Porównanie zrębów : Ruby on Rails i Lift

Zagadnienia Programowania Obiektowego

Mateusz Kopeć, Piotr Wojnarowski

Wydział Matematyki, Informatyki i Mechaniki
Uniwersytet Warszawski

14.12.2009

Spis treści

Spis treści

Ruby - wprowadzenie

Czym jest Ruby?

- Ruby to `interpretowany` język programowania
- Powstał w 1995 roku, stworzony przez Yukihiro Matsumoto
- Podstawa popularnego zrębu Ruby on Rails
- 2005 rok - 2 książki o Ruby i RoR najlepiej sprzedawanymi pozycjami z kategorii Programowanie
- Aktualna stabilna wersja - 1.9.1
- Polecam spróbować w przeglądarce: `http://tryruby.sophrinix.com/`

Najważniejsze cechy Rubiego

- wieloparadygmatowość (funkcyjność, obiektowość, imperatywność, refleksyjność)
- otwarte oprogramowanie
- przenośność (Unix, DOS, Windows, Mac OS X, BeOS itd.)
- wiele implementacji: Ruby MRI, YARV, JRuby, Rubinius
- prosta składnia
- automatyczne odśmiecanie pamięci
- przeciążanie operatorów
- obiektowość

Najważniejsze cechy Rubiego c.d.

- obsługa wyjątków
- wyrażenia regularne wbudowane w składnię
- liczby całkowite o dowolnych rozmiarach
- dodawanie metod do obiektów
- bloki i lambda wyrażenia
- „Duck typing” - dynamiczne typowanie
- moduły - rodzaj wielodziedziczenia
- możliwość zmiany praktycznie wszystkiego

Składnia

- Przypomina Perla, Pythona czy Smalltalka
- ..ale wcięcia nie mają znaczenia
- Zwięzłe, czytelne programy
- ..dzięki wszechobecnej obiektowości

Przykład

```
5.times { print "Witaj!" }
1 + 2    # to to samo, co:
1.+(2)   # i:
1.send "+", 2
```

Składnia

- Przypomina Perla, Pythona czy Smalltalka
- ..ale wcięcia nie mają znaczenia
- Zwięzłe, czytelne programy
- ..dzięki wszechobecnej obiektowości

Przykład

```
5.times { print "Witaj!" }
1 + 2     # to to samo, co:
1.+(2)    # i:
1.send "+", 2
```

Obiektowość

Przykład

```
class Person
  attr_reader :name, :age
  def initialize(name, age)
    @name, @age = name, age
  end
  def <=>(person)
    @age <=> person.age
  end
  def to_s
    "#@name (#@age)"
  end
end
```

Bloki, domknięcia, wyrażenia lambda

Przykład 1

```
search_engines =
  %w[Google Yahoo MSN].map do |engine|
    "http://www." + engine.downcase + ".com"
  end
```

Przykład 2

```
lambda{|a| p a+a}
# procedury to obiekty!
p = proc{|a| p a+a}
p.call 7
# wynik: 14
```

Mixin-y i dziedziczenie

Brak wielodziedziczenia, ale...

Przykład

```
class MyArray
  include Enumerable
end
```

Elastyczność Rubiego

Przykład

```
class Numeric
  def plus(x)
    self.+(x)
  end
end
y = 5.plus 6
# y wynosi teraz 11
```

Dodawanie metod do obiektów

Przykład

```
class Foo
    def greet
        print "Hello, world!\n"
    end
end

x=Foo.new; y=Foo.new

class << y
    def greet
        print "Goodbye, world!\n"
    end
end
```

Spis treści

Historia

- Twórca- Martin Odersky
  - Pizza - funkcyjność na JVM (funkcje wysokiego rzędu, klasy generyczne, dopasowywanie wzorców)
  - Generic Java
  - javac
  - "I wanted to start with a clean sheet, and see whether I could design something that's better than Java."
- Oparty o JVM, integracja z Javą (można używać na .NET)
- Początek prac: 2002, pierwsza publiczna wersja: 2003, wersja 2.0: 2006, obecnie zbliża się 2.8

System typów

Statyczne i silne typowanie. Ale kompilator jest inteligentny:

Przykład

```
scala> val x = "abc"
x: java.lang.String = abc

scala> def add(i: Int, j: Int) = i + j
add: (Int,Int)Int
```

Ważne cechy języka

- Skalowalny- skrypty i wielkie aplikacje, wydajny (jak Java)
- XML jako część języka

Przykłady

```
scala> val xml = <div>abc</div>
xml: scala.xml.Elem = <div>abc</div>
```

- Minimalna składnia, operatory są metodami

Przykład

```
scala> val x = (1).+(2)
x: scala.Int = 3
```

Funkcyjność: funkcje

Deklaracja

```
val f: Int => String = x => "Number: " + x
def w42(f: Int => String) = f(42)
```

Częściowa aplikacja

```
def plus(a: Int, b: Int) = "Sum is " + (a + b)
val p = plus(42, _: Int)
```

Funkcyjność: domknięcia

Przykłady

```
var state = 1
val f = (i: Int) => state += i
f(42)
//state == 43
```

Funkcyjność cd.

- Parametry typów
  - def append[U >: T](x: U) = ... //U musi być nadklasą T
  - def append[U <: T](x: U) = ... //U musi być podklasą T
- Dopasowywanie wzorców
- Niezmienność (val kontra var)

Obiektowość

- Funkcja też jest obiektem
- Nie ma typów prostych

Przykłady

```
scala> (1).toString
res2: java.lang.String = 1
```

- Obiekt jest obiektowy (a statyczność nie)

Trait: Cechy Scali

Trait =

- Interfejs z Javy (kontrakt)
- + Mixin z Rubiego (częściowa implementacja)
- bez wielodziedziczenia (linearyzacja)

Przykłady

```
class Animal
trait Furry extends Animal
trait HasLegs extends Animal
trait FourLegs extends HasLegs
class Cat extends Animal with Furry with FourLegs
//Cat -> FourLegged -> HasLegs -> Furry -> Animal
```

Współbieżność z aktorami

Współbieżność z aktorami

- Aktor- asynchroniczny komponent odbierający, wysyłający (actor ! msg) i odpowiadający (reply()) na wiadomości
- Aktor to nie wątek. Kiedy aktor czeka, wątek jest zwalniany. W pamięci mamy kontynuację.
- Niezmienność => nie potrzeba blokad

Spis treści

Popularność

Indeks TIOBE:

1. Java 18.373%
2. C 17.315%

...

10. Ruby 2.404%

...

34. Scala 0.244%

Twitter
Przykład z prawdziwego świata

Ruby:

1. "We like that it's such a full featured language, that it's fun to code in."
2. Kiepskie wątki, problemy z długo żyjącymi procesami, gorsza wydajność
3. "I think it may just be a property of large systems in dynamic languages, that eventually you end up rewriting your own type system, and you sort of do it badly. You're checking for null values all over the place."

Twitter
Przykład z prawdziwego świata

Scala:

1. ”Some of the core collection libraries in Scala are not quite up to snuff yet. And apparently they are working on that right now."
2. "It still seems like IDE and editor support is, perhaps not in its infancy, but in its awkward teenage years."

Spis treści

Ruby on Rails - filozofia

Ruby on Rails powstał w 2004 roku, stworzony przez Davida Heinemeiera Hanssona. Filozofia języka Ruby opiera się na:

- DRY - „Don't Repeat Yourself”
- „Convention over Configuration” (Konwencja Ponad Konfiguracją)
- REST (Representational State Transfer)

Lift - Historia

- Twórcą Lift jest David Pollak. Wersja 1.0 została opublikowany w lutym 2009. Obecnie mamy wersję 1.6
- Chyba najnowszy zrąb webowy, wyciąga wnioski z wcześniejszych:
  - Rails - znakomita obsługa CRUD
  - Seaside - podział na komponenty
  - Wicket - View first
- Lift -> Scala -> JVM (bliblioteki Javy, Tomcat, Maven, Jetty, ...)

Spis treści

Lift - Konfiguracja

Czego nam potrzeba? (Przykładowy zestaw)

- Eclipse + Eclipse Scala Plugin
- Maven + folder artefaktów http://scala-tools.org/

Żadnych konfiguracyjnych plików xml, zamiast nich klasa bootstrap.Boot

RoR - Jak zacząć?

Konieczna jest instalacja:

- języka Ruby
- RubyGems
- systemu zarządzania bazą danych

A jeślibyśmy chcieli się pochwalić innym:

- serwera WWW
- odpowiedniego dodatku pozwalającego uruchamiać Ruby na serwerze

RoR - Tworzenie projektu

Spróbujmy...

Spis treści

RoR - widoki

- Wyraźny podział MVC
- Wsparcie XML
- REST

Lift - Renderowanie: snippety i widoki

- Widok może się zakraść do kontrolera, logika do widoku nie (View first)
- Renderowanie oparte na komponentach i XML
- Snippet (strzęp)- funkcja wypełnia XML pożądanymi wartościami

Lift - Renderowanie: widoki

View (widok) - funkcja generuje kod XMLa (LiftView, InsecureLiftView, dispatch)

Widok

```
class ExpenseView extends LiftView {
 override def dispatch = {
    case "enumerate" => doEnumerate _
 }
 def doEnumerate () : NodeSeq = {
    <lift:surround with="default" at="content">
     { expenseItems.toTable }
    </lift:surround>
  }
}
```

Lift - Mapa strony

Z poziomu metody Boot możemy zarządzać dostępnością poszczególnych stron, dzielić je na grupy, ukrywać

- Menu.builder, .title, .group, .item
- Hidden, If (i Unless), LocGroup, Template (wskazuje wzorzec dla tego odsyłacza), Snippet, LocSnippet (wskazują snippet do wykonania)

Interaktywny chat, Comet

Pokaz praktyczny.

Spis treści

Lift

- Konfiguracja w Boot
- Dostępne zręby: Mapper, Recored (także integracja z JPA)
- Wszystko obiektowo (choć można też pisać zapytania)
  - By, NotBy, By_>, ByList (IN), NullRef, Like
  - TreeNode.findAll(ByRef(TreeNode,parent, TreeNode.id))
  - BySql(”amount between ? and ?”, lowVal, highVal))
- MetaMegaProtoUser

RoR a bazy danych

- Prosta konfiguracja
- Podział na bazę rozwojową, testową i produkcyjną
- Odsunięcie od SQLa

Spis treści

Główne cechy RoRa

+ Dobra dokumentacja
+ Zaangażowana społeczność
+ Szybki start
+ Skalowalność (mimo wszystko)
+ Wyraźny podział MVC

– Przeciętna wydajność
– Problemy wpasowania w inne systemy

Główne cechy RoRa

+ Dobra dokumentacja
+ Zaangażowana społeczność
+ Szybki start
+ Skalowalność (mimo wszystko)
+ Wyraźny podział MVC

– Przeciętna wydajność
– Problemy wpasowania w inne systemy

Główne cechy Lifta

+ Dobra wydajność
+ Szybko rosnąca społeczność
+ Oparty na języku, który daje duże możliwości (ale zawsze można pisać jak w Javie)
+ Możliwość integracji z Javą
+ Nie ma logiki w widoku
+ Wbudowane wsparcie dla „sieci czasu rzeczywistego”

– Mała społeczność (google group: ok. 1600 członków, 14 XII 09)
– Słaba dokumentacja
– Wymaga kompilacji

Główne cechy Lifta

+ Dobra wydajność
+ Szybko rosnąca społeczność
+ Oparty na języku, który daje duże możliwości (ale zawsze można pisać jak w Javie)
+ Możliwość integracji z Javą
+ Nie ma logiki w widoku
+ Wbudowane wsparcie dla „sieci czasu rzeczywistego”

– Mała społeczność (google group: ok. 1600 członków, 14 XII 09)
– Słaba dokumentacja
– Wymaga kompilacji

Zwycięzca

. . . czas pokaże

Zwycięzca

. . . czas pokaże

Więcej informacji I

- Oficjalna strona Ruby.
  (Także po polsku)
  http://www.ruby-lang.org
- Oficjalna strona Ruby on Rails.
  (Także po polsku)
  http://www.rubyonrails.org
- Przewodniki po Ruby on Rails.
  (Wersja angielska bardziej rozbudowana)
  http://apohllo.pl/guides/
- Strona domowa Scali
  http://www.scala-lang.org/

Więcej informacji II

Twitter on Scala. A Conversation with Steve Jenson, Alex Payne, and Robey Pointer

Bill Venners.

http://www.artima.com/scalazine/articles/twitter_on_scala.html

Strona domowa Lifta

http://liftweb.net/

The Definitive Guide to Lift

Chen-Becker, Danciu, Weir

http://groups.google.com/group/the-lift-book

Więcej informacji III

Lift: View first
David Pollak.
http://wiki.liftweb.net/index.php/Lift_View_First