Entites in Drupal 8

Sascha Grossenbacher
Christophe Galli

DRUPALCAMP VIENNA

Who are we?

Sascha (berdir)

- Active core contributor
- Entity system maintainer
- Porting and maintaining lots of D8 contrib projects

Christophe (cgalli)

- Part-Time Developer
- Site builder
- Author of ‘Content Entity Example Module’

What are we going to talk about

- Content vs Config Entities
- Minimalistic approach
- Installing and using
- Making them useful in Drupal
- Views integration
- Overview and tools

Before we start: The most important command!

```
drush cr
```

Content vs. Config Entities in Drupal 8

Config Entity

- Configuration Management
  - Defaults
  - Deployable
  - Dependencies
  - Config translation
  - Not scalable
- Examples
  - Node Type
  - Vocabulary
  - View
  - Filter format

Content Entity

- Stored in Tables (by Default)
- Configurable fields
- Views integration
- Content translation
- Scalable
- Examples
  - Node
  - Term
  - User
  - Comment

Minimalistic Approach: Use content entity solely for data storage!

Define Entity

```
/**
 * Defines the Contact entity.
 *
 * @ContentEntityType(
 *   id = "content_entity_example_contact",
 *   label = @Translation("Contact entity"),
 *   base_table = "contact",
 *   entity_keys = {
 *     "id" = "id",
 *     "label" = "name",
 *     "uuid" = "uuid"
 *   },
 * )
 */
class Contact extends ContentEntityBase {
```

Define fields (Content entity only)

```
class Contact extends ContentEntityBase {


  public static function baseFieldDefinitions(EntityTypeInterface $entity_type) {

    $fields['id'] = BaseFieldDefinition::create('integer')
      ->setLabel(t('ID'))
      ->setDescription(t('The ID of the Contact entity.'));
    $fields['uuid'] = BaseFieldDefinition::create('uuid')
      ->setLabel(t('UUID'))
      ->setDescription(t('The UUID of the Contact entity.'));
    $fields['name'] = BaseFieldDefinition::create('string')
      ->setLabel(t('Name'))
      ->setDescription(t('The name of the Contact entity.'))
      ->setSettings(array(
        'max_length' => 60,
      ))
    $fields['foo'].........

      return $fields;
  }
}
```

Minimalistic Approach: Install and Use

Installation

- When installing the module containing the content entity definition, the corresponding schema is automatically created.
- Core Bug: Without a schema in place, the module cannot be uninstalled! You must define the entity BEFORE installing it!

Usage (CRUD)

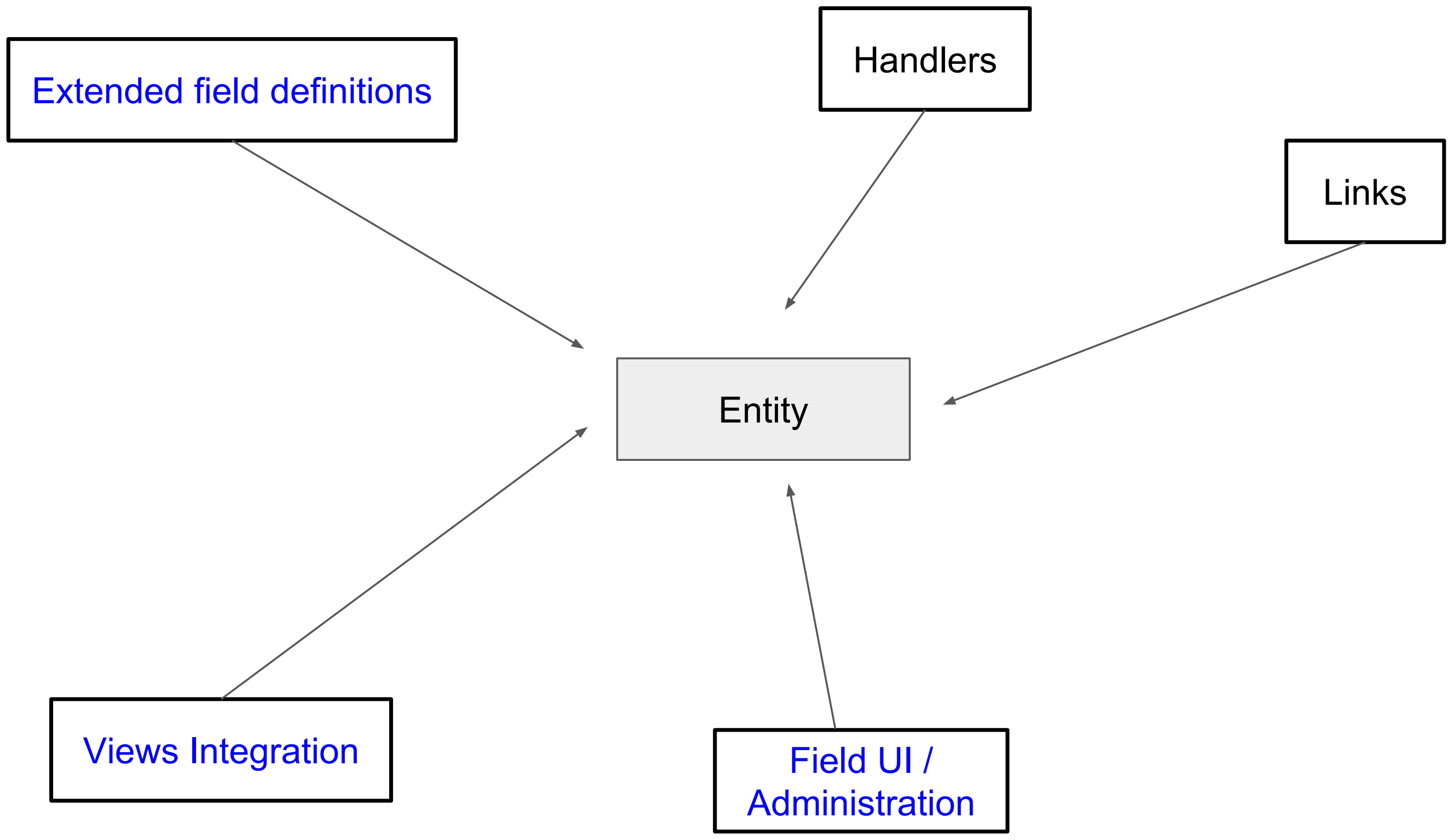
- $my_contact = Contact::create($value_array);
- $my_contact->name = ‘foo’;
- $name = $my_contact->name->value; (Content Entity specific)
- $my_contact->save();

- $ids = \Drupal::entityQuery(‘contact’)
  ->condition(‘name’, ‘Christophe’)
  ->sort(‘name’)
  ->execute();

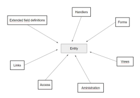
- Contact::load($id); or Contact::loadMultiple($ids);

- $my_contact->delete();

Making Content Entities useful in Drupal



Extended field definitions

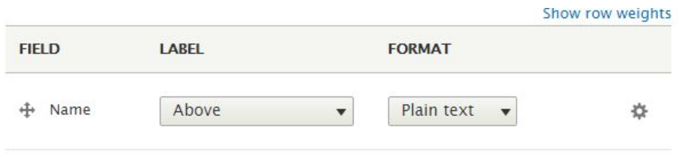


```
$fields['name'] = BaseFieldDefinition::create('string')
  ->setLabel(t('Name'))
  ->setDescription(t('The name of the Contact entity.'))
  ->setRequired(TRUE)
  ->setDefaultValue('Foo')
  ->setSettings(array(
    'max_length' => 60,
  ))
```

Basic definition

```
 ->setDisplayOptions('view', array(
   'label' => 'above',
   'type' => 'string',
   'weight' => -6,
 ))
```

Formatter

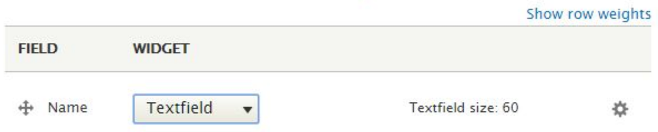


Show row weights

FIELD	LABEL	FORMAT
Name	Above	Plain text

```
->setDisplayOptions('form', array(
    'type' => 'string',
    'weight' => -6,
  ))
```

Form Widget



Show row weights

FIELD	WIDGET	
Name	Textfield	Textfield size: 60

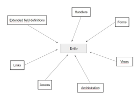
```
  ->setDisplayConfigurable('form', TRUE)
  ->setDisplayConfigurable('view', TRUE);
```

Show Form and Display Widgets in UI

DRUPALCAMP VIENNA

Handlers

Handler Classes, declared in the Entity Class, triggered by the routing definition



```
entity.content_entity_example_contact.canonical:
  path: '/content_entity_example_contact/{content_entity_example_contact}'
  defaults:
    _entity_view: 'content_entity_example_contact'
```

```
entity.content_entity_example_contact.collection:
  path: '/content_entity_example_contact/list'
  defaults:
      _entity_list: 'content_entity_example_contact'
```

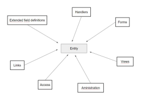
```
content_entity_example.contact_add:
  path: '/content_entity_example_contact/add'
  defaults:
    _entity_form: content_entity_example_contact.add
```

```
*   handlers = {
*     "view_builder" = "Drupal\Core\Entity\EntityViewBuilder",
*     "list_builder" = "Drupal\content_entity_example\Entity\Controller\ContactListBuilder",
*     "form" = {
*       "add" = "Drupal\content_entity_example\Form\ContactForm",
*       "edit" = "Drupal\content_entity_example\Form\ContactForm",
*       "delete" = "Drupal\Core\Entity\ContentEntityDeleteForm",
*     },
*     "access" = "Drupal\content_entity_example\ContactAccessControlHandler",
*   },
```

Controls create/edit/delete access for entity and fields
Shortcut: Specify admin_permission = “administer contacts”

DRUPALCAMP VIENNA

Links



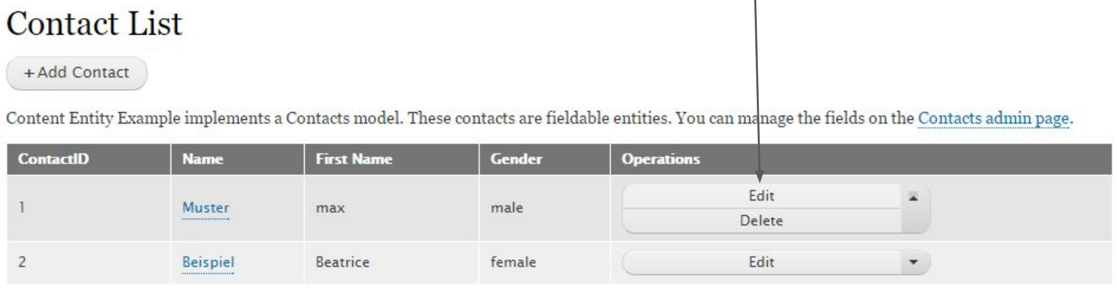
Routing

```
entity.content_entity_example_contact.canonical:
  path: '/content_entity_example_contact/{content_entity_example_contact}'
  defaults:
  # Calls the view builder, defined in the annotation of the contact entity
    _entity_view: 'content_entity_example_contact'
    _title: 'Contact Content'
  requirements:
  # Calls the access control handler of the entity, $operation 'view'
    _entity_access: 'content_entity_example_contact.view'
```

Additional Entity handler:

```
 *     "route_provider" = {
 *       "html" = "Drupal\Core\Entity\Routing\DefaultHtmlRouteProvider",
 *     },
```

```
 *   links = {
 *     "canonical" = "/content_entity_example_contact/{content_entity_example_contact}",
 *     "edit-form" = "/content_entity_example_contact/{content_entity_example_contact}/edit",
 *     "delete-form" = "/contact/{content_entity_example_contact}/delete",
 *     "collection" = "/content_entity_example_contact/list"
```



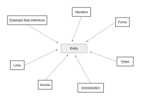
Contact List

+Add Contact

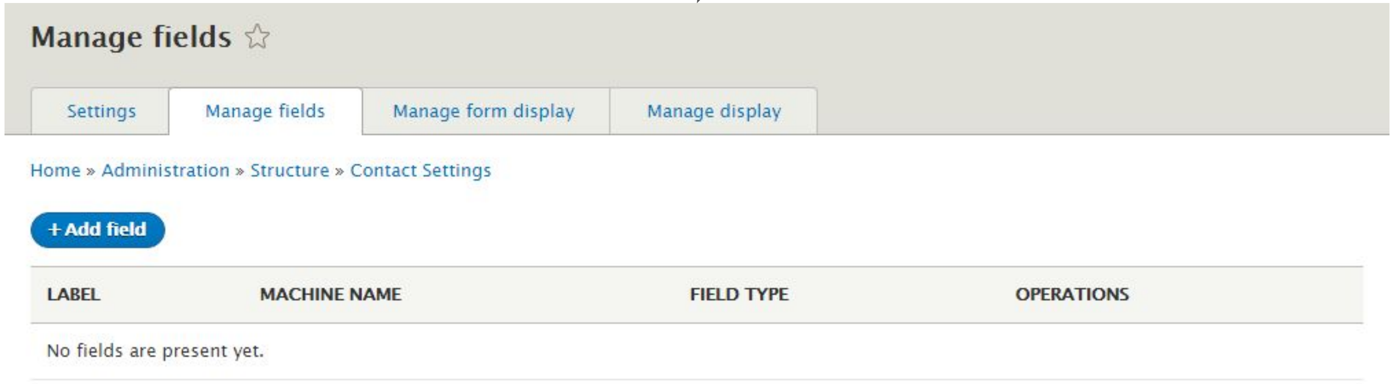
Content Entity Example implements a Contacts model. These contacts are fieldable entities. You can manage the fields on the Contacts admin page.

ContactID	Name	First Name	Gender	Operations
1	Muster	max	male	Edit / Delete
2	Beispiel	Beatrice	female	Edit

Field UI and Administration



* field_ui_base_route = "content_entity_example.contact_settings",



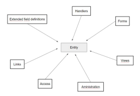
Manage fields ☆

Home » Administration » Structure » Contact Settings

LABEL	MACHINE NAME	FIELD TYPE	OPERATIONS
No fields are present yet.			

DRUPALCAMP VIENNA

Views Integration



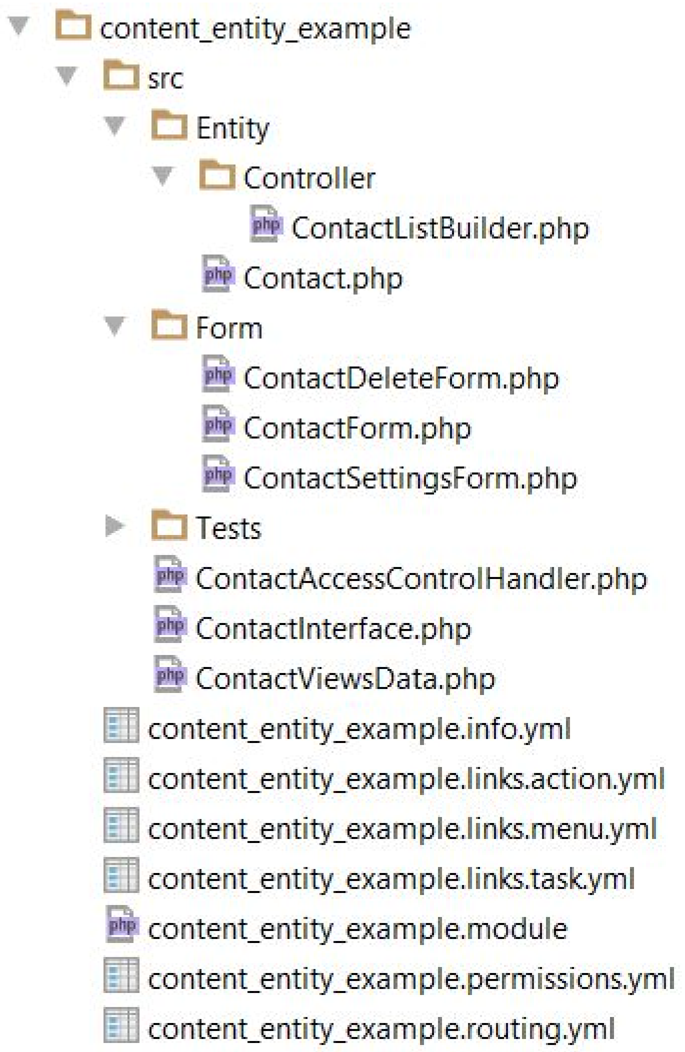
Additional Entity Handler

```
 *     "views_data" = "Drupal\content_entity_example\ContactViewsData",
```

Class definition

```
<?php

namespace Drupal\content_entity_example;

use Drupal\views\EntityViewsData;

class ContactViewsData extends EntityViewsData {

    public function getViewsData()
    {
      $data = parent::getViewsData();
      // Customize views data definitions...

      return $data;
    }

}
```

Summary



```
▼ content_entity_example
  ▼ src
    ▼ Entity
      ▼ Controller
          ContactListBuilder.php
        Contact.php
    ▼ Form
        ContactDeleteForm.php
        ContactForm.php
        ContactSettingsForm.php
    ▶ Tests
      ContactAccessControlHandler.php
      ContactInterface.php
      ContactViewsData.php
    content_entity_example.info.yml
    content_entity_example.links.action.yml
    content_entity_example.links.menu.yml
    content_entity_example.links.task.yml
    content_entity_example.module
    content_entity_example.permissions.yml
    content_entity_example.routing.yml
```

- It is non-trivial
- It ist very powerful
- There is much more to it
  - Bundles
  - Revisions

BUT

there is Drupal Console

1) drupal generate:module
2) drupal generate:entity:content

DRUPALCAMP VIENNA

Tools

- You need an IDE
- Examples Module for Developers
    - https://www.drupal.org/project/examples
- Content Entity Documentation
    - https://www.drupal.org/node/2192175
- Entity Cheat Sheet
    - http://wizzlern.nl/drupal/drupal-8-entity-cheat-sheet
- Drupal Console
    - http://drupalconsole.com/

The End

Questions?