Look Ma, No Hands
… Deployment

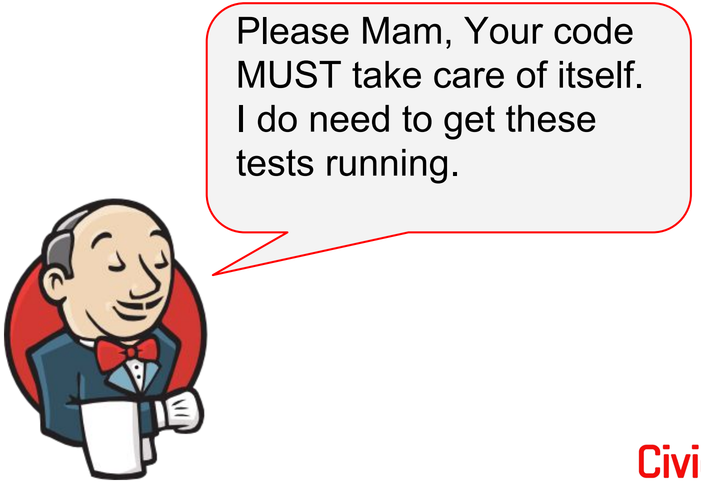
Steve Wirt

CivicActions → *Software Engineer*

Why Hands-off Deploy? - CAT

- Consistent steps increase reliability. [avoid ‘oops I forgot to ___’]
- Automated - Off peak deployers want a life too!

- Testable - (Travis, Jenkins...)


Please Mam, Your code MUST take care of itself. I do need to get these tests running.

Evolution of Deployment

✓ Click
✓ Click
✓ Click

Good?

```
✓ drush vset config1=TRUE
✓ drush feature revert ABC
✓ drush cache clear all
```

Better

drush updb

Better Yet

drush updb

I reverted Feature ABC successfully.
I enabled module XYZ successfully.
Warning! I was unable to enable ECK
I reverted Feature LMN but it remains overridden.

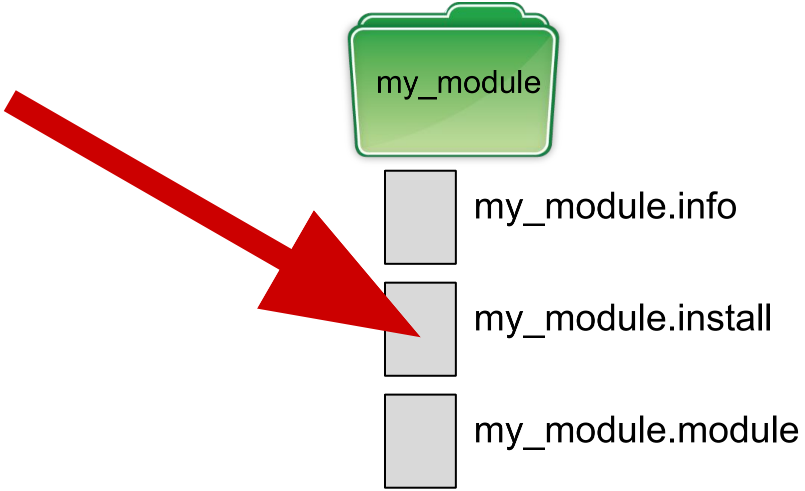
AWESOME

Anatomy of a hook_update_N()

Runs once upon ‘drush updb’ or update.php

```
/**
 *  Enable module Google Analytics.
 * /
MY_MODULE_update_7001(&$sandbox) {
  $modules = array('google_analytics');
  module_enable($modules);
}
```

This gets presented to person running the update.
Special case: Do not follow code style requirements.

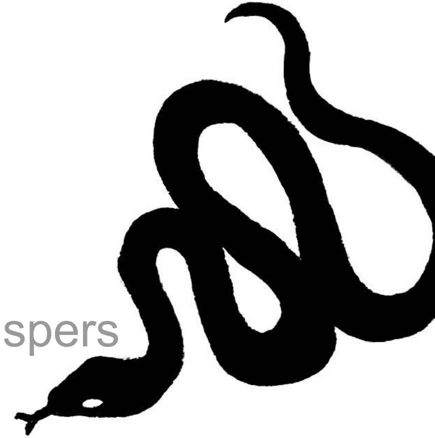


Anatomy of a hook_update_N()

In action...

```
/**
 *  Enable module Google Analytics.
 * /
MY_MODULE_update_7001(&$sandbox) {
  $modules = array('google_analytics');
  module_enable($modules);
}
```

Sneaks around and whispers
with a forked tongue.



```
> drush updb
My_module 7001 Enable Google Analytics
Do you wish to run all pending updates? (y/n): y
Performed update MY_MODULE_update_7001
>
```

Anatomy of a BETTER hook_update_N()

```
/**
 *  Enable module Google Analytics.
 * /
MY_MODULE_update_7001(&$sandbox) {
  $modules = array(‘google_analytics’);
  module_enable($modules);

  return “Google analytics was probably enabled.”;
}
```

Just telling you what you want to hear.

```
> drush updb
My_module 7001 Enable Google Analytics
Do you wish to run all pending updates? (y/n): y
Google Analytics was probably enabled.
Performed update MY_MODULE_update_7001
>
```

Anatomy of an EVEN BETTER hook_update_N()

```
/**
 *  Enable module Google Analytics.
 * /
MY_MODULE_update_7001(&$sandbox) {
  $modules = array(‘google_analytics’);
  $success = module_enable($modules);
  if ($success) {
    $msg =  “Google analytics was enabled.”;
  }
  else {
    // This module is not enabled, throw an exception.
    throw new DrupalUpdateException(“The module ‘google_analytics was
      supposed to be enabled by this update, but was not. Please investigate
      the problem and re-run update.”);
  }

  // Let’s set the variable too.
  variable_set(‘googleanalytics_account’, ‘UA-1-1’);

  return $msg;
}
```

1) Do something.

2) Verify it was done.

3) Return accurate feedback.

Anatomy of an AWESOME hook_update_N()

Use module Hook Update Deploy Tools

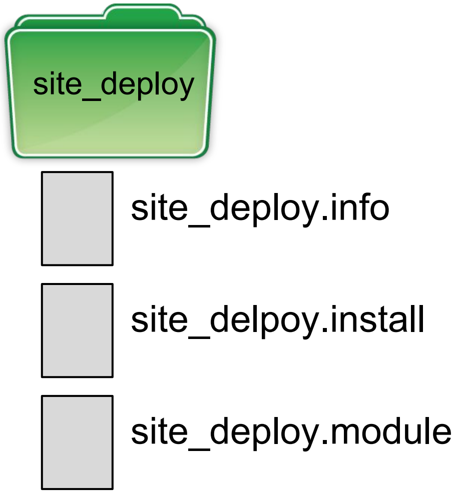
```
/**
 *  Enable module Google Analytics.
 * /
MY_MODULE_update_7001(&$sandbox) {
  $message = HookUpdateDeployTools\Features::revert('google_analytics');
  $message .= HookUpdateDeployTools\Settings::set('googleanalytics_account', 'UA-1-1');

  return $message;
}
```

All the tenets of a good update hook are baked in:

1. Does something.
2. Verifies it was done.
3. Gives accurate feedback and logs it.
4. Fails the update if the operation was not successful.

A Deploy Module

A simple custom module that does nothing but manage deploying your site changes.



Benefits:

1. One place to conduct the major releases of your site (module enables, disables, vocabulary creation, data wrangling…)
2. The install file becomes the site historian / release notes for your site.

* bonus: Using hook_update_deploy_tools run `` `drush side-deploy-init` `` and it will build the site_deploy module for you.

A Deploy Module | Best Practices

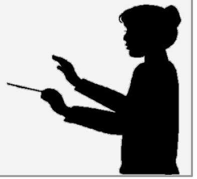
- **Don’t use fake dependencies to enable modules. (You’ll only ensure that site_deploy updates run last. So the module can only sweep-up, not conduct.)**

site_deploy.info

```
; Modules
dependencies[ ] = book
dependencies[ ] = search_api
dependencies[ ] = views

; Features
dependencies[ ] = cool_feature
dependencies[ ] = the_shiz
```

site_deploy.info

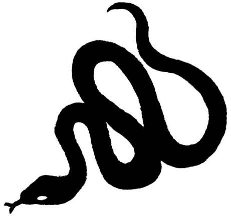
```
; Modules
dependencies[ ] =
hook_update_deploy_tools
```



Features that manage themselves

- Add a MY_FEATURE.install file to your Features.
- When a Feature needs to be reverted, call the feature revert from a hook_update_N().

```
/**
 *  Revert: Add new fields to blog.
 * /
cool_feature_update_7001(&$sandbox) {
  $features = array('cool_feature');
  features_revert($features);
}
```



Advantages

- Your .install file now reads like Feature release notes.
- Avoids feature revert all, which can be intensive.
- Avoids Git conflicts with editing site_deploy.install.

Features that manage themselves with hook_update_deploy_tools

- **To get validation and feedback, use:**

```
/**
 *  Revert: Add new fields to blog.
 * /
cool_feature_update_7001(&$sandbox) {
  $features = array(‘cool_feature’);
  return HookUpdateDeployTools\Features::revert($features);
}
```

Reverted cool_feature successfully.

- or -

cool_feature remains overridden after being reverted. Check for issues.

- or -

Revert request for cool_feature was skipped because it is not currently overridden.

hook update_n best practices

- Tell what you are going to do (meaningful docblocks)
- Validate what you did
- Tell what you validated (meaningful return messages)

Keep in mind

- Be aware of the order (or lack of). Dependencies run first. When orchestration is needed, use one hook_update_N() to conduct.
- Avoid re-numbering. Drupal tracks the last N run.
- Git collisions happen since you are always working on the last lines of the file.
- Use project/hook_update_deploy_tools

Open Discussion

Thank you.

Resources:

- dcycleproject.org/blog/44/what-site-deployment-module
- api.drupal.org/api/drupal/modules%21system%21system.api.php/function/hook_update_N/7
- www.drupal.org/project/hook_update_deploy_tools
- web-dev.wirt.us/modules/hook-update-deploy-tools
- http://nerdstein.net/blog/site-updates-drupal

What can hook_update_deploy_tools help you do?

- Generate a custom site_deploy module
- Enable, Disable, Uninstall Modules
- Revert Features
- Delete Fields
- Update node properties and simple field values
- Update node aliases
- Set Drupal variables {alter site config}
- Create / update Menus from menu import file
- Create / update rules from a rule import file
- Export rules with Drush
- Messages and logging