**Spark Configuration**

**Steps to install Spark in local mode:**

1. [Install Java 7 or later](http://www.oracle.com/technetwork/java/javase/downloads/index.html). To test java installation is complete, open command prompt type java and hit enter. If you receive a message 'Java' is not recognized as an internal or external command. You need to configure your environment variables, JAVA\_HOME and PATH to point to the path of jdk.
2. [Download and install Scala](http://www.scala-lang.org/download/).

Set SCALA\_HOME in Control Panel\System and Security\System goto "Adv System settings" and add %SCALA\_HOME%\bin in PATH variable in environment variables.

1. Install Python 2.6 or later from [Python Download link](https://www.python.org/downloads/windows/).
2. [Download SBT](http://www.scala-sbt.org/download.html). Install it and set SBT\_HOME as an environment variable with value as <<SBT PATH>>.
3. Download winutils.exe from [HortonWorks repo](http://public-repo-1.hortonworks.com/hdp-win-alpha/winutils.exe) or [git repo](https://github.com/steveloughran/winutils/tree/master/hadoop-2.6.0/bin). Since we don't have a local Hadoop installation on Windows we have to download winutils.exe and place it in a bin directory under a created Hadoop home directory. Set HADOOP\_HOME = <<Hadoop home directory>> in environment variable.
4. We will be using a pre-built Spark package, so choose a Spark pre-built package for [Hadoop Spark download](http://spark.apache.org/downloads.html). Download and extract it.

Set SPARK\_HOME and add %SPARK\_HOME%\bin in PATH variable in environment variables.

1. Run command: spark-shell
2. Open http://localhost:4040/ in a browser to see the SparkContext web UI.

Also go through below link for basic understanding of what is Spark and it’s capabilities –

<http://spark.apach.org>

**Reference:**

<http://stackoverflow.com/questions/25481325/how-to-set-up-spark-on-windows>

**Environmental setup: (Zookeeper + Kafka):**

**Zookeeper:**

* Unzip the contents to local disk
* Open a  command prompt window and change the directory to unzipped folder
* Run command “.\bin\zkServer”
* This starts Zookeeper service on port 2181

**Kafka:**

* Unzip the contents to local disk
* Open a new command prompt window and change the directory to unzipped folder
* Run command “.\bin\windows\kafka-server .\config\server.properties”
* This starts Kafka server
* Observe log on the command prompt of Zookeeper. It reports about the Kafka service being registered

Also check the file “<kafka>\config\server.properties”. By default at the end there will be a line –

“zookeeper.connect=localhost:2181”

This makes Kafka to register with local Zookeeper service running on port 2181. We can change it depending on the machine where Zookeeper service is running.

**Compile the code:**

javac -cp <kafka\_path>\libs\\* -d . SampleProducer.java

**Make a Jar:**

jar –cf test.jar tm

**Run the class:**

java -cp .\\*;<kafka\_path>\libs\\* tm.poc.SampleProducer

**Spark-kafka configuration:**

* The attachedproject(Zipped project (SparkKafkaPOC.zip)) contains some scala code which I tried in the beginning. You can delete it once you import it into your IDE to avoid any compilation issues.

Once the project is build, you can the run the application as below –

* Start Zookeeper
* Start Kafka server
* From a new command window go to Spark home
* Run “.\bin\spark-submit --packages org.apache.spark:spark-streaming-kafka\_2.10:1.6.3 --class tm.poc.JavaWordCount --master local[4] <path of JAR built from project> localhost:2181 localhost <topic\_name> 1”
  + In the above command “—master local[4]” means you want to run the job as a standalone with master being local with 4 worker threads
  + localhost:2181 is the Zookeeper address
  + localhost is ID for the Kafka consumer we are creating
  + 1 is integer value for “StorageLevel”. There are several options like NONE, DISK\_ONLY, MEMORY\_ONLY etc. Here we are specifying NONE are we are storing any incoming messages
* Then in another command window run our custom producer which we experimented yesterday say for 20 or 25 messages
* Then see in the Spark window that it is continuously printing the word counts for every 2 secs