Testing with Spock

Dan Durkin

Boston Grails User Group, March 28, 2012

Acknowledgements

- Peter Niederweiser & Luke Daley
  - Framework, slides, examples
- Broad Institute
  - Chemical Biology Informatics Platform
- Richard Rattigan
  - Examples

Spock

- A developer testing framework
- For Groovy and Java applications
- Based on Groovy
- Fully compatible with JUnit
- Adds additional features and capabilities that make testing more fun!

- Check out: http://code.google.com/p/spock/wiki/WhySpock

Frame of Reference

- Traditional testing
  - Setup thing to test
  - Do something to it
  - Check it's state

Casually Making it Formal

- First class support for Blocks
  - setup: cleanup: expect: given: when: then: where: and:
  - excellent support for inline documentation
- Fixture Methods
  - setup() cleanup() setupSpec() cleanupSpec()
  - @Before @BeforeClass @After AfterClass
- Instance and @Shared fields

Extra Goodness

- Parameterized Feature Methods
- Excellent support for exception testing
  - Thrown()
- Interactions (Stubbing and Mocking)
- Extensions
- Compatible
- Refreshing

Termonology Comparison to JUnit

Spock	JUnit
Specification	Test class
setup()	@Before
cleanup()	@After
setupSpec()	@BeforeClass
cleanupSpec()	@AfterClass
Feature	Test
Parameterized feature	Theory
Condition	Assertion
Exception condition	@Test(expected=...)
@FailsWith	@Test(expected=...)
Interaction	Mock expectation (EasyMock, JMock, ...)

Additional Info

- http://spockframework.org/
- https://github.com/spockframework
  - https://github.com/spockframework/spock-example
  - This talk: https://github.com/ddurkin/spock-example

Interactions (Stubbing and Mocking)

- Spock has it's own mocking/stubbing
  - Why?
    - Desire for flexible DSL for describing interactions
    - Goal of being able to specify enough but avoid being extremely specific resulting in fragile tests
  - Mocks are lenient
    - If a method call does not match any interaction, the default value for the method's return type is returned
- Spock combines mocking and stubbing
- Docs: http://code.google.com/p/spock/wiki/Interactions

Extensions

- *The* way to extend Spock Two flavors:
  - Annotation-driven
  - Global
- Abilities:
  - Register interceptors
  - Register listeners
  - Alter the spec's *model*

Some Existing Extensions

- @Ignore/@IgnoreRest/@IgnoreIf
- @Timeout
- @Stepwise
- @AutoCleanup
- @RevertMetaClass
- @Use
- @Rule/@ClassRule