stefan scherfke

Getting started with devpi

Posted on Thursday, November 09, 2017

Recently, I wanted to (re)evaluate devpi for use in our company. I have already worked with it some years ago but–by now–forgot what exactly it can do and how to set it up.

However, the marketing on the landing page of the docs and on GitHub was not very convincing and I nearly ended up working with another product.

A short Twitter discussion later, I decided to give devpi a try and write down my findings. Maybe they can contribute to improving devpi's docs and demonstrate how easy it is to use.

This is what I was trying to achieve with devpi:

- Mirror and cache PyPI
- Extend PyPI with an index for internal stable packages
- Extend the stable index with a staging index for experimental new features
- No free registration for users
- Only a single, authorized user (our GitLab CI)
- Use standard tools (pip, twine, ...) as much as possible

Installation

Devpi consists of a server, a command line client and a web front-end. The meta package `devpi` installs them all:

```
$ mkvirtualenv devpi
$ pip install devpi
...
Installing collected packages: ...
Successfully installed ...
```

Setting up the server

Devpi uses SQLite to store its data, so we don't need to setup a database (you can use PostgreSQL though, if you want).

When you start the `devpi-server` for the first time, you need to pass the `--init` option. You an also specify where it should store its data (the default is `~/.devpi`):

```
$ devpi-server --serverdir=/tmp/devpi --init
...
```

If you don't use the standard location, you must set the `--serverdir` option every time you start the server.

User management

We can now set a password for the root user and allow only root to create new users:

```
$ devpi-server --serverdir=/tmp/devpi --passwd root
enter password for root:
repeat password for root:

$ devpi-server --serverdir=/tmp/devpi --restrict-modify=root --start
...
starting background devpi-server at http://localhost:3141
...
```

Like the `--serverdir` option, you must always pass `--restrict-modify=root` when you start the server.

Once the server is running, we can use the devpi client `devpi` to create an additional user named *packages*. Prior to that, tell the devpi client on which server we want to operate with the following commands:

```
$ devpi use http://localhost:3141
...

$ devpi login root
password for user root:
logged in 'root', credentials valid for 10.00 hours

$ devpi user -c packages email=packaging@company.com password=packages
user created: packages

$ devpi user -l
packages
root
```

Package indexes

By default, devpi creates an index called *root/pypi*. It serves as a proxy and cache for PyPI and you can't upload your own packages to it.

However, devpi supports index inheritance: We can create our *stable* index in the *packages* namespace and set *root/pypi* as base. If we query *packages/stable*, devpi first searches this index and than falls back to *root/pypi* if it can't find the package on the first index.

```
$ devpi index -c packages/stable bases=root/pypi volatile=False
http://localhost:3141/packages/stable:
  type=stage
  bases=root/pypi
  volatile=False
  acl_upload=packages
  mirror_whitelist=
  pypi_whitelist=
```

Similarly, our *staging* index can inherit *stable*. Devpi will than search *packages/staging*, *packages/stable* and finally *root/pypi* for packages:

```
$ devpi index -c company/staging bases=company/stable volatile=True
http://localhost:3141/company/staging:
  type=stage
  bases=company/stable
  volatile=True
  acl_upload=company
  mirror_whitelist=
  pypi_whitelist=
```

The `volatile=True` option lets us perform destructive actions on the index (like overriding or deleting packages).

Install packages

Let's use our devpi to load a public package from PyPI:

```
$ pip install -i http://localhost:3141/company/stable click
Collecting click
  Downloading http://localhost:3141/root/pypi/+f/5e7/a4e296b3212da/click-6.7-py2.p
Installing collected packages: click
Successfully installed click-6.7
```

We can also configure pip to use our devpi as default index:

```
[global]
index-url = http://localhost:3141/company/stable
```

Upload packages

You can build and upload packages with your usual workflow. Just add devpi to your `~/.pypirc` (Do not store passwords in there!):

```
[distutils]
index-servers =
    devpi-stable
    devpi-staging

[devpi-stable]
repository = http://localhost:3141/packages/stable/
username = packages

[devpi-staging]
repository = http://localhost:3141/packages/staging/
username = packages
```

Now we can build some dists:

```
$ cd ~/Projects/simpy
$ python setup.py sdist bdist_wheel
...
```

And upload them:

```
$ pip install twine
...
Installing collected packages: ...
Successfully installed ...

$ twine upload -r devpi-stable dist/*
Uploading distributions to http://localhost:3141/packages/stable/
Enter your password:
Uploading simpy-3.0.10-py2.py3-none-any.whl
Uploading simpy-3.0.10.tar.gz
```

Congratulation!

You can now install your own packages from your own packages index:

```
$ pip install simpy
Collecting simpy
  Downloading simpy-3.0.10-py2.py3-none-any.whl
Installing collected packages: simpy
Successfully installed simpy-3.0.10
```

Devpi’s docs may appear a bit confusing and look a little demure, but devpi itself is actually really easy to setup and use – and powerful at the same time!

Tags: devpi , packaging , pypi , python