

## Case Study 5

### Task1

Create a Spark Application which streams data from a file on local directory on your machine and does the word count on the fly. The word should be done by the spark application in such a way that as soon as you drop the file in your local directory, your spark application should immediately do the word count for you.

### Code

```
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.log4j.{Level, Logger}

object SparkFileStreamingWordCount {

  def main(args: Array[String]): Unit = {

    println("hey Spark Streaming")

    val conf = new SparkConf().setMaster("local[2]").setAppName("SparkSteamingExample")
    val sc = new SparkContext(conf)
    val rootLogger = Logger.getRootLogger()
    rootLogger.setLevel(Level.ERROR)
    val ssc = new StreamingContext(sc, Seconds(15))
    val lines = ssc.textFileStream("/home/acadgild/case5") //for creating a Text Stream from
the file
    val words = lines.flatMap(_.split(" "))
    val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _) //calculating the wordcount
    wordCounts.print() // printing the count
    ssc.start() //starting streaming
    ssc.awaitTermination() //terminating the streaming

  }
}
```

## Task2

Create a Spark Application which should do the following:

1. Pick up a file from the local directory and do the word count
2. Then in the same Spark Application, write the code to put the same file on HDFS.
3. Then in same Spark Application, do the word count of the file copied on HDFS in step 2
4. Lastly, compare the word count of step 1 and 2. Both should match, other throw an error.

## Code

```
import java.io.File
import org.apache.spark.{SparkConf, SparkContext}
import scala.io.Source._
import org.apache.log4j.{Level, Logger}

object SparkHDFSWordCountComparison {

  private var localFilePath: File = new File("/home/acadgild/ test.txt") //location of the local
  file for which the wordcount is to be calculated
  private var dfsDirPath: String = "hdfs://localhost:8020/user/streaming" //location of
  directory on HDFS
  private val NPARAMS = 2

  def main(args: Array[String]): Unit = {
    //parseArgs(args)
    println("SparkHDFSWordCountComparison : Main Called Successfully")

    println("Performing local word count")
    val fileContents = readFile(localFilePath.toString()) //reading the contents of the
    local file

    println("Performing local word count - File Content ->" + fileContents)
    val localWordCount = runLocalWordCount(fileContents) //performing wordcount

    println("SparkHDFSWordCountComparison : Main Called Successfully -> Local Word Count
    is ->" + localWordCount)

    println("Performing local word count Completed !!")
    println("Creating Spark Context")

    val conf = new
    SparkConf().setMaster("local[2]").setAppName("SparkHDFSWordCountComparisonApp")
    val sc = new SparkContext(conf)
```

```

val rootLogger = Logger.getRootLogger()
    rootLogger.setLevel(Level.ERROR)

println("Spark Context Created")
println("Writing local file to DFS")
    val dfsFilename = dfsDirPath + "/dfs_read_write_test"
    val fileRDD = sc.parallelize(fileContents)           //creating RDD of the local file
    fileRDD.saveAsTextFile(dfsFilename)                 //saving the above as the text file over
                                                         the directory on HDFS
println("Writing local file to DFS Completed")

println("Reading file from DFS and running Word Count")
val readFileRDD = sc.textFile(dfsFilename)

//performing word count
val dfsWordCount = readFileRDD
    .flatMap(_.split(" "))
    .flatMap(_.split("\t"))
    .filter(_.nonEmpty)
    .map(w => (w, 1))
    .countByKey()
    .values
    .sum

sc.stop()           //forcefully stopping the streaming

//comparing the wordcounts and printing the message accordingly
if (localWordCount == dfsWordCount) {
    println(s"Success! Local Word Count ($localWordCount) " +
        s"and DFS Word Count ($dfsWordCount) agree.")
} else {
    println(s"Failure! Local Word Count ($localWordCount) " + s"and DFS Word Count ($dfsWordCount) disagree.")
}
}

private def printUsage(): Unit = {
    val usage: String = "DFS Read-Write Test\n" +
        "\n" +
        "Usage: localFile dfsDir\n" +
        "\n" +
        "localFile - (string) local file to use in test\n" +
        "dfsDir - (string) DFS directory for read/write tests\n"

    println(usage)
}

```

```

private def readFile(filename: String): List[String] = {
    val liner: Iterator[String] = fromFile(filename).getLines()
    val lineList: List[String] = liner.toList
    lineList
}

def runLocalWordCount(fileContents: List[String]): Int = {
    fileContents.flatMap(_.split(" "))
    .flatMap(_.split("\t"))
    .filter(_.nonEmpty)
    .groupBy(w => w)
    .mapValues(_.size)
    .values
    .sum
}
}

```

## Screenshots

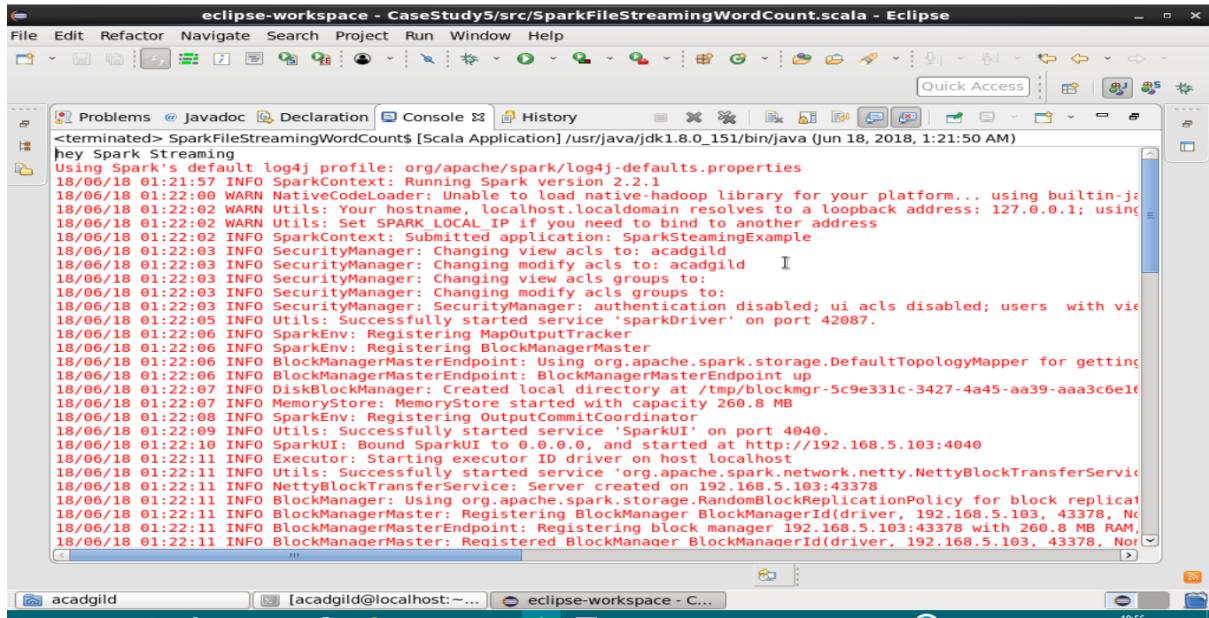
### Source files

```

[acadgild@localhost case5]$ cat f1.txt
Hey hi!
How are you?
[acadgild@localhost case5]$ cat f2.txt
Hi!
hey!
This is the first part of Case Study 5
[acadgild@localhost case5]$ cat f3.txt
Hello Hi!
Hi Hello!
[acadgild@localhost case5]$ cat f5.txt
Hey hey hey hey
Hello hello hello
[acadgild@localhost case5]$ cd
[acadgild@localhost ~]$ cat test.txt
This is a BDHS BDHS Session
This is a BDHS BDHS Session
This is a BDHS BDHS Session
This is a BDHS BDHS Session
[acadgild@localhost ~]$ █

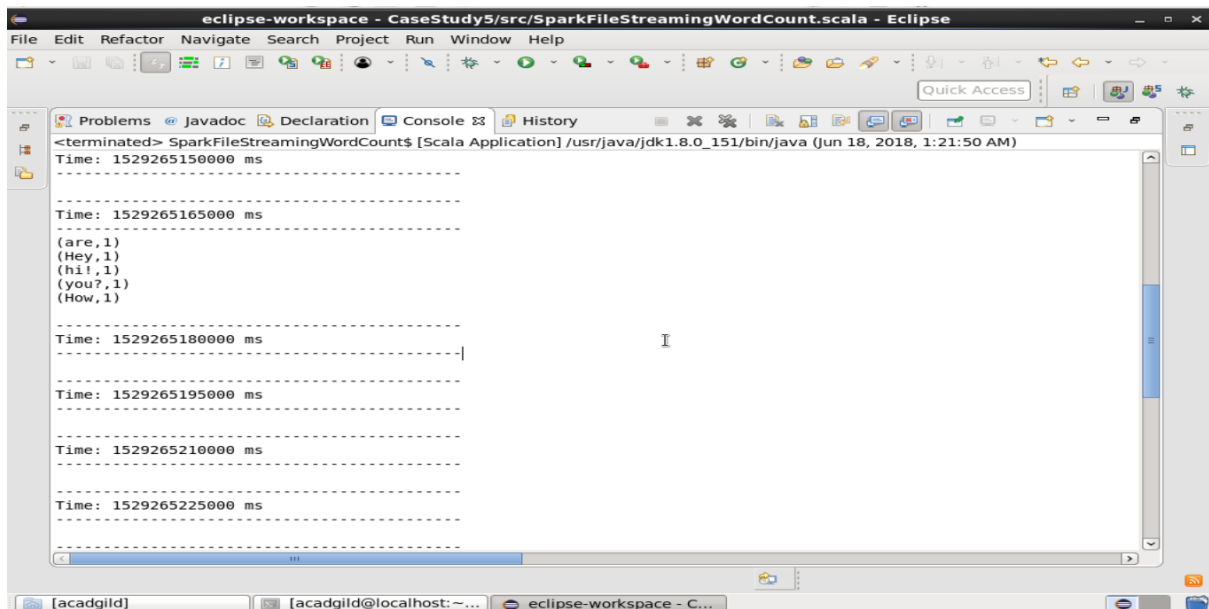
```

## Task1 outputs



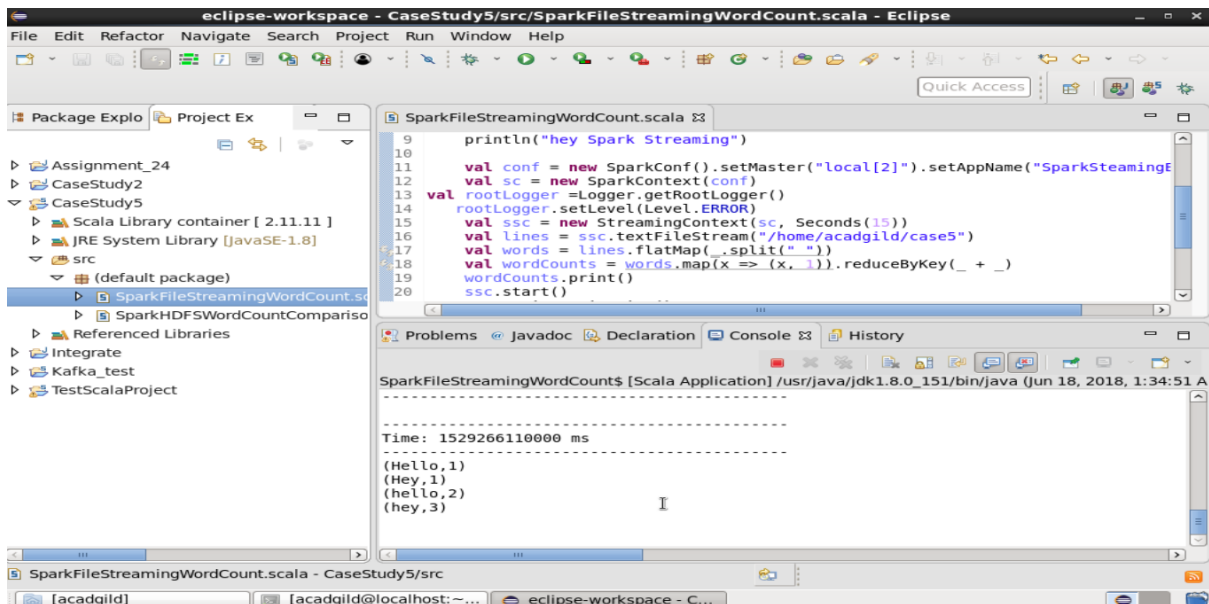
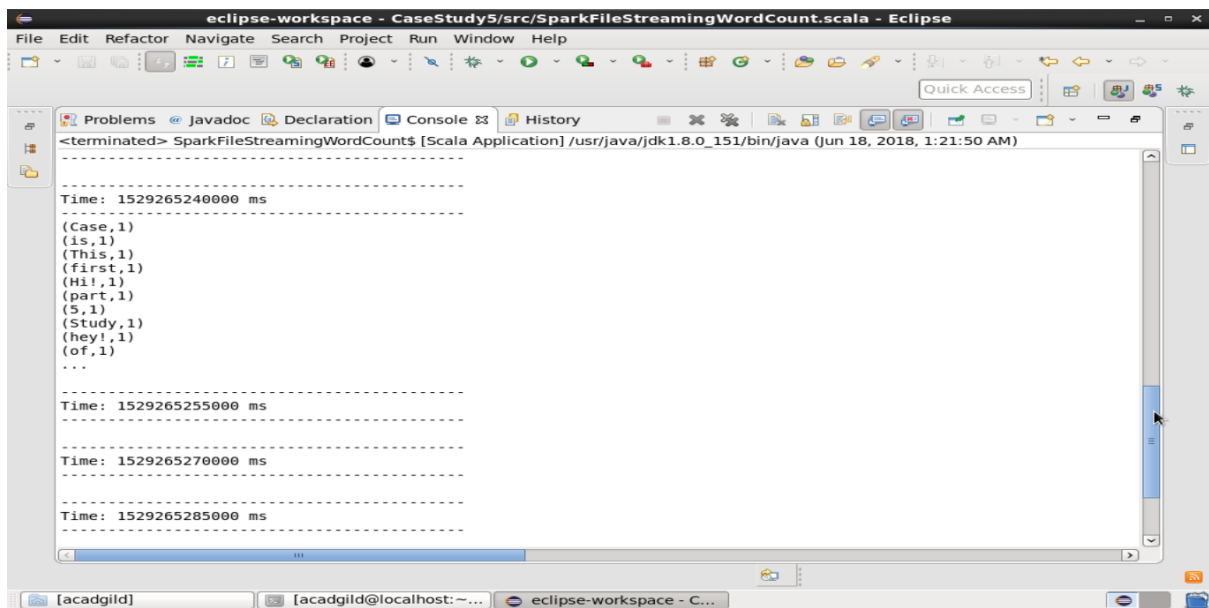
The screenshot shows the Eclipse IDE interface with the console window open. The title bar reads "eclipse-workspace - CaseStudy5/src/SparkFileStreamingWordCount.scala - Eclipse". The console output shows the Spark application starting. It begins with a red line indicating a terminated state, followed by a series of INFO and WARN messages. The logs include details about the SparkContext, security manager, and the successful start of the SparkEnv and SparkUI. The application is running on port 4040.

```
<terminated> SparkFileStreamingWordCount$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 18, 2018, 1:21:50 AM)
hey Spark Streaming
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
18/06/18 01:21:57 INFO SparkContext: Running Spark version 2.2.1
18/06/18 01:22:00 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-j
18/06/18 01:22:02 WARN Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using
18/06/18 01:22:02 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
18/06/18 01:22:02 INFO SparkContext: Submitted application: SparkSteamingExample
18/06/18 01:22:03 INFO SecurityManager: Changing view acls to: acadgild
18/06/18 01:22:03 INFO SecurityManager: Changing modify acls to: acadgild
18/06/18 01:22:03 INFO SecurityManager: Changing view acls groups to:
18/06/18 01:22:03 INFO SecurityManager: Changing modify acls groups to:
18/06/18 01:22:03 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with vie
18/06/18 01:22:05 INFO Utils: Successfully started service 'sparkDriver' on port 42087.
18/06/18 01:22:06 INFO SparkEnv: Registering MapOutputTracker
18/06/18 01:22:06 INFO SparkEnv: Registering BlockManagerMaster
18/06/18 01:22:06 INFO BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting
18/06/18 01:22:06 INFO BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
18/06/18 01:22:07 INFO DiskBlockManager: Created local directory at /tmp/blockmgr-5c9e331c-3427-4a45-aa39-aaa3c6e10
18/06/18 01:22:07 INFO MemoryStore: MemoryStore started with capacity 260.8 MB
18/06/18 01:22:08 INFO SparkEnv: Registering OutputCommitCoordinator
18/06/18 01:22:09 INFO Utils: Successfully started service 'SparkUI' on port 4040.
18/06/18 01:22:10 INFO SparkUI: Bound SparkUI to 0.0.0.0, and started at http://192.168.5.103:4040
18/06/18 01:22:11 INFO Executor: Starting executor ID driver on host localhost
18/06/18 01:22:11 INFO Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService
18/06/18 01:22:11 INFO NettyBlockTransferService: Server created on 192.168.5.103:43378
18/06/18 01:22:11 INFO BlockManager: Using org.apache.spark.storage.RandomBlockReplicationPolicy for block replicat
18/06/18 01:22:11 INFO BlockManagerMaster: Registering block manager BlockManagerId(driver, 192.168.5.103, 43378, No
18/06/18 01:22:11 INFO BlockManagerMasterEndpoint: Registering block manager 192.168.5.103:43378 with 260.8 MB RAM,
18/06/18 01:22:11 INFO BlockManagerMaster: Registered BlockManager BlockManagerId(driver, 192.168.5.103, 43378, No
```

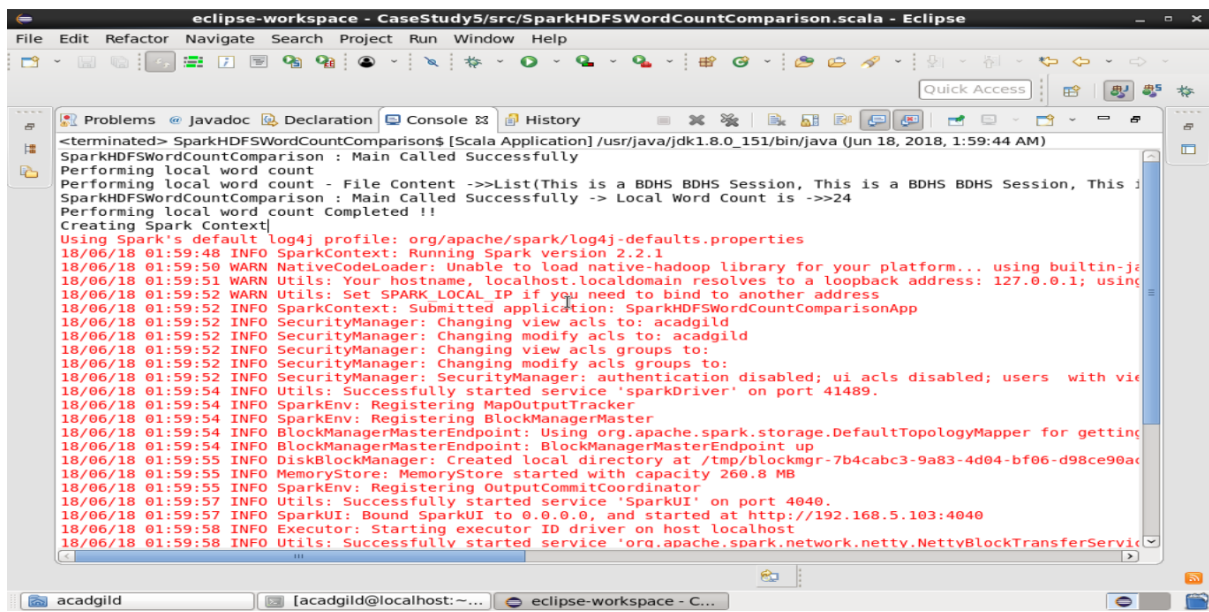


The screenshot shows the Eclipse IDE interface with the console window open. The title bar reads "eclipse-workspace - CaseStudy5/src/SparkFileStreamingWordCount.scala - Eclipse". The console output shows the results of the Spark application. It displays a series of timestamps and the output of the word count operation. The output shows the words "are", "hey", "hi", "you", and "how" with their respective counts.

```
<terminated> SparkFileStreamingWordCount$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 18, 2018, 1:21:50 AM)
Time: 1529265150000 ms
-----
Time: 1529265165000 ms
(are,1)
(hey,1)
(hi,1)
(you?,1)
(How,1)
-----
Time: 1529265180000 ms
-----
Time: 1529265195000 ms
-----
Time: 1529265210000 ms
-----
Time: 1529265225000 ms
-----
```

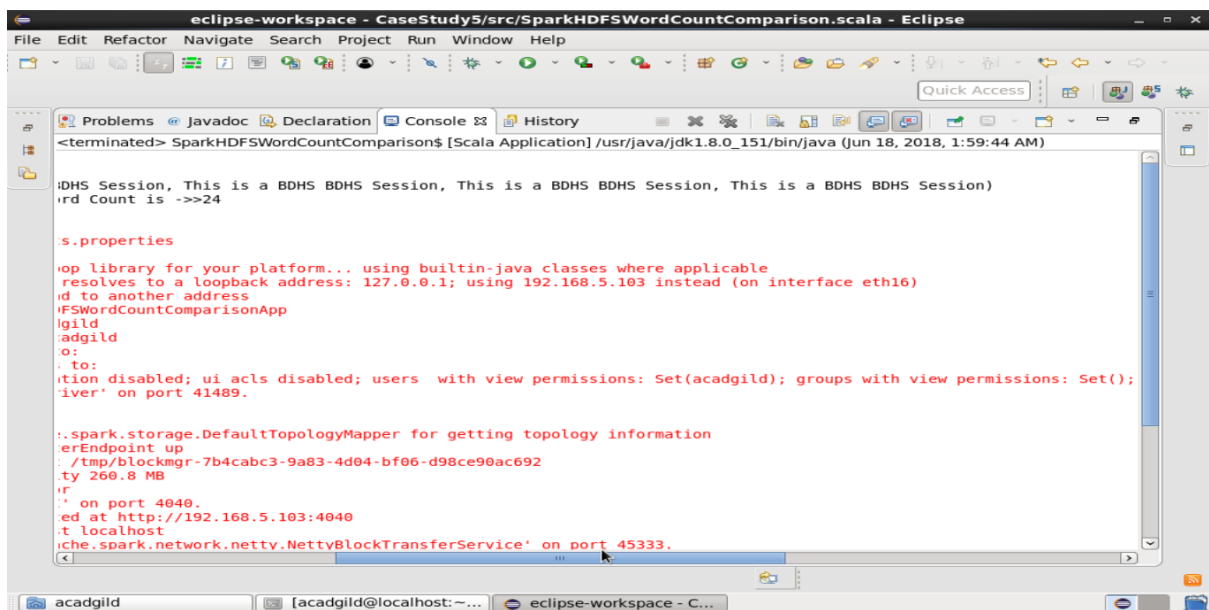


## Task2 outputs



```
eclipse-workspace - CaseStudy5/src/SparkHDFSWordCountComparison.scala - Eclipse
File Edit Refactor Navigate Search Project Run Window Help

<terminated> SparkHDFSWordCountComparison$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 18, 2018, 1:59:44 AM)
SparkHDFSWordCountComparison : Main Called Successfully
Performing local word count
Performing local word count - File Content ->List(This is a BDHS BDHS Session, This is a BDHS BDHS Session, This is a BDHS BDHS Session)
SparkHDFSWordCountComparison : Main Called Successfully -> Local Word Count is ->24
Performing local word count Completed !!
Creating Spark Context
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
18/06/18 01:59:48 INFO SparkContext: Running Spark version 2.2.1
18/06/18 01:59:50 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
18/06/18 01:59:51 WARN Utils: Your hostname, localhost.localdomain resolves to a loopback address: 127.0.0.1; using 192.168.5.103 instead (on interface eth16)
18/06/18 01:59:52 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
18/06/18 01:59:52 INFO SparkContext: Submitted application: SparkHDFSWordCountComparisonApp
18/06/18 01:59:52 INFO SecurityManager: Changing view acls to: acadgild
18/06/18 01:59:52 INFO SecurityManager: Changing modify acls to: acadgild
18/06/18 01:59:52 INFO SecurityManager: Changing view acls groups to:
18/06/18 01:59:52 INFO SecurityManager: Changing modify acls groups to:
18/06/18 01:59:52 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(acadgild); groups with view permissions: Set(); users with view permissions: Set(acadgild); groups with view permissions: Set();
18/06/18 01:59:54 INFO Utils: Successfully started service 'sparkDriver' on port 41489.
18/06/18 01:59:54 INFO SparkEnv: Registering MapOutputTracker
18/06/18 01:59:54 INFO SparkEnv: Registering BlockManagerMaster
18/06/18 01:59:54 INFO BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting topology information
18/06/18 01:59:54 INFO BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
18/06/18 01:59:55 INFO DiskBlockManager: Created local directory at /tmp/blockmgr-7b4cab3-9a83-4d04-bf06-d98ce90ac692
18/06/18 01:59:55 INFO MemoryStore: MemoryStore started with capacity 260.8 MB
18/06/18 01:59:55 INFO SparkEnv: Registering OutputCommitCoordinator
18/06/18 01:59:57 INFO Utils: Successfully started service 'SparkUI' on port 4040.
18/06/18 01:59:57 INFO SparkUI: Bound SparkUI to 0.0.0.0, and started at http://192.168.5.103:4040
18/06/18 01:59:58 INFO Executor: Starting executor ID driver on host localhost
18/06/18 01:59:58 INFO Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port 45333.
```



```
eclipse-workspace - CaseStudy5/src/SparkHDFSWordCountComparison.scala - Eclipse
File Edit Refactor Navigate Search Project Run Window Help

<terminated> SparkHDFSWordCountComparison$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 18, 2018, 1:59:44 AM)

BDHS Session, This is a BDHS BDHS Session, This is a BDHS BDHS Session, This is a BDHS BDHS Session)
rd Count is ->24

s.properties

op library for your platform... using builtin-java classes where applicable
resolves to a loopback address: 127.0.0.1; using 192.168.5.103 instead (on interface eth16)
rd to another address
rDFSWordCountComparisonApp
lgild
adgild
o:
: to:
tion disabled; ui acls disabled; users with view permissions: Set(acadgild); groups with view permissions: Set();
iver' on port 41489.

..spark.storage.DefaultTopologyMapper for getting topology information
erEndpoint up
: /tmp/blockmgr-7b4cab3-9a83-4d04-bf06-d98ce90ac692
ty 260.8 MB
'r
' on port 4040.
ed at http://192.168.5.103:4040
t localhost
che.spark.network.netty.NettyBlockTransferService' on port 45333.
```

```
eclipse-workspace - CaseStudy5/src/SparkHDFSWordCountComparison.scala - Eclipse
File Edit Refactor Navigate Search Project Run Window Help

Problems Javadoc Declaration Console History
<terminated> SparkHDFSWordCountComparison$ [Scala Application] /usr/java/jdk1.8.0_151/bin/java (Jun 18, 2018, 1:59:44 AM)
18/06/18 01:59:52 INFO SecurityManager: Changing view acls to: acadgild
18/06/18 01:59:52 INFO SecurityManager: Changing modify acls to: acadgild
18/06/18 01:59:52 INFO SecurityManager: Changing view acls groups to:
18/06/18 01:59:52 INFO SecurityManager: Changing modify acls groups to:
18/06/18 01:59:52 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view
18/06/18 01:59:54 INFO Uti ls: Successfully started service 'sparkDriver' on port 41489.
18/06/18 01:59:54 INFO SparkEnv: Registering MapOutputTracker
18/06/18 01:59:54 INFO SparkEnv: Registering BlockManagerMaster
18/06/18 01:59:54 INFO BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting
18/06/18 01:59:55 INFO DiskBlockManager: Created local directory at /tmp/blockmgr-7b4cab3-9a83-4d04-bf06-d98ce90ac
18/06/18 01:59:55 INFO MemoryStore: MemoryStore started with capacity 260.8 MB
18/06/18 01:59:55 INFO SparkEnv: Registering OutputCommitCoordinator
18/06/18 01:59:57 INFO Uti ls: Successfully started service 'SparkUI' on port 4040.
18/06/18 01:59:57 INFO SparkUI: Bound SparkUI to 0.0.0.0, and started at http://192.168.5.103:4040
18/06/18 01:59:58 INFO Executor: Starting executor ID driver on host localhost
18/06/18 01:59:58 INFO Uti ls: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService'
18/06/18 01:59:58 INFO NettyBlockTransferService: Server created on 192.168.5.103:45333
18/06/18 01:59:58 INFO BlockManager: Using org.apache.spark.storage.RandomBlockReplicationPolicy for block replicat
18/06/18 01:59:58 INFO BlockManagerMaster: Registering BlockManager BlockManagerId(driver, 192.168.5.103, 45333, No
18/06/18 01:59:58 INFO BlockManagerMasterEndpoint: Registering block manager 192.168.5.103:45333 with 260.8 MB RAM,
18/06/18 01:59:58 INFO BlockManagerMaster: Registered BlockManager BlockManagerId(driver, 192.168.5.103, 45333, No
18/06/18 01:59:58 INFO BlockManager: Initialized BlockManager: BlockManagerId(driver, 192.168.5.103, 45333, None)

Spark Context Created
Writing local file to DFS
Writing local file to DFS Completed
Reading file from DFS and running Word Count
Success! Local Word Count (24) and DFS Word Count (24) agree.

acadgild [acadgild@localhost:~... eclipse-workspace - C...
```