Drupal Best Practices

(or just good ideas)

Q. Why Is it important to practice good programming habits?

A. Following best practices, and thoughtful planning from the beginning, will ensure a well received project outcome, limiting mistakes while speeding development.

You are not #1

Don't run as admin, use an email address like webmaster or webmistress@yoursite.com. Then create a second account for yourself with administrator rights with your personal email address, and use that to control your site. Some sites even block user#1 so no-one can get in with those credentials.

Get update notices

Create a drupal.org account if you don't already have one.

Sign up for security updates here http://drupal.org/security

In each of your sites, go to /admin/reports/updates/settings d6 or 7 and enter that personal email address to receive notices from your site/s there

RUN CRON, hourly if possible

Organize your URLs

When creating views, set the paths for views pages according to your architecture

When constructing sites with custom node types, user's blogs, or a neat taxonomy use pathauto patterns to keep your url's organized.

Organize your file system

Create separate folders in your system for attachments to each of your content types, users, taxonomy terms

Multisite installs

Along with the default site which should be the master in a multisite, each site has to have it's own directory in sites/ that is the full name of the site

if each is a subdomain of the default site, some hosting control panels create a sub directory for you, you have to then delete that directory then create a symlink in the docroot that points back to the docroot so Drupal can handle addressing it

```
ln -s /var/www/html/ sub.sitename.com
```

Multisite installs

In sites/all and/or sites/sitename create a modules and themes directory, then in modules create custom and contrib

to keep drupal from slowing down keep this in mind drupal will read the directory structure it needs to build each page, so only keep what is being used by all your sites in sites/all

for instance, if one of the sites is ecommerce, put those modules in that sites

sites/sitename/modules/contrib directory

so that other sites don't read it, but any common modules, like views, token … belong in

sites/all/modules/contrib

Multisite installs

on that note, only put custom mods in sites/all/modules/custom if they are being used by more than one site otherwise put them in sites/all/sitename/modules/custom

the same can be said for custom themes although, I've never heard of them having custom and contrib in themes it's sufficient to put master themes in sites/all/themes and custom themes in sites/sitename/themes

if using the libraries module put your libraries directory into sites/all so they are available everywhere

There's one GOTCHA you should be aware of, DON'T name your module the same as your theme. Unusual behavior may ensue.

Indispensable Tools

- Drush, GIT & Project Management Software

INSTALL DRUSH

Use Git

registry_rebuild

git hosting, there are a few choices, github, beanstalk, assembla

some also offer issue tracking and project management

Module *musts!*

- you should be using almost all the top ten modules views, token, ctools, pathauto, admin_menu
- and some not so top ten that are just good suggestions like jquery_update, context, entity, module_filter, libraries, profile2, realname, file entity
- Advanced_help, google_analytics, workbench, rules, features and strongarm may be good ideas as well

Honorable mentions

calendar, ckeditor, logintoboggan, display suite, seo_checklist, seo_checker if your not using seven as your admin theme you might want to add quicktabs module

development mods include devel, coder, devel_themer, variable, drupal for firebug

Security mods security review, password_policy

some more suggestions FAQ, string overrides, "terms of use" or legal

in other words there's a mod for that

Theming

if you're creating a subtheme make your directory outside of the master theme's directory but you don't have to enable the master, but I think it would be a good idea so if there's any updates available to the master, drupal updates will let you know

copy the .info file and keep certain parts plus add base theme = mastertheme so it knows where to inherit things from

If you declare a region you have to declare all that you want from core too

Stark and Garland don't declare any regions and inherit drupals core regions

Theming

- core regions
  - Left sidebar
  - Right sidebar
  - Content
  - Header
  - Footer
  - Highlighted
  - Help

Theming

- Seven declares 5
  - Content
  - Help
  - Page top
  - Page bottom
  - First sidebar
  - then hides one (regions_hidden[] = sidebar_first) I imagine only during certain circumstances
  - and doesn't print or allow blocks to be entered into Page top and Page bottom and it doesn't inherit the others from drupal core
- This tells us that if you want to add your region to your theme, you have to re-declare the drupal core regions that you want to keep

Theming

```
BTW Zen declares 9,Genesis 10,AdaptiveTheme 12,Omega 18
```

Theming

Bartik declares 17

```
regions[header] = Header

regions[help] = Help

regions[page_top] = Page top

regions[page_bottom] = Page bottom

regions[highlighted] = Highlighted

regions[featured] = Featured

regions[content] = Content

regions[sidebar_first] = Sidebar first

regions[sidebar_second] = Sidebar second

regions[triptych_first] = Triptych first

regions[triptych_middle] = Triptych middle

regions[triptych_last] = Triptych last

regions[footer_firstcolumn] = Footer first column

regions[footer_secondcolumn] = Footer second column

regions[footer_thirdcolumn] = Footer third column

regions[footer_fourthcolumn] = Footer fourth column

regions[footer] = Footer
```

Code standards and secure code

- http://drupal.org/coding-standards
- http://drupal.org/writing-secure-code
- http://api.drupal.org/api/drupal
- http://www.php.net

The best way to fix your site if it's broken is Googling it

Google site:drupal.org “your warning or error message here”

Some last thoughts

Don't hack core, but use it to learn how to add to it properly, inspect the code and the api to add functionality in your theme or module.

If you want to modify an existing module, look through the issue queue for that mod, create a “feature request” issue or offer your modifications as patches.

If you are going to hack an existing module to customize it to your needs and it would be too custom to give back to the community, that's ok just namespace it. If your project is acme.com (some short but descriptive name, not more than one word or acronym/abbreviation without the .com) add that to the name of the module and global replace the module's name in the code with the same convention

Files get renamed acme_example.module acme_example.info acme_example.install

Functions get renamed acme_example_theme() acme_example_render() acme_example_form_alter()

This way you can leave the original in your module directory and receive updates to the module to adjust your version with whatever patching to remain secure and updated

Some last thoughts

Most Drupal code (core and contrib) is well documented and a lot of the community is working just as hard on that as on the code itself. Look in the code of the modules/themes you want to use, a lot of them have comments that are very revealing on how it gathers and displays it's variables. Views is well commented especially the template files. Themers that have to craft views template files should copy the appropriate tpl.php file from the views/theme directory.

In the views ui, expand the advanced tab on the right. At the bottom now is the theme info link, click it and an over lay will appear with the different template filenames available for each output. The one's in use are in bold, the others are suggested names for better targeting of that output. Choose the one filename that's as specific as you need, copy the bold tpl.php file from the views/theme to your custom theme and rename it with your selecion, and edit the markup and variables in that file to suit your needs. Use dsm() or dpm() to print variables you want to inspect while building the template.

One gotcha, clear the cache twice to see the results of any template changes.