Functional Programming using Scala

Faaiz Shah

faaiz@bfore.ai
Lead Data Science & Engineering

November 19, 2024

BforeAI

Currying and Partial Application in Scala

Definition:

- Currying is the process of transforming a function with multiple arguments into a sequence of functions, each with a single argument.

Example Code:

```scala
def add(x: Int)(y: Int): Int = x + y

val addFive = add(5)_
println(addFive(10)) // Output: 15

```

Explanation:

- add is a curried function.
- `addFive` partially applies add with `x = 5`.

Lazy vs. Eager Evaluation in Scala

Lazy Evaluation:

- Computations are deferred until their results are needed.
- Use `lazy val` to declare lazy variables.

Eager Evaluation:

- Computations are performed immediately when they are bound to variables.
- Standard in Scala for `val` and `var`.

Example:

```scala
lazy val x = { println("Computed x"); 10 }
println("Before accessing x")
println(x) // Triggers computation

```

Transformations in Apache Spark

Definition:

- Transformations are operations on RDDs/DataFrames that return a new RDD/DataFrame.
- They are **lazy** and not executed until an action is called.

Common Transformations:

- `map`, `filter`, `flatMap`, `groupByKey`

Example:

```scala
val data = spark.read.textFile("data.txt")
val filteredData = data.filter(line => line.contains("Spark"))

```

Actions in Apache Spark

Definition:

- Actions trigger the execution of transformations to produce a result.
- They are **eagerly** executed.

Common Actions:

- `collect`, `count`, `first`, `take`

Example:

```scala
val numLines = filteredData.count()
println(s"Number of lines containing 'Spark': $numLines")

```

Conclusion & Remarks

- **Flexibility**:
  - Functional Programming
- **Scalability**:
  - Scala
- **Big Data processing & ML**:
  - Apache Spark

Thank you for listening !

Faaiz Shah

faaiz@bfore.ai