

Scala

Scala je programski jezik (vrlo) visokog nivoa.

Ime je dobila iz opisnog izraza "SCALable LAnguage". Ubrzo ćemo vidjeti i zašto! :D

Objedinjuje koncepte iz objektno orijentisanog programiranja (OOP) i funkcionalnog programiranja (FP).

Pošto na raspolaganju imamo cijelu lepezu mogućnosti, treba znati pametno odabrati pravi skup "alata" za dati problem koji rješavamo.

Primarna platforma na kojoj se izvršava je Java Virtuelna Mašina (JVM), pa se mogu koristiti svi library iz Jave.

Scala ima statičke tipove, kao npr. C# ili Java, ali ima podršku čak i za dinamičke (Javascript, Ruby) i strukturalne tipove (kao Typescript)! Ovako je npr. podržana interoperabilnost sa Javascriptom, pogledajte ScalaJS projekat.

Ima vrlo fleksibilnu i lijepu sintaksu, pa je pogodna i za pisanje DSL-ova (Domain-Specific Language).

Postoje brojni library i frameworkci za konkurentno i distribuirano programiranje.

Neki od tih su `Future` (u standardnoj library), Akka `Actor s`, Apache Spark, Kafka i dr.

Uvod

Scala je prije svega potpuno objektno orijentisan jezik.

To znači da su sve vrijednosti u programu objekti, čak i brojevi i karakteri!

Ovo nije slučaj u Javi, gdje imamo 7 "primitivnih tipova" kao što su `int`, `char`, `boolean`...

U Scali su oni predstavljeni klasama `Int`, `Char`, `Boolean`...

U C# je sličan slučaj kao i u Scali, gdje je `int` samo alijas za `Int32`, ali je i dalje klasa.

Ovaj koncept se naziva i "unified types".

Da pogledamo deklaraciju konstante:

```
1 | val x: Int = 5
```

Scala Copy

Tip se piše nakon naziva varijable, kao u Pascalu.

U većini slučajeva kompajler nam može pomoći s "zaključivanjem tipa" (type inference).

Kompajler će "skontat" za nas da je to broj... Gdje ste bili do sada??? :/

Tako da je prethodni primjer ekvivalentan sljedećem:

```
1 | val x = 5
```

Scala Copy

Primijetite i to da tačkazarez nije potrebna! Napokon!!!

Izgleda da su pisci kompajlera vrlo lijeni ljudi... :D S razlogom, naravno.

Ključna riječ `val` (skraćeno od `value`) označava konstantu, isto kao `final` u Javi:

```
1 | final int x = 5;
```

Java Copy

Naravno, sljedeći kod se neće iskompajlirati.

Dobićemo grešku "reassignment to val".

```
1 | val x = 5
2 | x = 6
```

Scala Copy

Da vidimo sada dobru staru varijablu, koja se deklariše s `var`, logično:

```
1 | var y = 5
2 | y = 6
3 | y += 3
```

Scala Copy

Varijabla `y` na kraju ima vrijednost `9`.

U Scali imamo još jednu vrstu varijable, tzv. "lijena konstanta" (lazy value):

```
1 | lazy val z = 5
```

Scala Copy

Vrijednost `lazy val` će se izračunati samo jednom, i to tek kada se pozove u programu!

Npr. ako `z` poziva neku funkciju koja sadrži `println`, ispisaće se na ekran tek kada pozovemo `z` (tj. evaluiraće se funkcija).

Ovaj koncept je iz FP, i vrlo je koristan za keširanje vrijednosti.