Python on the toilet: WSGI

Patrice Neff
November 13, 2009

WSGI is a Python standard for how web servers can interact with web frameworks. It's one of my favorite standards: it's simple yet very powerful.

To write a WSGI web application you only need to create one function.

```
def my_app(environ, start_response):
    start_response('200 OK', [('Content-Type', 'text/plain')])
    return repr(environ)
```

The function receives a environ argument - a dictionary with all the environment variables. The `start_response` function can be called with the status line and a list of headers as soon as you're ready to send output.

New Python releases contain the library wsgiref which can be used to get started quickly with a simple web server (that should not be used in production).

```
from wsgiref.simple_server import make_server
httpd = make_server('', 8000, my_app)
httpd.serve_forever()
```

Save these two snippets in a file e.g. `myapp.py` and execute it with Python. This will serve the sample application on port 8000.

You don't usually want to write your own application directly on top of WSGI. But most frameworks now implement WSGI which has led to better interoperability. If you still want to use WSGI directly, there are a ton of good tools such as WebOb, Werkzeug or Paste. I used WebOb to easily build a REST service framework called WsgiService[1].

For more information I recommend the specification for WSGI [2] which is a good read.

This post is part of the Python on the toilet[3] series.

[1] http://github.com/pneff/wsgiservice
[2] PEP 333, http://www.python.org/dev/peps/pep-0333/
[3] http://weblog.patrice.ch/2009/03/05/pot-introduction.html