

JavaScript用の振る舞い駆動開発（BDD）テストフレームワーク

Jasmine



佐野 尚之



アジェンダ

- ・ Jasmineの特徴 5
- ・ Jasmineの使用 6
- ・ 参考情報 11



本ドキュメントのライセンスについて

この作品は、クリエイティブ・コモンズのAttribution 3.0 Unportedライセンスの下でライセンスされています。

この使用許諾条件を見るには、<http://creativecommons.org/licenses/by/3.0/>をチェックするか、クリエイティブ・コモンズに郵便にてお問い合わせください。

住所は：171 Second Street, Suite 300, San Francisco, California 94105, USA です。



原作者のクレジット（氏名、作品タイトルとURL）を表示することを守れば、改変はもちろん、営利目的での二次利用も許可される最も自由度の高いCCライセンス。

<http://creativecommons.jp/>





変更履歴

■第1版・・・2013/05/06

本ドキュメントは、オープンソースの「LibreOffice 4.0.1.2」を使用して作成。



Jasmineの特徴

- Ruby On Railsのプロジェクトでも利用できる。
- JavaScriptだけのテストができる Standalone バージョンもある。
- 値が正しいかのテスト、例外の有無のテスト、関数呼び出しが行われたかどうかのテストなど、さまざまなテストができる。

★前提条件

今回は「<https://github.com/pivotal/jasmine/downloads>」から、jasmine-standalone-1.3.1.zipをダウンロードして、ローカルPCの任意の場所に解凍済みであるという前提で説明します。



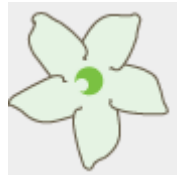
Jasmineの使用例 (1/5)

```
デスクトップ¥jasmine-standalone-1.3.1¥src¥testcode.js - sakura 2.0.5.0
ファイル(E) 編集(E) 変換(C) 検索(S) ツール(I) 設定(O) ウィンドウ(W) ヘルプ(H)

1 function trim(argValue) ←
2 { ←
3   // [ □]内には半角スペースと全角スペースを記述 ←
4   return String(argValue).replace(/^[ □]*/gim, "").replace(/[ □]*$/gim, ""); ←
5 } ←
6 ←
7 var MyClass; ←
8 MyClass = (function() { ←
9   // コンストラクタ ←
10  function MyClass(name) { ←
11    this.name = name; ←
12  } ←
13  // ゲッター ←
14  MyClass.prototype.getName = function() { ←
15    return this.name; ←
16  } ←
17  // セッター ←
18  MyClass.prototype.setName = function(name) { ←
19    this.name = name; ←
20  } ←
21  return MyClass; ←
22 })(); ←
[EOF]
```

23行 1桁 CRLF UTF-8 REC 挿入

テスト対象のjsファイル
(ローカルに解凍したフォルダ¥srcに作成)



Jasmineの使用例 (2/5)

```
デスクトップ¥jasmine-standalone-1.3.1¥spec¥TrimTest.js -...
ファイル(E) 編集(E) 変換(C) 検索(S) ツール(I) 設定(O) ウィンドウ(W)
ヘルプ(H)

1 describe('Trimのテスト', function() ←
2 { ←
3   it('Trim(前後の半角スペース)', function() ←
4   { ←
5     expect(trim(" test ")).toEqual("test"); ←
6   }); ←
7 ←
8   it('Trim(前後の全角スペース)', function() ←
9   { ←
10    expect(trim("□test□")).toEqual("test"); ←
11  }); ←
12 ←
13   it('Trim(前後の半角と全角のスペース)', function() ←
14   { ←
15     expect(trim("□ test □")).toEqual("test"); ←
16   }); ←
17 ←
18 }); ←
EOF
```

19行 1桁 CRLF UTF-8 REC 挿入

TrimTest.js
(ローカルに解凍したフォルダ¥specに作成。UTF-8で保存してください。)



Jasmineの使用例 (3/5)

```
1 describe('MyClassのテスト', function() ←  
2 { ←  
3   it('setNameメソッドのテスト', function() ←  
4   { ←  
5     var a = new MyClass("ムリ"); ←  
6     a.setName("ダメ"); ←  
7     expect(a.getName()).toEqual("ダメ"); ←  
8   }); ←  
9   ←  
10  it('getNameメソッドのテスト', function() ←  
11  { ←  
12    var b = new MyClass("ムリ"); ←  
13    expect(b.getName()).toEqual("ムリ"); ←  
14  }); ←  
15 }); ←  
[EOF]
```

10行 22桁 CRLF U+306E UTF-8 REC 挿入

MyClassTest.js
(ローカルに解凍したフォルダ¥specに作成。UTF-8で保存してください。)

Jasmineの使用例 (4/5)

- SpecRunner.htmlの編集(ローカルに解凍したフォルダ直下)

(1) 6行目に以下のコードを追加してください。ファイル保存時に文字コードをUTF-8にすることも忘れずに


```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" ↓
2 "http://www.w3.org/TR/html4/loose.dtd"> ↓
3 <html> ↓
4 <head> ↓
5 <title>Jasmine Spec Runner</title> ↓
6 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" /> ↓
```

(2) テスト対象の外部jsファイルとテストコードが書かれているの外部jsファイルの読み込みの記述を以下の内容に変更してください。

```
10 <script type="text/javascript" src="lib/jasmine-1.3.1/jasmine-html.js"></script> ↓
11 ↓
12 <!-- include source files here... --> ↓
13 <script type="text/javascript" src="src/testcode.js"></script> ↓
14 ↓
15 <!-- include spec files here... --> ↓
16 <script type="text/javascript" src="spec/TrimTest.js"></script> ↓
17 <script type="text/javascript" src="spec/MyClassTest.js"></script> ↓
18 ↓
19 <script type="text/javascript"> ↓
```

Jasmineの使用例 (5/5)

- SpecRunner.htmlをダブルクリックして、以下の画面が表示された場合は、全てのテストが成功しています。

```
Jasmine 1.3.1 revision 1354556913 finished in 0.068s  
  
● ● ● ● ●  
  
Passing 5 specs No try/catch   
  
Trimのテスト  
  Trim(前後の半角スペース)  
  Trim(前後の全角スペース)  
  Trim(前後の半角と全角のスペース)  
  
MyClassのテスト  
  setNameメソッドのテスト  
  getNameメソッドのテスト
```



参考情報(1/2)

Jasmine

<http://pivotal.github.io/jasmine/>

Javascriptテストフレームワーク Jasmineを試す

<http://nacika.com/entry/2013/01/03/055820/>

Jasmine ~ JavaScript Test フレームワーク

<http://atmarkplant-dj.blogspot.jp/2011/09/jasmine-javascript-test.html>

QUnit はオワコン!? Jasmine を使ってみる

<http://tnakamura.hatenablog.com/entry/20120313/jasmine>

PhantomJSとJasmineで振る舞い駆動開発なJavaScriptテスト (2/3)

http://www.atmarkit.co.jp/ait/articles/1210/10/news012_2.html

Jasmine で tDiary の JavaScript をテストする

<http://www.machu.jp/diary/20120831.html#p01>

Jasmine で 継続的なJavaScriptのテストをする

<http://d.hatena.ne.jp/dice-t/20110217/1297951716>

JasmineによるJavaScriptのテスト その1

<http://blog.serverworks.co.jp/tech/2010/11/30/jasmine-tutorial-1/>



参考情報(2/2)

javascriptの自動テスト化して楽をする！

(まだβ版のjasmine-standalone-2.0.0を使ってるよ) 導入編

<http://qiita.com/items/f1ad4e9331a11be20d66>

Jasmine tutorial

<http://qiita.com/items/0b8df9f41c7542086d4e>

261: JasmineでJavaScriptのテスト

<http://ja.asciicasts.com/episodes/261-testing-javascript-with-jasmine>