Group 42 - 2017

Website Migration

Updated: July 25, 2017

(Draft)

Table of Contents

Site Inventory

Users & Roles

- 2 users
- Only default roles

Node Types

Example	
Purpose	Provide a GIST style content type
URL Pattern	example/[node:title]
Notes	No content. Was never used.

External Note	
Purpose	Content for Technical Reference section. Defines an external link for listing on the reference page.
URL Pattern	None
Notes	

Note	
Purpose	Content for Technical Reference section. Defines a short article for the technical reference.
URL Pattern	note/[node:title]
Notes	Comments enabled and content has comments.

Page	
Purpose	Standard website page.
URL Pattern	None
Notes	Comments not enabled.

Podcast	
Purpose	Podcast episode
URL Pattern	None
Notes	No content, never implemented.

Story	
Purpose	Article/post content
URL Pattern	[node:title]
Notes	Comments enabled and content has comments. File uploads field (file field) with image and text files.

Taxonomy Vocabularies

All vocabularies use the default pattern: [term:vocabulary]/[term:name]

- **Example Type**

  Classifies example nodes. Feature was never implemented. No terms defined.
- **Index**

  The organization for the *Technical Reference* section.
- **Notes**

  Has terms that are similar to technical reference. Not sure how it's used.
- **Series**

  Empty. Was created for a feature that was never implemented.
- **Sphere**

  Empty. Was created for a feature that was never implemented.

- **Topics**

  The “tags” vocabulary.

Text Formats

Name	ID	Filters	Roles
Filtered HTML	1	Limit HTML tags Convert line breaks Correct HTML	authenticated anonymous
PHP code	2		
Full HTML	3	Authenticated Code filter Correct HTML	authenticated
WYSIWYG	4	Correct HTML	authenticated
Plain text	5		

Image Handling

All images are handled by storing images in *File Uploads* field and referencing the URL in content image tag. There are no additional modules (such as Media module) managing files, and there is no WYSIWYG plugin support.

Modules

The following modules are enabled.

Core

- Block (block)
- Comment (comment)
- Contact (contact)
- Contextual links (contextual)
- Database logging (dblog)
- Field (field)
- Field SQL storage (field_sql_storage)
- Field UI (field_ui)
- File (file)
- Filter (filter)
- Help (help)
- List (list)
- Menu (menu)
- Node (node)
- Number (number)
- Options (options)
- Path (path)
- PHP filter (php)
- RDF (rdf)
- Search (search)
- Shortcut (shortcut)
- System (system)

- Taxonomy (taxonomy)
- Text (text)
- Trigger (trigger)
- Update manager (update)
- User (user)

Contrib

- Administration Development tools (admin_devel)
- Administration menu (admin_menu)
- Administration menu Toolbar style (admin_menu_toolbar)
- Chaos tools (ctools)
- Features (features)
- Link (link)
- Automatic Nodetitles (auto_nodetitle)
- Backup and Migrate (backup_migrate)
- Code Filter (codefilter)
- Libraries (libraries)
- MediaElement.js (mediaelement)
- Mollom (mollom)
- Pathauto (pathauto)
- RSS enclosure (rss_enclosure)
- Strongarm (strongarm)
- Token (token)
- Global Redirect (globalredirect)
- Google Analytics (googleanalytics)
- Wysiwyg (wysiwyg)
- Views (views)
- Views UI (views_ui)

Custom

- Group 42 Site Module (group42site)

Features

- Group 42 Blog (group_42_blog)
- Group 42 Common (group_42_common)

Migration Overview

The migration will import content into a new Group 42 site, it will not attempt to upgrade the existing site. The new site does technical reference in a different way from the old site. The existing technical reference content will not be migrated.

Drupal Migrate

The following entities will be migrated with the *Migrate* module set:

- Users
- Topic vocabulary (transformed to Drupal 8 Tags vocabulary)
- Managed files
- Story nodes (transformed to Article)
- Comments

Manual Migration

- The *About* page will be manually created
- Technical Reference Section

  Technical reference is completely different in the new site. Still relevant information will be manually entered into new system.

  What to do with old pages is still under review, possibilities:

  - Drop completely
  - Remap URLs
  - Make a note on the 404 page

Special Handling

- Story node image references

  Recreating the images so they are managed by the CKEditor inline image plugin.

- Code examples (aka snippets) in content

  A variety of methods have been used for code snippets. The content will require a transform into the new method for Drupal 8.

- Managed files

  Managed file are moving to two directories based on extension: Images to inline-images directory and text files to the directory for Article Files field.

Migration Specifics

Migration details are referenced by destination. This is more easily compared to the YAML definition files. Because content is being dropped or moved it is also a conciser way of presenting the information.

User Import

- Default user migration provided by Drupal migrate

Tag Vocabulary Import

- All *Topic* vocabulary terms will be migrated to *Tags* vocabulary
- No other transforms are required

File Import

- Managed files are a combination of image and text files
- Managed image files are associated with images used in the node body. They should be place in the inline-image directory.
- Text files are additional content supporting the article and should go to the Article node *Files* field directory

Article Import (Story to Article nodes)

- *Story* nodes migrated to *Article* nodes
- *Tags* vocabulary imported from *Topics* vocabulary
- *Files* field:
  - Only text managed files should be added to this field
- *Body* Field:
  - Text formats: All content will be set to full_html
  - Where a managed image file exists for an image, the associate image tag should be updated to use the new src path; inline-image attribute set: data-entity-type, data-entity-uuid; and placement classes (e.g., align-left, align-right).
  - Code examples are managed with a new module, *Code Snippets*, which adds the CKEditor *Code Snippets* plugin. Existing body content needs to be updated with syntax for the new plug-in.
- *Section* Vocabulary:

  The value for *Section* vocabulary terms is inferred from the *Story* node *Topic* terms.

Comment Migration

- All *Story* node comments migrated to associated *Article* nodes.
- Comments for other node types must be excluded.

Features

Features functionality does not require migration.

Module Replacement Considerations

Because we are migrating content into a new site this section specifically concerns any transforms required because of module changes. Broader issues coming from module changes would be addressed in the specification and build for the new site.

Contrib

Link (link)	The link field is not used in any content being migrated. In any event, it is part of Drupal 8 core and has a migration implementation and would not have been an issue.
Automatic Nodetitles (auto_nodetitle)	Automatic Nodetitles is not used for content display and is therefore not an issue.
Code Filter (codefilter)	Replaced by the *CKEditor CodeSnippet* module. Content will require transform to new format.
MediaElement.js (mediaelement)	*MediaElement.js* used to implement an audio player for *Podcast* content type. It is not used for any content being migrated.
RSS enclosure (rss_enclosure)	RSS enclosure used to implement a podcast feed. It is not used for any content being migrated.
Wysiwyg (wysiwyg)	Replaced by CKEditor in Drupal 8 core. Some of the *Story* node content may require transform to work properly.

Custom

Group 42 Site Module (group42site)	Replacement is not required.

URL Transforms

Under investigation.

Questions:

- Will path-auto reproduce URLs for stories
- Do we want a map for Technical Reference content

Post Migration Tasks

- Scan for missing images
- Scan for broken URLs

Workflow

Overview

- Migration code work happens on a GIT feature branch
- GIT *rebase* is used to keep the migration feature branch in sync with other work
- A rebuild/reset script returns the site to it's pre-migration state
- A *make content* script adds associated new content that can't be migrated

Drush Scripts

- g428-reset

  Restores the site to its pre-migration state
- g428-make-content

  Runs the migration using Drush and adds any additional content

Commands

The following commands are normally run inside the scripts. It's sometimes useful to run them outside the script so they are documented here:

- `$ drush8 config-import --partial --source=modules/custom/g42migrate/config/install/ -y`
- `$ drush8 migrate-import --group=migrate_drupal_7 --update`
- `$ drush8 migrate-import g42_migrate_d7_node_story --update`
- `$ drush8 ms --group=migrate_drupal_7`
- `$ drush8 mrs g42_migrate_d7_node_story`

Appendix 1: Custom Migration Plugins

The following custom Migrate process plugins were written for the Group 42 migration project.

convert_to_codesnippets

Replace old code block tags with CodeSnippets module tag convention.

Module:

Group 42 Migrate (g42migrate)

YAML Usage:

```
field_article_body:
  plugin: iterator
  source: body
  process:
    value:
      plugin: convert_to_codesnippets
      source: value
```

infer_g42_section (Class: InferG42Section)

Takes Drupal 7 *Topic* term id as input and outputs a corresponding Drupal 8 *Section* term if the topic belongs in the section.

Module:

Group 42 Migrate (g42migrate)

YAML Usage:

```
field_section:
  plugin: iterator
  source: taxonomy_vocabulary_1
  process:
    target_id:
      process:
        plugin: infer_g42_section
        source: tid
```

setup_inline_images (Class: SetupInlineImages)

Iterate through all img tags in text. If filename matches a managed file, update the img tag to use the path and inline image attributes.

Module:

Group 42 Migrate (g42migrate)

YAML Usage:

```
field_article_body:
  plugin: iterator
  source: body
  process:
    value:
      plugin: setup_inline_images
      source: value
```

skip_unless_text_file (Class: SkipUnlessTextFile)

Skip processing the current row if the fid does not represent a text file.

Module:

Group 42 Migrate (g42migrate)

YAML Usage:

```
field_file_uploads:
  plugin: iterator
  source: upload
  process:
    target_id:
      plugin: skip_unless_text_file
      method: process
      source: fid
    display: display
    description: description
```