AVOIDING THE GIT OF DESPAIR

EMMA JANE HOGBIN WESTBY

SITE BUILDING TRACK

@EMMAJANEHW

http://drupal.org/user/1773
Avoiding
The Git of Despair

@emmajanehw
http://drupal.org/user/1773
www.gitforteams.com
With Apologies
To Those Who’ve Not Seen
The Princess Bride.
Happening

• How Git works in a deployment landscape.

• How to use branches for different environments.

• Commands you need to run.

• Why it’s hard to collaborate on Features.

• Commands to deal with merge conflicts
Not Happening

First-timer’s guide to:

• Drupal Module: Features

• Git (at the Command line)
20,000ft View

Drupal Module: Features
Features

A feature is a collection of Drupal entities which taken together satisfy a certain use-case.

Features allow us to export point-and-clicked configuration from one environment to another.

https://drupal.org/project/features
/* Sort criterion: Content: Post date */

$handler->display->display_options['sorts']
    ['created']['id'] = 'created';

$handler->display->display_options['sorts']
    ['created']['table'] = 'node';

$handler->display->display_options['sorts']
    ['created']['field'] = 'created';

$handler->display->display_options['sorts']
    ['created']['order'] = 'DESC';
Sharing Features

[Diagram of project hosting with user icons for Reviewer and Developer]
Sharing Features

1. start new_feature
2. upload new_feature
3. review new_feature
4. publish new_feature

Developer
Reviewer

or 4. reject changes
Deploying Code with Git
Deploying Code
(Actually)

Deploying Code
Branches allow you to store separate instances of ongoing work.
Git branching strategies are conventions we use based on common deployment setups.
Per-Environment Branches

Development

Staging

Production
Per-Environment Branches

Development
- Project Hosting
  - Local
  - Dev

Staging
- Project Hosting
  - Local
  - Release Prep

Production
- Project Hosting
  - Local
  - Production

Git Repository
- new_feature
- new_feature_blog
- new_feature_gallery
- dev
- qa
- master
Sharing Features with Git
Sharing Features with Git

- **Project Hosting** (e.g. Bitbucket or GitHub)
  - clone or pull

- **Developer**
  - push
  - `checkout -b new_feature`
  - `add <filename>`
  - commit

- **Reviewer**
  - push
  - `checkout --track origin/new_feature`
  - (do the review)
  - checkout dev
  - merge --no-ff new_feature
Remember This

**Features** (and its related export functions) is not always perfect but it is always better than using nothing.
5,000ft View

Improving Consistency with Drush
Remember This

The command line *can* provide a faster route to a more consistent experience.
Drush

• Drush is a command-line shell and scripting interface for Drupal.

• Execute cron, run update.php, work with Features, clear cache, and more.

• https://github.com/drush-ops/drush
Features focuses on code where Drupal would have normally focused on the database.
default

feature.module Z = database Z

 overridden

feature.module Z ≠ database X

mark as reviewed

code change

default

feature.module X = database X

revert
database change

 overridden

feature.module X ≠ database Y

revert
database change

 overridden

feature.module X = database Y

recreate

default

feature.module Y = database Y

https://www.drupal.org/node/582680
## Features + Drush
### Command Line Survival Guide

<table>
<thead>
<tr>
<th>Operation</th>
<th>Drush Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Export a feature</td>
<td><code>drush fu</code></td>
</tr>
<tr>
<td>Revert a feature</td>
<td><code>drush fr</code></td>
</tr>
<tr>
<td>Really revert your features</td>
<td><code>drush fra --force --yes</code></td>
</tr>
<tr>
<td>Clear all caches</td>
<td><code>drush cc all</code></td>
</tr>
</tbody>
</table>
5,000ft View
Avoiding Conflict
Hello. My name is Inigo Montoya. You killed my father. Prepare to die.
Remember This

Be Unique;
Avoid Overlap
Branch Reminder

Development

Project Hosting

Local

Dev

Staging

Project Hosting

Local

Release Prep

Production

Project Hosting

Local

Production

Git Repository

new_feature

merge

dev

merge

qa

merge

master

new_feature_blog

merge

dev

merge

new_feature_gallery

merge
PREPARE YOURSELVES

WALL OF TEXT IS COMING
INCONCEIVABLE!
Getting Ready (in Git)

• Start in the right "environment"
  $ git checkout [dev]

• Create a new feature branch
  $ git checkout -b [1234-new_feature]
Creating a Feature  
(Site Builder-friendly)

- Set all Features back to their factory defaults.  
  $ drush fra --force --yes

- Build your new feature with the pointy-clicky.

- Export your feature’s settings with the pointy-clicky.

- Put the downloaded file into:  
  /sites/all/modules/features

- Unpack your new feature  
  $ tar xvf feature_name.tar.gz
Updating a Feature
(Site Builder-friendly)

• Set all Features back to their factory defaults.
  $ drush fra --force --yes

• Build your new feature with the pointy-clicky.

• Update all features to use settings from the database
  $ drush features-update-all
  or
  $ drush fu-all
YOU SEEM A DECENT FELLOW
I HATE TO KILL YOU
Verify Your Feature is Right

• Your code is now changed to match the database. Using Git, see what’s changed.
  $ git diff

• Checklist:
  • Within an array, are the values in the same order?
  • Are strings (not) quoted?
  • Are there extra pieces?
  • Are there missing pieces?
You don't look so good.
You don't smell so good either.
Git It Up

• Check what is currently not in your repository
  $ git status

• Add the new Feature to Git
  $ git add [feature_directory]

• Save the new Feature to your repository
  $ git commit

• Add a really good commit message which describes
  what your Feature is, and all of its components.
I always think everything could be a trap. Which is why I'm still alive.
Share Your Feature

• Upload the Feature to your shared code hosting repository
  $ git push origin [1234-new-feature]
Testing Someone Else’s Feature

• Update your local list of branches
  $ git fetch

• Clean up your database by reverting all Features
  $ drush fra --force --yes

• Switch to the branch where the new Feature is
  $ git checkout --track origin/[1235-new-feature]

• Import the new Feature
  $ drush fr
Adding a Feature to a Shared Branch

- Checkout the branch you want to add the new Feature to.
  $ git checkout [dev]

- Ensure your copy of the branch is up-to-date.
  $ git pull --rebase=preserve

- Include the new Feature into the current branch
  $ git merge --no-ff [1234-new-feature]
10,000ft View

Dealing with Conflicts
Conflict is when there is an overlap at the same line.
Life is pain, highness; anyone who says differently is selling something
Investigating Conflict

• Determine the point of conflict:
  $ git mergetool

• Want to undo that merge? Back the truck up.
  $ git reset --merge ORIG_HEAD

• Take another look at the differences
  $ git diff [1234-new-feature]...[master]
Choose “ours”

$ git merge -s ours [branch]
Resources

• Building a Drupal site with Git
  https://www.drupal.org/node/803746

• Git for Teams
  http://gitforteams.com
More Resources

• Features - https://drupal.org/node/580026

• Drush - http://drush.ws/

• Introduction to Drush Series
  http://drupalize.me/series/introduction-drush-series

• Features & Drush Series
  http://drupalize.me/series/drupal-deployment-features-drush-series
WHAT DID YOU THINK?

EVALUATE THIS SESSION - LOSANGELES2015.DRUPAL.ORG/SCHEDULE

WWW.GITFORTEAMSCOM