Getting Started with Drupal

Tutorial Resource List

Table of Contents

Getting Started with Drupal

The Drupal Way: Manageable Abstraction

"Drupal strives to … provide its users with the tools they need to make their own content management solution, while still providing some pre-built components to help them get started. Thus, it can be described both as a content management system (CMS) and a content management framework (CMF) - one system which strives to have the strengths of both, without their deficiencies."

Drupal Community. (2013, January 4). Drupal Overview, Retrieved February 4, 2013 from: http://drupal.org/getting-started/before/overview

Introduction

This resource list was created to facilitate the sharing of numerous web addresses associated with this video tutorial. Within this list you will find links to obtain the suggested tools used in the tutorial, software packages, and more. Please keep in mind this tutorial and list act together to provide the minimal skills needed to build sites using **Drupal.** How this is done is by first installing the core software then adding various add-ons (modules) to make the site function as desired. Those demonstrated in the tutorial are among the most commonly used modules, and suffice to meet the needs of the example site. That is, a community oriented site for a regional computer user group including blogs and a forum.

One obvious decision made in choosing what tools to suggest was the desire to promote great open source software wherever possible. The use of such tools will certainly help reduce startup costs for those interested in building sites using **Drupal**. However, a need will always exist for good quality commercial software also. Links, therefore, to the programs used in this tutorial are also included.

On the subject of browser usage, my preference is to use **Mozilla's FireFox** for two reasons. Foremost is its good support of web standards as defined by the **World Wide Web Consortium (W3C)**, although others do equally well in this department. More importantly are the highly useful add-ons created just for this browser. These help me in getting my work done faster, especially when trying to figure out where a coding mistake has been made in building a custom theme. As work on a custom theme is drawing to a close, a review with other browsers – especially **Microsoft's Internet Explorer** – is in order.

Acknowledgements

I would particularly like to thank my husband and family for all their support over the years. My long-term friends and business mentors include Marlin Borsick, Dennis Timple, and Richard Gallagher. These people have always had faith in me and my skills, even when my own faltered.

At the same time, I must acknowledge the help provided by so many authors of books on **Drupal**. I have lost track of how many I have bought in all, but – thankfully – everyone was well written expanding my knowledge of the subject. One author I do wish to thank by name is Dani Norton for her series on **Drupal** design and theming subjects.

Others I want to thank include: Nathan Smith for the development of the **960** grid system, Sebastian Siemssen, Matt Smith, and Jake Strawn for the creation of the **Omega** theme; and Joshua Turton for his online presentation on creating an **Omega Sub-Theme in 45 Minutes** recorded at **Bad Camp** November 2012. Needless to say, these are only a few of those wonderful people donating their time and skills I have encountered.

My appreciation must also go out to those numerous others I have interacted with in the **Drupal Community**. In that, I count not only those I've encountered on the **Drupal** site itself, but **Drupal's IRC** channels, and **LinkedIn Drupal** discussion groups to name a few locations. What has continually amazed me is how helpful people are in answering questions, without the 'pay me first' attitude I have run into so often elsewhere.

Drupal Speak & Conventions

All software has its own terminology, and **Drupal** is no exception to that rule. Here is the meaning behind some of those terms, and conventions used in this tutorial:

Node – Within **Drupal** you will be using a number of different content types, all with their own unique names. However, when referring to content as a whole or its generation, the term node is used much as it is used in biology to refer to the point where a leaf is attached to a plant.

Content Fields – In my dialog, I have a bad habit of abbreviating this to just 'fields.' What I am referring to are the fields within a content form into which you enter data. Much like forms within **HTML;** fields can be configured to short or large areas of text, drop-down lists, radio buttons, etc.

Teasers – This is an abbreviated form of a specific content item, normally the first 600 characters or so used within summaries of content. Within the **Rich-Text** editor, control can be exerted over where the 'teaser break' occurs.

Role – Groupings of users so as to limit permissions.

Blocks – These are boxes that contain content that are visible on screen. Most blocks contain lists generated on-the-fly by other aspects of **Drupal** or pre-defined ones such as *Active Forum Topics*.

Feed – This refers to an RSS news feed normally attached to a page or block.

Module – Add-on software used to enhance the function of a **Drupal** site. Most modules used are small in nature, so as to minimize performance issues. Others are fully developed products in themselves intended to support ecommerce or CRM functions with more robust hosting requirements. All modules developed for use within **Drupal** are found on the **Drupal** site.

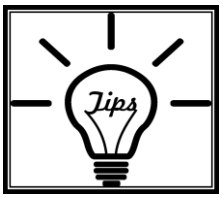
View – Some may confuse this with the **Views** module itself. In most cases, however, it refers to the stored settings used within **Views** to define a custom page, block, or feed.

Patch – Small code snippets intended to correct errors provided by others in the **Drupal** community.

Cron Job – While not really a **Drupal** term, one that is very important to the proper function of the site assuming it is based upon a **Linux** or **Unix** web server. **Cron** allows for scheduled tasks to be performed based upon date and/or time settings. The **Drupal** core software defaults to performing a **Cron Job** run every 3 hours. This can be adjusted within the *Admin* menu through: *Configuration, System, Cron*.

A much more detailed glossary to various terms used in connection with **Drupal** can be found here: http://drupal.org/glossary

When referring to a specific directory or file within a **Drupal** install or an address used to reach something within the test server, its name will be in the `Courier` typeface.



This box will appear in the tutorial when I am providing a Tip!

Video Tutorial Outline

Section One: Drupal Basics

Part One: Drupal Community (Running Time: 5:57)

Part Two: Resource List Introduction (Running Time: 3:51)

Part Three: Site Discovery Process (Running Time: 3:04)

Section Two: Building the Prototype

Part One: Tools & Organization (Running Time: 02:50)

Part Two: Installing XAMPP (Running Time: 06:31)

Part Three: Installing Drupal (Running Time: 07:29)

Part Four: Drupal Tour (Running Time: 03:42)

Part Five: Activating Modules (Running Time: 10:41)

Part Six: User Roles and Permissions (Running Time: 05:13)

Part Seven: User Accounts and Location (Running Time: 06:22)

Part Eight: Tokens within: Taxonomy, Paths, Titles (Running Time: 08:07)

Part Nine: Date Wizard and Calendar (Running Time: 01:57)

Part Ten: Content Types and Location (Running Time: 08:02)

Part Eleven: Advanced Help – Backup/Migrate (Running Time: 02:30)

Section Three: Multimedia

Part One: Multimedia Introduction (Running Time: 02:31)

Part Two: Adding Image Support (Running Time: 09:24)

Part Three: Adding a Rich Text Editor (Running Time: 7:48)

Part Four: Media (Running Time: 7:59)

Section Four: Refining Structure - Views

Part One: Blocks (Running Time: 7:55)

Part Two: Views (Running Time: 11:25)

Section Five: Drupal Theming

Part One: Theming Introduction (Running Time: 2:45)

Part Two: Delta (Running Time: 1:51)

Part Three: Omega (Running Time: 12:40)

Section Six: Deployment and Management

Part One: Migration – Prep Work (Running Time: 5:55)

Part Two: Migration – Final Steps (Running Time: 2:45)

Total Running Time for Course: 2 hours, 49 minutes, and 14 seconds

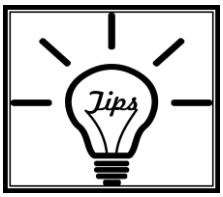
Tools, Mozilla Firefox Add-ons, and Library Packages

XAMPP from Apache Friends, download appropriate version for user's OS type; used to create test web server environment.

http://www.apachefriends.org/en/xampp.html

Guide to installing **XAMPP** on a **Microsoft Windows** machine, including several troubleshooting tips:

http://drupal.org/node/1763844



Windows users should change the properties setting for **XAMPP's Control Panel** shortcut (icon) to load it with ***Administrative*** privileges. While this doesn't get rid of all of the issues that can occur with running **XAMPP**, it will get rid of many of the more common problems!

Notepad++ is a source code editor, download appropriate version for users OS.

http://notepad-plus-plus.org/

NetBeans IDE used to write theme PHP and CSS files, easiest to download the ALL packages version.

http://netbeans.org/

Drupal's configuration tips for **NetBeans**, including those for *Drupal Coding Standards*:

http://drupal.org/node/1019816

Drupal's documentation page on installing patches on a **Windows** machine (I use **NetBeans** to do this):

http://drupal.org/node/60179

FileZilla FTP client for moving files between test server and hosted space.

http://filezilla-project.org/

7-Zip An open-source extraction tool for archives:

http://www.7-zip.org/

Drupal's security team notices are summarized here, along with links to sign up for e-mail notifications about security updates.

http://drupal.org/security

Many quick discussions between those in the **Drupal** community occur within various dedicated **IRC** channels. This page acts as an introduction to the various **IRC** chat channels for **Drupal**:

http://drupal.org/irc

There are quite a few IRC clients available for various OS types. I use the **Miranda IM** client for Windows found here, as it not only connects into IRC but a number of other IM networks:

http://www.miranda-im.org/

jQuery UI (library) – The jQuery UI library adds a number of user interface improvements with **Drupal**, and may be required for some modules.

http://jqueryui.com/

TinyMCE (library) * – This is the rich-text editor used in this tutorial.

http://www.tinymce.com/

Mozilla Add-on - Web Developer

https://addons.mozilla.org/en-US/firefox/addon/web-developer/

Mozilla Add-on - Firebug

https://addons.mozilla.org/en-US/firefox/addon/firebug/

Mozilla Add-on - No Squint

https://addons.mozilla.org/en-US/firefox/addon/nosquint/

Develop some method for storing notes of your own. How you do this is very much a matter of personal choice. Of late I've been making use of **Microsoft's OneNote's** ability to store data on the web that I can then share with others; even those not having that program. There are several other services of this nature out there, some being free for personal use. As you become more active in the **Drupal Community** setting up something of this nature will become highly useful.

A high quality, and most likely commercial, graphic/web design program is a must for theming. The time saved in preparing wireframes and eventually exported page concepts will more than pay for the cost. Look for a program that follows current web standards in the files exported, including **HTML 5** support. One that is smart enough to include specialized coding, based upon the design, for browsers or devices is a real treasure! Better quality programs will allow you to setup common elements shared across multiple pages, such as header and footer sections. Needless to say, upgrades to such a program must occur to remain current with changing coding practices. Shop around to get the most value, although I am a fan of **Xara's** products (read reviews of the latest version on my site). Whatever you choose to use, take the time to understand its features to accomplish tasks quickly.

Key factors would be built in mechanism for attaching **CSS** class coding to elements of a page that included within the output files. Provided the program saves that **CSS** in compliance with web standards, then it is a simply copy and paste operation to insert it into the custom theme files. Better quality ones will allow you to define all of your pages within one document thus allowing you to define '**global' CSS**

classes for shared elements. Another important point is to look for one that provides updates and/or revisions as coding standards change. If there is one thing that is certain on the web, it is that change is a constant factor!

Drupal and Add-on Modules

Latest core release for **Drupal** can always be found on the **Drupal** site.

http://drupal.org/

The following is a list of suggested add-on modules. Those featured in the tutorial are marked with a **(*)**.

AddToAny – Makes adding sharing buttons for popular social media, e-mail and more an easy task.

http://drupal.org/project/addtoany

Advanced Forum – Builds upon **Drupal's** forum included in the core, pulling data from other modules if installed. Effect is similar to that found with other forum software packages.

http://drupal.org/project/advanced_forum

Advanced Help (*) – If additional documentation on using a module has been provided outside of it in the form of HTML files, a link to it can be found here.

http://drupal.org/project/advanced_help

Author Pane – This creates a block that contains information about the author of the content appearing near it. It also integrates well with a number of other modules, including *Advanced Forum*.

http://drupal.org/project/author_pane

Avatar Selection – If you want control over permissions associated with avatars, this is the module that provides it. With it you can also upload pre-selected avatars that users may select from. (NOTE: This is one of those modules that are still a bit buggy in the **Drupal 7** release.)

http://drupal.org/project/avatar_selection

Backup & Migrate (*) - Used to create database backups and provides assistance in site deployment.

http://drupal.org/project/backup_migrate

Block Class (*) – Allows you to define the CSS class used by a block from within the block's configuration screen.

http://drupal.org/project/block_class

Calendar (*) – Creates a calendar type view associated with a given content type.

http://drupal.org/project/calendar

Chaos tool suite (ctools) (*) – This is a collection of API's needed to make a number of modules work properly, including **Views**. The included *Page Manager* is developing nicely as a replacement for **Panels**.

http://drupal.org/project/ctools

Context – Used to create contextual conditions and reactions within your site, such as the loading of different page layout templates based upon path.

http://drupal.org/project/context

Date (*) - Not only allows for date fields, but a date API of use in many other modules.

http://drupal.org/project/date

DB Maintenance – This is a nice addition to a site to automate the optimization of stored tables in the database during Cron jobs, thus reducing server load.

https://drupal.org/project/db_maintenance

Delta (*) – This will be used to create different page layouts as well as other tasks in the theming process.

http://drupal.org/project/delta

Devel - A specialized development module used in the tutorial to generate users and test content within the prototype. It can also provide data of use in analyzing the performance of a site. A word of warning, in my own recent usage it has failed in removing the generated content from a database requiring a database rollback to a previous state. Do be cautious, therefore, in its usage.

http://drupal.org/project/devel

Elements – This is required to make **HTML 5 Tools** work.

http://drupal.org/project/elements

Entity API – Although not featured in the tutorial, this module provides a unified API for working with fields defined within **Drupal**.

http://drupal.org/project/entity

Entity Reference – Used in combination with **Entity API**, allows for the creation of reference links between data. Example of use within prototype: using these two modules it would be fairly easy to add a field to the *Event* content type that would then contain the username of the speaker and provide a link to their **Profile**.

http://drupal.org/project/entityreference

External Links – Adds a small graphic icon to any links that lead outside of the site.

http://drupal.org/project/extlink

Field Permissions - Used to control permissions for fields added to content types, etc.

http://drupal.org/project/field_permissions

Global Redirect – Very simple module intended to help those trying to find content on your site.

http://drupal.org/project/globalredirect

Google Analytics - Facilitates information about a sites visitors with the free **Google Analytics** service.

http://drupal.org/project/google_analytics

HTML 5 Tools – The goal for this add-on is to make the creation of HTML 5 themes an easy process, regardless of theme in use.

http://drupal.org/project/html5_tools

IMCE (*) – Adds an image file management system into **Drupal**, with control over permissions by role.

http://drupal.org/project/imce

IMCE WYSIWYG Bridge (*) – Provides a bridge between IMCE and the installed rich-text editor.

http://drupal.org/project/imce_wysiwyg

JQuery Plugins - A collection of several *jQuery* plugins that may be needed by some modules.

http://drupal.org/project/jquery_plugin

JQuery Update - Updates the included version of **jQuery** in the **Drupal** core to a newer version. This may be needed to make some modules work.

http://drupal.org/project/jquery_update

Legal – Displays your *Terms & Conditions* for usage of a site to those wishing to register, manages updates to the same at need.

http://drupal.org/project/legal

Libraries API (*) - This is needed to integrate a rich-text editor into **Drupal** and similar packages.

http://drupal.org/project/libraries

Link - One of many field types that may be added to a content type's form, or used within a user profile via the **Profile2** module.

http://drupal.org/project/link

Location (*) – Adds location fields to content types or user profiles, depending upon how setup.

http://drupal.org/project/location

Login Security - Allows you to limit the number of login attempts made, set threshold for detecting automated attacks, and when blocking of host should occur.

http://drupal.org/project/login_security

Mail Editor - If using the **Subscriptions** module to send out e-mail notifications, this will allow you to edit the mail templates.

http://drupal.org/project/mail_edit

Media – Creates a framework for support and management of media assets of all types regardless of hosting location (local or through a service). Do pay attention to supportive requirements and/or libraries, depending on how it will be used!

http://drupal.org/project/media

Media: YouTube – Adds support within **Media** for **YouTube** videos as entities within a site:

http://drupal.org/project/media_youtube

Menu Breadcrumbs – Adds links to the top of displayed content indicating where the user is at within the menu structure.

http://drupal.org/project/menu_breadcrumb

Metatags - This module is a revised version of the old **NodeWords** found in **Drupal 6** for adding meta-tags to saved content. Support for most standard meta-tag types is built in.

http://drupal.org/project/metatag

Meta Tags Quick – If the requirements for a site call for custom meta-tags, this would be the module to use as it is based upon fields. It is not as user-friendly as the **Metatags** module.

http://drupal.org/project/metatags_quick

Mollom - Adds spam filtering and CAPTCHA protections to various aspects of a **Drupal** site.

http://drupal.org/project/mollom

Omega Tools (*) – Makes the creation of sub-themes based upon **Omega** an easy task.

http://drupal.org/project/omega_tools

Organic Groups - This popular module takes an alternative approach from the **Forum** found in **Drupal's** core to organizing discussions.

http://drupal.org/project/og

Page Title (*) - Combine the use of **Tokens** to set patterns for the page title tag associated with content.

http://drupal.org/project/page_title

Panels – This module allows for the creation of customized layouts within a page. The *Page Manager & Stylizer* found within **CTools** may accomplish what is desired, although there are still things **Panels** alone can do.

http://drupal.org/project/panels

Pathauto (*) - Combined with **Tokens** allows patterns for the path to be set when content is saved; a tremendous time saver in the organization of content on a site.

http://drupal.org/project/pathauto

Private Messages – Provides a mechanism for registered users (controlled by permissions) to send each other messages stored on the site, with notifications by e-mail. Also allows for admins the sending of messages to all roles.

http://drupal.org/project/privatemsg

Redirect – If there is one thing that is certain about a site, the way it is organized will change over time! This module helps when the path for content is change, or in redirecting traffic from removed content.

http://drupal.org/project/redirect

Revisioning – Drupal's core, in version 7, now includes some support for revisions. If more control over this process is needed, then it can be found with this module.

http://drupal.org/project/revisioning

Rules - Define actions based upon a given set of circumstances. Suggested examples of usage in prototype: order immediate publication of content generated by members that would ordinarily be held in the Revision queue, notify officers of new registrations on the site.

https://drupal.org/project/rules

SEO Compliance Checker - This module checks content when previewed or saved to ensure compliance with common SEO factors. Depending on a given sites requirements, activating it for all content types may not be wise, but only for those where indexing by search engines is most desired.

http://drupal.org/project/seo_checker

Site Verification - Allows for verifications – normally in the form of meta-tags – as required by major search engines.

http://drupal.org/project/site_verify

Skinr – Will require the install of a supportive library, but allows for the definition of re-usable CSS styles within a theme.

http://drupal.org/project/skinr

Subscriptions - If you wish to send users e-mail notices about new and/or updated content that is what this module will do.

http://drupal.org/project/subscriptions

Superfish – Integrates the *jQuery Superfish* plugin for menus into a **Drupal** theme.

http://drupal.org/project/superfish

Token (*) – Adds support for a number of additional tokens beyond those included in the core of **Drupal**.

http://drupal.org/project/token

Views (*) – Allows for the creation of custom pages, blocks, and feeds amongst other things. There are a number of excellent books published on mastering the usage of this module alone, along with cookbooks containing view recipes.

http://drupal.org/project/views

Views RSS (*) - Only one of the many modules intended to expand the functions of **Views** further. This one aids in the creation of *RSS* news feeds that can then be attached to other views.

http://drupal.org/project/views_rss

Views Slideshow – Create a slideshow, normally in a block, that will contain either text and/or images. Over time this project evolved to include so many features the decision was made to split some of those off into another project listed below.

http://drupal.org/project/views_slideshow

Views Slideshow Xtra – This provides a number of supplemental functions for the Views Slideshow module, involving the use of overlays.

http://drupal.org/project/views_slideshow_xtra

Webform – This module allows for the creation and management of interactive web-forms on a **Drupal** site.

http://drupal.org/project/webform

WYSIWYG (*) – This relies upon the **Libraries API** to integrate various rich-text editors into a **Drupal** site. WYSIWYG works in combination with **Text Formats** (part of Drupal's core) to establish limits for HTML usage, links, and more based upon role.

http://drupal.org/project/wysiwyg

XML Sitemap – Creates an *XML sitemap* in accordance with the *sitemaps.org* specification and submit it to search engines.

http://drupal.org/project/xmlsitemap

Drupal Themes

Omega (*) – This theme will be used within the tutorial, as it is built with *960* conventions in mind. Support for HTML 5 is also included.

http://drupal.org/project/omega

960 Grid – A theme entirely based upon the 960 grid conventions.

http://drupal.org/project/ninesixty

Adaptive Theme – This is another excellent starter theme with a heavy emphasis on web standards.

http://drupal.org/project/adaptivetheme

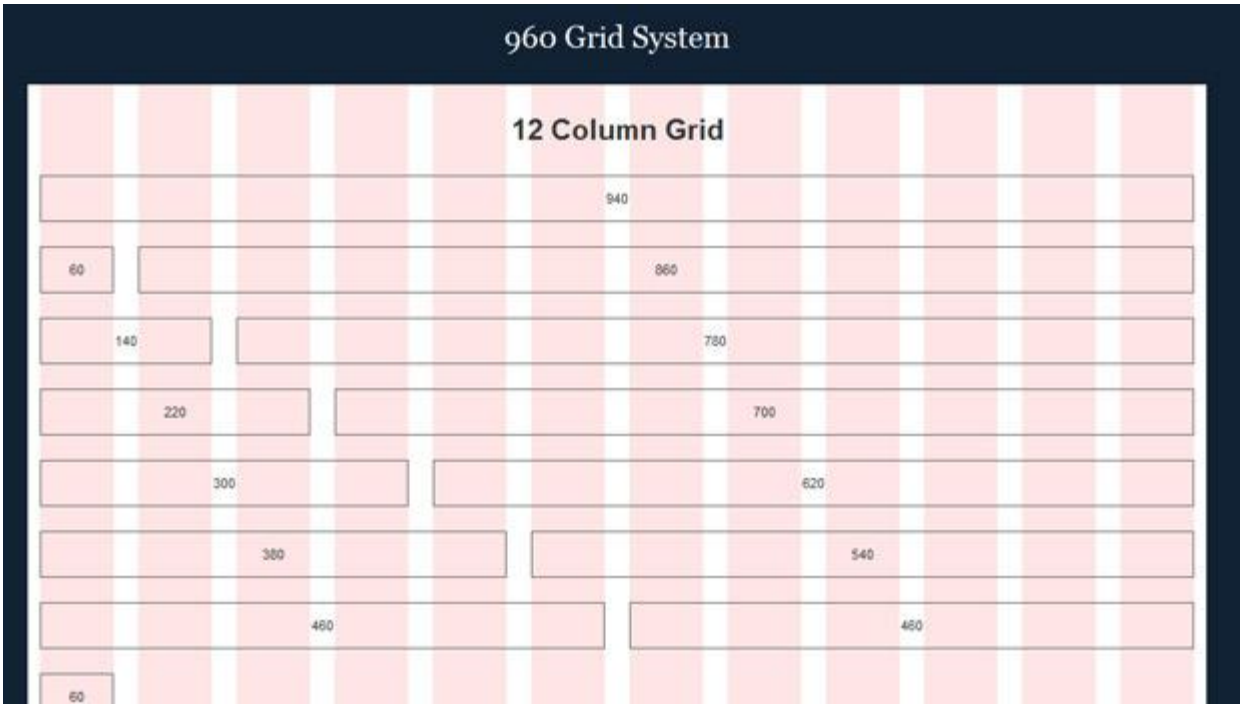
Zen – Zen is still very popular as a starter theme, with many guides to theming using it as a base.

http://drupal.org/project/zen

Drupal Theming Resources

Within this area will be some advice on various subjects relating to theming, including links were appropriate. To start, there's the subject of Nathan Smith's discovery involving commonly used grids in web design. His initial observation was that many of his themes could be broken down beginning with a

960 pixel page container into a 12 column grid with each one being 60 pixels wide with 20 pixel gutters between the columns and 10 pixel margins on either side of each column.



What he did that was so revolutionary was the decision to code, in advance, all of those numerous <DIV> containers in every combination possible thus saving tremendous time in the future. All he needed to finish the stylistic aspects for a theme was to add those CSS codes for that particular site. As he began to share his ideas with others, additional column grid widths with their own related code were created. On his **960** site, he provides information on this idea as well as a ZIP file containing samples of the grids and templates for various illustration and graphic design programs contributed by many others.

Ethan Marcotte, in putting forward his concept of **Responsive Web Design** was equally revolutionary. Marchotte's proposal is that designers should not create separate web designs for various devices such as desktops and mobile gadgets. Instead, the design should adapt dynamically to the viewpoint of the user's browser.

The **Omega** starter theme for **Drupal** was built to incorporate both of these design concepts. This was only part of the reason for choosing it as the starter base for the theme in the tutorial's prototype. Beyond that is the fact **Omega** moves control over many aspects of theming into the **Drupal** interface reducing the need to write actual PHP code. Taking into consideration the fact that the majority of those viewing this tutorial do not know what is involved in creating a sub-theme, adding in **Omega Tools** to accomplish that task made sense. Taken as a whole, the time needed to create a theme for a **Drupal**

managed site is reduced dramatically, assuming the work for creating artwork and stylistic CSS is ignored.

The bullet list that follows contains several tips for adding **CSS** into a **Drupal Theme**:

- Avoid working directly with a **Drupal** prototype in writing the **CSS** for as long as possible, as the server's cache files (found under *Configuration*, *Performance*) must be cleared before changes will be visible.
- Use the *Inspection* tool found within the *Firebug* add-on for **Mozilla's Firefox** to help identify class and ID attributes for <DIV>'s.
- Read more on the value found with high quality graphic/web design software in the section on **Tools**.
- Common questions in regard to theming with **Omega** are answered here: https://drupal.org/node/1416128
- The **Omega** related links page found on **Drupal**, containing numerous links to video presentations, tutorials, and more: http://drupal.org/node/819182

Here are numerous links for further study on the subject of **Drupal Theming**:

960 Website itself as managed by Nathan Smith:

http://960.gs/

Free video tutorial explaining concepts within **960** designs:

http://net.tutsplus.com/articles/news/a-detailed-look-at-the-960-css-framework/

Yet another free tutorial (Power Point slides) on **960** designs:

https://speakerdeck.com/u/nathansmith/p/accelerated-grid-theming

Style Tiles and their role in the visual design process:

http://styletil.es/

Article providing an introduction to the concept of **Responsive Web Design** as defined by Ethan Marcotte:

http://friendlymachine.net/posts/2011/when-a-trickle-becomes-a-flood

Great article aimed at explaining **Omega's** role in connection with both **960** and the concept of **Responsive Web Design**:

http://friendlymachine.net/posts/2011/drupal-omega-theme

In-depth walk-thru on creating a sub-theme using **Omega**, **Omega Tools**, **Delta**, and **Context** by Joshua Turton of Phase II Technology on creating an **Omega Sub-Theme in 45 Minutes** in November 2012:

http://2012.badcamp.net/program/sessions/omega-download-layout-45-minutes

Website capable of generating **Lorem Ipsum** dummy text used in testing web design work:

http://www.lipsum.com/

Localhost Links

In the course of these videos you will see me typing interacting with the test server via a browser by things typed into the address bar area. Here is a short summary of those:

To reach **XAMPP** for the first time after install: `localhost`

After **XAMPP** is fully setup and configured: `localhost/xampp`

To reach phpMyAdmin used to create and manage databases: `localhost/phpmyadmin`

As explained in the tutorial, how you install web software using **XAMPP** is by creating a directory under the `htdocs` folder found under **XAMPP**. Whatever name you give that folder is then used to reach that software via your browser following this pattern: `localhost/(folder name)`

On Backups and Drupal Site Updates

This is my one and only 'hosting horror' story I will share, as it illustrates the importance of backups and more in regard to using **Drupal**. In brief, there was a bug-release fix for one module (**Views**) that caused major conflicts with others to the point it was 'breaking' the display of content in the desired manner. The issue queue for this module heated up with numerous reports, with discussion between the *maintainers* of the **Views** project and other projects on how to devise the best fix. For those who had installed that update an immediate fix came from doing a database rollback to the state of the site before the update to **Views** was performed. (I should add, in my 5 years of using **Drupal** this was also the only time something on this order has ever occurred.)

Carrying out this rollback for one of the sites I maintained proved impossible in the manner desired as it had been experiencing deteriorating server support issues for roughly 7 months prior to this time. The first indicators for this were an increasing number of 'page not found' errors in the log for existing content. As more time passed, this escalated into failures of the site to come up at all when someone tried to reach it. What was occurring, in brief, was that the hosting company's server was experiencing too much demand upon its resources. Such situations arise under these conditions: the hosting

company having packed too many accounts on that server; or one or more of the accounts located on the server using up too much of the server's resources.

Although I had filed a number of complaints and support requests with the hosting company in question, they were pretty much ignored. When this response pattern was clear, a recommendation for a hosting change was made to the client involved. This came about only three months into a year's worth of hosting, which is why the client proved so resistant to my suggestions. Quite understandably he wanted to get his money's worth and was willing to put up with, what he felt were only minor outages in service.

Moving forward several more months to when the advice for a rollback was issued, the hosting server's response rate had continued to worsen. It was at that time I discovered, to my own horror, that all of the database backups made for the site for those months in question were zero (0) bytes in size. In other words, they were useless! After numerous tries the server did finally create a decent database backup of the site as it was at the time. However, in doing the rollback work had to begin with that last known good backup. All of the updates made since that time had to be done again, as well as copying over any new or altered content.

The lessons one can learn from an experience of this kind are numerous and (I hope!) valuable in preventing similar mistakes by others. Foremost is the importance of ensuring good backups of the database on each occurrence. A quick visual check of where the file was downloaded to ensure a file size larger than zero (0) suffices there. Next is the need to delay the install of normal updates for contributed projects for a few days, while monitoring activity in the issue queue. Another change made in support of this is a more active testing of such updates on my own test servers prior to rollout to supported sites. This should reduce the risk to production sites; as such significant problems should be evident there.

Getting back to the site in question that triggered this discussion for a moment… Thankfully by this time my client came over to my side of the fence and requested a change in hosting himself. He is back to being very happy with his site running on a new hosting companies server, with a significantly revised maintenance agreement in place as well!

Within the prototype usage of the **Backup and Migrate** module will be demonstrated. The importance of doing frequent a database backup cannot be understated. Do so prior to doing *any* major change on a site, especially prior to doing updates. It is so easy to do it can be forgotten in the pressing need of getting other things done. Even better is how the module automatically puts the current date and time stamp in the filename. What this means is that you can save a database backup pretty much as often, even every few minutes.

The process of backing up a **Drupal** site, as must be done when doing a core update, is explained here:

http://drupal.org/node/1223018

Thankfully, for those of you who may have tried out **Drupal 6**, the process of updating **Drupal** is much easier now! These all being what is considered the manual method of doing such an update.

Keep in mind when you put the site into maintenance mode, it will force all users currently logged in off (other than you as **User#1**, of course). Plan your time for doing your updates – where possible – to when there is little to no activity. By all means, make use of the **'Who's Online'** block demonstrated in the tutorial to see who – if any – are online. Beyond that point, you will have to work out your own plan to manage users still active on the site when updates are needed.

Managing Temporary Files

During my tutorial on copying the **Drupal** software files into place, an additional directory called `tmp` under: `sites/files` were added. Within the section on ***Deployment***, the temporary files folder setting within **Drupal** was changed to point to this location. This is my own opinion as to the best location for it, as will be discussed shortly. There are differing opinions on this subject, a fact that cannot be ignored.

One of the issues anyone running **Drupal** on a site will have to contend with is the opinions and attitudes of the hosting company themselves. Put another way, whenever (in my experience) that temporary folder was put in the root directory of the **Drupal** install, some hosting tech will come along and assign it to read-only access. Needless to say, this 'break's the site as **Drupal** cannot read and write to that directory as it should. Attempts to reach the site in question result in the infamous *White Screen of Death (WSOD)* with one line of an error message.

By putting that `tmp` directory under `sites/files`, issues of this sort rarely occur. This comes from the fact most hosting companies know (or should!) that everything located there must be kept *read/write* accessible to the software. This has only occurred with hosting companies unfamiliar with **Drupal**, something that should act as a red flag for their continued use. By all means, do take the advice on researching hosting provided in the next section seriously. Sadly enough, there are still far too many hosting companies claiming to support **Drupal** installs in their shared hosting accounts that really do not.

On Hosting in General

In this last section I am going to talk about the importance of choosing a **GOOD** hosting company to run your site. With that said, how do you find a good hosting company? The answer to that, in brief, is: research, Research, **RESEARCH**! Here I am speaking as someone who has utilized numerous hosting companies in the twenty plus years I have been building sites; you simply cannot do too much research. Earlier I shared my own worst hosting horror story. Then again, invaluable lessons were learned to improve the operations of my firm. Rather than dwell further on that subject, let's focus on what you should do to locate a good hosting company.

There are several excellent websites out there dedicated to helping you find hosting. Links to those won't be provided here as they are not hard to find. The better ones will include a method for defining

the requirements needed in the hosting package along with some sort of rating feedback system. By all means, pay attention to what others say! **Drupal's** site has always listed the minimal requirements needed:

http://drupal.org/requirements

There is a link on that page to several other pages on the subject of hosting, but the "*Webhosting Troubleshooting FAQ*" is of particular interest. Another good page to look at is the one that summarizes those hosting companies that are paid supporters via the **Drupal Association**. Not that there are not good hosting providers outside of that small list, but again those are some names to check out.

Of equal importance in locating good hosting, is the need to determine the proper type. One fact that will quickly become apparent is that the more modules added into a **Drupal** site, the more use such a site will put on the web server resources. If your hosting account is not up to supporting the site, then the only thing to do is either try to find better hosting or pay for a different hosting type.

How many modules are too many? That's a hard question to answer. The best attitude to take at all times is to stick to the absolutely essential for a sites operation. As experience grows not only in the *performance records* for a given hosting company, but in using **Drupal** as well, a determination on when and what can be added without a change in hosting becomes clearer.

As an aid to this aspect of things, check out the **Devel** module mentioned in the suggested projects list. One aspect of what it can do is providing feedback on processing time and memory allocations for supplying a given page to the viewer. To start, use this on the test server to see what sort of values it is indicating before and after the addition of modules. Not this is a totally accurate representation of how a real test server will act, but it will help.

If you plan to make use of rather large software packages, such as **CiviCRM** or **Ubercart**, then again these factors must be taken into account when choosing hosting too. These are very complicated software packages in themselves, designed to run within **Drupal** and need better than average server support. If there is a need for secure connections and security certificates, these things will also impact upon hosting cost. Both of these software packages also have external sites that may provide info on hosting.

One of the questions asked most often over on the **Drupal** site by novice site builders is if usage of a shared hosting account is possible. I, for one, can answer that with a very definitive ***yes*** with two stipulations. The first one of these being that the shared hosting really does provide the proper support to run the site in question. That is actually a harder target to reach in itself, as so many hosting companies say they support **Drupal** sites but really do not. Do your research by questioning them on aspects about **Drupal's** usage and that will help you avoid such companies. My second stipulation is that such a site must have a fairly low number of users and not have too much traffic. There must always be

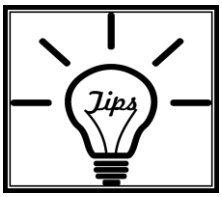
the assumption as the amount of traffic and/or users of a given site grow, the type of hosting needed to support it will increase in cost.

There is a great module for **Drupal** intended to work with **Google Analytics**, a free metrics collection service. That is, it gathers various facts about visitors to a site; where they come from, if they've been there before, etc. Combine that with **Google's Webmaster** tools now integrated into **Analytics**, you can begin tracking responses to campaigns housed on the site. This was not included in the building of the prototype, but its setup is really easy. Simply register on the **Analytics** site, provide the domain name for the site, get the code and plug it into the module's setup. This is one of those subjects you will probably want to study in depth so as to see how to get the maximum benefit. Thankfully, again, there are great books out there on this subject.

That pretty much summarizes what I can share on the subject of hosting. Other than to stress, again, the need for research!

Migration Checklist

This is my own take on a checklist of what needs to be done to prepare a **Drupal** prototype for movement to some other location. Not all steps should be followed in all situations, but having a list of this nature should help ensure the site works as well in its new location as it did in the old. If you are *piloting* the site prior to actual deployment (always recommended!), then the destination will probably be a password protected directory within your own web account. Some steps mentioned in this checklist should be delayed when piloting until actual deployment to the final destination.



I maintain a separate domain and hosted account specifically for the purpose of projects that have reached the *pilot* stage. The primary reasons for this are to avoid any impact upon my regular site and to better manage server load issues. If I had **VPS** (virtual private server) type of hosting, this would probably not be so important. I would still have to keep an eye on performance factors regardless.

- ☐ Check destination server for proper support for the prototype. This should include a review of **PHP** version, auxiliary services (such as **GD Image** support), **MySQL** or similar database, and memory allocations.
- ☐ Do a fresh install of the most recent release of **Drupal** on the destination server. This should, of course, match the version in use within your prototype. Play around with this test installs a bit to confirm the intended destination server is responding as it should.
- ☐ Leave a browser window open to the home page for that test install.
- ☐ Put site into maintenance mode.

- ☐ Check for any updates that need to be performed to **Drupal** or its add-ons within the prototype on the test server.
- ☐ Adjust temporary file path stored within **Drupal** to point to the `sites/default/files/tmp` directory.
- ☐ Change *Performance* settings as desired within **Drupal**. Adjusting the cache settings and other settings will help reduce server load. Do, however, take into consideration the activity level of the site.
- ☐ Remove any dummy content or unwanted users found in the prototype. Obtaining copies of content intended for use on the actual site is highly important. An analysis of the content intended to be on a site is a very critical part of designing its prototype. From that decisions can be made about the number of different content types needed, their settings, structure for the site (including *Views*), as well as display options.
- ☐ Within custom theme setup, remove uploaded *Logo* or *Shortcut* icons (if applicable), and restore usage of defaults. Save your theme settings! Why is this? The path to where those files are stored will differ from where they would be found in either a test server or piloting situation. When the site tries to load with such settings in a theme, errors will occur to the point a site will fail to load.
- ☐ Set as enabled, the default theme. (If custom *Logo* or *Shortcut* icons were uploaded to this theme, remove those and set back to defaults.)
- ☐ Make a fresh backup to the database.
- ☐ On the destination server, open **phpMyAdmin** or tool provided by the hosting company to work with the databases and drop all tables used by the test install.
- ☐ Import the fresh backup for the prototype within **phpMyAdmin**.
- ☐ Copy all directories and files found in the prototype under both `sites/all` directory to the destination server. (Preserve the `README.TXT` files located in the `modules` and `themes` directories from the most recent release of **Drupal**.)
- ☐ Delete any directories and files found in the `sites/default/files` on the destination server. Copy the contents of this directory within the prototype to the destination server.
- ☐ If changes are desired in the `.htaccess`, `robots.txt` files, etc. make the needed adjustments.

- ☐ Refresh the page within the browser window left open to the test install, which should now bring up the prototype. In most cases, your login state will not be preserved. Visit the `(path for migrated site)/users` to login.
- ☐ Check to ensure *cron job* settings are as desired for current server.
- ☐ Check to ensure paths for any linked files, such as images inserted within content using **IMCE**, are correct.
- ☐ Activate the prepared custom theme, including the upload of the custom logo and favicon. Check through the pages of the site to ensure all are loading as they should. Correct any issues found either with page displays, **Views**, and theme.
- ☐ If site is in *pilot* stage, it can now be taken out of maintenance mode for final checks, additions, and/or changes prior to deployment. Do remember to remove custom logos and favicons from stored theme settings prior to deployment.
- ☐ If the site has reached its final location, activate modules such as **Google Analytics**, **XML Sitemap**, or others intended to integrate with search engines, social media, or other sites.
- ☐ For sites at their final destination it is best to turn off the display of error messages. Errors should, of course, continue to appear in various log reports and the like.
- ☐ Take site out of maintenance mode and transition project to final phase.

There are many other checklists of this type out in addition to paid migration services. By all means, I encourage you to investigate those as they can be of aid in some situations. A more detailed list, for example, beyond my own can be found on the **Drupal** site here:

http://drupal.org/node/333675

Best of luck to you in your endeavors…