GETTING STARTED WITH AMAZON WEB SERVICES

Note: You are responsible for managing your own account and cluster. You should **terminate** your cluster after you are done. You should manage your account carefully so that you do not exceed the education credits provided by Amazon. The instructor is not responsible for any financial loss or charges that you may incur.

Pre-Requisites

Before getting started, make sure you have the following:

1. AWS account –

   Sign up for an account at https://aws.amazon.com/

   You can apply for AWS Educate credit here:

   https://aws.amazon.com/education/awseducate/

2. SSH client:

   Mac OS X: Already has a built-in SSH client using Terminal App

   Windows:

   - You can download a console emulator like **cmder** from: http://cmder.net/

   This is the recommended approach as it gives you a bunch of built-in tools and also is a better user experience than Windows command prompt

   - You can download a SSH client like PuTTY. You can find a list at:

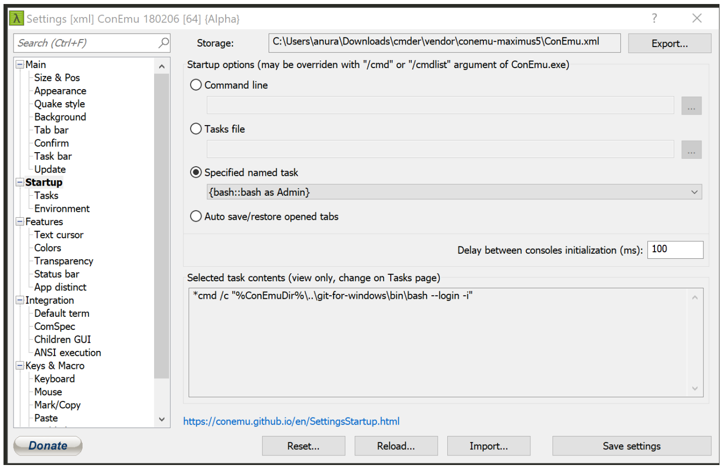
   https://www.ssh.com/ssh/client

   Unix: Already has lot of support for command line and SSH

3. Cmdr should be set up with a proper shell.

   Right click on the top bar of cmdr and select “Settings”. You can also press Win+Alt+p

   Go to “Startup -> Tasks” and select bash::bash as Admin in “Specified named task”



Settings [xml] ConEmu 180206 [64] {Alpha}

Storage: C:\Users\anura\Downloads\cmder\vendor\conemu-maximus5\ConEmu.xml

Startup options (may be overriden with "/cmd" or "/cmdlist" argument of ConEmu.exe)

○ Command line

○ Tasks file

● Specified named task

{bash::bash as Admin}

○ Auto save/restore opened tabs

Delay between consoles initialization (ms): 100

Selected task contents (view only, change on Tasks page)

```
*cmd /c "%ConEmuDir%\..\git-for-windows\bin\bash --login -i"
```

https://conemu.github.io/en/SettingsStartup.html

4. AWS Key Pair:

Follow the instructions below to create a key pair. Note the region where you created the key pair.
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html#having-ec2-create-your-key-pair

Note: If you are using a Windows SSH client like PuTTY, you need to convert .pem key to .ppk format. Follow the instructions here:
https://linuxacademy.com/howtoguides/posts/show/topic/17385-use-putty-to-access-ec2-linux-instances-via-ssh-from-windows

4. Local Development Tools:

You need a way to build projects locally and generate jar files. The following are required:

- **Java installation** with JDK (JDK 1.8 works best, JDK 9 is not compatible with many applications). You should be able to type **java -version** in cmder and get an output.

- **Scala installation**: can be downloaded from: https://www.scala-lang.org/download/. You should be able to type **scala -version** in cmdr and get an output. Scala version 2.11.8 is recommended.

- **Spark installation:**

For Mac OS X:
https://medium.freecodecamp.org/installing-scala-and-apache-spark-on-mac-os-837ae57d283f

For Windows 10 (Pro Edition):
You have to follow **all** instructions on this page:
https://hernandezpaul.wordpress.com/2016/01/24/apache-spark-installation-on-windows-10/

or on this page:
https://hernandezpaul.wordpress.com/2016/01/24/apache-spark-installation-on-windows-10/

- IntelliJ with Scala support:
Install the IDE IntelliJ and make sure to install the Scala plugin. Follow the instructions here:
https://docs.scala-lang.org/getting-started-intellij-track/getting-started-with-scala-in-intellij.html and make sure you can build and compile some sample code.

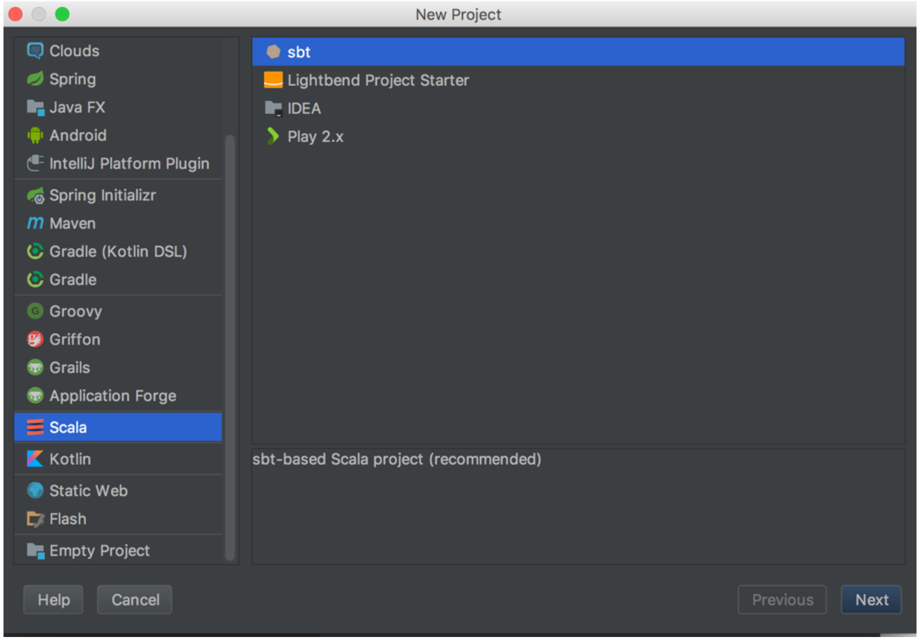
Building and Running a Simple Application

We are going to build a very simple application locally, generate jar file, and run it on AWS EMR. Please follow the steps below completely:

Step I: Building the Application

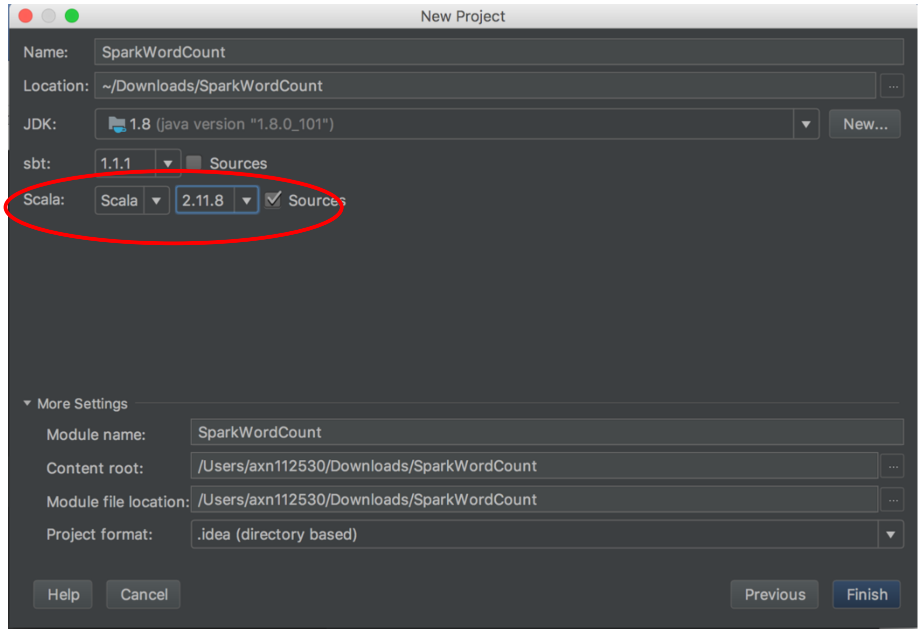
Note: You can download this project from
http://www.utdallas.edu/~axn112530/cs6350/spark/SparkWordCount.zip

Open IntelliJ Idea and choose “Create New Project”. Choose “Scala” on the left pane, and “sbt” in the center pane:



sbt-based Scala project (recommended)

In the next screen, choose a name for your application (“SparkWordCount”) and choose Scala version 2.11.8 (Important)



New Project

Name: SparkWordCount

Location: ~/Downloads/SparkWordCount

JDK: 1.8 (java version "1.8.0_101")

sbt: 1.1.1 □ Sources

Scala: Scala 2.11.8 ☑ Sources

More Settings

Module name: SparkWordCount

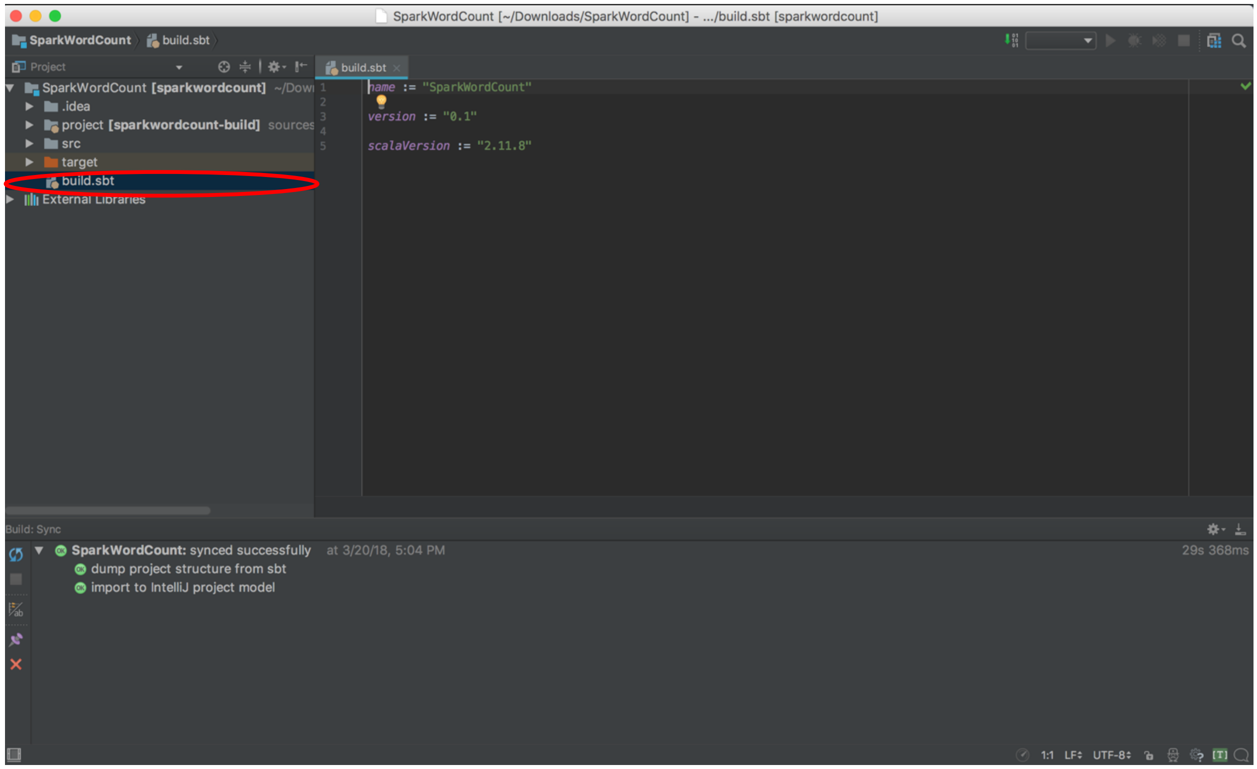
Content root: /Users/axn112530/Downloads/SparkWordCount

Module file location: /Users/axn112530/Downloads/SparkWordCount

Project format: .idea (directory based)

If you are performing these steps for the first time, it might take a bit for sbt project structure to be dumped. Be patient ☺

Open the build.sbt file. This file is used for managing dependencies and downloading jar files that your program needs.



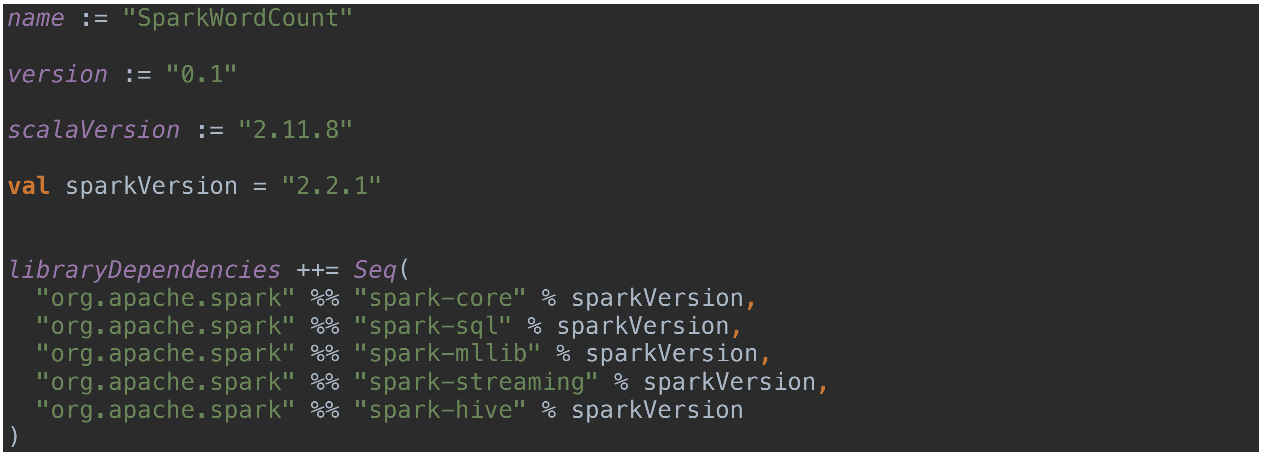
SparkWordCount [~/Downloads/SparkWordCount] - .../build.sbt [sparkwordcount]

```
SparkWordCount [sparkwordcount] ~/Dow
  .idea
  project [sparkwordcount-build] sources
  src
  target
  build.sbt
External Libraries
```

```
name := "SparkWordCount"

version := "0.1"

scalaVersion := "2.11.8"
```

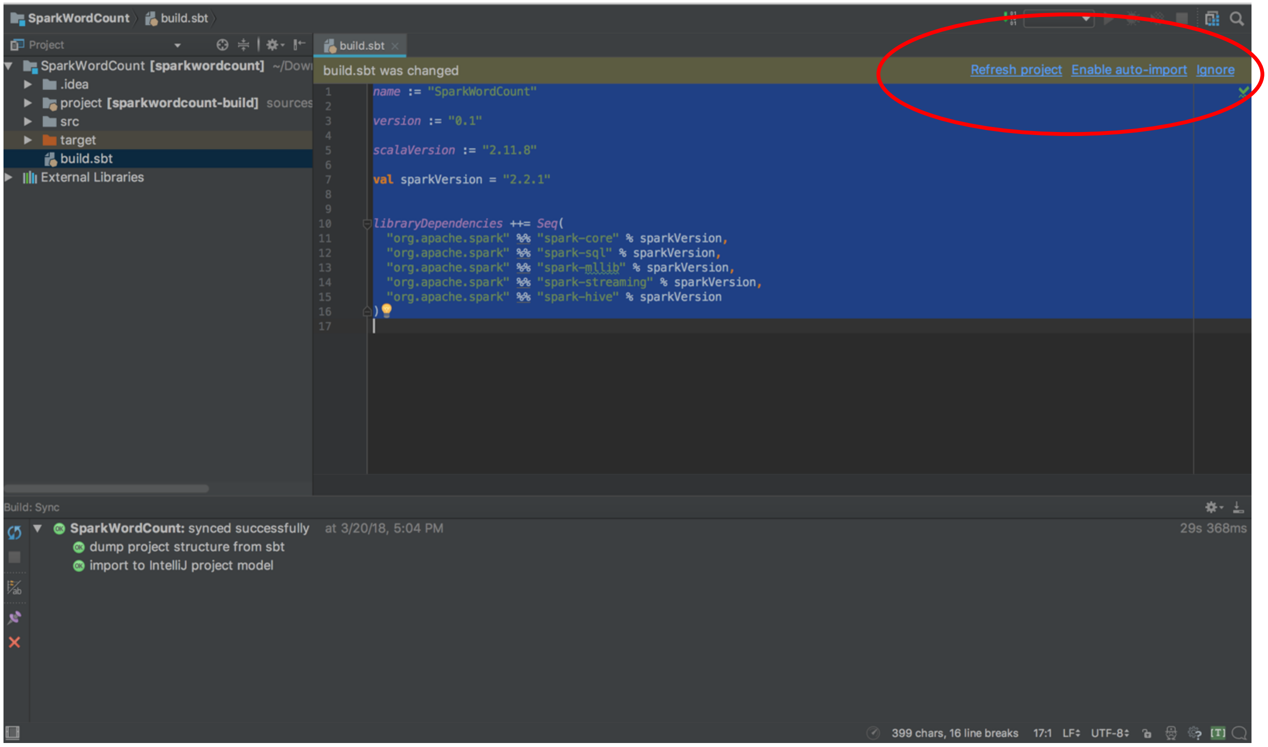
```
Build: Sync
SparkWordCount: synced successfully   at 3/20/18, 5:04 PM   29s 368ms
    dump project structure from sbt
    import to IntelliJ project model
```

Modify this file so that it contains the following:



```
name := "SparkWordCount"

version := "0.1"

scalaVersion := "2.11.8"

val sparkVersion = "2.2.1"


libraryDependencies ++= Seq(
  "org.apache.spark" %% "spark-core" % sparkVersion,
  "org.apache.spark" %% "spark-sql" % sparkVersion,
  "org.apache.spark" %% "spark-mllib" % sparkVersion,
  "org.apache.spark" %% "spark-streaming" % sparkVersion,
  "org.apache.spark" %% "spark-hive" % sparkVersion
)
```

When you do this, the IDE might show a message on top



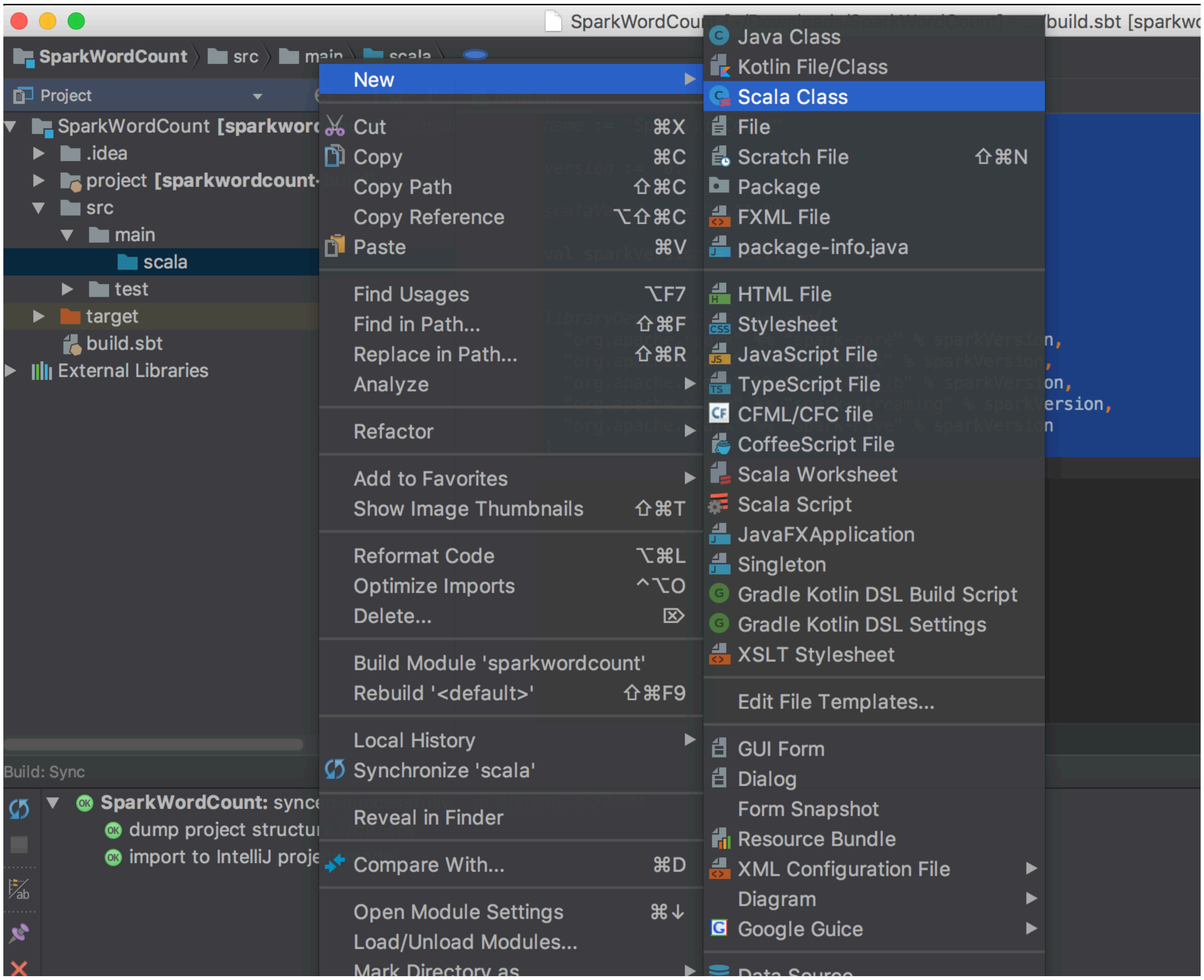
build.sbt was changed

```
name := "SparkWordCount"

version := "0.1"

scalaVersion := "2.11.8"

val sparkVersion = "2.2.1"


libraryDependencies ++= Seq(
  "org.apache.spark" %% "spark-core" % sparkVersion,
  "org.apache.spark" %% "spark-sql" % sparkVersion,
  "org.apache.spark" %% "spark-mllib" % sparkVersion,
  "org.apache.spark" %% "spark-streaming" % sparkVersion,
  "org.apache.spark" %% "spark-hive" % sparkVersion
)
```

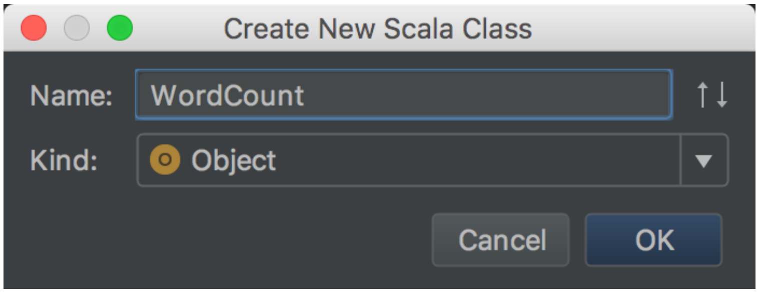
```
SparkWordCount: synced successfully   at 3/20/18, 5:04 PM   29s 368ms
  dump project structure from sbt
  import to IntelliJ project model
```

Click on either “Refresh Project” or “Enable auto-import” to import the dependencies. This might take a while if you are doing this for the first time. If everything succeeds, you should see a bunch of jar files appear under the “External Libraries” folder.

Now click on src -> main -> scala in the Project Explorer pane on the left, and then *right click* on "scala" to open the dialog box as shown in the next figure. Choose "Scala Class"



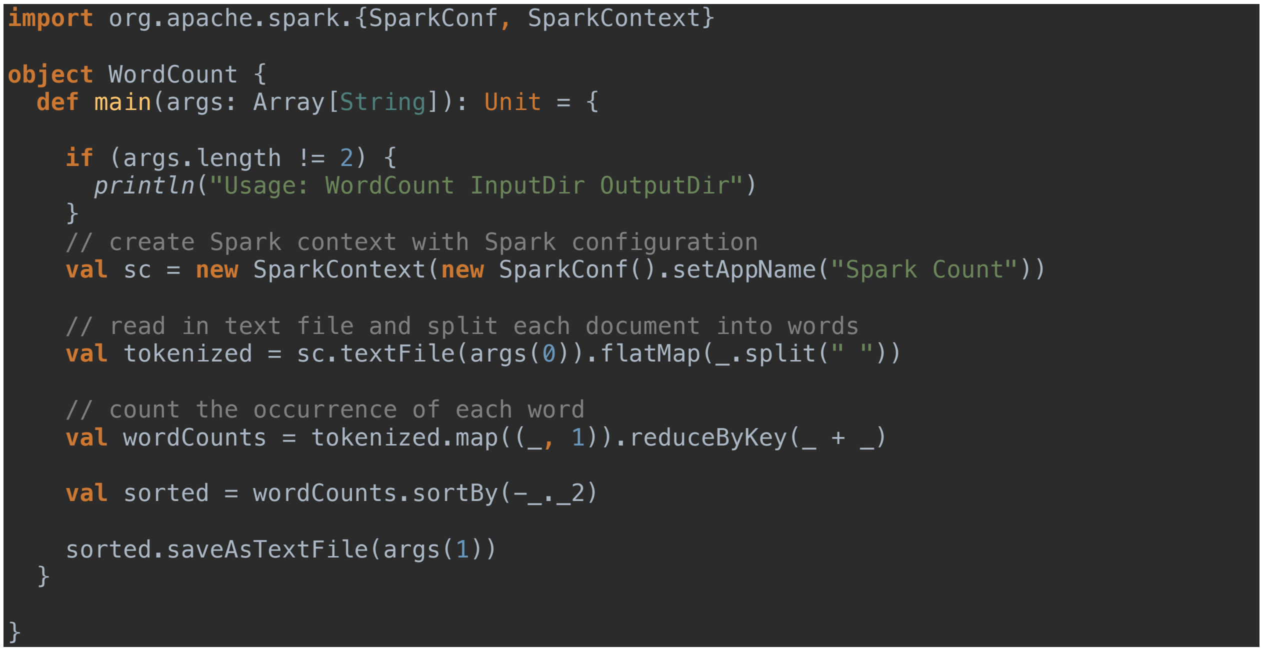
In the next dialog box, name your class and change the "kind" property to be "Object"



Name: WordCount

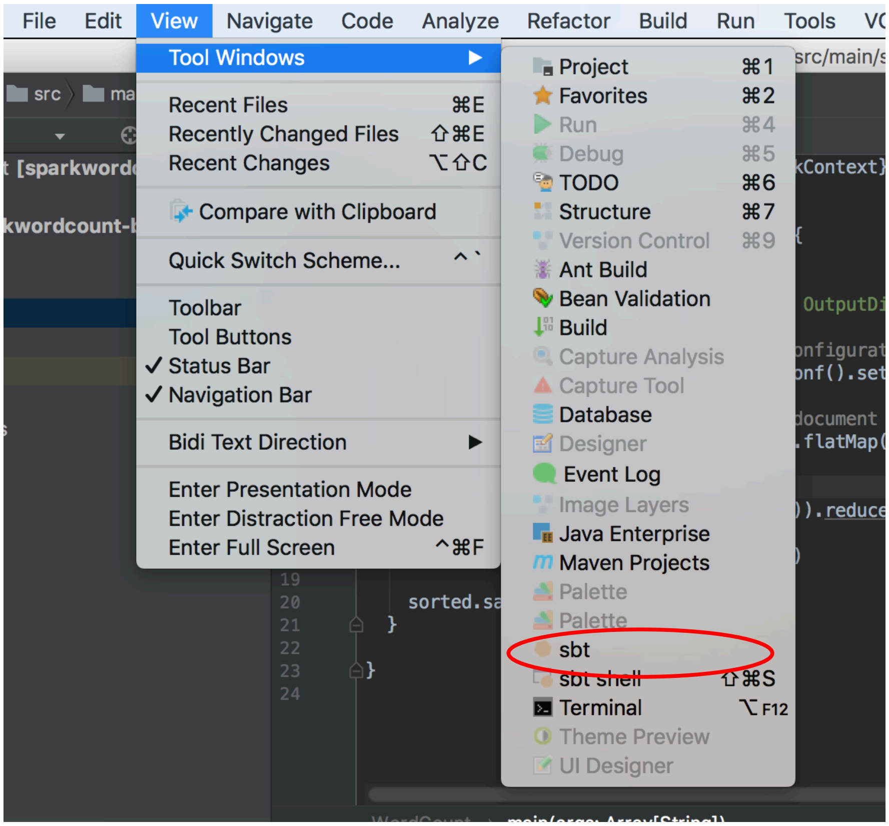
Kind: Object

Next, we will write the code below in the file WordCount.scala

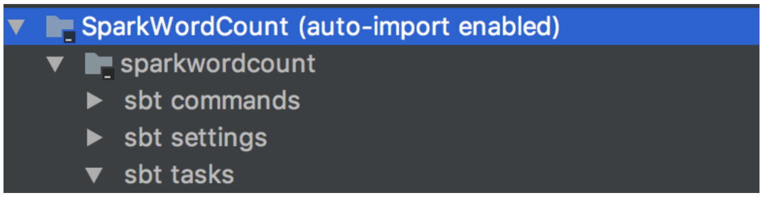


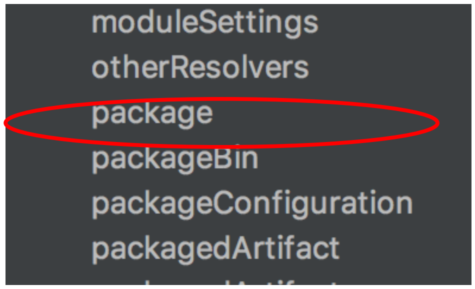
```
import org.apache.spark.{SparkConf, SparkContext}

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

    if (args.length != 2) {
      println("Usage: WordCount InputDir OutputDir")
    }
    // create Spark context with Spark configuration
    val sc = new SparkContext(new SparkConf().setAppName("Spark Count"))

    // read in text file and split each document into words
    val tokenized = sc.textFile(args(0)).flatMap(_.split(" "))

    // count the occurrence of each word
    val wordCounts = tokenized.map((_, 1)).reduceByKey(_ + _)

    val sorted = wordCounts.sortBy(-_._2)

    sorted.saveAsTextFile(args(1))
  }

}
```

Now go to “View” menu and then click on “Tool Windows” and then click “sbt”

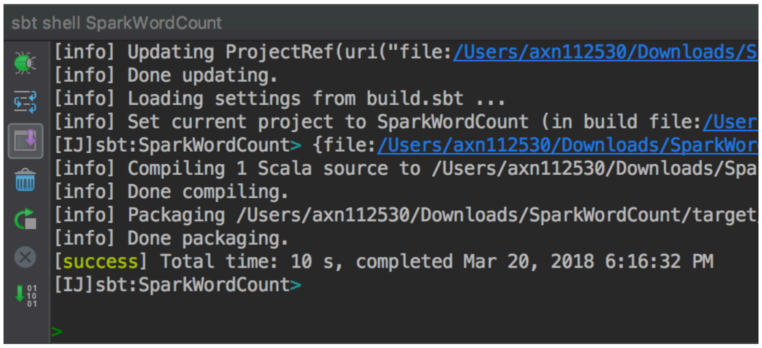


On the “sbt projects” pane on the right, expand “sbt tasks” and then go down to the “package” task and double click on it.



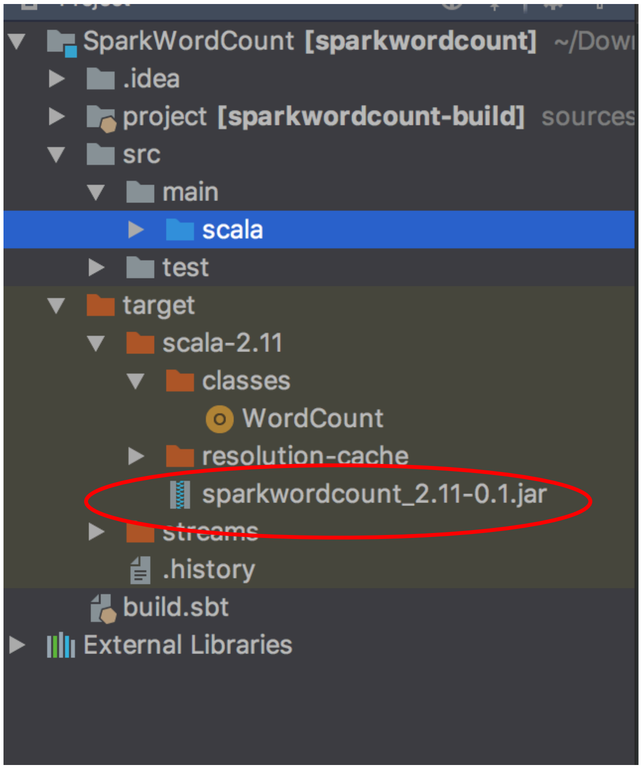


If everything goes OK, you should see a success message in the output window:



```
sbt shell SparkWordCount
[info] Updating ProjectRef(uri("file:/Users/axn112530/Downloads/S
[info] Done updating.
[info] Loading settings from build.sbt ...
[info] Set current project to SparkWordCount (in build file:/User
[IJ]sbt:SparkWordCount> {file:/Users/axn112530/Downloads/SparkWor
[info] Compiling 1 Scala source to /Users/axn112530/Downloads/Spa
[info] Done compiling.
[info] Packaging /Users/axn112530/Downloads/SparkWordCount/target
[info] Done packaging.
[success] Total time: 10 s, completed Mar 20, 2018 6:16:32 PM
[IJ]sbt:SparkWordCount>

>
```

You should also see a jar file in the target/scala-2.11/classes folder:



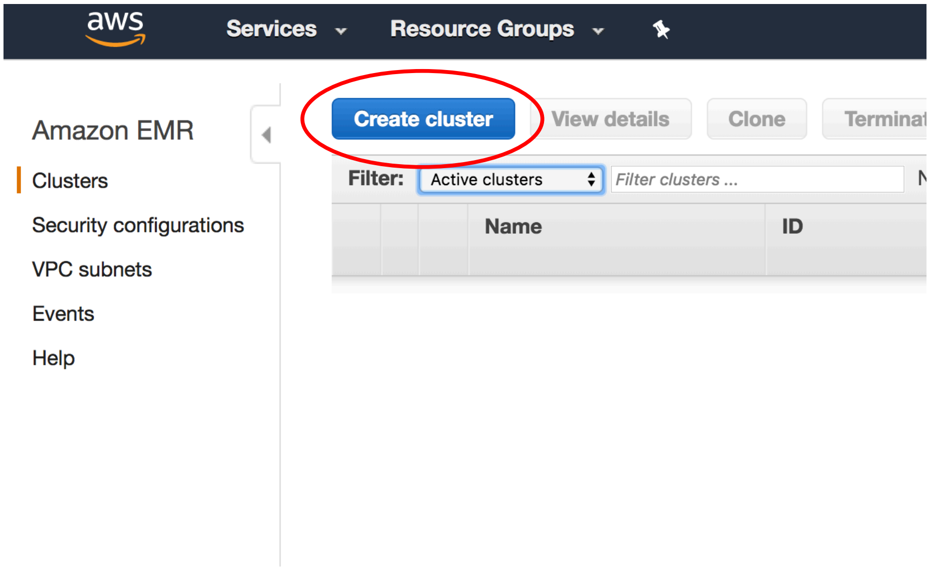
```
▼ SparkWordCount [sparkwordcount] ~/Dow
  ▶ .idea
  ▶ project [sparkwordcount-build] sources
  ▼ src
    ▼ main
      ▶ scala
    ▶ test
  ▼ target
    ▼ scala-2.11
      ▼ classes
          WordCount
      ▶ resolution-cache
        sparkwordcount_2.11-0.1.jar
    ▶ streams
      .history
    build.sbt
▶ External Libraries
```

Now, we can run this jar file on AWS cluster.

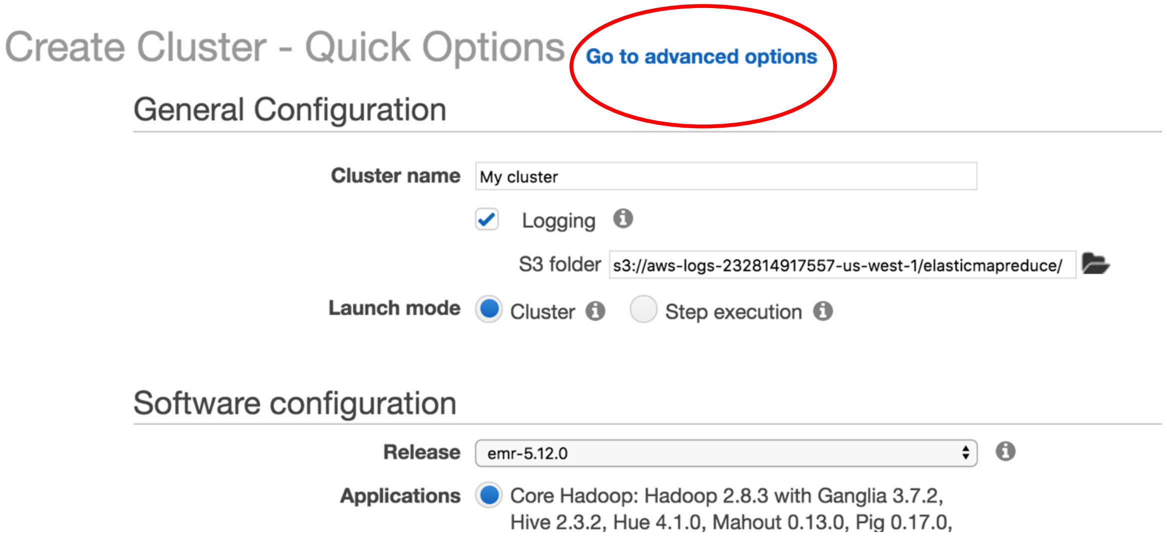
Step II: Launching AWS Cluster

Go to the link https://aws.amazon.com/emr/ and sign into your account. Make sure the region matches the region where you created your key pair. For example, if you created a key pair in N.California, you should start your cluster also in N.California.

Create on the "Create Cluster" button:



In the next screen, click on “Go to advanced options” link:



Create Cluster - Quick Options Go to advanced options

General Configuration

Cluster name: My cluster

☑ Logging

S3 folder: s3://aws-logs-232814917557-us-west-1/elasticmapreduce/

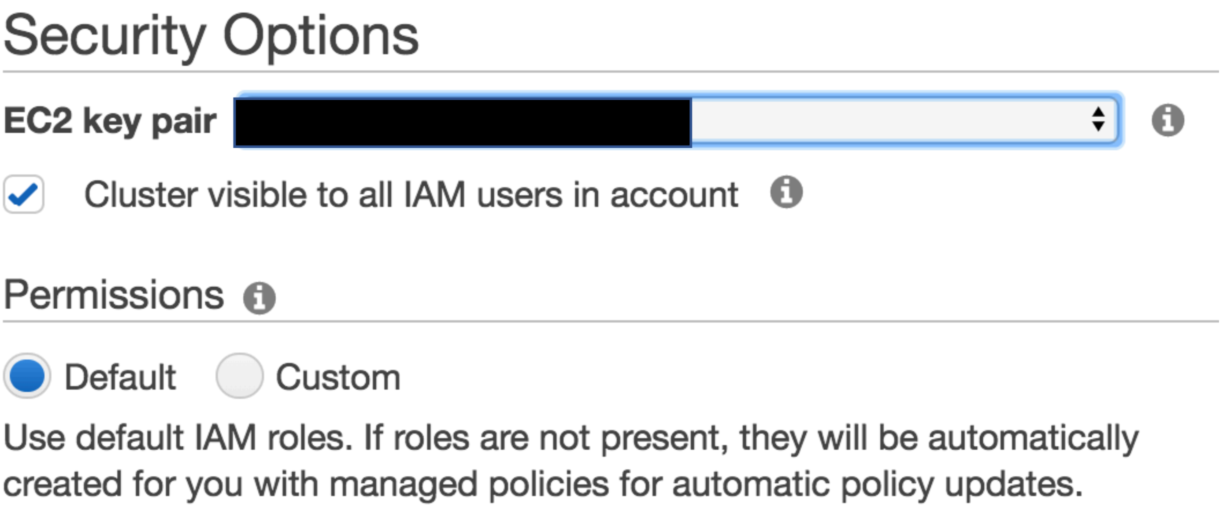
Launch mode: ● Cluster ○ Step execution

Software configuration

Release: emr-5.12.0

Applications: ● Core Hadoop: Hadoop 2.8.3 with Ganglia 3.7.2, Hive 2.3.2, Hue 4.1.0, Mahout 0.13.0, Pig 0.17.0,

In the advanced options screen, choose “Spark 2.2.1” and leave everything else as it is. Click next and accept other default values. Accept defaults on the next screen for Hardware Configuration also. In the General Options screen also, accept the default values. In the Security Options screen, make sure to select a valid key pair and default permissions:



Security Options

EC2 key pair: [redacted]

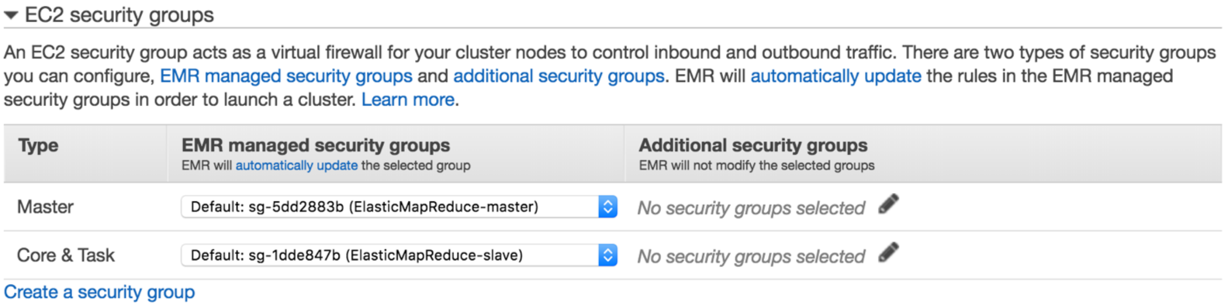
☑ Cluster visible to all IAM users in account

Permissions

● Default ○ Custom

Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

In the "EC2 security group", it should show you the name of the managed security group.



▾ EC2 security groups

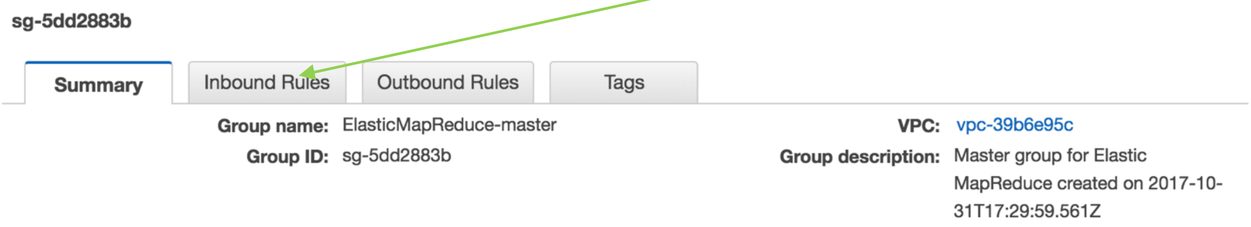
An EC2 security group acts as a virtual firewall for your cluster nodes to control inbound and outbound traffic. There are two types of security groups you can configure, EMR managed security groups and additional security groups. EMR will automatically update the rules in the EMR managed security groups in order to launch a cluster. Learn more.

Type	EMR managed security groups EMR will automatically update the selected group	Additional security groups EMR will not modify the selected groups
Master	Default: sg-5dd2883b (ElasticMapReduce-master)	No security groups selected
Core & Task	Default: sg-1dde847b (ElasticMapReduce-slave)	No security groups selected

Create a security group

Optional: Some people have found the step below useful. It's not required for most cases.

Note the name of the security group for the master. Then go to https://console.aws.amazon.com/vpc/ and then click on "Security Groups" on left pane. Look for the name of the security group and check the box next to it. Modify the inbound and outbound rules to allow traffic from all TCP or your own IP.



sg-5dd2883b

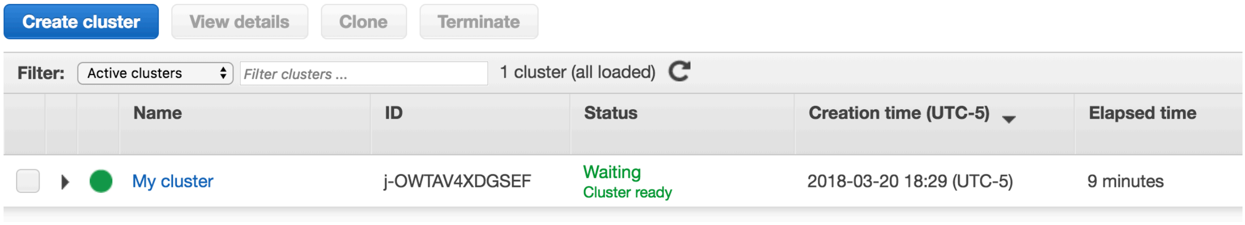
Group name: ElasticMapReduce-master

Group ID: sg-5dd2883b

VPC: vpc-39b6e95c

Group description: Master group for Elastic MapReduce created on 2017-10-31T17:29:59.561Z

Finally click on "Create Cluster" button. The cluster creation process should take about 10-15 minutes. If it takes much longer, it's likely that you made a mistake in one of the choices. When the cluster is ready, it will go to the "Waiting" stage:



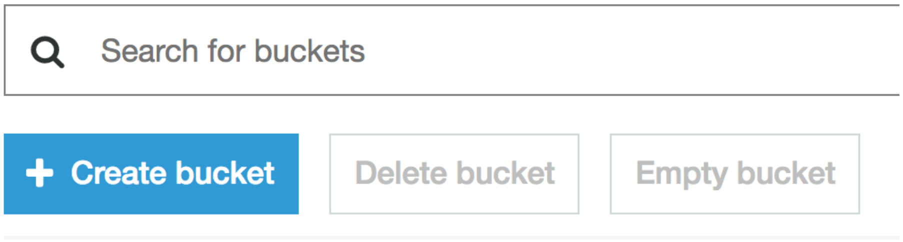
Filter: Active clusters | Filter clusters ... | 1 cluster (all loaded)

Name	ID	Status	Creation time (UTC-5)	Elapsed time
● My cluster	j-OWTAV4XDGSEF	Waiting Cluster ready	2018-03-20 18:29 (UTC-5)	9 minutes

Step III: Running the jar file

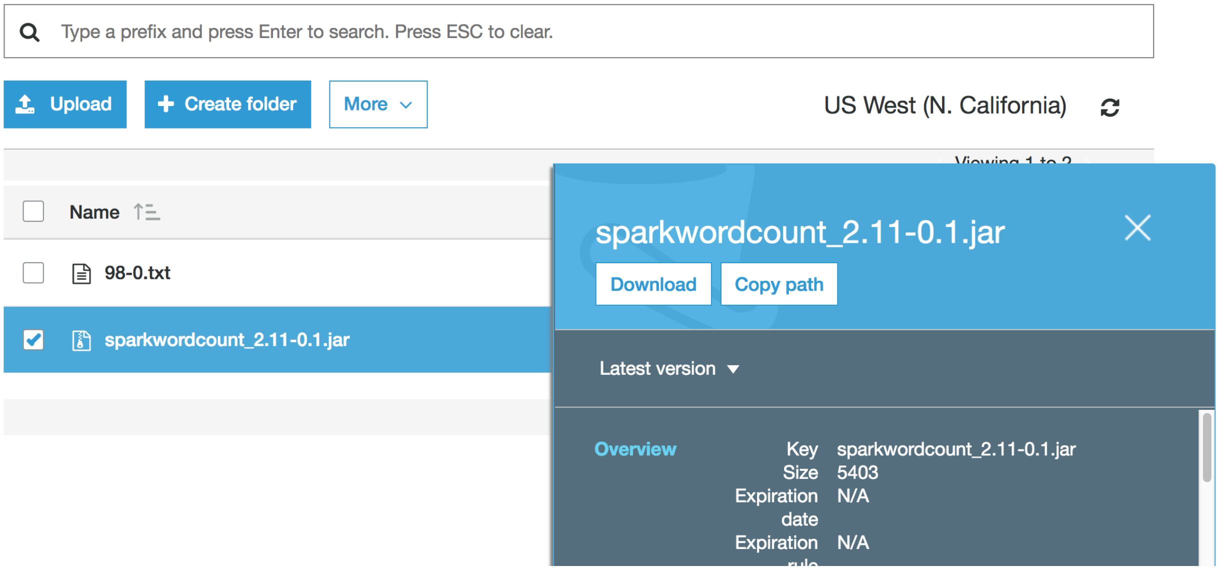
Run your jar file as below:

Login into your S3 account: https://aws.amazon.com/s3/ and create bucket in the same region (e.g. N. California)



Make sure your bucket name is unique. Accept all default values and create the bucket.

Download any text file from the Gutenberg project (http://www.gutenberg.org/) to your computer and then upload it to S3 bucket. Also upload the jar file that you created earlier to your S3 bucket. You can click on the files in S3 to copy their paths to your clipboard:



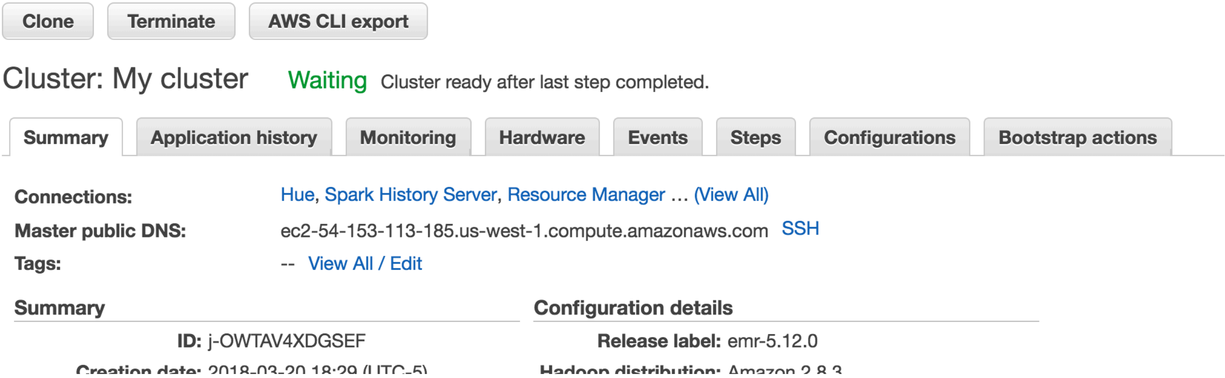
□ 98-0.txt

☑ sparkwordcount_2.11-0.1.jar

sparkwordcount_2.11-0.1.jar

Key sparkwordcount_2.11-0.1.jar
Size 5403
Expiration date N/A
Expiration rule N/A

Now go back to your EMR cluster, click on the cluster name and then click on the “Steps” tab.



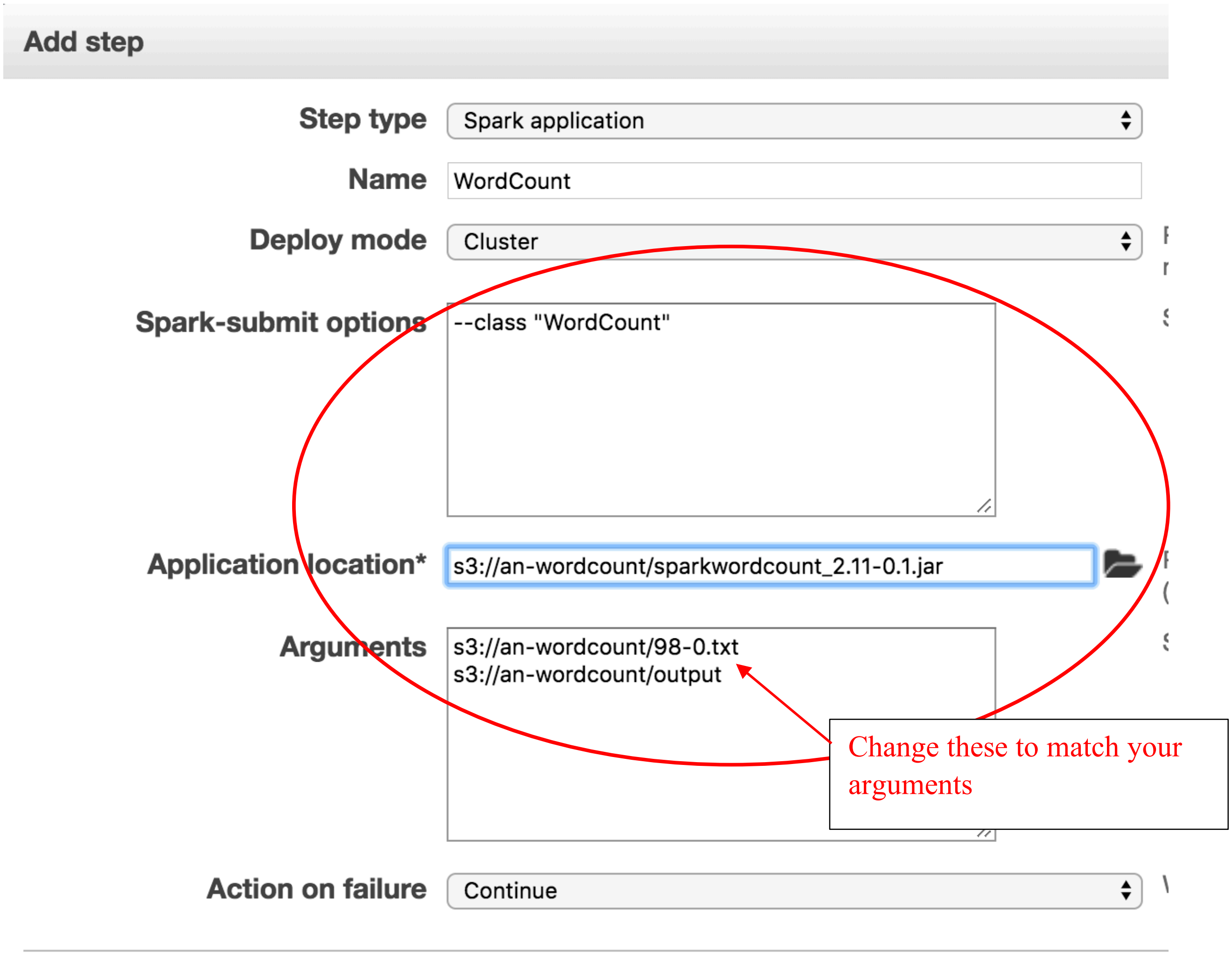
Cluster: My cluster Waiting Cluster ready after last step completed.

Connections: Hue, Spark History Server, Resource Manager … (View All)
Master public DNS: ec2-54-153-113-185.us-west-1.compute.amazonaws.com SSH
Tags: -- View All / Edit

Summary
ID: j-OWTAV4XDGSEF
Creation date: 2018-03-20 18:29 (UTC-5)

Configuration details
Release label: emr-5.12.0
Hadoop distribution: Amazon 2.8.3

Click on "Add Step" and then fill in the values as shown below:



Add step

Step type: Spark application

Name: WordCount

Deploy mode: Cluster

Spark-submit options:

```
--class "WordCount"
```

Application location*: s3://an-wordcount/sparkwordcount_2.11-0.1.jar

Arguments:

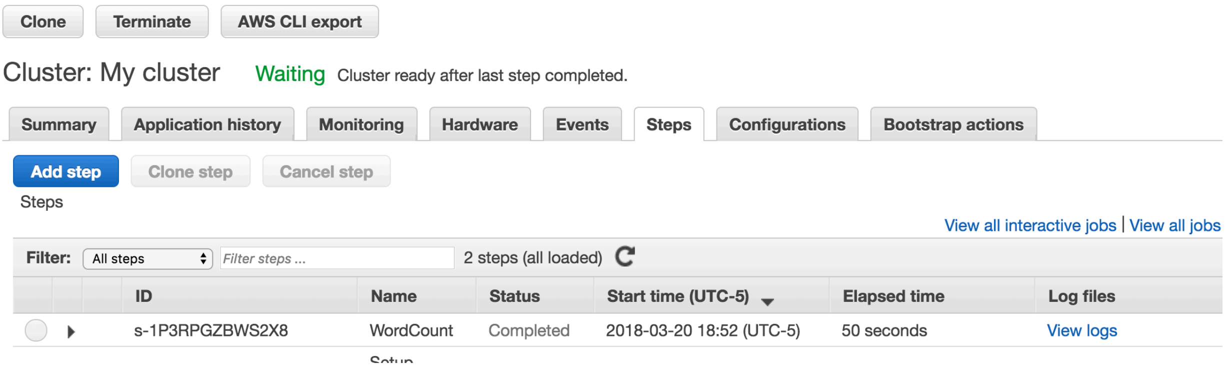
```
s3://an-wordcount/98-0.txt
s3://an-wordcount/output
```

Action on failure: Continue

Change these to match your arguments

Use your own bucket's path. My bucket name is "**an-wordcount**" -> change it to match your s3 path, and my file name was "**98-0.txt**" -> change it to match your file name.

Run this step and wait for it to be completed:

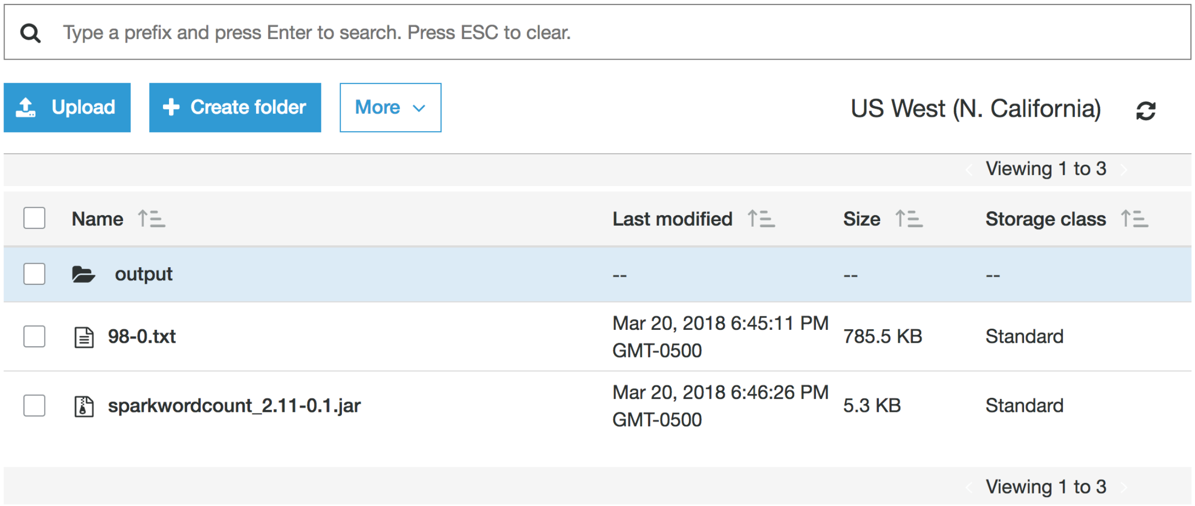


Cluster: My cluster Waiting Cluster ready after last step completed.

2 steps (all loaded)

ID	Name	Status	Start time (UTC-5)	Elapsed time	Log files
s-1P3RPGZBWS2X8	WordCount	Completed	2018-03-20 18:52 (UTC-5)	50 seconds	View logs
	Setup				

After completion, in your S3 bucket, you will have a directory called “output” (the second parameter), with the code output:


Type a prefix and press Enter to search. Press ESC to clear.
Upload
Create folder
More
US West (N. California)
Viewing 1 to 3

□	Name	Last modified	Size	Storage class
□	output	--	--	--
□	98-0.txt	Mar 20, 2018 6:45:11 PM GMT-0500	785.5 KB	Standard
□	sparkwordcount_2.11-0.1.jar	Mar 20, 2018 6:46:26 PM GMT-0500	5.3 KB	Standard

Please complete all the steps above. If you have questions, please post on Piazza. **Remember to terminate your cluster after you are done.**

Note: You are responsible for managing your own account and cluster. You should **terminate** your cluster after you are done. You should manage your account carefully so that you do not exceed the education credits provided by Amazon. The instructor is not responsible for any financial loss or charges that you may incur.