MetroStar Systems®

Powering Change®

Using Post Update Scripts to Deploy Content Across Environments

DrupalGovCon 2020

Matt Calvert & Jason Glisson

The Problem

- Massive amounts of content that needs to be deployed across environments
- Content must be launched in production quickly
  - During business hours deployments
  - Minimize any possible downtime or outage
- No access to some environments
  - Can’t execute shell commands
  - Minimal staff on hand with experience in Drupal content creation
- Copying databases UP from lower environments is a bad practice.

Use Cases for Post Update Scripts

- Deploying content across environments
  - Our project needed
    - 1 settings page for the backend UI of the interactive tool
    - 2 vocabularies
    - 53 taxonomy terms
      - Each one has a name, image, and description
      - Total of 159 fields needed to be filled out
    - 58 paragraphs
      - 227 individual pieces of content, some with multiple fields
    - 3-6 PDFs to upload
    - **597 individual fields that need data**   
  - This would have taken hundreds of hours to setup across multiple environments!

Post Update Scripts can help with...

Content

- Let your content curators do their job!
- While they may know how to use Drupal, don't force them to always be the ones copying content into the site.

Outages

- Avoid massive outages while content is entered
- Even with "all hands on deck" it can cost hours of time copying and pasting content into Drupal

Synchronization

- Make it easy to get all content/ fields exactly the same on all environments

Launching

- Launch large content applications in seconds
- Feed content from script into JSON endpoint

Difference Between Update and Post Update

- **hook_update_N**
  - Some Drupal APIs because the site isn't bootstrapped yet (Entities, Fields, Views, Etc)
  - Must follow numbering iteration in name
  - Can be used for creating, reading, updating and deleting entities (CRUD) but, again, not all APIs are available
  - Ideal for database schema updates
- **hook_post_update**
  - Runs AFTER hook_update_N
  - All Drupal APIs are available so ideal for CRUD
  - Can be named anything
    - moduleName_post_update_anything()

What about hook_install scripts?

These are similar in structure but should be used when:

You're installing a module	Setting up database schemas specifically for a module	You need to allow something to be undone when the module is uninstalled

hook_install functions are called from a full bootstrap so the same Drupal API scripts in post_update are also available in hook_install.

Before writing your script...

- Create all content types, fields, paragraph types, and even vocabularies ahead of time and get it into configuration.
- Decide what needs to be created first, in what order, and when:

  Example: You need a paragraph on a page that has a taxonomy terms reference field. You'll need to:

  1. Create vocabulary
  2. Create the terms
  3. Create the paragraphs
  4. Add the terms to paragraph field
  5. Create the page
  6. Append the paragraphs to the page
  7. Save

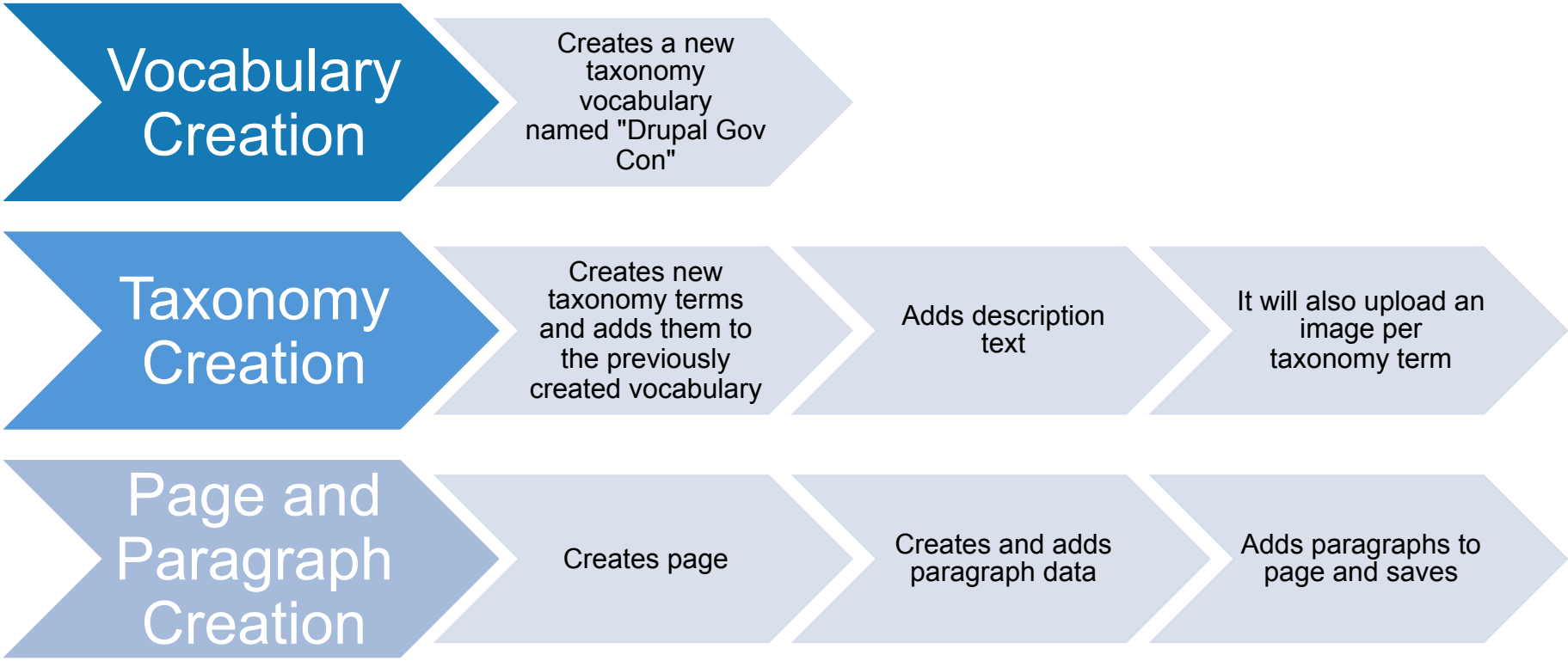
Code Setup In Module

- The post_update script needs to be inside of the module and named correctly:
  - **module_name.post_update.php**
- Functions inside this file need to be named correctly
  - **module_name_post_update_NAME()**
- Very Important: functions execute in alpha order and NOT in the order they are listed.
  - **module_name_post_update_abc** will execute first before **module_name_post_update_def**

Running Your Script

- The post update script can be executed using drush, with the usual command: **`drush updb`**
- It can also be executed as part of the update.php process through the UI.

Post Update Script : Examples

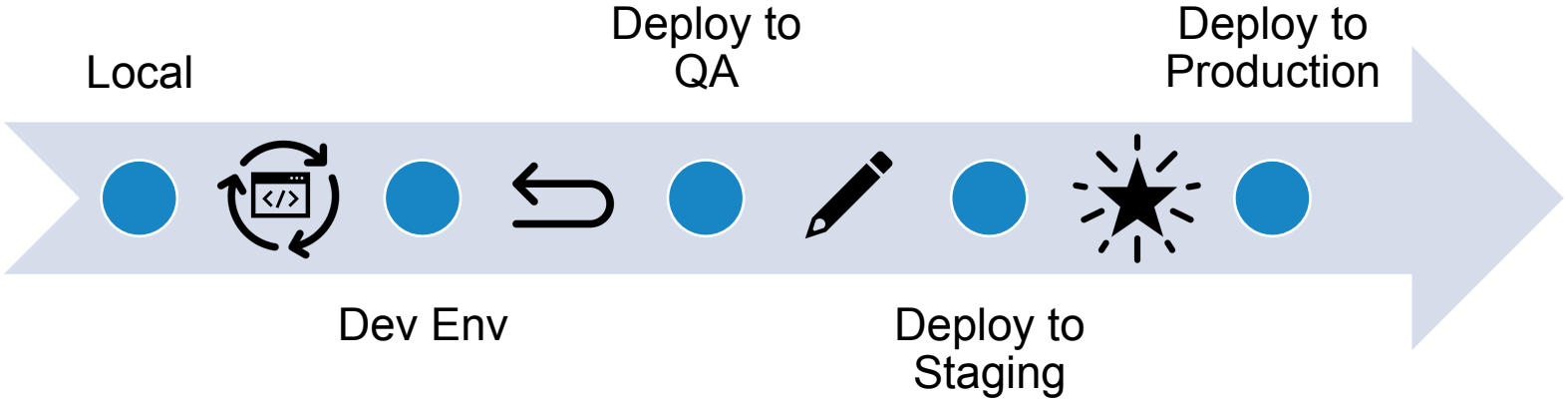


Demo of Post Update script

- This demo will be using Lando locally.
  You can find the code in a repo at:
  https://git.io/JURcc
- Repo includes example code and sample module
- Examples that will be executed in this demo
  - Vocabulary has already be created in configuration
  - Create taxonomy terms, add data, upload image, and add term to created vocabulary
  - Create paragraphs and add data
  - Create page
  - Append paragraphs to page and save

Deployment cycles with your script

- Develop script locally with finalized content coded IN the script
- Dummy content is fine for testing (but don't deploy it!)
- Test the script on the development environment often
- Don't spend much time QA'ing the script, just the content it creates
- Use staging as a final content edit with the client
- After deploying to production, all edits can be made directly in the site.
  - The script did the hard work of setting up all the fields!



Reverting Your Script

- In lower environments and locally as you are building your script, you'll need to revert it frequently using the following command:

```
drush php-eval '$update_hook_name = "module_name_post_update_name";
$key_value = \Drupal::keyValue('post_update');
$existing_updates = $key_value->get('existing_updates');
$index = array_search($update_hook_name,$existing_updates);
unset($existing_updates[$index]);
$key_value->set('existing_updates', $existing_updates);'
```

- Refrain from doing this in a production environment. The goal is to have this script run only once at launch.

Removing Test Data

- Running the script multiple times will duplicate content
- Remove all of the content it has created before running it again:

```
drush php-eval -e '$paragraphs = \Drupal::entityTypeManager()
    ->getStorage('paragraph')
    ->loadByProperties(array('type' => 'paragraph_type_machine_name'));
foreach ($paragraphs as $paragraph) {
    $paragraph->delete();
}'
```

- Backup your DB before your script and reimport each time

Pros & Cons of Update Script

Pros	Cons
Very quick to deploy large amounts of content	Significant time needs to be spent writing the script
Simple command to execute	Must be familiar with the Drupal API
Very flexible (can upload images, setup blocks and paragraphs, create taxonomy terms, etc)	This should be used as a “one shot” deployment plan
Useful to deploy the exact same content across environments	Caution must be exercised when using the script so as to not duplicate your content

Problems this solved for us...

Gov Agency office in charge of our deployments routinely ran drush updb for us so this would be familiar to them

Extremely fast setup of content, fields, images, etc

Not able to schedule deployments during off peak times.

Entire team didn't have to drop what they were doing to copy and paste content into site

Able to pinpoint exact time content will launch.

Tons of hours saved!

THANK YOU!
Questions?

Matt Calvert – Front End Developer
mcalvert@metrostarsystems.com

Jason Glisson – Drupal Developer
jglisson@metrostarsystems.com

MetroStar Systems
1856 Old Reston Avenue, Suite 100
Reston, VA 20190
www.metrostarsystems.com
703.481.9581