無料電子ブック

学習

protractor

Free unaffiliated eBook created from
Stack Overflow contributors.

#protractor

1: のい

は、AngularJSアプリケーションのためのエンドツーエンドのテストフレームワークです。

は、Selenium WebDriverのりにラッパーにされているですので、Selenium WebDriverでできるすべてのがまれています。さらに、は、AngularJSアプリケーションをするのにについくつかのしいロケータとをします。としては、waitForAngular、By.binding、By.repeater、By.textarea、By.model、WebElement.all、WebElement.evaluateなどがあります。

バージョン

バージョン	リリースデータ
0.0.1	2016-08-01

Examples

のとセットアップWindowsの

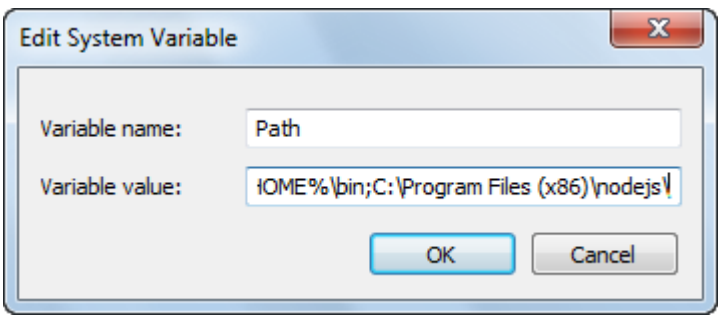
は、インストールにのをインストールするがあります。

- Java JDK 1.7
- Node.js v4

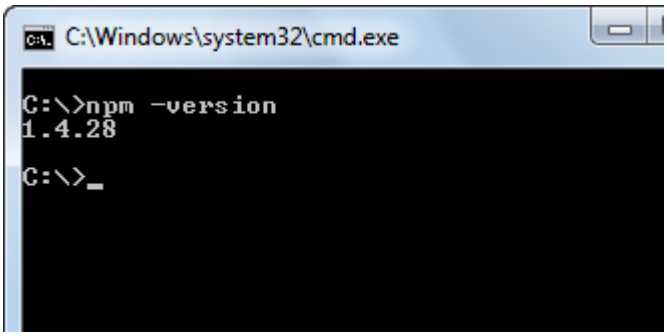
インストール

Node.jsをのURLからダウンロードしてインストールします。https : //nodejs.org/ja/

Node.jsのインストールがしたかどうかをするには、をしてください。システムの「パス」がにされます。



コマンドプロンプトで`npm -version`コマンドをすることでじことをすることもできます。これにより、インストールされているバージョンがされます。



きしは、ローカルまたはグローバルの2つのでインストールできます。

のフォルダまたはプロジェクトのディレクトリにをインストールできます。プロジェクトディレクトリにインストールするは、するたびにそのからのみするがあります。

プロジェクトディレクトリにローカルにインストールするには、プロジェクトフォルダにし、コマンドをします

```
npm install protractor
```

をグローバルにインストールするには、のコマンドをします。

```
$ npm install -g protractor
```

これにより2つのコマンドラインツール、 `protractor`と`webdriver-manager`がインストールされます。
`protractor --version`をして、がにインストールされたことをします。

`webdriver-manager`は、ブラウザドライバのバイナリをダウンロードし、セレンサーバをするためにされます。
ブラウザドライバのバイナリをダウンロードするには

```
$ webdriver-manager update
```

Seleniumサーバーをのようにします。

```
$ webdriver-manager start
```

Internet Explorerのドライバをダウンロードするには、コマンドプロンプトで`webdriver-manager update --ie`コマンドをします。これはあなたのseleniumフォルダにIEDriverServer.exeをダウンロードします

をったのテスト

は、のテスト、スペックテストコードファイル、およびコンフィギュレーションファイルをするために2つのファイルしかとしません。スペックファイルにはテストコードがまれ、もう1つには

スペックファイルパス、ブラウザの、テストURL、フレームワークパラメータなどのがまれています。のテストをくには、セレンサーバのアドレスとスペックファイルパスのみをします。ブラウザ、タイムアウト、フレームワークはデフォルトにめられます。

のデフォルトのブラウザはChromeです。

conf.js - ファイル

```
exports.config = {
  seleniumAddress: 'http://localhost:4444/wd/hub',
  specs: ['spec.js']
};
```

spec.js - テストコードファイル

```
describe('first test in protractor', function() {
  it('should verify title', function() {
    browser.get('https://angularjs.org');

    expect(browser.getTitle()).toEqual('AngularJS − Superheroic JavaScript MVW Framework');
  });
});
```

---

seleniumAddress - webdriverサーバーがされているサーバーへのパス。

specs - テストファイルのパスをむ。のパスは、コンマりのでできます。

describe - Jasmineフレームワークからの。staを`describe`

テストをく

しいコマンドラインまたはターミナルウィンドウをき、テストのクリーンフォルダをします。

は、するには2つのファイル、ファイルとファイルがです。

AngularJSウェブサイトのToDoリストのサンプルにし、しいToDoをリストにするなテストからめましょう。

を`spec.js`コピーする

'anglejsホームページtodoリスト'、function{it 'todo'をするはずです、function{browser.get ' https://angularjs.org ';

```
element(by.model('todoList.todoText')).sendKeys('write first protractor test');
element(by.css('[value="add"]')).click();

var todoList = element.all(by.repeater('todo in todoList.todos'));
expect(todoList.count()).toEqual(3);
expect(todoList.get(2).getText()).toEqual('write first protractor test');
```

```
// You wrote your first test, cross it off the list
todoList.get(2).element(by.css('input')).click();
var completedAmount = element.all(by.css('.done-true'));
expect(completedAmount.count()).toEqual(2);});});
```

ランニン グテスト

は、describeのわりにfdescribeをして、テストのグループをにできます。

```
fdescribe('first group',()=>{
    it('only this test will run',()=>{
        //code that will run
    });
});
describe('second group',()=>{
    it('this code will not run',()=>{
        //code that won't run
    });
});
```

は、itのわりにfitをしてグループのテストをにできます。

```
describe('first group',()=>{
    fit('only this test will run',()=>{
        //code that will run
    });
    it('this code will not run',()=>{
        //code that won't run
    });
});
```

fdescribeにfitがない、すべてのitがされます。しかし、fitはじdescribeまたはfdescribeでitびしをブロックします。

```
fdescribe('first group',()=>{
    fit('only this test will run',()=>{
        //code that will run
    });
    it('this code will not run',()=>{
        //code that won't run
    });
});
```

たとえfitがfdescribeのわりにdescribeにあっても、それはされます。また、fitをまないfdescribeのitもされます。

```
fdescribe('first group',()=>{
    it('this test will run',()=>{
        //code that will run
    });
    it('this test will also run',()=>{
        //code that will also run
    });
});
```

```
describe('second group',()=>{
    it('this code will not run',()=>{
        //code that won't run
    });
    fit('this code will run',(){
        //code that will run
    });
});
```

のテスト

は、をとしてすることをにする。これはがテストをしないで、わりにすることをします

```
Pending:
1) Test Name
Temporarily disabled with xit
```

xdescribeでにしたは、のようになります。

```
Pending:
1) Test Name
No reason given
```

みわせ

- xdescribeのxitはxitをします。
- fdescribeのxitはとしてとしてわれます。
- xdescribeのfitはされ、のテストはもしません。

エンターープライズアプリケーションのE2Eテスト

のとセットアップ

ステップ1 ここからNodeJSをダウンローードしてインストーールします。ノードのバージョンがインストーールされていることをしてください。ここでは、ノードv7.8.0をしています。セレンをするには、Java Development KitJDKがインストーールされているがあります。

ステップ2 をき、のコマンドをしてをグローバルにインストーールします。

```
npm install -g protractor
```

これはとウェブドライバマネージャのような2つのツールをインストーールします。 `protractor -version.`インストーールをするには、 `protractor -version.`コマンドをし`protractor -version.` Protractorがにインストーールされているは、インストーールされているバージョンつまり、バージョン5.1.1がされます。それのは、インストーールをするがあります。ステップ3Webdriverマネージャをして、なバイナリをダウンローードします。

```
webdriver-manager update
```

ステップ4のコマンドをすると、Selenium Serverがします。このステップでは、バックグラウンドでWebドライバマネ―ジャがされ、をしてされるテストがされます。

webdriver-manager startサーバーのステ―タスにするは
`http://localhost:4444/wd/hub/static/resource/hub.html`.

をったのテストケ―スの

テストケ―スのにるに、コンフィギュレ―ションファイルとスペックファイルの2つのファイルをするがあります。

ファイル

```
//In conf.js
exports.config = {
    baseUrl: 'http://localhost:8800/adminapp',
    seleniumAddress: 'http://localhost:4444/wd/hub',
    specs: ['product/product_test.js'],
    directConnect : true,
    capabilities :{
        browserName: 'chrome'
    }
}
```

ファイルでされるのな

baseUrl - テストのアプリケ―ションのベ―スURL。

seleniumAddress - にしているSelenium Serverにします。

specs - スペックファイルの

directConnect true - ブラウザのドライバにします。

- のブラウザでテストするは、capabilitiesオプションをします。のブラウザでテストするは、multiCapabilitiesをします。

ここからよりくのオプションをつけることができます。らは、すべてのなをそのとともにしてきました。

スペックファイル

```
//In product_test.js

    describe('Angular Enterprise Boilerplate', function() {
      it('should have a title', function() {
        browser.get('http://localhost:8800/adminapp');
        expect(browser.getTitle()).toEqual('Angular Enterprise Boilerplate');
      });
    });
```

specファイルでされるのな

デフォルトでは、Protractorはテストインタフェースとしてジャスミンフレームワークをします。"と 'それはジャスミンフレームワークからのものです。ここからをることができます。のテストケースをする

テストケースをするに、Webdriverマネージャとアプリケーションがのなるタブでしていることをしてください。

は、のようにテストをします。

```
Protractor app/conf.js
```

クロムブラウザがあなたのアプリケーションのURLでき、それをじるのがえます。テストは1テスト、1アサーション、0でなければなりません。

ブラボーあなたはのテストケースをにします。

オンラインでのいをむ https://riptutorial.com/ja/protractor/topic/933/のい

2: browser.waitでにする

Examples

browser.sleepとbrowser.wait

タイミングのをうには、「クイック」な`browser.sleep(<timeout_in_milliseconds>)`をくことがでです`browser.sleep(<timeout_in_milliseconds>)`。

は、いつかはするということです。するスリープタイムアウトには、ネットワークやパフォーマンスなどので、あるでページがみまれるやがされるまでにがかかるなど、ゴールデン/ジェネリックルールはありません。、あなたははにはにってわるだろう。

、 `browser.wait()`はがなります。 Protractor / WebDriverJSのExpected Conditionをして、のがtrueとされるまでします。 はをにし、のがであるか、なタイムアウトにしたでする。

みみのExpected Conditionsはされていますが、のカスタムをしてすることもできます ここでのサンプル。

オンラインでbrowser.waitでにするをむ https://riptutorial.com/ja/protractor/topic/8297/browser-wait--でにする

3: CSSセレクタ

- by.css 'css-selector'
- by.id 'id'
- by.model 'model'
- by.binding 'binding'

パラメーター

パラメータ	
CSSセレクター	'.class-name'ようなcssセレクタは、クラスのにあるをします。
id	domのID
モデル	DOMにされるモデル
	のにバインドするためにされるバインディングの

cssセレクタをくには

CSSセレクタをくためのもなはdomのclassとidです。インスタンスの、html domはのようにえます

```
<form class="form-signin">
    <input type="text" id="email" class="form-control" placeholder="Email">
    <input type="password" id="password" class="form-control" placeholder="Password">
    <button class="btn btn-block" id="signin-button" type="submit">Sign in</button>
</form>
```

に、メールフィールドをするには、のようにcssセレクタをします。

1. クラスの CSSセレクターのクラスはドットでまります。そのためのCSSセレクターは、`.form-control`ようになります。

   ```
   by.css('.form-control')
   ```

`form-control`クラスはのでされるので、ロケータののがじます。したがって、このようなでは、idがなは、にクラスのわりにidをするがよいでしょう。

2. **ID**の cssセレクターのidは、ハッシュでまります。したがって、メールのIDをするCSSセレクタは、のようにされます。

   ```
   by.css('#email')
   ```

3. のクラスの domにのクラスがあるは、クラスのみわせでcssセレクタとしてできます。たと

えば、DOMがのような

```
<input class="username-class form-control">
// css selector using multiple classes
by.css('.username-class.form-control')
```

4. ののタグの タグやそののをしてCSSセレクターをするなは、 `tagname[attribute-type='attribute-vallue']`です。 したがって、のに、サインインボタンのCSSロケータをのようにできます。

```
by.css("button[type='submit']") //or
by.css("button[id='signin-button']")
```

Examples

$と$$のCSSセレクタロケータのショートカット

APIにより、CSSロケータはjQueryのようなショートカット`$()`をできます。

の**CSS**ロケータ

```
element(by.css('h1.documentation-text[ng-bind="title"]'));
element(by.css('[ng-click="submit"]'));
```

ショートカット`$()` **CSS**エレメントロケータ

```
$('h1.documentation-text[ng-bind="title"]');
$('[ng-click="submit"]');
```

ロケータのでのをつけるには、 ショートカット`$$()`ます。

の**CSS**ロケータ

```
element.all(by.css('h1.documentation-text[ng-bind="title"]'));
element.all(by.css('[ng-click="submit"]'));
```

ショートカット`$$()` **CSS**エレメントロケータ

```
$$('h1.documentation-text[ng-bind="title"]');
$$('[ng-click="submit"]');
```

ロケータの

のロケータは、HTML DOMにするアクションをするためにされます。でされるもでのロケータは、CSS、ID、モデル、バインディングです。たとえば、にされるロケータはのとおりです。

```
by.css('css-selector')
by.id('id')
```

なHTMLでをする

なHTMLでをするには、cssロケータパターン[attribute = value]

```
//selects the first element with href value '/contact'
element(by.css('[href="/contact"]'));

//selects the first element with tag option and value 'foo'
element(by.css('option[value="foo"]'));

//selects all input elements nested under the form tag with name attribute 'email'
element.all(by.css('form input[name="email"]'));
```

されたをむHTMLでをする

されたをむHTMLでをするには、cssロケータパターン[attribute * = value]をします。

```
//selects the first element with href value that contains'cont'
element(by.css('[href*="cont"]'));

//selects the first element with tag h1 and class attribute that contains 'fo'
element(by.css('h1[class*="fo"]'));

//selects all li elements with a title attribute that contains 'users'
element.all(by.css('li[title*='users']'));
```

オンラインでCSSセレクタをむ https://riptutorial.com/ja/protractor/topic/1524/cssセレクタ

4: ページオブジェクト

き

ページオブジェクトは、コードのをなくし、メンテナンスをにし、をさせるデザインパターンです。

Examples

のページオブジェクト

```
/* save the file in 'pages/loginPage'
var LoginPage = function(){

};

/*Application object properties*/
LoginPage.prototype = Object.create({}, {
    userName: {
        get: function() {
            return browser.driver.findElement(By.id('userid'));
        }
    },
    userPass: {
        get: function() {
            return browser.driver.findElement(By.id('password'));
        }
    },
    submitBtn: {
        get: function() {
            return browser.driver.findElement(By.id('btnSubmit'));
        }
    }
});

/* Adding functions */
LoginPage.prototype.login = function(strUser, strPass) {
    browser.driver.get(browser.baseUrl);
    this.userName.sendKeys(strUser);
    this.userPass.sendKeys(strPass);
    this.submitBtn.click();
};

module.exports = LoginPage;
```

のページのオブジェクトファイルをテストでしましょう。

```
var LoginPage = require('../pages/loginPage');
describe('User Login to Application', function() {
    var loginPage = new LoginPage();

    beforeAll(function() {
```

```
        loginPage.login(browser.params.userName, browser.params.userPass);
    });

    it('and see a success message in title', function() {
        expect(browser.getTitle()).toEqual('Success');
    });

});
```

オンラインでページオブジェクトをむ https://riptutorial.com/ja/protractor/topic/9747/ページオブジェクト

5: デバッガ

- browser.pause
- browser.debugger

このセクションでは、テストをデバッグするについてします。

Examples

browser.pauseをする

`pause()`メソッドは、がコードをデバッグするためにするもなソリューションの1つです。コードをしてをするコードにするがあります。がになると、

1. あなたは`c`タイプCをってすることができます。このコマンドをするはしてください。cをすのがれたは、アサーションライブラリからのタイムアウトエラーがするがあるため、このコマンドをなくするがあります。

2. モードにるには`repl`とします。インタラクティブモードは、ブラウザのコマンドをブラウザのオープンインスタンスにするためにされます。えば、モードではのようなコマンドをすることができます

```
> element(by.css('#username')).getText()
> NoSuchElementError: No element found using locator: by.username("#username")
```

のコマンドのがされ、コマンドのしさをることができます。

*Chrome Dev Tools*をいているは、*Dev Tools*がいているときに*ChromeDriver*がしないため、テストをするに*Chrome Dev Tools*をするがあります。

3. `CTRL+C`をしてデバッグモードをすると、なCTRL + Cコマンドをしてデバッグモードからをりすことができます。

```
 it('should pause when we use pause method', function () {
    browser.get('/index.html');

    var username = element(by.model('username'));
    username.sendKeys('username');
    browser.pause();

    var password = element(by.model('password'));
    password.sendKeys('password');
    browser.pause();
});
```

4. dをしてのデバッガーのステートメントにみます

browser.debuggerの

browser.debuggerをしてをすることができます。コードにのをすることができ、するようにしないり、そののにをします。

デバッガモードでテストをするには、のようなコマンドをするがあります。

```
    `protractor debug <configuration.file.js>`
```

cをし、ブレークポイントのにけるか、するためにnextのフローののにcommand.The、のコマンドのを。

でされるデバッガはノードデバッガをし、のでをします。たとえば、のコードでは、username.sendKeys('username')がされたときにbrowser.debugger()がびされます。

これらはタスクであるため、スペックのデフォルトタイムアウトをやすがあります。それのは、デフォルトのタイムアウトがスローされます。

```
it('should pause when we use pause method', function () {
browser.get('/index.html');

var username = element(by.model('username'));
username.sendKeys('username');
browser.debugger();

var password = element(by.model('password'));
password.sendKeys('password');
});
```

replモードにるには、

```
debug > repl
> element(by.model('abc')).sendKeys('xyz');
```

これにより、のタスクとしてsendKeysコマンドがされ、デバッガにされます。

Port no.をすることができますPort no.らはデバッガメソッドにポートをすだけでスクリプトをデバッグしたいとっています。

```
browser.debugger(4545); //will start the debugger in port 4545
```

debugger()メソッドは、クライアントサイドをからブラウザにし、をフェッチするためにブラウザコンソールでコマンドをすることはほとんどできません。クライアントサイドスクリプトをするの1つはのとおりです。

```
window.clientSideScripts.findInputs('username');
```

オンラインでデバッガをむ https://riptutorial.com/ja/protractor/topic/3910/デバッガ

6: でのアプリをテストする

き

はアプリケーションをテストするためにられています。しかし、であればでのアプリケーションをテストすることはです。

Examples

でアプリをテストするためにな

`driver`わりに`browser.driver`をする

`browser.driver.ignoreSynchronization = true`し`browser.driver.ignoreSynchronization = true`

は、がウェブページににロードされるのをっているため、をします。しかし、たちのページはがないので、テストがタイムアウトしてするまで、「」がロードされるのをっています。だから、たちは、「」をたないように、ににえるがあります

オンラインでのアプリをテストするをむ https://riptutorial.com/ja/protractor/topic/8830/でのアプリをテストする

7: のXPathセレクタ

Examples

をしてDOMをする

は、CSS、モデル、バインディングセレクタとはに、xpath Viewをしてをすることもできます

```
<ul>
<li><a href='http://www.google.com'>Go to google</a></li>
</ul>
```

コード

```
var googleLink= element(by.xpath('//ul/li/a'));
expect(element.getText()).to.eventually.equal('Go to google','The text you mention was not
found');
```

のをつの

XPathセレクタをして、クラス、ID、タイトルなどののをつをできます。

クラス

```
<div class="HakunaMatata"> Hakuna Matata </div>
```

コード

```
var theLionKing= element(by.xpath('//div[@class="HakunaMatata"]'));
expect(theLionKing.getText()).to.eventually.equal('Hakuna Matata', "Text not found");
```

しかし、はのクラスをつことができます。このようなは、「む」をできます

```
<div class="Hakuna Matata"> Hakuna Matata </div>
```

コード

```
var theLionKing= element(by.xpath('//div[conatins(@class,"Hakuna")]'));
expect(theLionKing.getText()).to.eventually.equal('Hakuna Matata', "Text not found");
```

のコードは、class = "HakunaMatata"とclass = "Hakuna Matata"のをむをします。テキストがスペースでられたリストのであるは、のをできます。

```
var theLionKing= element(by.xpath('//div[contains(concat(' ',normalize-space(@class),' '),
"Hakuna")]'));
expect(theLionKing.getText()).to.eventually.equal('Hakuna Matata', "Text not found");
```

idによって

IDは、をするためにできるもでなロケーータのままです。

```
<div id="HakunaMatata">Hakuna Matata</div>
```

コード

```
var theLionKing= element(by.xpath('//div[@id="HakunaMatata"]');
expect(theLionKing.getText()).to.eventually.equal('Hakuna Matata', "Text not found");
```

クラスとに、containsをして、されたテキストをむをつけることができます。

そのの

のタイトルをつをする

ビューー

```
<div title="Hakuna Matata">Hakuna Matata</div>
```

コード

```
var theLionKing= element(by.xpath('//div[@title="Hakuna Matata"]'));
expect(theLionKing.getText()).to.eventually.equal('Hakuna Matata', "Text not found");
```

のテキストをつをする

ビューー

```
<div class="Run Simba Run">Run Simba</div>
```

コード

```
var runSimba= element(by.xpath('//div[text()="Run Simba"]'));
```

のテキストベースのとに、containsをして、なをむtextをつをすることができます。

ビューー

```
<div class="Run Simba Run">Run Simba,run</div>
```

コード

```
 var runSimba= element(by.xpath('//div[contains(text(),"Run Simba")]'));
expect(runSimba.getText()).to.eventually.equal('Run Simba, run', "Text not found"); //true
```

のをつをする

ビュー

```
<input type="text" name="FullName"></input>
```

コード

```
var fullNameInput= element(by.xpath('//input[@name="FullName"]'));
fullNameInput.sendKeys("John Doe");
```

オンラインでのXPathセレクタをむ https://riptutorial.com/ja/protractor/topic/7205/のxpathセレクタ

8: ファイル

き

ファイルには、スクリプトのにするがまれています。ここでは、いくつかのバリエーションをしようとします。

Examples

Simple Configファイル - Chrome

```
var config = {};
var timeout = 120000;

config.framework = 'jasmine2';
config.allScriptsTimeout = timeout;
config.getPageTimeout = timeout;
config.jasmineNodeOpts.isVerbose = true;
config.jasmineNodeOpts.defaultTimeoutInterval = timeout;
config.specs = ['qa/**/*Spec.js'];
config.browserName = 'chrome';

exports.config = config;
```

きのファイル - Chrome

```
var config = {};
var timeout = 120000;

config.framework = 'jasmine2';
config.allScriptsTimeout = timeout;
config.getPageTimeout = timeout;
config.jasmineNodeOpts.isVerbose = true;
config.jasmineNodeOpts.defaultTimeoutInterval = timeout;
config.specs = ['qa/**/*Spec.js'];
config.capabilities = {
    browserName: 'chrome',
    'chromeOptions': {
        'args': ['start-minimized', 'window-size=1920,1080']
    }
};

exports.config = config;
```

ファイルshardTestFiles - Chrome

このでは、2つのブラウザインスタンスでのスペックファイルをしてできます。のテストをするのにちます。にじてmaxInstancesをします。

テストがしていることをしてください。

```
var config = {};
var timeout = 120000;

config.framework = 'jasmine2';
config.allScriptsTimeout = timeout;
config.getPageTimeout = timeout;
config.jasmineNodeOpts.isVerbose = true;
config.jasmineNodeOpts.defaultTimeoutInterval = timeout;
config.specs = ['qa/**/*Spec.js'];
config.capabilities = {
    browserName: 'chrome',
    shardTestFiles: true,
    maxInstances: 2,
    'chromeOptions': {
        'args': ['start-minimized', 'window-size=1920,1080']
    }
};

exports.config = config;
```

ファイルマルチエミュレート - クロム

```
var config = {};
var timeout = 120000;

config.framework = 'jasmine2';
config.allScriptsTimeout = timeout;
config.getPageTimeout = timeout;
config.jasmineNodeOpts.isVerbose = true;
config.jasmineNodeOpts.defaultTimeoutInterval = timeout;
config.specs = ['qa/**/*Spec.js'];
config.multiCapabilities = [{
        browserName: 'chrome',
        shardTestFiles: true,
        maxInstances: 2,
        'chromeOptions': {
            'args': ['start-minimized', 'window-size=1920,1080']
        }
    },
    {
        browserName: 'chrome',
        shardTestFiles: true,
        maxInstances: 1,
        'chromeOptions': {
            'args': ['show-fps-counter=true'],
            'mobileEmulation': {
                'deviceName': 'Apple iPhone 6'
            }
        }
    }
];

exports.config = config;
```

オンラインでファイルをむ https://riptutorial.com/ja/protractor/topic/9745/ファイル

9: フローと

き

Protractor / WebDriverJSには、 フローとばれるこのメカニズムがあります。これは、のキューであり、コードのをしたにちます。

Examples

フローの

のテストをえてみましょう。

```
it('should test something', function() {
  browser.get('/dashboard/');

  $("#myid").click();
  expect(element(by.model('username')).getText()).toEqual('Test');

  console.log("HERE");
});
```

のテストでは、 `console.log()`がされ、コンソールで`HERE`がされたとき、のからのコマンドはされていません。これはにのです。コマンドはとしてされ、をしてするコントロールフローにかれました。

プロミスとコントロールフローのをしてください。

オンラインでフローとをむ https://riptutorial.com/ja/protractor/topic/8580/フローと

10: の

き

ページとやりとりできるようにするには、どのをすかをするがあります。のにされるはロケータです。は、なセレンセレクタをむだけでなく、よりでにえるロケータもえています。ただし、アプリケーションでも、のロケータをするがあります。

パラメータ

パラメータ	
セレクタ	セレクタのをするロケータに

Examples

のロケータベースのアプリケーションの

であれば、これらのロケータはアプリケーションのにしてであり、cssまたはxpathにづくロケータはにするがあるため、としてするがあります。

バインディングロケータ

```
by.binding('bind value')
```

ビュー

```
<span>{{user.password}}</span>
<span ng-bind="user.email"></span>
```

ロケータ

```
by.binding('user.password')
by.binding('user.email')
```

もサポート

```
by.binding('email')
```

なバインディングロケータ

bindingと、はされません。

```
by.exactBinding('exact bind value')
```

ビュー

```
<span>{{user.password}}</span>
```

ロケータ

```
by.exactBinding('user.password')
by.exactBinding('password') // Will not work
```

モデルロケータ

モデルディレクティブをつをします。

```
by.model('model value')
```

ビュー

```
<input ng-model="user.username">
```

ロケータ

```
by.model('user.username')
```

ボタンテキストロケータ

テキストについてボタンをします。ボタンのテキストがにされることがされないにのみするがあります。

```
by.buttonText('button text')
```

ビュー

```
<button>Sign In</button>
```

ロケータ

```
by.buttonText('Sign In')
```

なボタンテキストロケーータ

buttonTextとていますが、がです。ボタンのテキストがにされることがされないにのみするがあります。

```
by.partialButtonText('partial button text')
```

ビューー

```
<button>Register an account</button>
```

ロケーータ

```
by.partialButtonText('Register')
```

リピーータロケーータ

リピーータをつをします。

```
by.repeater('repeater value')
```

ビューー

```
<tbody ng-repeat="review in reviews">
    <tr>Movie was good</tr>
    <tr>Movie was ok</tr>
    <tr>Movie was bad</tr>
</tbody>
```

ロケーータ

```
by.repeater('review in reviews')
```

もサポーート

```
by.repeater('reviews')
```

なリピーータロケーータ

repeaterにていますが、がされていません

```
by.exactRepeater('exact repeater value')
```

ビュー

```
<tbody ng-repeat="review in reviews">
    <tr>Movie was good</tr>
    <tr>Movie was ok</tr>
    <tr>Movie was bad</tr>
</tbody>
```

ロケータ

```
by.exactRepeater('review in reviews')
by.exactRepeater('reviews') // Won't work
```

CSSとテキストロケータ

のテキストコンテンツをするCSSロケータ。

```
by.cssContainingText('css selector', 'text of css element')
```

ビュー

```
<ul>
    <li class="users">Mike</li>
    <li class="users">Rebecca</li>
</ul>
```

ロケータ

```
by.cssContainingText('.users', 'Rebecca') // Will return the second li only
```

オプションロケータ

オプションディレクティブをつをします。

```
by.options('options value')
```

ビュー

```
<select ng-options="country.name for c in countries">
    <option>Canada</option>
    <option>United States</option>
    <option>Mexico</option>
</select>
```

ロケータ

```
by.options('country.name for c in countries')
```

ディープCSSロケータ

シャドーDOMにがるCSSロケータ

```
by.deepCss('css selector')
```

ビュー

```
<div>
    <span id="outerspan">
    <"shadow tree">
        <span id="span1"></span>
        <"shadow tree">
            <span id="span2"></span>
        </>
    </>
</div>
```

ロケータ

```
by.deepCss('span') // Will select every span element
```

ロケータの

ロケータはそれがですることができるをさず、にをつけるをにするだけのです。

にアクセスするには、のをします。

```
element(locator);
element.all(locator);
```

は、アクションがされるまでにはアクセスされません。つまり、は、にしてgetTextなどのアクションがびされたときに、にをするだけです。

ロケータをして1つののみをするは、をし`element`。ロケータがのをしている、`element`はにつかった`element`をします。`element`は`ElementFinder`します。

ロケータをしてのをする、`element.all`はつかったすべてのをします。`element.all`は`ElementArrayFinder`し、のすべてのには、`map`などのさまざまなメソッドをしてアクセスできます。

```
element.all(locator).map(function(singleElement) {
```

```
        return singleElement.getText();
    }
});
```

チェインロケータ

のロケータをさせて、なアプリケーションのをすることができます。`locator`オブジェクトをチェインすることはできませんが、 `ElementFinders`チェーン`ElementFinders`するがあります。

```
element(by.repeater('movie in movies').element(by.linkText('Watch Frozen on Netflix')
```

できるチェーンのにはありません。に、ロケータにじて、の`ElementFinder`または`ElementArrayFinder`します。

オンラインでのをむ https://riptutorial.com/ja/protractor/topic/10825/の

クレジット

S. No		Contributors
1	のい	Bhoomi Bhalani, Community, Devmati Wadikar, Manuli Piyalka, olyv, Peter Stegnar, Praveen, Priyanshu Shekhar, SilentLupin, sonhu, Stephen Leppik
2	browser.waitでにする	alecxe
3	CSSセレクタ	alecxe, Droogans, leon, Priyanshu Shekhar, sonhu
4	ページオブジェクト	Barney, Suresh Salloju
5	デバッガ	Devmati Wadikar, Priyanshu Shekhar, Ram Pasala, Sakshi Singla, Stephen Leppik
6	でのアプリをテストする	Sakshi Singla
7	のXPathセレクタ	Shubhang
8	ファイル	Barney
9	フローと	alecxe
10	の	Sébastien Dufour-Beauséjour