



DRUPALCAMP
OTTAWA

#DCO2025



Session Recordings

Speakers, hit the rec/red button!

Attendees, remind speakers about the record button!

All recorded sessions will be available online on the DrupalYOW youtube channel.





DRUPALCAMP
OTTAWA

#DCO2025

Drupal Spring Cleaning

**Steven Stapleton - Coldfront Labs
Inc - May 2, 2025**

HELLO!

I am Steven Stapleton

I am here today to talk about Update Hooks!!!

1.

The Problem

Exploring
Solutions



What Is Technical **Debt** and Why Clean It Up?

Streamline
content creation

Reduces
maintenance

Improve site
performance



Migrations **vs** Update Hooks

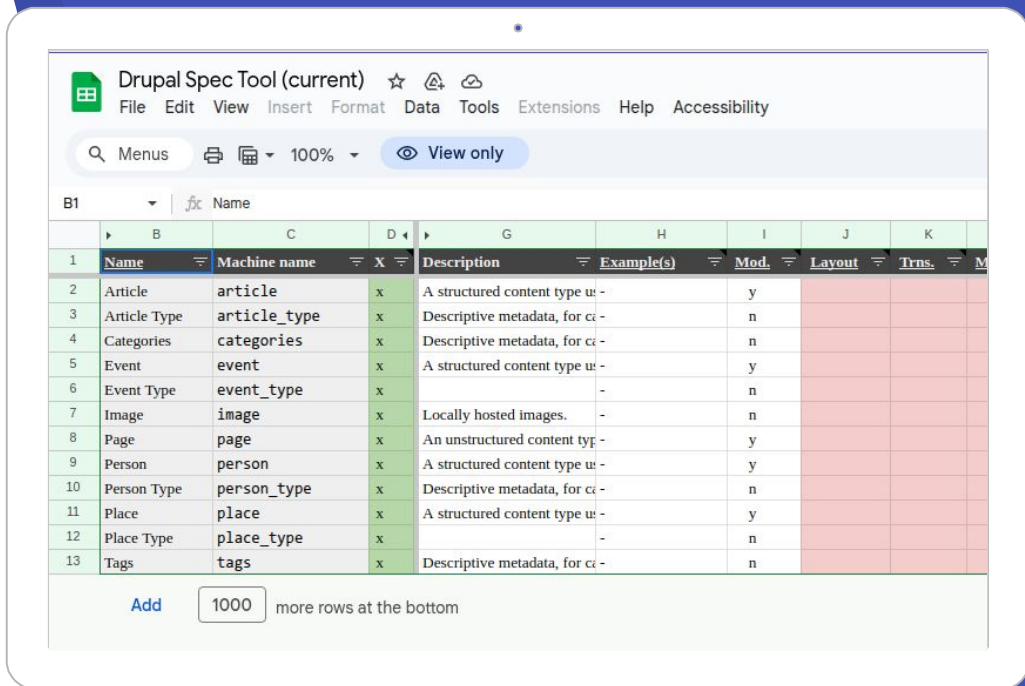
What are
migrations?

What are update
hooks?

When to use
update hooks?

Planning for Your Site's Clean Up (Drupal Spec Sheet)

<https://github.com/acquia/drupal-spec-tool>



The screenshot shows the Drupal Spec Tool (current) web application. The interface includes a menu bar with options: File, Edit, View, Insert, Format, Data, Tools, Extensions, Help, and Accessibility. Below the menu bar is a search bar with the text "Menus" and a "View only" button. The main content area displays a table with columns: B1, Name, Machine name, X, Description, Example(s), Mod., Layout, Trns., and M. The table lists various Drupal content types and their specifications.

B1	Name	Machine name	X	Description	Example(s)	Mod.	Layout	Trns.	M
1	Name	Machine name	X	Description	Example(s)	Mod.	Layout	Trns.	M
2	Article	article	x	A structured content type u	-	y			
3	Article Type	article_type	x	Descriptive metadata, for c	-	n			
4	Categories	categories	x	Descriptive metadata, for c	-	n			
5	Event	event	x	A structured content type u	-	y			
6	Event Type	event_type	x	-	-	n			
7	Image	image	x	Locally hosted images.	-	n			
8	Page	page	x	An unstructured content typ	-	y			
9	Person	person	x	A structured content type u	-	y			
10	Person Type	person_type	x	Descriptive metadata, for c	-	n			
11	Place	place	x	A structured content type u	-	y			
12	Place Type	place_type	x	-	-	n			
13	Tags	tags	x	Descriptive metadata, for c	-	n			

At the bottom of the table, there is an "Add" button and a text input field containing "1000", followed by the text "more rows at the bottom".

2.

Examples

Exploring
Solutions

WRITING AN UPDATE HOOK

- Naming convention

```
1  <?php
2
3  /**
4   * Implements hook_update_N().
5   */
6  function MODULE_NAME_update_N() {
7   ..// Load your data.
8   ..// ..
9   ..// Make your changes.
10  ..// ..
11  ..// Save your changes.
12 }
```

Loading Data

Entity Type Manager

```
\Drupal::entityManager()->getStorage($entity_type_id);
```



Node storage

...

Media storage

Loading Data

```
115  /**
116   * Implements hook_update_N().
117   */
118  function MODULE_NAME_update_N() {
119    // Get node storage to load nodes.
120    $entity_type_manager = \Drupal::entityTypeManager();
121    $entity_type_id = 'node';
122    $node_storage = $entity_type_manager->getStorage($entity_type_id);
123    $node_ids = [1, 2, 3, 4];
124
125    // Load nodes.
126    $nodes = $node_storage->loadMultiple($node_ids);
127
128    // Make your changes.
129    // ..
130    // Save your changes.
131  }
```

Mapping Fields on Content Types

```
115  /**
116   * Implements hook_update_N().
117   */
118  function MODULE_NAME_update_N() {
119    // Get node storage to load nodes.
120    $entity_type_manager = \Drupal::entityTypeManager();
121    $entity_type_id = 'node';
122    $node_storage = $entity_type_manager->getStorage($entity_type_id);
123    $node_ids = [1, 2, 3, 4];
124
125    // Load nodes.
126    $nodes = $node_storage->loadMultiple($node_ids);
127
128    // Make your changes.
129    foreach ($nodes as $node) {
130      if ($node->get('field_name')->isEmpty() == FALSE) {  Undefined method 'get'.
131        $field_value = $node->get('field_name');  Undefined method 'get'.
132        $new_value = $field_value . ' test';
133        $node->set('field_name', $new_value);  Undefined method 'set'.
134      }
135    }
136
137    // Save your changes.
138    $node->save();
139  }
```

Converting Files to Media Items

Label	Machine name	Field type
Image	field_image	Image
media	field_media	Entity reference Reference type: Media Media type: Image

Creating media items from
files



Get the file from the
field_image



Find the media item from the
file.



Attach the media item to
field_media

Creating media items from files

Loading file entities

```
115  /**
116   * Implements hook_update_N().
117   */
118  function MODULE_NAME_update_N() {
119    // Load the file entities.
120    $file_storage = \Drupal::entityTypeManager()->getStorage('file');
121
122    $files_ids = $file_storage
123      ->getQuery()
124      ->condition('fid')
125      ->sort('fid', 'ASC')
126      ->accessCheck(FALSE)
127      ->execute();
128
129    // Load the file entities.
130    $files = $file_storage->loadMultiple($files_ids);
131
132    // Create the media items.
133    // ..
134  }
135
```

Creating media items from files

Creating Media Items

```
136 /**
137  * Implements hook_update_N().
138  */
139 function MODULE_NAME_update_N() {
140   // Load the file entities.
141   // ...
142
143   foreach ($files as $file) {    Undefined variable '$files'.
144     $filemime = $file->get('filemime')->first()->getValue()['value'];
145
146     if($filemime == 'application/pdf') {    "filemime": Unknown word.
147
148       $media = Media::create([    Undefined type 'Media'.
149         'bundle' => 'document',
150         'uid' => $file->getOwnerId(),
151         'name' => $file->getFilename(),
152         'status' => $file->isPermanent(),
153         'created' => $file->getCreatedTime(),
154         'changed' => $file->getChangedTime(),
155         'field_media_document' => [
156           'target_id' => $file->id(),
157         ],
158       ];
159
160       $media->save();
161     }
162   }
163 }
164
```


Get the file from the field_image

Iterating over article nodes and getting the field_image

```
291 /**
292  * Implements hook_update_N().
293  */
294 function MODULE_NAME_update_N() {
295   $entity_type_manager = \Drupal::entityTypeManager();
296   $node_storage = $entity_type_manager->getStorage('node');
297   $article_nodes = $node_storage->loadByProperties(['type' => 'article']);
298
299   foreach ($article_nodes as $node) {
300     $file = $node->get('field_image')->entity;    Undefined method 'get'.
301     // ...
302   }
303 }
304
```

Find the media item from the file

Look up a media item based on some information from the file

```
291 /**
292  * Implements hook_update_N().
293  */
294 function MODULE_NAME_update_N() {
295   $entity_type_manager = \Drupal::entityTypeManager();
296   $node_storage = $entity_type_manager->getStorage('node');
297   $media_storage = $entity_type_manager->getStorage('media');
298   $article_nodes = $node_storage->loadByProperties(['type' => 'article']);
299
300   foreach ($article_nodes as $node) {
301     // Get the file entity from the article node.
302     $file = $node->get('field_image')->entity; Undefined method 'get'.
303     // Load the media item based on the file id.
304     $media = $media_storage->loadByProperties(['fid' => $file->id()]);
305
306     // ...
307   }
308 }
```

Attach the media item to field_media

Look up a media item based on some information from the file

```
291  /**
292   * Implements hook_update_N().
293   */
294   function MODULE_NAME_update_N() {
295     $entity_type_manager = \Drupal::entityTypeManager();
296     $node_storage = $entity_type_manager->getStorage('node');
297     $media_storage = $entity_type_manager->getStorage('media');
298     $article_nodes = $node_storage->loadByProperties(['type' => 'article']);
299
300     foreach ($article_nodes as $node) {
301       // Get the file entity from the article node.
302       $file = $node->get('field_image')->entity; Undefined method 'get'.
303       // Load the media item based on the file id.
304       $media = $media_storage->loadByProperties(['fid' => $file->id()]);
305       // Grab the first result.
306       $media = reset($media);
307       // Attach the media entity to the article node.
308       $node->get('field_image')->setValue([ Undefined method 'get'.
309         'target_id' => $media->id(),
310         'target_type' => 'media',
311       ]);
312
313       $node->save();
314     }
315   }
```

Handling Translations

```
291 /**
292  * Implements hook_update_N().
293  */
294 function MODULE_NAME_update_N() {
295   $entity_type_manager = \Drupal::entityTypeManager();
296   $node_storage = $entity_type_manager->getStorage('node');
297   $article_nodes = $node_storage->loadByProperties(['type' => 'article']);
298
299   foreach ($article_nodes as $node) {
300     // Get the available languages.
301     $translations = $node->getTranslationLanguages(); Undefined method '
302
303     // Get the translated node per language.
304     foreach ($translations as $langcode => $language) {
305       $translated_node = $node->getTranslation($langcode); Undefined met
306
307       // Perform your logic on the translated node then save after.
308
309       $translated_node->save();
310     }
311   }
312 }
```

Updating in Batches

```
291 /**
292  * Implements hook_update_N().
293  */
294 function MODULE_NAME_update_N(&$sandbox) {
295   // Initialize the sandbox, if not already initialized.
296   if (!isset($sandbox['progress'])) {
297     // If progress in the sandbox variable is not set.
298     // Then we set the progress to 0 we note the last node id as 0.
299     // Then we count the total number of nodes we are processing.
300     $sandbox['progress'] = 0;
301     $sandbox['last_nid'] = 0;
302     $sandbox['total'] = \Drupal::entityQuery('node')
303       ->condition('type', 'article')
304       ->accessCheck(FALSE)
305       ->count();
306     ->execute();
307   }
308
309   // Here we grab a list of node ids based on our current progress.
310   // So we only process these nodes for this batch.
311   $nids = \Drupal::entityQuery('node')
312     ->condition('type', 'event')
313     ->condition('nid', $sandbox['last_nid'], '>')
314     ->range(0, 20)
315     ->accessCheck(FALSE)
316     ->sort('nid', 'ASC')
317     ->execute();
318
319   // Perform your logic.
320   // Update sandbox last processed nid.
321   $sandbox['last_nid'] = max($nids);
322   $sandbox['progress'] += count($nids);
323
324   // Inform the batch engine whether we are finished or not.
325   if ($sandbox['progress'] >= $sandbox['total']) {
326     $sandbox['#finished'] = 1;
327   }
328   else {
329     $sandbox['#finished'] = $sandbox['progress'] / $sandbox['total'];
330   }
331 }
```

3.

Tips

Exploring
Solutions



BACKUP YOUR DATABASE!



DEPLOY NON DESTRUCTIVE
HOOKS FIRST!



**CHECK CONFIG EXISTS
BEFORE TRYING TO
REMOVE IT!**



**LOG THE STATUS OF
YOUR UPDATES!**

3.

Demo

Exploring
Solutions



THANKS!



Any questions?

You can find me at [@sstapleton](#) &
sstapleton@coldfrontlabs.ca





DRUPALCAMP
OTTAWA

#DCO2025





CREDITS

Special thanks to all the people who made and released these awesome resources for free:

- ▶ [Simple line icons](#) by Mirko Monti
- ▶ [E-commerce icons](#) by Virgil Pana
- ▶ [Streamline iconset](#) by Webalys
- ▶ Presentation template by [SlidesCarnival](#)
- ▶ Photographs by [Death to the Stock Photo](#) ([license](#))