

BEHAT KICKSTART FOR DRUPAL DEVELOPERS



Florida DrupalCamp 2016

Orlando, FL - March 5 - 6, 2016

Peter Sawczynec

Engineer

CivicActions

INTRO TO DRUPAL 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 development process:**

phpStorm
drush
composer
drupal console

Behat

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

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

INSTALL WITH COMPOSER

In Drupal 7 and 8 install and update Behat and all the additional libraries and components you need with Composer

STATE OF TESTING IN DRUPAL 8

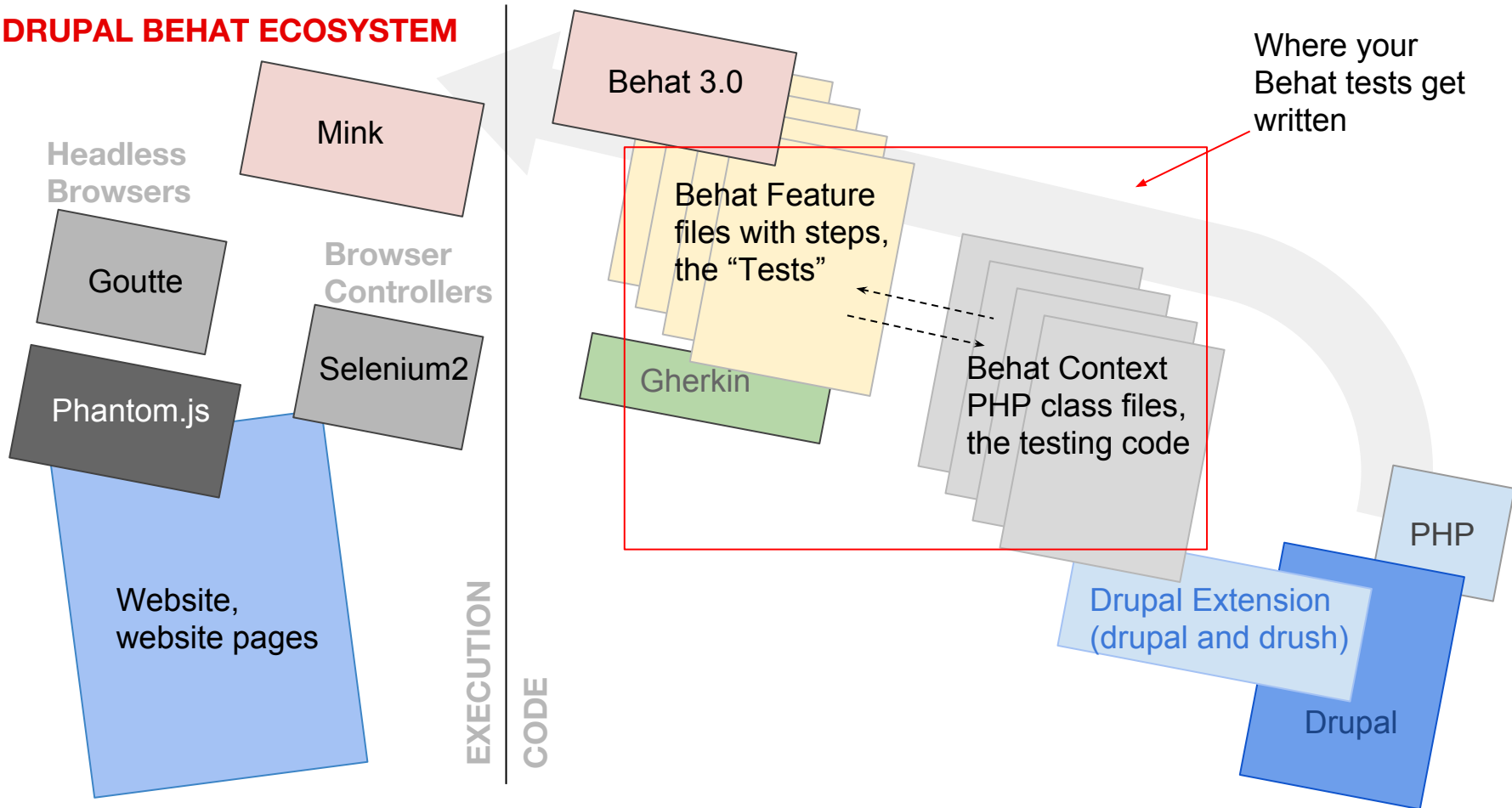
- **Drupal 8 ships with a vendor directory that contains Behat**
- **Drupal 8 ships with PHPUnit**
- **SimpleTest module also ships in D8 core, but must be enabled**

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

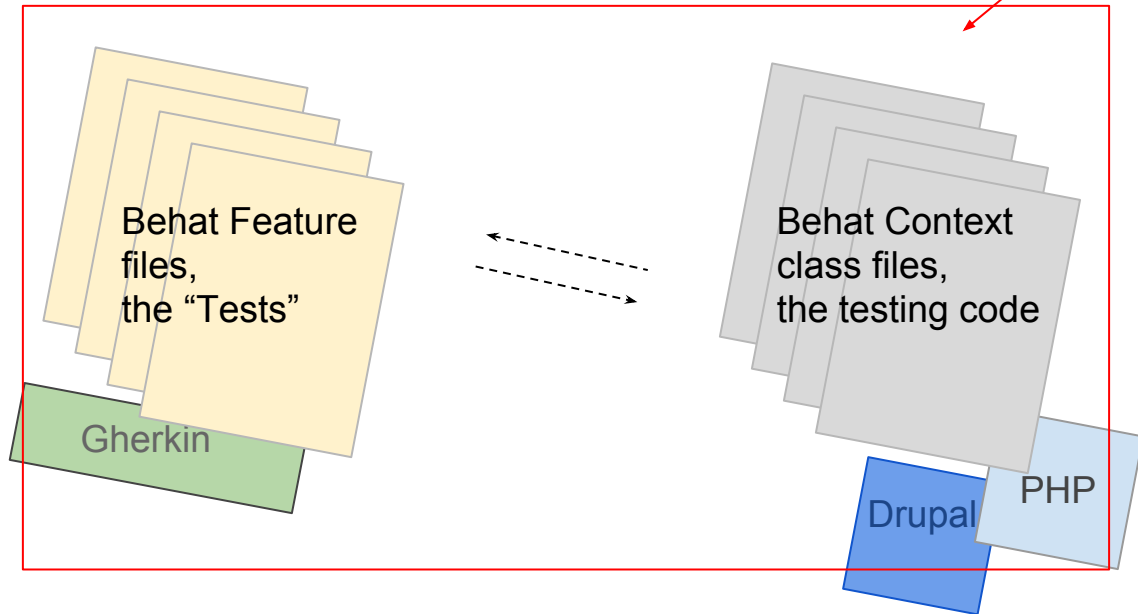
DRUPAL BEHAT ECOSYSTEM



INTRO TO BDD

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

Where your Behat tests get written



Writing Behat Testing for Drupal

Comments and Annotations

- More useful and required in your code.
- Used by Drupal, 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
```

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 ));
}
```

Gherkin

Beha
class
the t

PHP

Drupal

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
```

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 ));
}
```

Gherkin

Beha
class
the t

PHP

Drupal

INTRO TO BDD

Feature file
(Gherkin)

Context class file
(PHP)

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

```
/**  
 * 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);  
}
```

Gherkin

Be
cla
the

Drupal

PHP

STEPS

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

Steps in use in a Scenario:

The diagram illustrates the structure of a scenario with three main components labeled on the right: **Tags**, **Scenario Title**, and **Steps**. Red arrows point from these labels to the corresponding parts of the scenario text on the left.

- Tags:** Points to the `@smoke @access` line.
- Scenario Title:** Points to the `Scenario: Anonymous User permissions` line.
- Steps:** Points to the `When I go to "/admin"` line, which is the start of a sequence of steps.

The scenario text is as follows:

```
@smoke @access
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'
When I go to "/admin"
Then I should get a "403" HTTP response
And I should see "Access denied"
```

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

Tags to identify the
whole feature file

@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 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

Feature: Specific user permissions

I should not be able to access

So that I can verify my point.

Background

And load Feature: Specific

word I should not be ab

© 2013 Pearson Education, Inc. or its affiliate(s). All rights reserved.

Scenario Background:

Then I

usermanager

[Join our membership](#)

Scenario: Anonym

When I fall on the
Then I should go

And I should see

© 2014 Pearson Education, Inc. or its affiliate(s). All rights reserved.

Feature: Specific user permissions

I should not be able to see

permanencia

ed user

my permissions

users for all tests.

t creation mode as "default"

the homepage

the button 'Log in'

A set of features
intended to run together
for a purpose

FEATURES, SUITES, HOOKS, CONTEXTS: HOOKS

```
/**
 * 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
/**
 * @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.
     */
}
```

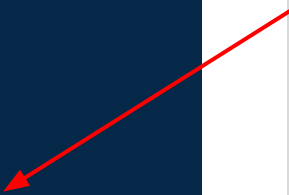


Typical declarations
and setup as PHP
class file

FEATURES, SUITES, HOOKS, CONTEXTS: CONTEXT CLASS FILES

```
/**
 * 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 can become a Gherkin step with the correct Annotation

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

Thank you.