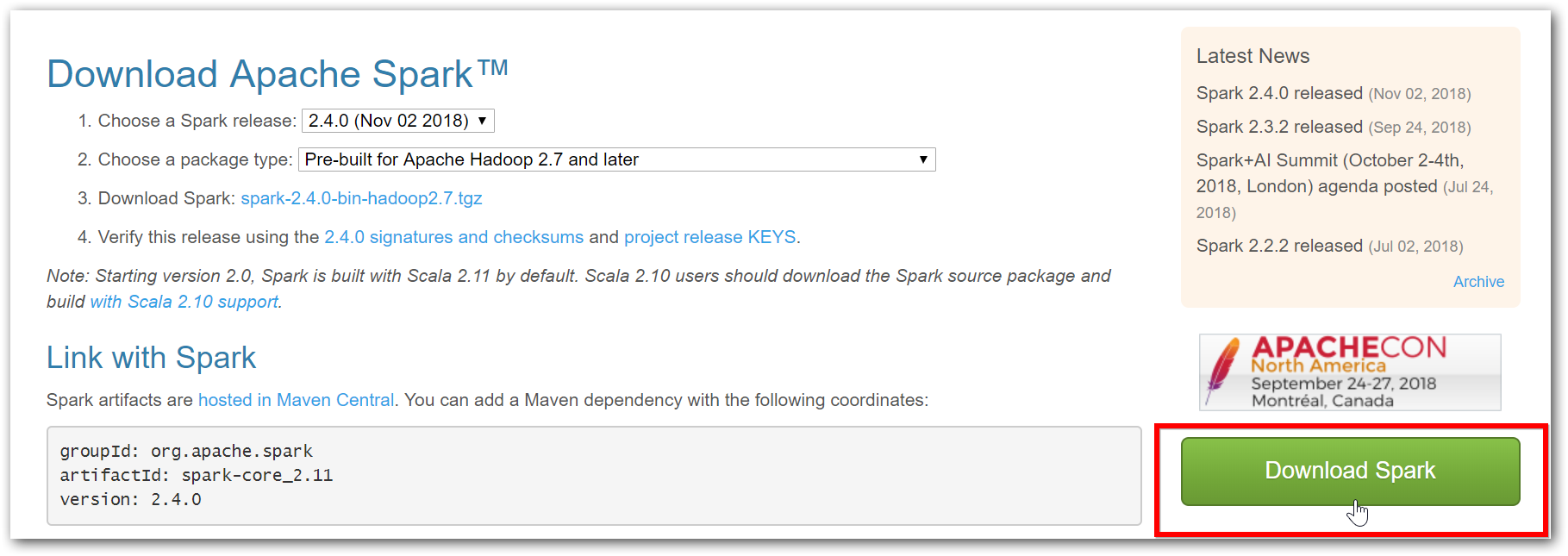
Hands on #2 – Spark with Hadoop + Azure Blob cluster

This is the continuation workshop after deploying Hadoop + Azure Blob cluster;

1. Download apache Spark stable release - <https://spark.apache.org/downloads.html>



1. Set SPARK\_HOME as environment variables (without bin),

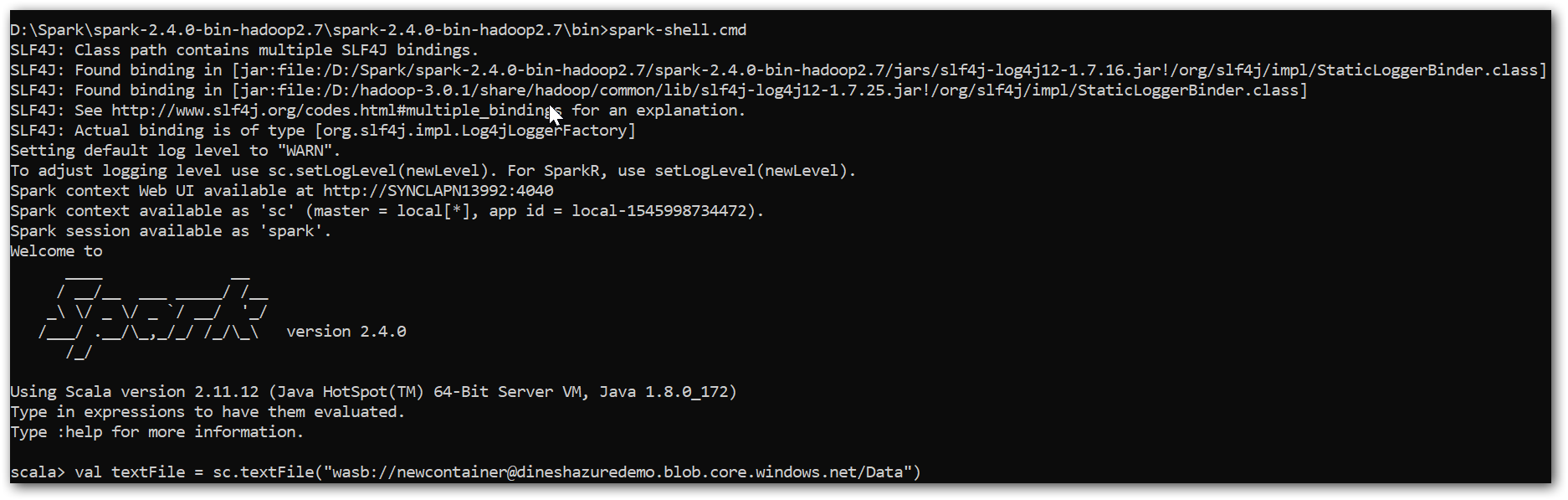
|  |  |
| --- | --- |
| SPARK\_HOME | D:\Spark\spark-2.4.0-bin-hadoop2.7\spark-2.4.0-bin-hadoop2.7\ |

1. Create a file called **spark-env.cmd** in directory “D:\Spark\spark-2.4.0-bin-hadoop2.7\spark-2.4.0-bin-hadoop2.7\conf\” and paste the following content in it,

|  |
| --- |
| @echo off  if exist %HADOOP\_HOME%\etc\hadoop (  set HADOOP\_CONF\_DIR=%HADOOP\_HOME%\etc\hadoop  )    if not defined SPARK\_CONF\_DIR (  set SPARK\_CONF\_DIR=%SPARK\_HOME%\conf  )  SET PATH=%PATH%;%JAVA\_HOME%\bin;%HADOOP\_HOME%\bin;  SET SPARK\_SUBMIT\_CLASSPATH=%SPARK\_HOME%\jars\derby\\*  set SPARK\_LOCAL\_HOSTNAME=%COMPUTERNAME%  SET SPARK\_DIST\_CLASSPATH=%SPARK\_HOME%\jars\\*;%HADOOP\_HOME%\share\hadoop\client\\*;%HADOOP\_HOME%\share\hadoop\common\\*;%HADOOP\_HOME%\share\hadoop\common\lib\\*;%HADOOP\_HOME%\share\hadoop\hdfs\\*;%HADOOP\_HOME%\share\hadoop\hdfs\lib\\*;%HADOOP\_HOME%\share\hadoop\mapreduce\\*;%HADOOP\_HOME%\share\hadoop\mapreduce\lib\\*;%HADOOP\_HOME%\share\hadoop\tools\lib\\*;%HADOOP\_HOME%\share\hadoop\tools\sources\\*;%HADOOP\_HOME%\share\hadoop\yarn\\*;%HADOOP\_HOME%\share\hadoop\yarn\lib\\* |

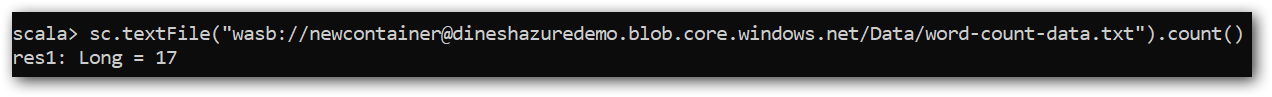
1. Download and copy the following jars to “D:\Spark\spark-2.4.0-bin-hadoop2.7\spark-2.4.0-bin-hadoop2.7\jars”
   1. azure-storage-8.0.0.jar ([link](http://central.maven.org/maven2/com/microsoft/azure/azure-storage/8.0.0/azure-storage-8.0.0.jar))
   2. hadoop-azure-2.7.7.jar ([link](http://central.maven.org/maven2/org/apache/hadoop/hadoop-azure/2.7.7/hadoop-azure-2.7.7.jar))
2. Open “spark-shell.cmd” and observe the output (perhaps with few WARN messages that you can simply disregard).

../spark-2.4.0-bin-hadoop2.7/bin> spark-shell.cmd



1. Tryout your first Spark-Scala code over file present in Azure Blob to find the number of lines,

scala> sc.textFile("wasb://newcontainer@dineshazuredemo.blob.core.windows.net/Data/word-count-data.txt").count()



1. And now perform wordcount over file present in Azure Blob and store the data into HDFS,

|  |
| --- |
| scala> val textFile = sc.textFile("wasb://newcontainer@dineshazuredemo.blob.core.windows.net/Data/word-count-data.txt")  scala> val counts = textFile.flatMap(line => line.split(" ")).map(word => (word, 1)).reduceByKey(\_+\_)  scala> counts.saveAsTextFile("/SparkOutput") |

You can find more samples at <https://spark.apache.org/examples.html>

1. Now let’s start to do samples with NASA logs,
   1. scala> :load "D:\Workshop\Scala Scripts\Scala Scripts\Exercise\_1.scala"