Java Testing on the Fast Lane

Be more effective while programming tests (and have some fun too!)

Goal

About the presenter

- Java Developer since the beginning
- Open Source believer since 1997
- Groovy development team member since 2007
- Griffon project co-founder

Agenda

- What is Groovy
- Groovy + Testing Frameworks
- How Groovy helps
- Mocking with Groovy
- XML Processing
- Functional UI Testing
- Resources

What is Groovy?

- Groovy is an agile and **dynamic** language for the Java Virtual Machine
- Builds upon the strengths of Java but has additional power features inspired by languages like Python, Ruby & Smalltalk
- Makes modern programming features available to Java developers with **almost-zero learning curve**
- Supports **Domain Specific Languages** and other compact syntax so your code becomes easy to read and maintain

What is Groovy?

- Increases developer productivity by **reducing scaffolding** code when developing web, GUI, database or console applications
- **Simplifies testing** by supporting unit testing and mocking out-of-the-box
- **Seamlessly integrates** with all existing Java objects and libraries
- Compiles straight to Java byte code so you can **use it anywhere you can use Java**

HelloWorld.java

```
public class HelloWorld {
    String name;

    public void setName(String name)
    { this.name = name; }
    public String getName(){ return name; }

    public String greet()
    { return “Hello “+ name; }

    public static void main(String args[]){
        HelloWorld helloWorld = new HelloWorld();
        helloWorld.setName(“Groovy”);
        System.err.println( helloWorld.greet() );
    }
}
```

HelloWorld.groovy

```
public class HelloWorld {
    String name;

    public void setName(String name)
    { this.name = name; }
    public String getName(){ return name; }

    public String greet()
    { return “Hello “+ name; }

    public static void main(String args[]){
        HelloWorld helloWorld = new HelloWorld();
        helloWorld.setName(“Groovy”);
        System.err.println( helloWorld.greet() );
    }
}
```

Equivalent HelloWorld 100% Groovy

```
class HelloWorld {
    String name
    def greet() { "Hello $name" }
}

def helloWorld = new HelloWorld(name:"Groovy")
println helloWorld.greet()
```

1st Mantra

Java is Groovy, Groovy is Java

- Every single Java class is a Groovy class, the inverse is also true. This means your Java can call my Groovy in vice versa, without any clutter nor artificial bridge.
- Groovy has the same memory and security models as Java.
- Almost 98% Java code is Groovy code, meaning you can in most cases rename *.java to *.groovy and it will work.

Common Gotchas

- Java Array initializers are not supported, but lists can be coerced into arrays.
- Inner class definitions are not supported (coming in Groovy 1.7).

2nd Mantra

Groovy is Java and Groovy is not Java

- Flat learning curve for Java developers, start with straight Java syntax then move on to a groovier syntax as you feel comfortable.
- Groovy delivers closures, meta-programming, new operators, operator overloading, enhanced POJOs, properties, native syntax for Maps and Lists, regular expressions, enhanced class casting, optional typing, and more!

Groovy + Testing Frameworks

- Any Groovy script may become a testcase
  - assert keyword enabled by default
- Groovy provides a GroovyTestCase base class
  - Easier to test exception throwing code
- Junit 4.x and TestNG ready, Groovy supports JDK5+ features
  - Annotations
  - Static imports
  - Enums

How Groovy helps

- Write less with optional keywords – public, return, arg types & return types
- Terser syntax for property access
- Native syntax for Lists and Maps
- Closures
- AST Transformations – compile time meta-programming

Accessing Properties

```
// Java
public class Bean {
  private String name;
  public void setName(String n) { name = n; }
  public String getName() { return name; }
}

// Groovy
Bean bean = new Bean(name: “Duke”)
assert bean.name == “Duke”
bean.name = “Tux”
assert bean.name == “Tux”
assert bean.name == bean.getName()
```

Native Syntax for Maps and Lists

```
Map map = [:]
assert map instanceof java.util.Map
map["key1"] = "value1"
map.key2 = "value2"
assert map.size() == 2
assert map.key1 == "value1"
assert map["key2"] == "value2"

List list = []
assert list instanceof java.util.List
list.add("One")
list << "Two"
assert list.size() == 2
assert ["One","Two"] == list
```

Closures (1)

```
int count = 0
def closure = {->
  0.upto(10) { count += it }
}
closure()
assert count == (10*11)/2

def runnable = closure as Runnable
assert runnable instanceof java.lang.Runnable
count = 0
runnable.run()
assert count == (10*11)/2
```

Closure (2)

```
// a closure with 3 arguments, third one has
// a default value
def getSlope = { x, y, b = 0 ->
   println "x:${x} y:${y} b:${b}"
   (y - b) / x
}

assert 1 == getSlope( 2, 2 )
def getSlopeX = getSlope.curry(5)
assert 1 == getSlopeX(5)
assert 0 == getSlopeX(2.5,2.5)
// prints
// x:2 y:2 b:0
// x:5 y:5 b:0
// x:5 y:2.5 b:2.5
```

AST Transformations

```
import java.text.SimpleDateFormat
class Event {
    @Delegate Date when
    String title, url
}
def df = new SimpleDateFormat("MM/dd/yyyy")
def oscon = new Event(title: "OSCON 09",
   url: "http://en.oreilly.com/oscon2009/",
   when: df.parse("07/20/2009"))
def so2gx = new Event(title: "SpringOne2GX",
   url: "http://www.springone2gx.com/",
   when: df.parse("10/19/2009"))

assert oscon.before(so2gx.when)
```

AST Transformations

- @Singleton
- @Lazy
- @Delegate
- @Immutable
- @Bindable
- @Newify
- @Category/@Mixin
- @PackageScope

But how do I run Groovy tests?

- Pick your favourite IDE!
  - IDEA
  - Eclipse
  - NetBeans
- Command line tools
  - Ant
  - Gant
  - Maven
  - Gradle
  - Good ol' Groovy shell/console

Testing exceptions in Java

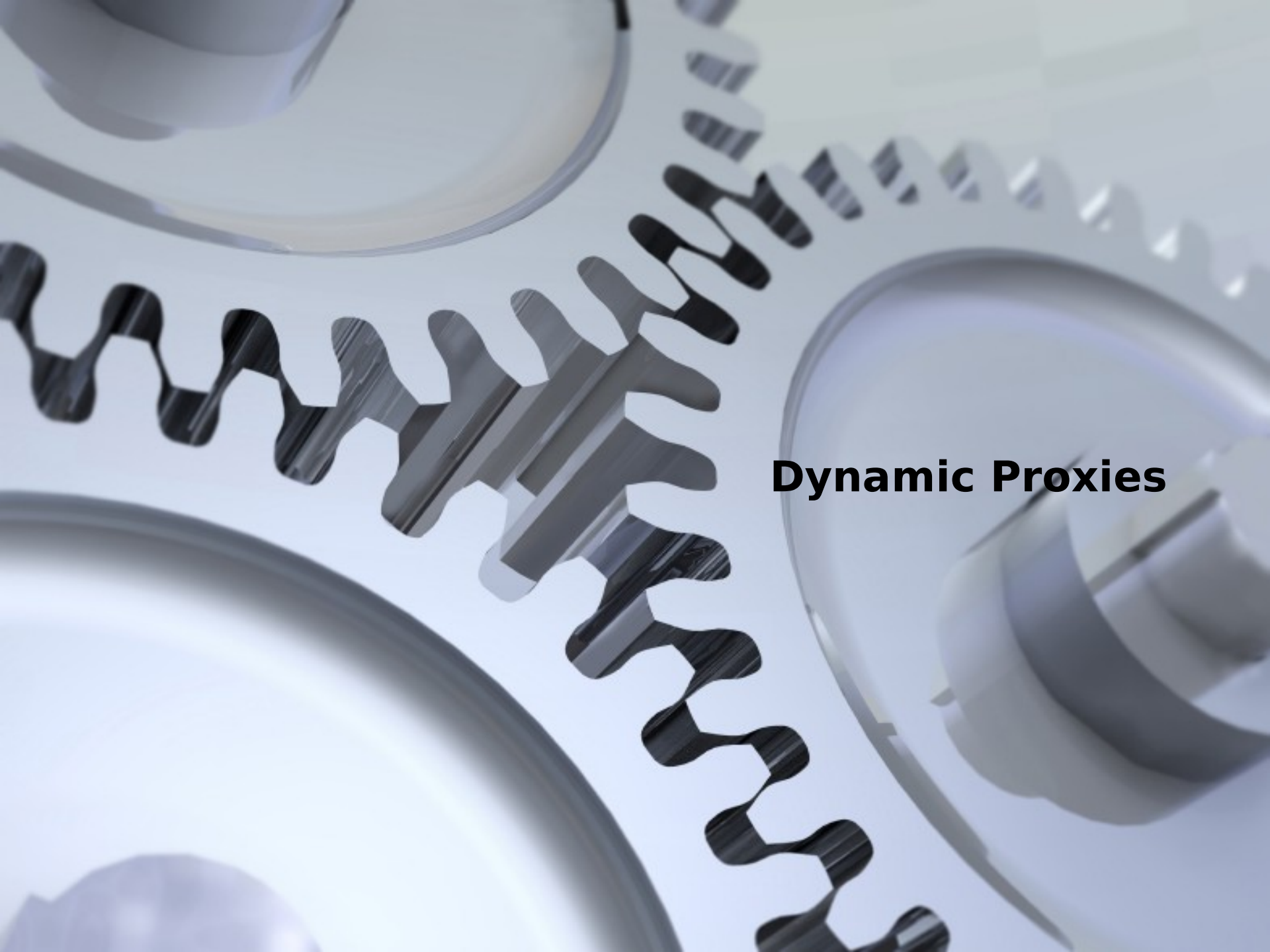
```
public class JavaExceptionTestCase extends TestCase {
    public void testExceptionThrowingCode() {
       try {
          new MyService().doSomething();
          fail("MyService.doSomething has been implemented");
       }catch( UnsupportedOperationException expected ){
          // everything is ok if we reach this block
       }
    }
}
```

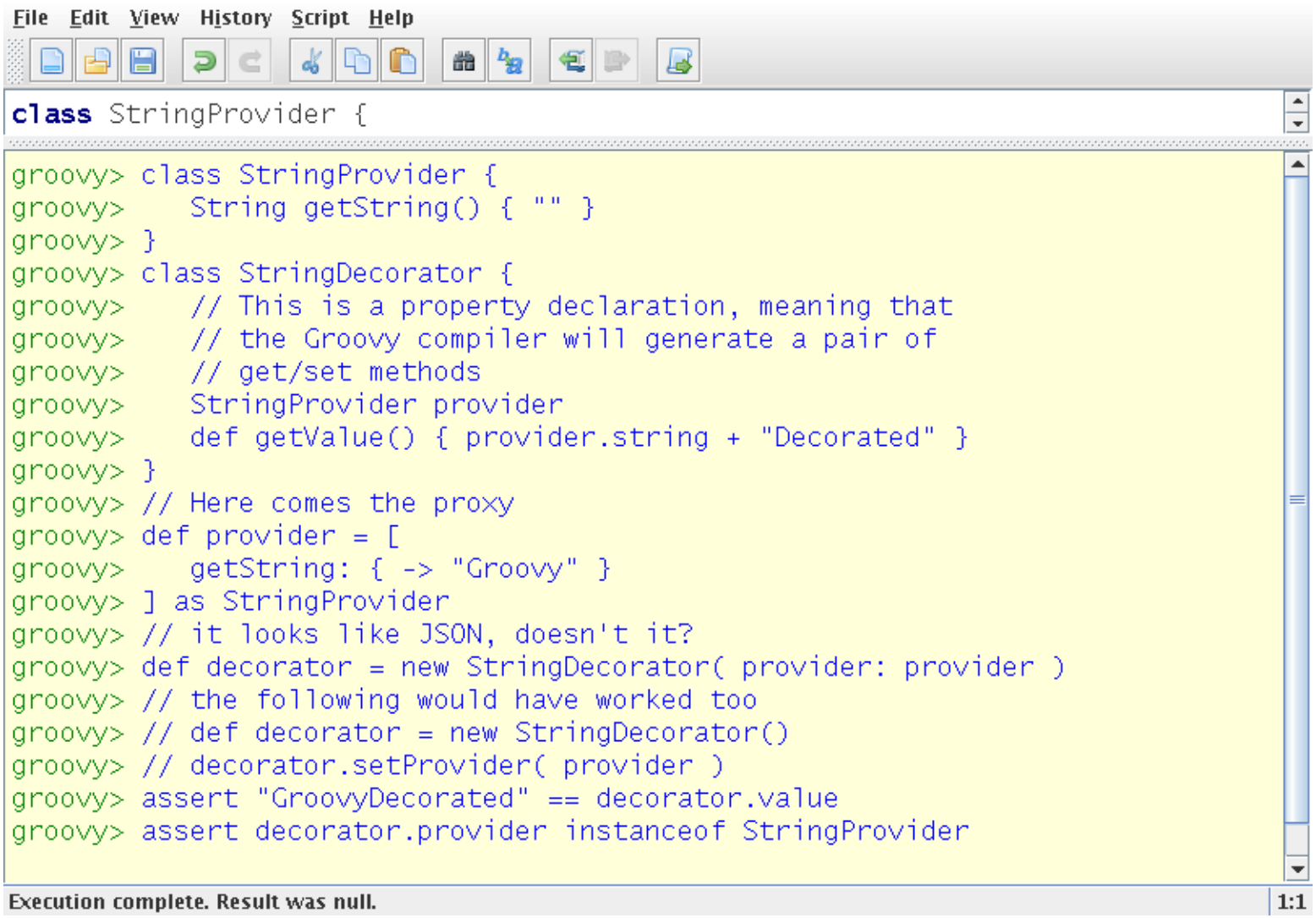
Testing exceptions in Groovy

```
class GroovyExceptionTestCase extends GroovyTestCase {
    void testExceptionThrowingCode() {
        shouldFail( UnsupportedOperationException ){
            new MyService().doSomething()
        }
    }
}
```

Mocking with Groovy

- Known (Java) mocking libraries
  - EasyMock – record/replay
  - Jmock – write expectations as you go
  - Mockito – the new kid on the block
- Use dynamic proxies as stubs
- Use StubFor/MockFor
  - inspired by EasyMock
  - no external libraries required (other than Groovy)

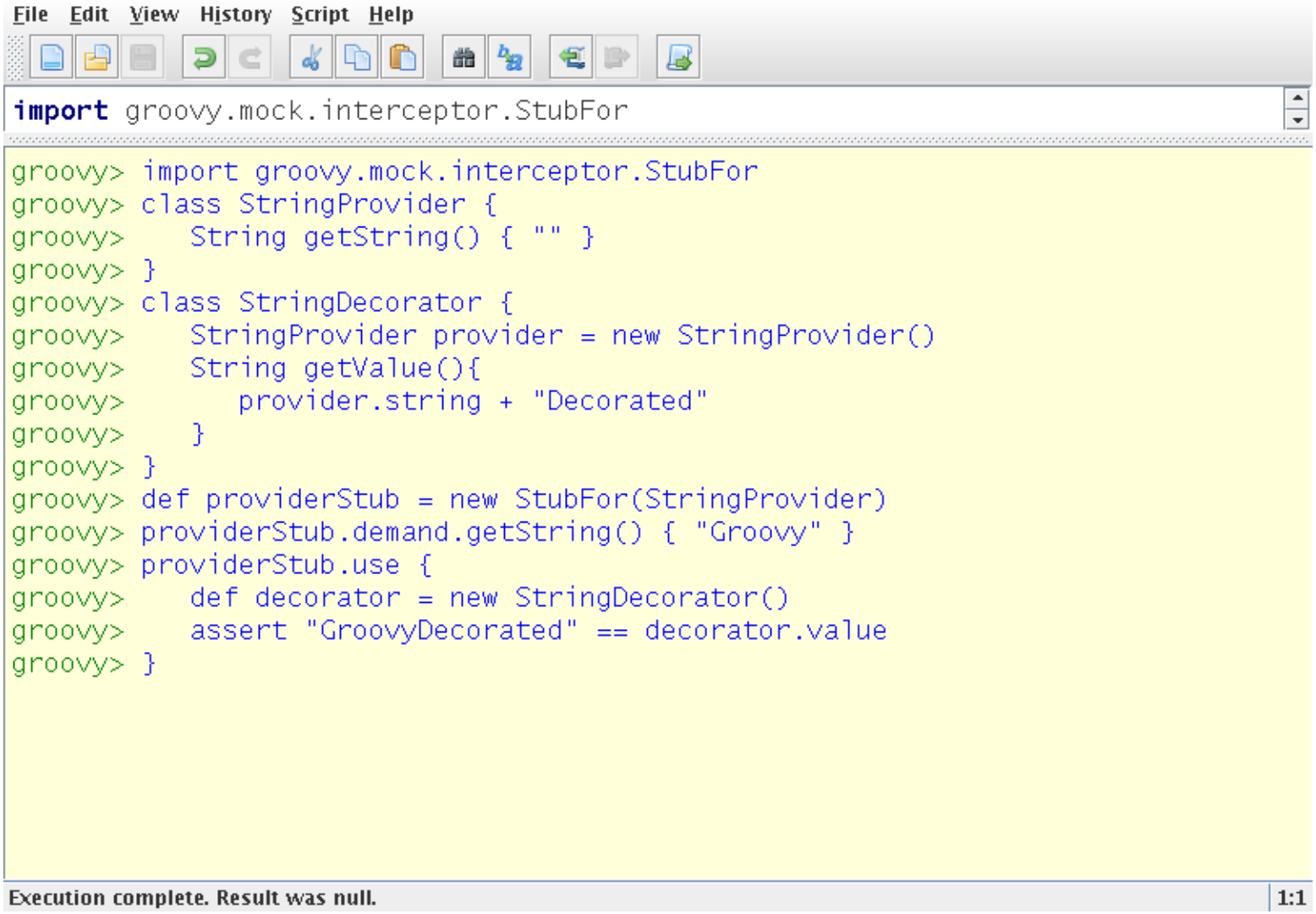

Dynamic Proxies


File Edit View History Script Help
class StringProvider {
groovy> class StringProvider {
groovy>   String getString() { "" }
groovy> }
groovy> class StringDecorator {
groovy>   // This is a property declaration, meaning that
groovy>   // the Groovy compiler will generate a pair of
groovy>   // get/set methods
groovy>   StringProvider provider
groovy>   def getValue() { provider.string + "Decorated" }
groovy> }
groovy> // Here comes the proxy
groovy> def provider = [
groovy>   getString: { -> "Groovy" }
groovy>] as StringProvider
groovy> // it looks like JSON, doesn't it?
groovy> def decorator = new StringDecorator(provider: provider)
groovy> // the following would have worked too
groovy> // def decorator = new StringDecorator()
groovy> // decorator.setProvider(provider)
groovy> assert "GroovyDecorated" == decorator.value
groovy> assert decorator.provider instanceof StringProvider
Execution complete. Result was null.
1:1

StubFor/MockFor

- caller – collaborator
- mocks/stubs define expectations on collaborators
- mocks are strict, expectation must be fulfilled both in order of invocation and cardinality.
- stubs are loose, expectations must fulfil cardinality but may be invoked in any order.
- CAVEAT: can be used to mock both Groovy and Java collaborators, caller must be Groovy though.


Groovy Mocks



XML Processing: testing databases

- DbUnit: a Junit extension for testing databases
- Several options at your disposal
  - Old school – extend DatabaseTestCase
  - Flexible – use an IDataBaseTester implementation
  - Roll your own Database testcase

Inline XML dataset

```
import org.dbunit.*
import org.junit.*

class MyDBTestCase {
   IDatabaseTester db

   @BeforeClass void init(){
      db = new JdbcDatabaseTester("org.hsqldb.jdbcDriver",
            "jdbc:hsqldb:sample", "sa", "" )
      // insert table schema
      def dataset = """
      <dataset>
         <company name="Acme"/>
         <employee name="Duke" company_id="1">
      </dataset>
      """
      db.dataset = new FlatXmlDataSet( new StringReader(dataset) )
      db.onSetUp()
   }

   @AfterClass void exit() { db.onTearDown() }
}
```

Compile-checked dataset

```
import org.dbunit.*
import org.junit.*
Import groovy.xml.MarkupBuilder

class MyDBTestCase {
   IDatabaseTester db

   @BeforeClass void init(){
      db = new JdbcDatabaseTester("org.hsqldb.jdbcDriver",
            "jdbc:hsqldb:sample", "sa", "" )
      // insert table schema
      def dataset = new MarkupBuilder().dataset {
         company( name: Acme )
         employee( name: "Duke", company_id: 1 )
      }
      db.dataset = new FlatXmlDataSet( new StringReader(dataset) )
      db.onSetUp()
   }

   @AfterClass void exit() { db.onTearDown() }
}
```

Functional UI Testing

- These tests usually require more setup
- Non-developers usually like to drive these tests
- Developers usually don’t like to code these tests
- No Functional Testing => unhappy customer => unhappy developer

Groovy to the rescue!

- Web:
  - Canoo WebTest - leverages AntBuilder
  - Tellurium - a Groovier Selenium
- Desktop:
  - FEST – next generation Swing testing
- BDD:
  - Easyb
  - Spock


FEST + Easyb

Resources

http://groovy.codehaus.org

http://junit.org
http://testng.org
http://www.dbunit.org
http://easyb.org
http://easytesting.org

http://groovy.dzone.come
http://jroller.com/aalmiray
twitter: @aalmiray


THINK

Credits

http://www.flickr.com/photos/patrick_pjm/3323599305/
http://www.flickr.com/photos/guitrento/2564986045/
http://www.flickr.com/photos/wainwright/1050237241/
http://www.flickr.com/photos/lancecatedral/3046310713/
http://www.flickr.com/photos/fadderuri/841064754/
http://www.flickr.com/photos/17258892@N05/2588347668/
http://www.flickr.com/photos/chelseaaaaaa/3564365301/
http://www.flickr.com/photos/psd/2086641/