ThoughtWorks®

Testing JavaScript with Jasmine

Rachel Laycock
Jo Cranford

Why Jasmine?

- Alternatives:
    - JsTestDriver
    - Screw.Unit
    - JSpec
    - YUI Test Framework

- Jasmine had good reports from colleagues
- Initial experimentation was positive

ThoughtWorks®

Behaviour-Driven vs Test-Driven

- The principle is exactly the same!
- Language shift
  - TestObject => ShouldDoSomething
- Thinking about ‘behaviour’ is more natural
- Business people can understand your tests
  - And tell you where the behaviour needs adjusting to meet requirements
- Tests become a living specification of what the code does

Let's start with a Project …

- Bowling Game Kata:
  - http://butunclebob.com/files/downloads/Bowling%20Game%20Kata.ppt
- What are the rules?

Getting started

- Download Jasmine from:
  - http://pivotal.github.com/jasmine/download.html
- Copy the Jasmine files:
  - Entire 'lib' folder
  - SpecRunner.html
- Edit SpecRunner.html to link to your own source and test files

- Jasmine documentation links:
  - Wiki: https://github.com/pivotal/jasmine/wiki
  - API docs: http://pivotal.github.com/jasmine/jsdoc/index.html

The first test

- If I roll and knock down zero skittles, my overall score should be zero
- Let’s write the code …

ThoughtWorks®

toBe or not.toBe, that is the question ...

- All matchers can be reversed by prefixing with “not.”

- Testing equality:
  - toBe()
  - toEqual()

- Testing booleans:
  - toBeTruthy()
  - toBeFalsy()

ThoughtWorks®

More matchers

- toBeDefined
- toBeUndefined
- toBeNull
- toBeGreaterThan
- toBeLessThan
- Arrays: toContain
- Exceptions: toThrow
- Regular Expressions: toMatch

Before and After

- Run before or after every test
  - beforeEach
  - afterEach
- Uses:
  - Keeps stubbed code together
  - Specific to a describe
  - Reset variables

Spying, Mocking, and Stubbing

- Spies in Jasmine can be used to mock and stub
  - Stub: canned responses
  - Mock: objects preprogrammed with expectations
    - Checking the messages they receive

Spying on a method

- Spies replace the method on the object with a Jasmine spy object
- To create a spy:
  - spyOn
  - createSpy
- To check if it was called:
  - toHaveBeenCalled
  - callCount
- Always create the spy before calling it!

Doing more than just spying

- If the spied method has to *return* something:
  - andReturn()
- If it has to *do* something:
  - andCallFake()
  - andCallThrough()

- Calling through isn't really unit testing!

spyOn versus createSpy

- spyOn
  - Keeps a reference to the original method
  - Allows you to do .andCallThrough()
- createSpy
  - Does not require the method to exist already

Spying on parameters

- toHaveBeenCalledWith
- mostRecentCall
- jasmine.any

Faking it

- Common uses:
  - Catching Ajax requests and calling the callback
  - Catching methods bound to events and calling them
- Use the fake function to:
  - Call the callback immediately
  - Store in a variable and call later

HTML Fixtures

- For testing DOM interactions, get the jasmine-jquery plugin here:
    - https://github.com/velesin/jasmine-jquery
- Create HTML Fixtures:
    - Separate files
    - Within tests
- Watch out!
    - Large fixtures cause maintenance headaches
    - Out of sync with real code

Other interesting stuff

* Custom matchers
* Waiting around
* In the build

Questions?