DRUPAL 8 BEHAT KICKSTART

2016

Peter Sawczynec
Engineer

CivicActions

INTRO TO DRUPAL 8 TESTING ECOSYSTEM

Behat **SimpleTest** **PHPUnit** **JMeter**

Drupal Extension

Mink Selenium 2.0 Phantom.js

Mockery Prophecy Codeception

Wraith Casper.js Slimer.js Phantom.css

Jenkins Travis CI Circle.ci

CrossBrowserTesting.com Sauce Labs New Relic

Consider using these tools in your Drupal 8 development process:

phpStorm

drush

composer

drupal console

Behat

CivicActions

Behaviour Driven Development (BDD)

Formal Explanation

Behat is a tool that makes behavior driven development (BDD) possible. With BDD, you write human-readable stories that describe the behavior of your application. These stories can then be auto-tested against your application.

Informal Explanation

Behat is about the concept of using a simple coding language (Gherkin) with simple words to describe things that get done (behaviors) on your website that you want to test using an ecosystem of software libraries/tools.

Behat in Drupal 8

CivicActions

Behat usage in Drupal 8 involves installing a set of software libraries that work quite seamlessly together

In Drupal 8 install and update Behat and all the additional libraries and components you need with Composer

STATE OF TESTING IN DRUPAL 8

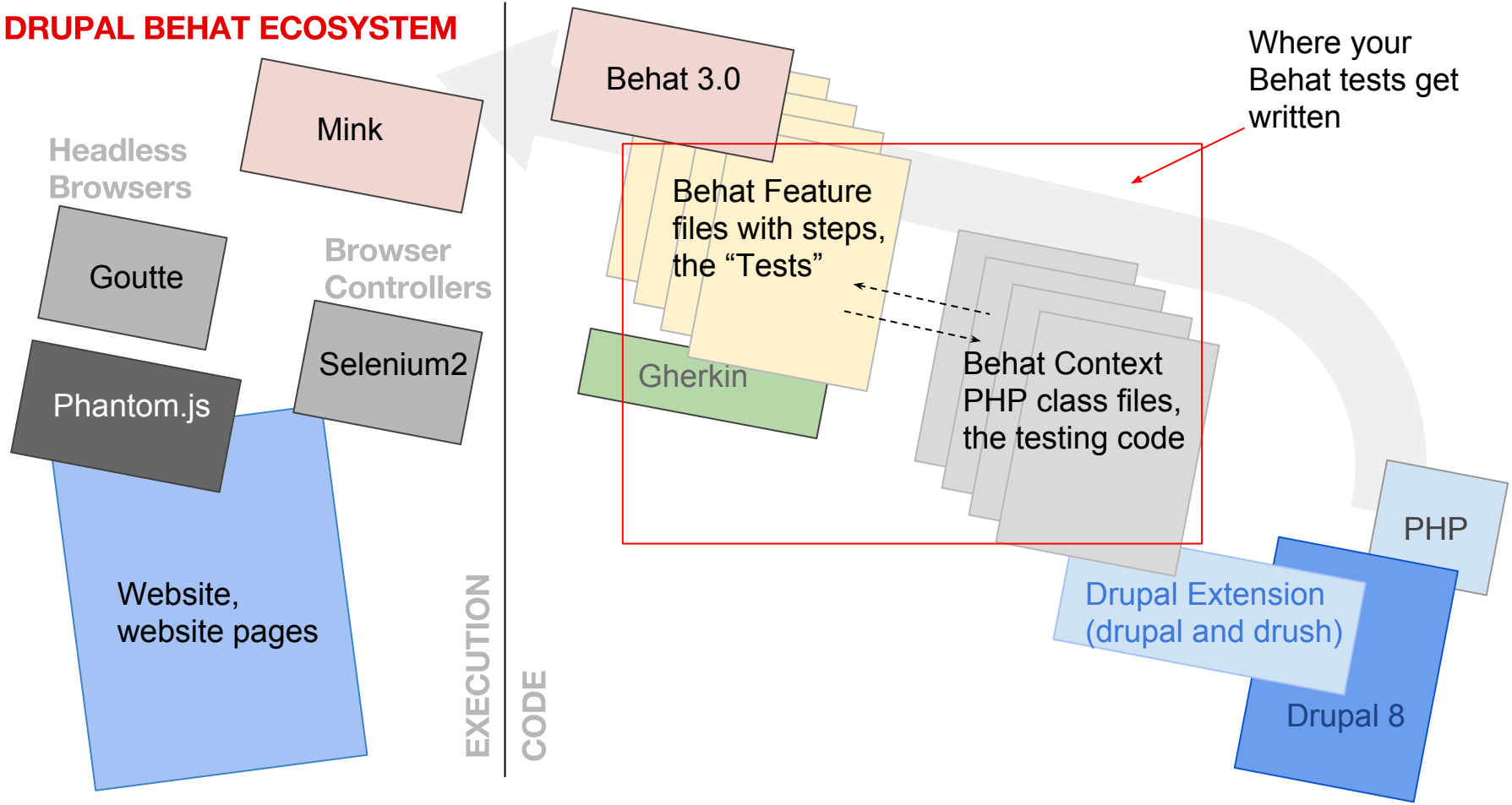
- **Drupal 8 ships with a vendor directory that contains Behat**
- **Drupal 8 ships with PHPUnit**
- **SimpleTest module also ships in D8 core, but must be enabled**

MINK, MINK EXTENSION, DRUPAL EXTENSION...

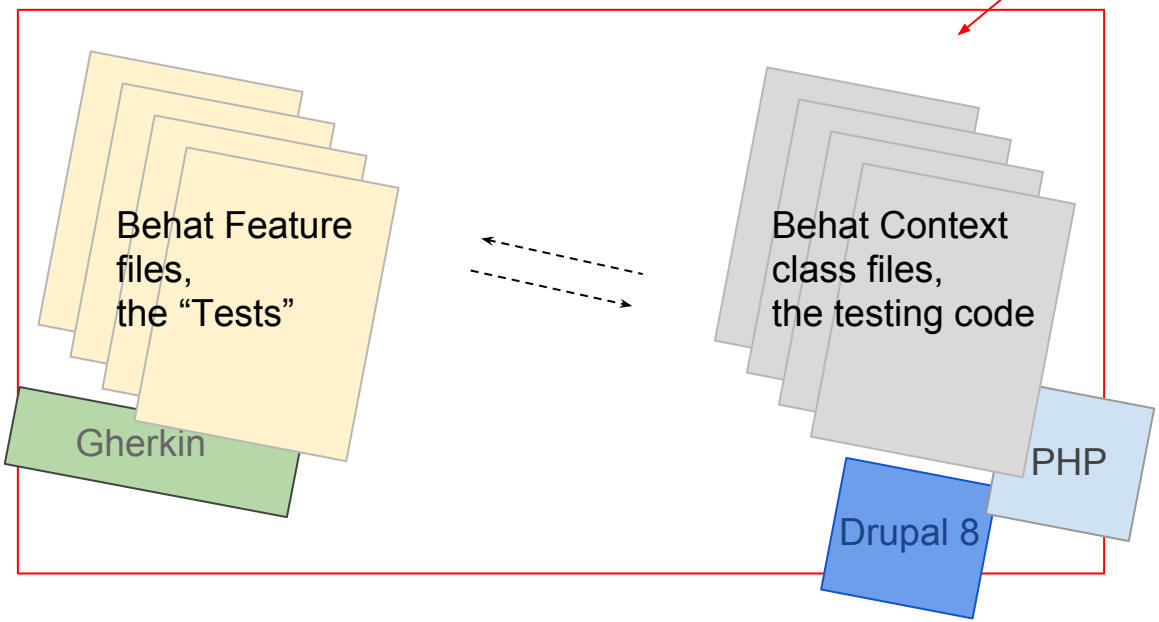
- **Mink, Mink Extension, Drupal Extension, Selenium, Phantom.js ...**
- **Software libraries like the above are important adjuncts to Behat, these libraries create the bridge to your web pages and to Drupal so you can run tests on your site**

Behat Ecosystem in Drupal 8

CivicActions


DRUPAL BEHAT ECOSYSTEM
Mink
Headless Browsers
Goutte
Browser Controllers
Selenium2
Phantom.js
Website, website pages
EXECUTION
CODE
Behat 3.0
Where your Behat tests get written
Behat Feature files with steps, the “Tests”
Gherkin
Behat Context PHP class files, the testing code
PHP
Drupal Extension (drupal and drush)
Drupal 8

The space where as a Drupal developer you write Behat tests and supporting code logic using Gherkin and PHP



Writing Behat Testing for Drupal 8

CivicActions

Comments and Annotations

- More useful and required in your code in Drupal 8.
- Used by Drupal 8, Behat, Symfony and other frameworks to “discover” your code and what it does

STEPS, SCENARIOS, BACKGROUND

Comments and Annotations Examples:

Comment

Annotation
(contains info for the framework)

```
/**
 * Step function to visit the last created node of a specific type.
 *
 * @param string $type
 *   The type of node that should be visited.
 *
 * @Given I visit the last created :type node
 */

/**
 * Views query plugin for an XML query.
 *
 * @ingroup views_query_plugins
 *
 * @ViewsQuery(
 *   id = "views_xml_backend",
 *   title = @Translation("XML Query"),
 *   help = @Translation("Query will be generated and run using the XML backend.")
 * )
 */
```

INTRO TO BDD

Feature file (Gherkin)

```
# Groups: Event, search
Given I run drush "search-api-index"
Then I login
And I click "Events"
Then I click "New Group Event"
Then I should see the text "Looking for members?"
And I follow the link element with xpath "//a[contains(@href,'/group-ela')]"
Given I visit the last created "article" node
```

Gherkin

Context class file (PHP)

```
/**
 * Step function to visit the last created node of a specific type.
 *
 * @param string $type
 *   The type of node that should be visited.
 *
 * @Given I visit the last created :type node
 */
public function iVisitTheLastCreatedNodeByType($type) {
  $node = $this->getLastCreatedEntityFromDb('node'  $type);
  $this->getSession()->visit($this->locatePath('/node/' . $node->nid );
}
```

Beha
class
the t

PHP

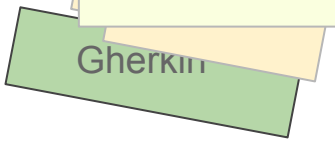
Drupal

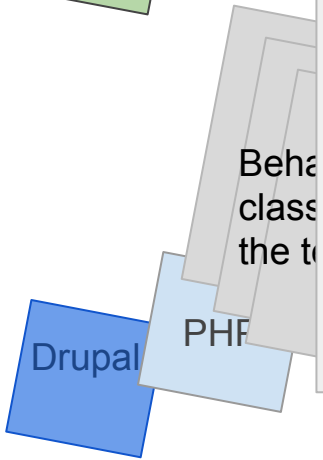
Feature file (Gherkin)

```
# Groups: Event, search
Given I run drush "search-api-index"
Then I login
And I click "Events"
Then I click "New Group Event"
Then I should see the text "Looking for members?"
And I follow the link element with xpath "//a[contains(@href,'/group-ela')]"
Given I visit the last created "article" node
```

Context class file (PHP)

```
/**
 * Step function to visit the last created node of a specific type.
 *
 * @param string $type
 *   The type of node that should be visited.
 *
 * @Given I visit the last created :type node
 */
public function iVisitTheLastCreatedNodeByType($type) {
  $node = $this->getLastCreatedEntityFromDb('node'  $type);
  $this->getSession()->visit($this->locatePath('/node/' . $node->nid );
}
```



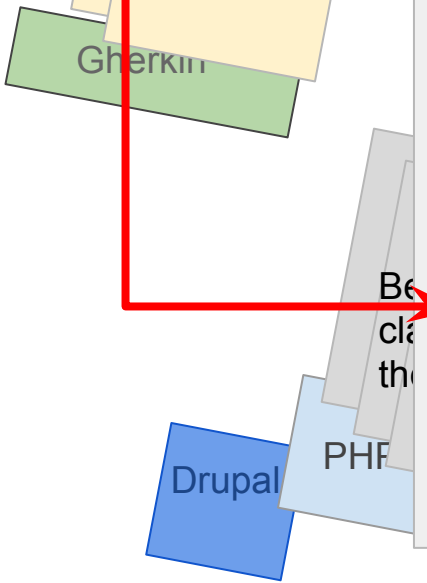


Feature file (Gherkin)

```
Then I login
Then I should see the text "Looking for members?"
Given I visit the last created "article" node
Then I logout
...
```

Context class file (PHP)

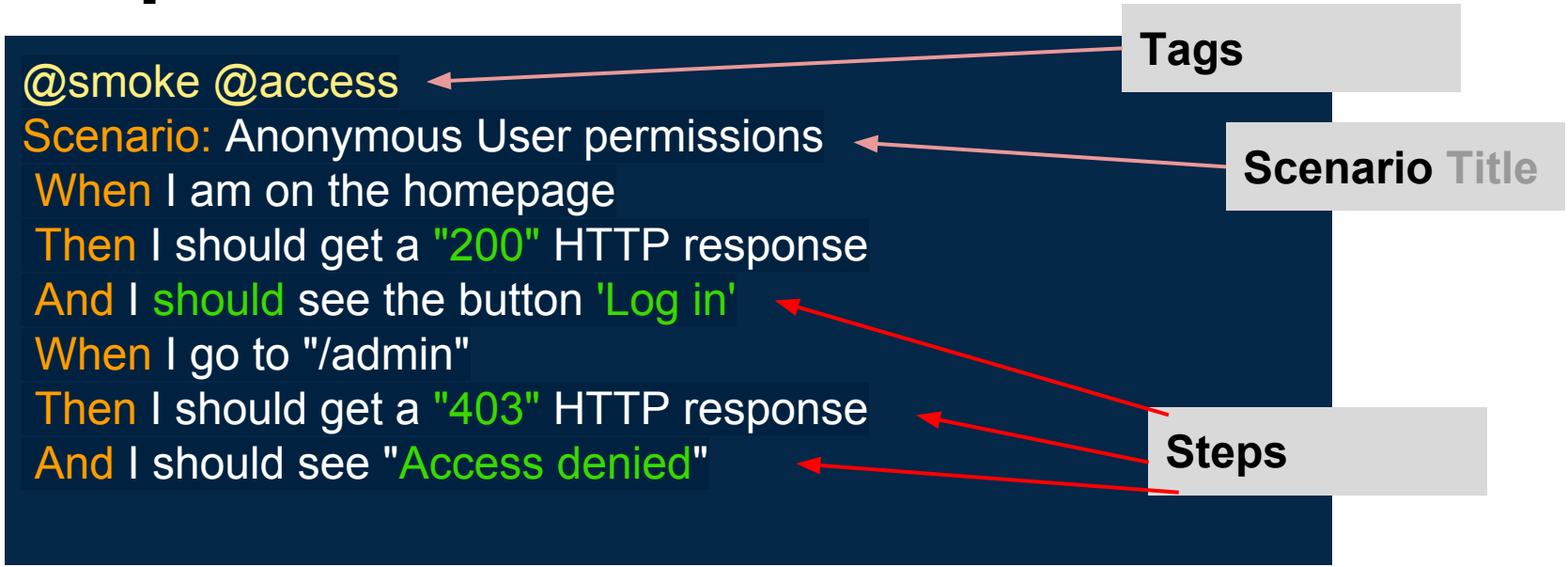
```
/**
 * Step function to visit the last created node of a specific type.
 *
 * @param string $type
 *   The type of node that should be visited.
 *
 * @Given I visit the last created :type node
 */
public function iVisitTheLastCreatedNodeByType($type) {
  $node = $this->getLastCreatedEntityFromDb('node'  $type);
```



Steps in Gherkin

```
When I am on the homepage
Then I should get a "200" HTTP response
And I should see the button 'Log in'
When I go to "/admin"
Then I should get a "403" HTTP response
And I should see "Access denied"
```

Steps in use in a Scenario:



TAGGING

```
@api @permissions
Feature: Specific user permissions
 As an authenticated user
 I should not be able to access admin pages
 So that I can verify my permissions

 @smoke
 Scenario: Anonymous User permissions
   When I am on the homepage
   Then I should get a "200" HTTP response
   And I should see the button 'Log in'
```

Tags to identify the whole feature file

Tags by scenario

STEPS, TAGS, SCENARIO, SCENARIO OUTLINE, BACKGROUND

```
@api @permissions
Feature: Specific user permissions
 As an authenticated user
 I should not be able to access admin pages
 So that I can verify my permissions

# Create standard users for all tests.
 Background:
   Given set content creation mode as "default"
   And load the background users:
     | user          |
     | uorgmanager1  |
     | umember1b     |

 @smoke
 Scenario: Anonymous User permissions
   When I am on the homepage
   Then I should get a "200" HTTP response
   And I should see the button 'Log in'
```

```
@member @group @access @info
Scenario Outline: Access user info, login history
  Given I am logged in as <user> with password <password>
  # View user login history
  And I visit the login history for <user_target>
  Then I should get a "<response>" HTTP response

  Examples:
    | user   | user_target | response | password      |
    | um1    | um11        | 404      | "xxx-xxx"     |
    | ug2    | um21        | 404      | "xxx-xxx"     |
    | ug1    | um32        | 200      | "xxx-xxx"     |
    | genl   | "qam"       | 200      | "xxx-xxx"     |
```

TABLE NODE

```
@api @permissions
Feature: Specific user permissions
 As an authenticated user
 I should not be able to access admin pages
 So that I can verify my permissions

# Create standard users for all tests.
 Background:
   Given set content creation mode as "default"
   And load the background users:
     | user          |
     | uorgmanager1  |
     | umember1b     |

 @smoke
 Scenario: Anonymous User permissions
   When I am on the homepage
   Then I should get a "200" HTTP response
   And I should see the button 'Log in'
```

Table Node data
Essentially how to pass an array to your Behat method in your context

BEHAT FEATURE FILE

```
@api @permissions
Feature: Specific user permissions
 As an authenticated user
 I should not be able to access admin pages
 So that I can verify my permissions

# Create standard users for all tests.
 Background:
   Given set content creation mode as "default"
   And load the background users:
     | user          |
     | uorgmanager1  |
     | umember1b     |

 @smoke
 Scenario: Anonymous User permissions
   When I am on the homepage
   Then I should get a "200" HTTP response
   And I should see the button 'Log in'
```

Feature file
A feature is a file filled with Gherkin code:
tags, steps, background, scenario, etc.
File suffix is .feature, e.g:
Smoke-access.feature

FEATURES, SUITES, HOOKS, CONTEXTS: SUITES

Suite
A set of features intended to run together for a purpose

FEATURES, SUITES, HOOKS, CONTEXTS: HOOKS

```php
/**
* This Behat hook runs after every step.
*
* @AfterStep
*/
public function failScreenshots(AfterStepScope $scope) {
    $this->saveScreenshot($filename, $this->screenshotDir);
    print 'Screenshot at: ' . $this->screenshotDir . $filename;
  }
 }
}

/**
* This Behat hook runs after every @access step.
*
* @AfterStep @access
*/
public function failScreenshots(AfterStepScope $scope) {
    $this->saveScreenshot($filename, $this->screenshotDir);
    print 'Screenshot at: ' . $this->screenshotDir . $filename;
  }
 }
}
```

Tagged hook

There are hooks:
before suite
after suite

before feature
after feature

before scenario
after scenario

before step
after step

transform

FEATURES, SUITES, HOOKS, CONTEXTS: CONTEXT CLASS FILES

```php
<?php
/**
 * @file
 * Operational Testing related context.
 */

namespace Tests\Drupal\Behat\Bootstrap\Context;

use Behat\Behat\Context\SnippetAcceptingContext,
 Drupal\DrupalExtension\Context\RawDrupalContext,
 Behat\Gherkin\Node\TableNode;

/**
 * Defines functionality for performing OT (Operational Tests).
 */
class OT extends RawDrupalContext implements SnippetAcceptingContext {

 use \Tests\Drupal\Behat\Bootstrap\Helper\All;

 private $currentUser;

 private $backgroundUsers;

 /**
  * Initializes context.
  *
  * Every scenario gets its own context instance.
```

Typical declarations and setup as PHP class file

FEATURES, SUITES, HOOKS, CONTEXTS: CONTEXT CLASS FILES

```php
/**
 * Initializes context.
 *
 * Every scenario gets its own context instance.
 * You can also pass arbitrary arguments to the
 * context constructor through a behat.yml.
 */
public function __construct() {
  $this->group_path = 'or1-gr1pu';
  $this->og = new Og();
}

/**
* Setting to create content on pre-existing demo content or not.
*
* @param string $mode
*   default|custom
*   E.g. default uses existing organization as base for all content.
*   custom creates now top level org.
*
* @When set content creation mode as :mode
*/
public function setContentCreationMode($mode) {
 $this->creationMode = $mode;
}
```

Every method in the class file gacn become a Gherkin step with the correct Annotation

CivicActions

Phantom.js

- The default Goutte driver grabs the page HTML one-time at page load and then works with that dom.
- Phantom.js can execute jQuery and then access new elements in the dom

BEHAT YAML FILE (behat.yml)

```
default:
 suites:
  default:
   contexts:
    - Tests\Drupal\Behat\Bootstrap\Context\Api
    - Tests\Drupal\Behat\Bootstrap\Context\Base extensions:
  Behat\MinkExtension:
   goutte:
    guzzle_parameters:
   selenium2:
    browser: chrome
    capabilities:
     version: ''
   files_path: "%paths.base%/media"
  Drupal\DrupalExtension:
   blackbox: ~
   api_driver: drupal
   drush:
    alias: 'local'
    root: '/var/www/docroot'
   drupal:
    drupal_root: '/var/www/docroot'
   region_map:
    main uppertabs:  "#tabs-0-main_uppertabs"
    main lowertabs:  "#tabs-0-main_lowertabs"
```

Top-level general config for all behat test sessions. Where you can set your drivers, etc.

BEHAT DEBUGGING STEPS

Then I print last response
Then I show last response
Then I break

BEHAT COMMAND PROMPT PARAMS

INFORMATIONAL

```
behat   --version
behat   -dl
behat   -dl | grep wait
behat   -df
```

RUNNING TESTS

```
behat   tests/behat/features
behat   tests/behat/features/checking.feature
behat   tests/behat/features  --tags '~@wip'
behat   tests/behat/features  --tags=@ot
behat   tests/behat/features  --tags=@ot&&~@access
```

DOCS

http://blog.lepine.pro/images/2012-03-behat-cheat-sheet-en.pdf

http://docs.behat.org/en/v3.0/

https://wiki.mahara.org/wiki/Testing/Behat_Testing/Characteristics_of_a_good_test

Open Discussion

CivicActions

Thank you.

CONFERENCE | PRESENTATION TITLE | NAME | @TWITTER | @CIVICACTIONS

CivicActions