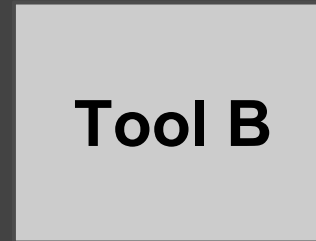# Virtual environments

Why we use them, how to start with virtualenv and how to go further with pyenv.

Example
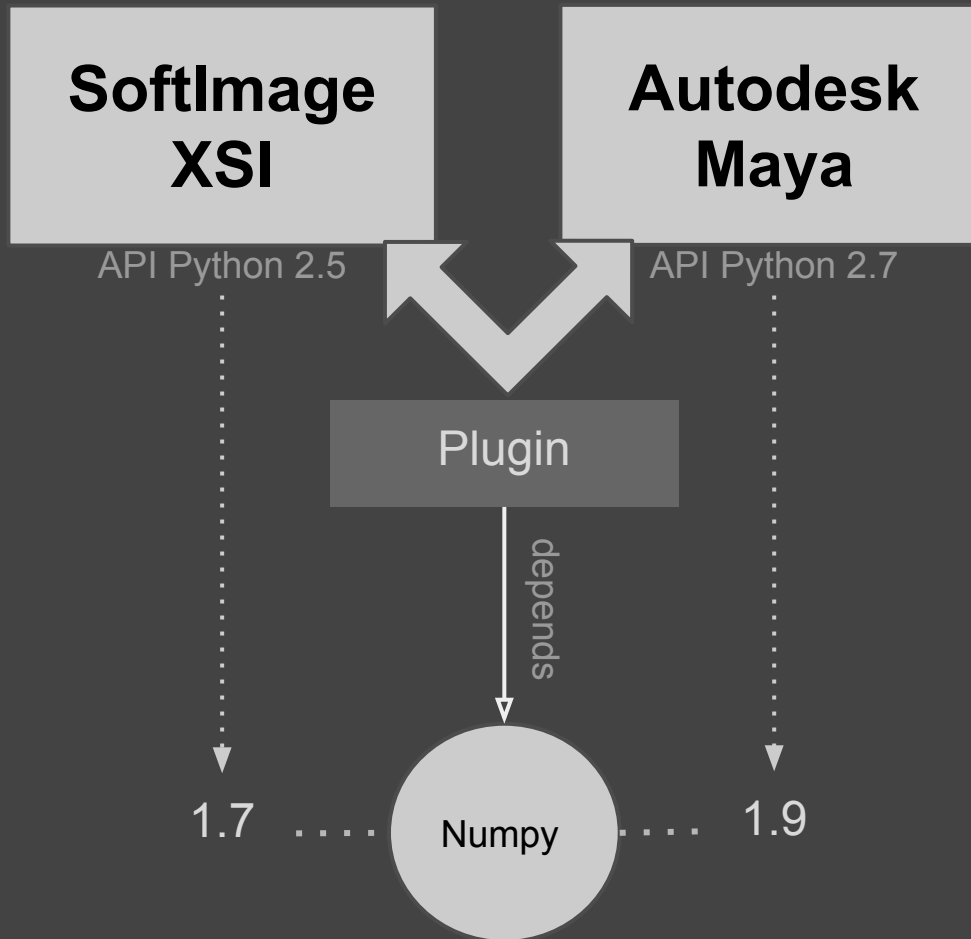
Tool A

Depends LibC 1.0.1

Tool B

Depends LibC 2.1.0

- 2 tools using different versions of the same library (or even two differents python versions)
- Between the 2 versions of the library, API changes might have occurred, and new features been introduced
- Tool A is stable, port the code to use newer version of the lib may not be the more efficient

Concrete example

SoftImage XSI

Autodesk Maya

API Python 2.5

API Python 2.7

Plugin

depends

1.7 · · · · Numpy · · · · 1.9

Why using virtual environments ?

- Isolate dependencies required by different projects

- Easily retrieve those dependencies

- Keep your main installation clean

- Create sandbox

- Switch easily from one environment to another

Virtualenv

Virtualenv is a tool to help you create isolated Python environments.

Installation :
```
pip install virtualenv
```

Create a new environment :
```
virtualenv my_env
```

Activate this new environment :
```
. my_env/bin/activate
```

Deactivate it :
```
deactivate
```

Specify a python exe on creation :
```
virtualenv -p /usr/bin/python26 my_env_2.6
```

More infos on virtualenv.pypa.io.

Pyenv

Pyenv is a tool to help you handle different versions of pythons :
- download and install
- switch the python version of your system, of a current shell or of a workspace

How does it work :
- based on Shell scripts,
- intercepts Python commands using shim executables injected into your PATH to pass your commands along to the correct Python installation.

Installation :
- Use homebrew on macOS,
- pyenv-installer,
- or clone the repository and set-up your env.

Pyenv is supported on MacOSX and Linux BUT not on Windows.

Pyenv : install a python

List available for download python versions :
```
pyenv install --list
```

Before installing a new version, you can set some configure options.
For example :
```
export CONFIGURE_OPTS="--enable-static --enable-unicode=ucs4 --with-pic"
```

Download and install a python version :
```
pyenv install <version>
```

```
pyenv rehash
```

Uninstall a python version :
```
pyenv uninstall <version>
```

Pyenv : check out versions

List installed python versions :
`pyenv versions`

Display current python version :
`pyenv version`

Locate current python :
`pyenv which python`

Pyenv : switch version

Change global python version :
`pyenv global <version>`

Set the python version of a workspace :
`pyenv local <version>`

Set the python version of the current shell :
`pyenv shell <version>`

More infos on github.com/yyuu/pyenv.

Pyenv-virtualenv

Pyenv-virtualenv is a pyenv plugin that allows you to create virtualenv via pyenv.

Installation:
```
git clone https://github.com/yyuu/pyenv-virtualenv.git ~/.pyenv/plugins/pyenv-virtualenv
```

Create a virtualenv from the current version:
```
pyenv virtualenv my_env
```

Or from a specific version:
```
pyenv virtualenv <version> my_env
```

List virtual envs:
```
pyenv virtualenvs
```

Pyenv-virtualenv

Activate a virtualenv :
`pyenv activate my_env`

Deactivate a virtualenv :
`pyenv deactivate`

Uninstall a virtual env :
`pyenv uninstall my_env`

More infos on github.com/yyuu/pyenv-virtualenv.

Questions ?

Presentation media on :
https://github.com/**mfe**/Prez/tree/master/virtualenv