org.apache.hadoop.mapreduce

Class Mapper<KEYIN,VALUEIN,KEYOUT,VALUEOUT>

java.lang.Object
  org.apache.hadoop.mapreduce.Mapper<KEYIN,VALUEIN,KEYOUT,VALUEOUT>

Direct Known Subclasses:

ChainMapper, FieldSelectionMapper, InverseMapper, MultithreadedMapper, RegexMapper, TokenCounterMapper, ValueAggregatorMapper, WrappedMapper

---

```
@InterfaceAudience.Public
 @InterfaceStability.Stable
public class Mapper<KEYIN,VALUEIN,KEYOUT,VALUEOUT>
extends Object
```

Maps input key/value pairs to a set of intermediate key/value pairs.

Maps are the individual tasks which transform input records into a intermediate records. The transformed intermediate records need not be of the same type as the input records. A given input pair may map to zero or many output pairs.

The Hadoop Map-Reduce framework spawns one map task for each `InputSplit` generated by the `InputFormat` for the job. `Mapper` implementations can access the `Configuration` for the job via the `JobContext.getConfiguration()`.

The framework first calls `setup(org.apache.hadoop.mapreduce.Mapper.Context)`, followed by `map(Object, Object, org.apache.hadoop.mapreduce.Mapper.Context)` for each key/value pair in the `InputSplit`. Finally `cleanup(org.apache.hadoop.mapreduce.Mapper.Context)` is called.

All intermediate values associated with a given output key are subsequently grouped by the framework, and passed to a `Reducer` to determine the final output. Users can control the sorting and grouping by specifying two key `RawComparator` classes.

The `Mapper` outputs are partitioned per `Reducer`. Users can control which keys (and hence records) go to which `Reducer` by implementing a custom `Partitioner`.

Users can optionally specify a `combiner`, via `Job.setCombinerClass(Class)`, to perform local aggregation of the intermediate outputs, which helps to cut down the amount of data transferred from the `Mapper` to the `Reducer`.

Applications can specify if and how the intermediate outputs are to be compressed and which `CompressionCodec`s are to be used via the `Configuration`.

If the job has zero reduces then the output of the `Mapper` is directly written to the `OutputFormat` without sorting by keys.

Example:

```
public class TokenCounterMapper
    extends Mapper<Object, Text, Text, IntWritable>{

  private final static IntWritable one = new IntWritable(1);
  private Text word = new Text();

  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);
    }
  }
}
```

Applications may override the `run(org.apache.hadoop.mapreduce.Mapper.Context)` method to exert greater control on map processing e.g. multi-threaded `Mapper`s etc.

See Also:

`InputFormat`, `JobContext`, `Partitioner`, `Reducer`

Constructor Summary

Constructors

Constructor and Description
`Mapper()`

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method and Description
protected void	`cleanup(org.apache.hadoop.mapreduce.Mapper.Context context)` Called once at the end of the task.
protected void	`map(KEYIN key, VALUEIN value, org.apache.hadoop.mapreduce.Mapper.Context context)` Called once for each key/value pair in the input split.
void	`run(org.apache.hadoop.mapreduce.Mapper.Context context)` Expert users can override this method for more complete control over the execution of the Mapper.
protected void	`setup(org.apache.hadoop.mapreduce.Mapper.Context context)` Called once at the beginning of the task.

Methods inherited from class java.lang.Object

`clone`, `equals`, `finalize`, `getClass`, `hashCode`, `notify`, `notifyAll`, `toString`, `wait`, `wait`, `wait`

Constructor Detail

Mapper

```
public Mapper()
```

Method Detail

setup

```
protected void setup(org.apache.hadoop.mapreduce.Mapper.Context context)
              throws IOException,
                     InterruptedException
```

Called once at the beginning of the task.

Throws:

`IOException`

`InterruptedException`

map

```
protected void map(KEYIN key,
                   VALUEIN value,
                   org.apache.hadoop.mapreduce.Mapper.Context context)
            throws IOException,
                   InterruptedException
```

Called once for each key/value pair in the input split. Most applications should override this, but the default is the identity function.

Throws:

IOException

InterruptedException

cleanup

```
protected void cleanup(org.apache.hadoop.mapreduce.Mapper.Context context)
                throws IOException,
                       InterruptedException
```

Called once at the end of the task.

Throws:

IOException

InterruptedException

run

```
public void run(org.apache.hadoop.mapreduce.Mapper.Context context)
         throws IOException,
                InterruptedException
```

Expert users can override this method for more complete control over the execution of the Mapper.

Parameters:

context -

Throws:

IOException

InterruptedException