# Plone

- Why is it important to use short names in Plone?
- Plone CMS Resources
- Plone 4 Tips and Tricks: Table of Contents
- How do I identify the stylesheets in Plone?
- How to get rid of icons in Plone
- Importing and exporting a Plone site
- Installing Plone v3.2 on Mac OS X 10.5
- Remove highlighting of search terms in Plone
- Is there a permission that allows a user edit content that s/he does not own in Plone?
- Why can't I add a photo using AT Photo in Plone?
- Shibboleth For Plone
- How do I get started with designing new/existing layouts in Plone?
- Backing up and packing Plone's database file (Data.fs)
- Zope/Plone usage statistics
- Should I use plonecustom.css when changing the layout for my Plone site
- Changing number of displayed news/events in Plone portlets
- Search across multiple Plone instances
- How can I undo changes in Plone?
- How do I remove the icons in Plone?
- How do I change the header image in Plone?
- Why are my excluded Plone items still showing up in navigation?
- Plone and Zope Screencasts
- How to add new slots in Plone
- Restricting Plone portlets to show up only on certain pages
- Can Plone display content from another site inside it?
- How can a rotating banner image be done in Plone?
- How do I make dynamic dropdown/pullup menus in Plone?

- How do I enable the advanced mode of the TinyMCE editor for Plone?
- Why aren't my font colors / scripts / Flash / Java applets showing up in my Plone site?
- Importing existing HTML content into Plone
- How do you enable short names in Plone?
- Plone 4 Tips and Tricks

# Why is it important to use short names in Plone?

Short names become apart of the URL for Plone sites. Instead of an auto generated URL, you can create an easy to remember URL. For example, say you created a “General Information” page, your URL can show up like this: generalinfo or general-info, as opposed to something like this: 2006-05-11.1561651564.

Also notice how underscores are not used in creating short names. The reason is for functionality: underscores disappear when you email the link (most word processors will format the text to become a URL so it’ll highlight and underline the text), the second reason is because it’s hard to explain to a person what an underscore is. Check out Plone’s road map for more in info:

http://plone.org/products/plone/roadmap/73

To enable short name editing, see this article: https://kb.ucla.edu/link/217

# Plone CMS Resources

Plone is an Open-Source Content Management System built on Python and Zope.

Departments at UCLA currently using Plone

- Office of Instructional Development
- Social Sciences Computing
- Psychology IT Department
- The School of Engineering
- UCLA Chancellor
- UCLA Executive Vice Chancellor and Provost
- For a list of sites using it UC-wide, visit http://plone.ucla.edu/links/ucplonesites/

Does Plone do what I'm looking for?

I'm evaluating a system to do X, Y and Z, can Plone do that? Is Plone the right tool for the job? http://plone.org/documentation/faq/is-plone-for-me

Useful Links

- http://www.plone.org/
- PHP with Plone
- Plone and MySQL Integration
- UCLA Plone Users Group
- UCLA's Plone Mailing List
- Plone Support and Mailing Lists
- For those of you still trying to figure out where everything is in Plone and how to change its look – Tutorial
- 18 Things I Wish Were True About Plone by Alexander Limi – Co-Founder of Plone
- GloWorm A great tool that works similar to Firebug but for Plone. Check out the video. Amazing.
- The Future of Plone's User Experience
- Weblion Penn State has done an amazing job at implementing Plone on their campus. Their support and documentation is amazing and they've done this through Weblion. I find myself constantly using their wiki (https://weblion.psu.edu/trac/weblion/wiki) for references on Plone. *Mike Takahashi*

- WebServerAuth – Shibboleth Integration WebServerAuth, which replaces apachepas and AutoMemberMakerPasPlugin, allows Plone to delegate authentication concerns to a web server like Apache or IIS. Using WebServerAuth, Plone can be configured so any user known to your LDAP, Kerberos, Shibboleth, or Pubcookie system—or any other system for which your web server has an authentication module—can transparently log in using enterprise-wide credentials.

Useful References for Plone and Zope

- The Definitive Guide to Plone – The complete book online. Great resource for Plone. Essentially the "Plone Bible". Mike Takahashi August 19, 2005
- The Book of Zope – a good book to help get you familiar with Zope. It's a little out dated (ignore the DTML references). However, it really goes into depth on the inner workings of Zope (acquisition, ZCatalog, roles, permissions, etc.) and is something everyone should learn if you plan to customize and develop your Plone site.
- Online, the The Zope Book
  is a nice free resource. A little outdated, but the same principles still apply. Ignore the DTML sections as any customizations you do with Zope/Plone should be with TAL/Python.

  This section is a must for learning TAL *(quoted with permission from Mike Takashi email Feb. 6, 2007)*
- http://del.icio.us/tag/plone

Plone Podcasts

- Plone at Disney – Interview with Scott Kelley
  "Users managing content with Vignette was becoming expensive from a licensing and training cost point. Installing Enfold Server to run Plone on Windows was the stategy Disney's Enterprise Operations team took to offloading content creation and security management to individual business units. Then re-integrating the content back into their intranet."
- Rob Miller from Burning Man on Plone
- Alan Runyan on Plone and Enfold Systems
- Plone Impressions – A Video Podcast About Plone

Plone Presentations

- UCCSC 2007 Conference Plone Presentation – by Mike Takahashi

# Plone 4 Tips and Tricks: Table of Contents

## Tiny MCE

# How do I identify the stylesheets in Plone?

In Plone, you can list, debug, enable and disable and change the order of all the stylesheets by:

- go into ZMI (Zope Management Interface)
- go to Root Folder of your site
- click on portal_css (CSS Registry). There you'll see a long list of all the CSS's affecting your site

# How to get rid of icons in Plone

Plone’s default style calls for an assortment of eye candy to decorate links, list items, and various other elements. Sometimes users don’t want to see these icons. Plone makes it easy to disable any of these individual elements, but sometimes tracking down all the icons can be difficult.

There are three places icons may appear in your site:

- Site actions (top right of the content pane, items such as print)
- Link decorations (for example, the globe next to external links)
- Portlet item decorations (such as events and news)

Disabling any type of icon must be done in the ZMI.

# Disabling site action icons

In the ZMI, go to /portal_actions. You’ll see a list of all actions. Simply uncheck the “Visible?” checkbox for each icon you want to get rid of.

# Disabling link decorations

Slightly more complex than site actions. In your ploneCustom.css, add a CSS rule for any link type you want to disable: set the background to “none” and the padding to zero. Here is an example:
`.link-parent {`
@ background: none;@
@ padding: 0;@
`}`

Here is a comprehensive list of all special link styles: “.link-parent, .link-user, .link-external, .link-https, .link-mailto, .link-news,
.link-ftp, .link-irc, .link-callto, .link-webcal, .link-feed, .link-comment”

You can simply turn this list into a CSS rule like so:
`.link-parent, .link-user, .link-external, .link-https, .link-mailto, .link-news,`
`.link-ftp, .link-irc, .link-callto, .link-webcal, .link-feed, .link-comment {`
@ background: none;@
@ padding: 0;@
`}`

All your link icons should now vanish.

# Disabling portlet icons

This is much trickier than disabling other icons. What you must do is customize whatever portlet whose icons you want to remove. Manually edit that portlet code to remove icon images. All portlets can be found in /portal_skins/plone_portlets. For example, to remove the icons from portlet_events, find this line: `<img src="#" alt="" tal:replace="structure here/event_icon.gif" />` and remove it. Other portlets should have similarly-defined icons.

As you can see, there is no easy way to disable **all** icons at once in a Plone site, but hopefully this article will provide a good checklist for places to check.

# Importing and exporting a Plone site

Zope has a feature that allows you to export files, folders, and even Plone sites using the Import/Export button. However, it’s important to note that this tool should not be used as a migration tool when moving an older Plone site to a newer one.

For example, you have a Plone 2.1 site that you want to export and then import into a Plone 3.x site, then upgrade it. This will not work.

Import/Export only works with identical systems. This means that both Zope instances must be identical. Down to the same version of Zope and the same Products installed. A complete mirror.

The correct way to upgrade a site is to move the Data.fs file. For more information, go to:

http://plone.org/documentation/manual/upgrade-guide

# 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.

# Remove highlighting of search terms in Plone

If you haven’t noticed already, go to Google and do a quick search for your Plone site. Then click on your site. Plone will then highlight all of the searched terms.

This can be helpful, but also distracting. Plone will try to match the highlight color with the over all color scheme of your site. However, if you want to disable this feature, here’s what you do:

1. Go to the ZMI → portal_skins → plone_ecmascripts → highlighsearchterms.js
2. Hit *Customize*
3. add *return false;* just before *function highlightSearchTerms(terms, startnode) {* and *function highlightSearchTermsFromURI() {*
4. Then hit save and you’re done!

The adjustment below will permanently turn off highlighting for your site. On the other hand, if you only want to temporarily turn off the highlighting, simply remove “/?searchterm=mySearchTerm” from the end of the URI in the browser’s address bar.

# Is there a permission that allows a user edit content that s/he does not own in Plone?

For Plone 2.x, 3.x

Question:

I notice that for a piece of content, both the Owner and the Manager can edit it. Naturally, I assumed that there a permission that allows the role Manager to be able to edit any content it does not own.

However, after I assign ALL the permission to a certain role, say, Reviewer, at the Plone root level (going to the Security tab after clicking on the Plone site in ZMI), a user with the role Reviewer is still not able to edit content that s/he does not own. So, it leads me to think that there is no such permission for editing content for other users? If that is the case, is the Manager role somehow “hardcoded” to be able to edit anything?

What I want to do is to have a role that can do anything (create, edit, delete) to any content of a Plone site, yet it has no access to Site Setup in Plone or whatever features that change the technical aspects of the site.

Answer:

In the ZMI, click on portal_workflow, then plone_workflow. Then click on the â€œStatesâ€ tab on the top.

As a test, try modifying the published state. Click on â€œpublishedâ€ and then select the â€œPermissionsâ€ tab on the top. Then check â€œModify portal contentâ€ for the Reviewer role.

Also, once you have made the changes you must update the security settings or your changes will not go into effect. This can be done by clicking on portal_workflow and scrolling all the way to the bottom of the page and clicking on â€œUpdate security settingsâ€.

# Why can't I add a photo using AT Photo in Plone?

If you've used the product ATPhoto or ATPhoto Album in Plone 2.1, but now it is breaking in Plone 2.5, before you rip your hair out, here's what you need to do:

1. copy the code from this site:

   http://dev.plone.org/collective/browser/ATPhoto/trunk/configure.zcml (you might need to clean it up a bit)
2. log into your server and go to your Zope folder
3. next go to /instance-home/Products/ATPhoto/configure.zcml (you might want to make a backup of this file)
4. remove the existing code and replace it with the one you've copied
5. restart Zope (you can do this via ZMI)

And volia! It's fixed.

# Shibboleth For Plone

**Updated as of June 25th, 2010**

UCLA Shibboleth 2.1+ Guides:

Installation guide

Configuration guide

Follow up with installation of WebServerAuth: http://plone.org/products/webserverauth

*Does "(null)" show up instead of the login name in Plone when all is said and done?*

Head over to your Apache SSL configuration (/etc/httpd/conf.d/ssl.conf) and modify your RequestHeader setting of X_REMOTE_USER to utilize the Shibboleth attribute you desire:

@RequestHeader set X_REMOTE_USER %{SHIBUCLALOGONID}e
@

**The most up to date instructions for the Shibboleth plug-ins for Plone are available from Ithaka.org:**

http://tid.ithaka.org/shibplone.pdf

**Here are older ones**

Thanks to Alan Brenner for creating these plug-ins and all the help.

http://tid.ithaka.org/software

Thanks to Datta Mahabalagiri at UCLA AIS

All my paths to files are for OS X

Please connect your Service Provider to www.testshib.org to make sure your installation is solid before connecting to UCLA

native.logger and shibd.logger should be set to DEBUG instead of INFO…

Native Logger

Shibd Logger

they are located here:

/opt/shibboleth-sp/etc/shibboleth/shibd.logger
/opt/shibboleth-sp/etc/shibboleth/native.logger

…for the log files located here

/opt/shibboleth-sp/var/log/httpd/native.log
/opt/shibboleth-sp/var/log/shibboleth/shibd.log

Check that you have the correct Attribute Acceptance Policy for the UCLA Identity Provider
/opt/shibboleth-sp/etc/shibboleth/AAP.xml

AAP.xml

Verify you have the correct metadata for the UCLA Identity Provider
/opt/shibboleth-sp/etc/shibboleth/ucla-metadata.xml

UCLA Metadata

mine is in the md namespace

Setup your Shibboleth.xml like so:

shibboleth.xml

here is my example vhost in my httpd.conf, it isn't that pretty.

vhost.conf

or check out what Alan Brenner did

Alan's Vhost

Make sure your Service Provider is receiving attributes correctly though a simple phpinfo() page or this page that can display Shibboleth attributes

Here is mine

https://test.psych.ucla.edu/secure/

Here is the code I found on google

Check Attributes Page

First Install

ApachePAS plugin

http://plone.org/products/apachepas

Then Install the Shib Plugins

AutoUserMakerPASPlugin
ShibbolethLogin
ShibbolethPermissions

from here

http://tid.ithaka.org/software

configure AutoUserMakerPASPlugin in the ZMI at /psych/acl_users/AutoUserMakerPASPlugin to look like this

http://www.psych.ucla.edu/shibfiles/autouserconf.jpg

I'm only using the first two HTTP_REMOTE_USER1 and HTTP_SHIB_DISPLAYNAME you can ignore the rest of the "User Setup Headers"

make sure you put whavever "User Setup Headers" you are using down below in the "User Mapping Headers"

Configure Shibboleth Login at /psych/acl_users/ShibbolethLogin to look like this

http://www.psych.ucla.edu/shibfiles/shibloginconf.jpg

When you login to your site select the "Log in with a UCLA user id" link

That's it. Kinda rough.

I don't have a logout function yet.

I haven't gotten around to using ShibbolethPermissions yet but maybe this might get you going:

http://tid.ithaka.org/software/shibbolethpermissions/

Gotcha's
"Session Creation Failure" errors were from having the wrong SessionInitiator in my shibboleth.xml

"Rejected Replayed Assertion ID" were from incorrect Host and Path in the RequestMapProvider

Good Luck

# How do I get started with designing new/existing layouts in Plone?

Taken from various posts on the Plone mailing list (http://www.nabble.com/Plone-f6741.html)

Stan McFarland wrote:

> â€œThe short answer is that you can make Plone look any way you want with a combination of template customization and CSS. You just need to learn how to do it. Andy McKay's "The Definitive Guide to Plone" is a good place to start, as well as plone.org.â€

J Cameron Cooper wrote:

> â€œThe best way is to start with the Plone pages, which have a fairly standard and general template, and customize them with CSS.

> If you want to work an existing design into Plone, you will have to do some slightly trickier stuff. (Which, basically, is replacing main_template with your own structure, though preserving the "signature"
> of main_template.)â€

Peter Fraterdeus wrote:

> â€œI highly recommend that you read the docs section in plone.org as a starting place for 'skins' customization, and best practices for building a "site product" which will instantiate your customizations.

>

> â€œAfter that, it’s probably best to have a good look at the way that the “main_template” is constructed and how the various layers of CSS are used to modify the look of the site. (on your *nix box, try “locate CMFPlone/skins/plone_templates/main_template.pt” or find it in the ZMI).â€

Matt Bowen wrote:

> â€œPlone uses Python for logic, Zope Page Templates for layout, and the Zope Object Database. If you don’t know Python, you’ll want to learn it — without it, you will be limited in your customizing. People seem to like Dive Into Python [http://www.diveintopython.org/], but there are lots of good tutorials online, and it’s a very nice language. ZPT you’ll pick up from the book and the tutorials.

> Finally, definitely check out the many, many good plone videos on Plone.org, at http://plone.org/about/movies and http://plone.org/events/conferences/seattle-2006/presentations/session-videos.
> There is a lot of good stuff there about all aspects of customization.â€

# Backing up and packing Plone's database file (Data.fs)

## Backing up the database

See “Backup Plone” and “Backup and recover Data.fs in linux” in Plone’s documentation.

Additional notes:

- Recent versions of Repozo can compress (gzip) the backup files in addition to doing incremental backups.
- When doing full backups, Repozo behaves exactly as using cp to copy Data.fs, except it uses ZODB’s code to determine where the last finished transaction is and therefore do not copy unfinished transactions at the end of Data.fs.
- Zope’s ZODB can use multiple “storages” (think of it as something like database backends). It comes with FileStorage (i.e. storing everything in Data.fs). DirectoryStorage is an alternative that is supposingly easier and faster to back up incrementally.

## Packing the database

Packing a database throws away old object versions (generated by undo actions) from the database. To pack a database:

Go to Zope’s admin panel, under Control Panel → Database → [database name, e.g. main], choose how much undo info to keep, and click “pack”. Note: `Data.fs.old` will contain a copy of the database file before packing.

(There is a Zope product called PloneMaintenance that can do scheduled backup, although I have not tried it yet.)

# Zope/Plone usage statistics

Since a Zope access log (Z2.log) has the same format as an Apache access log, Apache log analyzers will probably work for Zope/Plone too. I have tried AWStats and Webalizer and they both worked w/o needing to set anything special. I prefer Webalizer because:

- It's fast. (We have a 700 MB log. Webalizer took 70 seconds but AWStats took 10+ minutes.)
- It's easy to use. (No configuration was needed in my case.)
- Output is in the form of a bunch of static files (HTML, PNG), so no need to set up a CGI script just to view the stat.

# Should I use plonecustom.css when changing the layout for my Plone site

As stated in plonecustom.css, if you are going to be making heavy modifications to your layout, you should modify each CSS file accordingly (base.css, portlets.css, etc.).

plonecustom.css can be used for light modifications or any custom classes and idâ€™s you create.

# Changing number of displayed news/events in Plone portlets

Find out which version of Plone you are running. Versions prior to 2.5 keep their portlet code in Zope Page Templates and can be modified easily. Plone 2.5 uses a new programming method called Views that complicates things. Versions after the 2.5.x series are not covered in this guide since they haven't been released yet, but working with these versions will likely be the same as Plone 2.5.

# Versions prior to 2.5

Go into the ZMI and navigate to /portal_skins/custom. If you already have a customized portlet_events or portlet_news, change the one you find. Otherwise, we'll need to begin customizing a fresh copy. Go to /portal_skins/plone_portlets. Select the appropriate portlet you want to change (either portlet_events or portlet_news) and customize it.

Look for a line near the top of the file beginning with `tal:define="results`. This TAL define statement goes on for several more lines. Look for the end of it, which has an expression looking like: `[:5]`. The number after the colon controls the number of events to display, and you can change it to whatever you like.

# Version 2.5 and later

Because Plone 2.5 uses Views, the code that does the searching for news or events is part of the Zope server itself. You can modify this file directly if you only have one Plone site and don't think you'll be changing it frequently. However, this is not the best method, just the simplest.

# Directly modifying Zope

Note: this will affect all Plone sites on your server. The code is found at: instance_home/Products/CMFPlone/browser/portlets/events.py (or news.py) in the root of your Zope install directory. Look for the code that specifies `sort_limit`, and change it to the number of items you want to show. Then look for a line that ends with `[:5]` and change the number in brackets also to the number of desired items. Save the file and restart your Zope server.

# Modifying the ZPT

If you have more than one Plone site, and want to change the number of events on one site without affecting the others, this method is best. By borrowing some code from the old Plone version, we can make the needed changes. Customize /portal_skins/plone_portlets/portlet_events (or portlet_news) if you don't have a customized copy already.

Look for the first tal:define tag and remove it (all 4 lines, or 3 lines for the events portlet). Now replace it with this code:

```
tal:define="results python:here.portal_catalog.searchResults(portal_type='Event', end={'query': DateTime(), 'range': 'min'}, sort_on='start', sort_limit=5, review_state='published')[:5];"
```

(If you're changing the news portlet, be sure to set `portal_type='News Item'` and remove the `end=` part.)

Change both the number after `sort_limit=` and the number in brackets at the end to be the number of items you want to display.

If you changed the events page, now you need to replace all occurrences of `events_link` with `string:/events`, and replace `prev_events_link` with `string:/events/previous`. (These are the default names. If you changed the names of the Events or Previous Events pages, you'll need to use their new URLs here.)

If you changed the news page, replace all occurrences of `prev_events_link` with `string:/news`. (Again, this is the default name. If you changed it, use the correct URL for the News page.)

# Customizing the portlets in other ways

You can change sort order, sort criteria, and other things by altering the parameters to the searchResults function that we used. See the ZCatalog help page for more information.

# Search across multiple Plone instances

If you have multiple Plone instances and you would like the search feature to search across the instances instead of only the one you are performing the search in, there is an add-on called **PloneRSSSearch** and it is available from the following site:

http://ingeniweb.sourceforge.net/Products/PloneRSSSearch/

Another alternative is to use **External Site Catalog** but it is more general:

http://ingeniweb.sourceforge.net/Products/ExternalSiteCatalog/

# How can I undo changes in Plone?

Plone has a feature that lets you undo changes you make to Plone-managed pages and other items. To access it, log in, and in your personal toolbar (usually at the bottom of the screen) there will be an “undo” link. If your site’s templated has been customized so that the undo link is no longer there, you can access it via http://your.plone.site/undo_form.

The undo screen is easy to use. Simply check the box next to any change you want to undo, and press the “undo” button at the bottom. You can undo nearly any Plone action, even logins! (Not terribly useful, but maybe you can think of a use for it.)

The Plone undo feature is meant to nullify recent changes. It is not a long-term archiving tool. Depending on your version of Plone, the undo history may be wiped whenever the Zope server is restarted, or when its database is compacted. For keeping important revisions permanently, Plone 3.0 will have a versioning feature. Until that time, you should make regular backups of your site, relying on the undo feature only for short-term revisions.

Zope has its own more sophisticated undo feature inside the ZMI. To get to it, navigate to any object, and then click the “Undo” tab at the top of the screen. You can undo changes in just the same way as the Plone undo page. You can also view particular revisions in the object’s history by clicking the “History” tab. Keep in mind that you cannot undo changes made to Plone objects in the ZMI – you must use Plone to undo Plone changes.

# How do I remove the icons in Plone?

This will allow you to remove the navigation icons that appear in Plone’s navigation portlet.

- go into ZMI (Zope Management Interface)
- go to Root Folder of your site
- click on portal_css (CSS Registry). There you’ll see a long list of all the CSSes affecting your site
- disable generated.css

Additionally:

- under “Condition” put the following: not: portal/portal_membership/isAnonymousUser (This will allow the icons to appear to users who are logged in.)

# How do I change the header image in Plone?

Enter the the Zope interface by adding /manage to the end of your URL.
Go to the folder ‘/portal_skins’ and the subfolder ‘/plone_images’. Click on ‘logo.jpg’ Click on Customize and you will be directed to the /custom folder where you can upload your logo image.

For more info: http://plone.org/documentation/how-to/custom-logo

If you have already changed the logo image and would like to change it to another, start in the folder /portal_skins/custom, click on logo.jpg and upload the new image.

# Why are my excluded Plone items still showing up in navigation?

Plone comes with a useful way to hide certain items from the navigation menu. On the page's properties tab, you can select "exclude from navigation" and the item won't show up anymore in the navigation bar or site map.

… at least, that's what you would think. You'll notice that if you navigate to the page you excluded (by following a link or typing in its URL directly), it will show up in the navigation again!

The fix involves some Zope template code:

1. Open up the ZMI, and customize /portal_skins/plone_portlets/portlet_navtree_macro if you don't have it customized already.
2. Look for a line of code about 20 lines down that says:
   `tal:condition="python: bottomLevel <= 0 or level < bottomLevel-1">`
3. Change the line to read:
   `tal:condition="python: (bottomLevel <= 0 or level < bottomLevel-1) and (not item.exclude_from_nav)">`

Now your item will never show up in the navigation tools as long as it's excluded.

# Plone and Zope Screencasts

Collection of screencasts demoing Plone functionality and ease of development in Zope 3.

# How to add new slots in Plone

This document describes how to add an additional portlet slot to the two that exist already (left and right).

http://plone.org/documentation/how-to/add-slots

# Restricting Plone portlets to show up only on certain pages

Sometimes you may want certain Plone portlets to only show up on certain pages. This guide will walk you through doing exactly that.

If the portlets already appear on your site, you'll need to remove the portlet calls from the site properties page. In the Zope Management Interface, click on the name of your site at the top of the left-hand navigation bar. Then click on the properties tab at the top of the right-hand pane, and you'll see a list of properties. The two important ones are left_slots and right_slots. These contain lists of portlets you want to show up in the left and right panes, respectively, of your site. Remove the lines referring to the portlets you want to appear only on certain pages. Remember to click the "Save Changes" button near the bottom when you finish.

To get portlets to appear conditionally, we are going to manually add in code to the main template that references the portlets. If you have a customized main_template file in your custom folder, edit that. If not, create a customized version of main_template, found in /portal_skins/portal_templates. Find where you want the portlets to appear, and insert this code in the appropriate spot:

```
<div tal:condition="python: context.id in ('welcome', 'news')">
@

News
@
</div>
```

The first div element is used to conditionally filter the pages that the portlet shows up on. Replace `('welcome', 'news')` with a Python-style list (that is, strings separated by commas) of the IDs of the pages where you want the portlet to appear.

The second div calls the portlet code. Replace `portlet_news` in the above path expression with the name of the portlet you want. The text inside the div (in this case, "News") is just a placeholder and can be anything you want.

You only need one outer div (with the tal:condition code), provided your conditional portlets are going to go in the same area. Just add an inner div with the appropriate metal:use-macro attribute for each portlet. Calling portlets in this way also gives you the freedom to place them anywhere you like, not just in the pre-defined left and right parts of the screen.

Say you only want the News portlet to appear and nothing else, you can use a filler portlet, one that you know is not in use. For example: here/portlet_related/macros/portlet.

But what about the pages you that you didn’t the filler to show up (for example, you have the portlet set to show up on the left hand side, but for the pages without the portlet, the text should be flushed left). It’s a little tedious, but go to the folders in the ZMI that you want the filler portlet to be removed from. Go to the Properties tab. At the bottom of the form, add a property called *left_slots*, with *lines* as the type and no content. Press add. Also, anything that is a child of the folder will also carry this property.

# Can Plone display content from another site inside it?

Yes, there is a product that will do exactly this. It's called windowZ. Just use the quick installer to set it up, and you can add Window objects through the Plone interface. Window objects don't have their own content, but rather a URL that points to the content you want to display.

Additionally, the external content becomes searchable within Plone's powerful search tool. However, if the content is constantly being updated (i.e. a Blog) this will not be automatically updated. Also, the external content must be of fixed width and height because WindowZ does not support dynamic data.

# How can a rotating banner image be done in Plone?

The first step is to create a Python script that will serve up the image you want. This article assumes that you want a random image from a folder to be chosen. Python-savvy readers can use more advanced techniques, such as making a list of images to load in the same order, etc.

Through the ZMI go to /plone_skins/custom folder, and add a Script (Python) object from the drop down menu. Give this script a name (this example uses “randimg”) and put this code into it:

```
from random import choice
# Import a standard function, and get the HTML request and response objects.
from Products.PythonScripts.standard import html_quote
request = container.REQUEST
RESPONSE = request.RESPONSE
RESPONSE.setHeader('Content-Type', 'image/jpeg')
# Get the current folder's contents, choose an image and return it
images = context.objectValues(['ATImage'])
return choice(images).data
```

Now that the script is in place, you can test it by pointing your browser to http://your.plone.site/images_folder/randimg. (Replace `images_folder` with whatever your random images folder is called.) It will work just like a normal image file on a web server. To add the image anywhere in your site, add the usual image HTML: **<img src=“/images_folder/randimg”>**. Every time the page is reloaded, a randomly-selected image from images_folder will be shown.

If you are using images other than JPEGs, you will need to change the `setHeader` call so its second parameter reflects the image type you’re using. For example, image/gif or image/png.

Also, the `objectValues` function retrieves only the file types you ask it to, in this case “ATImage”, which means images uploaded through the Plone interface. Images uploaded through the ZMI have a different file type, “Image”. This is an important distinction, so change the parameter if you need to. Note however, all content should be added through Plone and not through the ZMI. All content management should be done in Plone. Working with the ZMI in this fashion can result in unforseen problems. You have been warned. An alternative is to leave the parameter to `objectValues` blank, so that it returns all objects in the folder. However, if you use this option, make sure your random images folder has only images in it!

---

Update from Joel Burton on Plone.org

Using ‘objectValues’ will actually *retrieve* each image from the folder, even though you only want to use one. Not very efficient. It also is making no checks for things like effective/expiration date.

Better would be:

```
images = context.randomimages.getFolderContents({'portal_type':'Image'})
return choice(images).getObject().data
```

Overall, though, I’d recommend a different strategy: rather than having this script pretend to be the image, I’d have a script that returns a randomly-selected image **tag**. This will allow you to get a good image tag (w/height, width, alt text, etc.) from a randomly-chosen image. This would look like:

```
from random import choice
return choice(context.randomimages.getFolderContents({'portal_type':'Image'})).getObject().tag()
```

Then, rather than having code like `<img src="myscript" />`, you’d have your Page Template say `<img tal:replace="context/myscript" />`. This will give you a good image tag of a randomly-chosen image.

# How do I make dynamic dropdown/pullup menus in Plone?

There is a Plone Drop Down Menu product (also called qPloneDropDownMenu) that creates navigation tabs for you that have a drop-down menu behavior. The product page above has complete instructions on using it.

If you want the menus to “pull up” rather than “drop down”, one line of code in the drop down CSS file can do this:

1. Make a copy of /portal_skins/qPloneDropDownMenu/drop_down.css into your custom folder.
2. In the custom file, add **bottom: 12px;** into the style for **#portal-globalnav li ul**.

Note: your pixel mileage may vary. If your menus end up floating too high (or low, sinking into the navigation bar itself) try altering the value for the **bottom** attribute. You can also try using values measured in em, so that it changes depending on the font size.

# How do I enable the advanced mode of the TinyMCE editor for Plone?

The TinyMCE editor is a feature-rich WYSIWYG editor for Plone, similar to Kupu. However, many of TinyMCE's best features are hidden away in so-called "advanced mode". Installing TinyMCE is straightforward – install it like you would any other Plone product.

To enable advanced mode:

1. In the ZMI, go to /portal_skins/tinymce/tinymce_wysiwyg_support. Make a copy into "custom".
2. Edit the file in custom. Near the top there is a line that says: *<script type="text/javascript" charset="iso-8859-1" tal:attributes="src string:${portal_url}/jscripts/tiny_mce/tiny_mce_init.js"> </script>* — Delete this line or comment it out.
3. In its place, add this code: *<script language="javascript" type="text/javascript">*
   *tinyMCE.init({*
   *theme: "advanced",*
   *mode:* "*specific*textareas"
   });
   </script>

Within the body of the tinyMCE.init function, you can add other lines to change the appearance of the toolbar, change its location, add more buttons, etc.

Here is a reference of all the settings for the TinyMCE editor:

http://tinymce.moxiecode.com/tinymce/docs/reference_configuration.html

Pay special attention to the function theme_advanced_buttons<1-n>_add: you can use it to add more buttons to the toolbar. For example, to add an undo button to row 2, put this line into tinyMCE.init:
*theme_advanced_buttons2_add: "undo"*

A full list of available buttons can be found here:

http://tinymce.moxiecode.com/tinymce/docs/reference_buttons.html

Remember that customizing the TinyMCE editor is only possible when using the Advanced theme.

# Why aren't my font colors / scripts / Flash / Java applets showing up in my Plone site?

By default, Plone employs an HTML-filtering system when it transforms Plone documents into marked-up web pages. Certain tags, such as <font> and other deprecated tags in HTML, are removed so that the generated page is closer to XHTML, which does not allow deprecated tags. Other tags such as <script> and <embed> are disabled because they present certain security issues.

While avoiding the use of these tags is certainly the preferred “Plone way of doing things”, this isn’t always an option. Certain tools such as the TinyMCE editor use <font> and other filtered tags, and won’t be displayed by default, forcing the user to change the filtering.

Instructions on changing tag filtering can be found at this Plone article. Look under the section labeled “Safe HTML”. Note that you must have Plone 2.1.2 or later in order to configure Safe HTML.

# Importing existing HTML content into Plone

I have an existing site consisting of HTML files, folders and images. Now I’d like to manage that site in Plone. How can I get my existing content into Plone?

Answer: https://docs.plone.org/develop/import/index.html

# How do you enable short names in Plone?

By default Plone 2.1.x generates short names for you based on the title of your document, folder, etc in the same manner https://kb.ucla.edu does.

For example, if you create a document with a title called “Plone is Great” the short name becomes “plone-is-great”. You can disable this and also allow users to choose their own short names by selecting it in Plone as admin via Site Setup → Portal Settings and selecting “yes” for the option "Show “Short Name” on Content?"

Each user must also enable this option.

To enable this options for each member of your Plone site, simply click on the “Preferences” link located at the bottom of the page and select “Personal Preference.” Scroll down and locate “Allow editing of Short Names” and check the box. hit save.

If you’re the admin and would like to give your managers permission without having to remind them to change their preferences, click on the same “Preferences” link. Then look to the left navigation menu and click on “Users and Groups Administration.” The names of your users should show up. Click on the names of the users whom you’ve assigned as managers. Scroll down and locate “Allow editing of Short Names” and check the box. hit save.

# Plone 4 Tips and Tricks

*Back to Table of Contents*

# Tiny MCE

***Adding a color picker to the toolbar***

Plone 4 comes installed with Tiny MCE as the default html editor. It has a WYSIWYG interface with a toolbar at the top. However, you may notice that some basic functions are missing (like a color picker). To activate the color picker, follow these steps.

1. Log in with Site Admin privileges.

2. Go to Site Setup > Tiny MCE Visual Editor > and choose the **Toolbar** tab at the top.

3. Scroll down until you find the **Forecolor** checkbox (off by default) and check it and save.

4. Next, go to **Html Filtering** (site setup).

5. Click the **Styles** tab and add the tag ‘color’ to permitted styles (Add permitted styles button) and click save.

That’s it! The next time you go to the html editor you will see the color picker icon in the toolbar.