Drupal 8 Module Development: Mad with Power!

Ted Bowman - Acquia
Me: Ted Bowman aka tedbow

- Acquia - Office of the CTO
- Working on Drupal 8 UX and REST
- Module Developer
- Trainer(former life)
You?

- Module developers?
- New to Drupal 8?
- New to Drupal?
- Symfony?
- OOP?
Mad with Power?
Drupal 8 changes the game
Today’s examples

- Powerful
- New
- Cool
- Hard to figure out
Code:

https://github.com/tedbow/Drupal-8-Power
Today:

- OOP
- Importance of an IDE
- Base Fields
- Extending Classes
- Creating Condition Plugins
- Importance of an IDE
An IDE and OOP
Drupal 8 is Object Oriented

- More Complicated but . . .
  - More encapsulated
  - More self documenting
  - Easier to learn
- If . . .
  - You use an Good IDE
Drupal 8 OOP & IDE

- Examples in each module
- Stop and demo
Base Fields
Base fields: Everything is a field

- Node: author, created, changed, content type
- User: name, mail, password, roles
Base fields: Advantages

- Use field widgets
- Use field formatters
- Can be configurable on view and form
- Can add new base fields
- Can alter existing base fields
Base Field Examples
Base fields: Node Author

- Entity Reference to user
- Can use all entity reference:
  - Formatters
  - Widgets
- Form: configurable
- View: not configurable
Base fields: User Roles

- Entity Reference to role (config entity)
  - What??????????
- Can use all entity reference:
  - Formatters - some
  - Widgets
- Form: not configurable
- View: not configurable
Example Module: Real Author

- Tracks who really wrote content
- Separates Drupal user from author
- Adds new base entity reference field
  - Target type: user
  - Could use content
- Remove editing and configuration of author field
Example Module: Real Author

- Implements
  - Hook_entity_base_field_info
    - Add new field
  - Hook_entity_base_field_info_alter
    - Remove access to author field
Content items can be edited using different form modes. Here, you can define which fields are shown and hidden when Article content is edited in each form mode, and define how the field form widgets are displayed in each form mode.

<table>
<thead>
<tr>
<th>FIELD</th>
<th>WIDGET</th>
<th>Show row weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>Textfield</td>
<td></td>
</tr>
<tr>
<td>Body</td>
<td>Text area with a summary</td>
<td></td>
</tr>
<tr>
<td>Tags</td>
<td>Autocomplete (Tags style)</td>
<td></td>
</tr>
<tr>
<td>Image</td>
<td>Image</td>
<td></td>
</tr>
<tr>
<td>Real Author</td>
<td>Autocomplete</td>
<td></td>
</tr>
</tbody>
</table>
### Add Field

<table>
<thead>
<tr>
<th>LABEL</th>
<th>MACHINE NAME</th>
<th>FIELD TYPE</th>
<th>OPERATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>body</td>
<td>Text (formatted, long, with summary)</td>
<td>Edit</td>
</tr>
<tr>
<td>Comments</td>
<td>comment</td>
<td>Comments</td>
<td>Edit</td>
</tr>
<tr>
<td>Image</td>
<td>field_image</td>
<td>Image</td>
<td>Edit</td>
</tr>
<tr>
<td>Tags</td>
<td>field_tags</td>
<td>Entity reference</td>
<td>Edit</td>
</tr>
</tbody>
</table>
Comment: The data table for node entities.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>real_author</td>
<td>int(11)</td>
<td></td>
</tr>
<tr>
<td>nid</td>
<td>int(10) unsigned</td>
<td></td>
</tr>
<tr>
<td>vid</td>
<td>int(10) unsigned</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>varchar(32)</td>
<td>The ID of the target entity.</td>
</tr>
<tr>
<td>langcode</td>
<td>varchar(12)</td>
<td></td>
</tr>
<tr>
<td>title</td>
<td>varchar(255)</td>
<td></td>
</tr>
<tr>
<td>uid</td>
<td>int(10) unsigned</td>
<td>The ID of the target entity.</td>
</tr>
<tr>
<td>status</td>
<td>tinyint(4)</td>
<td></td>
</tr>
<tr>
<td>created</td>
<td>int(11)</td>
<td></td>
</tr>
<tr>
<td>changed</td>
<td>int(11)</td>
<td></td>
</tr>
<tr>
<td>promote</td>
<td>tinyint(4)</td>
<td></td>
</tr>
<tr>
<td>sticky</td>
<td>tinyint(4)</td>
<td></td>
</tr>
<tr>
<td>revision_translation_affected</td>
<td>tinyint(4) NULL</td>
<td></td>
</tr>
<tr>
<td>default_langcode</td>
<td>tinyint(4)</td>
<td></td>
</tr>
</tbody>
</table>
Example Module: Real Author

- OOP?
  - Uses old school hooks
  - Benefits from Drupal 8 classes.
  - Let’s see.....
Other ideas for new base fields

- Term reference: Main Site Categories
- User reference: Taxonomy Term owner
- DateTime: First published date
- DateTime: User blocked date
- Integer: Original Import Id
Other ideas for altering base fields

- Remove access to published date
- Make Roles Widget Configurable
- ???????
Example Module: Show User Fields

- Makes hidden user base fields
  - Viewable
  - Configurable
- Fields
  - Roles
  - Last Login
  - Last Access
Example Module: Show User Fields

- Implements
  - hook_entity_base_field_info_alter
    - Change fields to be display configurable
Example Module: Show User Fields

- OOP?
  - Uses old school hooks
  - Benefits from Drupal 8 class.
  - Let’s see…..
admin

Member for 57 minutes 48 seconds

Last access
1 minute 36 seconds ago

Last login
38 minutes 53 seconds ago

Roles
Administrator
<table>
<thead>
<tr>
<th>FIELD</th>
<th>LABEL</th>
<th>FORMAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Picture</td>
<td>– Hidden –</td>
<td>Image ▼</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Image style: Thumbnail (100×100)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Linked to content</td>
</tr>
<tr>
<td>Member for</td>
<td></td>
<td>Visible ▼</td>
</tr>
<tr>
<td>Last access</td>
<td>Above ▼</td>
<td>Time ago ▼</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Future date: Past date:</td>
</tr>
<tr>
<td>Last login</td>
<td>Above ▼</td>
<td>Time ago ▼</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Future date: Past date:</td>
</tr>
<tr>
<td>Roles</td>
<td>Above ▼</td>
<td>Label ▼</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Link to the referenced entity</td>
</tr>
</tbody>
</table>
Example Module: No Access Reference Label

- Allows view of referenced entities’ labels without entity view access
- Links to viewable entities
- Acts like author label for other references.
- NO HOOKS!
post 1

Submitted by on Sat, 07/09/2016 - 07:01

asdf

Linked Content

art 2a

You cannot view me!
No Access Reference Label: OOP?

- Yes!
- Creates new plugin
  NoAccessEntityReferenceLabelFormatter
- Extends EntityReferenceLabelFormatter
- Overrides 2 methods
- Let’s see
Implementing Plugin:

Where to Start?
– Find a similar plugin
– Can you extend it?
– Can you extend its parent?
– Investigate its interface(s): What else implements them?
Find a similar plugin

IDE FTW!

Press ↑ or ↓ to navigate through the history
Other ideas for extending field plugins
Condition Plugins
Condition Plugins: Unified

- Simple conditions evaluate to TRUE or FALSE
- Used to determine Block Visibility in core
- Also by Panels and Rules!
Example Module: Author Conditions

- Condition: Node author has roles
- Extends core User Role condition
- NO HOOKS!
Questions?