



Drupal 8 Kickstart

An Overview for Developers

Drupal GovCon 2015

CivicActions

Peter Sawczynec
Engineer

Drupal 8 Ecosystem

CivicActions



Drupal 8 [Symfony](#) PHP OOP [Drush](#) [Git](#) [GitHub](#)
[Markdown](#) [Composer](#) **Linux shell** [zshell](#) SSH
[Behat](#) [Gherkin](#) [PHPUnit](#) [jMeter](#)
[MySQL Workbench](#) Regex JSON **jQuery**
[AngularJS](#) [Node.js](#) Twig [Compass](#) [SASS](#) [SMACSS](#)
Guzzle Memcache [Varnish](#) CDN Service
[Jenkins](#) [Chef](#) [Splunk](#) Apache Nginx
[phpStorm](#) [Sublime](#) [NetBeans](#)

Drupal 8: Mission



- **Drupal 8:** A service returning a response of format-agnostic data structures
- Whether the request comes from a desktop browser, mobile phone, or another website the response ***data*** will be returned consistently
- How the response data gets formatted is, as much as possible, a distinct and separate set of actions

Drupal 8: Mission



- Leverage existing industry-standard technologies so that **D8** can interface with and be programmed like other globally-recognized PHP frameworks using PHP OOP concepts
- To achieve the missions **D8** is built on top of the Symfony framework components

Drupal 8: Headless Drupal



- Drupal is a service, not an HTML output provider
- Drupal provides format-agnostic data for a request
- A website's browsing visitor may not even interact with Drupal's themed output
- For example, pages can be created by JS frameworks such as Angular or Backbone and Drupal is the data/content store

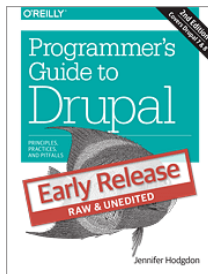
Drupal 8 Essential Reading

CivicActions



D8 API Reference: <http://api.drupal.org/api/drupal/8>

Programmer's Guide to Drupal
(2nd Edition)



Drupal 8 Configuration Management





- Check the documentation creation date to judge the timeliness and accuracy of online **D8** documentation
- Other than docs found on Drupal.org, **D8** online documentation older than October 2014 is very unlikely to be fully accurate

D8 / Symfony: Special Note

CivicActions



Drupal 8 will upgrade to Symfony 3.0 in a minor release and drop Symfony 2.x backwards compatibility

Posted by [catch](#) on *January 7, 2015*

While **Drupal 8.0.0** will likely ship using Symfony 2.7, in a subsequent minor release we will upgrade Symfony to use the 3.x branch. This will allow us to continue to get bug fixes and security releases more actively and for a longer time period.

Core, contrib and custom modules should not rely on any deprecated Symfony APIs, since these may be removed in any **Drupal 8** minor release...



Drush

Composer

YAML

PHP OOP



Drush

A command line tool for managing Drupal that provides uncountable shortcut commands

- Drush executes Drupal admin tasks 3 - 10x faster than using the admin pages
- Install drush with [Composer](#)

Drupal 8 and Drush

CivicActions



Drush can run update.php, clear cache, log you in, change user passwords, disable/enable modules, execute sql queries, manage features.

Example drush commands:

```
drush status      drush uli      drush cc all      drush updb
```

```
drush en devel -y      drush pmi devel
```

```
drush upwd --password="newsecurepasswoed" "admin"
```

```
drush sqlq 'SELECT schema_version FROM system WHERE name="views"'
```

```
drush sqlq "UPDATE system SET schema_version = 12 WHERE name='views'"
```

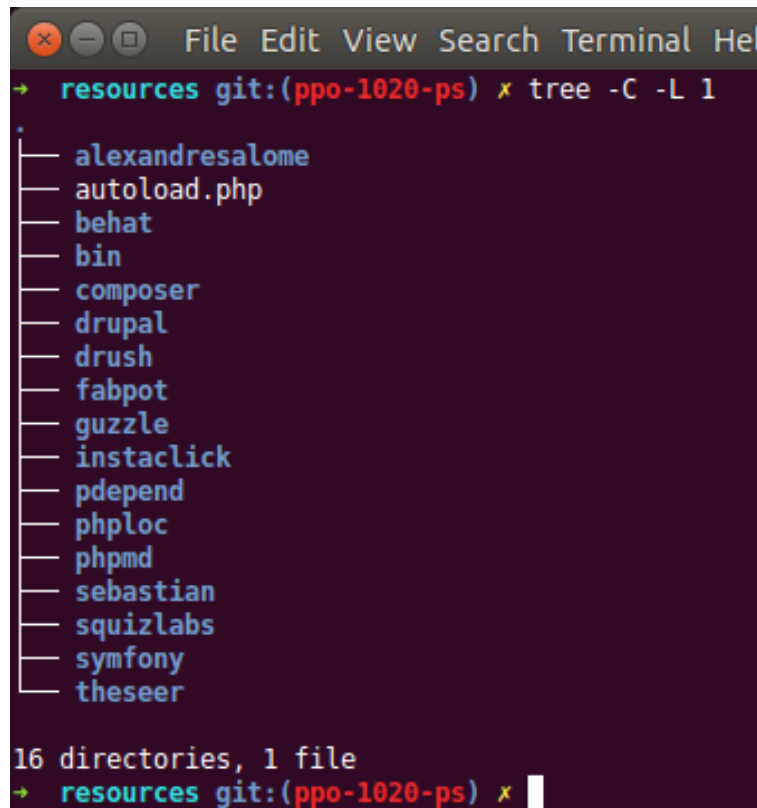


- Composer helps you declare, manage and install dependencies of PHP projects, ensuring you have the right stack everywhere
- Composer uses .json files to keep track of the versions of php libraries and other software that you might employ in your website. Then when you need with a single composer command one can download new or update all the software

Drupal 8 and Composer



- An enterprise **D8** website with a large resources directory like shown right can download and keep current all that software with two commands:
composer -install
composer -update



```
resources git:(ppo-1020-ps) x tree -C -L 1
├── alexandresalome
├── autoload.php
├── behat
├── bin
├── composer
├── drupal
├── drush
├── fabpot
├── guzzle
├── instaclick
├── pdepend
├── phploc
├── phpmd
├── sebastian
├── squizlabs
├── symfony
└── theseer

16 directories, 1 file
resources git:(ppo-1020-ps) x
```

Drupal 8 and Composer Manager

CivicActions



Composer Manager

Allows contributed modules and your own custom modules to manage the inclusion of PHP and other supporting libraries via Composer.



YAML (*.yaml files)

- A simple, clean format (similar to JSON) for storing structured data that is easier to read/write than XML
- YAML is a recursive acronym for:
"YAML Ain't Markup Language"
- All Drupal 8 configuration is created using YAML and during installation pulled from *.yaml files



- YAML is case sensitive
- YAML structure is created by using indenting with spaces. YAML does not allow the use of tabs
- Use 2 spaces for YAML indenting in Drupal

Schema Files (*.schema.yml files)

- Schema files define the expected structure and allowed elements of YAML files (like DTD for XML)

Sample YAML file: core-services.yml

```
parameters:
  session.storage.options: {}
  twig.config: {}
  renderer.config:
    required_cache_contexts: ['languages:language_interface', 'theme']
  factory.keyvalue:
    default: keyvalue.database
  factory.keyvalue.expirable:
    default: keyvalue.expirable.database
services:
  # Simple cache contexts, directly derived from the request context.
  cache_context.ip:
    class: Drupal\Core\Cache\Context\IpCacheContext
    arguments: ['@request_stack']
    tags:
      - { name: cache.context }
  cache_context.headers:
    class: Drupal\Core\Cache\Context\HeadersCacheContext
    arguments: ['@request_stack']
    tags:
      - { name: cache.context }
  cache_context.cookies:
    class: Drupal\Core\Cache\Context\CookiesCacheContext
    arguments: ['@request_stack']
    tags:
      - { name: cache.context }
  cache_context.request_format:
    class: Drupal\Core\Cache\Context\RequestFormatCacheContext
    arguments: ['@request_stack']
    tags:
      - { name: cache.context }
  cache_context.url:
    class: Drupal\Core\Cache\Context\UrlCacheContext
    arguments: ['@request_stack']
    tags:
      - { name: cache.context }
  cache_context.url.site:
    class: Drupal\Core\Cache\Context\SiteCacheContext
    arguments: ['@request_stack']
    tags:
      - { name: cache.context }
  cache_context.url.query_args:
    class: Drupal\Core\Cache\Context\QueryArgsCacheContext
    arguments: ['@request_stack']
    tags:
      - { name: cache.context }
  cache_context.url.query_args.pagers:
    class: Drupal\Core\Cache\Context\PagersCacheContext
    arguments: ['@request_stack']
    tags:
      - { name: cache.context }
```



Class

- A set of functions and properties organized in a file that offer a service
- **Controllers, Routers, Forms, and Plugins** are all major types of classes in **D8**.
- In general all functionality created for **D8**, including your custom modules, is expected to be created in class files



Interface

- A class with empty default methods that all other classes based on it must offer
- Every single method declared in an Interface will have to be implemented in the subclass. A class can implement multiple Interfaces

class MyClass implements ParentInterface

class MyClass implements SomeInterface, OtherInterface



Abstract Class

- A class with default abstract methods that classes based on it must offer
- Only Abstract methods have to be implemented by the subclass. A class can only implement one abstract class at a time.

class MyClass extends ParentClass

class MyClass extends ParentClass implements
SomeInterface, OtherInterface



Trait

- A set of php functions in one file that supply a useful set of related functions



Dependency Injection

- Initiating a class, but telling the class what you want it to use to work.
- See: [What is Dependency Injection?](#)
by Fabien Potencier

Sample class file

```
<?php

/**
 * @file
 * Contains \Drupal\tracking_inject\EventSubscriber\TrackingInjectEventSubscriber.
 */

namespace Drupal\tracking_inject\EventSubscriber;

use Symfony\Component\HttpFoundation\RedirectResponse;
use Symfony\Component\HttpKernel\KernelEvents;
use Symfony\Component\HttpKernel\Event\GetResponseEvent;
use Symfony\Component\EventDispatcher\EventSubscriberInterface;

class TrackingInjectEventSubscriber implements EventSubscriberInterface {

    /**
     * {@inheritdoc}
     */
    static public function getSubscribedEvents() {
        $events[KernelEvents::REQUEST][] = array('initiateTracking');
        return $events;
    }

    /**
     * Stub function.
     */
    public function initiateTracking(GetResponseEvent $event) {
        // Redirect example:
        // print "<br /> init event subscriber tracking REQUEST event<br />";
        // if ($event->getRequest()->query->get('redirect-me')) {
        //   $event->setResponse(new RedirectResponse('http://example.com/'));
        // }
    }
}
```



Services

Something a class offers, e.g. “map this node’s location by address, returns latitude and longitude”

Plugins

In **D8** plugins are used, for example, to make Blocks, in that your Block and what describes it, builds it, and controls access to the Block is found in a special kind of class called a plugin

D8 and Code Comments



Comments and special comments called Annotations are more important than ever in **D8**

Properly formatted comments are used by Drupal 8 to create documentation, identify tests, and in some cases for Drupal 8 to make discovery of services and other plugin functionality

Links: [Drupal Comments](#) [Annotations in Drupal](#)

Chaining (or Method Chaining)



Method Chaining (used by jQuery, PHP, Drupal)

Allows us to run a series of methods, one after the other (or in a chain), because each method in the chain after it executes returns a full object with the the changes applied to it

jQuery Method Chaining example:

```
$("#edit-button").css("color","red").slideUp(2000).slideDown(2000);
```

Chaining (or Method Chaining)



jQuery method chaining (multiline):

```
$("#p1").css("color", "red")  
    .slideUp(2000)  
    .slideDown(2000);
```

D8 Example (multiline):

```
db_update('example')  
    ->condition('id', $id)  
    ->fields(array('field2' => 10))  
    ->execute();
```

Above using the Database Abstraction Layer
where db_update returns an UpdateQuery object



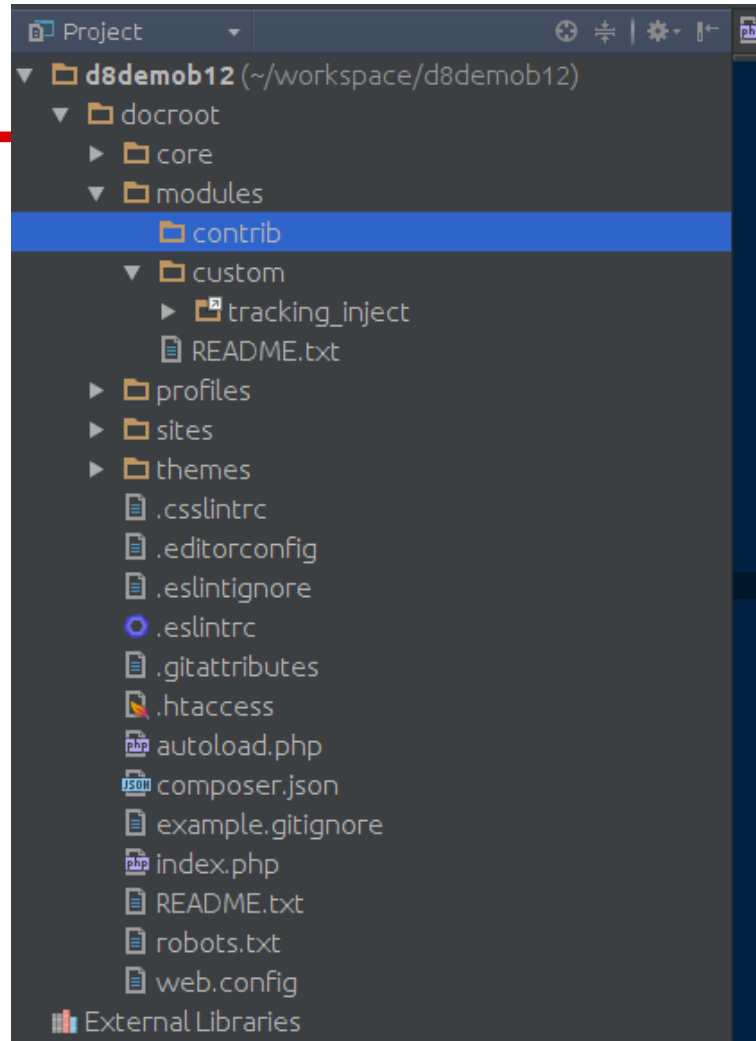
Drupal 8

D8 Top-level Directory Structure



- **/core** Core modules and files provided by **D8**
- **/libraries** Common 3rd party libraries, eg. a wysiwyg editor
- **/modules** Contrib and custom modules using sub-dirs **contrib** and **custom** (used to be sites/all/modules)
- **/profiles** Contrib and custom profiles
- **/sites** Site specific modules, themes and files. Including files uploaded by users, such as images.
The site's YAML configuration files, **active** and **staged**
- **/themes** Contrib themes, custom themes and subthemes

D8 top-level
directory structure.
Custom modules
reside in:
`/modules/custom`



D8 Core Directory Structure



Inside /core directory:

- **/core/assets** Various external libraries used by Core. jQuery, underscore, modernizer etc
- **/core/config** Configuration YAML files
- **/core/includes** Functionality that is too low level to be modular. Such as the module system itself
- **/core/lib** Drupal Core classes
- **/core/misc** Frontend libraries that Drupal Core depends on. (jQuery, modernizer, etc)
- **/core/modules** Drupal Core modules
- **/core/profiles** Drupal Core profiles. Empty at the time of writing
- **/core/scripts** Various CLI scripts, mostly used by developers
- **/core/tests** Drupal Core tests
- **/core/themes** Drupal Core themes
- **/core/vendor** Backend libraries that Drupal Core depends on. (Symfony, Twig, etc) <http://drupal.stackexchange.com/questions/84811/what-are-all-the-directories-for-in-the-new-drupal-8-structure>

Drupal 8 Module Structure



File Edit View Search Terminal Help

→ tracking_inject git:(8.x-1.x) tree -C -L 23

```
.
├── config
│   └── install
│       └── tracking_inject.settings.yml
├── README.txt
├── src
│   ├── EventSubscriber
│   │   └── TrackingInjectEventSubscriber.php
│   ├── Form
│   │   ├── TrackingInjectAdd.php
│   │   ├── TrackingInjectAdmin.php
│   │   ├── TrackingInjectDelete.php
│   │   ├── TrackingInjectEdit.php
│   │   └── TrackingInjectSettings.php
│   ├── TrackingInjectInterface.php
│   ├── TrackingInjectManagerInterface.php
│   ├── TrackingInjectManager.php
│   └── TrackingInject.php
├── tracking_inject.info.yml
├── tracking_inject.install
├── tracking_inject.links.action.yml
├── tracking_inject.links.menu.yml
├── tracking_inject.module
├── tracking_inject.permissions.yml
├── tracking_inject.routing.yml
└── tracking_inject.services.yml
```

5 directories, 20 files

→ tracking_inject git:(8.x-1.x) █



1. Bootstrap configuration

- Read the settings.php file, generate some other settings dynamically, and store them both in global variables and the Drupal\Component\Utility\Settings singleton object
- Start the **class loader**, takes care of loading classes
- Set the Drupal error handle.
- Detect if Drupal is actually installed. If it is not, redirect to the installer script



2. Create the Drupal kernel
3. Initialize the service container
(either from cache or from rebuild)
4. Add the container to the Drupal static class
5. Attempt to serve page from static page cache
6. Load all variables
7. Load other necessary include files



8. Register stream wrappers
(public://, private://, temp:// and custom wrappers)
9. Create the HTTP Request object
(using the Symfony HttpFoundation component)
10. Let DrupalKernel handle it and return response
11. Send response
12. Terminate request
(modules can act upon this event)

D8 YAML (*.yaml) Files



Replaces .info files and used for Configuration, Routes, Menu Links, and Services

Pronounced: “YA-MUL” is short for: “YAML Ain’t Markup Language”

D8

D7

<code><module_name>.info.yaml</code>	<code><--></code>	<code><module_name>.info</code> file
<code><module_name>.routing.yaml</code>	<code><--></code>	hook_menu for page paths
<code><module_name>.links.menu.yaml</code>	<code><--></code>	hook_menu for entries on admin menu
<code><module_name>.permissions.yaml</code>	<code><--></code>	hook_permissions
<code><module_name>.services.yaml</code>	<code><--></code>	Describes a class: machine name, class path, mandatory arguments

Services .yaml File



services:

tracking_inject.manager:

class: Drupal\tracking_inject\TrackingInjectManager

arguments: ['@database']

tags:

- { name: backend_overridable }

tracking_inject.response_event:

class: Drupal\tracking_inject\EventSubscriber\TrackingInjectEventSubscriber

tags:

- { name: event_subscriber }

tracking_inject.injections:

class: Drupal\tracking_inject\TrackingInject

arguments: ['@config.factory']

D8 Hooks to Events

CivicActions



Drupal 8 Hooks

Request Event Example



D8 uses Symfony kernel and events. Kernel events available in **D8** are as follows:

- **KernelEvents::CONTROLLER**

CONTROLLER event occurs once a controller was found for handling a request

- **KernelEvents::EXCEPTION**

EXCEPTION event occurs when an uncaught exception appears

- **KernelEvents::FINISH_REQUEST**

FINISH_REQUEST event occurs when a response was generated for a request



- **KernelEvents::REQUEST**

REQUEST event occurs at the very beginning of request dispatching

- **KernelEvents::RESPONSE**

RESPONSE event occurs once a response was created for replying to a request

- **KernelEvents::TERMINATE**

TERMINATE event occurs once a response was sent

- **KernelEvents::VIEW**

VIEW event occurs when the return value of a controller is not a Response instance

Services in Drupal 8



Core functionality in **D8** such as current user info, current path, node info, is logged in, module exists... these are all called services

- Core services in **D8** are declared in: `/core/core.services.yml`
- Services can be accessed throughout **D8** via the global Drupal namespace `\Drupal`

Services in Drupal 8



Examples of using **D8** core services:

```
\Drupal::moduleHandler()->moduleExists('content_translation');
```

```
$account = \Drupal::currentUser();
```

```
$config = \Drupal::config('some_module.settings');
```

Services in Drupal 8



Examples of using **D8** core services:

```
$id = $config->get('domain_id');
```

```
$request = \Drupal::request();
```

```
$exception = $request->attributes->get('exception');
```

```
$status = $exception->getStatusCode();
```

D8: variable_get, variable_set



- Replaced by using a **D8** core service...
(also understand states, settings and overrides)
- **Config** is the global **D8** configuration object and holds the changeable site or module configurations, e.g.:
`\Drupal::config('system.site') ->get('page.front');`

D8: variable_get, variable_set



- Getting a variable:

```
\Drupal::config('module_name.settings')->get('var_name');  
\Drupal::config('system.site')->get('page.front');
```

- Setting a variable:

```
\Drupal::configFactory()->getEditable('module_name.settings')  
->set('var_name', 'some_value')->save;
```

- Unsetting a variable value:

```
$config = \Drupal::config('system.performance');  
$config->clear('cache.page.max_age')->save();
```

D8: Config vs Settings



Settings is the global D8 settings object and holds site settings like the database settings that are in settings.php.

- A get settings example:

```
use\Drupal\Core\Site\Settings
```

```
$theme = Settings::get()->('maintenance_theme', 'bartik');
```

- A set settings in settings.php example:

```
$settings['maintenance_theme'] = 'my_custom_theme';
```



Routing System in Drupal 8

A route is a path which is defined for Drupal to return some sort of content on.

For example, the default front page, '/node' is a route. When Drupal receives a request, it tries to match the requested path to a route it knows about. If the route is found, then the route's definition is used to return content. Otherwise, Drupal returns a 404.

Drupal's routing system works with the Symfony HTTP Kernel.

The routing system is responsible for matching paths to controllers, and you define those relations in routes. You can pass on additional information to your controllers in the route. Access checking is integrated as well.

D8 Routes and Controllers



example.routing.yml

example.content:

path: '/example'

defaults:

_controller: '\Drupal\example\Controller\ExampleController::content'

_title: 'Example Route Response '

requirements:

_permission: 'access content'

Drupal 8 Blocks Plugin

CivicActions



Block plugin creation overview.



Configuration Management Initiative

- Saving **D8** global and module settings into and reading settings from *.yml files and also special **D8** CMI tables in the database. Links:

[Configuration Mangement Initiative](#)

[Principles of Configuration Management - Pt 1](#)

[Principles of Configuration Management - Pt 2](#)

[D8 CMI critical analysis](#)



Drupal Configuration Inspector

- A module that exposes the configuration settings in use throughout your site using nice visual organization. Links:

[Configuration Inspector Module](#)



Drupal Console

- An app that you can use to quickly make the scaffold of a **Drupal 8** module and a **Drupal 8** service within that module. Links:

[Drupal Console on Github](#)

[Install Drupal Console](#)

[Drupal Console Docs](#)

[Available Commands](#)



Partial list of **D8** Console commands:

<code>generate:controller</code>	Generate and register a controller
<code>generate:entity:config</code>	Generate a new "EntityConfig"
<code>generate:entity:content</code>	Generate a new "EntityContent"
<code>generate:form:config</code>	Generate a new "ConfigFormBase"
<code>generate:module</code>	Generate a module.
<code>generate:plugin:block</code>	Generate plugin block.
<code>generate:plugin:imageeffect</code>	Generate image effect plugin.
<code>generate:service</code>	Generate service

[See all Console commands](#)



Module Upgrader

- A **D8** module that can analyze your Drupal 7 module for needed changes and/or attempt the actual upgrade. Links:

[About Module Upgrader](#)

[Download D8 'Module Upgrader' Module](#)

Drupal 8 Training/Resources

CivicActions



[Drupalize.me](https://drupalize.me)

[Buildamodule.com](https://buildamodule.com)

[Safaribooksonline.com](https://safaribooksonline.com)

[http://youtube.com/user/DrupalAssociation](https://youtube.com/user/DrupalAssociation)