

DSLs and static meta-programming for Java developers

Václav Pech



<http://jroller.com/vaclav>

<http://www.vaclavpech.eu>

@vaclav_pech

Last time agenda

- Functional programming
- Scripting
- Dynamic typing
- Dynamic meta-programming



Agenda for today

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

Part 1

Groovy syntax and interoperability

Power assert

assert 5 == customer.score

Exception thrown

17.2.2012 12:30:12 org.codehaus.groovy.runtime.StackTraceUtils sanitize

WARNING: Sanitizing stacktrace:

Assertion failed:

assert 5 == customer.score

```
    | |      |
    | |      4
    | [score:4]
false
```

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)
```

Closure scope

owner

delegate

this

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
```


Part 2

Scripting

Scripting

Evaluate custom Groovy code

At run-time!!!

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

<http://groovyconsole.appspot.com/>

Part 3

Dynamic meta-programming

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

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)
```

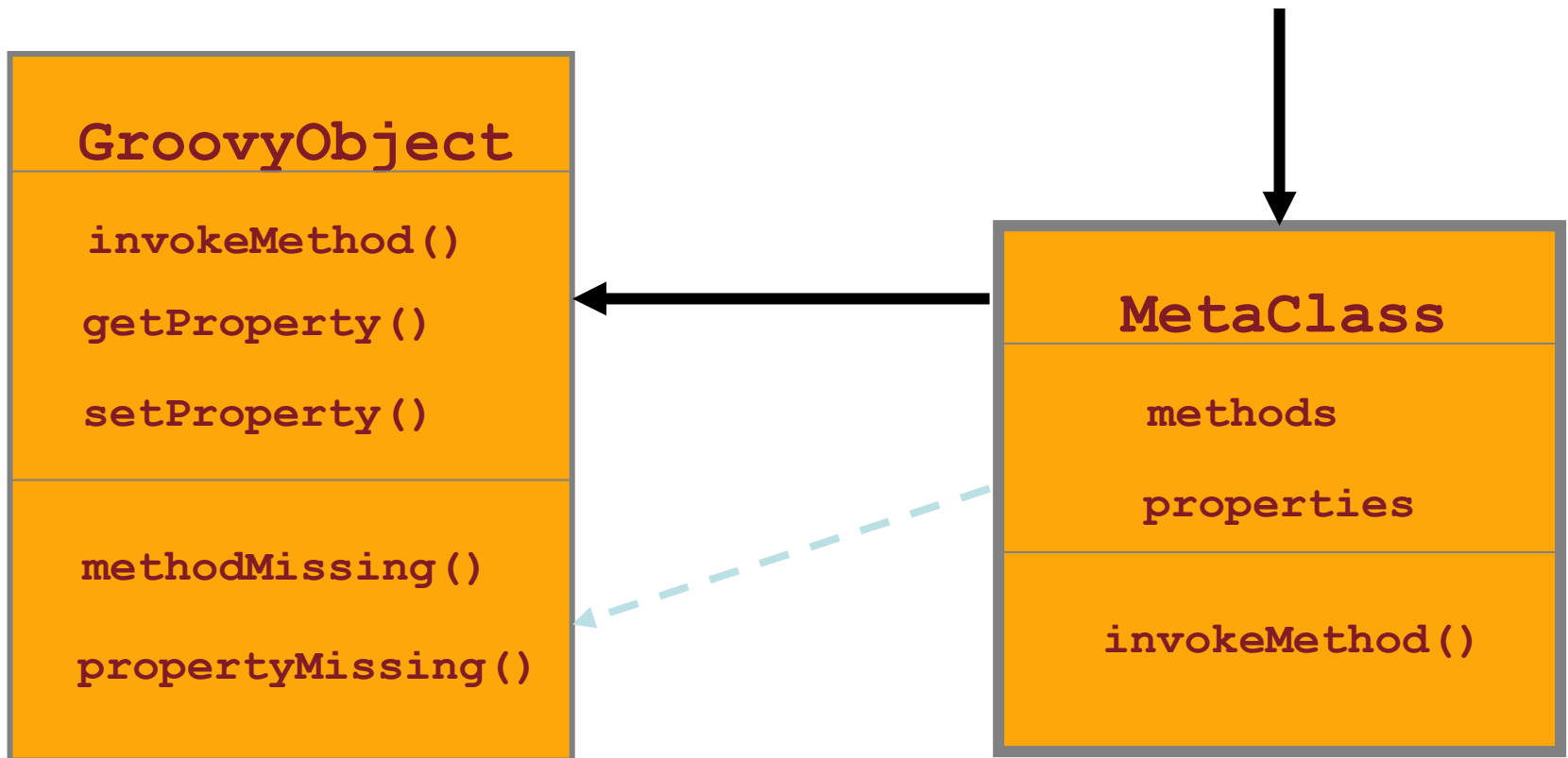
Categories

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



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

Dynamic method invocation



Part 4

Domain Specific Languages


```
File.metaClass.div = { path ->  
    new File(delegate, path)  
}
```

```
File.metaClass.div = { path ->  
    new File(delegate, path)  
}
```

```
def file = new File(".")/'test'/'hello'/'file.txt'
```

Agenda

- Domain-specific languages
- DSL frameworks – Grails, Griffon
- Builders

Grails

Web applications on JVM

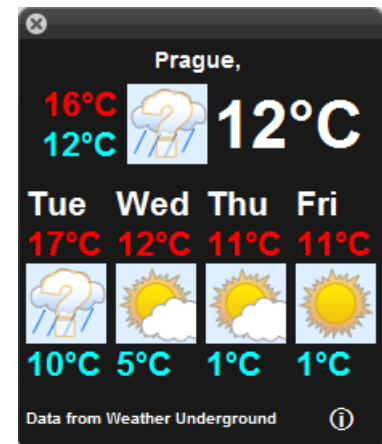
- Hibernate, Spring, ...
- Groovy DSLs
- Scaffolding
- Convention over configuration
- DRY
- KISS

Griffon



Rich-client applications on JVM

- Swing
- Groovy DSLs
- Scaffolding
- Convention over configuration
- DRY
- KISS



BDD - Spock

```
class DataDriven extends Specification {  
    def "maximum of two numbers"() {  
        expect:  
        Math.max(a, b) == c  
        where:  
        a | b | c  
        7 | 3 | 7  
        4 | 5 | 5  
        9 | 9 | 9  
    }  
}
```

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"
```

order cake with plums and apples
and cream to "Malostranske namesti"

```
order(cake) .with(plums) .and(apples)  
 .and(cream) .to("Malostranske namesti")
```

Builders

Construct hierarchies

- html, xml, json, swing, configuration
- objects
- db queries
- ...

Builders - GAnt

```
ant.sequential {  
    myDir = "target/AntTest/"  
    mkdir(dir: myDir)  
    copy(todir: myDir) {  
        fileset(dir: "src/test") {  
            include(name: "**/*.groovy")  
        }  
    }  
    List dirs = ['core', 'lib', 'engine', 'gui', 'db']  
    for(String currentDir:dirs) {  
        String targetDir="target/$currentDir"  
        mkdir(dir:targetDir)
```

Cli Builder

```
def cli = new CliBuilder (usage:'simpleHtmlServer -p PORT -d DIRECTORY')
cli.with {
  h longOpt:'help', 'Usage information'
  p longOpt:'port',argName:'port', args:1, type:Number.class,'Default is 8080'
  d longOpt:'dir', argName:'directory', args:1, 'Default is .'
}

def opts = cli.parse(args)
if(!opts) return
if(opts.help) {
  cli.usage()
  return
}
```

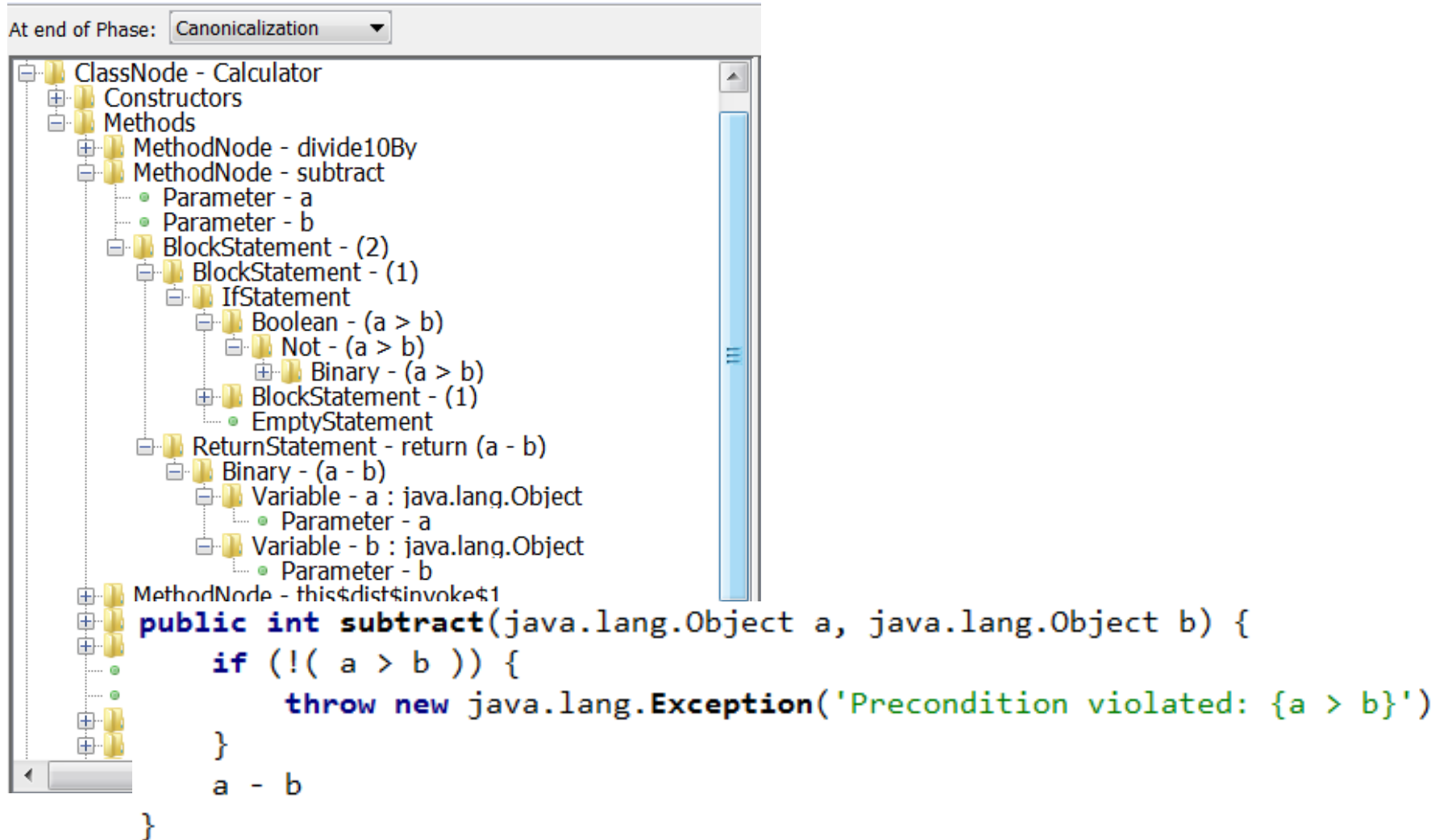
Builders – Spring config

```
dataSource(BasicDataSource) {  
    driverClassName = "org.hsqldb.jdbcDriver"  
    url = "jdbc:hsqldb:mem:shopDB"  
}  
  
sessionFactory(ConfigurableLocalSessionFactoryBean) {  
    dataSource = dataSource  
    hibernateProperties = ["hibernate.hbm2ddl.auto": "create-drop",  
        "hibernate.show_sql": true]  
}  
  
calculator(demo.shop.CalculatorImpl) {bean ->  
    bean.singleton = true  
    bean.autowire = 'byType'  
}
```


Part 5

Static meta-programming

AST



AST Transformations

```
class Registrations {  
    @Delegate List items = []  
}
```

```
def people = new Registrations()  
people.addAll(["Joe", "Dave"])  
assert ["Dave", "Joe"] == people.reverse()
```

@Delegate, @Immutable, @Singleton

@Lazy

@TupleConstructor

@InheritConstructors

@Canonical

@ToString

@EqualsAndHashCode

@Log, @Log4j, @Commons

@Synchronized

@WithReadLock

@WithWriteLock

@AutoClone, @AutoExternalize

...

Creating AST Transformations

```
new AstBuilder()
```

```
    .buildFromString()
```

```
    .buildFromCode()
```

```
    .buildFromSpec()
```

```
.buildFromString ("
    Integer.parseInt("$param")
")
```

```
.buildFromCode (  
    Integer.parseInt("$param")  
)
```



```
.buildFromSpec {  
  method('convertToNumber', ACC_PUBLIC, Integer) {  
    parameters { parameter 'parameter': String.class }  
    exceptions {}  
    block {  
      returnStatement {  
        staticMethodCall(Integer, "parseInt") {  
          argumentList {  
            variable "parameter"  
          }  
        }  
      }  
    }  
  }  
}
```

Type-checking/Static

@TypeChecked, @CompileStatic

@TypeChecked

```
String test(Object val) {  
    if (val instanceof String) {  
        val.toUpperCase()  
    } else if (val instanceof Number) {  
        val.intValue() * 2  
    }  
}
```

Summary



The joy of Ruby for Java programmers

<http://jroller.com/vaclav>
vaclav@vaclavpech.eu

References

<http://www.groovy.cz>

<http://groovy.codehaus.org>

<http://grails.org>

<http://groovyconsole.appspot.com/>

<http://www.manning.com/coenig2/>