

# Migrating Multilingual Content to Drupal 8

# Jigar Mehta aka Jerry



Drupal Developer @ Evolving Web

PHP since 2008

Drupal since 2013

**jigarius** on drupal.org, linkedin, twitter, etc.

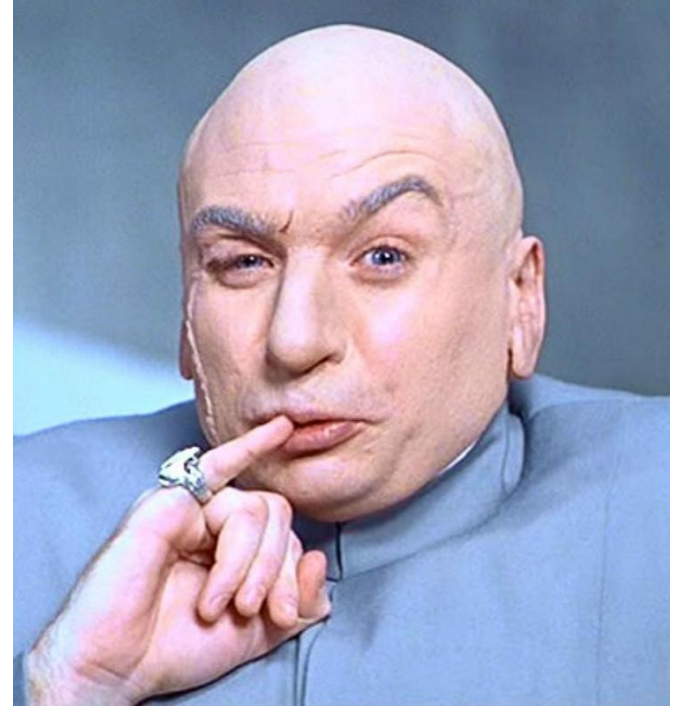
These slides are at [bit.ly/drupal8-i18n-migration](https://bit.ly/drupal8-i18n-migration)

# Our Clients



# Yoyo! What's this talk about?

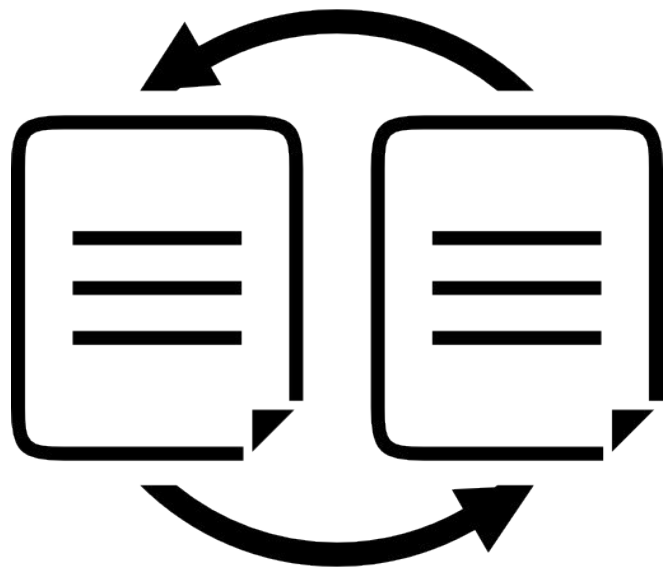
- A brief introduction to migrations and what they do.
- What's so special about migrating translated content?
- Migrating content from D7 to D8.
  - Migrating nodes translated with the **content\_translation** module to D8.
  - Migrating nodes translated with the **entity\_translation** module to D8.
- Migrating other translatable entities to D8.



# Introduction to Migrations

# Migrations: A brief introduction

- Migrate API helps us **migrate** data from a given **source** to a given **destination**.
- It **maintains a map** of old records and the new records and uses it to do cool stuff like **updating changed records, rolling back**, etc.
- **Provides a framework** for us to write migrations in a standardized way, so that **we can focus on the data to be migrated** (instead of trying to reinvent the “migrate” module).



# Migrations: Example usages

- Generate Drupal (node, terms, etc) entities from data in CSV / JSON / XML and other sources.
- Import entities from Drupal 5, 6 and 7 into Drupal 8.
- Import data from other content management systems like Wordpress, Joomla!, etc.
- Import data from the human brain into Drupal 8? Maybe that will be possible in Drupal 9! Just kidding.



# Migrations in Drupal 8



# Migrations in Drupal 8: Introduction

- Define a custom module to contain our migration-related code and migration definitions (optional).
  - Alternatively, you can maintain the yml files directly in the site's global config directory like `migration_plus.migration.fooobar.yml`.
- Write **YAML files to define migrations** and migration groups.
  - Put your migration `.yml` files inside your module's `config/install` directory.
  - To customize how things work, **write custom plugin** implementations.
- Use **drush** commands to see the status of and execute migrations.
  - `drush migrate-status (ms)`, `drush migrate-import (mi)`, `drush migrate-rollback (mr)`.
  - `--group=xx --tag=xx --all` to choose which migrations to work with.
  - `--update` to update records which have already been migrated (when source data is updated).

# Migration definition: Metadata

**id:** program\_data

**label:** Academic programs

**description:** Migrates academic program data.

**migration\_group:** c11n

**migration\_tags:**

- node
- academic program

**migration\_dependencies:**

optional:

- migration\_a

required:

- migration\_b

# Migration definition: Source

## source:

**plugin:** csv

**path:** 'public://import/program/program.data.csv'

**header\_row\_count:** 1

## keys:

- ID

## fields:

ID: Unique identifier for the program as in the data source.

title: Name of the program.

body: A description for the program.

## constants:

uid\_root: 1

# Migration definition: Process

## process:

**title:** heading

**uid:** constants/uid\_root

## **status:**

-

plugin: callback

source: published

callable: strtolower

-

plugin: static\_map

map:

yes: 1

no: 0

# Migration definition: Destination

**destination:**

**plugin:** entity:node

**default\_bundle:** program

# Migraciones Multilingües

Multilingual migrations

# Multilang Migrations: What's so different?

1. First, we migrate things in the base language.
  - At this stage, ignore translations. Migrate just the base data.
2. Next, we migrate the translations!
  - At this stage, ignore the base data or the “non-translations”.
  - Make sure that these translations get properly related to the entities in the base language (the ones we import in step 1 above).
  - Make sure the base data is imported before the translations are imported (with migration dependencies).

# Multilang Migrations: From D7 to D8

- Drupal 7 has more than one way to translate entities.
- Nodes can be translated using the core **content\_translation** module.
  - Each translation results in a separate node.
  - Since Drupal 8.3, core has support for migrating nodes translated with the *content\_translation* module.
- Nodes can be translated using the **entity\_translation** module.
  - Only one node - no matter how many translations.
  - Translations happen at the field level.
  - Drupal 8.x core doesn't yet have out-of-the-box support for migrating entity translations.





# Migrating nodes to D8

The content\_translation way

# Extract of the D7 *node* table

<b>nid</b> The primary identifier for a node.	<b>type</b> The node_type.type of this node.	<b>vid</b> The current node_revision.vid version identifier.	<b>tnid</b> The translation set id for this node, which equals...	<b>language</b> The languages.language of this node.	<b>title</b> The title of this node, always treated as non-mark...	<b>uid</b> The users.uid that owns this node; initially, this...	<b>status</b> Boolean indicating whether the node is published (...)
20	article	20	20	en	German shepherd dog	1	1
21	article	21	20	fr	Chien berger allemand	1	1
22	article	22	20	es	Perro pastor alemán	1	1

# migrate + content\_translation: Base data

**id:** program\_base

**label:** Programs in base language

**source:**

**plugin:** d7\_node

**node\_type:** article

**translations:** false

  ...

**destination:**

**plugin:** 'entity:node'

**translations:** false

**process:**

  title: title

  ...

# migrate + content\_translation: Translations

**id:** program\_i18n

**label:** Program translations

**source:**

**plugin:** d7\_node

**node\_type:** article

**translations:** true

...

**process:**

**nid:**

plugin: migration // migration\_lookup

source: tnid

migration: program\_base

...

**destination:**

**plugin:** 'entity:node'

**translations:** true

**process:**

title: title

...

**migration\_dependencies:**

required:

- program\_base

# Migrating nodes to D8


















The `entity_translation` way

# migrate + entity\_translation: An Overview



- D8 core has no out-of-the-box support for them - which means, we need to write some code.
- **Migration remains the same!**
- Need to **override the source plugin** `d7_node...`
  - To take the **entity\_translation** table into account.
  - To make the **translations** parameter work correctly.
  - To load field data in the correct language while reading translations.

# Extract of the *entity\_translation* table

		<b>entity_type</b> <small>The entity type this translation relates to</small>	<b>entity_id</b> <small>The entity id this translation relates to</small>	<b>revision_id</b> <small>The entity revision id this translation relates to</small>	<b>language</b> <small>The target language for this translation.</small>	<b>source</b> <small>The source language from which this translation wa...</small>	<b>uid</b> <small>The author of this translation.</small>		
<input type="checkbox"/>	 Edit	 Copy	 Delete	node	1	1	en		1
<input type="checkbox"/>	 Edit	 Copy	 Delete	node	1	1	es	en	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	node	1	1	fr	en	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	node	2	2	en		1
<input type="checkbox"/>	 Edit	 Copy	 Delete	node	2	2	es	en	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	node	2	2	fr	en	1

# Extract of the *field\_data\_title\_field* table

<b>entity_id</b> The entity id this data is attached to	<b>revision_id</b> The entity revision id this data is attached to, o...	<b>language</b> The language for this data item.	<b>delta</b> The sequence number for this data item, used for m...	<b>title_field_value</b>
1	1	en	0	Lernaean Hydra
1	1	es	0	Hidra de Lerna
1	1	fr	0	Hydre de Lerne
2	2	en	0	Cerberus
2	2	es	0	Cerbero
2	2	fr	0	Cerbère



# migrate + entity\_translation: Source Plugin

```
namespace Drupal\migrate_example_i18n\Plugin\migrate\source;
```

```
use Drupal\node\Plugin\migrate\source\d7\Node as D7Node;
```

```
/**  
 * Drupal 7 node migrate source with "entity_translation" support.  
 *  
 * @MigrateSource(  
 *   id = "d7_node_entity_translation"  
 * )  
 */
```

```
class D7NodeEntityTranslation extends D7Node {}
```

# migrate + entity\_translation: Source Plugin

```
protected function handleTranslations(SelectInterface $query) {  
    ...  
    // Entity translation data is kept in the entity_translation table.  
    $query->join('entity_translation', 'et',  
        "et.entity_type = :entity_type AND et.entity_id = n.nid",  
        [':entity_type' => 'node']  
    );  
    // Use only originals, or only translations, depending on our configuration.  
    $operator = empty($this->configuration['translations']) ? '=' : '<>';  
    $query->condition('et.source', "", $operator);  
    ...  
}
```

# migrate + entity\_translation: Source Plugin

```
public function getIds() {  
    $ids = parent::getIds();  
    // With Entity Translation, each translation has the same node ID.  
    // Therefore, we need both nid and language to identify source records uniquely.  
    $ids['language'] = [  
        'type' => 'string',  
        'alias' => 'et',  
    ];  
    return $ids;  
}
```

# migrate + entity\_translation: Source Plugin

```
// Make sure the current source record is loaded in the correct language by passing
// an additional $language parameter to the function getFieldValues().
public function prepareRow(Row $row) {}
// Load field data for the current record in the correct language by taking the $language
// argument into consideration.
protected function getFieldValues($entity_type, $field, $entity_id, $revision_id = NULL,
$language = NULL) {}
```

# Migrating other entities

like taxonomy terms, users, etc.

# Migrating other translated entities from D7 to D8

- Use the appropriate **source** plugin. Example: `d7_user`, etc.
- Override the source plugin if the D7 site uses the **entity\_translation** module.
- Set the **translations** parameter to **true** while migrating translations.
- While migrating translations, we must **lookup the ID of the entity** in the base language. This makes sure that the translations get attached to the base data correctly.
- Make the **translation migration depend on the base data migration** to ensure that the base data is imported before the translations.

# More on Migrations?

# Next steps

- See <https://github.com/jigarius/drupal-migration-example>
- See [https://github.com/evolvingweb/migrate\\_example\\_i18n](https://github.com/evolvingweb/migrate_example_i18n)
- Read migration related articles on migrations on [evolvingweb.ca/blog](https://evolvingweb.ca/blog)
  - Migrating translated content from D6 and D7 to D8.
  - Migrating XML / JSON / CSV content to D8.
  - Migrating taxonomy terms (entity references) to D8.
  - Migrating files / images to D8.
- Read (and contribute to) documentation on [drupal.org](https://drupal.org).
- Go through the **migrate\_plus** and **migrate\_tools** modules.
- Try writing your own migrations!



# Merci Beaucoup / Thank You

- See these slides on [bit.ly/drupal8-i18n-migration](https://bit.ly/drupal8-i18n-migration).
- I am **jigarius** on drupal.org, linkedin, twitter, etc.
- We are **Evolving Web**, visit our site for more Drupal stuff.

# Life-saving tips for developers!

- Make sure you get sufficient exercise to **have a healthy heart**. If the core has a bug, the site will crash!
- Make sure you stretch your arms, especially your fingers to avoid **carpal tunnel syndrome**.
- Use proper equipment to **make sure your eyes are relaxed**.
- Ensure a **good posture** to ensure a healthy back and neck.

