How custom is too custom?

Tips for coding (and when not too)

Brock Fanning

- brockfanning on drupal.org
- Drupal developer
- Currently contractor at a government agency
- Previously full-time developer for content marketing agency
- Party pooper (sort of)
- Question answerer

What is custom code?

Any Drupal code that does not have a presence on drupal.org. For example:

- PHP-driven blocks or pages
- alter hooks (hook_form_alter, hook_entity_view_alter, hook_query_alter)
- preprocess hooks (hook_preprocess_page, hook_preprocess_node)
- info hooks (hook_field_info, hook_theme, hook_menu)
- modules or javascript for unique site-specific functionality
- team-specific frameworks or helpers
- general-purpose modules that you haven’t put on drupal.org yet
- radical departures from the Drupal norms

What is contrib code?

Any Drupal code that has a presence on drupal.org

- modules, themes, install profiles
- patches to those things

Choose your own adventure

You walk into a room, and are given a feature request to implement on your Drupal site.

Do you...

1. Open up your text editor and custom code it?

2. Open up your browser and look for a contrib solution?

Why not custom?

1. Cost of training and documentation, difficulty of hand-offs
2. Lack of outside support, responsibility of maintenance
3. Risk of re-inventing the wheel

What happens when custom solutions get contributed?

Why not contrib?

1. Inflexible - unusual or particular requirements
2. Overkill - too complex, too many features, too hard on the server
3. Insufficient - not enough features

Some contrib solutions overcome these. And for cases where they do not, you can always change contrib.

Patching and contributing

Your escape route to programming fun (and happier clients).

- Patching is like acceptable hacking, and Drush/Composer makes it official
- Patches don’t need to be committed to be used, especially for Drupal 7
- Customize and contribute: best of both worlds

Caveat: Patches still need to make the project better, not just different

Analyzing contrib projects

Red flags:

- Breaks site
- Impacts performance
- Not supported
- Has show-stopping bugs
- No stable release
- Similar projects on drupal.og

Analysis tips:

- Go to project page
- Look at usage statistics
- Look at date of last commit
- Browse the issue queue
- Look at available releases
- Try it out
- Benchmark it on your site
- Look at the code

Case Study: Contrib was overkill

Site: large (hundreds of thousands of nodes)

Request: Add an og:image meta tag to all blog nodes

Contrib solution: Metatag module

Custom solution: drupal_add_html_head() inside hook_node_view_alter()

Analysis: Metatag project, gauge need for additional meta tags, difficulty of custom solution, installation/configuration of contrib solution, benchmarks

Case study: Obscure contrib recipe wins

Request: A system for automatically importing data into Drupal, and displaying it in arbitrary ways. (Eg: phone directory). Needs to scale (many instances of this) and allow for dynamic display (sorting and filtering) of data

Obvious contrib solution: Node types, fields, and Feeds. But, at scale, that gets out of control.

Custom solution: Parse feed sources, save serialized arrays in database, customize each instance with its own template file which outputs the data uniquely.

Not-so-obvious contrib solution: Data, Feeds, Feeds Data, and Views

Analysis: Vet Data and Feeds Data (sandbox), try out recipe (patch)

Case study: Patch contrib to meet specs

Request: A text-sizing widget to increase/decrease font size on the page. Must be able to be placed using Panels.

The catch: No cookies allowed.

Contrib solution: Text Resize

Problem: It uses cookies, and is incompatible with Panels

Solution: Write 2 patches and use them

Maintaining your custom code

- Over-comment the code
- Keep it in its own folder (sites/all/modules/custom/)
- How to handle Features modules?
- Tool for producing HTML docs?
- Anything else?

Giving estimates

Problem: We (developers) are optimists, people-pleasers, and puzzle-enthusiasts

Solutions:

- Be realistic about time estimates
- Mention the level of customization needed
- Offer contrib alternatives (even if they don’t meet the requirements)
- Educate managers about the costs of customization

Modules that imitate custom code

- Context
- Conditional Fields
- Data
- Views
- Entity Construction Kit
- Display Suite
- Field Group
- Views
- Page Manager
- Rules

Thanks for listening!

Questions?

You can also reach me at brockfanning@gmail.com