Univerzita Hradec Králové
Fakulta informatiky a managementu

Groovy – agilní Java

Pavel Kříž
Filip Malý

Úvod

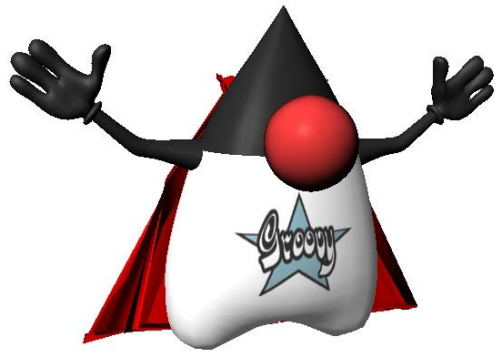
- Dynamický skriptovací jazyk pod JVM
- Navržen pro platformu Java, zcela interoperabilní (na úrovni bajtkódu) s běžnými programy v Javě, silně objektově orientovaný
  - Do značné míry „rozšířením“ jazyka Java
  - Berme ho jako doplněk této platformy
- Syntaxe vychází z Javy, přidává nové možnosti

Úvod

- Java silně statický jazyk (C++ nebo C#)
- „Konvenční jazyky“ z hlediska syntaxe přesné
  - Lze definovat přesně všechny typy = lepší předvídatelnost kódu
  - Na druhé straně může být právě tato konvenčnost tím, co vývojáře zpomaluje

Groovy

- Primárně navržen pro platformu Java
- Syntaxe vývojářům v Javě poměrně blízká
- Nové vlastnosti lze začít využívat i s nulovými zkušenostmi s tímto jazykem (při znalosti Javy)



Groovy

- Snadné psaní skriptů
- Dynamicky rozšiřuje třídy z JDK o nové metody
- Rozšiřuje syntaxi
- properties, closures, jmenné parametry, jednodušší práce se seznamy a mapami, traverzování stromem objektů, regulární výrazy, zpracování XML

Groovy

- Meta-programování (dynamické vyvolávání metod, definice metod nebo celých třídu za běhu)
- Koncept builderů pro XML, HTML, Swing,…
- Tvorba DSL
- Vkládání Groovy výrazů do řetězců

GDK, dokumentace

- Groovy JDK API Specification
- http://groovy.codehaus.org/groovy-jdk/

- „This document describes the methods added to the JDK to make it more groovy.“

Java a Groovy

Java

```
for (String it : new String[] {"Rod", "Carlos",
 "Chris"})
    if (it.length() <= 4)
        System.out.println(it);
```

Groovy

```
["Rod", "Tom", "Chris"].findAll{it.size() <= 4
 }.each{println it}
```

Grails

- Prostředí pro rychlý vývoj Java EE aplikací
- Silně inspirované **Ruby on Rails**
- Nosným jazykem **Groovy**
- Koncepčně je aplikace v Grails založena na MVC

Motivace pro dynamické jazyky

- Dynamické metody (findById, findByName v GORM atp.)
- Dynamicky přidané metody ke stávajícím třídám – příklad z JavaScriptu s insertAfter u DOMu
- Spouštění dynamicky vytvořeného kódu („uživatelská makra“)

Podpora v IDE

- Obecně nelze čekat „zázraky“
- Pluginy do majoritních IDE jsou různých kvalit
- Eclipse
  - http://docs.codehaus.org/display/GROOVY/Install+Groovy-Eclipse+Plugin
  - http://docs.codehaus.org/display/GROOVY/Create+Your+First+Groovy+Project

Groovy script

- Koncovka *.groovy
- Příklad „Hello world“
  - V Javě nutno vytvořit třídu
  - V Groovy rovnou **println "Hello world"**
- Spuštění skriptu:

  groovy script.groovy
- Kompilace do .class:

  groovyc script.groovy

Example1a, Example1b

Syntaxe

- Chybí středník
- Lze vynechat return
  - metody vrací vyhodnocenou hodnotu na posledního řádku, není třeba deklarovat návratový typ
- Nepovinné datové typy (v zásadě instance objektů)

Example2a, Example2b

Syntaxe

- Dynamické i statické typování
- Defaultní importy základních balíků
  - java.lang, java.util, java.io, java.net
- Bezpečné dereferencování pomocí ?.
  - Ošetření null
  - Pokud by mělo dojít k java.lang.NullPointerException, vrací

Example3a

Syntaxe

- Vše je objekt
  - Např. i číslo (použity objektové wrappery)
    - Při komunikaci s Javou zafunguje auto(un)boxing
- Lze přetěžovat operátory
  - Tím lze přidat další „syntaktický cukr“

    1+1 v groovy vede na volání
    1.plus(1)
  - Využitelné např. pro operace přidávání do pole, aritmetika s vektory atp.

Example4

Nekompatibility s Javou

- Omezení u vnitřních tříd
- Jiný význam operátoru ==
  - Je přetížen na .equals()
  - Původní porovnání odkazů lze provést pomocí .is()
- Nová klíčová slova

JavaBean v Groovy – GroovyBean

- Snaha redukovat množství kódu
- Žádné gettery, settery
  - Vytvořeny automaticky
  - Klasifikace proměnných pomocí nastavení viditelnosti, příp. deklarování jako final
  - private = žádný getter, setter
  - final pouze getter

Example3a, Example3b

GStrings

- Vložení proměnné do řetězce
  - Řetězec v úvozovkách
    “hodnota=$promenna”
  - U řetězce v apostrofech se proměnné nevkládají
  - Analogie s Perlem, PHP,…, JSP EL
  - Složitější výrazy se složenými závorkami

Example5

Kolekce

- Zkrácená inicializace kolekcí
  - List
  - Map
- Zkrácený přístup k prvkům
  - pole[1]
  - mapa[‘klic’]
  - mapa.klic

Example6

Closures - uzávěry

- Closure = blok kódu zapouzdřený do objektu (typu Closure)
  - Může obdržet parametry a vracet hodnotu
  - Blízká analogie k anonymním vnitřním třídám z Javy
  - Jeho instance vznikne blokem složených závorek

Example7

Closures – uzávěry

- V uzávěru lze pracovat s lokálními proměnnými z místa jeho definice
- Lze ovlivit na čem (delegate) se mají volat metody volané uvnitř uzávěru
  - Využito např. v tzv. Builderech (viz dále)
- Jejich význam oceníme při dobře navrženém API pro práci s nimi
- Groovy rozšiřuje mnohé třídy z JDK právě o uzitečné metody, které pracují s uzávěry

```
new File("soubor.txt").eachLine { println it }
```

Zpracování XML

- Standardní prostředky
  - SAX parser
  - DOM parser
- Groovy prostředky
  - DOMCategory
  - XmlParser
  - XmlSlurper
    - umí procházet strom XML tím, že dynamicky „předstírají“ existenci patřičných properties, př.: zamestnanci.zamestnanec[3].jmeno.text

Použití Groovy na UHK

- Projekt interaktivní učebnice fyziky pro střední školy (Pedagogická fakulta)
  - Preference opensource a multiplatformního SW
  - formát Scalable Vector Graphics (SVG) – XML
  - tvorba: editor InkScape
  - běh: XUL aplikace (Mozilla Gecko)
  - Dávkové zpracování
    - převedení textu na křivky
    - hromadné úpravy ve stránkách
    - hromadné generování stránek (multimédia)

Komplexní příklad: Graf v SVG

- Přečtení CSV souboru – data
- Přečtení SVG (XML) – šablona
- Přidání elementů do SVG (sloupce)
- Zápis SVG

- Groovy JDK, XmlSlurper

Komplexní příklad: Graf v SVG 2

- Úprava předchozího příkladu
- Místo CSV budeme číst XLS pomocí Apache POI

+ ExcelBuilder (elegantní čtení XLS)

Zkušenosti

- Vývoj v Groovy v mnoha ohledech jiný oproti Javě
- Kratší kód – snazší orientace, „snippets“, může i zhoršit čitelnost
- Nelze zajistit kvalitní doplňování kódu v editoru (nutnost časté konzultace s dokumentací, resp. s různými návody)
- Většina chyb se projeví až při běhu (testování se stává téměř nutností)
- Během ladění je třeba rozumět vnitřním principům (pro začátečníka mnohdy nesrozumitelné vyjímky)

Závěr

- Příjemně čitelná syntaxe, velká flexibilita
- Díky vazbě Groovy na platformu Java má velké možnosti využití jejích výhod, již existujícího kódu
- Výhodou přístupnost javovským vývojářům
- **Vychází syntakticky z Javy a lze začít používat nové užitečné vlastnosti postupně**
  - Nebývá příliš zdůrazněno, potenciální zájemce může po zhlédnutí ukázkových kódů v Groovy nabýt mylného dojmu, že je to zcela nový jazyk, jehož syntaxi se bude muset dlouho učit

Zdroje

- Root.cz http://www.root.cz/clanky/groovy-v-prikladech-uvod-do-jazyka/
- Java.cz http://www.java.cz/detail.do?articleId=8020
- Feeling Groovy http://www.ibm.com/developerworks/java/library/j-alj08034.html
- http://groovy.codehaus.org/Processing+XML
- Dierk König: Groovy in Action

Děkujeme za pozornost

Pavel Kříž, Filip Malý