Guido van Rossum

On his recent jaunt to New York, **Nick Veitch** caught up with Guido van Rossum, the creator of the Python programming language.

Linux Format: *While people may be familiar with the Python language, perhaps you could tell us a little about how you came to create it.*
Guido van Rossum: Well, in about 1990, I started creating Python. I was looking for a language that was much quicker to write in than something like C, but which was more like a programming language than the Bourne shell.

Previously I had worked on a language called ABC. I worked for a government funded research lab in Amsterdam, the CWI (Centre for Mathematics and Computer Science). I got to know a lot about language design as well as language implementation through that job. Later when I was working on different things, I realised there was a whole class of applications where doing the development in C++ or C wasn't an effective use of time. These were simple backup utilities, sysadmin tools – maybe even things I would only run once just to test something.

I love the shell when it's a ten line script, but when you get to thousands of lines it becomes unreadable and really unmaintainable. I wanted something that still had the interactive feel of the shell, but which you could use to write a proper program, with data structures, lists and so on.

There weren't a lot of alternatives at the time. Tcl and Perl began about the same time. I had heard of Perl, but it only ran on Unix systems then, and a quick look at the source made it clear that it wasn't going to be easily portable. One thing I wanted was something that was extensible. Had I known of Tcl, maybe Python would never have been created, or at least it may have influenced Python in different ways. In the end I had mostly positive influences from other languages. I realised that object orientation was the perfect solution to my desire to make the language extensible.

LXF: *Did you get much feedback?*

GvR: That was what kept me going. At first I just released it within the institute. There were a number of accomplished hackers there who loved it, which encouraged me to make a full release. In February 1991 I put all the source in a tar file, uuencoded it in 20 pieces and posted it to a usenet group. The response was amazing. I started receiving lots of comments, patches, bug fixes and wish-lists right away.

LXF: *Don't you find that unusual? You sort of expect that with an application that people can just go off and use, but a language requires more investment in terms of time from the people who are using it.*
GvR: You would think so. I think the initial users were people with a specific interest in languages. But Python is very easy to learn, and there were already quite a few applications for it, as we had been using it internally for a year. It came, in a sense, with batteries included – a large library of useful stuff.

The other thing is that at that time, a lot of people were looking for a language like Python, so it was also a case of being in the right place at the right time.

LXF: *If we can talk about ease of use for a moment, that's something that definitely comes across with Python compared to a language such as Perl which is easy when you understand it, but is a little daunting initially.*
GvR: Perl probably has a steeper learning curve, and it encourages the kind of tinkering that leads to complex and unmaintainable programs. The culture of the "one-liner" is very strong in Perl, and almost completely absent in Python. You can do the same thing in two lines, and when you look at them two weeks or even two years later, they will still be easy to understand.

LXF: *Is readability something that's very important to you?*
GvR: It's something that sometimes when I might get a bit overenthusiastic about adding new features, the users remind me that what they like about Python is that it's a small language, there aren't too many constructs that need to be learned either to use it, or understand other people's code. That's something which was probably inherited from ABC which was designed primarily to be an educational language.

LXF: *That seems to be part of the philosophy as well, that everybody can have a go with Python.*
GvR: You may be referring to Computer Programming for everybody, which was a big initiative about a year and a half ago. I still think that Python will make some inroads there, though I'm not actively involved with it myself, mainly because of a lack of funding for the project. Other people have taken it up though. High schools are taking up Python and some colleges. Some find that it is very effective to teach Python first, and then Java. They can concentrate on learning the concepts – algorithms, variables, code structure – and they don't have to worry so much about getting the code to compile correctly.

LXF: *What goals do you have for Python then, moving forwards?*
GvR: I don't have that many personal goals. I think

"wow this is just happening, I'm in the middle of a crowd of people who are very excited about this stuff, let's see if we can give them what they want". At the same time I think that Python is very well positioned to be the successor to Perl. It works very well with Java, we have the Java-Python bridge called Jython. The language can always be improved; we can keep polishing the diamond. We now have Unicode and XML support, and we'll see more and more large applications written in Python.

According to Tim O'Reilly, there are half a million Python users in the world. I don't know how we would verify that, but he sells books and he knows how many books he sells.

LXF: *In many ways Python and Perl are competitors – do you think there is a place for both?*

GvR: Larry Wall is a good friend of mine. He has a different philosophy about what makes a useable language and to be honest the jury is still out. He had the advantage early on, when Perl was the scripting language of choice for web applications, but that is changing now. Digital Creations, my new employer, has *Zope* which is a content management system as well as an application server, and all of that is written in Python. It is extensible through Python and all the system code is written in the language. So Python is useable to write large systems, more so than Perl. You can write such things in Perl, but the maintenance of the software takes over from developing it.

Python is more manageable for large projects. The way that it works tends to get people to write readable code, so less of the idiosyncrasies of the individual hacker come through. Perhaps the languages just attract a different type of person. Perl possibly attracts people who have that hackers pride, who get excited when they write a one-liner that nobody else thought could be done in one line, while the Python people are more excited when they create a piece of software and other people can understand it and make changes to it that work. It's more co-operative.

LXF: *It certainly seems that the more flamboyant hackers would tend towards Perl, and believe that their work is more of an art form than something which should be primarily functional.*

GvR: There is nothing wrong with seeing code as an art form. Maybe eventually we'll figure out how to produce software in an industrial fashion where you can predict maybe how long it will take and how well it runs, but right now the most productive programmers have something of the artist in them. At the same time, if you can get a bunch of people like that to work together, it's much better. At the other end of the spectrum if you want to write a simple program that scans a log file or something, Perl is a little faster, but Python is catching up here too. We have a much faster regular expression engine in 2.0, and in 2.1 we hope to have faster line input as well as more new language features.

LXF: *So, how are you finding your new job at Digital Creations?*

GvR: I love it. The funny thing is that Paul Everitt has tried to hire me about five times. The fifth time I was ready, and it was like coming home. Digital Creations has exactly the right corporate culture for Python and for me. They have been very good at supporting the Python team, and I'm having fun with *Zope* and learning about that too. *Zope* will become better because the Python link has been formed, and Python will become better because of Digital Creations support.

LXF: *It seems to be a common theme to a lot of open source projects now.*

GvR: It is a very classic open source story I would say. The difference may be that Digital Creations is a completely open source company. *Zope* is their only product, and that is open source, and I don't think they have ever regretted it.

"According to Tim O'Reilly, there are half a million Python users in the world."

LXF: *It also seems that many of the top open source developers find their way to the US.*

GvR: Whether they are from Europe or the US, I think there will always be a trend for people who start out as for example, a student developing an open source package and gathering a crowd of followers. Eventually they need a job. If you have developed a good open source package, that is a very big plus point on your resumé. I think US companies are currently picking up on that more than in Europe. The Python community, though, is fairly evenly split. I just found out today that there is a Korean Python users meeting running, and 400 people have signed up already. Python is definitely finding its way around the world. There are people in Sweden, Finland, Germany, all over the place contributing to the project.

LXF: *Will you ever get bored of Python?*

GvR: Probably as soon as Python gets bored of me. When the "benevolent dictator for life" joke wears thin it may be time to go.

LXF: *In other projects you often find that the main developers find the initial stages exciting, but once something becomes established there isn't so much fun in it for them anymore.*

GvR: The Python language is a framework and has so many applications, there is always more work to be done. It isn't like at some point you have developed all the options. I'm learning a lot about how Python is being used, and about how we can support those applications better. There are plenty of things to discover.

We are working on the 2.1 release, but we also have this mythical thing called Python 3000 in the back of our heads. It may never happen but it is to focus our energy on far out ideas for changes or improvements. Especially for things like when we discover a design bug – we can't just say we'll fix that, because it will change the language and affect all the code designed to run in it. There is a possibility that at some stage there may be a new version of Python that would be incompatible to previous versions to fix such things. For example, one divided by three currently returns zero. That's exactly what the typical C or Java programmer expects, but it isn't what a FORTRAN programmer, or someone who doesn't know programming expects. These people might expect it to return one third. Other stuff, we think about how we would design it to work in Python 3000, and then try to figure out how we could evolve the existing Python to work towards that.

For that reason we are including a warning framework in 2.1 so the language can issue warnings about outdated features. We currently have three different regular expression implementations that we carry around. One that had a design problem and one that is very close to the way Perl handles regular expressions – for which we imported foreign code which became hard to maintain and implement – for example when it ran out of memory it would just core dump, and we want Python to work nicely. We now have a third generation engine which was contributed by a very talented Swedish programmer, and it's much faster too. Now we want to issue warnings to people using the old modules. So there's always something to do... LXF

MORE INFO

www.python.org – main Python website, for latest releases, news, events listings and more
www.digicool.com – Digital Creations main website.
www.zope.org – for information on Zope .
www.vex.net/parnassus – Vaults of Parnassus, a huge python resources site.
www.python9.org – Ninth International Python Conference website.
www.python.org/~guido/ – Guido van Rossum's personal website.