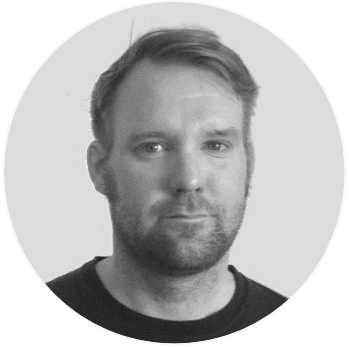
Setting up your Project with Virtual Environments



Reindert-Jan Ekker

@rjekker http://nl.linkedin.com/in/rjekker

Overview

Why virtual environments

Create and explore a virtual environment

Using virtual environments with projects

Project dependencies

Problems with System-wide Installs

Multiple projects with conflicting dependencies

Conflicts with system dependencies

Multi-user systems

Testing code against different python and library versions

Virtual Environments

Isolated context for installing packages

Always work inside a virtual environment

- No global installs anymore
- Create a virtual env. for every project

Isolate project dependencies

- No more conflicts with other projects

Demo

Starting a project

- Create a virtual environment
- Explore the virtual environment

Demo

Working inside a virtual environment

- Activating the environment
- Running python and pip
- Installing a package
- Deactivate

Creating a Virtual Environment

```
virtualenv myenv
# Need to install virtualenv package first


# Included with python >= 3.3: venv
python -m venv myvenv


# Deprecated
pyvenv myvenv
```

Activating a Virtual Environment

```
# Run the activate script inside the virtual environment

# On linux/Mac OS:

reindert@pc:~/dev/$ . myvenv/bin/activate


# On Windows:

C:\Users\reindert\dev> myvenv\Scripts\activate.bat
```

After Activation

```
# The prompt will show the name of the active venv
(myvenv) reindert@pc:~/dev/$
# You are now ready to install packages
# And work on your project


# When you're done, leave the virtual environment
(myvenv) reindert@pc:~/dev/$ deactivate
# To remove a virtual env., simply delete the directory
```

In an Active Virtual Environment

python **refers to interpreter in venv**

- Same for pip

Packages are installed inside the venv

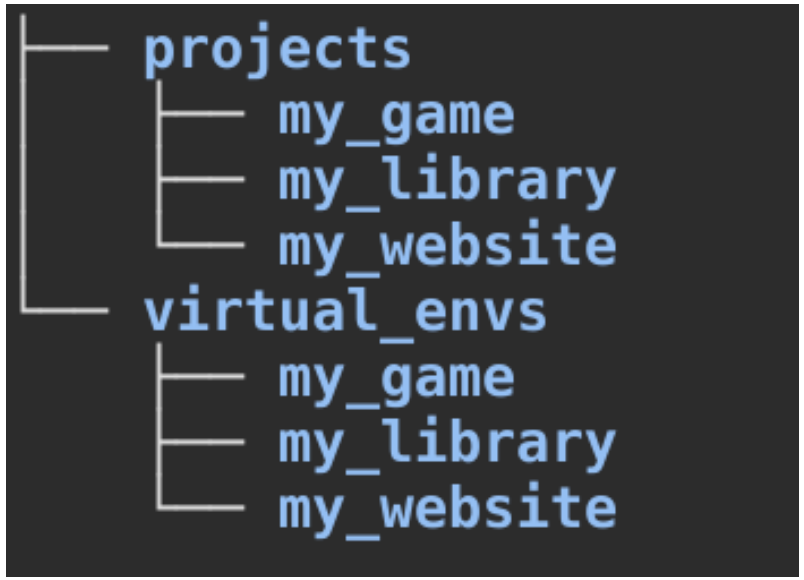
- Don't interfere with other projects

Demo

Requirements

- Syncing dependencies with your team

Projects and Virtualenvs



Projects

- Contain source code
- Are under version control

Virtual environments

- Contain packages, tools, python, etc.
- Keep them separate from your projects
- Usually: 1 venv per project
- Can have multiple venvs per project
- Or a single venv for multiple projects

requirements.txt

```
# After installing packages
python -m pip freeze > requirements.txt
# Resulting file (put this in version control):
certifi==2018.11.29
chardet==3.0.4
idna==2.8  (..etc..)

# To install all dependencies
python -m pip install -r requirements.txt
```

Specifying Versions

```
docopt == 0.6.1                 # Must be version 0.6.1

keyring >= 4.1.1                # Minimum version 4.1.1

coverage != 3.5                 # Anything except version 3.5
```

Versions and Pip

```
python -m pip install flask==0.9
python -m pip install 'Django<2.0'     # Mind the quotes!

# Upgrade to latest version
python -m pip install -U flask


# Upgrade pip itself
# Take care not to overwrite system pip
python -m pip install -U pip
```

Demo

A real-world github project

Overview

Why virtual environments

Create and explore a virtual environment

Using virtual environments with projects

Project dependencies