Testing Clientside JavaScript

Jasmine Spies

Joe Eames

@josepheames

www.testdrivenjs.com

www.pluralsight.com

pluralsight
hardcore developer training

Spies in Jasmine

- Built for mocking single functions
- Supports both replacement and pass-through mocking
- Has many built in matchers.

Creating Spies in Jasmine

```
Function doSomething(callback) {
   …
   if(success) {
      callback();
   }
}
```

```
it(‘tests something’, function() {
   var spyCallback = jasmine.createSpy(‘mySpy’);
   doSomething(spyCallback);
   expect(spyCallback).toHaveBeenCalled();
}
```

spyOn function

- Used to spy on a method of an object
- Global Function – not on the jasmine object
- Basic usage – spyOn(object, “methodName”)
- andReturn
- andCallFake
- andCallThrough
- andThrow

createSpyObj function

- Creates an object with only spy methods

```
var spy = jasmine.createSpyObj(‘name’, [‘method1’,
          ‘method2’, …]);
```

Jasmine Spy Matchers

- **toHaveBeenCalled**
- **not.toHaveBeenCalled**
- **toHaveBeenCalledWith(arg1, arg2…)**
- **not.toHaveBeenCalledWith(arg1, arg2…)**
- **calls[]**
- **calls[].args[]**
- **calls[].object**
- **callCount**
- **mostRecentCall**
- **argsForCall[]**

Jasmine Spy Utility Methods

- jasmine.isSpy
- reset

Summary

- Jasmine Spies
- spyOn()
- createSpyObj()
- matchers