Behavior-Driven Development

Jasmine

By Vijay Shivakumar

Seems like a lot of work
just for some lousy tests...

Enforces Good Design Practice
Flexible
Updatable / Scalable
Reusable

Tools available

Jasmine

QUnit

Mocha

etc,.

What is happens in BDD

Write your tests

Watch them fail

Refractor your code

Make them pass

Repeat

Organizing your tests

The describe block

```
describe("Name for the block", function(){
  // your test code;
})
```

```
describe("Name for the block", function(){
      describe("when walking", function(){
            // your test code;
      })
})
```

Writing the code

it() block

beforeEach() and afterEach()

Inbuilt Matchers

Custom Matchers

it() block

```
describe("test title", function(){
    it("should be walking",
    function(){
      // your code and assertions
    })
})
```

beforeEach() and afterEach()

```
describe("test title", function(){
  var user = null;
  beforeEach(){
      user = "username";
  }
  it("should be username", function(){
    // your code and assertions
  })
  afterEach(){
      user = null;
  }
})
```

Built-in Matcher

```
expect(x).toEqual(y) // if properties are equal (can be
                     objects, arrays etc.)
expect(x).toBe(y)    // uses === comparison
expect(x).toMatch(y) // uses regular expression
expect(x).toBeDefined(y) // not be undefined
expect(x).toBeUndefined(y) // to be undefined
expect(x).toBeNull(y)      // to be null
expect(x).toBeTruthy(y) // to be true
expect(x).toBeFalsy(y) // to be false
expect(x).toBeContain(y) // contain in array
expect(x).toBeLessThan(y) // less than
expect(x).toBeGreaterThan(y) // more than
expect(x).toThrow(y) // throws an error/exception
```

not the Negation

```
expect(x).not.toEqual(y)
expect(x).not.toBe(y)
expect(x).not.toMatch(y)
expect(x).not.toBeDefined(y)
expect(x).not.toBeUndefined(y)
expect(x).not.toBeNull(y)
expect(x).not.toBeTruthy(y)
expect(x).not.toBeFalsy(y)
expect(x).not.toBeContain(y)
expect(x).not.toBeLessThan(y)
expect(x).not.toBeGreaterThan(y)
expect(x).not.toThrow(y)
```

Creating custom matcher

```
Typically created in a beforeEach block
Use the method this.addMatchers()

beforeEach(function(){
  this.addMatchers({
      toBeBatman :  function(){
            {
            message : "you are not batman"
            }
            return this.actual === "batman"
      }
  })
})
```

vijay.shivu@gmail.com