GR8Conf Europe 2012
June 6th-8th
Copenhagen, Denmark

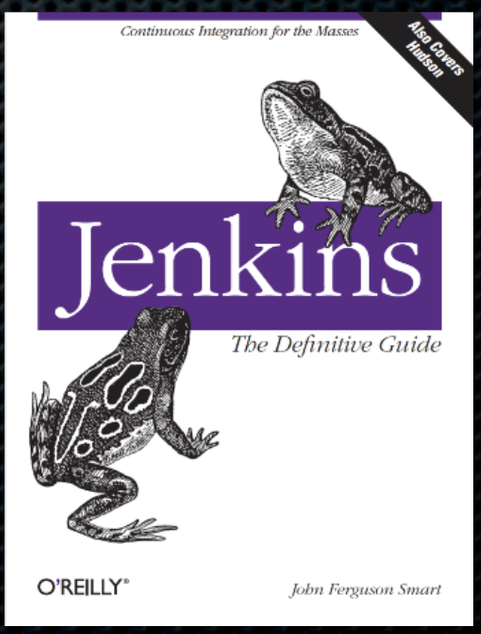
Spock Extensions Anatomy

Goldin Evgeny

JetBrains

Goldin Evgeny

- TeamCity and YouTrack Evangelist
- Groovy practitioner
- GroovyMag writer
- Blogger
- Open-Source developer
- @evgeny_goldin



Agenda

Agenda

- Why extending Spock?
- Global and annotation-driven extensions
- Interceptors and Listeners
- Demos and examples
- Cookbook and gotchas

Why?

Why Extending Spock?

Integrate - @Rule, @Autowired

Why Extending Spock?

Integrate - @Rule, @Autowired

Helper fields - @TestDir, @TempDir

Why Extending Spock?

Integrate - @Rule, @Autowired

Helper fields - @TestDir, @TempDir

Control - @Ignore, @FailsWith

Why Extending Spock?

Integrate - @Rule, @Autowired

Helper fields - @TestDir, @TempDir

Control - @Ignore, @FailsWith

Report - @Unroll, ProfilerExtension

Spock Dictionary

```groovy
class HelloSpock extends spock.lang.Specification {

    def setup      () { println 'setup()'       }
    def cleanup    () { println 'cleanup()'     }
    def setupSpec  () { println 'setupSpec()'   }
    def cleanupSpec() { println 'cleanupSpec()' }

    @TestDir
    File testDir

    @Unroll
    def "length of Spock's and his friends' names: [#name] / [#length]"() {
        expect:
        name.size() == length
        testDir.directory

        where:
        name   << [ 'Spock', 'Kirk', 'Scotty' ]
        length << [ 5, 4, 6 ]
    }
}
```

Spock Dictionary

specification

```groovy
class HelloSpock extends spock.lang.Specification {

    def setup      () { println 'setup()'       }
    def cleanup    () { println 'cleanup()'     }
    def setupSpec  () { println 'setupSpec()'   }
    def cleanupSpec() { println 'cleanupSpec()' }

    @TestDir
    File testDir

    @Unroll
    def "length of Spock's and his friends' names: [#name] / [#length]"() {
        expect:
        name.size() == length
        testDir.directory

        where:
        name   << [ 'Spock', 'Kirk', 'Scotty' ]
        length << [ 5, 4, 6 ]
    }
}
```

Spock Dictionary

fixture methods

```groovy
class HelloSpock extends spock.lang.Specification {

    def setup      () { println 'setup()'      }
    def cleanup    () { println 'cleanup()'    }
    def setupSpec  () { println 'setupSpec()'  }
    def cleanupSpec() { println 'cleanupSpec()' }

    @TestDir
    File testDir

    @Unroll
    def "length of Spock's and his friends' names: [#name] / [#length]"() {
        expect:
        name.size() == length
        testDir.directory

        where:
        name   << [ 'Spock', 'Kirk', 'Scotty' ]
        length << [ 5, 4, 6 ]
    }
}
```

Spock Dictionary

```groovy
class HelloSpock extends spock.lang.Specification {

    def setup      () { println 'setup()'      }
    def cleanup    () { println 'cleanup()'    }
    def setupSpec  () { println 'setupSpec()'  }
    def cleanupSpec() { println 'cleanupSpec()' }

    @TestDir
    File testDir

    @Unroll
    def "length of Spock's and his friends' names: [#name] / [#length]"() {
        expect:
        name.size() == length
        testDir.directory

        where:
        name   << [ 'Spock', 'Kirk', 'Scotty' ]
        length << [ 5, 4, 6 ]
    }
}
```

fields

Spock Dictionary

```groovy
class HelloSpock extends spock.lang.Specification {

    def setup      () { println 'setup()'      }
    def cleanup    () { println 'cleanup()'    }
    def setupSpec  () { println 'setupSpec()'  }
    def cleanupSpec() { println 'cleanupSpec()' }

    @TestDir
    File testDir

    @Unroll
    def "length of Spock's and his friends' names: [#name] / [#length]"() {
        expect:
        name.size() == length
        testDir.directory

        where:
        name   << [ 'Spock', 'Kirk', 'Scotty' ]
        length << [ 5, 4, 6 ]
    }
}
```

feature methods

Spock Dictionary

```groovy
class HelloSpock extends spock.lang.Specification {

    def setup      () { println 'setup()'      }
    def cleanup    () { println 'cleanup()'    }
    def setupSpec  () { println 'setupSpec()'  }
    def cleanupSpec() { println 'cleanupSpec()' }

    @TestDir
    File testDir

    @Unroll
    def "length of Spock's and his friends' names: [#name] / [#length]"() {
        expect:
        name.size() == length
        testDir.directory

        where:
        name   << [ 'Spock', 'Kirk', 'Scotty' ]
        length << [ 5, 4, 6 ]
    }
}
```

iterations

Extensions Dictionary

- Global extensions
- Annotation-driven extensions
- Interceptors
- Listeners

Extensions Dictionary

`OptimizeRunOrderExtension`	Listeners `@Stepwise`
Global Interceptors `RuleExtension`	Annotation-Driven `@Timeout`

Extensions Dictionary

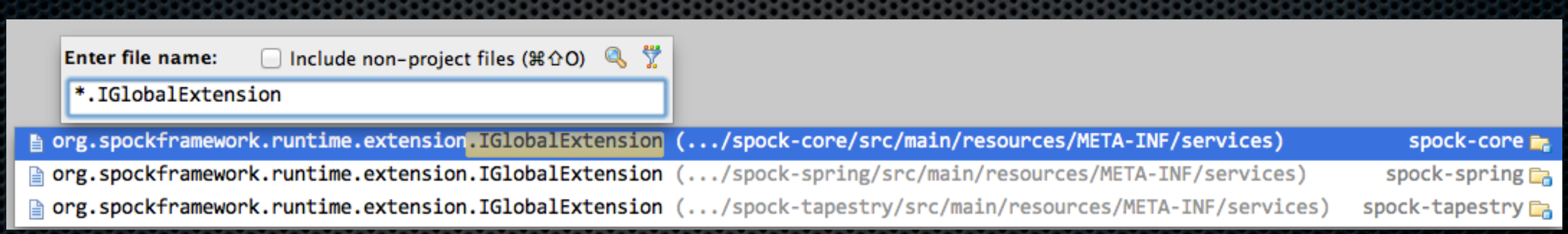
`OptimizeRunOrderExtension`	Listeners `@Stepwise`
Global Interceptors `RuleExtension`	Annotation-Driven `@Timeout`

Global Extensions

```
src/main/resources

src/test/resources

  META-INF/services/org.spockframework.runtime.extension.IGlobalExtension
```




```
org.spockframework.runtime.extension.builtin.IncludeExcludeExtension
org.spockframework.runtime.extension.builtin.RuleExtension
org.spockframework.runtime.extension.builtin.ClassRuleExtension
org.spockframework.runtime.extension.builtin.OptimizeRunOrderExtension
org.spockframework.runtime.extension.builtin.JUnitFixtureMethodsExtension
```

Global Extensions

```
dependencies {
   ...
   testCompile 'com.github.goldin:spock-extensions:0.1.4'
}
```

Or

```
dependencies {
   ...
   testRuntime 'com.github.goldin:spock-extensions:0.1.4'
}
```

Global Extensions

```
package org.spockframework.runtime.extension;

import org.spockframework.runtime.model.SpecInfo;

public interface IGlobalExtension {
   void visitSpec(SpecInfo spec);
}
```

SpecInfo

- Read fields, feature and fixture methods
- Add interceptors
- Add listeners
- Re-order, filter, sort, skip, exclude features

SpecInfo

- Read fields, features and fixture methods
- Add interceptors
- Add listeners
- Re-order, filter, sort, skip, exclude features

Annotation Driven Extension

```java
@Retention(RetentionPolicy.RUNTIME)
@Target({ElementType.TYPE, ElementType.METHOD})
@ExtensionAnnotation(IgnoreExtension.class)
public @interface Ignore {
  /**
   * The reason for ignoring this element.
   *
   * @return the reason for ignoring this element
   */
  String value() default "";
}
```

Annotation Driven Extension

```
public interface IAnnotationDrivenExtension<T extends Annotation> {
   void visitSpecAnnotation   (T annotation, SpecInfo    spec);
   void visitFeatureAnnotation(T annotation, FeatureInfo feature);
   void visitFixtureAnnotation(T annotation, MethodInfo  fixtureMethod);
   void visitFieldAnnotation  (T annotation, FieldInfo   field);
   void visitSpec             (SpecInfo spec);
}
```

Annotation Driven Extension

```
public interface IAnnotationDrivenExtension<T extends Annotation> {
  void visitSpecAnnotation   (T annotation, SpecInfo    spec);
  void visitFeatureAnnotation(T annotation, FeatureInfo feature);
  void visitFixtureAnnotation(T annotation, MethodInfo  fixtureMethod);
  void visitFieldAnnotation  (T annotation, FieldInfo   field);
  void visitSpec             (SpecInfo spec);
}
```

Annotation Driven Extension

```
public interface IAnnotationDrive Ext ion<T extends Annotation> {
  void visitSpecAnnotati n  (T annotation, SpecInfo    spec);
  void visitFeatureAnnota ion(T annotation, FeatureInfo feature);
  void visitFix reA no ation(T annotation, MethodInfo  fixtureMethod);
  void visi  ield nnotation  (T annotation, FieldInfo   field);
  void visitSpe              (SpecIn  sp c);
}
```

Annotation Driven Extension

```
public interface IAnnotationDrivenExtension<T extends Annotation> {
  void visitSpecAnnotation    (T annotation, SpecInfo    spec);
  void visitFeatureAnnotation(T annotation, FeatureInfo feature);
  void visitFixtureAnnotation(T annotation, MethodInfo  fixtureMethod);
  void visitFieldAnnotation   (T annotation, FieldInfo   field);
  void visitSpecAnnotation    (SpecInfo spec);
}
```

Listeners

- Added to spec
- Before and after spec, feature, iteration
- On error
- Skipped spec or feature
- Listen passively, do not throw exceptions
- Get passed **official** test results

Interceptors

- Added to spec, feature or fixture methods
- Intercept method invocation
- invocation.proceed()
- Can change test outcome and behavior
- Can throw exceptions
- But .. following interceptors can do the same

Interceptors

- Added to spec, feature or fixture methods
- Intercept method invocation
- invocation.proceed()
- Can change test outcome and behavior
- Can throw exceptions
- But .. following interceptors can do the same

Gotchas

- Use the right tool for the job
- Attach interceptors only where they belong
- ***Do not*** throw exceptions from listeners
- Reuse listeners or interceptors with caution
- Mark feature methods as skipped if needed
- Skipped (skip a few) vs. excluded (run a few)

Cookbook

- Extend AbstractAnnotationDrivenExtension

Fights Misplaced Annotation

- Extend AbstractMethodInterceptor

- Extend AbstractRunListener

To Summarize ...

Why Extending Spock?

Integrate - @Rule, @Autowired

Helper fields - @TestDir, @TempDir

Control - @Ignore, @FailsWith

Report - @Unroll, ProfilerExtension

Extensions Dictionary

- Global extensions
- Annotation-driven extensions
- Interceptors
- Listeners

Extensions Project

- github.com/evgeny-goldin/spock-extensions
- @TestDir
- @TempDir
- @Time
- @With
- ProfilerExtension

Additional Resources

- github.com/spockframework/spock/
- evgeny-goldin.org/javadoc/spock/
- github.com/evgeny-goldin/spock-example
- Blog post: Annotation Driven Extensions With Spock
- GroovyMag December 2011: Extending Spock

Q & A

See you at GrâConf 2013!