**Setting Up the Scala IDE: Preparing to Interact with Apache Spark**

Apache Spark is widely seen as a replacement for (or next generation of) Apache Hadoop’s MapReduce

Apache Spark allows a programmer to define *Resilient Distributed Datasets* (RDDs) in memory.

While you may often hear about Spark in the context of machine learning applications, it is important to note that Spark is not a machine learning framework. It defines methods for data transformation and manipulation over large clusters of computers (sound familiar?)

These methods are exposed to Spark users via Scala and Python libraries. It is through this exposure that machine learning libraries have been defined to leverage Spark’s native distributed computing capabilities.

In this course, we will be focusing on the Scala API for Spark. While the Scala API is available to us via an interactive shell program (more specifically a REPL[[1]](#footnote-1)), I find that this becomes inconvenient the more complex the task.

As a result, we will interact with Spark using Scala written on an Eclipse IDE specifically designed for that purpose.

**Step 1: Get the IDE**

Navigate to <http://www.scala-ide.org> (make sure you are doing this from the Cloudera virtual desktop)

Click on the download link:





Once the package is downloaded (it will be a gzipped tarball) you will need to unzip it. Navigate to its folder and extract the contents:

$ gzip zxvf path/to/ide.tar.gz

When you are doing this, ensure that you are not in the folder where your original eclipse files are. The extracted folder will overwrite your existing eclipse install and will not work. Recall that Cloudera ships with eclipse and the folder that contains those files is in the cloudera home folder.

The icon to run eclipse looks like this:



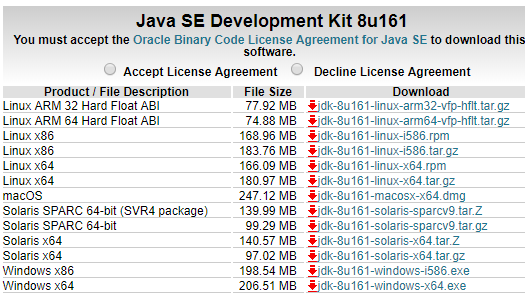
You will find it at the root of the newly extracted folder (that folder will be named “eclipse”). However, at this point, clicking on this file will throw an error. This is because Cloudera ships with Java 7 and this IDE requires Java 8.

**Step 2: Get and Install Java 8**

Eclipse needs the JDK, (not the JRE). The JDK is the java development kit and, critically, it includes a java compiler.

Navigate here: <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

Accept the EULA



Select this version for download

Once downloaded, you will need to unzip the gzipped tarball (use the same command in Step 1)

Now, you have to install it. The extracted folder should be called jdk1.8.0\_161 and you will need to move it to this directory: /usr/java/

$ mv jdk1.8.0\_161 /usr/java/

Now, you need to change the $JAVA\_HOME variable in the /etc/profile file.

$ sudo gedit /etc/profile

Toward the end of the file, you will see a line of code:

export JAVA\_HOME=/usr/java/jdk1.7.0\_67-cloudera

You will change that line to:

export JAVA\_HOME=/usr/java/jdk1.8.0\_161

Save the file.

Now, execute the code:

$ source /etc/profile

Test that this worked:

$ java -version

java version "1.8.0\_161"

Java(TM) SE Runtime Environment (build 1.8.0\_161-b12)

Java HotSpot(TM) 64-Bit Server VM (build 25.161-b12, mixed mode)

**Step 3: Set Up Eclipse**

You will need to change the eclipse.ini file to tell Eclipse what version of java it should use on startup. This file should look like this:

-startup

plugins/org.eclipse.equinox.launcher\_1.4.0.v20161219-1356.jar

--launcher.library

plugins/org.eclipse.equinox.launcher.gtk.linux.x86\_64\_1.1.500.v20170531-1133

-vm

/usr/java/jdk1.8.0\_161/bin/java

-vmargs

-Xmx2G

-Xms200m

-XX:MaxPermSize=384m

*I will walk you through the remainder of the eclipse set up as it is a bit too involved for this document. I will record that walkthrough and post to Blackboard.*

1. Read – Eval – Print Loop [↑](#footnote-ref-1)