Scripting and dynamic meta-programming for Java developers

Václav Pech

http://jroller.com/vaclav

http://www.vaclavpech.eu

@vaclav_pech

Agenda

- Closures
- Collections
- Operators
- Dynamic typing
- Scripting
- Dynamic meta-programming

Agenda (next lesson)

- Static meta-programming
- Builders
- Domain-specific languages
- DSL frameworks – Grails, Griffon, Gradle

Groovy

A JVM programming language

- Dynamic
- Dynamically-typed
- Scripting
- Object-oriented
- Building on Java syntax

Properties

```
class ProgrammingLanguage {
     String name
     String version
     boolean easy=true
}
def groovy=new ProgrammingLanguage(
          name:'Groovy', version:'1.5', easy:true)

def java=new ProgrammingLanguage(name:'Java')
java.version='1.6'
```

Closures

```
Closure multiply1 = {int a, int b -> return a * b}

Closure multiply2 = {int a, int b -> a * b}

Closure multiply3 = {a, b -> a * b}

def multiply4 = {a, b -> a * b}
```

Closures – implicit parameter

```
def triple1 = {int number -> number * 3}

def triple2 = {number -> number * 3}

def triple3 = {it * 3}
```

Groovy is functional

```
def multiply = {a, b -> a * b}
def double = multiply.curry(2)
def triple = multiply.curry(3)

assert 4 == multiply(2, 2)
assert 8 == double(4)
assert 6 == triple(2)
```

Memoize

```
def triple = {3 * it}

def fastTriple = triple.memoize()
```

Iterations

```
(1..10).each{number -> println number * 3}

1.upto(10) {println it * 3}


Closure triple = {it * 3}

1.step(11, 1){println triple(it)}
```

Collections

```
final emptyList = []
final list = [1, 2, 3, 4, 5]
final emptyMap = [:]
final capitals = [cz : 'Prague', uk : 'London']


final list = [1, 2, 3, 4, 5] as LinkedList
final emptyMap = [:] as ConcurrentHashMap
```

Some operators

```
[‘Java’, ‘Groovy’]*.toUpperCase()

customer?.shippingAddress?.street

return user.locale ?: defaultLocale
```

GDK = JDK + FUN

- java.util.Collection
  - each(), find(), join(), min(), max() …
- java.lang.Object
  - any(), every(), print(), invokeMethod(), …
- java.lang.Number
  - plus(), minus(), power(), upto(), times(), …
- …

Dynamic dispatch

The target method is decided at run-time using run-time type of the arguments

```
def calculate(String value)
def calculate(Integer value)

calculate('10' as Integer) ???
```

Dynamic object creation

```
Runnable r = {println 'Asynchronous'} as Runnable
```

Dynamic object creation

Duck-typing

```
Calculator c = [ add : {a, b, → a + b},
                 multiply : {a, b → a * b},
                 increment : {it + 1}
               ] as Calculator

assert 6 == c.multiply(2, 3)
```

Syntax enhancements

- Dynamic (duck) typing – optional!
- GDK
- Syntax enhancements
  - Properties, Named parameters
  - Closures
  - Collections and maps
  - Operator overloading
  - …

Scripting

Evaluate custom Groovy code

At run-time!!!

```
new GroovyShell().evaluate('println Hi!')
```

http://groovyconsole.appspot.com/

Categories

```
StringUtils.countMatches(myString, 'Groovy')
```

```
use(StringUtils) {
    myString.countMatches('Groovy')
}
```

DSL

- Limited purpose language
- Targeted to a particular domain
- Friendlier API to a framework
  - External
    - SQL, HTML, CSS, …
  - Internal

DSL – Date manipulation

```
use(org.codehaus.groovy.runtime.TimeCategory) {
    println "Tomorrow: ${1.day.from.today}"
    println "A week ago: ${1.week.ago}"
    println "Date: ${1.month.ago + 1.week + 2.hours - 5.minutes}"
    println "Date ${(1.month + 10.days).ago}"
}
```

DSL – Hibernate criteria

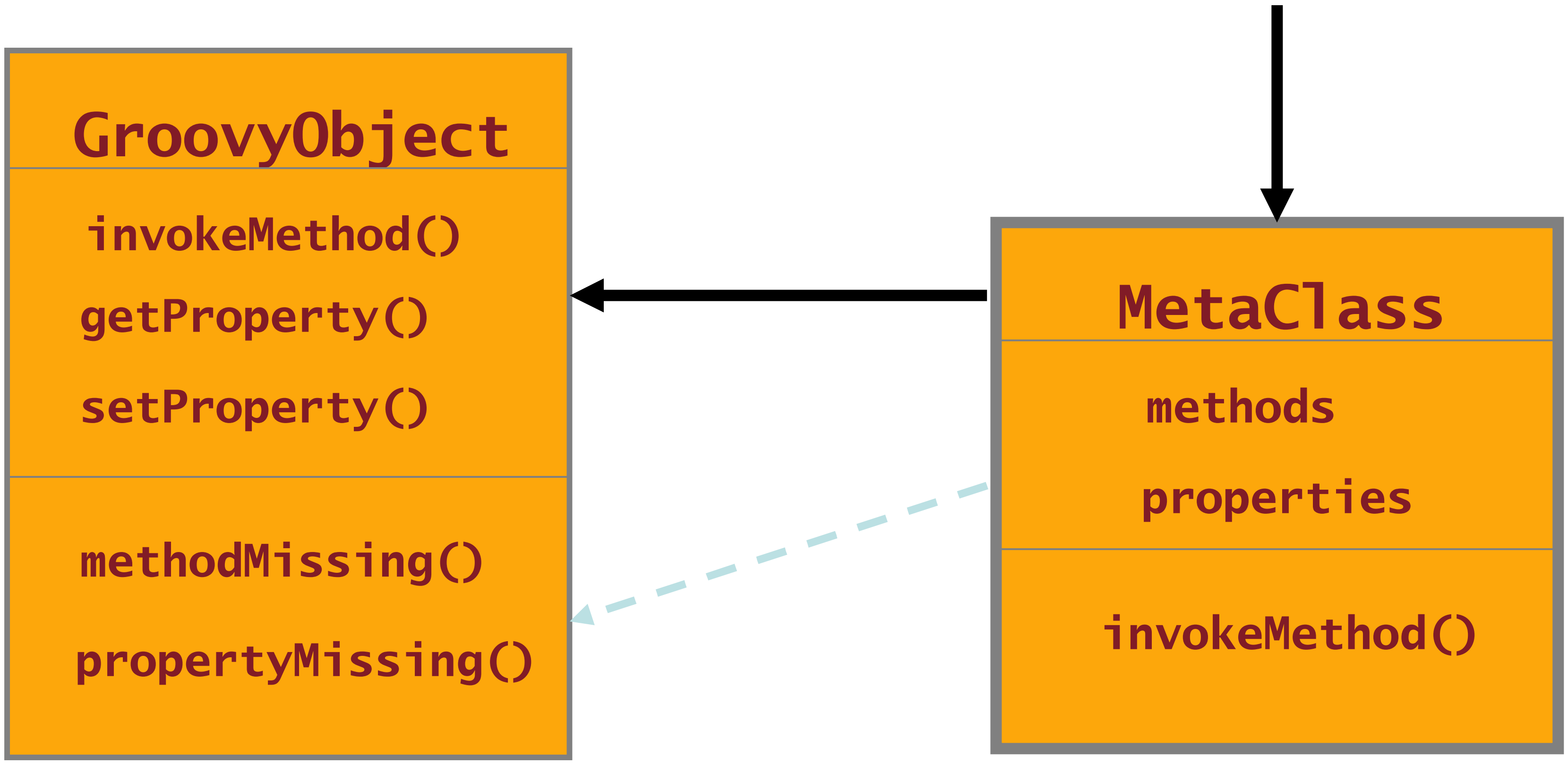
```
def participants = Participant.createCriteria().list {
    gt('age', age)
    or{
        eq('interest', 'Java')
        eq('interest', 'Groovy')
    }
    jug {
        ilike('country', 'de')
    }
    order('lastName', 'asc')
}
```

DSL – Account manipulation

```
Money money = new Money(amount: 350, currency: 'eur')
getAccount('Account1').withDraw money
getAccount('Account3').deposit money
```

```
"Account1" >> 350.eur >> "Account3"
```

Dynamic method invocation



Groovy eco-system

GPars – Groovy Parallel Systems

easyb, Spock

Gaelyk – lightweight Google App Engine framework

Gradle – much better Maven

Grails, Griffon

… (check out http://www.groovy.cz/)

Summary

The power of Ruby for Java programmers

http://jroller.com/vaclav
pech@d3s.mff.cuni.cz

References

http://www.groovy.cz

http://groovy.codehaus.org

http://grails.org

http://groovyconsole.appspot.com/