Installing Plone v3.2 on Mac OS X 10.5

Installing Plone v3.2 on Mac OS X 10.5

Instructions to install Plone v3.2 on Mac OS 10.5 Server and client.

Plone is a Python-based, multi-platform content management system. If you've stumbled on this document not knowing what Plone is, please take a look at the project site: Plone.org. The current recommended method of installing Plone is with the Unified Installer. The Unified Installer will download and install the necessary components for Plone to run.

- Plone Downloads Page

In most cases the Unified Installer will suit your needs. However, if you require more fine-tuned control over the base components, or are looking for a better understanding of the stack which forms Plone, this document will lay the groundwork to deploy Plone on Mac OS X 10.5.

Before installing Plone you should take some time to think about your system and how you're going to be using Plone. If you are using buildout, you are likely setting up a production or development environment. Consider mapping out your needs (services such as authentication, backup, and product installations) and infrastructure necessary for the installation.

These instructions will work on both Mac OS 10.5.x client and 10.5.x Server. They assume you know how to create users and have a basic understanding of the command line. Where appropriate, I will reference online documentation that has helped me with my deployment.

What you will need:

- A computer running Mac OS 10.5 Server or client.
- Mac OS X Developer Tools
- Python 2.4.x

These instructions cover the following steps

- Preparing To Install
- Creating Plone User Account
- Creating Plone Root Directory
- Installing Python v2.4
- Housekeeping and Setting Proper Paths
- Installing “Easy Install”
- Installing PIL & ZopeSkel
- Installing Plone
- Additional Steps

Preparing To Install

The initial steps to installing Plone are done in your administrator account. If you haven’t done it already, install the Apple xCode Developer tools that came with your installation of 10.5. If you want to check for the most recent version, go to Apple’s Developer Website and sign up. You can download a free copy and install it immediately.

If you are not logged in as your administrator, then do so.

Creating Plone User Account

Before you begin, you need to create a “non-privileged” user account that will be used to both launch and maintain your Plone buildout. This document does not go into user creation, but the key point here is that the account should be a staff account with no elevated privileges. For the purpose of these instructions, I will call the user account and store its home directory in /Users/plone.

Creating Plone Root Directory

Plone can be installed in any directory on your filesystem. If this is a merely a local install for your own development you could easily install it somewhere such as /Users/Shared. However, if this install will be for production services you should consider installing it on a non-system drive. The only requirement is that the user must be able to access the folder. So on a system with two drives, say: systemDrive and dataDrive, the following commands would create a working directory for your Plone buildouts:

```
sudo mkdir -p /Volumes/dataDrive/ploneBuildouts
```

```
sudo chmod 750 /Volumes/dataDrive/ploneBuildouts
```

```
sudo chown plone:admin /Volumes/dataDrive/ploneBuildouts
```

Note: Do not think of your ploneBuildouts as a typical web server. The files do not need to be world readable as it is the user that accesses and serves out the contents of the zodb files. You could also easily create a Plone development group that contains your Plone developers/administrators and grant them limited access to restart and troubleshoot the instance via the sudoers file.

Installing Python v2.4

Mac OS 10.5 comes pre-installed with Python v2.5. Unfortunately, Plone requires Python v2.4. We will have to download and install version 2.4 as an alternate install. This is quite easy and only takes a few minutes.

Note: When compiling code on Mac OS 10.5, it is always best to install your custom-built binaries in alternate locations on the filesystem rather than installing over the Apple installed versions. Doing the latter could break some dependencies and, in some cases, it is possible that Apple's Software Update may overwrite your custom-built binaries. Install in locations such as /usr/local or /opt.

Open Terminal and enter the following commands:

```
ls /usr
```

If the **ls** command does not list a directory called local you will have to create it and some subdirectories with the following commands:

```
sudo mkdir -p /usr/local/include
```

```
sudo mkdir -p /usr/local/lib
```

```
sudo mkdir -p /usr/local/bin
```

The first time you run **sudo** you will be asked for your administrator password. Go ahead and enter that and press return.

Next you will need to get the Python source code. Typically I compile code from a working directory that I download the source to. (If you don't have a working directory and want to create one `mkdir ~/working` from Terminal will create it for you.

From Terminal **cd** into your working directory: `cd ~/working`. Using curl, download the source code from Python.org.

Note: You may want to check to see if any newer, stable versions of 2.4 exist and download accordingly.

In Terminal enter:

```
curl -O http://www.python.org/ftp/python/2.4.6/Python-2.4.6.tgz
```

Since this is a compressed file we will need to uncompress it:

`gnutar -xzf ./Python-2.4.6.tgz`

GNUTAR will create a folder called Python-2.4.6. CD into it: `cd ./Python-2.4.6`.

The following command will compile and build Python-2.4.6 on your system. Each command will take a few minutes to complete and report any errors if there are any:

`./configure MACOSX_DEPLOYMENT_TARGET=10.5 --disable-tk`

`make`

`sudo make altinstall`

Once *sudo make altinstall* completes, Python v.2.4.6 will be installed on your system but will not affect any applications that might use the default install of v2.5.

You can now log out of your administrator account and login to the account. All the remaining work can be done remotely via **ssh**.

Housekeeping and Setting Proper Paths

Before continuing, we need to do a little housekeeping in the account to make sure the proper versions of Python are accessed and that any Python egg extensions are deployed in the correct location.

Login to the account either remotely or locally.

The default user shell used by Mac OS X 10.5 is **bash**. We will need to create a *.profile* file for bash to read and adjust its *PATH* and set the *PYTHONPATH*. Use the text editor of your choice.

Note: If you insist on using a GUI text editor, I suggest TextWrangler or Smultron. I prefer emacs from the command line as it generates a backup file in case you make a mistake. It is always a good idea to create backup files to fall back on should a modification go wrong. The .profile file needs to be saved in the root folder of your home directory.

`emacs ~/.profile`

> PATH=~/bin:/usr/local/bin:${PATH}
> export PATH
> export PYTHONPATH=~/Library/Python/2.4/site-packages/

Note: To save your file in emacs and exit use the following key combinations: `^x^s ^x^c`

The next file to create is *.pydistutils.cfg*

`emacs ~/.pydistutils.cfg`

> [install]

> install_lib = ~/Library/Python/$py_version_short/site-packages
> install_scripts = ~/bin

Finally, we need to make the directory Python will store its eggs in:

`mkdir -p ~/Library/Python/2.4/site-packages`

Logout of the account and then log back in. To check that your *PATH* is set up correctly type:

`which python2.4`

If */usr/local/bin/python2.4* is returned your path is set correctly.

It's also a good idea to check your *PYTHONPATH* variable:

`echo $PYTHONPATH`

This should return the full path to the site-packages directory we created. Depending on where you specified the home directory to be, it may be something like this:

/Users/plone/Library/Python/2.4/site-packages/

The *PYTHONPATH* variable tells Python2.4 where to install or find any eggs necessary to operate.

Installing "Easy Install"

Easy Install is a utility that Python uses to download, build, install and manage Python packages. The utility is well documented on the developer's website: Python Enterprise Application Kit.

Create a working directory to install your Python packages from and CD into it.

`mkdir ~/pythonInstallScripts`

`cd ~/pythonInstallScripts`

Download Easy Install.

`curl -O http://peak.telecommunity.com/dist/ez_setup.py`

Once the download has finished you can install with the following command:

```
python2.4 ez_setup.py
```

This will download and install the *Setuptools Python egg* in the account's *~/Library/Python/2.4/site-packages/* directory. It will also create a directory in the home folder called **bin**. If you `cd ~/bin` you will find the *easy_install* binaries when you ls the directory:

easy_install easy_install-2.4

You will see two versions of *easy_install*. You need to use the version of *easy_install* specific to the version of Python you are running. In our case, this is **easy_install-2.4**. If you don't trust yourself to remember this you can remove execute permissions on *easy_install* and move it elsewhere or delete it and create a virtual link.

```
cd ~/bin
```

```
chmod 440 easy_install
```

```
mkdir deprecated
```

```
mv easy_install ./deprecated/easy_install.dep
```

```
ln -s ./easy_install-2.4 ./easy_install
```

*Note: Apple has its own version of Easy Install in /usr/bin. The path we set above should grab the new version, however it's a good idea to check using which easy*install. I recommend specifying *easy_install-2.4* to leave out the guess work._

Installing PIL & ZopeSkel

Now that you have both Python v2.4 and Easy Install in place, you need to install PIL and ZopeSkel. To do so, type the following commands:

```
easy_install-2.4 --find-links http://dist.repoze.org/PIL-1.1.6.tar.gz PIL
```

Note: You will need to download the Unified Installer and use the included PIL: PILwoTk-1.1.6.3.tar.gz

```
easy_install-2.4 /path/to/PILwoTk-1.1.6.3.tar.gz
```

```
easy_install-2.4 -U ZopeSkel
```

It will download and install all necessary files for Zope to operate.

Installing Plone

Once you have installed ZopeSkel, you are ready to install Plone. **CD** into the directory you wish your plone deployment to live. In our case it is */Volumes/dataDrive/ploneBuildouts*.

```
cd /Volumes/dataDrive/ploneBuildouts
```

Your Plone instance will need a name for the directory it is stored in. For purposes of this documentation, we'll call it *demo*. At the command prompt enter:

```
paster create -t plone3_buildout demo
```

You will be asked a series of questions. For now the default answers will suffice, but be sure to set a non-trivial password for your Zope root admin user.

Selected and implied templates:

> ZopeSkel#plone3_buildout A buildout for Plone 3 projects

> Variables:
> egg: ploneWeb
> package: ploneweb
> project: ploneWeb

> Enter plone_version (Which Plone version to install) ['3.2.1']: 3.2.1
> Enter zope2_install (Path to Zope 2 installation; leave blank to fetch one) ['']:
> Enter plone_products_install (Path to directory containing Plone products; leave blank to fetch one) ['']:
> Enter zope_user (Zope root admin user) ['admin']:
> Enter zope_password (Zope root admin password) ['']:
> Enter http_port (HTTP port) 8080:
> Enter debug_mode (Should debug mode be "on" or "off"?) ['off']:
> Enter verbose_security (Should verbose security be "on" or "off"?) ['off']:

Note: Be sure to set a non-trivial password for the admin account.

Note: If you specify a HTTP port other than 8080, you should verify that the port is open in /etc/services and not in use by another service. If not, you'll need to append the port information to the /etc/services file.

Once *paster* has finished, **cd** into the demo directory and run the *bootstrap.py* script.

```
cd ./demo
```

```
python2.4 ./bootstrap.py
```

Note: The bootstrap.py script needs to be run only once per Plone instance.

The bootstrap script will build the rest of the files necessary to install Plone. When it has completed, you must run the buildout.cfg file created by the bootstrap script to install Plone.

```
./bin/buildout -v
```

Note: To customize your Plone installation, you can modify the buildout.cfg file to include products that extend Plone's capabilities.

Once buildout has completed you can launch your Plone instance by running the following command:

```
./bin/instance fg
```

*Note: The **fg** flag tells Plone to run in the foreground, posting debug information to the console. It's a good idea to test your install in debug mode to make sure Plone comes up cleanly. If you wish to start Plone in daemon mode, use ./bin/instance start.*

Plone should now be up and running. To connect to your instance, open a web browser and connect to: *http://yourHostName:8080/manage*

This will take you to the Zope Management Interface. Login with your admin account. In the right hand side of the main content window, you'll see a drop down menu labeled: *Select Type To Add*. Select **Plone Site** and click **Add**. A dialog will appear, enter the following info:

> **ID**: Site container name, think of this as the directory holding your site. This will appear in the URL path.
> **Title**: Your Website Name
> **Description**: Optional Information Describing the site.

Once you fill out this information **click** the "*Add Plone Site*" button and your site will be created. To visit your site point your web browser to: http://yourHostName:8080/ID where ID is the site ID you set when you added the Plone Site.

Additional Steps

Now that you have a standard install of Plone. You can begin modifying the setup by editing the buildout.cfg file and/or adding products to the Products directory.

Some third-party Products for Plone require additional libraries to be bound to Python, or installed as packages to Python; this is beyond the scope of this document at this time.

Some excellent sources of information include:

Plone CMS: Open Source Content Management

Weblion

Zope.org

Python Programming Language

This documentation wouldn't have been possible without the excellent tutorials at PSU's Weblion and Plone.org.

Revision #4
Created 2009-06-24 02:03:21 UTC by Brannon, Luke
Updated 2009-11-18 06:36:54 UTC by Brannon, Luke