canoo

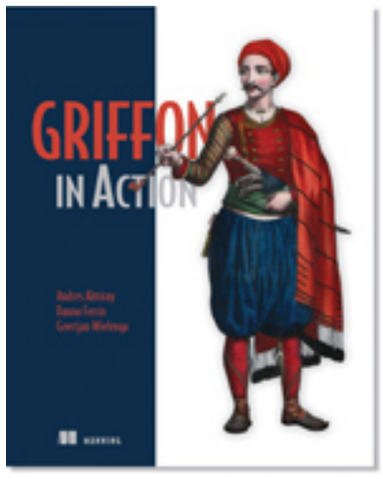
› your provider for business web solutions ›

Escribiendo Pruebas con Spock

¿Y quién es éste tipo?

- Desarrollador  desde el inicio (1995 y contando …)
- Creyente fiel del Código Abierto y Libre
- Miembro del equipo Groovy
- Líder del proyecto GRIFFON
- Actualmente trabajando con canoo

JAVA™ CHAMPIONS



Que es spock?

Donde, Quien, Como?

- http://spockframework.org
- Peter Niederwiser @pniederw

- Lenguage para pruebas basado en Groovy
- Manipulacion de codigo byte en tiempo de compilacion
- Inspirado en JUnit, Rspec, jMock, Mockito, Groovy, Scala y Vulcans

Porque?

- Mayor expresividad
- Facil de aprender
- Use multivariado (unitario a funcional)
- Potencializa la la sintaxis idiomatica de Groovy
- Compatible con Junit, IDEs, CI.
- Extendible via plugins

Primeras impresiones

```
import spock.lang.Specification

class HelloSpec extends Specification {
    def "length of Spock's and his friends' names"() {
        expect:
            name.size() == length

        where:
            name      | length
            "Spock"   | 5
            "Kirk"    | 4
            "Scotty"  | 6
    }
}
```

Introduciendo un defecto

```
import spock.lang.Specification

class HelloSpec extends Specification {
    def "length of Spock's and his friends' names"() {
        expect:
            name.size() == length

        where:
            name      | length
            "Spock"   | 5
            "Kirk"    | 4
            "Scotty"  | 7
    }
}
```

A wild error appears!

```
Condition not satisfied:

name.size() == length
|    |      |  |
|    6      |  7
Scotty      false
 <Click to see difference>

at HelloSpec.length of Spock's and his friends' names(HelloSpec.groovy:5)
```

Caracteristicas (1)

Bloques

- given: precondiciones, inicializacion de datos, etc.
- when: acciones que producen un resultado
- then: verificacion de expectativas
- expect: alternativa corta a when: y then:
- where: aplica entradas de datos varias
- and: sub-dividide (y enlaza) bloques
- setup: otro nombre para given:
- cleanup: post condiciones, limpieza de recursos, etc.

Caracteristicas (2)

Ciclo de Vida

- setup
- cleanup
- setupSpec
- cleanupSpec

Dirigido por Datos

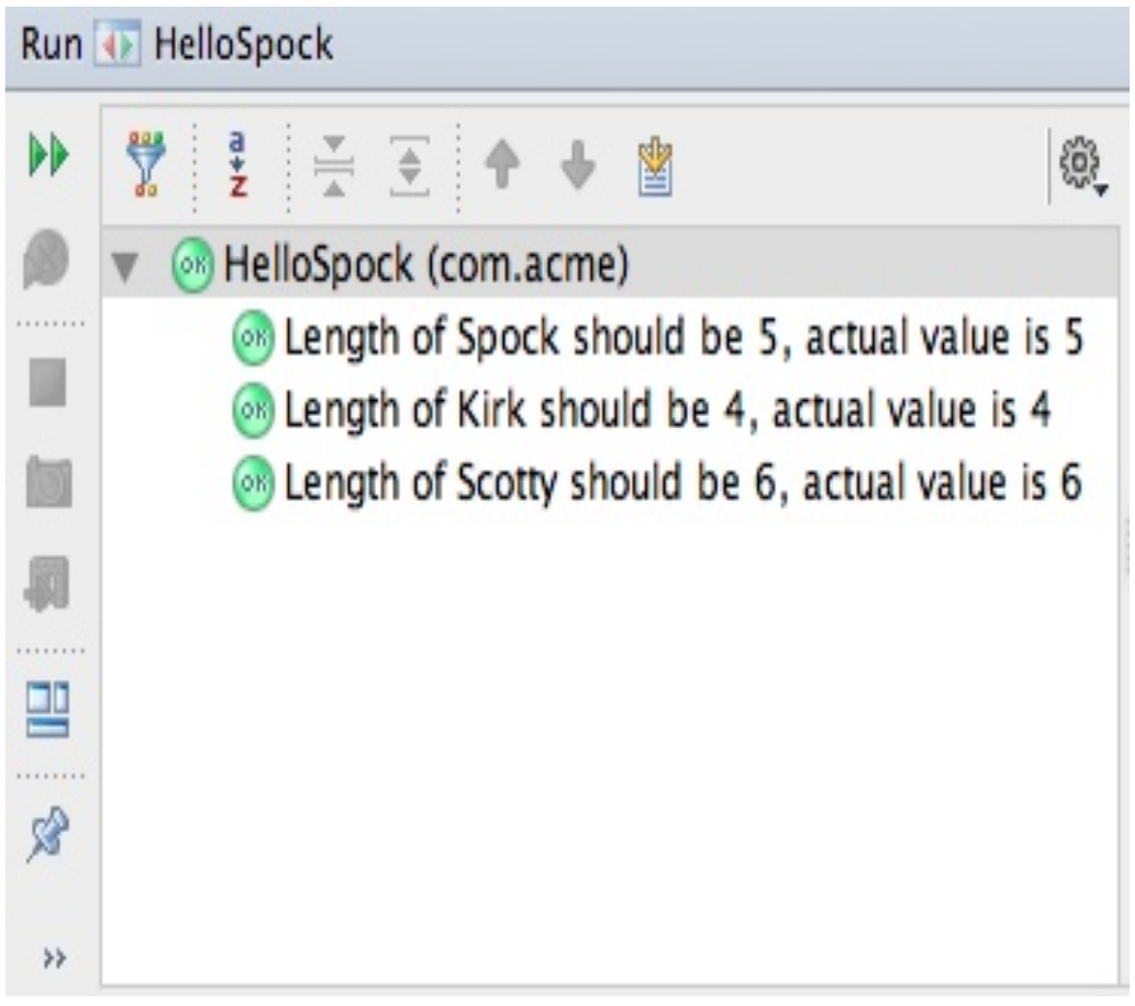
- Variables ligadas a Listas
- Variables ligadas a Tablas
- @Unroll
- @Shared

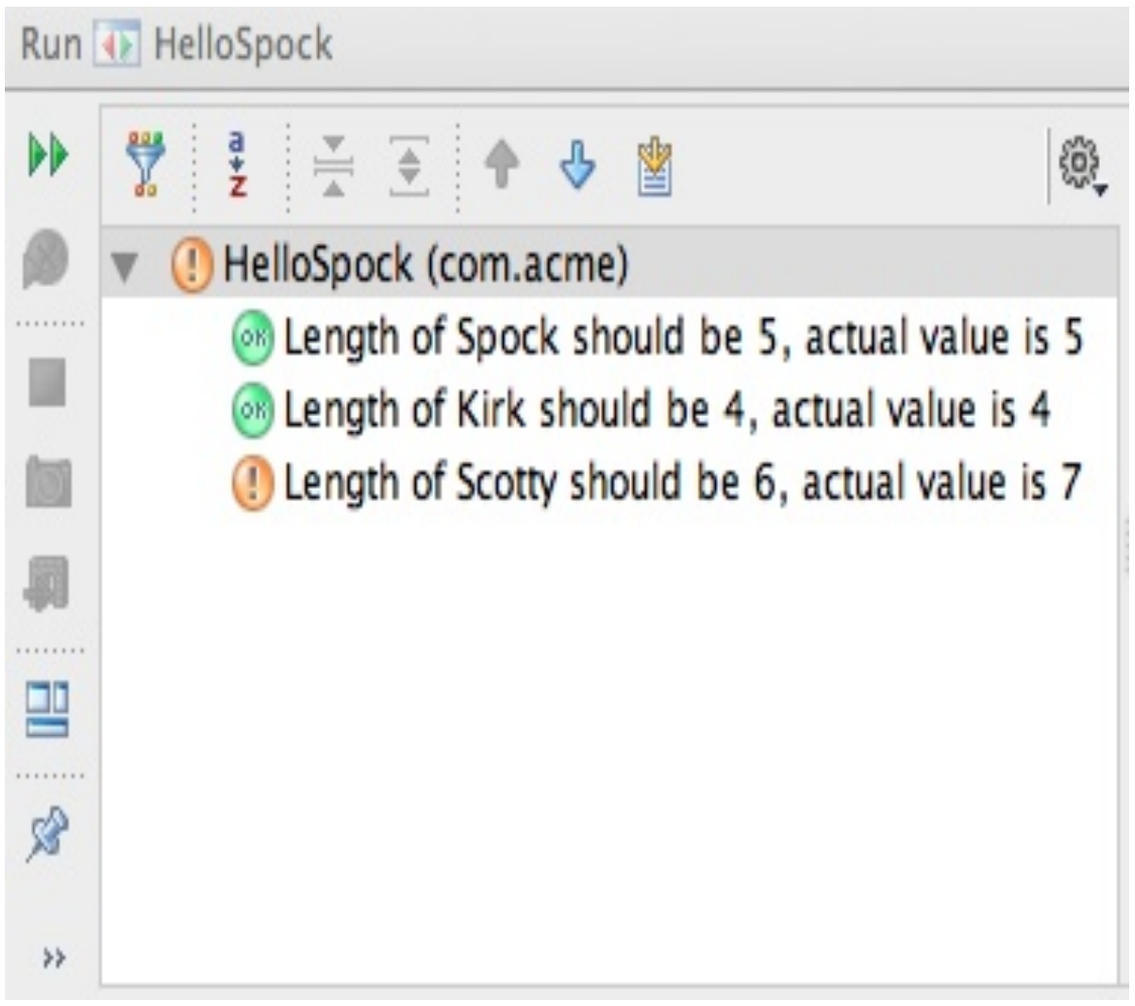
Unrolling

```
import spock.lang.Specification
import spock.lang.Unroll

@Unroll
class HelloSpec extends Specification {
  def "Length of #name should be #name.size(), value is #length"() {
    expect:
      name.size() == length
   where:
      name      | length
      "Spock"   | 5
      "Kirk"    | 4
      "Scotty"  | 6
    }
}
```

Unrolling visto por un IDE





Interactiones (Mocks)

```groovy
import spock.lang.Specification

import java.beans.PropertyChangeListener

class Model {
    @groovy.beans.Bindable String name
}

class BindableSpec extends Specification {
    def "Model properties are observable"() {
        given:
        def model = new Model()
        def listener = Mock(PropertyChangeListener)
        when:
        model.addPropertyChangeListener(listener)
        model.name = 'Groovy'
        model.name = 'Java'
        then:
        1 * listener.propertyChange({it.newValue == 'Groovy'})
        1 * listener.propertyChange({it.newValue == 'Java'})
    }
}
```

Espere, aun hay mas!

- http://docs.spockframework.org/en/latest/
- Spock es extendible via plugins
- Pruebas funcionales para aplicaciones web con GEB
- Plugins para Grails y Griffon
- Siguiente liberacion sera el codiciado 1.0.0

Gracias!

twitter: @aalmiray

SG VIRTUAL CONFERENCE