SETUP IDE FOR SPARK & SCALA ON WINDOWS

# Pre-Requisite:

# JDK 1.8: Scala, Spark require Java and JDK to develop and build the applications. Scala is JVM based programming language.

# Open command prompt and check java version installed in your machine. If java version is not 1.8, uninstall other version and install it.

# 

# Go to [official page of Oracle](https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html) where downloads are available

# Accept the terms and download correct version. Most modern days windows machine will be of 64 bit, so you might want to select 64 bit version.

# Once installed , use same command as shown above to validate.

1. **WINUTILS:** In the process of building data processing applications using Spark, we need to read data from files. Spark uses HDFS API to read files from several file systems like HDFS, s3, local etc. For HDFS APIs to work on Windows, we need to have WinUtils.

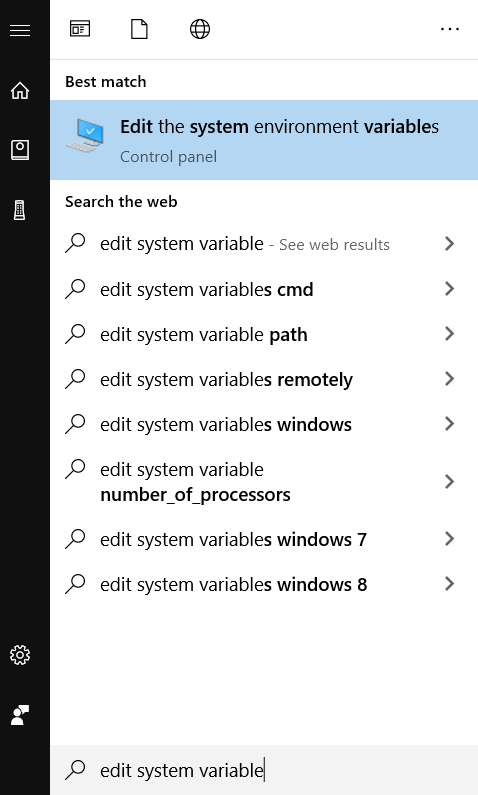
Download winutils from either of the following location –

1. <http://public-repo-1.hortonworks.com/hdp-win-alpha/winutils.exe>
2. <https://codeload.github.com/gvreddy1210/64bit/zip/master>. If you choose to download from this site, you have to extract it.

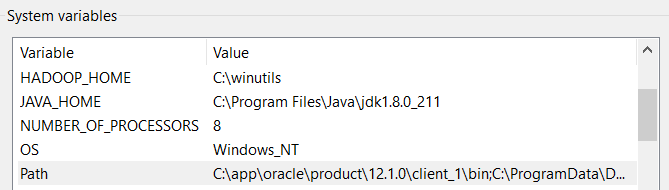
Keep the winutils.exe , in some folder, eg – C:/Hadoop/bin .

Make sure the directory you place Spark in is a directory that doesn't contain whitespaces – this is important.

1. **SETUP ENVIRONEMNT VARIABLE:** Type “edit system variable” in windows search box and select the highlighted option –

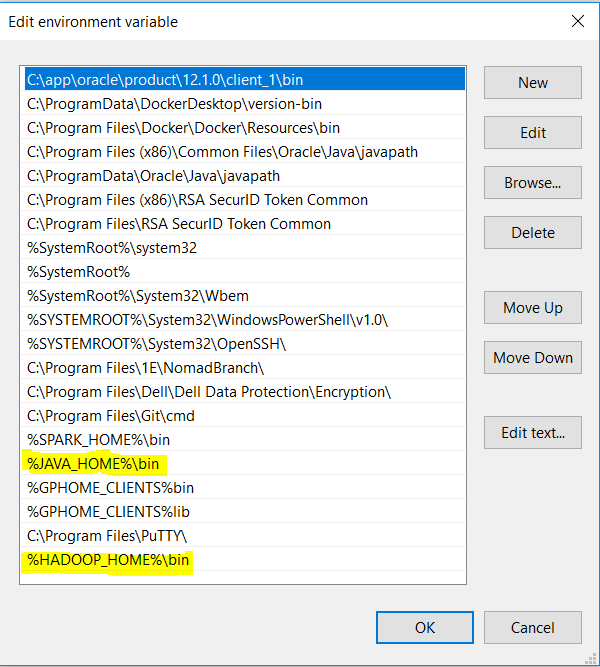
****

* Click on Add Environment Variables
* There will be 2 categories of environment variables. User Variables on top. System Variables on bottom
* Make sure to click on Add for System Variables
* Your entries should look like this –

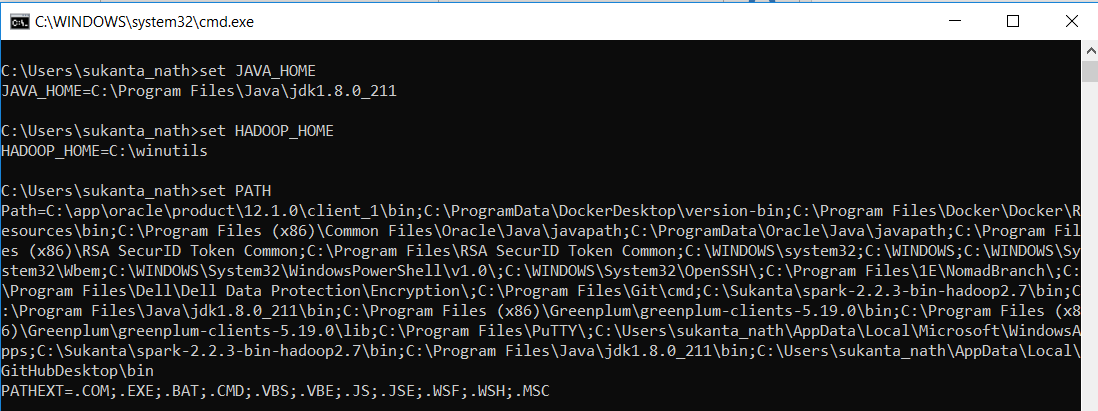


Notice, /bin has not been included in the environment variable values.

Also modify Path variable value, and make entries as shown below./bin has been included here. –



Once done ,verify the path values –



INSTALL INTELLIJ WITH SCALA & SPARK

### Go to [IntelliJ download page](https://www.jetbrains.com/idea/download/#section=windows) and install IntelliJ Community edition. It is a free edition and is popular among spark developers. It has code completion, integration with Git , code coverage and comparison feature.

### We will use Maven as our build tool. You can use SBT if your project requires so. Syntax of maintaining dependency will be different.

### Once Installation is complete, open IntelliJ and select File > Settings > Plugins and type Scala in search box. Select the Scala language and install it.

### 

### Next select, File > New > Project > Maven . Check “Create From ArchType” box and select scala-archtype-simple template. Notice how IntelliJ picked up Java 1.8 as Project SDK automatically.

### 

### Click next and enter GroupId & ArtifactId. Check out the standard naming convention for these 2 identifiers [here.](https://maven.apache.org/guides/mini/guide-naming-conventions.html) Here is an example -

### 

### Click next, review your group id, artifact id, move ahead and click on finish. Open the project in a “new window” when prompted.

### The project will open in new window and you’ll be prompted with 2 options – “Enable Auto Import” & “Import changes”. Click on later for time being and wait. The project will try to connect to maven repository central and download archtype template settings, necessary jars according to the pom.xml and setup correct directory structure. Once it is done, your project structure should look like this –

### 

### Notice, there is a folder named scala,the ArtifcatId becomes our Project Name and group id becomes the package. So, the folder structure becomes, ArtifactId > src > main > scala > GroupId and ArtifactId > src > test > scala > GroupId.

### Delete any scripts present under test folder. We do not want them right now.

### 

### Next, we have to update the pom.xml according to our requirement,

### The pom. xml file contains information of project and configuration information for the maven to build the project such as dependencies, build directory, source directory, test source directory, plugin, goals etc. Maven reads the pom. xml file, then executes the goal.

### Attached is a sample pom.xml. Copy the <repositories>, <dependencies>, <profiles> , <pluginRepositories> ,<build> tags contents. Do not replace your <groupId> and <artifactId>.

### 

### Once pom.xml is updated, IntelliJ will start downloading required jars from maven central repository. Given the fact you are setting up for first time, it will take considerable amount of time depending upon your internet speed,

### We will make sure scala sdk is added to project and auto import is enabled. Go to File > Settings > Build, Execution, Deployment > Build Tools > Maven and make sure following check boxes are ticked –

### 

### Now click on Compile option and make sure following check boxes are ticked.

### 

### Now open File > Project Structure > Libraries and add Scala SDK to project.

### 

### Select version 2.11.8 and click Ok (multiple times) to add this to your project.

### Now, it’s time to create a spark code in scala language to test our installation.

### Right click on your package > New > Scala Class > Object and name it as WordCount.

### Copy Paste the below code. Just be careful about the package name. Make sure it is the correct one as per your project setup.

**package** com.dell.it.dsc.eps  
  
**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(**"C:\\Users\\sukanta\_nath\\Documents\\My Received Files\\wc.txt"**)  
  
 test.foreach(*print*)  
  
 **val** counts = 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(**"C:\\Users\\sukanta\_nath\\Documents\\My Received Files\\output"**) *//Save to a text file  
  
 //Stop the Spark context* sc.stop  
 }  
}

### Save and Run it. It should finish with the message - Process finished with exit code 0

### Now go to the OutPut path and open the part-0000 file, you should be able to see the output.

### 

### If you can see, we are done with spark and scala setup! Happy Coding.

### .