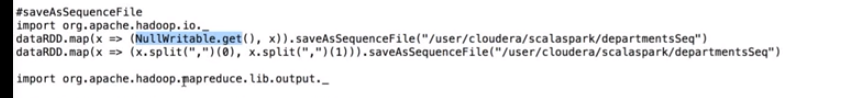
Sequence File: Binary Format where you generally give key and value.

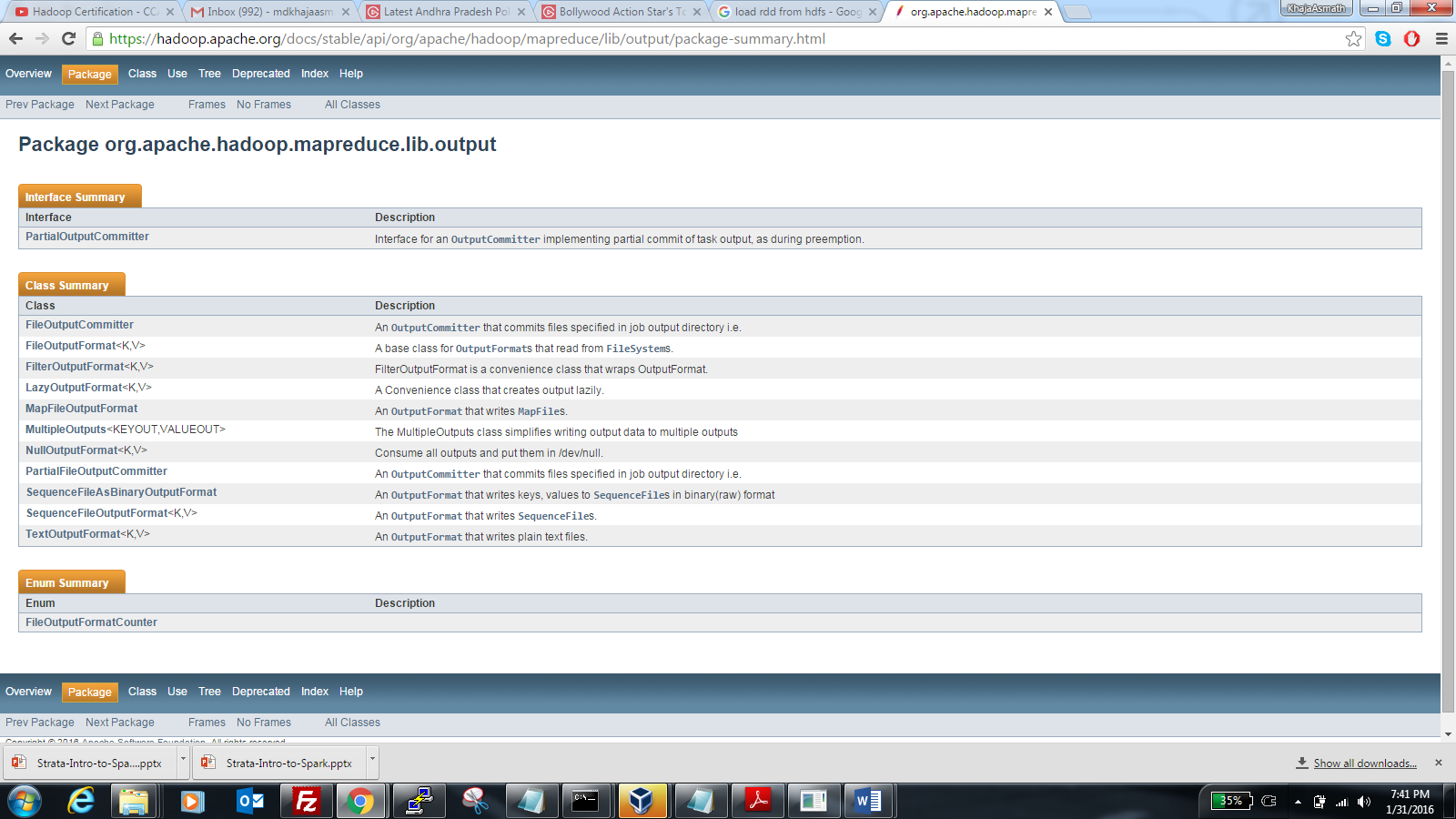
**dataRDD.map(x => (NullWritable.get(), x)).saveAsSequenceFile("/user/cloudera/scalaspark/departmentsSeq")**

**here you can see NullWritable is acting as key and entire line as value.**



\*\*\*\*\*\*Like sequence files, we can also create our own custom format to save and read. All these formats are present in the package **import org.apache.hadoop.mapreduce.lib.output.\_**

**See what this package has in google. MultiOut format for sparks. Custom names to the output file by LazyOutputFormat and MultiOut is to get different files.**



**// Save as sequence file is not present on RDD but present on other RDD. Lets google out.**

**Sequence file key and value so we need to have key and value before saving it as sequence file.**

dataRDD.map(x => (NullWritable.get(), x)).saveAsSequenceFile("/user/cloudera/scalaspark/departmentsSeq")

dataRDD.map(x => (x.split(",")(0), x.split(",")(1))).saveAsSequenceFile("/user/cloudera/scalaspark/departmentsSeq")

**Note we should have key and value for sequence file**

**Reading Sequence files**

**\*\*\* For reading sequence files, we need to give Type of class that we are reading for key and value. You will get error if you don’t give that type.**

#reading sequence file

sc.sequenceFile("/user/cloudera/spark/departmentsSeq", classOf[IntWritable], classOf[Text]).map(rec => rec.toString()).collect().foreach(println)

**CustomOutput Format:**

* **You can save the file in your own output format(i.e your own key and own value ) by extending the output package of Hadoop and implementing the call to saveAsNewAPIHadoopFile**
* **You have to import the import org.apache.hadoop.mapreduce.lib.output.\_ package also**
* **This can be done by using and calling saveAsNewAPIHadoopFile .**

**dataRDD.map(x => (new Text(x.split(",")(0)), new Text(x.split(",")(1)))).saveAsNewAPIHadoopFile(path, classOf[Text], classOf[Text], classOf[SequenceFileOutputFormat[Text, Text]])**

Here we are using Text as key and Text as value. Custom outputformat is saved as SequenceFile in this example. You can use any Hadoop format here only we are changing he keys and values based on the custom format.

You can also use textoutput with customoutput.