



OOP is Nothing To Be Afraid Of

Geoff Appleby



Geoff Appleby

drupal.org/u/gapple
twitter.com/@gappleca

Persistent Login
gacsp: Google Analytics
DrupalReleaseDate.com



myplanet



What is OOP?

data  code

Why OOP?

Why OOP?

- Structured Data
- Code Re-use
- Testing
- Performance

Class → Object

Class Definition

```
class MyClass {  
  
    public $myProperty;  
  
    public function __construct($initialValue) {  
        $this->myProperty = $initialValue  
    }  
  
    public function getMyProperty() {  
        return $this->myProperty;  
    }  
}
```

Instantiation

```
$node = (stdClass) array(  
    'title' => 'My Node',  
    'created' => 1462855238,  
);
```

```
$node = node_load(35);
```

Instantiation

```
$today = new DateTime('now');  
  
$halloween = DateTime::createFromFormat('Y-m-d', '2016-10-31');  
  
$canadaDay = $nationalHolidayFactory->create('CAN');
```

Calling Methods

```
php > $today = new DateTime('now');  
php > echo $today->format('Y-m-d h:i:s');  
2016-05-11 03:45:00  
php > $today->setTime(12, 0, 0);  
php > echo $today->format('Y-m-d h:i:s');  
2016-05-11 12:00:00
```

Encapsulation

Drupal 7 Visibility

```
/**  
 * This function should only be used by mymodule.  
 */  
function _mymodule_private_function() {  
  
}
```

Object Visibility

```
class MyClass {  
    private $myProperty;  
  
    public function __construct($initialValue) {  
        $this->myProperty = $initialValue  
    }  
  
    public function getMyProperty() {  
        return $this->myProperty;  
    }  
}
```

Object Visibility

```
php > $myObject = new MyClass('property value');  
php > echo $myObject->myProperty;  
PHP Warning: Uncaught Error: Cannot access private property  
MyClass::$property in php shell code:1  
Stack trace:  
#0 {main}  
    thrown in php shell code on line 1
```

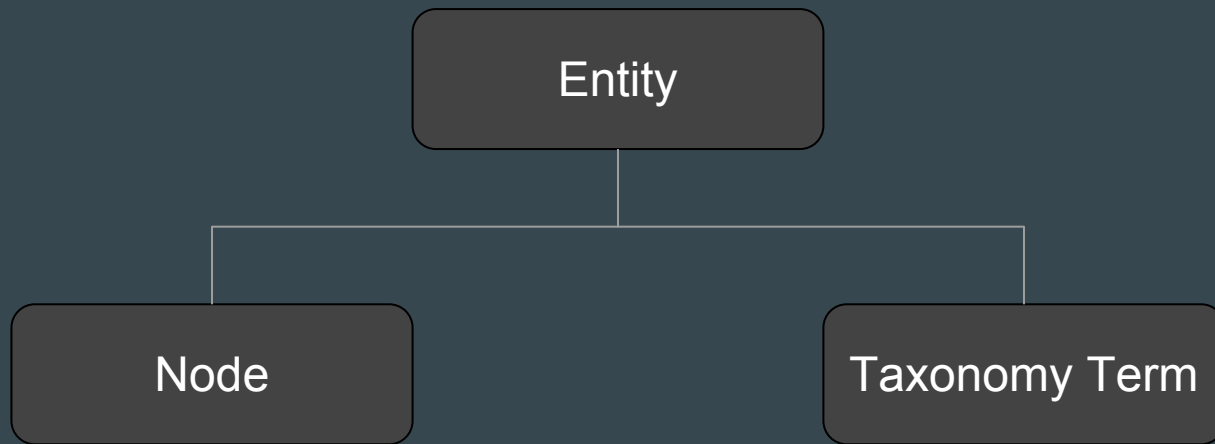
Object Visibility

```
php > $myObject = new MyClass('property value');  
php > echo $myObject->getMyProperty();  
property value
```

Visibility

- Public
- Protected
- Private

Inheritance



Inheritance

```
class MyBaseClass {  
    public function myBaseMethod() {  
  
    }  
}  
  
class MyChildClass extends MyBaseClass {  
    public function myChildMethod() {  
        $parentBaseValue = $this->myBaseMethod();  
    }  
}
```

Overriding Methods

```
class MyBaseClass {  
    public function myOverridableMethod() {  
  
    }  
}  
  
class MyChildClass extends MyBaseClass {  
    public function myOverridableMethod() {  
        $parentValue = parent::myOverridableMethod();  
    }  
}
```

Interfaces

Drupal 7 Interfaces - Module Hooks

```
/**  
 * Implements hook_menu().  
 */  
function mymodule_menu() {  
  
}
```

Interfaces

```
interface MyInterface {  
    public function myMethod();  
}  
  
class MyClass implements MyInterface {  
  
    public function myMethod() {  
  
    }  
}
```

Polymorphism

Namespaces

Drupal 7 Namespaces

```
/**
 * Implements hook_menu().
 */
function mymodule_menu() {

}

function mymodule_main_page_handler() {

}
```

PHP Namespaces

```
namespace Drupal\mymodule;  
  
class MyClass {  
  
}
```

PHP Namespaces

```
namespace Drupal\mymodule;

class MyClass {

    public function myMethod() {
        $structure = new \Drupal\othermodule\Structures\BaseStructure();
    }
}
```

PHP Namespaces

```
namespace Drupal\mymodule;

use Drupal\othermodule\Structures\BaseStructure;

class MyClass {

    public function myMethod() {
        $structure = new BaseStructure();
    }
}
```

PHP Namespaces

```
namespace Drupal\mymodule;

class MyClass {

    public function myMethod() {
        $today = new \DateTime('now');
    }
}
```

Autoloading

Drupal 7 File Loading

mymodule.module

```
require_once('includes/mymodule.util.inc');
```

```
function mymodule_menu() {  
  $items['mymodule/abc'] = array(  
    'page callback' => 'mymodule_abc_view',  
    'file' => 'mymodule.pages.inc',  
    'file path' => drupal_get_path('module', 'mymodule') . '/includes',  
  );  
  return $items;  
}
```

Drupal 7 File Loading

includes/mymodule.pages.inc

```
function mymodule_abc_view() {  
  module_load_include('inc', 'othermodule', 'includes/othermodule.util');  
  
  // Use functions from included file...  
}
```

Autoloading

`\Drupal\mymodule\Form\SettingsForm`

`modules/mymodule/src/Form/SettingsForm.php`

The diagram illustrates the autoloading process. It shows a class name `\Drupal\mymodule\Form\SettingsForm` at the top and a file path `modules/mymodule/src/Form/SettingsForm.php` at the bottom. Three arrows connect the segments of the class name to the file path: a yellow arrow from `mymodule` to `modules/mymodule`, a teal arrow from `Form` to `src/Form`, and a yellow arrow from `SettingsForm` to `SettingsForm.php`.

Autoloading

\Drupal\ **node** \Entity \Node



core/modules/**node**/src/**Entity**/**Node.php**

Services

Services Container

```
$sessionService = \Drupal::service('session');  
  
$siteConfig = \Drupal::config('system.site');  
$siteName = $siteConfig->get('name');
```

Dependency Injection

Dependency Injection

```
function my_function() {  
    $db = Database::getConnection();  
  
    $db->select('SELECT id, title FROM node');  
}
```

```
function my_function($db) {  
    $db->select('SELECT id, title FROM node');  
}
```

Dependency Injection

```
class MyItem {  
  
    private $db;  
  
    public function __construct($db) {  
        $this->db = $db;  
    }  
  
    public function myMethod() {  
        $this->db->select('SELECT id, name FROM items');  
    }  
}
```

Page Controller

core/lib/Drupal/Core/Controller/ControllerBase.php

```
namespace Drupal\Core\Controller;

abstract class ControllerBase implements ContainerInjectionInterface {

    public static function create(ContainerInterface $container) {
        return new static();
    }
}
```

Page Controller

modules/mymodule/src/Controller/MyController.php

```
namespace Drupal\mymodule\Controller;

class MyController extends ControllerBase {

    private $database;

    public function __construct($database) {
        $this->database = $database;
    }

    public static function create(ContainerInterface $container) {
        return new static($container->database());
    }
}
```

Learn More - Other Sessions

- [Drupal 8 Kickstart](#)
- [Altering, Extending, and Enhancing Drupal 8](#)
- [Dependency Injection in Drupal 8](#)

Helpful tools

- Drupal Console code generator
<https://drupalconsole.com/>

Thank you