Entity validation: The kick-ass road to data integrity

Kristiaan Van den Eynde (@Magentix) - Deeson

calibrate IMALINK

KRISTIAAN VAN DEN EYNDE

- Drupal developer
- Work at Deeson, UK
- Live near Antwerp, BE
- Happily married
- HSP: Highly Sensitive Person

Deeson.

WHAT IS ENTITY VALIDATION?

```
EntityMetadataWrapperException: Unknown data property next_slide in
EntityStructureWrapper->getPropertyInfo() (line 339 of /var/www/ghent/sites/all/
modules/entity/includes/entity.wrapper.inc).
```

ENTITY VALIDATION

- Validates content entities on multiple levels

Deeson.

ENTITY VALIDATION

- Validates content entities on multiple levels
- Returns a list of violations when validation fails

Deeson.

ENTITY VALIDATION

- Validates content entities on multiple levels
- Returns a list of violations when validation fails
- Happens automatically in entity form validation

ENTITY VALIDATION

- Validates content entities on multiple levels
- Returns a list of violations when validation fails
- Happens automatically in entity form validation
- Is not a part of the form system (unlike D7)

Deeson.

ENTITY VALIDATION

- Validates content entities on multiple levels
- Returns a list of violations when validation fails
- Happens automatically in entity form validation
- Is not a part of the form system (unlike D7)
- Guarantees the data integrity of your entities

Deeson.

Deeson.

WHEN TO USE ENTITY VALIDATION?

ON ENTITY MANIPULATION

- REST
- Custom entity generation code
- Migrations
- ...

Deeson.

HOW TO INVOKE ENTITY VALIDATION?

SIMPLE!

```
$entity->validate();
```

SIMPLE!

```
$entity->get('foo')->validate();
```

Deeson.

SIMPLE!

```
$entity->get('foo')->get(0)->validate();
```

WELL… ACTUALLY

```
$violations = $entity->validate();
```

Deeson.

Deeson.

WORKING WITH VIOLATIONS

CHECKING FOR VIOLATIONS

```
$violations->count();
implements \IteratorAggregate
```

Deeson.

MORE FUNKY METHODS

```
$violations->getEntityViolations();
$violations->getByField('foo');

\Drupal\Core\Entity\
EntityConstraintViolationListInterface
```

READING A SINGLE VIOLATION

```
$violation->getMessage();
```

Deeson.

MORE GROOVY METHODS

```
$violation->getPropertyPath();
$violation->getInvalidValue();
```

```
Symfony\Component\Validator\
ConstraintViolationInterface
```

Deeson.

Deeson.

WHERE DO VIOLATIONS COME FROM?

FOUR LAYERS OF VALIDATION

- The entity as a whole

FOUR LAYERS OF VALIDATION

- The entity as a whole
- A field on an entity

Deeson.

FOUR LAYERS OF VALIDATION

- The entity as a whole
- A field on an entity
- A field entry on an entity

FOUR LAYERS OF VALIDATION

- The entity as a whole
- A field on an entity
- A field entry on an entity
- A property of a field entry

Deeson.

OBLIGATORY CAR ANALOGY

- $car
- $car->field_wheels
- $car->field_wheels[0]
- $car->field_wheels[0][‘tyre_size’]

Deeson.

IN DRUPAL-SPEAK

- ContentEntityInterface
- FieldItemListInterface
- FieldItemInterface
- TypedDataInterface

Deeson.

THE FOUR LEVELS IN DETAIL

THE ENTITY AS A WHOLE

- Validation across multiple fields

THE ENTITY AS A WHOLE

- Validation across multiple fields

  Fuel tank + electric engine

Deeson.

THE ENTITY AS A WHOLE

- Validation across multiple fields

  Fuel tank + electric engine
- Field-independent validation

Deeson.

THE ENTITY AS A WHOLE

- Validation across multiple fields

  Fuel tank + electric engine
- Field-independent validation

  Third car when flat broke

Deeson.

A FIELD ON AN ENTITY

- Validation across multiple field entries

A FIELD ON AN ENTITY

- Validation across multiple field entries

  2 monster truck wheels + 2 minibike wheels

A FIELD ENTRY ON AN ENTITY

- Validation across multiple field properties

A FIELD ENTRY ON AN ENTITY

- Validation across multiple field properties

  16 inch rim + 15 inch tyre

Deeson.

A PROPERTY OF A FIELD ENTRY

- Validation of single properties

A PROPERTY OF A FIELD ENTRY

- Validation of single properties

  A tyre made out of wood

Deeson.

DEFINING YOUR OWN VALIDATION

TWO CLASSES REQUIRED

- A Constraint plugin
- The actual validator

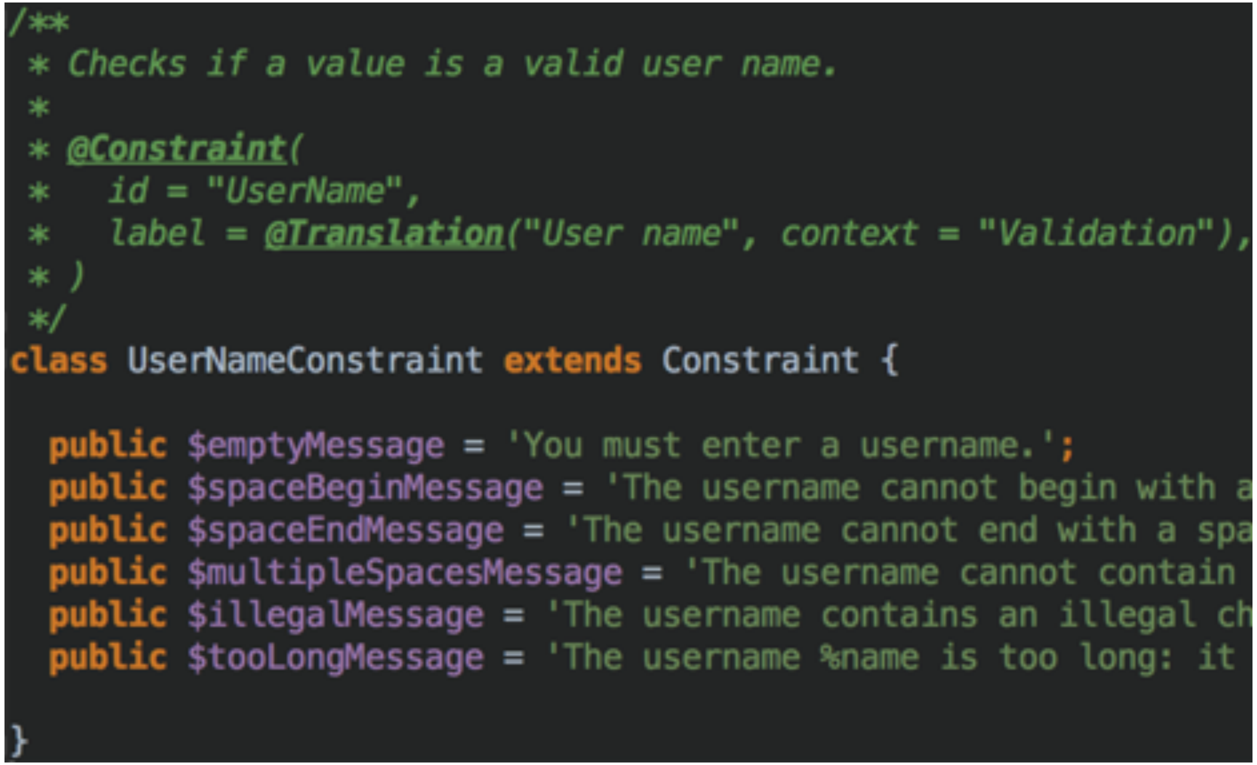
Deeson.

THE CONSTRAINT PLUGIN

CONSTRAINT PLUGIN DEFINITION

- A unique ID
- A human readable label
- A type (string) or list of types (array)
- In mymodule/src/Plugin/Validation/Constraint

Deeson.



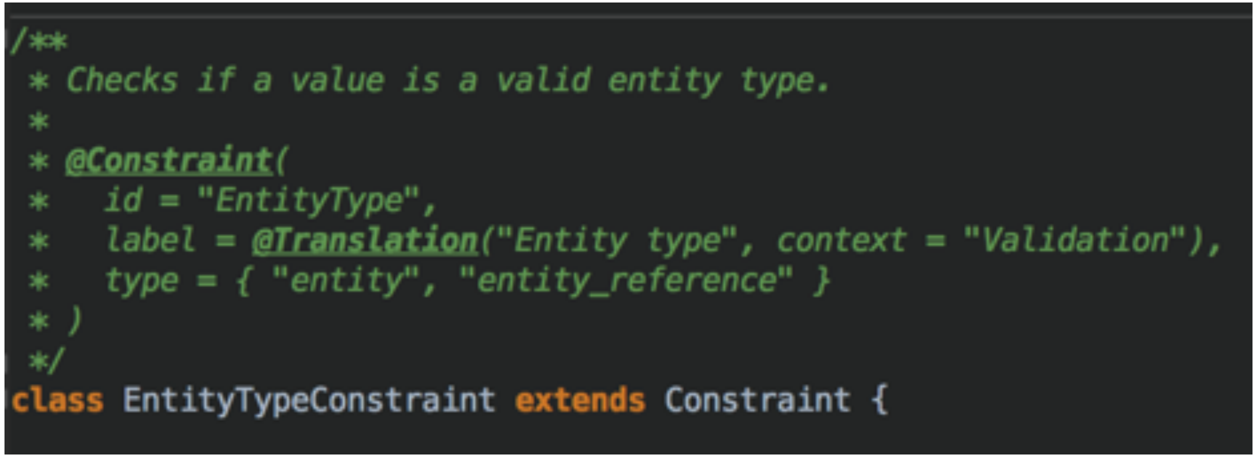
```
/**
 * Checks if a value is a valid user name.
 *
 * @Constraint(
 *   id = "UserName",
 *   label = @Translation("User name", context = "Validation"),
 * )
 */
class UserNameConstraint extends Constraint {

  public $emptyMessage = 'You must enter a username.';
  public $spaceBeginMessage = 'The username cannot begin with a
  public $spaceEndMessage = 'The username cannot end with a spa
  public $multipleSpacesMessage = 'The username cannot contain
  public $illegalMessage = 'The username contains an illegal ch
  public $tooLongMessage = 'The username %name is too long: it

}
```

Deeson.

WHAT ABOUT THE TYPE?

- Nothing: Empty array (the default)
- Everything: false
- Entities: entity, entity:node
- FieldItem: field_item, field_item:entity_reference
- FieldItemList: not supported?
- Other TypedData: string, integer, ...

SETTING THE TYPE KEY



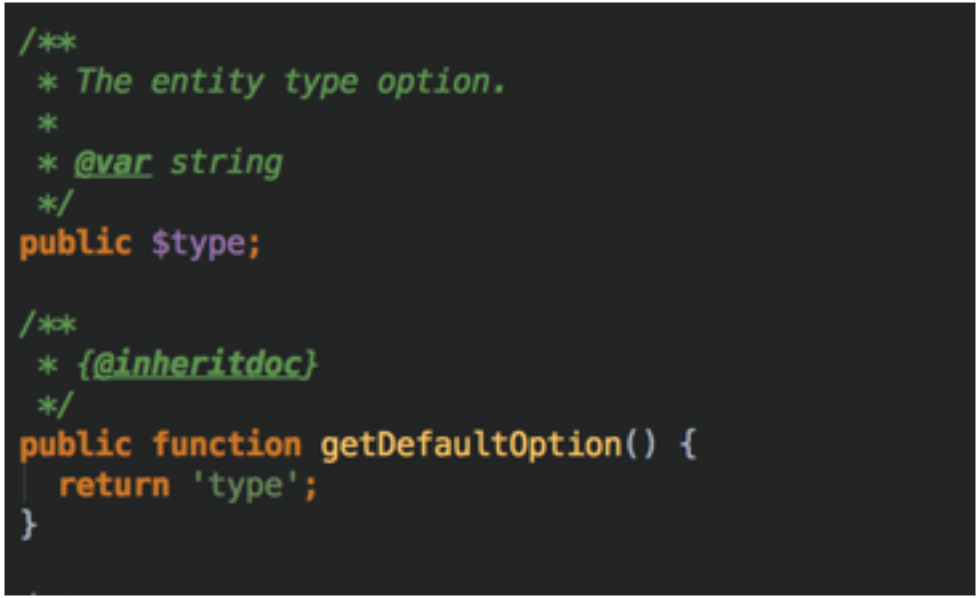
```
/**
 * Checks if a value is a valid entity type.
 *
 * @Constraint(
 *   id = "EntityType",
 *   label = @Translation("Entity type", context = "Validation"),
 *   type = { "entity", "entity_reference" }
 * )
 */
class EntityTypeConstraint extends Constraint {
```

CONSTRAINTS CAN TAKE OPTIONS

- Completely optional
- Simply define them as object properties
- Can accept any number of options as an array
- How to pass the options is covered later

Deeson.

ACCEPTING OPTIONS



```
/**
 * The entity type option.
 *
 * @var string
 */
public $type;

/**
 * {@inheritdoc}
 */
public function getDefaultOption() {
  return 'type';
}
```

Deeson.

Deeson.

THE VALIDATOR CLASS

MAGIC CLASS NAME

- Default: Name of constraint class + Validator
- Can change the default in

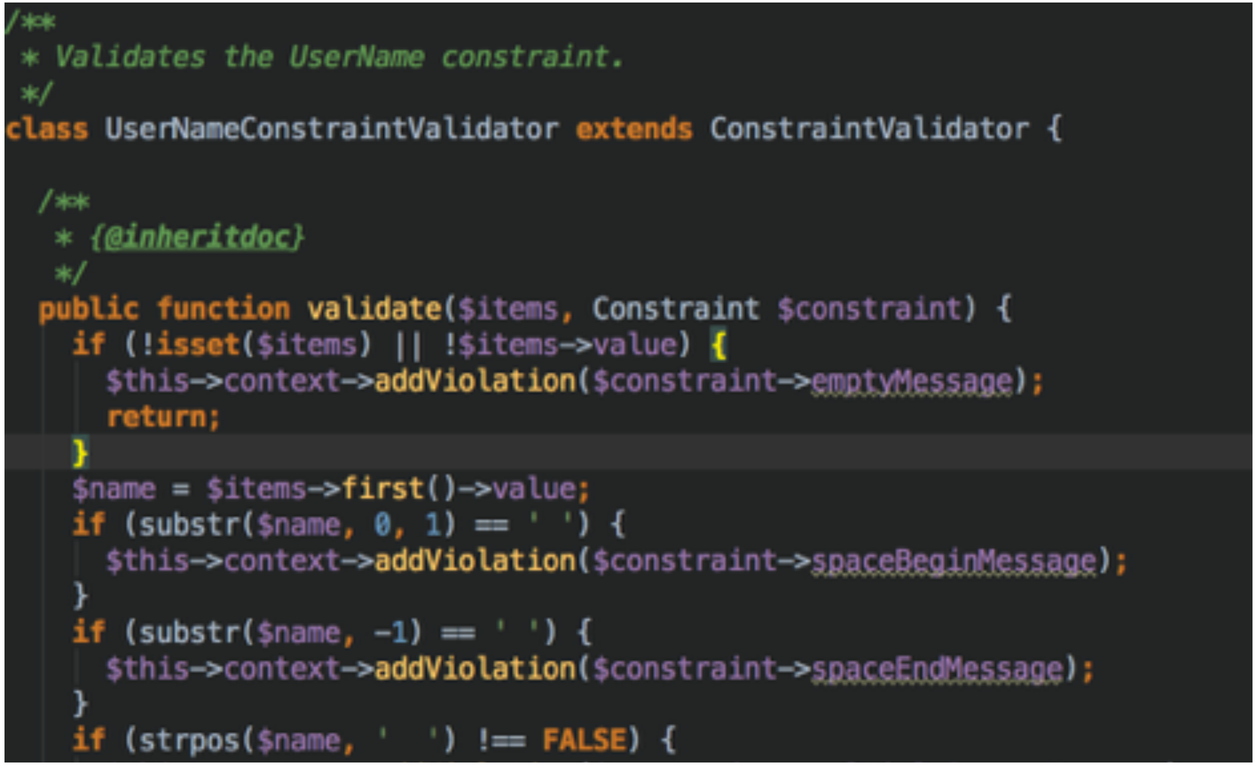
  `Constraint::validatedBy()`

Deeson.

THE VALUE IT RECEIVES

- For entities: The entity
- For fields: The FieldItemList
- For field entries: The FieldItem
- For properties: The raw value

Deeson.



```
/**
 * Validates the UserName constraint.
 */
class UserNameConstraintValidator extends ConstraintValidator {

  /**
   * {@inheritdoc}
   */
  public function validate($items, Constraint $constraint) {
    if (!isset($items) || !$items->value) {
      $this->context->addViolation($constraint->emptyMessage);
      return;
    }
    $name = $items->first()->value;
    if (substr($name, 0, 1) == ' ') {
      $this->context->addViolation($constraint->spaceBeginMessage);
    }
    if (substr($name, -1) == ' ') {
      $this->context->addViolation($constraint->spaceEndMessage);
    }
    if (strpos($name, '  ') !== FALSE) {
```

Deeson.

Deeson.

THE PROPERTY PATH

MENTIONING FOR COMPLETENESS

- Tells you where the error occurred
- Defaults to the item being validated
- Can be more specific in your validator
- Example: ValidReferenceConstraintValidator

Deeson.

EXAMPLES

- When validating an entity

  `field_foo.0.value`

  `field_bar.2`
- When validating a field

  `3.property_bar`
- When validating a field entry

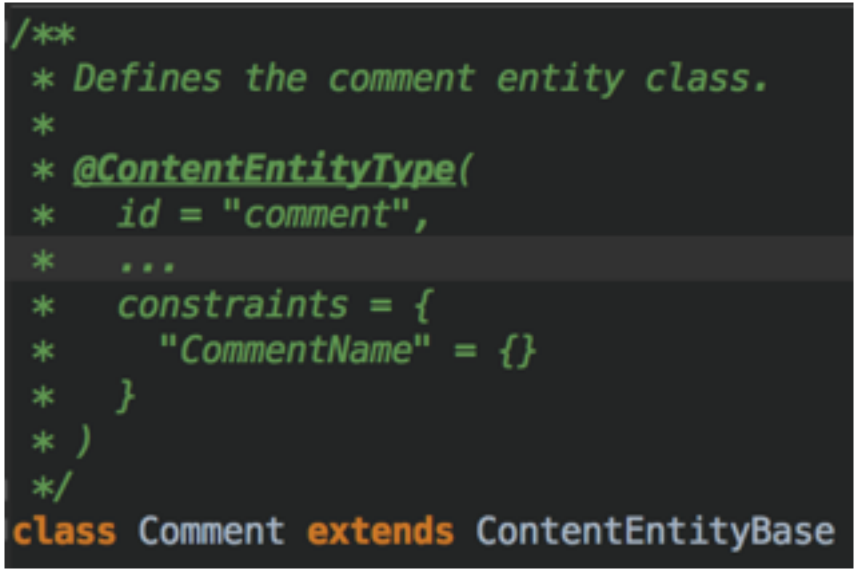
  `value`

Deeson.

Deeson.

HOW TO APPLY VALIDATORS

THE ENTITY LEVEL

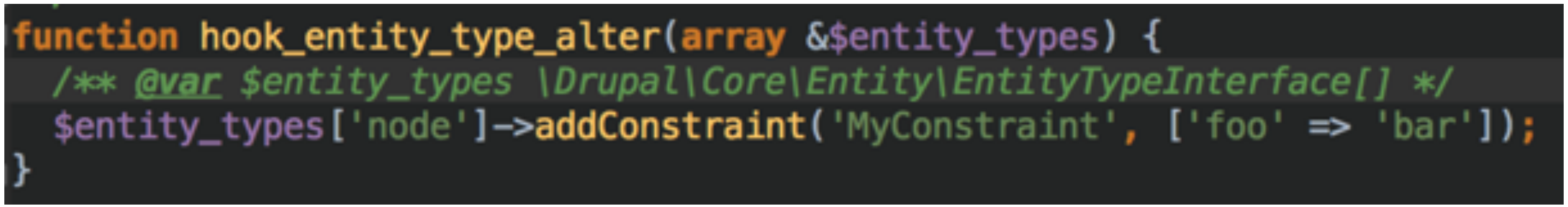


```
/**
 * Defines the comment entity class.
 *
 * @ContentEntityType(
 *   id = "comment",
 *   ...
 *   constraints = {
 *     "CommentName" = {}
 *   }
 * )
 */
class Comment extends ContentEntityBase
```

- As part of the annotation

THE ENTITY LEVEL

- As part of the annotation
- hook_entity_type_alter()

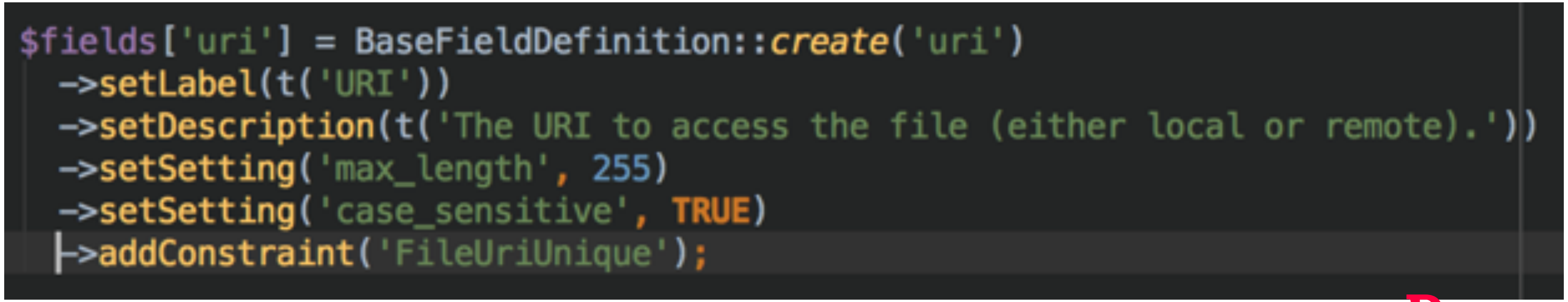


```
function hook_entity_type_alter(array &$entity_types) {
  /** @var $entity_types \Drupal\Core\Entity\EntityTypeInterface[] */
  $entity_types['node']->addConstraint('MyConstraint', ['foo' => 'bar']);
}
```

Deeson.

THE FIELD LEVEL

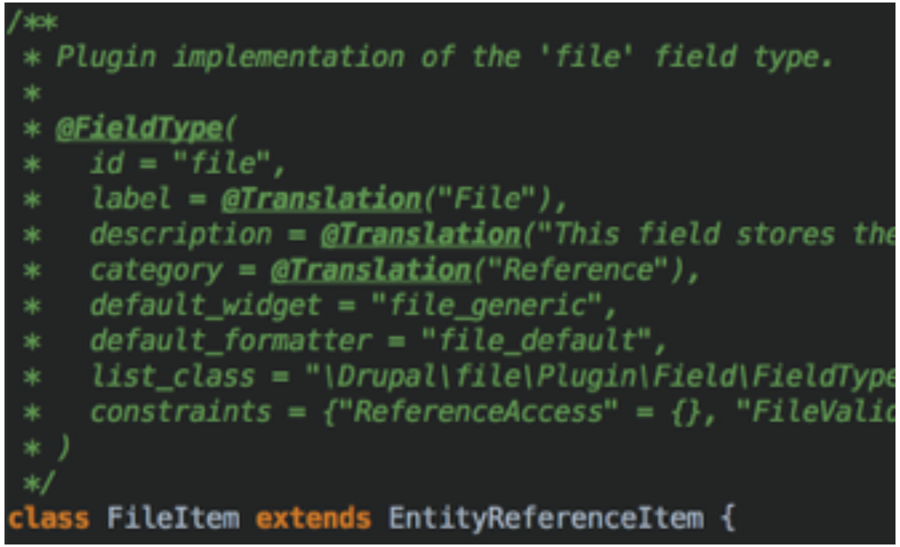
- BaseFieldDefinition::addConstraint()
- hook_entity_base_field_info_alter()
- hook_entity_bundle_field_info_alter()



```
$fields['uri'] = BaseFieldDefinition::create('uri')
  ->setLabel(t('URI'))
  ->setDescription(t('The URI to access the file (either local or remote).'))
  ->setSetting('max_length', 255)
  ->setSetting('case_sensitive', TRUE)
  ->addConstraint('FileUriUnique');
```

Deeson.

THE FIELD ENTRY LEVEL



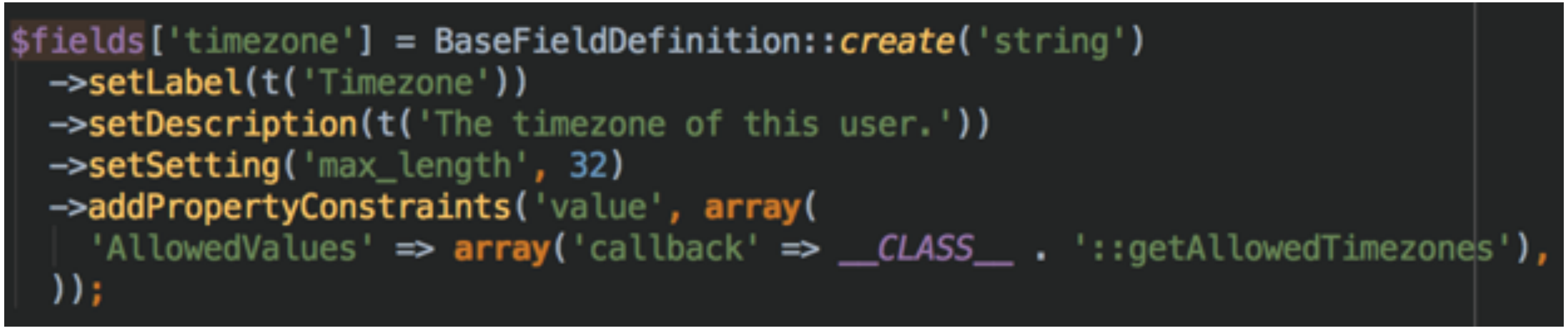
```
/**
 * Plugin implementation of the 'file' field type.
 *
 * @FieldType(
 *   id = "file",
 *   label = @Translation("File"),
 *   description = @Translation("This field stores the
 *   category = @Translation("Reference"),
 *   default_widget = "file_generic",
 *   default_formatter = "file_default",
 *   list_class = "\Drupal\file\Plugin\Field\FieldType
 *   constraints = {"ReferenceAccess" = {}, "FileValic
 * )
 */
class FileItem extends EntityReferenceItem {
```

- As part of the annotation
- `hook_field_info_alter()`

Deeson.

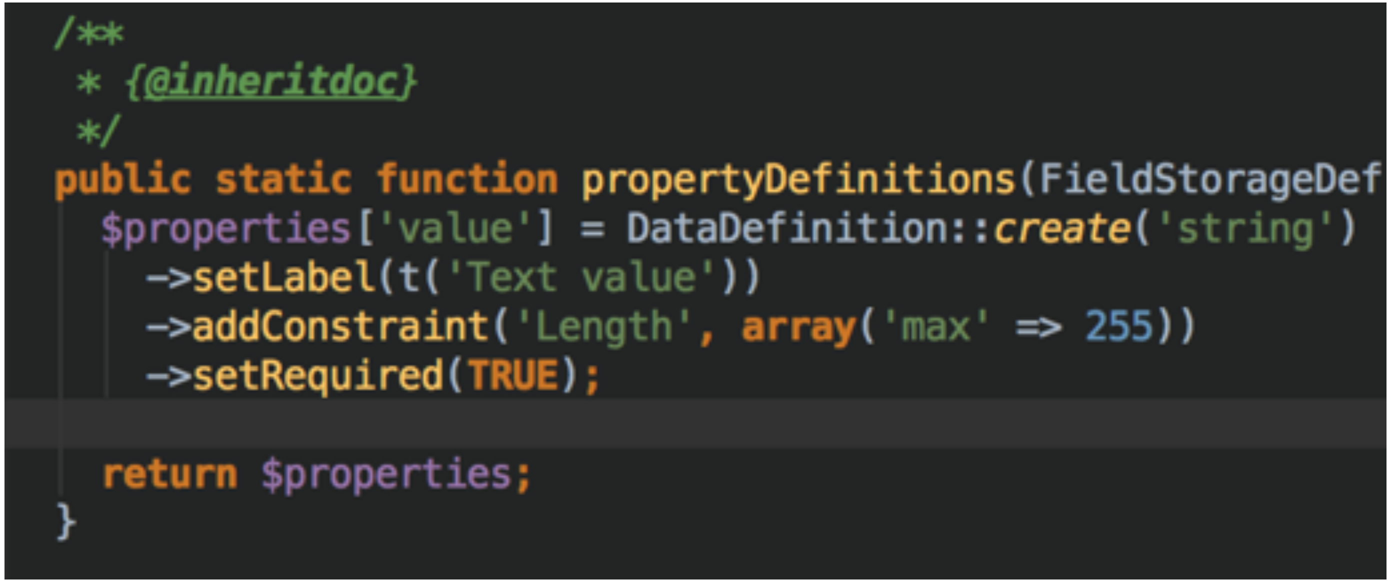
THE FIELD PROPERTY LEVEL

- BaseFieldDefinition::addPropertyConstraints()
- hook_entity_base_field_info_alter()
- hook_entity_bundle_field_info_alter()



```
$fields['timezone'] = BaseFieldDefinition::create('string')
  ->setLabel(t('Timezone'))
  ->setDescription(t('The timezone of this user.'))
  ->setSetting('max_length', 32)
  ->addPropertyConstraints('value', array(
    'AllowedValues' => array('callback' => __CLASS__ . '::getAllowedTimezones'),
  ));
```

THE FIELD PROPERTY LEVEL



```
/**
 * {@inheritdoc}
 */
public static function propertyDefinitions(FieldStorageDef
  $properties['value'] = DataDefinition::create('string')
    ->setLabel(t('Text value'))
    ->addConstraint('Length', array('max' => 255))
    ->setRequired(TRUE);

  return $properties;
}
```

Deeson.

Deeson.

THAT'S IT!

JOIN US!

Five week sabbatical

Unlimited training budget

Distributed work force across Europe

Deeson.

Deeson.

QUESTIONS?

DRUPALCAMP GHENT 2016

GROW SOME IDEAS

calibrate

IMALINK

corecrew
WE BUILD YOUR DIGITAL PRESENCE

nucleus
Hosting Solution Builder

Dropsolid
Makkelijk Digitaal Ondernemen

uniway
Business Critical Internet Solutions

2dotstwice
connecting the dots

AMPLEXOR

OPEN UP MEDIA

wunderkraut