**Different Methods of mapper Class**

Mapper class comprises of 4 methods and they are:

*1.Setup*

*2.Map*

*3.run*

*4.Cleanup*

***1.)SETUP method***

* It is known as at the start of the task
* And it is used for:
* For all mapper class it is used for initializing the common values.
* Eg- when MapSide is used as join
* Then setup method will be used in order to add files from the distributed cache to all the nodes present.
* Setup method is used to run the mapper.

*Example*

**@Override**

protected void setup(Context context) throws IOException,InterruptedException {

Path[] cacheFilesLocal = DistributedCache.getLocalCacheFiles(context.getConfiguration());

for (Path eachPath : cacheFilesLocal) {  
if (eachPath.getName().toString().trim().equals(“departments\_txt”)) {  
context.getCounter(MYCOUNTER.FILE\_EXISTS).increment(1);  
loadDepartmentsHashMap(eachPath, context);  
}

***2.Map Method***

* Map method will be called as “one key value pair” in input split.
* Line start byte offset it taken as key by the record reader in default
* The lines will be taken as value map method for each line in the file.

For Example

public void map(Object key, Text value, Context context) throws IOException, InterruptedException {

StringTokenizer itr = new StringTokenizer(value.toString());

while (itr.hasMoreTokens()) {

word.set(itr.nextToken());

context.write(word, one);

}

}

}

***3.Run Method***

* For a complete control can be handled over this method by experts using override command while executing the mapper.
* MR framework will call the run() which helps to call setup(), map() for n number of times and will cleanup()finally.

*cleanup method*

* It is also called as “once at the end of the task”.
* Will be used for writing the output of map on context once all the mapper gets over.

***2. Different Methods Of Reducer Class***

1.Setup

2.Reduce

3.run

4.Cleanup

**1.SETUP method**

* It is also called as “once at the start of the task”.
* And can be used for setting certain common values for all reducers.

**2.Reduce Method**

* This method is also called as “once for each unique key”.

**For Example:**

public static class ReduceForWordCount extends Reducer<Text, IntWritable, Text, IntWritable>

{

public void reduce(Text word, Iterable<IntWritable> values, Context con) throws IOException, InterruptedException

{

int sum = 0;

for(IntWritable value : values)

{

sum += value.get();

}

con.write(word, new IntWritable(sum));

}

**3.)Run method**

* It can be used only by advanced pplication writers
* [run(org.apache.hadoop.mapreduce.Reducer.Context)](https://hadoop.apache.org/docs/r2.6.2/api/org/apache/hadoop/mapreduce/Reducer.html#run(org.apache.hadoop.mapreduce.Reducer.Context)) is a method used to control the reduce task works.

**4.)Cleaup Method**

It is called as “once at the end of the Task”.

It is mainly used for finally to write the context for each key like reduce.

**Example:**

Protected void cleanup(Context context) throws IOException, InterruptedException {

Map sortedMap = sortByValues(countMap);

int counter = 0;

for (Text key: sortedMap.keySet()) {

if (counter ++ == 20) {

break;

}

context.write(key, sortedMap.get(key));

}