Introduction to Big Data

ECAP456

Dr. Rajni Bhalla

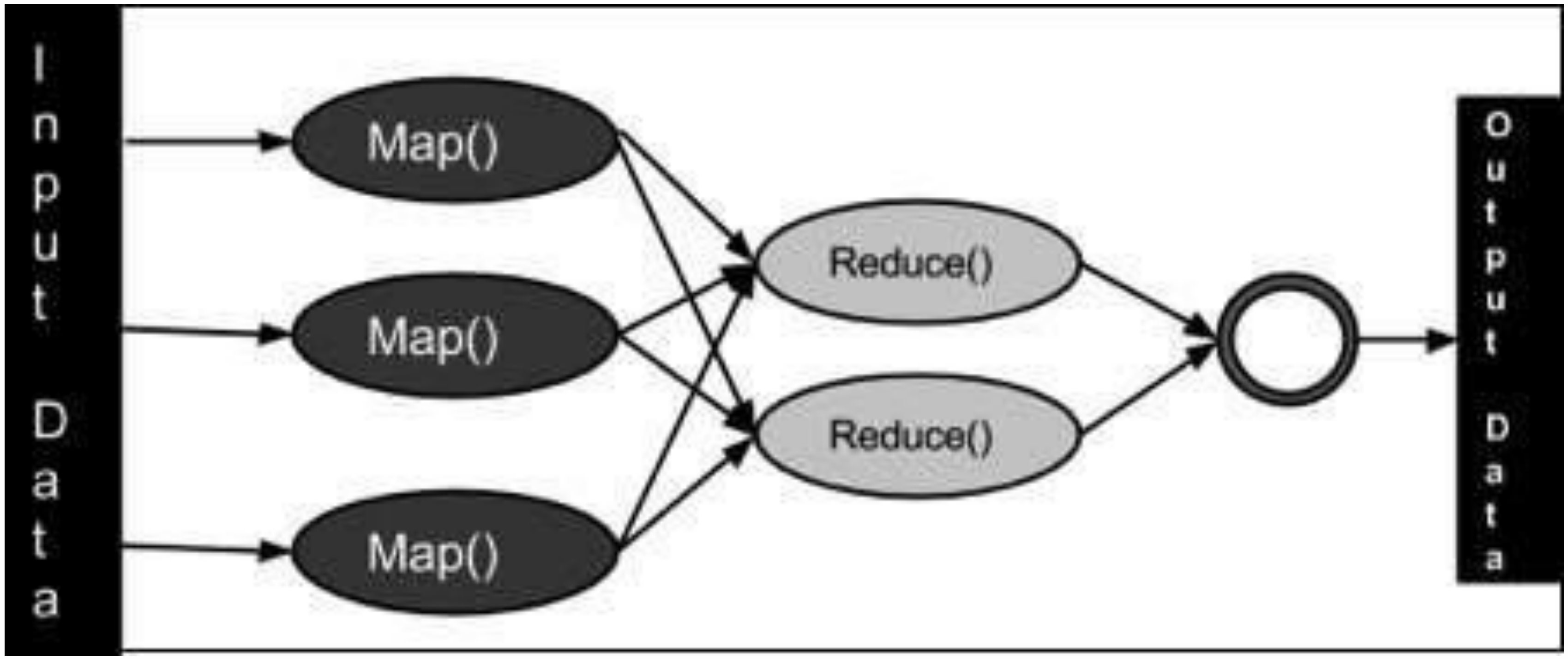
Associate Professor

Learning Outcomes

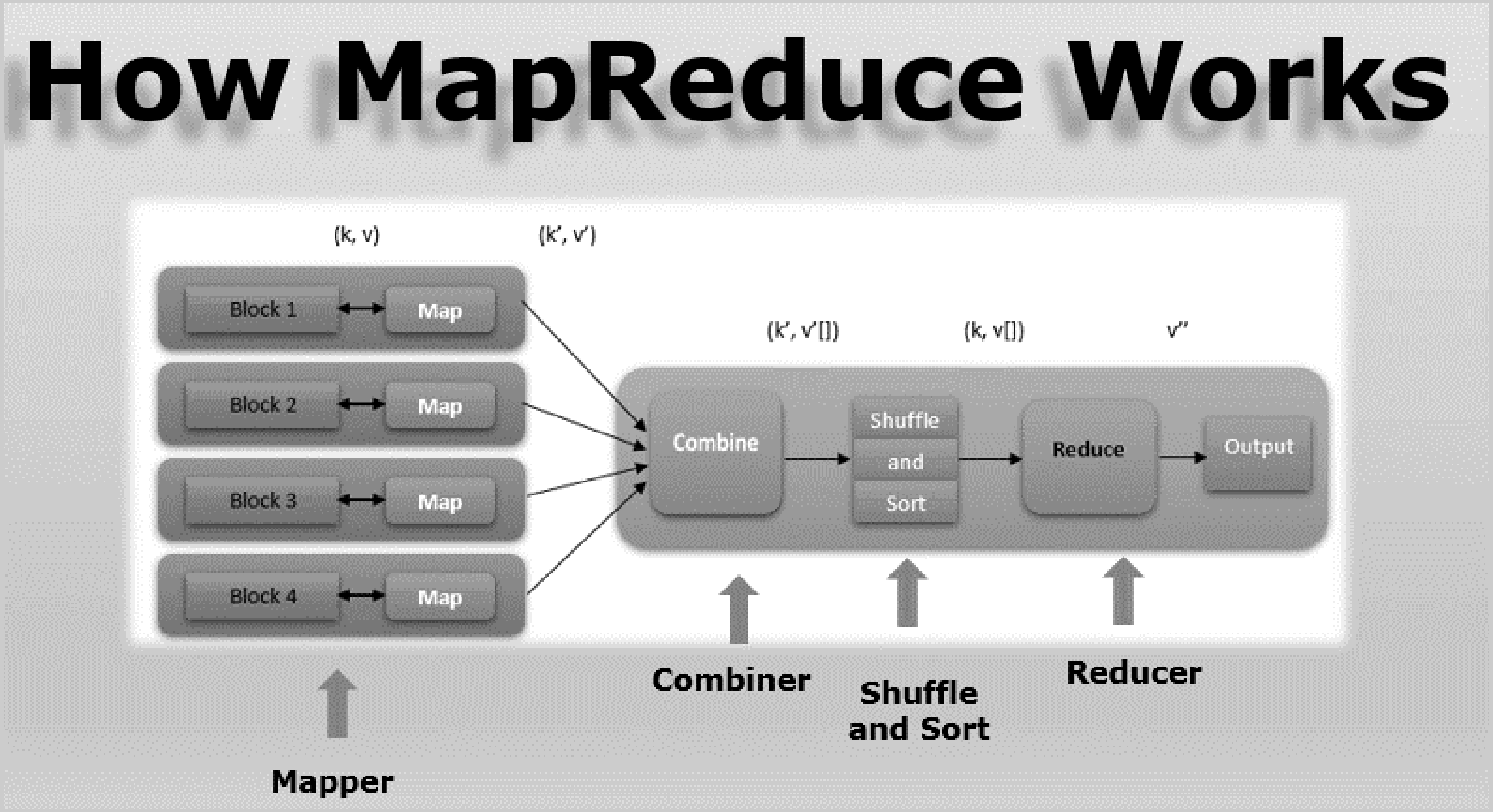
After this lecture, you will be able to

- learn anatomy of a MapReduce Job Run

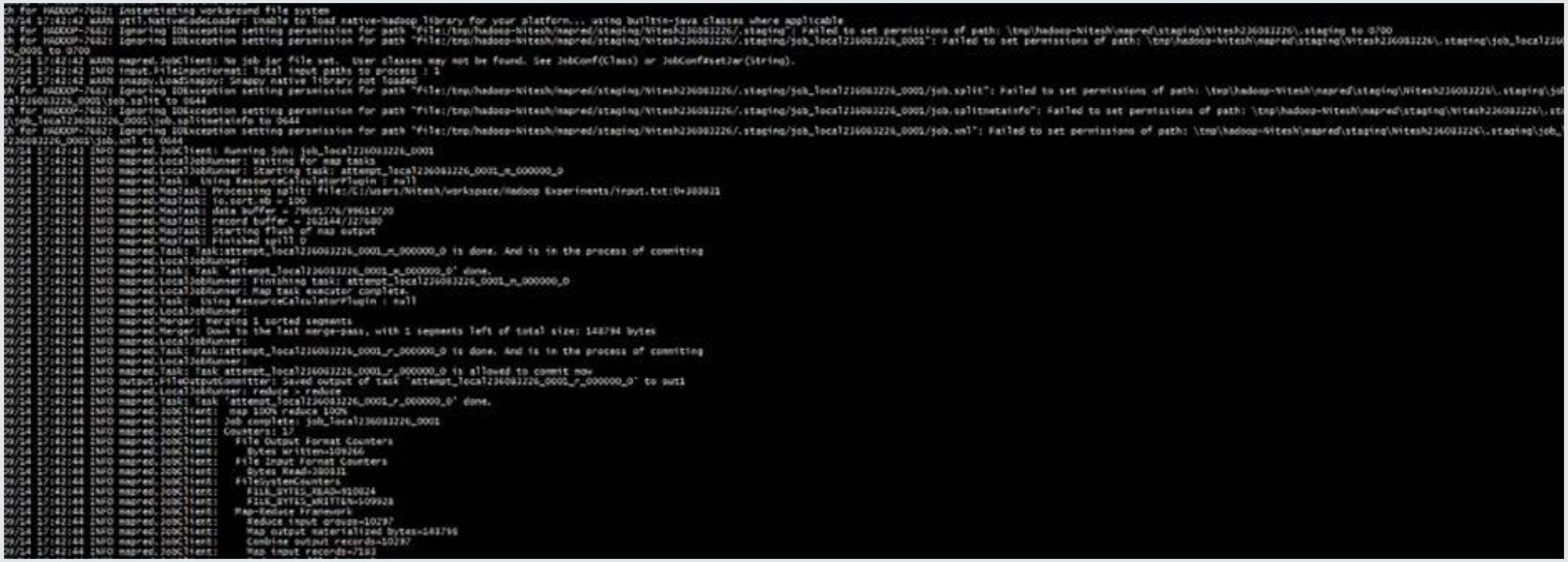
Introduction



Introduction



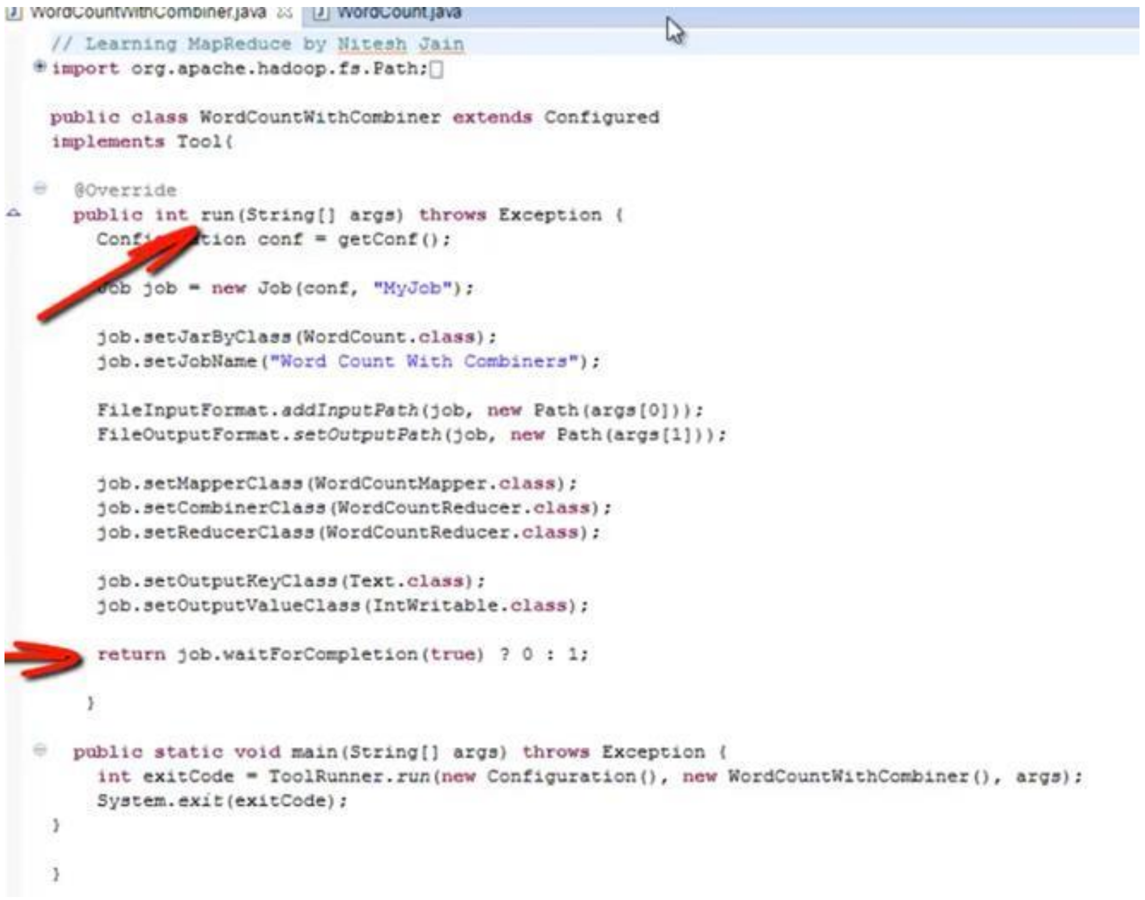
Introduction



```
ch for HADOOP-7682: Instantiating workaround file system
09/14 17:42:42 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
ch for HADOOP-7682: Ignoring IOException setting persmission for path "file:/tmp/hadoop-Nitesh/mapred/staging/Nitesh236083226/.staging": Failed to set permissions of path: \tmp\hadoop-Nitesh\mapred\staging\Nitesh236083226\.staging to 0700
ch for HADOOP-7682: Ignoring IOException setting persmission for path "file:/tmp/hadoop-Nitesh/mapred/staging/Nitesh236083226/.staging/job_local236083226_0001": Failed to set permissions of path: \tmp\hadoop-Nitesh\mapred\staging\Nitesh236083226\.staging\job_local236
26_0001 to 0700
09/14 17:42:42 WARN mapred.JobClient: No job jar file set.  User classes may not be found. See JobConf(Class) or JobConf#setJar(String).
09/14 17:42:42 INFO input.FileInputFormat: Total input paths to process : 1
09/14 17:42:42 WARN snappy.LoadSnappy: Snappy native library not loaded
ch for HADOOP-7682: Ignoring IOException setting persmission for path "file:/tmp/hadoop-Nitesh/mapred/staging/Nitesh236083226/.staging/job_local236083226_0001/job.split": Failed to set permissions of path: \tmp\hadoop-Nitesh\mapred\staging\Nitesh236083226\.staging\job
cal236083226_0001\job.split to 0644
ch for HADOOP-7682: Ignoring IOException setting persmission for path "file:/tmp/hadoop-Nitesh/mapred/staging/Nitesh236083226/.staging/job_local236083226_0001/job.splitmetainfo": Failed to set permissions of path: \tmp\hadoop-Nitesh\mapred\staging\Nitesh236083226\.sta
g\job_local236083226_0001\job.splitmetainfo to 0644
ch for HADOOP-7682: Ignoring IOException setting persmission for path "file:/tmp/hadoop-Nitesh/mapred/staging/Nitesh236083226/.staging/job_local236083226_0001/job.xml": Failed to set permissions of path: \tmp\hadoop-Nitesh\mapred\staging\Nitesh236083226\.staging\job_
236083226_0001\job.xml to 0644
09/14 17:42:43 INFO mapred.JobClient: Running job: job_local236083226_0001
09/14 17:42:43 INFO mapred.LocalJobRunner: Waiting for map tasks
09/14 17:42:43 INFO mapred.LocalJobRunner: Starting task: attempt_local236083226_0001_m_000000_0
09/14 17:42:43 INFO mapred.Task:  Using ResourceCalculatorPlugin : null
09/14 17:42:43 INFO mapred.MapTask: Processing split: file:/C:/users/Nitesh/workspace/Hadoop Experinents/input.txt:0+380831
09/14 17:42:43 INFO mapred.MapTask: io.sort.mb = 100
09/14 17:42:43 INFO mapred.MapTask: data buffer = 79691776/99614720
09/14 17:42:43 INFO mapred.MapTask: record buffer = 262144/327680
09/14 17:42:43 INFO mapred.MapTask: Starting flush of map output
09/14 17:42:43 INFO mapred.MapTask: Finished spill 0
09/14 17:42:43 INFO mapred.Task: Task:attempt_local236083226_0001_m_000000_0 is done. And is in the process of commiting
09/14 17:42:43 INFO mapred.LocalJobRunner:
09/14 17:42:43 INFO mapred.Task: Task 'attempt_local236083226_0001_m_000000_0' done.
09/14 17:42:43 INFO mapred.LocalJobRunner: Finishing task: attempt_local236083226_0001_m_000000_0
09/14 17:42:43 INFO mapred.LocalJobRunner: Map task executor complete.
09/14 17:42:43 INFO mapred.Task:  Using ResourceCalculatorPlugin : null
09/14 17:42:43 INFO mapred.LocalJobRunner:
09/14 17:42:43 INFO mapred.Merger: Merging 1 sorted segments
09/14 17:42:44 INFO mapred.Merger: Down to the last merge-pass, with 1 segments left of total size: 148794 bytes
09/14 17:42:44 INFO mapred.LocalJobRunner:
09/14 17:42:44 INFO mapred.Task: Task:attempt_local236083226_0001_r_000000_0 is done. And is in the process of commiting
09/14 17:42:44 INFO mapred.LocalJobRunner:
09/14 17:42:44 INFO mapred.Task: Task attempt_local236083226_0001_r_000000_0 is allowed to commit now
09/14 17:42:44 INFO output.FileOutputCommitter: Saved output of task 'attempt_local236083226_0001_r_000000_0' to out1
09/14 17:42:44 INFO mapred.LocalJobRunner: reduce > reduce
09/14 17:42:44 INFO mapred.Task: Task 'attempt_local236083226_0001_r_000000_0' done.
09/14 17:42:44 INFO mapred.JobClient:  map 100% reduce 100%
09/14 17:42:44 INFO mapred.JobClient: Job complete: job_local236083226_0001
09/14 17:42:44 INFO mapred.JobClient: Counters: 17
09/14 17:42:44 INFO mapred.JobClient:   File Output Format Counters
09/14 17:42:44 INFO mapred.JobClient:     Bytes Written=109266
09/14 17:42:44 INFO mapred.JobClient:   File Input Format Counters
09/14 17:42:44 INFO mapred.JobClient:     Bytes Read=380831
09/14 17:42:44 INFO mapred.JobClient:   FileSystemCounters
09/14 17:42:44 INFO mapred.JobClient:     FILE_BYTES_READ=910024
09/14 17:42:44 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=509928
09/14 17:42:44 INFO mapred.JobClient:   Map-Reduce Framework
09/14 17:42:44 INFO mapred.JobClient:     Reduce input groups=10297
09/14 17:42:44 INFO mapred.JobClient:     Map output materialized bytes=148798
09/14 17:42:44 INFO mapred.JobClient:     Combine output records=10297
09/14 17:42:44 INFO mapred.JobClient:     Map input records=7193
```

Job Execution and job Completion

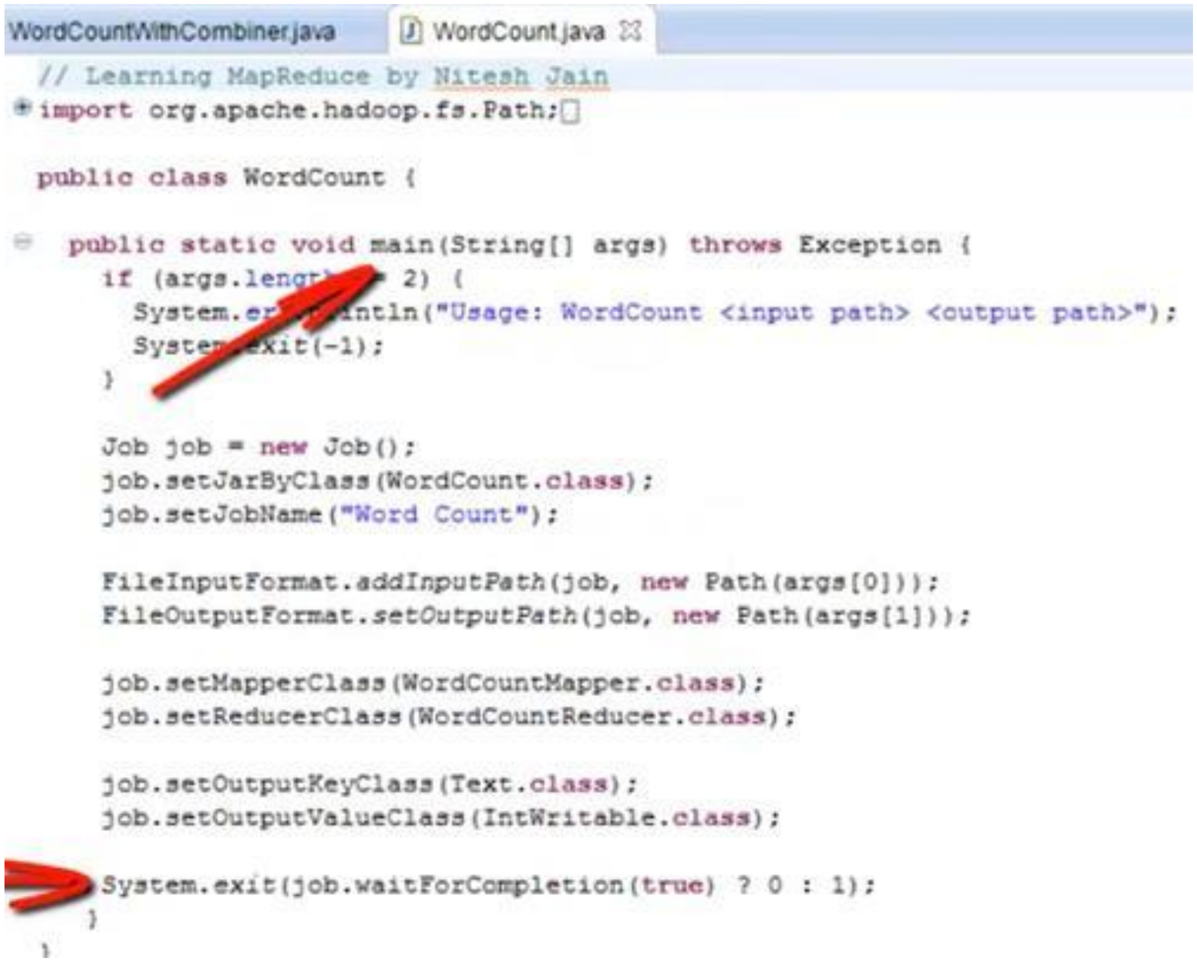
Introduction



```
// Learning MapReduce by Nitesh Jain
import org.apache.hadoop.fs.Path;

public class WordCountWithCombiner extends Configured
implements Tool{

  @Override
  public int run(String[] args) throws Exception {
    Configuration conf = getConf();

    Job job = new Job(conf, "MyJob");

    job.setJarByClass(WordCount.class);
    job.setJobName("Word Count With Combiners");

    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));

    job.setMapperClass(WordCountMapper.class);
    job.setCombinerClass(WordCountReducer.class);
    job.setReducerClass(WordCountReducer.class);

    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);

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

   }

  public static void main(String[] args) throws Exception {
    int exitCode = ToolRunner.run(new Configuration(), new WordCountWithCombiner(), args);
    System.exit(exitCode);
 }

 }
```

Last statement in run method

Introduction



```
// Learning MapReduce by Nitesh Jain
import org.apache.hadoop.fs.Path;

public class WordCount {

  public static void main(String[] args) throws Exception {
    if (args.length != 2) {
      System.err.println("Usage: WordCount <input path> <output path>");
      System.exit(-1);
    }

    Job job = new Job();
    job.setJarByClass(WordCount.class);
    job.setJobName("Word Count");

    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));

    job.setMapperClass(WordCountMapper.class);
    job.setReducerClass(WordCountReducer.class);

    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);

    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

Last line in main method

What happened behind the scene?

- Jobs gets submitted by WaitforCompletion().
- Properties that decide the job run:
  - “mapred.job.tracker” (for 0.22 release)
    - This is set in mapred-site.xml
    - Default is local.
    - Set as host:part
  - “mapreduce.framework.name” (for .23 and later(eg 1.X.X)):
    - Can be set to local,classic or yarn

What happened behind the scene?

- Jobs gets submitted by WaitforCompletion().
- Properties that decide the job run:
  - “mapred.job.tracker” (for 0.22 release)
    - This is set in mapred-site.xml
    - Default is local.
    - Set as host:part
  - “mapreduce.framework.name” (for .23 and later(eg 1.X.X)):
    - Can be set to local,classic or yarn

What happened behind the scene?

- Jobs gets submitted by WaitforCompletion().
- Properties that decide the job run:
    - "mapred.job.tracker" (for 0.22 release)
        - This is set in mapred-site.xml
        - Default is local.
        - Set as host:part
    - "mapreduce.framework.name" (for .23 and later(eg 1.X.X)):
        - Can be set to local,classic or yarn

What happened behind the scene?

- Jobs gets submitted by WaitforCompletion().
- Properties that decide the job run:
  - “mapred.job.tracker” (for 0.22 release)
    - This is set in mapred-site.xml
    - Default is local.
    - Set as host:part
  - “mapreduce.framework.name” (for .23 and later(eg 1.X.X)):
    - Can be set to local,classic or yarn

What happened behind the scene?

- Jobs gets submitted by WaitforCompletion().
- Properties that decide the job run:
  - “mapred.job.tracker” (for 0.22 release)
    - This is set in mapred-site.xml
    - Default is local.
    - Set as host:part
  - “mapreduce.framework.name” (for .23 and later(eg 1.X.X)):
    - Can be set to local,classic or yarn

What happened behind the scene?

- Jobs gets submitted by WaitforCompletion().
- Properties that decide the job run:
  - "mapred.job.tracker" (for 0.22 release)
    - This is set in mapred-site.xml
    - Default is local.
    - Set as host:part
  - "mapreduce.framework.name" (for .23 and later(eg 1.X.X)):
    - Can be set to local,classic or yarn

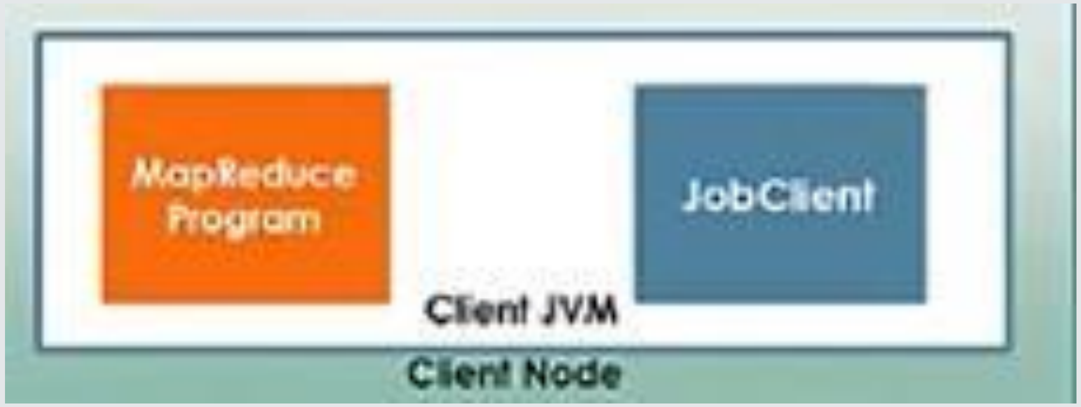
What happened behind the scene?

- Jobs gets submitted by WaitforCompletion().
- Properties that decide the job run:
    - “mapred.job.tracker” (for 0.22 release)
        - This is set in mapred-site.xml
        - Default is local.
        - Set as host:part
    - “mapreduce.framework.name” (for .23 and later(eg 1.X.X)):
        - Can be set to local,classic or yarn

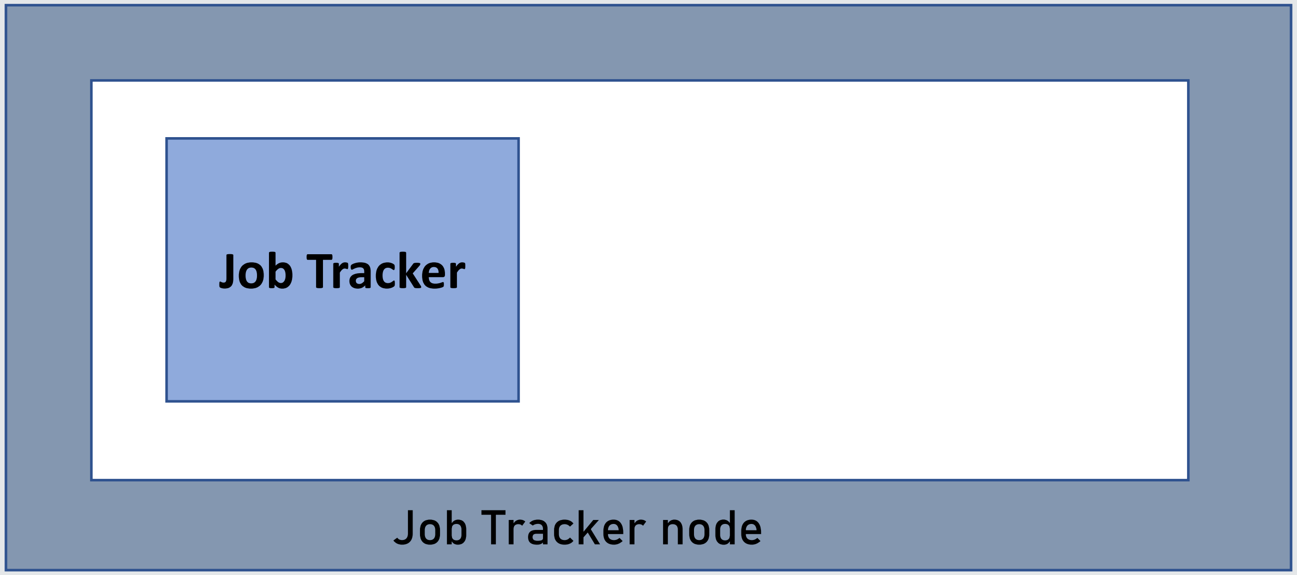
What happened behind the scene?

- Jobs gets submitted by WaitforCompletion().
- Properties that decide the job run:
  - “mapred.job.tracker” (for 0.22 release)
    - This is set in mapred-site.xml
    - Default is local.
    - Set as host:part
  - “mapreduce.framework.name” (for .23 and later(eg 1.X.X)):
    - Can be set to local,classic or yarn

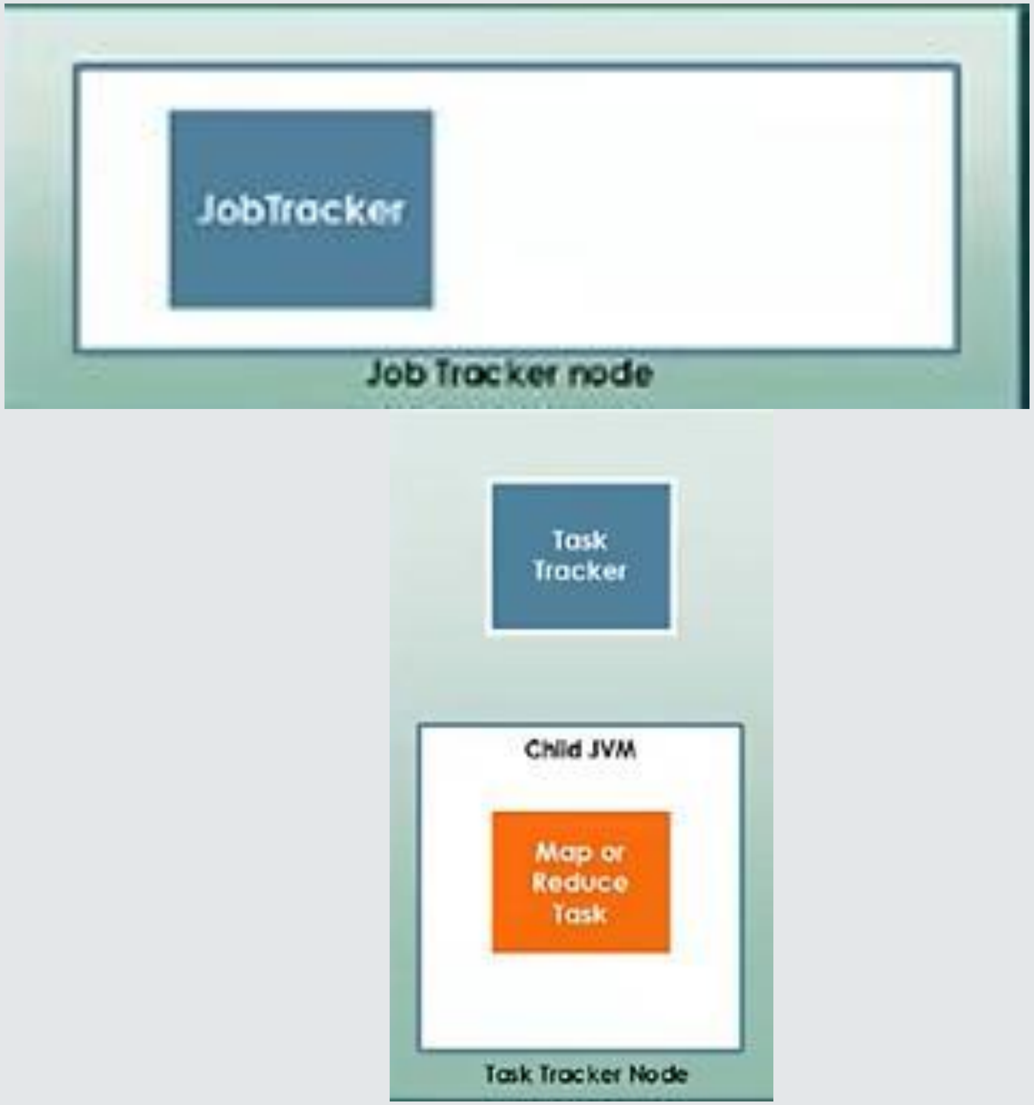
Classic MapReduce – Overview



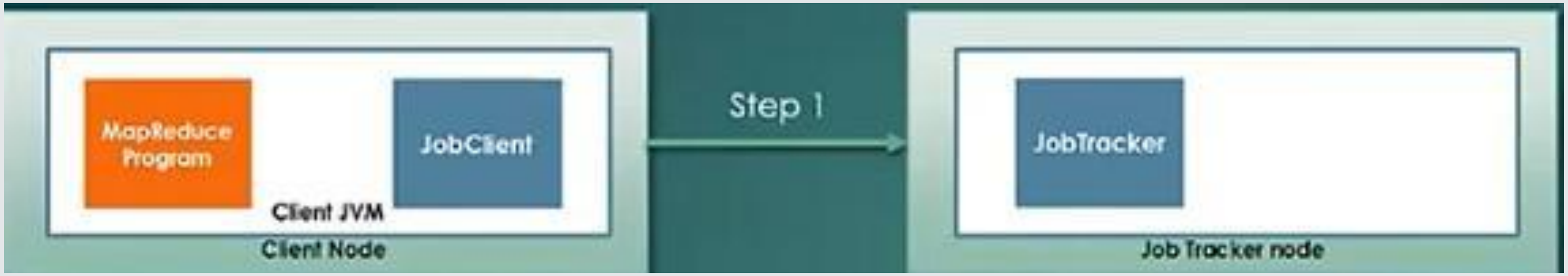
Classic MapReduce – Overview



Classic MapReduce - Overview

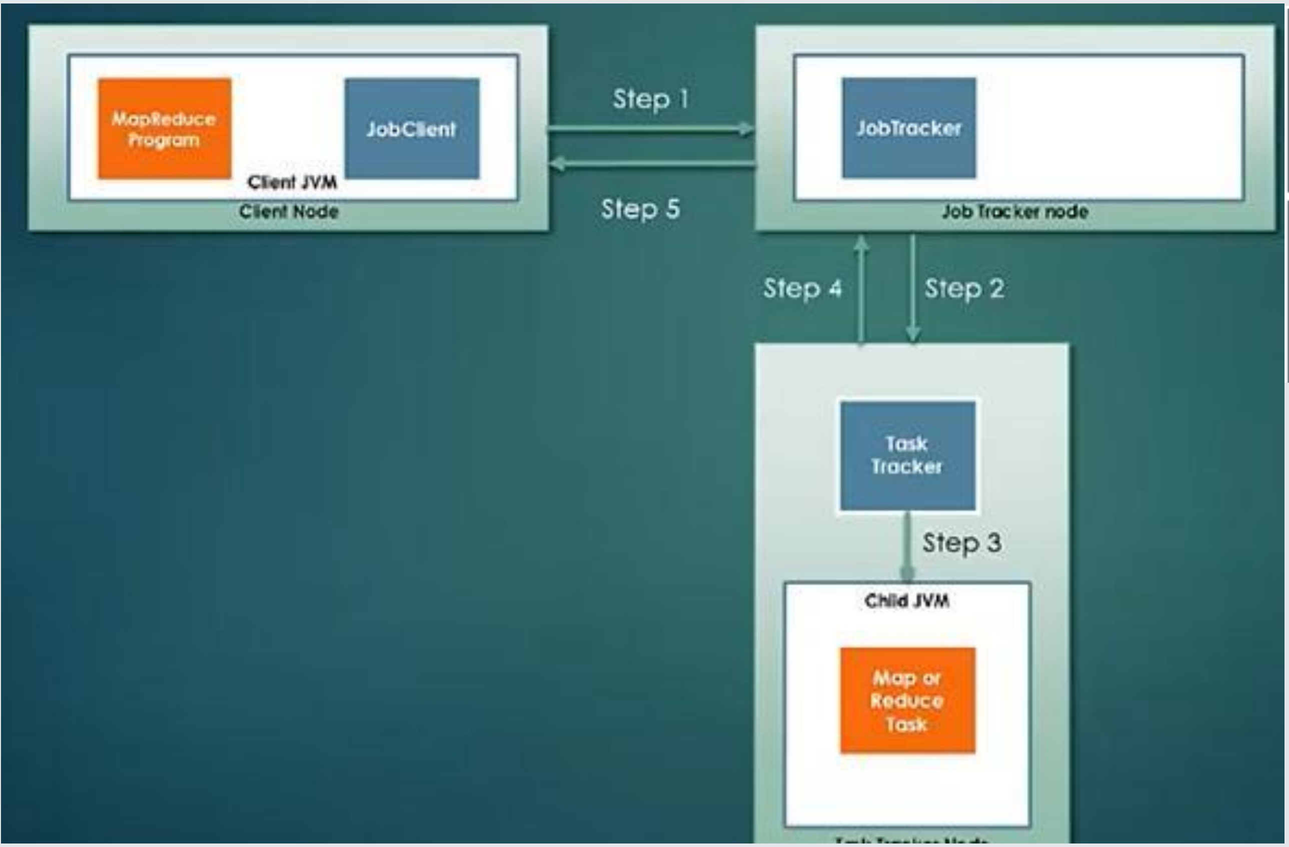


Classic MapReduce - Overview



1. Client submits the MapReduce job.

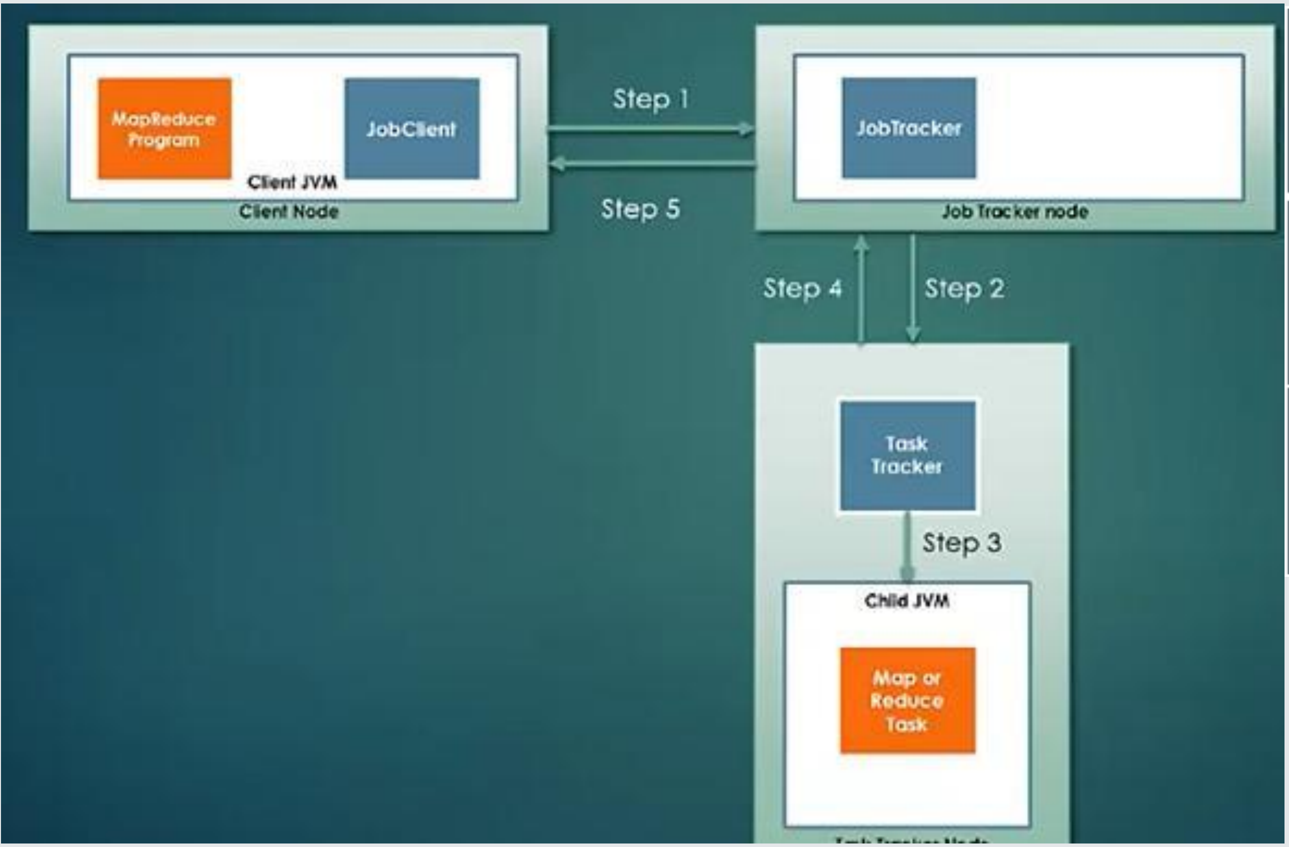
Classic MapReduce - Overview



1. Client submits the MapReduce job.

2. Job tracker co-ordinate the job run

Classic MapReduce – Overview

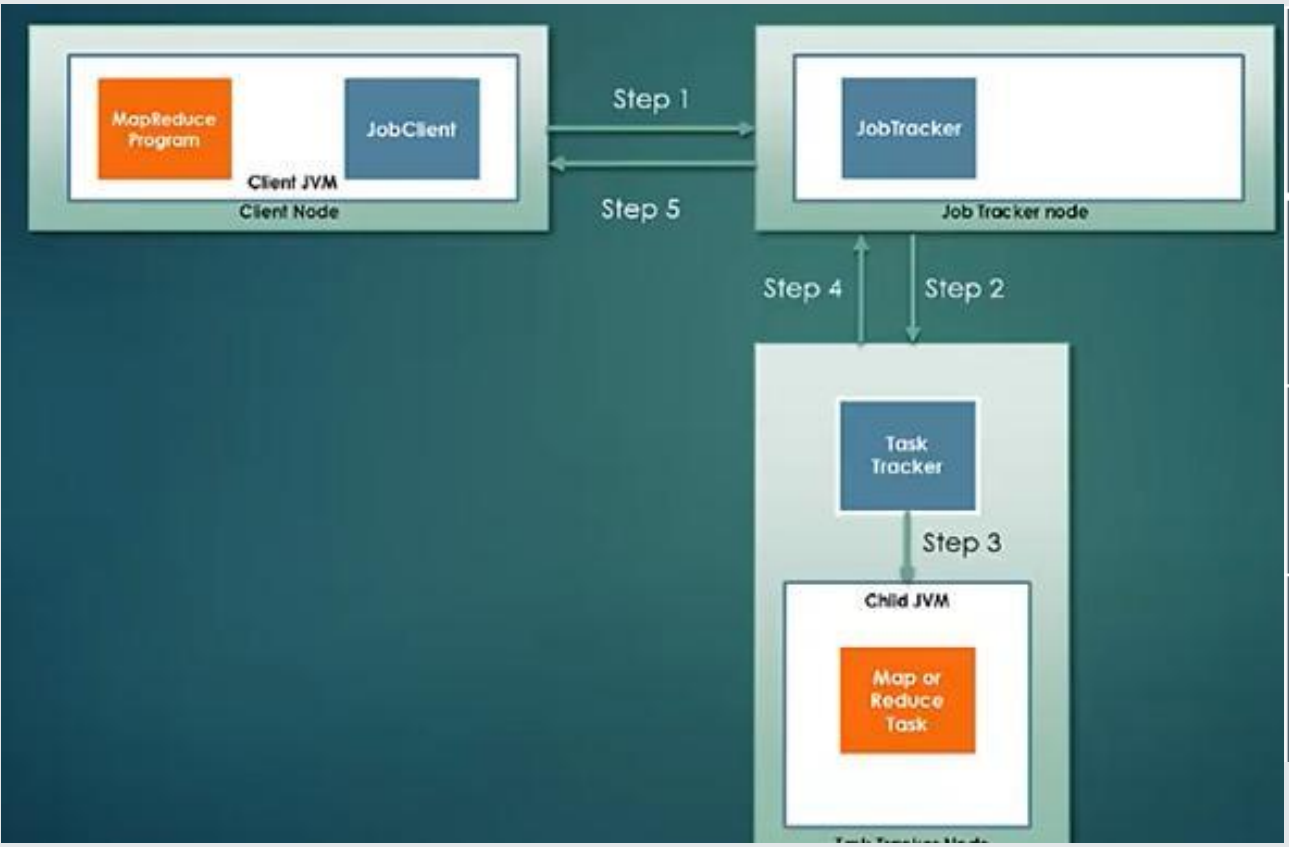


1. Client submits the MapReduce job.

2. Job tracker co-ordinate the job run

3. Task tracker run the map and reduce tasks.

Classic MapReduce - Overview



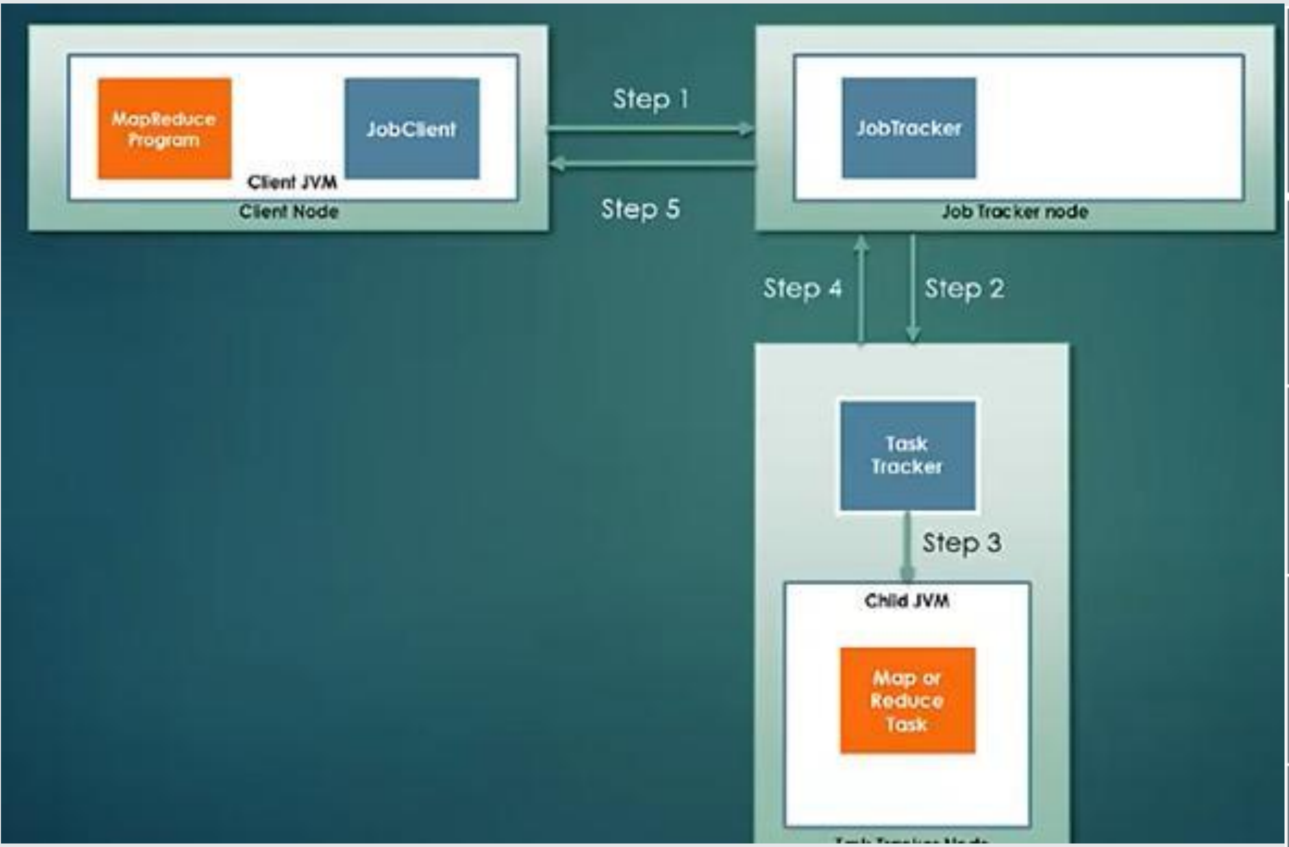
1. **Client submits the MapReduce job.**

2. Job tracker co-ordinate the job run

3. Task tracker run the map and reduce tasks.

4. Task tracker sends progress status

Classic MapReduce - Overview



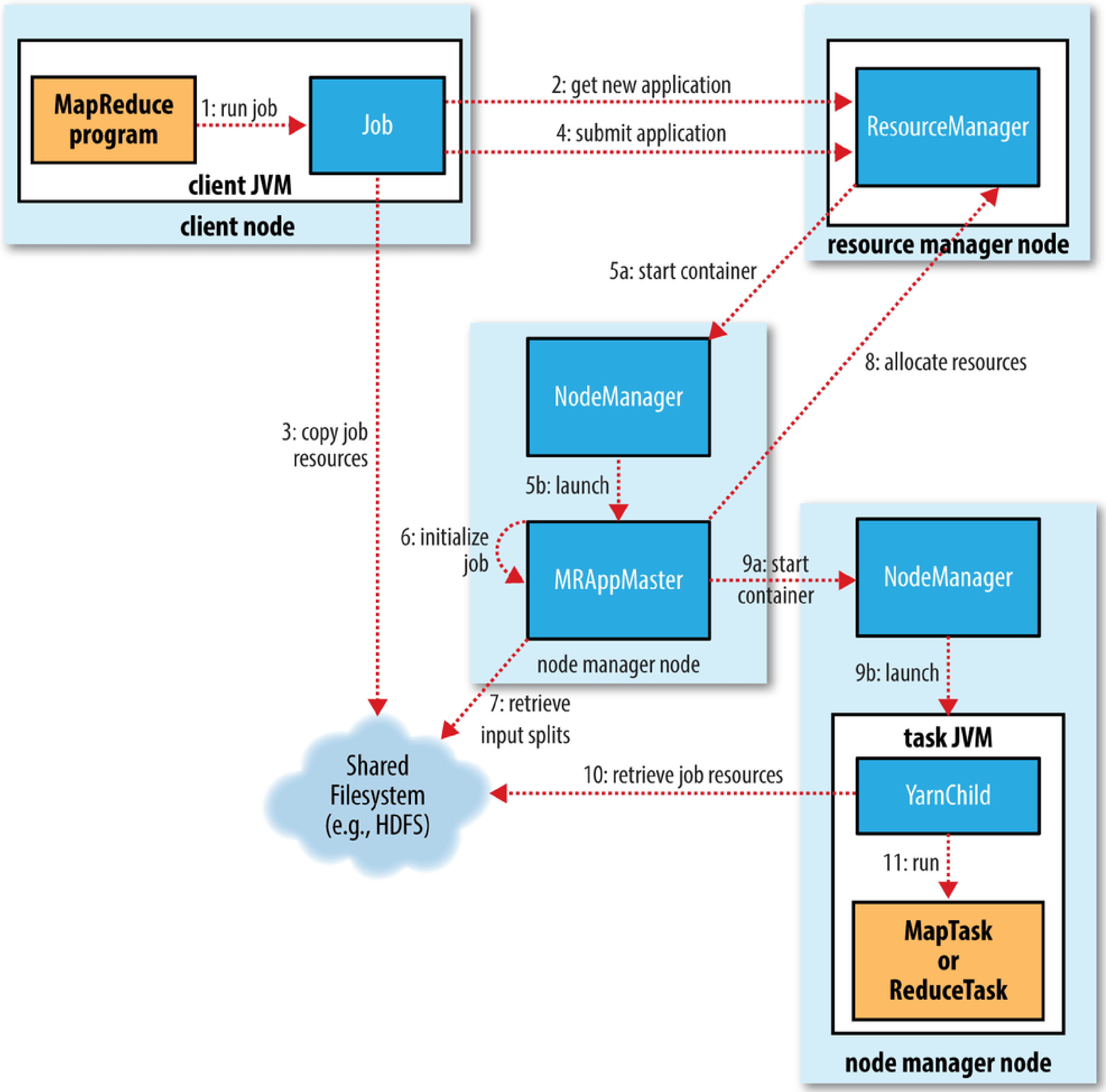
1. Client submits the MapReduce job.

2. Job tracker co-ordinate the job run

3. Task tracker run the map and reduce tasks.

4. Task tracker sends progress status

5. Job Tracker sends progress status and completion message to client.


MapReduce program
1: run job
Job
client JVM
client node
2: get new application
4: submit application
ResourceManager
resource manager node
5a: start container
8: allocate resources
3: copy job resources
NodeManager
5b: launch
6: initialize job
MRAppMaster
node manager node
9a: start container
NodeManager
9b: launch
7: retrieve input splits
task JVM
Shared Filesystem (e.g., HDFS)
10: retrieve job resources
YarnChild
11: run
MapTask or ReduceTask
node manager node

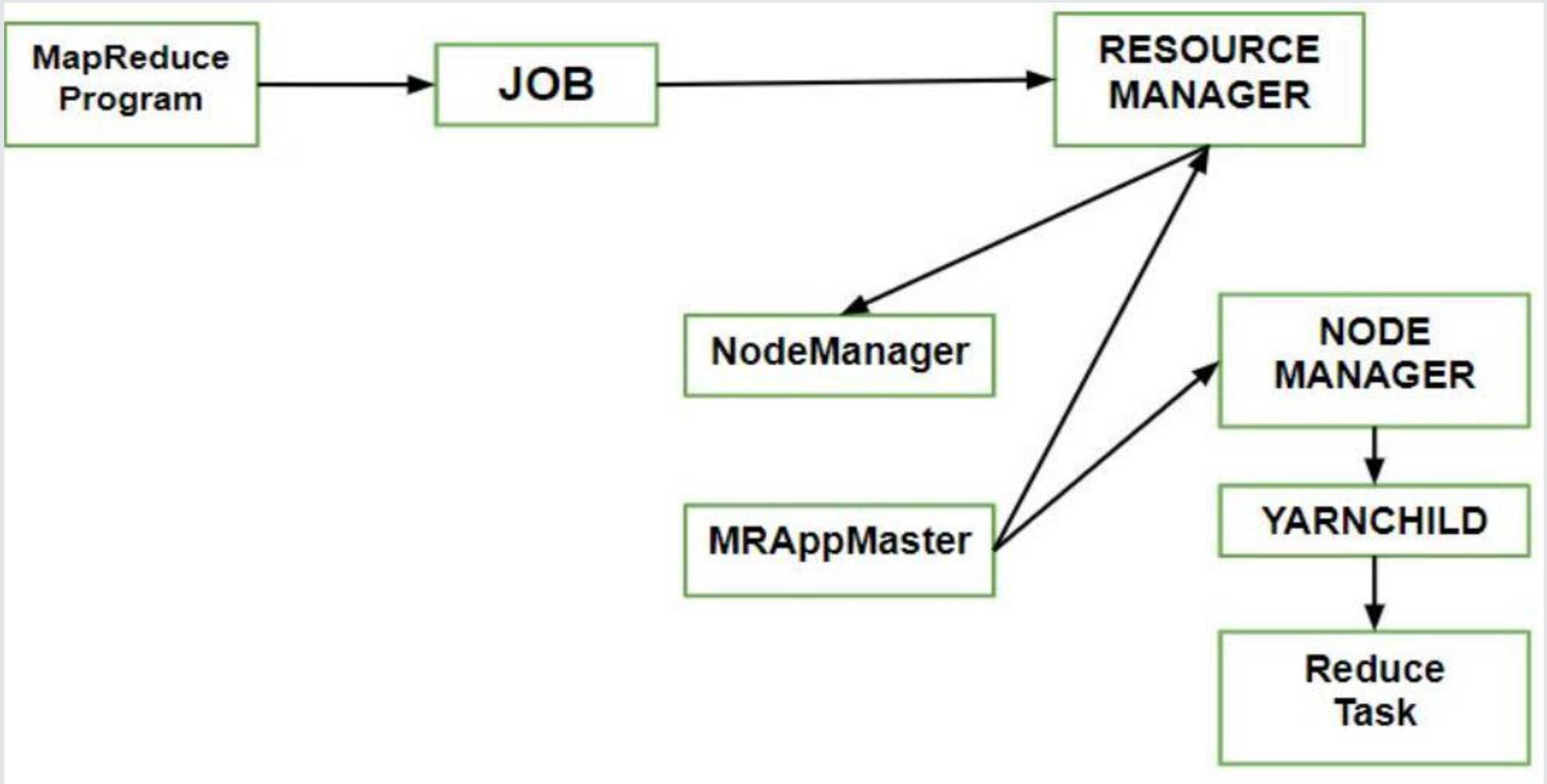
Components of MapReduce

1. Client
2. Yarn node manager
3. Yarn resource manager
4. MapReduce application master
5. Distributed Filesystem

Components of MapReduce

1. Client 2. Yarn node manager 3. Yarn resource manager

4. MapReduce application master 5. Distributed Filesystem



How to submit Job?

- use the submit()
- submitJobInternal()
- waitForCompletion()
- reports the progress to the console.
- job counters are displayed

How to submit Job?

- use the submit()
- submitJobInternal()
- waitForCompletion()
- reports the progress to the console.
- job counters are displayed

How to submit Job?

- use the submit()
- submitJobInternal()
- waitForCompletion()
- reports the progress to the console.
- job counters are displayed

How to submit Job?

- use the submit()
- submitJobInternal()
- waitForCompletion()
- reports the progress to the console.
- job counters are displayed

How to submit Job?

- use the submit()
- submitJobInternal()
- waitForCompletion()
- reports the progress to the console.
- job counters are displayed

Processes implemented by JobSubmitter for submitting the Job :

Processes implemented by JobSubmitter for submitting the Job :

1. The resource manager askes for a new application ID that is used for MapReduce Job ID.
2. Output specification of the job is checked.
3. If the splits cannot be computed, it computes the input splits for the job
4. Resources needed to run the job is copied

Processes implemented by JobSubmitter for submitting the Job :

Processes implemented by JobSubmitter for submitting the Job :

1. The resource manager askes for a new application ID that is used for MapReduce Job ID.
2. Output specification of the job is checked.
3. If the splits cannot be computed, it computes the input splits for the job
4. Resources needed to run the job is copied

Processes implemented by JobSubmitter for submitting the Job :

Processes implemented by JobSubmitter for submitting the Job :

1. The resource manager askes for a new application ID that is used for MapReduce Job ID.
2. Output specification of the job is checked.
3. If the splits cannot be computed, it computes the input splits for the job
4. Resources needed to run the job is copied

Processes implemented by JobSubmitter for submitting the Job :

Processes implemented by JobSubmitter for submitting the Job :

1. The resource manager askes for a new application ID that is used for MapReduce Job ID.
2. Output specification of the job is checked.
3. If the splits cannot be computed, it computes the input splits for the job
4. Resources needed to run the job is copied

Processes implemented by JobSubmitter for submitting the Job :

Processes implemented by JobSubmitter for submitting the Job :

5. It copies job JAR with a high replication factor, which is controlled by mapreduce.client.submit.file.replication property.

6. By calling submitApplication(), submits the job on the resource manager.

Processes implemented by JobSubmitter for submitting the Job :

Processes implemented by JobSubmitter for submitting the Job :

5. It copies job JAR with a high replication factor, which is controlled by mapreduce.client.submit.file.replication property.
6. By calling submitApplication(), submits the job on the resource manager.

That’s all for now…