific church/organization that you are involved with.)

4) Anyone with a pointer or story to tell about a company (anything other than a game or sport) that provides a great deal of **learning support for its users** at all levels, whether its through good tutorials, active online support forums, etc.

I'd love to hear from you, and it is entirely up to you if you want your real name in the book--anonymous is just fine, although if you've got a story of your own to tell, I will ask if I can quote you.

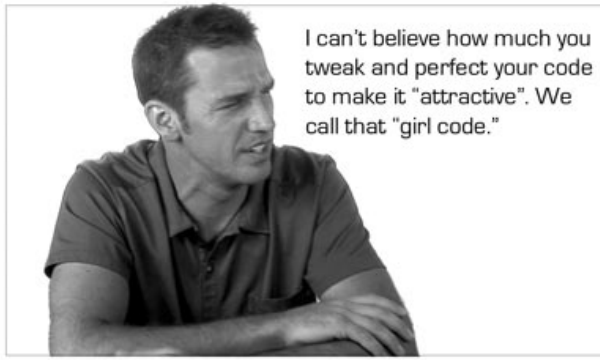
Thanks so much guys. Oh yeah, I know that a lot of YOU have products and services and causes that people are passionate about, so don't be shy about self-promoting that to me! If you think it meets the criteria I've listed, please let me know (although I'll ask for a reference to an actual passionate user I can talk to).

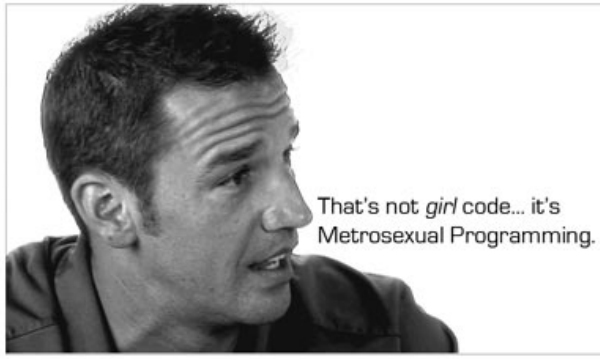
I realize this post does NOTHING to help you kick ass, but I appreciate any help you can give me. You can write to me directly -- headrush[at]wickedlysmart[dot]com, or just leave a comment (comment is probably better since it might trigger an idea from someone else).

http://headrush.typepad.com/creating_passionate_users/2006/03/are_you_a_passi.html

Code like a girl

By Kathy Sierra on March 29, 2006





Do engineers and programmers care about concepts like beauty and elegance? Should they? Designers have always known that looks *matter*--that the *outside* (interface) matters. But deep in

the heart of those building the *inside*--the technology most users never see--lies the sensibility of an artist. In a kind of "Design Eye for the Code Guy" way.

While I'm stereotyping with abandon, I might as well be honest. I've been going to tech conferences for the last 15 years, and I swear the ratio of pocket protectors to Urban Outfitter clothes has shifted dramatically. So maybe it's not accurate to say geeks today are better *looking*--but they're certainly better *dressed*. With hipper haircuts.

Does this *mean* anything? Maybe.

What prompted this post--and it's whimsical title--is a post by Jamis Buck titled Beautiful code, test first, which includes the following:

> *"He was telling me how he feels like he has to sit and tweak his code over and over until it not only acts right, but looks right. It cannot be merely functional, it must be beautiful, as well."*

But the best part was a comment by "Morten" that included the line:

> *"As for spending too much time on making the code look right down to the last indentation - my code has been called “girl code” for the same reason..."*

And there you have it. I think "girl code" is quite a compliment. Because caring about things like *beauty* makes us better programmers and engineers. We make better things. Things that aren't just functional, but easy to read, elegantly maintainable, easier--and more joyful--to use, and sometimes flat-out sexy. A passion for aesthetics can mean the difference between code that others *enjoy* working on vs. code that's stressful to look at. And whether we like it or not, most of the world associates an appreciation for beauty more with women than men (especially *geek* men). Women may have a genetic advantage here.

From one of my favorite books on aesthetics and technology, David Gelernter's Machine Beauty:

> *"This book explains how beauty drives the computer revolution: how lust for beauty and elegance underpinned the most important discoveries in*

computational history and continues to push research onward today....The best computer scientists are, like [Henri] Vaillancourt, technologists who crave beauty.

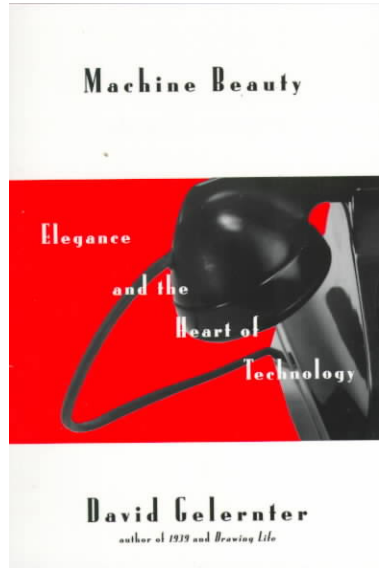
There is the ever-present danger when you discuss beauty in science, mathematics, and technology that readers will assume the word is being used metaphorically... And could a mathematical proof, scientific theory, or piece of software be "beautiful" in the real, literal way that a painting or symphony or rose can be beautiful?

Yes."

And from the back cover:

"Both hardware and software should afford us the greatest opportunity to achieve deep beauty, the kind of beauty that happens when many types of loveliness reinforce one another, when design expresses an underlying technology, a machine logic...

These principles, beautiful in themselves, will set the stage for the next technological revolution, in which the pursuit of elegance will lead to extraordinary innovations."



Yes, calling beautiful code "girl code" is both silly and some might believe sexist. But that doesn't mean there isn't some truth to it. As a female technologist in a heavily male-skewed industry, don't compliment my *hair*, but if you tell me my *code* is pretty, I might just give you some tips.

And if it makes you feel better, I'll refer to YOUR gorgeous code as *metrosexual*. *But we'll both know the truth.*

[full disclosure: though I'm 100% female, I have personally authored some of the worst-looking code in north america.]

[UPDATE: someone has Code like a girl shirts on CafePress.]

Bonus beauty links:

Don Norman's Emotional Design

Virginia Postrel's "the substance of style"

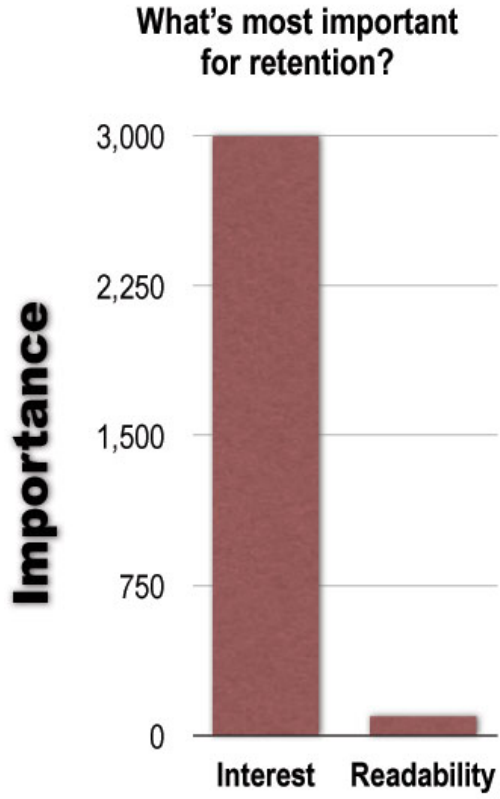
CSS Zen Garden: The Beauty in CSS Design

Flickr "Beauty is Simple" group

http://headrush.typepad.com/creating_passionate_users/2006/03/code_like_a_gir.html

Two more reasons why so many tech docs suck

By Kathy Sierra on March 31, 2006



A great deal of non-reference technical docs suck for a wide range of reasons, but the two most critical (yet fixable) are:

1) We assume the reader is inherently interested.

2) We drastically underestimate the role interestingness plays (especially in retention and recall).

When I say "non-reference", that's because the rules are completely different. Assuming reference materials are accurate, well-organized, and clear--whether the reader is "interested" matters very little. *Reference docs don't need to be memorable.* They aren't expected to communicate big, tough concepts. There's nothing to "get" when you're looking up a recipe, for example.

But when we're talking about learning--especially recall and retention--the reader's motivation means almost *everything*. And readers aren't even REMOTELY as interested as we might think. Intuitively, we often assume that if the reader has taken the time to find--possibly even *pay for*--the book, manual, tutorial, whatever, that they're obviously motivated. After all, they wouldn't be bothering if they weren't trying to learn something.

Sounds reasonsable. But it's almost...always...*wrong*.

Think about it. Almost *nobody* reading a tech document does it for intrinsically-motivated intellectual curiosity. They're reading it because they need to DO something! And anything that does not appear 100% relevant to that goal is just *in the way*. I said "appear", because this is the biggest issue of all. It's not how relevant a particular topic is that matters--it's all about whether you've proved that to the reader, up front. And you have to keep on proving it over and over and over.

It's not enough to simply *say* it's important. That won't keep their brain engaged, even if they trust you. You must *show* it. And you must show it *before* you expect them to pay attention and learn. If you want until after you've explained something to give examples of why it matters or how they can use it, that's too late.

One of the best techniques for creating and keeping interest is to make non-reference docs use-case driven. That way, each topic is framed by a context that matches something the user really wants to do. That way, each little sub-topic shows up just-in-time, instead of appearing to be there just-in-case.

The chart at the top of this post is a dramatic (albeit only partly related) study by Richard Anderson, Larry Shirley, Paul Wilson, and Linda Fielding. In the book Aptitude, Learning, and Instruction: Conative and Affective Process Analyses, they describe that they found--in some circumstances-- interest can be 30 times more important than readability in whether students *remember* what they read. Granted, students are often *forced* to read things they were never motivated by, but there's still something sobering about the results.

Consider all the people reading your docs, and ask yourself if they're reading them purely for intellectual joy, or because they're just trying to do something. Chances are, they're just

trying to DO something, and it's that THING -- and not your technology (and especially not your docs) -- that they're interested in. Bottom line: if we want them to remember, we must make it memorable. And the best way to make it memorable is to make sure they're interested every step of the way. And it's usually our job, not the readers, to build that interest!

The next step on that path is to help *create* an interest in things they didn't know they *could* do, but that's another topic... :)

http://headrush.typepad.com/creating_passionate_users/2006/03/two_more_reason.html

What can we learn from game developers?

By Kathy Sierra on April 3, 2006

Here's my first attempt at a couple of short audio remixes pulled from recordings of my public talks. Both are a partial answer to the question: ***what do game developers know that we can use on non-games to help create passionate users?***

What games do

games.m4a (5.8 mbs, 4 minutes)

Note: this particular talk was focused on technology products (including software and tech documentation), but the same principles apply to *any* kind of product, service, cause, etc...with a little imagination.

The difference between men and women

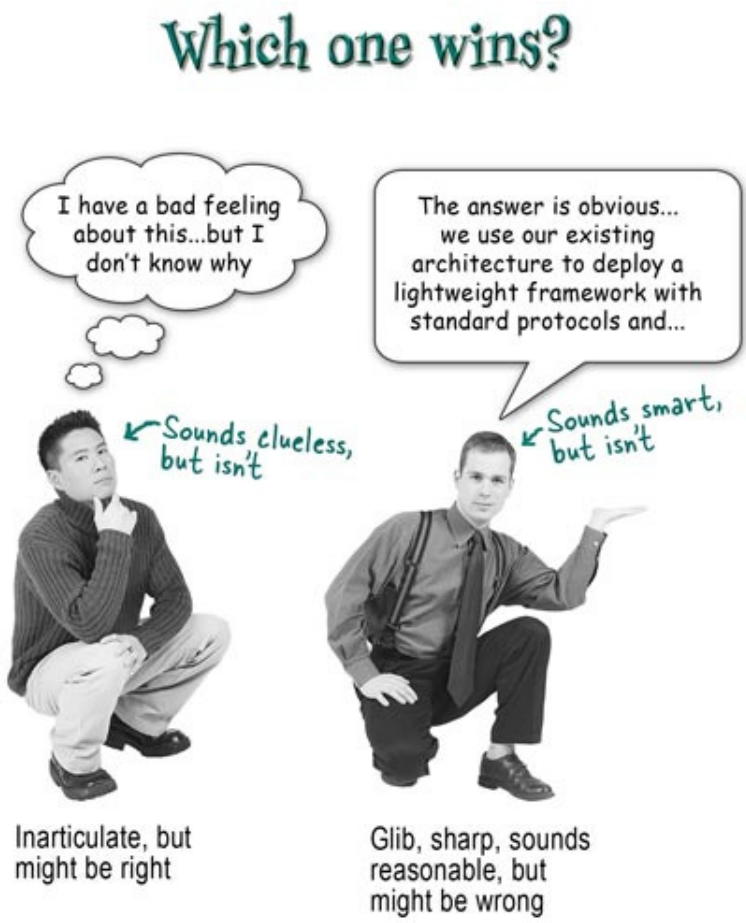
Differences.m4a (1.9 mbs, 2 minutes)

I'm still experimenting with the combinatorial explosion of possible audio settings, so these files might be way too big. I do plan on putting up more pieces from my talks, but don't worry--I'm not turning this into an audio-only podcast! I'll resume regular text tomorrow ;)

http://headrush.typepad.com/creating_passionate_users/2006/04/what_can_we_lea.html

When only the glib win, we all lose

By Kathy Sierra on April 5, 2006



In way too many meetings, the fastest talkers win. And by "fastest talkers", I mean those who are the first to articulate an idea, challenge, issue, whatever. Too many of us assume that if it *sounds* smart, it probably *is*, especially when we aren't given the chance to think about it. The problem is, the guy with the "gut feeling"--the one who senses that something's not right, but has no idea how to explain it, let alone articulate it *on the spot*--might be *right*. Too bad, though, because the glib usually rule.

Let's face it--the clever, witty, glib talkers can make the non-clever, non-witty, and non-glib sound like slow dolts. Slow-to-articulate is mistaken for slow-in-the-head. And as the world

speeds up and decisions have to be made *right frickin' NOW*, it just gets worse.

So there's the heart of the problem--if you're not able to explain your thoughts, ideas, and concerns quickly and articulately, you are often at a disadvantage. I've been there. I *am* there. I'm capable of thinking (some would debate that), willing to do the research, and reasonably articulate. *But I need time!* I have never been one of those think-on-your-feet types. With the exception of those few things in which I have a lot of expertise and experience, I pretty much suck at having to explain, defend, or promote something in real-time.

[Note: Making this a glib vs. thoughtful issue is not what I'm saying; lots of people *can* be thoughtful, right, and quick to articulate. Just because someone can think and speak fast on their feet doesn't necessarily mean they're automatically *wrong*. The problem is that too often they're assumed to be automatically *right*.]

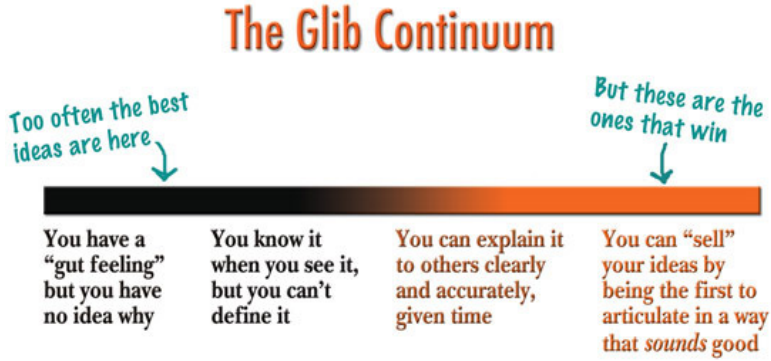
Perhaps we can attack this on two fronts:

1) We all need to fight the culture of the quick, by recognizing that giving people a little time to *think* will do more good than harm.

2) If you're not good at the glib game, there are some things you can do to improve, starting with the most important:
Memorize this: *"I have some concerns, but I need a little time before I can really articulate them."*

If you're a manager, or anyone who leads meetings and discussions, please PLEASE have respect for that phrase. It's unlikely (but possible, sure) that someone will abuse this, since "buying time" in the context of a work decision doesn't usually buy us anything other than the chance to think more deeply.

I do have a few other suggestions, and I hope to hear more from you...



Listen--and respect--your own "gut feelings"

(known as bad smells in the programming world). Read Blink for a greater appreciation of the research behind it. This doesn't mean that your instincts are always *right*, but they should NEVER be dismissed without thought.

Work to move up the glib continuum

Dig for the source of your feelings about the rightness or wrongness of an idea. If you're in the "I know it when I see it, but I can't explain why or how" stage, you need to do some serious analysis. Here are a few tips:

1) Compare the thing-that-feels-wrong to something that you know is right. Look for the deltas. Sometimes we miss the subtle but crucial things because we're looking in the wrong place. It might help to bring in an outsider who doesn't come with pre-existing biases. (Like the drawing on the right side of the brain thing, where you can draw a thing more accurately if you analyze it as lines and shapes and shades and try to ignore/forget what you *know* the thing is)

2) Use rubberducking to force yourself to explain something. Even if you have no idea what you're going to say, just start talking! (to a rubber duck or a helpful friend or co-worker) The act of speaking can engage other parts of your brain and help you shape an explanation. The tricky part is, we sometimes try so hard to find a reasonable, logical explanation that we just make s*** up without even realizing it. So explain it, then do the next thing:

3) Find someone to tear holes in your explanation. This is when you *need* a devil's advocate. Not necessarily to prove you're wrong, but to help you figure out why you're *not*. And of course, most of us need to practice defending our ideas.

4) If your communication skills are weak, work on them. You should be able to talk as fluently and naturally as the guys in marketing--only *authentically* and without the semantically-empty buzzwords and jargon. I hesitate to suggest this, but joining a toastmasters program could be a big help if you're struggling to make your point quickly and effectively, especially in front of a group.

5) Take an improv class! Nothing helps you learn to speak on your feet (and not get in your own way) better than improvisation. This is something Johnnie Moore talks about, and I think it's a lot more valuable than I previously recognized (on many different levels).

6) I know I don't have to say this, but for disclaimer purposes I will: don't use your glibness to *avoid* having to think more deeply or to "win." The best solution is to ask for time to think, research, analyze, evaluate, etc. Just because you *can* talk fast doesn't mean you *should*. But it helps to be quick enough to make the case for *why* you can't articulate your point on the spot, and why taking the time to do so could be of great value.

Again, if you're a manager, understand that most of us are biased to favor glibness (assuming it *sounds* smart at first listen). Most of us unconsciously link articulation with intelligence, and quicker is better. Don't be blinded by glibness. While it might be a huge asset for a rousing dinner party conversation, glibness can be potentially deadly in work.

I know there's more to say about this... and I might be completely wrong here, but I just can't put my finger on it right now... so I'll think some more and get back to you ;)

http://headrush.typepad.com/creating_passionate_users/2006/04/when_only_the_g.html

Pushing your skill set

By Kathy Sierra on April 11, 2006

Are you doing anything to keep up your skills? Some of you don't have a choice--especially if you're doing client work where each new job "forces" you to learn something new. But for those of us who--like me--are mostly working on our own stuff, we can get... a little lazy. The techniques we've been using are like old friends. Doing it the way we've been doing it feels comfortable and less risky.

Why learn a new API when the one we know works just fine? Sure, it's a little less efficient, and wasn't *really* designed to do *everything* the way we use it... but at least there's no new learning curve. It's more efficient to bend the things we know, then start up a new learning curve from scratch. Who wants to go through that initial "I suck at this" phase again?

At SXSW, the guys from skinnyCorp said that when they were finally earning enough from Threadless to quit doing client work, they noticed their skills began standing still. They tended to do only what they were already good at. They decided that if a *client* wasn't going to push their skills, they'd have to push *themselves*.

So, they chose to start a project that would force them into new knowledge/skill territory--Extra Tasty. I can't remember now which aspects of the drink recipe site were new for them, but I

was impressed that they recognized they needed to keep pushing. Of course, these guys have about a million ideas, so I shouldn't have been surprised.

Nobody wants to go through the suck stage. Being past that--where you start working in flow and everything clicks and you're good at what you do--is one of the *rewards* for going through it in the first place. But that whole "use it or lose it" thing applies to our brain. And by "use it", I don't mean using our brain to do the things we're already good at. If we don't continually keep pushing our brain in new ways, it starts to slide, just as parts of our body atrophy if we don't keep adding new physical stress (heavier weights, longer or faster runs, etc.)

What to do about it? I have only a few tips, so I'm hoping to hear more from you:

*** Use the Threadless approach**

Deliberately start a new project (even if it's just for fun) that you know will drive you into new research, experimenting, learning, etc.

*** Study for a certification exam.**

Yes, I'm biased on this one--I've been the lead developer on several of Sun's Java programmer and developer certifications, and I've written some certification books. But the reason I ended up working at Sun in the first place is because I was *already* a big advocate for certifications--devoting much of my free time to helping others prepare for and pass exams, although not for the reasons most people assume (and long before I ever thought about writing a book). I do think certifications are often a lousy way to evaluate a job candidate. And although the Sun exams are *extremely* difficult (and increasingly more "real world" than trivia-memorization), they certainly aren't a great reflection of realistic practical skill.

BUT--and this is the important part--studying for a certification exam *forces* you into new knowledge and skills *before* you need them. Most of us (and especially programmers) tend to use the same APIs and solutions that we've used before, applying them to each new somewhat related problem. We've gotten literally thousands of messages from people (between javaranch and emails from readers) who've said that they learned things from preparing for an exam that they wouldn't have otherwise known... and in some cases, those new things gave them a better

perspective when it came time to choose a tool for the next new problem. *Studying for a certification exam gave them a bigger toolbox.* Not everyone needs this kind of forced-learning approach. But for those who do, it can be the best kick in the ass to a skill upgrade that you can get.

[update: I forgot one of my favorites!... conferences]

* **Attend a conference**

And I'm not talking about your party skills. The best experience I've had at a conference was one where I actually left both my laptop and cell phone at home. Making wireless access available at conferences is, for me, a big mistake. When you're learning, you should be *learning*, not keeping up with work or surfing. I think people accomplished much more learning back in the days when you were forced to queue up to those few little stations where you could check your email.

Still, the beauty of most (by no means all) professional conferences or even just trade shows is that you absorb a lot more than just new technical knowledge. You learn what *other* people feel is important and useful, especially if you listen in or participate in hallway conversations. And for me, the enthusiasm you get by being around so many people who are *also* interested in the same topic is infectious. You can also get some of the same benefits just from attending user group meetings...

Some of the best experiences come from attending conferences or trade shows that are in a different domain. Yes, conferences are expensive, but often just the sub-$100 floor-only pass gives you much of the same benefits. [More on conferences in a separate post].

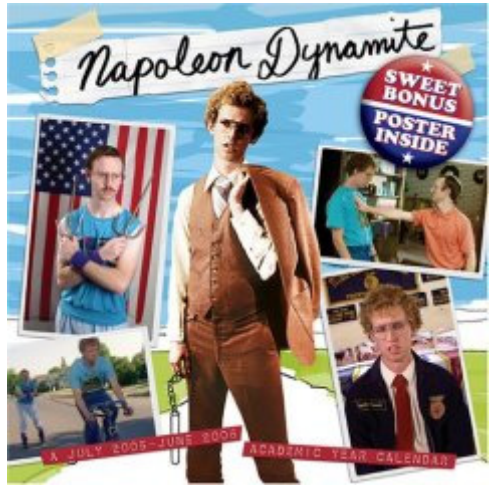
* **Blow your own mind**

Kevin Shockey (editor in chief of TUX magazine), sent me a couple of good links to brain-training articles:

1) Simple ways to make yourself far cleverer (is "cleverer" really a word?)

2) Brain training takes aging Japan by storm

So, what are *you* doing to improve your skills?



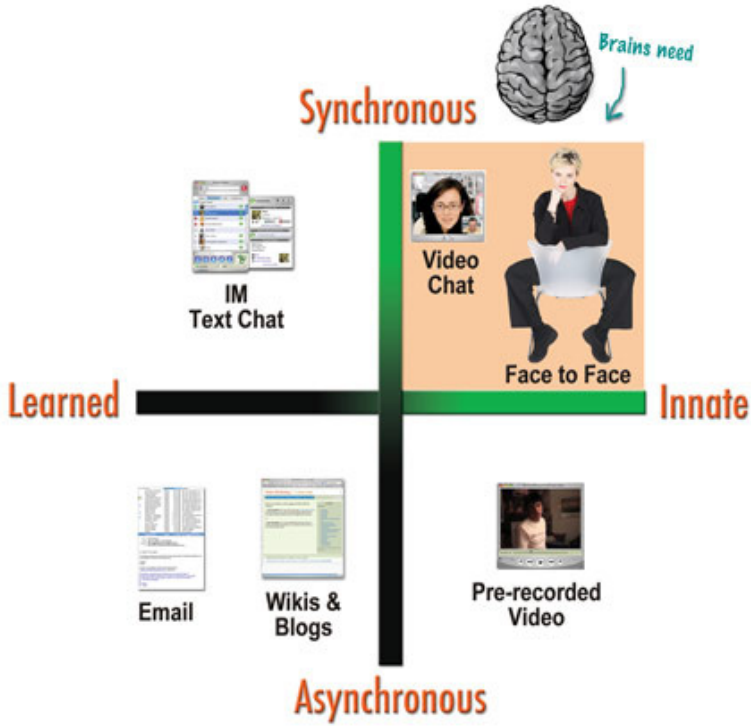
> *"You know, like nunchuck skills, bowhunting skills, computer hacking skills... Girls only want boyfriends who have great skills."*

[p.s. Does anyone have any idea what the guy in the photo at the top of this post is doing on that chalk board? I have no idea if it's real or random.]

http://headrush.typepad.com/creating_passionate_users/2006/04/pushing_your_sk.html

Why face-to-face still matters!

By Kathy Sierra on April 13, 2006



We email. We wiki. We blog. We IM. We convince ourselves that as long as we can write well, these are all good forms of communication. Perhaps in some ways even *better*, since we're not distracted (blinded, biased, seduced) by the person's physical presence.

And we are wrong.

According to the neuroscientists, anyway. I've just come back from a couple of days at the Conference on World Affairs, and attended a couple of different presentations where Dr. Thomas Lewis spoke. He has a particular interest in neurobiology (including the neurobiology of love), and what the brain does and does not want and need.

One of the key points he made was that we are fooling ourselves into thinking that text is even half as effective as face-to-face at communicating a message. He rattled off a ton of studies and

evidence, but I was too engrossed in the topic to take many notes, so I don't have references to most of them.

We all are aware of the notion that most of the information we get in a face-to-face communication is NOT from the words themselves, but rather from body language, facial expression, and tone of voice. What he finds troubling, though, is how we trick ourselves into thinking that (especially with all our text messaging tech) face-to-face is overrated. How we trick ourselves into thinking that we can truly *know* someone and experience real communication through text alone.

Although his explanation dove into the chemistry of face-to-face, LIVE interaction with another human vs. any other form of communication, one point was quite simple:

We never had to learn to process body language, facial expressions, and tone of voice. We evolved this capability...it's innate. But we had to spend *years* learning to read and write with any level of sophistication. The brain needs and *expects* these other--more significant--channels of information, and when they don't come... the brain suffers (and so does the communication). And the problem goes way beyond just an increased chance for misinterpretation.

Of course someone brought up smileys and winkies, etc. and he just gave us that "do you honestly believe that's somehow going to communicate anything remotely approximating subtle emotions?"

Part of the issue they've discovered in research is just how crucial the immediate response is. In still-face effect experiements with infants, for example, they learned that babies become immediately distressed when their mother maintains a "still face" that does not show any response/feedback with what the baby is doing. This makes sense, but what's really interesting is when they experimented with video. In some of these variations on the still-face effect, mothers and babies were on closed-circuit monitors where they could each see each other in real-time, through a television monitor. The babies were *much* happier when their mother's face was responsive to their own... less distressed than when the mother was right in front of the baby but maintaining a still face!

So, it was the *responsiveness* that mattered as well as the visual information. But just how quick does the feedback/response

need to be? When they took the same experiment but introduced a short delay (I can't recall the amount -- but it was less than a few seconds), the babies became distressed again. Even a small degree of latency killed the feedback/interaction/responsiveness the baby's brains were expecting and *needing*.

Of course, we're *adults*, and not babies, but again--Dr. Lewis pointed out that we still have the same basic neurochemistry, and that no matter how much we practice communicating through text, the brain still finds it stressful. He indicated that the *only* population whose lives have *improved* through the use of text over face-to-face are those with a serious problem of shyness. In the brains of the shy, he said, a previously unknown face triggers a fear or anxiety response in their amygdala which doesn't happen in text.

He said that video chat is *better* than any other form of non face-to-face, because you get facial expressions, tone of voice, body language, AND real-time responsiveness. But--he said there's still a very unsettling feature for the brain because there's really no way for BOTH speakers to make eye contact! If you look at the camera, then the other person sees *you* looking at *them*, but then your experience suffers. So you can either watch *the person you're chatting with*which helps *your* experience but causes theirs to sufffer (since you won't be looking into the camera, so to THEM you'll be looking down), OR you can look in the camera and improve *their* experience. But there's no way to have the camera right in your face, in a place where you can still look into the other person's eyes. Bottom line: You can see the camera or the person's eyes... but not both.

And even with the benefits of adding video to your chat, there's still a lot the scientists don't know about other factors surrounding human communication that can't be captured electronically. Smell, for example, might be far more powerful than we realize--even when below our conscious awareness...

This is just a small taste of the things he talked about, but I wanted to get it down while it was still fresh.

So, what to do if you're like me and work mostly remote from co-workers? Using AV chat is a HUGE improvement for the reasons I listed. But there's no substitution for face-to-face... so anything you can do to try to interact with people IN PERSON is critical. Even if it's just a once a year meeting, the very fact that

you've had a chance to see and hear that person and experience them in front of you goes a long way toward helping you when you get back to your remote office and return to text.

But there's more--he stressed that having face-to-face interaction is so crucial to the brain that even if you *can't* do face-to-face with your co-workers, we should *all* try to make sure we have a healthy amount of live social interaction. So, join a local user group. Spend more time with friends. Attend conferences. And--he stressed most of all--*stop watching television*. (more on that in another post, but part of it has to do with the way having television on tricks one part of your brain into thinking you're having a social interaction--all these people having conversations in your living room--but fails to give the brain what it *expects* and needs from that interaction.

I'll say more tomorrow, but for now I want to add that I'm thrilled to have met many of you in person at conferences and other events, and I'm hoping I'll have a chance one day to meet more of you. For now, you'll just have to trust that I have a smile on my face as I type this :)

[And photos of your face help too, so if you dare (and circumstances permit), you should post a picture of your face somewhere and make sure people have it. Ask the person you're emailing with if you can send a photo so they "know who they're talking to" a little better, and ask if they'll do the same. More on other tips a little later...]

http://headrush.typepad.com/creating_passionate_users/2006/04/why_facetoface_.html

Angry/negative people can be bad for your brain

By Kathy Sierra on April 17, 2006

Everyone's favorite A-list target, Robert Scoble, announced the unthinkable a few days ago: he will be moderating his comments. But what some people found far more disturbing was Robert's wish to make a change in his life that includes steering clear of "people who were deeply unhappy" and hanging around people who *are* happy. The harsh reaction he's gotten could be a lesson in scientific ingorance, because the neuroscience is behind him on this one.

Whether it's a *good* move is up to each person to decide, but I've done my best here to offer some facts. [Disclaimer: I'm not an authority on the brain! I have, however, spent the last 15 years doing research and applying it, both in my work and also because I have a serious brain disorder, and my brain knowledge could be a matter of life and death. Another disclaimer: I haven't spoken with Robert about this; I'm simply offering some science that supports the decision he may have made for entirely different reasons.]

A few things I'll try to explain in this post:

1) One of the most important recent neuroscience discoveries--**"mirror neurons"**, and the role they play in a decision like Robert's

2) The heavily-researched social science phenomenon known as **"emotional contagion"**

3) Ignorance and misperceptions around the idea of **"happy people"**

Mirror Neurons

Mirror neurons have been referred to by scientists like V.S. Rmachandran as one of the most important neuroscientific breakthroughs of recent history. This Nova video is a great introduction, but here's the condensed version:

There is now strong evidence to suggest that humans have the same type of "mirror neurons" found in monkeys. It's what these neurons do that's amazing--they activate in the same way when you're *watching* someone else do something as they do when you're doing it yourself! This mirroring process/capability is thought to be behind our ability to empathize, but you can imagine the role these neurons have played in keeping us alive as a species. We learn from *watching* others. We learn from *imitating* (mirroring) others. The potential problem, though, is that these neurons go happily about their business of imitating others *without* our conscious intention.

Think about that...

Although the neuroscientific findings are new, your sports coach and your parents didn't need to know the cause to recognize the effects:

"Choose your role models carefully."

"Watching Michael Jordan will help you get better."

"You're hanging out with the wrong crowd; they're a bad influence."

"Don't watch people doing it wrong... watch the experts!"

We've all experienced it. How often have you found yourself sliding into the accent of those around you? Spend a month in England and even a California valley girl sounds different. Spend a week in Texas and even a native New Yorker starts slowing down his speech. How often have you found yourself laughing, dressing, skiing like your closest friend? Has someone ever observed that you and a close friend or significant other had similar mannerisms? When I was in junior high school, it was tough for people to tell my best friends and I apart on the phone--we all sounded so much alike that we could fool even our parents.

But the effect of our innate ability and *need* to imitate goes way past teenage phone tricks. Spend time with a nervous, anxious person and physiological monitoring would most likely show *you* mimicking the anxiety and nervousness, in ways that affect your brain and body in a concrete, measurable way. Find yourself in a room full of pissed off people and feel the smile slide right off your face. Listen to people complaining endlessly about work, and you'll find yourself starting to do the same. How many of us have been horrified to suddenly realize that we've spent the last half-hour caught up in a gossip session--despite our strong aversion to gossip? The behavior of others we're around is nearly irresistible.

When we're consciously aware and diligent, we can fight this. But the stress of maintaining that conscious struggle against an unconscious, ancient process is a non-stop stressful drain on our mental, emotional, and physical bandwidth. And no, I'm not suggesting that we can't or should'nt spend time with people who are angry, negative, critical, depressed, gossiping, whatever. Some (including my sister and father) chose professions (nurse practitioner and cop, respectively) that demand it. And some (like my daughter) volunteer to help those who are suffering (in her case, the homeless). Some people don't want to avoid their more hostile family members. But in those situations--where we *choose* to be with people who we do *not* want to mirror--we have to be extremely careful! Nurses, cops, mental health workers, EMTs, social workers, red cross volunteers, fire fighters, psychiatrists, oncologists, etc. are often at a higher risk (in some cases, WAY higher) for burnout, alcholism, divorce, stress, or depression unless they take specific steps to avoid getting too sucked in to be effective.

So, when Robert says he wants to spend time hanging around "happy people" and keeping his distance from "deeply unhappy" people, he's keeping his brain from making--over the long term--negative structural and chemical changes. Regarding the effect of mirror neurons and emotional contagion on personal performance, neurologist Richard Restak offers this advice:

"If you want to accomplish something that demands determination and endurance, try to surround yourself with people possessing these qualities. And try to limit the time you spend with people given to pessimism and expressions of futility. Unfortunately, negative emotions exert a more

powerful effect in social situations than positive ones, thanks to the phenomena of emotional contagion."

This sounds harsh, and it is, but it's his recommendation based on the facts as the neuroscientists interpret them today. This is not new age self-help--it's simply the way brains work.

Emotional Contagion

Steven Stosny, an expert on road rage, is quoted in Restak's book:

"Anger and resentment are thet most contagious of emotions," according to Stonsy. "If you are near a resentful or angry person, you are more prone to become resentful or angry yourself. If one driver engages in angry gestures and takes on the facial expressions of hostility, surrounding drivers will unconsciously imitate the behavior--resulting in an escalation of anger and resentment in all of the drivers. Added to this, the drivers are now more easily startled as a result of the outpouring of adrenaline accompanying their anger. The result is a temper tantrum that can easily escalate into road rage."

If you were around one or more people with a potentially harmful contagious disease, you would probably take steps to protect yourself in some way. And if *you* were the contagious one, you'd likely take steps to protect others until you were sure the chance of infecting someone else was gone.

But while we all have a lot of respect for physical *biological* contagions, we do NOT have much respect for physical *emotional* contagions. (I said "physical", because science has known for quite some time that "emotions" are not simply a fuzzy-feeling concept, but represent physical changes in the brain.)

From a paper on Memetics and Social Contagion,

"...social scientific research has largely confirmed the thesis that affect, attitudes, beliefs and behaviour can indeed spread through populations as if they were somehow infectious. Simple exposure sometimes appears to be a sufficient condition for social transmission to occur. This is the social contagion thesis; that sociocultural phenomena can spread through, and leap between, populations more like outbreaks of measels or chicken pox than through a process of rational choice."

Emotional contagion is considered one of the primary drivers of group/mob behavior, and the recent work on "mirror neurons" helps explain the underlying cause. But it's not just about groups. From a Cambridge University Press book:

"When we are talking to someone who is depressed it may make us feel depressed, whereas if we talk to someone who is feeling self-confident and buoyant we are likely to feel good about ourselves. This phenomenon, known as emotional contagion, is identified here, and compelling evidence for its affect is offered from a variety of disciplines - social and developmental psychology, history, cross-cultural psychology, experimental psychology, and psychopathology."

[For a business management perspective, see the Yale School of Management paper titled The Ripple Effect: Emotional Contagion In Groups]

Can any of us honestly say we haven't experienced emotional contagion? Even if we ourselves haven't felt our energy drain from being around a perpetually negative person, we've watched it happen to someone we care about. We've noticed a change in ourselves or our loved ones based on who we/they spend time with. We've all known at least one person who really *did* seem able to "light up the room with their smile," or another who could "kill the mood" without saying a word. We've all found ourselves drawn to some people and not others, based on how we *felt* around them, in ways we weren't able to articulate.

So, Robert's choice makes sense if he is concerned about the damaging effects of emotional contagion. But... that still leaves one big issue: is "catching" only positive emotions a Good Thing? Does this mean surrounding ourselves with "fake" goodness and avoiding the truth? Does surrounding ourselves with "happy people" mean we shut down critical thinking skills?

Happy People

The notion of "Happy People" was tossed around in the Robert-Lost-His-Mind posts as something ridiculous at best, dangerous at worst. One blogger equated "happy people" with "vacuous". The idea seems to be that "happy people" implies those who are oblivious to the realities of life, in a fantasy of their own creation, and without the ability to think critically. The science, however, suggests just the opposite.

Neuroscience has made a long, intense study of the brain's fear system--one of the oldest, most primitive parts of our brain. Anger and negativity usually stem from the anxiety and/or fear response in the brain, and one thing we know for sure--when the brain thinks its about to be eaten or smashed by a giant boulder, *there's no time to stop and think!* In many ways, fear/anger and the ability to think rationally and logically are almost *mutually exclusive*. Those who stopped to weigh the pros and cons of a flight-or-fight decision were eaten, and didn't pass on their afraid-yet-thoughtful genes. Many neuroscientists (and half the US population) believes that it is exactly this fear != rational thought that best explains the outcome of the last US presidential election... but I digress.

Happines is associated most heavily with the *left* (i.e. logical) side of the brain, while *anger* is associated with the *right* (emotional, non-logical) side of the brain. From a Society for Neuroscience article on Bliss and the Brain:

"Furthermore, studies suggest that certain people's ability to see life through rose-colored glasses links to a heightened left-sided brain function. A scrutiny of brain activity indicates that individuals with natural positive dispositions have trumped up activity in the left prefrontal cortex compared with their more negative counterparts. "

In other words, ***happy people are better able to think logically.***

And apparently happier = healthier:

"Evidence suggests that the left-siders may better handle stressful events on a biological level. For example, studies show that they have a higher function of cells that help defend the body, known as natural killer cells, compared with individuals who have greater right side activity. Left-sided students who face a stressful exam have a smaller drop in their killer cells than right-siders. Other research indicates that generally left-siders may have lower levels of the stress hormone, cortisol."

And while we're dispelling the Happy=Vacuous myth, let's look at a couple more misperceptions:

"Happy people aren't critical."

"Happy people don't get angry."

"Happy people are obedient."

"Happy people can't be a disruptive force for change."

Hmmm... one of the world's leading experts in the art of happiness is the Dalai Lama, winner of the Nobel Peace Prize in 1989. Just about everyone who hears him speak is struck by how, well, *happy* he is. How he can describe--with laughter--some of the most traumatizing events of his past. Talk about *perspective*...

But he is quite outspoken with his criticism of China. The thing is, he doesn't believe that criticism *requires* anger, or that being happy means you can't be a disruptive influence for good. On happiness, he has this to say:

"The fact that there is always a positive side to life is the one thing that gives me a lot of happiness. This world is not perfect. There are problems. But things like happiness and unhappiness are relative. Realizing this gives you hope."

And among the "happy people", there's Mahatma Gandhi, a force for change that included non-violent but oh-most-definitely-disobedient behavior. A few of my favorite Gandhi quotes:

In a gentle way, you can shake the world.

It has always been a mystery to me how men can feel themselves honoured by the humiliation of their fellow beings.

But then there's the argument that says "anger" is morally (and intellectually) superior to "happy". The American Psychological Association has this to say on anger:

"People who are easily angered generally have what some psychologists call a low tolerance for frustration, meaning simply that they feel that they should not have to be subjected to frustration, inconvenience, or annoyance. They can't take things in stride, and they're particularly infuriated if the situation seems somehow unjust: for example, being corrected for a minor mistake."

Of course it's still a myth that "happy people" don't get angry. Of course they do. Anger is often an appropriate response. But there's a Grand Canyon between a happy-person-who-gets-angry and an unhappy-angry-person. So yes, we get angry. Happiness is not our only emotion, it is simply the outlook we

have chosen to cultivate because it is *usually* the most effective, thoughtful, healthy, and productive.

And there's this one we hear most often, especially in reference to comment moderation--"if you can't say whatever the hell you want to express your anger, you can't be authentic and honest." While that may be true, here's what the psychologists say:

"Psychologists now say that this is a dangerous myth. Some people use this theory as a license to hurt others. Research has found that "letting it rip" with anger actually escalates anger and aggression and does nothing to help you (or the person you're angry with) resolve the situation.

It's best to find out what it is that triggers your anger, and then to develop strategies to keep those triggers from tipping you over the edge."

And finally, another Ghandi quote:

"Be the change that you want to see in the world."

If the scientists are right, I might also add,

Be *around* the change you want to see in the world.

Remember the flight attendant's advice... you must put on your own oxygen mask first.

[UPDATE: I had seen so many blog posts painting "happy" as equivalent to any-synonym-for-brainless, that I didn't really care who used which word--and word "vacuous" was just one more example of what's been said about Robert and the Happy People. But, the author of the post that first used that word was Shelley Powers, who feels this to be a very bad move on my part, so, I'd like to correct that the original post with the word "vacuous", and Shelley's response to my post here.]

http://headrush.typepad.com/creating_passionate_users/2006/04/angrynegative_p.html

Cognitive seduction (a Typology of User Experience Pleasures)

By Kathy Sierra on April 19, 2006

Is Sudoku seductive? Is chess sexy? Is crafting code a turn-on? To our brains, absolutely. But while most of us don't use the word "seductive" in non-sexual contexts, good game designers do. They know what turns your brain on, and they're not afraid to use it. They're experts at the art of "cognitive arousal", and if we're trying to design better experiences for our users, we should be too.

I'm not talking about using *sex* to arouse your *brain*. I'm talking about the kind of "experiential pleasure" that comes from solving a puzzle, overcoming a challenge, exploring new territory, becoming swept up in a narrative, interacting with others in a social framework, and discovering something new about yourself. I'm talking about things that engage the brain in a way that Gregory Bateson describes in The Ecology of Mind, discussing games:

"... they are important emotions that we feel and go through and enjoy and find in some mysterious ways to enlarge our spirit."

In the book Rules of Play, by Katie Salen and Eric Zimmerman, game designer Marc LeBlanc defines 8 categories of experiences

in a "typology of pleasure". (A slightly different approach also described in the book was developed by Michael J Apter, developer of Reversal Theory.) I took a moment to tweak the "the kinds of experiential pleasure players derive from playing games" to apply it to the NON-game experiences we create for our users.

Typology of Cognitive Pleasures

(in no particular order)

1. Discovery

User experience as exploration of new territory

2. Challenge

User experience as obstacles to overcome, goals lying just beyond current skill and knowledge levels

3. Narrative

User experience as story arc (user on hero's journey) and character identification

4. Self-expression

User experience as self-discovery and creativity

5. Social framework

User experience as an opportunity for interaction/fellowship with others

6. Cognitive Arousal

User experience as brain teaser

7. Thrill

User experience as risk-taking with a safety net

8. Sensation

User experience as sensory stimulation

9. Triumph

User experience as opportunity to kick ass

10. Flow

User experience as opportunity for complete concentration, extreme focus, lack of self-awareness

11. Accomplishment

User experience as opportunity for productivity and success

12. Fantasy

User experience as alternate reality

13. Learning

User experience as opportunity for growth and improvement

I'm going to add this as one of my gazillion checklists to help stay focused on what's going on between the user's ears, and to keep motivating me to think about ways to give users a better experience. Clearly we can't--and wouldn't want to--design a user experience that includes *all* of those things, but even the best games don't. The point is to see if there are some we can add, or at least tune, to give our users a richer (hi-res) experience.

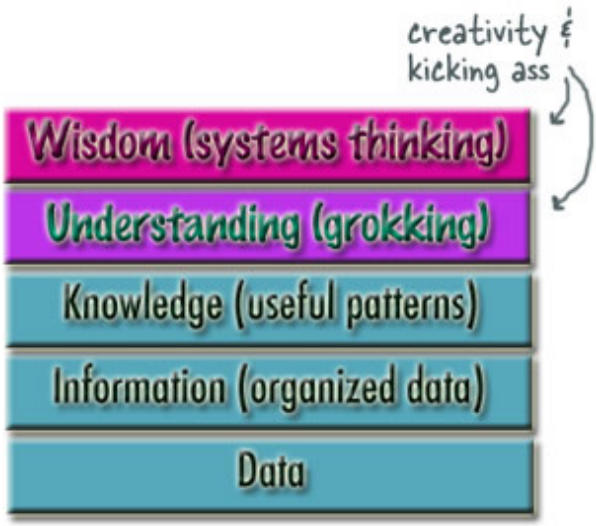
Given that I spent all of twenty minutes on this, I'm seriously hoping that you'll add to it or refine it for me and my co-author/teacher/developers.

And yes, the picture at the top was a blatant attempt to arouse your brain ;)

http://headrush.typepad.com/creating_passionate_users/2006/04/cognitive_seduc.html

Moving up the wisdom hierarchy

By Kathy Sierra on April 23, 2006



If you're an aggregator "harnessing collective intelligence", what are you aggregating? If it's data and information, you're competing with just about *everything*--Google searches, reference docs both online and printed, the majority of tech books and articles, etc. But if you're aggregating up the hierarchy through knowledge, and especially understanding and wisdom, you're adding *huge* value to someone's life.

If you're in knowledge management, what exactly are you capturing and managing?

If you're a teacher, what are you teaching? Facts and information, or practical knowledge and understanding? Are you teaching the What and the How but without the Why and the When? More importantly, what are you *testing*? (Not that in the US most public school teachers have a huge say in this, unfortuntately)

If you're a tech writer, what are you writing?

If you're creating tutorials and docs for your users, what are you focusing on? Remember, kicking ass and creativity usually doesn't happen at the data, information, and even the knowledge level. If you're not taking your users up the top tiers,

you might be missing the chance to give them more inspiring (cognitively arousing?) experiences.

The idea (and a zillion variations including mine) of the Data-Information-Knowledge-Wisdom (DIKW) hierarchy has been going around for quite some time (especially in knowledge management circles), but how come *everyone* isn't paying more attention to it?! Some are, of course--Richard Saul Wurman in particular, has made a point of referring to his work as "the understanding business", rather than stopping with information or even knowledge.

[Other links: Russell Ackoff wikipedia entry, data-to-wisdom curve, WIKID + Power model, a different take on the origin of the DIKW model]

And fortunately, even those focused on information architecture and information design often consider knowledge and understanding as well as information.

The thing is, the demand for tools and services that take us (mere mortals with our slowly evolving brains) keeps increasing, potentially exponentially. According to Wurmans' Information Anxiety (one of my all-time favorite books):

"There has been more information produced in the last 30 years than during the previous 5,000."

There are a gazillion places to get a roundup of basic data/facts and information (which of course we need). It's not tough to find "what to know and how to do" knowledge out there either. But it's not until we get to the higher levels that we start truly getting *if* and *when* we should use something. What are the long-term consequences? What are the ethical considerations?

If we could teach kids in elementary school just *one* thing (besides reading), my wish is that it would be systems thinking. But too much of even our adult training/education (including much of what I create) is focused on short-term "just-the-facts-mam" or how-to hacks and tutorials. We obviously *need* reference info and how-to's, or everything comes to a grinding halt, but without the higher elements of understanding and wisdom, we might never recognize that the thing we're learning to do is NOT the right choice!

One of the easiest ways to bump something up a level is to simply include a few things like:

When NOT to use something

(Our Design Patterns book, for example, talks about not just how and when to apply patterns but when *not* to.

Consequences

How to recognize when it was NOT a good idea to use or apply this [whatever]

Lessons learned from others, real case studies good AND bad

Links/referrals to communities of practice

Simulations (best of all--provide the tools and scenarios that let *them* discover what the long-term consequences could be)

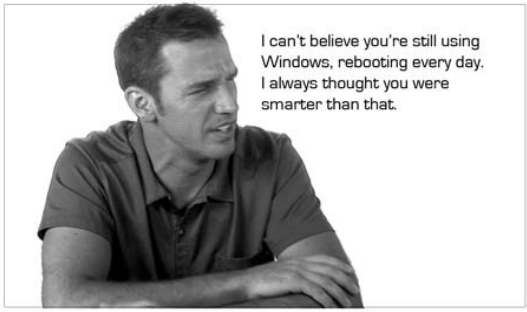
So, what are you doing to move up the hierarchy... or to move your *users* (students, customers, readers, members, guests, etc.) up the hierarchy?

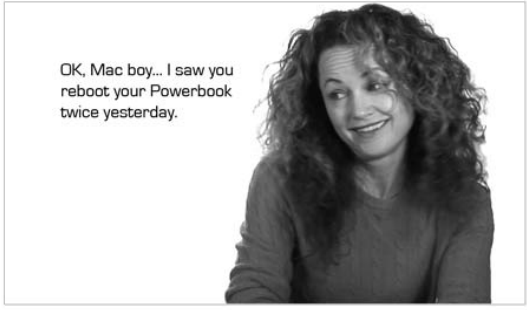
[Update: Shawn Callahan left a comment with an alternative perspective that's well worth the read.]

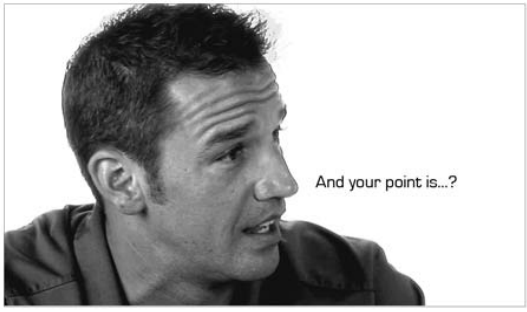
http://headrush.typepad.com/creating_passionate_users/2006/04/moving_up_the_w.html

My passion is awesome, your passion is lame

By Kathy Sierra on April 25, 2006







It's the golfer who spends an entire weekend hitting a little ball with a stick but slams his co-worker for "wasting time with mindless video games". It's the Java fan dissing the Ruby enthusiast. It's the audiophile who paid $8,000 for a home stereo but can't *believe* you spent $1,000 on that Nikon lens.

While we usually have no trouble justifying our passion to *ourselves* we have a tough time justifying it to *those who don't share that particular passion*. We all have passion double-standards. Mine makes sense, yours doesn't. The time and money I spend on my passion is worth it, yours is a waste.

In the past, I've said that one characteristic of people with a passion is that they are *irrational* about that passion. But by whose judgement? Who decides it's rational to spend an extraordinary amount on the best cooking pans and utensils, yet irrational to buy a digital SLR when "my $200 point-and-shoot Canon takes just as good a picture?" Who decides that you shouldn't watch too much TV, but *your* cause is judgemental and self-righteous? Who decides that it should be *obvious* why the iPod is worth the extra money, but come on--spending *that* much on *dog food*!?

The point is, one person's *irrational zealotry* is another person's reasonable passion. It all depends on your perspective. (Of course this *is* a continuum--with a few obsessed ones out on the extreme edge of the passion curve. And unfortunately, *the extreme zealots* are the ones that folks like to point out as examples of why your passion is so irrational.)

So, what can we do to help our users explain their passion to others? Sales and marketing people put a ton of effort into providing the "rational justification" for a buying, joining, whatever-it-is decision, but once they've accepted the justifications and paid their money, that's it. Yet as they start to become truly passionate, this is when they may need justifications the most! Not for themselves, but for others. And it's a different kind of justification...

Your passionate users don't need you to help justify the *product*, they need you to help them justify the *passion*.

It does me no good to explain the benefits of the $1,000 lens I just bought if my passion for photography makes no sense to you. It does you no good to justify why you joined rock climbing gym Foo instead of Bar, if I think your passion for climbing is not just irrational but *dangerous*.

And how can I possibly explain why I spent all that extra money for a Parelli-brand horse halter if you think "natural horsemanship" is nothing but marketing hype?

But this gets to the heart of one of the most important aspects of passion--that it goes beyond the product/service/cause itself. *Our passions often represent something about who we are*. For many of us, the thing we're passionate about is not just a hobby, product, service, cause, etc... *it's a way of life*.. Ted Leung explainined to me that as a result of his relatively recent passion for photography, he "sees the world differently now." Passionate golfers have apparently elevated golf to some kind of spiritual status--it is, for them, about much more than just hitting a ball with a stick. Ditto with fly fishing (it's apparently not about the fish or the flys). The guys from 37signals offer much more than software apps... they represent a philosophy (the whole "getting real/it-just-doesn't-matter" thing). MindJet's Mind Manager not a mind-mapping tool, it's a way of *thinking*.

If we truly want to support our *passionate* users (or help them to get there in the first place), we need to consider how this passion fits in the context of their work or personal life. If our user's spouse is complaining about all the time they spend doing [whatever it is], can we offer a friends/family free workshop? Educational demos? Create a DVD that helps define some of what this means to people who have a passion for this thing? Case studies? It might be recommending an influential film that speaks to the deeper nature of this passion (I must admit, I got teary-eyed over a golf movie once). At the very least, we should make sure that *existing* users have access to the same kinds of white papers, product comparisons, technical or financial justifications that we give to *prospective* users. (Although again, this addresses the THING and not the passion.)

Bottom line: if we want to help our users explain their passion to others, we need to help give them the tools to do so. It may often be unsuccessful--some people will always have a closed mind around things they can't see or feel for themselves--but it's way better than nothing.

Not that I should have to justify *my* passions, of course, because--DUH--the justification is self-evident to anyone who "gets it." ;)

http://headrush.typepad.com/creating_passionate_users/2006/04/my_passion_is_a.html

Can marketing be honest AND motivating?

By Kathy Sierra on May 4, 2006

Most of us who aren't Marketers have a dim view of Marketing. Yet here we are, non-Marketers, thinking about marketing. We have something we believe in that we want to promote. But we don't want to be dishonest or unethical. But what if we're like competitive athletes...where we're at a disadvantage if we're trying to run naturally and fairly while everyone *else* is on steroids?

If we don't claim that our product or service will get them rich, famous, or laid... we still need to think about motivation. We can be both honest AND motivating, but we can't assume that being honest is *inherently* motivating. Since our focus here is "passionate users", I'm making these assumptions:

1) There is something your product, service, or cause can help your users kick butt in. Something they can keep getting better and better at, where better means a higher-resolution experience. Where being better is better.

2) You are already working on ways to help your users pass the "Suck Threshold" and move more quickly toward the "Kick Ass" threshold. In other words, there are ways (either from you or a user community or third-party) for the user to keep learning.

[Note for those who haven't yet figured out what you can help your users kick ass in: that's the first crucial step. And remember, it's almost never about making them more of an expert on your *tool*, it's about helping them get really good at whatever it is they *do* with your tool. Nikon's tutorials aren't about making *camera* experts, they're making *photography* experts. Parelli horsemanship isn't making experts on *training equipment*, they're making expert *horse trainers*. Apple's iLife products aren't making software experts, they're making *home video, photography, and music* experts. (Yes, I'm using "expert" in the sense that even if most people never get there, the promise is that it's *possible*.)

And it isn't *always* directly related to what you offer. We've talked about this before--the guy who makes USB thumb drives, for example, could choose to help teach users to give kick-ass presentations.]

So, what's the *initial* motivation for someone to take the first step with your product, service, or cause? Why should they download your free trial? Why should they visit your gym/store/church for the first time? Marketers and Advertisers might delve into the psychology of human needs to answer that question (maybe a spin through some variation of Maslow's hierarchy), to figure out which they can tap into, but we think there's a simpler way to look at it.

The most common reason people take the first step toward something they may ultimately develop a passion for is because these THREE things are present:

1) There is a clear, compelling picture of what it might be like to be an expert (or at least really good) at this thing.

2) There is a clear path to getting there.

3) There is an obvious and relatively easy first step.

If you show me an example of what it could mean to be really good at this thing-you-can-help-me-kick-ass-in, I might find that motivating. Whether it's photos of people doing it, or the result of what they do using your thing, or video clips, or testimonials (users talking about how *they* kick ass, not how great *you or your product* are).

But it doesn't matter how motivating it looks to become really good at this if I can't imagine that I--a mere mortal--could ever get there. You must show me a realistic path to getting there. Do you have tutorials or training at all levels including total newbie? User support groups? Descriptions of each stage and what it takes to reach that stage, both financially (if that applies) and time/effort?

So, does your product, service, or cause need to be *motivating*? Not necessarily. But the thing-you-will-help-users-kick-ass-in needs to be. We assume that someone, somewhere loves being really good at whatever it is that you can help people get into and get better at. Whatever it is that they love about it, *that* is your motivating picture, even if it's nothing more than the glorious feeling of control I'll have when I've learned to use your productivity app in a meaningful, productive way.

It won't get them laid, it won't make them an instant millionaire, it won't help them lose 20 pounds (well, maybe that one could be true ;)

But you don't *need* those claims if you're able to paint a clear, realistic picture of something people will find *worth* the effort of getting good.

http://headrush.typepad.com/creating_passionate_users/2006/05/can_marketing_b.html

Don't give in to feature demands!

By Kathy Sierra on May 10, 2006

The more successful the product or service is, the stronger the pressure to give in to user requests. The more users you have, the more diverse the requests. One user's must-have-or-else feature is another user's deal-killer. And the more popular your product or service is, the more those *requests* start turning into demands and ultimatums, and finally *very* harsh criticisms.

The worst thing we can do is give in. But as the requests/demands and criticisms become louder and angrier, the harder it is to resist the siren call-- "But if we just added this *one* thing... these guys would ease up."

But when we've blended all the colors into one muddy blob, then nobody hates us, and nobody is delighted, excited, or turned on by what we do. We become *mediocre*. Usually the worst place to be.

But we can't just *ignore* the suggestions, requests, and criticism. So how do we filter the useful feedback from the crap? How do we know *who* we should listen to?

Let's say that most of the people who might make feature requests or demands fall into one of these categories:

1) The Active Haters

Those who hate you no matter what you do. The more popular you become, the louder they become.

2) The Extreme Critic

Those for whom "hate" is too strong a word, but they have a mile-long list of things you're doing wrong.

3) The Against-My-Will User

Those who are forced to use your product or service, but aren't happy about it. Maybe you're the only one in your category, or their employer or client made the decision--whatever the reason, they don't like it.

4) The Previously Satisfied, Now Overwhelmed User

Those who were satisfied once, but have now become unhappy because of updates and changes in the product. They just want to go back to the way things *were*! They could not care less how much more productive/creative/stimulated they could be if they embraced the changes. They worked hard to cross the "suck threshold", and they don't want to go back.

5) The Previously Satisfied, Now Impatient User

These folks are unhappy for the *opposite* reason as the Now Overwhelmed User. They're pissed off that your updates and changes haven't kept up with their requests which have recently turned into demands. They're the most likely to go elsewhere without much warning, because they feel they've outgrown you or that you just don't care. [Or just don't care *about them*.]

6) The Previously Passionate, Now Pissed Off User

Ahhhh... the most interesting users of all. They were your biggest fans, but You Just Wouldn't Listen to them and now they're threatening to go elsewhere, passionately, but they're giving you plenty of notice. They *want* to regain that spark they once felt, but the original bond only goes so far.

7) The Still Satisfied User

Whether they're too new to know what else they might want, or you've simply met all their expectations and requirements, these users are humming along without problems.

8) The Still Passionate User

Our favorites. They're thrilled with what your product or service lets them do. But... this doesn't mean they don't have requests. They can be passionate without necessarily being *satisfied*, but they'll do everything they can to *help* you improve. The trouble is, we're back to the same problem... one passionate user's idea of Useful Improvement is another's Worst Mistake You Could Make Ever.

9) The Uncaring User

They just don't *think* about your product or service. They use it, but if something else came along that did the same thing, they might not even notice the difference.

10) Marketing (and sales)

They ask for things based on research (often dubious), or their own feeling/judgement about what people want, or perhaps based on feedback from sales people, etc.

11) Engineers/Developers/Producers with little or no user contact

They may want to put in "this really cool thing we could do", regardless of whether anyone has asked for it. This could be the classic "solution in search of a problem."

12) Engineers/Developers/Producers in close contact with users

These are the folks that interact with the customer/clients/users on a regular basis, usually prior to and during development, with less contact *after* the thing is done.

13) Customer Service / Tech Support

Those on the front line! The ones who are in the closest contact with users.

[I've probably left out some crucial category, so help me out here.]

Now the big question is, *who do we listen to*?

I'd love for you to add *your* advice, but here's a rough cut of one way to think about it:

COMPLETELY IGNORE:

* The Haters

The Haters are too irrational about you to offer any criticism you can *trust*. They might have valid feedback, but if it's valid you'll hear the same feedback from other less-hateful users.

* The Uncaring

~~* Marketing~~ (couldn't resist)

KEEP ONE EAR OPEN:

But only for new ideas, not for specific requests and criticisms (again, if a criticism or request is really valid -- you'll hear about it from many others)

* The Extreme Critic

* The Against-My-Will user

* Engineers/developers with little or no user contact

Just because you *can* do something "really cool", doesn't mean you *should*.

LISTEN...

...but resist the pressure to give in until you've considered all the long-term implications to all the OTHER user groups:

* Previously Satisfied, Now Overwhelmed

I think we're often better off trying to *train* them -- to help them get up to speed on the new stuff without having to suck again.

* Still Satisfied

Their suggestions aren't coming from real motivation, but they might have a good idea...

* Marketing

(Unless you really REALLY know and trust that your marketing people deeply understand both the product *and* the users. Too often, they just want to make something more "sellable", regardless of the big picture.)

* Still Passionate

These are often the most difficult to resist! But they may ask for things that nobody needs, just because--like engineering--it would be cool.

PAY ATTENTION:

That still doesn't mean you'll give in, but feedback from these users should be given a lot of weight.

* Previously Satisfied, Now Impatient

These are the people who are on the verge of becoming passionate users, if you can keep up with them! One problem is that they may want to push you into territory that is out of your niche, and if you please *them* by adding advanced features, you have to make sure you don't create new "Now Overwhelmed" users as a result.

But anytime we can add advanced features without hurting the low-lever users (who may stay that way forever once they have the basics down), we should consider it. If we want users to be

passionate, we have to give them new challenges--new ways to learn and grow and apply their new skills and knowledge.

Another option is to create a separate more advanced product -- like Apple's three different music apps (when you outgrow GarageBand you go to Logic Express. When you outgrow Logic Express, you go to Logic, etc.)

* The Previously Passionate, Now Pissed Off User

It took a LOT to piss them off, because they've been extra forgiving and willing to overlook your faults, but only to a point. They know your product better than anyone else. One problem with these guys is that they tend to be so advanced that they don't necessarily reflect your average user. If you can find a way to work with them (and they desperately WANT you to succeed and win them back), and perhaps compromise, they'll be your biggest champions again. At the very least, we should tell them that we've listened carefully, and here's why we're having trouble adding or changing this thing... they might even have an answer we hadn't thought of. And at least we've given them the respect they deserve.

* Engineering/Developers in close contact with users

They're close to the product, they understand how it's being used. Best of all, they might know the best 80/20 way (most bang for the development buck?) to do the smallest thing that'll have a positive impact. They are the best source of ideas for compromise without hurting existing capabilities. (But I've spent enough time as a member of this group to know that we can't help ourselves from "adding this one cool thing" that we know "will take no time at all." So challenge our assumptions.)

PAY THEM FOR THEIR FEEDBACK:

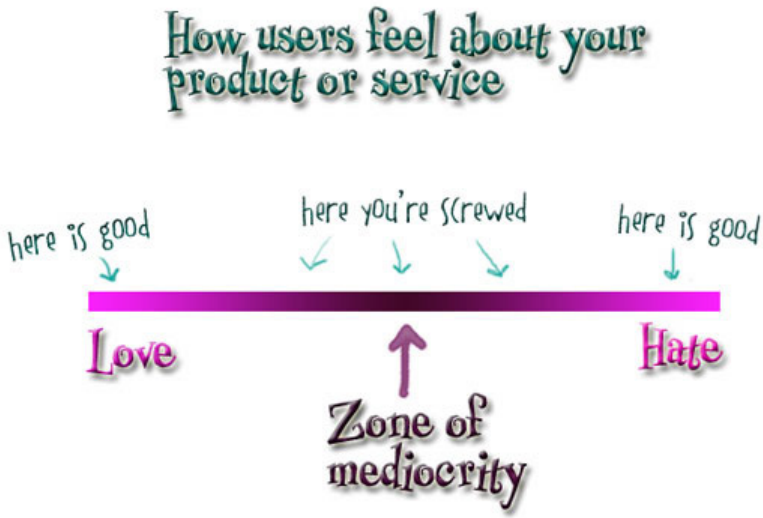
* Customer service / tech support

Nobody knows where the pain is the way the front line does, and they can probably get very specific about the exact source of that pain, which lets you do a more surgical fix rather than a sweeping change. But one question... why--in so many companies--are these the people who often get the *least* amount of respect and voice about this?

* You

The one user category I didn't mention. In the end, we have to trust ourselves. This is a controversial position--we're not supposed to be building things for *us*... it's not *about* us. But that doesn't mean we aren't the ones who can make the best overall decisions. We're the ONLY ones who get *all* the feedback and can think through the long-term implications, and can see how pleasing one user group might piss off another group, so which group do we choose?

If you have a product or service or cause where you "eat your own dog food", then you're in at least as good a position as any user to know what's broken and what *will* break if you listen to this request.



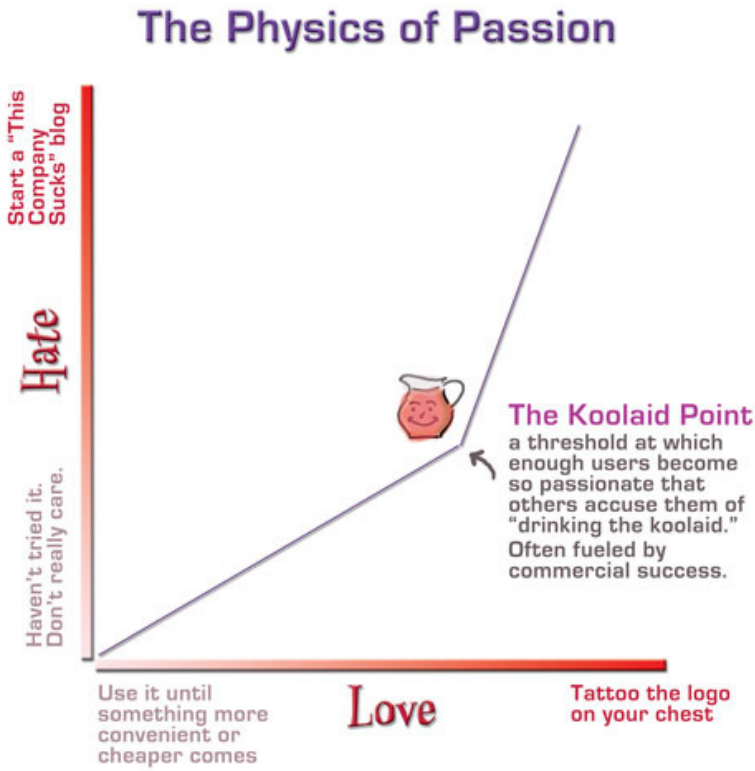
So, we have to listen but resist. The overwhelming pull of that right (hate) side slides you closer and closer to the middle. Those who hate it will hate it until you've neutered it into submission and taken away the magic some once loved.

Be brave ;)

http://headrush.typepad.com/creating_passionate_users/2006/05/dont_give_in_to.html

Popularity Breeds Contempt

By Kathy Sierra on May 11, 2006



Last year I talked about The Koolaid Point -- *the point at which enough users become passionate that others accuse them of "drinking the koolaid."* I offered no ideas for what to do when it happens other than "celebrate" and--the focus of yesterday's post--be brave. Don't give in was my main point then. But there *is* something else we can do when detractors start criticizing our *users*. Something so simple I was too thick to see it.

The tricky part is that the criticisms aren't always wrong. It really *might* be all hype. It *might be BS*. It might be just a fad, or the same s*** with a new name. But things are rarely that black and white. Where there is passion (not just fad or fashion), there is something real there. Something that *some* people see and feel. But the key point to keep in mind--and the one that offers a simple solution--is this:

People will sometimes diss things they know very little about.

They'll diss things they haven't tried.

They'll diss things based on sketchy, incorrect information--or the equivalent of urban-legend data ("My sister's friend's brother's girlfriend tried it and it was a disaster...")

Most importantly, the things people diss most passionately are often the things that challenge beliefs or ideas they're heavily invested in. They'll diss things because *embracing* those things might force them to re-examine thoughts and assumptions they care about, or because those things represent a change they don't want to make. And yes, sometimes they diss things simply because they *are* jealous, although most of us tend to dismiss critics as "just jealous" way more than we should. [Note to critics: snarky slams are far more likely to get written off as "jealousy" than specific, less emotional critiques.]

How do you deal with this challenge? The challenge faced, for example, by Seth Roberts, whose Shanri-la diet I've helped hype. The challenge our books have faced, where one prominent tech book author said--to my face--that the only reason my books were selling was because all the *smart* programmers had already learned it somewhere else. The challenge faced by Pat Parelli, whose "Parelli Natural Horsemanship" program drives intensely polarized fights in forums, where followers (or non-followers) are often outcasts at their stable, and where a Google search on parelli+cult returns over 1,000 hits.

So, I asked the guy who knows a whole lot about it--Pat Parelli. "What do you when your users are accused of being card-carrying, koolaid-drinking, Parelli cult members?"

He offers two simple suggestions we can use:

1) "Give users the tools to represent what you do accurately." (He gives his users a free-for-the-asking DVD that clearly demonstrates what the program is about.) "Don't expect--or ask--your users to defend you."

But the most important one--the simple 'doh-slapping-the-forehead' one for me--was this:

2) Ask the critics, "How long did you try it before you came to these conclusions? Because the feedback is really important to us."

With the Shangri-la diet, it was obvious how many people were slamming it without having read *all* of his research (or tried it themselves). With our books, we found early on that the strongest critics were those who had never actually *tried* to learn something from one of them--they'd never experienced it as a target-audience reader/learner would. With Parelli, 95% of their harshest critics have never actually tried it. Or perhaps someone has tried something, but in the wrong context--not in the way it's meant and designed to be used!

This "how long did you try it?" question will be met with, "I don't need to *try* something to know it's wrong." And for a ton of things, that's true. But as a sweeping statement, it doesn't hold up for most of the koolaid point, because until you've tried something or at least gotten *all* the facts, you cannot fully understand what others have found so compelling or practical or effective or engaging or productive or delightful or...

And I'm just as likely to be a koolaid accuser as koolaid drinker. I have never golfed, for example, and **I cannot imagine why anyone would spend that kind of money and time hitting a stupid little ball with a ridiculously expensive stick on grass that in Colorado costs a fortune in natural resources (water, in most cases) to keep green.** To which my co-author Bert says, "You just don't get it." ;)

http://headrush.typepad.com/creating_passionate_users/2006/05/popularity_bree.html

What if managers had to do tech support?

By Kathy Sierra on May 16, 2006

Years ago, Bert Bates worked at a software company where 95% of the 100+ employees had to spend time doing *everything*. Tech support. Customer training. Visiting clients and helping with installation and customization. Think about that.

How would a business change if...

Marketing had to spend two weeks in tech support.

Engineers had to spend two weeks doing customer training.

Managers had to spend two weeks doing testing.

Executives had to spend two weeks doing customer service.

Documentation writers had to spend two weeks teaching classes using only the manuals they'd created.

Accounting had to go on a 10-cities-in-14-days business trip and fill out their expense reports accurately, on time, and without a single missing receipt.

Sales people had to spend two weeks in a programming course.

Existing customers had to approve your marketing and advertising.

Clients attended your staff meetings.

And my personal favorite...

What if tech support and customer service reps were treated like the top sales people? What if they were sent on all-expense-paid trips based on how many customer complaints they fixed?

It's common in many industries (hospitality especially) to put everyone through cross-training that includes direct customer contact. In software development, it's rare. And according to Bert, the difference it made to that company was profound.

I've told the story before that I once attended an all-hands department meeting at Sun and asked the group of 100 to raise their hands if they'd spoken with a real customer in the last 12 months. The number of hands *not* raised was shocking, especially since a customer training facility was in the same building, and dozens of customers were on-site virtually every day. I don't blame the employees... I blame the upper management of that department for having a culture that saw customers as line items on a defects-per-million Six Sigma chart. For having a culture that made it easy to not think of customers as *actual humans* with names and needs.

[Note: The company Bert refers to was originally named Columbine, and developed broadcast scheduling software used in radio and television stations. They were eventually bought, and Bert has no idea what they're called--or doing--today.]

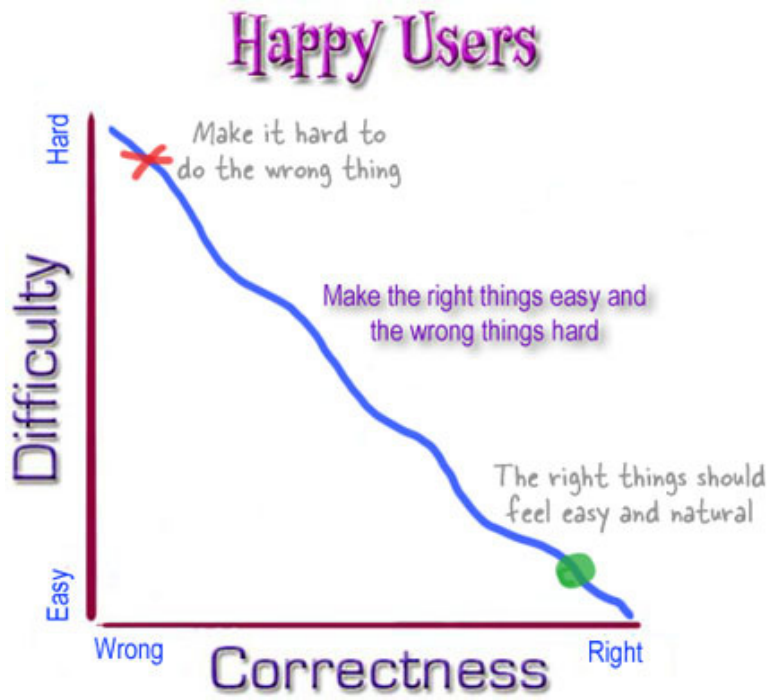
Can you think of any other "what if..." things that might apply to the tech industry?

Personally, I've had the occasional manager I thought could seriously use some time in janitorial... ;)

http://headrush.typepad.com/creating_passionate_users/2006/05/what_if_manager.html

Good usability is like "water flowing downhill"

By Kathy Sierra on May 19, 2006



"I prefer reactions in which the fabric of the organization is changed so that it's easier for people to do the "right" thing. Like water flowing downhill." Silkandspinach's Kevin Rutherford said that in a comment to David HH's post don't scar on the first cut, and I loved it on the spot.

I've talked about this many times before; my horse trainer's mantra is, "Make the right things easy and the wrong things hard"--but the *opposite* is everywhere. It's ridiculously easy for me to screw up the settings on my digital devices. The API methods that intuivitely feel right turn out to be dead wrong. I click the button I *think* will do X, and instead I get... WTF?

And sometimes, *many times*, those screw-ups are hard to undo. Sometimes, they're unrecoverable (or might as well be, since the documentation never seems to cover the most likely bad thing you'll do).

But while my earlier comments on this were mostly about usability, I hadn't thought of it as a *management* principle. (Works great with kids, too) Think about how many procedures we see in companies that feel like hacks... workarounds for a system that makes it too easy to make mistakes. And you see it from the highest levels of business right down to the duct tape someone put over the switch that you must NEVER EVER TURN OFF.

If those designing systems or software or houses or hardware or API's or policies or procedures or learning experiences or... if ~~they~~ *we* would all keep the image of water flowing downhill at the front of our minds, it might make a big difference. It might remind us just how much more elegant things could be if we made the right things easy and the wrong things hard.

It works with pets, and it works with employees (not that I would *ever* imply that employees are ever treated like... dogs). It works with kids and customers. It works with *nature*. And that's the best model of all--to make the right things seem *natural*. If a user/learner/employee couldn't imagine doing something any way other than the *right* one, they won't have to waste so much time and mental bandwidth finding and fixing mistakes.

That's not to say that *everything* should be easy and natural. But the challenges should never be in the *use* of a thing. The challenges should be in doing whatever it is the thing lets you do. The challenges are what makes the activity engaging and, in many cases, *worth it*. But the tools you use to meet those challenges should get out of your way!

Playing the game should be challenging. The interface should be brainless.

Figuring out what simulations to run for your new business idea should be challenging. Making the spreadsheet *do* it should be simple.

Defining what your code should do should be challenging. Figuring out which API methods will give you that capability should be simple.

Figuring out which music I want to buy for my perfect late-night-coding playlist should be challenging. Buying it from iTunes should be dead simple (and it is).

So, thanks for that quote Kevin.

And while we're on quotes... our blog friend Rimantas sent me another fantastic one, this time by Zed Shaw:

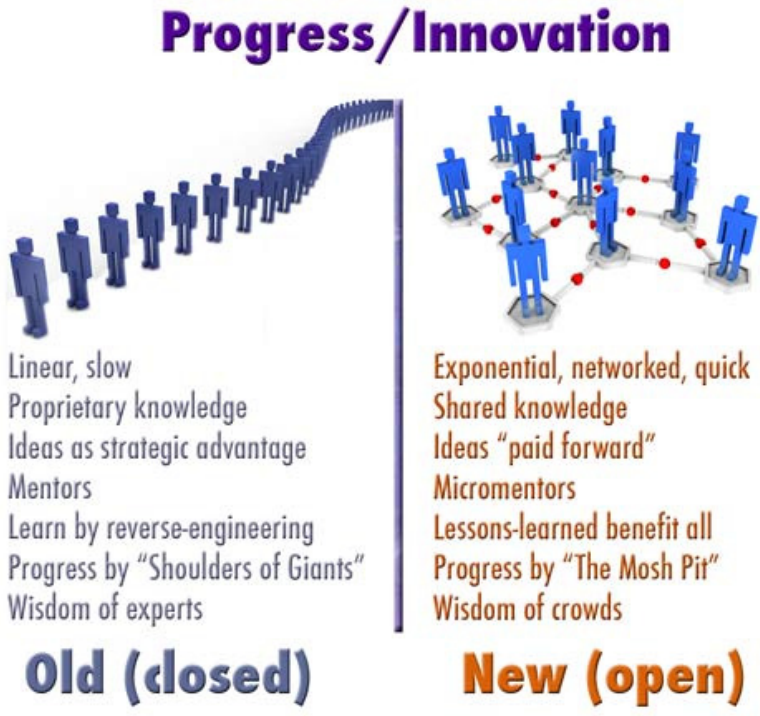
"If I KMFU (Know My F*ing Users) they won't have to RTFM."

(The quote is in an O'Reilly interview, and Rimantas got to it by way of this link from why.)

http://headrush.typepad.com/creating_passionate_users/2006/05/good_usability_.html

Mosh Pit as Innovation Model

By Kathy Sierra on June 10, 2006



"Professionals" in any field come in two flavors: Knowledge Sharers and Knowledge Hoarders. The hoarders believe in the value of their "Intellectual Property"(IP). The products of their mind must be carefully guarded lest anyone *steal their precious ideas*. But let's face it--if our *only* "strategic advantage" is our *ideas*, we're probably screwed. Or as CDBaby's Derek Sivers put in in this post:

> *"It's so funny when I hear people being so protective of ideas. (People who want me to sign an NDA to tell me the simplest idea.)*
>
> *To me, ideas are worth nothing unless executed. They are just a multiplier. Execution is worth millions."*

Yes, there are some crucial exceptions, but for most of us, **It's our *implementation*, not our *idea* that matters.** Even those who create something revolutionary are still

synthesizing... still drawing on the work of others, and making a creative leap. But even a big-ass gravel-hauling leap is still a *leap*, not a physics-violating idea that shimmered into the universe from nothin' but air.

It's how we *apply* those ideas.

How *creative* we are.

How *useful* we are.

How *brave* we are.

How *technically skilled* we are.

How we *anticipate* what our users will love.

How we *learn* from the ideas and work of others.

And from our (my co-authors and myself) perspective, **it's not about our ideas, it's about *what the ideas can do for our users*.**

Even if we *are* the only ones to have a specific new and protectable "idea" (unlikely), the moment we reveal it, everyone else will have it too. The barrier to entry today is way too low to use "intellectual property" as a main advantage. And all too often, we *think* we have a unique idea only to find that others are--independently--doing the same things.

I've found some wonderful discussions about this on other blogs (by people willing to share their ideas). The following are some snippets from recent and older posts on the topic:

Open Source Creativity from the *wonderful-go-read-it-now* Martini Shaker blog for creatives by Jeremy Fuksa:

> *"I used to work with a creative director who was (is) terribly paranoid about giving away trade secrets or any type of creative advantage to competitors. Now, if any of the things that he worried about were truly proprietary processes or special trade secrets that would be one thing: albeit very tinfoil hat-ish. But, all these "secrets" he was worried about...anything he was scared about losing control of was freely obtainable information in the first place. It just so happened that others in our area didn't obtain that information as voraciously as I do."*

> *"Case in point: I had an old colleague IM me to refresh his memory on how to add alpha channels into a Photoshop document. This CD got all freaky on me because I was "giving a competitor trade secrets and an unfair advantage."*

Whatever."

Jeremy's post pointed to another by Steve Hardy's Creative Generalist (another terrific blog). Steve's post linked to Mark Cuban's post, which talks about how Mark believes his "knowledge advantage" comes not from, say, buying, stealing, or inventing some incredibly new IP, but from relentlessly seeking out and consuming the same information that's freely "available to anyone who wanted it."

An Information Management blog, by Karl Nelson, has a post titled Open Source Knowledge that includes:

> *"A few years back a professor I had talked about the shelf-life of knowledge. His point was that information goes stale quickly, especially in the technology world. There isn't much value in keeping it locked away. The value, in the information and knowledge space, is in sharing what you know."*

And finally, Karl references one of my long-time favorites Evelyn Rodriguez. In Open Source Knowledge & Innovation, Evelyn quotes from David Maister's book, The Trusted Advisor, with:

> *"The conclusions many advisors draw are that they must be careful about giving away the store... The truth is, expertise is like love: not only is it unlimited, you destroy it only by not giving it away."*

This is not a none-of-us-is-smarter-than-all-of-us thing (which I hate). This is about each of us being smart at *different* things. Not as a "team", but as individuals with our own self-interests. If I help you, and you help him, and then he helps her, and she helps... and so on, sooner or later someone in that chain-reaction does something I benefit from directly or indirectly. It works in open-source software, where developers are practicing the idea of "code it forward", and all contributors utlimately benefit (as do the end-users of their work). Why should it be so different for many of the things-that-aren't-code?

It's also brainstorming on an impossibly large scale. And what's the worst that can happen?

A few weeks' back, I gave the closing keynote at Webstock, and I wanted to include slides (and quiz questions) on what went on during the conference. But what struck me the most during the week was how all these professionals gave away so much of their "secret sauce." How they helped their direct competitors--those fighting for the same clients and jobs--become better. In the end, I believe, everyone there recognized the benefit we *all* get in pushing the world forward, one user experience at a time.

And we'll get there a hell of a lot quicker if we stop guarding our knowledge like a jealous lover.

Our success is not about *what* we think up, but rather *who* we think about.

Issac Newton said, "If I have seen further, it is by standing on the shoulders of Giants." That was just fine in a world where knowledge doubled in half-centuries, not mere months. To make progress today, it's more like, "If I have seen further, it is by being thrown up by the mosh pit of my peers." And we all get a turn.

[Related link: Bill Kinnon on The Generous Web (he's been thinking about this a lot lately. I'm a fan.]

http://headrush.typepad.com/creating_passionate_users/2006/06/mosh_pit_as_inn.html

Changing the user experience without changing the product

By Kathy Sierra on July 8, 2006

How nice it would be to craft richer user experiences... without doing a damn thing to the product. Usually when we talk about the hi-res user experience, we say it's all about helping the user get better... usually through improved training, documentation, opportunities to practice, etc. But sometimes there's a shortcut--where a single event changes the user's experience *forever*.

That's what happened to little miss MySpace Skyler at last month's Bonnaroo, a ginormous music festival with over 100 artists ranging from Beck to Cat Power to Bela Fleck. But the high point of the featival, apparently, was Radiohead.

We have some Radiohead CD's lying around--OK Computer, Kid A, Amnesiac, I think. But they weren't anywhere near the top of Skyler's list. She was much more interested in Beck and Cat Power than Radiohead.

But that's all changed now.

More importantly-- the Kid A, OK Computer, Amnesiac CD's...*they* changed!

Because of her experience at Bonnaroo, the CDs just up and re-recorded and re-mixed themselves. Just like that. And all because of a single event. Skyler now hears layers and unusual time signatures and extra beats and... on it goes. She won't shut up about it. It is as if by magic*, the music pressed on those CDs really *did* evolve.

In the coming weeks I'll talk about *other* ways people are giving users a higher-res experience, but today is about Live Events. But does it have to be *live*? There are obviously a zillion ways to host *virtual* events. But the brain guys (and most of us feel this instinctively) say face-to-face *matters* in important ways--ways not yet replicated by technological communication, so I think we should at least consider ways in which we can connect with our users (or connect users to other users, at least) in the old analog way.

If we want to give our users a higher-res experience, some kind of live event in the Real World might be a *very* effective tool. I don't have any special ideas, just the usual:

User groups (sponsor, host, support, encourage them)

Conferences (attend a conference related to your industry, and host a special event for your users, even if it's just a demo, talk, and then party)

Very low-budget "Camps"like BarCamp (Tara has a video up about "What Is BarCamp" here--this woman knows more about them--and has *initiated* more of them than anyone).

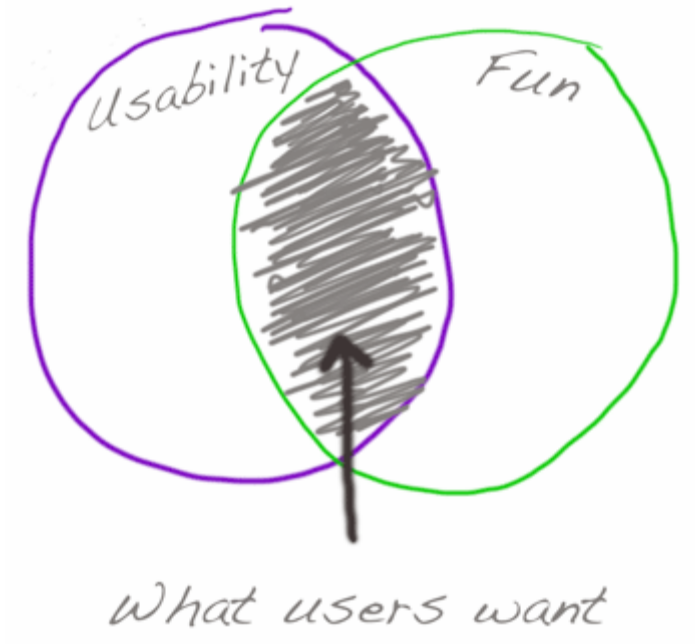
Road Trips (demos, free classes, etc.)

[* "magic" might not be that far off--I've attended enough festivals to have experienced the, ah, *contact* high.]

http://headrush.typepad.com/creating_passionate_users/2006/07/changing_the_us.html

Usability through fun

By Kathy Sierra on July 20, 2006



I've heard myself say that things *can* be both usable AND fun, but what if things might be more usable *because* they're fun? What if we started including *fun* in our specs? And I'm not talking about *games*. Can a spreadsheet be fun? A word processor? A can opener? A city government report? A church service? A technical document? A camera?

Before you start rolling your eyes, let me remind us all that **FUN** is not the same as **FUNNY**. Hugh's cartoons are *funny*. Chess is *not*. But most people who play chess still consider it *fun*. They *enjoy* it.

Funny

Fun

People are often turned off by the idea of adding "fun" to an otherwise serious product simply because they think it means "humour" or "silly." But while things which are "amusing" are often fun, things which are fun aren't necessarily amusing. The key phrase to link with fun is *enjoy doing it*. So, the kind of fun I'm talking about here includes both the chess kind (i.e. cognitive seduction) and the joke/cartoon kind.

Jakob Nielson defines usability with five components:

* **Learnability**

* **Efficiency**

* **Memorability**

* **Errors**

* **Satisfaction**

"Fun" can directly improve three, and potentially the other two as well. The key is this:

Brains reward play.

Brains like play, because play is important to survival. But how does the brain *know* that play is happening? The chemistry associated with having *fun*. If something is enjoyable, that registers in the brain, and the brain rewards us with reinforcing feelings and--more importantly--attention and memory! All things being equal, *fun* things are more memorable than things which are not enjoyable. (Remember: it need only be fun, not funny)

If something is made more memorable, more easily learned, and more sastifying... we've improved usability. What about efficiency and errors? Benefits of fun are more indirect here, but one connection is something like this:

The more fun something is, the more likely you are to keep doing it.

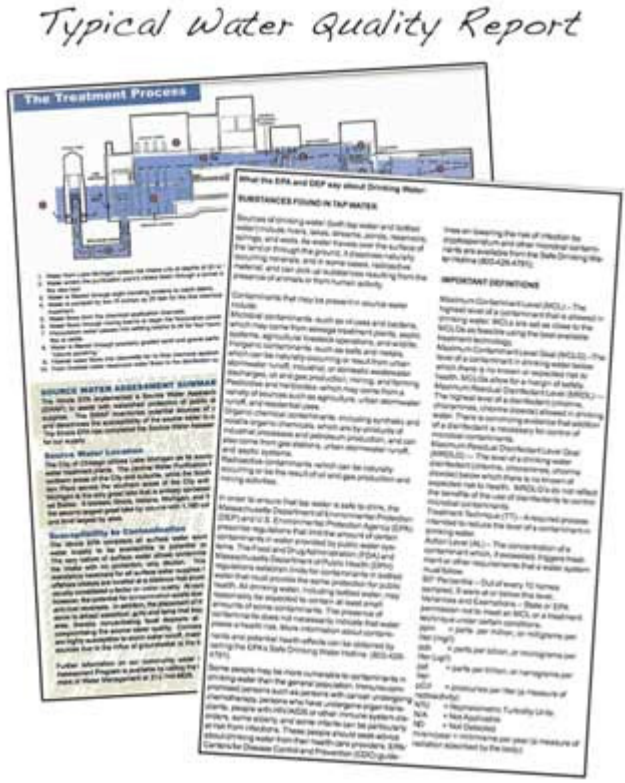
The more you do it, the better you'll get. By that logic, the more enjoyable a task is, the more likely you are to do it (i.e. practice), which often means an improvement in efficiency and error-reduction (assuming the product isn't otherwise a usability nightmare).

What got me thinking about fun today was the city of Bryan Texas Water Quality Report. US cities over a certain size are required to create them for city residents. Nobody reads them. Even if the data is made accesible and clear, it's still not *inviting* enough to get someone to take time out to read the damn thing. In other words, these reports aren't very usable.

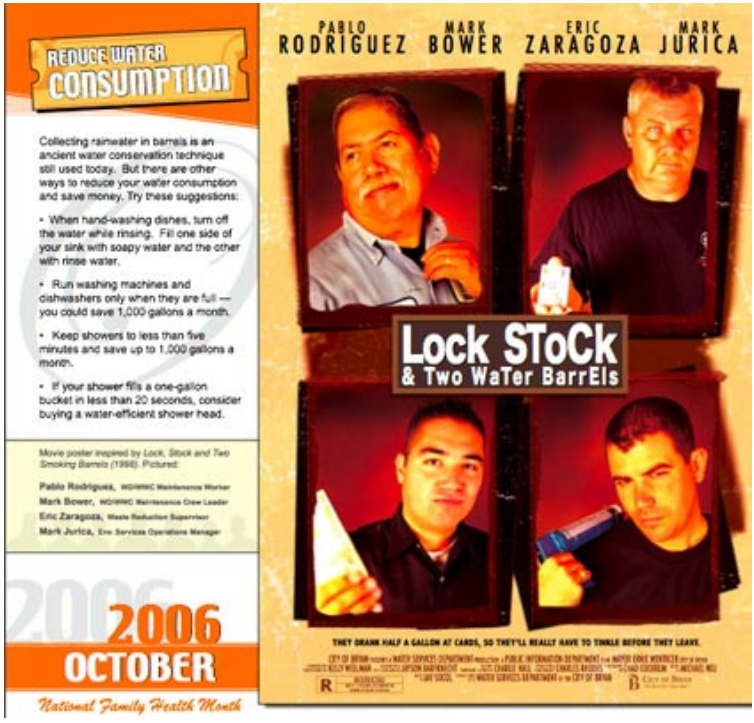
That might not be a big problem--that nobody reads them--but it's an opportunity the city has to communicate with their residents about something we (in the US) often take for granted--the city-supplied water that fills our pipes as if by magic. And cities want their residents to understand and appreciate what's really going on back/under there, and to learn more about what they should and shouldn't do to improve water usage and quality.

The city of Bryan started all this with last year's report--which I first talked about in Never Underestimate the Power of Fun. They raised the bar for a government report awfully high. But

the new one that just came out appears to top it. Keep in mind that *most* water quality reports look like this:



But the new Bryan, Texas report looks like *this*:



Jay Socol writes:

> *Before, residents threw the water report away and never read it, but now they not only read it, they look forward to receiving it... and they keep it all year! It makes people smile, laugh and believe their city government has a sense of humor.*
>
> *The older report didn't "say" anything about Bryan (or its residents), but this says we care enough to give you important info, but make it fun. (And this is also an image makeover like none other for our mayor. Look, he's having fun!)*
>
> *No one ever commented about the old report, good or bad, but today we get unbelievable amounts of unsolicited feedback from citizens, businesses, and peer cities."*

For me, one of the best parts is how Bryan, Texas made some of the "unknown heroes"--the folks who (literally) have the crappiest jobs possible--into minor celebrities. According to Jay

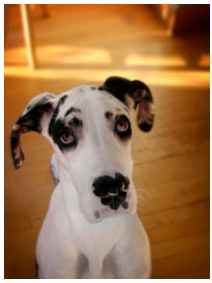
Socol, morale among these city employees has gotten quite a boost from this project. (And can you imagine someone walking down the street and saying, "Hey, you're the guy on October..." to one of the city plumbers? It's happening.)

Never underestimate the power of fun, and remember that while this calendar was actually laugh-out-loud funny (you have to read the posters... my favorite was "Flushdance"), you can still have "fun" without "funny." It's about the user's experience (i.e. cognitive seduction). And even if you aren't in a position to introduce more fun into your actual product, you can still add it to documentation and support!

http://headrush.typepad.com/creating_passionate_users/2006/07/usability_throu.html

More on blind spots

By Kathy Sierra on July 21, 2006



Just a quick public service announcement here about personal blind spots. We all have them, but the one I'm talking about in this post is the "It doesn't apply to ME" (where "ME" could be changed to "MY child", "MY job", "MY dog", etc.).

I was attacked by a Great Dane (like the one in the photo) three years ago. I was lucky--although he did serious, permanent damage to my arm (including severed nerves and many scars), he could have crushed it on a whim. And at the time, I wasn't worried about my arm--it was my *throat* that sent me into that time-slows-down mode.

But the point is not that the dog attacked me. The point is that the responsible pet owner--holding the leash at the time--never could have *imagined* that loveable, friendly, "Diego" was capable of this.

It was not provoked. I was standing there, arms at my side, silent, not making eye contact. Just standing. A minute before this happened, one of the owners got the (really big) dog out of the car and said to me, "Oh, he's friendly."

Witnesses said the dog just walked up to me and lunged. The owners--a couple who've been raising Great Danes for more than a decade--were horrified. Shocked. Stunned. How could this possibly happen? "He's never done ANYTHING like this!" I believed them. "He's the sweetest dog!" I believed them. [Witnesses later kicked around the "she's-an-alien-and-only-the-dog-knows" theory as a potential explanation.]

Until that moment, the dog's owners--and myself--were convinced that a "friendly" dog, especially on a leash, was completely safe.

But that's an illusion, especially when the dog weighs as much--or more--than the person holding the leash! And Great Danes are on the list of dogs more likely to be aggressive, including:

Bull Terrier

Cocker Spaniel

Chow Chow

Collie

Doberman Pinscher

German Shepherd

Great Dane

Pit bull

Rottweiler

Siberian Husky

But every person I know with a dog on that list would swear that it doesn't apply to *their* little Fluffy or Spike or whatever.

The reason I'm writing this is because I run on off-leash Boulder trails every morning, and today the thing I've been dreading finally happened: I came across a Great Dane. Off leash. His owner saw me cringing and said, "Oh, he's friendly." She was so certain. I froze up and could barely breathe, but she assumed that once she said the magic "he's friendly" words, I'd be fine.

Had I been able to unfreeze my face, I would have launched into a rant about how delusional this was and how could she have a dog that weighed more than *she* did and hope to control it and that oh I'd *heard* the "he's friendly" phrase before and yet look what happened to ME and on and on. What is *wrong* with people?

A little later--when I managed to start running again--I encountered a young girl on the trail.

"Oh, she's friendly", I said, when the girl warily eyed my unleashed, exuberant dog.

http://headrush.typepad.com/creating_passionate_users/2006/07/more_on_blind_s.html

Ignore the competition

By Kathy Sierra on July 23, 2006

I'm so tired of seeing so many products with the same features *that nobody wants*. It's bad enough to let feature requests from *users* get out of control, but when we start adding features just because our *competitors* have them, we're all screwed.

Why do we do it? My guesses are:

1) The Feature Arms Race. We're afraid of falling *behind* our competitors.

2) We assume that if one of the leading competitors added something, it's something users will want.

3) We assume that potential users will buy off a checklist, and we don't want to come up *short* in a side-by-side feature comparison.

4) We have a compulsive need to *add*, since the idea of an upgrade that *subtracts* features seems counterintuitive.

5) *New* features are easier to promote than *better/working versions of existing ones*. Or so we think...

What would happen if we completely, utterly, totally ignored the competition? What if we stopped thinking about competition *at all*? Perhaps if we devote *all* of our attention to users (and our own ability to innovate), we'll stop being dragged off into areas

that build our feature list, but often at the expense of users. That development time might be better spent.

It's The Feature Arms Race that leads to so much sameness among products!

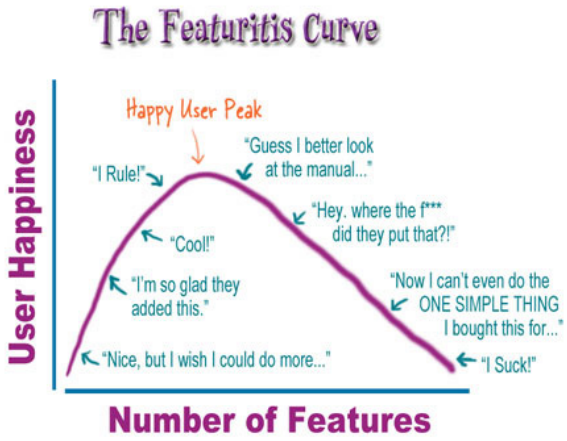
It's The Feature Arms Race that leads to the bloody kicking and clawing and fighting for market share. The Feature Arms Race is a form of group think, and we all know that design-by-committee does not produce *art*. We must wean ourselves off the obsession with the competition. If we're constantly trying to one-up them--or even just stay up with them--how does this really serve the users? How does it help the users kick ass if we're so focused making sure our *feature lists* kick ass? But it's hard to do.

"What if the competition comes up with something really good? Something users really like? "

Then you'll hear about it by staying in close contact with your user community.

"What if potential users do shop off a checklist?"

Then we should be educating them. In the absence of a deeper understanding of what's important and what we need and want, we DO often buy off a checklist--it feels like a better value to get more for our money. But of course the question is... more *what*? Certainly not usability, since the more features we add, the more danger there is of the dreaded featuritis:



If our only "competitive advantage" is by staying one step ahead of The Feature Arms Race, we're vulnerable. In my domain--

technical books--if my co-authors and I had completely given in to The Features Arms Race, we would have focused on making sure OUR Table of Contents had as much (or more) "coverage" of topics as the competing books on that topic. (Initially, that's what our editor was asking for.) But it would have come at the expense of the learner. We knew we couldn't help our learners kick ass unless we stopped trying to "cover" (and remember, what the hell does "cover" mean anyway?) the topics that would look good on a feature (ToC) comparison. Given the success of the books, we're so relieved that we resisted the pull to "compete."

I think in many cases, the more you *try* to compete, the less competitve you actually *are*.

Still, as much as I like to think I'm all about ignoring the competition, I feel (and often give in to) that pull every single day. So I'm looking for suggestions, thoughts, ideas about breaking the addiction to The Feature Arms Race.

[Note that I made this entire post without mentioning the web app company (name starts with a two-digit number less than 50) whose mission is to avoid The Feature Arms Race. But I was *thinking* about them the whole time.]

http://headrush.typepad.com/creating_passionate_users/2006/07/avoiding_the_fe.html

We can't leave innovation up to our users

By Kathy Sierra on July 31, 2006

> *"The world never needed Beethoven's Fifth Symphony until he created it. Now we could not live without it." -- Louis I. Kahn*

In this Web 2.0-ish world we're supposed to be all about the users being in control. Where the "community" drives the product. But the user community can't create *art*. (And I use "art" with a lowercase "a" as in software, books, just about anything we might design and craft.) That's up to us.

(Threadless excepted)

Our users will tell us where the pain is. Our users will drive *incremental* improvements. But the user community can't do the revolutionary innovation for us. That's up to us.

The world never needed the iPod until Apple created it. Now, look how many of us could not live without it.

[And before you snark about how we're just trying to look cool or be fashionable... no, this is about the way in which we're able to integrate music into our lives in a way that wasn't possible before. But that's for another post.]

The world never *needed* GUIs.

Or digital cameras.

Or cafe mochas.

Or skateboards.

But I have a hard time imagining my life without those things.

I can *survive* without them. But do those things give me pleasure and enhance my life in ways that I'd rather not give up? Just as Kahn says about Beethoven's Fifth? (Actually, I prefer the 7th, but whatever) Yes. Were these "needs" manipulatively planted in my brain *against my will?* I don't think so.

The point is that sometimes:

"The creation of art is not the fulfillment of a need but the creation of a need.

(this is the first part of the quote at the top of this post) -- Louis I Kahn

FYI -- I was inspired to do this by the documentary I saw last night, My Architect, a film by Kahn's son (searching for the secret to his father). I highly recommend it.

(Picture of one of Kahn's best, the Salk Institute.)

http://headrush.typepad.com/creating_passionate_users/2006/07/we_cant_leave_i.html

Declaration of (job) independence

By Kathy Sierra on August 1, 2006

If you're thinking about ditching the corporate job and heading out on your own, you can't do any better than to get help and inspiration from Pamela Slim's Escape from Cubicle Nation blog.

And to anyone who feels like a "corporate prisoner", or who has recently taken the leap and could use a gentle reminder of what this is about, I urge you to watch her little Flash movie, Declaration of Independence. It might be the most inspirational 3 minutes I've experienced in quite a while.

Pam told me her goal was, "...a simple desire to spend 3 minutes whispering something positive and encouraging in their ear."

I hear so many people underestimate/devalue/dismiss the importance of motivation. Yet so often, a lack of motivation is the only thing standing between you and something you really want to try.

Once again, I'm reminded that life is just too damn short not to go for it. Or too damn long. Take your pick ;)

http://headrush.typepad.com/creating_passionate_users/2006/08/declaration_of_.html

Are your users stuck in "P" mode?

By Kathy Sierra on August 8, 2006

How many things do you own where you can't use more than 10% of what they can actually can do? The home stereo you play CDs on but gave up on Surround Sound. The cell phone that can fry eggs, but you still can't get it to *vibrate*. The software app where half the menus might as well be Latin. So what are we doing to make sure this doesn't happen to *our* users?

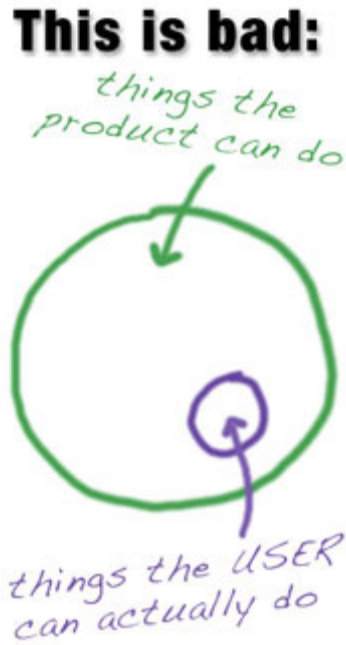
Several weeks' back I took a one-night Digital SLR class, and at the beginning the teacher asked us each to say why we were there. All 18 of us said the same thing, one after the other: "I know I have an SLR that can do so many things, but I'm still stuck in "P"--Program Mode--and I don't know how to use anything else." In other words, we were all using our pricey bazillion-megapixel cameras like point-and-shoot disposables.

Here we are with all this power and flexibility, and we can't get past AUTOMATIC. Why? It's tempting to just write it off as a usability flaw. But that's not the case with my camera--the Nikon D200 is *dead easy* to adjust. For most of us, the problem was NOT that we couldn't learn *how* to use anything but

automatic "P" mode. The problem was that we didn't know *why* or *when* to use anything else.

It wasn't simply a *camera* problem--it was a *photography* problem. The camera manuals describe precisely how to turn the dials and push the buttons, **but never tell us why we'd *want* to.** They focus on the *tool* rather than the thing the tool enables (taking pictures). What good does it do to master a tool if we haven't understood (let alone mastered) the thing we're using the tool *for*?

As we've talked about a zillion times on this blog--where there is passion, there is always a user kicking ass. If users are stuck in permanent beginner mode, and can't really do anything interesting or cool with a thing (product, service, etc.), they're not likely to become passionate. They grow bored or frustrated and then that "tool" turns to shelfware.



[Note: I'm not talking about a scenario where the green circle is just too damn big because they've added too damn many features. This is about where the user is stuck not being able to do any of the *good* stuff. Remember, this is the "passionate users" blog...]

What's *your* product or service equivalent of "P" mode?

Are your users stuck with a small purple circle of capability within a huge green circle of possibilities? We have to keep asking ourselves:

1) Are we focusing too much on the tool (e.g. *camera*) rather than the thing our users are trying to *do* with the tool (e.g. *photography*)? And by "focusing", I mean that your documentation, support, training, marketing, and possibly product design are all about the *tool* rather than whatever the tool enables.

If we want passionate users, we have to help them do something cool... fast. And "do something cool" does NOT mean, "learn to use the interface." (Keep in mind that "cool" is in the eye of the beholder... one man's "really cool pivot tables" is another man's "lame Excel tricks")

2) Is the product just too damn hard to use even if a user *does* know what they want to do with it?

3) Do we encourage/support a user community that emphasizes mastery of the thing the tool is *for*? In other words, does your product/service have the equivalent of a FlickR community... to help give users the motivation for pushing past the "P"?

4) Do we train our users to become better at the thing they use the tool *for*, in a way that helps make the need for all those other features seem obvious?

If our users are stuck in "P", they'll never get into the flow state. They'll never have that hi-resolution experience. They'll never become *passionate*.

Soooooo... let's assume we do all that--we help our users get past "P" and into the good stuff. The challenging stuff. They learn, they practice, they master the tool. *Then* what? What is the implication of a user who *does* master the tool?

On the surface, simply increasing the size of the user's purple circle relative to the product's big-ass green circle seems like the right thing to do. But is it? Is there a limit? Should there always be a little buffer zone of green just beyond the user's capabilities? And capabilities for *what*? How would *you* label the purple and green circles? Would you include the capabilities of the tool AND the potential things the tool could let you do?

I'd love to hear your thoughts about:

* why users (of some things) are so often stuck in "P"

* how this applies to things other than *tools*

* what we can do to help push users out of that little comfort/automatic zone and into the more interesting things

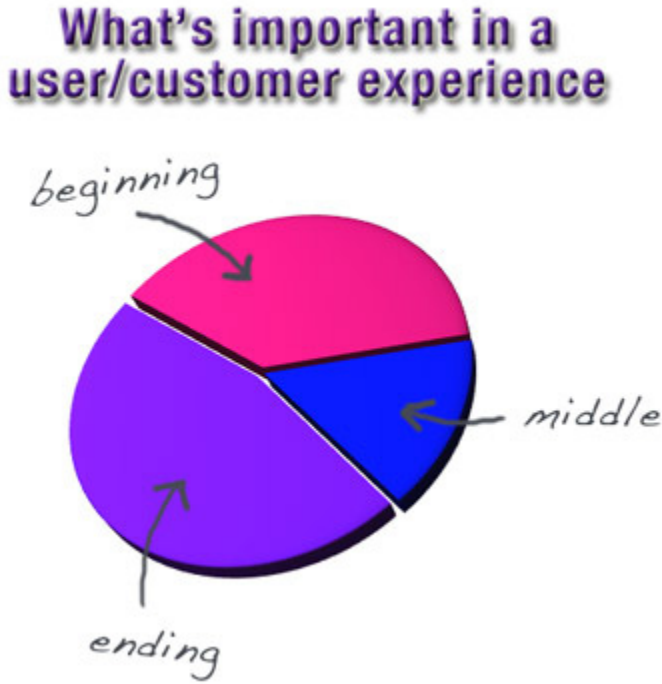
* what does it mean when the purple circle starts to fill the green circle, and how we might relabel/rethink these circles as the product and/or user capability matures

* anything else (heard any good jokes lately?)

http://headrush.typepad.com/creating_passionate_users/2006/08/are_your_users_.html

Give users a Hollywood ending

By Kathy Sierra on August 16, 2006



We can all take a lesson from filmmakers: *endings* matter. The way we end a conversation, blog post, user experience, presentation, tech support session, chapter, church service, song, whatever... is what they'll remember most. The end can matter more to users than *everything* we did before. And the *feeling* they leave with is the one they might have forever.

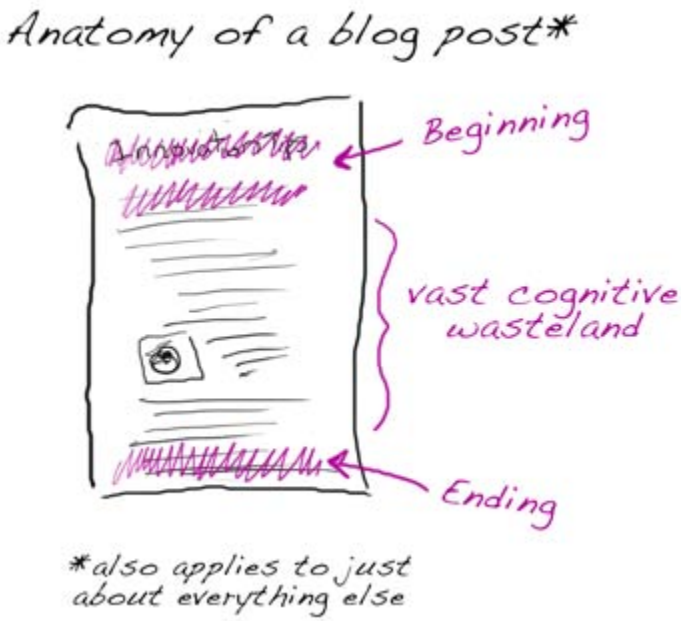
Think of all the movies where the *best* song is saved for the ending. A big chunk of "Best Original Song" Academy Award winners have been songs that played only during the closing credits. *They want you to leave the theater with the feeling that song evoked.* When a movie goes through "beta" (a test screening), the studios aren't looking for feedback on the whole damn movie...they're measuring audience reaction to the *end*. If the audience hates the ending (too sad, too absurd, too unresolved, etc.), *that's* what they reshoot.

I was reminded of the power of endings when I went to another Red Rocks concert a few weeks' back--this time it was David

Gray (with Aimme Mann and Beth Orton). Whatever you may think of David Gray's music, the guy gives good encore. They're like a whole separate show, and he leaves you feeling with a powerful, emotional, energetic, finale.

It's not just filmmakers that appreciate The End--learning theory has known this for a long time. Students in a classroom are more likely to remember what they learned/heard/did first and *last* than whatever happened in the middle. It's the Recency Effect (along with its counterpart for beginings, the Primacy Effect). Good teachers try to have more beginnings and endings by breaking up lessons into small chunks, rather than doing a single 45-minute lecture.

In fact, here's what matters in my blog posts:



From a retention and recall view, middles suck. So let's talk about endings since they're one of my personal weak spots. Even when psychology/cognitive science *tells* us that the end can matter more than the middle, it feels counterintuitive. We focus so heavily on the meaty-middle while the ending is just a tacked on afterthought. So what if we left the customer feeling frustrated and unsatisfied with our tech support as long as they know we spent a ton of time *trying*? Who cares if the presentation just... sort... of...fades...out... if the rest of it was

killer? And the ending of a chapter is just another paragraph, right?

Yes, I want to think more like a filmmaker on this. As Sacha Molitorisz put it in Now that's an ending:

"When a film resolves itself well, audiences leave satisfied and content, even if the preceding 90 minutes have been uninspiring. If, however, the climax is forced or implausible, the preceding scenes will be stripped of any poignancy. ***In other words: a terrific ending can make an excellent film a masterpiece; a dud ending can ruin an otherwise intriguing offering."***

But even if you buy into the power of the ending, the next question is, "What *kind* of ending?" Should it be a *Hollywood* ending? As opposed to, say, an *indie* finish? That depends on your definition and the circumstances, of course. There's hollywood endings and then there's **HOLLYWOOD ENDINGS.**

Not all *Hollywood* endings must be *happy*, and not all *indie* films must end in complete and utter *incomprehensability* (in that "I'm more unresolved than thou" way.) It all gets back to what we hope our users will think and feel at the end. I need to be asking the right questions about my goals, to figure out how to end:

* Do I want to help my users ***memorize*** something?

Then I should stick that at the end, or at least *repeat* it at the end.

* Do I want to help and motivate my users to ***do*** something?

Then I should end with what the sales/ad/preachers refer to as an inspiring Call To Action.

* Do I want my users to ***think*** more deeply (or more creatively) about something?

Then I should end with some things still unresolved (easy for me since I've rarely figured anything all-the-way out).

* Do I want my users to be **curious**?

Then I should end with a teaser... something that hints at what's to come, whether it's new products, new capabilities the user

will have, new and exciting ways for them to participate, etc. Leave them with a question...

* Do I want my users to ***care*** about something?

Then I should end by giving them a damn good reason... something that touches the emotional side of their brain. (Note: by "care" I'm talking about things like, "care about writing software tests" or "care about creating good user docs" or "care about the importance of endings.")

* Do I want my users to know that ***we care*** about *them*?

Then make sure the user experience has a satisfying ending, and that means *every session*. (Think of how many times you've bought something online and while the shopping part is compelling, once they've taken your credit card info you're lucky to even get a text confirmation on the screen.)

* Do I want my users to ***feel like they kick ass***?

Then I should focus less on what they think of *me* or *my product*, and more on how they'll feel about *themselves* as a result of the interaction. If they experience frustration, confusion, fear, anxiety, intimidation, and so on, that can be an "I suck" experience.

So, endings are crucial. They're what sticks. But why, then, are there so many examples of bad (or at least wimpy) endings?

What do YOU think?

Do you have any examples of good or bad endings?

[Bonus link: Top 50 Movie Endings]

Oh, and stay tuned because *soon* we're going to talk about *very cool* things to do with *beginnings*, including how to *seduce* your users into wanting more...

The End.

(or is it?)

http://headrush.typepad.com/creating_passionate_users/2006/08/give_users_a_ho.html

Geek marketing should be like a good lover

By Kathy Sierra on August 20, 2006

To the typical geek, *marketing your wares* ranks only slightly higher than *selling your soul*. It's unethical, compromising, inauthentic. ***Not Real.*** One advantage this view offers is an easy way out--we can always claim moral superiority if nobody buys/reads/uses our stuff. After all, *we* didn't "sell out" to be popular.

I used to *live* that view. But today I believe it's based on logic you could drive a FedEx truck through. And if we don't get past our marketing aversion, we may have no business whining about our lack of success. This isn't about trying to push something we know is *wrong* for users--this is about feeling comfortable (and even skilled) at helping people discover and explore the things we believe in.

The real issue is about how you define "authentic", "honest", "real", and "selling out." That's where the marketing-as-good-lover model comes in. A good lover is NOT afraid of finding out what his (or her) partner wants. A good lover does NOT view it as "selling out" if he does things simply because it's what the other person *wants*. A good lover does NOT believe it's a compromise to try to be more popular, if being popular means making things more stimulating, exciting, *sexy*, enticing, compelling, appealing, and attractive. A good lover respects that

our perception matters. A good lover respects and trusts us. ***A good lover takes a shower and puts on a clean shirt.***

In other words, maybe we should stop assuming that marketing means lying, and start treating our customers/users as people we value and care about enough to make their life a bit more enjoyable. *Even if that means little more than sexing up the packaging!* Life is short, and a good lover appreciates that a little extra attention to non-essential yet *sensual* pleasures is being *caring*, not inauthentic.

So, that's the real test of authenticity: do you genuinely *care* about the quality of your users' time and experience? Then there's nothing wrong with increasing your chances of "getting laid" (and by "getting laid", I mean, "having users find your efforts delightful").

GEEK MARKETING MYTHS

Geeks hate being marketed to

Truth: Geeks hate being *insulted*. If geeks hated being marketed to, the tech conferences wouldn't be teeming with iPods and Macs.

Geeks hate being treated as though they're too stupid to recognize when you're lying, so don't bullshit. But if you go out of your way to make something sexy, there's no reason you should be afraid to flaunt it. It's not hype if it's *true*.

Geeks are logical and rational, and don't care about superficial "sexiness". They care only about the specs

Truth: There's no such thing as a "logical and rational" human, geek or otherwise. Need proof? Throw a centerfold of Miss July in front of a geek (male OR female) and an MRI will show their brain lit up like a fireworks show. We are all human, and caring about the way something looks and feels does not mean we're superficial--it means we're *human*. We don't need to *exploit* sex to recognize that a certain amount of sexiness is both pleasurable and natural.

If the product is high quality, the packaging shouldn't matter.

Truth: For many of us, the packaging is part of the experience. Just because you're going to be naked soon doesn't mean the shirt you're wearing right NOW doesn't matter. After all, undressing you is part of the fun. Trying to be attractive to your partner does NOT mean selling out.

Sometimes, in fact, it can make all the difference. My dentist goes out of her way to make the office feel like a spa. We aren't called "patients", we're called "guests". There is no medical window in the waiting area; there is wine and espresso. The rooms where they do the work are indistinguishable from a salon. All those extras make NO difference to the technical quality of their procedures, but they sure make me enjoy it more (or at least *hate and fear it* less).

Seduction is evil

Truth: Seduction without a genuine concern for the seducee probably *is* evil, but seduction-as-part-of-a-fun-experience is one of life's great pleasures. Humans are tuned for seduction and curiosity. Of COURSE seduction can be used for evil, but so can pillows and cornflakes.

Characteristics of a good lover/marketer

DO:

Be desirable

Be appealing

Be creative

Be brave

Be thoughtful

Be attractive (but don't worry about fitting some classic definition of perfection)

Be kind and caring

Be stimulating

Be exciting

Be entertaining

Be encouraging

Be enticing

Be experimental

Be flexible and adaptable

Be playful

Be unique

DON'T:

Be dull

Be rude

Be sloppy

Be selfish/self-centered

Be arrogant

Be abrupt/impatient

Be boring (or bored!)

Be overly formal and dignified

Be exactly like everyone else

Be judgemental

Be depressing

Be ~~rigid~~ inflexible

Why does a lover go out of his way to do things for us? (besides the obvious--that he's hoping for a repeat)

Because it's *rewarding*. Full stop.

[Bonus link: John Dodds has a great little piece on Geek Marketing 101 you should check out]

http://headrush.typepad.com/creating_passionate_users/2006/08/geek_marketing_.html

Assumptions have a Sell By date

By Kathy Sierra on August 21, 2006

We can't expect to innovate new products, services, techniques, etc. without challenging our assumptions. Have some of your assumptions "gone off"? How frequently are you checking? In other words, do you have a plan in place for regularly *sniffing the milk*? I swear that half my battles at Sun were about questioning assumptions... many of which had been around long enough to be science fair projects.

When you're stuck with the inertia of outdated assumptions, you're stuck with incremental (not revolutionary) improvements. The Head First books, for example, would never have happened if we hadn't been able to convince Tim O'Reilly that typical programming books were based on an assumption that was just plain *wrong*.

We all *talk* about challenging assumptions, but what does that really mean? Because if we don't go deep enough--deep enough to get to the foundation on which all subsequent assumptions

are based--we might as well not waste our time. Here's a typical scenario:

Fred: Let's challenge our assumptions here people... are we *certain* that customers won't like this?

Jim: Yes.

Fred: How do we know? Where's the data?

Jim: It came out clearly in focus group testing.

Fred: But how *recent* were those focus groups?

Jim: Very recent--less than a year ago.

Fred: But what did they actually test?

Jim: They tested this *exact* feature.

Fred: OK, then let's move on. Tell engineering to cut that from the spec.

There's a textbook example of challenging an assumption, without challenging the assumptions *below*. The underlying, unchallenged assumption here is that *focus groups work* (when we know focus groups are notoriously unreliable for many things).

It's assumptions all the way down.

A few tips:

1) List them.

Yes, that's a "duh" statement, but seriously... how many times do you actually SEE assumptions explicitly called out?

2) Give them a Sell By date.

Slap a date on these puppies and have a system in place for knowing when to sniff them! Whether its a database or spreadsheet or just a big chart on the wall that y'all agree to review once a month or quarter or whatever, the point is to guarantee that you really WILL sniff them all on a regular basis.

3) Challenge them all the way down.

Question something and then question what it's based on, and then what *that* is based on, and so on... until you get to the bottom. And when you hit bottom, *keep questioning* until you're absolutely positively sure it's the bottom.

4) When you challenge an assumption, make it fight for its life.

Put it on trial. Force it to defend itself. Be relentless. Be skeptical. Be *brutal.*

These are all rather obvious tips, yet so often overlooked. But simply listing and challenging our assumptions on a regular basis isn't the biggest problem.

The *really* big problem is the assumptions which are so ingrained that we don't even *know* they're assumptions. They become an accepted Law of Physics, as good as gravity.

It does little good to list (and date) our assumptions, if the most crucial ones--the ones that could lead to the biggest innovations and breakthroughs--never make it to the list. It's not enough to say, "So, what are our assumptions here?" We have to ask--and keep asking--"So, what are we accepting as fact and not questioning as an assumption?" In other words, "What are our *hidden* assumptions? What do we believe *implicitly*?"

It's not enough to "sniff the milk." We have to recognize that some of the things which we believe are part of the fabric of our universe might just be milk in disguise.

And while I'm using Fundamental Laws of Physics as a metaphor for the things we believe implicitly about customers, products, etc. it seems that even the *real* laws of physics need a sniff from time to time.

UPDATE: but the universe appears to persist. Or does it? Assumptions all the way down... ;)

http://headrush.typepad.com/creating_passionate_users/2006/08/assumptions_hav.html

Why marketing should make the user manuals!

By Kathy Sierra on August 29, 2006

Why do so many companies treat *potential* users so much better than *existing* users? Think about it. The brochure is a thing of beauty, while the user manual is a thing of boredom. The brochure gets the big *budget* while the manual gets the big *index*. What if we stopped making the docs we give away for *free* SO much nicer than the ones the user *paid* for? What if instead of seducing *potential* users to *buy*, we seduced *existing* users to *learn*?

Let's take the whole damn ad/marketing budget and move it over to product manuals and support. Let's put our money where our users are. If we're in it for the short term, then sure--it makes sense to do everything to *get* a new user, while doing as little as possible once we've got them. But if we're really in it for the long haul--for customer retention and loyal users--then

shouldn't we be using all that graphic design and pro writing talent for the people we care about the most? Our users?

Most of you know our philosophy here on Creating Passionate Users:

Truly passionate users will evangelize to others.

The better users get at something, the better (higher res) the user experience.

The better the user experience, the more likely they are to keep trying to get better.

Nobody is passionate about something they completely suck at.

Helping your users learn and (ultimately) kick ass is the best way to up the odds they'll become passionate.

Creating fabulous learning materials might be a far better use of the budget than creating fabulous ads and brochures. If traditional advertising and marketing is becoming less and less effective, why not move all that talent (designers, artists, copywriters, other "creatives") from *before the sale* to *after the sale*? We keep wondering why users won't RTFM, but just *look* at our FMs! Nice brochures are printed on that coated silky paper that begs to be touched, while the manual is printed on scratchy office-grade paper. Even just that one change--making the user manual as *touchable* as the marketing material would be a good start.

And if your company insists on having fancy, slick, colorful brochures... why not take the new fancy, slick, colorful *product manuals* and use THEM as your promotional material? As a *potential* customer, I'll find your attention to user learning a lot more convincing than your attention to new sales. Rather than using your brochure to show how much YOU kick ass, I'd much rather see no-marketing-spin hard evidence of how you're going to help ME kick ass.

If the best way to help create passionate users is by helping users learn and get better, then we should put our power to entice, motivate, and inspire someone to *buy* more, and use it to entice, motivate, and inspire someone to *learn* more. In the end, those passionate users will evangelize our product or service far more credibly and honestly than we can.

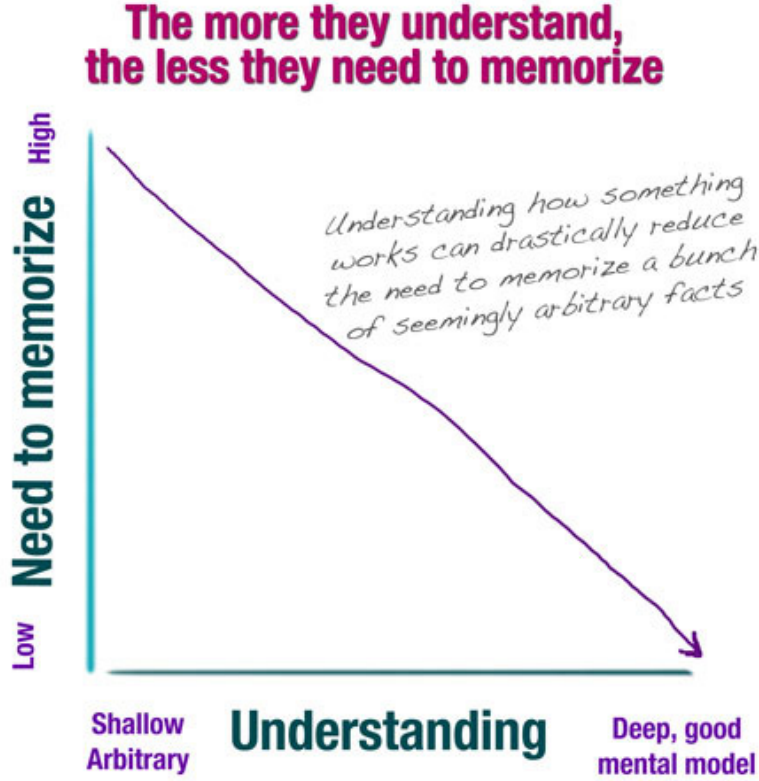
So, are you as sexy *after* the sale as you are *before*? Do you know anyone who is? (I know a few, including Electric Rain)

And stay tuned for Part Two of this post--probably tomorrow--where we'll look at how to get them to RTFM even without the big budget. And hey, I missed you guys. I was out sick for a while and then travelling for a few days. Thanks for keeping me in your feeds. ;)

http://headrush.typepad.com/creating_passionate_users/2006/08/why_marketing_s.html

How to get users to RTFM

By Kathy Sierra on September 1, 2006



The "F" in RTFM is the biggest clue that most of us blame the user for not reading the manual. But if "reducing guilt is the killer app", companies should take more responsibility for whether readers use their manuals. And since we can't *force* our users to do anything, if we want them to RTFM, we need to make a better FM.

[This post is a follow-on to two earlier posts, Are your Users Stuck in P Mode (worth reading for the very useful comments from others) and my previous Why Marketing Should Make the Manuals]

Qualifiers: this post is about products for which someone might have--or develop--a passion for whatever the product lets them DO. So, we aren't talking about, say, watches or portable radios--although these simple products still need good manuals. Also,

these tips are for those who are NOT professional tech writers. Too many times, being asked to create the manual is almost a form of punishment because the company doesn't appreciate the importance of the manual, let alone the skill (and time!) required to craft a good one.

Note: This is one long-ass post, with most of the words living in the vast cognitive wasteland known as *the middle*. If you look at the graphic and skim only the headings, you'll get most of what's in here. The rest is just support details.

In no particular order, a few tips on making a better FM:

Make Reading the Manual Unnecessary

In theory, "If your product is good enough, they shouldn't *need* a manual." In practice, that's a meaningless sentence without context. If your car radio *does* need a manual (oh, how I hate that mine does), blame the designers. But if your pro video editing software *doesn't*, then it's probably *not* a "pro" app. A complex product that needs a manual does not necessarily mean there's a design flaw.

If you *are* lucky enough to be in a position to influence a product's design, then of course you'll try to reduce the need for a manual. All the standard usability, information architecture, and user experience advice applies here (e.g. make the product intuitive, make the right things easy and the wrong things hard (to do), rely on what Don Norman refers to as "knowledge in the *world*" rather than "knowledge in the *head*", whenever possible (more on that in a minute), follow sound user interface guidelines (burn Steve Krug's ideas into your DNA), etc.)

And if *are* a product developer, please, PLEASE kick the crap out of anyone on the team who utters the phrase all tech writers fear and hate , "We'll fix that in documentation" -- a phrase that actually means, "We f'd up, but the tech writers will make up for our mistakes by putting a bunch of extra stuff in the manual to deal with it."

But... I wrote this post because most of us don't have the luxury of *modifying* (let alone designing from scratch) the products we want to help people use, so the rest of this assumes that you're pretty much stuck with what you've got.

Separate Reference from Learning

A good manual for a complex product should usually include at least FIVE *distinct* sections: Reference Guide, Tutorial, Learning/Understanding, Cookbook/Recipe, and Start Here. The single biggest problem with most less than stellar manuals is that they're usually *only* reference. But even with the best index on the planet, a reference guide suffers from one huge mother-of-an-assumption: ***that the user knows exactly what to look up!***

A reference guide, no matter how technically correct and concise and clear and all those other referencey goodness attributes, does virtually NOTHING to get me out of "P" mode if I don't even know what (or even *that*) I should be looking up.

If I have a specific thing I know I need to do, AND I know what it's called in the product, then I'm fine with the reference guide. But reference guides fail:

* If I have only a vague idea of what the "thing" is called that I want to look up.

OR

* If I don't even *know* the thing exists

OR

* If I want to get better not at the tool, but at the *thing the tool lets me do*.

(This is my problem with the Nikon D200 manual (which is actually a really good reference guide)--it helps me use the camera controls, but it's teaching me only about the camera--the thing I care least about. What I *really* want is to *take better photos*, and if the manual helped me figure out which features of the camera mapped to which cool photography things I might want to do, I'd come up the curve more quickly--which means a better chance of becoming passionate, yada, yada, yada).

1) The Reference Guide

Despite the problems with having *only* a reference guide, it's still one of the most important parts of the manual and deserves the best you can give it including a great index, a user-friendly, relevant organization scheme (make it easy for me to look things up regardless of whether I know what they're called, etc.), and clear, concise information *with visuals* whenever possible. Since

reference is where most manuals concentrate, and far more people are skilled at doing it, I'm not going to say anything else about it here.

2) The Tutorial

By "tutorial", we mean walking the user through a concrete example of using the product to do a specific task. A lot of manuals skip this in favor of giving abstract instructions for how to do something, but not all brains can learn that way. For many people, a step-by-step tutorial is the *only* thing that helps them "get it." So, even though some users will never need or read the tutorial sections, they can mean the difference between a new user who actually *uses* the product and one who never gets past opening the box.

A few tips on good tutorials:

* Choose examples that best reflect what most new users will want to do.

* Keep the cognitive overhead as low as possible by NOT using an example that requires domain-specific knowledge. For example, a tutorial on a desktop publishing program would be better using a pizza shop as the scenario rather than, say, a tax attorney. The learner should be able to focus 100% of their brain cells on doing the tutorial and not a single neuron wasted in figuring out why that particular business would need to do that particular thing...

* Include a LOT of them. Some of us learn by "triangulation".

* Watch for cliffs! The deal-killer in a tutorial is when a crucial piece--however small--is left out either inadvertently or because the author *assumed* this part of the step was so frickin' *obvious*. Of course, the level of granularity you choose for a "part" depends on who your audience is. Most software apps today should not have to explain how a mouse works, for example; a lot of pre-existing user knowledge is rightly encapsulated in the "Select FOO from the BAR menu..." But, rarely do people complain that the manual contained too *many* step-by-step details. For most of us, the pain starts when we suddenly turn the page and say, "Uh... what did I just miss? How the hell did they get THERE? My dialog box looks nothing like that..."

(If you want to know more about making good tutorials, a great place to look is at the Visual QuickStart Guides like the ones Tom and Dori write.)

3) Learning/Understanding

The more they understand, the less they have to memorize. The only way to *reduce* the time and pain of the learning curve is to *increase* learning. Too many manuals mistake reference information and memorization for *learning*, but we've all experienced the frustration of looking something up, doing it, and then forgetting everything the next time we need to do it again. The only way to really make it "stick" is by helping them 'get it' on a deeper level, where the mental model (thought bubble) in their head matches the one you were trying to communicate... the mental model that lets them extrapolate and infer and be creative about things that weren't in the tutorial.

While a tutorial is a concrete step-by-step use-case, without a deeper understanding of how and why things work the way they do, the user/learner can have trouble *adapting* what they did in the tutorial to their own unique use-cases. We've all seen users who end up bending and shoehorning their own work into something that more closely matches what the tutorial did, simply because that's the only way they know how to do it!

By including a learning and understanding section in the manual, you have the best chance to help users get out of "P mode and--most importantly--*want to*. This is where you take them from surface users who must call tech support at the first instant anything goes wrong to users who become engaged with the product and enjoy co-discovering all the ways in which the product will help them kick ass.

It is this part of the manual where the advertising, marketing, and entertainment folks have something to teach us. It is up to us to get the user/learner motivated to not just Open The Manual, but to want to actually... learn new things. Learning new things usually sucks at first, because it means we have to pass through the phase of frustration, confusion, stupidity, anger, etc. Most humans avoid learning anything that comes with a manual, and will try to do as little as possible to get the bare minimum level of competency.

This won't do if we're actively *trying* to create and inspire passionate users. We have to make this part of the manual

especially enticing, compelling, seductive, entertaining, informative, useful... all without (as many of you warned) making it *appear* like a shallow sales pitch.

This is also where it really needs to be *memorable*. It's amazing how much written documentation is meant to be remembered, but anything BUT *memorable*. Memory is tied to chemistry, and the brain pays attention and records to long-term memory that which the brain finds important. What your conscious mind wants has nothing to do with it--this is about your legacy brain recognizing that this new thing is useful for long-term survival. And let's face it--most of the content in user manuals are far, far, far, from something the brain thinks is important enough to store.

So, we have to *trick* the brain into thinking that [insert widget A into widget B and configure server C...] is just as important (or life-threatening) as the tiger licking his lips outside your cave. And who better to help us make things stand out than advertisers and marketers? Entertainers and graphic designers? Really Good Storytellers? There are a ton of things that can get past the brain's crap filter, but we have to care enough to create a manual that's *brain*-friendly, not just *user*-friendly. And the brain is sooooo much pickier about what it attends to and records.

Knowledge in the Head vs. Knowledge in the World

When we're talking about memory, we have to define what *should* be remembered, and what should exist Out There. Clearly, the more we can rely on external clues the better, and there's no reason we should have to memorize the reference steps for doing something we'll never do again, but we have to memorize *some* things. The trick is to figure out *what*. What will make their experience much better if they just *know it*?

Don Norman talks about defining the difference between Knowledge in the World (external clues) vs. Knowledge in the Head (things you memorize and "know"). Although he's talking about product design, it applies to manuals and to the relationship between the product and the manual.

4) Cookbook/Recipe

A tutorial is a specific, concrete example. Reference is, well, *reference*. Learning and understanding is the place where you start to really "get it" on a deeper level. A Cookbook/Recipe

section is where a series of steps that might otherwise live in different places in the manual are all brought together under something like a "How do I..." heading. It's kind of like a playlist; it rarely contains fundamentally new information, and simply groups a collection of otherwise separate pieces into one action/goal-based context.

For example, the Nikon D200 can do continuous high-speed shooting, but to do it right, you need to change *several* different settings on the camera including shooting mode, auto-focus, and metering. The problem is, all the different pieces you need are scattered in different places in the manual, and you have to figure out which you need only if you happen to stumble on each section and put the pieces together in your head. A simple, "How do I take high-speed shots?" or even something like, "How do I take action shots?" that described all the things you need to do... in one place... would be a HUGE help.

If page count is a concern, the cookbook section *could* simply list the different places in the manual you need to look (e.g. reference page 27, learning page 82, reference page 120, and the last half of the FOO tutorial). And although having to flip pages isn't as user-friendly, it's still way better than leaving it up to chance that the reader will put those pieces together on his own.

5) Start Here

There you are, with the freshly opened box, staring at this huge manual and wondering how long it'll take to just do the One Simple Thing you bought this thing for. A "Start Here" section, which many product manuals include, is a low-intimidation way to help the new user jump in and get *something* happening. Some early success.

The problem with many Start Here guides is that they are about configuration and set-up and very little else. There's a Grand Canyon between the Start Here and... *the rest of the manual.* Ideally, the Start Here would go beyond the initial set-up and function as a kind of mini version of the whole manual. And the Start Here makes a great sales piece for the manual itself. "Hmmmm... the Start Here guide was great, so if the rest of the manual is like that, this shouldn't be too painful. Maybe even fun..."

Of course we've all seen Start Here guides that were total bait-and-switch--they do indeed look as if they were produced by an

inspired and caring design department, but then the REAL manual looks like it was done by someone who really, really doesn't like you.

One of the best product manual goals we've ever heard is the Electric Rain "User must do something cool in 30 minues" mission. What would it mean to have a goal like that? Would it change anything about the manual?

Change the "F" in RTFM to "Fun"

Not funny, just fun. Fun as in chess. Fun as in writing elegant code. Fun as in doing something you're good at... something that lets you have a high-resolution experience. What would it mean if you asked, "How can we make the manual a fun experience?" Don't jump to the "nobody wants humour in a manual" argument--you don't need "humour" to have fun.

Change the "F" in RTFM to "Flow"

If you can keep the reader/learner/user in a flow state--where the rest of the world drops away--they'll love you. Seriously. And they'll want their friends and family to love you as much as they do. The flow experience is one of the most enjoyable and enriching times in most people's lives, and we all have the chance to help give our users just a little more of this.

There's only so much we can do if the product itself is a flow killer, but the manual can go a long way toward helping the user stay in the moment doing whatever the tool is supposed to help them do. The less they have to stop and refer to the manual (because you helped them *remember* how the thing works so that the next right step feels natural and intuitive), the more they get to keep doing the do.

Care

About the right thing. Instead of caring what the user/learner thinks about the manual or even the product, care about what the user thinks about himself. Care about how the user feels during--and as a result of--his reading the manual and learning this

thing. Care about helping the user kick ass. Care about giving the user an "I Rule!" experience, rather than the default "I Suck" experience we usually get from trying to parse a product manual.

Yes, this is a big "duh", but orientation is *everything*. When you ask someone to create a manual, be sure they know who and what it's for. Be sure they know that the goal is not simply To Accurately Document The Thing, but to Help The User Kick Ass. This one shift in perspective could change a user's world.

Resources

For learning tips, see my Crash Course in Learning Theory and the many wonderful education/learning blogs including a few of my favorites:

elearnspace

Viki Davis: Cool Cat Teacher Blog

Jay Cross: Internet Time Blog

Cognitive Daily (not technically a learning blog, but so relevant)

Judy Breck's Golden Swamp

elearningpost

Usable Help

and most definitely Darrren "I can't believe the hubris of Kathy Sierra" Barefoot, who knows a ton about this (and a zillion other things too).

Also:

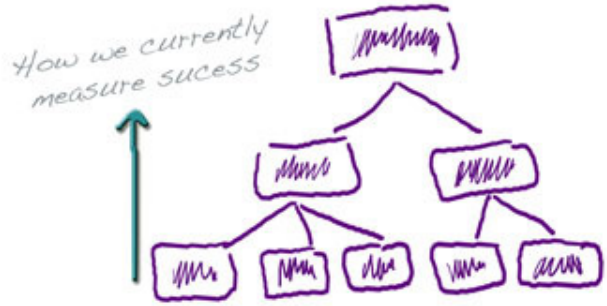
Boxes and Arrows

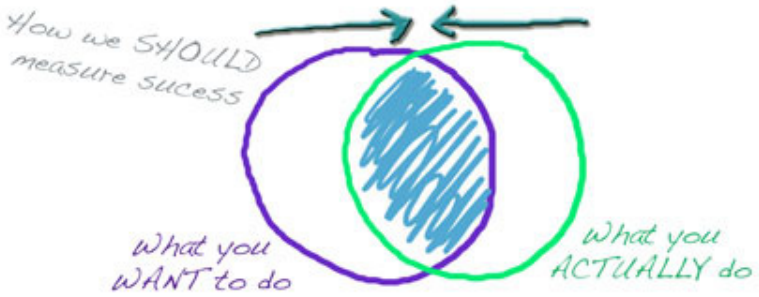
Solveig's Open Office blog (Solveig's also a pro tech writer)

http://headrush.typepad.com/creating_passionate_users/2006/09/how_to_get_user.html

"Success" should not mean "Management"

By Kathy Sierra on September 6, 2006





We might all *say* that career success should be measured by how fulfilled you are on the job, but in practice, most people and companies still measure success by how high you climbed the corporate ladder. Clawed yourself near the top of the org chart? You ARE successful. But if you're not on a leadership track, playing the "my number of direct reports is bigger than yours" game, you're probably not. We all know this is lunacy, especially in the tech world, so why do so many companies still have only a single path for promotions? You either move into management or your career (pay, benefits, perks, control, etc.) stands still.

Isn't it about time we quit measuring professional success in one dimension, *vertically*, and start considering how much your actual work matches your desired work?

And isn't it about time more companies started offering *multiple* career tracks, where management is no more valuable or important than the highly-skilled work of an *individual contributor*? (Sun is a good example of a company that offers two clear paths--one for management, and one for *individual contributors* who'd rather bathe cats than be a boss.)

What happens when a company gives an employee no option for growth other than management? Yes, lots of individual contributors (even programmers) *want* the challenge of a management role, but some of the best feel *forced* into trading the work they love best for more "advancement opportunities". How senseless is it to take a star programmer and make her do Gantt charts? How lame it is to take your best designer and make him run budget meetings, review TPS reports, and consolidate time sheets?

This post was partly inspired by Anne 2.0's Where Are The Women Redux, which (among other things) talks about conferences that claim they can't find enough women speakers because their aren't enough women in those top leadership roles. Anne makes a fabulous point with:

"It's no surprise that you might find more "smart" women speakers elsewhere than in the upper reaches of large tech and media companies. Part of being smart is weighing your options and making tradeoffs. Women face a radically different opportunity landscape than do men. I'm not going to say one or the other landscape is better-they're different. But if you care about having more women as speakers at your tech conference, you might have to go with someone other than a senior level executive or dealmaker type."

So, yes, I'm thinking that we should wean ourselves from evaluating professional success on management level (even if it's within a company *we* started and own). Rather than asking about someone's rank, position, job title, number of direct reports, power, etc. we should focus on one simple question: how closely does the work you do match the work you *want* to do? We should start thinking of ways to make sure that kick-ass individual contributors can be compensated just as well as managers, so that they aren't torn between getting a promotion that sucks (into management) or sticking with what they're good at and love, for less pay.

[And yes, I realize that this is all way over-simplified with tons of big, tricky issues including the whole ugly mess about how some jobs (engineers) are considered so much more valuable than others (teachers), etc. But even if I were smart enough to take that all on (I'm not), we can't do it all in one blog.]

We should start thinking in Venn diagrams instead of hierarchical org charts. But I want to know what *you* think--I've seen this from only one side--as an individual contributor who'd rather program in punch cards than do an Employee Performance Review.

http://headrush.typepad.com/creating_passionate_users/2006/09/success_should_.html

Why "duh"... isn't.

By Kathy Sierra on September 7, 2006

Critics of this blog love to say, "Duh!" or "Thanks for stating the obvious." My response is, "While the idea is dead obvious--the problem is that we don't *do* the obvious." When I hear comments like, "You wasted all that space to say, "Care about your customers", I wonder why we *don't*. Or rather, I wonder why we all *say* we care about them, yet our actions reflect a more selfish view. When it comes to our users/customers...

I don't think they think what we think they think.

It's similar to all those other statistics you hear about, like that way more than 50% of the population rate themselves "Above Average" in everything from looks to smarts. We think our customers generally love us, although of course we're not *perfect*, but then... who is? Sure we have a few issues, but we're working on it. And besides, we're *so* much better than the competition.

When we first came out with the Head First books, and talked about brain-friendly learning principles, people said, "Duh. There's nothing new here." And we said, "Of course not. We didn't *invent* anything. We just *applied* it. And if implementing these principles were truly "duh" (which they should be), then

everyone would be doing some variation of it, and readers/learners would not be struggling to learn tough technical topics.

If *helping your users kick ass* were truly "duh", then our users wouldn't feel frustrated, confused, angry, stupid, humiliated, or furious. If writing good user manuals were truly "duh", then there'd be no acronym for RTFM.

This is no different from any other part of our lives, of course. Eating healthy is a "duh." Exercising five times a week is a "duh." Saving money is a "duh." Keeping our kids off TV is a "duh." Flossing is definitely "duh." Managing stress is a "duh." Greeting your significant other and kids with a smile and full attention is a "duh." Empowering our employees is a "duh." Changing the oil is a "duh." Being on time is a "duh." And I might as well end this paragraph with a totally lame cliche:

There's a big difference between saying, "Eat an apple a day" and actually *eating the apple*.

If "duh" is so damn obvious, why aren't we DOING it? (I say "we" because I'm just as guilty) More importantly, why do we drastically overestimate the extent to which we *are* doing "duh" things?

There are too many reasons to list, and many I hope you'll add, but a few highlights include:

Downplaying the importance

Denial (we think we *are*)

Inertia

Fear of change

Too risky

If the competition isn't doing it, why should we?

Ego (making a change means admitting you weren't doing something right)

etc....

But I think the most important one is that we never actually take the time to *really* think about the "duh" thing. I try to ask people, "Sure, taking care of the customer yada yada yada is "duh", but what would it actually mean if you really REALLY did

it? Stop. Think. Deeply. How much of what you do might *feel* like it's for the customer... or you tell yourself that story, anyway... but it's more about what's good for *you*? What would it mean if you took the "duh" thing and spent one hour--just ONE hour--brainstorming what that *really* means?

When people ask for the secret sauce guaranteed recipe for success, we say that it's quite simple: just *do* the "duh" thing. The Big Secret is not about knowing what magical thing to do--it's about taking the "duh" things seriously enough and actually *doing them*. If you could pick just one "duh" thing to work on, what would it be?

And yes, this post is one big "duh." A "meta-duh", if you will. ;)

What are *your* thoughts?

[Update: In comments to this recent post on Tara's blog, Martin Wells said something similar:

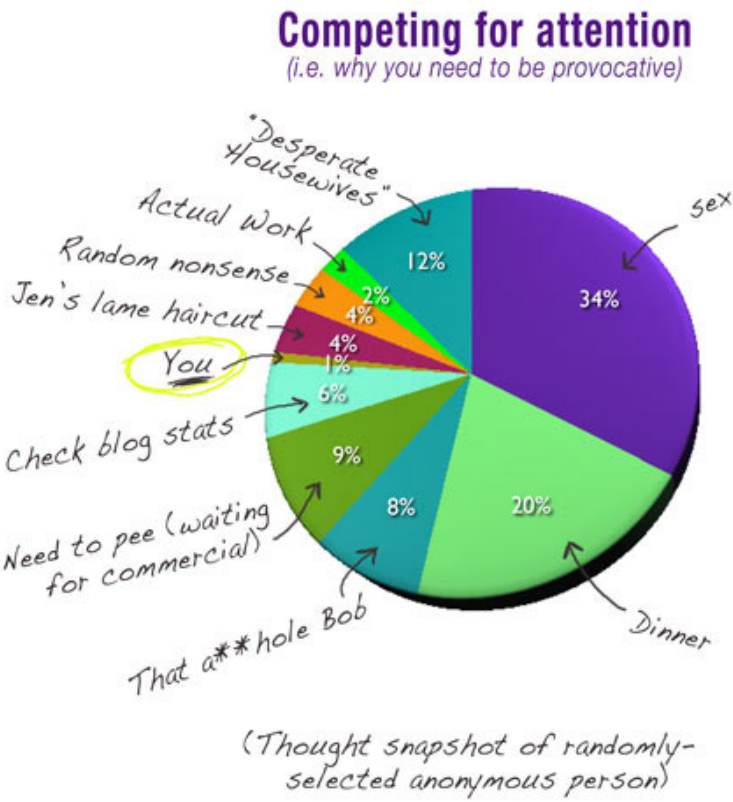
"And readers continue to buy into the idea that if they can just somehow find the right formula -- the "secret" -- they'll succeed.

The irony is most of the books are right, it's just a matter of applying all that knowledge correctly and intelligently."]

http://headrush.typepad.com/creating_passionate_users/2006/09/why_duh_isnt.html

Be Provocative

By Kathy Sierra on September 13, 2006



When you want to get--and especially *keep*--someone's attention, what's your competition? What *else* could they choose to focus on at any given moment? The belief that we have 100% conscious control over what we pay attention to is a myth. The belief that users can and will *choose* to pay attention to our message/ad/docs/product/lesson, etc. is a mistake. So what *can* we do to up the odds of getting and keeping attention?

I just returned from two weekends of intense horse/human training, including the annual Parelli conference, and you'll just have to suffer through several posts in which I map *everything* into some all-I-needed-to-know-I-learned-from-my-horse principle. Starting with *this* post. At the first clinic, master trainer David Lichman said of our horse-human relationships:

"The secret is to be more provocative and interesting than anything else in their environment."

If we want our *users* (members, guests, students, potential customers, kids, co-workers, etc.) to pay attention, *we have to be provocative*. We can moan all we want about how the *responsible* person should pay attention to what's *important* rather than what's *compelling*. But it's not about responsibility or maturity. It's not even about *interest*. It's about the brain.

Remember, the *brain* and the conscious *mind* don't always see neuron-to-neuron. The *brain* pays attention to survival of the species. *No matter what the mind wants!* If you want the *mind's* attention, you can't ignore the *brain*. In other words, you can't assume that users will pay attention to what you say *even when they're genuinely interested*. Unless, that is, you throw a bone to the brain as well. Or *trick* it.

So this isn't about having to *bribe* people into paying attention by sexing things up with graphics, sound, or shock. This is about helping the mind *and* the brain agree on what's worth paying attention to. And if you want it to be *you*, then you better be the most provocative and interesting thing in their environment.

With horses, there's not as much competition. There's no HorseBox 360 or PonyMail. No Horse 2.0. No PonyMeme. Yet it's still a battle to be more compelling than the grass, the wind, a plastic bag, other horses (*especially*), playing the whoever-moves-their-feet-first-loses game with me, etc. And as smart and complex as my [fabulous Icelandic] horses are, they're still way easier to interest than a human.

Being Provocative

Provocation is in the eye of the provoked, obviously, so there's no clear formula. But there's plenty we can try, depending on the circumstances, including:

* Be Visual

Pictures are more important to the brain than words, and unless you've already got their attention *and* are a good enough writer to *paint pictures in their head*, you'll do better with visuals. The more stimulating the better. Even graphs and charts are a huge help... it doesn't have to be pictures of naked women (although that would work, of course. Just try to get past a rack of men's

magazines (without the "protective covers") without at least a glance. Your brain can't help it, so let yourself off the hook ;)

*** Be Different--Break Patterns and Expectations**

As long as we're doing what everyone else is doing (or what *we* have always done), the brain can relax and think, "Nothing *new* here... whew... what a relief, that means I can now go back to scanning for something that *is*". Ways to be different include doing the opposite of what you normally do, or doing something expected in a different domain, but which is wildly unique in *yours*.

*** Be Daring**

You know the story on this one--being *safe* is often incompatible with being *provocative*.

*** Change Things Regularly**

This is about *continually* breaking your *own* patterns. Consistently shaking things up whether it's look and feel of your website to the product itself. (Obviously the definition of "regularly" and "things" varies dramatically depending on the type of product or service. MySpace can change daily to the delight of its core audience, while a financial app better keep its UI stable for a much longer time and find something *else* to change regularly (like the website, tutorial style, or online forums).

*** Inspire Curiosity**

Humans often find puzzles and even *questions* irresistible. Just *try* to walk by a TV playing a quiz show and *not* think about the answer to the question you heard walking by. How many times have you watched to the end of a movie you didn't particularly like, just because you *had* to find out how the story ends? Our legacy brains love curiosity because it usually means more learning. (FYI - my horse finds orange traffic cones irresistible)

*** Pose a Challenge**

The level and nature of the challenge work only if they're within boundaries that work for your audience, of course. Ask me to solve a calculus problem and I'll keep on walking. Ask my co-author *Bert*, and he'll find it impossible to do anything else *but* work on it.

* **Be Controversial and Committed**

Take a stand. Mediocrity is not a formula for holding attention.

* **Be Fun**

Remember, brains *love* fun because fun=play, and play=practicing-to-survive. (And as we've said many times here, *fun* does not have to mean *funny*. Chess can be fun but isn't funny. *Except when I play*.)

* **Be Stimulating. Be Exciting. Be *Seductive***

Keep in mind that seduction does not have to mean *sexual*. A good storyteller can seduce me into sticking with the story. A good teacher can seduce me into learning. A good software app can seduce me into getting better and better.

* **Help them have Hi-Res Experiences**

This gets back to the notion of being-better-is-better. The more your users know and can do, the higher resolution experience they have. Whatever you can do to give them more expertise will help keep them interested in wanting to know and do *more*. But they need to be up the skill curve a ways before this really kicks in, so we must do whatever we can to help get new users past the rough spots (i.e. the "suck threshold").

* **???**

Your turn. What are your ideas for how we (or you) can be more provocative? Who's doing a good job? Who is *not*?

(Note: I'm currently in the middle of a difficult multi-country work trip in Europe, so I'm having a tough time getting online. I apologize for not responding to your comments here recently, but they're HUGELY motivating for me, so... thanks :)

http://headrush.typepad.com/creating_passionate_users/2006/09/be_provocative.html

Screaming users considered good

By Dan Russell on September 18, 2006

We all know our users can have really strong opinions about our stuff. And we all know we need to listen carefully.

My very favorite user comment of all time came from a young woman who had been using this cool design tool (something I'd built) for about 12 weeks. At the end of the project, I was doing the standard debriefing of the users, asking what they liked and didn't like, what worked, what didn't work, what was frustrating: the usual sort of post mortem on a project.

She told me that my software, my baby, the thing I'd been working on for the past 2 years was *"..the most white male fascist tool I've ever had the misfortune to use:"*

I was somewhat taken aback.

"Ah, yes:" I stalled for time, desperately trying to hold it together. "And what made you feel this way?"

The conversation went on for some time after that (as you can imagine). And while it was a painful episode, it was a really valuable learning experience.

Although I knew intellectually that not everyone would see my system as I did, I was floored by her reaction. But it made the point: as a designer, you really have to be aware of other folks opinions (even when they don't jibe with yours), and you have to know what it is you're building. Sometimes, your product is going to passionately piss people off. Sometimes, that's okay. In many cases, you simply can't design a product that will make everyone sing your praises and want to send you roses. I love my iPod, but I know there are some people who think it's devil spawn. If Steve can't get everyone to love his things, I'm not sure I can.

So I'd succeeded in creating a passionate user. Sadly, it was passion in the wrong direction.

After I recovered my composure a few weeks later, I realized I was really glad she'd told me. The ten users I'd interviewed before her were all pretty nice and even-keeled. "Oh yeah: it worked well:" or even the sweetly positive comment "I could do things with it that I could never have done before."

But in retrospect, I didn't learn much from the nice folks who told me everything was fine and ducky. I did learn a great deal from the ones who struggled, my users that just didn't get it, had really strong reactions or failures.

As Henry Petroski writes in *To Engineer Is Human: The Role of Failure in Successful Design*, we learn more from our failures than our successes. But only if we pay attention to the failures and figure out what to do right the next time.

The trick is to figure out what the message is from the user. I did have the presence of mind to ask her what "fascist" meant. Sure, I know the dictionary definition, but I couldn't figure out what it meant in the context of the tool I'd built.

My question opened up the sluice gates and I heard an awful lot about "not letting the user have a choice" and how our design tool "forced the user to do things in a particular order."

Gee. We did it that way because we knew it was more efficient. But provably correct didn't win the heart and mind of this user-she did things in a different way, and the tool was forcing her to go along a different path. It felt fascist to her.

Okay. Got it. So it wasn't the Gestapo of all software, but it really was at variance with her approach. In an instant it

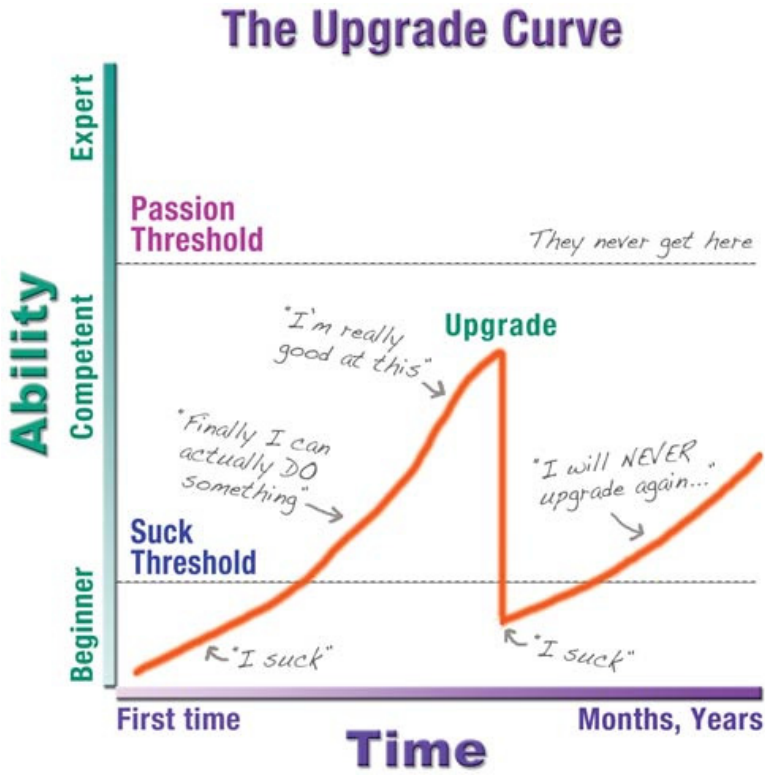
became clear what we could do differently the next time around.

Bottom line: Every product evolves. It's the rare (or trivial) that gets it right the first time and sticks with it for the rest of time. Listen to the screamers and whiners and people writing nasty blog posts. It's painful and tough, but worth it. The screamers may not know it, but they're really helping you out with the next release.

http://headrush.typepad.com/creating_passionate_users/2006/09/screaming_users.html

Why they don't upgrade (and what to do about it)

By Kathy Sierra on September 22, 2006



Why is it that--after we bust our ass to produce a shiny new version of our product--users are so slow to upgrade? WE know it's better. WE know it'll help them kick ass in new ways. WE know that if they stick with their current version, they'll never truly become passionate...because they'll never touch that high level of expertise where things get *really really interesting*. But there our users sit, apparently content to hang out in the "competent" zone, happy they no longer suck, but unmotivated to push forward.

That's a problem.

And I'm not talking about the *financial* side. Even if we make no money off our upgrades, we still want our users learning and growing and improving and reaching for new challenges and

doing more complex, cool things. (Assuming you ultimately want *passionate* users, which if you're reading this blog...)

So why are users dragging their feet? Why aren't they desperate to get the latest and greatest spanky new release? Conventional wisdom says it's because of the expense, or that users fear change, or that users are simply too lazy. But there's a simpler explanation:

People don't upgrade because they don't want to move back into the "Suck Zone."

They worked too damn hard to reach a level of competence and the thought of sinking back down--however briefly--into that awful state they clawed their way out of--is too unpleasant. We've trained users to *fear* upgrades. Raise your hand if you've ever installed an upgrade only to find yourself back in that confused I-have-no-frickin'-clue-where-they-put-that-dialog-box state? Raise your hand if you felt the upgrade just wasn't worth it, *even though you knew that the way you did things in the current version was pretty much an inefficient hack*. Raise your hand if you felt intimidated and maybe even a bit humiliated that after upgrading you could no longer do some of the simplest things.

It's not usually the *upgrade* that sucks. It's that the upgrade makes the *users* suck. Or at least makes them *feel* that it's their fault for not instantly getting it.

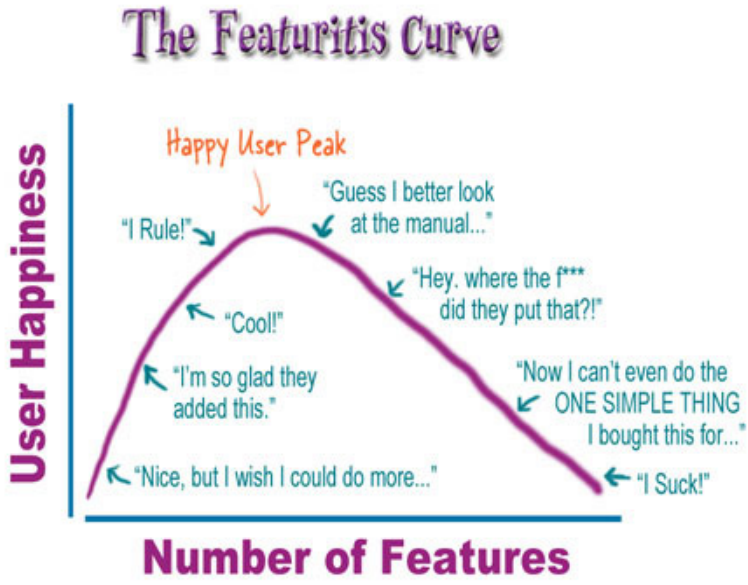
Bottom line: *nobody likes doing things they suck at. If there's a way to avoid it, we will.*

Back in the late 90's, I attended a Macromedia conference, and one of the sessions was a panel of web pioneers discussing what were then the earliest days of web development, especially the whole browser incompatability problem (that we of course thought would be LONG gone by *now*... lol). The panel host asked one simple question of each of the panelists, "So, which browser do you have on your machine right now?" The response was shocking. Almost every panelist--and keep in mind that these were hard-core web developers/entrepreneurs--gave the same response, "Whatever was installed on my machine at the time I got it." One of those panelists was none other than Jerry Yang, co-founder of Yahoo! We were stunned. If even *Jerry Yang* doesn't bother with upgrades...

(Many of us later confessed that we would have answered the same way.)

How to inspire users to upgrade

Don't give in to featuritis



Make the upgrade *worth it*.

More importantly...

Make sure the users KNOW it's worth it.

Provide a compelling benefit, and do your best job of painting that compelling picture for the users.

Go over the top with documentation

Geez... I hate it when I get an upgrade and it comes with a whopping 1-page ReadMe. Make sure users know you're going to hold their hand and walk them through the new things in the friendliest, most accessible, most encouraging way.

Try not to break things that were previously important to them

Yeah, another "duh" thing, but so often ignored. Users should feel like the new upgrade simply *adds* capability, performance,

etc. without sending them back to the "suck zone." In other words, they should feel like the upgrade is an *extension* not a radical *modification*. This isn't always possible for forward progress, of course, and you don't want to be locked in to your former design mistakes, etc. but at least think about ways to help a user transition gracefully from one version to another.

Don't tell me what cool things *YOU* did to the new version, tell me what cool things *I* can do with the new version.

Never, ever forget that it's *all about me*. For most products, and most users, they don't give a duck about your new specs. They care about what it means to *them*. Connect the dots for them in the most vivid, compelling, motivating way.

The pain of an upgrade begins with download and installation

Even if the new version itself is natural and easy to get used to, if the install and set-up is a pain in the ass, they'll remember that the next time (and tell their friends not to bother unless it's REALLY REALLY worth it).

Don't make me pay for YOUR bug fixes

The more users perceive your upgrade as simply correcting things you should have had working in the first place (bugs, performance problems, etc.), the more likely they are to start taking hostages if you expect them to *pay* for the privilege of having what they *thought* they were paying for with the previous rev. It's OK to make a performance/bug-fix release, but don't charge for it unless you've done something earth-shattering to the technology which gives you a huge increase in performance (as opposed to correcting poor performance).

Seed the community early

Get beta versions to your key community of users so that *they* can start evangelizing why the new version is worth it. (Of course, this assumes that the new version IS worth it.)

Set the tone for *future* upgrades

If you lie about the upgrade--either by downplaying the learning curve or overselling the benefits, you're screwed.

Users will remember the pain of THIS upgrade when it comes time for the NEXT one.

The better the first upgrade experience is for them, the more likely they'll be to ever do it again.

Try making more frequent, smaller/incremental upgrades

While this does't work for most non-software products, continuously "refreshing" and modifying the product in tiny ways adds up to big changes down the road without those huge jump-off-the-cliff slides back to the "suck zone". The ultra-fast release cycles of many of the Web 2.0 companies is an example (and of course ANY web app has a potential advantage here since the user doesn't need to *choose* to upgrade).

Entice, bribe, or potentially *force* them to upgrade

This is extremely dangerous, but if you are absolutely certain that your upgrade will be universally loved by users--and that the upgrade will be relatively bloodless--you could potentially hold them hostage, like the way Apple did recently with the new iTunes. If you want to download the new shows at the new hi-resolution, you have no choice but to upgrade/install the new version of iTunes. Again, very few of us will ever have *Apple* loyalty, but there are scenarios where you might just have to say, "Sorry, but there is no way we can--in good conscience--let you continue without this upgrade." This approach will likely backfire spectacularly if the upgrade is not *free*.

Start the buzz early (practice T-Shirt-First Development)

By the time Apple releases a new version of Mac OSX, the Faithful are so excited that they line up by the *thousands* outside Apple stores at midnight, braving the cold, just to get the new OS a full 24 hours ahead of their friends. How do I know? I've done it, twice. Once when it was snowing.

New releases can be a source of great enthusiasm and energy. Exploit that.

In the right situations, upgrades are like crack. (In a good way)

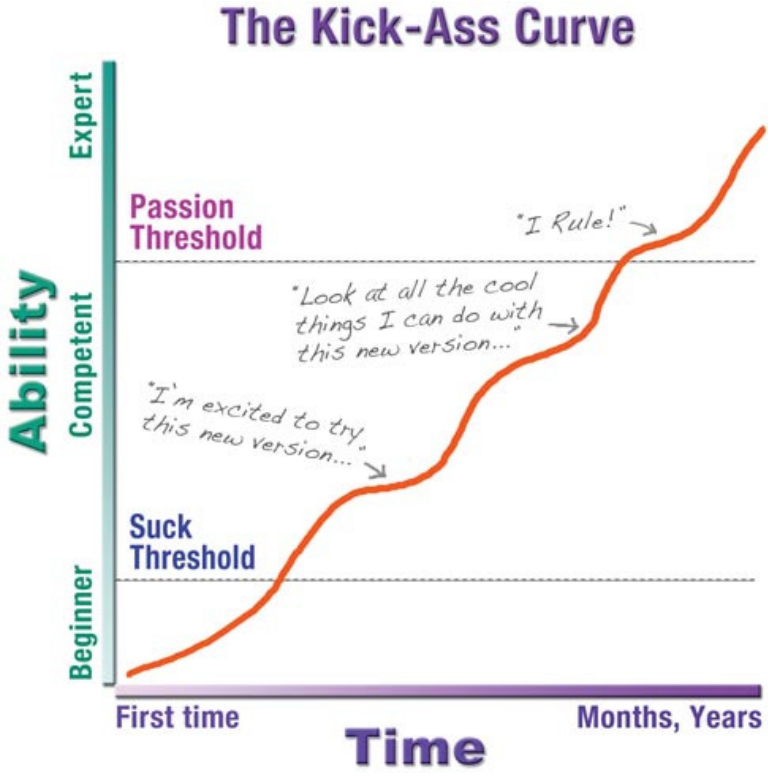
Remember, *reducing guilt is the killer app*. Nobody wants to go back to the "suck zone", so it's your job to make sure that:

A) The new upgrade must not send them back to the Suck Zone

and

B) You must convince *users* that they won't land back in the Suck Zone

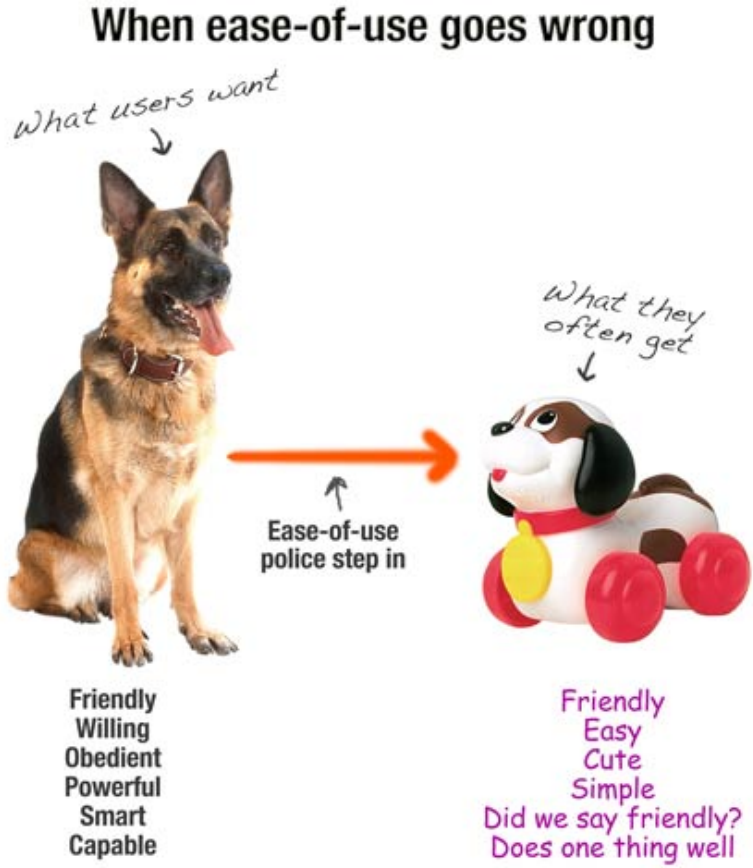
In the ideal world, the curve looks like this:



http://headrush.typepad.com/creating_passionate_users/2006/09/why_they_dont_u.html

Ease-of-use should not mean neuter-the-software

By Kathy Sierra on September 25, 2006

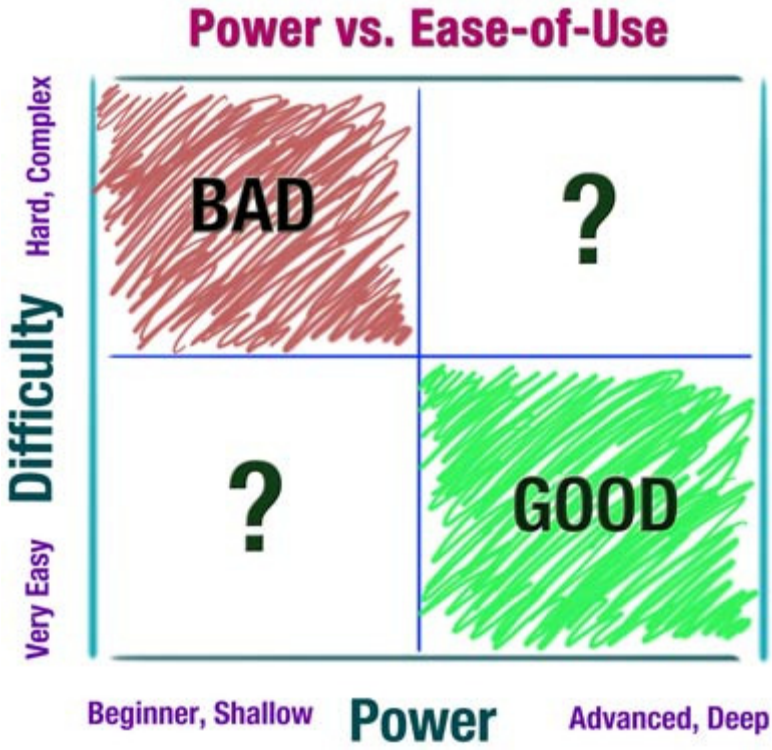


Is our heart in the right place but our execution flawed when we *neuter* a product in the name of newbie-friendliness? In the push to make programs "so simple even your [mom/kid/dog] could use it", there are a lot of dumb products out there. Or rather, dumbed-*down* products. It's like we're throwing the *power* baby out with the *poor UI* bathwater. But if we want passionate users, ease-of-use should NOT be the Big Design Goal. **Good usability is the enabler for what we (users) *really* want--*more superpowers*.**

We want to *do* things. Cooler things. Advanced things. More creative things. We don't want to be better at *using the tool*, we

want to be better at doing *whatever it is the tool supports!* Usually when we talk about this it's-not-about-being-better-at-the-tool thing, we're coming from the perspective of what and how we *teach* our users. This post, however, is about the software, product, web site, service, itself.

Take a look at this chart, and ask yourself how you'd describe the two boxes with question marks. We know the bottom right quadrant is awesome, and in the top left, there be dragons. But what of the top right? What of the bottom left? Think about it for a moment before you continue (or before you, as most of you will do, skip to the next graphic ;)



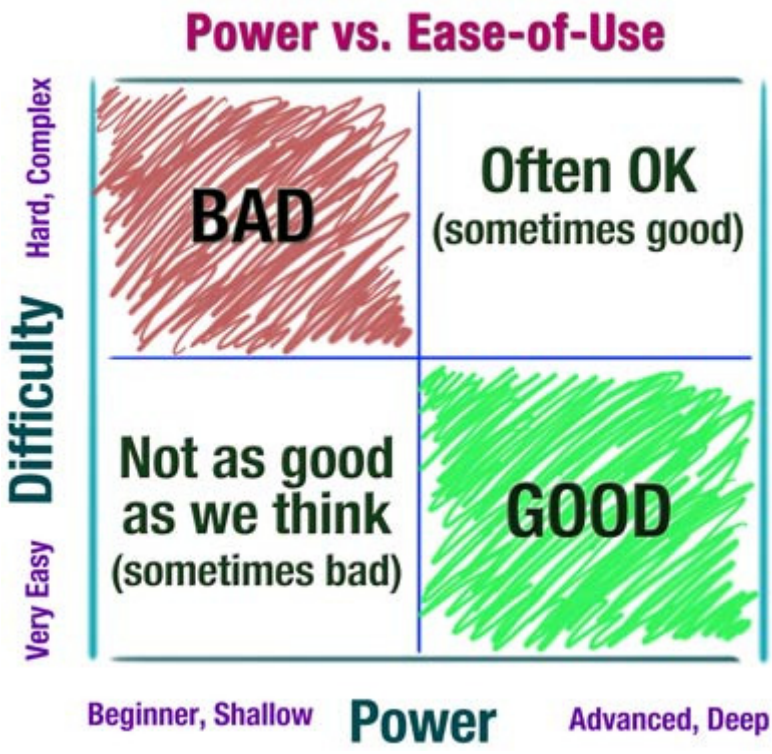
It's great that so many are putting the "user" back in "user interface", but using a brain-dead-simple tool does NOT necessarily mean an "I Rule!" experience. Maybe we need to spend more time thinking about *providing superpowers* and a little less time on *simplifying*. The last thing we want is to build the Tic-Tac-Toe equivalent of software, when the user ultimately wants to play Chess. (Note: I said *"when*" the user wants to play Chess. Sometimes the ultra-beginner-only product is exactly

what's needed, and might be extremely successful, but it does mean that users will outgrow it before they ever become *passionate*. This post is about the products that do NOT want to be newbie-*only*.)

Without challenge, there can be no growth and no flow state. And wherever you find real passion, you *always* find challenges. Alan Kay once said something like, "We do a great job of helping people practice being beginners. We help them get really *good* at being beginners. What we *need* are ways to help people start at an *intermediate* level so they can start doing something rewarding right away."

Of *course* usability is absolutely crucial, and it's a prereq for pretty much everything. A UI that gets in the way of the thing the user is trying to do is a deal-killer (or at least a *flow*-killer). But there's a difference between "Good UI" and "Ease-of-use"! If I'm doing something complex, by choice, then focusing on making it *easier* might not be the right move. Assuming the power is there, the main goal should be to keep the UI the hell out of the way of what I'm trying to do. You may not have made it technically an *easier* product to use, but you've made it a product that supports more time in flow, doing the thing I want to do (which is NOT "use the software", but rather "edit video" or "write a letter" or "mix audio").

Perhaps I need a qualifier for the word "easy", because while the thing I'm doing with the tool may be quite difficult, it's true that I want the how-I-communicate-with-the-program to be as easy as possible. I do want it to be extremely easy to figure out how to tell the software what to do, but I do NOT want the software to pat me on the head and say, "don't worry your little head... I'll take care of it all for you with this nicey-wicey wizard and this fuzzy-wuzzy dialog box and all the helpful things I can do for you like capitalize words (since you're too stupid and lazy to do it yourself)" . And I do NOT want the software to simply strip out all the functionality that's too complex to simplify. So I guess there's at least two different forms of easy: easy-as-in-*natural-usability* and easy-as-in-*dumbed-down*.



But what about Featuritis?

"Featuritis" comes from adding horizontal (broad) capabilities, rather than adding vertical (deep) capabilities. Rather than add 25 new ways to do the same shallow things, add 5 new *advanced* capabilities. I don't want to do more *things*, I want to *be more advanced*.

Photoshop, for example, would be adding horizontal features if they added new painting tools, or yet more ways to configure your tool bars, etc. But adding new capability to their color correction and RAW tools, for example, would be adding vertical features. One leads to featuritis, the other leads to more powerful users.

Featuritis is not so much about feature *quantity*... it's about feature *shallowness*. By all means, please give me more features. But they must be the *right* features, and to figure out "right", we have to know our users, we have to narrow down the domain in which they use our products (e.g. is Photoshop about

photography or digital painting?) and make some assumptions about their goals, motivation, and background.

1) Adding power through different products or product *editions*

When we add features, they should be the next natural things an advanced user wants to do, but the new superpowers don't necessarily need to be in the same product... Apple provides a dumbed-down (but still wonderful) *free* music making tool in Garageband. It's extremely easy to get started (which is great), and ridiculously simple to use, but even a non-musician who really gets into Garageband starts to bump into Garageband's limitations pretty quickly. And as if by magic, Apple's non-free product Logic Express just coincidentally happens to have the features you find yourself wanting after you've started to max out Garageband. (And the same thing repeats when you bump into Logic Express limitations, there's always the much more expensive full-featured Logic. Apple uses the same thing 3-tier/first-one-is-free-crack-model with their video editing tools as well).

2) Adding power through user-created extensions

Many excellent, successful products take the approach of, "Sure, we could keep piling more and more features on after listening to all the requests, but each added feature would just annoy everyone except the one person who asked for that particular feature... so instead, we'll let YOU add new features."

IBM once had an Expert System tool called TIRS, and when users wanted to do more, rather than continuing to add to their API, they opened up the system so that you could embed your own C functions in a rule. Allowing plug-ins, extensions, macros, third-party modules, etc. is (sometimes) a great way to add far more vertical/deep *power* than you could ever come up with when YOU are both the developer and, well, *the decider* of what should be there.

3) Adding power through "advanced modes"

Yeah, yeah, yeah I know that this is controversial, and I have virtually no credibility in developing a product with different user modes, but... it can work.

4) Adding power through a willingness to sacrifice newbies

We can't be all things to all people. The cliche of "catering to the lowest common denominator" is one we all consider negative, yet we still do it. It can be a form of greediness! If we can only do one thing well, then we have to choose carefully. And if can't find a clear way to add advanced capability while hiding the complexity from the newbie, then some of us must choose to leave the newbies to someone else. (Yes, there are a million implicit qualifiers, conditions, disclaimers, exceptions, etc.)

Being the one who nails it for the newbies is a very successful strategy for a lot of products and services. But again, once those newbies leave the nest, they have to look elsewhere. And the better we are at inspiring enthusiasm and loyalty in beginners, the more they want to continue with *us* ("I trusted you this far, and you really came through... now help me get to the next level").

5) Adding power by truly knowing (and not underestimating) your users

Recently someone showed me a very early alpha of a start-up web app he's working on. The target audience is "regular people" (i.e. not programmers or other hard-core geeks). I was horrified when he showed me what amounted to a command-line style of input. Command-line?! But then he reminded me that most people who aren't right in front of their computers *write things down*. Most adults are not just comfortable but *experts* at writing on paper. When I write someone's phone number on a piece of paper, I don't have a separate little piece of paper for [first name] [last name] [area code] [phone] [cellphone] and on and on. I just write the damn thing down as a string of characters. So... why *not* allow (or even encourage) users to simply type things in rather than forcing them to go through complex wizards and dialog boxes.

This particular example is fraught with UI landmines including the problem of users needing to memorize the exact order and syntax they have to type things in. And yes, that *could* render the whole thing virtually unusable. But if the software had some very intelligent, clever parsing and could just look at the text and figure it out (or at least make a high-probability smart guess), then you've got a way to enable a ton of power without having to add endless dialog boxes and windows and choices and other get-in-the-way features.

The point is, that while it looked DRASTICALLY counter-intuitive to me to have "regular users" essentially work in a command-line interface (fortunately it *looked* and felt like a simple text edit window as opposed to, say, a DOS prompt), when the guy showing it to me framed it as, "Writing things as text is the most natural thing there is" I felt my brain shift a little. And then I began to consider all the software that babies us, insults us, and ultimately gets in our way... something Jason talked about when he delivered Opening Remarks at SXSW (classic example: "I can capitalize words mySELF thank you very much...").

[Here's a podcast/mp3 recording of the talk. Incidentally, *I* am the ~~seared sh**less~~ lucky one delivering the Opening Remarks for the *next* SXSW Interactive in March. I'm hoping Jason will give me some tips; he gave a wonderful talk]

So... as always, now it's your turn. This is a tricky topic, and everyone has something to add to the story. Are too many of us dumbing down our products? What are other ways we can help our beginners without leaving them stuck in that lower left quadrant? Remember, an "I Rule!" doesn't come from *success*... it comes from doing something challenging.

http://headrush.typepad.com/creating_passionate_users/2006/09/easeofuse_shoun.html

Motivating others: why "it's good for you" doesn't work

By Kathy Sierra on September 29, 2006

"What matters is what they do when the clicking stops." That was the central theme in the New Media Interaction Design courses I taught at UCLA Extension (Entertainment Studies dept). We all want to motivate our users (customers, learners, kids, employees, members, etc.), but motivate them for *what?* What do we hope they'll *do* when they stop clicking/listening/reading? More importantly, *how do we make it happen?*

Question 1: What do we want our users to *do*?

And no, we don't get to say, "know more." That's not an action. "Like us more" is not an action. Even my favorite, "kick ass" is not an action. How many people take a course in Design Patterns and then go right back to work and write the same clunky code, reinventing the flat tire? How many customers interact with a web app and then... just leave? How many people say they care deeply about a cause, but do nothing beyond bumper-sticker activism? How many people listen to a lecture on the dangers of smoking, but *keep smoking?*

There is nearly *always* an action (or set of actions) you're hoping users will take, and most of you already know what that is. But we also know that this sometimes involves a change in *behavior*, something that's *extremely* hard to do. So it's really the *next* question that matters more:

Question 2: *How* do we motivate them to do it?

That's where broccoli and optimism come in (I promise I'll get there in a moment).

We all know we can't simply slap motivation on another person. All we can do is **design an experience to help them motivate *themselves***. If we get them to spend time on our web site, and they have a good experience, but then leave without doing anything--and never come back--does it really *matter* that they had a Good User Experience? Is a good experience an end in itself, or is it a means to something else? For much of what we design, what matters is what happens when the clicking stops (or for many web apps, just *before* the clicking stops).

So, we really have *two* levels of motivation... motivation to *interact* and motivation to do something as a result of that interaction. Motivation to *interact* is something we've talked about quite a bit here... things like the flow state, levels/superpowers, spiral experience design, painting a compelling picture with clear steps to getting there, blah blah blah. *This* post is about inspiring post-interaction *action*.

And it all comes back to broccoli. And optimism. The main points are:

1) Trying to motivate someone to action by telling them it's *good* for them doesn't... actually... work.

There's way too much statistical evidence (not that any of us need more evidence than our own personal experience), that not only is "... because it's good for you" NOT motivating, even the extreme case of, "... because you will DIE if you don't..." often fails! Smoking, weight loss, lack of exercise, too much alcohol or drugs. We all know what is and isn't "good for us," yet too many of us still aren't motivated enough to DO something about it. So we must ask ourselves:

"If people don't aren't motivated to make changes even under the threat of *death*, what on earth *will* motivate them?" In a controversial but *powerful* article in Fast Company (from May 2005) called "Change or Die", there are some insights and examples. You need to read the whole thing for the full context, but this quote gives a strong hint:

> *"The conventional wisdom says that crisis is a powerful motivator for change. But severe heart disease is among the most serious of personal crises, and it doesn't motivate -- at least not nearly enough. Nor does giving people accurate analyses and factual information about their situations. What works? Why, in general, is change so incredibly difficult for people? What is it about how our brains are wired that resists change so tenaciously? Why do we fight even what we know to be in our own vital interests?*
>
> *Kotter has hit on a crucial insight. "Behavior change happens mostly by speaking to people's feelings," he says. "This is true even in organizations that are very focused on analysis and quantitative measurement, even among people who think of themselves as smart in an MBA sense. In highly successful change efforts, people find ways to help others see the problems or solutions in ways that influence emotions, not just thought."*
>
> *Unfortunately, that kind of emotional persuasion isn't taught in business schools, and it doesn't come naturally to the technocrats who run things -- the engineers, scientists, lawyers, doctors, accountants, and managers who pride themselves on disciplined, analytical thinking. There's compelling science behind the psychology of change -- it draws on discoveries from emerging fields such as cognitive science, linguistics, and neuroscience -- but its insights and techniques often seem paradoxical or irrational."*

Or to put it another way, **telling you to eat broccoli because it's good for you doesn't work because it doesn't invoke the right feelings.**

And even the threat of *death* doesn't invoke the right feelings. (Not that fear isn't a powerful motivator, but it's not motivating in the ways we might think...)

Which brings us back to, what *does* motivate?
Optimism. Hope.

In the Fast Company article, they talk about **reframing / recasting** the reasons why you should do something. Rather than using "it's good for you" or even the hard-to-believe-it-doesn't-work "you'll DIE if you DON'T," some health-related programs have much more success by emphasizing *pleasure*. From a doctor in the article: "joy is a more powerful motivator than fear."

Yes, this whole "duh" post is to reinforce the cliche: ***focus on the positive.*** (And if you're wondering why an article on making health changes is in a business magazine, you'll have to read the whole thing to see how they apply it to work behavior and culture as well, especially in the area of *change*.)

But what prompted me to dig out that old article was the most *recent* Fast Company article, Moving Pictures, about the Oscar-nominated entrepreneur Jeff Skoll, the man behind Participant Productions--"the first film company to be founded on a mission of social impact through storytelling." Skoll is also the guy who made Al Gore's new film, An Inconvenient Truth happen.

Skoll recognizes that simply "raising awareness" of issues is of little value unless people *take action*. From the article:

"For each project, Participant execs with nonprofit backgrounds reach out to public-sector partners, from the ACLU to the Sierra Club, for their opinions. If those partners don't think they can build an effective action campaign around the film, it's a no-go..."It can't be good-for-you spinach, or it's not going to work."

[I used broccoli instead of spinach because that whole recent Killer Spinach thing in the US wrecked the metaphor]

And here's the optimism part:

> *"In the face of challenges ranging from global warming to threats to civil liberties, Skoll aims to inspire hope, then action. "Time and time again, you see this outpouring from people once they're made aware they*

> *can do something," he says. "That's the principle that drives this company."*

And even if you're *not* trying to get someone to take action for social change or to save their life--something Meaningful with a capital "M"--remember that meaningful with a *lowercase* "m" matters too. If your software, book, or service helps me learn more, spend more time in flow, kick ass a bit more at work, or even just have fun playing a game...you're bringing a bit more joy into my day. And THAT is meaningful to *me*.

http://headrush.typepad.com/creating_passionate_users/2006/09/motivating_othe.html

Aerons and Air Hockey... dot com excess or essential tools?

By Kathy Sierra on October 2, 2006

The Aeron chair. Air hockey. Espresso machines. Hip, urban design with a touch of MOMA. In the pre-bubble dot coms, *aesthetics* mattered. Having a *fun* workspace mattered. Having the best toys (including workstations, ginormous monitors, etc.) mattered. But when the dot coms went south they took the chairs and the toys and the stimulating workspace with them. We want them back. We *need* them back. You can keep the lame (and by "lame" I mean "WTF were they thinking?") business models, thank-you, but bring back our Aerons!

The thing is, we all *expect* and understand why *designers* have--and need--creative work spaces, yet we somehow think programmers (or just about any other role that's not considered one of the "creatives") *don't*. We act as if programmers don't *care* about their environment. But you don't need to know an Eames from an Eero to appreciate the impact your environment has on your energy, creativity, productivity, and *happiness*.

Way before the dot com days, I spent several years in Los Angeles working at design/creative shops, often as the sole programmer in a sea of artists. The first thing I noticed when I started working at these places was how *good* it felt to be in a place where the aesthetics were taken very seriously. Lighting, walls, materials, colors, floors, layouts, offices-with-doors (you can't be creative without some alone time!). I swear I wrote better code in those environments.

Then I started working at game companies, where it was expected that everyone's workspace would be knee-deep in toys--light sabers, life-size Capt. Kirk standees, Lego masterpieces, vintage robots and other sci-fi kitsch, and of course--Nerf weapons. Once I learned to duck at the right moment, I swear I wrote better code in those environments.

Then I went to work at Sun. Not the engineering part, in California, but the huge new Colorado campus. And while you were certainly free to dress up your cube any way you liked, and the coffee was pretty good, this was NOT the Sun that I'd heard about. No weird MIT-style pranks where someone's car is reassembled in their office over lunch. No, it was more like Office Space out here. Not that it wasn't light years better than a lot of--or most--tech companies, but the Colorado campus just didn't have the geek/festive mojo I'd expected.

But then it got worse... *I started working from home*. It took me a long time to realize that it wasn't so much the other *people* that were missing, it was the stimulating work environment. I tried coffee shops and considered shared office spaces where other self-employed or work-from-home people can have some of the benefits of an office, but I actually *prefer* to work alone. It's not the people I miss... it's being in an environment that makes me feel creative and energetic. I want a space that matches my enthusiasm.

All that changed when I learned that Dori Smith had rented a 1957 Airstream office. I'd lusted after Airstreams for years, and when I went to visit her, I knew it was exactly what I'd been looking for.

Finally, after two years of looking (and saving), I found and bought a vintage 1966 (recently restored) 23-foot Silver Streak trailer. (Silver Streak is a "fork" of the original Airstream.) This is my new baby, with my dog Clover in the doorway:

And it's perfect. It's parked exactly two feet away from the side of the house (a house I share with my horse trainer and his wife), and the wifi from the house works beautifully. I haven't felt this good working in years. During my search I found a variety of people who use vintage trailers as their work studios, all equally thrilled. But I also discovered an incredibly passionate vintage travel-trailer community, especially over on Tincantourists.

I don't have an Aeron, but I do get to work at a retro dinette with the original formica ;)

Don't underestimate the importance of your work environment, and don't be quick to consider things like Aerons and office aesthetics and toys *wasteful*. It's just the opposite. Apparently Joel Spolsky agrees.

A few other relevant links on the importance of a *playful* environment:

Nial Kennedy on incentives.

Joel's Field Guide To Developers

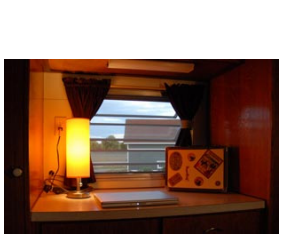
Metropolis article

37signals on the importance of not being interrupted (I'll say more on the danger of distractions--and the need for plenty of 'alone time'-- in another post)

And my earlier post on the science of how dull environments hurt your brain.

Don't for a moment think that the aesthetics and stimulation of your work environment don't matter! So, what have been some of *your* most stimulating work environments? And if you work from home, what are you doing to make it inspirational?

And here are a few more pictures of my new office than you really want to see:






http://headrush.typepad.com/creating_passionate_users/2006/10/post.html

"Oops... we forgot about the users."

By Kathy Sierra on October 3, 2006

If I attend one more tech company meeting where NOBODY talks about the users/customers (or at least not a *positive* one), I'm throwing myself out the window of this trailer. Think long and hard right now about some of the company meetings you've attended where the entire effen meeting is about everything *but* what's good for the users. We talk about deliverables and budgets and TPS reports and why we all need to help keep the refridgerator clean and how "upper management" has a new policy and why filling out those timesheets *really helps the company* and how we didn't make our numbers last quarter and how *somebody* is taking more than their share on Bagel Morning Wednesdays and, oh yeah, don't forget the team-building workshop next Tuesday.

Gag me with a motivational poster.

Until talking about the users/customers/members/clients becomes **the most important thing**, we're going nowhere good. And no matter how many companies pay it lip service, the meetings tell the real story. It's staggering how many meetings I've been to where nobody is advocating for the users. Nobody. Yet *everybody* is advocating for ways to do what "upper management" wants or ways to save money or ways to... you know, many of you probably work in the companies I'm thinking of.

That doesn't mean that most of *you* don't think and care about the customer, it's just that you don't always get the opportunity to *talk about it*. It's hard being the first one to stand up in a meeting and say, "Um, excuse me, but we just made a decision that hurts the customers a *lot*, and... EVERYONE'S OK WITH THAT?" And who wants to be the one who stands up and asks, "And this helps the customers... how?" or the one who says, "Do you think we could all take a look at the customer feedback reports?" or the one who says, "Has anyone actually TALKED to a real user lately?"

Caring what the users think is something that just about any company *claims* to do, but even when they say it, what they really mean is, "Of COURSE we care what the customers think of *us*" when they *should* care about what the customer thinks of

himself in relation to the product. It's the biggest companies that usually are the worst, since there are so many layers upon layers of mid-to-upper managers who are so far removed from the real users that they've got a distorted, naive, unrealistic, or just plain *wrong* idea of who their users are and what their users need and want.

But I hear it outside the big companies as well. It's the author who is writing the book to help his resume or gain visibility, so he focuses on what readers will think... of the *author*. It's the blogger who complains about not having the readers they deserve, but not *once* acknowledges that what the *readers* actually *value* (and decide is worth their time to look at) is what matters (assuming more readers is the goal--for most bloggers, it isn't). We make excuses. We blame everyone and everything but ourselves when it's so often *not* about the things we point to (ad budget is too small, marketing sucks, not enough funding, the A-list won't link to us, we're the wrong gender/race/age, etc.), but rather the simple fact that we just don't have enough respect for the people who are our users/readers/customers/members...

So, to help reinforce the message, I've made a couple of take-offs on the Buzzword/BS Bingo game (here's a marketing bingo card), where you go to a meeting and check off each BS/buzzword as it comes up. But since that's only reinforcing what's *wrong* with many of these meetings, I thought I'd make one that reinforces what's *right*... talking about the users. In this game, you check off a box when someone says something loosely related.

Better yet, randomly pick one of the boxes on the card and just *say it*, every 5 minutes ;)

CLUEFUL BINGO:

How does this help users kick ass?	Customers need this	Because it's the right thing to do	Our customer support totally SUCKS	We need a better user experience
I don't care what the competition is doing	It's not about us... it's about how customers feel	Customers don't want to hear how hard it is for us to do this	Who are we helping by doing this?	This policy hurts the customers... let's ditch it
Our manual sucks too	If it's bad for the customer, don't do it	How does this help users kick ass?	Where is the "pain" for our customers?	Our users are not stupid
Let's not forget who pays for all this	We should be more open and honest with users	Hurting customers for short-term profit is a lame idea	Are we sure this isn't featuritis?	How can we make this upgrade less painful?
Do this help the user stay in flow?	I don't care what upper mgmt says if it's good for our users	If we're doing it ONLY for the money, we're screwed	Reducing user guilt is the killer app	How does this help users kick ass?

And here's another bingo game, although like Buzzword Bingo, it is focused on what's *wrong* (but I still like it). It's a card that brings up a variety of ways we deflect responsibility for a product-that-does-not-respect-the-users by blaming everything *but* the fact that we just didn't take the users seriously enough. That we just didn't value their time, money, energy enough. There are, of course, scenarios in which things on the blame card are valid, and not simply excuses, but you can usually tell when someone (or some company) is either in denial or looking for a scapegoat.

Blame Bingo

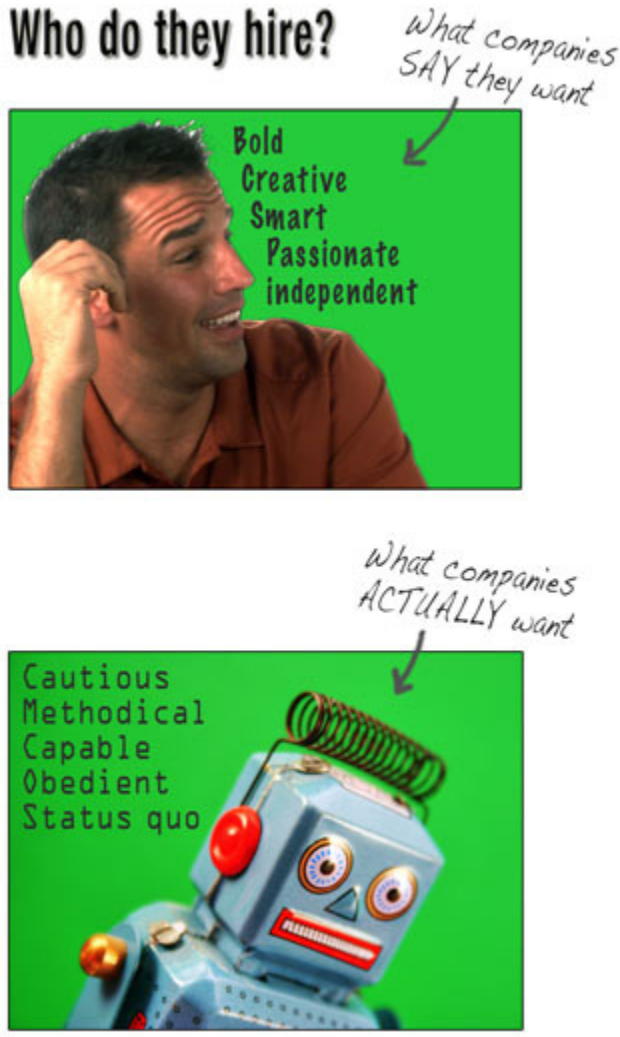
Our users won't RTFM	Our marketing sucks	We don't have enough flashy features	The competition has lower prices	Our PR people suck
The economy is bad	The competition stole our idea	We didn't get enough VC funding	We need a bigger ad budget	Customers only care about fads
How can we compete with such a big company?	We can't because that would be "selling out"	Our campaign wasn't "viral" enough	Our users are retarded	We needed to keep our margins up
Our employees are lazy	The top bloggers won't talk about us	The competition got there first	It's what "upper mgmt" wants	The schedule was more important than quality
The accountants insisted	We need the revenue in THIS quarter	Our vendors suck	It's not MY fault...	HELLO -- it's BETA, people.

Whoa... guess I have a little 'tude about this, but it's frustrating trying to help companies who want passionate users when their culture keeps users as a low-priority background thread. But things are changing quickly--the days of "he who has the biggest marketing campaign wins" are just about over, and it's shifting to "he who cares enough to deliver what users really want wins." And *that* means the little one-person-start-up might have just as much chance to win customer hearts as the Big Guys.

http://headrush.typepad.com/creating_passionate_users/2006/10/oops_we_forgot_.html

Knocking the exuberance out of employees

By Kathy Sierra on October 6, 2006



In an earlier post I said, *"If you asked the head of a company which employee they'd prefer: the perfect team player who doesn't rock the boat or the one who is brave enough to stand up and fight for something rather than accept the watered-*

down group think that maintains the status quo (or makes things worse), who would they SAY they'd choose? Who would they REALLY choose?

In his book Re-imagine", Tom Peters says, "We will win this battle... and the larger war... only when our talent pool is both deep and broad. Only when our organizations are chock-a-block with obstreperous people who are determined to bend the rules at every turn..."

So yes, I'm thinking Mr. CEO of Very Large Company would say that their company should take the upstart whatever-it-takes person over the ever-compromising team player. "If that person shakes us up, gets us to rethink, creates a little tension, well that's a Good Thing", the CEO says. riiiiiiiiight. While I believe most CEOs do think this way, wow, that attitude reverses itself quite dramatically the futher you reach down the org chart. There's a canyon-sized gap between what company heads say they want (brave, bold, innovative) and what their own middle management seems to prefer (yes-men, worker bees, team players). "

I'm not done with my horse-training-as-universal-metaphor phase, so here's another thing I learned from the Parelli Natural Horsemanship conference:

> ***"Too many people fall into the my robot is better than your robot trap... and knock the exuberance out of their horse. What you're left with is a well-trained robot, not a curious, playful, mentally and emotionally balanced living creature."***

"Hmmmm", I thought, "that sounds an awful lot like some of the companies I've worked for." Not that you'd ever in a million years get them to admit that. Possibly not even to themselves. But the proof is in their practices. Of course some argue that *exuberance* on the job is not necessarily a good thing. That too much *passion* leads to problems. I say BS on that one. Real passion means you love the *profession*, the *craft*, the *domain* you're in. And that may or may not happen to coincide with a passion for your current *employer*. When some folks talk about too much passion for a *job*, they're usually referring to something a little less healthy... the thing that lets your employer take advantage of you, having you work round the clock because of their bad scheduling, or because they refuse to

say "no" to clients, or because you have a manager that wants to look good to *his* manager... and you're the lucky one chosen to be the "hero."

If you knock out *exuberance*, you knock out *curiosity*, and curiosity is the single most important attribute in a world that requires continuous learning and unlearning just to keep up. If we knock out their exuberance, we've also killed their desire to learn, grow, adapt, innovate, and *care*. So why do we do it?

Why Robots Are the Best Employees

1) They don't challenge the status quo

2) They don't ask those *uncomfortable* questions

3) They're 100% obedient

4) They don't need "personal" days.

5)... because they don't have a personal *life*

6) They never make the boss look bad (e.g. stupid, incompetent, clueless, etc.)

7) They dress and talk the way you want them to

8) They have no strongly-held opinions

9) They have no passion, so they have nothing to "fight" for

10) They are *always* willing to do whatever it takes (insane hours, etc.)

11) They are the ultimate team players

12) They don't complain when you micromanage (tip: micromanaging is in fact one of the best ways to *create* a robot)

13) They don't care what their workspace is like, and don't complain if they don't have the equipment they need

14) They'll never threaten *your* job

15) They make perfect scapegoats

16) They get on well with zombies

And while I'm here... parents do this as well. Admit it. We have *all* wished that our children (for whom we worked so hard to instill a fierce independence) would be strong-willed, exuberant, questioning--everywhere but at *home*. I've never really wanted Skyler to *be* a robot, but oh how I've wished for a robot *mode*... ;)

http://headrush.typepad.com/creating_passionate_users/2006/10/knocking_the_ex.html

Reducing fear is the killer app

By Kathy Sierra on October 14, 2006

The high-pitched screech of the drill. The sickly smell of antiseptic and fear. The long nervous wait for the attendant to call your name and take you... back *there*. We assume that people are afraid of the *dentist*, but we don't usually think of *software* as scary. Maybe we should rethink that. Our users might be more afraid of us and our products than we think. And those who can reduce or eliminate that fear have a huge advantage. Not to mention a passionately loyal following.

Something extraordinary happened to me yesterday, but before I tell that story I want you to look at these pictures:

One, a drab but typical medical office. The other, a warm, inviting, spa-like environment. *The spa-like place is actually my dentist's office.* And I would drive 100 miles to go there, because the people there work their a** off to reduce my fear. And the pictures don't do it justice because you're missing the smell (freshly ground coffee beans and warm cookies) and the sounds (jazz, not drills).

Here's another picture, of the Boulder Community Foothills Hospital, the first hospital in the US to earn the LEED certification for being "green."

It doesn't look like a hospital. It doesn't feel like a hospital. And it doesn't *smell* like a hospital. I'm not sure how they do it, but no matter when you go, it smells like fresh popped popcorn. Think about that... almost nobody has a bad association with the smell of popcorn. I instantly think movies and theme parks. (And the *live* piano music reminds me of shopping in Nordstrom's.)

In a medical scenario, reducing fear means a *lot*. But think about all the ways *our* users (or potential users) might be afraid. Not in mortal terror, but afraid nonetheless. The fear of not being smart enough to *learn* a new product, programming language, or procedure. The fear of being taken advantage of by an unscrupulous company and/or sales person. The fear of making the wrong purchasing decision. *The fear of looking stupid or slow in front of our co-workers.*

I've often said that reducing *guilt* is the killer app, but now I'd put reducing *fear* way up there too. He who reduces fear better than the competition can, potentially, stop competing on price, convenience, or just about anything else. Reduce my fear, and I'll be grateful forever.

So here's my story:

Y'all have probably seen a lot of *pink* lately, inspired by the fight against breast cancer. Yesterday, I went to the Boulder Foothills Hospital (in the picture) where I was scheduled for a mammogram. I was terrified. I'm not exaggerating. As many of you know, my mother was diagnosed with breast cancer at a

young age, *younger than I am now*. She did not survive. The most tragic part was that she probably *would* have, if it had been detected earlier. But she was too afraid to have the exam... afraid of hearing the results she ultimately got.

Cancer has been on my mind a lot this year. Less than a year ago both myself *and* my daughter were diagnosed with a form of cancer that had not yet become invasive, but that could have killed both of us had we not been tested.

But worst of all, I have--quite irrationally--not had a mammogram in 10 years. A monumentally stupid choice, given that I'm at very high risk for breast cancer. But... I am more terrified of that test than anything I've ever done, and I've spent the last few years convinced that it was already too late. Thinking about it sends me straight to the childhood moment when I learned the results of my mother's mammogram (and the awful period that followed). It was selfish of me, as a mother myself, to not do everything I can to stay healthy and alive, but fear does bizarre, irrational things to the brain. Finally, though, all the pink-awareness and a visit to this extraordinary hospital convinced me.

When I arrived, I told the technician my story, and literally begged her to rush the results. "7-10 days is how long it takes for the doctor to review it and get the results to *your* doctor," she said. "There's nothing I can do to speed that up." I could barely breathe or walk, but I managed to get through the exam. But now the *worst* part begins... The *Wait*. The first wait is for the ten minutes it takes for the tech to review the film to make sure the pictures aren't too dark, light, or blurred. Once they've checked the film, they either walk you back to repeat the test, or send you home to start The Wait. So there I sat, waiting for the tech.

Five minutes passed. Ten minutes. *20* minutes I sat in that little room. Finally she walked in and said, "The film is fine, you're free to go." And then something happened that I'll remember for the rest of my life. She sat down next to me and said, "Oh, so how would you like to enjoy your weekend?" I was confused. "I convinced the doctor to break protocol. He said everything's perfect and we'll see you in a year." We both cried.

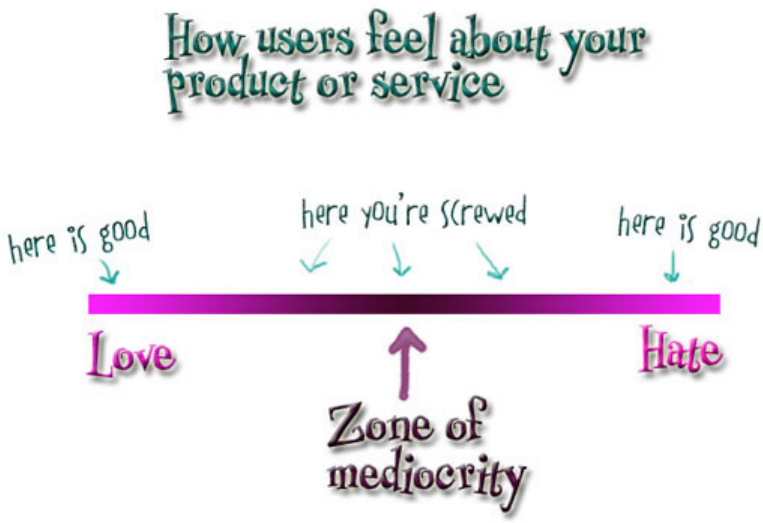
Reducing fear doesn't have to be about life or death or pain to be meaningful and powerful. If you can help your users feel more confident and less stressed, you've given them a wonderful gift. Whether it's a policy change, better documentation and support, or more user-friendly design, *anything* you do to genuinely reduce my fear improves my life. Why *not* ask customers about their needs before you agree to sell them something? (And be willing to "downsell" rather than trying to convince them to buy something *more* expensive than they need.) Why *not* keep Consumer Report magazines in your dealership, or give potential customers a quote from your competitors, *even if it means you lose that sale?* Why *not* work harder to make sure new users (or students) realize that they really ARE going to be able to "get" this, and that you'll be there every step of the way?

Reduce my fear and I'll love you forever. :)

http://headrush.typepad.com/creating_passionate_users/2006/10/reducing_fear_i.html

Dilbert and the zone of mediocrity

By Kathy Sierra on October 18, 2006



How brave are you? How far will you (or your employer) go to avoid the Zone of Mediocrity? Until or unless you're willing to risk passionate *hate*, you may never feel the love. Scott Adams agrees. In a recent post on the Dilbert blog, he said, *"If everyone exposed to a product likes it, the product will not succeed... The reason that a product "everyone likes" will fail is because no one "loves" it. The only thing that predicts success is passion, even if only 10% of the consumers have it."*

This is NOT about being *remarkable*-- it's about being *loveable*. And that almost always means being *hated* as well. Our Head First Java book, for example, has 139 Amazon reviews, and most are either five stars ("love it, best technical book ever, I learned a lot") or one star ("hated it, worst technical book ever, authors should be shot.") But crafting a book that people would either love or hate was not our intention. We set out to make a more brain-friendly learning book format, and we were just clueless and naive enough to not realize how many implicit "rules" we were violating. It wasn't until O'Reilly editors started a mini revolt against it that we knew we'd crossed a Line That Shall Not Be Crossed and created something potentially *embarrasing*.

Today, it is often far *more* risky to create something "safe" than to take a big frickin' chance on something deeply provocative, dangerously innovative, or just plain weird.

Think about all the things you love today that once seemed very, very weird. Things that someone took a huge frickin' chance on.

Today, the more you try to *prevent* failure, the more likely you are to fail.

That wasn't always true, but geez... how many more [whatevers] do we *need* today? There are way too many of all the things we *already have* and not enough introductions of things we *don't* have. We all know the reasons why companies play it safe, and why employees are often forced to play it safe, but this me-tooism isn't helping anyone.

What does it take to move out of the Zone of Mediocrity?

Normally at this point I'd talk about the usual things *everyone* talks about... how to come up with breakthrough ideas, where to look for opportunities, being innovative, blah blah blah. You know all that. I think it really comes down to this:

To avoid the Zone of Mediocrity, you must suspend disbelief.

You must be willing and able to turn off (temporarily) The Voice inside that says, "We'll never get away with this. People will hate it." That doesn't necessarily mean The Voice is *wrong*, but until you can shut if off, you're virtually guaranteed to stay with safer, incremental ideas. But remember--"safer" really *isn't* safer anymore, unless you're looking only to avoid criticism. Safe will keep you safely out of the spotlight. If that's what you want (and sometimes that's the best approach), then fine. But if not...

(side note: this is somewhat like The Inner Game approach or Drawing on the Right Side of the Brain or any of the other approaches to creativity that get your logical "talking" mind out of the way so all the more *useful* but non-speaking parts of your brain can get on with the important things you're trying to accomplish.)

And it's not just suspending disbelief about what *users* (or critics) will say... you must also suspend disbelief about what your *company* will let you do. I first experienced this at Sun, where it was almost impossible to creatively brainstorm about

ways to improve things without someone jumping in with, "Yeah, but they'd never let us do that." End of discussion. End of chance to do something amazing. Every time I do an internal workshop, the partipants are far more negative than when some of those *same* people are in a public version of my passionate users workshop. By taking them outside their company and having them brainstorm or work on fictional or other people's projects, their minds are free to move about. I've nearly quit doing in-house workshops because the "they'll never let us do that" syndrome is so strong.

You can't help users kick ass until your employer lets YOU kick ass. Easy for the unemployed ME to say ;)

(Thanks to Karl Nieberding, Kyle Maxwell, and John Radke for telling me about the Dilbert post!)

And one more follow-up note: I heard from the guy who designed the Airstream 75th Anniversary Trailer (wow -- if ONLY I could afford that one, it would have been my first choice). His studio builds custom and restored vintage trailers, and even if you don't want one now, you should still check out his Vintage Trailering site just to see his work. There's *nothing* mediocre here!

http://headrush.typepad.com/creating_passionate_users/2006/10/dilbert_and_the.html

Better Beginnings: how to start a presentation, book, article...

By Kathy Sierra on October 22, 2006

You are in a dimly lit room. You are alone on a stage before an audience of 1,000. 10 minutes into your presentation, your hands no longer shake or sweat. This is going well, you think. But just then you notice a vaguely familiar sound--*tap, tap, clickety-clack*--which in one horrifying moment you *recognize*--it's your audience. IMing, checking email, *live blogging* ("wifi sucks at this hotel and OMFG this is the most boring speaker ever")

What went wrong? How did you lose them in the first 10 minutes? *How can you get their attention?*

Nobody knows more about the importance of beginnings than novelists and screenwriters, but too often we think their advice doesn't apply to *us*. After all, we give *technical* presentations. Lectures. Sermons. We cover *professional* topics, not *fiction*. Not *entertainment*.

Oh really? Regardless of your topic, the only way they'll *read* or *listen* to it is if you get them hooked from the beginning. And like your mother always said, *"You never get a second chance to make a first impression."* (Or as one writer put it, "You can't be in the room with the reader to say, 'trust me...it gets better.')

So, we took some tips on making a good beginning from those whose work depends on it.

1) Do NOT start at the beginning!

Advice for first-time novelists is often, "Take the first chapter and throw it away. Chances are, chapter 2 is where it just starts to get interesting, so start THERE." Start where the *action* begins! What happens if you remove the first 10 minutes of your presentations? What happens if you remove the first chapter? Or the first page, paragraph, whatever?

Yes, this means dropping the user straight in to the fray without all the necessary context, but if the start is compelling enough, *they won't care*, at least not yet. They'll stick with you long enough to let the context *emerge*, just in time, as the "story" goes along. One of my biggest mistakes in books and talks is overestimating the amount of context the listener/reader really needs *in advance*.

2) Show, Don't Tell

If you have to TELL your audience that they *should* care, you're screwed. The motivation for why they should care should be an inherent part of the story, scenarios, examples, graphics, etc.

3) For the love of god, DO NOT start with history!

If I read just ONE more book about the web that starts with a history of the internet, I will have to take hostages. Seriously. Do any of us really *need* to know about DARPA and CERN and...? Do most web designers and programmers really *care*? No, and No. And it's not just web design books that suffer from this worst-thing-to-put-in-chapter-one syndrome. WHY DO AUTHORS KEEP PUTTING THE HISTORICAL OVERVIEW AT THE BEGINNING OF THE BOOK?? If you feel driven or morally obligated to include the history of whatever, fine, but *don't put it at the front*. Stick it in an appendix or on a web page, where it'll do the least damage. (To be fair, there *are* plenty of topics where the history *is* interesting and useful, but rarely is the historical overview the grabby get-them-hooked thing you need up front.)

If you do have context that matters--including history (although I'd fight like a mother tiger to convince you it wasn't needed)--let it emerge *during* the talk or book, not *before*, when they're the least motivated to hear it. Think about all the things you've pursued where the history became interesting to you only AFTER you developed a strong interest in and knowledge of the subject.

4) DO NOT start with prereqs

Decide what is absolutely, positively, crucial and then... stick it in an appendix. If you write for an audience that you assume probably *has* those prereqs, then why ruin the first chapter for them? Why slow them down? Chances are, they won't just skip chapter 1 and start at chapter 2. Chances are, they'll just skip the whole book.

5) MYTH: you must establish credibility up front

How many talks do you see where the speaker has multiple bullet points and slides just on their background? I did it once because I thought it would help people understand the context of my talk, and it did NOT go over well because:

A) Nobody cares

B) Bullet points do not equal credibility

C) Nobody cares

D) You already HAVE credibility going in... you don't have to *earn* it, you just have to make sure you don't *lose* it.

E) Nobody cares

But I also see this in books, where it feels like the author is trying to *prove* to you how smart he is. A better approach might be to prove to the reader how smart HE is, by not dumbing it down. And by demonstrating to the reader/listener that he's capable of "getting" this really tough thing. I have no illusions about this--the reader/listener cares about himself waaaaaaay more than he cares about me.

Trying to establish credibility is backwards. Don't try to get the reader to respect YOU... the reader wants to know that you respect HIM!

Demonstrate that respect by caring about his time. By caring about the *quality* of time. Your audience should know right up front that you're grateful for the time they're giving you, and you show that by being entertaining, engaging, compelling, interesting, or at least *useful*. You demonstrate it by assuming they're smart. By recognizing what they already bring to the discussion. By not insulting their intelligence. *By being prepared*.

IDEAS FOR BEGINNINGS

A few tricks of the novelists, screenwriters, and world's best teachers. Use one or more of the following to open with an impact:

Begin with a question. A question the listener wants to have answered

It doesn't have to be a *literal* question, just something they want to find out. In a good movie or novel, you find yourself thinking, "Who is this guy? Why is he in this situation? Will he get out of it? What's this secret thing they keep referring to?" **Make them curious.** Curiosity is seduction. I'm astonished by how often we suck the life out of technical topics, when they could be fascinating. Find the passion. If YOU don't care about the answer, why should they?

Be provocative

Challenge a belief. Even if they instantly disagree, they'll stick with it long enough to find out where you got that crazy idea. Start with your most dramatic and/or unpopular assertion.

Evoke empathy

Start with a story about real people, or about a fictional character they can identify with.

Do something surprising... VERY surprising

They'll want to stick around to see what strange thing you do *next*.

Start with something funny

Forget the advice to "open with a joke", unless you happen to be one of those rare funny people. But you don't have to start with a joke to get them laughing early. Sometimes a picture, story, or just a quote can get them to stick around because you entertained them... at least for a moment.

Promise there will be conflict

We would rarely read a novel or see a movie if not for the promise of conflict. Tension and suspense are compelling. How will this turn out? How will you ever scale that thing? How can

you build this system in this ridiculous amount of time using only duct tape and a tin of Altoids?

Start with a dramatic key event or turning point

Mystery, suspense, intrigue

How many *bad* books and movies have you stuck with just because you had to find out who did it? Look at your topic and find a way to set up a little mystery. ANYTHING worth talking or writing about has potential for mystery (which leads to curiosity).

Deliver an emotional experience

Your job is to touch their emotions in some way. Not a "I laughed I cried I was moved" thing, but remember: people pay attention to that which they *feel*. Look at your first set of slides and your first few pages and ask yourself, "what *feeling* does this evoke?" Raise your hand if you've been to way too many talks and read way too many books where nobody asked that question.

"Always grab the reader by the throat in the first paragraph, send your thumbs into his windpipe in the second, and hold him against the wall until the tagline." -- Paul O'Neil

That's the goal, but only the truly talented can actually do that. Me? I'll settle for getting the reader to give me just one more moment. Then another. Then another. And I value deeply (and feel lucky for) each moment y'all are willing to give me.

http://headrush.typepad.com/creating_passionate_users/2006/10/better_beginnin.html

Two simple words of passion...

By Kathy Sierra on November 3, 2006

Seattleduck's Kevin Broidy captured the essence of user passion when he said, "Passion starts with two simple words: F***ing Cool!" In Kevin's words (and I think he nailed it):

> *"That's where passion begins. Those are the words I want every user of my product to utter. Ideally followed up by something like:*
>
> *'Dude, you have to check this out. It's so f**king cool!'*

> *I don't want their reaction to be a measured, rational, dispassionate analysis of why the product is better than the alternatives, how the cost is more reasonable, feature set more complete, UI more AJAXified. I don't want them to pause to analyze the boring feature comparison chart on the back of the box.*
>
> *I want **'f**king cool!'** Period.*
>
> *I want that pure sense of wonder, that kid-at-airshow-seeing-an-F16-on-afterburners-rip-by-so-close-it-makes-your-soul-shake reaction, that caress-the-new-Blackberry-until-your-friends-start-to-question-your-sanity experience. I want an irrational level of sheer, unfiltered, borderline delusional joy."*

But "f***ing cool" is not a "business appropriate" phrase. It's unprofessional. So while we may want our customers to feel it, sure, we certainly can't have one of our *employees* saying it. Heavens no. According to some folks within Sun, anyway. It seems that the insightful tech blogger Tim Bray--*who happens to be a Sun employee*--used the words, "f***ing cool" (but without the asterisks) to describe Sun's Project Blackbox, and he took some interesting heat for it from both outside and inside the company.

> *"Out of public view, the Sun internal bloggers alias exploded, opinions ranging from those saluting me as an exemplar of New Age Marketing and Proactive Transparency to others who felt my mouth ought to be washed out; one person related that he'd heard from a Sun shareholder who was going to sell as a consequence."*

But this is who Tim is. He didn't use the F-word to attract attention. He used it because he honestly believes that Project Blackbox is, "totally drop dead f***ing cool." So he said it. And of course it brought up all sorts of issues related to honesty, authenticity, professionalism, personal vs. corporate blogs, etc.

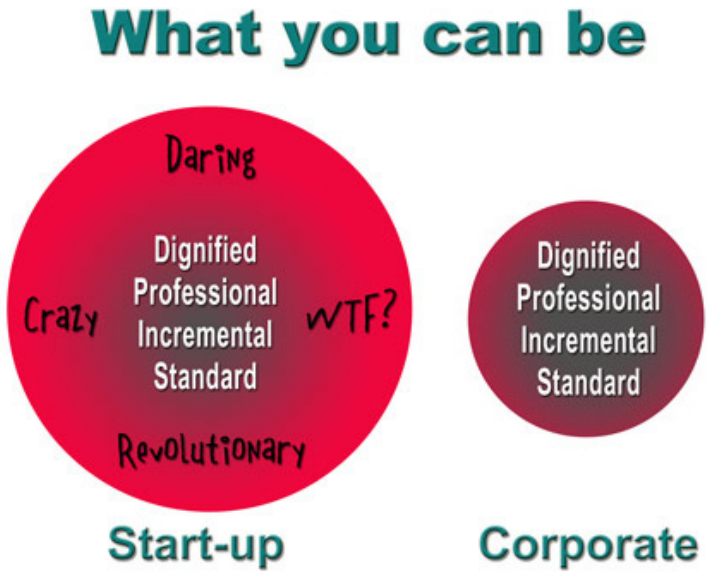
I would be proud and thrilled to have someone describe something I made in those terms. Those exact, most passionate, terms.

But here's why this whole story made me smile (in that but-I'm-not-bitter kind of way)... when *I* was still a Sun employee, I got

into *serious* trouble for using just ONE of the words in that phrase... *cool*. Yes, cool. It came up as a black mark in my annual employee performance evaluation. So Tim, times have changed when *you* call a Sun product "f***ing cool", and all they care about is the F-word.

We all have to decide what constitutes "professionalism" for our own business. And my standards might be much lower (or rather *different*) from yours. One of my favorite exchanges was between Hugh MacLeod and one of his commenters, some time back. Hugh, not known for self-censorship, was told, "Maybe you'd have more clients if you stopped using such inappropriate language." His response: "If that's what I'd have to do to win those clients, they aren't the people I'd want to work with anyway."

Which reminds me of Paul Graham's Dignity is Deadly speech.



http://headrush.typepad.com/creating_passionate_users/2006/11/two_simple_word.html

The Zone of Expendability?

By Kathy Sierra on November 20, 2006

In Don Norman's words, "If someone doesn't hate your product, it's probably mediocre." If playing it safe today is considered a risk in business, what about in a *job*? If all managers *like* you, are you safer than if some think you're amazing while others think you're the poster child for Bad Hiring Decisions?

My little trip back to Sun was a perfect reminder of this... during the time I was there as an employee, one manager would give me an award while another would dig up dirt for my next performance review. Marketing gave me a bonus (for having a tech article published in a major trade magazine) while another department gave me a reprimand for not getting all the proper approvals. One boss went out of her way to use the downturn as an excuse to give me the one job she knew I hated, while another used his remaining time to give me a ridiculously large salary increase. The thing is, through all of this... I was always the same person. In a single year, I went from best thing since canned beer to best reason for having a Prompt Exit Plan (otherwise known as the, "We're almost certainly going to fire you unless you do this [insert thing they know you won't do], but we'll give you the chance to leave quietly if you go now...].

I'm rightly and frequently criticized for celebrating the trouble-maker... for making the rule-breaker into some kind of hero. But I agree that just because one challenges the status quo does NOT mean they're helping. And just because one has bold, risky ideas doesn't mean those ideas are *good*. Sometimes a rule-breaker,

non-team-playing upstart is simply... a pain in the ass. But too many managers appear too threatened to figure out whether their trouble-maker is the one person who can really push things forward, or the one who simply thrives on being disruptive.

There are no guarantees, of course. Especially now. But while in the past the safest move was to keep your head down and stay off any radars... being a good little trooper... that's no longer any more likely to help you keep your job. If you're on *nobody's* radar, you've probably got nobody defending you like a tiger to *their* boss. In either case, the freedom to push for what you believe in, and to challenge the status quo is a lot more *stimulating* than deciding to just not care.

If everyone is a lot more expendable today, and we ALL are "short-timers" whether we know it or not, we might as well *act* like short-timers by taking the risks we were too afraid to make before. It probably won't make us any *less* at risk, and today... it might even make us safer.

[Footnote: this seems to be true for students as well... like mother like daughter I suppose, but Skyler's report cards always made me smile when it came to the little extra notes each teacher attached to the grade. It was amazing how a single person--Skyler--could simultaneously be "a joy to have in class!" an "inspiration" and "disrupts the class" and "disrespectful", all in a single semester. Again, this is not the strategy I'd ever recommend, especially for a child who wants into a great college, but this kid has other plans for the world, and I can't say I'm not secretly delighted. Life... is just too darn short not to speak up.]

http://headrush.typepad.com/creating_passionate_users/2006/11/the_zone_of_exp.html

Still listening

By Dan Russell on November 21, 2006

I know I said we should listen to our outraged users.

But I didn't mean we should ignore all our happy, contented users as well. Nor do I suggest that we pay attention to the folks who like some features, but dislike others. (Ah, they inhabit the zone of mediocre reactions!)

Instead, we have to listen to them all, then show good taste in figuring out which of the voices have value to them. Sometimes it's the screamers that make the good points about your product (no matter how hard it is to get over the emotional tumult of their delivery), and sometimes it's the quiet voices with the critical commentary.

Let me go all Zen on you for a minute and suggest that what you--as skilled practitioners of software / product / services / education-need to do.

Listen while being still.

My previous post about listening to the screamers was all about how to set aside your immediate emotional reaction to their delivery and look for the nuggets of truth and insight within the scream.

The same is true for anyone who's willing to give you some feedback. Listen without reacting so you can hear the valuable bits of what they say. I know this means you have to do a little emotional work on your part, mostly suppressing your own reaction to their reaction: but you can do it. If Spock can do it, so can you.

I mentioned the value of hearing someone describe my early software as "white, male, fascist." It stung to hear that. That was a great example of listening to a screamer's voice.

But just a few weeks ago I was doing a field study, listening to a user talking about how hard it is to do some kinds of web searches. "I don't know," she said, "I think there's got to be a way to find this, but how?"

This was a busy Mom with three little kids (one in her arm as we were talking), a dog and the plumber all wandering through the house. Even though her house was busy, she literally spoke quietly and calmly.

Of course, I could tell her how to use an advanced operator, maybe show her the advanced search page. "Ah.. that's it! I'll show her the advanced search!" I think to myself, "get her onto the road to being a power user."

Proudly I showed how with one click she could get to a page with all kinds of power search features. Tools that I knew would give her exactly the skills and capabilities she needed to do an instant, precise and potent web search.

"Oooh." She said, upon seeing that page with all the options. It wasn't a happy "oooh" either. I looked at her eyes to see what she was looking at, and I could immediately see that she didn't know what to focus on, her eyes revealed the truth as they swept from side-to-side, looking for something familiar. There are a lot of features and options on the page, perhaps a few too many.

So I asked a very non-techy question: "Umm: How do you feel about this?" It's a low-tech but high-touch question. It's deliberately non-leading and open-ended.

And she proceeded to talk for another minute about how that particular page was "scary and intimidating." What do you know.

I've never thought about a web page, especially a search page, as being "scary" -- but here she was, telling me that it's a frightening thing. Unpacking WHY it gave her that moment's pause has been the most illuminating thing I've learned this month.

So the flip side of the screaming user is the user that says "ooh" in a quiet voice. Those voices are important too. Our job is to hear all the tones and semitones in what our users are saying, and be still enough in ourselves to be able to understand what it all means.

http://headrush.typepad.com/creating_passionate_users/2006/11/still_listening.html

Rhythm method

By Dan Russell on November 24, 2006

Life runs on a pulse: without that basic rhythm, you're dead.

What's more, that sense of a beat measuring out time is intrinsic to almost everything we do, be it conversations, living our lives or learning how to use software.

A story that illustrates the point:

A few years ago, while studying the way teachers teach Japanese in a classroom, I spent a lot of time watching videos of language classes. If you've ever spent time learning another language, you know what it's like-lots of drilling and practice speaking. It's an essential part of class; you need to get good at hearing and saying different word forms and that requires a great deal of back and forth between teacher and students.

But as I spent hours doing video analysis of the classes, I ended up doing a lot of rewinding and fast-forwarding. As you'd expect, I watched a lot of Japanese classes go by on the video monitor in 2X or 3X real time. And I noticed that when you watch these classes fly by on video, there's an amazingly regular pulse to the class. The teacher prompts, the students reply: the teacher prompts, the students reply: on and on.

Curious about this unexpectedly regular pulse in the class, I actually went through one of the classes and noted each prompt and each response, wrote down the timecode, then plotted them out as a graph. To my amazement, the back-and-forth of the interactions was incredibly regular - each event showed up as a regularly spaced dot on my chart.

I'd found that there is a rhythm to a language class, a regular beat to the back-and-forth that makes the class work. I quickly noticed in the analysis that anytime the pulse was disrupted by more than a second or two, that was when something had

broken down in the class-a student couldn't answer a question, or when the teacher moved onto another segment of the class.

Unfortunately, looking for rhythmic structure in the classroom wasn't the topic of that research study, so I filed this under "interesting stuff to think about in the future" and pressed on with my work.

But what I found completely fascinating was the incredible regularity and structure of the interactions. Was this true of other kinds of interaction settings as well?

Years later I found Edward T. Hall's book, *Dance of Life,* that pretty much confirms that observation I'd made. Hall's an anthropologist who's made a career out of watching how people interaction in time and space. In this book he shows how people not only interlock the rhythms of conversation, but also how they move closer and farther apart, effectively dancing in their day-to-day interactions.

More recently, I've been looking at the rhythms of people doing search queries. Lo and behold, a similar pattern stands out: If you watch someone's timing of searches, you can see striking patterns of when they post a query, how long they go between queries and how each day is similar to other days.

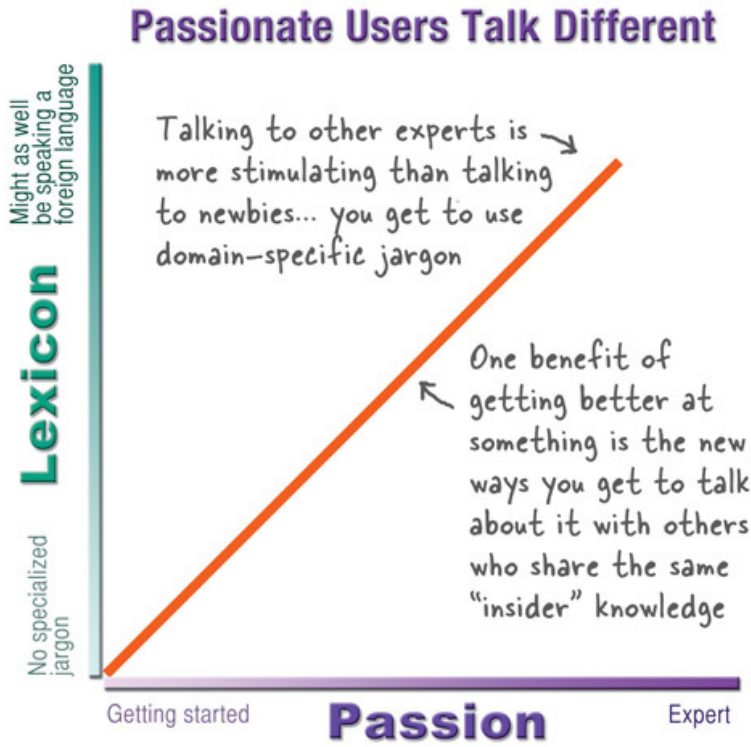
Although it doesn't always feel that way, people are amazingly rhythmic in their behaviors, up to and including their use of software. This is a point beautifully noted by Bo Begole, John Tang and Roscoe Hill in their paper Rhythm modeling, visualizations and applications.

What does this have to do with Creating Passionate Users? It prompts me to ask you, dear readers, a question. What rhythmic patterns do you see in the course of your work? Do passionate users have a rhythm that works well, signaling happy use? I'm interested in hearing your notes on the time course of events. Do you find people using software in an interesting repeating pattern? Just using something everyday is dull, but perhaps you see other pulses, rhythms and beats that go beyond the ordinary. What about it? Is rhythm a good thing, or the sign that things have become mechanical? Is rhythm a part of a flow experience, or its nemesis?

http://headrush.typepad.com/creating_passionate_users/2006/11/rhythm_method.html

Why Web 2.0 is more than a buzzword

By Kathy Sierra on November 26, 2006



Many people hate the phrase "Web 2.0" even more than they hate *what they believe it represents*. No, that's not quite right... many people hate the phrase precisely because they think it represents *nothing*. Or they're annoyed by the idea of a web *version number*. Or they think it's "elitist." Or they're convinced it's so much marketing hype. But what if it's *not* an empty phrase? What if it's simply a way of representing a concept that some people DO understand? What if it's like so many other domain-specific terms that sound like nonsense to everyone else?

That doesn't mean zillions of people haven't abused the term for everything from sounding tech-savvy to getting a piece of the hype-fueled-please-god-bring-back-the-bubble-and-I-promise-I-won't-piss-it-away-this-time VC pie. And it doesn't mean that there's all that much consensus even among those who think

they DO know what "Web 2.0" means. But to say it means *nothing* (or WORSE--to say it's just a marketing label) is to mistake jargon (*good*) for buzzwords (*bad*). Where **buzzwords** are used to *impress* or mislead, **jargon** is used to communicate more efficiently and *interestingly* with others who share a similar level of knowledge and skills in a specific area.

Part of the benefit of being "into" something is having an *insider* lexicon.

It's not about *elitism*--it's about *efficiency*. It's not about impressing others--it's about a shared understanding of specific concepts. It's about being able to talk about ideas or processes or even *parts* with fewer words and (potentially) greater meaning. If two heart surgeons debate the merits of a new medical procedure, I'd be lost. Hell, I'm over my head when the conversation turns to *cooking*. But I can talk about *cantles* and *pommels*, and I know exactly what *topline* means in the context of *collection*. And I can talk about recursion and dependency-injection and backward-chaining. Just don't ask me how to *carmelize*.

Dinner conversations around my house often *are* about one of those two things--programming or horses--and most non-horse, non-developer folks might wonder if we're just making s*** up. But if you took away our *jargon*, the conversations would not just be *slower*, they'd be *dumber*. We couldn't converse on some of the more sophisticated, complex, higher-level ideas about horses or software development. The experience wouldn't be as rich, productive, or engaging. Strip away the specialized words and you strip away part of why being better is better.

One of the biggest mistakes I see community builders make (however well-intentioned) is fretting over inclusivity and newbie-friendliness. They want the beginners to feel welcome, and few experiences are more daunting than stepping into a new domain where you have no idea what anyone's talking about. It feels... uncomfortable. Confusing. Discouraging. But in our quest to cut the jargon and perceived (or even real) elitism, we risk ruining one of the biggest benefits of sticking with it. Not only should we *allow* domain-specific jargon or expert-speak, we should be driving it! We should help *invent* short-cuts and specialized words and phrases to make communication among our most passionate--our *experts*--even more stimulating and useful.

If you're afraid of newbies feeling intimidated or unwelcome, by all means give them a separate safe zone. Whether the newbie space is the *default* while the advanced users have their own special area (site, forum, club, whatever), or just the opposite--the advanced users are the default and the *newbies* get their own special **beginner** area, the key is to not sacrifice your advanced users in an effort to make beginners *feel* better. That's a short-term benefit to the beginner but a long-term wet blanket over those who might otherwise be more motivated to move up the ranks.

So... back to "Web 2.0"--I'll admit that this one's trickier than most domain-specific phrases because it wraps *many* different--and big and ill-defined--concepts. But when Tim O'Reilly and Dale Dougherty (the guy who first coined the term) talk about Web 2.0, it represents something real and specific and meaningful. Over time, a *lot* of other people (especially those who've spent time around them, including me) have come to understand at least a *part* of what they've encapsulated in that one small phrase. "Web 2.0" may be the least understood phrase in the history of the world, but that *still* doesn't make it meaningless.

Think of all the *other* words or phrases that mean nothing to us simply because we're not in that profession or hobby. Pop Quiz: From which domains do these sets of words or phrases come from? (And hey, try to see how much you can get *without* Google.)

A) The flop, the turn, and the river

B) purlwise, stockinette, double-pointed

C) snowman, gimmie, duck hook

D) blowbag, escutcheon, gas cock

E) grind, fakie, bluntslide

F) abseil, hexcentric, friend

G) sente, tiger's mouth, "black is thick"

H) break, build, "train wreck"

I) vermin type, use-activated, swarm subtype

J) ruck, maul, blood-bin

K) HIWAS, option approach, DOD FLIP

L) clipping, phantom power, patch bay

M) flashback, freelist, Scott

N) Class M, dilithium, positronic

First person to get *all* of them gets a surprise.

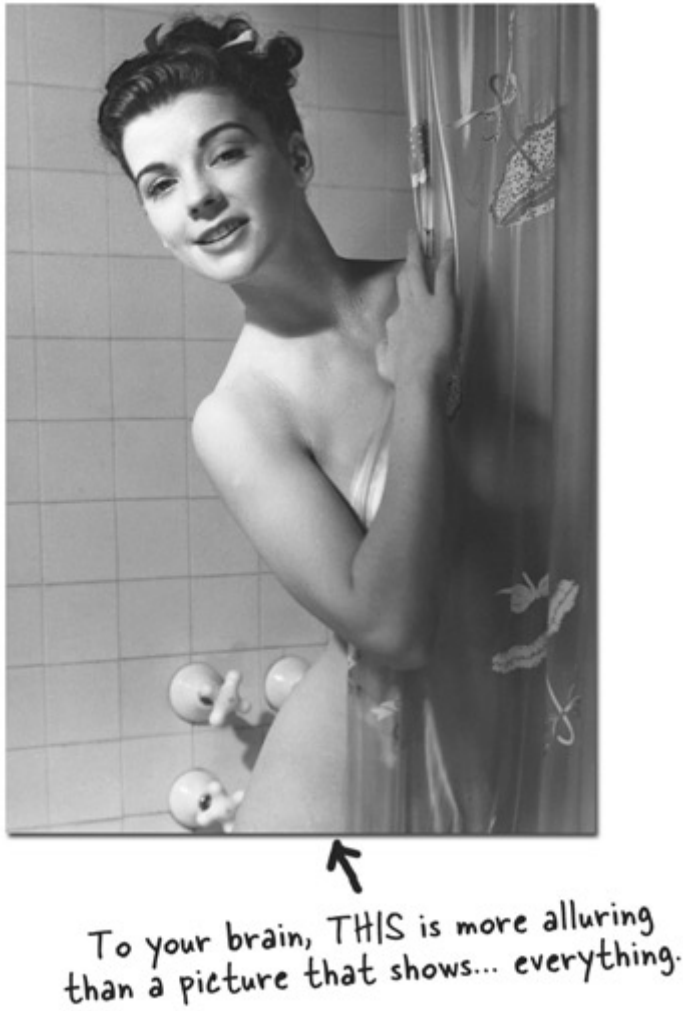
[UPDATE: once you look at the comments, you'll see everyone else's answers so... watch for the spoilers.]

[UPDATE: OK, new challenge... since everyone guessed mine so quickly, I'd love to hear YOUR idea for a set of three words/phrases from some domain/profession/hobby that the rest of us have to guess...]

http://headrush.typepad.com/creating_passionate_users/2006/11/why_web_20_is_m.html

Cognitive Seduction and the "peekaboo" law

By Kathy Sierra on November 28, 2006



Brains are turned on by puzzles. Brains are turned on by figuring things out. Brains are turned on by even the smallest "aha" moments. And despite what some of you *(*cough* men *cough*)* might believe, the brain is more turned on by seeing just the *arms* of a naked woman behind a shower curtain than it is by seeing *all* of her. So if you're trying to engage someone's brain, don't show *everything*. Let their brain connect the dots.

At least, that's what the neuroscientists say in the latest issue of Scientific American Mind. In their article The Neurology of Aesthetics, our favorite brain guy V.S. Ramachandran and Diane Rogers-Ramachandran describe a series of "laws" of aesthetics (they put "laws" in quotes) and how they're supported by what we know of the brain. My favorite--and one that we've been talking about (minus the festive name) for a long time here--is known as **Peekaboo**.

From the article:

> *"An unclothed person who has only arms or part of a shoulder jutting out from behind a shower curtain or who is behind a diaphanous veil is much more alluring than a completely uncovered nude. Just as the thinking parts of our brain enjoy intellectual problem solving, the visual system seems to enjoy discovering a hidden object.*
>
> ***Evolution has seen to it that the very act of searching for the hidden object is enjoyable, not just the final "aha" of recognition--lest you give up the chase.***
>
> *Otherwise, we would not pursue a potential prey or mate glimpsed partially behind bushes or dense fog."*

If something dangerous is hiding in the bushes, it's damn useful for the brain to reconstruct a complete tiger from just a few bits of orange and black peeking out between the leaves. Apparently it's all the little **mini-aha moments** that send messages to the brain that prompt still more searches and more mini-ahas until the final BIG aha where your brain nails it.

It goes on:

> *"The clever fashion designer or artists tries to evoke as many mini "ahas," ambiguities, peak shifts and pardoxes as possible in the image."*

We're always trying to leave something to the reader/learner/observer's imagination. Something for *them* to fill in. (This relates to our earlier space between the notes post).

In my workshops and talks, I show a series of photos where things are not fully resolved... a face hidden behind a hand, a (potentially naked) woman staring intently at an object you can't

quite see, the lower half of a young man suspended in air next to a tree, where you can't see the ground OR anything above his waist (is he hanging from the tree? on a trampoline? in the midst of an alien abduction?) To the brain, these "Hmmm... what's the story here?" images are virtually irresistible. The brain *needs* to figure it out, and enjoys the experience.

This applies to non-visual things as well, of course. In learning, the more you fill things in and hold the learner's hand, the less their brain will engage. If they don't need to fire a single neuron to walk through the tutorial, lesson, lecture, etc., they're getting a shallow, surface-level, non-memorable exposure of "covered" material, but... what's the point? Obviously this doesn't mean you just never tell them anything period. This is about graduated hints, mental teasing, cognitive treasure hunts, sparking curiosity, etc. Things that engage the brain. (This is part of the brain-friendly strategy we use in our books.)

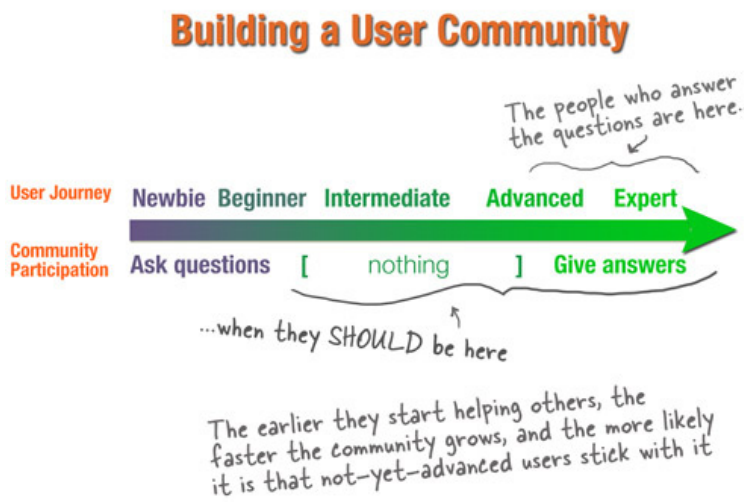
Whether you're trying to *get* someone's attention, *keep* their attention, *motivate* them to stick with something, or help them to learn more deeply and retain what they've learned, leave something for their brain to resolve. Do something to *turn their brain on.*

[Disclaimer: this does NOT apply to something like reference docs, where you don't *want* their brain to become engaged. With reference material, I want to get them in and out as quickly as possible--with the accurate info they need--and where retention and recall is not a goal.]

http://headrush.typepad.com/creating_passionate_users/2006/11/cognitive_seduc.html

How to Build a User Community

By Kathy Sierra on December 3, 2006



Most user communities take a typical path--the newbies *ask* questions, and a select group of more advanced users *answer* them. But that's a slow path to building the community, and it leaves a huge gaping hole in the middle where most users drop out. If we want to keep beginning and intermediate users more engaged (and increase the pool of question *answerers*), we need them to shift from asker to answerer much earlier in their learning curve. But that leaves two big questions... 1) How do we motivate them? 2) How do we keep them from giving lame answers?

Actually, this isn't the *biggest* problem with most user communities. The real deal-killer is when a new or beginning user asks a "dumb" question. Most supportive, thriving user communities have a culture that encourages users to ask questions, usually through brute-force moderation with a low-to-no-tolerance policy on ridiculing a question. In other words, by forcing participants to "be reasonably nice to newbies", beginners feel safe posing questions without having to start each one with, "I know this is probably a dumb question, but..."

It was precisely that idea that led to the original javaranch... in 1997, the comp.lang.java newsgroup was just too nasty a place to ask questions. Even if you *were* brave enough to ask an

obviously stupid one, the slamming you got was enough to make it your last. And without users asking questions, the community evaporates.

But most user communities--especially the new ones--aren't hurting for people *asking* for help, they're in desperate need of people willing to help the newbies. And one of the quickest ways to keep a user community from emerging is when questions go unanswered. So the *real* problem is getting people to *answer* questions.

Encouraging a "There Are No Dumb *Questions*" culture is only part of the solution. What we really need is a "There are No Dumb *Answers*" policy.

The best way to grow a user community is to get even the beginners to start answering questions. The more they become involved, the more likely they are to stick with it through the rough spots in their own learning curve, and we all know that having to teach or explain something to another person accelerates our *own* understanding and memory of the topic. The problem, of course, is that the beginners are... *beginners*. So, here are a few tips used by javaranch, one of the most successful user communities on the planet (3/4 million unique visitors each MONTH):

1) Encourage newer users--especially those who've been active *askers*--to start trying to answer questions

One way to help is by making sure that the moderators are not always the Ones Who Know All. Sometimes you have to hold back the experts to give others a chance to step in and give it a try.

2) Give tips on how to answer questions

Post articles and tips on how to answer questions, which also helps people learn to communicate better. You can include tips on how to write articles, teach a tough topic, etc.

3) Tell them it's OK to guess a little, as long as they ADMIT they're guessing

4) Adopt a near-zero-tolerance "Be Nice" policy when people answer questions

Don't allow other participants (especially the more advanced users) to slam anyone's answer. A lot of technical forums

especially are extremely harsh, and have a culture where the regulars say things like, "If you think that, you have no business answering a question. In fact, you have no business even DREAMING about being a programmer. Better keep your paper hat day job, loser."

5) Teach and encourage the more advanced users (including moderators) how to *correct* a wrong answer while maintaining the original answerer's dignity.

And again, zero-tolerance for a**holes. All it takes is one jerk to stop someone from ever trying it again.

6) Re-examine your reward/levels strategy for your community

Is there a clear way for new users to move up the ranks? Are there achievable, meaningful "levels"?

I'd love to hear some examples of other user communities you think are doing a good job at this. Javaranch isn't perfect, but it's one of the best I've seen (again, all the best stuff there happened *after* I turned it over to Paul Wheaton, so I can't really take credit).

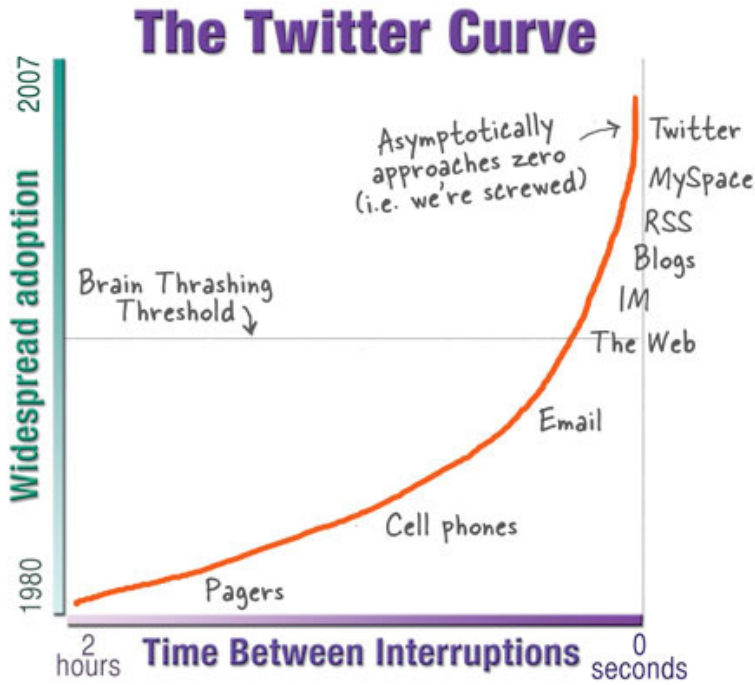
Also, before you point out counter-examples of successful communities like slashdot... remember, I'm talking about *user* communities--people using a particular product or service--and not just *any* community. I'm sure there are tons of, say, political forums where a "be nice" policy is not only unneccessary, but most likely *impossible*.

Your ideas?

http://headrush.typepad.com/creating_passionate_users/2006/12/how_to_build_a_.html

The Asymptotic Twitter Curve

By Kathy Sierra on December 7, 2006



We've all been at the brain bandwidth breaking point for the last five years. Email is out of control. IM'ing sucks up half the day. And how can we not read our RSS feeds, post to our blogs, and check our stats? If my Cingular cell phone sends me a MySpace alert and I'm not there to get it, do I *exist*? But email, IMs, social networking, and blogs are *nothing* compared to the thing that may finally cause time as we know it to cease. I'm talking, of course, about Twitter.

For those of you who don't know about Twitter, it has one purpose in life--to be (in its own words)--**A global community of friends and strangers answering one simple question: *What are you doing?*** And people answer it. And answer it. And answer it. Over and over and over again, every moment of every hour, people type in a word, fragment, or sentence about what they're doing right then. (Let's overlook the fact that there can be only one *true* answer to the question: "I'm typing to tell twitter what I'm doing right now... which is typing

to tell twitter what I'm doing right now." Or something else that makes my head hurt.)

Twitter, it seems, is the solution to the one problem we all have: it's just too damn hard to keep updating our blog every few minutes to tell the world *what we're doing at that very moment*. Twitter lets you make tons of nano-posts (postlets?) to a kind of nano-blog (bloglet?) And indeed, it's every bit as stimulating as it sounds. Here's an ACTUAL SAMPLE from earlier today:

(names removed to protect the utterly bored):

"Missed the bus again."

"Attempting to figure out why the cat is hiding."

"I'm signing off."

"On bus going in to the office."

"Scanning pictures of 12-year old girls in mini skirts..."

"Going to bed now."

"Thinking about eating."

"About to start a conference call."

"I'm watching my dog chase the reflection from his tags and wish I had a laser pointer!"

"Feeling so bored at work I'm going to die. Wonder if it is my attitude or the work."

"Washing hair. Fetching groceries."

And there you have it. But don't take my word for it... go to the Twitter Public Timeline and find out what people are doing... *right now*. Right this very moment.

I'm making fun of Twitter, but this isn't really a funny topic. Moore's law for the brain doesn't quite work. We're evolving much, much, much too slowly... Brain 2.0 isn't coming anytime soon. And we're all feeling the enormous weight of not being able to keep up. We can't keep up with work. We can't keep up with our social life. We can't keep up with the industry, our hobbies, our *families*. We can't keep up with current events. We'll never read a fraction of those books on our list. And we are hurting.

Worst of all, this onslaught is keeping us from doing the one thing that makes most of us the happiest... *being in flow*. Flow requires a depth of thinking and a focus of attention that all that context-switching prevents. Flow requires a challenging use of our knowledge and skills, and that's quite different from mindless tasks we can multitask (eating and watching tv, etc.) Flow means we need a certain amount of time to load our knowledge and skills into our brain RAM. And the more big or small interruptions we have, the less likely we are to ever get there.

And not only are we stopping ourselves from ever getting in flow, we're stopping ourselves from ever getting really *good* at something. From becoming experts. The brain scientists now tell us that becoming an expert is not a matter of being a prodigy, it's a matter of being able to *focus.*

Lots of people are talking about this, and perhaps nobody more eloquently than Linda Stone. Linda talks about the problem of *Continuous Partial Attention*. She says:

"To pay continuous partial attention is to pay partial attention -- CONTINUOUSLY. It is motivated by a desire to be a LIVE node on the network. Another way of saying this is that we want to connect and be connected. We want to effectively scan for opportunity and optimize for the best opportunities, activities, and contacts, in any given moment. To be busy, to be connected, is to be alive, to be recognized, and to matter.

We pay continuous partial attention in an effort NOT TO MISS ANYTHING. It is an always-on, anywhere, anytime, any place behavior that involves an artificial sense of constant crisis. We are always in high alert when we pay continuous partial attention. This artificial sense of constant crisis is more typical of continuous partial attention than it is of multi-tasking."

Read more on her wiki!

But this whole problem is also tied up with the notion of Alone Time, something Jason Fried believes is absolutely essential to both creativity and productivity. I strongly suggest reading his post on How to Shut Up and Get to Work (don't forget to look at the comments).

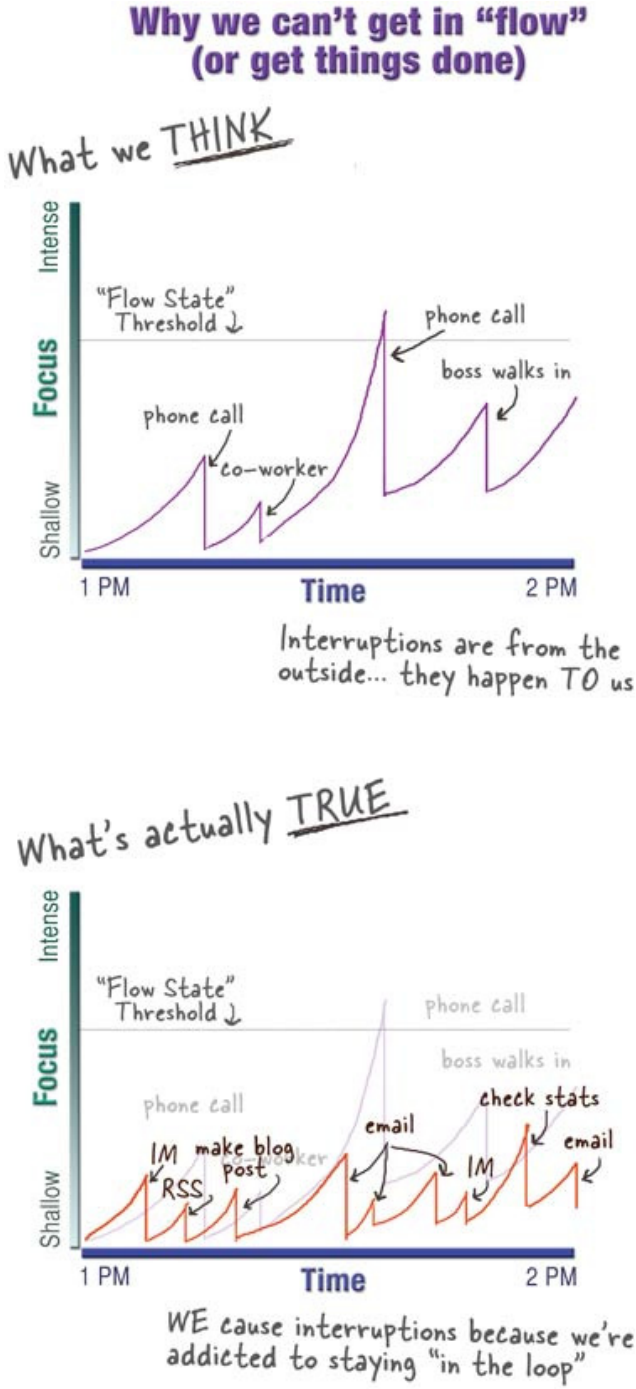
Joel Spolsky also appreciates the value of Alone Time, and makes sure that those working for him have a chance--and a space--in which to *think* without distractions.

And finally, a lot of other people are musing about the effects of Twitter, including Kevin Tofel who wonders the same thing I do--Is it Too Much Information? (The answer, Kevin, is YES. I know enough about the brain and learning to recognize that sucking the last bit of mystery and curiosity out of our lives is not a good thing.) Also Frank Paynter, who talks about the distinction between multi-tasking and Linda's Continuous Partial Attention, and where Twitter might fit in to this.

A few of my earlier posts on this (pre-Twitter, when I still *had hope*) were:

Multitasking makes us stupid? (a follow on to the earlier Your brain on multitasking) and The Myth of "keeping up" (which is where I created the book picture I re-used in yesterday's big book list).

Also, this post helps explain some of the science behind why we really ARE addicted to checking IM, blogs, email, and now Twitter. The most important thing, I think, is to stop being in denial about the profound impact this is having on us and those around us. Until we stop seeing interruptions as something that happens TO us, and understand the role we play in causing them, we're in big trouble.


Why we can't get in "flow"
(or get things done)
What we THINK
Intense
Focus
Shallow
"Flow State"
Threshold
phone call
boss walks in
phone call
co-worker
1 PM
Time
2 PM
Interruptions are from the outside... they happen TO us
What's actually TRUE
Intense
Focus
Shallow
"Flow State"
Threshold
phone call
boss walks in
phone call
check stats
email
email
IM
make blog post
RSS
IM
1 PM
Time
2 PM
WE cause interruptions because we're addicted to staying "in the loop"

Fortunately, there's help... a kind of 12-step program for geeks who want to stay connected but also *get something done* (and without losing our minds completely). While you're out surfing, you might as well check out the tips and techniques on 43 Folders, Lifehacker, and Steve Pavlina.

So, OK, yeah, I stretched a LOT on my Twitter curve (I'm determined to make an asymptotic curve once a year whether I need to or not, and I hadn't met my quota for '06). Obviously the time between interruptions is not asymptotically approaching zero.

Or is it? ;)

[cue end-of-world sci fi music, with maybe a voice-over of Terrence McKenna discussing Time Wave Zero]

[UPDATE: Against my will, I found myself reading the Twitter timeline again after I posted this (I told you it was addicting) and had just about the biggest laugh of the week when I found people Twittering about... this post on Twittering. ;) I love you guys (Sarah and Arabella you made my night!) And I can think of dozens of reasons why Twitter is a wonderful thing (like for separated families, etc.) But talk about an event horizon... Twitter is the new Crackberry.]

http://headrush.typepad.com/creating_passionate_users/2006/12/httpwww37signal.html

Rhythm method 2

By Dan Russell on December 9, 2006

The comments to the first Rhythm Method post are really intriguing. They point out how thoughtful and remarkable the CPU readers really are. It's readers like this that make writing a blog so much fun. It's not just an exercise in writing stuff down, it's a many-way conversation! (If a "dialog" is between 2 people, is this blog a "polylog"?)

If I were to organize the comments into logical clusters, I'd group them like this:

a. **use of rhythms to coordinate actions** (sea shanties to pull together better, precise timing of conversational actions back and forth, entrainment between participants in a group behavior)

b. **rhythm as defining element of flow states** (flow seems to happen with rhythmic patterns)

c. **non-rhythmic events that disrupt behavior**, primarily exogenous events such as hunger (chocolate!) or software imposed interruptions ("you have mail!") and delays (waiting on Perforce).

What I found especially interesting were the practical suggestions of rhythm use. It's obvious what use rhythm has in coordinating teams to work together in close synchrony. But it's also fascinating to think about using rhythm to go around bottlenecks (as Tim O'Reilly speculates about Larry Bird breaking rhythms in driving to the basket around defenders).

This leads me to point out that rhythms have many uses - as organizers of time and as ways of coordinating groups of people (and processes). But if you can't perceive the rhythms, you're in a heap of trouble.

It's clear that some of our tools help to coordinate behaviors - IM, email, calendars - they all help to align people in time. And it's true that every tool seems to have its own natural rhythm: IM beats to a faster pulse than email, and if you've got multiple IMs going, it's often a polyrhythm as beats go against other pulses in time.

And we've all had those moments of sudden expectation in IM or email when someone doesn't respond at the right moment. ("Has she forgotten about me?" you wonder: then a few seconds later, the reassuring IM arrives. Whew!) It's the rhythm telling you when you should begin to worry.

The ability to perceive a rhythm is a fundamental one: as is the ability to generate a rhythm. Yet, it's still a bit of a mystery how people recognize a rhythmic beat. Some things are clearly rhythmic - think of your favorite Sousa march or Ludacris rap - they both pulse and move along regularly, with both short-term and long-term pulses going on in layers. But how, neurologically speaking, does your brain do this neat trick? I'll spare you the various theories, but I don't think we've figured this one out yet.

Regardless of what mystery mechanism we use to pick up on rhythms, it's clear that we humans can detect both simple beats as rhythms (a heartbeat, the gallop of a horses' hooves) and the wonderful layering of many kinds of events over longer periods of time (the rhythm of weather systems moving across the Pacific Northwest, the rise and fall of Orion in the night sky).

And it's clear that we coordinate our actions based on what rhythmic devices we sense and the regular pulse of time we feel. Breaking that pulse is a terrible thing to do.

While latency in responding to a user input is bad manners, creating an unpredictable delay that breaks the perception of rhythm is even worse.

In a kind of extension to the variable reinforcement schedule Kathy discussed earlier, unpredictable delays in response only serve to make the entire experience awful. Passionately bad, in fact.

But note the difference! While unpredictable ***rewards*** are great for training, for a system that requires moment-to-moment interaction, ***unpredictable response times*** are the antithesis of flow. Using such an irregularly reacting system takes up lots of cognitive attention just to recognize when the

next event is going to happen. The user ends up having to be constantly vigilant to know when the next event's going to happen. It's ultimately tiring and a pain to use. Worse of all, the irregularly responding system is *generating interruptions:* that's the one thing we know we **really** shouldn't be doing.

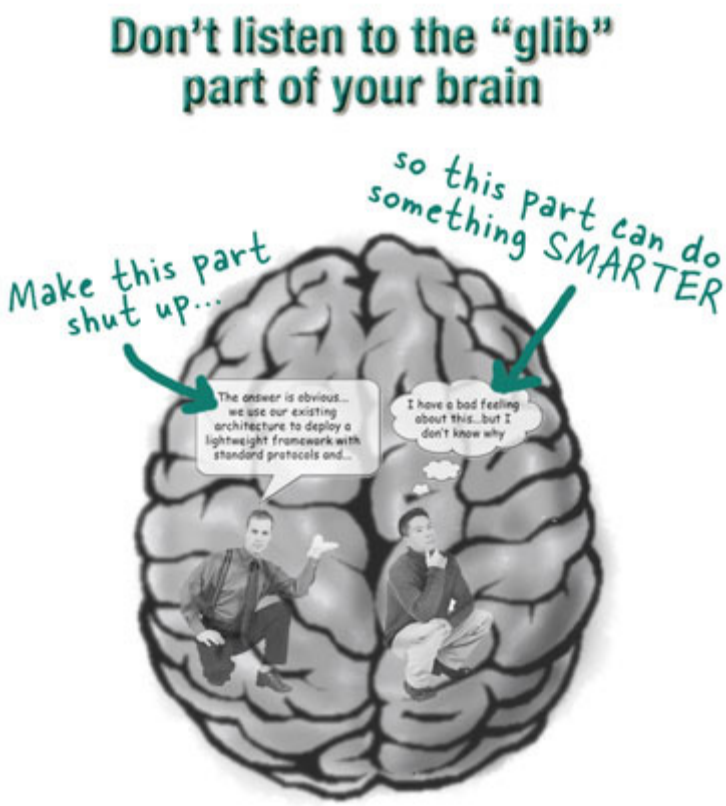
If you (as a designer) have to make the tradeoff, go with the slower, but more regular response so people can entrain their rhythms and get into that flow use condition. That will create the sensation of a more regular and reliably responding system. Faster isn't always better!

http://headrush.typepad.com/creating_passionate_users/2006/12/rhythm_method_2.html

How to be better at almost anything

By Kathy Sierra on December 11, 2006

Earlier we talked about why the fast-talking guy sounds smarter than the guy who understands more than he can say. We talked about how *wrong* that is, and how if the glib *always* win, we all lose. But the more important battle is not between articulate vs. less-articulate *people*... it's between the articulate vs. non-articulate parts of your own head. Your brain has both a quick-talker and a quick-thinker, but the good-talker "know-it-all" gets the glory. In other words, there's a smart part and a dumb part of your brain, and the problem is...the dumb part *talks*.



If we can get the dumber part to STFU, we can learn much faster and perform much better at just about anything. The dumber chatty part is hurting us.

It's the part that makes you self-conscious:

"Do I look OK? Am I going to say something stupid?"

"Am I overlooking something in this code?"

It's the part that criticizes:

"DOH! I can't believe I just did that. Idiot!"

"This code is inefficient... you need to fix it."

"That paragraph reads like a six-year old wrote it. It's dull.

It's the part that gives you "helpful" instructions:

"Make eye contact for three seconds. Watch your posture, don't look at your slides."

"Look for a design pattern to apply. Don't duplicate that code over there."

"Stay close to the fall-line. Don't lock your knees. Turn quicker!"

"It's safer to use formal terminology than risk looking silly."

While your brain is chatting away evaluating, judging, instructing, criticizing, directing, etc... the smarter parts sit in the corner, ignored. Alan Kay--often called the father of object-oriented programming and one of the greatest thinkers/researchers/designers/teachers/engineers of our time-- talks about the implications of this for education... something we talked about earlier.

But Alan Kay was inspired by the work of Tim Gallwey, whose work arose from one simple question, "... is all this inner dialogue really necessary? Is it helping...or is it getting in the way?" Until I heard Alan Kay talk about it (and explain some of the cognitive science behind it), I had always thought Gallwey's "inner game" thing was just one more bit of 60's self-help new-age nonsense. I was dead wrong.

Gallwey showed that the parts of our brain that learn from experience are far more capable than the parts that learn from talking through it. We *think* we need to tell ourselves things like, "keep your weight over your front don't press so hard on the violin bow..." when we're trying to learn something new or improve our performance, when that's exactly the thing that *inhibits* learning and improvement.

We did learn to walk, after all. And we did it with virtually no explicit "talking" instruction. Nobody compared our first steps to the steps of an expert (i.e. a parent) and "told" us how to adjust. Nobody outside or inside our head was evaluating, judging, or correcting. Think about times when people are telling you what to do when you're trying to concentrate and you finally yell at them to STFU. All we need to do is take that attitude we have to people *outside* our head and apply it to the chattering *inside* our head.

Easier said than done, of course. Gallwey makes the point that most of us can't turn off the talking parts with brute force *will*, although that's the basis for so many ineffective self-help or creativity books that tell you to *change* the way you talk to

yourself or "silence your inner critic" simply by *telling* yourself to do so. (pretty tough to tell yourself not to tell yourself...)

For Gallwey, the answer is focus of attention. In tennis, for example, he has people learn to focus on the ball--the seams turning, the way it bounces, and the moment at which someone hits it. Bounce-hit. Bounce-hit. Nothing about feet, arms, rackets, weight shifts. Nothing talking to--or about--you. (Yes, technically your brain is still 'talking' through this "here's the bounce, there's the hit, etc." but the point is that it's not annoying and influencing *you*.)

Example Techniques

* *Art*

Nobody does a better job of this than Drawing on the Right Side of the Brain, a program created by Betty Edwards. I'd recommend it to *everyone* whether you ever care about drawing or not, just for the way it changes your brain. Betty (and a zillion students) demonstrate that the part of your brain that talks is also the part that draws like a three-year old. That part talks its way through, say, what a car or horse or human looks like, and it does a really lame job. But if you get your brain to stop saying, "this is a horse, and they have four legs and..." and instead focus on *seeing* shapes and lines, the better-performing parts of your brain can kick in.

* *Writing non-fiction*

If you plan a book by making outlines, you're indulging the talking (linear, step-by-step, rational) part of your brain. The focus is on what you do and say and when and how you say it. With our books, we do not use outlines--we do everything from storyboards. By focusing on the story of the learner's journey, it keeps the brain focused on the learner's experience rather than what WE do/say/write. This is not a trivial thing--last week our books represented 25% of the O'Reilly Top 20 bestsellers. And we're not all that good at writing. It really *is* about focusing on the reader instead of focusing on what the reader will think of *us*.

* *Design*

Mind-mapping--if you do it *quickly*--stops the talking parts from jumping in and evaluating what you're writing, so the

creative parts can do things. That's what makes mind-mapping so powerful and fun--after you're done, you look at the paper and find things you'd never thought about... things that wouldn't have come out while talking your way through an outline.

Programming

Pay attention to Code Smells, which is another way of saying a gut "bad feeling" that tells you something is wrong even if you can't yet say why.

Everything

Read Malcolm Gladwell's book Blink, which I talked about in this post. And while you're making sure that glib people don't always win, try to do the same within your own brain.

Listen to the comments of our readers, who I'm sure will have suggestions for other resources :)

http://headrush.typepad.com/creating_passionate_users/2006/12/how_to_be_bette.html

Two more words that might change your life (or at least your lunch hour)

By Kathy Sierra on December 11, 2006

Things I learned from my horse trainers #42: practice saying, "Hmmmm... how *interesting*." Say it when you're frustrated. Say it when you're mad. Most importantly, say it *before* you say or do anything else (including hit the "send" or "post" button).

It should be the first thing out of your mouth when things go wrong--or don't meet your expectations--because:

1) It inserts a pause and gives you a moment to *think* before you react.

2) It keeps you from taking things too personally

If someone calls you an idiot (or worse) saying "hmmm...how interesting..." changes your reaction from purely emotional to more curious and detached.

3) It helps you ask more questions instead of jump to conclusions.

With horses, the main goal of the "how interesting" technique is to keep you from losing patience and blaming the horse. If you say "how interesting," it helps you explore reasons, including what your own role in this might be. It makes problems feel more like puzzles.

I learned this trick only a few months' ago, and it helped when I had my little incident with Leira. But it also helped my perspective after my Web 2.0 post. Instead of being purely pissed off and defensive at some of the harsher things said about me on other blogs, for example, I thought about my horses and said, "Hmmmmm... how interesting... " which brought me to a new question, "I wonder what it is about Web 2.0 that leads to such strong emotional reactions in some people...?"

And *that* changed everything ;)

Imagine how it would effect you if you said "hmmm...how interesting" to yourself when a co-worker puts *that* picture of you on Flickr. Imagine saying this when your dog chews your digital camera's USB cable. Imagine saying this when your six-year old calls her teacher an ass. *In class*. Imagine saying this when your girlfriend flirts with your roommate (the one that looks like Brad Pitt). Imagine saying this when your clients make you crazy expecting you to, say, make their marketing "viral". Imagine saying this to the compiler.

Imagine saying "hmmm...how interesting" when you tell someone you're mad at them and they cock their head and say, "hmmmm... how interesting..."

So... in what situations could *you* say "hmmm... how interesting"?

http://headrush.typepad.com/creating_passionate_users/2006/12/two_more_words_.html

Become the thing that replaces you

By Kathy Sierra on December 14, 2006

I asked Little Miss MySpace what happens when something *new* comes along... when someone else makes a MySpace-Killer. Skyler said, "Why does it have to be someone *else*? MySpace can just *become* that (whatever 'that' is)." She knows nothing of the business or politics of MySpace--she's simply a passionate user. And she's never read The Innovator's Dilemma. But she still has a point: why *shouldn't* we be the ones to build our own "killer"?

Whether we're trying to innovate around our existing products and services or trying to find a completely new idea, we have to back up to the meta-level rather than focus on implementation. Obviously implementation matters... a *lot*. But implementation of *what?* Why build a better XYZ if all that matters to users is

the Z? What if XYZ is just one way to give users what they *really* want--JKL--and there's actually a much better way to help them do JKL? A way that makes XYZ unnecessary?

If we're not careful, we can take our existing success and misattribute it to an implementation detail that was never important.

Or worse... we can misattribute our success to something that's actually a *problem* but that users managed to cope with. Right up to the time we *upgraded* that thing-they-never-liked to give it a bigger role.

Yes, this is just another one more of those DUH topics, but as with so many others--it's too easy to get sidetracked by either our own success or the success of someone else's product or service that we're trying to build a better version of. And that's one reason why trying to reverse-engineer the success of a product is tricky. We get stuck rationalizing why some implementation detail is important, when it may be nothing more than noise.

An example of outstanding implementation that ignored the meta-level

The now-dead Purple Moon software company was the result of millions of dollars and years of research at Paul Allen's Interval Research think tank. They had finally found the secret sauce to getting girls--the great untapped market--into gaming. By the time the company's first product launched (1997), they knew just about everything you *could* know about what young (10-14 years) girls wanted and how they differed from boys. So they took their exhaustive and expensive research effort and created the ultimate implementation.

The implementation was awesome... beautiful graphics, clever characters and story, slick marketing, and a world-class leader who many of us still practically worship, Brenda Laurel. If *anythng* could finally bring girls to the games, it would be the perfectly-pedigreed Purple Moon's first game, Rockett's New School.

Except it sucked.

At the meta-level.

Because what *is* the meta-level for a game? Oh yes, *fun*. Purple Moon got the individual implementation details right, and

applied all that they'd learned from the research, but forgot the forest. Fun.

Skyler was an early beta-tester, and had been looking forward to the game I'd been hyping for so long. But the first thing she said when she read the overview document was, "Why would I want to play a game about a girl trying to fit into a new school? HELLO! I've DONE that in real life and it wasn't fun then. I'd rather play Blobbo."

[Warning: gratuitous kid photo... this is Skyler *today*]

[Fortunately, she reads this blog only twice a year--she'd kill me if she knew I posted this. Let's just keep it our little secret.]

Granted, Skyler wasn't the typical pre-teen. She didn't do Barbie. (She would have given a kidney for her My Little Pony collection, however). But still, when you strip away Purple Moon's research and implementation details, Occam's Razor applies: **Just. Wasn't. Fun.**

With our books, that meta-thing is *learning*. And if we get off track by focusing on and EQ'ing our implementation details without remembering that, we're sunk. So if we try to figure out

our own "killer", we'll do it only by staying true to the meta-level forest.

Many of us are creating products or services where the barrier to entry for a competitor is not all that high. The only thing we have to really protect us is a willingness to throw out even our most successful products in order to build a better reflection of what matters to users at the meta-level. And that might look *nothing* like our current, successful product. Keeping focused on meta-levels is also the key to avoiding being trapped by fads or fashion. Fads and fashion ("rounded", "glossy", "extreme", "twittery" [sorry, couldn't resist ;)] tend to be implementation details, not meta-level concepts ("have fun", "kick ass", "be smarter", "have more time in flow", etc.)

Finding the meta-level

The best trick we know for finding the meta-level is to play the five-why's / why-who-cares-so-what game. Ask your users (or even just yourself) what's important about a product. When they answer, ask, "Why?" When they answer *that*, say, "So?" and when they answer *that*, say, "Who cares?" and keep going until you get to the heart of it. (And if you haven't played this before, most people stop WAY too early and miss what matters the most.) Only then do you discover that this feature the users--and you--believed to be meaningful was simply a tolerable way to do what they *really* wanted. When they say that X is important, dig deep enough and you might find that it was only because X let them do Z, and that there's a much better way to make that happen.

Again, I know we all know this. But it's so hard to do, and the more successful your product or service, the harder it becomes. "Don't mess with success" is often the biggest barrier to becoming your own "killer".

A prominent tech book author wrote on a public forum, "Your Head First books will be fine just until the next hot new thing comes along to replace it." I said, "Yes, and that's why I want to be the one to replace it."

http://headrush.typepad.com/creating_passionate_users/2006/12/become_the_thin.html

Tech t-shirts aren't sexy enough

By Kathy Sierra on December 15, 2006

I've been to seven JavaOne conferences. I've paid more than $10,000 of my own money, *just for the attendance fee*. You'd think--just once--they'd give me a show shirt that didn't hide the fact that I have, say, *breasts*. You'd think--just once--they'd take part of the $2000 entrance fee and spend, oh, .1% extra to print up some shirts that sub-6-foot folks can wear. And it's not just Sun's JavaOne show, of course--practically every tech company out there is guilty. If I had a dime for every booth vendor who's smiled and said , "Here, you can sleep in it!", I'd be typing this from my ocean-front villa. (Pssst--tech companies: most of us women don't sleep in *anything*, but I digress...)

The formula we've done to death on this blog is pretty simple:

How are you helping your users kick ass?

I put "helping them look good" in the "kicking ass" category.

But that's not even the point. The point is showing us that you *care* about more than just saving a few bucks on a t-shirt print run. That you care about ALL your users, not just the Big Burly

Men. And even if you do *not* care, you'd think the marketers would get a clue that people aren't going to be wearing your logo around giving you free advertising if the shirt doesn't fit.

The bar's been set pretty low on this, so even a MEN'S SMALL would make me happy. But Webstock went all the way to give the gals *women's* shirts. I actually wear mine all the time. I've even been photographed wearing it at *another* conference.

I so don't want a lecture on logistics or saving money by making shirts for the largest common denominator. And I don't want to hear that, after all, it IS mostly men at these things. So *what* if you have some leftover shirts? Give them out at other tech events. Send them to user groups. *Donate them to a homeless shelter*.

Yes, you could argue that as a web-focused show rather than a pure programming event, Webstock was likely to have more women than JavaOne, so it made sense. And that's true, but doesn't explain why I also got a fitted, flattering, rather sexy blue tee at GUADEC (the GNOME user's and developer's european conference) which was not expecting but a very few women attendees. But they treated us like we mattered too. Like we weren't the tacked-on not-really-target-audience people. Besides, this isn't even a gender thing... it's a SIZE thing. There are plenty of men who don't look that much better in an XXL Hanes Beefy T than I do.

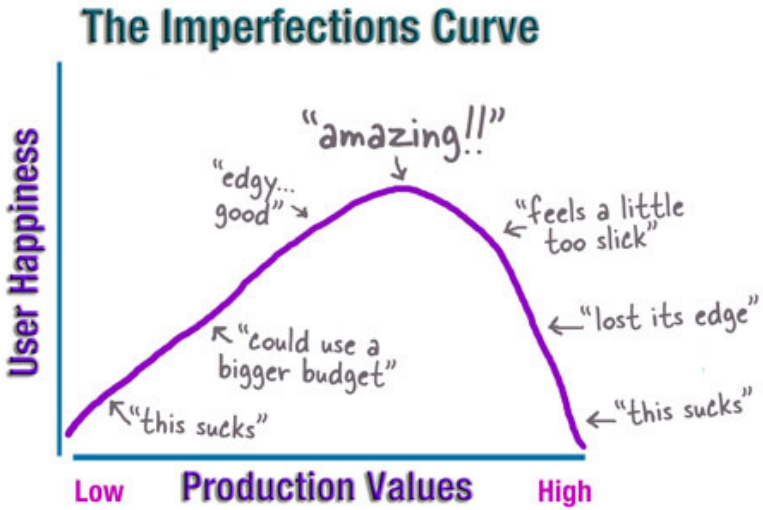
This is partly tongue-in-cheek, but still...the t-shirts are a metaphor for--or at least a reflection of--the way the company feels about users as individual people. The shirts *matter*, and they speak volumes about your company.

And hey, tech companies, I AM available to beta test your freshly-minted women's T's (size small or x-small). In fact, for any tech company that tells me they'll be keeping plenty of women's shirts on hand for trade shows, user groups, etc., I'll post a picture of the shirt on this blog. But it better make me look good. ;)

http://headrush.typepad.com/creating_passionate_users/2006/12/tech_tshirts_ar.html

Sometimes the magic is in the imperfections

By Kathy Sierra on December 19, 2006



What makes indie films more appealing than so many of the huge Hollywood productions? What makes indie music more interesting than the slick big label big production records? What's the magic that disappears when you hear the studio-mix version of something you once heard live? Not that most of us have the problem of too big a budget for our own good, but still... maybe we should think about whether some imperfections might be a *good* thing. Maybe we should consider whether we're trying too hard to smooth all the rough edges.

I'm not even sure this applies to much *other* than music and movies, but it's definitely a big deal there. A couple examples:

David Gray

His breakout hit, Babylon, was on the album *White Ladder* -- an album he produced mostly in his apartment, using electronic sounds to back up his acoustic guitar and piano. One of the *best* parts of that record was the combination of cheesy drum machine sounds and the faint hint of traffic noise from the street below. It definitely had that home-made, soulful feel.

Fast forward-- he gets big and ultimately his last album was a big, slick, high production value studio recording that sucked the life right out of the music. (In my opinion)

Howie Day

If you had a chance to see this kid play live during the 2000-2001 time especially, you know what I'm talking about. I saw him at a small, used record store in Denver the first time in 2000, and he walks out with an acoustic guitar... and some strange foot controller devices. He constructs the songs in realtime, using looping and other effects to layer in percussion, other voices, different guitar parts, etc. It was captivating. But then his records--where he has an Actual Band to do the work--all sound like so much pop music crap. It lost the imperfections that came with building a song on-the-fly with one person and some gear.

I happened to see Thomas Dolby the other night, in the fantastic DeviantART show with BT, and Dolby also does a lot of his music through layering in pieces in realtime. I've never much cared for his music--certainly not enough to buy it and play it at home--but his show was amazing, and now I'm much more interested in his records. One of the really fun things in his show was his head-cam-- you get to see what he's looking at when he's fiddling the midi controls, switching rapidly between various input devices, etc.

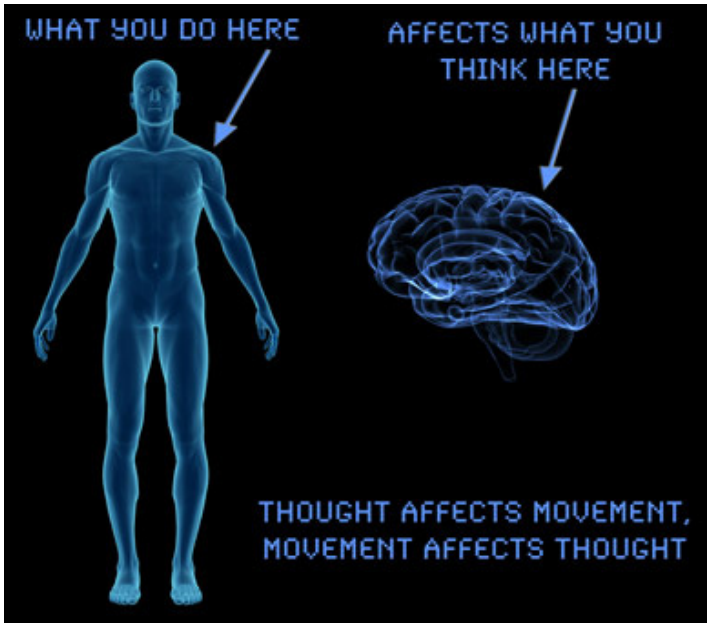
If you happen to be near a city where the tour is (MD, PA, VA, NY are the remaining shows, I think), you should check it out. And the BT show will make your head explode in a good way.

I'm not sure how much this notion of overproduction vs. imperfection applies to other products, but I suspect it does, depending on how you define perfection. A typo in a book would be a *mistage*, not an imperfection that gives it life. But a more casual tone that occasionally violates your fourth-grade Grammar Rules might be just the imperfection it needs. I don't know. What do you think? Is there an "indie sensibility" that applies to other things besides music, film, and fashion?

http://headrush.typepad.com/creating_passionate_users/2006/12/sometimes_the_m.html

What you DO affects what you THINK

By Kathy Sierra on December 22, 2006



What our users DO with our products--or even what they see someone *else* do--has a bigger effect on their brains than we might believe. How we move (or imagine moving) our bodies changes our thoughts. And if there's a mismatch between thought and physical action, brains don't like it. Whether you're designing interfaces or instructional materials, you can't afford to ignore the research on this.

The rest of this post won't make sense unless you do the following exercises, so... you've been warned.

EXERCISE ONE:

1) Say the word "zeal" out loud. Twice, clearly.

2) Without changing the position of your mouth and lips, *imagine* yourself saying "zeal." Make sure you "hear" it.

3) Now--this is the important one--**open your mouth as wide as you possibly can**, keep it open, and imagine yourself saying the word "zeal."

So, what happened when you tried to imagine saying "zeal" with your mouth wide open? Did the state of your physical body--in this case your mouth--affect your ability to *think*?

EXERCISE TWO:

1) Tighten your whole body, grit your teeth, clench your jaw, tense the muscles in your shoulders and arms, and clench your fists. Hold that position and imagine yourself pushing a piece of big heavy furniture across the room.

2) Now relax all your muscles. Let them go as limp as you can, unclench your jaw, relax the face, shoulders, hands, all the way down. Picture yourself lying on a beach listening to the sound of the waves. KEEP THAT RELAXATION in your body for the next step.

3) Holding that completely relaxed position, imagine yourself pushing a piece of big heavy furniture across the room. No matter what, do NOT let your muscles tense up.

What happened? Did the state of your muscles affect your ability to *think*?

Both of these exercises are from neuroscientist Richard Restak's latest book, The Naked Brain. From the book:

"Our mental processes are sufficiently tethered to our bodily senses that we have difficulty with situations when the brain and other parts of the body aren't in sync."

Even more dramatic, in a test measuring the association of arm posture and attitude, he demonstrates that the way you hold your arms affects how you feel about items you encounter. Apparently even just *imagining* items while your arms are extended in the "pushing away" position can cause you to like those items less than if you imagined them while your arms were in a flexed "bringing-to me" position.

One experiment he describes:

"Half of the participants in the experiment were asked to push a lever away from them if they reacted positively to a particular word but pull it toward them if the word gave rise to negative associations, while the other half of the participants were told to do the opposite, pulling forward with positive words and pushing away with negative words. Overall, people were faster to respond to positive words when they were

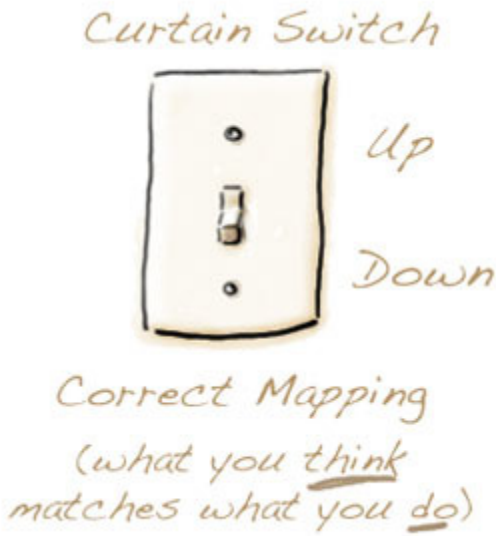
pulling instead of pushing the lever, and faster to respond to negative words when they were pushing rather than pulling the lever."

In another experiment, half the participants were told simply to push a lever the instant they saw a word on the screen, regardless of any like or dislike. The other half were told to *pull* a lever the instant they saw a word. You can imagine what happened... those asked to *push* the lever reacted more quickly to negative rather than positive words, and those asked to *pull* reacted more quickly to positive words.

And of course scientists have found evidence now that simply *thinking* about an action--or watching someone else do the action--activates the same brain regions that would be involved in actually *doing* it yourself.

I'm sure you can imagine the implications, including one of the key design principles known as natural mapping, outlined so well by Don Norman in The Psychology of Everyday Things (the title was later changed to "The *Design* of Everyday Things).

A quintessential example of mapping for those new to design:



A switch for moving something up or down should have "up" position move the thing up and the "down" position move the thing down. The most ridiculous example of bad/incorrect

mapping would be to reverse that--the "up" switch position moves the thing down, and vice-versa.

Usually we think of mapping in the context of usability and "mental models"--the most *natural* mapping helps the user intuitively *do the right thing* without having to consciously think, learn, or remember the switch positions. But Restak's brain/body link goes beyond just mapping, and into attitudes. And when you factor in mirror neurons, then even just the pictures you use on your website can matter.

Consider the following two pictures, and think about your feelings related to each one:

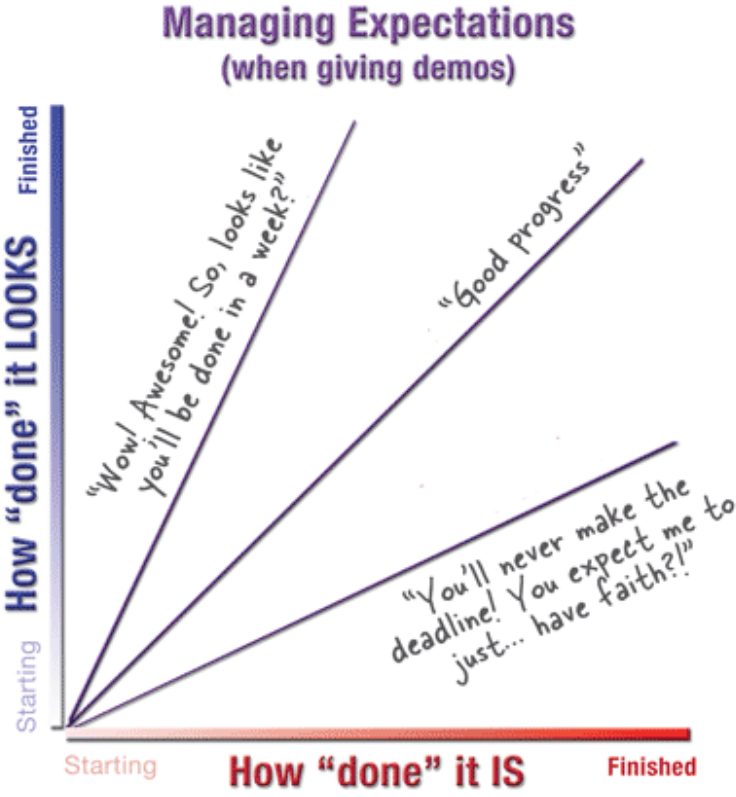
Even if you don't *consciously* notice anything significant, your brain is still doing that "pushing away = dislike, bringing in = like" thing. (Technically, this is an unfair apples-to-oranges comparison because I was forced to switch genders (I couldn't find suitable pictures of two men or two women), but you still get the idea.)

I'd very much like to hear your thoughts about the body/thought connection, and how it might relate to the kinds of work we're all doing...

http://headrush.typepad.com/creating_passionate_users/2006/12/what_you_do_aff.html

Don't make the Demo look Done

By Kathy Sierra on December 27, 2006



When we show a work-in-progress (like an alpha release) to the public, press, a client, or boss... we're setting their expectations. And we can do it one of three ways: dazzle them with a polished mock-up, show them something that matches the reality of the project status, or stress them out by showing almost nothing and asking them to take it "on faith" that you're on track.

The bottom line:

How 'done' something *looks* should match how 'done' something *is*.

Every software developer has experienced this many times in their career. But desktop publishing tools lead to the same headache for tech writers--if you show someone a rough draft that's perfectly fonted and formatted, they see it as more done

than you'd like. We need a match between where we *are* and where others *perceive* we are.

Joel Spolsky talked about this way back when in The Iceberg Secret, Revealed. The secret:

"You know how an iceberg is 90% underwater? Well, most software is like that too -- there's a pretty user interface that takes about 10% of the work, and then 90% of the programming work is under the covers... That's not the secret.

The secret is that People Who Aren't Programmers Do Not Understand This."

He goes on to add corollaries including:

"If you show a nonprogrammer a screen which has a user interface that is 90% worse, they will think that the program is 90% worse."

and

"If you show a nonprogrammer a screen which has a user interface which is 100% beautiful, they will think the program is almost done."

You'll have to read the rest to get the other corollaries, and to see where else he takes the topic.

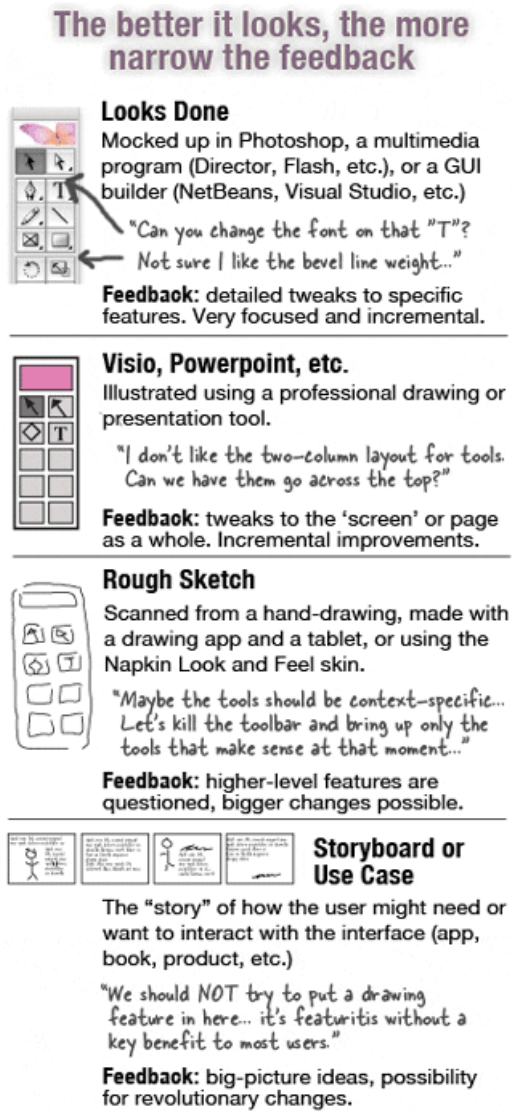
~~My First~~ Robert Scoble talked about this recently, in a story about the early *fake* prototypes of Vista:

"Later... I found out that all we really saw were Macromedia Director-based movies. They looked so cool...how good they made us feel... This actually was NOT a good thing for Microsoft...when you build up expectations and you aren't able to meet them you look pretty silly.

But behind the scenes things were even worse. Why? Because executives bought into the Flash and Mirrors song and dance too. They thought what they were seeing was possible... it's very possible that what you are dreaming of is simply not possible."

So, overpromising by delivering a flashy (or Photoshopy or Powerpointy or Visioy) demo is tempting, but it's short-term gain (you're a hero to your client, boss, the public) with long-term pain. But there's another problem with overdone demos that's just if not more damaging than wrong expectations:

The more "done" something appears, the more narrow and incremental the feedback

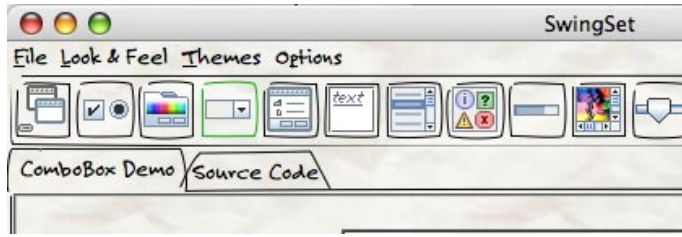


We see this with books and software all the time. Show them something polished and pretty, and you'll get feedback on font sizes. The reviewers make incremental tweaks, blinded by what's in front of them. But show a napkin sketch, and they don't just see what's *there*, they see what's *possible*. Obviously you need to tell reviewers about the kind of feedback you DO want at this

stage... you don't *want* big-picture-forest feedback when you've really moved on to the tree stage. My point is: all you'll *get* is tree-tweaks when you show something finished-looking, so if you want big picture, make it fuzzy!

Finally, it's great to know that there are tools to help make the *look* match the *state*, with my favorite being the Napkin Look and Feel, a GUI "skin" for Java that makes the interface look--quite literally--like it was scrawled on a napkin. I think it's brilliant, and creator Ken Arnold (Java guru, fellow Sun refugee) paid astonishing attention to detail. For example, if the radio buttons were all exactly the same, no matter how sketched they look, their exact sameness would break the spell. So, there's more than one of nearly all of the GUI components from sliders to buttons to tabs to...

Here's just one picture, but I urge you to go check out the snapshots on Sourceforge or even better, try the actual Java demo (you can get to the demo from the link above).



I realize that there are a million exceptions and caveats (like, for example, when you'll be *fired* if you don't show something jaw-dropping early on), but in general, the more closely what you SHOW matches what you HAVE, the more likely you are to have less pissed-off people down the road, *and* the more likely you are to get much better feedback, at the stage you need it. Bottom line: when it's an early demo, think fuzzy. Think sketchy. Think underpromise-and-overdeliver.

[Related links:

* 37Signals blog talks about how to make sure you fix the 'placeholder' stuff before the final release

UPDATE: flow/state discusses the same thing in Matching Design sketches to the desired level of design feedback]

http://headrush.typepad.com/creating_passionate_users/2006/12/dont_make_the_d.html

Five(ish) Things I Don't Know About You

By Kathy Sierra on December 30, 2006

Imagine you want to get to know someone and you can ask only five questions. What would you ask? What would you most want to know? Obviously it depends on the context--you'd ask different questions of someone you're dating vs. a job interviewee vs. a customer vs. someone your *daughter* is dating. But I wonder, should the questions we ask our *users* be that different from the ones we'd ask our *dates*?

Think of all those surveys you've taken. If you're signing up for a conference, magazine, or online news site, you usually get things like:

1) What do you do for a living? [Choose an industry and job title]

2) How big is your budget?

4) What are your purchasing plans for [whatever the domain is, with timeframes]

3) Are you "the decider"?

In other words, "How useful are you to our advertisers and sponsors?" Those surveys say to me, "The only thing we care about is how much you can buy."

If it's a customer survey, you often get things like:

1) How old are you?

2) What gender are you?

3) How old are your kids?

4) What's your total household income?

5) Do you own or rent your home?

In other words, "The only thing we care about is selling more things to you and other people who fit your demographics."

Those questions tell you little about me as a *person*. If we want passionate users, shouldn't we care about what *they* care about? Obviously there are personal questions that might not be appropriate for customers, but most of us here are trying to have a more personal connection with users, and that means doing more to get to know them as *individual people*.

By now, you've probably seen the "Five Things You Probably Don't Know About Me" meme floating around, which I've enjoyed reading. And several *wonderful* bloggers have tagged me for this including:

Joe McCarthy

passionate analyst Dylan Lewis

Cool Cat Teacher (aka Vicki Davis)

Tamar Weinberg

Passionate Communicator Lee Hopkins

and the must-must-must read creativity guy Roger von Oech

Well, I'm much more interested in knowing something about *you*--our wonderful readers we learn so much from. So, I'm asking *you* for a huge favor--to answer some or all of the following five questions here in comments--or on your own blog (please let us know). Which brings me back to the start of this post, *which five questions should I ask?* Normally my first question would be, "Which books do you wish others would

read?", but fortunately I've already asked y'all that and got a fantastic reading list. But, here are the other five(ish) things I'd love to know:

0) What's your name and website URL? (optional, of course)

1) What's the most fun work you've ever done, and why? (two sentences max)

2) A. Name one thing you did in the past that you no longer do but wish you did? (one sentence max)

B. Name one thing you've always *wanted* to do but keep putting it off? (one sentence max)

3) A. What two things would you most like to learn or be better at, and why? (two sentences max)

B. If you could take a class/workshop/apprentice from anyone in the world living or dead, who would it be and what would you hope to learn? (two more sentences, max)

4) A. What three words might your best friends or family use to describe you?

B. Now list two *more* words you *wish* described you...

5) What are your top three passions? (can be current or past, work, hobbies, or causes-- three sentences max)

6) (sue me) Write--and *answer*--one more question that YOU would ask someone (with answer in three sentences max)

[Bonus: What is one question you wish people would ask *themselves*?]

Thanks! And best wishes to y'all for a wonderful 2007. I know 2006 sucked for many of us, but the new year is a powerful metaphor for 'starting new.' At the very least, you get to start a new calendar and crack open a crisp, fresh, Moleskine ;)

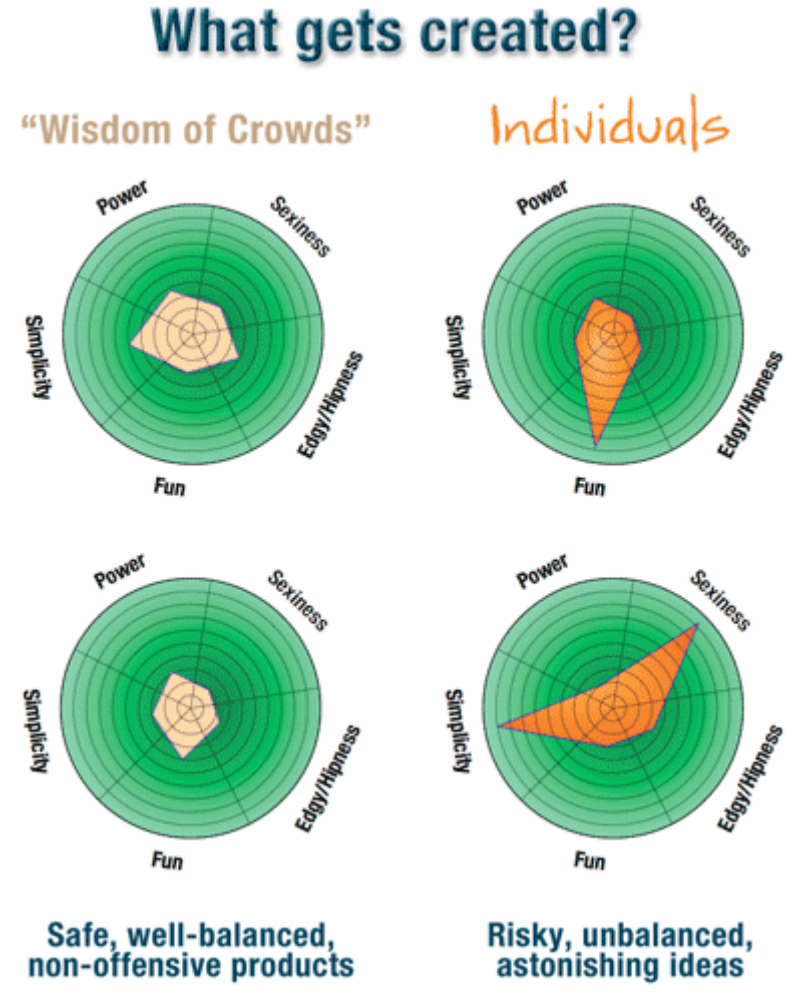
[Photo of Daniel knitting at Foo Camp by Brian Sawyer]

[Photo of *me* is so old I can't remember who took it. And would you get a look at the size of those *trucks*? Not to mention the 80's hairstyle...]

http://headrush.typepad.com/creating_passionate_users/2006/12/fiveish_things_.html

The "Dumbness of Crowds"

By Kathy Sierra on January 2, 2007



Community. Wisdom of Crowds. Collective Intelligence. The new emphasis on net-enabled collaboration is all goodness and light until somebody gets an ~~eye~~ I poked out. Is it merely a coincidence that Apple, run by (as James Gosling put it) "a dictator with good taste" leads the way in tech design, while risk-averse companies using design-by-committee (or consensus) are churning out bland, me-too, incremental tweaks to existing products? And if that's true about *companies*, why do

we think consensus will work on an even larger scale with "users" in Web 2.0?

Jaron Lanier, in his controversial Edge essay Digital Maoism, has a great quote:

"In the last year or two the trend has been to remove the scent of people..."

All geeks-and-personal-hygiene-jokes aside, *we need the smell.* And the most frustrating part for me is how the "Wisdom of Crowds" idea has been twisted and abused to mean virtually the opposite of what New Yorker columnist James Surowiecki says in the book of the same name. He opened a talk at ETech telling us that while *ants* become smarter as the number of collaborators increases, *humans* become dumber. In what is potentially the most misleading book/idea title in the history of the world, the "Crowds" in "The Wisdom of Crowds" was never meant to mean "mobs", "groups acting as one", "committees", "consensus" or even "high collaboration".

By "crowd,", I think he meant "more people", sure, but he also defined a big ol' set of *constraints* for how much togetherness people can have before the results became dumber. And it turns out, not that much. By "crowd", he was referring to *a collection of individuals*. Individuals whose independent knowledge (and "independent" is a key word in what makes the crowd "smart") is aggregated in some way, not smushed into one amorphous Consensus Result.

Web 2.0 and putting the Community in Control

One of the high-profile concepts of the Web 2.0 meme is *community*. Giving community the control. Letting the community make decisions. Trusting the community. And--if you're a lucky bubble-2.0er--*letting the community do all the work while you collect the money*. But this idea of consensus-community is not at all what I've heard Tim O'Reilly talk about when he uses the phrase, "harnessing collective intelligence" or when he describes Web 2.0 as something whose value *to users* grows with the number of users.

What's the difference between Collective Intelligence and Dumbness of Crowds? A few examples:

"Collective intelligence" is a pile of people writing Amazon book *reviews*.

"Dumbness of Crowds" is a pile of people collaborating on a wiki to collectively author a *book*.

(Not that there aren't exceptions, but that's just what they are--rare exceptions for things like reference books. I'm extremely skeptical that a group will produce even a remotely decent novel, for example. Most fiction suffers even with just *two* authors.)

"Collective Intelligence" is all the photos on Flickr, *taken by individuals on their own*, and the new ideas created from that pool of photos (and the API).

"Dumbness of Crowds" is expecting a group of people to create and edit a photo *together*.

"Collective Intelligence" is about getting *input* and ideas from many different people and perspectives.

"Dumbness of Crowds" is blindly averaging the input of many different people, and expecting a breakthrough.

(It's not always the averaging that's the problem it's the *blindly* part)

"Collective Intelligence" is about the community on Threadless, voting and discussing t-shirts *designed by individuals*.

"Dumbness of Crowds" would be expecting the Threadless community to actually *design* the t-shirts together as a group.

Art isn't made by committee.

Great design isn't made by consensus.

True wisdom isn't captured from a crowd.

At least not when the crowd is acting as a single entity. Clearly there IS wisdom in the many as long as you don't "poison" the crowd by forcing them to agree (voting doesn't mean agreeing). According to Surowiecki, even just sharing too much of your own specialized knowledge with others in the group is enough to *taint* the wisdom and dumb-down the group.

It's the sharp edges, gaps, and *differences* in individual knowledge that make the wisdom of crowds work, yet the trendy (and misinterpreted) vision of Web 2.0 is just the opposite--get us all collborating and communicating and conversing all

together as one big happy collborating, communicating, conversing *thing* until our individual differences become superficial.

Imagine a community--let's say the Dog Lover's Society--that through a genetic breakthrough is given the chance to design *the perfect dog*. Everyone gets to contribute. Everyone's idea counts. The dog will be the perfect reflection of the wisdom of the dog-loving crowd. What will they come up with?

I see two options:

1) A non-descript Generic Dog--the average of every possible dog attribute. It would look something like the abstract DOG used in pre-school books where you teach two-year olds to "point to the DOGGY"

OR

2) Frankendog--a hideous patchwork of dog parts that were *never* meant to go together. It would look something like a Star Trek transporter accident.

Of course most of what I've been dissing is the popular, rampant misinterpretation of Wisdom of Crowds, not what Surowiecki actually meant. Read the book and you'll see just how significant

and powerful the *aggregation of individual knowledge* really is, and how in the right circumstances with the right constraints, the wisdom found in that group CAN be smarter than the smartest individual in the group. But he never says the group itself *becomes* smarter when they work together to produce a result *as a group*.

20Q - a perfect example of the *real* Wisdom of Crowds

If you're one of the twelve people who haven't yet played with the 20Q "toy", you have no idea how scarily well this thing "guesses" what you're thinking about. The creepy thing isn't necessarily that it figures out you were thinking of thermometer, bra, microscope, painting, mp3 player, or lightbulb (all things it guessed correctly for me yesterday). The *really* creepy thing is how it got there from the questions it asked. Although the program clearly changes questions based on your answer to the previous questions, it doesn't change them nearly as much as you'd think it would need to.

If I didn't know better, I'd swear it's using voice recognition to cheat. But no, it turns out there's a perfect explanation for its supernatural accuracy. The creator harnessed collective intelligence. Hundreds of thousands of people "taught" the program over a period of years, by playing software versions of his game. The program uses a neural net, and learned. It learned so well, in fact, that it learned a few *dumb* things. For example, way too many people think a dolphin is a fish, so even if you say "yes" when asked by the device if what you're thinking of is a fish, *it can still figure out you mean dolphin from other cues*. (Apparently it also thinks human beings may not actually be animals, based on the "collective intelligence" of those who've played the online version.)

Finally, while I disagree with much of what Jaron talks about in his essay, I know how damn smart this guy is (we were on a panel together a long time ago at the Junos in Canada). A few of my favorite quotes:

> *"Meanwhile, an individual best achieves optimal stupidity on those rare occasions when one is both given substantial powers and insulated from the results of his or her actions.*

"If the above criteria have any merit, then there is an unfortunate convergence. The setup for the most stupid collective is also the setup for the most stupid individuals.

"Every authentic example of collective intelligence that I am aware of also shows how that collective was guided or inspired by well-meaning individuals. These people focused the collective and in some cases also corrected for some of the common hive mind failure modes."

No matter what, I believe that in our quest to exploit the "We" in Web, we must not sacrifice the "I" in Internet.

http://headrush.typepad.com/creating_passionate_users/2007/01/the_dumbness_of.html

Reverse-engineering user reviews

By Kathy Sierra on January 5, 2007

Which would you rather hear a user rave about in a review... your *product* or your *company*? How should they describe you? I've seen a lot of startups analyze their competitor's *bad* reviews... to look for opportunities where the competitor's product or company is screwing up. But we can learn even *more* from analyzing the *good* reviews. This little exercise has made a huge difference for us, and it might help you too.

The homework assignment we give our new authors before they come to a bootcamp workshop is this:

1) Write your ideal review--the detailed review you'd most like to see from a user.

2) Analyze the positive (in this case, 5-star only) reviews of one of the other books in the series then analyze the positive reviews of the closest competitor for that book. (Make sure you pick one that has at least 40 reviews to get more data).

3) Describe how closely each set matches your ideal review, and what the differences are.

4) Describe any differences between the two sets.

We believe this homework has been been the single most important part of our process, although we often do a several-hour debrief/discussion about what they come up with. And of course when we're done with the exercise, the rest of our effort is in figuring out what to do to *cause* us to get those ideal reviews.

YOUR CHALLENGE

Try doing just #1, right now. I mean it, stop and write your ideal user review. Make it detailed, at least a few paragraphs. Think about it.

I'll wait.

I'll trust that you've done it ;)

Now, did your ideal user review focus on the product or the company? What did they talk about?

SECRET ANSWER C

I'm sure most of you already know that the question was a trick. We don't want our users talking about the company *or* the product. All that matters is how they feel about *themselves* as a result of interacting with our product. How they feel about *us* has little impact on whether they'll become loyal (let alone passionate) users. All that matters is what we've helped *them* do or be.

So, when you analyze user reviews, **look for first-person language.** Look for the word "I". Do a statistical analysis on the number of times users talked about something *they* were able to do as a result, rather than a run-down of oh-how-great-this-company-is. View your competitor's positive reviews the same way.

For us, we look for other book-specific things as well. For example, we analyze the number of times reviewers use the author's first name vs. last name vs. both, and then we compare that to our competitors. I'll leave you to consider why that matters to us. We also look for more emotionally-tinged language.

HAVE YOUR EMPLOYEES DO THIS EXERCISE

Everyone connected with your product or service should sit down and write their ideal user review. And if it's nothing but raves about how fabulous the product or company is, there's a problem. In our case--with technical learning books--an author with a goal of what readers will think about *the book or the author* can lead to a book that's bad for the user (too advanced, too much content, etc.)

If you're creating something to win awards, or to impress people, or to gain praise and recognition, that might lead to an award-winning, impressive product that leaves the user behind. I hear a lot of companies claim to care about what the user *thinks*, but they're still focused on what the user thinks of them or the product. I don't want people to praise *us*. I want them to *thank* us for helping them earn the praise of others.

I mean it. Have your employees do this exercise. Count the number of times the employee's ideal review includes the word "I" (or rather, does NOT include the word "I"). It's enlightening.

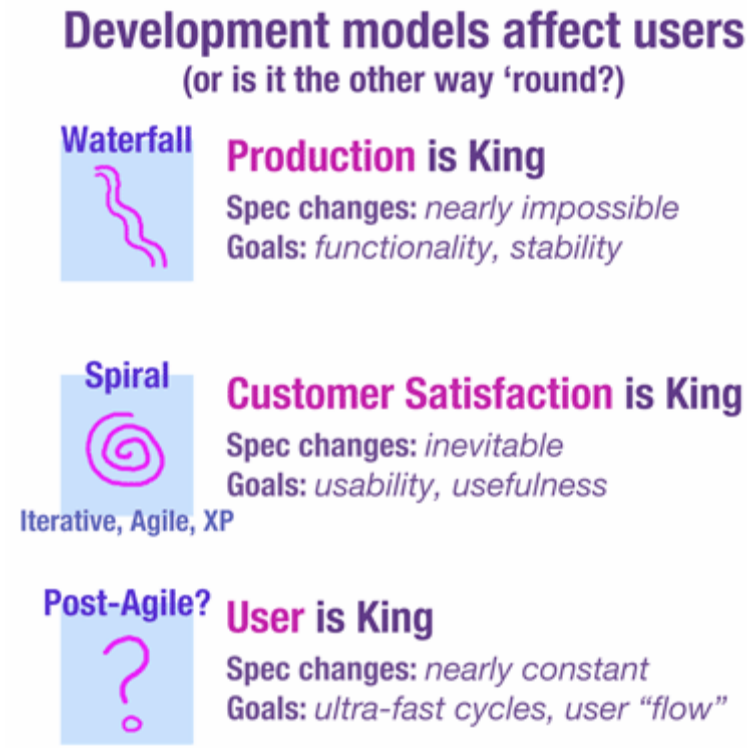
[bonus link: Tom Asacker has a lot of great things share on this topic. Including books.]

[And yes, I know I've talked about this exact thing before, but hey -- it's a whole new year. And a new picture!]

http://headrush.typepad.com/creating_passionate_users/2007/01/reverseengineer.html

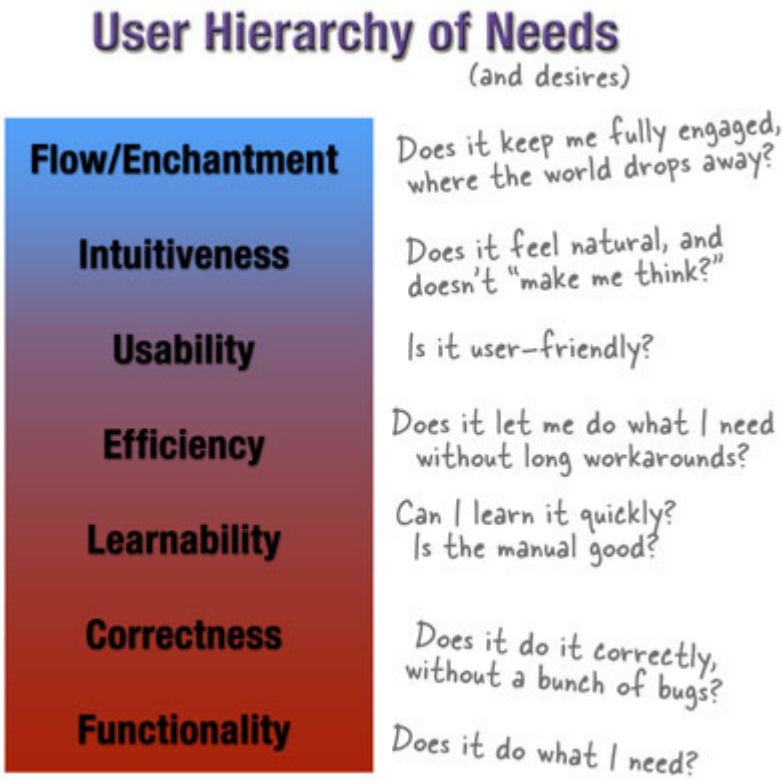
What comes after usability?

By Kathy Sierra on January 7, 2007



The software development process usually drives what users get. In the beginning, there was the Waterfall model based on a world where everything is known in advance and specs don't change (i.e. a figment). Users got something functional, just not what they wanted or needed by the time the software shipped. Then came various spiral flavors: Iterative, Agile, XP. Unlike waterfalls (which run in one direction and don't back up), spirals can produce software much more likely to match what users want. Spirals support usability, and usability drives the need for spiral development. But what comes *after* usability? And will new development approaches emerge to support it?

So, I guess I'm really asking two somewhat-related questions. This is just a first crack very rough look for me, so please feel free to hack away, remix, rearrange, and add your own more credible (or just as wild-ass as mine) ideas.



After Usability comes *Flow*

"Thanks for giving me something useable, well-designed, and useful. Now, can you make it as engaging as a game or sport? Can you keep me so immersed that time and all the clutter of daily existence drops away? Where I'm under a spell that's never broken by an intrusion from the software itself? Where the challenge is NOT in using the software, but in what I'm using the software TO DO?"

Even if users don't start *demanding* Flow... it's a huge opportunity and advantage for those whose products support it. (And one of the key attributes of products with passionate users)

To learn more about flow--and I strongly encourage us all to make a study of it--some resources include:

The original FLOW book

An excellent overview of flow theory by award-winning game designer Jenova Chen, including an enchanting, seductive game that implements a theory of flow.

Game design guy Raph Koster's blog, and don't forget his excellent book on A Theory of Fun.

And two of my earlier posts on this:

User Enchantment and Cognitive Seduction.

What about development models?

I really have no idea. People like Jonathan Kohl and Jason Gorman are talking about Post-Agilism, but there's no agreement on what that looks like or even means. Some see it as nothing more than the practical approach of taking the best of what works without being such a hard-core Agile zealot. In other words, "agile" with a lower-case "a" rather than The Church of Agile.

And in this post I'm really talking about the web-app world, not necessarily big corporate enterprise systems that don't have the pressure the new class of web-apps (or games) have. It's the new web-app development that needs ultra-fast releases and near-instant responses to user needs (not necessarily user *requests*). Is there a Post-Agile model that works for this? Or is just faster, tighter iterations?

Do we speed up the spirals or do we do something completely different? How does this fit with things like, say, the 37Signals somewhat controversial Getting Real approach? I say "controversial" because some see it as nothing new at all, or worse--a fly-by-the-seat thing that absolutely Does Not Scale, or...

So, what do you think?

(And if I were smarter than I actually am, I would have done this artwork more napkin-sketchesque so you'd realize just how rough my thoughts are on this. As if that wasn't obvious...)

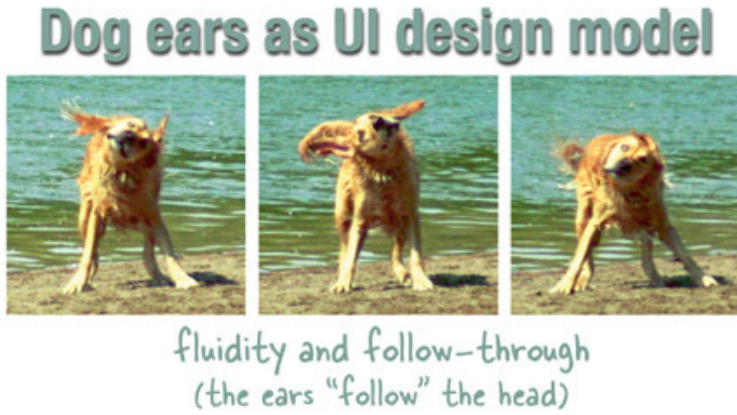
http://headrush.typepad.com/creating_passionate_users/2007/01/what_comes_afte.html

iPhone and the Dog Ears User Experience Model

By Kathy Sierra on January 14, 2007

I was at the Steve Jobs keynote. And like everyone else in that room, I was thrilled by the iPhone demo. The UI *is* spectacular, but for reasons you can't see in a photograph, or get from the online keynote video. The best part of the iPhone is simply this: the UI is *alive*. By implementing one of the key principles of animation, the designers have shown us the stunning power of using Dog Ears as a user experience model.

In the real world, we have physics. We have inertia. Things bounce and stretch and squash. We have *follow through*. Imagine a dog with long floppy ears sprinting for a frisbee. Now picture the dog coming to a screeching halt in front of the disc. What happens to the ears? *They keep going*. Then they "bounce" back. And it's a big part of what separates a good animator from an amateur.



Even if you don't notice it consciously, an animation (even of just *words*) feels more appealing and alive when things move in the virtual world more like things do in the real world (or even more exaggerated). It feels more lyrical, fluid... *less abrupt*. And *that* is what the iPhone UI does.

Yes the touch-screen is cool. And the multi-touch gestures are so very minority-reportish. But it wasn't the scrolling that made my jaw drop... it was what happened when the scrolling *stopped*: **it *bounced*!** The thing actually bounced if you flicked it hard and fast enough to send it flying up to the very (or bottom) of the list before it had a chance to slow down and stop. It actually bounced. And until you've seen it slow down and bounce, you haven't *felt* that visceral, life-like, fluidity.

Someone was quoted as saying, "You had me at scrolling." Well, for me it was, "You had me at what happened when the scrolling *stopped*."

And bouncing wasn't the only nod to a fluid user-experience... it also uses audio fades when you're listening to music (iPod mode) and a call comes in. Think about it. I attended a talk by Marc Canter in the mid-90's, and it changed the way I think about sound and users forever. In that talk, he railed against us--the interactive CD-ROM developers--for committing one of the worst sound sins--chopping the sound off when a user navigated from one place to another. He demonstrated it by making a huge verbal ruckus and then--dead silence--then back to a huge verbal ruckus. It was annoying. It was stressful. It was *what we were doing to our users*.

And all it took to fix it was a fade! An f'n fade. Not a long, elaborate, complicated cross-fade. Just a very short fade-out of the audio as you left an area where the sound was not going to continue.

From that moment on, I became hyper sensitive to how stressful it is when sound--especially loud sound--just cuts off. And now, if I'm listening to anything--music, a DVD movie, whatever--if I have to stop the sound for some reason, I attenuate. I grab the knob and rotate it to the left. It's one of those tiny gestures that my companions might not even notice, but on some level they appreciate it.

Life is abrupt enough as it is.

Why not reduce some of that for our users? If we can make a user experience where things don't come to a slamming, smashing, halt but instead move and fade as lyrically as a dancer, we've just added something to their life.

Try it. Turn the music up in your car or home stereo to a pretty strong (but good) volume. Ask a friend to join you. At one point, when they're in the flow, cut the sound completely. Kill the power. Notice their response. Now do it again, but this time fade the volume.

This is not a trivial thing.

And although Apple and the iPhone certainly aren't the first to use this kind of "absence-of-abruptness" to the user experience, they've done it in an elegant, subtle, flow-supporting, enchanting way.

Consider it UI research to sit in a dog park and watch some ears. Big, floppy, ears.

[FIY: after leaving San Francisco, I was home for less than 12 hours before getting on a plane for Australia, where I am now for the wonderful linux.au conference. So, my apologies for being off-line for the last week. It looks like I have a decent connection here in my hotel, so I should be checking in regularly again while I'm here. And oh GOD how I love summer. It was below zero F as I left Denver.]

http://headrush.typepad.com/creating_passionate_users/2007/01/iphone_and_the_.html

Who'd you make smile today?

By Kathy Sierra on January 15, 2007

Marketers and managers tell us to "delight" the customer. But they're usually talking about heroic gestures, "empowering the front line", and virtually always about how to use this "happy customers" focus as a competitive advantage. But sometimes it's the smallest of things that can make all the difference. Things that aren't bullet points in the brochure or check marks in product comparisons. Things that just... make you smile. Things the one who made you smile didn't need to do.

In the midst of a two-day travel hell to get to Australia two days ago, I landed in the Honolulu airport for a 9,256 hour layover. I was sleep-deprived, jet-lagged, and still mourning the loss of my lotion at Security Checkpoint Theater. And then I saw it. Marking the entrance to the Men's Restroom on the airport concourse is the typical international "MAN" symbol with one

little upgrade... the little guy's wearing a Hawaiian shirt (even has a little lei). I smiled. For the first time in 12 hours.

Too often we see formal *institutionalized* smile-strategies... like the Southwest airline flight attendants inserting jokes into their safety speech. But some of those attendants are simply repeating the script, and it shows. It's a lot *more* smile-inducing when the flight attendant just blurts something out spontaneously, in response to something in realtime. Or when they announce to the entire plane that there's a couple in coach celebrating their 50th wedding anniversary, having just returned from a romantic second honeymoon. Or when the pilot comes on and starts describing the joy of flying by telling you way too much about the physics of flight. THESE things make me smile, and for those of us who can't afford first-class, it comes just when we need it the most. And I want to know, "What causes these smile-inducing people to behave like this even though they don't *need* to?"

What are some other non-institutionalized things you can do to make someone smile? And what does it take to support that in your company *without* trying to institutionalize it? (which of course never actually works). I think we can all assume that someone who goes out of their way to bring a smile to your face--for no reason other than *they want to*--they must be feeling genuinely good. Those phoning-it-in aren't likely to make you smile. They aren't likely to smile themselves, let alone to care whether YOU do.

A few things that make me smile...

(That they didn't need to do)

* A thoughtful, almost imperceptible feature in a product. Something that surprises you that they'd have that attention to detail on something that appears to exist solely to make you a little happier, but adds nothing to the actual capability of the product. (Or so you might think... in reality, of course, it's those little things that can be the deal-makers or breakers in keeping us in flow).

* Easter eggs in software (*good* software... as an earlier commenter pointed out, if your software has big flaws and faults and I find an easter egg, it'll really piss me off that you spent resources on THAT instead of making a product that actually works)

* Insider references or homages (a form of easter egg) inside a product manual. (e.g. using TPS reports in a sample)

* Whimsical names for dishes on a menu (often this IS formalized, but sometimes it just feels like someone cared enough to make an otherwise dull diner a bit more festive)

* Playing foreign-language training tapes in the bathrooms of an ethnic restaurant.

* Fresh-baked cookies in the lobby. GOOD coffee in the reception area, not that crap instant with fake creamer. When I taste that first sip of really good coffee, I *always* close my eyes and smile. Bliss :)

Really, though, there's one really simple thing that we can do to make someone *else* smile.

Smile.

We've talked before about how scientists know that smiling produces physiological changes in your body. And thanks to mirror neurons, we know that seeing someone *else* smile or laugh can trigger the same neurons reponsible for making *us* smile.

But this cannot be faked:

But we can tell a *real* one by looking at the eyes, not the lips:

Crinkly eyes = real smile. No crinkly eyes? Faker. (or too much botox)

Bonus: take this BBC quiz to test your ability to spot the difference between someone who's smile is genuine vs. a faker.

Perhaps the real question should not be, "Who'd you make smile today?" but rather, "How can you get *yourself* to smile more?" We all know it's true... real, genuine, authentic, natural smiles are infectious. Picture the people you know who can light up a room when they walk in with a big, REAL, smile. We can all be those people. Imagine if someone told you that you had the power to instantly alter someone else's blood/brain chemisty in a positive way, potentially improving their immune system and giving them more physical energy. And all you had to do was flash them a smile. We can all be those people.

But for so many of us, we don't interact with our users face-to-face, so the next step, then, is to figure out how we can inspire

ourselves and especially our employees and co-workers to "virtually" smile at our users by doing something with our products or services that causes them to smile. (Quickest change: do something with your online tech support pages. Next quickest, do something with the user documentation)

Remember, it's often the smallest of things. Like a bathroom sign that changed the rest of my day.

[Tip: keep a 'smile journal' for a week or two, and try to write down every little thing that caused you to smile when you interacted with a product or service or person, and look for a pattern (or at least some ideas you can use). You can also track another person you interact with, and write down how often THEY smile, and what caused it. If the notebook is nearly blank at the end of two weeks, time to rent some Monty Python.]

So, who'd you make smile today? Who made *you* smile today?

http://headrush.typepad.com/creating_passionate_users/2007/01/whod_you_make_s.html

Let them do the thing everyone else tells them not to

By Kathy Sierra on January 30, 2007

This sign at the Royal Botanic Gardens in Syndey, Australia took me by surprise. So many signs tell us what we *can't* do, and it's delightful to see the opposite. We need more of this. And I love the, "Entry is free -- but if you would like to help preserve this wonderful place..." How can you refuse?

So, yes, I'm back. The Linux conference was wonderful (the Jeff and Pia Waugh are awesome), and soon I'll have a lot more to say about some of what I learned there. Sydney is a fantastic city, and I've now moved Australia up to the number two place I'd love to live, just under New Zealand. But I got very, very, very sick while I was there (no fault of the country ;) although I managed to find doctors who had no trouble giving me drugs I'd need an act of congress to get here. Fortunately, I got to take the time to recover *there*, in a very peaceful resort up the coast from Sydney, until I was strong enough to travel back home yesterday.

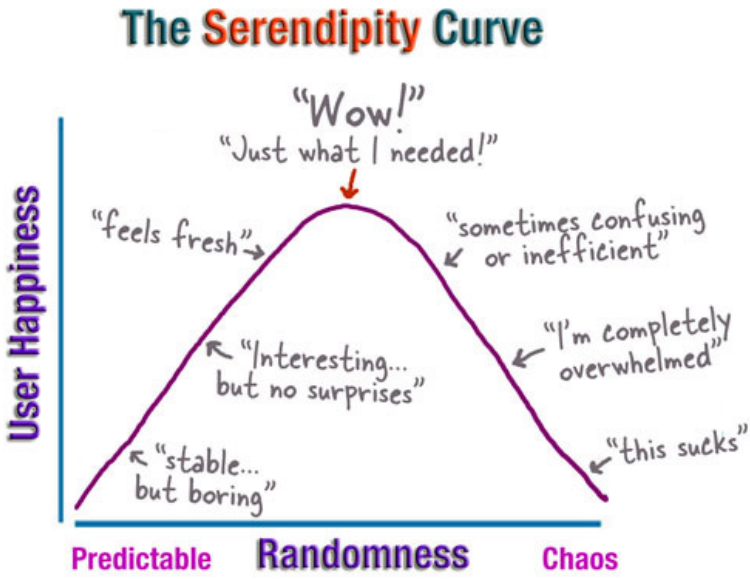
Sorry about the off-lineness, I haven't seen email in almost two weeks. But I'd like to publicly thank my co-hort Dan for stepping in here, and for y'all for sticking around. We're just about to put up some changes and fixes to the blog, too, so stay tuned.

I missed you guys. That's something I couldn't have predicted two years ago when I started this blog. but I really did.

http://headrush.typepad.com/creating_passionate_users/2007/01/let_them_do_the.html

Add a little more random to your product

By Kathy Sierra on January 30, 2007



You know the feeling: You follow a near-random trail of blog links and land on the post that solves your big business problem. You randomly flip through a physics book and find next week's sermon. You're shopping for discount dog food when you find your dream date. It's the powerful charm of the iPod Shuffle ("How did it KNOW that's just the song I needed to hear right now..."). It's serendipity. And maybe we should build more opportunities for it into our products, services, and *lives*.

In user experience design, especially, we often work our asses off to *remove* unpredictability. That's a *good* thing, mostly. An interface that does what you expect drops away so you can focus on whatever it is you're using the product to *do*. While we assume that randomness plays a big role in games, we do our best to strip it from "serious" products and services. But there are plenty of ways to keep a user experience consistent while still supporting--even encouraging--the chance for serendipity. And serendipity is delightful, astonishing, sexy, rewarding, inspiring...

When the iPod Shuffle first came out, the ads were based on the theme, "Life is random." I thought it was one of the lamest marketing spins ever. I imagined the meetings, "Let's spin the lack of display as a *feature*. Yeah, *that's it*. We'll sell the inability to choose your music as a benefit!"

But I was so so so wrong. Within a few weeks' of the Shuffle's release, the serendipity effect had kicked in. "OMG! That was the *perfect* song for this!" "Seriously. It can't be random. It's putting songs together that just... *work**" The Shuffle was getting people out of their playlist ruts. Out of the music comfort zones we all fall into (emo, anyone?). Exposing them to songs they'd loaded onto their pre-Shuffle iPod but that never seemed to be one of The Chosen Ones. Think about it. Think about all the music on *your* (non-Shuffle) iPod, computer, or vintage CD rack. Now think about the subset you actually listen to regularly. For most of us, it's a pathetically small set. By literally *forcing* people to listen to randomly-chosen songs, the Shuffle was constantly delighting, surprising, rewarding, *stretching* users. And users *loved* it.

Filters drive a bigger need for randomness today

We're all on info overload, and filters are the best antidote. Whether it's a tech or politics aggregator like Techmeme or Memeorandum, a topic-specific blog/online news site like Slashdot or Engadget, or our own hand-crafted custom news page like My Yahoo, we're all looking for ways to narrow the funnel. Even semi-smart online shopping sites like Amazon become a filter, telling us what we're most likely to be interested in, and even letting us help tune it to be more precise. But all this filtering, tuning, and pruning keeps us stuck! We end up seeing only what we think we *want* to see--what we're already familiar with--and slashes our chances for serendipity. And that means slashing our ability to create and innovate, or even to be truly surprised and delighted.

How can we add more chances for serendipity into our products, services, and even lives?

Of course it depends greatly on the product, with random-by-design (like the Shuffle) on the extreme edge of the Predictable/Random continuum. But here are a few (randomly-chosen) examples:

1) Staff picks of the Day/Week/Month

The bestseller lists reflect the popularity of the many. "Recommended for you" picks reflect what people *just like us* have bought. Both of these narrow the funnel, but the "Staff Picks" can introduce something new, especially when the staff pickers go out of their way to introduce things you might have otherwise missed.

2) Encourage other users to post "off-label" uses of the product

Don't just showcase examples of how the product can be used *in the usual way*. Get users to submit stories, pictures, examples, etc. of ways they used the product to do something nobody (or at least YOU) never imagined.

3) Randomly introduce things from completely unrelated domains

If you aggregate home improvement stories, for example, have a place where you insert a semi-random--but high quality--post from a *non*-home-improvement field.

4) Use cards from a shuffled "idea" deck

The idea is simple: select a card from a shuffled deck, and act as though whatever the card says is directly relevant to your current problem.

Some favorites:

Brian Eno's Oblique Strategies (designed for musicians and artists, but works for *anything*.)

Roger von Oech's original Creative Whack Pack (a long-time favorite of mine... I started using it more than a decade ago, and it's a big part of why I'm such a fan of Roger).

The IDEO Method Cards

I've never used them, but they're visually stunning, and others have recommended them.

5) The old standby: subscribe to magazines from unrelated domains

Walk through a large newstand, and linger in some of the sections you usually skip. You just never know when Cat Lover Today is going to have that perfect answer for you.

Even more challenging--but interesting--is to go beyond newstand magazines and flip through professional journals you find at the home of a friend or business associate (or waiting in the dentist's office). Who knows how many times we 'reinvent the flat tire' simply because it's never been solved in *our* world, while a gazillion solutions are out there in unrelated fields.

6) Find SOME means to add randomness (or pseudo-randomness) directly into your product or service

While a random tip-of-the-day is *one* implementation (and just because it's so often done badly, annoyingly, intrusively, and obnoxiously doesn't mean it HAS to be that way), there are probably a lot of other ways to introduce--perhaps as a user option--some element of random or pseudo-randomness. Drum machines (and other electronic/midi music software) sometimes let you choose to automatically insert subtle, somewhat random shifts in the music to make it just a little less perfect... which means a little more *real*-sounding.

Google has an "I'm Feeling Lucky" button that takes you straight to the first web page returned for your search query, but there's nothing random there. And while that's a useful feature, it might be equally useful to add an "I'm feeling **bored**" button that takes you straight to, say, the 42nd returned hit.

Photoshop has a kind of mutation feature that while not random, lets you instantly view your current image with a variety of different color adjustments. Perhaps they could add a "apply random filter" menu item, and let you see the image with some wild--and semi-random--combination of tweaks. It might never have occurred to you that the "plastic wrap" look is exactly what you needed to use on that picture of your ex you're about to put online for the world.

One of the problems with e-books...

Another area where randomness could matter a lot is in e-books. One of the complaints you hear from dead-tree-book-lovers is that they don't get to flip through the pages. On the surface, this might sound like yet another silly argument, and indeed I've heard e-book champions tell us we'll get over it, we won't care, or--hey--they'll just add a page flipping sound+animation to make us feel better.

But it's not the sound or tactile feel of the page turning that matters... it's the chance for serendipity you lose when you can't

easily, randomly flip through the pages! How many times have you flipped to a page in a non-fiction book and--viola!--as if by magic, the thing you need-but-didn't-know-it-until-you-saw-this-page appeared? And no, presenting you with a linear list of thumbnails doesn't count.

There is, however, a fairly simple and at least partial solution I've seen in older experimental prototypes, but have no idea if they're implemented in any current e-book readers: a random "flip through the pages" button. But it can't just be a sudden HERE IS PAGE 267 thing. It needs to have a visual that shuffles through the pages (like the Apple cover-art thing on iTunes) in a way that displays them large enough to *see* something potentially interesting. In other words, it's the serendipity of a simple flip through the book that needs to be retained.

Finally...

Perhaps the best way for us all to up our chances for serendipity is to cultivate diversity wherever, however, whenever we can. Like I said earlier about filters, the bright side of efficiency and focus comes with a dark side of narrow vision. The good news? Remembering to keep a bit of random (or at least semi-random) input goes a long way. Think of the implications. You really, really, really don't want *your* kids to think about *your* music tastes (or potential music *stagnation*) the way you felt about your parents (who still listen to the music they played in college), do you? Seriously. Who knows which hot hipster band of today is the Barry Manilow of tomorrow... so don't get stuck.

Apple's original Shuffle promo said "Life is Random", but that's stating the obvious. Perhaps a better mantra would be, "Random is Life." We could all use more of it, and if we can give our users a few more moments of serendipity, we're giving them a wonderful gift.

Bonus related links:

*The randomness of the iPod Shuffle is hotly debated (including but not limited to the computer science issues of "random"). Read more here, here, and--if you're a math/stats geek--here.

Randomness, artificial intelligence, and art.

One of Maggie Boden's wonderful essays on unpredictability and creativity. A sample:

"With the help of this mental discipline, even flaws and accidents may be put to creative use. Oliver Sacks reports the case of a jazz drummer suffering from Tourette's syndrome.24 He is subject to sudden, uncontrollable, muscular tics. These occur, though with reduced frequency, even when he is drumming. As a result, his drumsticks sometimes make unexpected sounds. But this man's musical skill is so great that he can work these supererogatory sounds into his music as he goes along. At worst, he "covers up" for them. At best, he makes them the seeds of unusual improvisations which he could not otherwise have thought of. (Similar remarks apply to jazz musicians who use Hodgson's program to help spawn interesting musical ideas, or to artists and designers who use "evolutionary" computer-systems in developing ideas which they could not have thought up by themselves.)"

The Future of the Book

Questions to ask randomly

So... what are YOU doing to keep random input in your life and/or the lives of your users?

http://headrush.typepad.com/creating_passionate_users/2007/01/add_a_little_mo.html

Difference between Jeff Bezos and Bill Gates?

By Kathy Sierra on February 2, 2007

I've given presentations on "creating passionate users" at both Amazon and Microsoft. 2 big companies, 2 CEOs. Guess which CEO has been to the talk? And he didn't just sit there, he participated. His hand shot up when I asked a question. *He quit fondling his Blackberry*. But far more importantly--he *asked* an amazing question.

And it was a question I can't imagine being asked by Bill Gates, *even in the alternate universe where Bill Gates WOULD choose to see a talk on "creating passionate users."*

So there he was, Jeff Bezos, third row. (The talk he came to was at O'Reilly's Foo Camp, not the one I did at Amazon.)

I tried to imagine what *he* was doing there. You're Jeff f'n Bezos. More than 10,000 people work for you. You're building a space ship!

After the talk, Jeff came up (patiently waiting his turn) and said he was *really* going to think hard about the implications of the some of the things we talked about, especially the part on levels and rewards. Then he asked, *"How can I do more for our reviewers? These people do so much, and work so hard--especially the ones who do a lot of reviews -- and the 'Top [some number] Reviewer' badges are not enough."* I was speechless. Not because I couldn't think of an answer, but because I couldn't believe someone this far up the food chain would even think--let alone *care* about this. My talk is geared toward--and usually attended by--founders of tiny start-ups, not Big Company CEOs. [Visualize me doing one of those cartoon double-take head shake eye-pop things]

But then I remembered my trip to Amazon, where Paul Graham gave a fabulous talk on what a company loses when it gets big, and how important it was to hang on to a start-up sensibility as you grow. Paul said to the Amazon folks, "You're a big company now, but how can you still act like a start-up in the ways that really matter."

And there's Jeff Bezos, doing exactly that. Acting just like the enthusiastic start-up folks I usually see--the ones whose chance for success hangs on their ability to make and keep users happy. The ones who don't have 10,000 other people to do it for them.

I have no idea if he thought about it ever again, and yes we all have our stories of bad Amazon customer service. But the point is, *Jeff Bezos, CEO, chose to spend time and attention on a talk about user passion*. And remember, what we (and Jeff) were talking about is at the *implementation level*, not some abstract concept of "we must be good to our customers." I believe most CEOs *do* care--at least strategically--about having happy users, but wouldn't waste a single synapse actually thinking about specific ways to make it happen.

So, to my original question on the difference between Jeff and Bill--*would Bill Gates attend a talk like this?* (And I'm not talking about my talk in particular, but anything--by anyone--on these kinds of user-happiness topics.) Ask yourself, would *your* CEO choose to hear a talk on how to create passionate users? How about your upper-level managers? Anyone besides *you*? When I worked at Sun, I gave many lunch-time "brown-bag" talks on this over my 4 years, and it would never have occurred to me that, say, Scott McNealy might drop in. Or any mid-to-

upper-level manager. (Although I'm delighted that there *are* some kick-ass higher-up Sun folks I know now that are working on this... but I hadn't met any of them until recently.)

I didn't charge a fee to do this talk at Microsoft, or any other company that has convinced me they'll take it seriously and try to make a difference. I don't do this for pay; I do it because I believe it matters. The problem is, the places it matters the most are the ones least likely to think about it. No, they have too many other--apparently more important--things to think about. And for the rest of us? That means there's plenty of opportunities for small companies and start-ups that DO make user happiness (not the same as "customer satisfaction") the top priority.

p.s. the question Jeff raised his hand on (and answered) was, "Who in popular culture speaks English without using contractions?" Jeff answered, correctly, "Data!" (thus reinforcing his geek cred). If you want to know why that question--and answer--was relevant, you'll have to read my earlier post on conversational language ;)

http://headrush.typepad.com/creating_passionate_users/2007/02/difference_betw.html

Inspiring your user-evangelists

By Kathy Sierra on February 6, 2007

In the past 30 days, did you enthusiastically recommended something--anything--to someone else? Maybe it was a new restaurant, web app, game, sport, note pad, band, indie film, car, micro-brew, lotion, operating system, environmental cause, dog food, or pillow. Chances are, you did. More than once. Our users *want* to recommend or (if they're passionate) *evangelize* things they believe in, and it's our job to give users the tools to do it. Indie bands often have "street teams" of loyal (unpaid) fans who hit the streets to post flyers, etc. Do *you* have an unpaid street team?

There are at least two ways to inspire evangelists: the sleazy way and the authentic way. Fortunately, the authentic, ethical way doesn't need a big budget. The sleazy, expensive, and often unethical way is to *hire* people to "pretend" to evangelize. There are companies that will assemble a team of faux street-teamers to spread the word, ranging from the despicable--like the sexy woman in the bar who fakes interest in a man while casually mentioning the product (without disclosing her "job") --to the less harmful but even creepier--the person who is paid to tell their *friends* about a product, albeit with full disclosure.

Here's the thing...

If you have to PAY people to evangelize your product or service, you probably don't have a product or service worth evangelizing.

(If it's about simply getting the word out on something too new to have customer/user evangelists, there are plenty of ways to 'seed' potential users to get the ball rolling.)

Users will want to evangelize on your behalf for two main reasons:

1) You're small--or in trouble--and they want you to succeed.

(When there's no guarantee you will) Apple was in this position at one time; I remember handing out the "50 things you can do to save the mac" handbook! This is especially true for independent bands, stores, products, restaurants, etc. but big, well-funded companies aren't immune, obviously. Non Apple-fans still marvel at why a crowd of thousands cheers so loudly when Steve Jobs shows how much money the company is making. They don't realize that all we (the faithful) see is assurance that our beloved devices will survive, new ones will be developed, and that more developers will find it worthwhile to create for this platform, etc.

2) They *believe* in the benefits of whatever you offer, and want others to experience that (especially their close friends and family)

How to Create Evangelists The Authentic Way

1) You have a product or service or cause that helps users learn and grow and kick ass at *something*.

2) You give users tools to help them evangelize.

3) You do not ever, ever, *ever* pay users for doing this.

Remember, even if your product has problems, you can often make up for a ton of flaws by building up the ecosystem *around* the product. A killer user community site. A breakthrough manual. Stunning customer support. If you're helping your users learn and grow and improve, you're inspiring them to be better and--as we know--being better at something is a lot more fun than being a frustrated newbie or mediocre just-getting-by user or participant. If you can inspire your users to learn and grow, they'll naturally want to get others to share in this experience.

Tool ideas

(most of these are dead-obvious, but all too often overlooked)

** A short, free DVD*

One that isn't a sales/marketing pitch, but simply *explains* why the evangelizing user is so interested in getting others to see what they see. A truly passionate user would love nothing more than to be able to give someone a DVD that gets the other person to say--after watching it--"Hmmm...now I'm starting to understand why this means so much to you."

** Posters and stickers*

In other words, things to spread around in public to help raise awareness. The Sticker Guy is one of many good sources for stickers. (And check out this fun Wired story about Apple stickers. I have one on my car.)

** Free tickets*

The Parelli organization goes on tour across the US and gives members of their official club up to 10 free tickets so that they can bring the non-converted to experience for themselves what Parelli-folks call "the magic."

** Friends and Family nights*

** Testimonials from credible people!*

Is there someone trusted and respected in your domain who uses your product? Your users need to know! Our Design Patterns book had endorsements from some of the key figures in the software development world, and we've had hundreds of emails from people telling us that this was the only reason they decided to give it a try. In the Parelli world--where members of the cult (like me) are constantly battling with those who dismiss it--the endorsement by two US Olympic Equestrian medalists--Karen and David O'Connor was huge. When they made a video about it (which we have to pay for), they gave us perhaps the best possible ammunition--"Think Parelli doesn't apply to anyone except cowboys? Don't listen to me, pop this in your DVD player for a few minutes..."

** Make it REALLY obvious how users can get involved in evangelizing*

For inspiration, check out:

Oxfam's What You Can Do page or Greenpeace's Get Involved page.

** Private behind-the-scenes website areas for members only*

...that they can share with their friends and which highlight the *real* reasons your user is so passionate.

** Free tickets to learning webcasts they can give to their friends.*

Not *marketing* webcasts... I mean actual training courses that most people have to pay for.

** Materials, support, and recognition for user group leaders.*

Sun has done a lot to recognize and reward JUG (Java User Group) leaders, for example, including special meetings and receptions at conferences, and giving them special access to some key Java folks at Sun.

** Create a "Street Team", and a toolkit*

Have *some* kind of affinity club, user group, *something* that users can join and become members of. And make sure that members can get an evangelism toolkit whether it's a PDF poster to download or a full-blown package in the mail with flyers, stickers, t-shirts, CDs, etc.

A great example of a very active (and apparently successful) street team are the Petal Pushers. (Click on Petal Pushers from the side menu). I encourage everyone to check it out. Another example of an indie-band-on-a-budget street team is here.

The street team is an interesting phenomenon because it is often a lot more successful for bands that *aren't* well known. In fact, part of the appeal to hardcore street teamer fans is that they get to be *the first one in their group to have discovered the band.* Being the first to tell/show something cool to a friend brings considerable social "points". I don't know much about street teams, but Skyler has been *extremely* active in two of them, including (a looooong time ago) the Sugarcult">Sugarcult street team. She whipped up a lot of interest in local shows, and these guys would even recognize her at events and talk to her. But once they started becoming more "known", she lost interest. But that's a whole different topic...

[Side note: There are two kinds of companies you can hire to help you (band or otherwise) create a "street team". The sleazy

kind will take your money in exchange for providing you with street team "members" who may never have heard of you and don't particularly like you. The authentic kind are simply marketing/community helpers who will help you create a street team program *for your existing loyal fans*.

My Personal Opinion on What NOT to Do

* Do not EVER pay your members/fans/users to do this.

Limited edition t-shirts and stickers? Absolutely. Free evangelism products for friends (like the tickets and CDs) -- absolutely. But money for referrals? Never. (This is a big topic in itself that we'll save for another time)

Paying them, or even doing a "refer a friend and get YOUR next thing free..." program changes the incentive. And while it may not change the users motivation, it taints the incentive. Irrevocably, in my opinion.

If you have truly passionate users, paying them is not only not necessary, it could *hurt*. That doesn't mean you don't *reward* them, of course, there are gazillion great ways (and reasons) to reward your loyal users. But that's for their continued loyalty, support, patience, feedback, etc... not for some kind of paid referral program.

http://headrush.typepad.com/creating_passionate_users/2007/02/inspiring_your_.html

Don't ask employees to be passionate about the company!

By Kathy Sierra on February 6, 2007

People ask me, "How can I get our employees to be passionate *about the company?*" Wrong question. Passion for our employer, manager, current job? Irrelevant. Passion for our *profession* and the kind of work we do? Crucial. If I own company FOO, I don't need employees with a passion for FOO. I want those with a passion for the work they're doing. The company should behave just like a good user interface --

support people in doing what they're trying to do, and stay the hell out of their way. Applying the employer-as-UI model, the *best* company is one in which the employees are so engaged in their *work* that the company fades into the background.

Given a choice, I would work ONLY on projects that followed the Hollywood Model, where people come together with their respective skills and talents, and DO something. Make a web app. Create a book. Build a game. Develop and deliver learning experiences. The happiest moments of my work life were on projects where we pulled all-nighters because we *wanted to*, not because the corporate culture said we weren't a true team-player/trooper if we didn't.

Employees shouldn't be sleeping in cubes to prove they're "passionate employees." I want to work with people who have a particular set of skills (and interests) who view themselves and one another as either **professionals/craftspeople** (programmers, designers, engineers, animators, editors, scientists, authors, educators, architects, entertainers, etc.) or as **producers and assistant producers** (the people who pull it all together, support the craftspeople, and make it happen).

[**UPDATE:** I do not consider "caring about the user" as separate from "our work." In other words, I consider one who is truly passionate about their work to have "the effect it has on the user" as a fundamental part of that work. A tech book author/teacher who has brilliant wordsmithing and technical breadth but no effect on the reader is not a professional. A software developer who crafts

brilliant code that doesn't include that code's effect on the user is not a professional. Part of what makes us professional/craftspeople is that we value and never forget the POINT of our work, and the point is--for most of us--what it means for the user. It's quite sad that many of our professions have rewarded work without making the user the most important attribute of how we asses that work.]

I realize these aren't mutually exclusive--one can be passionate about their employer *and* the work they do, but it's a matter of which one employers value. And all too often, it's the wrong one.

The simple 4-quesetion test to see if someone has a passion for their *work*:

* When was the last time you read a trade/professional journal or book related to your work? (can substitute "attended an industry conference or took a course")

* Name at least two of the key people in your field.

* If you had to, would you spend your own money to buy tools or other materials that would improve the quality of your work?

* If you did not do this for work, would you still do it (or something related to it) as a hobby?

DIFFERENCE BETWEEN PASSION FOR *EMPLOYER*, vs. PASSION FOR *WORK*

Passionate about the *company*:

* The ultimate team player who goes along with the group rather than voice dissent

* Works late nights and weekends because "everyone needs to pitch in on this project"

* Defends the company to anyone, anywhere that criticizes or questions its products, policies, or practices

* Puts responsibility to employer above responsibility to customers, without question

* Questions, but does not challenge the status quo

* Is well-liked because they do whatever is asked, enthusiastically

* Accepts (and uses) phrases like, "this is what corporate needs us to do."

* Cares a lot about his career path in the company; focused on getting management recognition.

Passionate about the *work:*

* Scores well on the 4-question test:

- keeps up with trade/professional journals

- knows who the key people in the industry are

- would spend his own money, if necessary, for better tools

- if they were NOT doing this as their job, they would still do something related to it as a hobby

* Works late nights when, "I'm just one-compile away from this awesome refactoring that's going to make this thing run 40% faster." In other words, they work late when they're driven by something they know they can do better on.

* Defends the quality of his own work (and, in the Hollywood Model, the work of his team).

* Puts responsibility to his own ethics and values--especially related to quality of work--over responsibility to employer.

* May not be extremely well-liked, but is highly respected and tolerated because he's known as one who, "cares deeply about doing the best possible job, and is very good at what he does." [update: the person must be liked well enough for people to want to work with him again... the Hollywood Model has a way of screening out a**holes... nobody calls them for their next project.]

* Does not accept, "this is what corporate needs us to do" when it conflicts with quality and ethics. Must be given a damn good reason why a corporate decision is worth the downsides.

* Does not care about upward mobility in the company. Cares about doing fabulous work and possibly the recognition of his peers in the industry. May stive for professional recognition.

Am I, as always, glorifying the maverick? It only looks that way if your perspective is a Big Company that puts teamwork and company loyalty above *all* else. In the Hollywood Model, our ability to get work--which means new projects--depends entirely on whether anyone on previous projects wants to work with us again. What you hope for--and what happens--in the Hollywood Model is that when a team is being assembled, someone says, "Hey, last time I worked on the Bar project, Roger did the graphics and he was awesome." And the assistant producer or project manager says, "What's his phone number?"

In the Hollywood Model--despite the glamorous name--whether the *project* is exciting or sexy has very little to do with whether we view our *work* together as exciting and sexy. The sound guy pushes the edge with intelligently-adaptive audio that changes subtly as the user navigates into different "places." It doesn't matter that the project is a boring bank's interactive annual

report. The programmer (usually my role) builds an authoring tool to help the artists sync their work to the sound way before the engine is ready. The artists decide at the last moment that they aren't happy with something that *nobody but they can see*, and spend days tweaking something that they swear will have a subconscious impact (for the better) on the user.

There are plenty of companies--even big ones--who are able to foster this kind of enviornment (including some parts of Google, I've heard). And in many small start-ups there is virtually no distinction between passion for the company and passion for the work--they are, essentially, the same thing, driven by the same overall desire to succeed. The companies that have the greatest chance, in my opinion, are the ones who can hang to that. And I would start by thinking of project managers as "producers" and treating the "talent" like gold ;)

Finally, if you really want your employees to be passionate about the company, take lessons from UI and Usability: let people do what they want and need to do, and get the hell out of their way. Unfortunately, too many of our employers are like really *bad* software--frustrating us at every turn, behaving inconsistently, not giving us a way to learn new things and develop new, cool capabilities, etc.

Remember, when I say I have a passion for a particular piece of software, it's not *really* the software I'm passionate about. **It's always about my passion for what the software lets me DO.** Companies should work the same way. By acting like a good UI and letting employees express the passion they have for *their work*, you'll end up with employees who'd never consider going elsewhere.

http://headrush.typepad.com/creating_passionate_users/2007/02/dont_ask_employ.html

Marketing should be education, education should be marketing

By Kathy Sierra on February 11, 2007

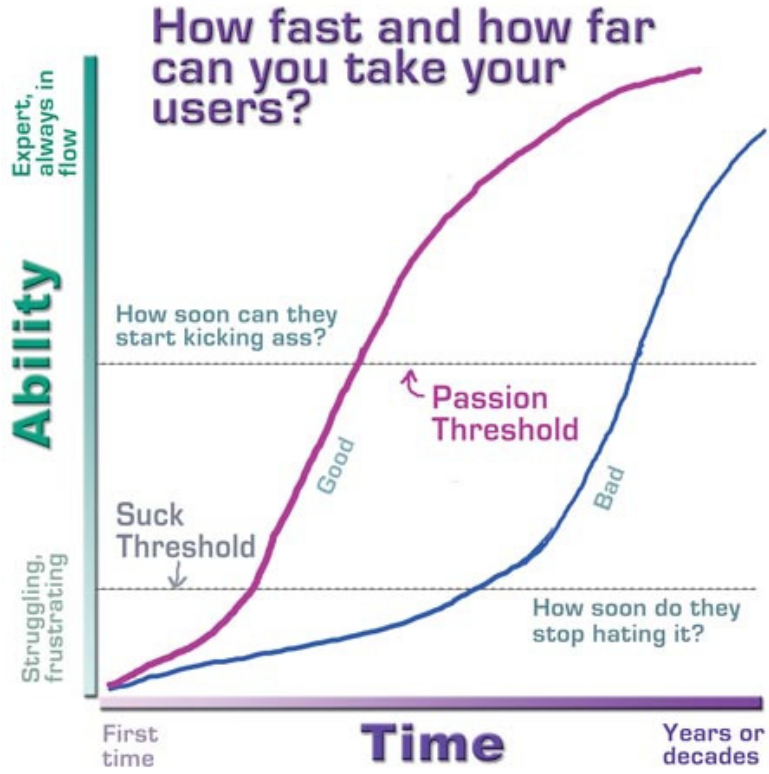
Do you want passionate users? *Educate them.* Do you want passionate learners? *Sell them.* If ever there were two groups who ought to trade places--and especially *research* -- it's teachers and marketers. Our mantra here is, "Where there is passion, there is a user kicking ass..." and by "kicking ass" we mean "being really good at something." In the post-30-second-spot world, the *marketing* department should become the *learning* department. Meanwhile back in schools, *teachers* should become...*marketers.*

The tragedy is this: the amount of money spent in the US each year on *marketing* research is orders of magnitude more than the amount spent on *learning theory* research. Big business probably spends more in a week on brain research than the US Department of Education spends in a year.

[I don't have the real data, but I've been trying to piece it together enough to make wild-ass estimates like this.]

The good news is this: in the he-who-does-the-best-job-of-getting-his-customers-past-the-suck-threshold-wins world that's beginning to emerge, companies may need teachers more than marketers. And in my perfect world, marketers and

teachers exchange research and techniques, and by applying marketing to teaching and teaching to marketing, everyone benefits.



What Marketers Could Do For Teachers

Marketers *know* what turns the brain on (currently, not last week). Teachers need that more than ever today.

Marketers have access to fMRIs. Teachers rarely do.

Marketers are dangerously close to finding the Buy Button in the Brain. Think what teachers could do with that research... after all, that Buy Button could be modified into a Learn Button with very little effort.

Marketers know how to motivate someone almost *instantly*. Teachers could sure use that.

Marketers know how to manipulate someone's thoughts and feelings about a topic. Teachers could use that to 'manipulate' a learner into thinking, say, "math IS cool."

Marketers know how to get--and keep--attention. I know some teachers who'd give a kidney for that research.

Marketers spend piles of money on improving retention and recall. Teachers--and *students* need all the help they can get.

[Yes, I'm aware how horrifying this notion sound -- that we take teachers and make them as evil as marketers? Take a breath. You know that's not what I'm advocating, so keep reading.]

What Teachers Could Do For Marketers

(Marketers who want passionate users, that is)

Teachers know the importance of honesty and integrity. The good teachers *care*. Some--perhaps many--marketers could use a lot more of that, especially now that the internet has made it far harder for marketers to get away with deceptions. Those damn users *talk*! They email, they youtube their bad experiences, they blog it.

Teachers know how to help people think on a deeper level, to get beyond the surface level of understanding. In old-school advertising, only the most superficial attributes were used ("This product will make your neighbors envious!") Clearly, those days are dwindling.

[And don't even get me started on how bad most product manuals are--where the difference between *pre*-sales and *post*-sales material is huge, and completely backwards. "Yes, once they've actually paid us and become a customer, who cares how the manual reads or what it looks like?"]

Teachers help people think about *thinking*. In fear-based (or any emotion-based) marketing campaign (especially politics!), thinking was inhibited. But people can't learn and improve without *thinking*, so any marketing approach based on helping users get better needs to use emotions to enhance thinking, not prevent it.

Teachers know how to help people through the rough spots... where the learner is still firmly in the suck zone. Marketers need that more than ever, since so many of the most sophisticated products can't be mastered in 5 minutes.

Should we be worried about the hot new research known as neuromarketing? Yes. But it's going to happen regardless of what we do. Why not start demanding that marketers be transparent about the research and their applications? Big Marketing is not about to stop using techniques to manipulate us into wanting things, and about the *only* defense we have is to *know* that this is happening.

If we're to be smart consumers (and voters), we *must* stay one step ahead of those who are trying to manipulate us without our knowledge. And for that, we must know as much as possible about how our brains work, and how we're being tricked, spun, and seduced. We should all be comfortable thinking, "Oh, that's obviously my amygdala talking."

But rather than rail against the research and bemoan the fact that the marketers (and politicians) have these "secrets of persuasion", we can put these tools to good use--one of the main goals of this blog. To help ourselves, our students, and our users *learn*!

Of course, there should be full disclosure *everywhere in which these techniques are used*. We should demand it from marketers, and expect it from teachers. In the Head First books, for example, the beginning of each book describes exactly what we're trying to do to your brain (i.e. how we try to trick your legacy brain into thinking the code is as important as a tiger).

Public education in the US is in a pretty sad state, but I'm reminded of an old anti-war bumper sticker that went something like:

"It Will Be a Great Day When Our Schools Get all the Money They Need and the Air Force Has to Hold a Bake Sale to Buy a Bomber" "

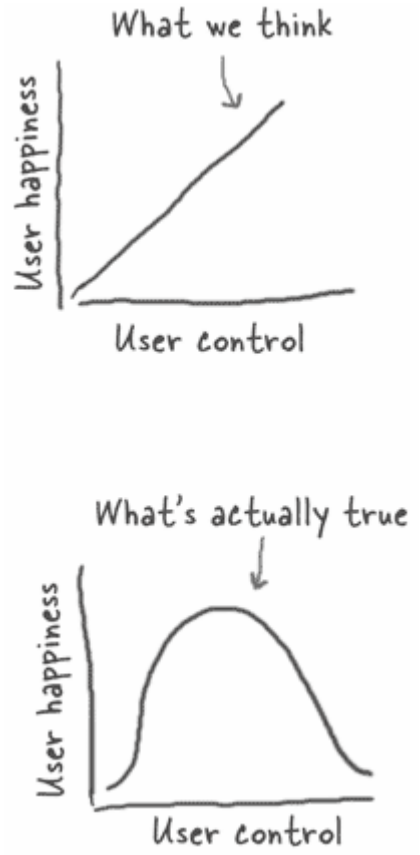
I don't have anything clever, but I like the idea:

"It Will Be a Great Day When Our Schools Get all the Brain Research the Marketers Have, and the Marketers Have to Hold a Bake Sale to Buy an fMRI."

http://headrush.typepad.com/creating_passionate_users/2007/02/marketing_shoul.html

How much control should our users have?

By Kathy Sierra on February 13, 2007



We all know Featuritis is bad, but what about User Control? Is more always better? The notion that a user-centric focus means putting users in control of everything--their software (and other tools), their learning, their conferences, the companies they support (the now-over-used "community")--is pervasive. But even when users *do* have the expertise to make good decisions, do they *want* to?

In some scenarios, of course. But when applied with abandon, user control can mean user suffering. In the 80's, the big thing in education was *Learner* Control. With hypertext tools came CBT programs and learners were finally put in charge of their

own paths through material. *The learner was empowered!* Just one problem: most people pretty much suck at making sound learning decisions, especially when they don't already know the material. So, the era of more-is-better-for-learner-control was over.

Then in the 90's -- Whoo-Hoo! Interactive Movies! Interactive Television shows! Interactive Fiction! Outside of rare novelties and a few good story-driven *games*, most of us would rather leave our storytelling to Steven King or Steven Spielberg, thank-you. A huge part of the *point* of movies and novels is to be swept into another world--a world we do *not* have any responsibility for.

Worst of all, though, is the ongoing trend toward more-is-better for the products we purchase. More choices, more options, more control. In the book The Paradox of Choice, Barry Schwartz looks at how the overabundance of products today makes buying even *toilet paper* stressful. We shut down when we're faced with too many choices, *even when those choices are about relatively simple things*.

Yet we expect people to make decisions over some of the most complex things, regardless of whether they have any knowledge or training in those areas. I look at product checklists and comparisons for electronic devices and think, "WTF are they talking about?" I have no idea what this thing-with-the-check-mark-next-to-it is or why I'd want it. And we don't just agonize *before* we choose, the vast array of possibilities has us agonizing *afterwards* as well. Second-guessing ourselves, continuing to check reviews, etc. Like we don't have enough stress.

And in software programs, especially, we expect users to choose their workflow configurations way before they have the slightest idea why they'd care. Or we give them ten different ways to do the same thing--so each person can do it in the way that best suits them--when the new user just wants to *do* the thing -- not grapple with the cognitive overload of ten ways to do *the thing they still can't do*.

How much control should users have?

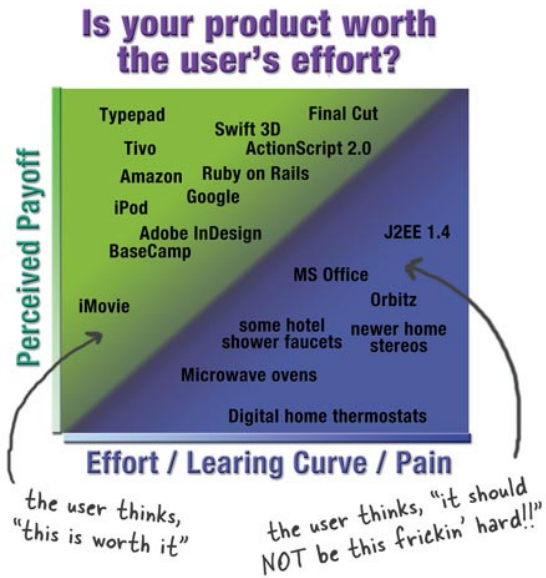
Obviously this is a big "it depends", but the main point is to focus on the relationship between user control and user capability. As user capability (knowledge, skill, expertise) increases, so should control -- at least for a lot of things we

make, especially software, and especially when we're aiming not just for satisfied users but potentially *passionate* users. The big problem is that we make our beginning users suffer just so our advanced users can tweak and tune their configurations, workflow, and output. [For the record, I'm a big fan of splitting capabilities into different products, or having a really good user-level modes--where you use wizards or simpler interfaces for new users, etc. Yes, they're often done badly, but they don't have to be.]

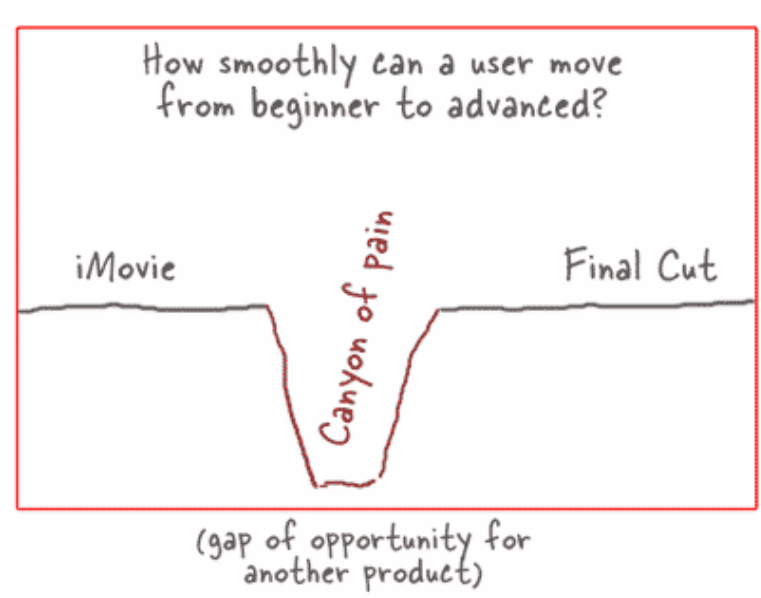
The simple rule we so often forget is:

The amount of pain and effort should match the user's perceived payoff.

In other words, the user has to think it's worth it. Yes, another "duh" thing... but if it *is* that "duh", then why oh why haven't some of the biggest producers on the planet taken it to heart? How come I *still* can't tune my Denon receiver? Or adjust my home thermostat properly? How come I find myself in hotel bathrooms staring at the shower faucet, wondering how annoyed the front desk will be when I ask them to help me take a bath. How come I can't turn off automatic Capitalization in Word? (trust me, it's not as simple as it seems...)



But we'll accept (and sometimes even *value*) pain and effort when it's worth it. Apple's Final Cut, for example is *much* more difficult than TextEdit. But I *expect* Final Cut to be hard... and it's worth it. The pain-to-perceived-payoff ratio *works*. My stereo receiver, on the other hand, just pisses me off. The sad thing is, I'm probably just two button-presses away from success, but I swear the possible combinations of button-presses on my remote exceeds the number of particles in the known universe.



On the other extreme is Apple's iMovie. It gives you almost no control, but the payoff is high right out of the shrinkwrap. It exceeds my expectations of pain-to-payoff. But pretty quickly, anyone who gets into iMovie--and is bitten by the movie-making bug--starts wanting things that iMovie doesn't let you control. So... Apple says, "not to worry -- we have Final Cut Express HD for just $299". The problem is, the learning curve jump from iMovie to Final Cut Express is DRASTIC. There needs to be something in the middle, to smooth that transition.

User Control in Web 2.0

I realize that part of the Web 2.0 "sensibility" is that users are in charge, but I'm pretty sure even Tim O'Reilly doesn't mean that Web 2.0 means the inmates should be running the asylum. There's an ocean of difference between user *contribution* and

user *control*. I'm sometimes afraid that the Age of User Participation will lead to the Age of Too Many People Doing Things They Are Not Qualified To Do But That Everyone Is OK With. Amateur Mash-up videos on YouTube? Hell yes. But what's next... *amateur minor-surgery* mash-ups? (that is actually, scarily, already happening, and I won't even link to it).

Putting users *first* does not necessarily mean putting users *in charge*.

I believe with all my heart in working with the user's happiness in mind (i.e. helping the user kick-ass), but part of my role is to use my specialized skills and knowledge to make that happen.

Even the poster kid of community-based business, Threadless, does not *really* put its community in control. In charge of voting on t-shirts, yes. In charge of whether Threadless is successful, yes -- but no more so than most businesses--they all live or die on whether customers want their product, experience, or both. But the Threadless community does not do the company's books, decide who to hire, choose their factory location, etc. The community has a very strong voice, and the Threadless guys listen--and respond--much better than most, but the company still controls the company. User *contribution*, not user *control*.

User Control and Capability enables Passion

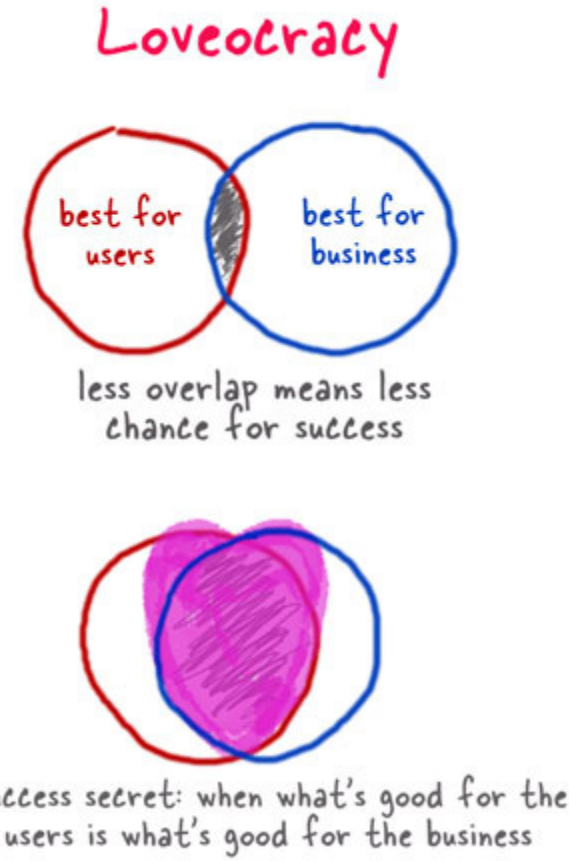
In the end, though, having more control and capability represents a higher-resolution experience. It's part of what makes being GOOD at something so much better than being bad or even average. And it's that high-resolution experience that inspires people to passion. (A passionate snowboarder is usually on black-diamonds, not the bunny slope) So we *should* be trying to give users more capability and control...and encouraging them to take it. But we must balance that with the learning they need to take that responsibility without being overwhelmed.

Like everything else, it all comes back to user education. The more we help them learn and improve, the more control they can handle... and appreciate. By putting the user *first*, it's our job to give them the responsibility they want, but only when we know they're ready to handle it.

http://headrush.typepad.com/creating_passionate_users/2007/02/how_much_contro.html

The real secret to a successful blog/book/business...

By Kathy Sierra on February 14, 2007



For the last three years, Bert and I have tried to explain the "secret" to the success of the Head First books. We've tried to explain the "secret" to how a little non-news, non-scandal blog could land in the Technorati Top 100. We've tried to explain the "secret" to why Javaranch is one of the largest, most active, and well-loved developer communities on the internet. One big clue: we're not that talented. There is a secret, yet, but it's mostly a if-WE-can-do-it-ANYONE-can-do-it thing.

I've revealed the Big Secret before, but perhaps the *bigger* secret is that almost nobody takes it seriously. It seems too simple. Business books make it complicated. Consultants make it

complicated. Those who don't want to try make it complicated. But it's not. Hard work? Yes. But the hardest part is simply taking it seriously. After that, it's implementation details. The details *matter*, but it's what *drives* the implementation that matters most.

So, on this Valentine's Day, I thought it was time for a reminder to myself and my co-authors:

Success no longer has to be a *meritocracy* (or *advertocracy*), today it's just as much a *loveocracy*.

The secret is simply this: you have a much better chance for success when your business model makes what's good for the users match what's good for the business, and vice-versa. Our books are best-sellers not because we're better authors or teachers (a meritocracy), but because they were literally labors of love. We wrote them with one very clear goal:

* The only way the books will be successful is if people actually *learn* from them.

* The only people will actually learn from them is if they actually *read* them.

* We must do everything we can to get people to read more than most people read in a tech book, and in such a way that they learn--and realize how much they've actually learned.

What's good for the readers is what's good for the books. Where I think so many potentially better books go wrong is that they're really good books (meritocracy), but they're written with a focus on Being A Really Good Book. (Which is often completely at odds with a book that's good for the reader.)

And why do you read this blog? I always ask myself, "how can I help my readers in some way?" Whether it's a tip or trick, a post you can use to help make your case to your boss, a new way of looking at something, a potential source for an idea, a pointer to something useful...I try to make 90% of the posts here for *you*. And you in turn respond with the most amazing, insightful, inspirational, and often entertaining comments.

What's good for you is what's good for the blog. And for me.

This is not to say you still can't succeed with a business model where what's good for the business is bad for the user and vice-versa, but next time you're in a product design meeting or a

business development meeting or you're planning a book or a blog or... ask the question we keep bringing up here, "What will this help the user do?" Not, "How can we make a great product?" Nobody cares about your company, and nobody cares about your product. Not really. *They care about themselves in relation to your product*. What it means to them. What it does for them. What it says about them that they use your product or believe in your company. You're still just the delivery guy, and your package helps the user kick ass at something. However, when you DO have a product that truly helps the user, they might just love you for it. :)

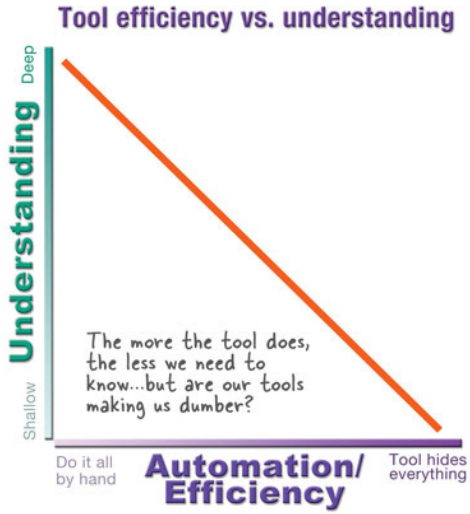
Happy Valentine's Day.

I heart my readers :)

http://headrush.typepad.com/creating_passionate_users/2007/02/test.html

Are our tools making us dumber?

By Kathy Sierra on February 21, 2007



It's lunchtime at the cafe and you give the cashier a $20 bill for an $8 purchase. She gives you $32.78 in change. You mention the mistake. She says, "But that's what the cash register says I owe you." She can't cope with the cognitive dissonance between reality and What The Machine Said. Later that day you get a frantic call from a co-worker--a recent addition to the programming team. "I keep getting this error message that it can't find the classes I'm using!" You ask, "By 'it' do you mean the compiler?" He answers "I don't know. I'm using an IDE." That night, you're helping your 12-year old son with his math homework when you realize--in horror--that while he's quite good with the calculator, he couldn't multiply two three-digit numbers using only paper and pencil if his Wii depended on it. These tools were designed to make us more efficient, so that we can focus on something more important than the tedious task of, say, giving change, organizing source code, and doing calculations. But are they helpful timesavers, or are we dumbing ourselves--and our users--down?

Obviously this depends greatly on the tool, the operator, and the task itself. If we all had to understand what every tool was doing/hiding for us, we'd waste brain bandwidth that could be used for something more important--like what we're using the

tool *for*. But in my examples above, think about how fragile the user's ability is if they don't understand what the cash register, IDE, or calculator is really doing. Without that understanding, what happens if the tool stops working? (In college I worked in a small surfboard shop in SLO, California, and the owner said, *"I don't care what happens to the cash register, always take the customer's money!"* Power outage? Use a damn cardboard box for the cash drawer until it comes back up...)

But should a web designer need to be an HTML coder? Or can he just use a WYSIWYG tool? The debates still rage in the web development world, although the issue should be resolved soon enough. In desktop publishing, for example, you will *never* hear, "Oh, you can't just use Quark or Adobe InDesign... you really need to tweak the Postscript by hand to do it right."

Some people think even automatic transmissions are dumbing people down. (I've offered to let friends borrow my car and I'm always shocked when I hear, "No, I can't drive a stick.") A flight instructor friend said there are some planes they don't want you to learn on, because those planes *do too much for you*. Some people think convenience foods like TV dinners are keeping generations from learning to cook. My sister's boyfriend could fix his own VW bus, but that was before cars became computers, before master mechanics were often reduced to part-orderers.

Tools can reduce errors, handle the tedious work, and potentially let us spend more time in flow. Still, when I see those cashiers and programmers, I think we need to keep a few things in mind:

Tool developers

If you make a tool that's hiding things the user should *understand*, maybe you could provide a tutorial or even an *understanding mode* where the user can ask the tool exactly what it's doing and how it made the decisions it made. But there's another issue for tool developers, and that's where passion comes in. Consider a point-and-shoot digital camera with presets for things like Portrait, Sunny Day, etc. The camera *hides* the complexity of making adjustments for exposure, white balance, etc. For most people, that's the whole *point* of these cameras--they don't WANT to mess with the settings of an SLR. But it's staying in point-and-shoot mode that keeps most people from developing a passion for photography (and ultimately, buying more expensive cameras and lenses).

But what if you could use your point-and-shoot as a way to *learn* more about photography? It would be so helpful if you could put the camera into a kind of "teach me" mode, where it explained *what* it adjusted and *why* it did it. That would make a great bridge to help you feel more confident moving into a (more expensive) digital SLR and avoiding what most first-time SLR owners do--*keep it in program/automatic mode*.

Tool teachers

Consider forcing students to do some things *the old-fashioned way* before letting them get their hands on the tool that'll automate much of the drudgery. My first semester of college stats wouldn't let us use computer apps for anything. Just us and our HP calculators. I hated it. But by the time we started running (and writing) our own programs, we had no doubt what was happening at each step and how to troubleshoot. When I teach Java, I *always* teach it using nothing but a simple text editor and the command-line. I do advocate tools for development, but never, never, NEVER for someone who doesn't understand Java at a fundamental level (compiler options, packages, namespaces, access modifiers, etc.)

Tool users

90% of the time we probably don't need to know how things work under the covers. I only barely understand why 747's ever leave the ground. I've never changed my own motor oil. (I have, she says proudly, topped off my windshield wiper fluid.) But I shouldn't think about putting a bit in my horse's mouth before I understand everything from horse anatomy to the principles of leverage that bit was designed for. I don't have to know how to create a microchip to use this MacBook, but if I don't understand the basics of its UNIX OS underpinnings, I can get into trouble figuring out where things are, how to set up security, etc. And just because there's one Starbucks per every 20 square feet in the US does NOT mean you shouldn't know how to make good, strong coffee the old-fashioned way.

Just something to think about, and as always... I'd love to hear your thoughts about tools, dumbing down, and strong coffee.

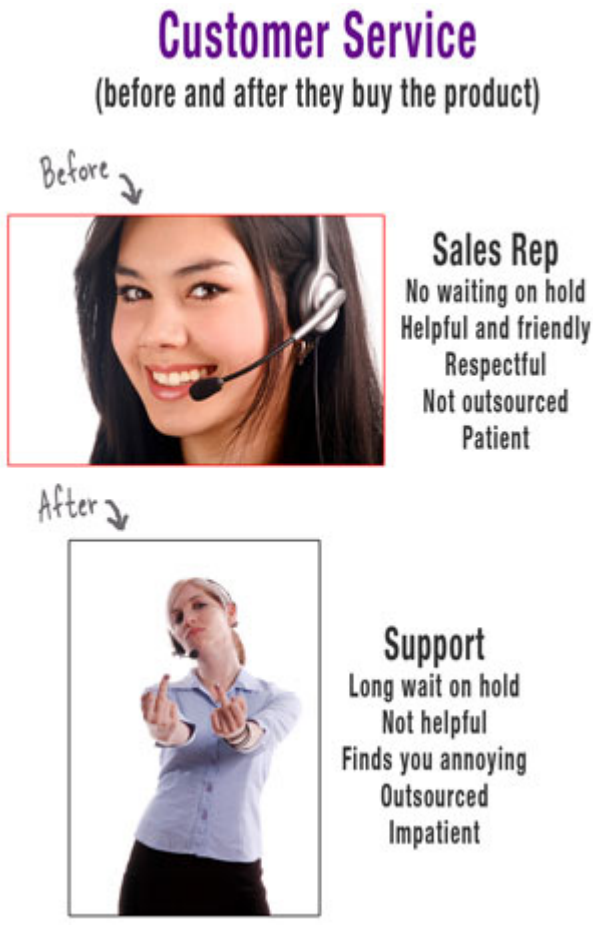
http://headrush.typepad.com/creating_passionate_users/2007/02/its_lunchtime_a.html

Too many companies are like bad marriages

By Kathy Sierra on February 24, 2007

It's been said that the secret to a good marriage is... *don't change*. In other words, be the person you were when you were merely *dating*. Don't stop paying attention. Don't stop being kind. Don't gain 50 pounds. Don't stop flirting. Stay passionate, stay sexy, stay caring. Answer their calls. Unfortunately, too many companies are all candle-lit dinners, fine wine, and "let's talk about *you*" until the deal is sealed. Once they have you (i.e. you became a paying customer), you realize you got a bait-and-switch relationship.

This is such a big bowl of wrong. I don't understand this in *personal* relationships, and I don't understand it in *business-to-customer* relationships. Shouldn't you treat the people you're in a relationship with *better* than you treat anyone else? Shouldn't you treat your *existing* customers better than the ones who've given you nothing?



Most companies would never outsource their sales reps, but we all know what happens with most tech support.

Most companies would never make a brochure with the same (lack of) quality in the product manual.

Most companies would never make their main website as uninviting as the tech support site.

If we want passionate users, we should take a lesson from successful marriages and *keep the spark alive*. Just because they're now a "sure thing" doesn't mean we take them for granted.

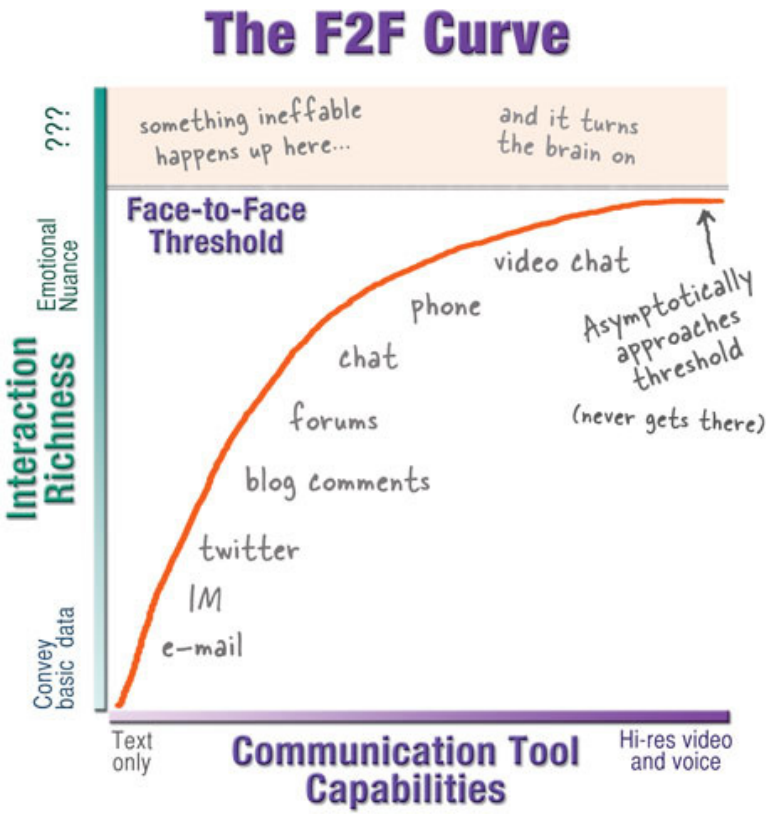
Besides, if we shift that marketing and ad budget from pre-sales to *post*-sales, we won't have to worry about getting new customers. Our loyal, cared-for customers will take care of that.

http://headrush.typepad.com/creating_passionate_users/2007/02/too_many_compan.html

Face-to-Face Trumps Twitter, Blogs, Podcasts, Video...

By Kathy Sierra on March 15, 2007

SXSW Interactive had more attendees than ever before. A lot more. But here's the confusing part: the people attending are the same people who create and evangelize the tools that make attending totally unnecessary. I started my keynote by asking if anyone was live-blogging. Hands shot up across the room. Someone yelled "Twitter!" The whole thing was recorded on video and audio. So... if nobody *needed* to be there, *why were they*?



Lots of people claimed it was all about the side/hallway conversations. But chat and Twitter and all the other non-face-to-face tools are pretty damn good at letting you have conversations. Some said it was for the free beer, but seriously, if you needed to come to Austin, pay a few hundred bucks, and stand in lines just to get a beer, you have other issues. The only *logical* excuse was for those few (or perhaps more) singles who were hoping for some... action. So it must be something *illogical*, or rather--some ineffable human thing that defies a simple rational explanation.

The point is, face-to-face still matters. And in fact all our globally-connecting-social-networking tools are making face-to-face *more*, not *less* desirable. Thanks to the tools y'all are building, we now have more far-flung friends--including people we've never met f2f--than ever before. We now have *more* people we want to connect with in the human world, often after years of electronic-only contact.

All we (and the scientists researching this) know is that there is something we still haven't managed to replicate in a meaningful way, even with the highest-resolution video conferencing tools. One theory is that it has something to do with smell. Whatever it is, it exerts a powerful pull on our legacy brain... a brain that still has no idea we aren't living in caves where human contact and social face-to-face interaction are key to our survival.

The most underrated benefit of the face-to-face effect of conferences is INSPIRATION.

For me, the single biggest reason to attend an event like SXSW is the feeling of motivation and--as David Seah so aptly put it --"Rededication". Almost everyone I talked with at SXSW said they were newly inspired. Was it from the ideas they were hearing in the sessions? Some of it, sure. But again, those same ideas are going out to everyone with a browser. No, there's more to it. There's mirror neurons, for one thing, and the effect of emotional contagion that happens when you're around a pile of people who share the same interest and enthusiasm. *Everyone* comes out re-energized. *And you don't need to go to SXSW to get that benefit!* Simply attending *any* live event--from the three-person lunch meetup to the 100-person local user group can give you the most positive effect of being at an event like SXSW.

In my talk, I mentioned two implications of the importance of face-to-face:

1) We should encourage our (human) users to get together in the offline world.

2) We should add more human-ness to the interactions in our software.

In this post, I want to mention a few ideas for the first one.

Get your users to meet other users in the real world!

Where there is passion around an activity or product or service or cause, there are always people wanting to connect with others who share that passion. The more we can help put our users together with one another offline, the more likely we are to get--or increase--user passion.

Suggestions:

1) Put together a "How To Start A Local User Group or Club" document.

Include tips on things like finding a space, topic ideas, and getting speakers. (If anyone has one of these, please let us know!)

2) Offer free materials for the user groups

User group meetings often start or end with prize drawings; give the user group leaders plenty of swag for the meetings. It'll make the leaders look good, etc.

3) Treat your user group leaders like royalty

Sun puts JUG (Java User Group) leaders on a pedastal--helping them promote their groups, giving them special receptions at the annual JavaOne conference, etc.

4) Instead of a traditional user group, provide guidelines for a Study Group

(Or a related book club.) Collect advice and lessons from other existing groups. Provide a list of suggested books to read, and 6-months' worth of topic plans. For example, "Month One: read [insert book related to your domain], and have each attendee discuss the following key points..."

5) Hold a very low-cost annual weekend conference.

Make it ridiculously easy for people to get there. Find sponsors to help. Even better if you hold several mini-conferences a year, in different locations.

6) Encourage users to start a local BarCamp (or other *Camp).

Direct your users to the BarCamp Wiki where they can learn how to do it.

7) Use Meetup.com as a resource!

8) If you already have online user forums, enlist moderators to try to form an offline meetup.

This is often one of the best places to start.

9) Hold special cocktail receptions/parties for user group leaders at industry conferences in your domain.

10) Advertise/promote your user group events on your main page!

Remember, passionate users MUST connect with others who share that passion, so this is not a nice-to-have... it's an essential part of any product, service, or cause for which people are passionate.

Bottom line: Face-to-Face matters, and the more people we meet *online*, the more people we now want to connect with *offline*. Perhaps one day in the future, the technology will finally catch up with real-life and we'll get the same brain/health benefits from a non-real-world experience. Personally, I hope not. I'd rather see technology that lets us come together in the real world as cheaply and easily as possible, despite wide geographic distances.

And to all who bothered to come to my SXSW talk when you clearly didn't *need* to, I so SO much appreciate it. I've only recently been *speaking* at conferences, but I've been attending them for almost 20 years like a junkie. And I don't even go to the parties. I go because I always come back motivated, even by the things I had begun to take for granted.

You don't *need* to go to SXSW to take advantage of the ideas there... just read the coverage and listen to the podcasts. But to get the *real* benefit of SXSW without being there, find a local group of people to meet up with! Even if it's three people having a coffee morning, it's important. A lot more important than most of us twittering, IM'ing, blogging, video-chatting folks like to acknowledge.

http://headrush.typepad.com/creating_passionate_users/2007/03/sxsw_interactiv.html

Is Twitter TOO good?

By Kathy Sierra on March 16, 2007

Twitter scares me. For all its popularity, I see at least three issues: 1) it's a near-perfect example of the psychological principle of *intermittent variable reward*, the key addictive element of slot machines. 2) The strong "feeling of connectedness" Twitterers get can *trick* the brain into thinking its having a meaningful social interaction, while another (ancient) part of the brain "knows" something crucial to human survival is missing. 3) Twitter is yet another--potentially more dramatic--contribution to the problems of always-on multi-tasking... you can't be Twittering (or emailing or chatting, of course) and simultaneously be in deep thought and/or a flow state.

[Disclaimer: I'm SO in the minority on this one... it looks like about a hundred-to-one in favor of Twitter, so I'm most likely way wrong on this one (but it doesn't stop me from trying). And this post is mostly a mashup of a variety of earlier posts I've made on related subjects.]

I'll look at each of the three points in more detail:

1) The Twitter Slot Machine

One of Skinner's most important discoveries is that behavior reinforced intermittently (as opposed to consistently) is the most difficult to extinguish. In other words, intermittent rewards beat predictable rewards. It's the basis of most animal training, but applies to humans as well... which is why slot machines are so appealing, and one needn't be addicted to feel it.

From a Time magazine feature story on multitasking:

> *Patricia Wallace, a techno-psychologist,...believes part of the allure of e-mail--for adults as well as teens--is similar to that of a slot machine." You have intermittent, variable reinforcement," she explains. "You are not sure you are going to get a reward every time or how often you will, so you keep pulling that handle."*

2) The feeling of connectedness

The biggest benefit most people seem to be deriving from Twitter is the ability to feel more connected to others. Carson Systems' Lisa put it this way in a comment to Tara Hunt's defense of Twitter:

> *"Twittering fills in those gaps...recording our friends' feelings, geographic location and actions as if we were spookily almost there. That makes us feel *really* connected..."*

Is this really a *good* thing?

Probably, yes. For most people, perhaps. But I think it's worth a critical look as opposed to an automatic connected-is-awlays-implicitly-good response. UCSF neurobiologist Thomas Lewis claims that if we're not careful, we can *trick* a part of our brain into thinking that we're having a real social interaction--something crucial and ancient for human survival--when we actually aren't. This leads to a stressful (but subconscious) cognitive dissonance, where we're getting *some* of what the brain thinks it needs, but not enough to fill that whatever-ineffable-thing-is-scientists-still-haven't-completely-nailed-but-might-be-smell. He didn't make this claim about Twitter... I attended his talk at The Conference on World Affairs, and he was addressing e-mail, chat, and even television (brain recognizes it's looking at "people", and feels it *must* be having a

social connection (GOOD), but yet it knows something's missing (BAD).

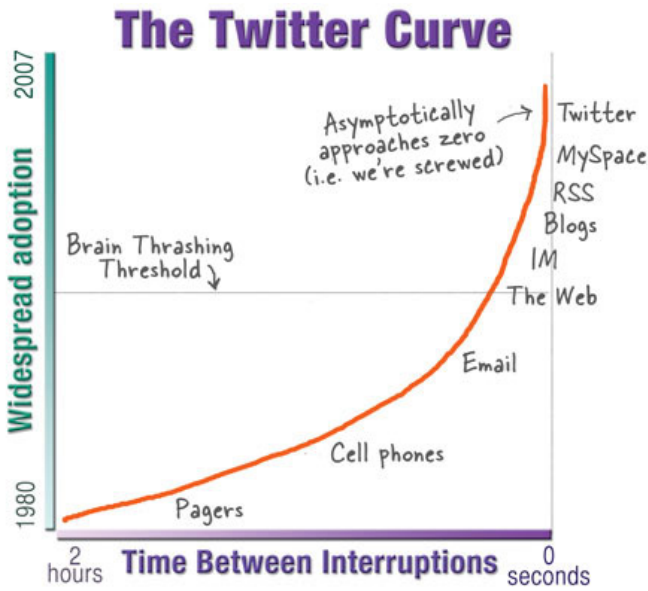
Dr. Lewis cited a ton of studies which I didn't write down, so you can take this with a grain of salt. Plus, I'm extending his issues from e-mail and chat to Twitter. But part of the reasons he talks about are that our brain has evolved an innate ability to interpret body language, facial expression, tone of voice, etc. so the brain *expects* these channels of information and becomes distressed when the social interaction appears to be there, but these innate, legacy-brain pieces are missing.

Again, this doesn't mean that it's not worth it and highly valuable for people TO stay connected to far-flung family and friends, I'm just saying that it's worth a look at whether that might be lulling some folks into a false sense of "I'm connected" at the expense of *real-life* connections.

Coffee with your next-door neighbor could do more for your brain than a thousand Twitter updates.

While this same argument has been going around forever, and is the same claim made about television, that doesn't make it untrue. (There's that study about the isolated Canadian village whose collective IQ went down once cable finally came to the village... Lewis cites it in his talks, although I can't find it referenced online).

Ironically, services like Twitter are simultaneously leaving some people with a feeling of *not* being connected, by feeding the fear of not being in the loop. By elevating the importance of being "constantly updated," it amplifies the feeling of missing something if you're not checking Twitter (or Twittering) with enough frequency.



3) Twitter is the best/worst cause of continuous partial attention

From an earlier post of mine:

Worst of all, this onslaught is keeping us from doing the one thing that makes most of us the happiest... being in flow. Flow requires a depth of thinking and a focus of attention that all that context-switching prevents. Flow requires a challenging use of our knowledge and skills, and that's quite different from mindless tasks we can multitask (eating and watching tv, etc.) Flow means we need a certain amount of time to load our knowledge and skills into our brain RAM. And the more big or small interruptions we have, the less likely we are to ever get there.

And not only are we stopping ourselves from ever getting in flow, we're stopping ourselves from ever getting really good at something. From becoming experts. The brain scientists now tell us that becoming an expert is not a matter of being a prodigy, it's a matter of being able to focus.

We're already seeing a backlash response to info overload, and it seems like a good chunk of Web 2.0 VC investments are going to companies that promise to help us get/stay organized. There's a reason 43 Folders is a Top 100 blog, and it's got to be more than just Merlin Mann's good looks ;)

Lots of people are talking about this, and perhaps nobody more eloquently than Linda Stone:

"To pay continuous partial attention is to pay partial attention -- CONTINUOUSLY. It is motivated by a desire to be a LIVE node on the network. Another way of saying this is that we want to connect and be connected. We want to effectively scan for opportunity and optimize for the best opportunities, activities, and contacts, in any given moment. To be busy, to be connected, is to be alive, to be recognized, and to matter.

We pay continuous partial attention in an effort NOT TO MISS ANYTHING. It is an always-on, anywhere, anytime, any place behavior that involves an artificial sense of constant crisis. We are always in high alert when we pay continuous partial attention. This artificial sense of constant crisis is more typical of continuous partial attention than it is of multi-tasking."

Bottom line(s):

Do I think Twitter has benefits? Clearly, and Tara does a great job of defining them (although not everyone agrees that these things are all *benefits*, they are for her and that's what matters).

Do I think people can use Twitter responsibly, without letting it get out of control or become too much of a distraction or encourage the same kind of voyeurism that makes tabloid news and TV so pervasively popular in the US?

Yes, definitely.

All I'm saying is that beyond the hype, we should consider just how far down the rabbit hole of always-on-attention we really want to go.

I am not in the target audience for Twitter--I am by nature a loner. I don't *want* to be *that* connected. And I also have a huge appreciation for the art of *keeping the mystery alive*. I don't *want* to know *that* much about so many people, and I sure don't want people to know that much about me... mundane or otherwise. So, that puts me in the minority, and my Twitter fears are probably based solely on my own--quirky and less common--personality traits.

http://headrush.typepad.com/creating_passionate_users/2007/03/is_twitter_too_.html

How to host a product/feature design party

By Kathy Sierra on March 18, 2007

Want to design the next great web app? Upgrade your product, but can't decide what to add or change? Add a new feature to your product, but can't decide how to implement it? Forget

focus groups. Forget endless meetings and brainstorming sessions. Throw an ultra-rapid-design party, and do it in a single day. This approach exploits the wisdom-of-crowds through a process of enforced idea diversity and voting, so no consensus, committe, or even agreement is needed. And it's way more fun.

The Product Design Dinner Party takes 9 people, a pile of diverse "inputs", and has each of the 9 people voting on--and pitching--one another's ideas to continuously reconfigured groups of 3 people, letting the best ideas rise to the top. The process is a little complicated, but it's derived/modified from an existing rapid-prototyping design I'll talk about later in the post.

The basic idea looks like this, although there are a million ways to modify it:

1) Pick 9 people, ideally from different parts of your company and including some customers. (If you don't have a company yet, pick 9 friends--preferably those who don't know each other well)

2) Buy/borrow/find at least 20 "input materials" including books, magazines, a short film, graphic novels, etc. (a list of possibilities is a little lower in this post)

3) Assign (randomly) at least 2 "inputs" to each person. Do NOT let them choose (it's important they not be allowed to gravitate toward things they're already comfortable with)

4) Give the group 30 minutes to generate 4 ideas (if it's a feature/upgrade party, then 4 different features or feature sets... if it's a feature implementation party, then 4 different ways to implement the already-decided feature, etc.) These 4 ideas don't have to come directly from their input materials, although participants should be highly encouraged to describe at least one new thing they learned that inspired their idea.

5) Round One begins: split into 3 groups of 3 people (see chart below). Each person gets no more than 10 minutes to "pitch" four ideas to the other two in their group. There are 12 total ideas for this group, so allow about 30 minutes. Record (anonymously) the selections of each person, which represent a "vote" for the ideas.

6) At the end of Round One, each person must select their two favorite ideas from each of the other two members of their group. So if Group One had Fred, Mary, and Sue... then Fred

must select his two favorite ideas from the four that Mary pitched, and his two favorites that Sue pitched.

7) Round Two begins: reconfigure the groups so that each person is now with different people (see chart below). Instead of pitching their *own* four ideas, each person pitches the four ideas they chose from their previous group members. Again, they have about 10 minutes to pitch the four ideas. Remember, the point is that each person is no longer pitching their own ideas!

8) At the end of Round Two, each person must again select their two favorite ideas from each of the other two members of this new group. Record (anonymously) the selections of each person, which represent a "vote" for the ideas.

9) Round Three begins: reconfigure the groups again. Each person in the group now pitches the four ideas (two from each of the two members of their most recent group) they chose in the previous (Round Two) round.

10) At this point, each person has pitched a total of 12 ideas:

* Round One: pitch your own four ideas

* Round Two: pitch four ideas from your Round One group to your new Round Two group -- two ideas from each of your previous group's other members.

* Round Three: pitch four ideas from your Round Two group to your new Round Three group, as before.

11) At the end of Round Three, *again* each person selects their top two favorite ideas from the ones pitched by the other two members. Record these as a vote.

12) You should now have a total of 108 votes. Choose the top 9 vote-getters (you'll have to be creative about tie-breaking... you could choose more than 9, for example).

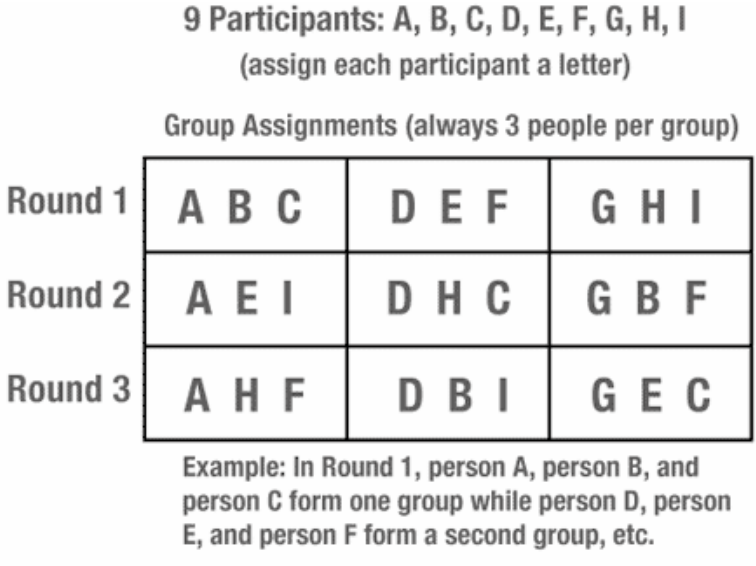
13) Give each person a copy of the 9 ideas, and send them back for another round of "inputs." Again, assign each person *different* materials from the ones they used at the beginning.

14) Give the participants 30 minutes to use their inputs and flesh out a single idea from the nine. Their one idea can be a modified version of one of the nine, based on their "research." Their one idea could be a mashup of two or more of the nine ideas. It cannot, however, be something completely new. Participants should be prepared to explain how something they

got from their inputs helped in some way (not an absolute requirement).

15) Now it's up to you what to do with the ideas. You might choose just one, or take all 9 "winners" with their pitches back to another person or group, etc.

Group Configurations (just an idea to get you started):



While it might be hard to believe a process like this could lead to any useful ideas, it's actually derived from a well-desiged, heavily-field-tested rapid-prototyping/development process from one of the leading training consultants on the planet, Thiagi. Granted, that doesn't mean my modifications haven't completely messed it up, but the main goals and benefits of doing it this way are:

1) Time constraints

Constraint-fueled creativity is something we've talked about earlier, so I won't discuss it here.

Build something cool in 24 hours

Creativity on Speed

How to make something amazing right now

and a little in Don't wait for the muse

2) Forced lack of attachment

By having to pitch someone *else's* ideas instead of your own (after Round One), it keeps people from getting stuck/married/attached to their own idea.

3) Random, outside-your-domain inputs

By having to use pre-selected (and pre-assigned) materials from outside your domain, participants have a better chance for a diversity-driven inspiration.

The whole thing is based on the assumption that you have all the knowledge you need -- the *wisdom* within your own company and your customers... you just need a way to tap into it that doesn't dilute the idea (as design-by-consensus would do) or prevent innovation (as design-by-listening-to-customers would do).

Ideas for "input materials"

Books on a wide range of topics outside your domain including architecture, astronomy, pop culture, filmmaking, comic books, wedding planning, education, children's book, romance-novel-writing, crafts magazine, travel book, sports, history, environment, etc.

If it's a software product, you might assign people to look at a variety of pre-chosen sites or web apps that are way outside your domain.

I've used this in the training world -- as a tool for learners to help come up with what they ought to be learning, but I've never used it in the way I've described here. I'm looking forward to trying it...

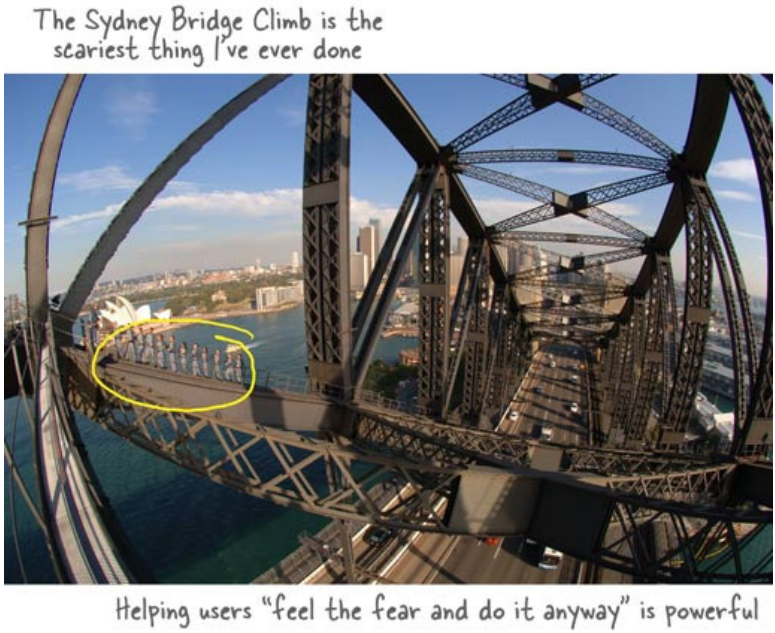
(And yes, I took a little artistic license with the photo at the top--pizza and coke might be better than alcohol. Then again...)

I'd love to hear ideas for modifying this, or from anyone who's done anything like this!

http://headrush.typepad.com/creating_passionate_users/2007/03/how_to_host_a_p.html

Helping users "feel the fear and do it anyway"

By Kathy Sierra on March 18, 2007



We've said before that *reducing* fear might be a killer app... making something users were previously afraid of feel less threatening. Wesabe does this for personal finances. Dr. Laurie Kemet does this for a trip to the dentist. And Electric Rain does this for 3D. Our books try to do this for programming. But what about a step beyond that... where you help them do something that just IS *really, seriously, scary*? Making only things which are friendly and easy is *not* the holy grail of design.

Reduce my fear or guilt, and I'll be grateful. Help me do something that really IS scary, and I'll be grateful *and exhilarated*. I'll be forever changed, and your company, product, or service will be linked to that change. To *reduce* fear means taking something *perceived* as scary and showing users that it's not. But not everything can be made to appear friendly and easy and safe. Like Apple's Logic. The learning curve is steep, it looks overwhelming and intimidating, but the payoff can be high. What if instead of removing advanced features that make a

product inherently daunting, it's OK to say to users, "This IS hard. Really, frickin' hard. But we'll get you through it."

Sometimes, with some products, it's OK to say, "We can't make this any easier or less scary, but we *can* help you come out the other side."

A short time ago I went on the Sydney Bridge Climb. At night. It was the most frightening thing I've ever done. But when it was over, I felt braver, stronger, and *different*. I'll never forget the Bridge Climb, and I'll probably be recommending it for the rest of my life.

We can reduce guilt:

We can reduce fear:

But helping a user be afraid and do it anyway is a powerful force. We shouldn't be too quick to over-simplify a product or experience. Of course, it's up to us to get our users through the big, challenging, thing--there's a big responsibility for stellar documentation and support. And we're talking *moral* support, not just *tech* support, so building a user community is even more important with something really, really, scary.

I've talked before about the benefits to *us* when we do something scary, but maybe we can help give those benefits to our advanced (or trying to be) users.

So, what scary thing have YOU done lately?

http://headrush.typepad.com/creating_passionate_users/2007/03/helping_users_f.html

Is your app an ass-kisser?

By Kathy Sierra on March 20, 2007

If your app was an employee, what kind of employee would it be? When it's employee performance review time, how would you rate it? These are just a few of the apps I've worked with recently...

The Paper Hat Guy

Friendly... very, very friendly. But can't do much, and you wouldn't trust him with anything important.

Brilliant but tempermental

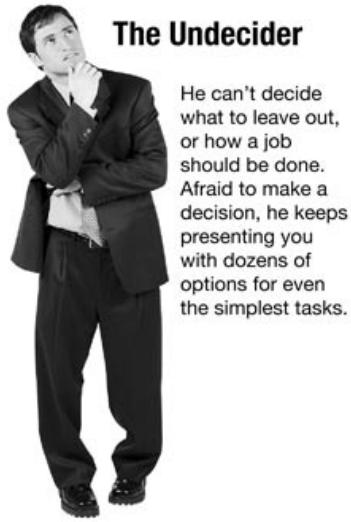
Can do amazing work, but the slightest thing can set him off. Difficult to keep happy. Pain in the ass, but worth it... until someone else comes along who can do what he does without all the "issues".

Anal-Retentive Guy

Meticulous but inflexible. There is only ONE RIGHT WAY to do things, and if you don't do it perfectly, he'll make you do it again. And this time it better be *exactly* the way he wants it. Assuming you can ever figure out what that is...

The Show-off

Does amazing tricks, just not the things you actually want him to do



What other app/product employee-types are there? Know any apps that need an employee appraisal?

http://headrush.typepad.com/creating_passionate_users/2007/03/is_your_app_an_.html

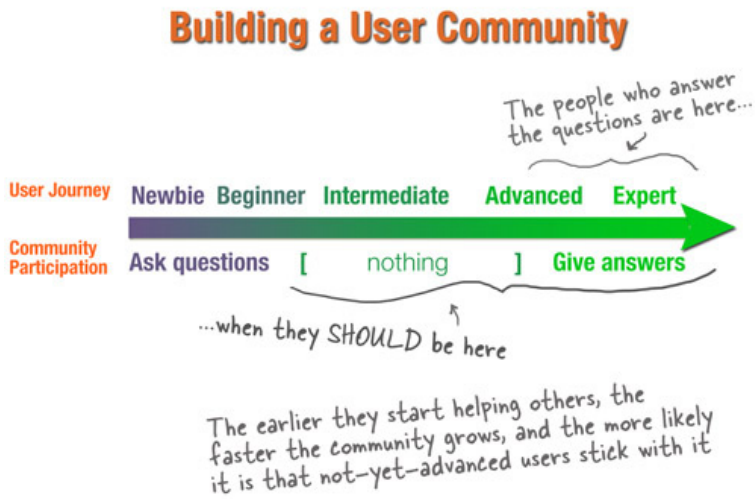
User Community and ROI

By Kathy Sierra on March 21, 2007

Every time I give a talk, someone always asks, "That's all good and nice that helping users learn is the key to creating passionate users... but who's going to do all that extra work? Who's going to make the extra tutorials and better docs?" Answer: *your user community*. Think about all the things a strong user community can do for you: tech support, user training, marketing (evangelism, word of mouth), third-party add-ons, even new product ideas. And that's not including any extra sales you might make on community/tribe items like t-shirts, stickers, and other gear.

Yes, there's still a budget... but we've all seen third-party fan/user groups that got *no* support at all from "the mother ship" and yet thrived and gave users a level of support and training the company didn't provide. But there's still that little of issue of getting users *involved*, and for that--the single biggest factor is getting users involved at a much earlier path on their learning journey than typically happens.

This picture is from an earlier post:



In [Building a User Community Part 1] we talked about the importance of not only a strict "There Are No Dumb Questions" policy, but also an even more dedicated "There Are No Dumb *Answers*" message.

Today, this post will offer a few more tips on how to use your marketing budget (tiny as it may be) to build, support, and grow a user community from the beginning.

* Host some kind of discussion forum (can include chat, wikis, and blogs as well), and do whatever it takes to get people there as soon as possible, ideally while the thing is still in beta (but it's never too late to start!)

* Look on *other* third-party forums where users are discussing (which usually means struggling) your product, and find the most active people. Reach out to your earliest adopters or strongest new users and offer them *non-paid* incentives for becoming active. Chances are, if you have any users at all and your product is even the least bit complicated, people are

discussing it *somewhere*. This could be anywhere from Amazon product reviews to technical discussion boards and even comments on related blogs.

* Make these folks life-time "charter members" with special privileges and recognition as 'founders' that nobody else will ever get.

* Have levels and rewards for participating (but again, not *money*--that totally changes the motivation, or at least the *perceived* motivation). The rewards can simply be status, early access to betas, and especially restricted access to the developers where they can discuss their ideas or at least listen to the engineers and designers describe why they made the choices they did, etc. [Don't reward people for post quantity alone... if post-count is the only criteria, you end up with a zillion useless posts]. Study successful user group communities for examples (like, say, javaranch.com--3/4 million unique visitors a *month*).

* *Teach* users how to help other members by creating documents (or getting other users to write them) on how to ask and answer questions in the most productive way.

* Include some just-for-fun activities in your community, like one (usually ONLY one) totally off-topic forum.

* Make sure there are interesting, easy-access ways for users to get to know more about one another. Be SURE to have user profile pages that include gender, photos, and some other personal info in addition to the specifics related to this particular community. Which leads to...

* Encourage members to meet *offline*! Hold a dirt-cheap User's Conference, ideally in more than one city, to get things started. Start a forum from the people who sign-up for the conference, and offer user group or forum leaders free entry to the event (and be sure to have a private user group or forum leader cocktail reception). Tips for that are in this recent post on face-to-face). Create a document on *How To Start A User Group*, and make sure users know how to get it. There is a *great* series of posts on how to start a user group written by the guys behind the Edmonton .NET User Group. (Thanks guys)

* Encourage forum moderators or other community leaders to have their own private discussion space.

* Don't tolerate abuse of the beginners, but don't force the experts to have to put up with newbie issues. As any community matures, you must provide separate areas for newbies and experts... if the community culture is one of generosity and motivation, there will still be enough experts who *want* to spend time helping newbies.

* Why not help your top community leaders get a book deal? You never know... if it's a tech topic, direct them (or yourself) over to Wiley publisher Joe Wikert for some excellent and candid advice (search his archives, and you'll find everything from how to write a proposal, whether you need an agent, etc.)

* Consider starting a monthly "official" user group membership subscription, with something that *comes in the real mail* each month. Think about it. Think about how you feel when Fedex or UPS pulls up with that little Amazon box with the smile on the side. Each month, send them a newsletter or DVD. Where's the budget for *that* content? Get your users involved! Have *them* submit things, and use the small monthly membership fee to cover the cost of materials and mailing, etc. Maybe you can partner with a sponsor on this, to include other things in the monthly "kit."

* Create limited-edition, not-for-sale t-shirts, stickers, and other gear JUST for the founding community members (if you're just getting started in building a community). For ongoing communities, do the same thing and distribute them randomly, for free. Use the principle of "intermittent variable reward" that works so well to make slot machines and twitter so addicting ;)

* Make your community leaders or even just active participants HEROES. Create "superhero" Moo cards for them. Plaster their photos everywhere. (Cute story I heard from a reader here -- she met her husband online while they were both moderators for an Autodesk CAD forum.)

* Host an offline retreat just for the key community leaders. Can't afford to do what Microsoft does with its Search Champs? Can't afford to put people up at the "W"? Have a campout. Supply the marshmallows.

* Above all, keep teaching members to teach other members. Give *everyone* a crash course in learning theory. The better they become at helping others--the more skills they develop in

mentoring/tutoring others--the more meaningful and motivating it is for them to *keep on doing it*..

These are just a few tips for now. Stay tuned for more. And of course, please add your own... while I have quite a lot of user group/community experience having launched several groups from scratch, they were all technology-related, and many of you are from very different domains.

http://headrush.typepad.com/creating_passionate_users/2007/03/user_community_.html

The End?

EDITOR'S NOTE: As of April 2, 2007, Kathy Sierra is no longer posting to Creating Passionate Users due to a number of harassing messages and death threats that she was receiving. Although Kathy posted several messages to the blog about the messages and circumstances surrounding the situation, they have not been reprinted here due to the graphic nature of the content. For those interested, you can visit the pages at:

http://headrush.typepad.com/creating_passionate_users/2007/03/as_i_type_this_.html

http://headrush.typepad.com/creating_passionate_users/2007/04/updatejoint_sta.html