Coding Horror

http://www.coding-horror.com/blog/

So You Want to be a Programmer

I didn’t intend for ***Please Don’t Learn to Code*** to be so controversial, but it seemed to strike a nerve. Apparently a significant percentage of readers stopped reading at the title.

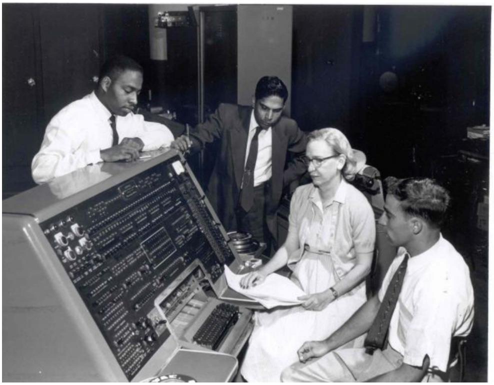
So I will open with my own story. I think you’ll find it instructive.

> My mom once told me that the only reason she dated my father is because her mother told her to stay away from that boy, he’s a bad influence.
>
> If she had, I would not exist.

True story, folks.

I'd argue that the people who *need* to learn to code will be spurred on most of all by honesty, not religious faith in the truthiness of code as a universal good. Go in knowing both sides of the story, because ***there are no silver bullets in code***. If, after hearing both the pros and cons, you still want to learn to code, then by all means *learn to code*. If you're so easily dissuaded by hearing a few downsides to coding, there are plenty of other things you could spend your time learning that are more unambiguously useful and practical. Per Michael Lopp, you could ***learn to be a better communicator***. Per Gina Trapani, you could ***learn how to propose better solutions***. Slinging code is ***just a tiny part of the overall solution*** in my experience. Why optimize for *that?*

On the earliest computers, everyone *had* to be a programmer because there was no software. If you wanted the computer to do anything, you wrote code. Computers in the not so distant past booted directly to the ***friendly blinking cursor of a BASIC interpreter***. I view the entire arc of software development as a field where we programmers spend our lives writing code so that our fellow human beings no longer *need* to write code (or even worse, become programmers) to get things done with computers. So this idea that “everyone must know how to code” is, to me, going backwards.



I fully support a push for basic Internet literacy. But in order to be a competent driver, does everyone need to know, in detail, how their automobile works? Must we teach all human beings the basics of being an auto mechanic, and elevate shop class to the same level as English and Mathematics classes? Isn't knowing how to change a tire, and when to take your car in for an oil change, sufficient? If your toilet is clogged, you shouldn't need to take a two week in depth plumbing course on ***toiletcademy.com*** to understand how to fix that. Reading a single web page, ***just in time***, should be more than adequate.

What is code, in the most abstract sense?

> code (kōd) …
>
> 3. a. A system of signals used to represent letters or numbers in transmitting messages.
>
>   b. A system of symbols, letters, or words given certain arbitrary meanings, used for transmitting messages requiring secrecy or brevity.
>
> 4. A system of symbols and rules used to represent instructions to a computer…

> — *The American Heritage Dictionary of the English Language*

Is it punchcards? Remote terminals? Emacs? Textmate? Eclipse? Visual Studio? C? Ruby? JavaScript? In the 1920s, it was considered important to learn how to use slide rules. In the 1960s, it was considered important to learn mechanical drawing. None of that matters today. I'm hesitant to recommend any particular approach to coding other than the fundamentals as outlined in ***Code: The Hidden Language of Computer Hardware and Software***, because I'm not sure we'll even recognize coding in the next 20 or 30 years. To kids today, perhaps coding will eventually resemble ***Minecraft***, or ***building levels in Portal 2***.

But everyone should try writing a *little* code, because it somehow sharpens the mind, right? Maybe in the same abstract way that ***reading***

the entire Encyclopedia Brittanica from beginning to end does. Honestly, I'd prefer that people spend their time discovering what problems they love and find interesting, first, and researching the hell out of those problems. The toughest thing in life is not learning a bunch of potentially hypothetically useful stuff, but figuring out ***what the heck it is you want to do***. If said research and exploration leads to coding, then by all means learn to code with my blessing … which is worth exactly what it sounds like, nothing.

So, no, I don't advocate learning to code for the sake of learning to code. What I advocate is **shamelessly following your joy**. For example, I received the following email yesterday.

> I am a 45 year old attorney/C.P.A. attempting to abandon my solo law practice as soon as humanly possible and strike out in search of my next vocation.

I am actually paying someone to help me do this and, as a first step in the "find yourself" process, I was told to look back over my long and winding career and **identify those times in my professional life when I was doing something I truly enjoyed**.

Coming of age as an accountant during the PC revolution (when I started my first "real" job at Arthur Andersen we were still billing clients to update depreciation schedules manually), I spend a lot of time learning how to make computers, printers, and software (VisiCalc anyone?) work. This quasi-technical aspect of my work reached its apex when I was hired as a healthcare financial analyst for a large hospital system. When I arrived for my first day of work in that job, I learned that my predecessor had bequeathed me only a one page static Excel spreadsheet that purported to “analyze” a multi-million dollar managed care contract for a seven hospital health system. I proceeded to build my own spreadsheet but quickly exceeded the database functional capacity of Excel and had to teach myself Access and thereafter proceeded to stretch the envelope of Access’ spreadsheet capabilities to their utmost capacity – I had to retrieve hundreds of thousands of patient records and then perform pro forma calculations on them to see if the proposed contracts would result in more or less payment given identical utilization.

I will be the first to admit that I was not coding in any professional sense of the word. I did manage to make Access do things that MS technical support told me it could not do but I was still simply using very basic commands to bend an existing application to my will. **The one thing I do remember was being happy.** I typed infinitely nested commands into formula cells for twelve to fourteen hours a day and was still disappointed when I had to stop.

My experience in building that monster and making it run was, to date, my most satisfying professional accomplishment, despite going on to later become CFO of another healthcare facility, a feat that should have fulfilled all of my professional ambitions at that time. More than just the work, however, was the group of like-minded analysts and IT folks with whom I became

> associated as I tried, failed, tried, debugged, and continued building this behemoth of a database. I learned about Easter Eggs and coding lore and found myself hacking into areas of the hospital mainframe which were completely offlimits to someone of my paygrade. And yet, **I kept pursuing my "professional goals" and ended up in jobs/careers I hated doing work I loathed.**

Here's a person who a) found an interesting problem, b) attempted to create a solution to the problem, which naturally c) led them to learning to code. *And they loved it.* This is how it's supposed to work. I didn't become a programmer because someone told me learning to code was important, I became a programmer because ***I wanted to change the rules of the video games I was playing***, and learning to code was the only way to do that. Along the way, ***I too fell in love***.

> All that to say that as I stand at the crossroads once more, I still hear the siren song of those halcyon days of quasi-coding during which I enjoyed my work. My question for you is whether you think it is even possible for someone of my vintage to learn to code to a level that I could be hired as a programmer. I am not trying to do this on the side while running the city of New York as a day job. Rather, I sincerely and completely want to become a bona fide programmer and spend my days creating (and/or debugging) something of value.

Unfortunately, ***calling yourself a "programmer" can be a career-limiting move***, particularly for someone who was a CFO in a previous career. People who work with money tend to make a lot of money; ***see Wall Street***.

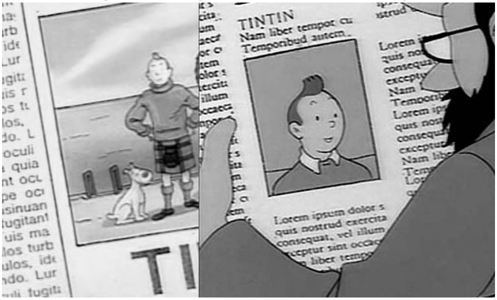
But this isn't about money, is it? ***It's about love***. **So, if you want to be a programmer, all you need to do is follow your joy and fall in love with code.** Any programmer worth their salt immediately recognizes a fellow true believer, a person as madly in love with code as they are, warts and all. ***Welcome to the tribe***.

And if you're reading this and thinking, "screw this Jeff Atwood guy, who is he to tell me whether I should learn to code or not", all I can say is: ***good! That's the spirit!***

Table 1 none

The Eternal Lorem Ipsum

If you've ***studied design*** at all, you've probably encountered ***Lorem Ipsum placeholder text*** at some point. Anywhere there is text, but the meaning of that text isn't particularly important, you might see Lorem Ipsum.



Most people recognize it as Latin. And it is. But it is arbitrarily rearranged and not quite *coherent* Latin, extracted from a ***book*** Cicero wrote in 45 BC. Here’s the complete quote, with the bits and pieces that make up Lorem Ipsum highlighted.

> Nemo enim ipsam voluptatem, quia voluptas sit, aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos, qui ratione voluptatem sequi nesciunt, neque porro quisquam est, qui dolorem ipsum, quia dolor sit amet, consectetur, adipisci[ng] velit, sed quia non numquam [do] eius modi tempora inci[di]dunt, ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis

> autem vel eum iure reprehenderit, qui in ea voluptate velit esse, quam nihil molestiae consequatur, vel illum, qui dolorem eum fugiat, quo voluptas nulla pariatur?
>
> At vero eos et accusamus et iusto odio dignissimos ducimus, qui blanditiis praesentium voluptatum deleniti atque corrupti, quos dolores et quas molestias excepturi sint, obcaecati cupiditate non provident, similique sunt in culpa, qui officia deserunt mollitia animi, id est laborum et dolorum fuga.

But *what does it all mean?* Here's an English translation with the same parts highlighted.

> Nor again is there anyone who loves or pursues or desires to obtain pain of itself, because it is pain, but occasionally circumstances occur in which toil and pain can procure him some great plea-

sure. To take a trivial example, which of us ever undertakes laborious physical exercise, except to obtain some advantage from it? But who has any right to find fault with a man who chooses to enjoy a pleasure that has no annoying consequences, or one who avoids a pain that produces no resultant pleasure?

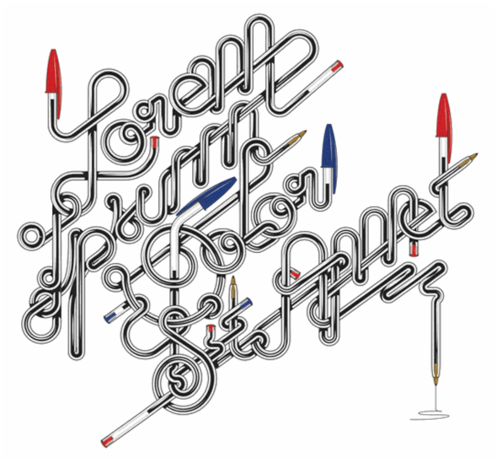
On the other hand, we denounce with righteous indignation and dislike men who are so beguiled and demoralized by the charms of pleasure of the moment, so blinded by desire, that they cannot foresee the pain and trouble that are bound to ensue; and equal blame belongs to those who fail in their duty through weakness of will, which is the same as saying through shrinking from toil and pain.

Of course the whole point of Lorem Ipsum is that the words aren't *supposed* to mean anything, so attempting to divine its meaning is somewhat … unsatisfying, perhaps by design. Lorem Ipsum is a specific form of what is generally referred to somewhat cheekily as ***"Greeking"***:

> Greeking is a style of displaying or rendering text or symbols, not always from the Greek alphabet. **Greeking obscures portions of a work for the purpose of either emphasizing form over details or displaying placeholders for unavailable content.** The name is a reference to the phrase "Greek to me", meaning something that one cannot understand, so that it might as well be in a foreign language.

So when you need filler or placeholder text, you naturally reach for Lorem Ipsum as the standard. The theory is that, since it's unintelligible, nobody will attempt to *read* it, but instead **focus on other aspects of the design**. If you put readable text in the design, people might think the text is important to the design, that the text represents the sort of content you expect to see, or that the text somehow itself needs to be copyedited and updated and critiqued.

(Regular readers of this blog may remember that I am fond of using ***Alice in Wonderland*** in this manner, when I need a bit of text to demonstrate something in a post.)



However, not everyone agrees that relying on a standard boilerplate greeked placeholder text is appropriate, even going so far as to ***call for the death of Lorem Ipsum***. I think it depends what you're trying to accomplish. I once noted that it's better to use real content to ***avoid Blank Page Syndrome***, for example.

There are quite a few websites that helpfully offer up the classic Lorem Ipsum text in various eminently copy-and-pastable forms.

Classic Lorem Ipsum

- ***The original site***
- ***With bold, tags, lists, quotes, code blocks, etc***
- ***With many HTML structure variants on a single page***
- ***With choice of paras, words, sentences***
- ***With download and gibberish options***
- ***With non-repeating text and diacriticals/punctuation added***
- ***With short/long paras and lists***

- ***With English, tags, headings, bold and italic***
- ***WordPress standard content oriented***

Beyond that, if you just want a bunch of, uh, *interesting* text to fill an area, there a lot – and I mean *a lot* – of websites to choose from. So many in fact that I was a little overwhelmed trying to index them all. I’ve tried to broadly categorize the ones I did find, below. If you know of more, feel free to leave a comment and I’ll update the list.

Novelty

- ***Meat***
- ***Vegetables***
- ***Vegan***

- ***Fish***
- ***Beer***
- ***Desserts***
- ***Hipster***
- ***Skaters***
- ***Space***
- ***Zombies***
- ***Comics***
- ***Moustaches***
- ***Baseball***
- ***Postmodernism***

- ***Marketing, Hillbilly, Metropolitan, Pseudo German, Pseudo-Code, Techno-Babble***

Clever English Tricks

- ***Random text in 16 different languages***

- ***Pangrams (and other oddities)***

- ***Random dictionary words***

- ***Homophonic transformations***

- ***Gibberish***

Literature

- ***Classic writers: Melville, Baum, Burroughs, Carroll, London, Verne, Wells***

- *Classic books: Bible, Kama Sutra, Fun with Mathematics, Alice in Wonderland, Wuthering Heights, Ulysses, All about Coffee*

Professions

- *Journalist*
- *Corporate*
- *50+ professions and fields*

Social Networks

- *Facebook, Digg, Twitter*
- *Twitter*

TV, Movies and Media

- *Samuel L. Jackson*
- *Arnold Schwarzenegger*

- ***Chuck Norris***
- ***Snoop Doggy Dogg***
- ***Charlie Sheen***
- ***Web 2.0 company names***
- ***Internet Memes***
- ***Arrested Development***
- ***The Big Lebowski***
- ***Geeky movies and TV shows***
- ***80s TV shows***
- ***TV theme songs***
- ***TV variety***

Possibly NSFW

- ***Swearing***

- ***Trolling***

Regional

- ***Yorkshire***
- ***Newfoundland***
- ***New Zealand / Australia***
- ***Spanish***

This is a lot to go through. If I had to pick a favorite, I'd say ***Fillerati*** because it's all dignified and stuff. But I think truer to the spirit of Lorem Ipsum are **definitely the** ***homophonic transformations***, which consistently *blow my mind* every time I attempt to read them. Isn't that the implied goal of any properly greeked text? You were one deliciously perverse professor of romance languages, ***Howard L. Chace***.

In today’s Pinteresting world, images are arguably more important than text. But **what is the Lorem Ipsum of images?** Is there even one? I guess you could just slap some Lorem Ipsum text in an image, but where is the fun in that? Anyway, there are also plenty of websites offering up placeholder images of various types to go along with your Lorum Ipsum placeholder text.

Images

- ***Creative Commons images***
- ***Flickr***
- ***Kittens***
- ***Dogs***
- ***Bears***

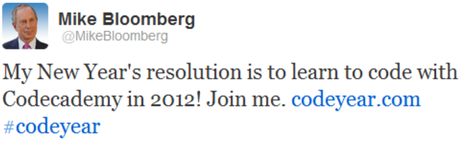
- ***Plain blocks of specific sizes, colors, text***
- ***More plain blocks of specific sizes, colors, text***
- ***Gray boxes***
- ***More gray boxes***
- ***Gray boxes on App Engine***
- ***Local JavaScript generated gray boxes***

I’m not sure the world needs any more Lorem Ipsum-alikes than we already have at this point. Like the market for ironic t-shirts, the Internet has ensured that our placeholder greeked text needs have not merely been met but *vastly* exceeded for the forseeable future. But after discovering all the creative things people have done with Lorem Ipsum, and text placeholders in general, it’s sure tempting to dream yet another one up, isn’t it?

Table 1 none

Please Don’t Learn to Code

The whole “everyone should learn programming” meme has gotten so out of control that the mayor of New York City actually ***vowed to learn to code in 2012***.



Mike Bloomberg
@MikeBloomberg

My New Year's resolution is to learn to code with Codecademy in 2012! Join me. codeyear.com #codeyear

A noble gesture to garner the NYC tech community vote, for sure, but if the mayor of New York City actually needs to *sling JavaScript code to do his job*, something is deeply, horribly, terribly wrong with politics in the state of New York. Even if Mr. Bloomberg did “learn to code”, with

apologies to Adam Vandenberg, I expect we'd end up with this:

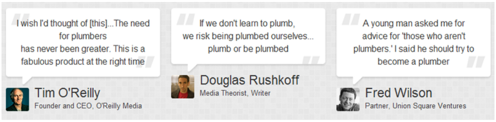
```
10 PRINT "I AM MAYOR"
20 GOTO 10
```

Fortunately, the odds of this technological flight of fancy happening – even in jest – are zero, and for good reason: the mayor of New York City will hopefully spend his time doing the job taxpayers paid him to do instead. According to the ***Office of the Mayor*** home page, that means working on absenteeism programs for schools, public transit improvements, the 2013 city budget, and … do I really need to go on?

To those who argue programming is an essential skill we should be teaching our children, right up there with reading, writing, and arithmetic: **can you explain to me how Michael Bloomberg would be better at his day to day job of leading the largest city in the USA if he woke up one morning as a crack**

Java coder? It is obvious to me how being a skilled reader, a skilled writer, and at least high school level math are fundamental to performing the job of a politician. Or at *any* job, for that matter. But understanding variables and functions, pointers and recursion? I can't see it.

Look, ***I love programming***. I also believe programming is important … in the right context, for some people. But so are a lot of skills. I would no more urge everyone to learn programming than I would urge everyone to learn plumbing. ***That'd be ridiculous, right?***



"I wish I'd thought of [this]...The need for plumbers has never been greater. This is a fabulous product at the right time"

Tim O'Reilly
Founder and CEO, O'Reilly Media

"If we don't learn to plumb, we risk being plumbed ourselves... plumb or be plumbed"

Douglas Rushkoff
Media Theorist, Writer

"A young man asked me for advice for 'those who aren't plumbers.' I said he should try to become a plumber"

Fred Wilson
Partner, Union Square Ventures

The "everyone should learn to code" movement isn't just wrong because it falsely equates coding with essential life skills like reading, writing, and math. I wish. It is wrong in so many other ways.

- It assumes that more code in the world is an inherently desirable thing. In my thirty year career as a programmer, I have found this … not to be the case. Should you learn to write code? No, I can't get behind that. You should be ***learning to write* as little code as possible**. Ideally none.

- It assumes that coding is the goal. Software developers tend to be software addicts who think their job is to write code. But it's not. ***Their job is to solve problems***. Don't celebrate the creation of code, celebrate the creation of *solutions*. We have way too many coders addicted to doing just one more line of code already.

- It puts the method before the problem. Before you go rushing out to learn to code, *figure out what your problem actually is.* Do you even have a problem? Can you explain it to others in a way they can understand? Have you researched the problem, and its possible solutions, deeply? Does coding solve that problem? Are you *sure?*

- It assumes that adding naive, novice, not-even-sure-they-like-this-whole-programming-thing coders to the workforce is a net positive for the world. I guess that’s true if you consider that ***one bad programmer can easily create two***

new jobs a year. And for that matter, ***most people who already call themselves programmers can't even code***, so please pardon my skepticism of the sentiment that "everyone can learn to code".

- It implies that there's a thin, easily permeable membrane between learning to program and getting paid to program professionally. Just look at these new programmers who got offered jobs at an average salary of **$79k/year** after ***attending a mere two and a half month bootcamp!*** Maybe you too can ***teach yourself Perl in 24 hours!*** While I love that programming is an egalitarian field where degrees and certifications are irrelevant in the face

> of experience, you still gotta ***put in your ten thousand hours like the rest of us***.

I suppose I can support learning a tiny bit about programming just so you can recognize what code is, and when code might be an appropriate way to approach a problem you have. But I can also recognize plumbing problems when I see them without any particular training in the area. The general populace (and its political leadership) could probably benefit most of all from a basic understanding of how computers, and the Internet, work. Being able to get around on the Internet is becoming a basic life skill, and we should be worried about fixing *that* first and most of all, before we start jumping all the way into code.

Please don’t advocate learning to code just for the sake of learning how to code.

Or worse, because of the fat paychecks. Instead, I humbly suggest that we spend our time learning how to …

- Research voraciously, and understand how the things around us work at a basic level.

- Communicate effectively with other human beings.

These are skills that ***extend far beyond mere coding*** and will help you in every aspect of your life.

w off all of your hard work from Stack Overflow, Gith

Table 1 none

This Is All Your App Is: a Collection of Tiny Details

Fair warning: this is a blog post about automated cat feeders. Sort of. But bear with me, because I'm also trying to make a point about software. If you have a sudden urge to click the back button on your browser now, I don't blame you. I don't often talk about cats, but ***when I do, I make it count***.

We've used automated cat feeders ***since 2007*** with great success. (My apologies for the picture quality, but it *was* 2007, and camera phones were awful.)

http://blogs.vertigo.com/personal/jatwood/Blog/Lists/Posts/Post.aspx?ID=25

Feeding your pets using robots might sound impersonal and uncaring. Perhaps it is. But I can't emphasize enough **how much of a daily lifestyle improvement it really is to have your pets** ***stop associating you with ritualized, timed feedings***. As my wife so aptly ***explained***:

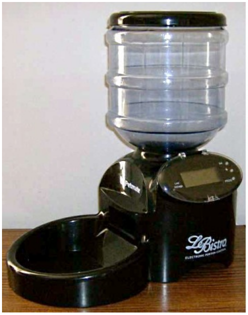
> I do not miss the days when the cats would come and sit on our heads at 5

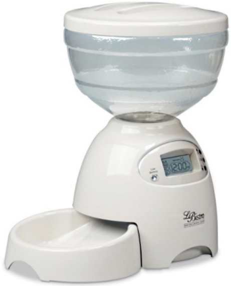
> AM, wanting their breakfast.

Me neither. I haven't stopped loving our fuzzy buddies, but this was also before we had onetwothree children. We don't have a lot of time for random cat hijinks these days. Anyway, once we set up the automated feeders in 2007, it was a *huge* relief to outsource pet food obsessions to machines. They reliably delivered a timed feeding at 8am and 8pm like clockwork for the last five years. No issues whatsoever, other than changing the three D batteries about once a year, filling the hopper with kibble about once a month, and an occasional cleaning.

Although they *worked*, there were still many details of the automated feeders' design that were downright terrible. I put up with these problems because I was so happy to have automatic feeders that worked at all. So when I noticed that the ***2012 version of these feeders*** appeared to be considerably updated, I went ahead and upgraded immediately on faith alone. After all, it had been nearly five years! Surely the company had improved their product a bit since then … right? Well, a man can dream, can't he?

When I ordered the new feeders, I assumed they would be a little better than what I had before.





The two feeders don't look so radically different, do they? But pay attention to the details.

- **The food bowl is removable.** It drove me crazy that the food bowl in the old version was permanently attached, and tough to clean as a result.
- **The food bowl has rounded interior edges.** As if cleaning the non-removable bowl of our old version wasn't annoying enough, it also had sharp interior edges,

which tended to accrete a bunch of powdered food gunk in there over time. Very difficult to clean properly.

- **The programming buttons are large and easy to press.** In the old version, the buttons were small watch-style soft rubber buttons that protruded from the surface. The tactile feedback was terrible, and they were easy to mis-press because of their size and mushiness.

- **The programming buttons are directly accessible on the face of the device.** For no discernable reason whatsoever, the programming buttons in the old version were under a little plastic clear protective “sneeze guard” flap, which you had to pinch up and unlock with your thumb before you could do any programming at all. I guess the theory was that a pet could somehow accidentally brush against the buttons and do … something … but that seems incredibly unlikely. But most of all, unnecessary.

- **The programming is easier.** We *never* changed the actual feed schedule, but just changing the time for daylight savings was so incredibly awkward and contorted we had to summarize the steps from the manual on a separate piece of paper as a "cheat sheet". The new version, in contrast, makes changing the time almost as simple as it should be. Almost.

- **There is an outflow cover flap.** By far the number one physical flaw of the old feeder: the feed slot invites curious paws, and makes it ***all too easy to fish out kibble on demand***. You can see in my original photo that we had to mod the feed slot to tape (and eventually bolt) a wire soap dish cover over it so the cats wouldn't be able to manual feed. The new feeder has a ***perfectly aligned outflow flap*** that I couldn't even dislodge with my finger. And it works; even our curious-est cat wasn't able to get past it.

- **The top cover rotates to lock**. On the old feeder, the top cover to the clear kibble storage was a simple friction fit; dislodging it wasn't difficult, and the cats did manage to do this early on with some experimentation. On the new feeder, the cover is slotted, and rotates to lock against the kibble storage securely. This is the same way the kibble feeder body locks on the base (on both old and new feeders), so it's logical to use this same "rotate to lock into or out of position" design in both places.

- **The feed hopper is funnel shaped**. The old feed hopper was a simple cylinder, and holds less in the same space as a result. When I transferred the feed over from the old full models (we had literally just filled them the day before) to the updated ones, I was able to add about 15-20 percent more kibble despite the device being roughly the same size in terms of floor space.

- **The base is flared**. Stability is critical; ***depending how adventurous your cats are***, they may physically attack the feeders and try to push them over, or hit them hard enough to trigger a trickle of food dispensing. A flared base isn't the final solution, but it's a big step in the right direction. It's a heck of a lot tougher to knock over a feeder with a bigger "foot" on the ground.

- **It's off-white**. The old feeder, like the Ford Model T, was available in any color customers wanted, so long as it was black. Which meant it did a great job of *not* blending in with almost any decor, and also showed off its dust collection like a champ. Thank goodness the new model comes in "linen".

These are, to be sure, a **bunch of dumb, nitpicky details**. Did the old version feed our cats reliably? Yes, it did. But it was also a pain to clean and maintain, a sort of pain that I endured weekly, for reasons that made no sense to me other than arbitrarily poor design choices. But when I bought the ***new version of the automated feeder***, I was shocked to discover that nearly every single problem I had with the previous generation was addressed. I felt as if the Petmate Corporation™ was actually listening to all the feedback from the people who used their product, and actively refined the product to address *our* complaints and suggestions.

My point, and I do have one, is that details matter. Details matter, in fact, *a hell of a lot.* Whether in automatic cat feeders, or software. As ***my friend Wil Shipley once said***:

This is all your app is: a collection of tiny details. This is still one of my favorite quotes about software. It's something we internalized heavily when building Stack Overflow. **Getting the details right is the difference between something that delights, and something customers *tolerate*.**

Your software, your product, is nothing more than a collection of tiny details. If you don’t obsess over all those details, if you think it’s OK to concentrate on the “important” parts and continue to ignore the other umpteen dozen tiny little ways your product annoys the people who use it on a daily basis – you’re not creating great software. Someone else is. I hope for your sake they aren’t your competitor.

The details are hard. Everyone screws up the details at first, just like Petmate did with the first version of this automatic feeder. And it’s OK to screw up the details initially, provided …

- you’re getting the primary function more or less right.

- you’re listening to feedback from the people who use your product, and actively refining the details of your product based on their feedback every day.

We were maniacal about listening to feedback from avid Stack Overflow users from the earliest days of Stack Overflow in August 2008. Did you know that we didn't even have *comments* in the first version of Stack Overflow? But it was obvious, based on user feedback and observed usage, that we desperately needed them. There are now, at the time I am writing this, ***1,569 completed feature requests***; that's more than one per day on average.

Imagine that. Someone who **cares about the details just as much as you do**.

ers (you!) with the best employers. You can search

Table 1 none

Buying Happiness

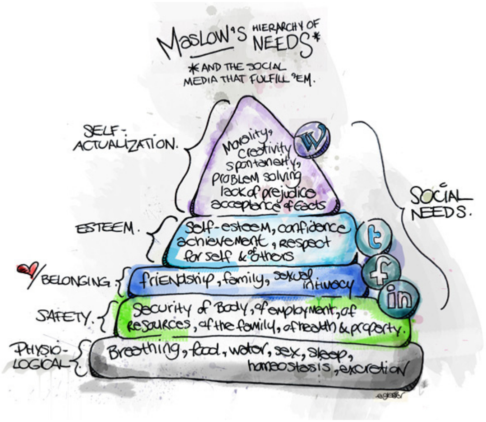
Despite popular assertions to the contrary, science tells us that ***money* can *buy happiness***. To a point.

> Recent research has begun to distinguish two aspects of subjective well-being. Emotional well-being refers to the emotional quality of an individual's everyday experience — the frequency and intensity of experiences of joy, stress, sadness, anger, and affection that make one's life pleasant or unpleasant. Life evaluation refers to the thoughts that people have about

> their life when they think about it. We raise the question of whether money buys happiness, separately for these two aspects of well-being. We report an analysis of more than 450,000 responses to the Gallup-Healthways Well-Being Index, a daily survey of 1,000 US residents conducted by the Gallup Organization. […] When plotted against log income, life evaluation rises steadily. **Emotional well-being also rises with log income, but there is no further progress beyond an annual income of ~$75,000.**

For reference, the ***federal poverty level*** for a family of four is currently **$23,050**. Once you reach a little over 3 times the poverty level in income, you've achieved peak happiness, as least far as money alone can reasonably get you.

This is something I’ve seen echoed in a number of studies. Once you have “enough” money to satisfy the basic items at the foot of the ***Maslow’s Hierarchy of Needs*** pyramid – that is, you no longer have to worry about food, shelter, security, and perhaps having a bit of extra discretionary money for the unknown – stacking even more money up doesn’t do much, if anything, to help you scale the top of the pyramid.


Maslow's hierarchy of needs*
* and the social media that fulfill 'em.
Self-actualization.
Morality, creativity, spontaneity, problem solving, lack of prejudice, acceptance of facts
Esteem.
Self-esteem, confidence, achievement, respect for self & of others
Belonging.
friendship, family, sexual intimacy
Safety.
Security of body, of employment, of resources, of the family, of health & property.
Physiological.
Breathing, food, water, sex, sleep, homeostasis, excretion
Social needs.
eglasser

But even if you're fortunate enough to have a good income, how you spend your money has a strong influence on how happy – or unhappy – it will make you. And, again, there's science behind this. The relevant research is summarized in ***If money doesn't make you happy, then you probably aren't spending it right*** (pdf).

> **Most people don’t know the basic scientific facts about happiness — about what brings it and what sustains it — and so they don’t know how to use their money to acquire it.** It is not surprising when wealthy people who know nothing about wine end up with cellars that aren’t that much better stocked than their neighbors’, and it should not be surprising when wealthy people who know nothing about happiness end up with lives that aren’t that much happier than anyone else’s. Money is an opportunity for happiness, but it is an opportunity that people routinely squander because the things they think will make them happy often don’t.

You may also recognize some of the authors on this paper, in particular Dan Gilbert, who also wrote the excellent book ***Stumbling on Happiness*** that touched on many of the same themes.

What is, then, the **science of happiness**? I'll summarize the basic eight points as best I can, but ***read the actual paper*** (pdf) to obtain the citations and details on the underlying studies underpinning each of these principles.

1. Buy experiences instead of things Things get old. Things become ordinary. Things stay the same. Things wear out. Things are difficult to share. But experiences are totally unique; they shine like diamonds in your memory, often more brightly every year, and they can be shared forever. Whenever possible, spend money on *experiences* such as taking your family to Disney World, rather than *things* like a new television.

2. Help others instead of yourself Human beings are intensely social animals. Anything we can do with money to create deeper connections with other human beings tends to tighten our social connections and reinforce positive feelings about ourselves and others. Imagine ways you can spend some part of your money to help others – even in a very small way – and integrate that into your regular spending habits.

3. Buy many small pleasures instead of few big ones Because we adapt so readily to change, the most effective use of your money is to bring frequent change, not just "big bang" changes that you will quickly grow acclimated to. Break up large purchases, when possible, into smaller ones over time so that you can savor the entire experience. When it comes to happiness, frequency is more important than intensity. Embrace the idea that lots of small, pleasurable purchases are actually *more* effective than a single giant one.

4. Buy less insurance Humans adapt readily to both positive and *negative* change. Extended warranties and insurance prey on your impulse for loss aversion, but because we are so adaptable, people experience far less regret than they anticipate when their purchases don't work out. Furthermore, having the easy "out" of insurance or a generous return policy can paradoxically lead to even *more* angst and unhappiness because people deprived themselves of the emotional benefit of full commitment. Thus, avoid buying insurance, and don't seek out generous return policies.

5. Pay now and consume later Immediate gratification can lead you to make purchases you can't afford, or may not even truly want. Impulse buying also deprives you of the distance necessary to make reasoned decisions.

It eliminates any sense of anticipation, which is a strong source of happiness. For maximum happiness, savor (maybe even prolong!) the uncertainty of deciding whether to buy, what to buy, and the time waiting for the object of your desire to arrive.

6. Think about what you're not thinking about We tend to gloss over details when considering future purchases, but research shows that our happiness (or unhappiness) largely lies in exactly those tiny details we aren't thinking about. Before making a major purchase, consider the mechanics and logistics of owning this thing, and where your actual time will be spent once you own it. Try to imagine a typical day in your life, in some detail, hour by hour: how will it be affected by this purchase?

7. Beware of comparison shopping Comparison shopping focuses us on attributes of products that arbitrarily distinguish one product from another, but have nothing to do with how much we'll *enjoy* the purchase. They emphasize characteristics we care about while shopping, but not necessarily what we'll care about when actually using or consuming what we just bought. In other words, getting a great deal on cheap chocolate for $2 may not matter if it's not pleasurable to eat. Don't get tricked into comparing for the sake of comparison; try to weight only those criteria that actually matter to your enjoyment or the experience.

8. Follow the herd instead of your head
Don't overestimate your ability to independently predict how much you'll enjoy something. We are, scientifically speaking, very bad at this. But if something reliably makes others happy, it's likely to make you happy, too. Weight other people's opinions and user

reviews heavily in your purchasing decisions.

Happiness is a lot harder to come by than money. So when you *do* spend money, keep these eight lessons in mind to maximize whatever happiness it can buy for you. And remember: **it's science!**

Table 1 none

Trust Me, I’m Lying

We reflexively instruct our children to always tell the truth. It’s even ***encoded into Boy Scout Law***. It’s what adults do, isn’t it? But do we? Isn’t telling the truth too much and too often a bad life strategy – perhaps even dangerous? Is telling children to always tell the truth *even itself the whole truth?*

One of the most thought provoking articles on the topic, and one I keep returning to, year after year, is ***I Think You're Fat***. It's about the **Radical Honesty movement**, which proposes that adults follow their own advice and *always tell the truth*. No matter what.

> The [Radical Honesty] movement was founded by a sixty-six-year-old Virginia-based psychotherapist named Brad Blanton. He says everybody would be happier if we just stopped lying. Tell the truth, all the time. This would be radical enough – a world without fibs – but Blanton goes further. He says we should toss out the filters between our brains and our mouths. If you think it, say it. Confess to your boss your secret plans to start your own company. If you're having fantasies about your wife's sister, Blanton says to tell your wife and tell her sister. It's the only path to authentic relationships.

It’s the only way to smash through modernity’s soul-deadening alienation. Oversharing? No such thing.

Yes. I know. One of the most idiotic ideas ever, right up there with Vanilla Coke and giving Phil Spector a gun permit. Deceit makes our world go round. Without lies, marriages would crumble, workers would be fired, egos would be shattered, governments would collapse.

And yet … maybe there’s something to it. Especially for me. I have a lying problem. Mine aren’t big lies. They aren’t lies like “I cannot recall that crucial meeting from two months ago, Senator.” Mine are little lies. White lies. Half-truths. The kind we all tell. But I tell dozens of them every day. “Yes, let’s definitely get together soon.” “I’d

> love to, but I have a touch of the stomach flu." "No, we can't buy a toy today – the toy store is closed." It's bad. Maybe a couple of weeks of truth-immersion therapy would do me good.

The author, A.J. Jacobs, is a great writer who has made something of a cottage industry of treating himself ***like a guinea pig***, such as attempting to ***become the smartest man in the world***, spend ***a year living exactly like the Bible tells us to***, and to ***become the fittest person on Earth***. Based on the strength of this article, I bought two of his books; experiments like Radical Honesty are right up his alley.

Radical honesty itself isn't exactly a new concept. It's been parodied in any number of screwball Hollywood comedies such as ***Liar, Liar*** (1997) and ***The Invention of Lying*** (2009). But there's a big difference between milking this concept for laughs and exploring it as an actual lifestyle among real human beings. Among the ideas raised in the article, which ***you should go read now***, are:

- Telling someone that something they created is terrible: is that cruelty, because they have no talent, or is it compassion, so they can know they need to improve it?

- Does a thought in your head that you never express to anyone represent your truth? Should you share it? This is particularly tricky for men, who ***think about sex twice as much as women***.

- How much mental energy do you expend listening to a conversation trying to determine how much of what the other person is saying is untrue? Wouldn't it be less fatiguing if everything they said was, by definition, the truth? And when you're talking, always telling the truth means you never have to decide just how much truth to tell, how to hedge, massage, and spin the truth to make it palatable.

- In a hypothetical future when every action we take is public and broadcast to the world, is that exposing the real truth of our lives? Should we become more honest today to ready ourselves for this inevitable future?

- Always telling the truth can be thrilling, a form of risk taking, as you intentionally violate taboos around politeness that exist solely for the sake of avoiding conflict.

- Total honesty can lead to new breakthroughs in communication, where politeness prevented you from ever reaching the root, underlying causes of discontent or unhappiness.

- Honesty is more efficient. Rather than spending a lot of time sending messages back and forth artfully dancing *around* the truth, go directly there.

- If people see you are willing to be honest with them, they tend to return the favor, leading to a more useful relationship.

What we often don't acknowledge is that **the truth is kind of scary**. That's why we have a hard time being honest with *ourselves*, much less those around us. Reading through all these ambiguous situations that A.J. put himself through, you start to wonder if you understand what truth is, or what it means to decide that something is "true". After summarizing the article in bullet form, I'm surprised there are so many points in favor of honesty, maybe even radical honesty.

But uncompromisingly committing to the whole truth, and nothing but the truth, has a darker side.

> My wife tells me a story about switching operating systems on her computer. In the middle, I have to go help our son with something, then forget to come back.
>
> "Do you want to hear the end of the story or not?" she asks.
>
> "Well...is there a payoff?"
>
> "F**k you."
>
> It would have been a lot easier to have kept my mouth closed and listened to her. It reminds me of an issue I raised with Blanton: Why make waves? "Ninety percent of the time I

> love my wife," I told him. "And 10 percent of the time I hate her. Why should I hurt her feelings that 10 percent of the time? Why not just wait until that phase passes and I return to the true feeling, which is that I love her?"
>
> Blanton's response: "Because you're a manipulative, lying son of a bitch."

Rather than embrace the truth, as Radical Honesty would have us do, Adrian Tan advises us to ***be wary of the truth***.

> Most of you will end up in activities which involve communication. To those of you I have a second message: be wary of the truth. I'm not asking you to speak it, or write it, for there are times when it is dangerous or impossible to do those things. The truth has a great capacity to offend and in-

> jure, and you will find that the closer you are to someone, the more care you must take to disguise or even conceal the truth. Often, there is great virtue in being evasive, or equivocating. There is also great skill. Any child can blurt out the truth, without thought to the consequences. It takes great maturity to appreciate the value of silence.

I think he's right. But Radical Honesty isn't altogether wrong, either. Let me be clear: Radical Honesty, as a lifestyle, is ridiculous and insane. **Advocating telling the truth 100%of the time, no matter what, is harmful extremism.** But it's also wonderfully seductive as a concept, because it illustrates how needlessly afraid most of us are of truth – even truths that could potentially help us. Radical Honesty teaches us to be more brave. That is, when it's not *destroying our lives and the lives of every-*

one around us.

Ask yourself: what is the *purpose* of this truth? What effect will sharing this truth have on the other person, on yourself, on the world? What change will come about, positive or negative, from choosing to voice a particular truth at a particular time?

I believe that the true lesson of Radical Honesty is that **truth, real truth, is honesty with a *purpose***. Ideally a noble purpose, but any purpose at all other than "because I could" will suffice. By all means, be brave; embrace the truth. But if your honesty has no purpose, if you can't imagine any positive outcome from this honesty, I suggest you're better off keeping it to yourself.

Or even lying.

companies, whether you’re looking for opportunitie

Table 1 none

Geekatoo, the Geek Bat-Signal

To understand this story, you need to understand that **grandchildren are like crack cocaine to grandparents**. I'm convinced that if our parents could somehow snort our children up their noses to get a bigger fix, they would. And when your parents live out of state, like ours do, access to the Internet isn't just important. No. It is *life threatening*.

Like ***Gator in Jungle Fever***, grandparents just gotta get their fix of the grandkids every month. And if they don't, if their Internet is broken for any reason, you're going to get an earful via telegraph and facsimile and registered letter until you fix it.

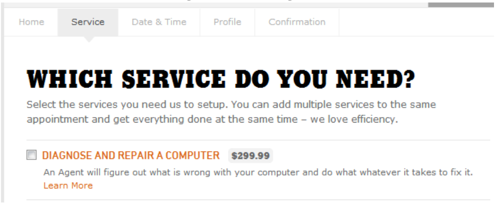
Either way, *they're gonna get high*. On your kids.

My mom is no exception. So when her computer suddenly stopped working, and she couldn't get updates on her three grandkids, I got frantic calls. Which is odd, because everything had been working fine for a few years now. Once Henry was ***born*** in 2009, I set her up with a ***netbook*** that had Skype and Firefox set to auto update and she'd been able to video chat with us regularly, no problem at all, since then. So what happened?

My first thought was to hell with it, I'll just buy her a new iPad online via the Apple Store. I'm a ***big fan*** of the retina display, and surely the touchy-feely iPad would be more resistant to whatever problem she was having than a netbook, what with its archaic "operating system" and "updates" and "keyboard" and "mouse".

With some urging from my wife (I married well), cooler heads prevailed. What if her problem had nothing to do with the computer, but her Internet connection in some way? Then I'd just be trading one set of problems for another with the iPad. I have no idea how things are set up over there, thousands of miles away. I needed help. **Help from a fellow geek who is willing to drive out and assist my poor mom.**

My mom doesn't live near where I grew up any more, so I have no friend network there. All I could think of was ***Geek Squad***. I've seen their trucks around the neighborhood, and they've been around a while, so I checked out their website. Maybe they'd work?



WHICH SERVICE DO YOU NEED?

Select the services you need us to setup. You can add multiple services to the same appointment and get everything done at the same time – we love efficiency.

☐ DIAGNOSE AND REPAIR A COMPUTER $299.99

An Agent will figure out what is wrong with your computer and do what whatever it takes to fix it.

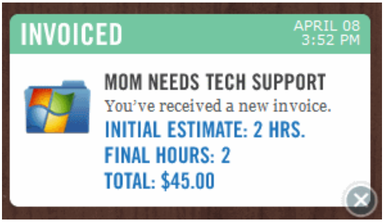
Learn More

When I can buy my mom a new iPad for $399, **the idea of paying $299 just to have someone come out and fix her old stuff starts to feel like a really bad idea**. But I suppose it's a preview of our disposable computer future, because it's cheaper to buy a new one than it is to fix the old one. This is the stuff that my friend and ***iFixit*** founder Kyle Wiens' nightmares are made of. I'm sorry, Kyle. But it's coming.

I posted my discontent on Twitter, and received an interesting recommendation for a site I'd never heard of – ***Geekatoo***.

I was intrigued, first because the site didn't appear to suck which is more than I can say for about half the links I click on, and second because it appealed to my geek instincts. I could post a plea for help for my mom, and a fellow geek, one of my kind who happened to be local, would be willing to head out and assist. **I could send out the geek bat-signal!** But I was still skeptical. My mom lives in Charlotte, North Carolina which, while not exactly the sticks, isn't necessarily a big tech hub city, either. I figured I had nothing to lose at this point, so I posted the request titled "Mom Needs Tech Support" with the info I had.

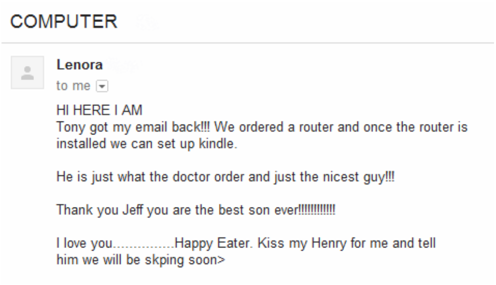
Much to my surprise, I got two great bids within 24 hours, geeks with good credentials, and I picked the first one. The estimate was two hours for **$45**, and he was on-site helping my mom within 2 days from the time I posted.


INVOICED
APRIL 08
3:52 PM
MOM NEEDS TECH SUPPORT
You've received a new invoice.
INITIAL ESTIMATE: 2 HRS.
FINAL HOURS: 2
TOTAL: $45.00

Geekatoo-case

It turns out that my wife's intuition was correct: the cable internet installer had inexplicably decided to connect my mother's computer to a neighbor's wireless, instead of setting up a WiFi access point for her. So when that neighbor moved away, calamity ensued.

And the results? Well, I think they speak for themselves.



COMPUTER

Lenora
to me

HI HERE I AM
Tony got my email back!!! We ordered a router and once the router is installed we can set up kindle.

He is just what the doctor order and just the nicest guy!!!

Thank you Jeff you are the best son ever!!!!!!!!!!!

I love you...............Happy Eater. Kiss my Henry for me and tell him we will be skping soon>

Thank you Jeff you are
the best son ever!!!!!!!!!

My mom, as usual, exaggerates about her only son. I am far, far from the best son ever. But any website that can make me look like a hero to my mom, *and* keep my fellow ***Super User geeks*** gainfully employed doing super-hero work on my behalf gets a huge thumbs up from me.

Needless to say, strongly recommended. If you need reliable local tech support that won't break the bank, and you want to **support both your family and your local geek community at the same time**, check out ***Geekatoo***.

w off all of your hard work from Stack Overflow, Gith

Table 1 none

Will Apps Kill Websites?

I've been an eBay user since 1999, and I still frequent eBay as both buyer and seller. In that time, eBay has transformed from a place where geeks ***sell broken laser pointers to each other***, into a global marketplace where businesses sell anything and everything to customers. If you're looking for strange or obscure items, things almost nobody sells new any more, or grey market items for cheap, eBay is still not a bad place to look.

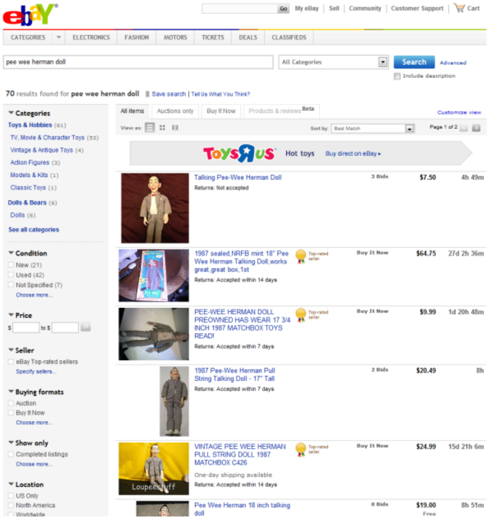
At least for me, eBay still basically works, after all these years. But one thing hasn't changed: **the eBay website has always been difficult to use and navigate**. They've updated the website recently to remove some of the more egregious cruft, but it’s still *way* too complicated. I guess I had kind of accepted old, complex websites as the status quo, because I didn’t realize how bad it had gotten until I compared the experience on the eBay website with the experience of the eBay apps for mobile and tablet.

eBay Website

eBay Mobile App

eBay Tablet App

Unless you’re some kind of super advanced eBay user, you should probably avoid the website. The tablet and mobile eBay apps are just plain simpler, easier, and faster to use than the eBay website itself. I know this intuitively from using eBay on my devices and computers, but there’s also ***usability studies*** with



pee wee herman doll

70 results found for **pee wee herman doll**

Categories
Toys & Hobbies (61)
TV, Movie & Character Toys (53)
Vintage & Antique Toys (4)
Action Figures (3)
Models & Kits (1)
Classic Toys (1)
Dolls & Bears (6)
Dolls (6)
See all categories

Condition
New (21)
Used (42)
Not Specified (7)
Choose more...

Price
$ to $

Seller
eBay Top-rated sellers
Specify sellers...

Buying formats
Auction
Buy It Now
Choose more...

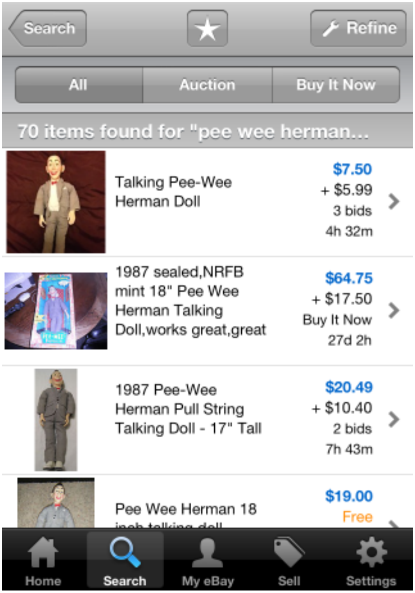
Show only
Completed listings
Choose more...

Location
US Only
North America
Worldwide

Talking Pee-Wee Herman Doll Returns: Not accepted		3 Bids	$7.50	4h 49m
1987 sealed,NRFB mint 18" Pee Wee Herman Talking Doll,works great,great box,1st Returns: Accepted within 14 days	Top-rated seller	Buy It Now	$64.75	27d 2h 36m
PEE-WEE HERMAN DOLL PREOWNED HAS WEAR 17 3/4 INCH 1987 MATCHBOX TOYS READ! Returns: Accepted within 7 days	Top-rated seller	Buy It Now	$9.99	1d 20h 48m
1987 Pee-Wee Herman Pull String Talking Doll - 17" Tall Returns: Accepted within 7 days		2 Bids	$20.49	8h
VINTAGE PEE WEE HERMAN PULL STRING DOLL 1987 MATCHBOX C426 One-day shipping available Returns: Accepted within 14 days	Top-rated seller	Buy It Now	$24.99	15d 21h 6m
Pee Wee Herman 18 inch talking doll		0 Bids	$19.00 Free	8h 51m

Ebay-web

data to prove it, too. To be fair, eBay is struggling under the massive accumulated design debt of a website originally conceived in the late 90s, whereas their mobile and tablet app experiences are recent inventions. **It's not so**



70 items found for "pee wee herman...

Talking Pee-Wee Herman Doll
$7.50
+ $5.99
3 bids
4h 32m

1987 sealed,NRFB mint 18" Pee Wee Herman Talking Doll,works great,great
$64.75
+ $17.50
Buy It Now
27d 2h

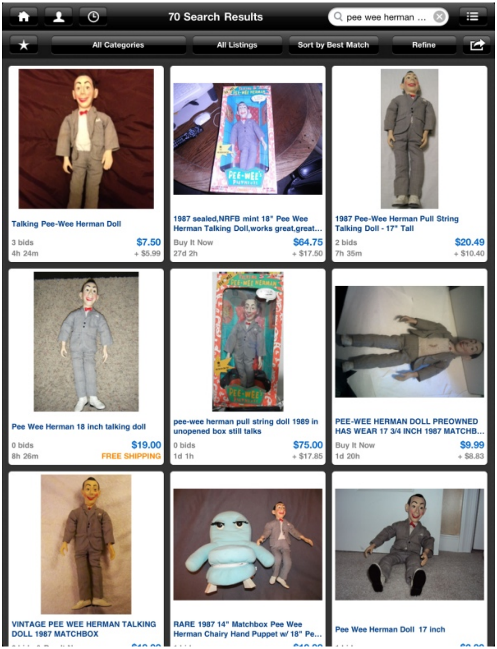
1987 Pee-Wee Herman Pull String Talking Doll - 17" Tall
$20.49
+ $10.40
2 bids
7h 43m

Pee Wee Herman 18 inch talking doll
$19.00
Free

Ebay-iphone-app

much that the eBay apps are great, but that the eBay website is so very, *very* bad.

The implied lesson here is to **embrace con-**



Talking Pee-Wee Herman Doll
3 bids $7.50
4h 24m + $5.99

1987 sealed,NRFB mint 18" Pee Wee Herman Talking Doll,works great,great...
Buy It Now $64.75
27d 2h + $17.50

1987 Pee-Wee Herman Pull String Talking Doll - 17" Tall
2 bids $20.49
7h 35m + $10.40

Pee Wee Herman 18 inch talking doll
0 bids $19.00
8h 26m FREE SHIPPING

pee-wee herman pull string doll 1989 in unopened box still talks
0 bids $75.00
1d 1h + $17.85

PEE-WEE HERMAN DOLL PREOWNED HAS WEAR 17 3/4 INCH 1987 MATCHB...
Buy It Now $9.99
1d 20h + $8.83

VINTAGE PEE WEE HERMAN TALKING DOLL 1987 MATCHBOX

RARE 1987 14" Matchbox Pee Wee Herman Chairy Hand Puppet w/ 18" Pe...

Pee Wee Herman Doll 17 inch

Ebay-ipad-app

straints. Having a limited, fixed palette of UI controls and screen space is a *strength*. A strength we used to have in early Mac and Windows apps, but seem to have lost somewhere along the way as applications got more powerful and complicated. And it’s endemic on the web as well, where the eBay website has been slowly accreting more and more functionality since 1999. The nearly unlimited freedom that you get in a modern web browser to build whatever UI you can dream up, and assume as large or as small a page as you like, often ends up being harmful to users. It certainly is in the case of eBay.

If you’re starting from scratch, you should always ***design the UI first***, but now that we have such great mobile and tablet device options, consider designing your site for the devices that have the strictest constraints first, too. This is the thinking that led to ***mobile first design strategy***. It helps you stay focused on a simple and uncluttered UI that you can scale up to bigger and beefier devices. Maybe eBay is just going in the wrong direction here; **design simple things that scale up; not complicated things you need to scale down.**

Above all else, simplify! But why stop there? If building the mobile and tablet apps first for a web property produces a better user experience – why do we need the website, again? Do great tablet and phone applications make websites obsolete?

Why are apps better than websites?

1. **They can be faster.**
   No browser overhead of CSS and HTML and JavaScript hacks, just pure native UI elements retrieving precisely the data they need to display what the user requests.

2. **They use simple, native UI controls.**
   Rather than imagineering whatever UI designers and programmers can dream up, why not pick from a well understood palette of built-in UI controls on that tablet or phone, all built for optimal utility and affordance on that particular device?

3. **They make better use of screen space.**
   Because designers have to fit just the *important* things on 4 inch diagonal mobile screens, or 10 inch diagonal tablet screens, they’re less likely to fill the display up with a bunch of irrelevant noise or design flourishes (or, uh, advertisements). Just the important stuff, thanks!

4. **They work better on the go and even offline.**
   In a mobile world, you can’t assume that the user has a super fast, totally reliable Internet connection. So you learn to design apps that download just the data they need at the time they need to display it, and have sane strategies for loading partial content and images as they arrive. That’s assuming they arrive at all. You probably also build in some sort of offline mode, too, when you’re on the go and you don’t have connectivity.

Why are websites better than apps?

1. **They work on any device with a browser.**

Websites are as close to universal as we may ever get in the world of software. Provided you have a HTML5 compliant browser, you can run an entire universe of “apps” on your device from day zero, just by visiting a link, exactly the same way everyone has on the Internet since 1995. You don’t have to hope and pray a development community emerges and is willing to build whatever app your users need.

2. **They don’t have to be installed.** Applications, unlike websites, can’t be visited. They aren’t indexed by Google. Nor do applications magically appear on your device; they must be explicitly installed. Even if installation is a one-click

affair, your users will have to discover the app before they can even begin to install it. And once installed, they'll have to **manage *all those applications like so many Pokemon***.

3. **They don't have to be updated.**
   Websites are always on the ***infinite version***. But once you have an application installed on your device, how do you update it to add features or fix bugs? How do users even know if your app is out of date or needs updating? And why should they need to care in the first place?

4. **They offer a common experience.**
   If your app and the website behave rad-

> ically differently, you’re forcing users to learn two different interfaces. How many different devices and apps do you plan to build, and how consistent will they be? You now have a community divided among many different experiences, fragmenting your user base. But with a website that has a decent mobile experience baked in, you can deliver a consistent, and hopefully *consistently great*, experience across *all* devices to all your users.

I don’t think there’s a clear winner, only pros and cons. But **apps will always need websites**, if for nothing else other than a source of data, as a mothership to phone home to, and a place to host the application downloads for various devices.

And if you’re obliged to build a website, why not build it out so it offers a *reasonable* experience on a mobile or tablet web browser, too? I have nothing against a premium experience optimized to a particular device, but shouldn't all your users have a premium experience? eBay's problem here isn't mobile or tablets per se, but that they've let their core web experience atrophy so badly. I understand that there's a lot of inertia around legacy eBay tools and long time users, so it's easy for me to propose radical changes to the website here on the outside. Maybe the only way eBay can redesign *at all* is on new platforms.

Will mobile and tablet apps kill websites? A few, certainly. **But only the websites stupid enough to let them.**

companies, whether you're looking for opportunitie

Table 1 none

Make Your Email Hacker Proof

It’s only a matter of time until your email gets hacked. Don’t believe me? Just read ***this harrowing cautionary tale***.

> When [my wife] came back to her desk, half an hour later, she couldn’t log into Gmail at all. By that time, I was up and looking at e-mail, and we both quickly saw what the real problem was. In my inbox I found a message purporting to be from her, followed by a quickly proliferating stream of concerned responses from friends and ac-

quaintances, all about the fact that she had been "mugged in Madrid." The account had seemed sluggish earlier that morning because my wife had tried to use it at just the moment a hacker was taking it over and changing its settings—including the password, so that she couldn't log in again.

...

The greatest practical fear for my wife and me was that, even if she eventually managed to retrieve her records, so much of our personal and financial data would be in someone else's presumably hostile hands that we would spend our remaining years looking over our shoulders, wondering how and when something would be put to damaging use. At some point over the past six years, **our [email] corre-**

> **spondence would certainly have included every number or code that was important to us – credit card numbers, bank-account information, medical info, and any other sensitive data you can imagine.**

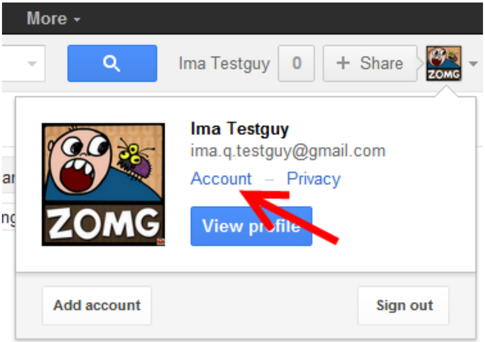
Now get everyone you know to read it, too. Please. It's for their own good.

Your email is ***the skeleton key to your online identity***. When you lose control of your email to a hacker – not if, but *when* you lose control of your email to a hacker – the situation is dire. Email is a one stop shop for online identity theft. You should start thinking of security for your email as roughly equivalent to the sort of security you'd want on your bank account. It's exceedingly close to that in practice.

The good news, at least if you use GMail, is that **you can make your email virtually hacker-proof today, provided you own a**

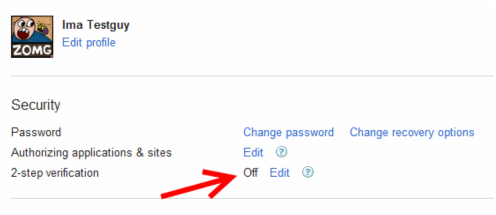
cell phone. The fancy geek technical term for this is ***two factor authentication***, but that doesn't matter right now. What matters is that until you turn this on, your email is vulnerable. So let's get started. Not tomorrow. Not next week. **Right. Freaking. Now.**

Go to your Google Account Settings



Google-account-settings

Make sure you're logged in. Expand the little drop-down user info panel at the top right of most Google pages. From here, click “Account” to view your account settings.


Ima Testguy
Edit profile

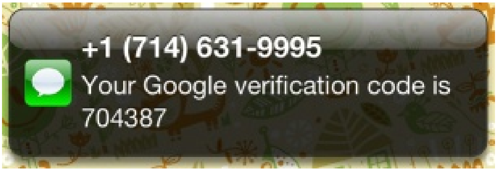
Security

Password	Change password Change recovery options
Authorizing applications & sites	Edit
2-step verification	Off Edit

Google-enable-two-factor-auth

On the account settings page, click “edit” next to 2-step verification and turn it on.

Have Your Cell Phone Ready GMail will walk you through the next few steps. You just need a telephone that can receive SMS text messages. Enter the numeric code sent through the text message to proceed.



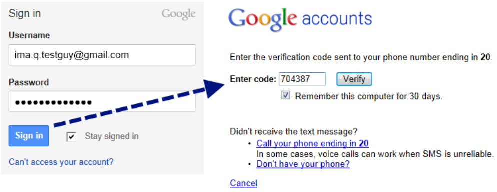
+1 (714) 631-9995

Your Google verification code is 704387

Google-text-email-verification

Now Log In With Your Password and a PIN

Now your password alone is no longer enough to access your email.



Sign in

Username: ima.q.testguy@gmail.com

Password: ••••••••••••••

[✓] Stay signed in

Can't access your account?

Google accounts

Enter the verification code sent to your phone number ending in **20**.

Enter code: 704387

[✓] Remember this computer for 30 days.

Didn't receive the text message?

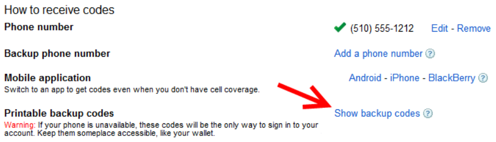
- Call your phone ending in **20**
  In some cases, voice calls can work when SMS is unreliable.
- Don't have your phone?

Cancel

Google-two-factor-login

Once this is enabled, **accessing your email always requires the password, *and* a code delivered via your cell phone**. (You can check the “remember me for 30 days on this device” checkbox so you don’t have to do this every time.) With this in place, even if they discover your super sekrit email password, would-be hackers can’t do anything useful with it! To access your email, they’d need to somehow gain control of your cell phone, too. I can’t see that happening unless you’re in some sort of hostage situation, and at that point I think email security is the least of your problems.

What If I Lose My Cell Phone? Your cell phone isn’t the only way to get the secondary PIN you need to access your email. On the account page there are multiple ways to generate verification codes, including adding a secondary backup phone number, and downloading ***mobile applications that can generate verification codes*** without a text message (but that requires a smart phone, naturally).


How to receive codes

Phone number ✓ (510) 555-1212 Edit - Remove

Backup phone number Add a phone number

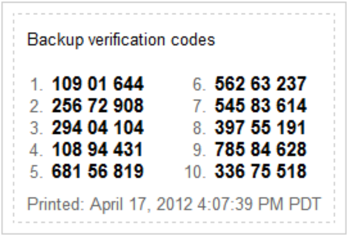
Mobile application Android - iPhone - BlackBerry
Switch to an app to get codes even when you don't have cell coverage.

Printable backup codes Show backup codes
Warning: If your phone is unavailable, these codes will be the only way to sign in to your account. Keep them someplace accessible, like your wallet.

Google-backup-email-codes

This also includes the never-fails-always-works option: **printing out the single-use backup verification codes on a piece of paper**. Go do this now. *Right now!* And keep those backup codes with you at all times. Put them in your wallet, purse, man-purse, or whatever it is that travels with you most often when you get out of bed.

What About Apps That Access Email? Applications or websites that access your email, and thus necessarily store your email address and password, are also affected. They have no idea that they now need to enter a PIN, too, so they'll all be broken. You'll need to **generate**



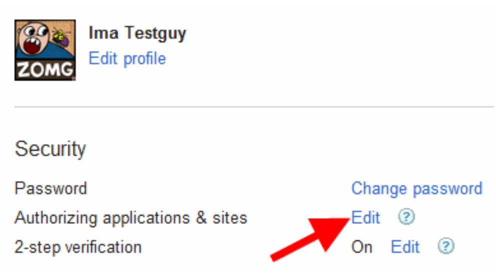
Backup verification codes

1.	109 01 644	6.	562 63 237
2.	256 72 908	7.	545 83 614
3.	294 04 104	8.	397 55 191
4.	108 94 431	9.	785 84 628
5.	681 56 819	10.	336 75 518

Printed: April 17, 2012 4:07:39 PM PDT

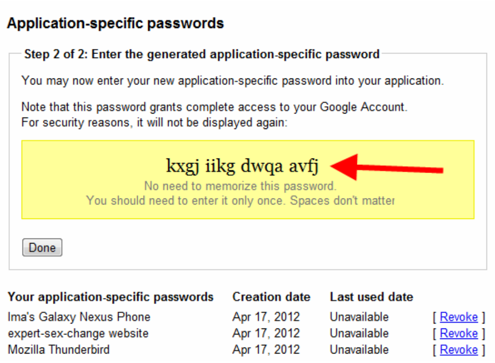
Backup-verification-codes

app-specific passwords for your email. To do that, visit the accounts page.



Google-enabling-apps

Click on authorizing applications & sites, then enter a name for the application and click the Generate Password button.



Application-specific passwords

Step 2 of 2: Enter the generated application-specific password

You may now enter your new application-specific password into your application.

Note that this password grants complete access to your Google Account.
For security reasons, it will not be displayed again:

kxgj iikg dwqa avfj

No need to memorize this password.
You should need to enter it only once. Spaces don't matter

Your application-specific passwords	Creation date	Last used date	
Ima's Galaxy Nexus Phone	Apr 17, 2012	Unavailable	[Revoke]
expert-sex-change website	Apr 17, 2012	Unavailable	[Revoke]
Mozilla Thunderbird	Apr 17, 2012	Unavailable	[Revoke]

Google-generated-app-password

Let me be clear about this, because it can be confusing: **enter that specially generated password in the application, *not* your master email password**.

This effectively creates a list of passwords specific to each application. So you can see the date each one was last used, and revoke each app's permission to touch your email individually as necessary without ever revealing your primary email password to *any* application, ever. See, I told you, there is a method to the apparent madness.

But I Don't Use Gmail Either nag your email provider to provide two-factor authentication, or switch over. Email security is critically important these days, and switching is easy(ish). GMail has had fully secure connections for ***quite a while now***, and once you add two-factor authentication to the mix, that's about as much online email safety as you can reasonably hope to achieve short of going back to snail mail.

Hey, This Sounds Like a Pain! I know what you're thinking. Yes, this is a pain in the ass. I'll fully acknowledge that. But you know what's an even *bigger* pain in the ass? Having your entire online identity stolen and trashed by a hacker who happens to obtain your email password one day. Remember that article I exhorted you to read at the beginning? Oh, you didn't read it? ***Go freaking read it now!***

Permit me to ***channel Jamie Zawinski*** one last time: "OMG, entering these email codes on every device I access email would be a lot of work! That sounds like a hassle!" **Shut up. I know things. You will listen to me. Do it anyway.**

I've been living with this scheme for a few months now, and I've convinced my wife to as well. I won't lie to you; it hasn't all been wine and roses for us either. But it is inconvenient in the same way that bank vaults and door locks are. The upside is that once you enable this, your email becomes **extremely secure**, to the point that you can (and I regularly do) email yourself highly sensitive data like passwords and logins to other sites you visit so you can easily retrieve them later.

If you choose not to do this, well, at least you've educated yourself about the risks. And I hope you're extremely careful with your email password and change it regularly to something complex. You're making life all too easy for the hackers who make a fabulous living from stealing and permanently defacing online identities *just like yours*.

Table 1 none

Learn to Read the Source, Luke

In the calculus of communication, writing coherent paragraphs that your fellow human beings can comprehend and understand is ***far more difficult*** than tapping out a few lines of software code that the interpreter or compiler won't barf on.

That's why, when it comes to code, ***all the documentation probably sucks***. And because writing for people is way harder than writing for machines, the documentation will *continue* to suck for the forseeable future. There's very little you can do about it.

Except for one thing.

Read-the-source-luke

You can **learn to read the source, Luke**.

The ***transformative power of "source always included" in JavaScript*** is a major reason why I coined – and continue to believe in – ***Atwood's Law***. Even if "view source" isn't built in (but it totally should be), you should demand access to the underlying source code for your stack. **No matter what the documentation says, the source code is the ultimate**

truth, the best and most definitive and up-to-date documentation you're likely to find. This will be true *forever*, so the sooner you come to terms with this, the better off you'll be as a software developer.

I had a whole entry I was going to write about this, and then I discovered ***Brandon Bloom's*** brilliant ***post*** on the topic at Hacker News. Read closely, because he explains the virtue of reading source, and in what context you *need* to read the source, far better than I could:

> I started working with Microsoft platforms professionally at age 15 or so. I worked for Microsoft as a software developer doing integration work on Visual Studio. More than ten years after I first wrote a line of Visual Basic, I wish I could never link against a closed library ever again.

Using software is different than building software. When you're using most software for its primary function, it's a well worn path. Others have encountered the problems and enough people have spoken up to prompt the core contributors to correct the issue. But when you're building software, you're doing something new. And there are so many ways to do it, you'll encounter unused bits, rusty corners, and unfinished experimental code paths. You'll encounter edge cases that have been known to be broken, but were worked around.

Sometimes, the documentation isn't complete. Sometimes, it's wrong. The source code never lies. For an experienced developer, reading the source can often be faster... especially if you're already familiar with the package's architecture. I'm in a medium-sized co-working space with several startups. A lot of the other CTOs and engineers come to our team for guidance and advice on occasion. **When people report a problem with their stack, the first question I ask them is: "Well, did you read the source code?"**

I encourage developers to git clone anything and everything they depend on. Initially, they are all afraid. "That project is too big, I'll never find it!" or "I'm not smart enough to understand it" or "That code is so ugly! I can't stand to look at it". But you don't have to search the whole thing, you just need to follow the trail. And if you can't understand the platform below you, how can you understand your own software? And most of the time, what inexperienced developers consider beautiful is superficial, and what they consider ugly, is battle-hardened production-ready code from master hackers. Now, a year or two later, I've had a couple of developers come up to me and thank me for forcing them to sink or swim in other people's code bases. They are better at their craft and they wonder how they ever got anything done without the source code in the past.

When you run a business, if your software has a bug, your customers don't care if it is your fault or Linus' or some random Rails developer's. They care that your software is bugged. Everyone's software becomes my software because all of their bugs are my bugs. When something goes wrong, you need to seek out what is broken, and you need to fix it. You fix it at the right spot in the stack to minimize risks, maintenance costs, and turnaround time. Sometimes, a quick workaround is best. Other times, you'll need to recompile your compiler. Often, you can ask someone else to fix it upstream, but just as often, you'll need to fix it yourself.

- Closed-software shops have two choices: beg for generosity, or work around it.

- Open source shops with weaker developers tend to act the same as closed-software shops.

> - Older shops tend to slowly build the muscles required to maintain their own forks and patches and whatnot.
>
> True hackers have come to terms with a simple fact: If it runs on my machine, it’s my software. I’m responsible for it. I must understand it. Building from source is the rule and not an exception. I must control my environment and I must control my dependencies.

Nobody reads other people’s code for fun. Hell, ***I don’t even like reading my own code***. The idea that you’d settle down in a deep leather chair with your smoking jacket and a snifter of brandy for a fine evening of reading through someone else’s code is absurd.

But we need access to the source code. We *must* read other people’s code ***because we have to understand it to get things done***. So don’t be afraid to **read the source, Luke** – and follow it wherever it takes you, no matter how scary looking that code is.

[advertisement] ***Stack Overflow Careers*** matches the best developers (you!) with the best employers. You can search our ***job listings*** or ***create a profile*** and even let employers find you.