1x INTERNET

Leveraging the Drupal Core Entity API for your custom Entity Types

July 2023

Hristo Chonov @ 1xINTERNET GmbH

https://1xinter.net/Zxv

Hristo Chonov

Studied Computer Science at TU Darmstadt, Germany

Co-Maintainer for Drupal 8/9/10 Core Entity API

Author / maintainer of multiple contrib extensions for Drupal
- Autosave
- Conflict 2.x
- Prefetch Cache

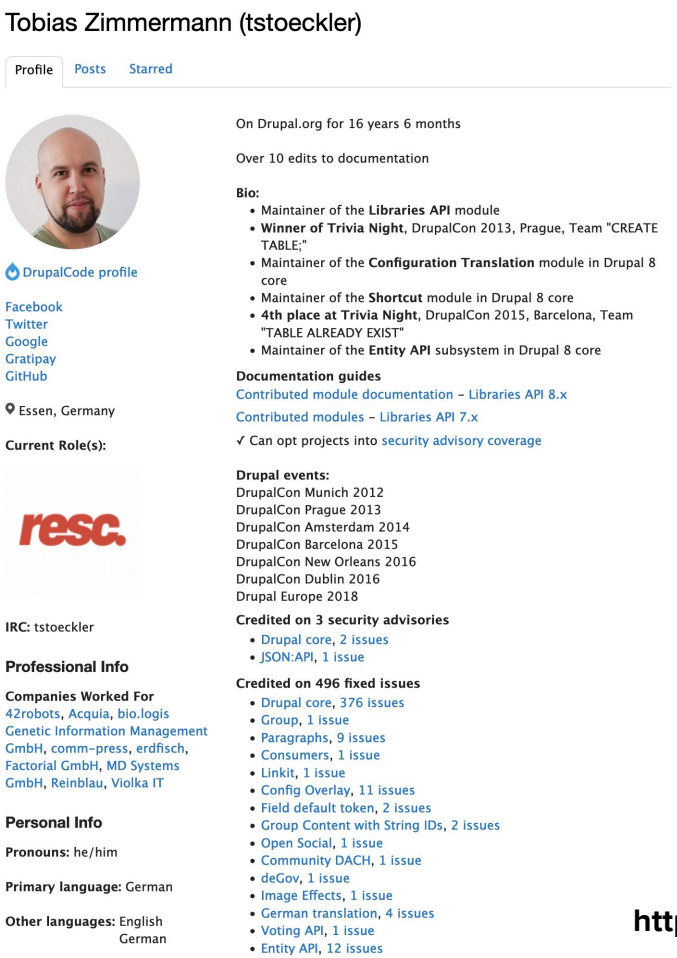
Lead Software Engineer at 1xINTERNET GmbH

https://1xinter.net/Zxv

Credits

Entity Type Walkthrough

https://git.io/d8entity



Tobias Zimmermann (tstoeckler)

Profile | Posts | Starred

On Drupal.org for 16 years 6 months

Over 10 edits to documentation

Bio:

- Maintainer of the **Libraries API** module
- **Winner of Trivia Night**, DrupalCon 2013, Prague, Team "CREATE TABLE;"
- Maintainer of the **Configuration Translation** module in Drupal 8 core
- Maintainer of the **Shortcut** module in Drupal 8 core
- **4th place at Trivia Night**, DrupalCon 2015, Barcelona, Team "TABLE ALREADY EXIST"
- Maintainer of the **Entity API** subsystem in Drupal 8 core

Documentation guides

Contributed module documentation – Libraries API 8.x

Contributed modules – Libraries API 7.x

✓ Can opt projects into security advisory coverage

Drupal events:
DrupalCon Munich 2012
DrupalCon Prague 2013
DrupalCon Amsterdam 2014
DrupalCon Barcelona 2015
DrupalCon New Orleans 2016
DrupalCon Dublin 2016
Drupal Europe 2018

Credited on 3 security advisories

- Drupal core, 2 issues
- JSON:API, 1 issue

Credited on 496 fixed issues

- Drupal core, 376 issues
- Group, 1 issue
- Paragraphs, 9 issues
- Consumers, 1 issue
- Linkit, 1 issue
- Config Overlay, 11 issues
- Field default token, 2 issues
- Group Content with String IDs, 2 issues
- Open Social, 1 issue
- Community DACH, 1 issue
- deGov, 1 issue
- Image Effects, 1 issue
- German translation, 4 issues
- Voting API, 1 issue
- Entity API, 12 issues

DrupalCode profile

Facebook
Twitter
Google
Gratipay
GitHub

Essen, Germany

Current Role(s):

resc.

IRC: tstoeckler

Professional Info

Companies Worked For
42robots, Acquia, bio.logis Genetic Information Management GmbH, comm-press, erdfisch, Factorial GmbH, MD Systems GmbH, Reinblau, Violka IT

Personal Info

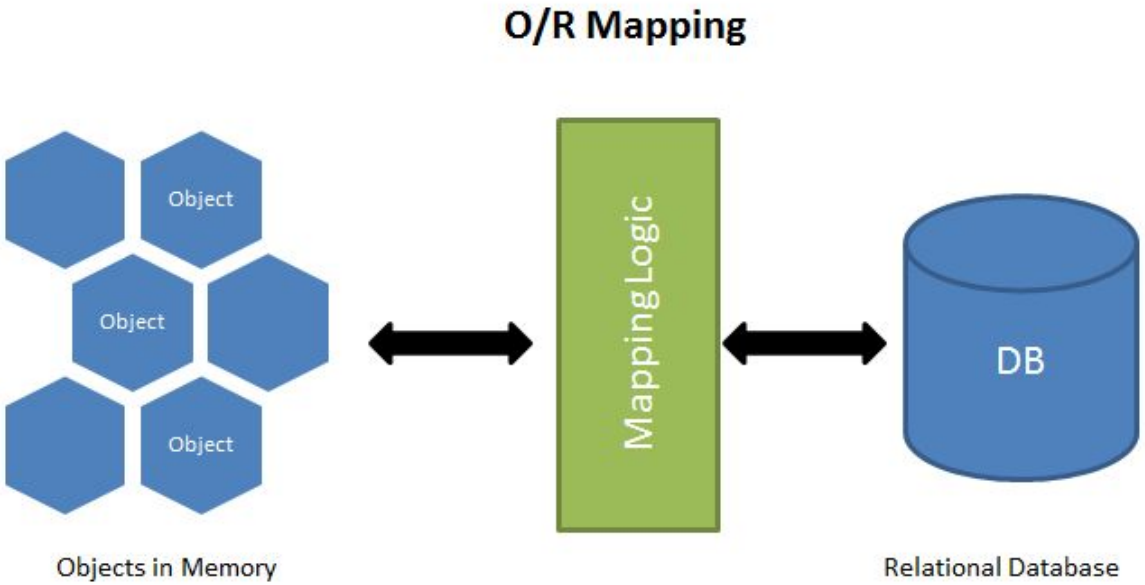
Pronouns: he/him

Primary language: German

Other languages: English
German

https://1xinter.net/Zxv

What is ORM?



https://1xinter.net/Zxv

Drupal - Entity API

- ORM implementation

- supporting different variants of the same entity type

- supporting fields/properties defined in code

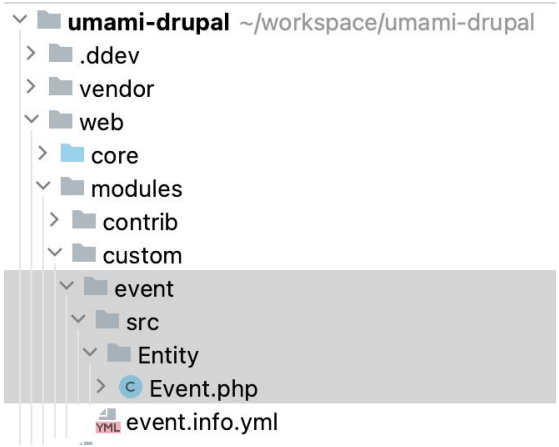
- supporting custom defined fields through configuration

- supporting translatability for each field

- supporting versions / revisions

Basics

1x



Event.php

```php
<?php

namespace Drupal\event\Entity;

use Drupal\Core\Entity\ContentEntityBase;

/**
 * @ContentEntityType(
 *   id = "event",
 *   label = @Translation("Event"),
 *   base_table = "event",
 *   entity_keys = {
 *     "id" = "id",
 *     "uuid" = "uuid",
 *   },
 * )
 */
class Event extends ContentEntityBase { }
```

event	/* The base table for event entities. */
uuid	varchar(128)
id	int(10) unsigned

https://1xinter.net/Zxv

Basics...

1x

```php
<?php

use Drupal\event\Entity\Event;

$event = Event::create();
$event->save();
```

event

1 row | Tx: Auto | DDL

WHERE

	id	uuid
1	1	6fd5e14f-6ba8-464c-b10a-1274c8557958

https://1xinter.net/Zxv

Entity Keys

1x

```php
<?php

namespace Drupal\event\Entity;

use Drupal\Core\Entity\ContentEntityBase;

/**
 * @ContentEntityType(
 *   id = "event",
 *   label = @Translation("Event"),
 *   base_table = "event",
 *   entity_keys = {
 *     "id" = "id",
 *     "uuid" = "uuid",
 *     "owner" = "author",
 *     "published" = "published",
 *   },
 * )
 */
```

event /* The base table for event entities. */	
uuid	varchar(128)
author /* The ID of the target entity. */	int(10) unsigned
published	tinyint(4)
id	int(10) unsigned

```php
25  class Event extends ContentEntityBase implements EntityOwnerInterface, EntityPublishedInterface {
26
27    use EntityOwnerTrait, EntityPublishedTrait;
28
29    /**
30     * {@inheritdoc}
31     */
32    public static function baseFieldDefinitions(EntityTypeInterface $entity_type) {
33      // Get the field definitions for 'id' and 'uuid' from the parent.
34      $fields = parent::baseFieldDefinitions($entity_type);
35
36      // Get the field definitions for 'author' and 'published' from the traits.
37      $fields += static::ownerBaseFieldDefinitions($entity_type);
38      $fields += static::publishedBaseFieldDefinitions($entity_type);
39
40      return $fields;
41    }
42
43  }
```

https://1xinter.net/Zxv

Base fields

1x

```
/**
 * @ContentEntityType(
 *   id = "event",
 *   label = @Translation("Event"),
 *   base_table = "event",
 *   entity_keys = {
 *     "id" = "id",
 *     "uuid" = "uuid",
 *     "owner" = "author",
 *     "published" = "published",
 *     "label" => "title",
 *   },
 * )
 */
```

```
public static function baseFieldDefinitions(EntityTyp
  // Get the field definitions for 'id' and 'uuid' fr
  $fields = parent::baseFieldDefinitions($entity_type

  // Get the field definitions for 'author' and 'publ
  $fields += static::ownerBaseFieldDefinitions($entity_type);
  $fields += static::publishedBaseFieldDefinitions($entity_type);

  $fields['title'] = BaseFieldDefinition::create( type: 'string')
    ->setLabel(t( string: 'Title'))
    ->setRequired( required: TRUE);

  $fields['date'] = BaseFieldDefinition::create( type: 'datetime')
    ->setLabel(t( string: 'Date'))
    ->setRequired( required: TRUE);

  $fields['description'] = BaseFieldDefinition::create( type: 'text_long')
    ->setLabel(t( string: 'Description'));

  return $fields;
}
```

event /* The base table for event entities. */

uuid	varchar(128)
author /* The ID of the target entity. */	int(10) unsigned
published	tinyint(4)
title	varchar(255)
date /* The date value. */	varchar(20)
description__value	longtext
description__format	varchar(255)
id	int(10) unsigned

https://1xinter.net/Zxv

Creating and saving an Entity

```php
$event = Event::create([
  'title' => 'Leveraging the Drupal Core Entity API for your custom entity types',
  'date' => '2023-07-21T10:45:00',
  'description' => [
    'value' => '<p>This session is a walk-through defining a custom entity type. Some of the major topics will be about:</p>
                <p>
                  <ul>
                    <li>Bundles</li>
                    <li>Revision and translation support</li>
                    <li>Configuring the form display of the entity type</li>
                    <li>Configuring the view display of</li>
                    <li>Constraints and validation</li>
                    <li>Access control</li>
                    <li>Entity queries</li>
                    <li>Configuration vs content entity</li>
                  </ul>
                </p>',
    'format' => 'restricted_html',
  ],
]);
$event->save();
```

event

1 row | Tx: Auto | DDL

WHERE | ORDER BY

	1
id	1
uuid	b61526f7-54f0-4ece-bb0b-8d6e5f04a7a7
author	0
published	1
title	Leveraging the Drupal Core Entity API for your c…
date	2023-07-21T10:45:00
description__value	<p>This session is a walk-through defining a cus…
description__format	restricted_html

Base fields - Getters and Setters

```php
/**
 * Returns the title of the event.
 *
 * @return string
 *   The event title.
 */
public function getTitle() {
  return $this->get('title')->value;
}
```

```php
/**
 * Sets the title of the event.
 *
 * @param string $title
 *   The title.
 *
 * @return $this
 */
public function setTitle($title) {
  return $this->set( name: 'title', $title);
}
```

```
> $event->get('title');
= Drupal\Core\Field\FieldItemList {#7951
    0: Drupal\Core\Field\Plugin\Field\FieldType\StringItem {#7955},
  }

> $event->get('title')->value;
= "Leveraging the Drupal Core Entity API for your custom entity types"
```

Entity view

- requires a route on which the entity's field values are output on a given path

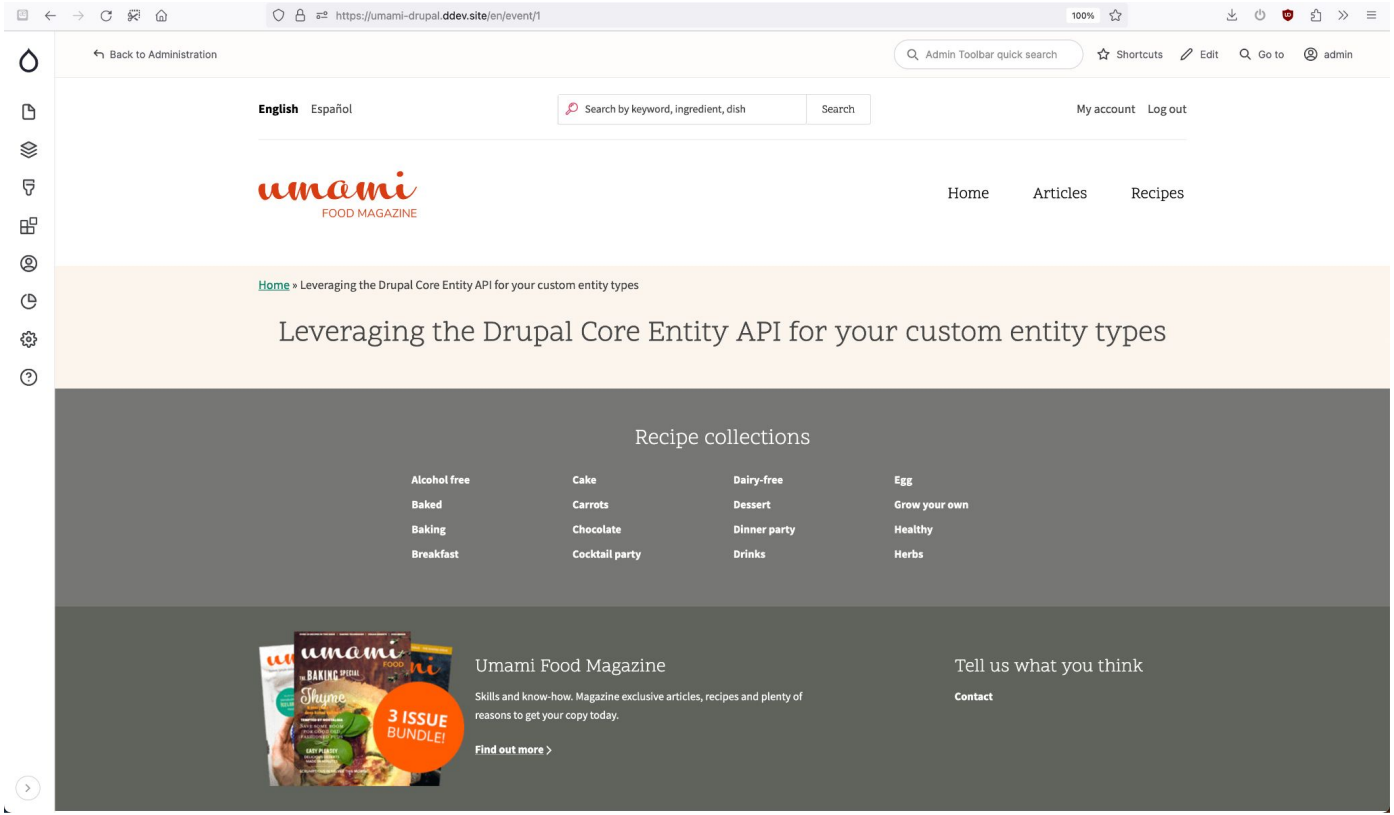
- can be automated by amending the entity annotation

- contributed Entity API provides various enhancements to the core Entity API. One such enhancement is the ability to more easily provide permissions for entity types.

https://www.drupal.org/project/entity

```
* @ContentEntityType(
*   id = "event",
*   label = @Translation("Event"),
*   base_table = "event",
*   entity_keys = {
*     "id" = "id",
*     "uuid" = "uuid",
*     "owner" = "author",
*     "published" = "published",
*     "label" = "title",
*   },
*   handlers = {
*     "access" = "Drupal\entity\EntityAccessControlHandler",
*     "permission_provider" = "Drupal\entity\EntityPermissionProvider",
*     "route_provider" = {
*       "default" = "Drupal\entity\Routing\DefaultHtmlRouteProvider",
*     },
*   },
*   links = {
*     "canonical" = "/event/{event}",
*   },
*   admin_permission = "administer event",
* )
*/
```

Entity view...


https://umami-drupal.ddev.site/en/event/1
100%
Back to Administration
Admin Toolbar quick search
Shortcuts
Edit
Go to
admin
English Español
Search by keyword, ingredient, dish
Search
My account
Log out
umami
FOOD MAGAZINE
Home
Articles
Recipes
Home » Leveraging the Drupal Core Entity API for your custom entity types
Leveraging the Drupal Core Entity API for your custom entity types
Recipe collections
Alcohol free
Baked
Baking
Breakfast
Cake
Carrots
Chocolate
Cocktail party
Dairy-free
Dessert
Dinner party
Drinks
Egg
Grow your own
Healthy
Herbs
3 ISSUE BUNDLE!
Umami Food Magazine
Skills and know-how. Magazine exclusive articles, recipes and plenty of reasons to get your copy today.
Find out more
Tell us what you think
Contact

Entity view...

```php
$fields['date'] = BaseFieldDefinition::create( type: 'datetime')
  ->setLabel(t( string: 'Date'))
  ->setRequired( required: TRUE)
  ->setDisplayOptions( display_context: 'view', [
    'label' => 'inline',
    'settings' => [
      'format_type' => 'short',
    ],
    'weight' => 0,
  ]);
```

```php
$fields['description'] = BaseFieldDefinition
  ->setLabel(t( string: 'Description'))
  ->setDisplayOptions( display_context: 'view',
    'label' => 'hidden',
    'weight' => 10,
  ]);
```

umami
FOOD MAGAZINE

Home Articles Recipes

Home » Leveraging the Drupal Core Entity API for your custom entity types

Leveraging the Drupal Core Entity API for your custom entity types

Date: 07/21/2023 - 13:45

This session is a walk-through defining a custom entity type. Some of the major topics will be about:

- Bundles
- Revision and translation support
- Configuring the form display of the entity type
- Configuring the view display of the entity type
- Constraints and validation
- Access control
- Entity queries
- Configuration vs content entity types

Entity form

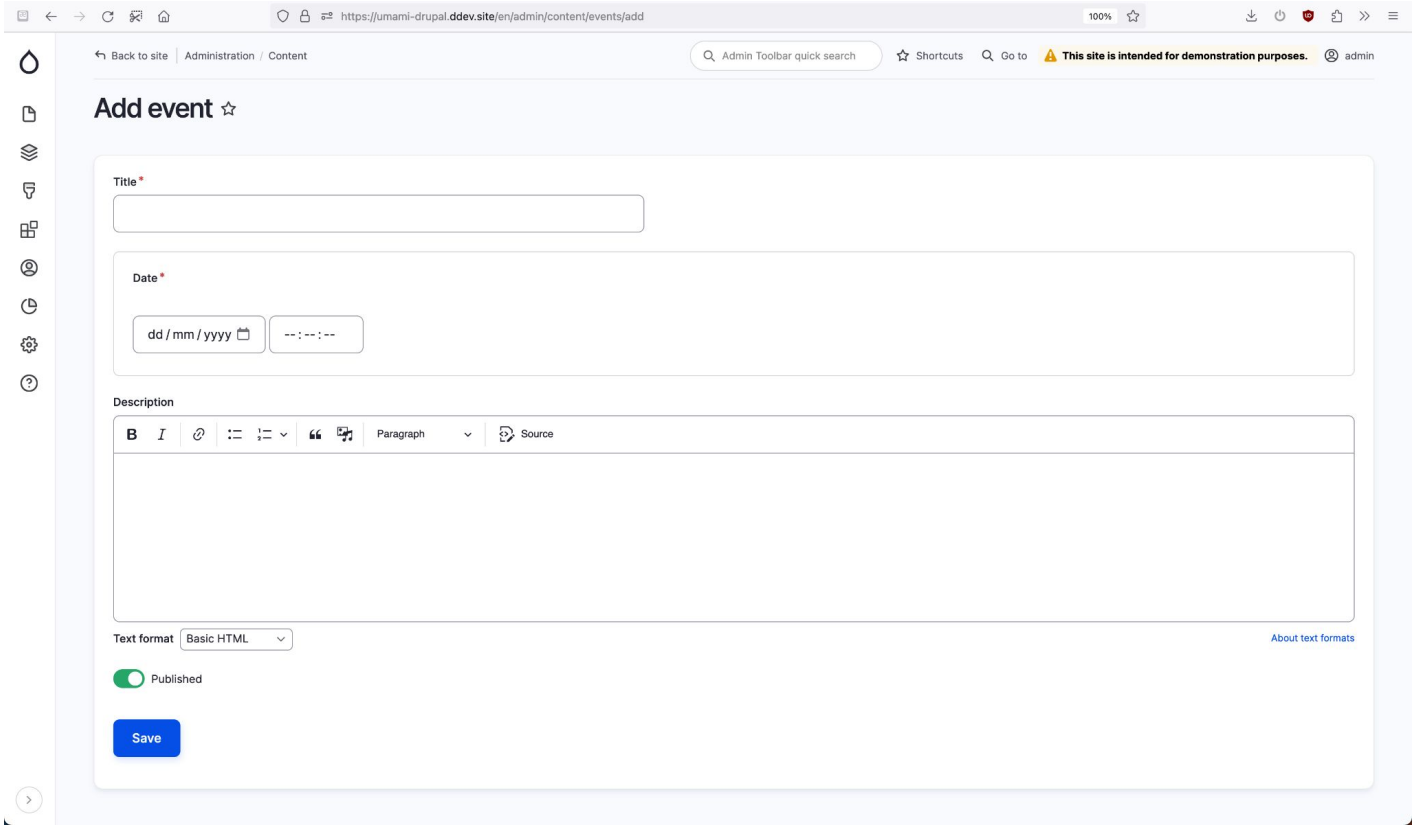
```
*   handlers = {
*     "access" = "Drupal\entity\EntityAccessControlHandler",
*     "permission_provider" = "Drupal\entity\EntityPermissionProvider",
*     "route_provider" = {
*       "default" = "Drupal\entity\Routing\DefaultHtmlRouteProvider",
*     },
*     "form" = {
*       "add" = "Drupal\Core\Entity\ContentEntityForm",
*       "edit" = "Drupal\Core\Entity\ContentEntityForm",
*       "delete" = "Drupal\Core\Entity\ContentEntityDeleteForm",
*     },
```

```
*   links = {
*     "canonical" = "/event/{event}",
*     "add-form" = "/admin/content/events/add",
*     "edit-form" = "/admin/content/events/manage/{event}",
*     "delete-form" = "/admin/content/events/manage/{event}/delete",
*   },
```

```
$fields['title'] = BaseFieldDefinition::create( type: 'string')
  ->setLabel(t( string: 'Title'))
  ->setRequired( required: TRUE)
  ->setDisplayOptions( display_context: 'form', ['weight' => 0]);

$fields['date'] = BaseFieldDefinition::create( type: 'datetime')
  ->setLabel(t( string: 'Date'))
  ->setRequired( required: TRUE)
  ->setDisplayOptions( display_context: 'view', [
    'label' => 'inline',
    'settings' => [
      'format_type' => 'short',
    ],
    'weight' => 0,
  ])
  ->setDisplayOptions( display_context: 'form', ['weight' => 10]);

$fields['description'] = BaseFieldDefinition::create( type: 'text_long')
  ->setLabel(t( string: 'Description'))
  ->setDisplayOptions( display_context: 'view', [
    'label' => 'hidden',
    'weight' => 10,
  ])
  ->setDisplayOptions( display_context: 'form', ['weight' => 20]);

$fields['published']->setDisplayOptions( display_context: 'form', [
  'settings' => [
    'display_label' => TRUE,
  ],
  'weight' => 30,
]);
```

1x

Entity form...


https://umami-drupal.ddev.site/en/admin/content/events/add
100%
Back to site
Administration
Content
Admin Toolbar quick search
Shortcuts
Go to
This site is intended for demonstration purposes.
admin
Add event
Title *
Date *
dd / mm / yyyy
--:--:--
Description
Paragraph
Source
Text format
Basic HTML
About text formats
Published
Save

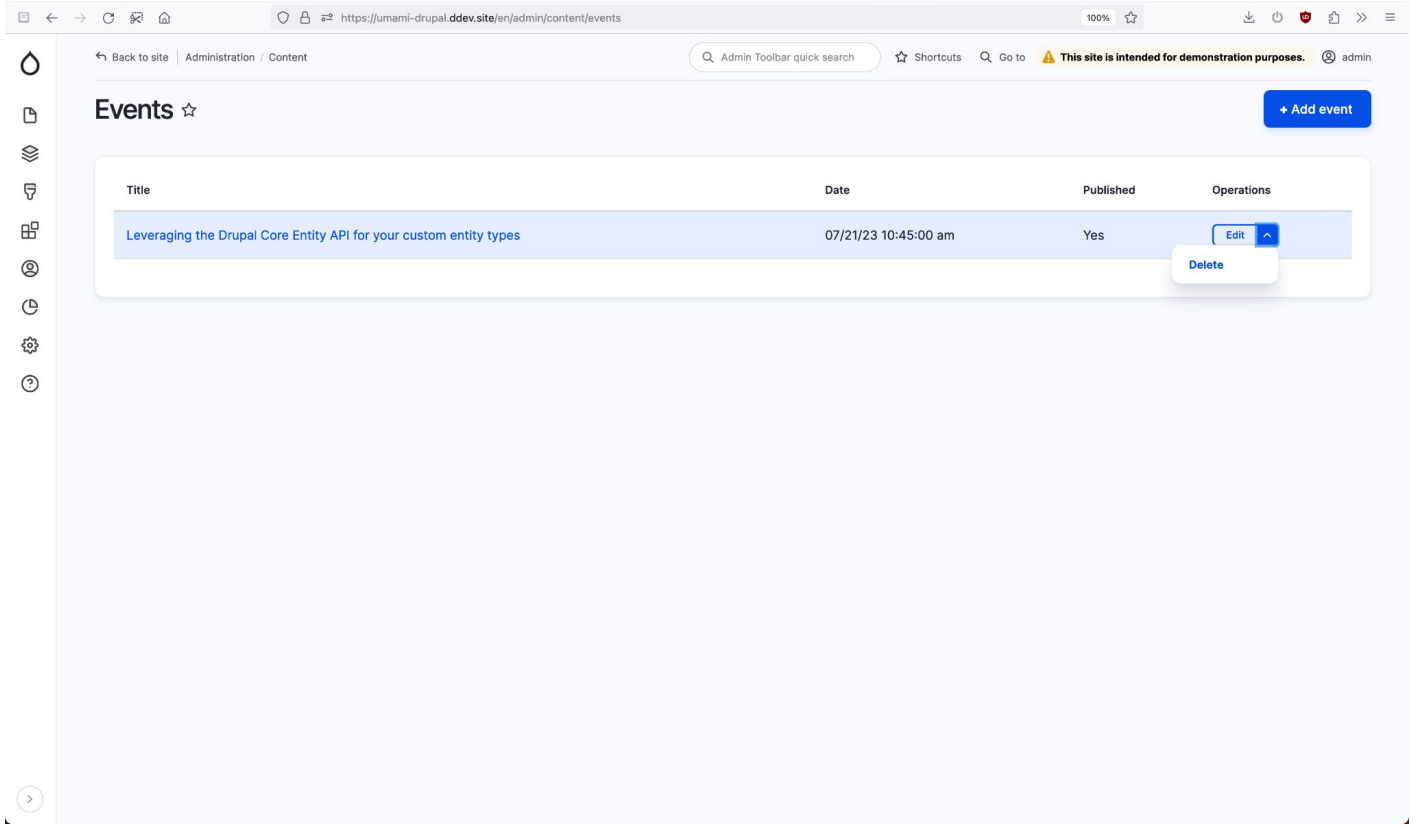
Entity listing

1x

```php
<?php

namespace Drupal\event\Controller;

use Drupal\Core\Entity\EntityInterface;
use Drupal\Core\Entity\EntityListBuilder;

/**
 * Defines an implementation to build a listing of event entities.
 */
class EventListBuilder extends EntityListBuilder {

  /** {@inheritdoc} ...*/
  public function buildHeader() {
    $header = [];
    $header['title'] = $this->t( string: 'Title');
    $header['date'] = $this->t( string: 'Date');
    $header['published'] = $this->t( string: 'Published');
    return $header + parent::buildHeader();
  }

  /** {@inheritdoc} ...*/
  public function buildRow(EntityInterface $event) {
    /** @var \Drupal\event\Entity\Event $event */
    $row = [];
    $row['title'] = $event->toLink();
    $row['date'] = $event->date->date->format('m/d/y h:i:s a');
    $row['published'] = $event->isPublished() ? $this->t( string: 'Yes') : $this->t( string: 'No');
    return $row + parent::buildRow($event);
  }

}
```

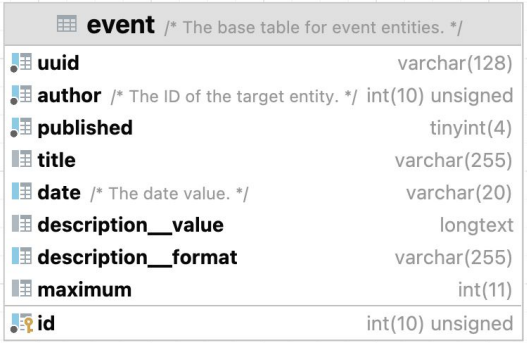
```php
/**
 * @ContentEntityType(
 *   id = "event",
 *   label = @Translation("Event"),
 *   label_collection = @Translation("Events"),
 *   label_singular = @Translation("event"),
 *   label_plural = @Translation("events"),
 *   base_table = "event",
 *   links = {
 *     "canonical" = "/event/{event}",
 *     "add-form" = "/admin/content/events/add",
 *     "edit-form" = "/admin/content/events/manage
 *     "delete-form" = "/admin/content/events/mana
 *     "collection" = "/admin/content/events",
 *   },
 *   handlers = {
 *     "access" = "Drupal\entity\EntityAccessControlHandler",
 *     "permission_provider" = "Drupal\entity\EntityPermissionProvider",
 *     "route_provider" = {
 *       "default" = "Drupal\entity\Routing\DefaultHtmlRouteProvider",
 *     },
 *     "form" = {
 *       "add" = "Drupal\Core\Entity\ContentEntityForm",
 *       "edit" = "Drupal\Core\Entity\ContentEntityForm",
 *       "delete" = "Drupal\Core\Entity\ContentEntityDeleteForm",
 *     },
 *     "list_builder" = "Drupal\event\Controller\EventListBuilder",
 *     "local_action_provider" = {
 *       "collection" = "Drupal\entity\Menu\EntityCollectionLocalActionProvider",
 *     },
 *   },
```

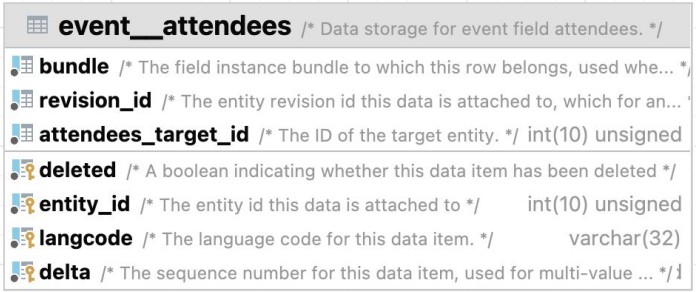
Entity listing



Entity constraints and validation

```php
$fields['maximum'] = BaseFieldDefinition::create( type: 'integer')
  ->setLabel(t( string: 'Maximum number of attendees'))
  ->setSetting( setting_name: 'min',  value: 1)
  ->setDisplayOptions( display_context: 'form', ['weight' => 23]);

$fields['attendees'] = BaseFieldDefinition::create( type: 'entity_reference')
  ->setLabel(t( string: 'Attendees'))
  ->setSetting( setting_name: 'target_type',  value: 'user')
  ->setCardinality( cardinality: FieldStorageDefinitionInterface::CARDINALITY_UNLIMITED)
  ->setDisplayOptions( display_context: 'view', ['weight' => 20])
  ->setDisplayOptions( display_context: 'form', ['weight' => 27]);
```





Entity constraints and validation

AttendeeCountConstraint.php

```php
<?php

namespace Drupal\event\Plugin\Validation\Constraint;

use Symfony\Component\Validator\Constraint;

/**
 * @Constraint(
 *   id = "AttendeeCount",
 *   label = @Translation("Attendee count"),
 * )
 */
class AttendeeCountConstraint extends Constraint {

  /**
   * @var string
   */
  public $message = 'The event %title only allows %maximum attendees.';

}
```

```php
$fields['attendees'] = BaseFieldDefinition::create( type: 'entity_reference')
  ->setLabel(t( string: 'Attendees'))
  ->setSetting( setting_name: 'target_type', value: 'user')
  ->setCardinality( cardinality: FieldStorageDefinitionInterface::CARDINALITY_UNL
  ->addConstraint( constraint_name: 'AttendeeCount')
  ->setDisplayOptions( display_context: 'view', ['weight' => 20])
  ->setDisplayOptions( display_context: 'form', ['weight' => 27]);
```

AttendeeCountConstraintValidator.php

```php
<?php

namespace Drupal\event\Plugin\Validation\Constraint;

use Symfony\Component\Validator\Constraint;
use Symfony\Component\Validator\ConstraintValidator;

class AttendeeCountConstraintValidator extends ConstraintValidator {

  public function validate($value, Constraint $constraint) {
    /* @var \Drupal\Core\Field\FieldItemListInterface $value */
    /* @var \Drupal\event\Plugin\Validation\Constraint\AttendeeCountConstraint $constraint */
    /* @var \Drupal\Core\Entity\Plugin\DataType\EntityAdapter $adapter */
    $adapter = $value->getParent();
    /* @var \Drupal\event\Entity\Event $event */
    $event = $adapter->getEntity();
    $maximum = $event->maximum->value;
    if (count($value) > $maximum) {
      $this->context->buildViolation($constraint->message)
        ->setParameter('%title', $event->getTitle())
        ->setParameter('%maximum', $maximum)
        ->addViolation();
    }
  }

}
```

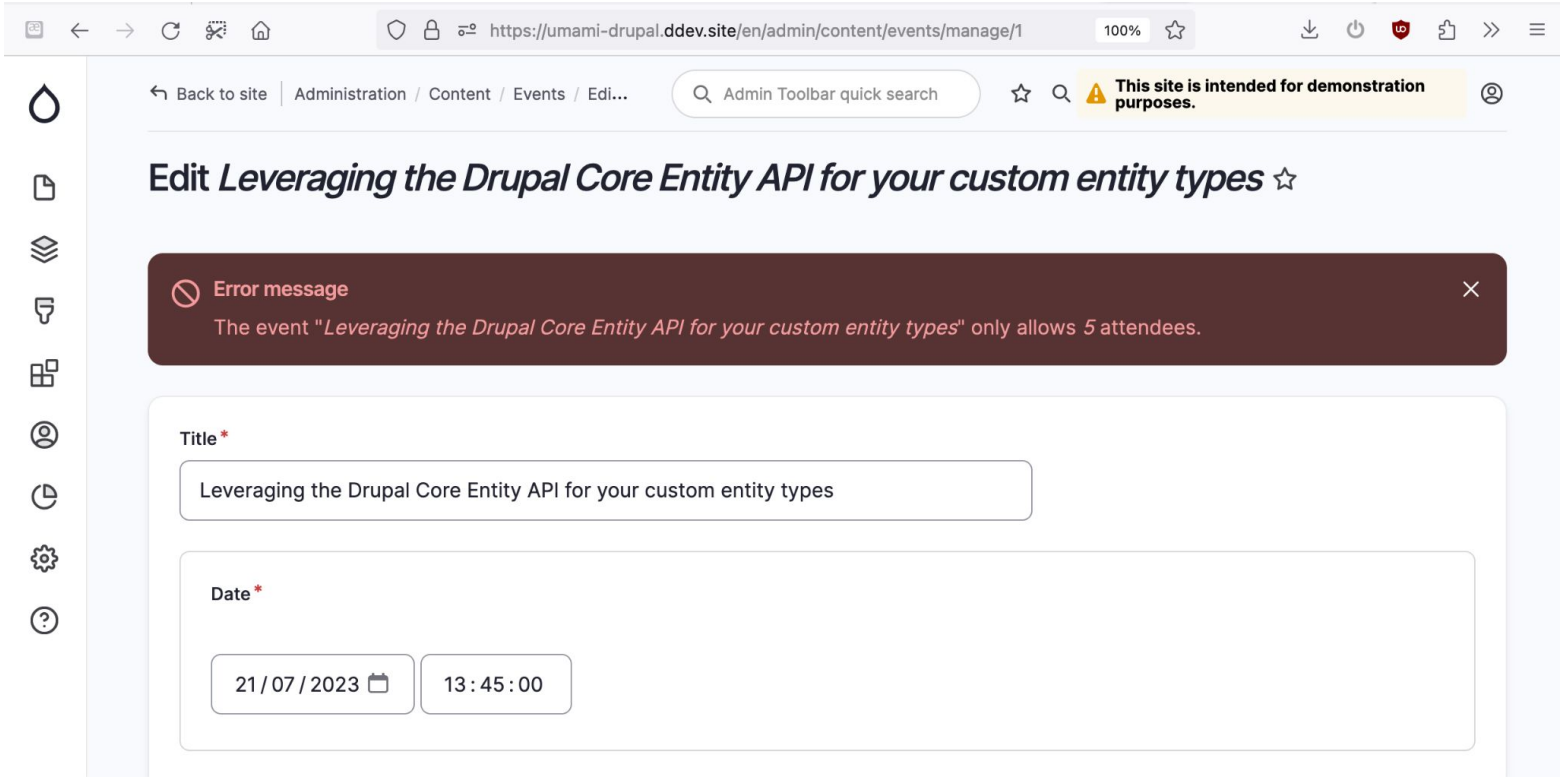
Entity constraints and validation

1x

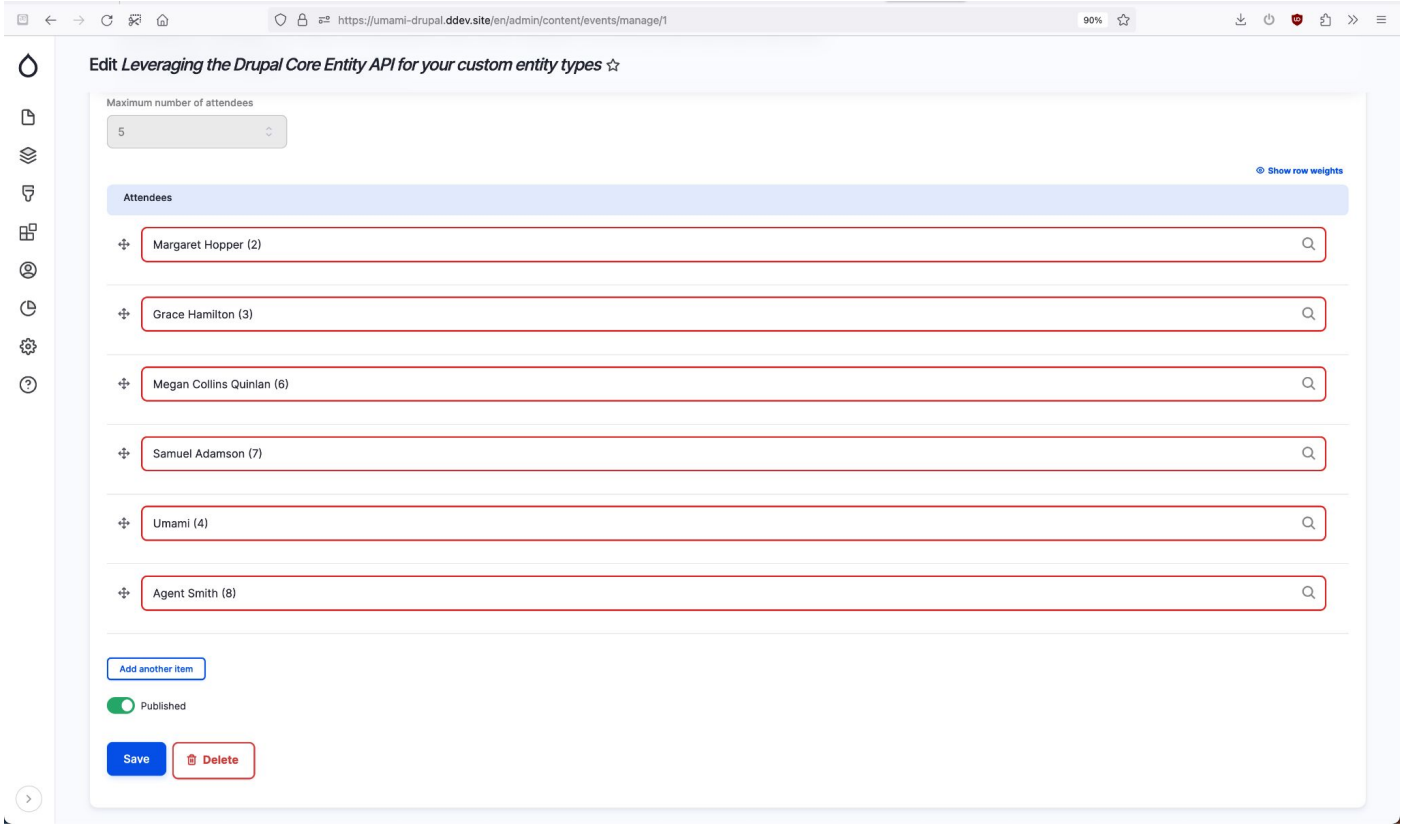
event.module

```php
<?php

use Drupal\Core\Form\FormStateInterface;

/**
 * Implements hook_form_BASE_FORM_ID_alter() event entity forms.
 */
function event_form_event_edit_form_alter(&$form, FormStateInterface $form_state, $form_id) {
  /** @var \Drupal\event\Entity\Event $entity */
  $entity = $form_state->getFormObject()
    ->getEntity();
  if (\Drupal::currentUser()->id() != $entity->getOwnerId()) {
    $form['maximum']['#disabled'] = TRUE;
  }
}
```

Entity constraints and validation



Entity constraints and validation



Tracking changes to entities

1x

```php
class Event extends ContentEntityBase implements EntityOwnerInterface, EntityPublishedInterface, RevisionLogInterface {

  use EntityOwnerTrait, EntityPublishedTrait, RevisionLogEntityTrait;

  /**
   * {@inheritdoc}
   */
  public static function baseFieldDefinitions(EntityTypeInterface $entity_type) {
    // Get the field definitions for 'id' and 'uuid' from the parent.
    $fields = parent::baseFieldDefinitions($entity_type);
    $fields += static::revisionLogBaseFieldDefinitions($entity_type);
```

OR

```php
class Event extends RevisionableContentEntityBase implements EntityOwnerInterface, EntityPublishedInterface {

  use EntityOwnerTrait, EntityPublishedTrait;
```

Tracking changes to entities

1x

```
*   entity_keys = {
*     "id" = "id",
*     "revision" = "vid",
*     "uuid" = "uuid",
*     "owner" = "author",
*     "published" = "published",
*     "label" = "title",
*   },
*   entity_keys = {
*     "id" = "id",
*     "revision" = "vid",
*     "uuid" = "uuid",
*     "owner" = "author",
*     "published" = "published",
*     "label" = "title",
*   },
*   revision_table = "event_revision",
*   revision_metadata_keys = {
*     "revision_user" = "revision_author",
*     "revision_created" = "revision_created",
*     "revision_log_message" = "revision_log_message",
*   },
*   show_revision_ui = true,
```

```
*   handlers = {
*     "access" = "Drupal\entity\EntityAccessControlHandler",
*     "permission_provider" = "Drupal\entity\EntityPermissionProvider",
*     "route_provider" = {
*       "default" = "Drupal\entity\Routing\DefaultHtmlRouteProvider",
*       "revision" = \Drupal\Core\Entity\Routing\RevisionHtmlRouteProvider::class,
*     },
```

```
*   links = {
*     "canonical" = "/event/{event}",
*     "add-form" = "/admin/content/events/add",
*     "edit-form" = "/admin/content/events/manage/{event}",
*     "delete-form" = "/admin/content/events/manage/{event}/delete",
*     "collection" = "/admin/content/events",
*     "version-history" = "/event/{event}/revisions",
*     "revision" = "/event/{event}/revisions/{event_revision}",
*     "revision-revert-form" = "/event/{event}/revisions/{event_revision}/revert",
*   },
```

Tracking changes to entities

1x

event /* The base table for event entities. */

vid	int(10) unsigned
uuid	varchar(128)
author /* The ID of the target entity. */	int(10) unsigned
published	tinyint(4)
title	varchar(255)
date /* The date value. */	varchar(20)
description__value	longtext
description__format	varchar(255)
maximum	int(11)
id	int(10) unsigned

event__attendees /* Data storage for event field attendees. */

bundle /* The field instance bundle to which this row belongs, used whe... */	
revision_id /* The entity revision id this data is attached to */	)) unsigned
attendees_target_id /* The ID of the target entity. */	int(10) unsigned
deleted /* A boolean indicating whether this data item has been deleted */	
entity_id /* The entity id this data is attached to */	int(10) unsigned
langcode /* The language code for this data item. */	varchar(32)
delta /* The sequence number for this data item, used for multi-value ... */	1

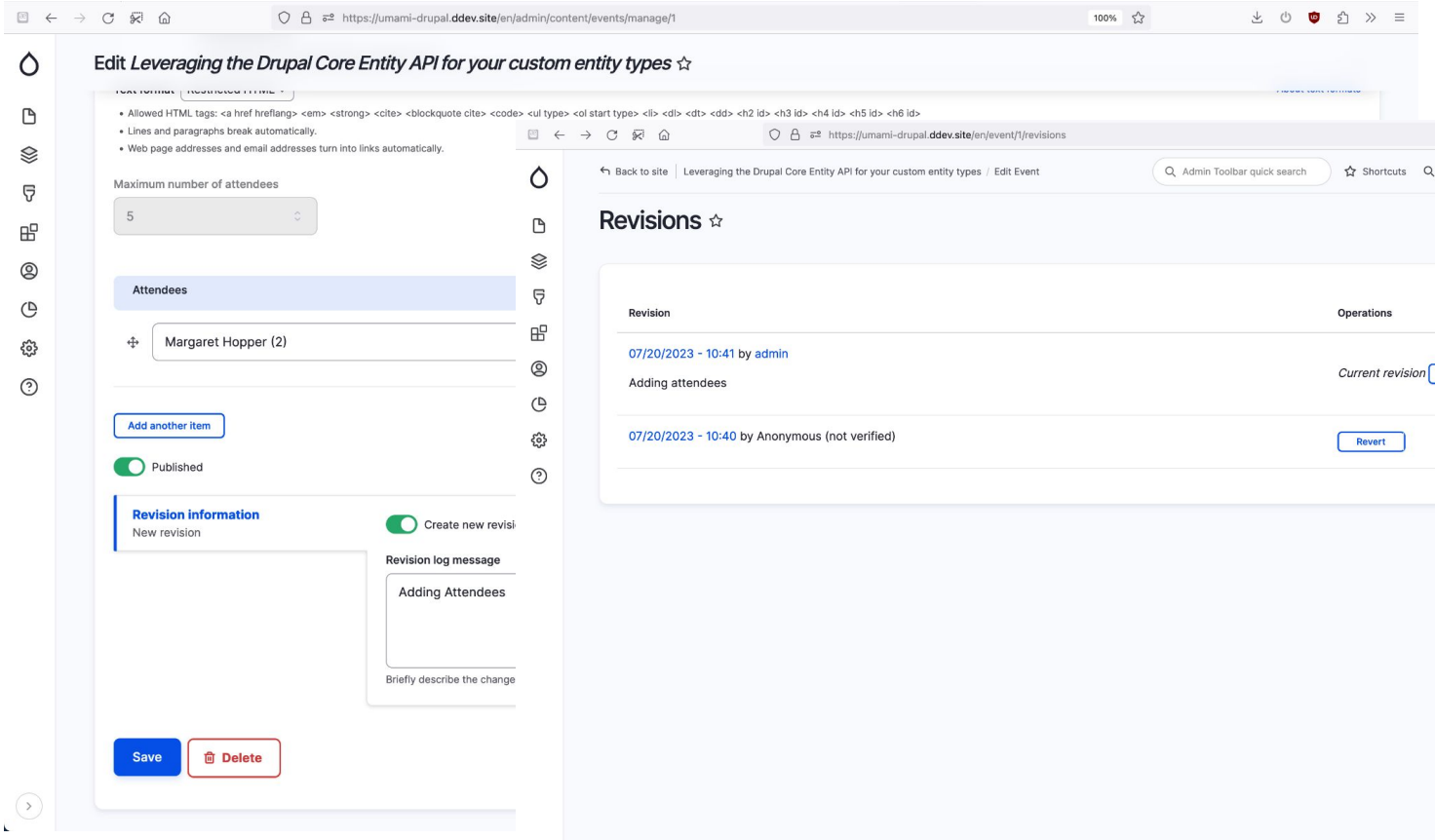
event_revision /* The revision table for event entities. */

id	int(10) unsigned
revision_created	int(11)
revision_author /* The ID of the target entity. */	int(10) unsigned
revision_log_message	longtext
published	tinyint(4)
revision_default	tinyint(4)
vid	int(10) unsigned

event_revision__attendees /* Revision archive storage for event field attendees. */

bundle /* The field instance bundle to which this row belongs, used whe... */	varchar(128)
attendees_target_id /* The ID of the target entity. */	int(10) unsigned
deleted /* A boolean indicating whether this data item has been deleted */	tinyint(4)
entity_id /* The entity id this data is attached to */	int(10) unsigned
revision_id /* The entity revision id this data is attached to */	int(10) unsigned
langcode /* The language code for this data item. */	varchar(32)
delta /* The sequence number for this data item, used for multi-value ... */	int(10) unsigned

Tracking changes to entities



Config entity - Event Type

EventType.php

```php
<?php

namespace Drupal\event\Entity;

use Drupal\Core\Config\Entity\ConfigEntityBase;

/**
 * @ConfigEntityType(
 *   id = "event_type",
 *   label = @Translation("Event type"),
 *   label_collection = @Translation("Event types"),
 *   label_singular = @Translation("event type"),
 *   label_plural = @Translation("event types"),
 *   config_prefix = "type",
 *   config_export = {
 *     "id",
 *     "label",
 *   },
 *   entity_keys = {
 *     "id" = "id",
 *     "label" = "label",
 *   },
 * )
 */
class EventType extends ConfigEntityBase {

  /** The machine name of this event type. ...*/
  protected $id;

  /** The human-readable name of the node type. ...*/
  protected $label;

}
```

umami-drupal 〉 web 〉 modules 〉 custom 〉 event 〉 config 〉 schema 〉 event.schema.yml

Project

event.schema.yml

```yaml
event.type.*:
  type: config_object
  mapping:
    id:
      type: string
      label: 'ID'
    label:
      type: label
      label: 'Label'
```

Config entity - Event Type

```
*   handlers = {
*     "access" = "Drupal\entity\EntityAccessControlHandler",
*     "list_builder" = "Drupal\event\Controller\EventTypeListBuilder",
*     "local_action_provider" = {
*       "collection" = "Drupal\entity\Menu\EntityCollectionLocalActionProvider",
*     },
*     "permission_provider" = "Drupal\entity\EntityPermissionProvider",
*     "route_provider" = {
*       "default" = "Drupal\entity\Routing\DefaultHtmlRouteProvider",
*     },
*   },
*   links = {
*     "collection" = "/admin/structure/event-types",
*   },
*   admin_permission = "administer event_type",
```

EventTypeListBuilder.php

```php
<?php

namespace Drupal\event\Controller;

use Drupal\Core\Entity\EntityInterface;
use Drupal\Core\Entity\EntityListBuilder;

/**
 * Defines an implementation to build a listing of event type entities.
 */
class EventTypeListBuilder extends EntityListBuilder {

  /** {@inheritdoc} ...*/
  public function buildHeader() {
    $header = [];
    $header['label'] = $this->t( string: 'Label');
    return $header + parent::buildHeader();
  }

  /** {@inheritdoc} ...*/
  public function buildRow(EntityInterface $event) {
    $row = [];
    $row['label'] = $event->label();
    return $row + parent::buildRow($event);
  }

}
```

Config entity - Event Type

```
*   handlers = {
*     "access" = "Drupal\entity\EntityAccessControlHandler",
*     "list_builder" = "Drupal\event\Controller\EventTypeListBuilder",
*     "local_action_provider" = {
*       "collection" = "Drupal\entity\Menu\EntityCollectionLocalActionProvider",
*     },
*     "permission_provider" = "Drupal\entity\EntityPermissionProvider",
*     "route_provider" = {
*       "default" = "Drupal\entity\Routing\DefaultHtmlRouteProvider",
*     },
*     "form" = {
*       "add" = "Drupal\event\Form\EventTypeForm",
*       "edit" = "Drupal\event\Form\EventTypeForm",
*       "delete" = "Drupal\Core\Entity\EntityDeleteForm",
*     },
*   },
*   links = {
*     "collection" = "/admin/structure/event-types",
*     "add-form" = "/admin/structure/event-types/add",
*     "edit-form" = "/admin/structure/event-types/manage/{event_type}",
*     "delete-form" = "/admin/structure/event-types/manage/{event_type}/delete",
*   },
```

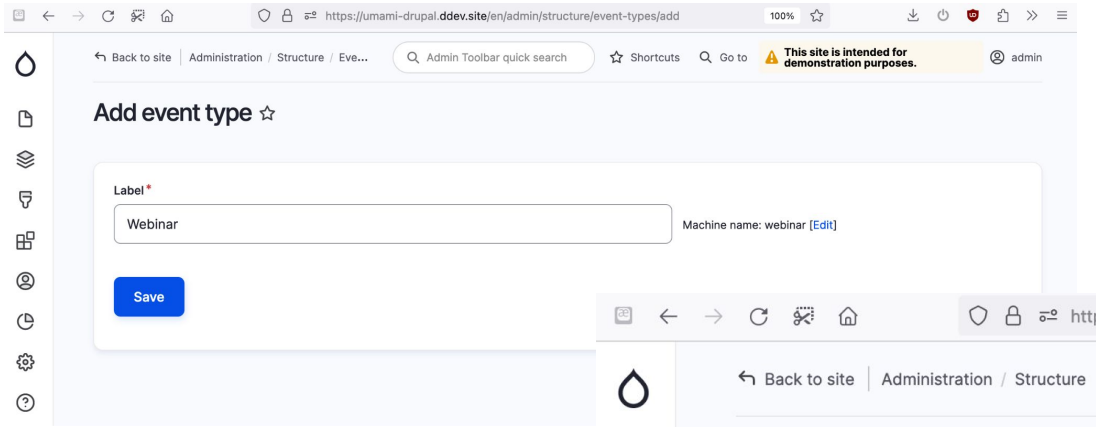
EventTypeForm.php

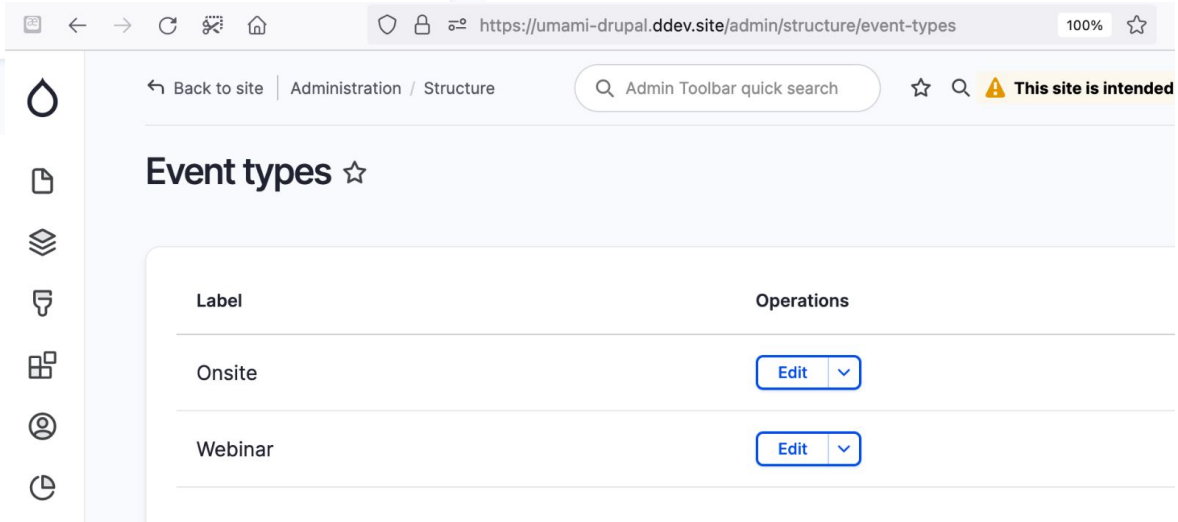
```php
<?php

namespace Drupal\event\Form;

use Drupal\Core\Entity\BundleEntityFormBase;
use Drupal\Core\Entity\EntityTypeInterface;
use Drupal\Core\Form\FormStateInterface;

/**
 * Form handler for event type forms.
 */
class EventTypeForm extends BundleEntityFormBase {

  /** {@inheritdoc} ...*/
  public function form(array $form, FormStateInterface $form_state) {
    $form = parent::form($form, $form_state);

    $event_type = $this->getEntity();

    $form['label'] = [
      '#type' => 'textfield',
      '#title' => $this->t( string: 'Label'),
      '#default_value' => $event_type->label(),
      '#required' => TRUE,
    ];

    $form['id'] = [
      '#type' => 'machine_name',
      '#title' => $this->t( string: 'ID'),
      '#maxlength' => EntityTypeInterface::BUNDLE_MAX_LENGTH,
      '#default_value' => $event_type->id(),
      '#machine_name' => [
        'exists' => [$event_type->getEntityType()->getClass(), 'load'],
      ],
      '#disabled' => !$event_type->isNew(),
    ];

    return $form;
  }
```

Config entity - Event Type

```php
/**
 * {@inheritdoc}
 */
public function save(array $form, FormStateInterface $form_state) {
  parent::save($form, $form_state);

  $entity = $this->getEntity();
  $entity_type = $entity->getEntityType();

  $arguments = [
    '@entity_type' => $entity_type->getSingularLabel(),
    '%entity' => $entity->label(),
    'link' => $entity->toLink($this->t( string: 'Edit'),  rel: 'edit-form')->toString(),
  ];

  $this->logger($entity->getEntityTypeId())->notice( message: 'The @entity_type %entity has been saved.', $arguments);
  $this->messenger()->addMessage($this->t( string: 'The @entity_type %entity has been saved.', $arguments));

  $form_state->setRedirectUrl($entity->toUrl( rel: 'collection'));
}
```

Config entity - Event Type





Configuring bundles in the user interface

1x

Event.php

\Drupal\event\Entity

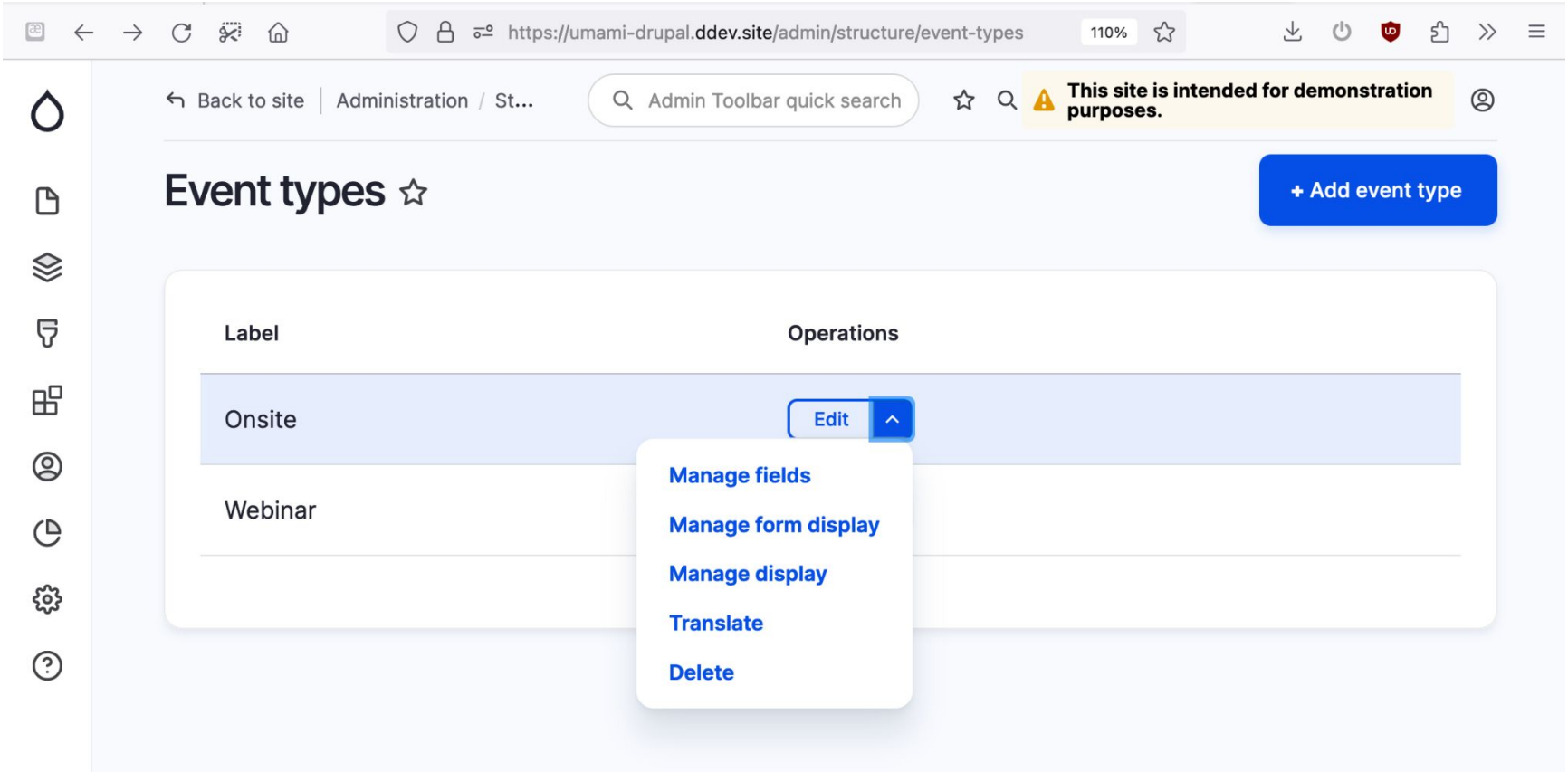
```
59   *   links = {
60   *     "canonical" = "/event/{event}",
61   *     "add-form" = "/admin/content/events/add/{event_type}",
62   *     "edit-form" = "/admin/content/events/manage/{event}",
63   *     "delete-form" = "/admin/content/events/manage/{event}/delete",
64   *     "collection" = "/admin/content/events",
65   *     "version-history" = "/event/{event}/revisions",
66   *     "revision" = "/event/{event}/revisions/{event_revision}",
67   *     "revision-revert-form" = "/event/{event}/revisions/{event_revision}/revert",
68   *   },
69   *   bundle_entity_type = "event_type",
70   *   field_ui_base_route = "entity.event_type.edit_form",
```

EventType.php

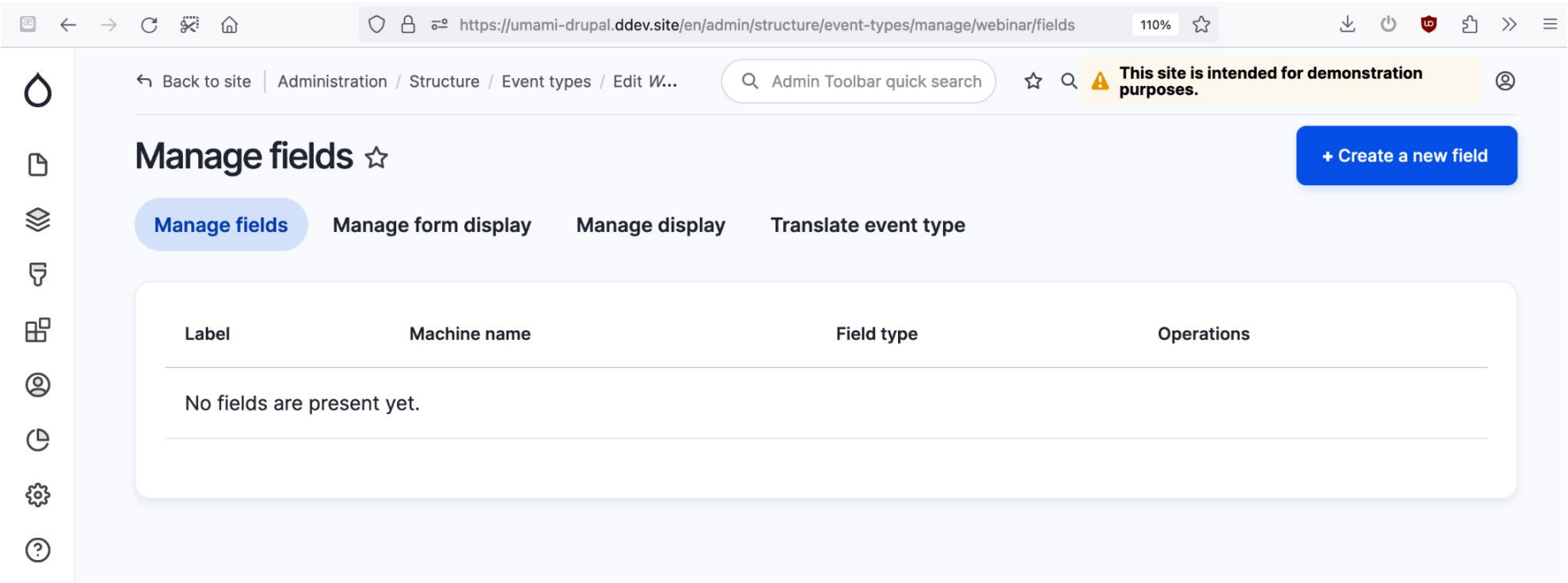
\Drupal\event\Entity

```
38   *   },
39   *   links = {
40   *     "collection" = "/admin/structure/event-types",
41   *     "add-form" = "/admin/structure/event-types/add",
42   *     "edit-form" = "/admin/structure/event-types/manage/{event_type}",
43   *     "delete-form" = "/admin/structure/event-types/manage/{event_type}/delete",
44   *   },
45   *   bundle_of = "event",
```

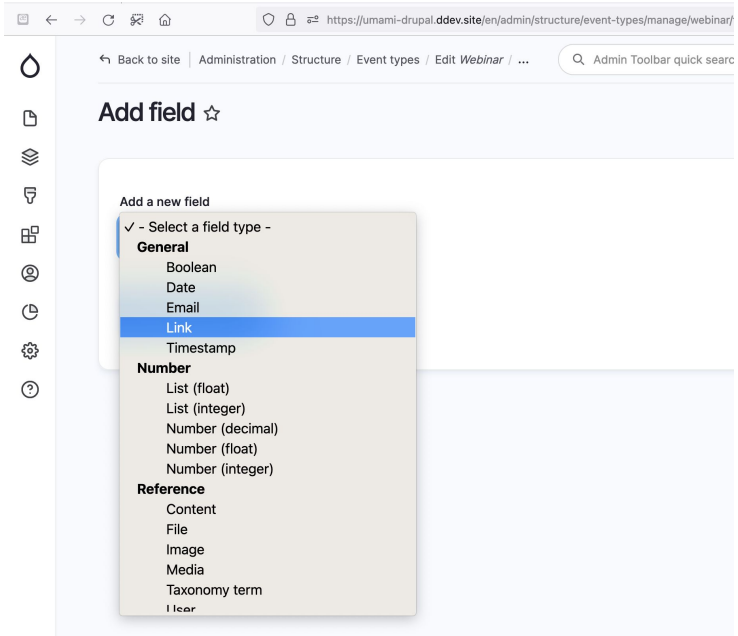
Configuring bundles in the user interface

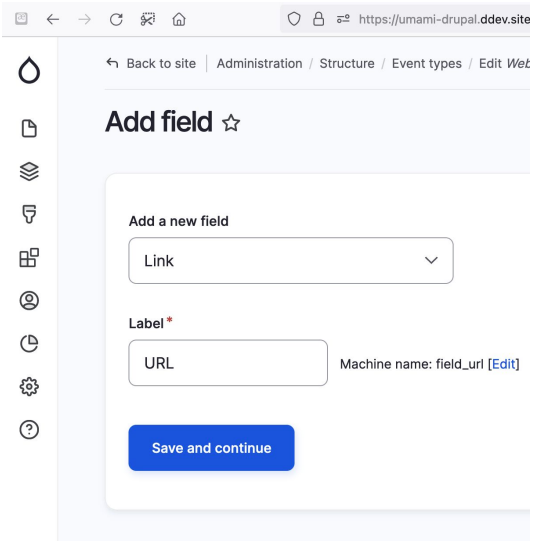


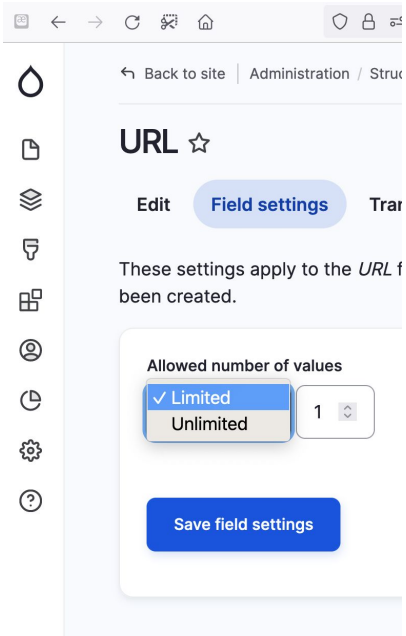
Configuring bundles in the user interface



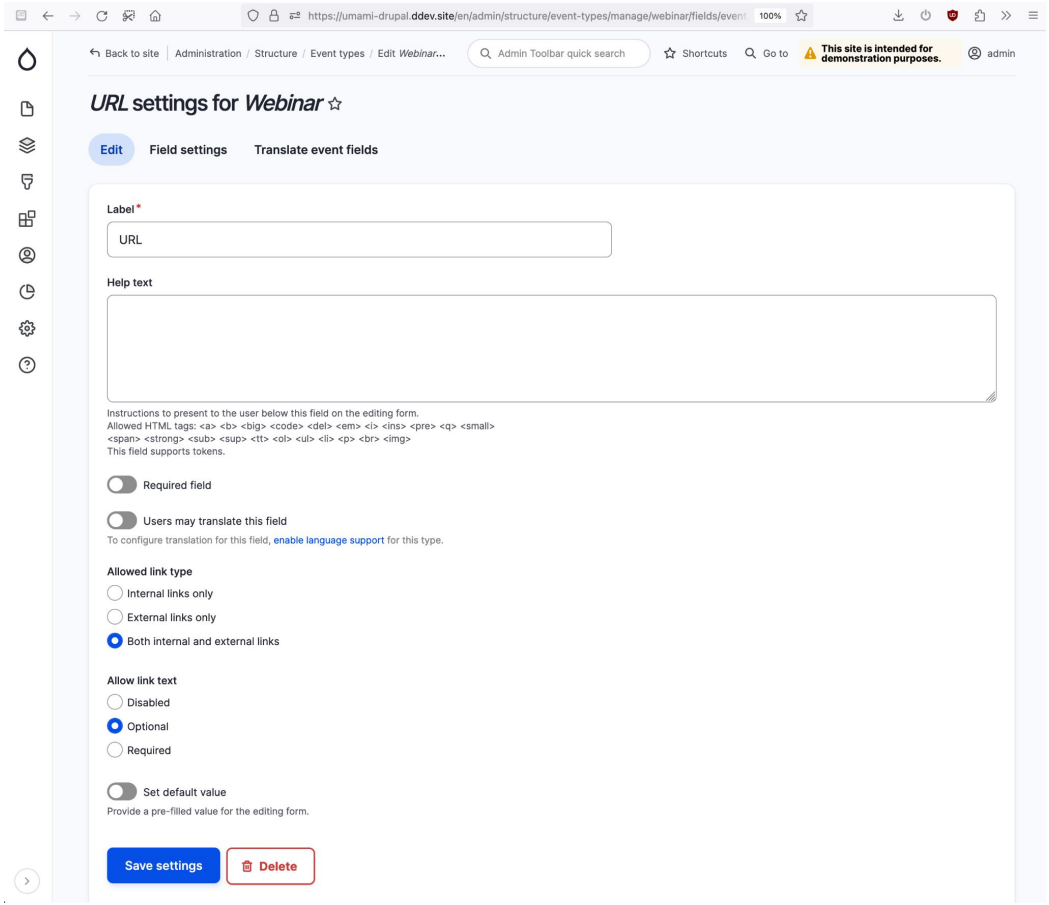
Configuring bundles in the user interface





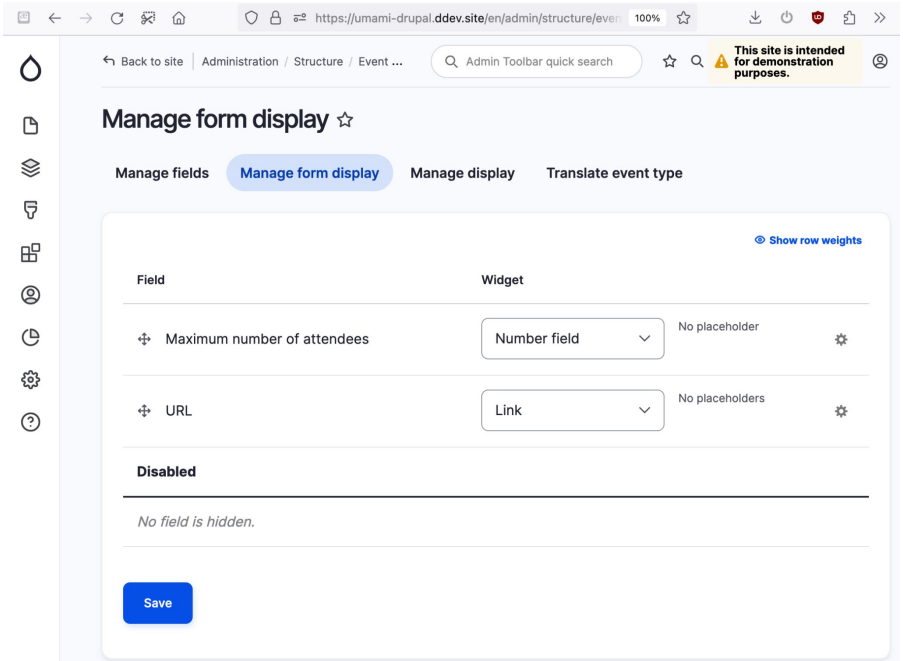


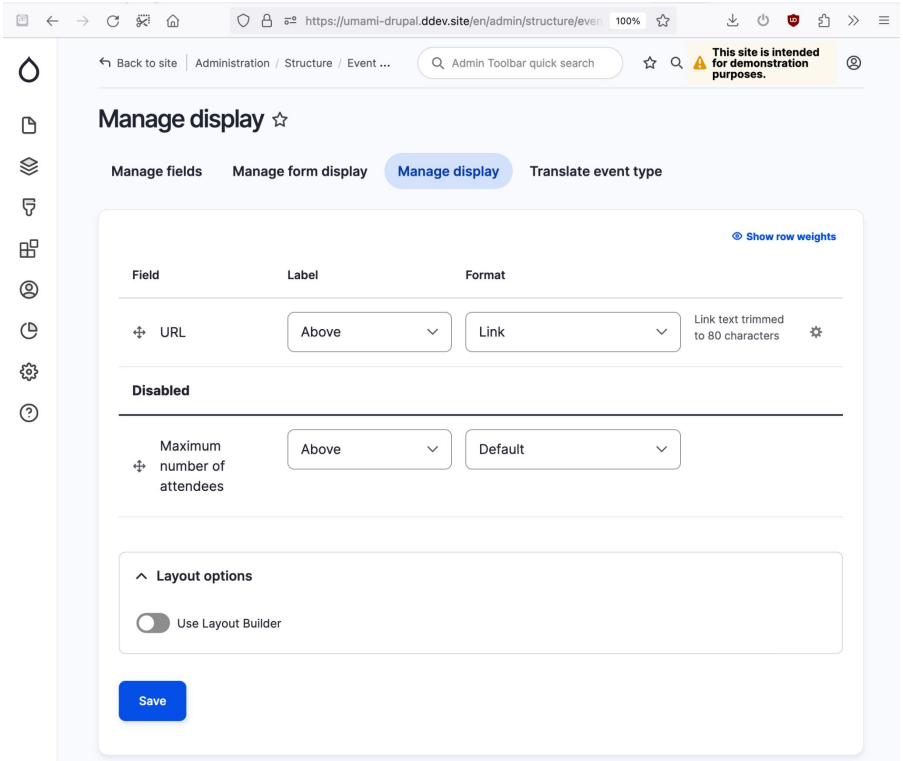
Configuring bundles in the user interface



Configuring bundles in the user interface

```
$fields['maximum'] = BaseFieldDefinition::create( type: 'integer')
  ->setLabel(t( string: 'Maximum number of attendees'))
  ->setSetting( setting_name: 'min',  value: 1)
  ->setDisplayOptions( display_context: 'form', ['weight' => 23])
  ->setDisplayConfigurable( display_context: 'form',  configurable: TRUE)
  ->setDisplayConfigurable( display_context: 'view',  configurable: TRUE);
```





Translatable entities

```
*   entity_keys = {
*     "id" = "id",
*     "revision" = "vid",
*     "bundle" = "type",
*     "langcode" = "langcode",
*     "uuid" = "uuid",
*     "owner" = "author",
*     "published" = "published",
*     "label" = "title",
*   },

*   translatable = TRUE,
*   data_table = "event_field_data",
*   revision_data_table = "event_revision_field_data",
```

event_field_data /* The data table for event entities. */

vid	int(10) unsigned
type /* The ID of the target entity. */	varchar(32)
author /* The ID of the target entity. */	int(10) unsigned
published	tinyint(4)
title	varchar(255)
date /* The date value. */	varchar(20)
description__value	longtext
description__format	varchar(255)
maximum	int(11)
default_langcode	tinyint(4)
revision_translation_affected	tinyint(4)
id	int(10) unsigned
langcode	varchar(12)

event_revision_field_data /* The revision data table for event entities. */

id	int(10) unsigned
published	tinyint(4)
default_langcode	tinyint(4)
revision_translation_affected	tinyint(4)
vid	int(10) unsigned
langcode	varchar(12)

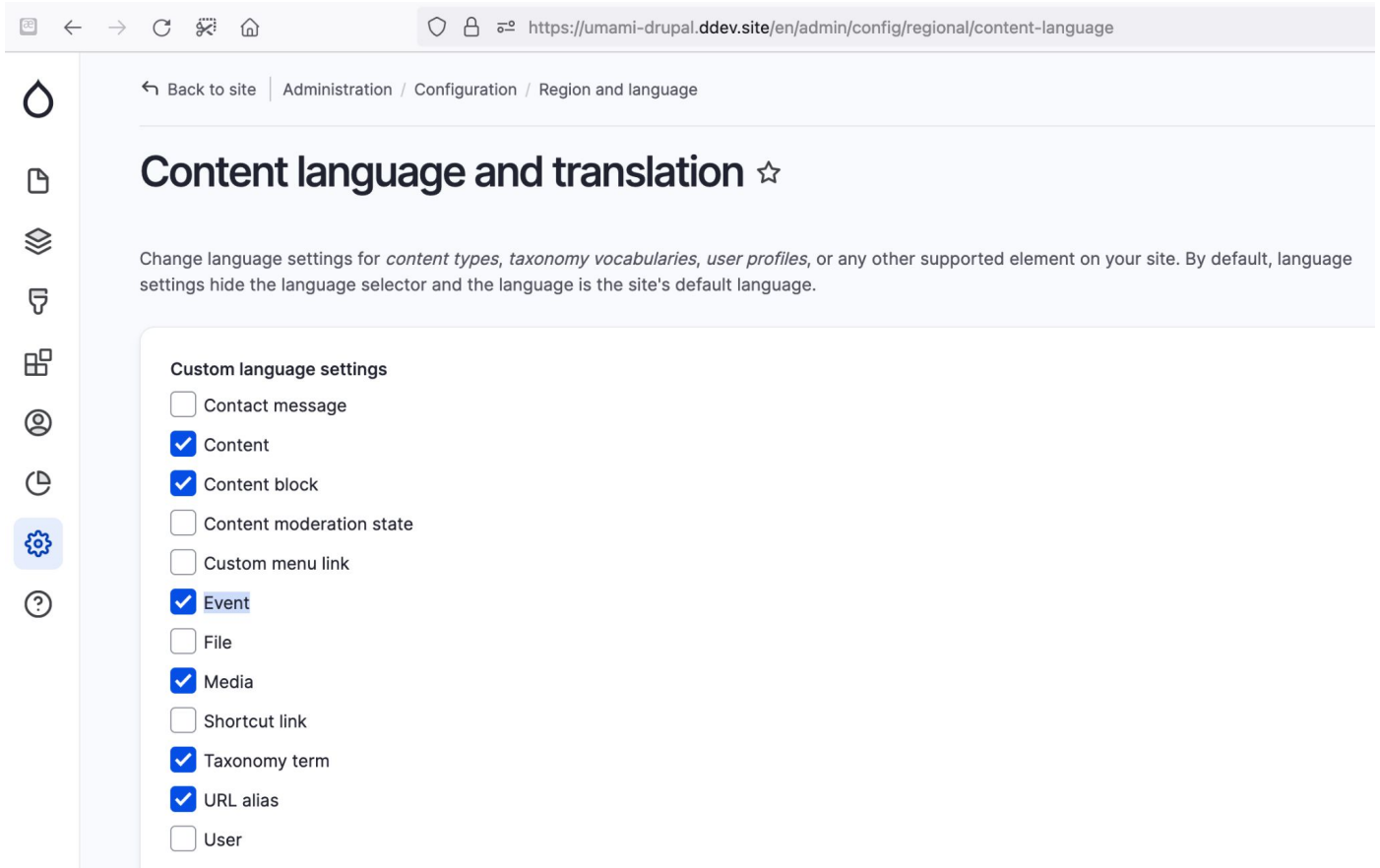
Translatable and revisionable fields

```
$fields['title'] = BaseFieldDefinition::create( type: 'string')
  ->setLabel(t( string: 'Title'))
  ->setRequired( required: TRUE)
  ->setTranslatable( translatable: TRUE)
  ->setRevisionable( revisionable: TRUE)
  ->setDisplayOptions( display_context: 'form', ['weight' => 0]);
```

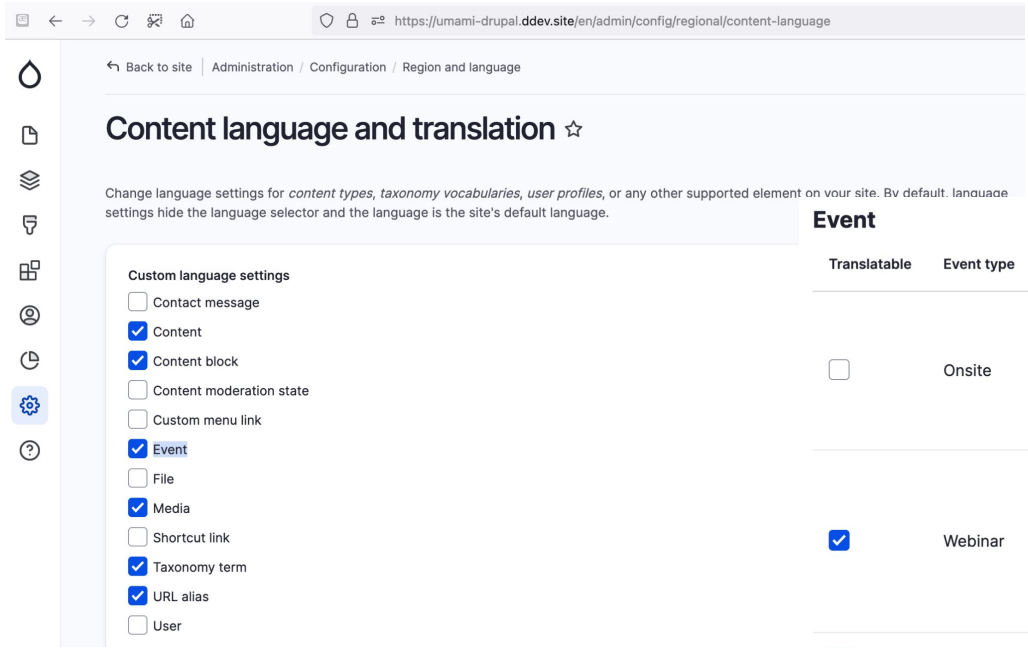
event_revision_field_data /* The revision data table for event entities. */	
id	int(10) unsigned
published	tinyint(4)
default_langcode	tinyint(4)
revision_translation_affected	tinyint(4)
vid	int(10) unsigned
langcode	varchar(12)

event_revision_field_data /* The revision data table for event entities. */	
id	int(10) unsigned
published	tinyint(4)
title	varchar(255)
default_langcode	tinyint(4)
revision_translation_affected	tinyint(4)
vid	int(10) unsigned
langcode	varchar(12)

Configuring translations per bundle



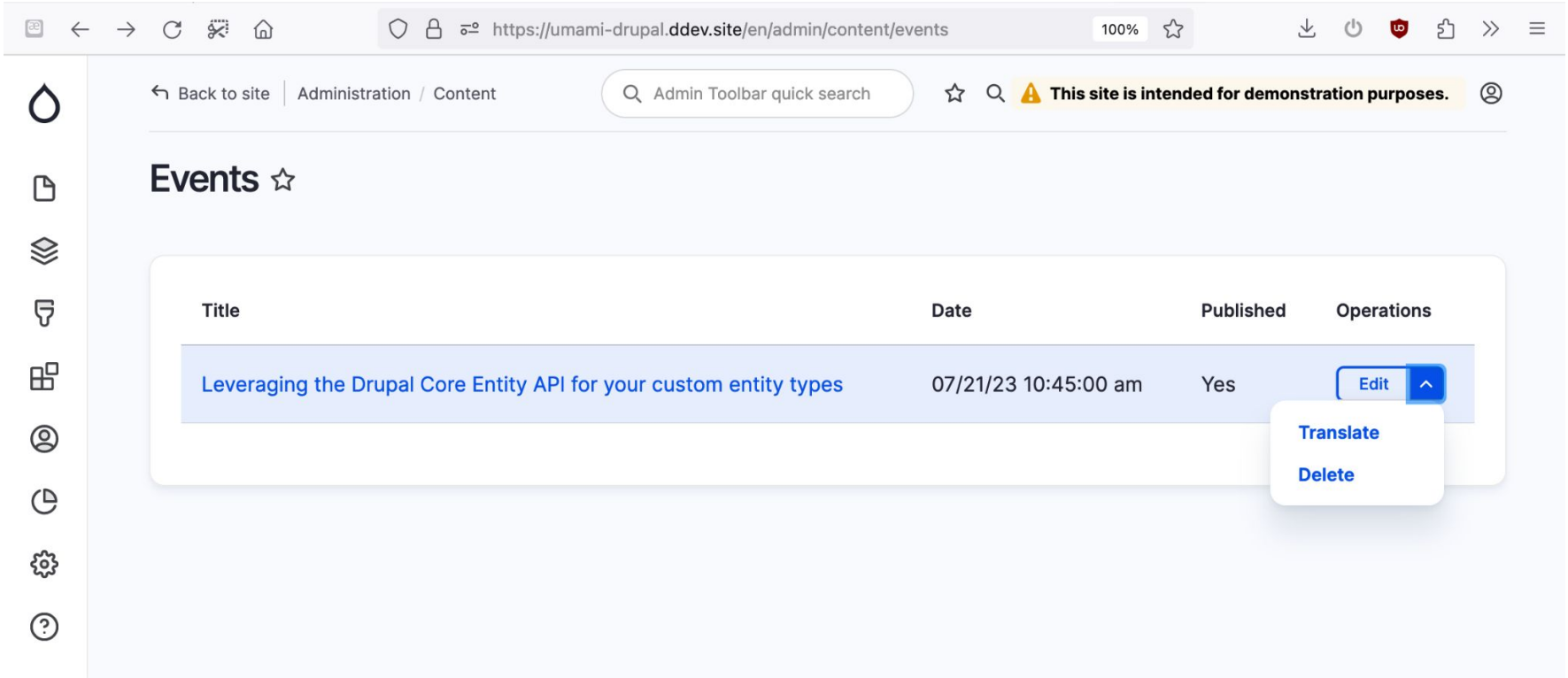
Configuring translations per bundle



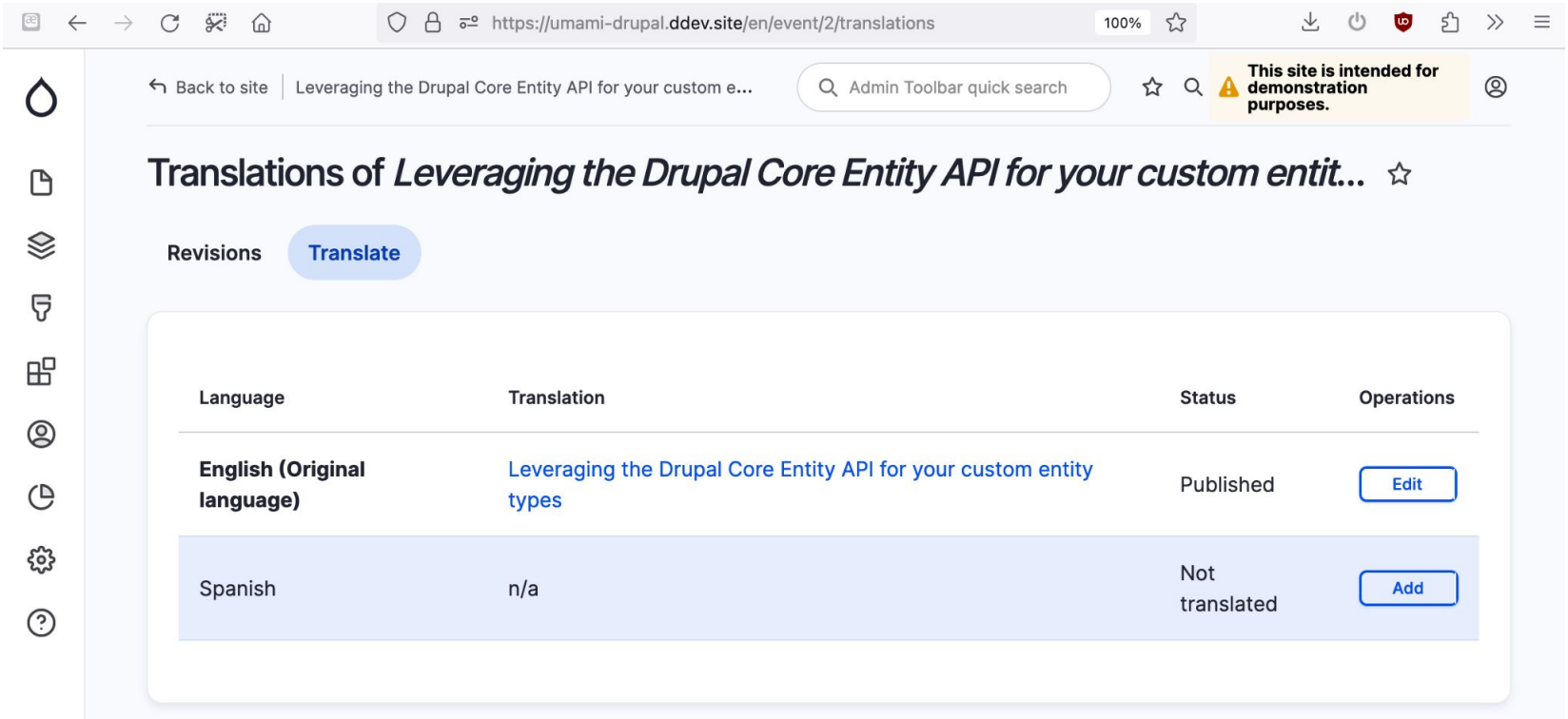
Event

Translatable	Event type	Configuration
☐	Onsite	Default language: Site's default language (English). Explanation of the language options is found on the languages list page. ○ Show language selector on create and edit pages
☑	Webinar	Default language: Site's default language (English). Explanation of the language options is found on the languages list page. ● Show language selector on create and edit pages ○ Hide non translatable fields on translation forms
☑	User ID	
☑	Published	
☑	Title	
☑	Description	
☐	URL	

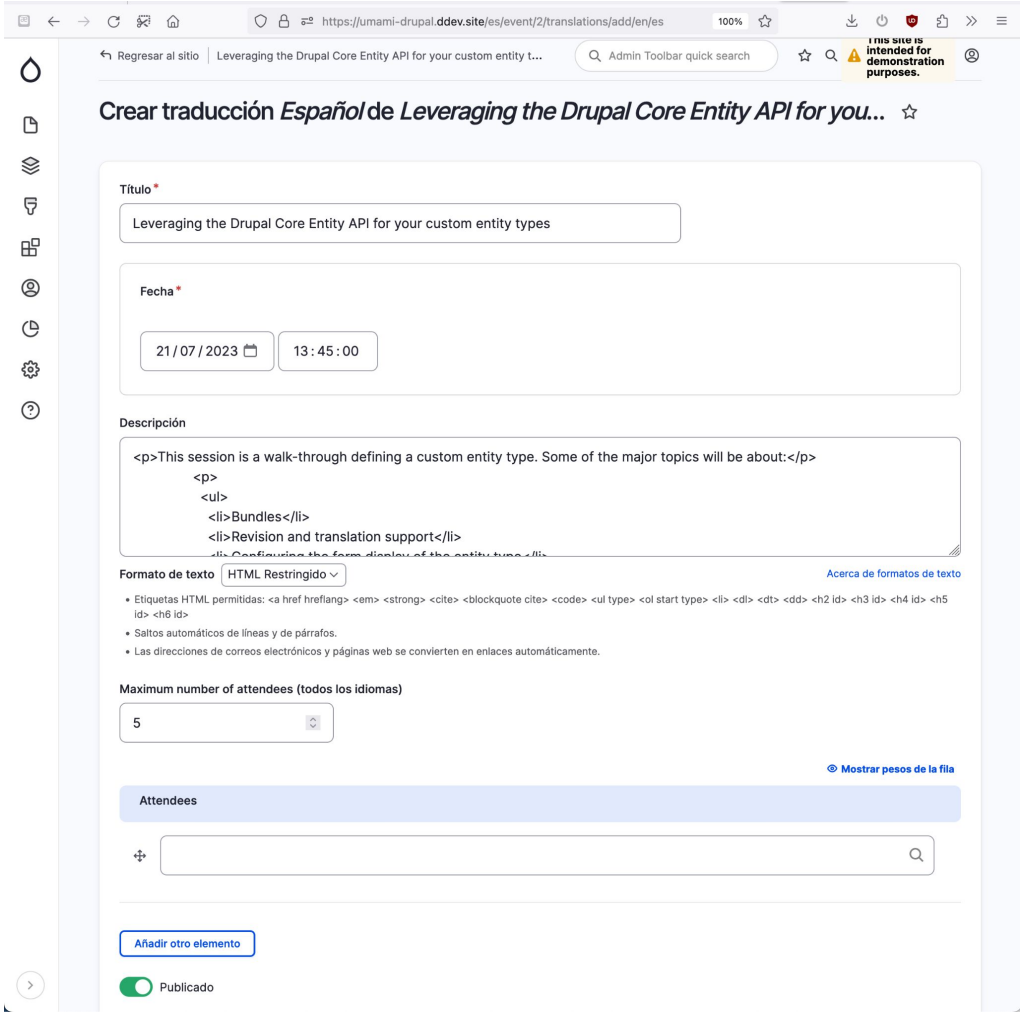
Translating entities



Translating entities

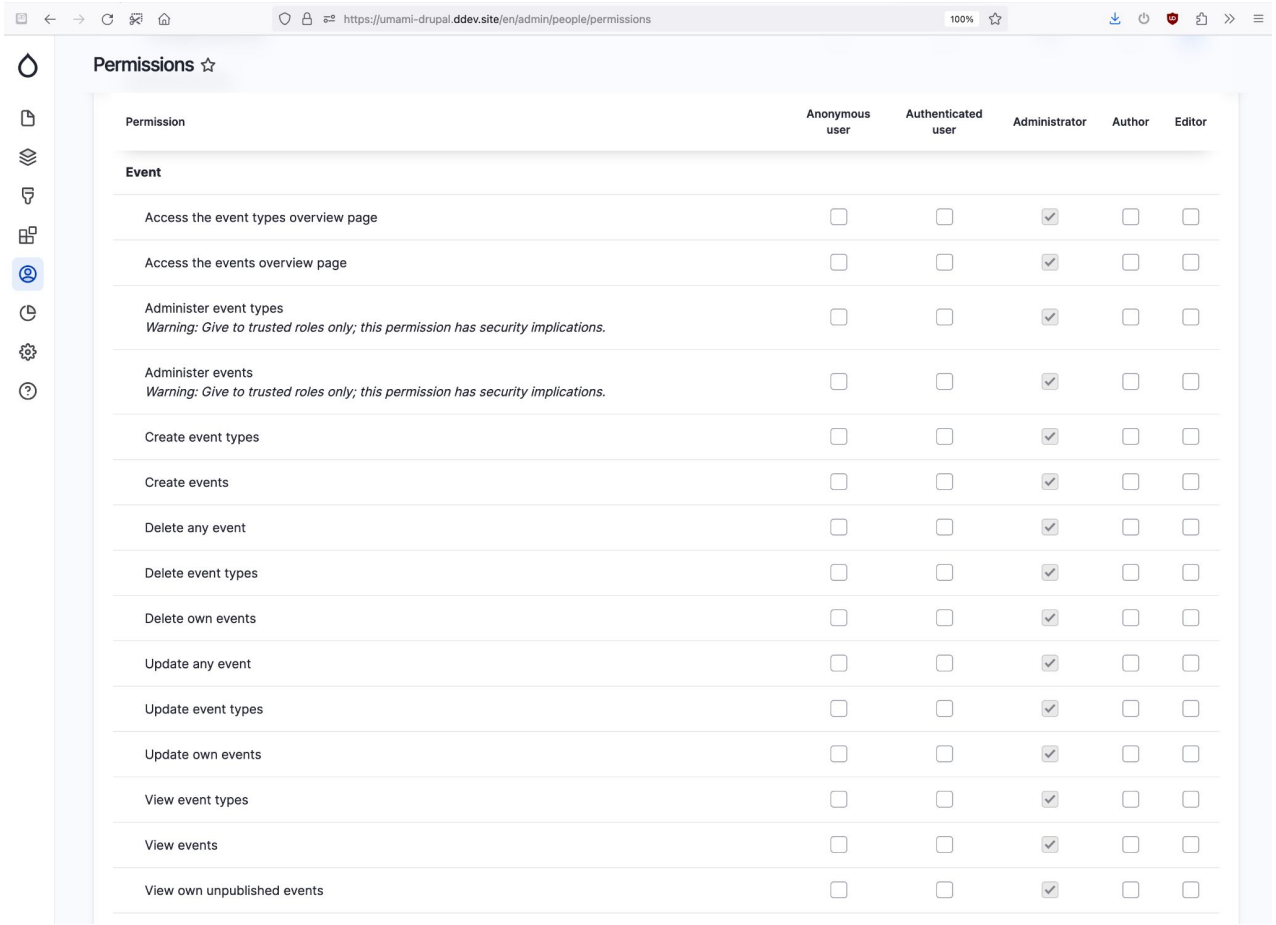


Translating entities



Access control and permissions

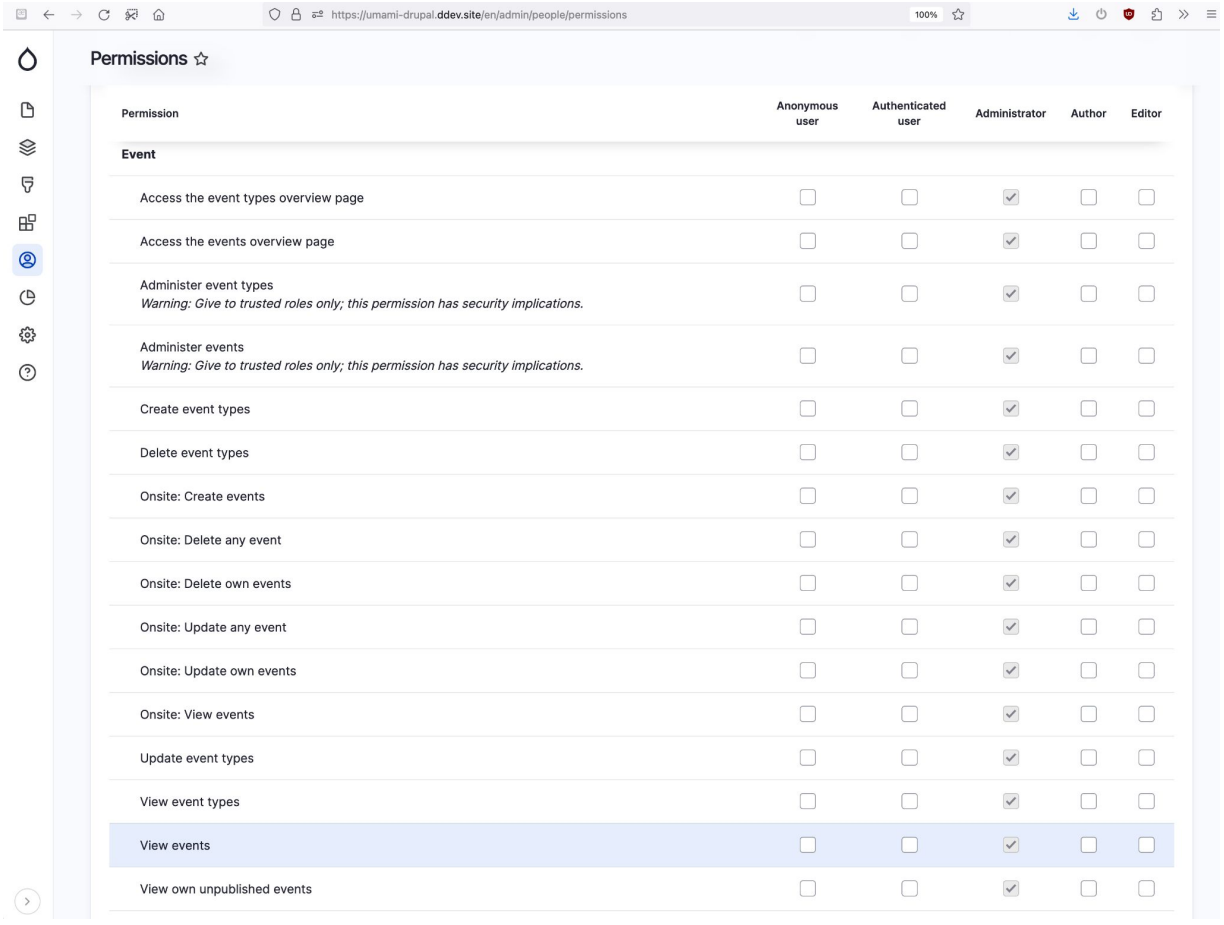
permission_granularity = “entity_type”



Permission	Anonymous user	Authenticated user	Administrator	Author	Editor
Event					
Access the event types overview page	☐	☐	☑	☐	☐
Access the events overview page	☐	☐	☑	☐	☐
Administer event types *Warning: Give to trusted roles only; this permission has security implications.*	☐	☐	☑	☐	☐
Administer events *Warning: Give to trusted roles only; this permission has security implications.*	☐	☐	☑	☐	☐
Create event types	☐	☐	☑	☐	☐
Create events	☐	☐	☑	☐	☐
Delete any event	☐	☐	☑	☐	☐
Delete event types	☐	☐	☑	☐	☐
Delete own events	☐	☐	☑	☐	☐
Update any event	☐	☐	☑	☐	☐
Update event types	☐	☐	☑	☐	☐
Update own events	☐	☐	☑	☐	☐
View event types	☐	☐	☑	☐	☐
View events	☐	☐	☑	☐	☐
View own unpublished events	☐	☐	☑	☐	☐

Access control and permissions

permission_granularity = “bundle”



Entity queries

```
\Drupal::entityTypeManager()
  ->getStorage( entity_type_id: 'event')
  ->getQuery()
  ->accessCheck( access_check: FALSE)
  ->condition( field: 'type',  value: 'webinar')
  ->condition( field: 'date',  value: '2023-07-21%',  operator: 'LIKE',  langcode: 'es')
  ->execute();
// Returns [2 => "2"]
```

```
\Drupal::entityTypeManager()
  ->getStorage( entity_type_id: 'event')
  ->loadMultiple($entity_ids);
```

Entity type annotation

```
/**
 * @ContentEntityType(
 *   id = "event",
 *   label = @Translation("Event"),
 *   label_collection = @Translation("Events"),
 *   label_singular = @Translation("event"),
 *   label_plural = @Translation("events"),
 *   base_table = "event",
 *   entity_keys = {
 *     "id" = "id",
 *     "revision" = "vid",
 *     "bundle" = "type",
 *     "langcode" = "langcode",
 *     "uuid" = "uuid",
 *     "owner" = "author",
 *     "published" = "published",
 *     "label" = "title",
 *   },
 *   revision_table = "event_revision",
 *   revision_metadata_keys = {
 *     "revision_user" = "revision_author",
 *     "revision_created" = "revision_created",
 *     "revision_log_message" = "revision_log_message",
 *   },
 *   show_revision_ui = true,
 *   translatable = TRUE,
 *   data_table = "event_field_data",
 *   revision_data_table = "event_revision_field_data",
 *   handlers = {
 *     "access" = "Drupal\entity\EntityAccessControlHandler",
 *     "permission_provider" = "Drupal\entity\EntityPermissionProvider",
 *     "route_provider" = {
 *       "default" = "Drupal\entity\Routing\DefaultHtmlRouteProvider",
 *       "revision" = \Drupal\Core\Entity\Routing\RevisionHtmlRouteProvider::class,
 *     },
 *     "form" = {
 *       "add" = "Drupal\Core\Entity\ContentEntityForm",
 *       "edit" = "Drupal\Core\Entity\ContentEntityForm",
 *       "delete" = "Drupal\Core\Entity\ContentEntityDeleteForm",
 *     },
 *     "list_builder" = "Drupal\event\Controller\EventListBuilder",
 *     "local_action_provider" = {
 *       "collection" = "Drupal\entity\Menu\EntityCollectionLocalActionProvider",
 *     },
 *   },
 *   links = {
 *     "canonical" = "/event/{event}",
 *     "add-form" = "/admin/content/events/add/{event_type}",
 *     "edit-form" = "/admin/content/events/manage/{event}",
 *     "delete-form" = "/admin/content/events/manage/{event}/delete",
 *     "collection" = "/admin/content/events",
 *     "version-history" = "/event/{event}/revisions",
 *     "revision" = "/event/{event}/revisions/{event_revision}",
 *     "revision-revert-form" = "/event/{event}/revisions/{event_revision}/revert",
 *   },
 *   bundle_entity_type = "event_type",
 *   admin_permission = "administer event",
 *   permission_granularity = "bundle",
 *   field_ui_base_route = "entity.event_type.edit_form",
 * )
 */
```

Thank you for joining us!

Stay connected with Drupal Austria for more exciting events, news, and updates!

Also visit our website:

https://ddd23.drupalcamp.at/

drupalaustria

And don't forget to visit our sponsors' stands.

Their support helps make events like this possible!

1x INTERNET

I/O amazee.io part of Mirantis

Droptica

agiledrop Trusted Drupal teammates

Kraut.Hosting No Clouds Just Sunshine

eloo

Druid

drop solid