**Experiments with Default and Custom Partitioner**

Here, We will explore the default setting and Explore the Map and Reduce Framework.

Recap of Map- Reduce, Combine Function:: Map takes (key, Value ) as Input and Produces the list of Keys and Values.

Combine Function takes input as key , list of values and corresponding to keys and produces the list of key, Value..

And Exactly same case with Reduce Phase Also…

Map (K1,v1 ) -🡪 list (k2,v2)

Combiner (k2,List (v2)) -> list (k2,v2)

Reduce ((k2,List (v2)) -> list (k3,v3)

Hence While Implementation Combiners extents reducer Class.

( Combiner logic we write in Reduce Pgm, But It Runs on Map phase)

\*\* Another thing to be noted here is that for a Single input key pair the map or combine or reduce can emits multiple (key, Value ) pair ..

Now, Gets introduce New Function Partitioner

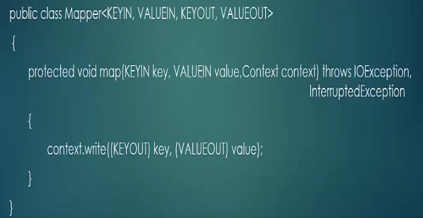
\*\*Partition takes Key, Value pair as a Input and Produces integer as a Result…

Partition (k2, v2) 🡪 Integer

And This Integer will be used to decide which reducer (Key, Value ) pair would go to…

Even IN Case, We can Write Custom Partitioner also….

Let’s see with an Example Program::



public class DefaultDriver extends Configured implements Tool {

@Override

public int run(String[] args) throws Exception {

Job job = new Job(getConf());

job.setJarByClass(getClass());

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

return job.waitForCompletion(true) ? 0 : 1;

}

public static void main(String[] args) throws Exception {

int exitCode = ToolRunner.run(new DefaultDriver(), args);

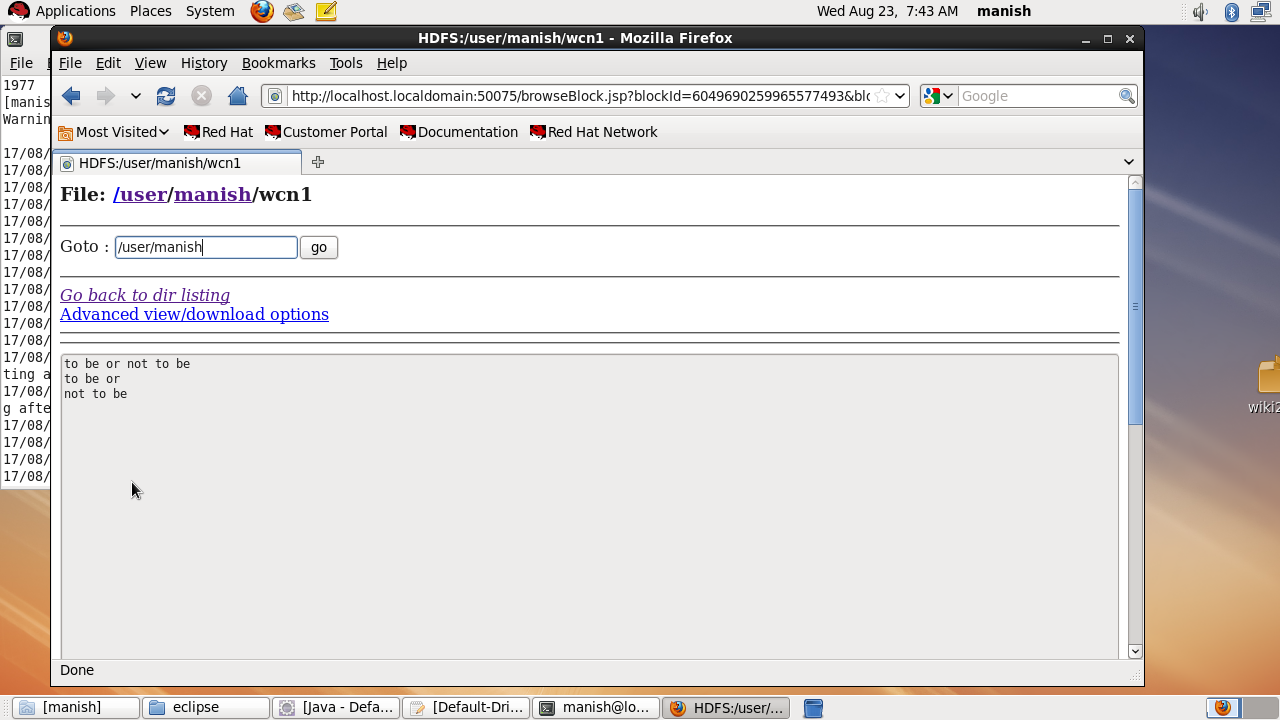
System.exit(exitCode);

}

}

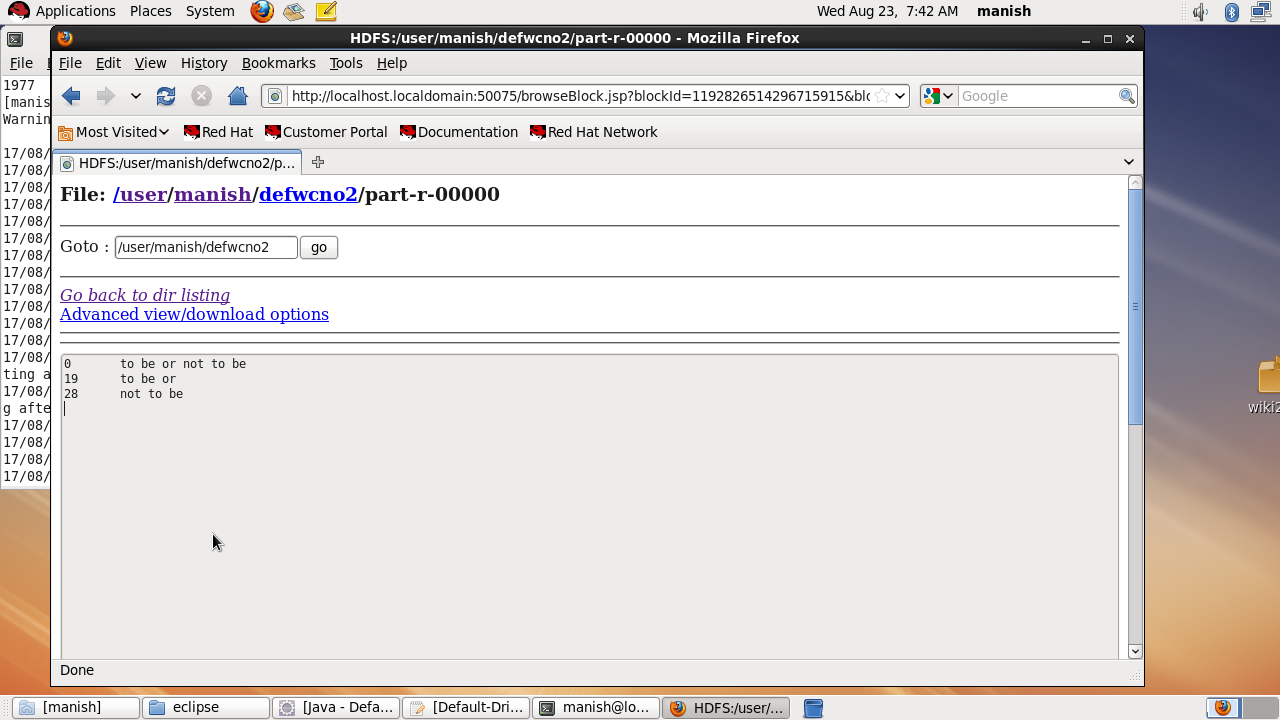
* As We have seen that There is no job is configured in this class , and we havn’t specify map class, reduce class nor combine class..
* Neither we have specified input data types nor output Data types…
* Lets Run this Code and see the Output …
* Input File:::: to be or not to be

That is the question

Input:: 

Run the code::

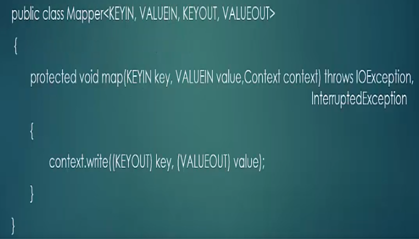
OutPut::



This code will convert the input into Byteoffset, Line as Key, Value Pair…

This is the Default run of the mapper or Reducer..

Now , Let’s Understand How Default Mapper and Reducer Look Like ???



\*\* The mapper Class as we know has (key, Value as a input and Key, Value as a Output…

\*\* Line 1 – Where we put Data Types

\*\* Line 2 -- this map function otherwise we overwrite. As we already know that has 3 parameters , Key, Value, Context…

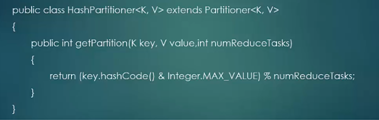
Line -3 :: in the processing it just emits the key, value pairs which it receives .

The Default Key Input data types is long Writable. ( Because it can handle a large no )

Default Input Values are Text

Default Output key and Values are same as INPUT…(i.e. Longwritable, Text )

* After map has produced its output the partitioner has its responsibility for dividing the result and distributing the values among various Reducers..
* By Default there is no Combiner class.
* Default Partioner is HASH partioner



* Get Partion Function takes Key, Value and No. of Reducers as Input and It Produces the Hash Code of the key, and Performs and ADD operation with the Max Value and Modulo It to which reducer it should go..

Ex: Hi , 1 ,1,1,1,1,1 = 6 modulo no of reducer

Hello 1,2,2,2,22, =13 modulo no of reducer

* Lets Suppose We have 3 Reducers the results of all the keys will come to 0,1,2.
* \*\* Depending upon the result Key , Value pair will be sent to particular Reducers…
* It is to be noted that only KEY is considered on which reducer the key, Value pair should go..

\*\*\* It may be possible that We want some certain key to be processed by a Certain REDUCER ONLY

Lets Take an Example on this :::

Let us assume , We have a File with Customer Fname, Lname, and Colors they Like )

Manish , Singh , Blue, Red, Pink

Vishakha , Joshi , Black, Blue, Pink

Anshul , Agarwal, maroon, Blue,

Mary, Ficher, Black, Cyan

James, Ficher, Red, White

Mary, Ficher, Cyan, Sky

David , Dumbler, Red, Blue

\*\* Suppose We are Looking to search for a Color Pattern, If there is any correlation between Names and Colors of their Choice…

So, According to our Query or file We may decide to take

Key as the second Name in the Map Function…..

This will help us to group the colors with the help of second name..

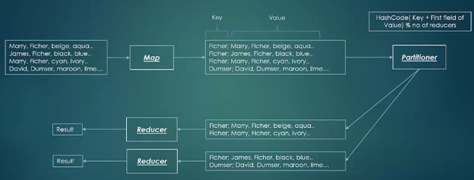
So, Map output will be : Ficher : Mary , colors

This fits to Partitioners:::

\*\*\* If we want to send the people with the same first name should go to same reducer…

In this Case, We add first name as well to the hash Partitioners to calculate the hash Code.

As Ficher James has been sent to different reducer eventhough its key value is same as Ficher marry..



This is because of custom Partioners defined..