

Custom entities in Backdrop what, how and why

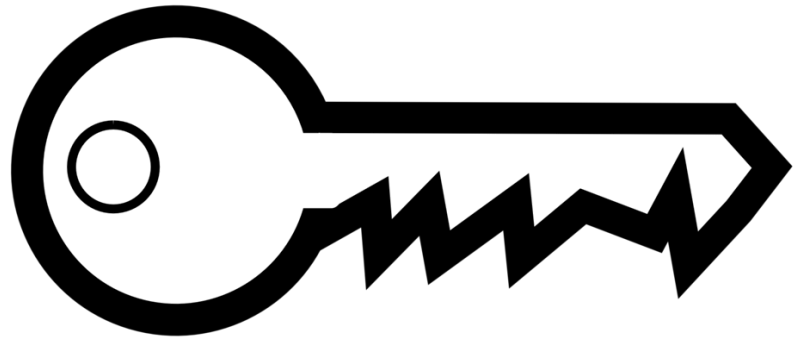


Alejandro Cremaschi (@argiepiano)

What is an entity?

A PHP object that:

- Contains data gathered from the database, or
- Contains data that will be saved to the database



Node object (nid: 1)

... (Object) Node

is_active_revision	(String, 1 characters) 1
nid	(String, 1 characters) 1
vid	(String, 1 characters) 1
type	(String, 4 characters) post
langcode	(String, 3 characters) und
title	(String, 16 characters) Your first post!
uid	(String, 1 characters) 1
status	(String, 1 characters) 1
created	(String, 10 characters) 1731468406
changed	(String, 10 characters) 1731468406
scheduled	(String, 1 characters) 0
comment	(String, 1 characters) 0
promote	(String, 1 characters) 1
sticky	(String, 1 characters) 0
tnid	(String, 1 characters) 0
translate	(String, 1 characters) 0
revision_timestamp	(String, 1 characters) 0
revision_uid	(String, 1 characters) 0
in_preview	(NULL)
log	(String, 0 characters)
comment_close_override	(String, 1 characters) 0
body	(Array, 1 element)
field_image	(Array, 0 elements)
field_tags	(Array, 0 elements)
path	(Array, 5 elements)

Database

node table

Server: localhost:8889 » Database: bdttesting » Table: node » The base table for nodes

Browse Structure SQL Search Insert Export Import

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	D
1	nid	int(10)		unsigned	No	N
2	vid	int(10)		unsigned	Yes	N
3	type	varchar(32)	utf8mb4_general_ci		No	
4	langcode	varchar(12)	utf8mb4_general_ci		No	
5	title	varchar(255)	utf8mb4_general_ci		No	
6	uid	int(11)			No	0

field_data_field_tags

Server: localhost:8889 » Database: bdttesting » Table: field_data_field_tags

Browse Structure SQL Search Insert Export

Table structure Relation view

#	Name	Type	Collation	Attributes	Null
1	entity_type	varchar(128)	utf8mb4_general_ci		No
2	bundle	varchar(128)	utf8mb4_general_ci		No
3	deleted	tinyint(4)			No
4	entity_id	int(10)		unsigned	No
5	revision_id	int(10)		unsigned	Yes

field_data_body table

Server: localhost:8889 » Database: bdttesting » Table: field_data_body

Browse Structure SQL Search Insert

Table structure Relation view

#	Name	Type	Collation
1	entity_type	varchar(128)	utf8mb4_general_ci
2	bundle	varchar(128)	utf8mb4_general_ci
3	deleted	tinyint(4)	
4	entity_id	int(10)	

field_data_field_image

Server: localhost:8889 » Database: bdttesting » Table: field_data_field_image

Browse Structure SQL Search Insert Export

Table structure Relation view

#	Name	Type	Collation	Attributes
1	entity_type	varchar(128)	utf8mb4_general_ci	
2	bundle	varchar(128)	utf8mb4_general_ci	
3	deleted	tinyint(4)		
4	entity_id	int(10)		unsigned
5	revision_id	int(10)		unsigned

Node object (nid: 1)

... (Object) Node

is_active_revision	(String, 1 characters) 1
nid	(String, 1 characters) 1
vid	(String, 1 characters) 1
type	(String, 4 characters) post
langcode	(String, 3 characters) und
title	(String, 16 characters) Your first post!
uid	(String, 1 characters) 1
status	(String, 1 characters) 1
created	(String, 10 characters) 1731468406
changed	(String, 10 characters) 1731468406
scheduled	(String, 1 characters) 0
comment	(String, 1 characters) 0
promote	(String, 1 characters) 1
sticky	(String, 1 characters) 0
tnid	(String, 1 characters) 0
translate	(String, 1 characters) 0
revision_timestamp	(String, 1 characters) 0
revision_uid	(String, 1 characters) 0
in_preview	(NULL)
log	(String, 0 characters)
comment_close_override	(String, 1 characters) 0
body	(Array, 1 element)
field_image	(Array, 0 elements)
field_tags	(Array, 0 elements)
path	(Array, 5 elements)

Database

node table

Server: localhost:8889 » Database: bdttesting » Table: node » The base table for nodes

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	D
1	nid	int(10)		unsigned	No	N
2	vid	int(10)		unsigned	Yes	N
3	type	varchar(32)	utf8mb4_general_ci		No	
4	langcode	varchar(12)	utf8mb4_general_ci		No	
5	title	varchar(255)	utf8mb4_general_ci		No	
6	uid	int(11)			No	0

field_data_field_tags

Server: localhost:8889 » Database: bdttesting » Table: field_data_field_tags

Table structure Relation view

#	Name	Type	Collation	Attributes	Null
1	entity_type	varchar(128)	utf8mb4_general_ci		No
2	bundle	varchar(128)	utf8mb4_general_ci		No
3	deleted	tinyint(4)			No
4	entity_id	int(10)		UNSIGNED	No
5	revision_id	int(10)		UNSIGNED	Yes

field_data_body table

Server: localhost:8889 » Database: bdttesting » Table: field_data_body

Table structure Relation view

#	Name	Type	Collation
1	entity_type	varchar(128)	utf8mb4_general_ci
2	bundle	varchar(128)	utf8mb4_general_ci
3	deleted	tinyint(4)	
4	entity_id	int(10)	

field_data_field_image

Server: localhost:8889 » Database: bdttesting » Table: field_data_field_image

Table structure Relation view

#	Name	Type	Collation	Attributes
1	entity_type	varchar(128)	utf8mb4_general_ci	
2	bundle	varchar(128)	utf8mb4_general_ci	
3	deleted	tinyint(4)		
4	entity_id	int(10)		UNSIGNED
5	revision_id	int(10)		UNSIGNED

Entity types provided by core in Backdrop CMS

Node

File

Comment

Taxonomy term

User

Commonalities among entities

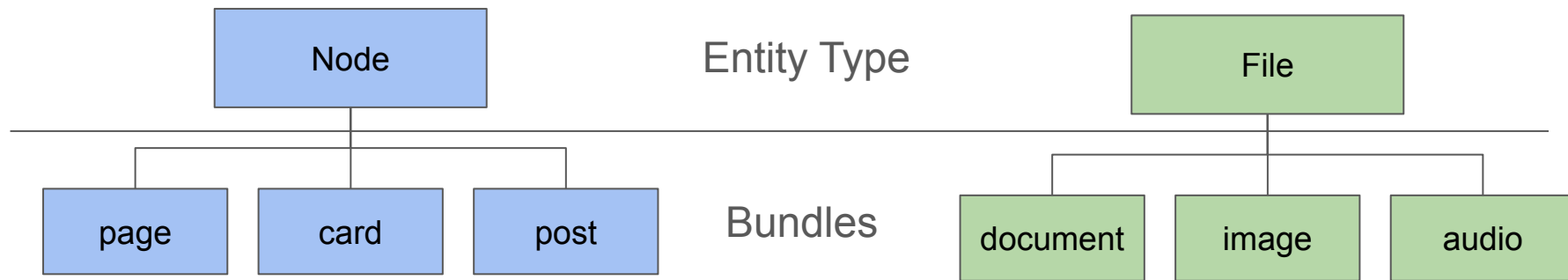
Common API to create, load, save, delete and display them

A few properties ("fields") with the same purposes:

- A property to hold its unique ID
- A property to hold its label (human-readable name)
- A property to hold its bundle

Bundles

A "bundle" is like a "subtype" of an entity type



"Bundles" receive different names in Backdrop

- For nodes, they are called "content types" (page, card, etc)
- For taxonomy terms, they are "vocabularies" (tags, etc)
- For files, they are "file types" (audio, video, document, etc)
- (Users and Comments don't have bundles)

Bundles allow us to...

Attach different fields to each bundle (e.g. `field_tags` to *post* but not *page*)

Change the display settings for each bundle

Filter Views and Layouts by bundle (e.g. create a View that displays *posts*)

Two flavors of fields

Properties

They are a "fixed" part of the entity type

Common to all bundles of an entity type

Can't be deleted/detached from the entity

Saved in the entity type table

E.g. for node entities: nid, title, vid, etc

Field API fields

Attached in the "Manage fields" tab for each bundle

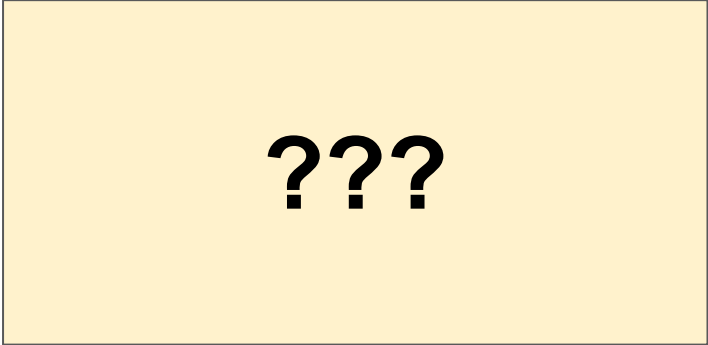
Can be deleted/detached from the bundle

Configurable per bundle

Saved in their own tables (field_data_X)

Custom entity types

Backdrop makes it easy to create new entity types



???

Contrib modules that create custom entity types

Registration

Organic groups

Ubercart

Commerce

Paragraphs

Field collections

Entity forms

Etc

Why use custom entities?

Nodes can be adapted to store custom information, but there is a lot of overhead....

Custom entities provide more targeted flexibility



How does Backdrop help you create entity types?

Common API for all entity types (CRUD)

- `entity_create()`
- `$entity->save()` [Save capability (provided by the entity controller)]
- `entity_load()`
- `entity_delete_multiple()`
- `entity_view()`

Field API provides ready-made UI to attach fields to ANY bundle of ANY fieldable entity type

What are the basic requirements for an entity type?

We need at least:

Database table

- Column to store the **unique ID**
- Column to store the **label**
- Column to store the **bundle name**

PHP object

- Property to store the **unique ID**
- Property to store the **label**
- Property to store the **bundle name**

- Extend class **Entity**
- Implement methods `id()`, `entityType()`, `label()`, `bundle()` and `uri()`

A node object

Stores the unique ID

Stores the bundle name

Stores the label

... (Object) Node

is_active_revision (String, 1 characters) 1

nid (String, 1 characters) 1

vid (String, 1 characters) 1

type (String, 4 characters) post

langcode (String, 3 characters) und

title (String, 16 characters) Your first post!

uid (String, 1 characters) 1

status (String, 1 characters) 1

created (String, 10 characters) 1731468406

changed (String, 10 characters) 1731468406

scheduled (String, 1 characters) 0

comment (String, 1 characters) 0

promote (String, 1 characters) 1

sticky (String, 1 characters) 0

tnid (String, 1 characters) 0

The node table

Stores the unique ID

Stores the bundle name

Stores the label

Server: localhost:8889 » Database: bdtesting » Table: node "The base table for nodes."

Browse Structure SQL Search Insert Export Import

Table structure Relation view

	#	Name	Type	Collation	Attributes	Null	Default	C
<input type="checkbox"/>	1	<u>nid</u>	int(10)		UNSIGNED	No	None	TI no
<input type="checkbox"/>	2	<u>vid</u>	int(10)		UNSIGNED	Yes	NULL	TI no id
<input type="checkbox"/>	3	<u>type</u>	varchar(32)	utf8mb4_general_ci		No		TI
<input type="checkbox"/>	4	<u>langcode</u>	varchar(12)	utf8mb4_general_ci		No		TI no
<input type="checkbox"/>	5	<u>title</u>	varchar(255)	utf8mb4_general_ci		No		TI al m
<input type="checkbox"/>	6	<u>uid</u>	int(11)			No	0	TI th th
<input type="checkbox"/>	7	<u>status</u>	int(11)			No	1	B th (v ac

Other "nice things" you may want

- A form to enter or edit values of the entities
- A list or View that displays all created entities of that type
- Administration form and list to configure the entity type bundles (attach fields etc)

What I'll show

Four progressive examples of a custom entity type: **student**

Examples available at <https://github.com/argiepiano/ed>



Demo 1: basic setup for a non-fieldable entity type (no bundles). Form and list

Demo 2: hard-coded bundles and Field API UI

Demo 3: dynamic bundles

Demo 4: Views and tokens integration (Entity Plus)