



Júlio de Lima
iam@juliodelima.com.br

Automação de Testes de Aceitação com PHP



Júlio de Lima
iam@juliodelima.com.br

Sobre mim

- Lead QA Engineer na (evino)
- Formação em Tecnologia
- Pós graduação em Docência no Ensino Superior
- Articulista na DevMedia
- Palestrante e Coordenador da Trilha Testes do The Developers Conference São Paulo
- Professor a mais de 8 anos
- PHP, Javascript, Java e Node.js



Júlio de Lima
iam@juliodelima.com.br

Sobre mim

- ~~Lead QA Engineer na (evino)~~
- **Senior Consultant na Capco**
- Formação em Tecnologia
- Pós graduação em Docência no Ensino Superior
- ~~Articulista na DevMedia~~
- ~~Palestrante e Coordenador da Trilha Testes do The Developers Conference São Paulo~~
- Professor a mais de 8 anos
- **Instrutor na Udemy do curso de Automação com WebDriver em Java (bit.ly/5etapasCursoWebDriver35)**
- PHP, Javascript, Java e Node.js



Júlio de Lima
iam@juliodelima.com.br

Sobre a disciplina

1º encontro (16 de setembro)

Falaremos sobre a criação do ambiente e de seus primeiros scripts.

2º encontro (11 de novembro)

Vamos discutir sobre estratégias de teste e execução na nuvem.

3º encontro (18 de novembro)

Veremos alguns frameworks de teste utilizados no lugar do PHPUnit e falaremos sobre testes de performance.



Júlio de Lima
iam@juliodelima.com.br

Automação de Testes

Automatizar testes é uma prática que vem ganhando espaço no mercado de engenharia de software. O principal motivador é a constante mudança e evolução das aplicações desenvolvidas e a necessidade de manter tudo o que funciona, comportando-se como esperado.



Júlio de Lima
iam@juliodelima.com.br

Níveis de teste

1 Unidade

Onde são exercitadas as Menores partes de uma aplicação, por exemplo, um método.

2 Integração

Nesse nível, apenas as comunicações entre os componentes são exercitadas.

3 Sistema

Todo o software, já integrado, é testado com o objetivo de validar seu comportamento.



Júlio de Lima
iam@juliodelima.com.br

Níveis de teste

1 Unidade

Onde são exercitadas as Menores partes de uma aplicação, por exemplo, um método.

2 Integração

Nesse nível, apenas as comunicações entre os componentes são exercitadas.

3 Sistema

Todo o software, já integrado, é testado com o objetivo de validar seu comportamento.

Os testes de aceitação ocorrem nesse nível de teste. Com o objetivo de comprovar que o software comporta-se como esperado.



Júlio de Lima
iam@juliodelima.com.br

Teste de Aceitação

*Como usuário, desejo agendar tarefas
para que me lembre de executar
as atividades antes da data final*

Cenário: Adicionar uma nova tarefa

Dado que estou autenticado

Quando acesso o gerenciador de tarefas

E submeto o formulário com dados da tarefa

Então vejo a mensagem “Parabéns, tudo certo!”



Júlio de Lima
iam@juliodelima.com.br

Técnicas de projeto de teste

Há muitas técnicas para projetar casos de teste, mas dentre as mais conhecidas estão a partição de equivalência e os testes exploratórios.

1 Partição de equivalência

Técnica sistemática baseada na identificação das variáveis de entrada e suas faixas de valores

2 Testes exploratórios

Testes baseados no conhecimento empírico do testador, olhando para a tarefa, o que ela influi e o que a influencia.



Júlio de Lima
iam@juliodelima.com.br

Teste de Aceitação Automatizado

```
public function testAdicionarNovaTarefa () {  
    // Arrange  
    $navegador = RemoteWebDriver::create( 'http://localhost:4444',  
DesiredCapabilities::chrome() );  
    $navegador->get( 'http://www.juliodelima.com.br/taskit' );  
  
    // Act  
    $navegador->findElement(WebDriverBy::id( 'signup' ))->click();  
    $navegador->findElement(WebDriverBy::id( 'login' ))->sendKeys( 'julio' );  
}
```



Júlio de Lima
iam@juliodelima.com.br

Teste de Aceitação Automatizado

```
public function testAdicionarNovaTarefa () {  
    // Arrange  
    $navegador = RemoteWebDriver::create('http://localhost:4444/  
DesiredCapabilities::chrome());  
    $navegador->get('http://www.juliodelima.com.br/taskit');  
  
    // Act  
    $navegador->findElement(WebDriverBy::id('signup'))->click();  
    $navegador->findElement(WebDriverBy::id('login'))->sendKeys('julio');  
}
```

PHPUnit

Framework de testes de unidade
usado em nosso projeto com o
intuito de estruturar os testes



Júlio de Lima
iam@juliodelima.com.br

Teste de Aceitação Automatizado

WebDriver

Simulador de ações do usuário
disponível em diversas linguagens

```
public function testAdicionarNovaTarefa () {  
    // Arrange  
    $navegador = RemoteWebDriver::create( 'http://localhost:4444',  
DesiredCapabilities::chrome());  
    $navegador->get( 'http://www.juliodelima.com.br/taskit' );  
  
    // Act  
    $navegador->findElement(WebDriverBy::id( 'signup' ))->click();  
    $navegador->findElement(WebDriverBy::id( 'login' ))->sendKeys( 'julio' );  
}
```



Júlio de Lima
iam@juliodelima.com.br

Ambiente

1

PHP 7

[http://php.net/
downloads.php](http://php.net/downloads.php)

2

Composer

[https://
getcomposer.org/
download/](https://getcomposer.org/download/)

3

PHPUnit

<https://phpunit.de/>

4

ChromeDriver

[https://
chromedriver.storage.goo
gleapis.com/index.html?
path=2.32/](https://chromedriver.storage.googleapis.com/index.html?path=2.32/)



Júlio de Lima
iam@juliodelima.com.br

Ambiente

2 Composer

<https://getcomposer.org/download/>

```
// composer.json

{
    "require": {
        "phpunit/phpunit": "6.3.0",
        "facebook/webdriver": "1.4.1"
    }
}
```



Júlio de Lima
iam@juliodelima.com.br

Ambiente

4 ChromeDriver

https://
chromedriver.storage.goo
gleapis.com/index.html?
path=2.32/

chromedriver --port=4444



Júlio de Lima
iam@juliodelima.com.br

Escrevendo o primeiro script

A criação do arquivo que abrigará os testes é uma classe PHP que estende a classe TestCase. Esta classe possui métodos públicos sem retorno. Dentro dos métodos, adicionamos comandos do WebDriver, capazes de simular as ações do usuário real.

```
<?php
use PHPUnit\Framework\TestCase;
use Facebook\WebDriver\Remote\DesiredCapabilities;
use Facebook\WebDriver\Remote\RemoteWebDriver;
use Facebook\WebDriver\WebDriverBy;
```




Júlio de Lima
iam@juliodelima.com.br

Escrevendo o primeiro script

Como dito anteriormente, usamos o PHPUnit como base fundamenta da execução e estruturação dos testes com execução via binário contido dentro do diretório vendor.

```
class SignInTests extends TestCase {  
    public function testSignInWithAValidUserAndSeeTheGreeting() {  
        // Arrange  
        // Act  
        // Assert  
    }  
}
```

```
$ ./vendor/bin/phpunit tests/SigninTests.php --testdox-html "reports/  
index.html"
```



Júlio de Lima
iam@juliodelima.com.br

phpunit.xml

O arquivo phpunit.xml serve para definir uma série de configurações referentes à forma com que os testes serão executados.

```
<phpunit bootstrap="vendor/autoload.php" colors="true" stopOnError="false">
  <testsuites>
    <testsuite name="smoke">
      <file>tests/SigninTests.php</file>
    </testsuite>
    <testsuite name="regression">
      <directory>tests</directory>
    </testsuite>
  </testsuites>
  <logging>
    <log type="testdox-html" target="reports/index.html"/>
  </logging>
</phpunit>
```

```
$ ./vendor/bin/phpunit --testsuite smoke
```



Júlio de Lima
iam@juliodelima.com.br

Arrange

A área de preparação para execução do testes que será executado.

```
// Arrange
$driver = RemoteWebDriver::create( 'http://localhost:4444',
DesiredCapabilities::chrome() );
$driver->manage()->window()->maximize();
$driver->manage()->timeouts()->implicitlyWait(5);
$driver->get( 'http://www.juliodelima.com.br/taskit' );
```



Arrange

A área de preparação para execução do testes que será executado.

ChromeDriver

Parser que envia e recebe os comando do browser.

```
// Arrange
$driver = RemoteWebDriver::create('http://localhost:4444',
DesiredCapabilities::chrome());
$driver->manage()->window()->maximize();
$driver->manage()->timeouts()->implicitlyWait(5);
$driver->get('http://www.juliodelima.com.br/taskit');
```



Júlio de Lima
iam@juliodelima.com.br

Método do Elemento

Métodos para interação com o browser.
Eg. click(), getText(), etc.

Estratégias de identificação

Método de identificação dos elementos
Eg. 'className', 'cssSelector', 'id', 'name', 'linkText', 'partialLinkText', 'tagName' and 'xpath'.

Act

As ações que, basicamente, irão dar sentido ao método de teste.

```
// Act
```

```
$driver->findElement(WebDriverBy::linkText('Sign in'))->click();  
$driver->findElement(WebDriverBy::cssSelector('#signinbox  
input[name="login"]'))->sendKeys('unicesumar');  
$driver->findElement(WebDriverBy::cssSelector('#signinbox  
input[name="password"]'))->sendKeys('123456');  
$driver->findElement(WebDriverBy::cssSelector('#signinbox'))->  
>findElement(WebDriverBy::linkText('SIGN IN'))->click();  
$username = $driver->findElement(WebDriverBy::className('me'))->  
>getText();
```




Júlio de Lima
iam@juliodelima.com.br

Assert

Asserções

Métodos para comparação entre valores
eles mostram ao runner os resultados.

Fechando o browser

O método quit fecha todas as abas abertas

As ações que, basicamente, validam se um dado atende as perspectivas.

```
// Assert  
$this->assertContains('Unicesumar', $username);  
$driver->quit();
```



Júlio de Lima
iam@juliodelima.com.br

Let's refactor!



Júlio de Lima
iam@juliodelima.com.br

Before

O conteúdo com método que possui essa anotação será executado todas as vezes, antes de cada método de teste.

```
private $driver;
```

```
public function setUp() {  
    $this->driver = RemoteWebDriver::create( 'http://localhost:4444' ,  
DesiredCapabilities::chrome() );  
    $this->driver->manage()->window()->maximize();  
    $this->driver->manage()->timeouts()->implicitlyWait(5);  
    $this->driver->get( 'http://www.juliodelima.com.br/taskit' );  
}
```




Júlio de Lima
iam@juliodelima.com.br

After

O conteúdo com método que possui essa anotação será executado todas as vezes, antes de cada método de teste.

```
public function tearDown() {  
    $driver->quit();  
}
```



Júlio de Lima
iam@juliodelima.com.br

Estratégias de identificação

Facebook\WebDriver\WebDriverWait

1. WebDriverBy::className
<div **class="mensagem"**></div>

2. WebDriverBy::id
<div **id="mensagem"**></div>

3. WebDriverBy::name
<input **name="nome"** />

4. WebDriverBy::linkText
TASK IT!

5. WebDriverBy::partialLinkText
TASK IT!

6. WebDriverBy::tagName
<**h1**>Task it!</h1>

7. WebDriverBy::css selector
<div class="mensagem btn sucesso"></div>
.mensagem.btn.sucesso

8. WebDriverBy::xpath
<div data-id="mensagem">Ver mensagem</div>
//div[@data-id="mensagem"][text()="Ver mensagem"]



Júlio de Lima
iam@juliodelima.com.br

Keyword-Driven Testing

Uma forma de intermediar o conhecimento de testes manuais com os testes automatizados.

Objeto	Palavra-chave	Valor
Sign in	Clicar	
#signinbox input[name="login"]	Digitar	unicesumar
#signinbox input[name="password"]	Digitar	123456
SIGN IN	Clicar	



Júlio de Lima
iam@juliodelima.com.br

Combobox

Facebook\WebDriver\WebDriverSelect

Caixas de seleção são elementos bastante específicos, uma vez que são compostos por funcionalidades compartilhadas entre o grupo e seus itens. Há uma classe específica para este objeto, chamada WebDriverSelect.

```
$fieldType = $driver->findElement(WebDriverBy::name('type'));  
$comboType = new WebDriverSelect($fieldType);  
$comboType->selectByValue('email');
```



Júlio de Lima
iam@juliodelima.com.br

Outras ações que podem ser executadas

selectByIndex(int \$index)

Selecionar um item a partir do seu índice numérico

selectByValue(string \$value)

Selecionar um item a partir do valor contido na propriedade "value"

selectByVisibleText(string \$text)

Selecionar um item a partir do texto apresentado na combo

selectByVisiblePartialText(string \$text)

Selecionar um item a partir de parte do texto apresentado na combo

getFirstSelectedOption()

Obter o valor que encontra-se selecionado na combo



Espera explícita

Facebook\WebDriver\WebDriverWait

Facebook\WebDriver\WebDriverExpectedCondition

Esperar até que uma condição seja atendida. Geralmente essa condição está relacionada a um elemento que será apresentado no DOM da página.

```
$wait = new WebDriverWait($this->driver, 5, 500);  
$wait->until(  
WebDriverExpectedCondition::visibilityOfElementLocated(WebDriverBy::id("mens  
agem"))  
);
```

Timeout em segundos

Intervalo de checagem em milisegundos



Júlio de Lima
iam@juliodelima.com.br

Outras condições disponíveis

WebDriverExpectedCondition::presenceOfElementLocated(WebDriverBy \$by)

Aguardar até que um determinado elemento seja adicionado ao DOM

WebDriverExpectedCondition::titleIs(string \$title)

Aguardar até que o título da página seja igual a \$title

WebDriverExpectedCondition::elementTextContains(WebDriverBy \$by, string \$text)

Aguardar até que o texto contido no elemento seja igual a \$text

WebDriverExpectedCondition::elementToBeClickable(WebDriverBy \$by)

Aguardar até que \$by seja clicável

WebDriverExpectedCondition::alertIsPresent()

Aguardar até que uma mensagem Javascript seja apresentada

<http://facebook.github.io/php-webdriver/1.4.0/Facebook/WebDriver/WebDriverExpectedCondition.html>

2017 | WEBDEV - UniCesumar



Júlio de Lima
iam@juliodelima.com.br

Javascript

Facebook\WebDriver\WebDriver

Janelas Javascript não possuem sua estrutura baseada em DOM. Ao invés disso, são recursos carregados diretamente pelo browser, por isso, possuem tratativa diferente feita por uma classe específica.

```
$alerta = $this->driver->switchTo()->alert();  
$alerta->getText();  
$alerta->accept();
```

sendKeys(string \$text)

Digitar texto no prompt

dismiss()

Clique no botão cancelar

<https://facebook.github.io/php-webdriver/latest/Facebook/WebDriver/WebDriverAlert.html>

2017 | WEBDEV - UniCesumar



Júlio de Lima
iam@juliodelima.com.br

Screenshots

Facebook\WebDriver\WebDriver

Obter um screenshot da tela que está aberta no momento da execução do script.

```
$this->driver->takeScreenshot( "evidencias/screenshot.jpg" );
```



Júlio de Lima
iam@juliodelima.com.br

Report

Há diversos formatos de relatórios para execução dos testes no PHPUnit, talvez o que mais se adeque à questão de testes de aceitação seja o formato TestDox, uma vez que gera uma descrição em texto descritivo:

```
./vendor/bin/phpunit test --testdox-html "reports/index.html"
```



Júlio de Lima
iam@juliodelima.com.br

Outros formatos de report

--log-junit <file>	Logar a execução dos testes em formato JUnit XML.
--log-teamcity <file>	Logar a execução dos testes em formato TeamCity
--testdox-html <file>	Logar a execução dos testes em formato ágil HTML
--testdox-text <file>	Logar a execução dos testes em formato ágil TXT



Júlio de Lima
iam@juliodelima.com.br

Sobre a disciplina

1º encontro (16 de setembro)

Falaremos sobre a criação do ambiente e de seus primeiros scripts.

2º encontro (11 de novembro)

Vamos discutir sobre estratégias de teste e execução na nuvem.

3º encontro (18 de novembro)

Veremos alguns frameworks de teste utilizados no lugar do PHPUnit e falaremos sobre testes de performance.



Júlio de Lima
iam@juliodelima.com.br

Componentize seus testes!

Você decide como fará isso, a idéia é fazer algo com manutenibilidade maior

Scripts de teste são semelhante a código de aplicação, logo, use sua criatividade para fazer com que seu projeto seja mais fácil de dar manutenção e mais fácil de usar.

Imagine que que irá criar os testes do PHPUnit são pessoas que não manjam de WebDriver, mas conhecem um pouco de PHP.



Júlio de Lima
iam@juliodelima.com.br

Page Objects (Dojo)

Modularizando seus testes e ganhando manutenibilidade

- Cada página possui métodos que representam as ações que podem ser feitas na página
- Todo método retorna a próxima página que irá aparecer após a execução do método
- Toda página possui um construtor que recebe a instância do WebDriver e armazena em uma propriedade
- Todo teste é iniciado pela instanciação da primeira página que será utilizada



Júlio de Lima
iam@juliodelima.com.br

Data-Driven Testing

Um mesmo script de teste, com dados diferentes

Quando há um teste que terá a mesma estrutura, independente de quais dados sejam passados a ele, temos a possibilidade de usar essa estratégia de testes.

Basicamente, todos os testes terão parâmetros que serão alimentados por um método de Data Source.



Júlio de Lima
iam@juliodelima.com.br

Data-Driven Testing

Um mesmo script de teste, com dados diferentes

```
public static function dataDoSigninUsingAnExistentUser() {
    return array(
        'Aluno Unicesumar' => array('unicesumar', '123456', 'Unicesumars'),
        'Julio' => array('julio0001', '123456', 'Julio')
    );
}

/**
 * @dataProvider dataDoSigninUsingAnExistentUser
 */
public function testDoSigninUsingAnExistentUser($login, $senha, $nome) {
    // Act
    $this->driver->findElement(WebDriverBy::linkText('Sign in'))->click();
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox input[name="login"]'))->sendKeys($login);
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox input[name="password"]'))->sendKeys($senha);
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox'))->findElement(WebDriverBy::linkText('SIGN IN'))->click();
    $username = $this->driver->findElement(WebDriverBy::className('me'))->getText();

    // Assert
    $this->assertContains($nome, $username);
    $this->driver->takeScreenshot("$nome.jpg");
}
```




Júlio de Lima
iam@juliodelima.com.br

Data-Driven Testing

Um mesmo script de teste, com dados diferentes

Pesquise e implemente uma forma de fazer com que os dados sejam capturados de um arquivo externo, como um CSV, JSON ou XLS.



Júlio de Lima
iam@juliodelima.com.br

Execução de Testes na Nuvem

Que tal ter mais flexibilidade e cobertura?

Há serviços pagos que provêm uma larga variação de SOs e Browsers de diversas Versões para execução de testes de software. Essas ferramentas, permitem que você dispare seus testes apontando quais são as capacidades necessárias, e então eles executaram os testes e fornecem um rico relatório de execução.

www.browserstack.com

www.saucelabs.com



Júlio de Lima
iam@juliodelima.com.br

Execução de Testes na Nuvem

Que tal ter mais flexibilidade e cobertura?

```
$browserStack = 'https://juliocesardelima2:wUCEnr7XjcXSEfLZsSvp@hub-cloud.browserstack.com/wd/hub';  
$this->driver = RemoteWebDriver::create($browserStack, array(  
    "browser" => "Chrome",  
    "browser_version" => "62.0",  
    "os" => "Windows",  
    "os_version" => "10",  
    "resolution" => "1280x800"  
));
```

Veja mais Capabilities:

<https://www.browserstack.com/automate/php>



Júlio de Lima
iam@juliodelima.com.br

Execução de Testes na Nuvem

Que tal ter mais flexibilidade e cobertura?

- Pesquise e implemente uma forma de reportar no BrowserStack os testes que falharam.
- Faça com que seus testes rodem na SauceLabs



Júlio de Lima
iam@juliodelima.com.br

Sobre a disciplina

1º encontro (16 de setembro)

Falaremos sobre a criação do ambiente e de seus primeiros scripts.

2º encontro (11 de novembro)

Vamos discutir sobre estratégias de teste e execução na nuvem.

3º encontro (18 de novembro)

Veremos alguns frameworks de teste utilizados no lugar do PHPUnit e falaremos sobre testes de performance.



Júlio de Lima
iam@juliodelima.com.br

Behavior-Driven Development

Que tal desenvolver algo pensando no comportamento?

No BDD, definimos as funcionalidades das aplicações a partir de exemplos práticos escritos através de uma estrutura sistemática, mas ao mesmo tempo, livre.

Os exemplos usam palavras chave como Given When e Then. Além de conectores como Mas e E. Também é permitido o uso de tabelas e do conceito de Data-Driven Development.



Júlio de Lima
iam@juliodelima.com.br

Feature: Contact

In order to change info from my profile
As a ordinary user
I need to be able to manage my contacts

Rules:

- I can add phones
- I can add telephones
- I can add duplicated data

Scenario: Adding a Phone

Given I am logged in

And I am seeing my contact data

When I add a new phone

Then I see It was my phone



Júlio de Lima
iam@juliodelima.com.br

Behat

Um framework para automatizar suas estórias!

Behat é um framework baseado no Cucumber, um framework Ruby que também automatiza estórias. Te dá a possibilidade de implementar as estórias com o uso de bibliotecas de automação de testes.

Adicione ao “dependencies” do composer.json:

```
"behat/behat": "3.4.1"
```



Júlio de Lima
iam@juliodelima.com.br

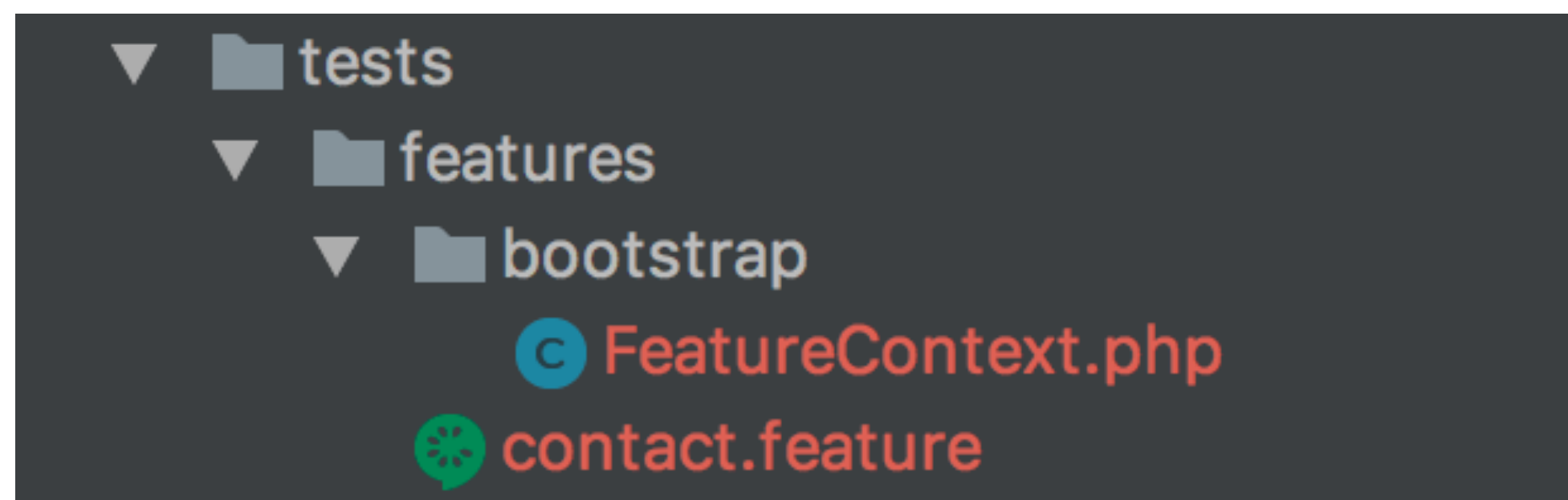
Iniciando o projeto

Tudo tem um começo, né?

Dentro do diretório test:

```
../vendor/bin/behat --init
```

Estrutura:



A definição da funcionalidade



Júlio de Lima
iam@juliodelima.com.br

// Arquivo: `contact.feature`

Feature: Contact

In order to change info from my profile

As a ordinary user

I need to be able to manage my contacts

Rules:

- I can add phones
- I can add telephones
- I can add duplicated data

Scenario: Adding a Phone

Given I am logged in

And I am seeing my contact data

When I add a new phone

Then I see It was my phone

As importações iniciais



Júlio de Lima
iam@juliodelima.com.br

```
use Behat\Behat\Context\Context,  
    Facebook\WebDriver\Remote\RemoteWebDriver,  
    Facebook\WebDriver\Remote\DesiredCapabilities,  
    Facebook\WebDriver\WebDriverBy,  
    PHPUnit\Framework\Assert;
```

Construindo e preparando o browser



Júlio de Lima
iam@juliodelima.com.br

```
private $driver;

/**
 * Initializes context.
 *
 * Every scenario gets its own context instance.
 * You can also pass arbitrary arguments to the
 * context constructor through behat.yml.
 */
public function __construct()
{
    $this->driver = RemoteWebDriver::create(
        "http://localhost:4444",
        DesiredCapabilities::chrome()
    );
    $this->driver->manage()->window()->maximize();
    $this->driver->manage()->timeouts()->implicitlyWait(5);
    $this->driver->get('http://www.juliodelima.com.br/taskit');
}
```



Júlio de Lima
iam@juliodelima.com.br

Executando os cenários

Eita, agora eu fiquei impressionado

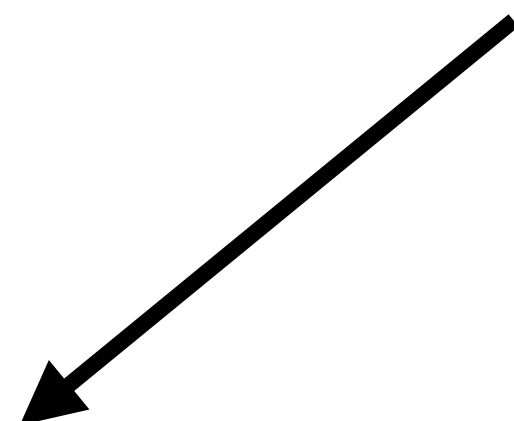
Dentro do diretório test:

```
../vendor/bin/behat
```

Resultado:

```
/**
 * @Given I am logged in
 */
public function iAmLoggedIn()
{
    throw new PendingException();
}
```

Given I am logged in





Júlio de Lima
iam@juliodelima.com.br

```
/**
 * @Given I am logged in
 */
public function iAmLoggedIn()
{
    $this->driver->findElement(WebDriverBy::linkText('Sign in'))->click();
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox input[name="login"]'))
        ->sendKeys('unicesumar');
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox input[name="password"]'))
        ->sendKeys('123456');
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox a'))->click();
    $actual = $this->driver->findElement(WebDriverBy::className('me'))->getText();

    Assert::assertContains('Unicesumar', $actual);
}
```

← **Given** I am logged in

Parâmetros inline



Júlio de Lima
iam@juliodelima.com.br

// Arquivo: `contact.feature`

Feature: Contact

In order to change info from my profile
As a ordinary user
I need to be able to manage my contacts

Rules:

- I can add phones
- I can add telephones
- I can add duplicated data

Scenario: Adding a Phone

Given I am logged in as "unicesumar"

And I am seeing my contact data

When I add a new phone

Then I see It was my phone



Júlio de Lima
iam@juliodelima.com.br

```
/**
 * @Given I am logged in as :login
 */
public function iAmLoggedInAs($login)
{
    $this->driver->findElement(WebDriverBy::linkText('Sign in'))->click();
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox input[name="login"]'))
        ->sendKeys($login);
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox input[name="password"]'))
        ->sendKeys('123456');
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox a'))->click();
    $actual = $this->driver->findElement(WebDriverBy::className('me'))->getText();

    Assert::assertContains('Unicesumar', $actual);
}
```

← **Given** I am logged in as “unicesumar”

Parâmetros de step

// Arquivo: contact.feature



Júlio de Lima
iam@juliodelima.com.br

Feature: Contact

In order to change info from my profile

As a ordinary user

I need to be able to manage my contacts

Rules:

- I can add phones
- I can add telephones
- I can add duplicated data

Scenario: Adding a Phone

Given I am logged in as:

login	password
unicesumar	123456

And I am seeing my contact data

When I add a new phone

Then I see It was my phone



Júlio de Lima
iam@juliodelima.com.br

```
use Behat\Gherkin\Node\TableNode;
```

```
/**
 * @Given I am logged in as:
 */
public function iAmLoggedInAs (TableNode $table)
{
    $hashUsuario = $table->getHash()[0];
    $this->driver->findElement(WebDriverBy::linkText('Sign in'))->click();
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox input[name="login"]'))
        ->sendKeys($hashUsuario['login']);
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox input[name="password"]'))
        ->sendKeys($hashUsuario['password']);
    $this->driver->findElement(WebDriverBy::cssSelector('#signinbox a'))->click();
    $actual = $this->driver->findElement(WebDriverBy::className('me'))->getText();

    Assert::assertContains('Unicesumar', $actual);
}
```

Given I am logged in as:

login	password
unicesumar	123456

Data-Driven usando Scenario Outline



Júlio de Lima
iam@juliodelima.com.br

// Arquivo: contact.feature

[...]

Scenario Outline: Adding a Phone

~~Given~~ I am logged in as:

login	password
-------	----------

<login>	<password>
---------	------------

~~And~~ I am seeing my contact data

When I add a new phone

Then I see It was my phone

Examples:

login	password
unicesumar	123456
julio0001	123456



Júlio de Lima
iam@juliodelima.com.br

```
Scenario Outline: Adding a Phone # features/contact.feature:11
  Given I am logged in as:       # FeatureContext::iAmLoggedInAs ()
    | login    | password  |
    | <login>   | <password> |
  And I am seeing my contact data
  When I add a new phone
  Then I see It was my phone
```

Examples:

```
| login      | password |
| unicesumar | 123456   |
| julio0001  | 123456   |
```

Failed asserting that 'Hi, Julio' contains "Unicesumar".

--- Failed scenarios:

features/contact.feature:22

2 scenarios (1 failed, 1 undefined)
8 steps (1 passed, 1 failed, 6 undefined)
0m16.22s (8.70Mb)

Categorizando



Júlio de Lima
iam@juliodelima.com.br

// Arquivo: contact.feature

[...]

@smoke

Scenario Outline: Adding a Phone

Given I am logged in as:

login	password	
<login>	<password>	

And I am seeing my contact data

When I add a new phone

Then I see It was my phone

Examples:

login	password	
unicesumar	123456	
julio0001	123456	



Júlio de Lima
iam@juliodelima.com.br

Executando os cenários categorizados

Eita, agora eu fiquei impressionado

Dentro do diretório test:

```
../vendor/bin/behat --tags=@smoke
```



Júlio de Lima
iam@juliodelima.com.br

Salvando resultados em arquivo

As vezes é necessário gerar relatório, não é?

Dentro do diretório test:

```
../vendor/bin/behat --format pretty --out report.txt
```



Júlio de Lima
iam@juliodelima.com.br

Workshop Apache JMeter

Vamos testar a performance das aplicações web!

Apache JMeter é uma ferramenta para simular usuários virtuais simultâneos, buscando validar que a aplicação suporta grande carga de utilização.

Download:

http://jmeter.apache.org/download_jmeter.cgi

Vídeo tutorial:

<https://www.youtube.com/watch?v=jtgyUlggO9o>



Júlio de Lima
iam@juliodelima.com.br

Dúvidas?

Ah, vai, fala que tem alguma!