

Tutorials

Tutorials are very detailed step-by-step articles. They commonly explain a pathway to success for an overarching goal. Creating a photography site that allows multiple content contributors to submit pictures is an example of a tutorial. Explaining how to leverage the assetmanager module's features to grab just the pictures you want for a certain node would be a HowTo. Get it now? Good!

Go forth and share your knowledge

Now that you understand the difference, its time to write one of your own. If you've accomplished something not listed here, please click the 'add a child page' link near the bottom of this page and become a tutorial contributor. Trust us, it will be so satisfying to have other users reading the story of how you did X with Y & Z!

Links to tutorials in other sections of the Handbook

In order to avoid duplication, here are some links to tutorials that live in other handbooks:

- [Multi-Site installation](#)

Configuring menus, navigation and blogroll for newbies

A Four Part Tutorial on How to Set up A Basic Navigation, A Hierarchical Navigation, Separate Blocks for Your Navigation and A Blogroll using Drupal 6.1 (Garland Theme)

I wanted to set up a basic, hierarchy site nav for my site, but I found all the posts and documentation that I read so confusing and (forgive me pls.) over-geeked that I was ready to give up. Surely it couldn't be this difficult for the most basic function of a site. And it's not. In truth, once I 'saw' the answer, I was appalled at how easy it was, and how close I had come to throwing the whole thing out.

This is how it is done, as close to step by step as I can make it. I will say this before you begin. Go through the steps. You will learn as you go, and once you start to see your results, you will begin to catch on how to go about the rest, or how to change it, or how to make it do EXACTLY what you want. But go through the basics, till you understand it first.

Also, remember that you MUST CREATE ACTUAL CONTENT for most of your links to begin to show up on your pages. However, the structure will be there for when you do.

SETTING UP THE BASIC, FIRST LEVEL NAVIGATION.

Go to Admin > Menus
Select Primary Links

Primary links are the links that will show up in both the blue banner bar of the Garland theme, and as in an item called (Primary Links) in Administer > Site Building > Blocks, which you can move to any location you like. If you do not wish to add any links in the Blue Bar, then simply go back Administer > Site Building > Menus, and select Add Menu for your links.

This tutorial assumes however, that you want at least a few links to show up here, i.e. Home, or About Us, or Contact.

To return to the steps. On the Primary Links page:

Select Add Item.

You will get a page that asks you for Path, Menu Link Title, etc. For Path: type in node/add. For Menu Link Title, type in the name that you want to show as the Link. i.e. Home, or About Us or whatever link you want.

***NOTE concerning the Path URL:**

This I just realized ~ When you type in node/add as the Path URL, all the links that you create under Primary Links that show up in the Blue Banner bar of the Garland Theme will show up to your users (including you as the Admin) as links to 'create content' according to the content that they are permitted to create. This condition is only for the links that show up in the Blue Banner Bar. It does not pertain to the links that will show up in your sidebar under the Primary Links Block. If you do not want this (the create content link), then you will have to change the Path URL to node/(node number) according to the instructions further on in this tutorial on how to find the node number. You can also change this Path URL at any time through the edit function once you begin to create content, and so change the target of your links then.

You can add a description that will show up on a mousehover if you want by filling in the description box. You can also fill this in later if you want, as you refine your menu's and links by using the edit functions.

Going down the page ~

Toggle Enabled. This will make your link show up. Ignore Expanded for the time being. This used if you want your Navigation to remain in the expanded state (toggle ON) or closed, and expanded only when clicked (toggle OFF).

Parent Item: Select Primary Links. This will set the links to show up in the Primary Links Block, as well as in the Blue Banner of the Garland Theme.

Save.

Keep adding Items in the Primary Links Menu until you have the links that you want for your Primary or First Level Navigation. You should see these same links showing up in the Blue Banner of the Garland Theme as you save them.

Now: Go to BLOCKS.

Locate the Block that says Primary Links, and move it to your preferred location. This menu list will show up where you place it, and it will be a duplicate of the one that also shows up in the Blue Banner of the Garland Theme. If you do not want a duplicate, you can skip this step. Likewise, if you want to break up your Primary Nav into separate blocks, you can skip this step and follow the tutorial for this procedure further on down the page.

SETTING UP SECOND LEVEL NAVIGATION:

Go to Admin > Menus > Primary Links.

Select Add Item

Fill in Path (use node/add) and the Menu Link Title should be what you want the name of the actual link to be. Enter the description as you choose, toggle Enabled, and now you come to :

Select Parent Item. Scroll through the list, and find the Menu that you want this link to be set under. Choose that and you now have your second level hierarchy.

For example: You set up a Primary Link called About Us. Now you want to set up subsections under About Us. Toggle About Us, and your new subsection that you just created will show up here. If however, all you want to do is just list separate pages under About Us, you can do this through the page creation process and you do not need to set up a second level navigation.

You can continue doing this for each subsequent level to a depth of nine levels of hierarchy. Each Item you add under node/add as the Page url, you are essentially creating a 'folder'.

BREAKING UP YOUR NAVIGATION INTO SEPARATE BLOCKS

Not everyone wants their navigation to be one long list down the side, on all pages. Some would like to break up this navigatin and distribute it on the page, especially if they have a two column theme, as Garland does. There are a number of posts in the forums asking how to break this up and have only certain blocks appearing on certain pages. This portion of the tutorial will tell you how to break up your basic navigation to do this very thing.

Go to Admin > Menus > Add Menu

Select Add Menu.

You will see a page that has Menu Name, Title Name, a place for a short description, and a Save button.

The Menu name will be an internal ID name, all lower case letters with no spaces. Title Name will be the Title of the Menu. This title will also show up as a Block on the BLOCK section of your Admin. It will be the title of the Block of Nav you want to show up on your pages.

Add your description if you want. This description is for your benefit, and will show up on your Menus Page. SAVE.

You should now be back on the main Menus page, and your new Menu should be listed there with your description, if any, underneath it.

Select your Menu that you created. You should now be at a page that has the name of your Menu at the Top.

Select Add Item.

Go through the same procedure that was listed at the beginning of this tutorial, the way you did to create your primary links, and list all the particular Link Titles that you want to show up in this block. SAVE.

Go to BLOCKS.

Locate your Blocks that you just created, toggle them to go to whichever location you want them to go on your pages. SAVE BLOCKS. These blocks will now show up on all your pages according to

the locations that you have selected.

Want to have only certain blocks show up on certain pages?

Go to the block that you just activated on the BLOCKS page, and Select Configure.

You will now see a page that asks you for:

Block Title: This will be the title that you gave the block in the Menu. If you want to override this and use a different name, you can do so here. If you do not want the name to show at all, you can enter . Overriding the name does not change the name, it merely overrides it for the purposes of the block.

Select your configuration settings to suit your need.

Now you will reach the portion that says: SHOW block on specific pages.

This is where you can toggle to eliminate pages, or you can toggle to select certain pages for this particular block to go on. If you want a Blogroll to go only on blog pages (next section will tell you how to do that), then enter into the space, blog/*. The * is the wildcard that will select all blog pages.

If you only want the block to show up on specific pages, this is where you add that page. To do this, call up the actual page you want the block to show up on and look at the address bar. It will give you an address that will end in node/___ with ___ being a number. Enter this number into the SHOW block space on your block configuration page as simply node/35 or whatever your node number was. You will now have this specific block show up on only that particular page.

If you chose page/* then your block will only show up on content created as pages. Likewise with books, forums etc. If you only want it to show up on certain pages, then you will have to find their node numbers and add these in.

SAVE your configurations, and that is it.

NOW, you will have to start creating content if you want any of your blocks to start showing up. Once they start showing up, you will begin to 'see' how it works, and can alter or adjust your navigatin to suit your taste.

As a word of advice, DO NOT USE the 'NAVIGATION' Menu to set up your links. This is the default navigation for all your site permissions according to user. You can toggle this block to be where you want it to be through the BLOCKS menu, or you can have it not show up at all, by untoggling it.

Any of the Menu items that you want to move around, you can do so through using the drag and drop handles on Drupal 6.1 or through using the 'weight' function on Drupal 5.7.

If you try to use the Secondary Links, what you will end up with is by having your secondary links show up on the Blue Banner of the Garland theme. They do not necessarily show up under the primary link, and they are not drop downs either. As such Secondary Links is not a menu that will accomplish your purpose in setting up a basic nav with a hierarchy. What else you would use it for, I do not know. Using Primary LInks has resolved the Nav and Link issue admirably.

HOW TO CREATE A BLOGROLL OF EXTERNAL LINKS. (again, using the Garland Theme)

Go to Admin > Menus > Add Menu

Your Menu Name is an internal ID that should be all lowercase with no spaces.

Title Name is the title that you want to call your blogroll.

Enter a brief description if you want, and SAVE. You will be taken back to the Main Menu's list. The name of the menu that you just created should show up here.

Select your new Menu.

Select Add Item.

You will now see a page that asks for Path, and a Menu Link Title etc.

In the Path area, enter the full URL that you want for your blogroll link.

In the Menu Link Title, enter the name you want to appear on your blogroll.

i.e. Path could be #http://www.abcdefg.com# (the #'s are so it won't show up as a link, I hope) but the Menu Link Title could be 'Tiny Tots', which when selected will take you to abcdefg.com

Add a description of the site link. A brief word of what you are linking to.

Toggle Enable

Ignore Expanded, unless you want to repeat the steps of a hierarchy (maybe someone has multiple sites)

Parent Item: Select to the name you gave your new Menu or blogroll. If you ARE creating a Blogroll hierarchy, repeat all the same steps for creating a Nav hierarchy and set it to the parent item that you want it to show up under.

Weight can be set optionally, or you can use the drag and drop handles on the Menu page to set the order of your links.

SAVE.

AND THAT IS IT! No coding, no extra modules or plugins, all basic and straightforward.

As you begin to see it in action, and you begin to add content, you can play around with it, and tweak it to your hearts content (i.e. adding CSS and the like).

Dedicated and virtual host install

This article is "complete", but has not been properly reviewed. Please note any suggestions or comments.

This tutorial is a description of how to install Drupal on a virtual or dedicated host with root access.

This document presumes you are reasonably comfortable with the command line of your operating system and Drupal in general. It also assumes you already have a working base install of your operating system.

General instructions

1. Upgrade the operating system

2. Install the lamp stack. Depending on your server, you might have to install apache server, php, mysql server, php-mysql, php-mbstring, php-gd.
3. Install control panel of choice if desired. Webmin is a popular choice.
4. Enable the Apache and mysql daemons to start at boot. You can do this in Webmin. Remember that Webmin runs in its own Apache process.
5. Create a regular user. You may want to change ownership of /var/www/html (assuming that is the directory Apache uses on your system) files and directories to this owner. If it is the only user, you might wish to set the /var/www/html directory as the home directory.
6. Edit the httpd.conf file as needed.
7. Create any virtual hosts desired in Apache. This can be done in the httpd.conf file or with Webmin.
8. Create a mysql database for Drupal.
9. Download and unpack Drupal
10. Change Drupal permissions as desired. You may want to change the file ownership to your ftp user and the group to apache or nobody or whatever user (not process) you have apache running as. You can then add a files directory as /sites/default/files if desired. Then you can change the permissions for the directory /sites/default giving Apache ownership and make it belong to your user's group if desired. Chmod the /sites/default directory to make sure it is writeable by your Apache server. Something like 775 would probably be fine at this point.
11. Install Drupal from your browser and make sure it changes the permissions of your default folder and settings.php file to something secure. If Drupal has permission, it should do this automatically.

Configuring a Centos VPS for Drupal

Hi all

Target of this writeup is keeping track how I set up my VPS, asking for comments and possibly be of some assistance to people who would like to do the same. It's my first VPS ever, it's for absolute beginners so experts please comment.

Base OS:
Centos 5

Target setup:
Apache 2.2, with mod_ssl and mod_ruid
PHP 5.2, with APC and Uploadprogress
Mysql 5
Proftp
Webmin

Tools used:
[Putty](#), [SSH client](#)

Install Utter Ramblings Yum repository

CentOS 5 comes standard with PHP 5.1, some other repositories come standard with 5.3, which will either limit functionality or provide errors. So I will use the [Utter Remblings Yum Repository](#).

Login to your server as root with Putty, speedup Yum a little and update your server:

```
yum install yum-fastestmirror
yum update
```

Now for installing that repository:

I personally like to use the simplest of editors: vi. Open or create a file as above, start editing with "i", stop editing with help button "F1", save the file with "ZZ" and exit with ":q!".

```
vi /etc/yum.repos.d/uterramblings.repo
```

and paste (right mouse button in Putty) the following in it after clicking "i":

```
[uterramblings]
name=Jason's Utter Ramblings Repo
baseurl=http://www.jasonlitka.com/media/EL$releasever/$basearch/
enabled=1
gpgcheck=1
gpgkey=http://www.jasonlitka.com/media/RPM-GPG-KEY-jlitka

Read more: http://www.jasonlitka.com/yum-repository/#ixzz0iK46UmBx
```

Press "F1" and "ZZ" to save and exit.

Run update again:

```
yum update
```

Install Apache and PHP

Installing Apache, with development tools, mod_ssl and mod_php:

```
yum install httpd httpd-devel mod_ssl mod_php
```

You could be running into problems here when packages cannot be found. Some setups of Centos protect against certain updates to make sure the system keeps on working. If so, check:

```
vi /etc/yum.conf
```

and look for a line that start with "exclude". If it's there, comment it out with "#" after pressing "i". Press "F1" and save with "ZZ". Repeat the yum command before.

Point your browser to your domain or ip-address and see something working there.

Make a info.php file to check if PHP is working and it's configuration.

```
vi /etc/www/html/info.php
```

and copy, paste (use i) and save the following in it.

```
<?php
phpinfo();
?>
```

Point your browser to www.yourdomain.com/info.php or 123.123.123.123/info.php. You should now see the text above appearing in your browser. Since we just installed PHP, Apache is not booted with PHP. So we just have to restart it:

```
service httpd restart
```

And we reload the browser to see the PHPinfo page. As you should be able to see, PHP needs some more goodies to be really useful.

Dressing up PHP

Installing APC (a cache, which should work?) and uploadprogress (to get rid of those annoying status report errors).

```
yum install php-pear php-devel
```

Install a C++ compiler:

```
yum install gcc gcc-c++ autoconf automake
```

Then:

```
pecl install apc uploadprogress
```

Include in php

```
echo "extension=apc.so" > /etc/php.d/apc.ini
```

Which will make a small file in /etc/php.d/ containing "extension=apc.so", and so telling PHP that it should be included.

and (dito)

```
echo "extension=uploadprogress.so" > /etc/php.d/uploadprogress.ini
```

And restart apache

```
service httpd restart
```

Check info.php and find both APC and uploadprogress installed now.

Now for some graphics. Let's install GD2

```
yum install gd php-gd
```

and Imagemagick

```
yum install ImageMagick ImageMagick-perl ImageMagick-devel
```

Then:

```
pecl install imagick
```

Add the ini file

```
echo "extension=imagick.so" > /etc/php.d/imagick.ini
```

Install the php mbstring module Drupal needs and php-xml for Drupal 7 (dom).

```
yum install php-mbstring php-xml
```

Restart Apache to see the changes:

```
service httpd restart
```

And see in info.php ([Thanks!](#))

Installing Webmin

Why [Webmin](#)? I searched and tried around for some packages, and found Webmin one of the most practical for studying my own system. Instead of obscuring settings and changing lot's on it's own, it helps by making those easier accessible.

Making the Webmin repository:

```
vi /etc/yum.repos.d/webmin.repo
```

and paste there:

```
[Webmin]
name=Webmin Distribution Neutral
baseurl=http://download.webmin.com/download/yum
```

```
enabled=1
gpgkey=http://www.webmin.com/jcameron-key.asc
```

Then just do

```
yum install webmin
```

Now point your web browser to www.mydomain.com:10000 and login as root. ([Thanks!](#)) You could now surf to Others, PHP Configuration, Resource Limits and change the Maximum memory allocation to 128M. Save and check your info.php and see the memory_limit has changed to 128M.

Installing ProFTPD

ProFTP is probably the easiest FTP server on Centos, which is not shipped with Centos, so we will install another repository first.

```
vi /etc/yum.repos.d/dag.repo
```

And add

```
[dag]
name=DAG RPM Repository
baseurl=http://apt.sw.be/redhat/el$releasever/en/$basearch/dag
gpgcheck=1
enabled=1
gpgkey=http://dag.wieers.com/rpm/packages/RPM-GPG-KEY.dag.txt
```

Update the system and install Proftpd

```
yum update
yum install proftpd
```

Done! ([Thanks!](#))

Make Proftpd start at startup and start now:

```
chkconfig --levels 235 proftpd on
service proftpd start
```

Open Webmin again (www.yoursite.com:10000) and go to System - Users and Groups - Create a new user.

```
Username = "username"
Home directory = Automatic
Shell = bin/bash
Password = "password"
Primary group = New group with same name as user
Create home directory? = Yes
Copy template files to home directory? = Yes
Create user in other modules? = Yes
```

Startup your ftp program and connect to your web server using the username and password above. Make a new directory public_html and another one in there public_html/drupal6 and upload all Drupal files to that directory. ([Thanks!](#))

Compile mod_ruid

Why mod_ruid? Not really sure, it makes Apache change username and group to the files it uses at the moment. So if it's working in the home directory of drupaluser, it will behave like it is drupaluser, opening and writing files like it is you and limiting itself to drupalusers home directory. Another option would be SuPHP, but that's considered slow.

Official development of [mod_ruid](#) seemed to have stopped a few years ago. [Somebody in the Netherlands](#) apparently picked it up and provided a new version. We will use that one.

Install dependencies:

```
yum install libcap libcap-devel
```

Download to /tmp and unpack

```
cd /tmp/
wget http://www.monshouwer.eu/download/mod\_ruid/mod\_ruid-0.7.1.tar.bz2
tar -xvzf mod_ruid-0.7.1.tar.bz2
```

Compile mod_ruid and restart Apache

```
cd /tmp/mod_ruid-0.7.1/
apxs -a -i -l cap -c mod_ruid.c
service httpd restart
```

Careful! If you update Apache to a newer version, you might have to do the step above again.

Add a mod_ruid.conf file:

```
vi /etc/httpd/conf.d/mod_ruid.conf
```

Containing:

```
# Should be here but automaticly put in httpd.conf
# LoadModule ruid_module    modules/mod_ruid.so
RMode                        stat
RGroups                       apache
```

([Thanks!](#))

And restart Apache

```
service httpd restart
```

Installing Mysql

```
yum install mysql-server mysql php-mysql php-pdo
```

Boot on startup and boot now

```
chkconfig --levels 235 mysqld on
service mysqld start
```

Make your configuration safe.

```
/usr/bin/mysql_secure_installation
```

You will be asked some questions, answer them like this

```
Enter current password for root (enter for none): "enter"
Set root password? [Y/n] "y"
Remove anonymous users? [Y/n] y
Disallow root login remotely? [Y/n] y
Remove test database and access to it? [Y/n] y
Reload privilege tables now? [Y/n] y
```

Login to Webmin and make a database for your Drupal installation

Click on MySQL Database Server under servers (if it is not there you might need to do a "Refresh

Modules" on the left below.
Tell Webmin the root user (root) and your new password.

Create a new database for your Drupal website.

-> Create a new database.

Name: anything you like

Character set: utf8

Collation order: default

Initial table: none

Now make a new user

-> User Permissions

-> Create new user.

Username: drupal

Password: anything you like

Hosts: localhost

Permissions:

Select table data

Insert table data

Update table data

Delete table data

Create tables

Drop tables

Manage indexes

Alter tables

Create temp tables

Lock tables

Restart Mysql and Apache

```
service mysqld restart
service httpd restart
```

([Thanks!](#))

Finalize

Make a new virtual host for your domain.

In Webmin, go to:

-> Servers

-> Apache Webserver

-> Create virtual host

Handle connections to address = Specific address = your domain name

Document Root = path to your installation of Drupal

Copy directives from = Nowhere

Safe and "Apply Changes" in the upper right.

Open your new virtual server

-> Edit Directives

and add

AllowOverride All

just above ""

Safe and "Apply Changes" in the upper right.

And your server should be working.

Now those pesky e-mails.

Installing Drupal on CentOS 5/Red Hat/Fedora

This is a general description of how to install Drupal on a Linux CentOS/RedHat/Fedora server with root access. Your mileage might vary somewhat depending on your particular environment.

Comments on various CentOS/RedHat/Fedora combinations and set ups are most welcome in this section. This is based on a CentOS 5/RedHat installation. Comments that only concern Fedora should be added to and not displace CentOS/RedHat information.

1. Upgrade the operating system
yum update
2. Install common programs
yum install php mysql-server php-mysql php-mbstring php-gd
3. Download and install Webmin (optional)
wget <http://prdownloads.sourceforge.net/webadmin/webmin-1.440-1.noarch.rpm>
wget <http://webmin.com/jcameron-key.asc>
rpm --import jcameron-key.asc
yum install webmin-1.440-1.noarch.rpm
4. Reboot
5. Login to webmin through your browser. On a virtual host, how to do so may be non-obvious, but your host should provide details somewhere in the documentation of how to access your host through a url.
6. In webmin, go to "Bootup and Shutdown" and enable the httpd and mysqld daemons.
7. Create a new user. I made the home directory "/var/www/html" and the shell "/bin/bash".
8. Edit the httpd.conf file. I did this through webmin in the "Apache Webserver" section by clicking on the "Global configuration" tab and then going to "Edit config files". Change the line "Options Indexes FollowSymLinks" to "Options Includes FollowSymLinks" or even something more liberal.

While you are there, go to the following section:

```
# AllowOverride controls what directives may be placed in .htaccess files.
# It can be "All", "None", or any combination of the keywords:
#   Options FileInfo AuthConfig Limit
#
AllowOverride None"
```

Change "AllowOverride None" to "AllowOverride All" if you want clean urls.

9. In webmin, go to "Apache Webserver" and create a virtual host if you want to host multiple hosts on the same server instance.
10. Create a new empty database in mysql. I did this through webmin. You should also add passwords for the root mysql user as well as a regular user for security reasons.
11. In my case, I changed the permissions of /var/www/html to my user and the apache group.
chown -R myuser /var/www/html
chgrp -R apache /var/www/html
12. Get Drupal and unpack it in the /var/www/html/ directory or if you are using virtual hosts in /var/www/html/example.com
wget <http://ftp.drupal.org/files/projects/drupal-6.6.tar.gz>
tar -xvzpf drupal-* --strip-components=1

```
cp ./sites/default/default.settings.php ./sites/default/settings.php
```

13. If you want a "files" directory for file uploads, do the following in your Drupal root directory


```
mkdir ./sites/default/files
```
14. The following permissions should allow you complete access to Drupal though your regular user account except for the ./sites/default folder which should be set to a Drupal's default upon install.


```
chgrp -R apache .
chown -R apache ./sites/default
chgrp -R myuser ./sites/default
chmod -R g+w sites/default
```
15. Reboot
16. Install Drupal as normal by going to the url of your home page

Drupal 6.x on Fedora 9 using PostgreSQL

Installing Drupal on Fedora can be challenging because of the different way in which Fedora installs Drupal and PostgreSQL, so the standard ways that you will find in the generic README files and even on the official websites won't work. Artificially trying to make your installs like you see on the sites is fraught with problems.

At least parts of this tutorial can be found elsewhere, but this page represents one single place for all your needs for Fedora 9 and PostgreSQL.

PostgreSQL on Fedora

- PostgreSQL installs to `/var/lib/pgsql`
- the superuser `postgres` is created automatically, without a password. Once you `su root`, you can then `su postgres` (or another other user) without a password.
- although `root` cannot use PostgreSQL, `root` initializes the database and starts PostgreSQL

Starting PostgreSQL

1. If you already have PostgreSQL running, skip to step 5.
2. As `root`, in a command line, enter `service postgresql initdb`. (Depending on how you have paths set up, you may have to use `/sbin/service` for the command.) You should get an OK as feedback when it's done.
3. Still as `root`, enter `service postgresql start`. Once again, wait for the OK.
4. To have PostgreSQL start up on boot, as `root` enter `chkconfig --level 345 postgresql on`. There is also a GUI way of starting/stopping services, but I prefer the command line.
5. Now you can `su postgres` to `createuser` – we'll call him *drupal*:


```
createuser --pwprompt --encrypted --no-adduser --no-createdb drupal
```

 which leads to a prompt for the password
6. Create the database (as `postgres`) – which we'll call *drupaldb*:


```
createdb --encoding=UNICODE --owner=drupal drupaldb
```

It used to be that you would get some feedback on the command line after `createuser` and `createdb`, but that doesn't seem to happen anymore.

Now Edit `pg_hba.conf`

You will need to do this to allow Drupal to access your database.

As `root`, use your favorite editor to edit the file `/var/lib/pgsql/data/pg_hba.conf`

Go down to this section:

```
# TYPE      DATABASE          USER                CIDR-ADDRESS          METHOD
# "local" is for Unix domain socket connections only
```

```
local  all          all          ident sameuser
```

After this above line enter:

```
host      drupaldb      all          127.0.0.1/32      md5
```

Save the file, then, as root, `service postgresql restart`

Drupal on Fedora

- Drupal installs to `/usr/share/drupal`
- Inside Drupal, the `sites` links to `/etc/drupal`, `files` links to `/var/lib/drupal`
- You use `httpd` to control access to Drupal. This is what keeps visitors from getting access to your system, since `httpd` forbids going up above allowed directories by default.
- The base setup of `httpd` and Drupal is set in `/etc/httpd/conf.d/drupal.conf`, where you find

```
Alias /drupal /usr/share/drupal

<Directory /usr/share/drupal/>
    Order Deny,Allow
    #Comment the following line and uncomment the next for public use
    Deny from all
    #Allow from all
    #Uncomment the following line for setup
    #Allow from 127.0.0.1
    AllowOverride All
</Directory>
```

The first line is the link. On your machine, go to a browser, enter `http://localhost/drupal`, you will access Drupal, but not with a file like this one – instead you get the *403 Forbidden* page. Remove the '#' from the `Allow from 127.0.0.1` (this is what *uncomment* means, add the '#' to *comment* the line.) Needless to say perhaps, but you need to do this as root, and when you edit this file you have to stop and start `httpd` again, the quickest way being `service httpd restart`.

More steps before diving into Drupal

You will get permissions error messages from Drupal as it tries to establish your site, unless and until you do the following:

- As root, go to `/etc/drupal` and `chmod 777 default`.
- Enter the `default` directory, and `cp default.settings.php settings.php`.
- `chmod 666 settings.php default.settings.php`
It's not quite clear whether you also need to make `default.settings.php` writable, perhaps `644` is Ok for `default.settings.php`.
- *You do not need to restart httpd after these steps.*

Now, Go For It!

At this point, you should be able to enter `http://localhost/drupal` in your browser, and bring up the forms for starting up your Drupal site, then entering your basic administrator information, after which you're off and running!

The last step, as you will read when you're all done, is to go back to `/etc/drupal/` and

```
chmod 755 default
and inside the default directory,
chmod 644 settings.php default.settings.php
```

A Final Note about SELinux

It's possible there will not always be problems with SELinux, but for many tasks in Fedora 9, it simply helps to change the enforcing mode of SELinux to *Permissive*. Ideally, some day there may be some helpful documentation about SELinux to create some targeted policies that work, but for now, this seems to be what many are doing, and not specifically just about Drupal.

How to Install Drupal on your Home Windows Computer

Using the following steps, I successfully installed Drupal 6.16 on computers running Windows 7, 32-bit, with Explorer 8 and Norton 360 security. Drupal - or any content management system for that matter - isn't easy to install because of the database and webserver component that are required, which are not normally installed on desktop computers.

***Please Note: These instructions are only intended for installing Drupal on your home computer to try out Drupal. DO NOT use these instructions to install Drupal for a website on the internet. Always make sure you have the latest Windows updates installed on your computer, and have an anti-virus program installed, updated and running. ***

PART I – Getting and Installing XAMPP

1. Remove any previous versions of XAMPP on your computer only if you tried installing it before. Go to "All Programs" > "XAMPP" > "Uninstall".
2. Execute the following steps
 - A.) Go to www.apachefriends.org.
 - B.) Click on XAMPP (on the icons in the navigation bar on the top of the website).
 - C.) Click on XAMPP for Windows.
 - D.) You'll see a list titled "Jump off Point".
 - E.) Click on "XAMPP Lite".
 - F.) Click on the red EXE link (this means self extracting file).
 - G.) An ad will pop up on the next page; ignore it.
 - H.) Look above the ad, and under the Source Forge logo, you will see: "Your XAMPP download will start shortly...Problems with the download? Please use this direct link or try another mirror."
 - I.) Click on "Use this direct link."
 - J.) A "Getting Information" "Open" "Open Folder" "Cancel" box will pop up.
 - K.) A "Do you want to run or save this file?" box will pop up on top of the "Getting Information" box (I think this is a Norton thing. Without Norton, you just might get the "Getting information" box).
 - L.) Select "Run" or "Open" – whichever option it gives you.
 - M.) XAMPP will want to extract into the "C" folder . Let it.
 - N.) When it finishes downloading, a black box will pop up and ask you questions. Keep hitting enter. Don't change anything.
 - O.) On the final black box, where you have to enter something to make it go away, type a lower case "x" and then enter.
 - P.) You now have XAMPP on your computer, but DON'T PLAY WITH IT YET. There's other stuff to do first.

PART II – Getting and Installing Drupal

- A.) Go to www.drupal.org
- B.) Click on "Download Drupal 6.16", or whichever is the latest version
- C.) Click on "Open"
- D.) Drupal will download.
- E.) Pay attention which folder Drupal is downloaded to.
- F.) Use Windows Explorer (not the browser, but the file viewing program that is under All Programs > Accessories > Window Explorer) to find the Drupal file. You might have to check folders under documents, downloads, etc. If you can't find it, create a folder in documents and download it again

- F.) Close the window.
 G.) Open a browser.
 H.) Type <http://localhost/drupal> in the address bar (If you only type in <http://localhost> you'll get the XAMPP program)
 I.) A Drupal screen should open. If not, check your XAMPP controller box (double click the XAMPP icon on your desktop if you need to open it) and make sure both XAMPP and MySQL are started.
 J.) Select install in English.
 K.) Drupal will verify your requirements. Give it a minute.
 L.) In the Set Up Database screen, enter drupal (all lowercase) for Database Name. For E-mail address, enter your e-mail address (you won't be able to use the e-mail feature right now, but just put it in there) and TRICKY PART: for Database user name enter root (all lowercase) and for Database password LEAVE THIS BLANK. Do not enter anything.
 M.) Administrator Account Setup: this is what you will use to access the site on your home computer. I just used my first name as my user name, and then put a 1 after my name for my password. It's easy to remember.
 N.) Once you're finished, you will get an error message about your e-mail account. Ignore that for now. Your Drupal install is now complete and usable, and I would highly recommend Drupal for Dummies (<http://drupalfordummies.com/>) to learn what to do next.

To access Drupal whenever you login to your computer, open a browser window, go to <http://localhost/drupal> and enter your user name and password. Remember to make sure both Apache and MySQL are running in the XAMPP controller box (double click on the desktop XAMPP icon).

Enjoy!

How to redirect users after login in drupal?

Step 1 : Enable the trigger module if you haven't done so. (To enable the trigger module, go to administer->site building->module. check the trigger module and press save.)

Step 2 : Go the administration panel and click on action. Under **Make a new advanced action available**, select Redirect to URL... and hit create.

Enter a **description** and the **url** to redirect to. You can use internal url like node/123 to redirect the user on login.

Press Save.

Step 3 : Go to the administration panel and click on triggers under site building. Select users in the tab.

Under **Trigger**: After a user has logged in, select the trigger you have just created and hit assign.

Logout and log back in to test.

Mirroring a drupal-site for offline viewing using a Perl script

(also in German at <http://www.drupalcenter.de/handbuch/26659>)

This description describes the steps done to produce a copy of a live Drupal site that could be viewed "offline" e.g. from a CD by just using a browser accessing static html files without the need for a web-server or a database.

Such a copy could be used for archiving purposes, if the online access to the site isn't always possible, or - as in the case that triggered the creation of the script described in here - to have a portable "snapshot" version of the site's content for reviewing purposes.

Also - the approach described here is not a general purpose mirroring technique - there might be easier ways to achieve this using tools like e.g. wget (<http://www.gnu.org/software/wget/>), httrack (<http://www.httrack.com/>) or Drupal's Boost (<http://drupal.org/project/boost>) or HTML export (http://drupal.org/project/html_export) modules but they might not allow for as close control of the result and the mirroring process as the approach described below.

The situation:

- an SSL secured Drupal site requiring log in to access any content
- Drupal is installed in a subdirectory "secrets" of the webroot being pointed to by the URL <https://www.mysecrets.com>
- the site's content is organized as a Drupal "book"
- file attachments are all placed in a "private" directory
- additional modules "private upload", "private download", and "custom filter" are being used to prevent the "private" directory being spied into.
- the result of the "mirroring" process should be a set of HTML, CSS, image, and attachment files, that reflect the current contents of the drupal site statically
- all URL, directory names, user IDs, passwords, etc. for this description are fictitious, any similarities to real domains are merely accidental and by no means intended.
- this description and the Perl script snippets are provided under the conditions of the GNU Public License (GPL), which could be found in lots of places on the web and is not attached here for simplicity purposes.

Prerequisites:

In order to use the script being developed in this description you'd need Perl plus some Perl packages containing HTML::TreeBuilder for parsing and manipulating html content, LWP as the "browser", and in case https is used also libcrypt-ssleay-perl to be able to use https together with LWP.

Disclaimer:

This description is not intended to teach Perl. It is rather assumed, that the reader has some Perl knowledge already. And it might not be the best possible and most elegant or efficient way to write Perl scripts - I'm more or less an Perl newbie myself. Apologies if the coding style is offending - its not meant to hurt anyone.

The method described below might not be applicable to all kinds of Drupal sites, but should at least work for simple things like e.g. Drupal books.

The script:

Lets start with a more or less typical declarative section of the Perl script containing the more globally used variables:

```

#
use strict;
use warnings;
use LWP 5.64;
use HTML::TreeBuilder;

my $browser = LWP::UserAgent->new;
my %urls2retrieve; # a hash to store what we still have to retrieve
my %urlsretrieved; # another hash containing what we retrieved already (to avoid
duplicate retrieval)
my $outfile;
my @urlkeys;
my $nexturl2retrieve;
my $typeofnexturl;

```

Now - as we need to log on to the Drupal site and do the retrieval work while being logged on, we need to keep the cookies that the site sends us to maintain our "logged on" state across the duration of this script. Then we do our log-on.

```

# the cookie store is in memory
$browser->cookie_jar({});
# Initial contact
my $host = 'https://www.mysecrets.com'; # the host
my $url = $host . '/secrets'; # the subdirectory Drupal is installed in
my $response = $browser->get($url);
die "Can't get $url -- ", $response->status_line
unless $response->is_success;
my $html = $response->content;
# now post the login
$response = $browser->post( $url . "/user",
[
'name' => 'JamesBond', # the log-in ID
'pass' => 'agent-007', # the password for the above ID
'op' => 'Log%20in',
'form_id' => 'user_login'
],
);
$html = $response->content;

```

Now the script is logged in and can start retrieving the contents. The site is set up so that node 3 is the starting screen, that you get, when logged in, so we start with retrieving node 3 and go down all we can find from there.

The hash %urls2retrieve is used in a way, that the key contains the partial URL and the corresponding value contains the type of what the URL point to. We have types of "node" and "css" where we not only need to "get" them from the site, but also need to do some processing, and a type of "file", which describes something that we just need to retrieve and store as it is.

```

# initialize the urls to retrieve
$urls2retrieve{"secrets/?q=node/3"} = 'node';

# iteratively retrieve everything
while (%urls2retrieve) {
@urlkeys = keys(%urls2retrieve);
$nexturl2retrieve = $urlkeys[0];
$typeofnexturl = $urls2retrieve{$nexturl2retrieve};
if ($typeofnexturl eq "node") {
RetrieveNode();
ProcessNode();
} elsif ($typeofnexturl eq "css") {
RetrieveCss();
ProcessCss();
} elsif ($typeofnexturl eq "file") {
RetrieveFile();
}
$urlsretrieved{$nexturl2retrieve} = 1; # record this url as retrieved
delete $urls2retrieve{$nexturl2retrieve}; # don't handle this anymore
} # end while (%urls2retrieve)

```

That is basically it. We're done and have retrieved the entire site. The real work is being done in the subroutines described below.

This routine simply retrieves a node and puts the entire html page into the \$response variable.

```

sub RetrieveNode {
print "retrieving: $nexturl2retrieve -----\n";
my $url = $host . $nexturl2retrieve;
$response = $browser->get($url);
die "Can't get $url -- ", $response->status_line unless $response->is_success;
} # end of RetrieveNode()

```

Now the a bit more complicated routine to go through the retrieved page and it's html structure, extracting things that we will have to retrieve next and modify the related references to our intended (flat) target structure.

```

sub ProcessNode {
print "processing: $nexturl2retrieve -----\n";
$nexturl2retrieve =~ /.*/\(.*/;
$html = $response->decoded_content; # to get e.g. all the non-standard-ascii characters
like german umlauts right
my $root = HTML::TreeBuilder->new();
$root->parse_content($html);

```

So the entire page is parsed and now we can start looking for things that point to something, that we need to retrieve too or modify. We're starting with the "link" tags.

```

my @links = $root->look_down('_tag', 'link'); # get a list of "links"

```

Now we have a list of all the "links" in this page, which we're going to process one by one. Three different types of links are recognized in the script below. If the "link" points to another node, then first we check, whether this other node has been retrieved already or has been noted for retrieval earlier, if not, then we add this other node onto our hash %urls2retrieve. Then we change the reference to the node from the ".../?q=node/" pattern to a simple "node.html" filename.

In a similar way link references to CSS files and other files are handled.

```

foreach my $link (@links) {
my $linkhref = $link->attr('href');
if ($link->attr('href') =~ /\?q=node/ ) {
# push the link to the to-download list
if (! defined $urlsretrieved{$linkhref} && ! defined $urls2retrieve{$linkhref}) {
$urls2retrieve{$linkhref} = 'node'; }
# change the link for local filesystem use
$link->attr('href') =~ /\?q=node\(.*/;
my $nodenum = $1;
$link->attr('href', "node" . $nodenum . ".html" ); #set a new href
} elsif ( $link->attr('href') =~ /\.css\?y/ ) {
# push the link to the to-download list
if (! defined $urlsretrieved{$linkhref} && ! defined $urls2retrieve{$linkhref}) {
$urls2retrieve{$linkhref} = 'css'; }
# change the link for local filesystem use
$link->attr('href') =~ /.*/\(.*/.css\?y/;
my $css = $1;
$link->attr('href', $css . ".css?y" ); #set a new href
} else {
# push the link to the to-download list
if (! defined $urlsretrieved{$linkhref} && ! defined $urls2retrieve{$linkhref}) {
$urls2retrieve{$linkhref} = 'file'; }
# change the link for local filesystem use
$link->attr('href') =~ /.*/\(.*/;
my $file = $1;
$link->attr('href', $file ); #set a new href
}
}
}

```

Now that we have handled all "link" tags we do the same kind of processing for anchor tags. Due to the use of some additional Drupal modules (Private Upload, Private Download, Custom Filter) special handling needs to be done for attachment URLs pointing to the "private" folder. References to anything else than a node or a file in private will be removed. This includes all the usual links to the administrative pieces of the Drupal site as they don't make a lot of sense in a "static" copy of the site's content. Furthermore any references to the "starting page" (in our example <https://www.mysecrets.com/secrets>), which is used e.g. in the site's header and in the breadcrumb navigation, is changed to point to our main page (node3.html).

We're also not following any reference pointing to somewhere else than the site being mirrored - i.e. external links.

```
my @as = $root->look_down('_tag', 'a'); # get a list of "as"
foreach my $a (@as) {
    my $ahref = $a->attr('href');
    next if (! defined $ahref);
    if ($a->attr('href') =~ /\?q=node/ ) {
        # push the ahref to the to-download list
        if (! defined $urlsretrieved{$ahref} && ! defined $urls2retrieve{$ahref}) {
            $urls2retrieve{$ahref} = 'node'; }
        # change the a for local filesystem use
        $a->attr('href') =~ /\?q=node\/(.*)/;
        my $nodenum = $1;
        $a->attr('href', "node" . $nodenum . ".html" ); #set a new href
    } elsif ( $a->attr('href') =~ /\?q=system\/files/ ) {
        # files in private
        if (! defined $urlsretrieved{$ahref} && ! defined $urls2retrieve{$ahref}) {
            $urls2retrieve{$ahref} = 'file'; }
        # change the a for local filesystem use
        $a->attr('href') =~ /\.*\//(.*)/;
        my $file = $1;
        $a->attr('href', $file ); #set a new href
    } elsif ( $a->attr('href') =~ /\?q=/ ) {
        # ?q=anythingotherthannode will be removed
        $a->delete();
    } else {
        # check for external link
        if ($ahref =~ /\$host/ ) {
            # push the link to the to-download list
            if (! defined $urlsretrieved{$ahref} && ! defined $urls2retrieve{$ahref}) {
                $urls2retrieve{$ahref} = 'file'; }
            # change the link for local filesystem use
            $a->attr('href') =~ /\.*\//(.*)/;
            my $file = $1;
            $a->attr('href', $file ); #set a new href
        } elsif ($ahref eq '/secrets/') {
            # redirect it to our starting page, which is node 3
            $a->attr('href', 'node3.html' ); #set a new href
        } else {
            print "--- external link: $ahref\n";
        }
    }
} # end foreach @as
```

Now that we have handled the anchor tags, we need to look for "img" tags and process any reference to images we find there. Again - as this is a secured site where only logged-in users are allowed to access any content (including images), such content needs to be retrieved from the "private" folder.

```
my @imgs = $root->look_down('_tag', 'img'); # get a list of "img"
foreach my $img (@imgs) {
    my $src = $img->attr('src');
    if ($src =~ /\?q=system\/files/ ) {
        # files in private
        if (! defined $urlsretrieved{$src} && ! defined $urls2retrieve{$src}) {
            $urls2retrieve{$src} = 'file'; }
        # change the a for local filesystem use
        $src =~ /\.*\//(.*)/;
        my $file = $1;
        $img->attr('src', $file ); #set a new src
    } elsif ( $src =~ /\secrets\/sites\/default\/files/ ) {
        if (! defined $urlsretrieved{$src} && ! defined $urls2retrieve{$src}) {
            $urls2retrieve{$src} = 'file'; }
        # change the a for local filesystem use
        $src =~ /\.*\//(.*)/;
        my $file = $1;
        $img->attr('src', $file ); #set a new src
    } else {
        # check for external link
        if ($src =~ /\$host/ ) {
            if (! defined $urlsretrieved{$src} && ! defined $urls2retrieve{$src}) {
                $urls2retrieve{$src} = 'file'; }
            # change the link for local filesystem use
            $src =~ /\.*\//(.*)/;
            my $file = $1;
            $img->attr('src', $file ); #set a new href
        }
    }
}
```

```

    } else {
        print "--- unhandled img src: $src\n";
    }
}
} # end foreach @imgs

```

As now all the contents of this node / page has been scanned for references to other places and modified appropriately we can save it to a file. The file name is chosen to be "node" followed by the number of the node and then the extension ".html".

```

$outfile = '';
#determine the name of the file to be written
if ($nexturl2retrieve =~ /\?q=node\/\//) {
    $nexturl2retrieve =~ /\?q=node\/(.*)\/ ;
    $outfile = "node" . $1 . ".html";
    unlink($outfile);
    open (OUTPUT, ">$outfile") or die "cannot open $outfile for output \n";
    my $x = $root->as_HTML();
    print OUTPUT $x;
    close OUTPUT;
}
$root->delete(); # clear this html tree
} # end of ProcessNode()

```

This concludes the ProcessNode routine. The next is the RetrieveCss routine, which is similar to the RetrieveNode routine except that we directly retrieve the CSS file to a local file.

```

sub RetrieveCss {
    print "retrieving: $nexturl2retrieve ----- \n";
    my $url = $host . $nexturl2retrieve;
    $nexturl2retrieve =~ /\.*\//(.*)\.css/;
    my $cssfile = $1 . ".css";
    $response = $browser->get($url, ':content_file' => $cssfile );
    die "Can't get $url -- ", $response->status_line unless $response->is_success;
} # end of RetrieveCss()

```

The retrieved CSS file could contain url references to graphics being used (e.g. for menu bullets or theme-coloring), which need to be retrieved too. Processing needs to consider the relative placement of the referred graphic, which is relative to the CSS file's location.

```

sub ProcessCss {
    print "processing: $nexturl2retrieve ----- \n";
    my $line;
    my $cssurl;
    my $url = $host . $nexturl2retrieve;
    $nexturl2retrieve =~ /\.*\//(.*)\.css/;
    my $cssfile = $1 . ".css";
    open (CSSFILE, $cssfile);
    while (<CSSFILE>) {
        chomp;
        $line = $_;
        if ($line =~ /url\((.*)\)/ ) {
            $line =~ /url\((.*)\)/ ;
            $cssurl = $1;
            $nexturl2retrieve =~ /(\/.*\/)/ ;
            my $retfromdir = $1;
            $cssurl = $retfromdir . $cssurl;
            # push the link to the to-download list
            if (! defined $urlsretrieved{$cssurl} && ! defined $urls2retrieve{$cssurl}) {
                $urls2retrieve{$cssurl} = 'file';
            }
        }
    }
    close (CSSFILE);
} # end of ProcessCss()

```

The only remaining part is to retrieve plain attachment files and graphics files, which is handled in the following routine.

```

sub RetrieveFile {
    print "retrieving: $nexturl2retrieve ----- \n";
    my $url;
    if ($nexturl2retrieve =~ /$host/) {
        $url = $nexturl2retrieve;
    } else {
        $url = $host . $nexturl2retrieve;
    }
}

```

```

return if ($nexturl2retrieve eq "/secrets/");
$nexturl2retrieve =~ /\.*\/(.*)/;
my $file = $1;
$response = $browser->get($url, ':content_file' => $file );
print "Can't get $url -- \n", $response->status_line unless $response->is_success;
} # end of RetrieveFile()

```

This concludes my description on how the contents of a Drupal site could be mirrored for offline viewing - or at least how I did it for some sites I take care of.

Feel free to adapt it to any similar situation, where you'd like to have some control on the mirroring process. Any comments are welcome.
(Richard)

Posting video from mobile phones into Drupal

Posting video from mobile phones to drupal website is reasonable easy and there are few methods of doing it. Below we cover the "independent" path, to avoid any proprietary mms gateways: just simple email with attachment.

Introduction

Posting video from mobile phone to your drupal website via email requests from user to configure a mobile phone with smtp settings and reasonable Internet connection (3g). Costs depends from your mobile phone provider: at some networks you pay only for transfer volume, at others you might pay for "event" of establishing connection. Video from mobile phones is usually around 0.5 MB is size, so limit is in size of allowed attachments: on decent phones you can attach up to 1 MB.

On server side you will probably need to setup smtp server and provide access to it to users of your website. Transcoding of posted video employs ffmpeg, its standard software on proper hosting server.

Process Overview

1. Email with video attachment from mobile phone is sent to a defined mailbox
2. Drupal website downloads Mail on cron (mailhandler module)
3. Emails are turned into nodes with videos as attachments
4. Media Mover runs on cron, trans-coding mobile video formats and creating thumbnails
5. Transcoded flash video files are added to cck file field
6. Thumbnail added to file field
7. Nodes are themed using swftools to display video using "JWplayer"

Modules used

- Mailhandler
- Mailssave
- Mediamover
- SWFTools
- FFMPEG_wrapper
- cck

- filefield
- imagefield

Implementation

Node Setup

1. Define a node type that will be used to "host" your video data. Call it for example "video" type
2. Add a cck field for storing your flash video
3. Set display settings to use video player
4. Add a cck file field for thumbnail
5. At time of writing Imagefield is not properly released for D6, so Media Mover is unable to add filed to image fields properly. This will mean that image thumbnails will need to be manually themed for now.
6. Assign permissions to allow anonymous users to create and view this node type (but not edit it)
7. if users are already members, and they email in from their registration address, video will be owned by them, so then they will be able to add metadata.
8. Allow attachments for this node type

File Upload settings

Set upload settings, so that every user posting via mail (usually anonymous) will be able to publish video type:

- upload file size
- upload file extensions (jpg jpeg gif png txt doc xls pdf ppt pps odt ods odp avi mov wmv mpeg mp4 mpeg2 dv 3gp 3g2 mpeg4 flv mpg)

Note: media mover is using the upload module to insert transcoded files into cck filed: remember to allow proper size and extension (flv).

Mailhandler

Download and install mailhandler and mailsave modules. (There is mail module for media mover but it makes media mover process too complex). However, you can use Media Mover's implementation of Mail Handler to keep everything inside of Media Mover.

Set up mail box

Set up a new mailbox in admin/content/mailhandler page

```
E-mail address: mobile@example.org
Folder: INBOX
POP3 or IMAP: POP3
Mailbox domain: mail.example.org
Mailbox password: xxxx
Security: Disabled
Send error replies: Disabled
```

There is an area in the mailbox entitled "Default Commands" which is IMPORTANT towards assigning where you want your mail to go and adjust other default settings that aren't sent in emails but are standard on drupal nodes.

Working settings are below, tune it for your website:

```
type: video
promote: 1
comments: 1
published: 1
```

For more info on these settings see [this page](#)

Another couples of options to note below:

```
Delete messages after they are processed? Deselected
Cron processing: Disabled
```

However in a **production** site you will want to set these as

```
Delete messages after they are processed? Selected
Cron processing: Enabled
```

Now mailhandler should be able to receive emails from your admin user and create nodes from them.

In order to make sure your email attachments are stored on the site and attached to a node, adjust settings for this and set who can email files to your site (see the next couple of entries).

Permission settings

The mailhandler module uses the email address that the message is sent from to determine which user the message is coming from. If this email address does not match the email address of a user on your website then the message will be assigned as being from an anonymous user

Thus we can use the Drupal Core permission roles to determine who can send emails, attachments etc. to the website.

For our site we wanted anonymous users to be able to send in emails with attachments so in admin/user/permissions we selected the following options for anonymous users

```
mailsave module
* access content
* save attachments

node module
* create video content

upload module
* upload files
* view uploaded files
```

Transcoding with ffmpeg

To convert video from attachments into flv format you will need 2 pieces: ffmpeg binary installed on server where your drupal is installed and ffmpeg wrapper module (covered below). Ffmpeg is a standard open source package for multimedia handling available on many distributions so ask your host if you dont have it. Note it might be very tricky to make it running:

- Standard debian package is striped from proprietary formats so you will not be able to transcode from/to avi, mp4, rm flv or mp3 but developer of Media mover is providing [<http://mediamover.24b6.net/content/ffmpeg-binary-static-binary>]. You still need vhooks libs installed (part of ffmpeg) on system.

Usually shared hosts don't let your php to run shell commands, if you run into passthru limit or php error 127, it means you are in shell jail. Explain to your host what you are going to do, if it not helps, change your host.

- If test transcode is not running, try "flush all caches" from [http://drupal.org/project/admin_menu admin menu]. Its strange behavior , but it helped us.

To transcode in drupal you need to install [http://drupal.org/project/ffmpeg_wrapper ffmpeg wrapper module] and test it via settings in admin menu. Best way to test is to trans code small video file with common extension like 3gp or avi and media encoded with popular codec. Extension of the test file have to match one from list available formats of ffmpeg, other wise your test will fail (it might happend for example with .mpg files).

MediaMover

Media mover does the video conversion in a set of stages. Each stage processes new nodes with video attachments. (These nodes will have been created by Mailhandler from email set to the video inbox.)

Media mover rules should be weighted so that they run in order. This will allow nodes to be made public only when all content has been processed.

Pass 1: convert video to standard format

Set up a media mover rule to do the following:

- harvest nodes with attachments
 - attachments should have allowed video extensions
 - exactly which extensions are possible will depend on your ffmpeg setup
- Process your incoming attachments to transcode video to flv
- store transcoded video in a cck file field on the original node

Pass 2: Create a thumbnail

- Harvest nodes with video attachments - exactly as above
- Process your video to create thumbnails
- Store the thumbnail in a image field
- Complete your processing by publishing your node and promoting to front page if you want

SWFTTOOLS setup

1. configure flash video playing with swfttools
2. download the swfobject code
3. download player code (JWplayer)
4. Set up your swf file handling to use the player
5. add a custom skin for the player if you want to - download from JWplayer site

Final notes

Security

Providing post-in gateway to your website might lead to **security or spam issue**. When you provide public smtp server to your users its god idea to limit allowed destination email addresses

to only one, with your drupal website checks.

Credits

The document have been put together as part of Transmission Network`s [Drupal Video documentation](#) effort at the [Drupal Transmission Sprint 2009 uk](#)

In creation of this tutorial have been involved [the greenman](#), [benced](#), [szczym](#), [Mick Fuzz](#) and others.

Remove Duplicate Nodes based on title

I have tested only once, and it did work. But use it on your own risk.

Based on code from [Robert Douglass](#).

I have added few lines to make it copy-paste-able to a module.

Simply copy this to dedupe.module, and create module named "dedupe". More info about creating modules can be found [here](#).

```
<?php
/**
 * @file
 * Remove Duplicate Nodes.
 */
/**
 * Implementation of hook_menu().
 */
function dedupe_menu() {
  $items['admin/content/dedupe'] = array(
    'title' => 'Dedupe',
    'description' => 'Delete Duplicate Nodes',
    'page callback' => 'drupal_get_form',
    'page arguments' => array('dedupe_content_command'),
    'access arguments' => array('administer site configuration'),
    'type' => MENU_NORMAL_ITEM,
  );
  return $items;
}
/**
 * Build Form
 */
function dedupe_content_command() {
  $options = node_get_types('names');
  $form['dedupe_node_types'] = array(
    '#type' => 'select',
    '#title' => t('You can select content type from which duplicates are removed.'),
    '#options' => $options,
    '#default_value' => variable_get('dedupe_node_types', array('page')),
    '#description' => t('Duplicate content from the selected content types
will be deleted.'),
  );
  $form['submit'] = array(
    '#type' => 'submit',
    '#value' => t('Dedupe'),
  );
  return $form;
}
/**
 * Call delete function and set message.
 */
function dedupe_content_command_submit($form, $form_state) {
  $type= $form_state['values']['dedupe_node_types'];
  $dedupe_m = 'Duplicates deleted from content type '.$type;
  dedupe_delete($type);
}
```

```

    drupal_set_message(t($dedupe_m));
  }
  /*
  * Delete duplicates from selected content type, based on title. By Robert Douglass.
  * @see http://robshouse.net/blog-post/remove-duplicate-nodes-dedupe-based-title
  */
function dedupe_delete($type) {
  $previous = array();
  $result = db_query("SELECT nid, title FROM {node}
  WHERE title IN
    (SELECT title FROM {node}
    WHERE type = '%s'
    GROUP BY title HAVING count(*) > 1)
  ORDER BY title, created DESC", $type);
  while ($row = db_fetch_array($result)) {
    if ($row['title'] == $previous['title']) {
      node_delete($previous['nid']);
    }
    $previous = $row;
  }
}
}

```

Revisoning, moderation and content publication workflows

The [handbook page on Revisoning](#) will have given you a basic understanding of how you can set up a **workflow to create, moderate and publish content revisions** using the [Revisoning](#) module.

The tutorials below go further:

- On bigger sites, with many authors producing lots of content, you may want to associate categories with both roles and content, so that only “sports authors” have access to “sports” content. See [Revisoning workflow for categorised content](#)
- You may want to grant exclusive access to content based on the state it's in, e.g. while authors are still working on a new revision, moderators shouldn't be allowed to touch it also. Conversely, while a moderator is reviewing revisions, authors should not be able to apply further edits. Enhance your revisoning and publication workflow with [state-based access control](#)

Translation, multilanguage content, and internationalization

Translating the user interface

You can translate the interface text (like the “Log in” button and the “Add new comment” text). For many languages, there are complete or partly complete translations available. You need:

- [Locale](#) (core module)
- [Translations for download](#)

See also:

- [Translator's guide](#) (how to contribute to translations)
- [Localization client](#) (contributed module, makes interface translation easier)

- [Localization server](#) (community localization editor; including list of the translation teams which use the server)
- [Concept article: Translations - Drupal 5](#)

Right to left support

If you want a website with right-to-left text direction, you'll need a theme which supports this, like the core theme Garland for 6.x, or [Garland BIDI](#) for 5.x. For more RTL themes, look at the [theme download page](#).

Multilanguage content

If you want a site with content in more than one language, you have several options. The main ones are:

- [Content translation](#) (core module, version 6.x only)
- [#translatable](#) (contributed module)
- [Internationalization](#) (contributed module) ([handbook pages](#))
- [Localizer](#) (contributed module) ([handbook pages](#))

See also:

- [All contributed modules for multilingual functionality](#)
- [Tutorial - Building a multi-language site](#)

Other internationalization challenges

There are numerous other challenges when creating websites for a specific part of the world or language group, like date formats, alphabetic sort of special characters, currencies, and tax calculations. See the links below for places to look for more information, and please add a sub page if you have something to share.

Read more:

- [Localization API](#) (translation and localization information for developers)
- [Translation support forum](#)
- [Troubleshooting](#)
- [List of support sites in various languages](#)
- [groups.drupal.org - Internationalization](#) (for content translation and other internationalization issues)
- [groups.drupal.org - Translation](#) (for interface translation)
- [Translation mailing list](#)
- [groups.drupal.org - Various groups for specific geographical areas](#) (some groups for specific languages are listed under "Working groups", not "Geographical")

step by step Website Tutorial (Not done yet!!!!!!)

This tutorial covers the creation of a website using drupal.

I am creating a site for learning purposes and wanted to document how I did it. This will help other people understand the processes I performed. This will also help me remember the things I have done especially when I need to explain them for several times in the future.

So here it goes:

background info:

The site I will be creating is going to be based on the needs of my employer Uroboros glass, and hopefully when I am fully finished with it, I can present it to them and they will let me implement it. With a raise??!!

The first modules you should install are: [Advanced Help](#), [Administration Menu](#)

The needs of this site are as follows.

General modules: [Nice Menus](#), [Rules](#), [Taxonomy](#), [Views Slideshow](#)

Multiple page status: [Twitter](#)

WYSIWYG HTML editor: [IMCE](#), [FCKeditor](#)

Product Catalog: [Views](#), [CCK](#), [Tokens](#), [ImageField](#), [FileField](#), [FileField Paths](#), [ImageField Tokens](#)

Glass Artists Profiles: [Advanced Profile](#), User gallery, User tutorials/How-to's, User video tutorials

Retail online store: [Ubercart](#)

Whole-sale point of sale software: I don't know if this will ever be implemented but its nice to dream lol.

Forum: [Voting API](#), [Fivestar](#), [Advanced Forum](#), [Flag](#)

File Organization: File upload

PDF, email, print: [Printer](#), [e-mail and PDF versions with wkhtmltopdf](#),

Image: thumbnail creation, [Image Upload](#), [Lightbox2](#)

SEO: [Page Title](#), [Seo Checklist](#)

SEO friendly url creation: [PathAuto](#)

Statistics reports: [Google Analytics](#)

Ping pages: [Add to Any Share/Bookmark Button](#)

Site map: [XMLsitemap](#)

MetaTag: [Nodewords](#)

Email: [Email Field??](#)

Email list manager: [Simplenews](#), [Mime Mail](#)

Theming: Panels

Automatic backups: [Backup and Migrate](#)

Calendar of events: [Calendar](#), [Date](#)

Retail store search: [Location](#), [Gmap](#)

Spam prevention: [Mollom](#)

Messages: [Notifications](#), [messaging](#)

Feeds: [FeedAPI](#)

Classes: [Signup](#)