Python Virtual Environments

By Aroonav Mishra (ID: B113008)

virtualenv

Features

- Fully isolated environments
- Uses its own site-packages directory
- Ignore the default system's site-packages directory.

site-packages

- Third-party installed modules go here.
- e.g. /usr/lib/python2.6/site-packages/

(Standard library in /usr/lib/python2.6)

How do site-packages work?

- Python imports site.py at startup.
- e.g. usr/lib/python2.6/site.py

  Or Lib/site.py in the Python source.

site.py

```
# Prefixes for site-packages;
PREFIXES = [sys.prefix, sys.exec_prefix]

def getsitepackages():
    """Returns a list containing all global site-packages
    directories (and possibly site-python).

    For each directory present in the global ``PREFIXES``,
    this function will find its `site-packages` subdirectory
    depending on the system environment, and will return a
    list of full paths.
    """
    sitepackages = []
    seen = set()

    for prefix in PREFIXES:
        if not prefix or prefix in seen:
            continue
        seen.add(prefix)

        if sys.platform in ('os2emx', 'riscos'):
            sitepackages.append(os.path.join(prefix, "Lib", "site-packages"))
        elif os.sep == '/':
            sitepackages.append(os.path.join(prefix, "lib",
                                             "python" + sys.version[:3],
                                             "site-packages"))
            sitepackages.append(os.path.join(prefix, "lib", "site-python"))
        else:
            sitepackages.append(prefix)
            sitepackages.append(os.path.join(prefix, "lib", "site-packages"))
        if sys.platform == "darwin":
            # for framework builds *only* we add the standard Apple
            # locations.
```

Where does sys.prefix come from?

- Python/sysmodule.c and then subsequently Modules/getpath.c

Modules/getpath.c

```c
#ifndef LANDMARK
#define LANDMARK "os.py"
#endif

static int
search_for_prefix(char *argv0_path, char *home)
{
/* ... */

    /* Search from argv0_path, until root is found */
    copy_absolute(prefix, argv0_path);
    do {
        n = strlen(prefix);
        joinpath(prefix, lib_python);
        joinpath(prefix, LANDMARK);
        if (ismodule(prefix))
            return 1;
        prefix[n] = '\0';
        reduce(prefix);
    } while (prefix[0]);
```

Where does sys.prefix come from?

- If PYTHONHOME is set, that's sys.prefix.
- Starting from the location of the Python binary, search upwards.
- At each step, look for lib/pythonX.X/os.py.
- If it exists, we've found sys.prefix.
- If we never find it, fallback on hardcoded configure --prefix from build.

Hardcoded landmark

- os.py is the hardcoded landmark which has been defined in Modules/getpath.c as follows:

```
#ifndef LANDMARK
#define LANDMARK "os.py"
#endif
```

Experiment I

```
$ mkdir scratch; cd scratch

$ mkdir bin

$ cp /usr/bin/python bin/

$ tree
.
`-- bin
    `-- python

$ ./bin/python -c "import sys; print(sys.prefix)"

/usr
```

Result

sys.prefix is still /usr.

- No PYTHONHOME set.
- Binary is in /home/epsilon/scratch/bin/.
- No /home/epsilon/scratch/bin/lib/python2.6/os.py
- No /home/epsilon/scratch/lib/python2.6/os.py
- No /home/epsilon/lib/python2.6/os.py
- No /home/lib/python2.6/os.py
- /usr is the fallback build prefix.

Experiment II

Create the landmark.

```
$ mkdir -p lib/python2.6

$ touch lib/python2.6/os.py

$ tree
.
|-- bin
|   `-- python
`-- lib
    `-- python2.6
        `-- os.py
```

Result

```
$ ./bin/python -c "import sys; print(sys.prefix)"
'import site' failed; use -v for traceback
/home/epsilon/scratch

$ ./bin/python -c "import sys; print(sys.path)"
'import site' failed; use -v for traceback
['',
 '/home/epsilon/scratch/lib/python2.6/',
 '/home/epsilon/scratch/lib/python2.6/plat-linux2',
 '/home/epsilon/scratch/lib/python2.6/lib-tk',
 '/home/epsilon/scratch/lib/python2.6/lib-old',
 '/usr/lib/python2.6/lib-dynload']
```

Conclusion

We now have been able to create a trivial virtual environment.

As it doesn’t contain the standard library, it is still unusable.

This can be made usable by placing our own, modified site.py in lib/python2.6 which adds system stdlib paths to sys.path.

Thank you.