

プログラミング言語: Scala の紹介

2024-01-20

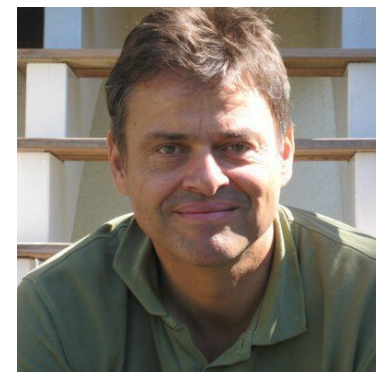
Affiliation: JAIST Ph.D. Student

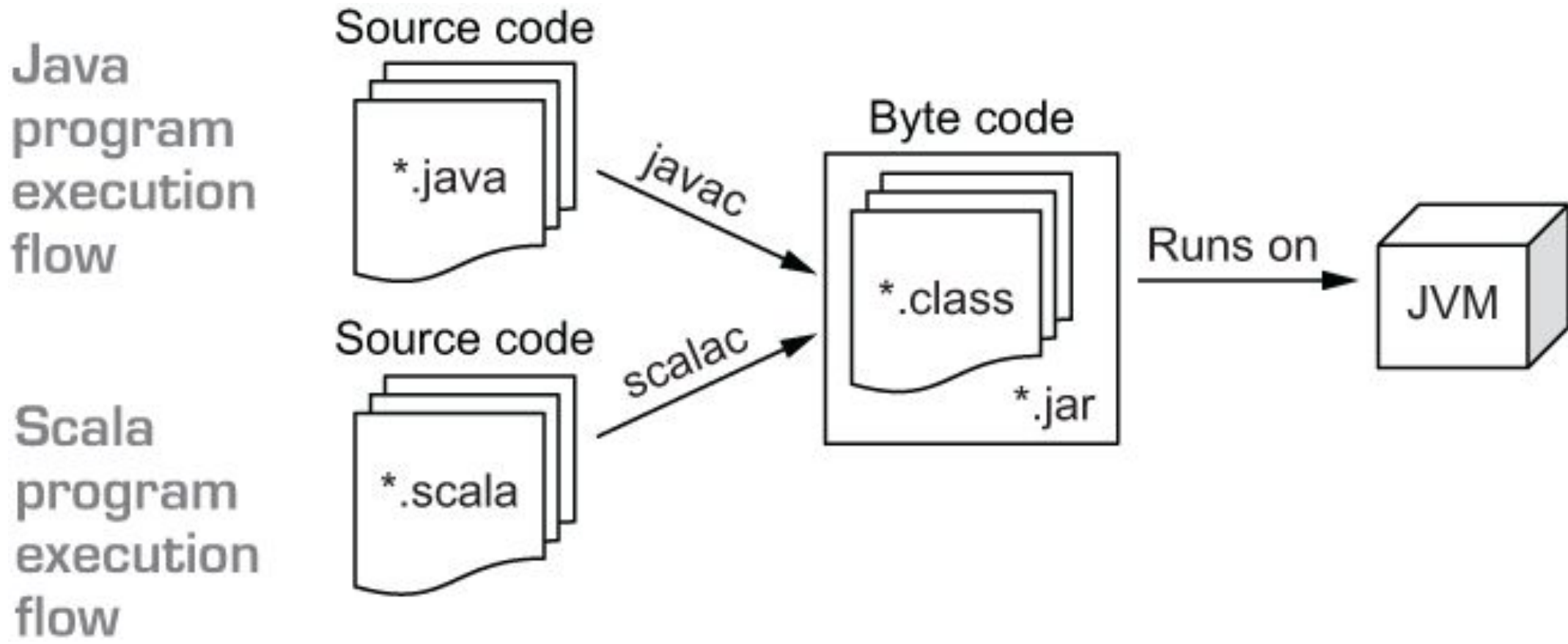
Name: ADACHI Yuya

E-mail: s2120001@jaist.ac.jp

- Scala 歴は 2.13.1 がリリースされたところなんで 5 年ぐらい
- 何か 1 言語以上、プログラミング言語を知っている前提で発表します
- 圏論、モナド、型システムなどの話は出てきません
 - この辺りをキチンと説明できる自信がない

- Scalable Language に由来する
- マルチパラダイムプログラミング言語
- 静的型付け言語、非純粋関数型言語
- JVM 言語、AltJS 言語、コンパイラ言語、.NET 言語 (開発中止)
- Martin Odersky 氏によって設計された
- Scala Center によって開発・保守がされている





1 Why Scala?: <https://livebook.manning.com/book/get-programming-with-scala/chapter-1/27>

コソスキポイント (1): 手軽にスクリプトを作って遊べる


```
1 import requests
2
3 def fetch_data(url):
4     response = requests.get(url)
5     if response.status_code == 200:
6         return response.json()
7     else:
8         return None
9
10 url = "https://jsonplaceholder.typicode.com/posts/1"
11 data = fetch_data(url)
12 if data is not None:
13     print(data)
14 else:
15     print("データの取得に失敗しました。")
```



```
1 import scala.io.Source
2 import scala.util.{Try, Using}
3
4 def fetchData(url: String): Try[String] = {
5     Using(Source.fromURL(url))(_.mkString)
6 }
7
8 val url = "https://jsonplaceholder.typicode.com/posts/1"
9 fetchData(url) match {
10     case scala.util.Success(data) => println(data)
11     case scala.util.Failure(ex) => println(s"データの取得に失敗しました: ${ex.getMessage}")
12 }
```



コソスキポイント (2): Java の資産を好きに使える



```
1  import java.time.LocalDateTime
2  import java.time.format.DateTimeFormatter
3
4  object JavaLibExample extends App {
5      val currentDateTime = LocalDateTime.now()
6      val formatter = DateTimeFormatter.ofPattern("yyyy/MM/dd HH:mm:ss")
7      val formattedDateTime = currentDateTime.format(formatter)
8
9      println(s"現在の日時: $formattedDateTime")
10 }
```

コソスキポイント (3):コード量が少なく記述できる

```
1 import java.util.Arrays;
2 import java.util.List;
3 import java.util.stream.Collectors;
4
5 public class Main {
6     public static void main(String[] args) {
7         List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
8         List<Integer> evenNumbers = numbers.stream()
9                                         .filter(n -> n % 2 == 0)
10                                        .collect(Collectors.toList());
11         System.out.println(evenNumbers);
12     }
13 }
14
```



```
val x = (1 to 10).filter(x => x % 2 == 0)
println(x) (): scala.Unit
```




ココスキポイント (4): 開発環境構築が簡単 & ツールが優秀

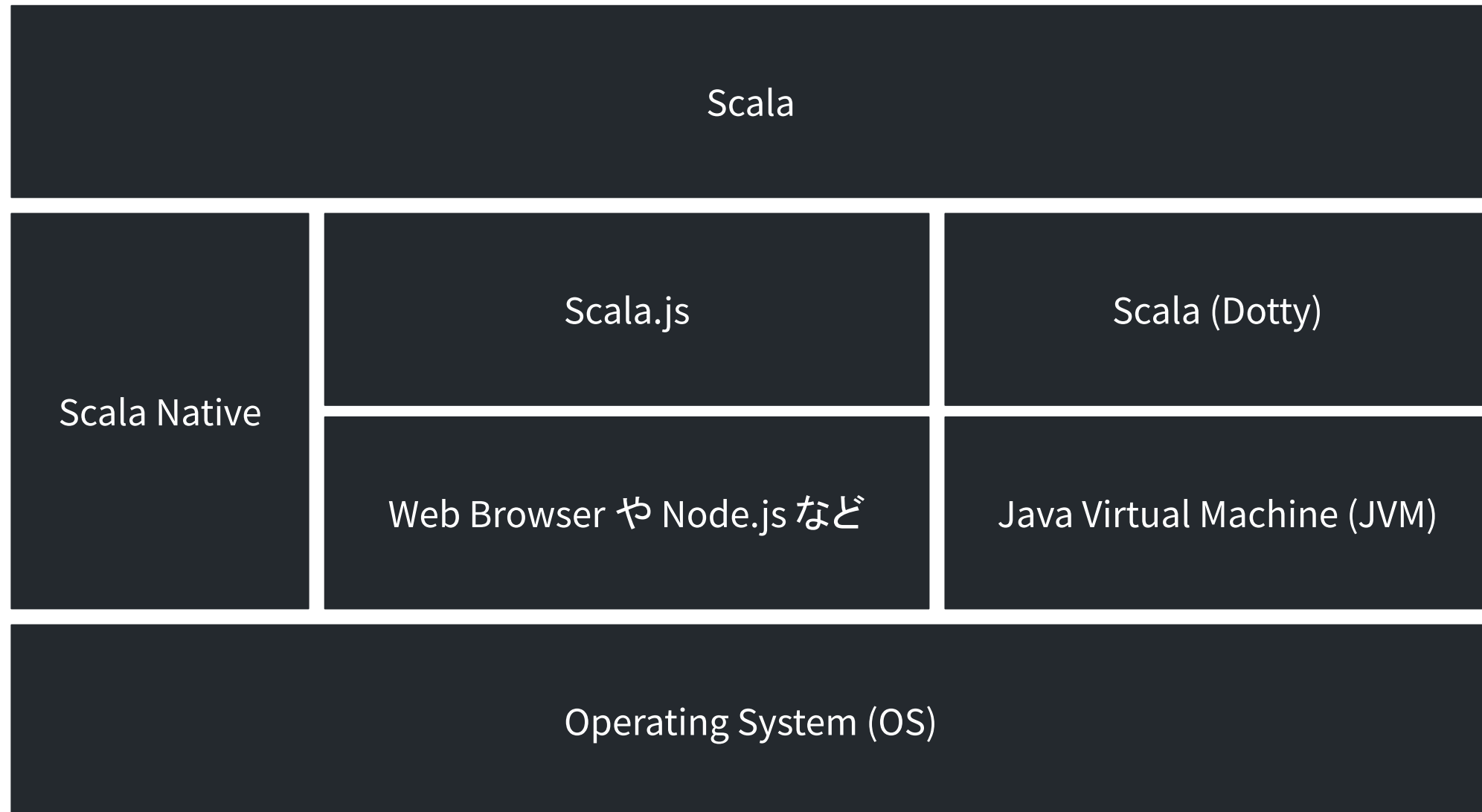
```
// Linux の場合は、コマンド一発でJDK のインストール、パスの設定、ビルドツールのインストールが完了する
$ curl -fL <https://github.com/coursier/coursier/releases/latest/download/cs-x86_64-pc-linux.gz> | gzip -d > cs && chmod +x cs && ./cs
setup -y
$ source ~/.profile
$ java --version
openjdk 17 2021-09-14
OpenJDK Runtime Environment Temurin-17+35 (build 17+35)
OpenJDK 64-Bit Server VM Temurin-17+35 (build 17+35, mixed mode, sharing)
$ scala-cli -S 2.13.5 Main.scala // Scala のバージョンを指定できる
$ scala-cli --jvm adopt:11 Main.scala // JVM のバージョンを指定できる
```


ココスキポイント (5): 並列処理が超手軽に実装できる

```
1  //> using dep "org.scala-lang.modules::scala-parallel-collections:1.0.4"
2
3  import scala.collection.parallel.CollectionConverters._
4
5  (1 to 10).par.map { i =>
6    |   println(i)
7    |   Thread.sleep(500)
8    | }
```



ココスキポイント (6): 色々なプラットフォームにスケラブルに拡張できる



- 型システムが優秀
- Option / Either / Try でヌルポを撲滅できる

ウ〜ンポイント: 波括弧とインデントの混在が許可されている

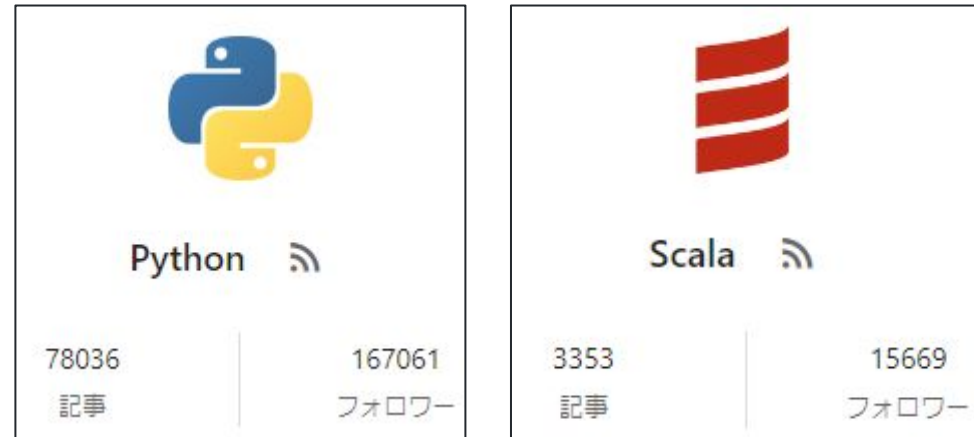
```
1  import scala.io.Source
2  import scala.util.{Try, Using}
3
4  def fetchData(url: String): Try[String] = {
5    | Using(Source.fromURL(url))(_._mkString)
6  }
7
8  val url = "https://jsonplaceholder.typicode.com/posts/1"
9  fetchData(url) match {
10   | case scala.util.Success(data) => println(data)
11   | case scala.util.Failure(ex) => println(s"データの取得に失敗しました: ${ex.getMessage}")
12 }
```



```
1  import scala.io.Source
2  import scala.util.{Try, Using}
3
4  def fetchData(url: String): Try[String] =
5    | Using(Source.fromURL(url))(_._mkString)
6
7  val url = "https://jsonplaceholder.typicode.com/posts/1"
8  fetchData(url) match
9    | case scala.util.Success(data) => println(data)
10   | case scala.util.Failure(ex)    => println(s"データの取得に失敗しました: ${ex.getMessage}")
```



イマイチポイント: ユーザーや認知度がまだまだ少ない



- Stack Overflow Developer Survey 2022: 26位
 - <https://survey.stackoverflow.co/2022>
- Stack Overflow Developer Survey 2023: 27位
 - <https://survey.stackoverflow.co/2023>

- バックエンドとデータサイエンスは得意
 - Twitter のバックエンドで採用されたことで話題になった
 - Apache Spark も Scala で記述されている
 - Python の Matplotlib みたいな手軽に使えるプロットライブラリがない
- Scala.js を活用すればフルスタックでシステムを開発できる
- 以下の分野は苦手かな
 - 組み込み系: C/C++、Rust が強い
 - ゲーム: C/C++ (UE) や C# (Unity) が強い
 - モバイルアプリ: Swift、Kotlin、Dart (Flutter) が強い

- 名前の由来通りスケラブルな言語
 - 手軽なスクリプトから大規模システムまで作れる
- 難しいとか怖いイメージがあるけれど難易度は高くない
 - 2つ目以降、何のプログラミング言語を勉強しようか迷っている人にオススメできる言語だと思う
- ユーザー数や認知度がまだまだ未熟なところ
- 分野ごとに得意・不得意がある