Programming/Development: Minicoin, GNU Gengetop and Python

By *Roy Schestowitz*
Created *04/06/2019 - 3:48pm*
Submitted by Roy Schestowitz on Tuesday 4th of June 2019 03:48:36 PM Filed under Advertisement [1]

- **Building and testing on multiple platforms ? introducing minicoin [2]**

  While working with large-scale (thousands of hosts), distributed (globally) systems, one of my favourite, albeit somewhat gruesome, metaphors was that of ?servers as cattle? vs ?servers as pets?. Pet-servers are those we groom manually, we keep them alive, and we give them nice names by which to remember and call (ie ssh into) them. However, once you are dealing with hundreds of machines, manually managing their configuration is no longer an option. And once you have thousands of machines, something will break all the time, and you need to be able to provision new machines quickly, and automatically, without having to manually follow a list of complicated instructions.

  When working with such systems, we use configuration management systems such as CFEngine, Chef, Puppet, or Ansible, to automate the provisioning and configuration of machines. When working in the cloud, the entire machine definition becomes ?infrastructure as code?. With these tools, servers become cattle which ? so the rather unvegetarian idea ? is simply ?taken behind the barn and shot? when it doesn?t behave like it should. We can simply bring a new machine, or an entire environment, up by running the code that defines it. We can use the same code to bring production, development, and testing environments up, and we can look at the code to see exactly what the differences between those environments are. The tooling in this space is fairly complex, but even so there is little focus on developers writing native code targeting multiple platforms.

  For us as developers, the machine we write our code on is most likely a pet. Our primary workstation dying is the stuff for nightmares, and setting up a new machine will probably keep us busy for many days. But this amount of love and care is perhaps not required for those machines that we only need for checking whether our code builds and runs correctly. We don?t need our test machines to be around for a long time, and we want to know exactly how

they are set up so that we can compare things. Applying the concepts from cloud computing and systems engineering to this problem lead me (back) to Vagrant, which is a popular tool to manage virtual machines locally and to share development environments.

- **GNU Gengetopt - News: 2.23 released** [3]

  New version (2.23) was released. Main changes were in build system, so please report any issues you notice.

- **Abolishing SyntaxError: invalid syntax ...** [4]

  Do you remember when you first started programming (possibly with Python) and encountered an error message that completely baffled you? For some reason, perhaps because you were required to complete a formal course or because you were naturally persistent, you didn't let such messages discourage you entirely and you persevered. And now, whenever you see such cryptic error messages, you can almost immediately decipher them and figure out what causes them and fix the problem.

- **Sending email with EZGmail and Python** [5]

- **Creating and Importing Modules in Python** [6]

Source URL: http://www.tuxmachines.org/node/124529

Links:
[1] http://www.tuxmachines.org/taxonomy/term/140
[2] https://blog.qt.io/blog/2019/06/04/introducing-minicoin/
[3] http://savannah.gnu.org/forum/forum.php?forum_id=9442
[4] https://aroberge.blogspot.com/2019/06/abolishing-syntaxerror-invalid-syntax.html
[5] http://www.blog.pythonlibrary.org/2019/06/04/sending-email-with-ezgmail-and-python/
[6] https://stackabuse.com/creating-and-importing-modules-in-python/