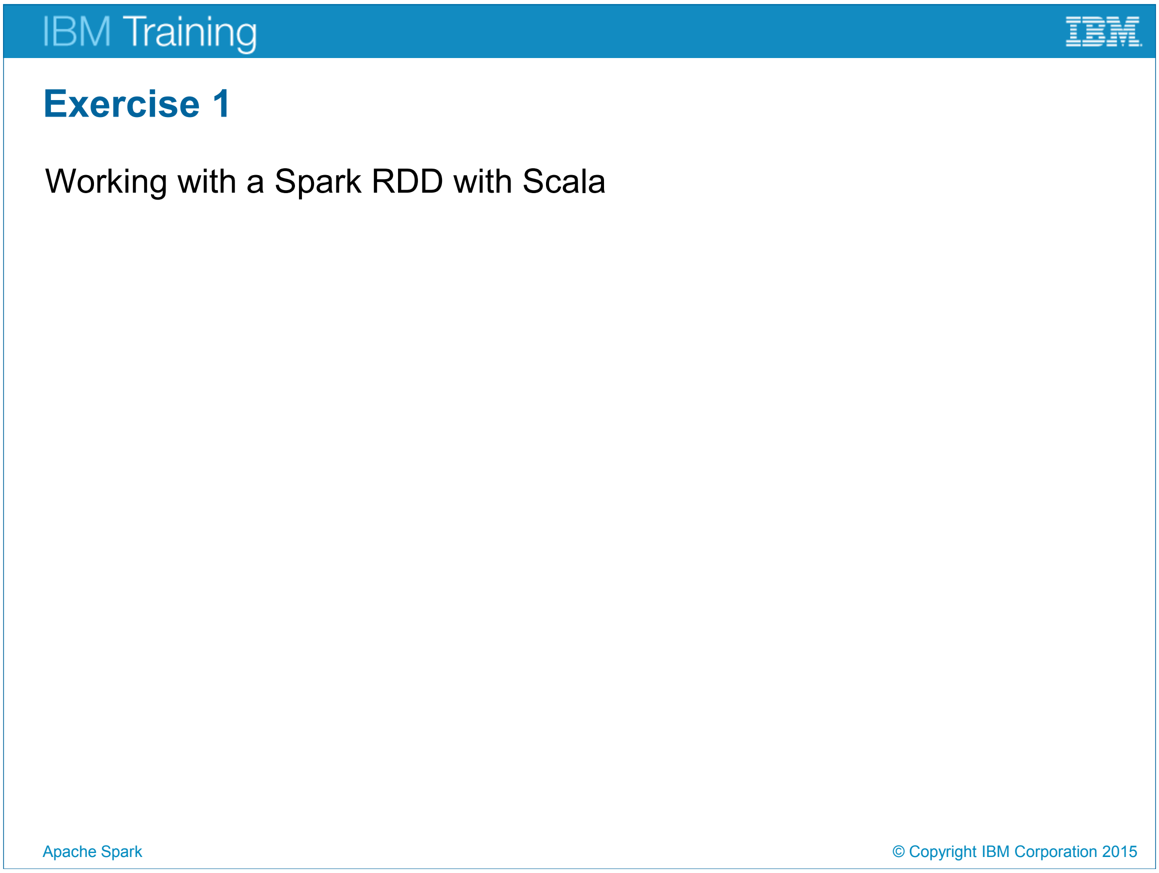

IBM Training

Exercise 1

Working with a Spark RDD with Scala

Apache Spark

© Copyright IBM Corporation 2015

Exercise 1: Working with a Spark RDD with Scala

Exercise 1:
Working with a Spark RDD with Scala

Purpose:
In this Exercise you will learn to use some of the fundamental aspects of running Spark in the Open Data Platform environment.

Additional information on Spark and additional lab exercises using Spark (with Scala, Python, and Java) are available in the BigDataUniversity.com (BDU) course *Spark Fundamentals* available at http://bigdatauniversity.com/bdu-wp/bdu-course/spark-fundamentals.

Good locations for additional information on Scala and introductory exercises in Spark with Scala and Python can be found at:

Scala Community: http://www.scala-lang.org

Quick Start: https://spark.apache.org/docs/latest/quick-start.html

Blogs, e.g.: http://blog.ajduke.in/2013/05/31/various-ways-to-run-scala-code

Note:

You may need verify that your hostname and IP address are setup correctly as noted in earlier units. Note that if you shut down your lab environment, this verification of hostname and IP address should be repeated. This is particularly important if you find that you cannot connect to Spark. Resetting the IP values may require that you reboot the VMware Image (as root: `reboot`).

Task 1. Connect to the VMware Image & to the Spark server.

1. Connect to and login to your lab environment with user **biadmin** and password **biadmin** credentials.
2. In a new terminal window, type `cd` to change to your home directory.
3. To set an environmental variable $SPARK_HOME, type `export SPARK_HOME=/usr/iop/current/spark-client.`

4. To start the Spark shell, type **`$SPARK_HOME/bin/spark-shell`**.

```
[biadmin@ibmclass Desktop]$ cd
[biadmin@ibmclass ~]$ $SPARK_HOME/bin/spark-shell
15/06/05 11:06:04 INFO spark.SecurityManager: Changing view acls to: biadmin
15/06/05 11:06:04 INFO spark.SecurityManager: Changing modify acls to: biadmin
15/06/05 11:06:04 INFO spark.SecurityManager: SecurityManager: authentication
disabled; ui acls disabled; users with view permissions: Set(biadmin); users with
modify permissions: Set(biadmin)
15/06/05 11:06:04 INFO spark.HttpServer: Starting HTTP Server
15/06/05 11:06:04 INFO server.Server: jetty-8.y.z-SNAPSHOT
15/06/05 11:06:04 INFO server.AbstractConnector: Started
SocketConnector@0.0.0.0:34520
15/06/05 11:06:04 INFO util.Utils: Successfully started service 'HTTP class server'
on port 34520.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.2.1
      /_/

Using Scala version 2.10.4 (OpenJDK 64-Bit Server VM, Java 1.7.0_65)
Type in expressions to have them evaluated.
Type :help for more information.
15/06/05 11:06:07 INFO spark.SecurityManager: Changing view acls to: biadmin
15/06/05 11:06:07 INFO spark.SecurityManager: Changing modify acls to: biadmin

. . .

15/06/05 11:06:10 INFO scheduler.EventLoggingListener: Logging events to
hdfs://ibmclass.localdomain:8020/iop/apps/4.0.0.0/spark/logs/history-server/local-
1433520369438
15/06/05 11:06:10 INFO repl.SparkILoop: Created spark context..
Spark context available as sc.

scala>
```

You now have a Scala prompt where you can enter Scala interactively.

Note that the Spark context is available as sc.

To exit from Scala at any time, you type **`sys.exit`** and press **Enter** (the official approach) or use Ctrl-D.

The tab key provides code completion.

5. Type `sc.` (the period is needed!), and then press **Tab**.

```
scala> sc.
accumulable                accumulableCollection
accumulator                addFile
addJar                     addSparkListener
appName                    applicationId
asInstanceOf               binaryFiles
binaryRecords              broadcast
cancelAllJobs              cancelJobGroup
clearCallSite              clearFiles
clearJars                  clearJobGroup
defaultMinPartitions       defaultMinSplits
defaultParallelism         emptyRDD
files                      getAllPools
getCheckpointDir           getConf
getExecutorMemoryStatus    getExecutorStorageStatus
getLocalProperty           getPersistentRDDs
getPoolForName             getRDDStorageInfo
getSchedulingMode          hadoopConfiguration
hadoopFile                 hadoopRDD
initLocalProperties        isInstanceOf
isLocal                    jars
killExecutor               killExecutors
makeRDD                    master
metricsSystem              newAPIHadoopFile
newAPIHadoopRDD            objectFile
parallelize                requestExecutors
runApproximateJob          runJob
sequenceFile               setCallSite
setCheckpointDir           setJobDescription
setJobGroup                setLocalProperty
sparkUser                  startTime
statusTracker              stop
submitJob                  tachyonFolderName
textFile                   toString
union                      version
wholeTextFiles
```

Note, if your type `sc` and press **Tab**, without the period after *sc*, you will get an abbreviated output, since only three keywords start with *sc*, whereas a lot of functionality is provided by the Spark context ("sc.").

```
scala> sc
sc       scala    schema
```

Task 2. Load data into an RDD and perform transformations and actions on that data.

1. To do an RDD transformation by reading in a file that was previously loaded to HDFS, type the following:
   `val pp = sc.textFile("Gutenberg/Pride_and_Prejudice.txt")`

```
scala> val pp = sc.textFile("Gutenberg/Pride_and_Prejudice.txt")
15/06/05 12:05:48 INFO storage.MemoryStore: ensureFreeSpace(274073) called with
curMem=0, maxMem=278302556
15/06/05 12:05:48 INFO storage.MemoryStore: Block broadcast_0 stored as values in
memory (estimated size 267.6 KB, free 265.1 MB)
15/06/05 12:05:49 INFO storage.MemoryStore: ensureFreeSpace(41821) called with
curMem=274073, maxMem=278302556
15/06/05 12:05:49 INFO storage.MemoryStore: Block broadcast_0_piece0 stored as bytes
in memory (estimated size 40.8 KB, free 265.1 MB)
15/06/05 12:05:49 INFO storage.BlockManagerInfo: Added broadcast_0_piece0 in memory
on localhost:46250 (size: 40.8 KB, free: 265.4 MB)
15/06/05 12:05:49 INFO storage.BlockManagerMaster: Updated info of block
broadcast_0_piece0
15/06/05 12:05:49 INFO spark.SparkContext: Created broadcast 0 from textFile at
<console>:12
pp: org.apache.spark.rdd.RDD[String] = Gutenberg/Pride_and_Prejudice.txt MappedRDD[1]
at textFile at <console>:12
```

The result is a pointer to the file. The file is not actually read at this time, as is evidenced by noting that you do not get any errors if you misspell the file name. Now **pp** is a pointer to the RDD.

We can perform some RDD actions on this data. One simple action is to count the number of items (lines, records) in the RDD.

2. To count the number of items in the RDD, type **`pp.count()`**.

```
scala> pp.count()
15/06/05 12:02:27 INFO mapred.FileInputFormat: Total input paths to process : 1
15/06/05 12:02:27 INFO spark.SparkContext: Starting job: count at <console>:15
15/06/05 12:02:27 INFO scheduler.DAGScheduler: Got job 0 (count at <console>:15) with
2 output partitions (allowLocal=false)
15/06/05 12:02:27 INFO scheduler.DAGScheduler: Final stage: Stage 0(count at
<console>:15)
15/06/05 12:02:27 INFO scheduler.DAGScheduler: Parents of final stage: List()
15/06/05 12:02:27 INFO scheduler.DAGScheduler: Missing parents: List()
15/06/05 12:02:27 INFO scheduler.DAGScheduler: Submitting Stage 0
(Gutenberg/Pride_and_Prejudice.txt MappedRDD[7] at textFile at <console>:12), which
has no missing parents
15/06/05 12:02:27 INFO storage.MemoryStore: ensureFreeSpace(2536) called with
curMem=1263720, maxMem=278302556

...

15/06/05 12:02:28 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 0.0, whose tasks
have all completed, from pool
15/06/05 12:02:28 INFO scheduler.DAGScheduler: Stage 0 (count at <console>:15)
finished in 0.600 s
res2: Long = 13030

scala> 15/06/05 12:02:28 INFO scheduler.DAGScheduler: Job 0 finished: count at
<console>:15, took 0.958171 s
```

The number of lines in the file is 13030.

3. In a new terminal window, to verify the number of lines in the file, use the Linux command **wc** on the original file that we uploaded to HDFS:
   wc -l /home/biadmin/labfiles/Pr*

```
[biadmin@ibmclass ~]$ wc -l /home/labfiles/Pr*
13030 /home/labfiles/Pride_and_Prejudice.txt
[biadmin@ibmclass ~]$
```

4. Restart the Spark Shell, and then to read the first record of the RDD, type **pp.first().**

```
scala> pp.first()
15/06/05 12:40:14 INFO mapred.FileInputFormat: Total input paths to process : 1
15/06/05 12:40:14 INFO spark.SparkContext: Starting job: first at <console>:15
15/06/05 12:40:14 INFO scheduler.DAGScheduler: Got job 1 (first at <console>:15) with
1 output partitions (allowLocal=true)
15/06/05 12:40:14 INFO scheduler.DAGScheduler: Final stage: Stage 1(first at
<console>:15)
15/06/05 12:40:14 INFO scheduler.DAGScheduler: Parents of final stage: List()
15/06/05 12:40:14 INFO scheduler.DAGScheduler: Missing parents: List()
15/06/05 12:40:14 INFO scheduler.DAGScheduler: Submitting Stage 1
(Gutenberg/Pride_and_Prejudice.txt MappedRDD[3] at textFile at <console>:12), which
has no missing parents
15/06/05 12:40:14 INFO storage.MemoryStore: ensureFreeSpace(2560) called with
curMem=631860, maxMem=278302556
15/06/05 12:40:14 INFO storage.MemoryStore: Block broadcast_3 stored as values in
memory (estimated size 2.5 KB, free 264.8 MB)
15/06/05 12:40:14 INFO storage.MemoryStore: ensureFreeSpace(1901) called with
curMem=634420, maxMem=278302556
15/06/05 12:40:14 INFO storage.MemoryStore: Block broadcast_3_piece0 stored as bytes
in memory (estimated size 1901.0 B, free 264.8 MB)
15/06/05 12:40:14 INFO storage.BlockManagerInfo: Added broadcast_3_piece0 in memory
on localhost:46250 (size: 1901.0 B, free: 265.3 MB)
15/06/05 12:40:14 INFO storage.BlockManagerMaster: Updated info of block
broadcast_3_piece0
15/06/05 12:40:14 INFO spark.SparkContext: Created broadcast 3 from broadcast at
DAGScheduler.scala:838
15/06/05 12:40:14 INFO scheduler.DAGScheduler: Submitting 1 missing tasks from Stage
1 (Gutenberg/Pride_and_Prejudice.txt MappedRDD[3] at textFile at <console>:12)
15/06/05 12:40:14 INFO scheduler.TaskSchedulerImpl: Adding task set 1.0 with 1 tasks
15/06/05 12:40:14 INFO scheduler.TaskSetManager: Starting task 0.0 in stage 1.0 (TID
1, localhost, ANY, 1343 bytes)
15/06/05 12:40:14 INFO executor.Executor: Running task 0.0 in stage 1.0 (TID 1)
15/06/05 12:40:14 INFO rdd.HadoopRDD: Input split:
hdfs://ibmclass.localdomain:8020/user/biadmin/Gutenberg/Pride_and_Prejudice.txt:0+348
901
15/06/05 12:40:14 INFO mapred.LineRecordReader: Found UTF-8 BOM and skipped it
15/06/05 12:40:14 INFO executor.Executor: Finished task 0.0 in stage 1.0 (TID 1).
1796 bytes result sent to driver
15/06/05 12:40:14 INFO scheduler.TaskSetManager: Finished task 0.0 in stage 1.0 (TID
1) in 21 ms on localhost (1/1)
15/06/05 12:40:14 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 1.0, whose tasks
have all completed, from pool
15/06/05 12:40:14 INFO scheduler.DAGScheduler: Stage 1 (first at <console>:15)
finished in 0.021 s
15/06/05 12:40:14 INFO scheduler.DAGScheduler: Job 1 finished: first at <console>:15,
took 0.030293 s
res2: String = PRIDE AND PREJUDICE

scala>
```

The first actual line in the file has the string: *PRIDE AND PREJUDICE*.
This string is the title of the book.

Scala, Python, and Java are each substantive languages. It is not our goal to teach you the complete Scala language in this unit, but merely to introduce you to it.

Scala is an interpreted language, like Java, and has a compiler scalac just as Java has its compiler javac.

The next stage in your learning should be to take the free BigDataUniversity (BDU) course on Spark, which has programming exercises in Scala and Python that carry on from what you have learned here. The BDU course uses a free, downloadable VMware Image based on BigInsights v4 and Open Data Platform software. From there you would progress to one of the textbooks on learning Scala, but be warned that the best ones are often over 500 pages long.

Close all open windows.

Results:
You learned to use some of the fundamental aspects of running Spark in the Open Data Platform environment.