GOOMICS

Google's corporate culture revealed through internal comics

Volume I

2010 – 2015

Glitches in Wonderland

Drawings and text by Manu Cornet

For my parents, who have long been looking for a way to understand what it is that I do everyday. Thanks to this book, they will still have absolutely no clue.

Many thanks to Chris Achille, Shane Almeida, Talha Amin, Doug Arnett, Shane Brennan, Christopher Carpenter, Bruno Cauet, Géraldine Chachoua, Olivia Chang, Poonam Chawla, Willa Chen, Tal Cohen, Brad Conte, Damien Desfontaines, Victoria Dukes, Katlyn Edwards, Alejandra Estanislao, Corey Fry, Andre Hamrah, Scott Holzer, Jeremy Hurwitz, Rob Judd, Akshay Lal, Camden Lindsay, Jerome Li, Sean Lip, Ohad Lutzky, Andrew Lytvynov, Julia Ma, Grace Mollison, Sven Mueller, David Murphy, Cliff Redeker, Phillip Remaker, Dave Rensin, Jess Renteria, Phil Rollet, Leif Schelin, Piotr Sieklucki, Erin Soderberg, Ilya Sherman, Alexandru Totolici, Andy Walker, Kathy Walrath, Hans Wennborg, Sharon Wong and Julie Wu for reading early versions of this book and providing extremely valuable feedback.

This book was made exclusively with open source software. It was initially released in September 2018. The version you are reading was produced on June 29, 2021. If it is now a much later date, you can blame the distributor (for being slow), the author (for not updating) or yourself (for procrastinating). Later revisions may contain small fixes and enhancements, but certainly do not warrant purchasing the book again (my sales consultant strongly advised against writing this).

ISBN 978-0-98-852384-5

Foreword

Forewords are boring. I never read them myself. Please go ahead and turn the page now.

What on Earth is this?

I joined Google as a software engineer in 2007. A couple of years later, I started drawing clumsy little cartoons about life at Google, which evolved over the years into a full-on series with hundreds of "episodes".

Being written by a Google employee for other Google employees, most of the drawings require a little bit of explanation. But if you need to spend time explaining the context of a cartoon, well, the cartoon isn't very funny anymore. This book is therefore designed as a peek into Google's corporate culture rather than a promise of endless laughter. Sorry.

For confidentiality reasons, the most recent cartoons aren't yet ready to be shared publicly. That is why this book contains the 200 episodes published between 2010 and 2015.

Who is this book for?

In general, some amount of curiosity about Google's internal culture is all that's required.

- **Normal people**: This book is for you! No prior knowledge of technical jargon is required, everything is explained as we go. If anything is still unclear, the glossary (page 136) and the index (page 139) are here to help.
- **Geeks and IT workers**: You'll feel right at home in this world of nerdiness, and you can probably skip over most of the explanations.
- **Googlers**: I don't know how I tricked you into buying this book! You can already read 100% of these comics for free. If you're a new hire, though, you might learn a few things about the company's history.

Time to turn the page now, yes?

Google

This book contains many things about Google, its leaders, and its employees. Kind, snarky, cynical, controversial — regardless of the tone, I hope you can tell that I'm very fond of this company, its founders and executives, and my coworkers. If that weren't true, I wouldn't still be delighted to go to work every morning since 2007. Just look at the diversity that the list of names on page 2 hints at — one of the reasons why I love this place.

To Google's credit, it's also fair to say that I have not once been approached by anybody from human resources, legal, or leadership asking me to tone it down. The fact that you're holding this book says a lot about Google's open culture.

Now for the usual disclaimer: everything in this book expresses **only my own views** and not, in any way, my employer's.

How to read this book?

- Reading **in order is easiest**; **cherry-picking works too**, with an occasional glance at the **glossary**. Words that have an entry in the glossary are printed in color the first time they appear in a given chapter.
- Please take everything in this book with a healthy amount of humor. The jokes are sometimes barbed. If you find yourself offended by any drawing, please understand that they are drawn from a sense of love and constructive criticism; none should be taken too seriously. Also keep in mind that cartoons about exclusively positive behaviors are seldom funny. For every quirky thing that deserved a comic, there are hundreds of great things that didn't.
- If you can understand the cartoons without reading the accompanying text, good for you. None of the text is interesting — just stare at the colorful pictures!
- Always follow the author's advice about turning the page.
- When you get to the end of a line on the left page, drop to the line below instead of going across to the other page, or it won't make much sense. Similarly, when you get to the end of a page, ignore the page number on the side — it's no use trying to make sense of that in the sentence.
- When the left page is thicker than usual and the right page looks a lot like your hand, you've reached the end of the book. You can then put it away and tell all your acquaintances how great it was and that they should all buy it right now.

Ha. Now there's nothing left to read. You really do have to turn the page.

Oh, come on.

A is for Android

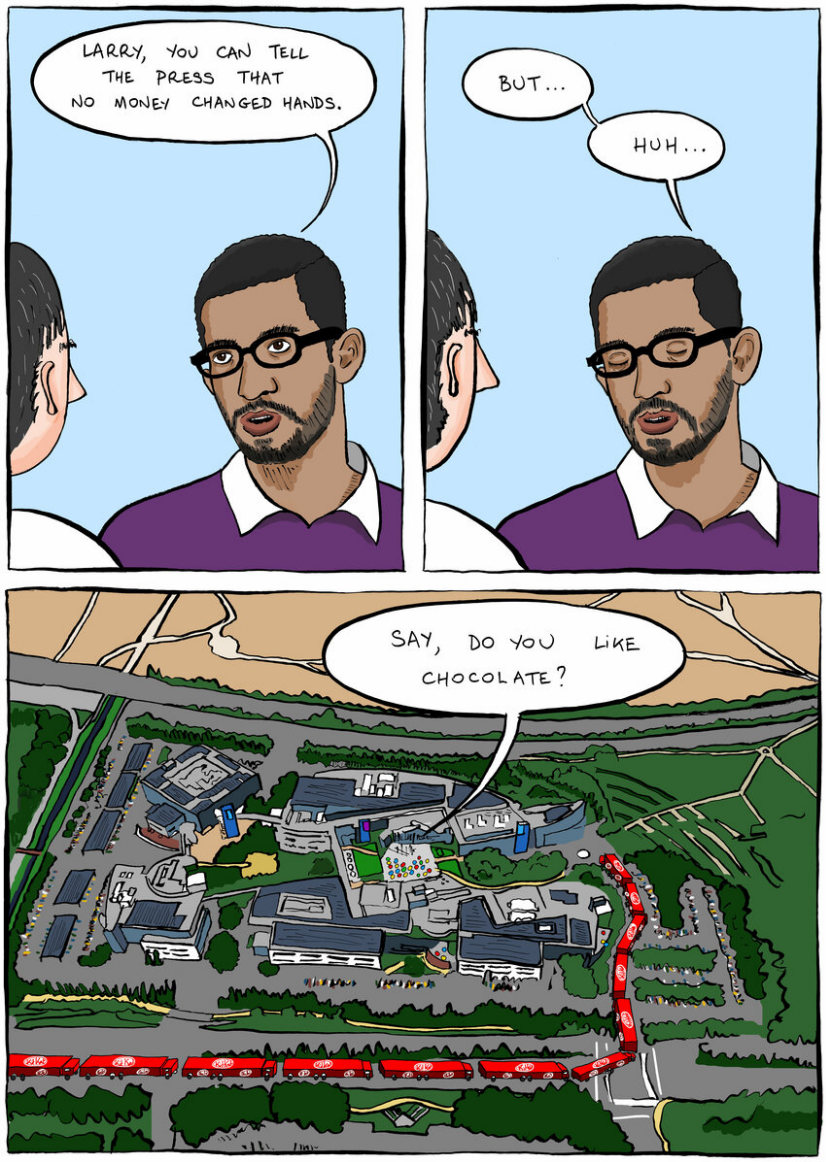
ANDROID is Google's mobile software platform. If you have a phone that isn't an iPhone, it probably runs Android.

Android was initially a separate company, co-founded by Andy Rubin, before Google acquired it in 2005. From early on, every new major version of the software has been named after a dessert starting with the next letter in the alphabet: Cupcake, Donut, Eclair, etc. (The first two were named "Astro Boy" and "Bender".)

Episode #161 (updated with the "desserts" until 2015) 2012.11.30

As Android gained popularity, its most loyal users have been trying to guess the name of the next release. When it was time for the letter K, Google surprised everyone (including yours truly) who thought they were going to unveil "Key Lime Pie":

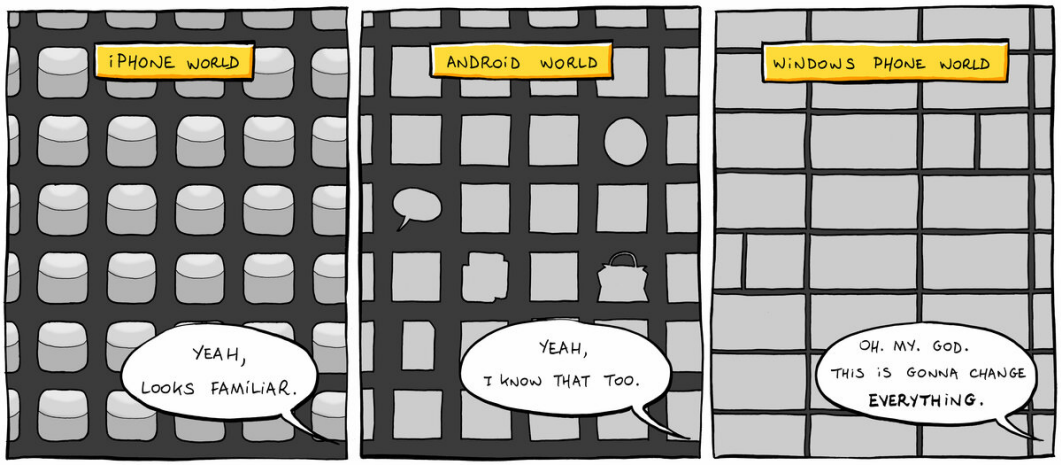
(Well, that was a wasted drawing.) Instead, they called it "KitKat". Larry Page (Google's CEO at the time, backed by Sundar Pichai who was running Android) was prompt to state that "no money changed hands".



Episode #176 2013.09.09

Perhaps not coincidentally, this was around the time when Kit Kat chocolate bars started appearing as a snack option in many Google offices.

Back in late 2010, Microsoft had re-entered the smartphone arena with its “Windows Phone” software, featuring a user interface based on “tiles” rather than application icons.



Episode #81 2011.09.13

(Microsoft ended up discontinuing the Windows Phone platform in 2017.)

The most visible difference between Android phones and iPhones is that only Apple produces iPhones, while Google makes it easy for any manufacturer to build a phone that runs Android. Apple has traditionally released a new iPhone every year, while new Android phones come out much more frequently, each time prompting the press to come up with more clickbait-y "The new iPhone killer?" headlines. The strength of Android, however, is more about choice than about a single flagship device.

"Killer..." That's a word that Google employees (or Googlers) usually refrain from using in this context. Most companies prefer to avoid having their employees talk about the competition with such strong words, or any vocabulary that could be later perceived as the seeds of an evil world domination plan in an antitrust court case. I guess I hadn't read the memo before drawing the following. Oops.

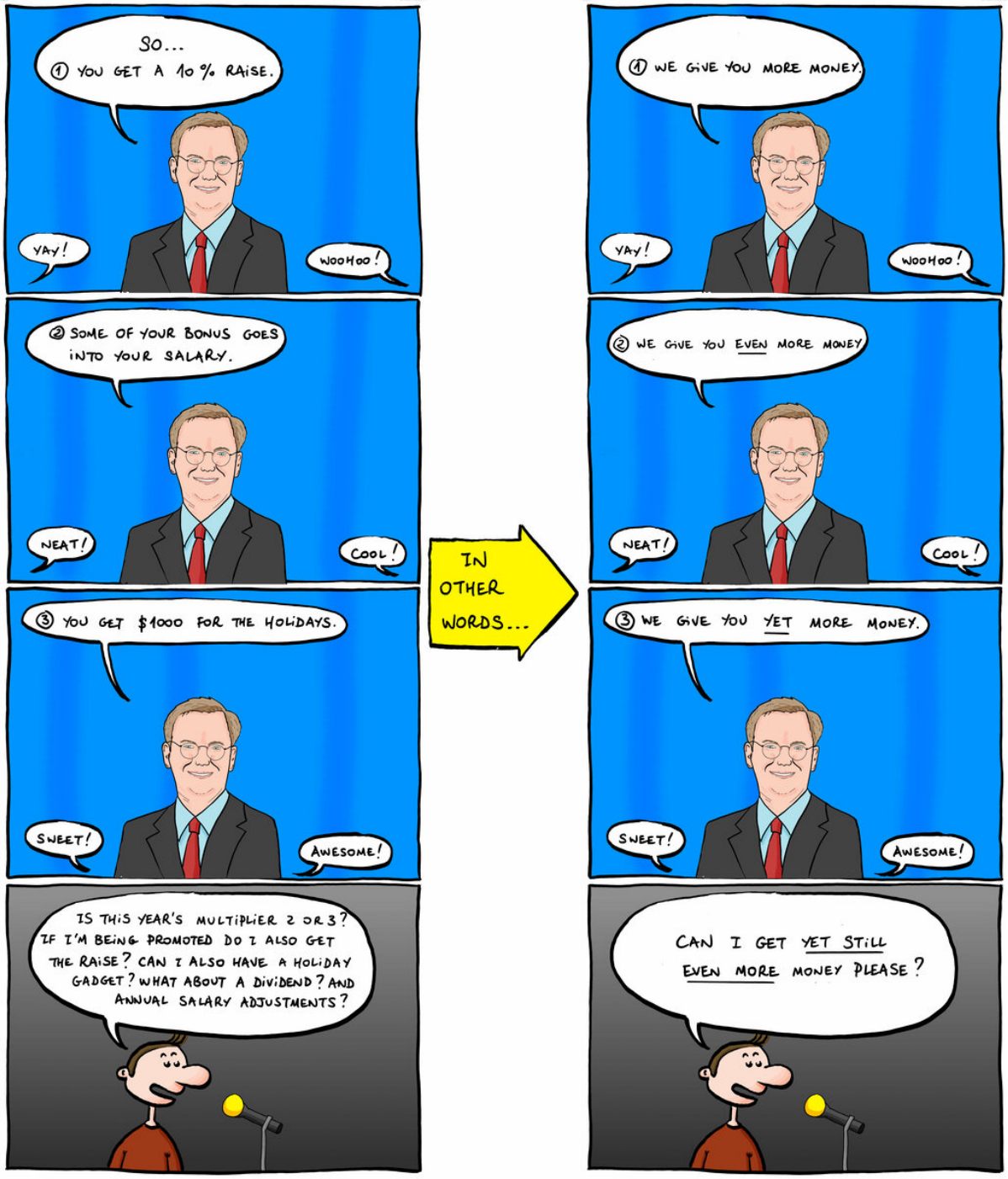
Episode #113 2011.12.30

Thankfully, the iPhone's formidable position in the market (back then and for the foreseeable future) means that Android is far from world domination and I still have my job. For now.

B is for Bonus

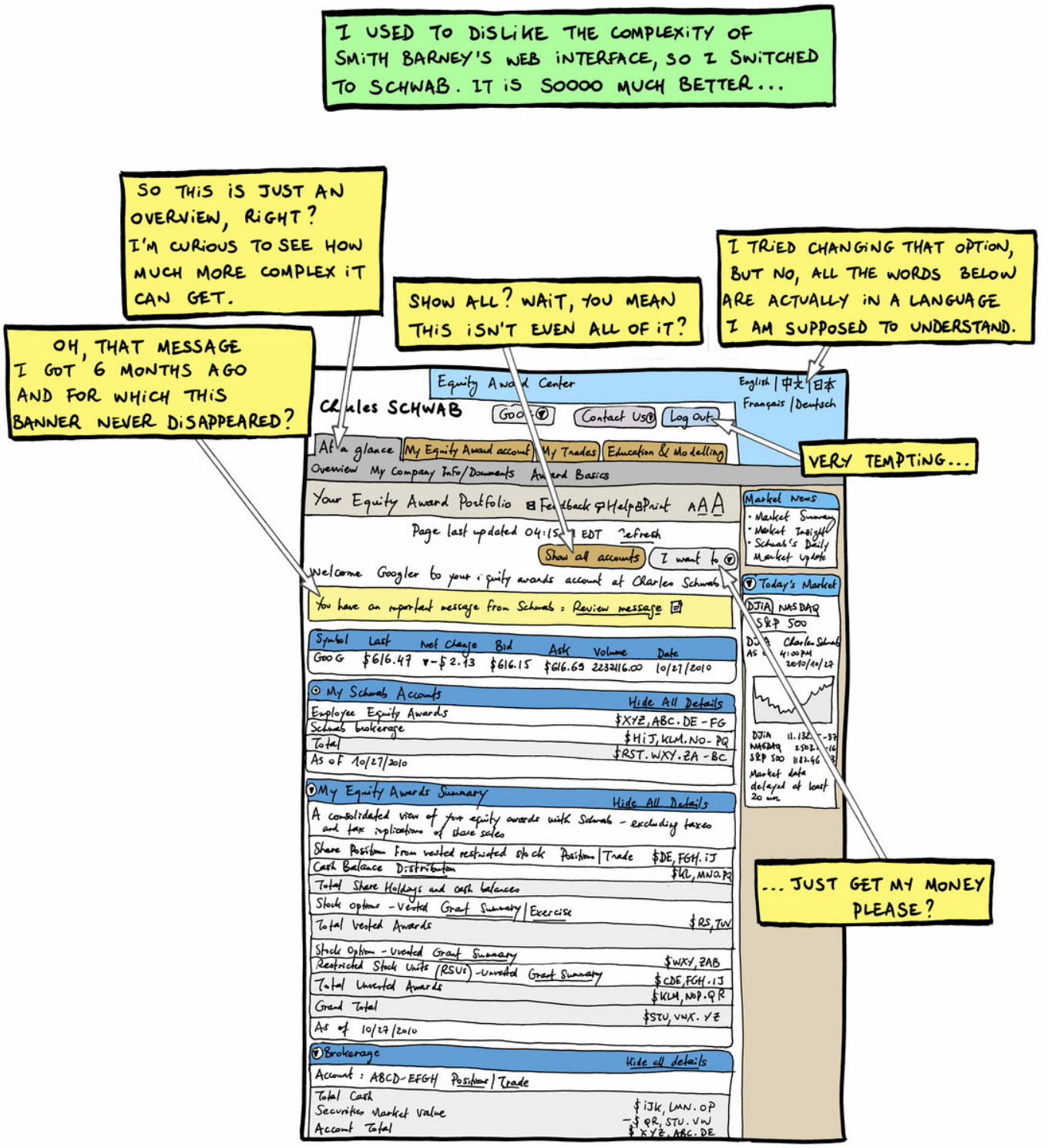
Google, like many other companies, compensates its employees with a base salary as well as a yearly bonus that depends on the employee's (and the company's) performance. Performance is, in turn, measured by how well the various Objectives and Key Results (OKRs) have been met and graded (0.0 for total failure, 1.0 for complete success). And as if that wasn't enough, Googlers used to get an extra little something towards the end of the year.

At a company gathering in late 2010 Eric Schmidt, then Google CEO, announced not only a holiday gift and cash bonus but also major compensation changes aimed at making Google more attractive in the very competitive market of "high tech" jobs. He also announced the doubling of the maximum amount that Google would match for donations to charities. Most of the audience cheered, but as usual a grumpy vocal minority had some extra questions about the change.



I'd like to point out that Googlers can and do ask questions like these to executives, most often at TGIF, a weekly company-wide gathering (more on TGIF later). This isn't very common at other large companies.

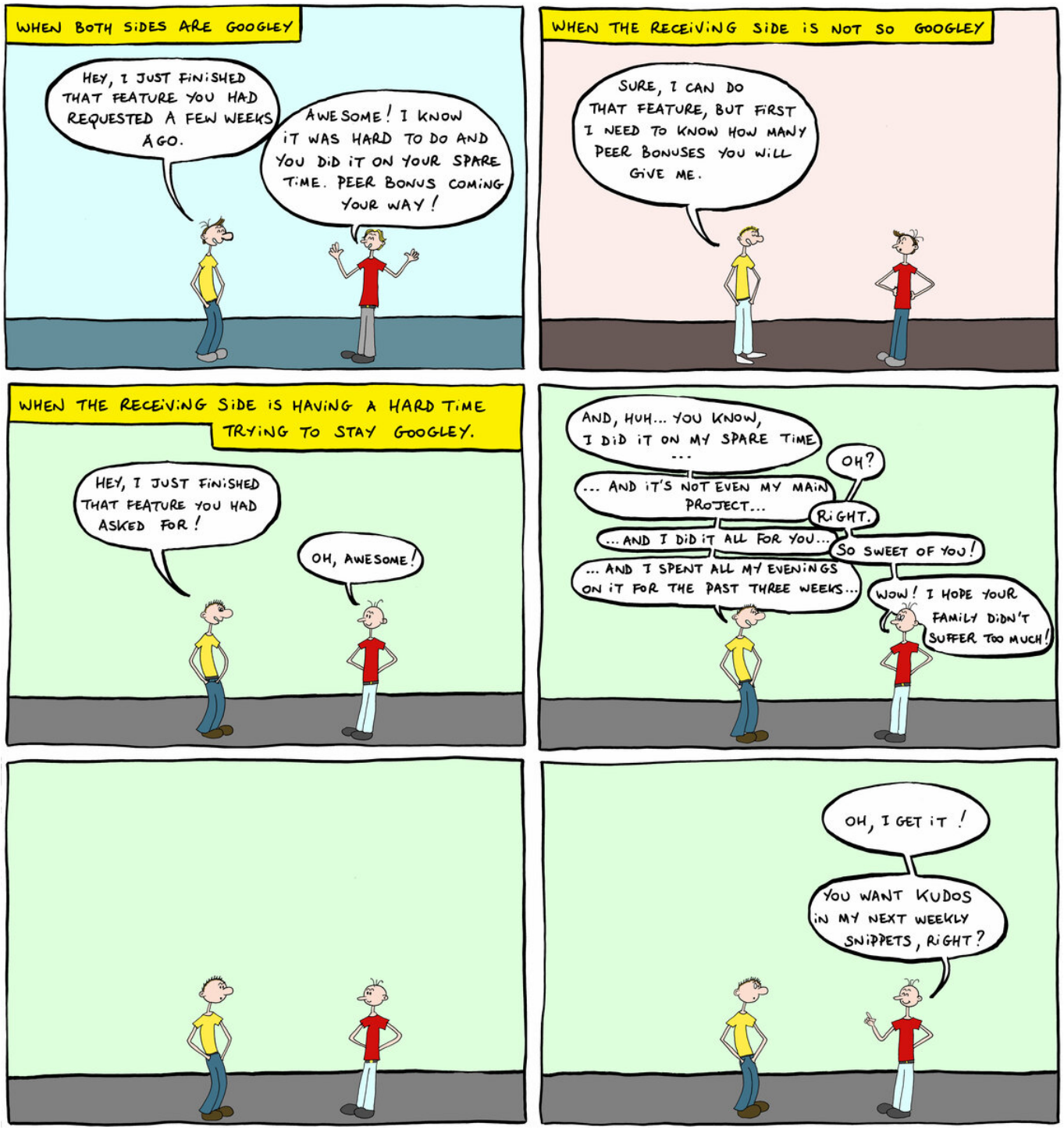
In any case, if you feel a sudden empathy for those poor underpaid Googlers, be reassured: there are also Google stock grants, which employees manage through a financial firm's website. Alas — looking at that website, it seems that tech companies have already hired all the best user interface designers. Life is harsh.



This firm has since redesigned its website. It used to be extremely confusing, and now it is entirely unusable.

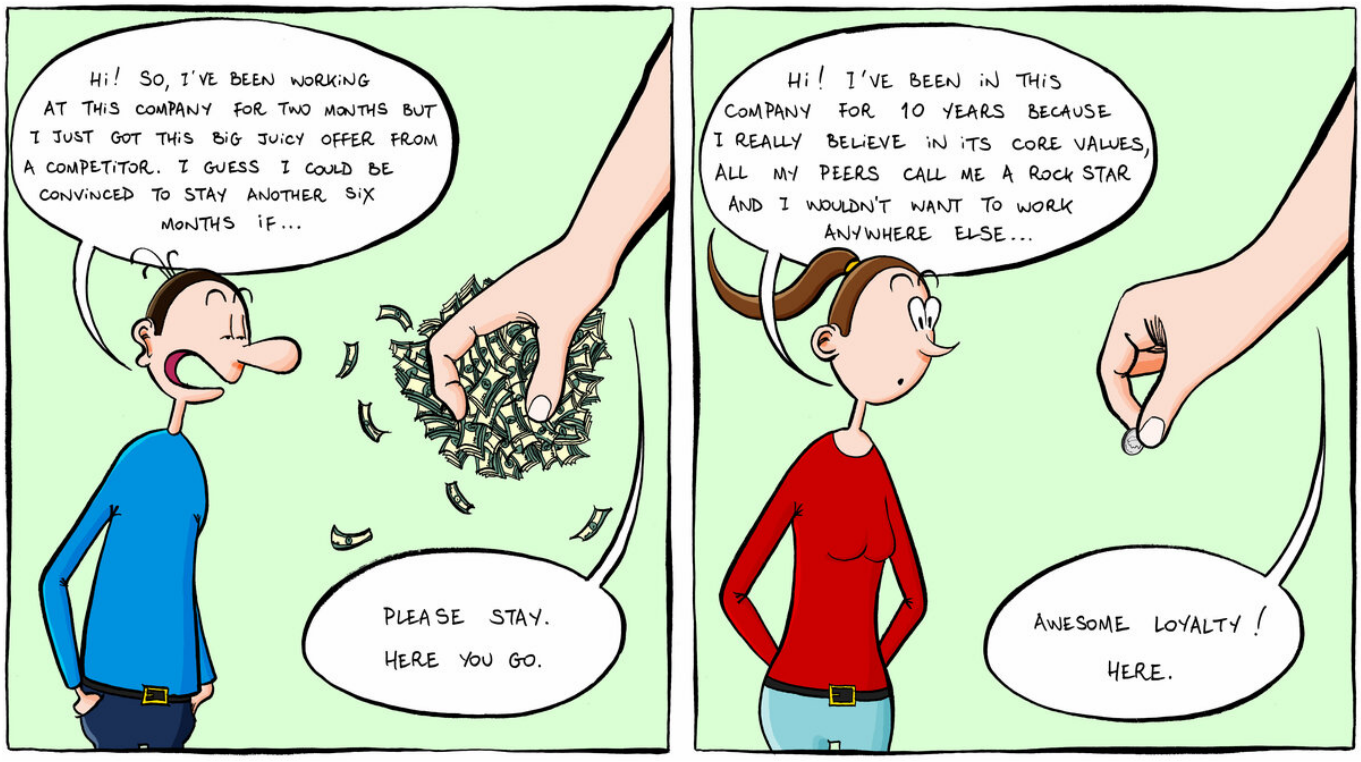
As a Googler, you can also nominate a coworker for a "peer bonus", which results in a little extra income. This is generally encouraged as a way to reward people who behave in a "Googley" way by spending extra time and effort to be helpful to their coworkers.

Another, more casual, way for Googlers to express gratitude is by sending out kudos, a simple shoutout where no monetary compensation is involved. This used to be done with a quick mention in their "weekly snippets" — tidbits that some employees write to summarize what has kept them busy over the week, and that coworkers can read at their leisure.



Episode #162 2012.12.06

Despite these various lucrative channels and Google's efforts over the years to keep their compensation competitive, some talented and money-grubbing individuals always find ways to let companies compete for their talent, at the expense of other less greedy but no less talented coworkers.



Episode #194 2014.09.16

C is for CL

This chapter is a little more advanced. Feel free to skip at first!

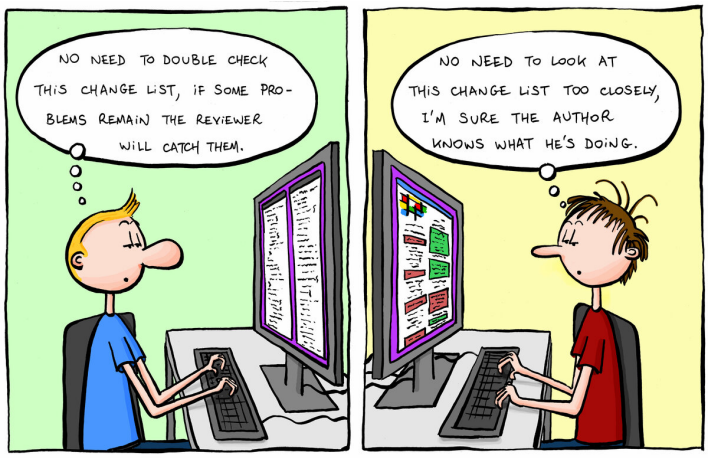
JUST as the cell is the smallest self-contained building block of living organisms, the smallest unit of Google life is the change list or CL.

All the recipes for Google's software (Google Search, Gmail, Google Maps, etc.) are expressed in millions of lines of program code stored in a central repository that's nearly as well guarded as Coca Cola's secret recipe. This repository is called `google3` (it is the third iteration of the way the code is organized). When an engineer wants to implement a new feature, fix a bug, or make an enhancement on any of Google's existing software, they need to change, and potentially add, some pieces of code. This is called a code change. Regardless of how simple the enhancement is, however, a single change to a small area of the code base is seldom enough. If you wanted to raise your bed by a couple of inches, you would need to lengthen all four of its legs, and you wouldn't want to do just one of them today and the others tomorrow. Likewise, an engineer usually needs to perform multiple changes concurrently to achieve the desired outcome. All those changes go into a CL.

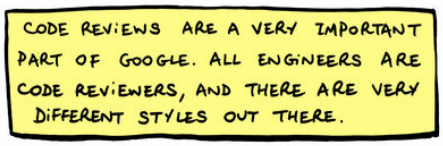
If you ever wander the corridors of a Google building, you will no doubt realize CLs are all that engineers talk about. "I'm nearly done with my CL", says an exhausted Googler. "Sorry I haven't looked at the CL you sent me yesterday yet", replies a busy colleague. "It's quite large — could you split it into two smaller CLs?"

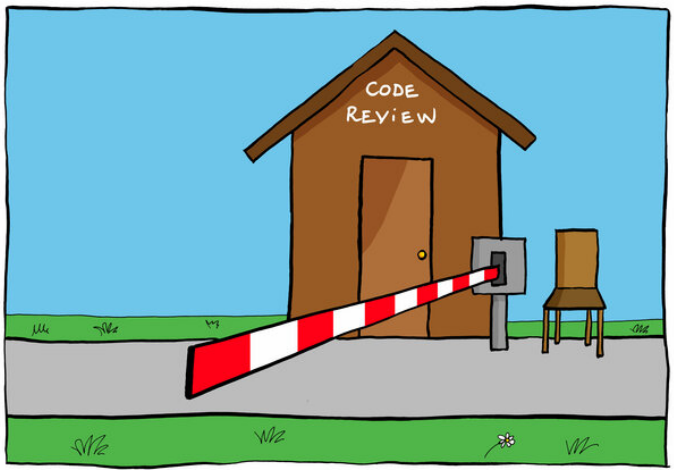
Why would a Googler need to look at someone else's change list? Well, it would be bad if any engineer could change the code for any product without asking or notifying anybody. Multiple processes are therefore in place to guard against potential mistakes. The most obvious of these is the code review: an engineer who proposes a change needs to have it reviewed by at least one other person. In addition to catching more mistakes, this helps "spread the knowledge" about how the code works and how it is changing.

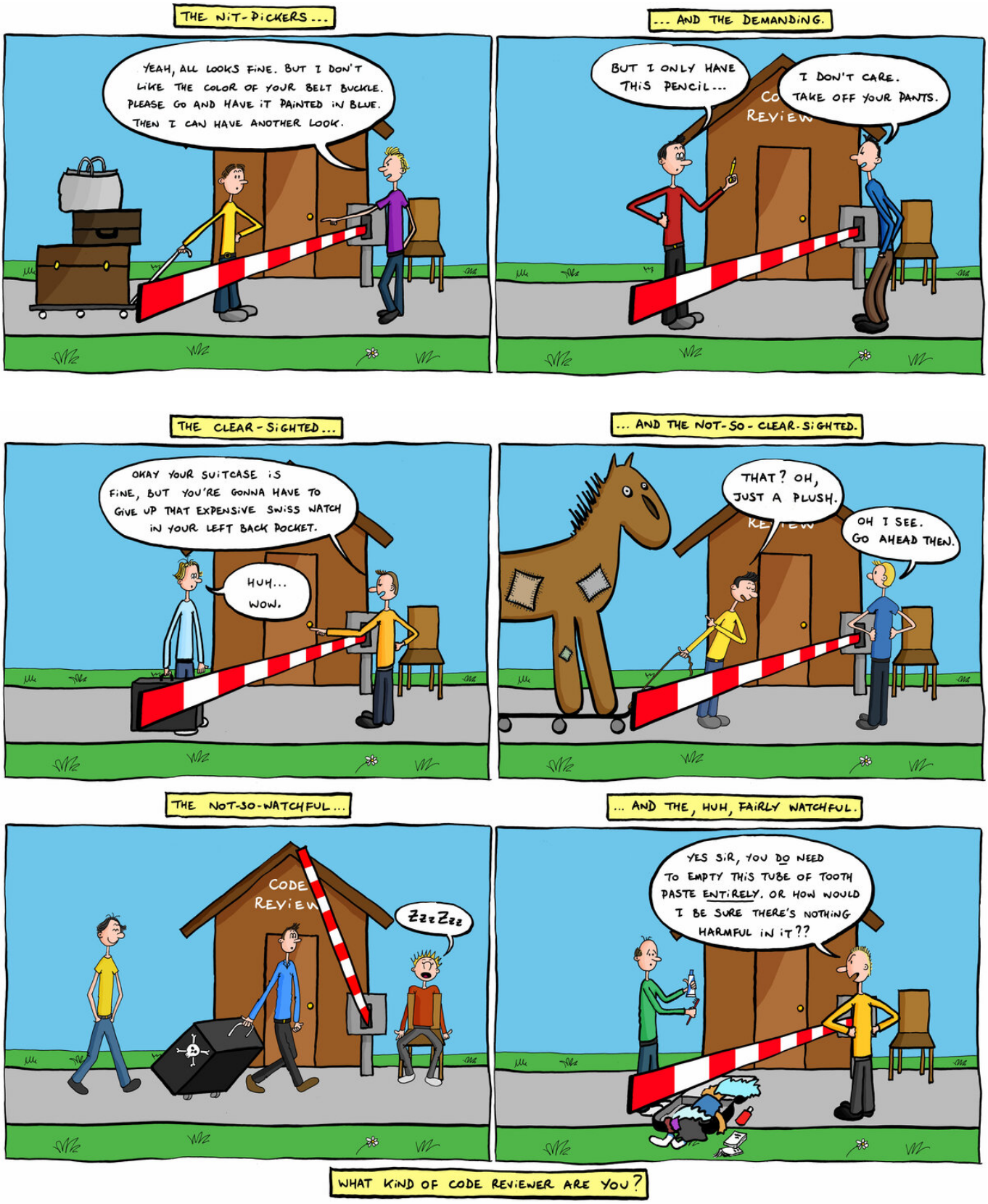
Unfortunately, both the author of the change and its reviewer are usually busy people with other fish to fry, and they might not always put their heart and soul into this necessary review process.



Episode #67 2011.07.18

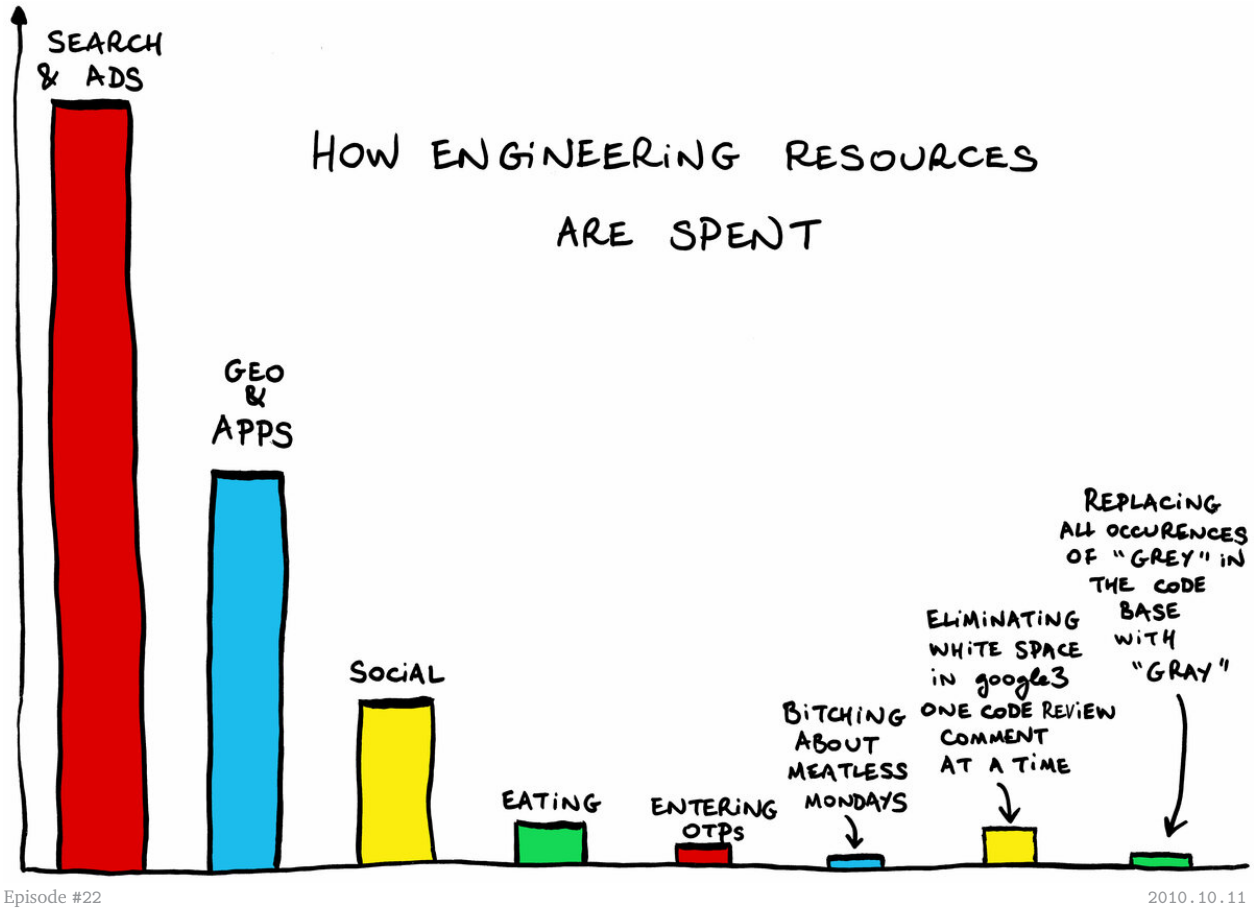






Episode #40 2011.01.05

Of course, some reviewers are picky. Moreover, software engineers can be obsessed with things as silly as making sure no trailing whitespace (unnecessary albeit invisible space characters at the end of a line of code) tarnishes Google's code base. Combine all of that with a company's worth of nitpickers and you get a substantial amount of questionably spent resources.



(OTP stands for one-time password — more on that later!) Thankfully, since that cartoon was published, the Google engineer's toolkit has improved enough that nobody needs to manually take care of white space anymore (it gets removed automatically).

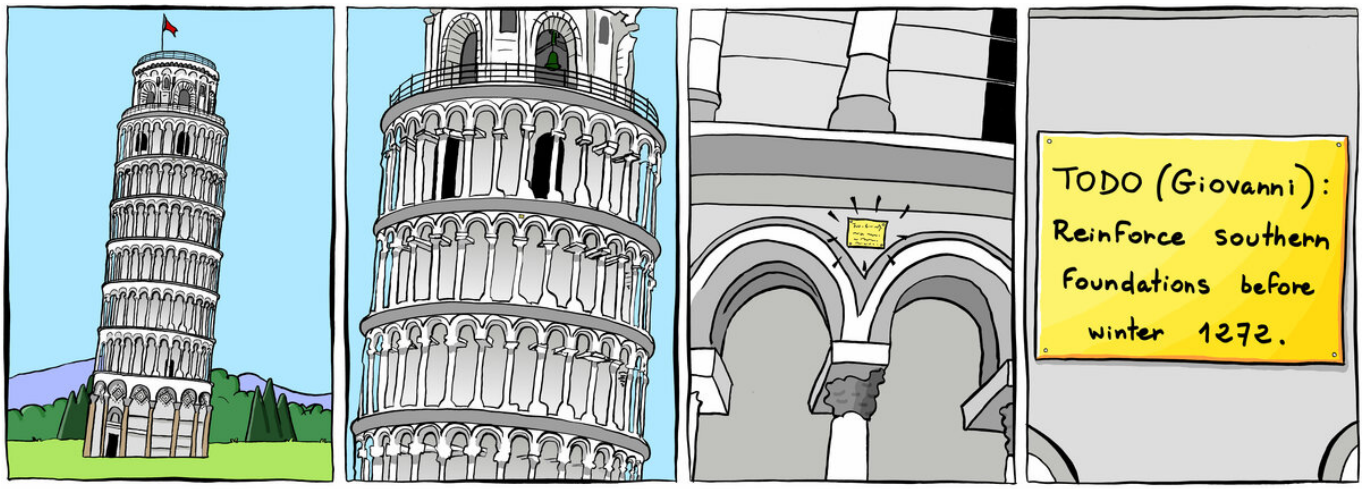
Often, during a code review, the reviewer — let's call her Rebecca — will ask for something to be enhanced in the code that the change list author — let's call him Albert — has written, but the enhancement would take more time and effort than Albert is prepared to spend right away. To build on the previous metaphor, maybe Albert is raising the bed's four legs by a few inches, and Rebecca argues that the bedside table should also be raised so that the heights match. But all Albert wants is to quickly get the bed raised and be off to his next task, which is already overdue, so he replies that whoever is going to lie down here can just reach down a couple extra inches to whatever is on the table, at least for now.

After a couple of round-trips where Rebecca insists and Albert pushes back, they both agree to meet in the middle by having Albert write a TODO. He is going to move forward without implementing Rebecca's suggestion in this particular change list, but he agrees to do it at a later time. To show that and to obtain Rebecca's LGTM ("Looks Good To Me", meaning that she's satisfied as a reviewer), he leaves a written note embedded in the code describing what needs to be done and by whom. For instance:

```
TODO(Albert):  Also lengthen the bedside table's legs.
```

A TODO can also be written by an engineer of their own accord, to leave a reminder for a potential future improvement.

So what happens next? As you might guess, it's not rare for everyone to proceed with their busy lives, and for both Albert and Rebecca to forget about the pending task. Thus, the supposedly temporary code stays around for months or years. By now, Google's code is littered with more TODOs than any Googler would probably like to admit.

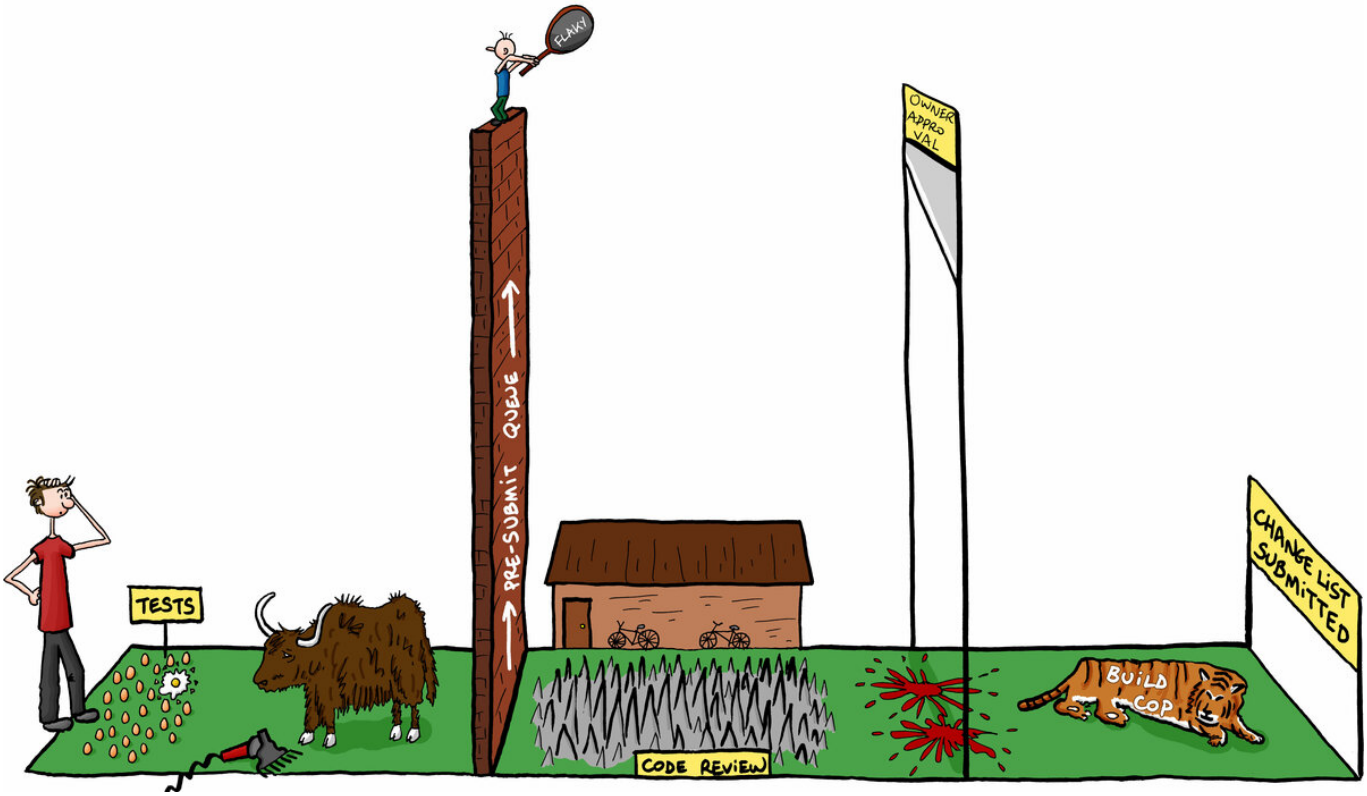


Episode #152 2012.08.15

Code reviews are only a small part of the story, however; a software engineer's life is no walk in the park. Many hurdles need to be overcome between the time an engineer starts implementing their idea and the time the resulting change list actually makes it into the central code base (we say that it has been "submitted").

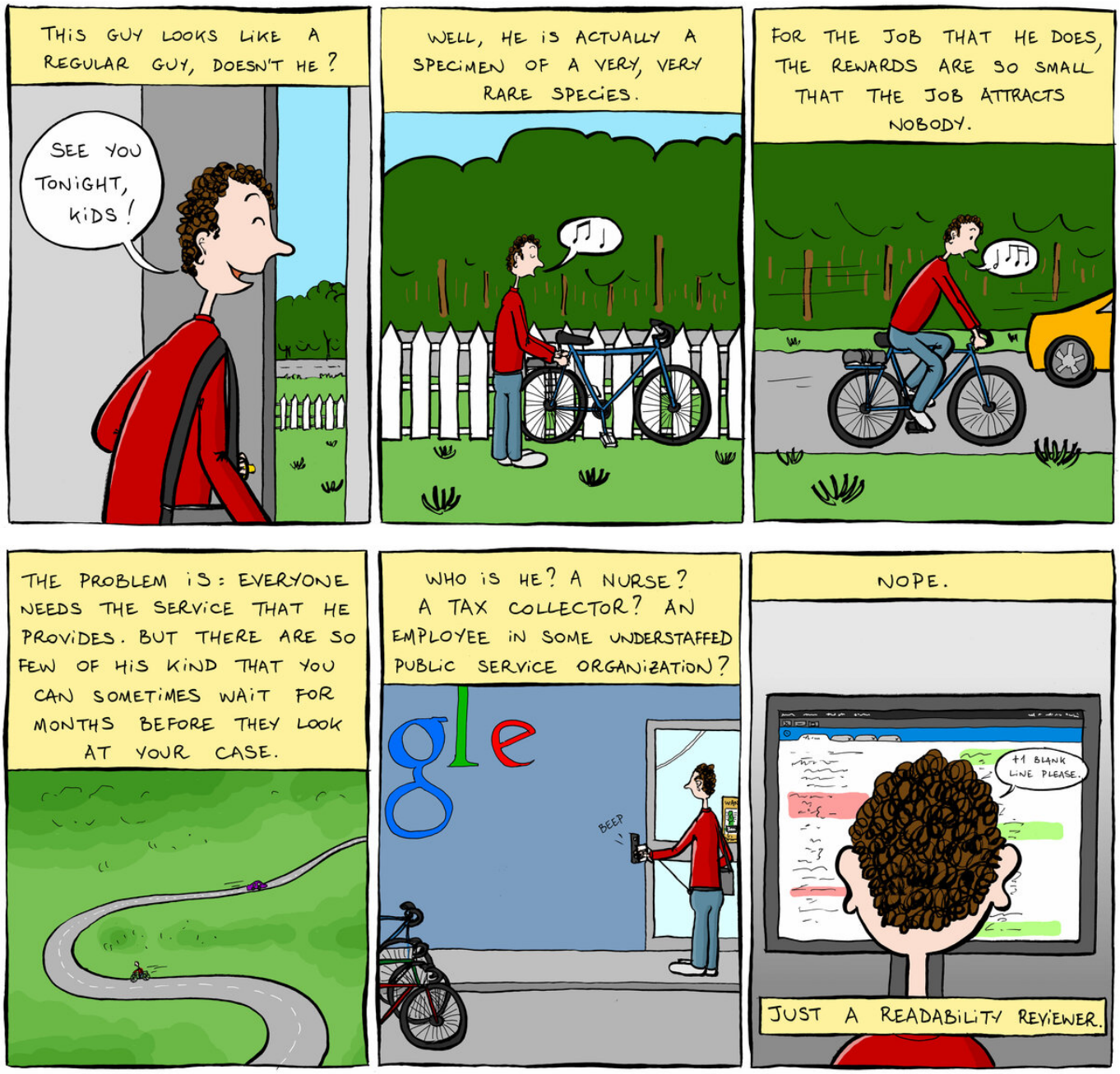
There are the tests, auxiliary pieces of code executed regularly to ensure that the main code keeps doing what it should, despite the constant changes. Those tests are easily broken by a pending change if you're not careful. Then there is what is referred to as yak shaving: the seemingly endless process of clearing up one layer of hairy complications at a time, and just when you think you're done, discovering that there's yet another layer underneath.

Once your change list is ready to be sent out for review, it is automatically added to the back of a "presubmit queue" where it will wait its turn to undergo another long series of tests. Some of those can be flaky (meaning that they can randomly fail for no good reason), in which case your change list needs to be queued up again. After the tests pass, you'll need an approval from one of the "owners" of the code (this is so that one team cannot randomly change another team's code without their approval). And finally, if your change has caused anything to stop working (it has "broken the build"), beware the wrath of the "build cop".



Episode #10 2010.08.16

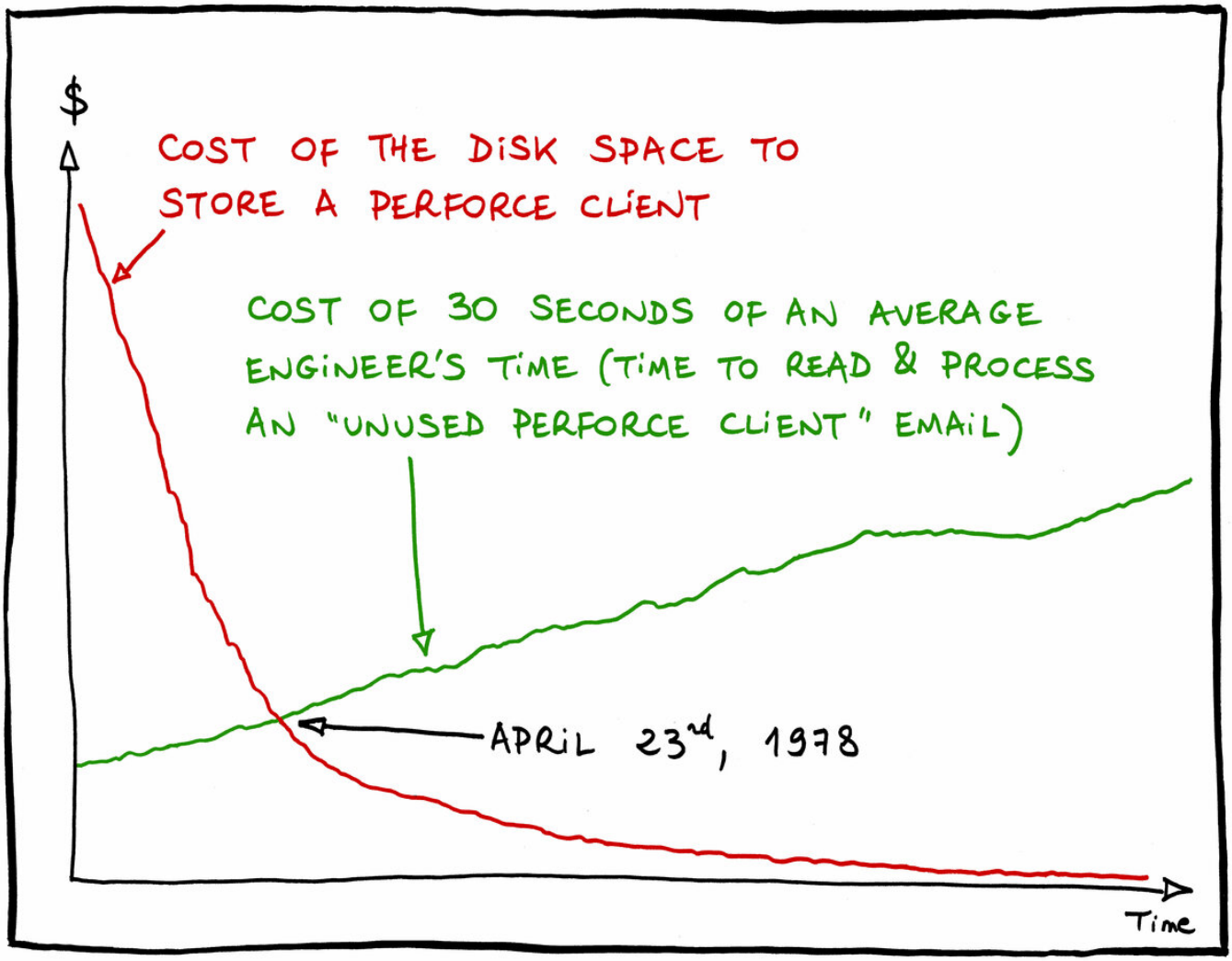
And that's not even the end of it, because code can be written in a variety of programming languages, and not every engineer is proficient in all of them. They're like different modes of transportation, used for different purposes. If you're visiting a friend in the countryside, chances are you can just drive there. If you're planning a trip by yourself on a small island, you'll need a few different means of transportation: a car or a bike, a boat, etc. Likewise, if an engineer is making a complex change involving several parts of Google's code base, they will often need to write parts of the change list in different programming languages. The engineer must thus demonstrate that they are proficient in each language: they must have readability (meaning that their code is good enough to be read and understood by others), a sort of driver's license for a given kind of vehicle. Once you have the driver's license (readability) for a given means of transportation, you're good to go for your whole career at Google. And how do you obtain this driver's license? Behold Google's equivalent of a driving instructor: the "readability reviewer".



Episode #180 2014.01.08

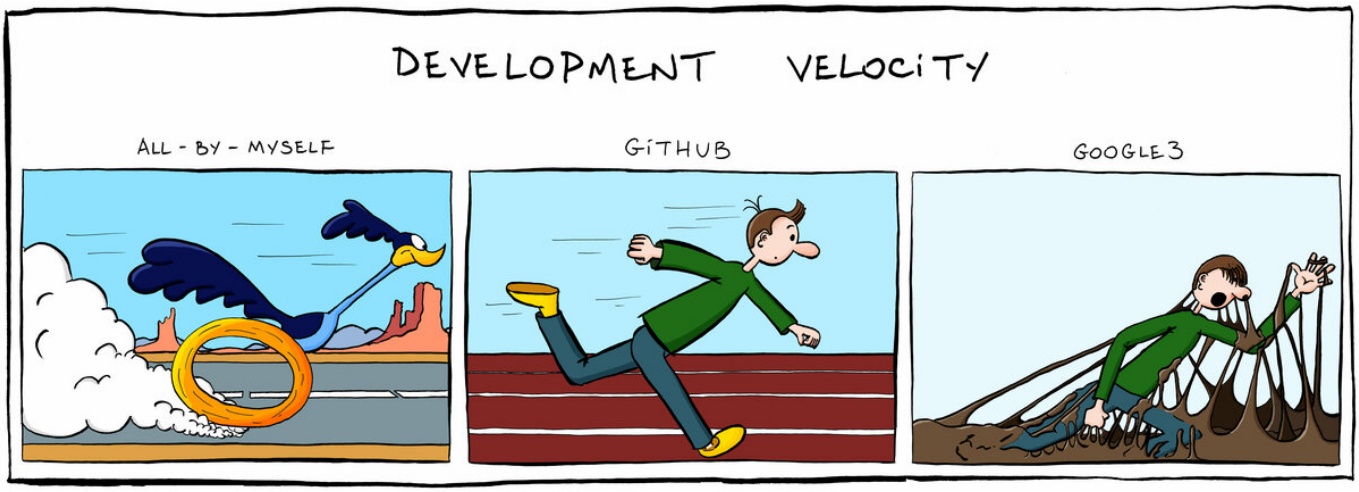
It used to be the case that creating a change list, and seeing it all the way through to submission, required a small workspace called a "Perforce client". Each of these clients could be reused for several change lists, but Googlers would usually keep a few of them within reach at any given time (in case they needed to work

on several change lists simultaneously) until nagging emails reminded them to clean up their unused clients — presumably because each of them needlessly used up a smidgen of computer storage.

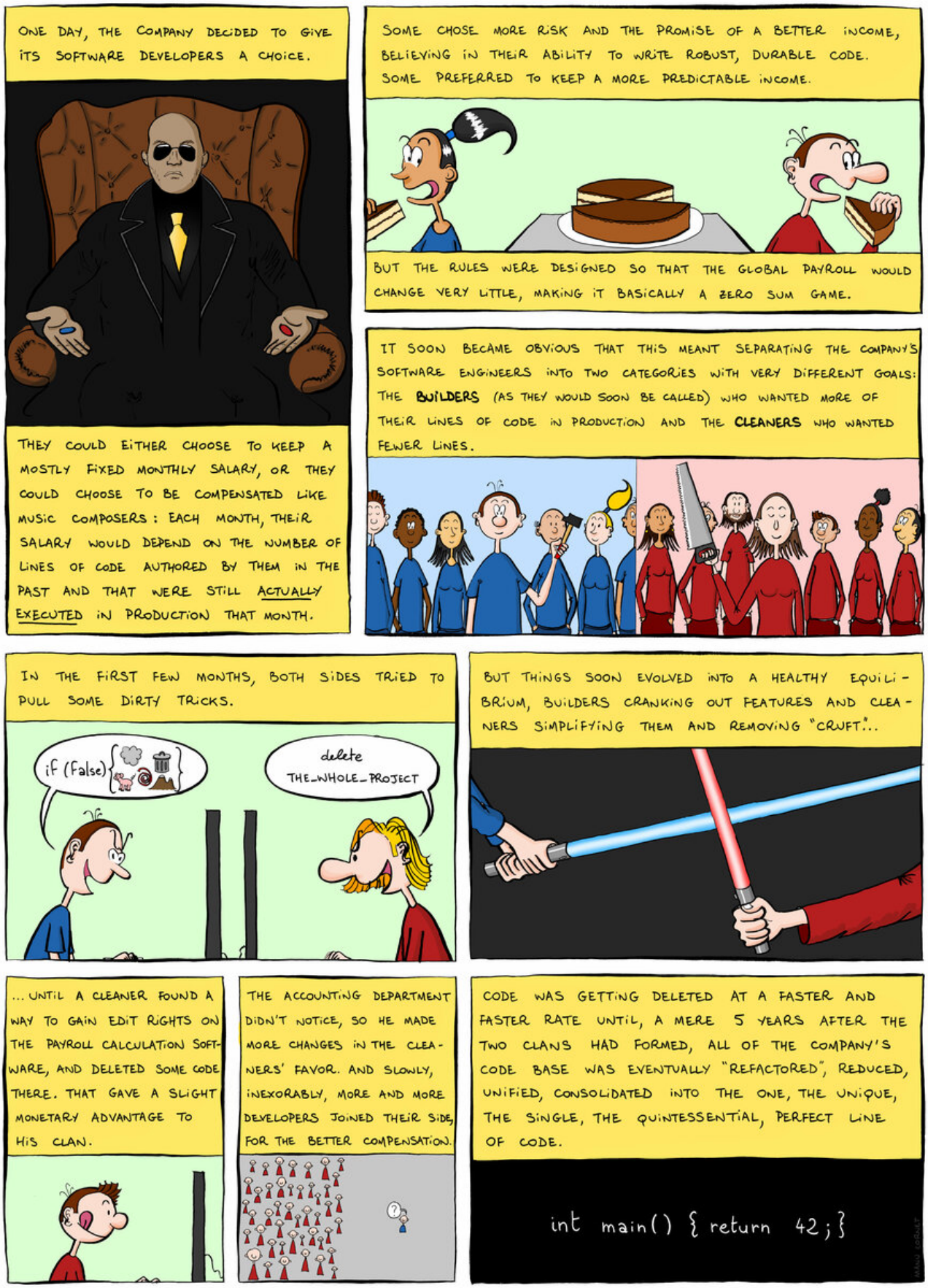


Episode #77 2011.09.07

As you can imagine, all this overhead made it significantly slower to work on Google's code base than on a public collaborative platform such as GitHub, let alone compared to working solo.



Episode #138 2012.05.23



Episode #174 2013.04.01

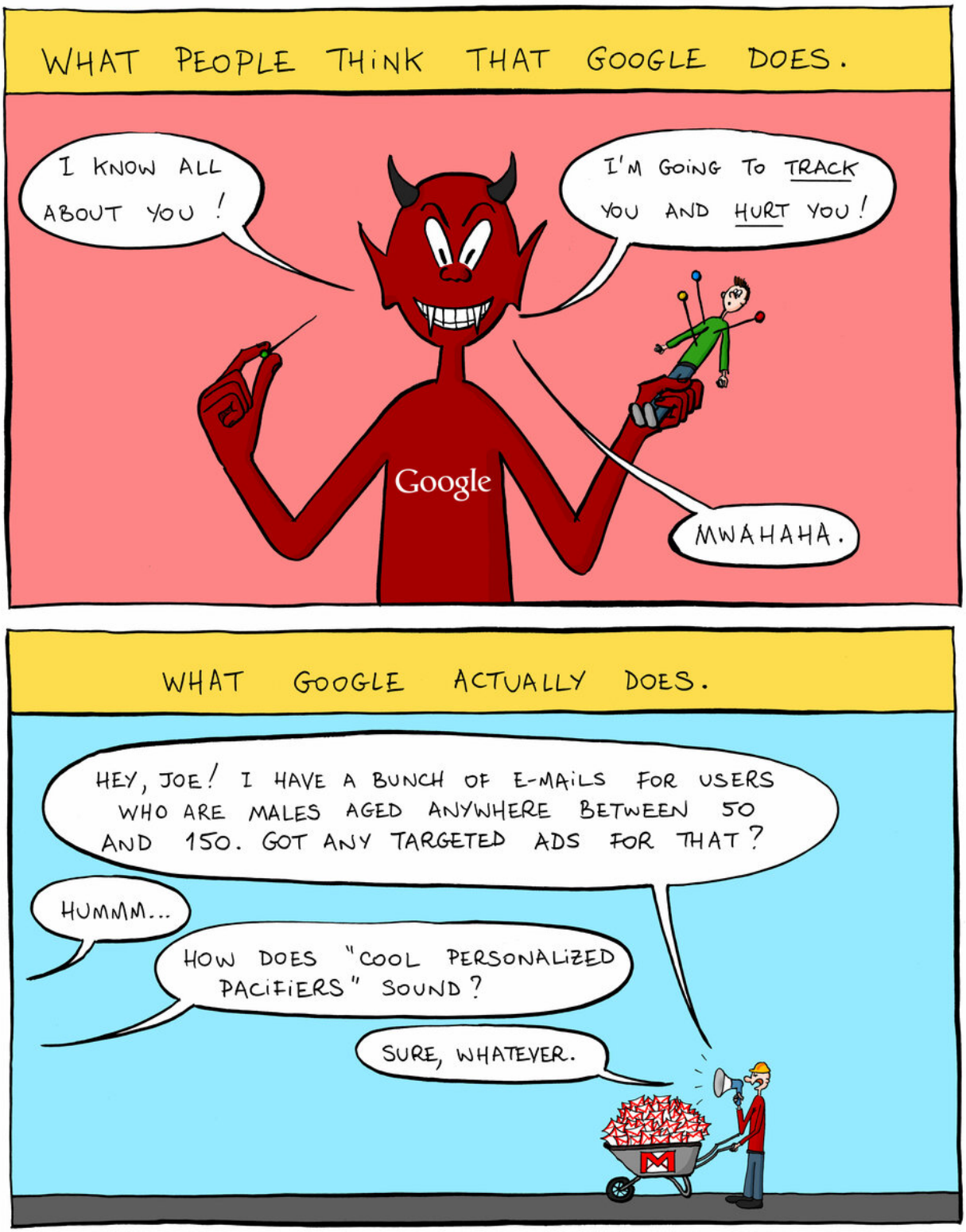
This last panel is a reference to *The Hitchhiker's Guide to the Galaxy*, where a supercomputer, on being asked the "Answer to the Ultimate Question of Life, The Universe, and Everything", thinks for 7.5 million years before producing the answer: 42.

D is for Don't be evil

"Don't be evil" has been Google's motto since its beginnings, and until it became Alphabet in 2016. That lofty ideal became a target for disgruntled people who found the company's actions questionable.

There was of course the general outrage when Gmail users realized that some of the advertisements displayed alongside their messages were actually targeting the email's content — which seemed to imply that Google was "reading their email". Despite the company's explanations, there's always been a discrepancy between some of the public's perception that their privacy was being violated and Google's defense that only machines were reading messages, which must happen anyway otherwise the messages could not be shown to the user.

Another common source of queasiness has been how much Google "knows" about you and how it is able to present you with uncannily targeted advertisements. Personally, I often find the targeting a bit... off.

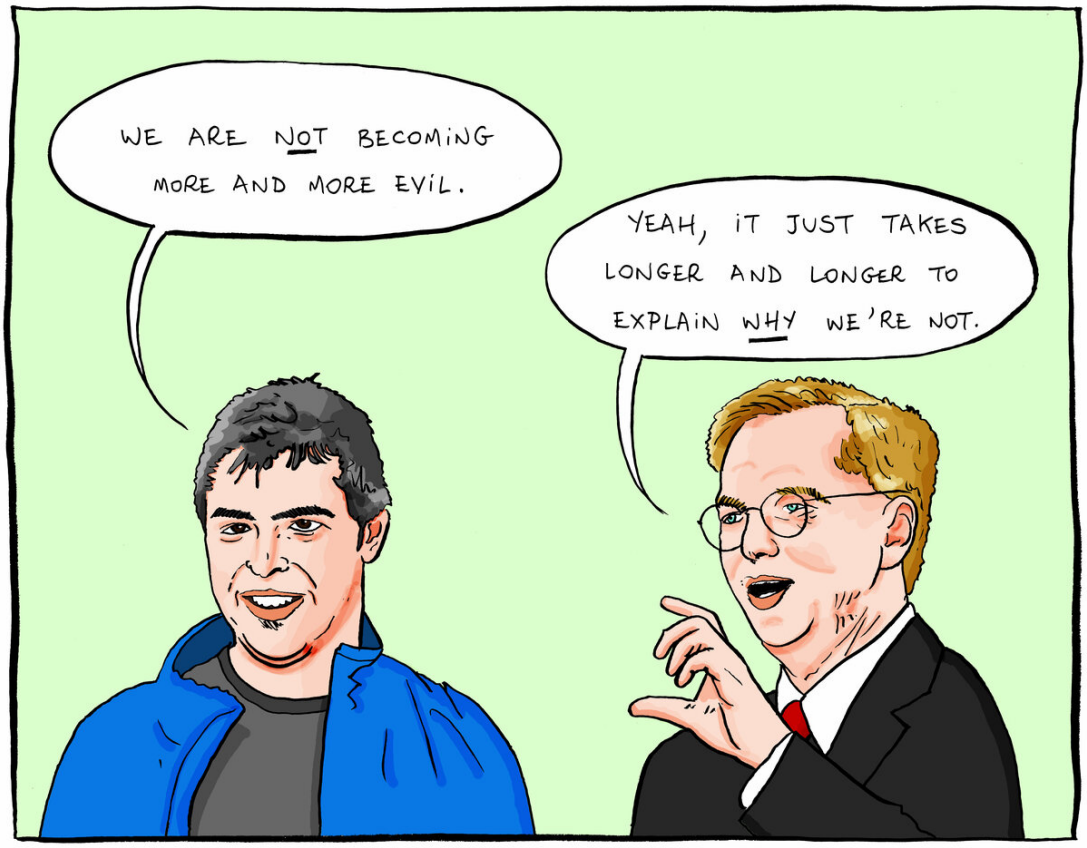


Episode #126 2012.03.06

In late 2011 and early 2012, a series of unrelated events happened in quick succession, causing the press to declare that Google had "finally gone evil" (a phrase that tended to get repeated over the years, not unlike the "iPhone killer" headline). First, there was the inadvertent collection of personal information over wi-fi by cars gathering location data for Google Maps (that happened a couple of years earlier, but the legal consequences were still unfurling). There was also a general backlash against the policy that required users of the Google+ social network to use their real name and not a pseudonym (more on this in the next

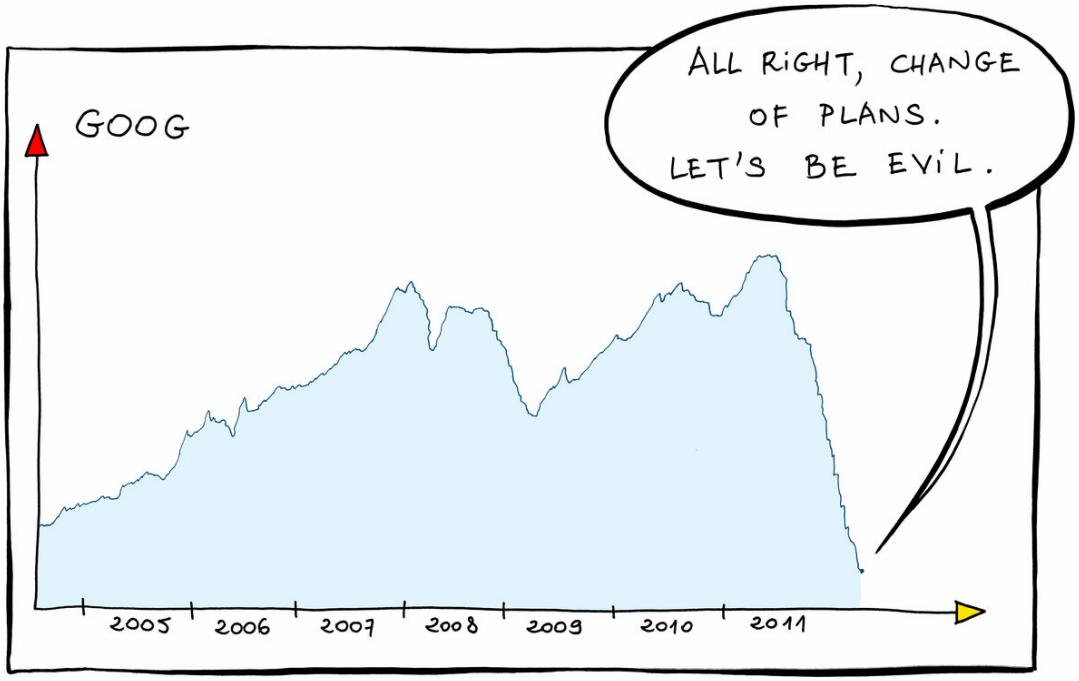
chapter). And in February 2012, Google was accused of working around the privacy settings of Apple's Safari browser to be able to track its users.

Personally, I do not believe that there was any deliberate wrongdoing in any of those examples. Engineers often focus on solving a particular technical problem (Kant would call it a hypothetical imperative) and can be oblivious of whether it makes ethical sense in the bigger picture (categorical imperative). Nevertheless, Google found itself needing to explain its way out of evilness more often than before.



Episode #122 (these are Larry Page and Eric Schmidt) 2012.02.23

Lastly, there's always the crucial question: under how much duress does a company need to be before it decides that its survival (as measured by its stock price) trumps its original values?



Episode #71 2011.08.05

E is for Emerald Sea

In early 2010, Google's leadership was worried. Facebook's ascension seemed inexorable. The Internet was becoming more and more social. Squinting towards the horizon, Google's top executives were already seeing the day when their company would become irrelevant due to lack of integration with the world of social networks.

They couldn't let this come to pass. A state of emergency was declared, and thus was born project Emerald Sea (which a blogger visiting the Google campus mis-overheard and reported as "Emerald City" — I actually like that name better). A flood of social features would be summoned, and its majestic tides would inundate all of Google's products with their green glittery social goodness. Facebook would see that we meant business. All hands on deck!

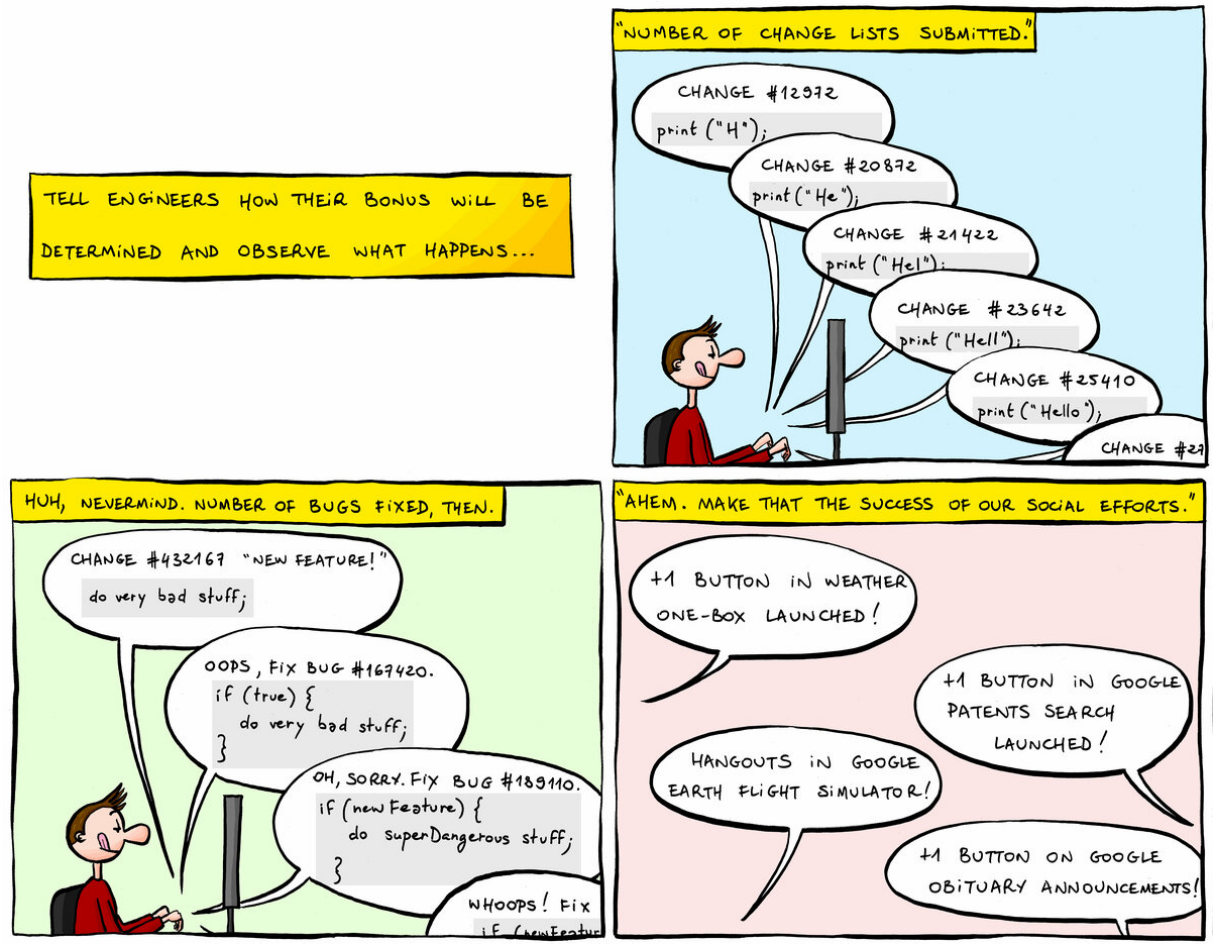
How do you mobilize an army of overpaid engineers? Well, of course, you give them more money. Larry Page, having just become CEO again, announced that 2011's bonus would depend on the success of this social effort, through a "social multiplier". All products were to acquire a social dimension and become better connected to one another, and the center hub would be a page showing a stream of recent updates from your friends, similar to other existing social networking sites.

This complex puzzle of social features would need to be built quickly: executives decreed that, within 100 days, a first version of the software would see the light of day. Experienced engineers left their current projects to join the social teams. Some called this event an Ursquake, from the name of an executive who contributed in declaring the emergency. As the deadline approached, however, Googlers started to ask when they could dogfood (test an early internal version) the shiny new hotness. Just like a carrot attached to the end of a wooden pole held by a donkey rider, that date didn't seem to get any closer.

Episode #24 2010.10.12

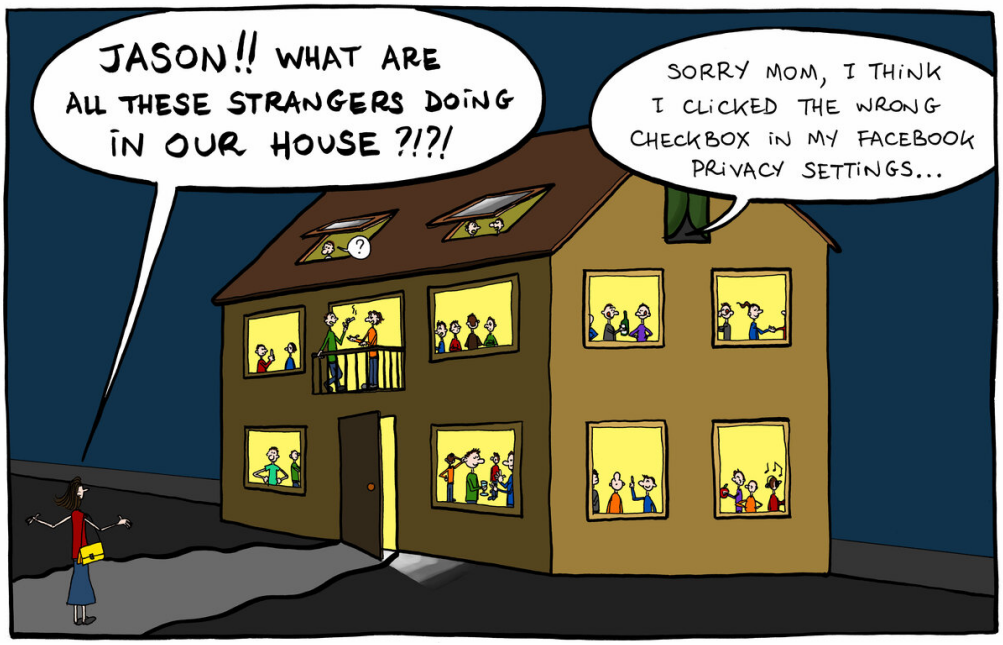
But no matter, the whole company was mobilized, digging irrigation ditches to ensure that the wave of precious emeralds would reach inland far enough to irrigate every single product. The most straightforward way of integrating anything with the social effort was to add a "+1" button, the equivalent of Facebook's

"Like" button, somewhere in the user interface. Another option was to integrate with "Hangouts", the instant messaging piece of Emerald Sea.



Episode #72 2011.08.08

A key part of the Emerald Sea project was to provide a nifty interface for people to control exactly what to share and with whom. That would contrast with Facebook's (back then) confusing privacy settings.



Episode #86 2011.09.22

Another key part of the project was the "real names policy": users could appear on Google's social network only under their real name, and not a pseudonym.

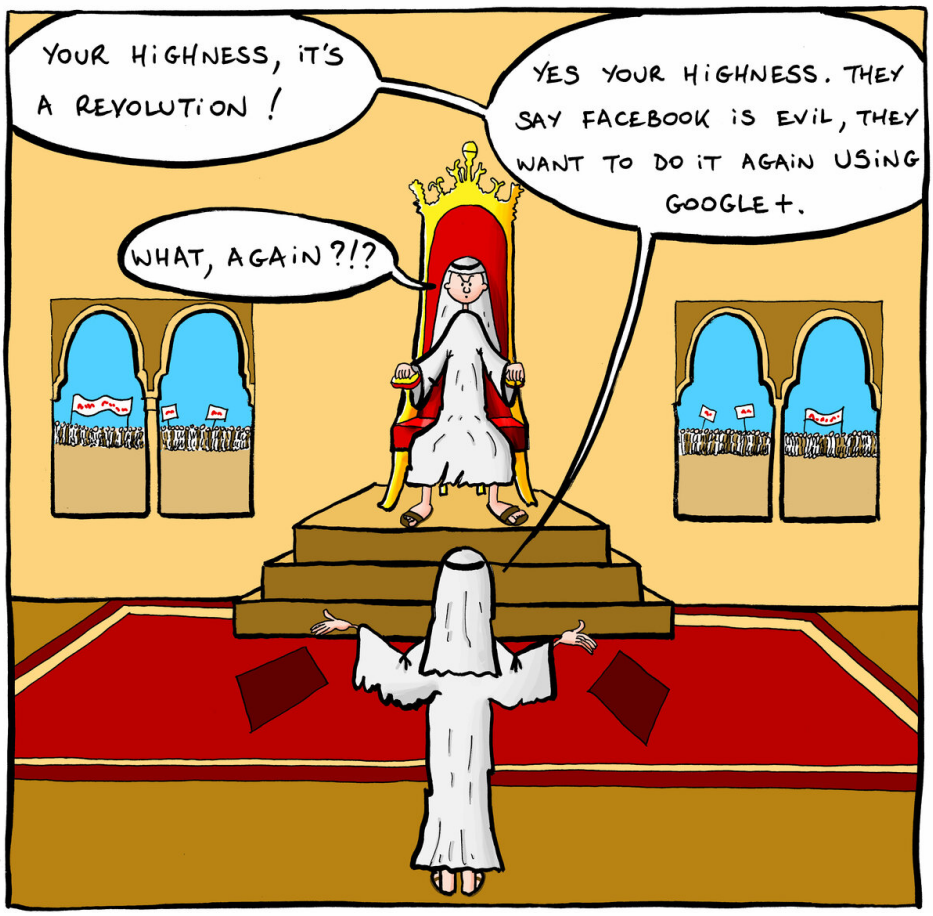
Although "dogfooders" had a clear negative reaction to this policy, company leaders continued to support it. The assumption was that people would be better behaved without the cover of anonymity. To demonstrate this, an executive shared a comparison of two screenshots showing two different streams of comments on PicasaWeb, the firm's existing photo-sharing site, that seemed to have been hand-picked to make his point.

Episode #41 2011.01.18

When "Google+" was eventually released to the public, it was time to rejoice. Google was saved. The competition was in awe. Lars Rasmussen, the high-profile co-creator of Google Maps and Google Wave who had left for Facebook a few months earlier, must have been having second thoughts...

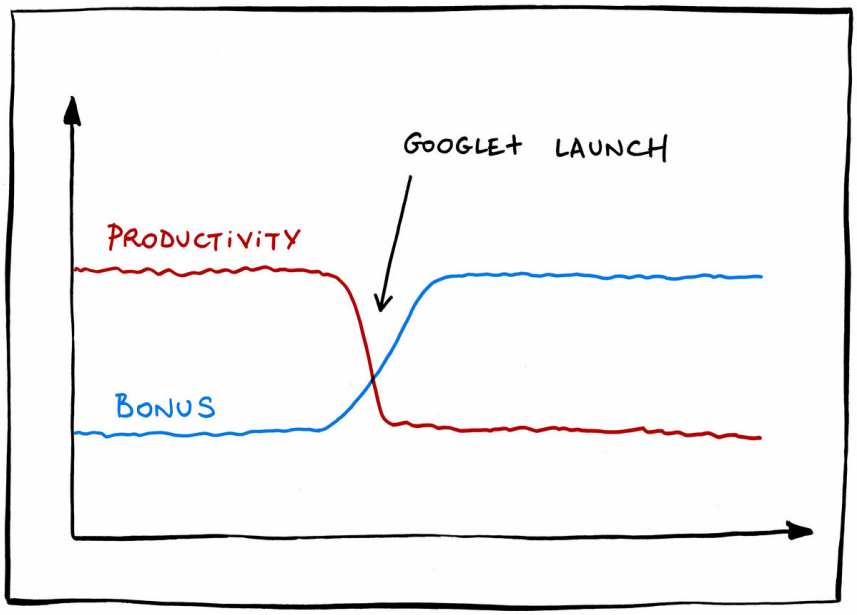
Episode #63 2011.06.29

...and the 2011 Arab Spring, which had been facilitated by Facebook, was ready to happen all over again.



Episode #68 2011.07.21

The social network's Google-internal-only version allowed employees to share their opinions without fear of releasing confidential information to the public, and was widely praised for better connecting employees.

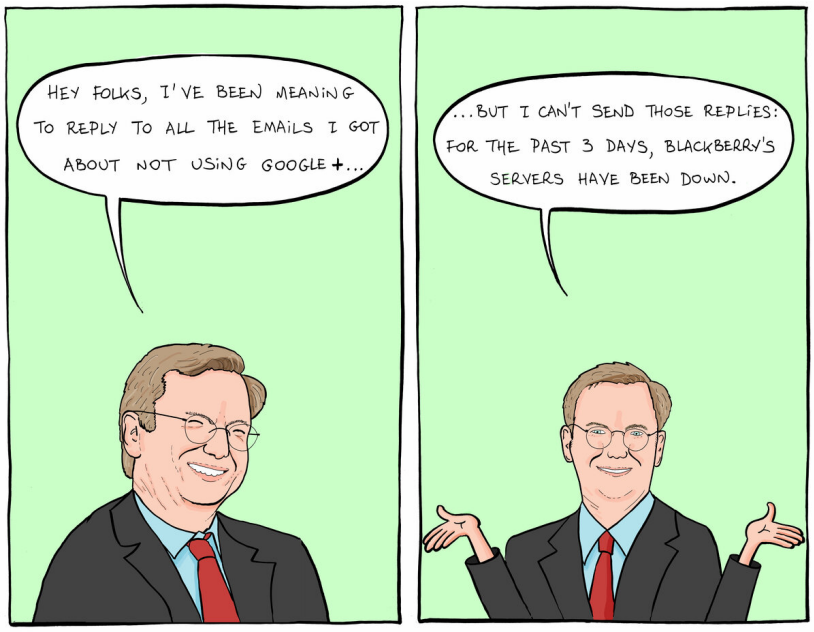


Episode #65 2011.07.13

But doubts soon started to appear. Googlers used the new software to interact with coworkers, but their friends and family didn't have enough incentive to make the jump from Facebook. The public also started to notice that, after the heat of the launch, well-known Google employees didn't seem to be using the public Google+ at all. A quick glance at laptop screens on the company shuttles could easily confirm that.

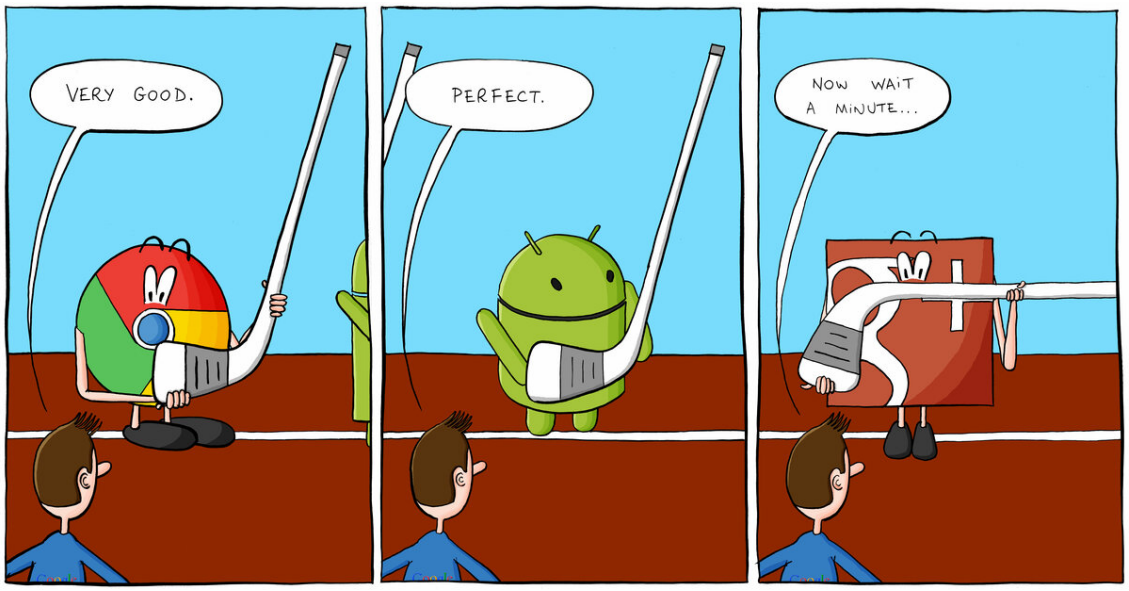
Episode #120 2012.02.15

Episode #92 2011.10.10



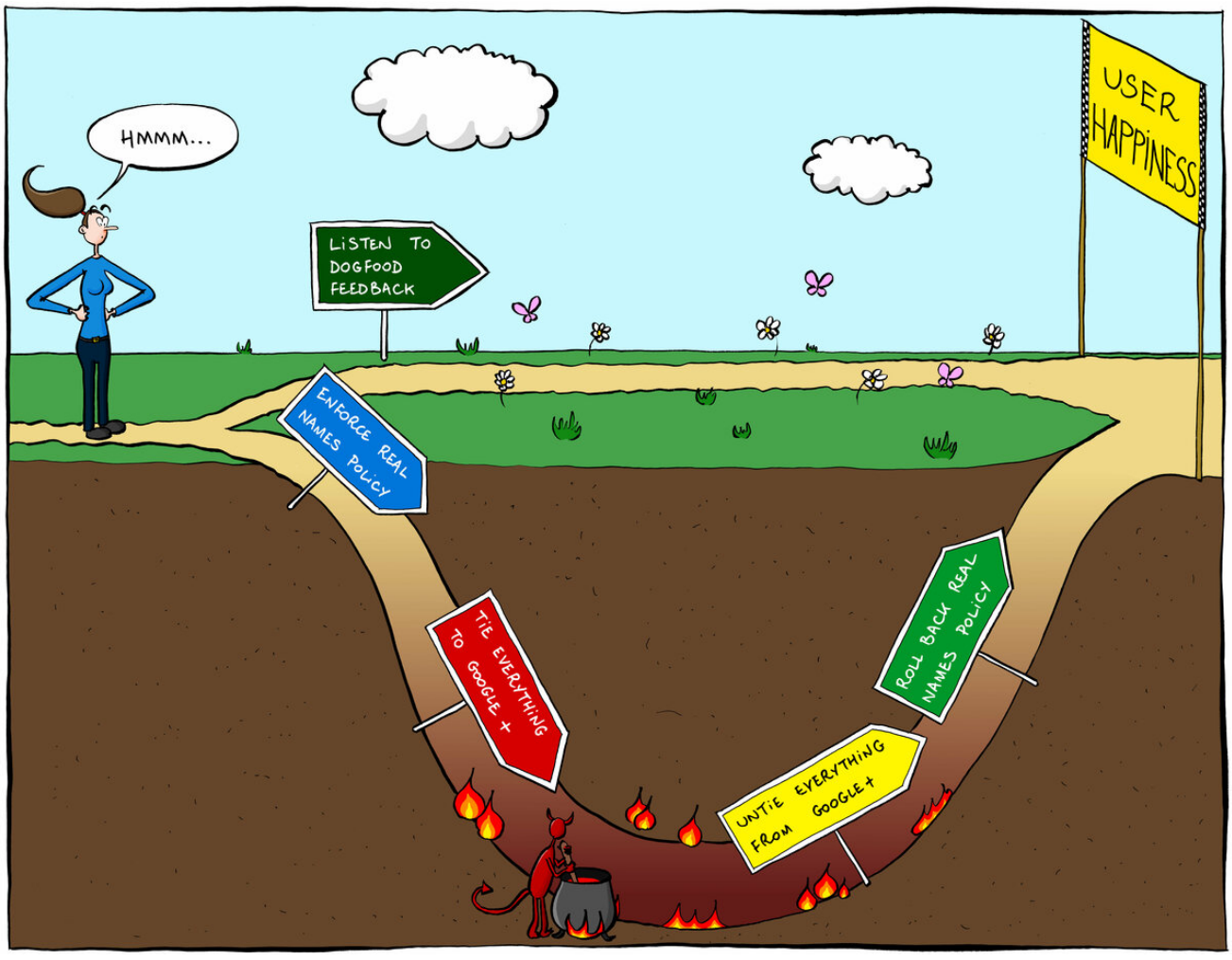
Episode #94 2011.10.13

In short, Google+ didn't seem to embark on the "hockey stick"-shaped growth curves so dear to executives.



Episode #127 2012.03.08

In hindsight, when the company abandoned the Google+ real names policy (in July 2014) and started dismantling the Google+ monolith, it appeared that some Googlers' concerns (about the real names policy, and tying everything to Google+) during the Emerald Sea "dogfood" were somewhat justified.



Episode #193 2014.07.11