Curso Experto Aseguramiento de la Calidad Del Software

TESTING CON SPOCK
(Pruebas Unitarias)

RUBÉN GONZÁLEZ MARTÍN

OBJETIVOS

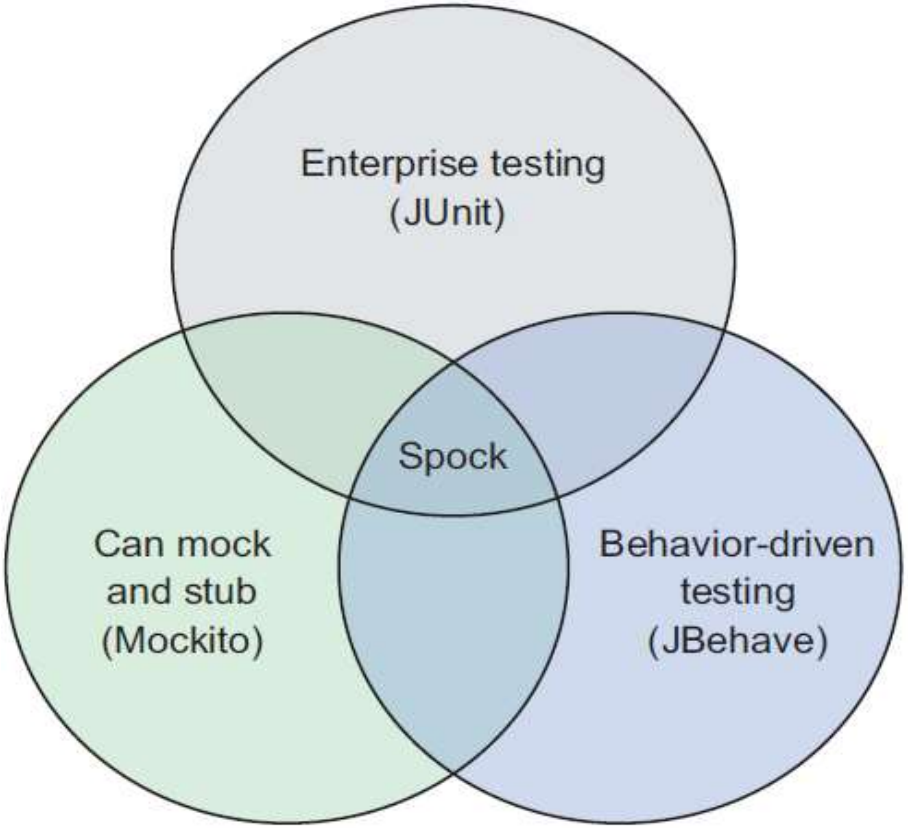
PRIMERA TOMA DE CONTACTO CON TESTING SPOCK

- Configuración en IDE.
- Análisis de características framework Spock para pruebas unitarias
- Realización con Spock de algunos ejercicios iniciales del curso.

SPOCK. INTRODUCCIÓN I

- Testing Framework que usa Groovy
  - Para Java
  - Para Groovy
- Gratuito
- Ejecutable desde JUnit

SPOCK. INTRODUCCIÓN II



Java Testing with Spock (Manning 2016)

CONFIGURACIÓN I

- Eclipse (versiones nuevas a partir de Indigo).
  - Problema Plugin de Integración con Groovy.
- IntelliJ ➔ Integración por defecto.
- Proyectos Maven y Gradle.

CONFIGURACIÓN II (POM)

```
<plugin>
    <groupId>org.codehaus.gmavenplus</groupId>
    <artifactId>gmavenplus-plugin</artifactId>
    <version>1.4</version>
    <executions>
        <execution>
            <goals>
                <goal>compile</goal>
                <goal>testCompile</goal>
            </goals>
        </execution>
    </executions>
</plugin>
<plugin>
    <artifactId>maven-surefire-plugin</artifactId>
    <version>2.6</version>
    <configuration>
        <useFile>false</useFile>
        <includes>
            <include>**/*Spec.java</include>
            <include>**/*Test.java</include>
        </includes>
    </configuration>
</plugin>
```

CONFIGURACIÓN III (POM)

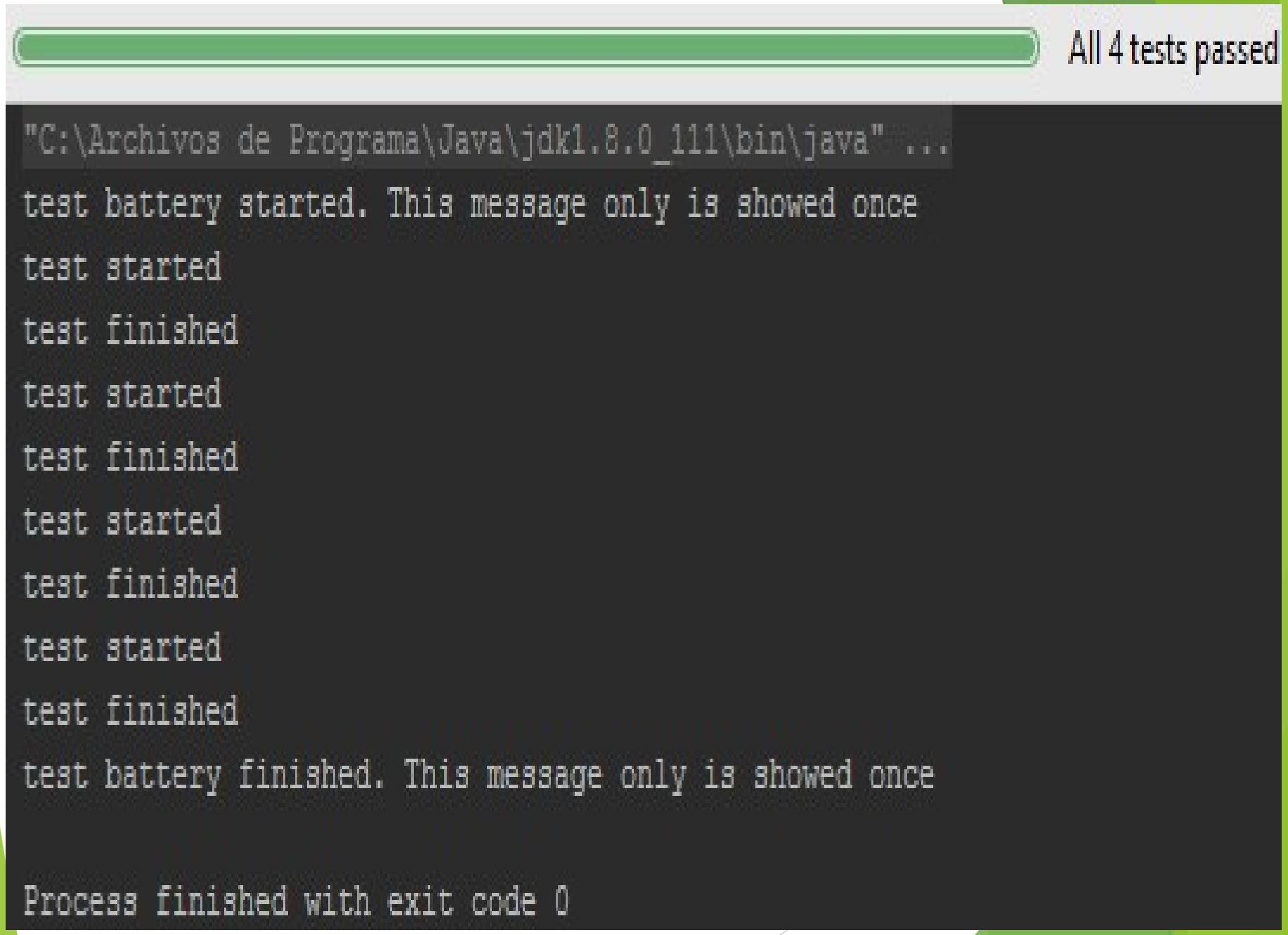
```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.12</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.spockframework</groupId>
        <artifactId>spock-core</artifactId>
        <version>1.0-groovy-2.4</version>
        <scope>test</scope>
    </dependency>
    <dependency> <!-- enables mocking of classes (in addition to interfaces) -->
        <groupId>cglib</groupId>
        <artifactId>cglib-nodep</artifactId>
        <version>3.1</version>
        <scope>test</scope>
    </dependency>
    <dependency> <!-- enables mocking of classes without default constructor (together with
            CGLIB) -->
        <groupId>org.objenesis</groupId>
        <artifactId>objenesis</artifactId>
        <version>2.1</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.codehaus.groovy</groupId>
        <artifactId>groovy-backports-compat23</artifactId>
        <version>2.3.7</version>
    </dependency>
</dependencies>
```

EJEMPLOS

```groovy
class AppSpec extends spock.lang.Specification{
    def "Adding two numbers to get the sum"() {
        when: "a new Calculadora class is created"
            Calculadora calc = new Calculadora ();
        then: "1 plus 1 is 1"
            calc.suma(1, 1) == 2
    }


    def "Subtracting two numbers to get the sum"() {
        when: "a new Calculadora class is created"
            Calculadora calc = new Calculadora ();
        then: "1 minus 1 is 0"
            calc.resta(1, 1) == 0
    }

}
```

Setup, setupSpec,clean, cleanup

```
def setupSpec(){
    println "test battery started. This message only is showed once"
}

def setup(){
    println "test started"
}



.......
.......

def cleanup(){
   println"test finished"
}

def cleanupSpec(){
    println "test battery finished. This message only is showed once"
}
```


All 4 tests passed
"C:\Archivos de Programa\Java\jdk1.8.0_111\bin\java" ...
test battery started. This message only is showed once
test started
test finished
test started
test finished
test started
test finished
test started
test finished
test battery finished. This message only is showed once
Process finished with exit code 0

MATCHERS

```
def "Given Zero Complex when Add To Complex1_1_ thenComplex 1_1 IsObtained \
     using Given_When_Then"() {
    given: "a zero Complex"
        Complex complexZero = new Complex (0,0);
    when: "Add Complex1_1 is added"
        Complex complexOneOne = new Complex (1,1);
        Complex resultSumComplexZero_1_1 = complexZero.add(complexOneOne)
    then: "the result is the Complex 1_1"
        expect (resultSumComplexZero_1_1, equalTo (complexOneOne))
}
```

```
def "Given Zero Complex then Real Part Zero And Imaginary Part Zero"() {
    when: "a zero Complex is created"
        Complex complex = new Complex (0.0,0.0);
     then: "Real part is zero"
        expect ( complex.getRealPart(), equalTo (0.0.doubleValue()))
     and: "Imaginary part is zero"
        expect ( complex.getImaginaryPart(), equalTo (0.0.doubleValue()))
}
```

Tests Parametrizados

```
class ComplexAbsSpec extends spock.lang.Specification{
    def "Given some complex numbers tests if abs value is right"() {

        given:
            Complex complex = new Complex(a,b)
        when: "Complex "
            double resultAbs = complex.abs()
        then: " The result od the absolute values of the comples "

        expect: "Abs value of each complex is the right"

        AbsOfComplex == resultAbs

        where: "the following cases are"
            a | b | AbsOfComplex
            1 | 1 | 1.4142135623730951
            0 | 0 | 0
            2 | 2 | 2.8284271247461903
            20| 2 | 20.09975124224178

    }
}
```

CONCLUSIONES

Framework muy completo

Apto para testing de código Java

Groovy aunque muy similar no = Java

Código de Testing sencillo y poco verboso.

Dificultades configuración en eclipse

REFERENCIAS

- Java Testing with Spock (Manning 2016)
- Groovy in action (Manning 2017).
- http://spockframework.org/
- Repositorio: https://github.com/ruglez/Pruebas_Software_Actividad1

FIN