A Guide To

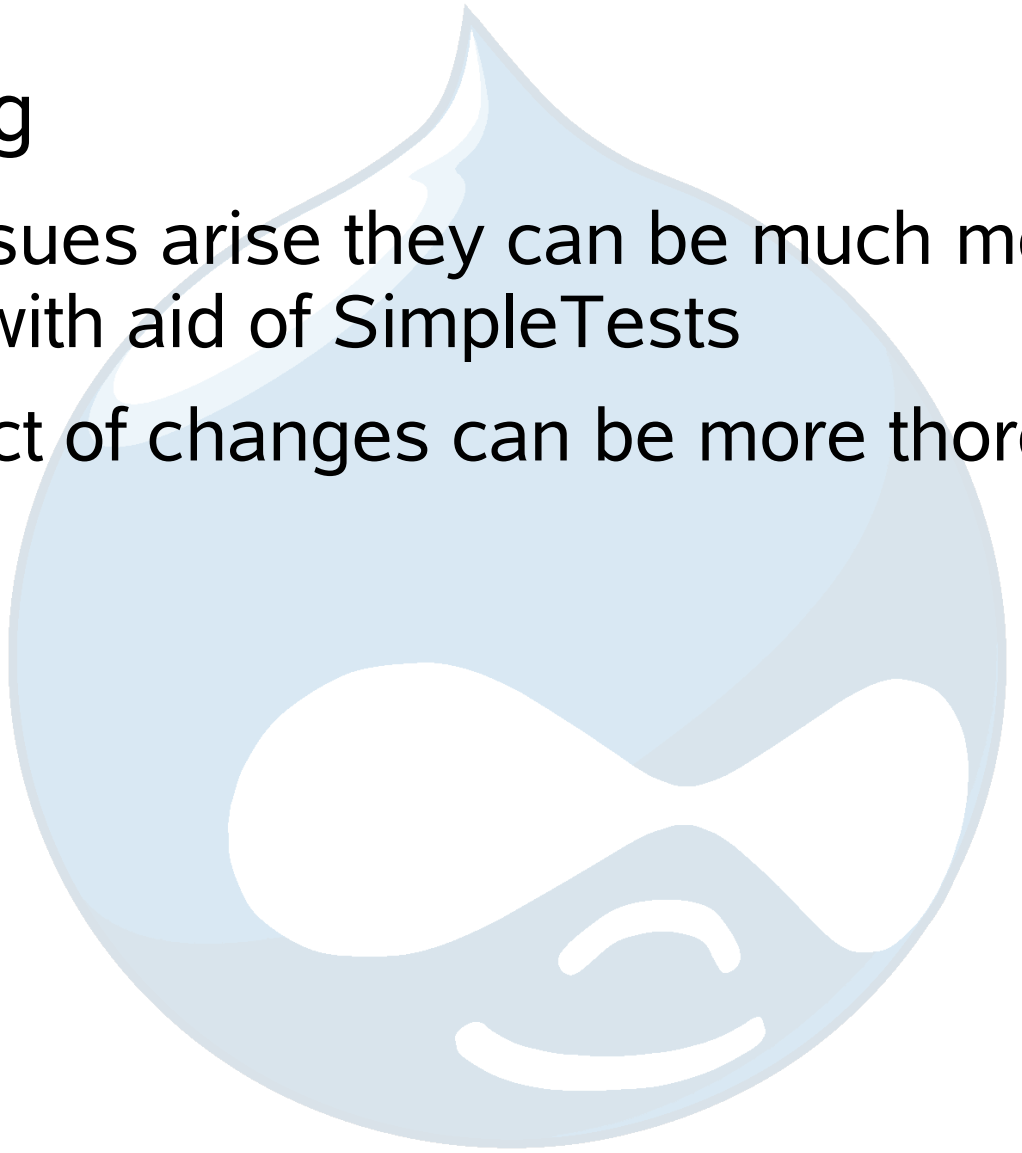
Drupal SimpleTests

Benefits of unit testing

- Ensure that modules are tested
- Provide an easy method for validating a patch
  - Tests can be run on the patch to confirm that everything still works
  - Prevent catchable mistakes from being committed
- Facilitate change
  - By allowing a developer to make sure that changes to core functionality do not have detrimental effects on dependent modules

Benefits of unit testing

- Debugging
  - When issues arise they can be much more easily located with aid of SimpleTests
  - The effect of changes can be more thoroughly tested



What is unit testing?

- Definition
  - "...unit testing is a procedure used to validate that individual units of source code are working properly." —Wikipedia
  - "A unit test is a method of testing the correctness of a particular module of source code." —JavaWorkshop

What is SimpleTest?

- SimpleTest is an open source PHP framework
  - Allows for unit testing to be done quickly and easily
- The framework provides:

Feature	Benefit
• Abstract class that automatically calls test functions	• Makes it easy to separate tests and have them executed
• Common testing functions	• Helps facilitate quick test development
• Internal web browser to imitate user requests	• Necessary to test web interface
• Feedback system that displays the test results	• Allows the developer to easily distinguish what is wrong

Drupal SimpleTesting

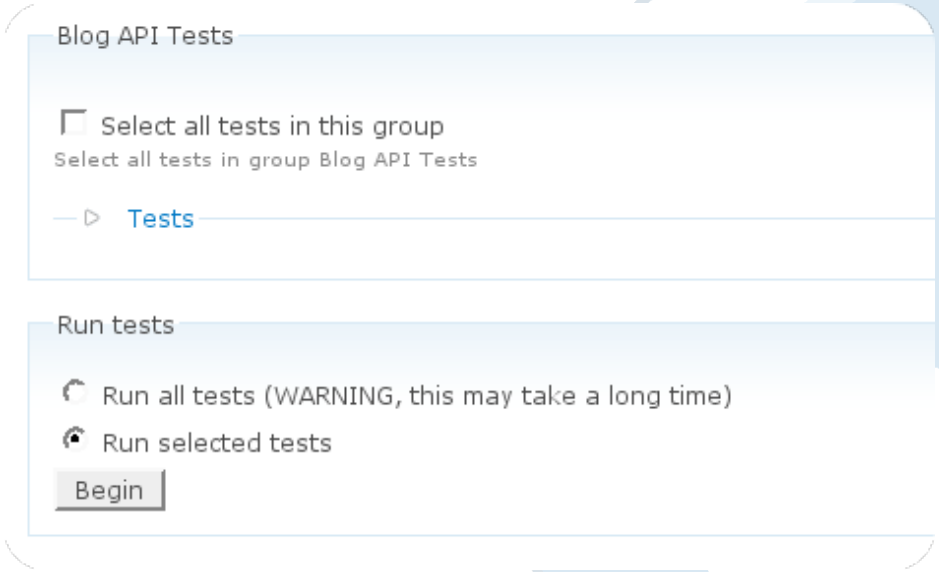
- Integrated SimpleTest environment
  - Easy to configure SimpleTest module
  - Administration page to run and review test results
  - Convenience functions for common Drupal testing tasks
  - Specialized internal browser
    - Format urls for Drupal system
    - Wraps functions to keep them consistent with Drupal standards

Drupal SimpleTesting

- SimpleTest Automator module provides a quick and easy way to create SimpleTests for Drupal
  - Configure user and permissions
  - Login that user
  - Record actions performed through Drupal interface
  - Clean up
    - The code can then be cleaned up and with a few additions can be completed

*http://drupal.org/project/simpletest_automator

Interface



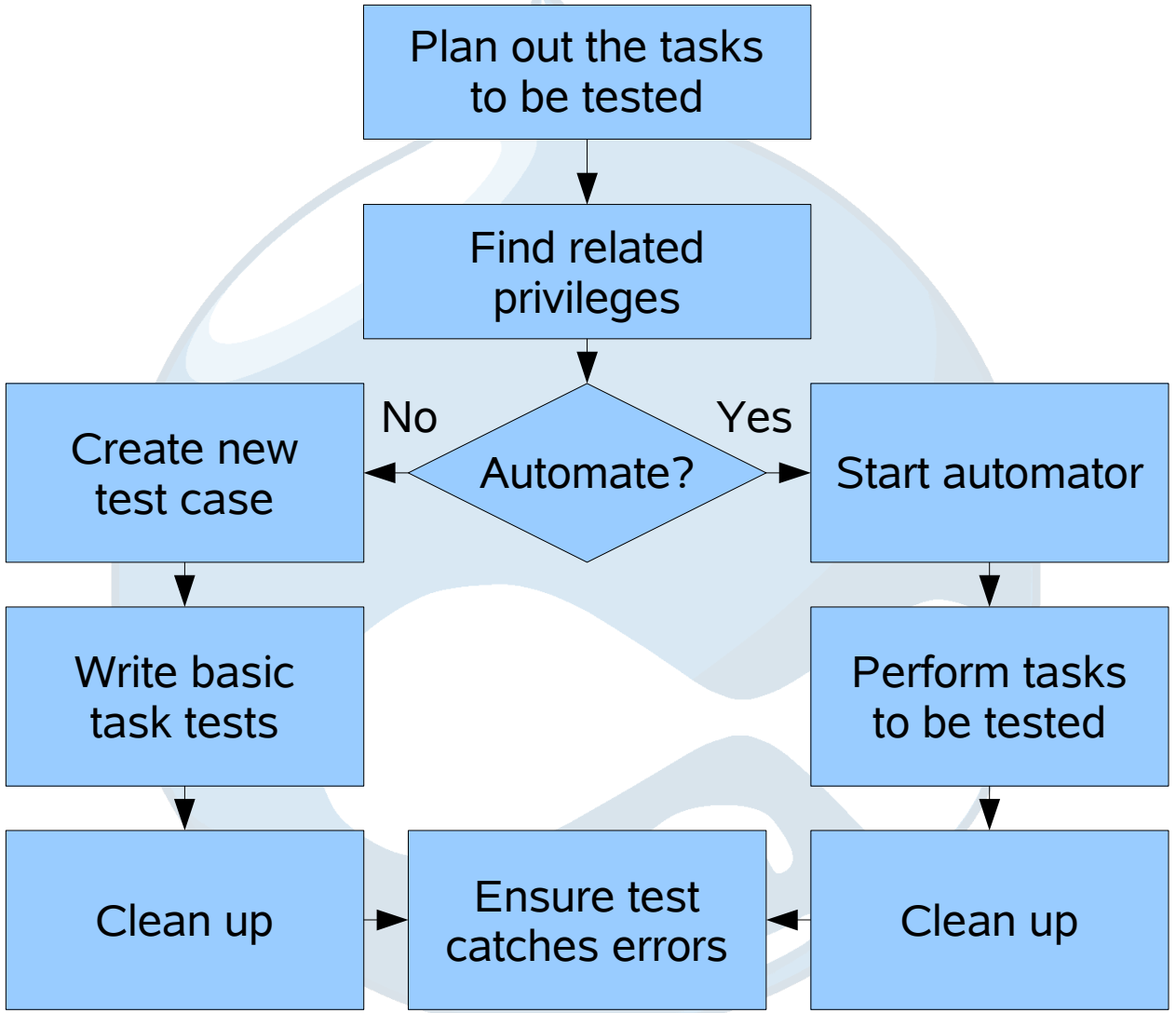
- Simple interface
  - Select tests
  - Run
- Tests are categorized to make them easy to find

Test Results

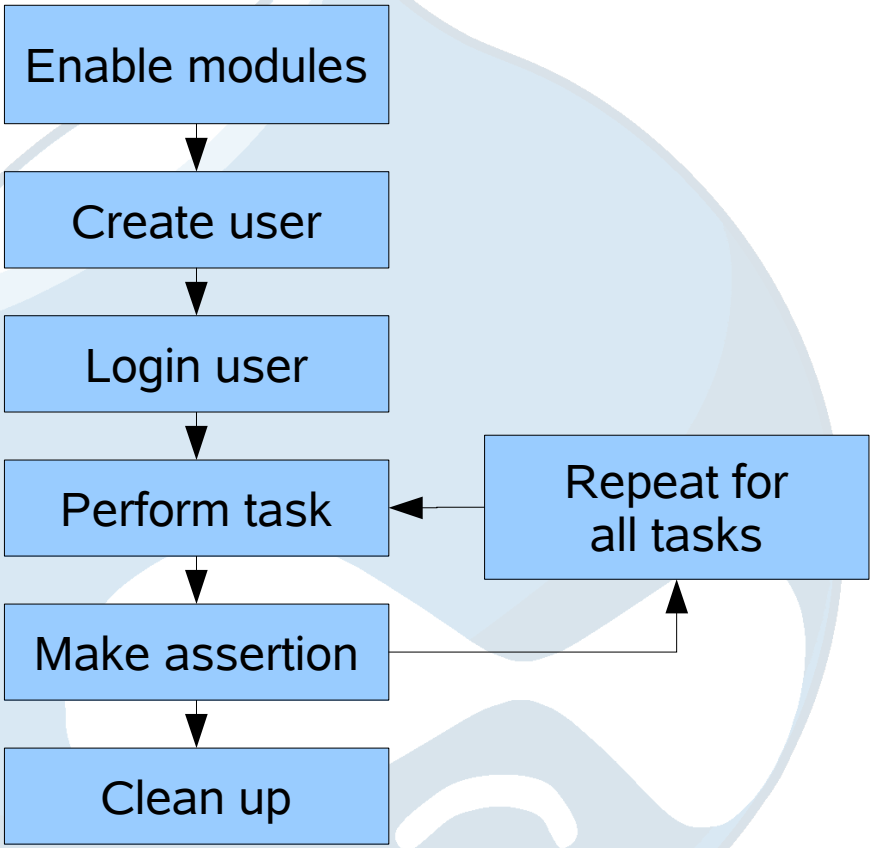
- Easy to read
  - Green—passed
  - Red—failed

Post successfully modified. at [/home/jimmy/public_html/drupal/modules/simpletest/tests/blogapi_module.php line 75]	OK
File successfully uploaded. at [/home/jimmy/public_html/drupal/modules/simpletest/tests/blogapi_module.php line 84]	OK
Uploaded contents verified. at [/home/jimmy/public_html/drupal/modules/simpletest/tests/blogapi_module.php line 90]	FAIL
Post categories set. at [/home/jimmy/public_html/drupal/modules/simpletest/tests/blogapi_module.php line 95]	OK
Category list successfully retreived. at [/home/jimmy/public_html/drupal/modules/simpletest/tests/blogapi_module.php line 99]	FAIL

Creating a SimpleTest



General SimpleTest Flow



SimpleTest Example

- Any tests that you create need to extend the *DrupalTestCase* class
  - Your class will then gain all the benefits of the SimpleTest library

```
class PresentationModuleTestCase extends DrupalTestCase
```

SimpleTest Info

- All SimpleTests should implement *get_info()*
  - This provides information about the test to the administration interface to help developers understand what the test will do

```
function get_info() {
  return array(
    'name' => t('Presentation abilities'),
    'desc' => t('Does a complete test of presentation abilities'),
    'group' => t('Presentation Tests'),
  );
}
```

SimpleTest Test Function

- Create a function beginning with the word “test” so that it will be executed as part of the test

```
function test_presentation()
```

Enable Modules

- Enable all used modules
  - This ensures that the testing environment is configured for the test to be performed

```
$this->drupalModuleEnable('presentation');
```

Create Test User

- Create a user for test
  - By creating a user with just the privileges required to complete the test the privilege system can be tested as well
  - Ensures that the testing environment is not the problem

```
$admin_user = $this->drupalCreateUserRolePerm(array('administer content types'));

$this->drupalLoginUser($admin_user);
```

Make POST Request

- POST data can be sent to a page to simulate user interaction
  - Allows user interface to be tested
  - Allows modules responding to interface to be tested

```
$edit = array();
$edit['name'] = 'John Doe';
$edit['foo'] = 'bar';
$this->drupalPostRequest('presentation/test', $edit, 'Save');
```

Make Assertion

- Use assertions to check results of actions
  - Simplifies code by removing conditional logic
  - Provides easy way to display text explaining test

```
$this->assertText(t('Saved Changes.'), 'Changes were saved successfully.');
```

Complete Example

```php
<?php
// $Id:

class PresentationModuleTestCase extends DrupalTestCase {
  function get_info() {
    return array(
      'name' => t('Presentation abilities'),
      'desc' => t('Does a complete test of presentation abilities'),
      'group' => t('Presentation Tests'),
    );
  }

  function test_presentation() {
    $this->drupalModuleEnable('presentation');
    $admin_user = $this->drupalCreateUserRolePerm(array('administer content types'));

    $this->drupalLoginUser($admin_user);

    $edit = array();
    $edit['name'] = 'John Doe';
    $edit['foo'] = 'bar';
    $this->drupalPostRequest('presentation/test', $edit, 'Save');

    $this->assertText(t('Saved Changes.'), 'Changes were saved successfully.');
  }
}
```

Limitations

- Lacks support for
  - JavaScript
    - Included script files cannot be checked
    - The result of scripts cannot be evaluated since the internal browser does not support JavaScript
  - Cascading Style Sheets
    - The visual aspect of the pages cannot be checked
    - The style sheets themselves are not accessible by SimpleTest

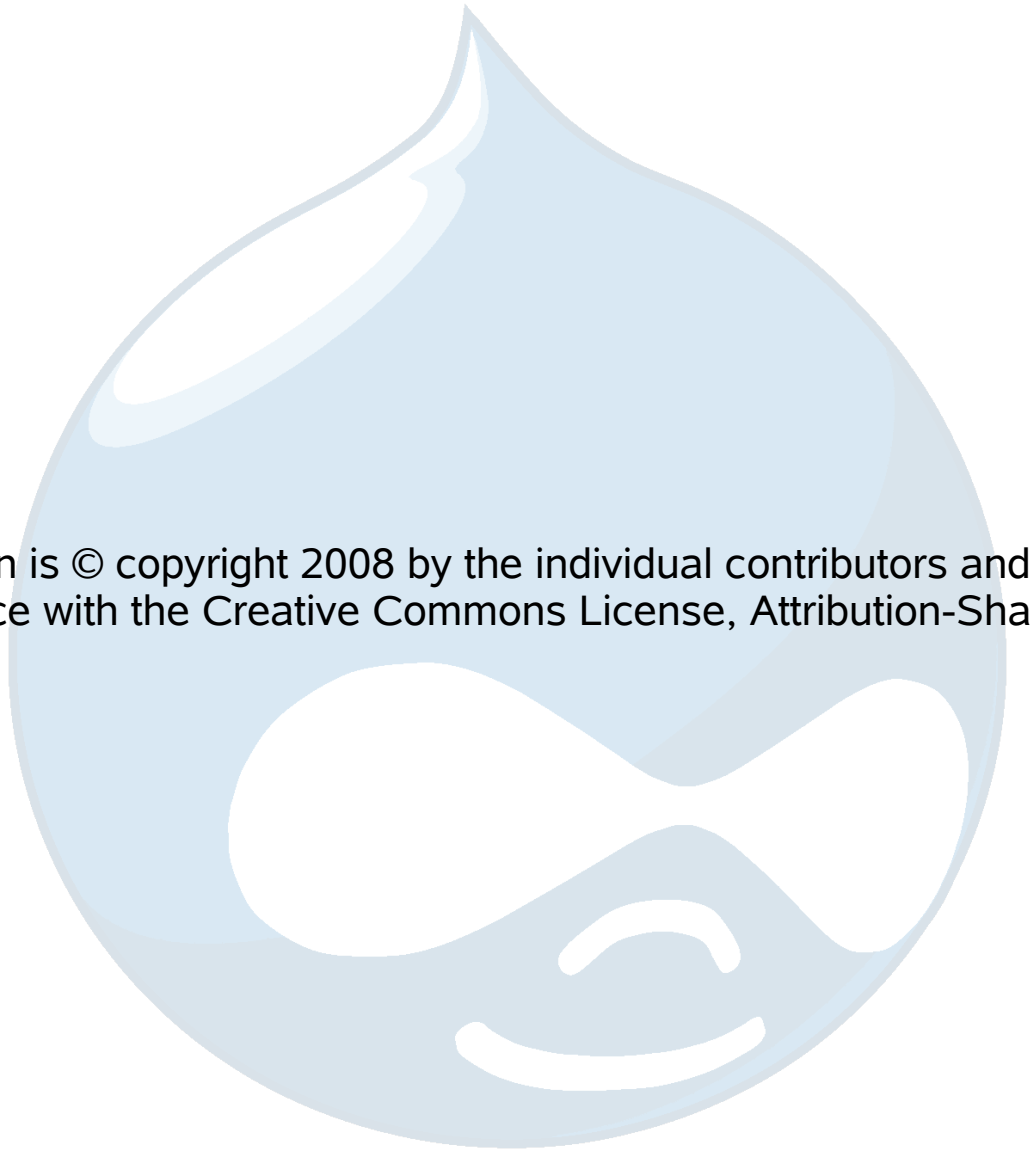
References

- Modules
  - SimpleTest—http://drupal.org/project/simpletest
  - SimpleTest Automator—http://drupal.org/project/simpletest_automator
- Documentation
  - SimpleTest—http://simpletest.org/
  - Drupal SimpleTest—http://drupal.org/simpletest
- Articles
  - http://www.lullabot.com/articles/introduction-unit-testing
  - http://www.lullabot.com/articles/drupal-module-developer-guide-simpletest

Author

- Jimmy Berry
  - I created a number of SimpleTests and was very impressed with the framework.
  - This presentation allowed me to compile my thoughts and provide an organized way to share them.

License



This presentation is © copyright 2008 by the individual contributors and can be used in accordance with the Creative Commons License, Attribution-ShareAlike 2.0.