Stanford eCorner

Lessons Learned from Breaking Things

Jocelyn Goldfein, *Facebook*

May 22, 2013

Video URL: http://ecorner.stanford.edu/videos/3163/Lessons-Learned-from-Breaking-Things

Director of Engineering Jocelyn Goldfein shares the illustrative story of learning the true meaning of Facebook's motto: move fast and break things. In just her fifth week at the company, Goldfein caused a major problem, but quickly learned that Facebook's culture sincerely believes it is every employee's right to take risks.

Transcript

This was when I was in bootcamp, it was week five and so I was brand new to the company, nobody knew me from Adam and I noticed that my first few weeks in bootcamp I was assigned a bunch of bugs to fix and it turned out like three out of the five were obsolete. Someone had already fixed them or the code had been refactored and just wasn't an issue anymore. And so I realized that there were actually a lot of bugs in the bug database that were kind of - they were Kroft basically. And if you - like, bugs don't sound like a very sexy thing, but if you are determined to ship high-quality software, one of the things you need is information about what is your state, like what is the state of my software. And a bug database can be a faithful representation of the state of the world, if you are really scrupulous about your bug hygiene and about closing out bugs that - at about tracing a bug, so about looking at every bug, and about closing out the ones that aren't relevant, including ones that exist, but you're never going to do anything about because they don't matter. And so at VMware, I built a name for myself by being like this monstrous triager of bugs, like I triaged like 1,000 bugs in my first month at VMware. And at Facebook, I was walking into a much more mature environment that already had a lot of bugs open, and then I obviously had sort of reflection in hindsight that it's just not - what I did at VMware, didn't really work for me, but also wasn't incredibly scalable. And so I wanted to try something new, I wanted to try automating bug triage in a way and writing a script that would essentially if a bug had been untouched for three months. At Facebook, things move so fast, if something has been untouched for three months, it's a pretty good sign that is irrelevant. And so after three months, it would sort of just post an update to the bugs saying, hey, is this thing still relevant or can you close it out? And that would trigger an e-mail to everybody CC'ed on the bug and then if another three months went by, and nobody responded to that or updated the issue, then it would just auto close it.

So I was building this I call it the Task Reaper and I was writing this script and I was testing it and I wasn't careful enough in my testing and I accidentally - just was moving some code around, it was copy paste error, and I moved something out of an If-block, and you're probably guessing where this is going - I accidentally ran the test on live data and it pinged 14,000 bugs, because the first time you run this, there is a lot of untouched bugs. It pinged 14,000 bugs. And that means it generated email to everybody on the CC list of all 14,000 bugs, which meant I basically launched a denial of service attack on our e-mail infrastructure. So this brought the bug system to its knees, but it also brought email to its knees for the whole company, which is like a pretty big deal. I mean, it doesn't take the site down for users, but it means like our sales team can't interact with customers or like the engineers aren't - they can't do code reviews. The email's kind of the lifeblood of the company, even more so in those days. We use Facebook itself more for communication now. And so I was just kind of blankly terrified, like, what I have done and I really expected to be tarred and feathered for this. And there was definitely a vocal company response to what just happened to me, why do I have 200 emails from the Task Reaper. But what was really visible to me, two things were really striking to me.

One was the Exchange team and the bug tools team, like the people I expected to be most mad at me, actually just sort of

rolled up their sleeves, waded in and started fixing the problem. They spun up another process to process the e-mails faster; they threw extra capacity onto the server, like they just got in the trenches and had my back. And I was a stranger, they didn't know me like, they didn't like I had no right to be messing with their systems, they just had it. And then on the flipside the communication from the company, there was no one saying how dare you, you're new, this you didn't - why didn't you ask permission, nobody said that. People were saying what was the point of this? And they were saying could you may be send a digest email instead of an email for each one? And they were like - but no one acted like I didn't have the right to try, like they clearly didn't like the result that their e-mail was down, but they didn't act like I shouldn't try. And it was like a light bulb went on in my head, I realized oh, my God, this is what lets this company still innovate, even when it's already - at that time it had just passed 0.5 billion monthly users and like if any company - like it was just success for them. What is - why did Facebook still have appetite to mess with this thing that is so successful, why do we launch like complete redesigns of the homepage, when the homepage is already the most trafficked piece of property on the web? And I sort of got it in that moment; it was because we are willing to take risks. We are willing to face up to the consequences of failure, if it was in the spirit of trying for something, of trying to innovate. So that was just like an amazing experience for me and I think, it's not that you can come to Facebook and like be incompetent or do things wrong all the time. I mean, I think we - there is a lot of feedback and actions if that's happening.

But we view it as every employee's right to try things to take risks. And we expect you to deal with the consequences of your failures, but we also rally and help you and have your back when you fail and we expect you to rally and have our backs when we fail. So it was kind of magical introduction to the company.