ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ

УДК 681.3

СРАВНИТЕЛЬНЫЙ АНАЛИЗ ДВУХ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ: JAVA И SCALA

© 2022 Т. В. Аветисян, А. М. Бородай

Воронежский институт высоких технологий (Воронеж Россия)

В статье обсуждаются основные различия, которые характерны для двух языков программирования: Java и Scala.

Ключевые слова: программирование, Scala, Java, приложения.

Java – один из самых известных и широко используемых языков программирования в мире. Его синтаксис достаточно прост в изучении, присутствует хорошая обратная совместимость. Однако, java не всегда эффективна в использовании для конкретных задач. В этой связи, даже в рамках одной платформы JVM были разработаны другие языки, имеющие свои преимущества и недостатки [1]. Кроме того, были созданы реализации для работы других языков программирования такую же платформу, например python. [1]

Прежде всего, отметим, что и Scala, а также и Java представляют собой объектно-ориентированные языки программирования на основе JVM. JVM — это виртуальная машина, которая дает возможность компьютеру запускать программы не только на языке Java, но и на других языках, которые компилируются в байт-код для JVM. Кроме того, Scala помимо объектно ориентированной парадигмы поддерживает функциональную [2].

Для того, чтобы сделать выбор на каком из этих двух языков программирования стоит писать программы, необходимо сравнить их по нескольким ключевым пунктам.

Первое, на что стоит обратить внимание – это область применения [3].

Java применяется в многочисленных областях, начиная с e-commerce веб-сайтов и завершая приложениями для Android. Круг применения scala несколько скромнее, здесь мы рассмотрим области, в которых эти два языка пересекаются [4, 5].

Java очень обширно применяется в финансовой сфере. Многие банки, например, Сбербанк, Альфа банк, Тинькофф и другие используют Java для написания фронт-энд и бэк-энд офисных электронных систем, систем регулирования и конфирмации, проектов обработки данных и некоторых других.

Преимущественным образом Java используется при написании серверных приложений, в большинстве своём без какого-либо пользовательского интерфейса. Они получают данные с одного сервера, обрабатывают их и отправляют дальше.

Scala используют известные банки, представленные выше, преимущественно scala используют для реализации потоковой обработки данных или анализа больших данных. Вот типовые задачи и примеры проекторов в этих сферах от компании Neoflex:

Online рекомендации и принятие решения (Next Best Action, Next Best Offer);

Аветисян Татьяна Владимировна – Воронежский институт высоких технологий, старший преподаватель, e-mail: avvetis_tatyanka@yandex.ru.
Бородай Александр Михайлович – Воронежский институт высоких технологий, студент, e-mail: RedCrabAlex@yandex.ru.

Выявление аномалий и предотвращение мошенничества (Anomaly Detection, Anti-Fraud);

Предиктивная аналитика;

AI и ML для голосовых помощников и чат-ботов;

Задачи Internet of Things (IoT).

Также Java широко используется в электронной коммерции и в области вэб-приложений. Огромное количество RESTful сервисов было создано с использованием Spring MVC, Struts 2.0 и похожих фреймворков.

Scala как Java имеет широкий набор библиотек и фреймворков для созданий веб приложений, например akka-http, play framework. Помимо бэкенда, scala так же может использоваться на фронтенде, это scalajs. ScalaJs – это реализация scala, которая компилируется в байт код, и поддерживает совместимость с javascript.

Второе, что мы можем сравнивать — это синтаксис языков. Возможности scala 2.x, которых нет в java: автоматический вывод типов, кейс классы, паттерн-матчинг, значение параметров по умолчанию и именные параметры, неявные преобразования, ленивые вычисления, не требуется отлавливать исключения, упрощенный синтаксис для больших строк (текстовые блоки), множественное наследование, функции высшего порядка, тело класса является его конструктором [6, 7].

Стоит заметить, что язык java и scala также развиваются, и на момент выхода данной статьи в java 17 уже поддерживается: упрощенный синтаксис для больших строк (текстовые блоки), паттерн-матчинг, аналог кейс классов.

Лучше всего продемонстрировать различия между языками можно на практике, поэтому ниже мы привели коды для Java и Scala для двух почти эквивалентных программ.

Пример вычисления факториала-Java:

```
class MainJava {
   public static BigInteger getFactorial(int f) {
      if (f <= 1) {
         return BigInteger.valueOf(1);
      }
      else {
         return BigInteger.valueOf(f).multiply(getFactorial(f - 1));
      }
   }
   public static void main(String[] args) {
      System.out.println("Hello world, Java!");
      List<Integer> number = List.of(2, 3, 4, 5, 100);
      for (var i : number) {
         System.out.printf("f(%d) = %d\n", i, getFactorial(i));
      }
   }
}
```

Пример вычисления факториала-Scala:

```
object MainScala {
 def getFactorial(f: Int): BigInt = if (f <= 1) 1 else f * getFactorial(f - 1)

 def main(args: Array[String]): Unit = {
  println("Hello world, Scala!")
  val number: List[Integer] = List(2, 3, 4, 5, 100)
  val result = for (i <- number) yield {
   s"f($i) = ${getFactorial(i)}"
  }
  println(result.mkString("\n"))
 }
}
```

Первое, что бросается в глаза, это синтаксис объявления функций и переменных. Объявления типа данных несмотря на то, что scala так же, как и java, являться статических типизированным в языке программирования можно опустить. В некоторых случаях, из-за этой особенности тип объявляется справа от переменной, чтобы парсер мог продолжить свою работу.

Видно отсутствие ключевого слово “return”, и то, что такие конструкции как “if/else” и “for” возвращают значения.

Полностью ООП подход, когда происходит работа с данными, ориентируется на то, что все есть объект. Из-за этой особенности в Scala возможна перегрузка операторов.

Это так же помогает сделать язык более понятным и выразительным. Пример можно увидеть при работе Java с большими числами.

Приходится вызвать статический метод, для конвертации целого числа объектов, после чего для этого объекта вызвается метод с соответствующим названием multiply, в то время как в Scala достаточно написать знак *.

Работа с выводом в консоль достаточно лаконична, присутствует комфортный синтаксис для работы со строками.

После разбора синтаксиса можно рассмотреть совместимость этих двух языков.

Scala не требует отказа от платформы Java. Этот язык позволяет повышать ценность существующего кода, то есть опираться на то, что у вас уже есть, поскольку он был разработан для достижения беспрепятственной совместимости с Java.

Программы на Scala компилируются в байт-коды виртуальной машины Java (JVM). Производительность при выполнении этих кодов находится на одном уровне с производительностью программ на Java. Код Scala может вызывать методы Java, обращаться к полям Java классах, поддерживать наследование из классов Java и реализовывать интерфейсы Java.

Для всего перечисленного не требуется ни специального синтаксиса, ни явных описаний интерфейса, ни какого-либо связующего кода. По сути, весь код Scala интенсивноым образом использует библиотеки Java, зачастую даже без ведома программистов.

Какой язык выбрать для написания программы? Это, конечно, выбор каждого. Популярность вокруг языка Scala стала меньше, но качество языка постоянно растет, и разработчики его выбирают, так как это надежный инструмент, который они могут использовать для решения своих технических или бизнес-задач, не связанных со Scala.

Язык Java-будет точно существовать долгое время, так как его используют во многих сферах, и уже существует большая кодовая база, которую нужно поддерживать.

СПИСОК ИСТОЧНИКОВ

1. JavaScript Higher Order Functions and Arrays // oodlesTechnologies URL: https://www.oodlestechnologies.com/blogs/javascript-higher-order-functions-and-arrays

2. Fast Data // Neoflex URL: https://www.neoflex.ru/expertises/fast-data

3. Прокопец А. Конкурентное программирование на Scala / А. Прокопец. – Саратов: Профобразование, 2019. – 342 с. // Электронный ресурс цифровой образовательной среды СПО PROFобразование: [сайт]. – URL: https://profspo.ru/books/87977.

4. Синюков Д. С. Экспериментальное исследование системы автоматического поиска и устранения неисправностей в базе данных / Д. С. Синюков, А. В. Потудинский // Моделирование, оптимизация и информационные технологии. – 2022. – Т. 10. – № 1 (36). – С. 20-21.

5. Потудинский А.В. Модели для определения моментов контроля в многоуровневых организационных системах / А. В. Потудинский, А. П. Преображенский // Моделирование, оптимизация и информационные технологии. – 2020. – Т. 8. – № 2 (29).

6. Вульфин А.М. Интеллектуальный анализ данных пользовательского окружения в задаче обнаружения удаленного управления / А. М. Вульфин // Моделирование, оптимизация и информационные технологии. – 2020. – Т. 8. – № 2 (29).

7. Вульфин А. М. Анализ защищенности веб-приложения для доступа к системе хранения критически важных данных / А. М. Вульфин // Моделирование, оптимизация и информационные технологии. – 2021. – Т. 9. – № 4 (35).

COMPARATIVE ANALYSIS OF TWO LANGUAGES PROGRAMMING: JAVA AND SCALA

© 2022 T. V. Avetisyan, A. M. Boroday

Voronezh Institute of High Technologies

The paper discusses the main differences that are characteristic of two programming languages: Java and Scala.

Keywords: programming, Scala, Java, applications.