

The background features a dark navy blue field with several overlapping, semi-transparent geometric shapes. On the left, there are shapes in shades of green, cyan, orange, red, and light blue. On the right, there are shapes in shades of cyan, lime green, purple, and red. The shapes are layered, creating a sense of depth and movement.

UNIT TESTING IN DRUPAL



HOWDY!

I am Mateu

I am here because I love quality code.

You can find me at @e0ipso



HELLO!

My name is Christian

... and I assert true.

You can find me at @penyaskito



1. **WHAT TO TEST**

Types of tests and when to choose
one or the other

Types of testing within Drupal

Unit Test

Kernel Test

Simple Test

JavaScript Test

When to choose one or the other?

Unit test

- Testing a class

Kernel test

- API tests
- Integration tests

Simple test

- Testing output (HTML)

JavaScript Tests

- Testing behavior

When to choose one or the other?

	Unit	Kernel	Simple
Complexity	High	Middle	Low
Speed	High	Middle	Low
Stability*	Low	Middle	High
Reliability	High	High	Middle

* During the dev process

What is **unit** testing?

Units of code

Take *a method* in a class that produces an output and/or changes the state.

Testing

Feed that method with several sets of inputs, and *assert* that the output is what you expect and the state is what you expect.

Testing procedural code

- › Cannot be unit tested.
- › It can be tested using SimpleTest.
- › It poisons OO code when called from a class.

Use as little procedural code as possible!

What about hooks?

- › Declare a service and include the logic there
- › Inject dependencies to your service
- › Test your service
- › The hook only calls your service

DEPENDENCY INJECTION



Do not use `new` in your code.
Pass in all the objects to your
class constructor.

```
my_module.module.php UNREGISTERED
OPEN FILES
x my_module.module.php
x my_module.services.yml
x EntityTypeManager.php
1 <?php
2
3 use \Drupal\Core\Entity\EntityInterface;
4
5 /**
6  * Implements hook_entity_save().
7  */
8 function my_module_entity_create(EntityInterface $entity) {
9     \Drupal::service('my_module.entity_manager')
10     ->postCreateAlter($entity);
11 }
12
```

Line 12, Column 1 Spaces: 2 PHP

Procedural hook
**DO NOT INCLUDE
ANY LOGIC!**

```
my_module.services.yml
UNREGISTERED

OPEN FILES
x my_module.module.php
x my_module.services.yml
x EntityTypeManager.php

1 services:
2   my_module.entity_manager:
3     class: Drupal\my_module\EntityManager
4     arguments: [
5       '@entity_type.manager',
6       '@logger.channel.default'
7     ]
8

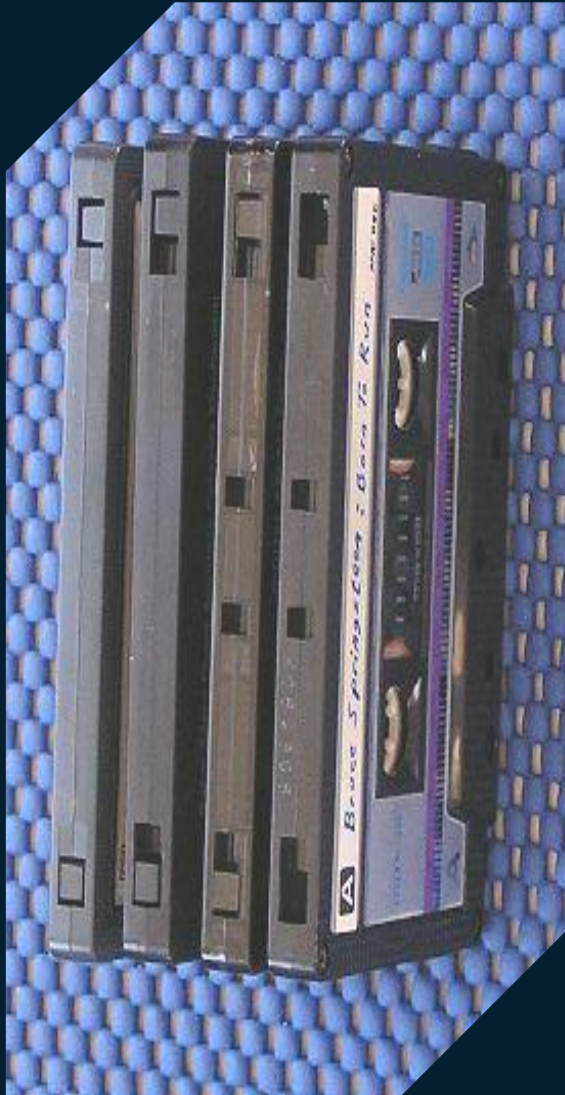
Line 8, Column 1
Spaces: 2
YAML
```

Service declaration
**INJECTS YOUR
DEPENDENCIES**

```
EntityTypeManager.php UNREGISTERED
my_module.module.php x my_module.services.yml x EntityTypeManager.php x
OPEN FILES
x my_module.module.php
x my_module.services.yml
x EntityTypeManager.php

1 <?php
2
3 namespace Drupal\my_module;
4
5 class EntityManager {
6
7     public function __construct(
8         EntityTypeManagerInterface $entity_type_manager,
9         LoggerChannelInterface $logger
10    ) {
11        $this->entityTypeManager = $entity_type_manager;
12        $this->logger = $logger;
13    }
14
15    public function postCreateAlter($entity) {
16        // This method is now testable!
17    }
18
19 }
20
```

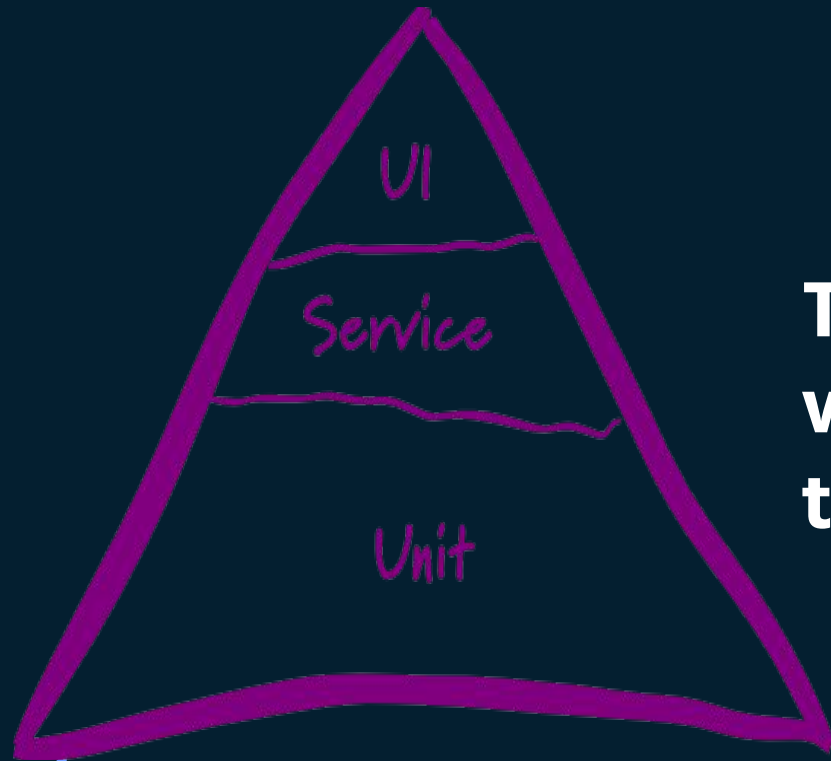
Testable Service
**THIS CLASS HAS
YOUR LOGIC!**



Inject your dependencies to **replace** them


During testing the injected objects **can** be replaced by simplified substitutes called **stubs**.

We'll see how in a moment.



Testing **pyramid** wants more unit tests

According to [Mike Cohn](#) End to end tests are less reliable and more brittle.



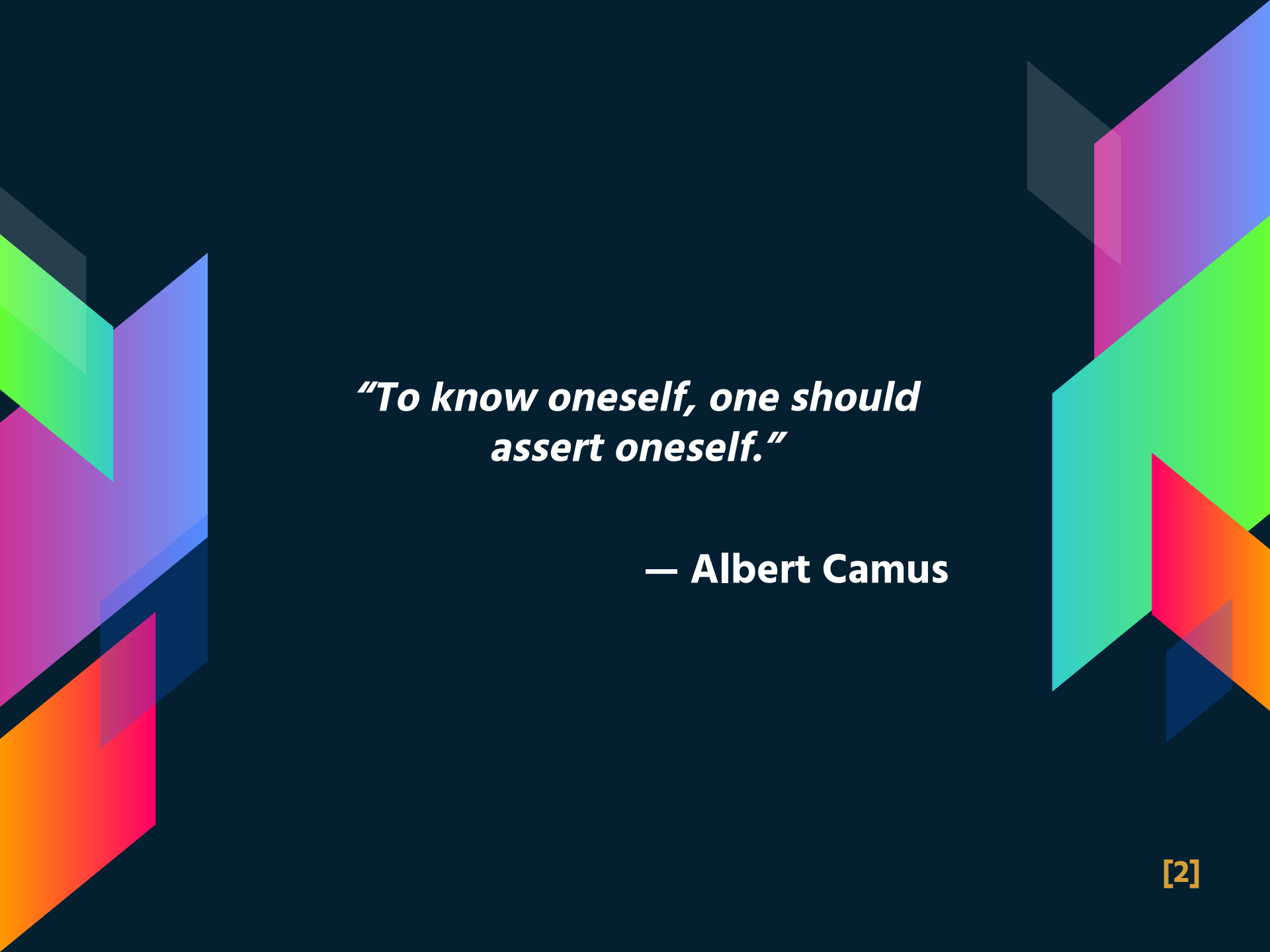
“If you get a failure in a high level test, not just do you have a bug in your functional code, you also have a missing unit test”

— Martin Fowler



2. **HOW TO TEST**

Common techniques and tools for
Unit and Kernel testing in Drupal



***“To know oneself, one should
assert oneself.”***

— Albert Camus

- ▶ PageCache
- ▶ ParamConverter
- ▶ Password
- ▶ Path
- ▶ PathProcessor
- ▼ Plugin
 - ▶ Context
 - ▼ Discovery
 - ▶ Fixtures
 - ContainerDerivativeDiscoveryDec
 - DerivativeDiscoveryDecoratorTes
 - HookDiscoveryTest.php
 - TestContainerDerivativeDiscover
 - TestDerivativeDiscovery.php
 - TestDerivativeDiscoveryWithObje
 - YamlDirectoryDiscoveryTest.php
 - YamlDiscoveryDecoratorTest.php
 - YamlDiscoveryTest.php
 - ▶ Fixtures
 - CategorizingPluginManagerTraitTest
 - ContextHandlerTest.php
 - DefaultLazyPluginCollectionTest.php
 - DefaultPluginManagerTest.php
 - DefaultSingleLazyPluginCollectionTe

```

*
* @see \Drupal\Core\Plugin\Discovery::getDefinition()
*/
public function testGetDefinition() {
    $this->moduleHandler->expects($this->exactly(4))
        ->method('getImplementations')
        ->with('test_plugin')
        ->will($this->returnValue(array('hook_discovery_test', 'hook_discovery_test2')));

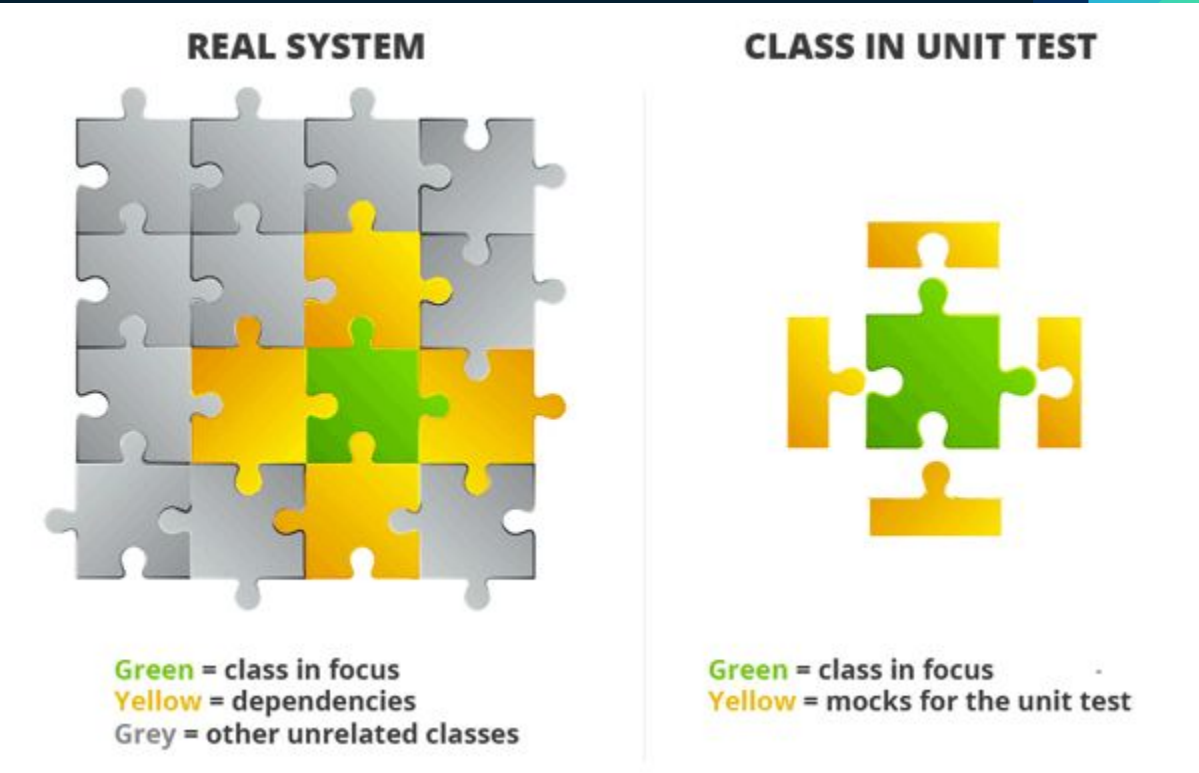
    $this->moduleHandler->expects($this->any())
        ->method('invoke')
        ->will($this->returnValueMap(array(
            array('hook_discovery_test', 'test_plugin', array(), hook_discovery_test_test_plugin(),
                st_plugin()),
        ));

    $this->assertArrayEquals(expected : array, actual : array, [mes... void, FALSE));
    $this->assertArrayHasKey(key, array : array|\ArrayAccess, [mess... void in\plugin_
    $this->assertArrayNotHasKey(key, array : array|\ArrayAccess, [m... void );
    $this->assertArraySubset(subset : array|\ArrayAccess, array : a... void test\fruit\Apple');
    $this->assertAttributeContains(needle, haystackAttributeName : ... void
    $this->assertAttributeContainsOnly(type : string, haystackAttri... void
    $this->assertAttributeCount(expectedCount : int, haystackAttrib... void in\plugin_
    $this->assertAttributeEmpty(haystackAttributeName : string, hay... void );
    $this->assertAttributeEquals(expected, actualAttributeName : st... void test\fruit\Orange');
    $this->assertAttributeGreaterThan(expected, actualAttributeName... void in\plugin_
    $this->assertAttributeGreaterThanOrEqual(expected, actualAttrib... void π');
    $this->assert
    }

```

assert methods
 True/false, equals,
 same, null, array ops...

MOCKING OBJECTS



We care about interactions

PHPUnit mocks

Basic stubs:

```
$mock  
->expects($this->any())  
->method('get')  
->with('param1', 'param2')  
->willReturnValue('some return value');
```

Multiple calls:

```
$mock  
->expects($this->any())  
->method('get')  
->willReturnMap([  
    ['key1', 'return value 1'],  
    ['key2', 'return value 2'],  
]);
```

Prophecy mocks

Basic stubs:

```
$prophecy  
->get('param1', 'param2')  
->willReturn('some return value');
```

Multiple calls:

```
$prophecy  
->get('key1')  
->willReturn('return value 1');  
$prophecy  
->get('key2')  
->willReturn('return value 2');
```

MOCKING WITH
PHPUnit vs Prophecy

Data Providers

- tests
 - Drupal
 - FunctionalJavascriptTests
 - KernelTests
 - Tests
 - Component
 - Core
 - Access
 - Ajax
 - Annotation
 - Asset
 - Authentication
 - Batch
 - Block
 - Breadcrumb
 - Cache
 - Context
 - BackendChainImplementationUnitTe
 - CacheableMetadataTest.php
 - CacheCollectorHelper.php
 - CacheCollectorTest.php
 - CacheFactoryTest.php
 - CacheTagsInvalidatorTest.php
 - CacheTest.php
 - ChainedFastBackendTest.php

```
\Drupal\Tests\Core\Cache\CacheTest mergeTagsProvider
    $this->assertNull(Cache::validateTags($tags));
}

/**
 * Provides a list of pairs of cache tags arrays to be merged.
 *
 * @return array
 */
public function mergeTagsProvider() {
    return [
        [[], [], []],
        [['bar'], ['foo'], ['bar', 'foo']],
        [['foo'], ['bar'], ['bar', 'foo']],
        [['foo'], ['bar', 'foo'], ['bar', 'foo']],
        [['foo'], ['foo', 'bar'], ['bar', 'foo']],
        [['bar', 'foo'], ['foo', 'bar'], ['bar', 'foo']],
        [['foo', 'bar'], ['foo', 'bar'], ['bar', 'foo']],
    ];
}

/**
 * @covers ::mergeTags
 *
 * @dataProvider mergeTagsProvider
 */
public function testMergeTags(array $a, array $b, array $expected) {
    $this->assertEquals($expected, Cache::mergeTags($a, $b));
}
```

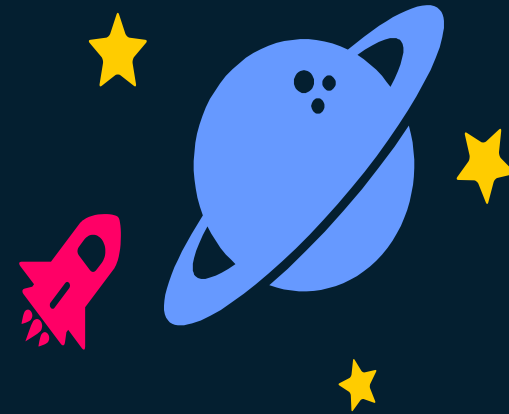
Refactor your tests :-)

Trick: Testing code that deals with procedural code



KERNEL TESTING

Lies in the middle of Unit and Simple
testing

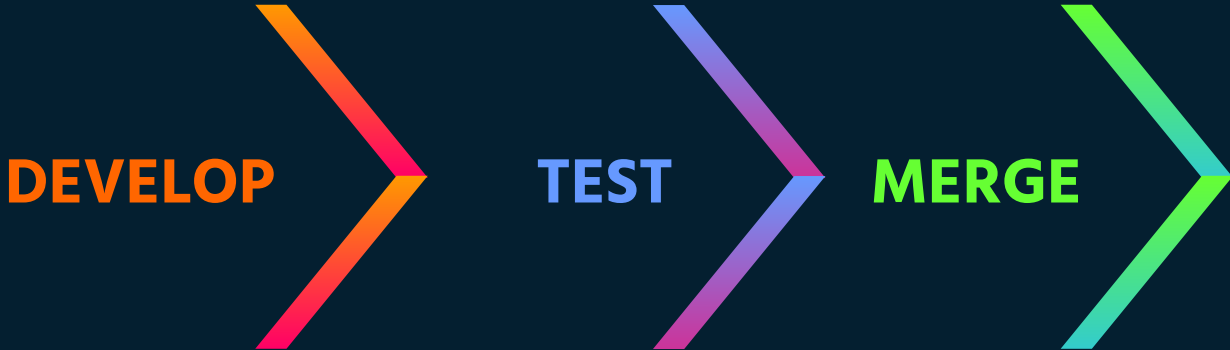




3. WHEN TO RUN YOUR TESTS

What good are tests if you don't run them

Running tests consistently



Running tests consistently



TDD IS NOW EASY IN DRUPAL!

Have a CI tool?
RUN YOUR TESTS THERE

Pre commit

Lint your code.

Pre push

Run your tests.

Pull request

Check coverage



Environment	7.x-3.x-dev	7.x-4.x-dev	8.x-1.x-dev	7.x-7.x-dev
PHP 5.3 & MySQL 5.5	<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>
PHP 5.4 & MySQL 5.5	<input type="checkbox"/>	<input type="checkbox"/>		<input type="checkbox"/>
PHP 5.5 & MySQL 5.5	<input type="checkbox"/>	<input type="checkbox"/>	Issues & on commit <input type="checkbox"/>	<input type="checkbox"/>
PHP 5.6 & MySQL 5.5	<input type="checkbox"/>	<input type="checkbox"/>	Issues & on commit <input type="checkbox"/>	<input type="checkbox"/>
PHP 7 & MySQL 5.5			Issues & on commit <input type="checkbox"/>	
PHP 5.5 & SQLite 3.8			<input type="checkbox"/>	
PHP 7 & SQLite 3.8			<input type="checkbox"/>	
PHP 5.5 & PostgreSQL 9.1			<input type="checkbox"/>	
PHP 7 & PostgreSQL 9.1			<input type="checkbox"/>	

Branch daily

Run tests for the branch once a day.

Branch on commit

Run tests for the branch on commit.

Issues & on commit

Run tests for all of the branch's issues, and on branch commit. Users may run other enabled tests.

Save

Current results

8.x-1.x-dev

PHP 5.5 & MySQL 5.5, D8.2 213 pass PHP 5.6 & MySQL 5.5, D8.2 213 pass PHP 7 & MySQL 5.5, D8.2 213 pass [Add test](#)

```
53 - mysql -e 'create database rules'
54 # Export database variable for kernel tests.
55 - export SIMPLETEST_DB=mysql://root:@127.0.0.1/rules
56 # Download Drupal 8 core from the Github mirror because it is faster.
57 - travis_retry git clone --branch $DRUPAL_CORE --depth 1 https://github.com/drupal/drupal.git
58 - cd drupal
59 # Temporary core hack to get debug information when random test fails occur.
60 # See https://www.drupal.org/node/2659954
61 - wget -q -O - https://www.drupal.org/files/issues/core-debug_0.txt | patch -p1
62
63 # Reference rules in build site.
64 - ln -s $TESTDIR modules/rules
65
66 # Run composer install for Drupal 8.1 and up. We need an up-to-date composer
67 # when installing Drupal 8.1.
68 - if [ "$DRUPAL_CORE" != "8.0.x" ]; then travis_retry composer self-update && travis_retry composer install; fi
69
70 # Start a web server on port 8888, run in the background.
71 - php -S localhost:8888 &
72
73 # Export web server URL for browser tests.
74 - export SIMPLETEST_BASE_URL=http://localhost:8888
75
76 # Install PHPCS to check for Drupal coding standards.
77 - travis_retry composer global require drupal/coder
78 - ~/.composer/vendor/bin/phpcs --config-set installed_paths ~/.composer/vendor/drupal/coder/coder_sniffer
79
80 script:
81 # Run the PHPUnit tests which also include the kernel tests.
82 - ./vendor/bin/phpunit -c ./core/phpunit.xml.dist ./modules/rules
83 # Check for coding standards violations
84 - cd modules/rules && ~/.composer/vendor/bin/phpcs
```

[Back to Project](#)

[Status](#)

[Changes](#)

[Console Output](#)

[View Build Information](#)

[History](#)

[Parameters](#)

[Test Result](#)

[Previous Build](#)

[Next Build](#)

Test Result : Lingotek

0 failures (±0)

213 tests (±0)

[Took 0 ms.](#)

All Tests

Class	Duration	Fail	(diff)	Skip	(diff)	Pass	(diff)	Total	(diff)
Drupal\Tests\lingotek\Unit\LingotekUnitTest	0 ms	0		0		9		9	
Drupal\Tests\lingotek\Unit\Remote\LingotekApiUnitTest	0 ms	0		0		5		5	
Drupal\lingotek\Tests\ChineseBulkTranslationTest	0 ms	0		0		5		5	
Drupal\lingotek\Tests\Form\LingotekAccountDisconnectFormTest	0 ms	0		0		3		3	
Drupal\lingotek\Tests\Form\LingotekAccountFormTest	0 ms	0		0		3		3	
Drupal\lingotek\Tests\Form\LingotekProfileFormTest	0 ms	0		0		6		6	
Drupal\lingotek\Tests\Form\LingotekSettingsTabConfigurationFormTest	0 ms	0		0		3		3	
Drupal\lingotek\Tests\Form\LingotekSettingsTabContentFormTest	0 ms	0		0		5		5	
Drupal\lingotek\Tests\LingotekAccountTest	0 ms	0		0		5		5	
Drupal\lingotek\Tests\LingotekBulkDeleteTest	0 ms	0		0		4		4	
Drupal\lingotek\Tests\LingotekChangeAccountDefaultsTest	0 ms	0		0		3		3	

CI WITH JENKINS (actually, D.O)

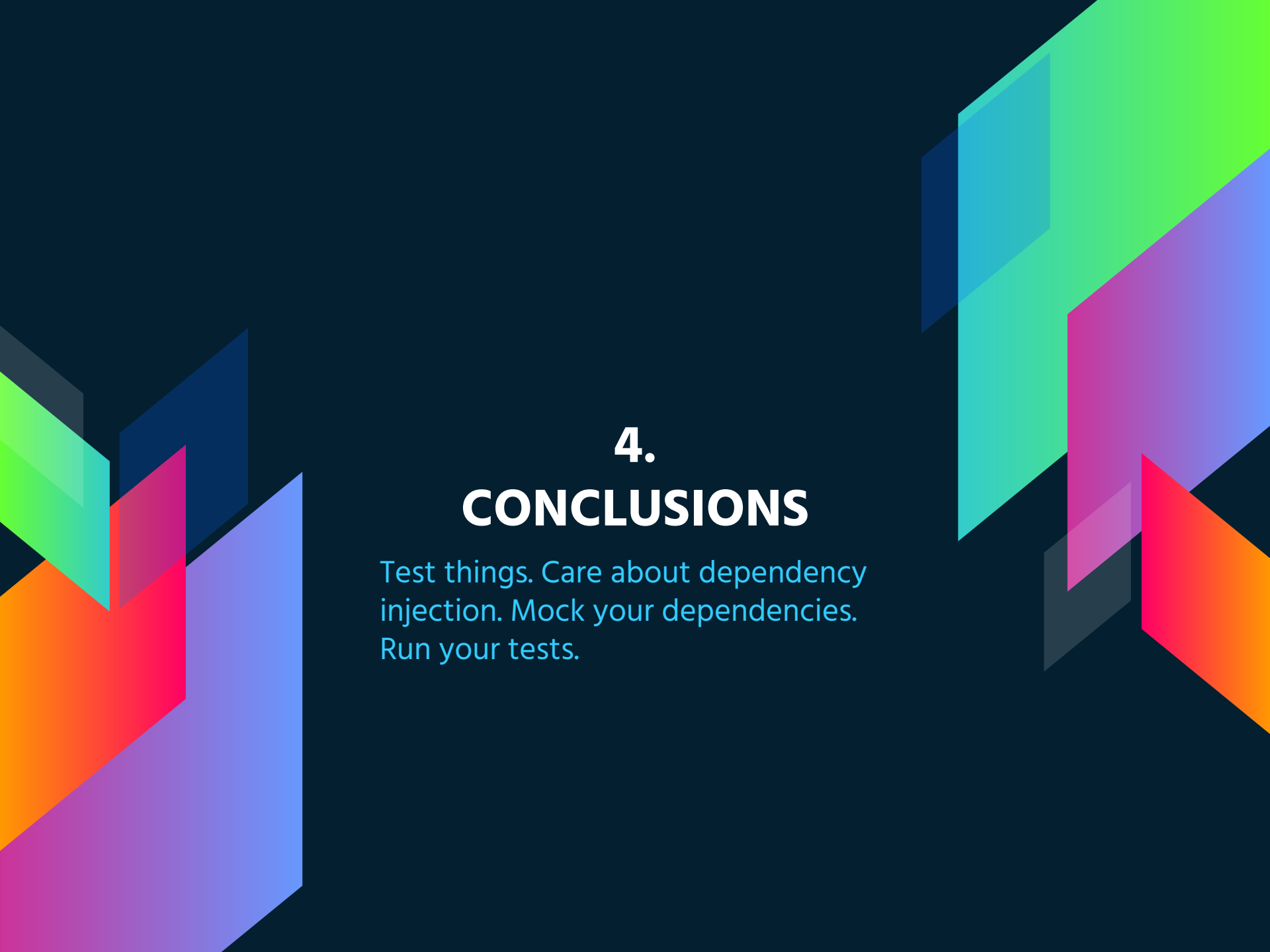

```
my_module.module.php
UNREGISTERED

OPEN FILES
my_module.module.php
my_module.services.yml
EntityTypeManager.php

1 <?php
2
3 use \Drupal\Core\Entity\EntityInterface;
4
5 /**
6  * Implements hook_entity_save().
7  */
8 function my_module_entity_create(EntityInterface $entity) {
9     \Drupal::service('my_module.entity_manager')
10     ->postCreateAlter($entity);
11 }
12
```

Line 12, Column 1 Spaces: 2 PHP

Procedural hook
**DO NOT INCLUDE
ANY LOGIC!**



4. **CONCLUSIONS**

Test things. Care about dependency injection. Mock your dependencies. Run your tests.

THE END

