Python

macOS, like Linux, ships with Python already installed. But you don't want to mess with the system Python (some system tools rely on it, etc.), so we'll install our own version(s). There are two ways to install Python, (1) Homebrew and (2) Pyenv. If you plan to use multiple versions of Python (e.g. 2, 3, and anaconda) then you should use pyenv.

Installation

Homebrew method

Python 3 is the default version when installing with Homebrew, so if you want to install Python 2.7 you'll have to be explicit about it.

Python 3

```
$ brew install python
```

Python 2.7

```
$ brew install python@2
```

Installing Python also installs pip (and its dependency Setuptools), which is the package manager for Python. Let's upgrade them both:

```
$ pip install --upgrade setuptools
$ pip install --upgrade pip
```

Executable scripts from Python packages you install will be put in `/usr/local/share/python`, make sure it's on your `PATH`.

Pyenv method

`pyenv` is a Python version manager that can manage and install different versions of Python. Works very much like `rbenv` for Ruby. First, we must install `pyenv` using Homebrew:

```
$ brew install pyenv
```

To upgrade `pyenv` in the future, use `upgrade` instead of `install`. After installing, add `pyenv init` to your shell to enable shims and autocompletion (use `.zshrc` if you're using `zsh`).

```
$ echo 'eval "$(pyenv init -)"' >> ~/.bash_profile
```

Restart your shell so the path changes take effect.

```
$ exec $SHELL
```

You can now begin using `pyenv`. To list the all available versions of Python, including Anaconda, Jython, PyPy and Stackless, use:

```
$ pyenv install --list
```

Then install the desired versions:

```
$ pyenv install 2.7.12
$ pyenv install 3.5.2
```

Use the `global` command to set global version(s) of Python to be used in all shells. For example, if you prefer 2.7.12 over 3.5.2:

```
$ pyenv global 2.7.12 3.5.2
$ pyenv rehash
```

The leading version takes priority. All installed Python versions can be located in `~/.pyenv/versions`. Alternatively, you can run:

```
$ pyenv versions
  system (set by /Users/your_account/.pyenv/version)
* 2.7.12
* 3.5.2
```

This shows an asterisk `*` next to the currently active version.

Application-specific Python version

The `local` command will set local application-specific Python version(s) by writing the version name to a `.python-version` file in the current directory. This version overrides the global version. For example, to install anaconda3-4.1.1 in `path/to/directory`:

```
$ pyenv install anaconda3-4.1.1
$ cd path/to/directory
$ pyenv local anaconda3-4.1.1
$ pyenv rehash
$ pyenv versions
  system
  2.7.12
  3.5.2
* anaconda3-4.1.1 (set by /Users/your_account/path/to/directory/.python-version)
```