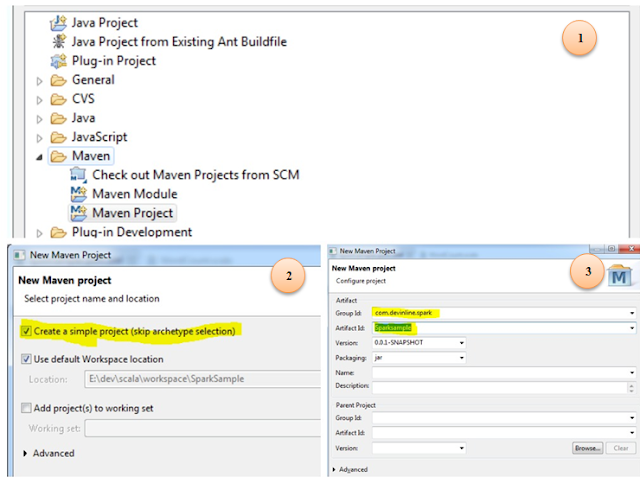
**Create a Maven project**:-  
Maven is a popular package management tool for Java-based languages that allows us to link libraries present in public repositories.We can use Maven itself to build our project, or use other tools like Scala’s sbt tool or Gradle.

**1**. Go to: **File-> New -> Project -> Maven project**  and create a maven project.Fill Group Id and Artifact Id & click finish.   
Group Id = com.devinline.spark and Artifact Id = SparkSample

[](http://1.bp.blogspot.com/-z4iPl0QKLFM/Vo4yhE6mw2I/AAAAAAAAJ3o/A51_1FEJ8zw/s1600/11111.PNG)

**2.** Update pom.xml:- [Download pom.xml](https://drive.google.com/file/d/0B-ur4R5mlgGLRGNETk5nX1JUNVE/view?usp=sharing) sample and update it in above maven project. It has spark dependency jar entry which will be downloaded while building.

**3.** Add Scala Nature to this project :-

Right click on project -> configure - > Add Scala Nature.

**4.** Update Scala compiler version for Spark:-

Scala IDE by default uses latest version(2.11) of Scala compiler, however Spark uses version 2.10.So we need to update appropriate version for IDE.

Right click on project- > Go to **properties -> Scala compiler -> update Scala installation** version to 2.10.5 

**5.** Remove Scala Library Container from build path :- (Optional)

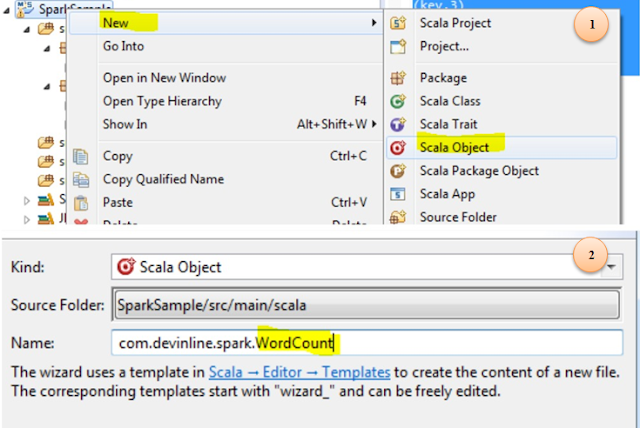
Jars required in already added via spark core(via pom.xml), so multiple jars is not required.

Right click on the project -> Build path -> Configure build path  and remove Scala Library Container.

**6.** Update source folder **src/main/java to src/main/scala** (Right click -> Refactor -> Rename  to scala).Now create a package under this name it as com.devinline.spark.

**7.**Create a Scala object under package created above name it as WordCount.scala

Right click on package -> New -> Scala Object  and add WordCount at the end of Name.

[](http://4.bp.blogspot.com/-3FQPyNgywv8/Vo4z6FR1r9I/AAAAAAAAJ4A/FPL2QnGyEzE/s1600/111111_345.PNG)

**8.** Update WordCount.scala with following code lines

**package** **com.devinline.spark**

**import** **org.apache.spark.SparkConf**

**import** **org.apache.spark.SparkContext**

**import** **org.apache.spark.rdd.RDD.rddToPairRDDFunctions**

**object** **WordCount** {

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

//Start the Spark context

**val** conf **=** **new** **SparkConf**()

.setAppName("WordCount")

.setMaster("local")

**val** sc **=** **new** **SparkContext**(conf)

//Read some example file to a test RDD

**val** test **=** sc.textFile("input.txt")

test.flatMap { line **=>** //for each line

line.split(" ") //split the line in word by word.

}

.map { word **=>** //for each word

(word, **1**) //Return a key/value tuple, with the word as key and 1 as value

}

.reduceByKey(**\_** + **\_**) //Sum all of the value with same key

.saveAsTextFile("output.txt") //Save to a text file

//Stop the Spark context

sc.stop

}

}

**Explanation**:- On applying **flatmap** unction on RDD test, each line is split with respect to space and array of string is obtained. This string array is converted into **map** with each word of list as key and 1 as value (collection of tuple is produced).Finally, **reduceByKey** is applied on for each tuple and aggregated output (unique word and corresponding count) is written to file. Lets take an example and understand the flow of method used in the above program unit.Suppose input.txt has two lines :  
 This is spark time  
 Learn spark

|  |
| --- |
| [http://1.bp.blogspot.com/-cJIKDouPW7k/VqrrYBZ6e5I/AAAAAAAAJ9U/XWCdNusPKmU/s640/Capture.PNG](http://1.bp.blogspot.com/-cJIKDouPW7k/VqrrYBZ6e5I/AAAAAAAAJ9U/XWCdNusPKmU/s1600/Capture.PNG) |
| Flow of method's used in word count example |

**9.**[Download sample input file](https://drive.google.com/file/d/0B-ur4R5mlgGLbTNjUFFaZ19jRkk/view?usp=sharing) and place is at some location as per your convenience. Modify location of input.txt in above sample code accordingly(sc.textFile("<Your\_input.txt\_Location>")).   
  
**10.** Execute wordcount program :-  Right click on WordCount.scala - > Run as -> Scala application. It should create an output directory output.txt  and it should contain two file : part-00000 and \_SUCCESS.  
Sample output in part-00000 is :-  
(spark,2)  
(is,1)  
(Learn,1)  
(This,1)  
(time,1