written by Oliver Hall on 26 August 2014 in devops, engineering

incidental scripting

You may or not be aware, but here at MetaBroadcast, we have a pretty structured system for support. One of the pillars around which our support rota is built is the 'Support Handover'. This is a weekly event, where one week's support team hands over to the next, and we summarise what incidents occurred, and if there's any work we can do to improve things. The general idea is that by constantly reviewing what incidents are occurring, we can weed out false positives, and identify re-occurring issues and solve (or at least reduce) them with some targeted engineering time.

passing the torch

For the longest time, Support Handover has strictly been the domain of our very own Tom, but given that he's not exactly twiddling his thumbs for lack of work, I volunteered to take on the duties. After having a quick look through what we tend to run through, I couldn't help noticing a few potential improvements ripe for the making. I'm going to run through the first of these that I've got around to. Hopefully, in a month or two, I'll be able to discuss further improvements...

automaton

Everyone likes scripting tasks, right? As part of our Support Handover, we run through the past week of incidents, referring to the records held in PagerDuty, our alerting system of choice. Now, PagerDuty has a nice enough display for the various incidents, and you can click around just fine. But that's boring, and requires scrolling about on a webpage. What about an API? Well, turns out, PagerDuty has one of those for incidents, which is pretty handy. Tom wrote a quick cURL script to grab the first page of incidents some while back. It simply used some shell magic to sort and list the incidents along with counts, making for a simple but useful display of the most numerous incidents of the past week.

This was certainly handy, but it wasn't exactly the most sophisticated tool known to man. So, I decided to 'improve' it. First things first, I rewrote it in Python. I may not be the world's best Python programmer, but I know enough for it to be my scripting language of choice. Within about half an hour I had something which paginated over all incidents for the past week (not merely the first page), and ordered by count. I decided, for an extra flourish, to separate incidents by escalation policy. This meant we could see at a glance whether the incidents under examination were critical production problems, or mostly minor little bugs cropping up. However, it was at this point still dumping all this out to stdout. Around then I (perhaps stupidly) thought: This'd look grand on a webpage!

enter the jinja

Or, more precisely, Jinja2. Jinja is a templating engine for Python, which makes outputting different formats (like HTML) very simple. Just create a template for your page, then render it, passing in your data, et voila: you have an HTML page containing all your pretty datas.

Now, what I have now isn't the most sophisticated incident report in the world, but it's certainly a start. Hopefully, it should be easily extensible to cover other parts of our Support Handover, as well as things such as incidents in/out of office hours (to see whether people were woken up while on support), and much more.

Have you plumbed into PagerDuty's APIs? What have you managed to build from them? Do let us know!