07: Getting started with Spark-submit

Posted on November 9, 2018

This extends the tutorial Setting up & getting started with Spark local mode with Sbt & Scala and Setting up Spark standalone on Mac.

Spark-submit is used to submit a Spark job. In this tutorial we will be using the "local" master in "client" mode. When you have a Hadoop installation with multiple nodes managed by the "YARN" resource manager, you can run it on "yarn" in "cluster" mode.

Step 1: Add the Spark dependency to "**build.sbt**" file.

```
organization := "com.sbt-tut"
```

300+ Java Interview FAQs

300+ Java FAQs
16+ Java Key Areas Q&As
150+ Java Architect FAQs
80+ Java Code Quality Q&As
150+ Java Coding Q&As

300+ Big Data Interview FAQs

300+ Big Data FAQs
Tutorials - Big Data
TUT - Starting Big Data
TUT - Starting Spark & Scala

```

version:= "0.1"

scalaVersion := "2.11.12"

libraryDependencies += "org.apache.spark" %% "spar

```

Step 2: From the command-line run the following command, and it may take some time to download the dependencies.

```
~/projects/sbt-tutorial]$ sbt eclipse

```

Step 3: Write a very simple Spark code in Scala as shown below within a package named "com.scalaproject" and the Scala file is named "**SimpleSpark.scala**".

```
package com.scalaproject

import org.apache.spark.sql.SparkSession;
import org.apache.spark.sql.Dataset;

object SimpleSpark {

  def main(args: Array[String]): Unit = {
     val spark = SparkSession.builder.appName("Si
                              .config("spark.mast
                             .getOrCreate()

     val list = List(1,2,3,4,5)
     val rdd = spark.sparkContext.parallelize(lis

     val multipleBy2Rdd = rdd.map(_ * 2)
     println(multipleBy2Rdd.collect().toList)

  }
}

```

Step 4: You can package it as a jar file in the command-line.

```
~/projects/sbt-tutorial]$ sbt package

```

projects/sbt-tutorial/target/scala-2.11/**sbt-tutorial_2.11-0.1.jar** will be built.

Step 5: On a command-line run the spark-submit command.

```
~/projects/sbt-tutorial]$ spark-submit --class com

```

Output:

```
....2018-11-09 23:01:35 INFO  TaskSetManager:54 -
2018-11-09 23:01:35 INFO  TaskSchedulerImpl:54 - I
2018-11-09 23:01:35 INFO  DAGScheduler:54 - Resul
2018-11-09 23:01:35 INFO  DAGScheduler:54 - Job 0
List(2, 4, 6, 8, 10)
2018-11-09 23:01:35 INFO  SparkContext:54 - Invok
2018-11-09 23:01:35 INFO  AbstractConnector:318 -
2018-11-09 23:01:35 INFO  SparkUI:54 - Stopped Sp
2018-11-09 23:01:35 INFO  MapOutputTrackerMasterE
.......

```

The .bash_profile

This assumes that Hadoop has been installed as per one of the other tutorials so that you can do $(hadoop classpath). Also, when you install it with “brew install” as opposed downloading the *.tar.gz and unzipping it.

```
//.....
export SPARK_HOME=/usr/local/Cellar/apache-spark/2
export SPARK_DIST_CLASSPATH=$(hadoop classpath)
export PATH=$SPARK_HOME/bin:${PATH}
//...
```

Activate the change:

```
$ source ~/.bash_profile
```

Starting the Spark master

```
sudo spark-class org.apache.spark.deploy.master.Ma
```

Note down the **host:port** where the Spark master is listening via “http://localhost:8080”.

Spark worker joins the Spark master

In a separate shell window start the worker node, specify the master host:location to join.

```
sudo spark-class org.apache.spark.deploy.worker.Wo
```

You can submit a job as shown below. If you don’t specify the URL of the master, it defaults to “local[*]”.

```

$ spark-submit --master spark://192.168.0.18:7077 
```

‹ 06: Setting up Spark-shell on Mac & getting started

08. Setting up & getting started with Hadoop on Mac ›