L2. Agile Software Development (I)

Introduction to Groovy

Dr A Boronat

Table of Contents

Challenge in Software Development

Goal

- Software release: software that is developed and tested, i.e. our goal
- Build: software that is compiled and assembled (a jar file), intermediate goal to achieve a release

Problems in release management

- Does the code compile?
- Does the code pass the tests? (unit tests)
- Does the code meet the business requirements? (functionality)
- Does the code meet the quality criteria? (performance, security, etc.)

Challenge in Software Development

Solution: continuous delivery

- automatically produce build artifacts (jar files)
- release often and small
- produce a Minimum Viable Product (MVP)
    - to obtain fast feedback from customers
    - reducing risks
    - ensuring continuous progress

How? Agile Methodology

- Iterative, incremental and evolutionary
- People not process (when sensible)
- Focus on quality and on maintaining simplicity
  - continuous delivery: automate as much as possible
    - optimise resources: save time
    - increase quality: to achieve repeatable and consistent processes
  - automated testing
    - quantitative measures
    - consistency
    - release readiness
- Embrace change: very short feedback loop and adaptation cycle

HOW? Tooling

Groovy

gradle

compile > test > build > release

spring by Pivotal

JPA Java Persistence API

HIBERNATE

MySQL

GitHub

SPOCK FRAMEWORK

Gradle: a DSL for Build Automation

Gradle uses Groovy as scripting language to automate builds

Groovy

- Scripting language:
  - no need to declare types
  - facilities for dealing with regular expressions and files
  - versatile syntax
- JVM language: Java-like syntax and Java integration
- In the top 20 of the most popular programming languages:
  - according to TIOBE's index – September'16: #1 Java, #16 Groovy
  - according to TIOBE's index – September'15: #1 Java, #34 Groovy
  - according to RedMonk's ranking – June'16: #2 Java, #20 Groovy
  - according to RedMonk's ranking – June'15: #2 Java, #19 Groovy

Who uses Groovy?

Groovy: syntax

- Java syntax supported (with a few differences):

```
system.out.println("Hello, World!");
```

- but more versatile:

```
println "Hello, World!"
```

- Declaration of variables

```
def var = "Hello, World!"
```

- Strings:
    - single quotes: a string
    - double quotes: string interpolation
    - triple single/double quote: multiline strings

```
def course ='CO2006'
def string="""
Hello:
$course
"""
println string
```

Groovy: lists and ranges

```
def letters = ['a', 'b', 'c', 'd']
// accessing a member of the list
assert letters[0] == 'a'
// appending
letters << 'e'
// looping a list
for (letter in letters) {
   println letter
}
// ranges
for (number in 1..3) {
   println number
}
```

Groovy: functions

- function declaration

```
def isEven(num) {
  num % 2 == 0
}
```

- function call

```
isEven(2)
```

- function composition

```
def isEven(num) {
  num % 2 == 0
}
def mult2(num) {
  num * 2
}
isEven(mult2(15))
```

Groovy: closures

- block of code that may have parameters

```
{ it -> println it }
{ println it } // it is always included implicitly
```

- calling a closure

```
def printMe = { println it }
printMe 'hello'
```

- closure composition

```
def plus2  = { it + 2 }
def times3 = { it * 3 }
def times3plus2 = times3 >> plus2
times3plus2(5) // result is 17
(times3 >> plus2)(5) // result is 17
```

- closures can be used as parameters

```
[1,2,3].collect({ it + 2 }) // output: a new array [3, 4, 5]
[1,2,3].each({println "Number $it"}) // it may modify the input
[1,2,3,4].find({it % 2 == 0}) // output: 2
[1,2,3,4].findAll({it % 2 == 0}) // output: [2, 4]
[1,2,3,4].any({it % 2 == 0}) // output: true
[1,2,3,4].every({it % 2 == 0}) // output: false
```

Goals for this sprint

- Getting familiar with software infrastructure to develop the mini project (a secure web application with Java):
  - Agile practices
  - Gradle
  - Eclipse
- Test: Agile, Groovy and Gradle

Goals for this week

TODO list (sprint backlog) on Blackboard:

- ☐ Eclipse videos (revision) - optional but recommended
- ☐ Set up your GitHub repository
- ☐ Agile methodologies: videos and learning check
- ☐ Groovy: videos and learning check
- ☐ Groovy exercises

Feedback

Exercises

- solutions to be released next Monday
- laboratory session next Monday to discuss any questions you may have about the exercises

I'm stuck...

- **DO NOT** wait until Monday
- **ASK** in the discussion forum on Blackboard:
  - check if your question is related to an existing thread (it may have been answered already)

Learning resources

- Pluralsight
- Exercises on GitHub
- Groovy documentation
- Groovy cheatsheet
- Groovy web console