

# Drupal 7 Entities

Neal Burns

San Diego Drupal Users Group

# Entities: What are they?

- A unifying abstraction for Drupal-aware objects
- Entities in Drupal core:
  - Comments
  - Nodes
  - Files
  - Taxonomy terms
  - Taxonomy vocabularies
  - Users

# Parts of an entity

- An entity is typically a wrapper around a database table  
(the 'base table' key in `hook_entity_info()`)
- Entities can have fields  
(set 'fieldable' to TRUE in `hook_entity_info()`)
- Entities can have revisions  
(the 'revision table' key in `hook_entity_info()`)
- All of the parts are optional

# The entity subsystem uses objects

- By default, an entity in memory is a StdClass object
- There is additionally a “controller” class
  - This is an object factory for your entity type
  - Drupal provides a default controller that provides query-building and caching capabilities

# The entity object

- Properties
  - By default, correspond to the fields in the base and revision tables
  - Must contain a unique id
    - The id is typically the primary key in the base table
- The database schema is entirely up to you
  - The `hook_entity_info()` fields that describe the database are used by `DrupalDefaultEntityController`
  - You can substitute any controller (factory) class, even one that doesn't use a database

# The controller class

```
• interface DrupalEntityControllerInterface {  
•  
• /**  
• * Constructor.  
• *  
• * @param $entityType  
• * The entity type for which the instance is created.  
• */  
• public function __construct($entityType);  
•  
• /**  
• * Resets the internal, static entity cache.  
• *  
• * @param $ids  
• * (optional) If specified, the cache is reset for the entities with the  
• * given ids only.  
• */  
• public function resetCache(array $ids = NULL);  
•  
• /**  
• * Loads one or more entities.  
• *  
• * @param $ids  
• * An array of entity IDs, or FALSE to load all entities.  
• * @param $conditions  
• * An array of conditions in the form 'field' => $value.  
• *  
• * @return  
• * An array of entity objects indexed by their ids. When no results are  
• * found, an empty array is returned.  
• */  
• public function load($ids = array(), $conditions = array());  
• }  
•
```

Not much is required:

Just a loader for a list of entity ids

There is also built-in support for caching

# Entities can have **fields**

- **Fields** in Drupal 7 correspond to CCK in Drupal 6
- **Fields** are implemented by the **Field** module in core
- The **Field** module adds the concept of **bundles** to entities
- A **bundle** can usually be thought of as a subtype of an entity

# A **bundle**?

- A **bundle** is like a subtype of a fieldable entity
- Each **bundle** has its own set of fields
- Example: nodes
  - Entity type: ‘node’
  - **Bundles**: ‘article’, ‘page’
- Why is it called a **bundle**?
  - Maybe because it’s a “bundle” of fields?





# Bundles: to clarify

- Bundles are not part of the entity subsystem proper, they are part of the field subsystem
- Bundles are only made up of fields
  - Bundles are commonly described as “subtypes” of entities, but it’s only an analogy
  - You don’t have a separate database table per bundle
  - The list of bundles for an entity type can be hard-coded or stored dynamically in the database
  - Entities are not required to have bundles, unless they are fieldable

# Properties and Fields

- There are two ways to attach information to an entity: “Properties” and Fields
  - Properties live in the entity’s own tables, and are managed entirely by the entity
    - Drupal will load them for you, but that’s it
  - Fields live in the field system, and are completely taken care of on behalf of the entity
    - Fields have “widgets” to provide form fields, “formatters” to render output, configuration forms, etc.

# A simple core Entity: File

Entity info  

[Home](#)

```
... (Array, 5 elements)
  file (Array, 14 elements)
    label (String, 4 characters ) File | (Callback) File();
    base table (String, 12 characters ) file_managed
    entity keys (Array, 4 elements)
    static cache (Boolean) FALSE
    fieldable (Boolean) FALSE
    controller class (String, 29 characters ) DrupalDefaultEntityController
    field cache (Boolean) TRUE
    load hook (String, 9 characters ) file_load | (Callback) file_load();
    bundles (Array, 1 element)
      file (Array, 2 elements)
    view modes (Array, 0 elements)
    translation (Array, 0 elements)
    schema_fields_sql (Array, 1 element)
    save callback (String, 9 characters ) file_save | (Callback) file_save();
    deletion callback (String, 27 characters ) entity_metadata_delete_file | (Callback) entity_metadata_delete_file();
  node (Array, 20 elements)
  taxonomy_term (Array, 20 elements)
  taxonomy_vocabulary (Array, 17 elements)
  user (Array, 19 elements)
```

Krumo version 0.2.1a | <http://krumo.sourceforge.net>

# hook\_entity\_info()

```
/**  
 *  
 * Entity's machine name  
 */  
function system_entity_info() {  
  return array(  
    'file' => array(  
      'label' => t('File'),  
      'base table' => 'file_managed',  
      'entity keys' => array(  
        'id' => 'fid',  
        'label' => 'filename'  
      )  
    )  
  );  
}
```

Entity's machine name

Database table

Tells Drupal which property on the entity\* contains the entity's *id*

There are no bundles, because files can't have fields\*

\* see next slide

# Subtleties

- The 'entity keys' refer to properties on the entity object in memory. These ordinarily correspond to fields in the database table, which are automatically loaded into object properties by `DrupalDefaultEntityController`.
- Drupal automatically gives each entity at least one bundle, even if it doesn't declare one. This bundle only matters if the entity is fieldable. (The name is the same as the entity's.)

# hook\_schema()

```
$schema['file_managed'] = array(  
  'description' => 'Stores information for uploaded files.',  
  'fields' => array(  
    'fid' => array(  
      'description' => 'File ID.',  
      'type' => 'serial',  
      'unsigned' => TRUE,  
      'not null' => TRUE,  
    ),  
    'uid' => array(  
      'description' => 'The {users}.uid of  
      'type' => 'int',  
      'unsigned' => TRUE,  
      'not null' => TRUE,  
      'default' => 0,  
    ),  
    'filename' => array(  
      'description' => 'Name of the file with no path components. This may differ from the basenan  
      'type' => 'varchar',  
      'length' => 255,  
      'not null' => TRUE,  
      'default' => '',  
    ),  
    'uri' => array(  
      'description' => 'The URI  
      'type' => 'varchar',  
      'length' => 255,  
      'not null' => TRUE,  
      'default' => '',  
    ),  
    'filemime' => array(  

```

fid: This is the *id* in  
hook\_entity\_info()

filename: This is the *label* in  
hook\_entity\_info()

The other database fields are  
just file-specific properties

# The EntityAPI Module

- EntityAPI (machine name: entity) is a contrib module that provides a more sophisticated Controller class than comes with core
- EntityAPIController adds full CRUD (Create, Read, Update, Delete) capabilities, which takes care of most of the boilerplate code in a typical custom entity
- EntityAPI also provides a UI implementation that your entity can plug into

# The EntityAPI module, cont'd

- With EntityAPI, you can create a new kind of entity with almost no code
- There is no reason not to use it, unless your entity is doing something really unusual, like residing somewhere other than the database
- EntityAPI also provides a host of other features, like a common API for loading and saving any kind of entity, including core entity types



# Model Entities

- The “Model Entities” module provides an example of how to use EntityAPI to create a custom entity
- Some caution:
  - The code is difficult to understand if you are new to entities, because it uses dynamically-created bundles which are themselves entities
  - It didn’t work for me out of the box, so I gave up on trying to learn from it

# More information

- Entities in core
  - `hook_entity_info()` docs:  
[http://api.drupal.org/api/drupal/modules--system--system.api.php/function/hook\\_entity\\_info/7](http://api.drupal.org/api/drupal/modules--system--system.api.php/function/hook_entity_info/7)
- Fields
  - API docs: <http://api.drupal.org/api/drupal/modules--field--field.module/group/field/7>
- EntityAPI
  - Project page: <http://drupal.org/project/entity>
  - `entity.api.php` in the EntityAPI module
  - Online docs: <http://drupal.org/node/878784>
- Model Entities
  - Project page: <http://drupal.org/project/model>