Introduction to spark with scala

Just a simple and quick but interesting introduction -

Be Happy With Scala!!!

Adekunle Babatunde - Seamfix Nig Ltd.

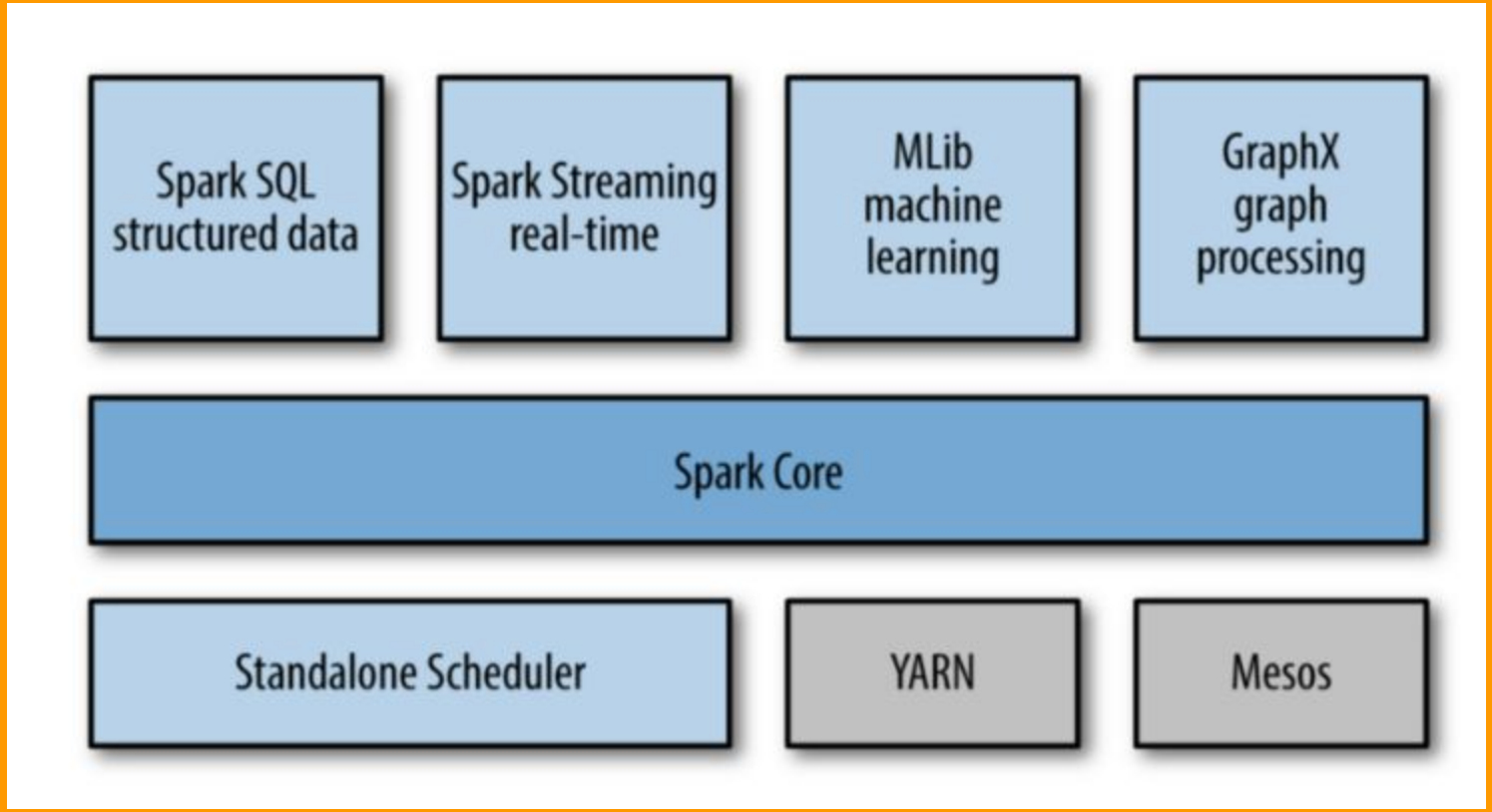
What About Spark

- The underlying Idea
- It's Capability
- Introducing RDD
- Transformation and Actions
- Brief Intro to Dataset/DataFrame API
- A simple tweet analysis

The Underlying Idea

- A fast and general-purpose cluster computing system.
- Unified engine for big data applications
- Why??? A cluster computing platform???
  - Single processor maxed out
  - Hadoop
- Why??? A new big data applications
  - Schedule, Good distribution systame
  - A good monitoring
  - Speed

The Capability



What Spark Components Mean

Spark SQL Structured Data:

- ★ Optimized for SQL like processing
- ★ SQL (SQL/HQL) and Dataset API

Spark Streaming

- ★ Ingest and Process data in Real-time
- ★ Abstraction with DStream
- ★ Initiated by StreamingContext()

Spark ML/MLLIB

- ★ Machine Learning API for Spark
- ★ ML is DataSet/DataFrame based
- ★ MLLib is RDD based

Spark GRAPHX Graph Processing

- ★ Graphs and graph parallel processing API
- ★ Has some cool other stuffs.

Initiating the different APIs.

```scala
val spark: SparkSession = SparkSession.builder()
  .master( master = "local[*]")
  .appName( name = "simple-count-app")
  .getOrCreate()
```

Initiate SparkSession for SQL, DataFrame/Dataset API

```scala
spark.read.textFile( path = "textfile.csv")
```

```scala
val sc: SparkContext = spark.sparkContext
```

Initiate sparkContext for RDD operations

```scala
val rdd = sc.textFile( path = "textfile.csv")
```

Spark RDD

- ➔ Create and RDD

```scala
val data: Array[Int] = Array(1, 2, 3, 4, 5, 6, 6, 7, 7)
val dataRdd: RDD[Int] = sc.parallelize(data)
```

- ➔ Operations on RDD
  - ◆ Transformations
  - ◆ Actions
- ➔ Transformations

```scala
val lcStopWords = Set("the").map(_.trim.toLowerCase)
val tokens: RDD[String] = rdd.flatMap(_.split( regex = " ").map(_.trim.toLowerCase))
val words: RDD[String] = tokens.filter(token => !lcStopWords.contains(token) && (token.length > 0))
val wordPairs: RDD[(String, Int)] = words.map((_, 1))
val wordCounts: RDD[(String, Int)] = wordPairs.reduceByKey(_ + _)
```

Transformations and Actions

➔ Transformations

```
val lcStopWords = Set("the").map(_.trim.toLowerCase)
val tokens: RDD[String] = rdd.flatMap(_.split( regex = " ").map(_.trim.toLowerCase))
val words: RDD[String] = tokens.filter(token => !lcStopWords.contains(token) && (token.length > 0))
val wordPairs: RDD[(String, Int)] = words.map((_, 1))
val wordCounts: RDD[(String, Int)] = wordPairs.reduceByKey(_ + _)
```

➔ Others

- join()
- sortByKey()
- coalesce()
- etc

Transformations and Actions II

➔ Actions

```
val collects: Array[(String, Int)] = wordcounts.collect()
val counts: Long = wordcounts.count()
val save: Unit = wordcounts.saveAsTextFile(outputFile)
```

➔ Others

- take()
- takeOrdered()
- saveAsSequence()
- etc.

Dataset/DataFrame

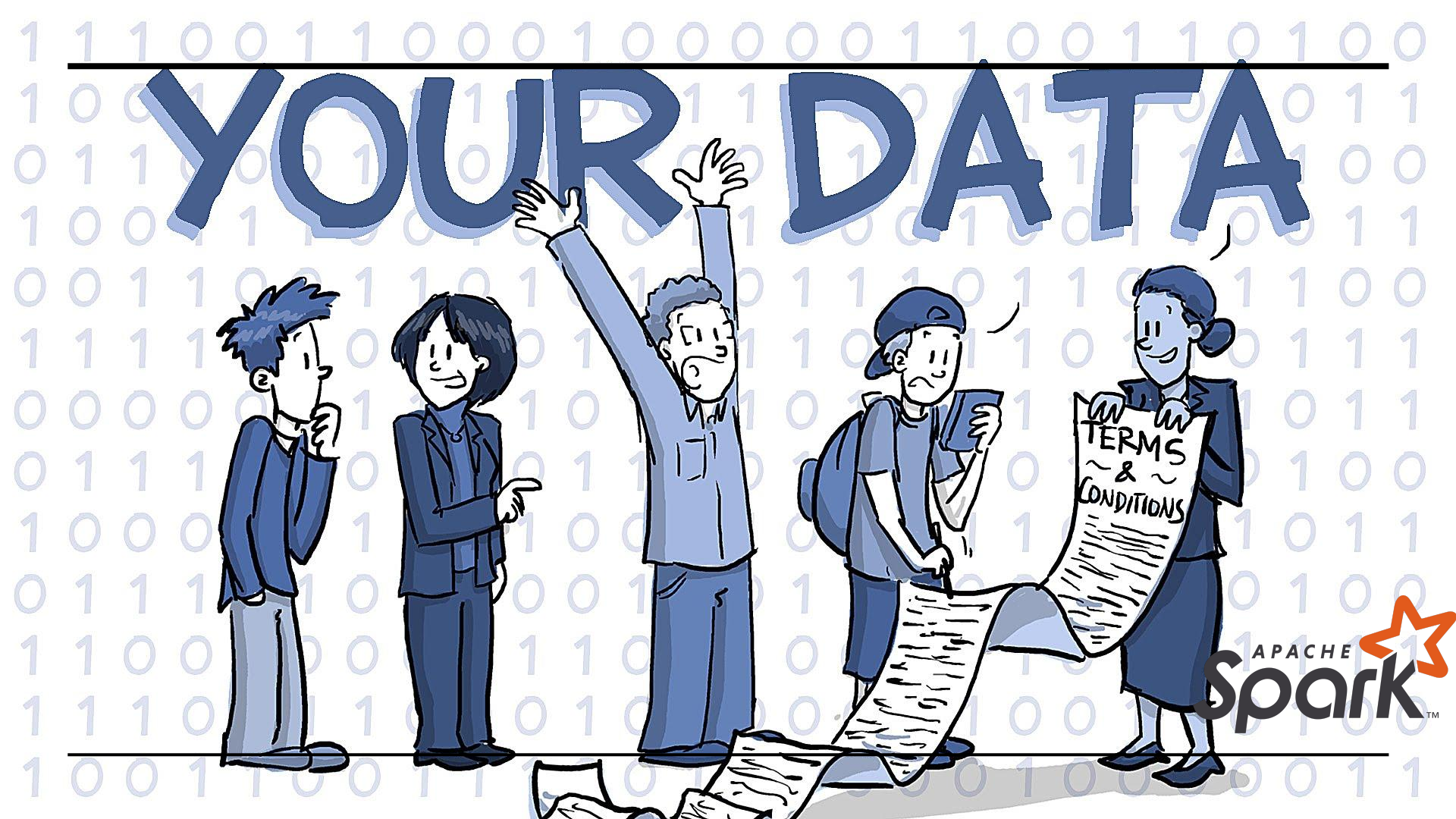
A Quick Look

➔ **Dataset**

```
val dataSet: Dataset[String] = spark.read.textFile( path = "textfile.csv")
val filteredDataSet: Dataset[String] = dataSet.filter(x => x.contains("a"))
val splitDataSet: Dataset[String] = filteredDataSet.flatMap(_.split( regex = " "))
```

➔ **DataFrame**

```
val df: DataFrame = dataSet.toDF() //Converts to DataFrame
val colList: Array[String] = df.columns
val selectData: DataFrame = df.select( col = "col1",  cols = "col2")
                              .withColumnRenamed( existingName = "someCol1",  newName = "someCol2")
```


YOUR DATA
TERMS & CONDITIONS
APACHE Spark

Thank You !