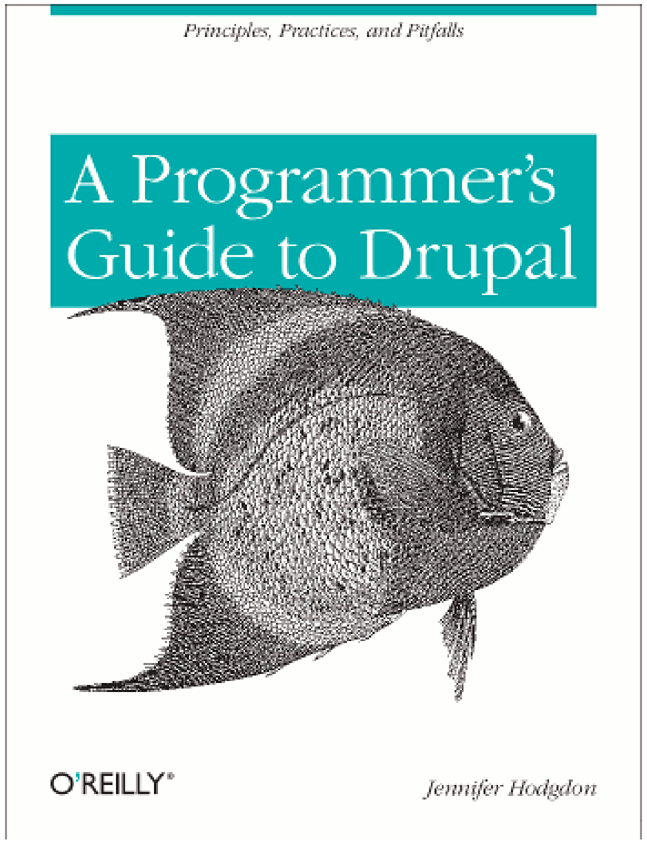
Mistakes Programmers Make

Jennifer Hodgdon
(jhodgdon)

Poplar ProductivityWare
(poplarware.com)

Pacific Northwest Drupal Summit
October 2012
Seattle, WA

Shameless Plug



Principles, Practices, and Pitfalls

A Programmer's Guide to Drupal

O'REILLY®

Jennifer Hodgdon

http://shop.oreilly.com/product/0636920027799.do

Mistake: Hacking (vs. Altering, Patching)

NO!

Hacking: Editing files you downloaded from Drupal.org (Drupal Core, modules, themes) – Makes it difficult to update!

Altering and Patching:

YES!

- Creating a sub-theme (http://drupal.org/node/225125)
- Theme hooks: altering core/module output in your theme (http://drupal.org/node/341628)
- Hooks & alter hooks: Extending or altering core/module functionality in a module (http://drupal.org/developing/modules)
- Fixing a bug or implementing a new feature and providing a patch (http://drupal.org/node/707484)

Mistake: Custom Programming (vs. Config)

NO!

Custom programming: Creating your own module (you will have to maintain it forever!)

YES!

Configuration and Site Building: Using an existing module and customizing/configuring

- Views
- Context or Panels
- Fields/Display Suite or Panels
- Taxonomy
- Flag
- Rules
- Custom breadcumb
- …

Mistake: User-Entered or User-Edited PHP Code

NO!

User-entered PHP code (block body, node body, calculated fields); editing PHP code via Drupal interface: security issues, bugs

Alternatives:

YES!

- Context module (block placement)
- Custom module (create blocks, pages)
- Views (create blocks, pages)

Mistake: High-Impact Programming

Some hooks get invoked a lot, and cause extra page processing time:

- hook_form_alter() vs. hook_form_FORM_ID_alter() & hook_form_BASE_FORM_ID_alter()
- hook_init() & hook_boot() vs. page redirects, permissions, hook_menu(), hook_menu_alter(), etc.

Mistake: No Tests or Documentation

Standards for documentation headers: drupal.org/node/1354

Automated testing framework: drupal.org/simpletest

YES!

Write tests and documentation before code! Why?

- Think through what the code will do before you write it.
- Make sure it does what it is supposed to after you write it
- When you make a change, you can test for regression and avoid breaking existing code

Mistake: Ignoring Usability and Accessibility

YES!

- Make your admin interfaces look and work like the rest of Drupal
- Make sure your drag-and-drops degrade gracefully for non-mouse users
- Think about color contrast, alt tags, etc.
- Follow WCAG and other accessibility standards (there are accessibility testing tools out there!)
- Remember that search engines will see pretty much what a screen reader sees!
- Use existing Drupal Core widgets

Mistake: Not Following Coding Standards

drupal.org/coding-standards

drupal.org/project/coder

YES!

Why adopt these standards?

- Code uniformity
- Maintainability by others
- Some day you may want to contribute modules to drupal.org and coding standards adherence is recommended if not required

Internationalization Mistakes

Incorrect internationalization:

NO!

```
print “Node was modified”; // no internationalization
print t($message); // variables in t()
print t(“Posted by $username”); // variables in t()
print t('Posted by ') . $username; // Doesn't work for some languages
```

Better internationalization:

YES!

```
print t(“Node was modified”);
$message = t('Node was modified'); print $message;
print t(“Posted by @username”, array( '@username' => $username));
```

Data Sanitization Mistakes

YES! All user-entered data should be sanitized!

NO! Incorrect data sanitation:

print '<p>' . $user_entered_text . '</p>'; // no sanitization
print l(check_plain($link_title), $url); // double-sanitization
db_query("UPDATE {users} SET name = '$name'") // no sanitization

YES! Better data sanitization:

print '<p>' . check_plain($user_entered_text) . '</p>';
print l($link_title, $url);
Use database API!

Data Sanitization Mistakes

YES! All user-entered data should be sanitized!

NO! Incorrect data sanitation:

```
print '<p>' . $user_entered_text . '</p>'; // no sanitization
print l(check_plain($link_title), $url); // double-sanitization
db_query(“UPDATE {users} SET name = '$name'”) // no sanitization
```

YES! Better data sanitization:

```
print '<p>' . check_plain($user_entered_text) . '</p>';
print l($link_title, $url);
```

Use database API!

Mistake: Working in Isolation (vs. Community)

Connect with the local, regional, and world-wide Drupal community:

YES!

- Join a local/topical group on groups.drupal.org
- Ask questions via IRC (http://drupal.org/irc)
- Use the forums (http://drupal.org/forum)
- Find partners for projects at local group meetings
- Learn best practices at regional and international conferences
- Use community-produced tools (Coder, Devel, Drush, ...)
- Build your reputation by contributing expertise (code, documentation, theme designs)
- Give back: open-source does not work without community