わんくま同盟 富山勉強会 #01

JavaからScalaへ

西本 圭佑 (NISHIMOTO Keisuke)

keisuken@cappuccino.ne.jp

2008.11.08 **富山県民会館**

お品書き

- 概要
- 文法
- 開発
- ライブラリ・フレームワーク
- Javaとの関係
- まとめ
- デモ
- 質疑応答

自己紹介

■緒言 

- ●西本 圭佑 (NISHIMOTO Keisuke)
- ●Twitter: @keisuke_n (follow,remove,block自由に)

■趣味

- ●Java処理系でプログラムを書くこと
- ●最近はScalaがお気に入り

■仕事

- ●主にWebアプリケーション開発
- ●GUI/マルチメディアも

自己紹介

■最近の活動

- ●第29回 Ruby/Rails勉強会@関西
  - •Ｓｃａｌａ on JRuby by keisukenさん
- ●Scala勉強会@岡山-1, @関西-1, @関西-2
- ●某Webフレームワーク

■今後の予定

- ●交通博物館へGo! (鉄分補給)
- ●某Webフレームワーク勉強会@ねっと

概要

■What is Scala? (本家より)

- Scala is a general purpose programming language designed to express common programming patterns in a concise, elegant, and type-safe way. It smoothly integrates features of object-oriented and functional languages. It is also fully interoperable with Java.
- Scala**は簡潔で、上品で、型安全な方法で一般的なプログラミングパターンを表現できるように設計された汎用のプログラミング言語です。それはスムーズにオブジェクト指向と関数型言語を統合します。また、**Java**と共に完全に相互運用できます。**

*概要

■Scalaとは

- ●EPFL(スイス連邦工科大学)を中心に開発
- ●オブジェクト指向と関数型を融合
- ●静的型付のコンパイル型言語

■処理系

- ●実行環境
  - Java 5以上の実行環境 (JRE/JDK)
    - JavaME CLDC/Androidでも動いているらしい
  - .Netでも動くらしい
- ●ライセンス: Scala Lisence (BSD-style Lisence)

概要

■ロゴ

(新しくなりました)

*概要

■普及

- 比較的新しい (4-5年程度)の言語
- 注目され始めて2年くらい

■特徴

- Javaや.NET(CLR)のバイトコードにコンパイル
  - JavaやC#のようにクラスファイル/実行型ファイル
  - パッケージ化可能(JAR, EXE)
- 軽量な文法
  - 開発のハードルを下げる
- インタプリタ
  - 対話環境
  - スクリプト

概要

■Scalaのオブジェクト指向言語としての側面

●便利なアプローチ

- クラス, メソッド
- 継承
- アクセッサ
- インスタンス
- etc.

●グループでの開発

- 情報隠蔽
- パッケージ
- etc.

概要

■Scalaの関数型言語としての側面

●便利なアプローチ

- クロージャ
- カリー
- パターンマッチング
- etc.

●より厳格なプログラミング

- 副作用をなくすプログラミングのサポート
  - immutableな代入: val
  - immutableなコレクション: scala.collection.immutable._

概要

■関数型言語概要

- ラムダ計算の概念をプログラミング言語として体現したものである。すべての計算は関数の評価によって行われる
- 関数型言語の多くは、カリー化、遅延評価などの機能を備えている。また、静的型付けの物の多くは型推論の機能を持つ
- Wikipediaより

概要

■関数型言語の種類

●純粋関数型・静的型付け

- Haskell, Cleanなど

●非純粋関数型

- 静的型付け
  - F#, ML, OCaml, Scala
- 動的型付け
  - Erlang, Lisp, Scheme

概要

■関数型言語概要

●主な機能･文法

- 無名関数･クロージャ
- カリー化
- 再帰呼び出し(および最適化)
- 遅延評価
- 型推論/パターンマッチ/モナド

*概要

■ツール

- scala
  - インタプリタ/コマンドコンソール
- scalac
  - コンパイラ: Javaのクラスへ変換
- fsc
  - scalacの高速版: デーモン化して起動を速くする
- scaladoc
  - APIリファレンスの生成: javadocのScala版
- sbaz
  - パッケージ管理

*文法

■リテラル

●数値, 論理値, 文字, 文字列, シンボル

1234 , true/false , 'c' , "Hello, world!" , 'id

●XML

```
val title = "Hello, world!"
val html =
  <html>
    <head><title>{title}</title></head>
    <body>{title}</body>
  </html>
```

*文法

■オブジェクト指向 (Javaに近い)

- ●class
  - Javaのclassとほぼ同じ
- ●object
  - Singleton class
  - インスタンスは1つ
  - Javaからはstatic method/fieldに見える
  - mainメソッドはここに書く

*文法

■オブジェクト指向 (Javaに近い)

- ●trait
  - 抽象クラス
  - インスタンス化はできない
  - Javaのinterfaceに似ている
  - 実装をかける
  - classで多重継承できる
  - Mix-inで使用

*文法

■関数型

●クロージャ

```
val func = {(x: Int, y: Int) => x + y}
println(func(1, 2)) // 3
```

●カリー

```
def add(x: Int)(y: Int) = x + y
val add1234 = add(1234) _
val result = add1234(5678)
println(result) // 6912
```

文法

■その他

- ●型推論
- ●パターンマッチング
- ●値の束縛
- ●Structural typing
- ●Generics
- ●Annotations
- ●Implicit Parameters
- ●Implicit Conversions
- ●etc...etc...

*文法

Hello, world! (Java)

```
// Hello, world
package examples;

public class HelloWorld {
  public static void main(String[] args) {
    System.out.println("Hello, world!");
  }
}
```

*文法

- Hello, world! (Scala)

```
// Hello, world
package examples

object HelloWorld {
  def main(args: Array[String]) {
    println("Hello, world!")
  }
}
```

*文法

Foo class (Java)

```
public class Foo {
  private String name;
  public Foo(String name) {
    this.name = name;
  }
  public void name(String name) {
    this.name = name;
  }
  public String name() {
    return name;
  }
}
```

*文法

Foo class (Scala)

```
class Foo(var name: String)
```

*文法

- trait

```
trait Name {
  def name
  def length = name.length
}
class SmithName extends Name {
  def name = "Smith"
}
println((new SmithName).length) // 5
```

*文法

■型推論

●型を推論して型の定義を省略できる機能

●例

- Java

```
String str = "Hello, world!";
int i = 1234;
int[] values = new int[] {1, 2, 3};
```

- Scala

```
val str = "Hello, world!"
val i = 1234
val values = Array(1, 2, 3)
```

*文法

Implicit Conversions

- 型の拡張を行う
  - メソッドの追加など
- 継承は行う必要がない
- 型を変えることなく、型を拡張できる
- 例:

```
// str: java.lang.String
val str = "1234"
// toIntメソッドは拡張されている
val length = str.toInt // 1234
```

*文法

Implicit Conversions

```
class Person(name: String)

class RichPerson(person: Person) {
  def splitName =
    person.name.split("[¥s]+")
}
implicit def psn2rPsn(person: Person) =
  new RichPerson(person)

val person = new Person("Foo Boo Bar")
println(person.splitName)
// Array("Foo", "Boo", "Bar")
```

*文法

■パターンマッチング

- ●条件分岐の一種
- ●オブジェクトツリーがパターン(条件)かどうか調べる
- ●switch case 文に似ている(が非なるもの)
- ●型や値を条件に加えることができる

文法

■パターンマッチング (Java)

```
Object value = new Integer(1234);
String message;
if (value instanceof Integer) {
  message = "Int: " + value;
} else if (value instanceof String) {
  message = "String: " + value;
} else {
  message = "Not match";
}
```

*文法

■パターンマッチング (Scala)

```
val value: Any = 1234
val message = value match {
  case i: Int => "Int: " + i
  case s: String => "String: " + s
  case _ => "Not match"
}
// "Int: 1234"
```

*文法

■パターンマッチ

```
val regex =
  """¥s*([0-9]+)¥s*/¥s*([0-9]+)¥s*""".r

val (n1, n2) = " 123 / 456 " match {
  case regex(n1, n2) =>
    (n1.toInt, n2.toInt)
  case _ => (0, 0)
}
// (123, 456)
```

開発

■インストール

●Javaランタイム(JDK/JRE)のインストール

- http://java.sun.com/javase/ja/6/download.html
- **環境変数**JAVA_HOME**の設定**

開発

■インストール

●Scalaのインストール

- http://www.scala-lang.org/downloads
- **方法**
  - scala-2.7.1.final.tar.gz(or .zip)**で適当なディレクトリに展開**
    - » **環境変数**SCALA_HOME(**インストール場所**)**の設定**
    - » **環境変数**PATH**に**$SCALA_HOME/bin**を追加**
  - IzPack Java Installer**でインストール**
  - MacPorts(Mac OS X)**でインストール**
  - **その他パッケージ管理ツールでインストール**

*開発

■コマンドラインでの開発

●対話環境

```
bash-3.2$ scala
Welcome to Scala version 2.7.1.final…

scala> for (i <- 1 to 4) {
     |   println(i)
     | }
1
2
3
4
scala>
```

*開発

■コマンドラインでの開発

●コンパイル

- ソース作成 (HelloWorld.scala)

```
object HelloWorld {
  def main(args: String) {
    println("Hello, world!")
  }
}
```

- コンパイル

```
bash-3.2$ scalac HelloWorld.scala
```

- 実行

```
bash-3.2$ scala HelloWorld
Hello, world!
```

開発

■IDEでの開発

●Eclipseの場合

- Eclipse 3.4 Classicをインストール
- Eclipse Scala Plug-inをインストール
  - http://www.scala-lang.org/node/94
- デモ
  - コード作成
  - コンパイル・実行

開発

■IDEでの開発

●NetBeansの場合

- NetBeans 6.5 nightly buildをインストール
- NetBeans Scala Plug-inをインストール
  - http://wiki.netbeans.org/Scala
- デモ
  - コード作成
  - コンパイル・実行

*ライブラリ

■ライブラリ

- ●Scala標準ライブラリ
  - リテラルの補完
    - Array(1, 2, 3)
  - コレクション
  - Actor
  - Parser Combinator
  - XML
  - etc
- ●これ以外はJava APIなどを使う

ライブラリ

■ライブラリ

- ●Scalax
  - The Scala Community Library
- ●Scala/xml
  - XQery, XSLT …
- ●Jiva
  - Genetic Algorithms (GA) toolkit
- ●Scala-rel, scala.dbc
  - ORM/SQL

ライブラリ

■テスト

- ●ScalaCheck
  - QuickCheck(Haskell)のScala実装.
- ●Rehersal
  - SUnitの代わり (Sunitよりも便利)
- ●ScalaTest
  - OSSテストツール (Scala/Java).
- ●Specs
  - BDD(Behavior-Driven-Design)を実践するフレームワーク
  - RSpecに近い

*フレームワーク

lift Web Framework

- Java Servletコンテナで動作するフレーワーク
- Rails/Djangoなどのいい点を継承
- Scalaで書かれており、Scalaで開発する
- XMLリテラルによるView
- Actorによる軽量処理(Cometなど)
- 標準ORM
- メニューフレームワーク

*フレームワーク

Web Flavor

- Servletベースの Webフレームワーク
- すぐ書いてすぐ実行 (オートコンパイル)
- Scalaで書かれており、Scalaで開発する
- Servlet APIのラッパ
- XMLリテラルによる簡単なView
- Administrator menuでソースの編集が可能
  - 配置してしまえばWeb上で開発可能
- keisuken(日本の方)という人が作ってるらしい

Web Flavor

そんなことより聞いてくれよ

>>1よ

今日のオススメは

Web Flavor

JavaとかScalaの話なんてどうでもいいんだよ

Web Flavor

■フレームワークのポリシー

●強くて硬い枠組を排除する

- シンプルにする
- 学習コストをおさえる
- 開発者に手段を選択させる

●冗長な定義を書かせない

- DRY: Don't Repeat Youreself: **重複させない**
- CoC: Convention Over Configuration：**設定より規約**

●コードで記述する

- プログラマに主導権を

*Web Flavor

- Hello, world!

```
// HelloWorld.scala
val TITLE = "Hello, world!"
<html>
<head><title>{TITLE}</title></head>
<body><h1>{TITLE}</h1></body>
</html>
```

*Web Flavor

HelloWorldPOHP

```
<!-- HelloWorldPOHP.html -->
<html>
<head><title><span flavor:id="title"/></title></head>
<body>
<h1><span flavor:id="title"/></h1>
<p><span flavor:id="message"/></p>
</body>
</html>

// HelloWorldPOHP.scala
Template("",
  "title" -> Text("Hello, world!"),
  "message" -> <strong>Web Flavor world!</strong>
)
```

Web Flavor

PrintRequestParameters

```
// PrintRequestParameters.scala
val TITLE = "PrintRequestParameters"
<html>
<head><title>{TITLE}</title></head>
<body>
<h1>{TITLE}</h1>
<dl>{
  for ((name, value) <- params) yield
    <dt>{name}</dt><dd>{value}</dd>
}</dl>
</body>
</html>
```

Web Flavor

■今後

- ●ORM対応
  - 独自に作るかも
- ●ステートフルGUIコンポーネント
  - 現在プロトタイプは存在
- ●国際化
  - LocaleによるPOHPテンプレートなどの差換え

*Javaとの関係

■概要

- ●Java処理系で動く言語の1つ
  - Java Scripting Frameworkには対応していない
- ●親和性が高い
  - 類似の文法
  - 相互呼び出し
    - JavaからScala, ScalaからJava
  - 開発スタイル
    - コンパイル
    - パッケージ化(Jarファイル)
    - 実行

*Javaとの関係

- Javaクラスファイル/JARファイル
  - scalacでコンパイル
  - Javaのクラスファイルに
  - JAR/WAR/EARファイル化可能
- Java処理系での実行
  - scala-library.jar(Scalaランタイム)をCLASSPATHに
  - それ以外は通常のJavaプログラムと同じ

Javaとの関係

■Java API/ライブラリをScalaで使う

●ほとんど意識することなく使用可能

```
import javax.swing._
val frame = new JFrame("Foo")
...
frame.setVisible(true)
```

●JavaのGenericsはScalaと透過

```
import java.util._
val list = new ArrayList[String]
list.add("abc")
...
```

Javaとの関係

■Scalaのライブラリを Javaから使う

●気をつけることがある

- **プロパティメソッド**
  - Scala: def name_=(value: String) {…}
  - Java: void name_$eq(String value) {…}
- object
  - **クラス名の最後に$がつくときがある**
- **クロージャ**
  - Function{**引数の数**}[**型**, …]**クラスに**
- Generics
  - Scala 2.7.2**でないと**Java**から使えない**

*まとめ

■Javaへの貢献

- ●Java標準言語としてどう?
  - まだ難しい
  - 選択肢の1つになる (裾野が広がる)
  - Java言語の拡張のサンプルとなりえる
- ●Scalaによる開発
  - ハードルを下げる
  - 成果物はJavaそのもの

*まとめ

■Scalaの将来

- ●利点がいっぱい
  - 軽量な文法
    - スクリプト的なアプローチで作成
  - 関数型アプローチ
    - 副作用を取り除くことによるバグの減少
  - パフォーマンスも良好
    - ほとんどJavaと変わらない
- ●普及は?
  - Javaによく似てる/熱狂的なファンがいる
  - 今後は増えていく

デモ

- Eclipse Scala Plug-in
- **対話環境**
- **サンプルプログラム**
- WebFlavor

参考

- Scala **本家**
  - http://www.scala-lang.org/
- Scala Wiki
  - http://scala.sygneca.com/
- Scala Blog
  - http://scala-blogs.org/

参考

- Lingr - Scala-ja
  - http://www.lingr.com/room/scala-ja
- Scala-sandbox (CodeRepos)
  - http://coderepos.org/share/wiki/Scala-sandbox
- mixi - Scala**コミュニティ**
  - http://mixi.jp/view_community.pl?id=3111016
- lift Web Framework
  - http://liftweb.net/
- Web Flavor
  - http://webflavor.sourceforge.net/

質問など

ご静聴ありがとうございました

Any Questions?