Today marks Rust's sixth birthday since it went 1.0 in 2015. A lot has changed since then and especially over the past year, and Rust was no different. In 2020, there was no foundation yet, no const generics, and a lot organisations were still wondering whether Rust was production ready.

In the midst of the COVID-19 pandemic, hundreds of Rust's global distributed set of team members and volunteers shipped over nine new stable releases of Rust, in addition to various bugfix releases. Today, "Rust in production" isn't a question, but a statement. The newly founded Rust foundation has several members who value using Rust in production enough to help continue to support and contribute to its open development ecosystem.

- Six Years of Rust [2]

The Bourne shell lets you set variables in if expressions [3]
This isn't the first time I've seen SC2181 and as always, I rolled my eyes at it because it seemed obviously wrong, because of course you can't merge these two lines together. But this time I went off to the Shellcheck repository to report it as an issue, and before I reported it as an issue I did a search, and that was when I discovered that Shellcheck was not wrong.

To my surprise, the Bourne shell allows you to perform command substitutions and capture the output in variables in if expressions. You really can write my two lines in a single one as: [...]

*What Python 3.3 did to improve exception handling in your code | Opensource.com*[4]

This is the fourth in a series of articles about features that first appeared in a version of Python 3.x. Python 3.3 was first released in 2012, and even though it has been out for a long time, many of the features it introduced are underused and pretty cool. Here are three of them.

*Macros in the Shell: Integrating That Spreadsheet From Finance Into a Data Pipeline*[5]

While I no longer use it regularly for the purposes of analysis, I will always have a soft spot in my heart for excel1. Furthermore, using a ?correct? set of data science tools often requires a bridge2. Integrating a rigorous component into a messy spreadsheet based pipeline can be an initial step towards the pipeline or team or organization starting on a path of continuous improvement in their processes3. Also, spreadsheets are foundational to many (probably most) BizOps teams and therefore are sometimes unavoidable?

In this post I will walk through a short example and some considerations for when you might decide (perhaps against your preferences) to integrate your work with extant spreadsheets or shadow ?pipelines? within your organization.

*Extend the language*[6]

It?s almost eerie to me how a programming problem can seem completely unsolvable and then you extend the language a bit and suddenly it?s easy.

It?s happened to me most often with Lisps of course, but it can happen with pretty much any language. I remember a few Java occasions, personally, where I put in a functor framework and previously challenging problems suddenly became easy.

*Scraping the latest EU VAT rates for e-services from the European Commission?s web site with Node.js*[7]
So, you now know how to verify an EU VAT number with Node.js.

* Using the European Commission EU VAT Number validation API with Node.js [8]

You may know of the VIES site where you can manually validate EU VAT numbers but did you know that the European Commission also has an API for programmatically doing this?1

**Source URL:** [http://www.tuxmachines.org/node/151219](http://www.tuxmachines.org/node/151219)

**Links:**
[3] https://utcc.utoronto.ca/~cks/space/blog/programming/BourneIfCanSetVars