Amsterdam Python meetup, november 2018

My summary of the 28 November Python meetup at the Byte office. I myself also gave a talk (about cookiecutter) but I obviously haven’t made a summary of that.

Trip Report: C++ Standards Meeting in San Diego, November 2018

A few weeks ago I attended a meeting of the ISO C++ Standards Committee (also known as WG21) in San Diego, California. This was the third committee meeting in 2018; you can find my reports on preceding meetings here (June 2018, Rapperswil) and here (March 2018, Jacksonville), and earlier ones linked from those. These reports, particularly the Rapperswil one, provide useful context for this post.

This meeting broke records (by a significant margin) for both attendance (~180 people) and number of proposals submitted (~270). I think several factors contributed to this. First, the meeting was in California, for the first time in the five years that I’ve been attending meetings, thus making it easier to attend for Bay Area techies who weren’t up for farther travels. Second, we are at the phase of the C++20 cycle where the door is closing for new proposals targeting to C++20, so for people wanting to get features into C++20, it was now or never. Finally, there has been a general trend of growing interest in participation in C++ standardization, and thus attendance has been rising even independently of other factors.

This meeting was heavily focused on C++20. As discussed in the committee’s standardization schedule document, this was the last meeting to hear new proposals targeting C++20, and the last meeting for language features with significant library impact to gain design approval. A secondary focus was on in-flight Technical Specifications, such as Library Fundamentals v3.
To accommodate the unprecedented volume of new proposals, there has also been a procedural change at this meeting. Two new subgroups were formed: Evolution Incubator (?EWGI?) and Library Evolution Incubator (?LEWGI?), which would look at new proposals for language and library changes (respectively) before forwarding them to the Evolution or Library Evolution Working Groups (EWG and LEWG). The main purpose of the incubators is to reduce the workload on the main Evolution groups by pre-filtering proposals that need additional work before being productively reviewed by those groups. A secondary benefit was to allow the attendees to be spread out across more groups, as otherwise EWG and LEWG would have likely exceeded their room capacities.

- **The Future of OpenJDK at Red Hat** [4]

  With the release of Java 11, the transition of Java into an OpenJDK-first project is finally complete. The days of most Java installations using the proprietary OracleJDK binaries are at an end. This increased focus on Open and Free Java naturally brings the contributions of companies other than Oracle into greater prominence. InfoQ recently spoke with Rich Sharples, Senior Director of Product Management for Middleware at Red Hat, to discuss OpenJDK and Red Hat's involvement with it.

- **PyBites: 3 Cool Things You Can do With the dateutil Module** [5]

- **Subtleties of Python** [6]

  A good software engineer understands how crucial attention to detail is; minute details, if overlooked, can make a world of difference between a working unit and a disaster. That's why writing clean code matters a lot?and clean code isn't just about neat indentation and formatting; it's about paying attention to those details that can affect production.

  In this article, you'll see a couple of short cases of problematic code in Python and how they can be improved. Please note that these are just examples and in no way must you interpret them to universally apply for real-world problems.

- **A Tale of Two Commits** [7]

  I've discussed and linked to articles about the advantages of splitting patches into small pieces to the point that I don't feel the need to reiterate it here. This is a common approach at Mozilla, especially (but not just) in Firefox engineering, something the Engineering Workflow
group is always keeping in mind when planning changes and improvements to tools and processes.

Many Mozilla engineers have a particular approach to working with small diffs, something, I?ve realized over time, that seems to be pretty uncommon in the industry: the stacking of commits together in a logical series that solves a particular problem or implements a specific feature. These commits are generally authored, reviewed, updated, and even landed as a set. They tell a complete story; indeed, you could view this process as similar to writing a novel: the book is written, edited, and published as a complete unit.

- **Common architectural elements for modern integration architectures**[8]

  In Part 1 of this series, we explored a use case around integration being the key to transforming your customer experience.

  I laid out how I?ve approached the use case and how I?ve used successful customer portfolio solutions as the basis for researching a generic architectural blueprint. The only thing left to cover was the order in which you?ll be led through the blueprint details.

**Development**

**Source URL:** [http://www.tuxmachines.org/node/117999](http://www.tuxmachines.org/node/117999)

**Links:**