Introduction to Big Data

ECAP456

Dr. Rajni Bhalla

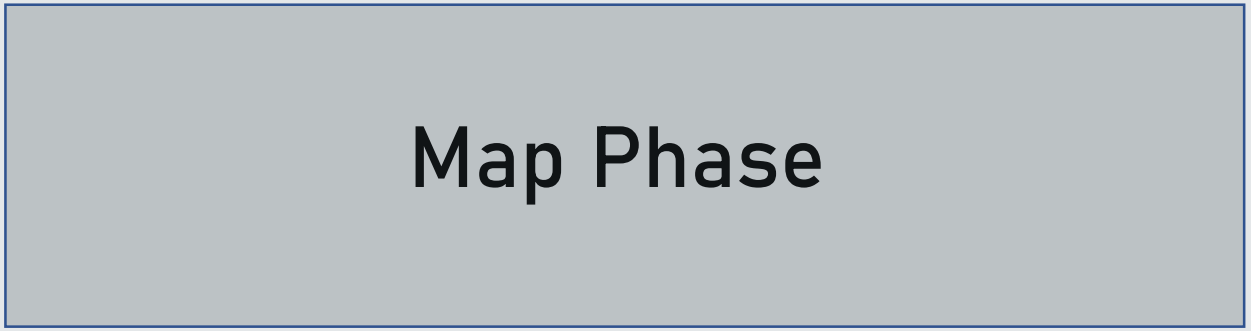
Associate Professor

Learning Outcomes

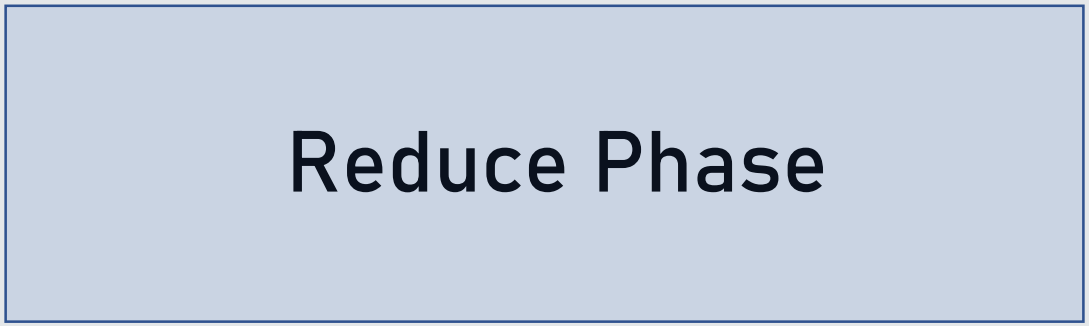
After this lecture, you will be able to

- developing and testing MapReduce jobs with MRUnit,
- developing and testing MapReduce jobs running in local mode

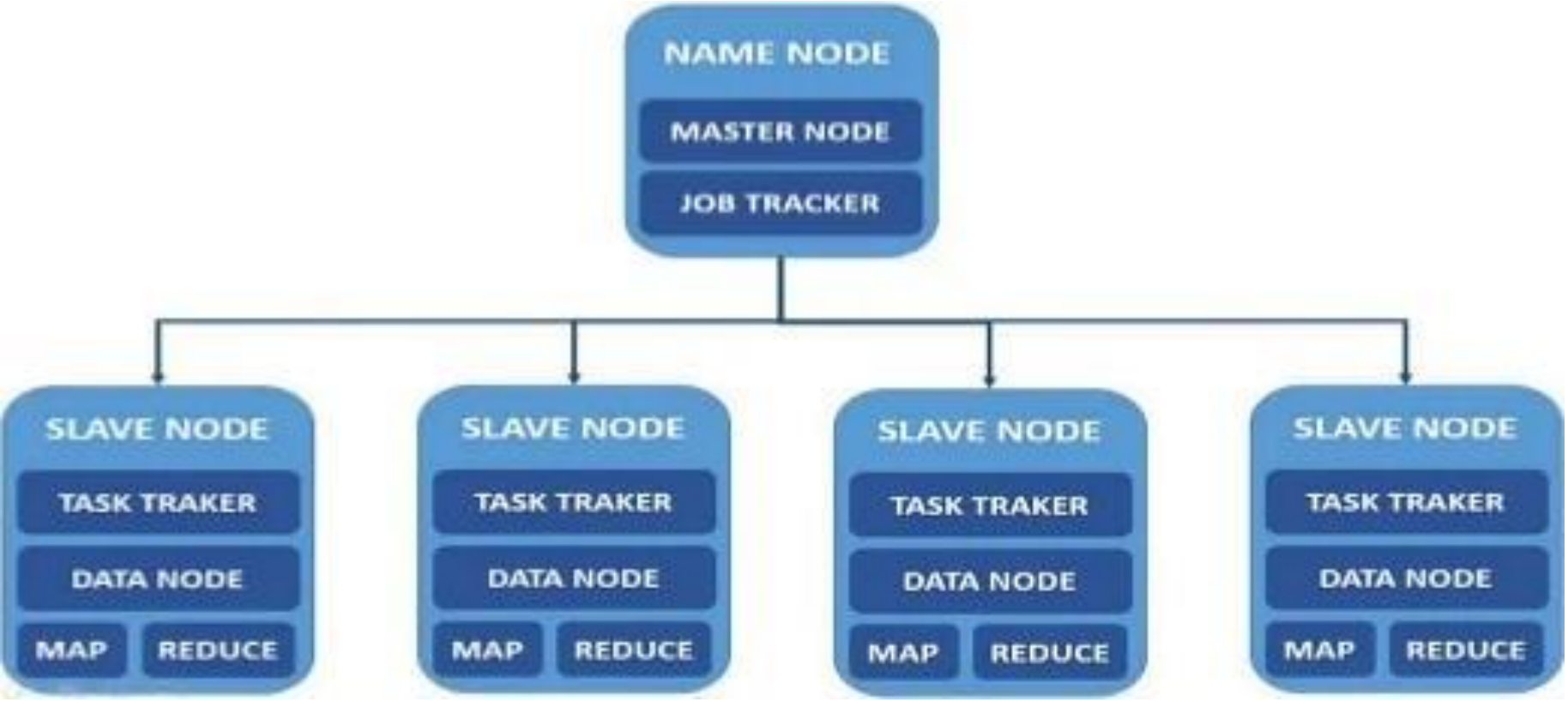
Developing and testing MapReduce jobs with MRUnit



Developing and testing MapReduce jobs with MRUnit



Developing and testing MapReduce jobs with MRUnit



Complexities arise due to the distributed nature of Hadoop

Developing and testing MapReduce jobs with MRUnit

Prior to the release of MRUnit by Cloudera, even the simplest tests running in local mode would have to read from the disk and take several seconds each to set up and run.

Developing and testing MapReduce jobs with MRUnit

- MRUnit removes as much of the Hadoop framework.
- The focus is narrowed to the map and reduce code.
- Developing and testing MapReduce code can be done entirely in the IDE.

Developing and testing MapReduce jobs with MRUnit

- How MRUnit uses the IdentityMapper provided by the MapReduce framework in the lib folder.
- It takes a key-value pair as input.

Developing and testing MapReduce jobs with MRUnit

- The Identity Mapper takes a key-value pair as input and emits the same key-value pair, unchanged.

Developing and testing MapReduce jobs with MRUnit

Step1: Download the latest version of MRUnit from http://mrunit.apache.org/ general/downloads.html

Step2: Create a new Java project

Step3: Add the mrunit-X.Y.Z-incubating-hadoop1.jar file and other Hadoop JAR files to the build path of the Java project

Step4: Create a new class named IdentityMapperTest

Step5: For the full source, review the IdentityMapperTest.java file in the source code.

Developing and testing MapReduce jobs with MRUnit

Step1: Download the latest version of MRUnit from http://mrunit.apache.org/ general/downloads.html

Step2: Create a new Java project

Step3: Add the mrunit-X.Y.Z-incubating-hadoop1.jar file and other Hadoop JAR files to the build path of the Java project

Step4: Create a new class named IdentityMapperTest

Step5: For the full source, review the IdentityMapperTest.java file in the source code.

Developing and testing MapReduce jobs with MRUnit

Step1: Download the latest version of MRUnit from http://mrunit.apache.org/ general/downloads.html

Step2: Create a new Java project

Step3: Add the mrunit-X.Y.Z-incubating-hadoop1.jar file and other Hadoop JAR files to the build path of the Java project

Step4: Create a new class named IdentityMapperTest

Step5: For the full source, review the IdentityMapperTest.java file in the source code.

Developing and testing MapReduce jobs with MRUnit

Step1: Download the latest version of MRUnit from http://mrunit.apache.org/ general/downloads.html

Step2: Create a new Java project

Step3: Add the mrunit-X.Y.Z-incubating-hadoop1.jar file and other Hadoop JAR files to the build path of the Java project

Step4: Create a new class named IdentityMapperTest

Step5: For the full source, review the IdentityMapperTest.java file in the source code.

Developing and testing MapReduce jobs with MRUnit

Step1: Download the latest version of MRUnit from http://mrunit.apache.org/ general/downloads.html

Step2: Create a new Java project

Step3: Add the mrunit-X.Y.Z-incubating-hadoop1.jar file and other Hadoop JAR files to the build path of the Java project

Step4: Create a new class named IdentityMapperTest

Step5: For the full source, review the IdentityMapperTest.java file in the source code.

Developing and testing MapReduce jobs with MRUnit

Follow these steps to test a mapper with MRUnit

1. public class IdentityMapperTest extends TestCase

2. private Mapper identityMapper;
   private MapDriver mapDriver;

3. @Before

public void setup() { identityMapper = new IdentityMapper(); mapDriver = new MapDriver(identityMapper); }

Developing and testing MapReduce jobs with MRUnit

Follow these steps to test a mapper with MRUnit

1. public class IdentityMapperTest extends TestCase
2. private Mapper identityMapper;
   private MapDriver mapDriver;
3. @Before

public void setup() { identityMapper = new IdentityMapper(); mapDriver = new MapDriver(identityMapper); }

Developing and testing MapReduce jobs with MRUnit

Follow these steps to test a mapper with MRUnit

1. public class IdentityMapperTest extends TestCase
2. private Mapper identityMapper;
   private MapDriver mapDriver;
3. @Before

public void setup() { identityMapper = new IdentityMapper(); mapDriver = new MapDriver(identityMapper); }

Developing and testing MapReduce jobs with MRUnit

Follow these steps to test a mapper with MRUnit

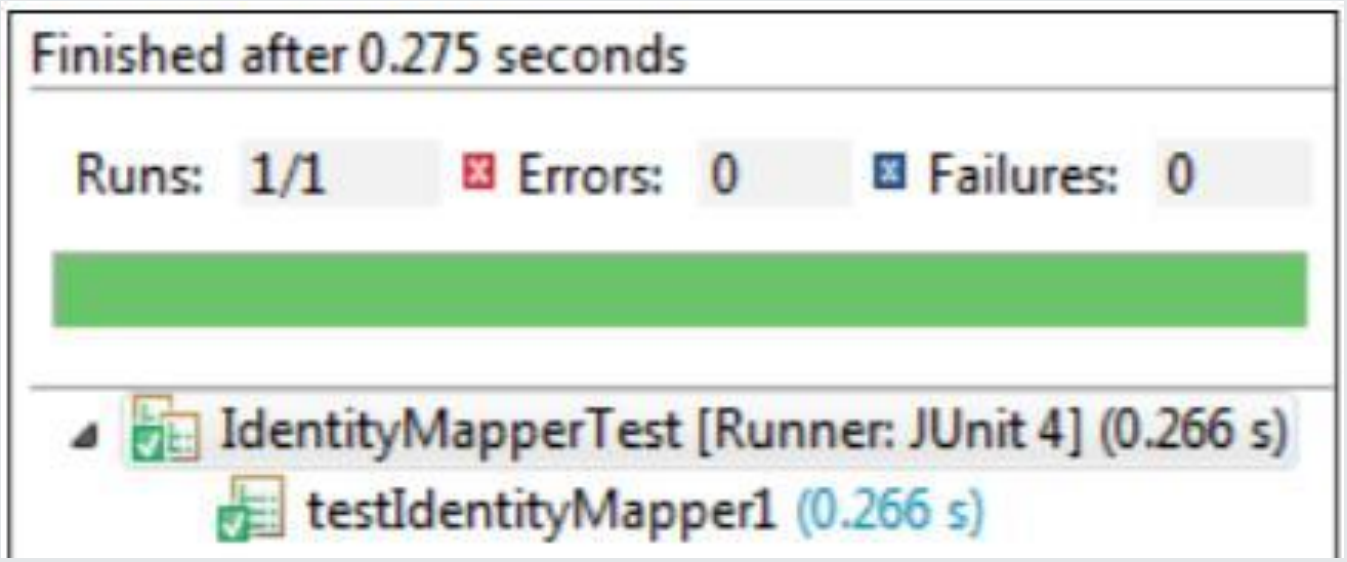
4. @Test

```
public void testIdentityMapper1() {

mapDriver.withInput(new Text("key"), new Text("value"))

mapDriver.withOutput(new Text("key"), new Text("value"))

.runTest();

}
```

Developing and testing MapReduce jobs with MRUnit

Follow these steps to test a mapper with MRUnit

5. Run the application

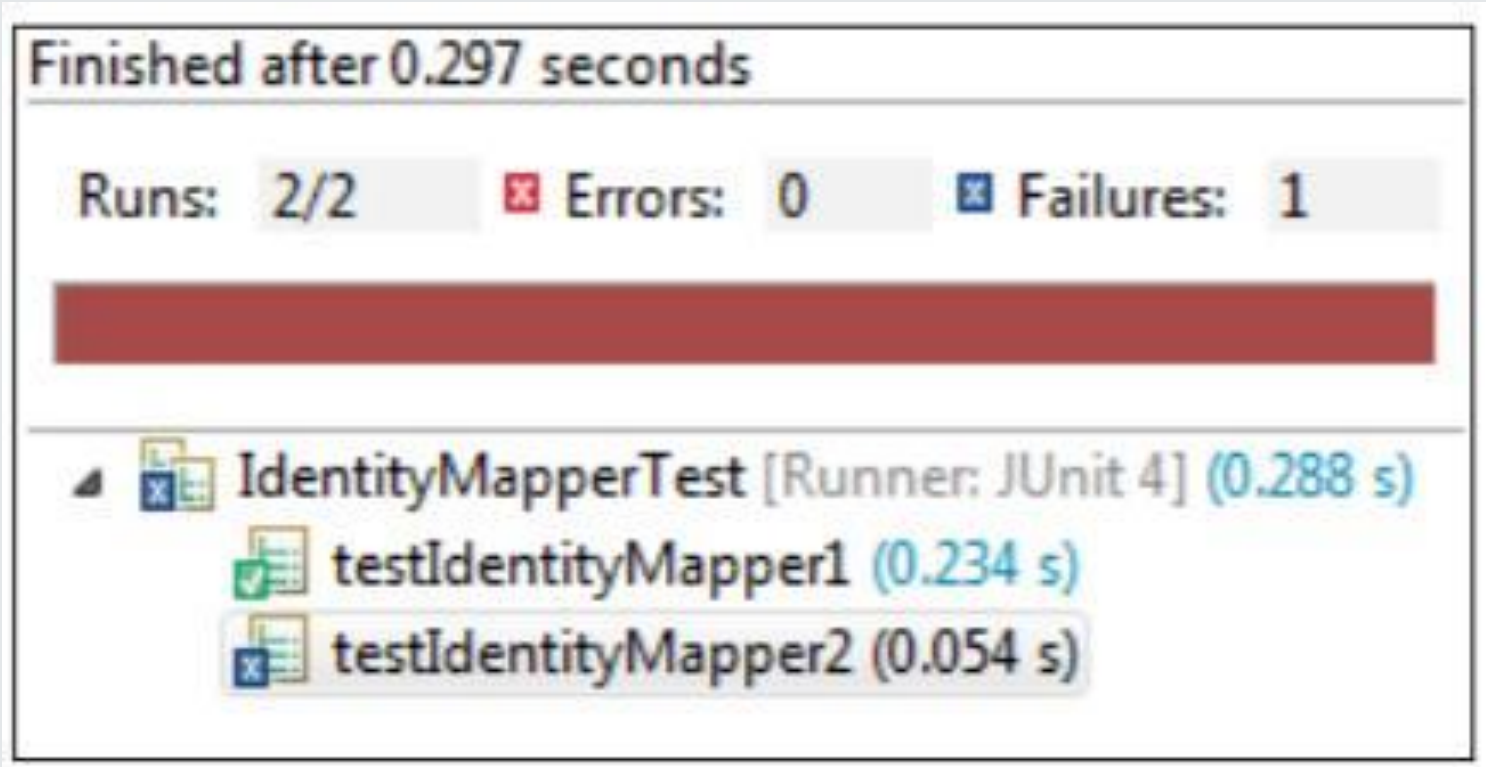


Developing and testing MapReduce jobs with MRUnit

6.
```
@Test

public void testIdentityMapper2() {

mapDriver.withInput(new Text("key"), new Text("value"))

mapDriver.withOutput(new Text("key2"), new Text("value2"))

 mapDriver.runTest();

}
```

Developing and testing MapReduce jobs with MRUnit

7. Run the application again



Developing and testing MapReduce jobs running in local mode

Developing and testing MapReduce jobs running in local mode

Getting ready

Download the weblog_entries_bad_records.txt dataset from the Packt website, http://www.packtpub.com/support.

Developing and testing MapReduce jobs running in local mode

Getting ready

Download the weblog_entries_bad_records.txt dataset from the Packt website, http://www.packtpub.com/support.

CounterExample.java class

Developing and testing MapReduce jobs running in local mode

Getting ready

Download the weblog_entries_bad_records.txt dataset from the Packt website, http://www.packtpub.com/support.

CounterExample.java class

Using Counters in a MapReduce job to track bad records recipe.

Developing and testing MapReduce jobs running in local mode

- How to do it...

Step1: Open the $HADOOP_HOME/conf/mapred-site.xml file in a text editor.

Step2:

```
<property>

<name>mapred.job.tracker</name>

<value>local</value>

</property>
```

Developing and testing MapReduce jobs running in local mode

How to do it...

core-site.xml file

3. Set the fs.default.name property value to file:///:

<property>

<name>fs.default.name</name>

<value>file:///</value>

</property>

Developing and testing MapReduce jobs running in local mode

How to do it...

4. hadoop-env.sh file and add the following line:

```
export HADOOP_OPTS="-
agentlib:jdwp=transport=dt_socket,server=y,
suspend=y,address=7272"
```

Developing and testing MapReduce jobs running in local mode

How to do it...

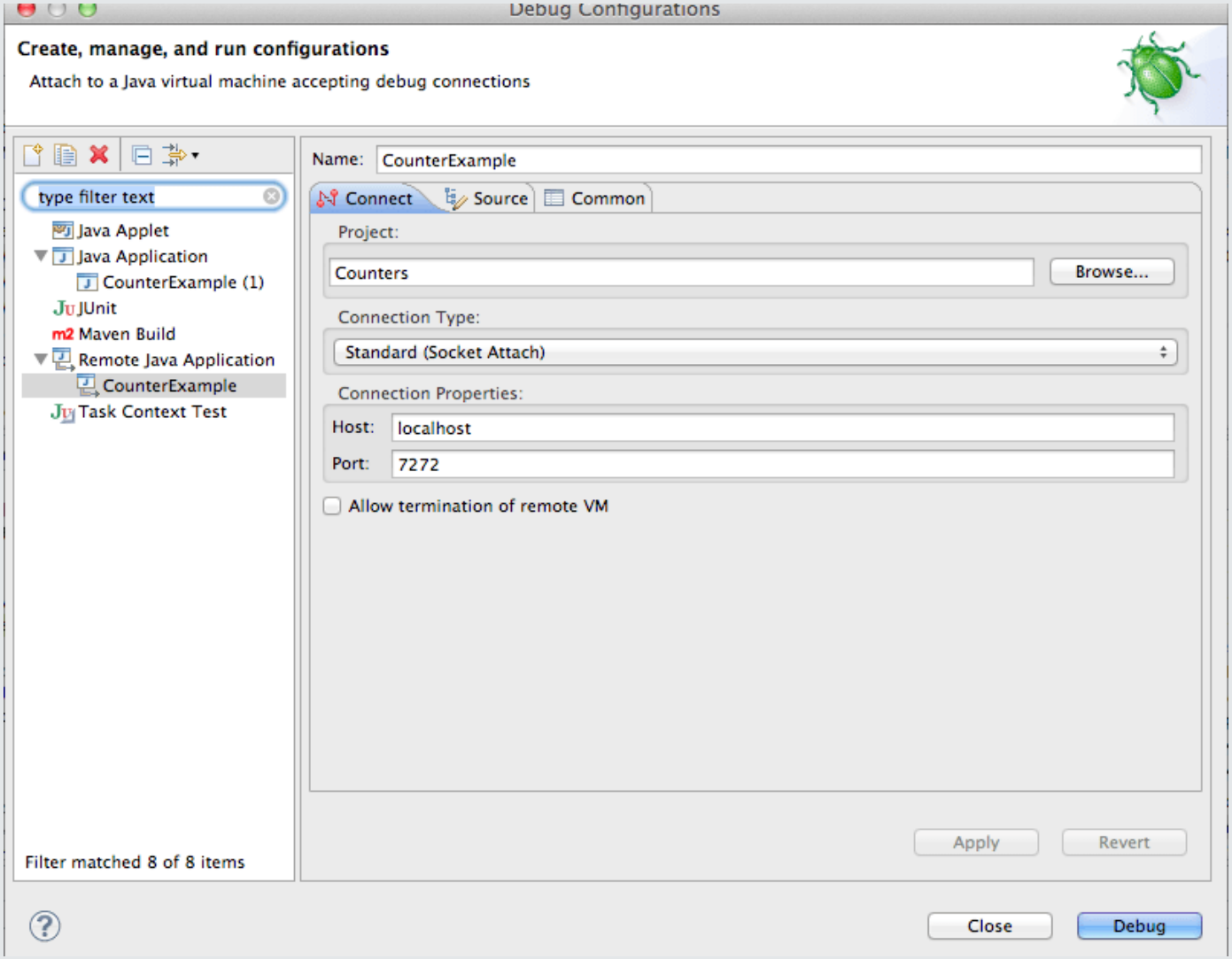
5. $HADOOP_HOME/bin/hadoop jar ./CountersExample.jar com.packt. hadoop.solutions.CounterExample /local/path/to/weblog_entries_bad_ records.txt /local/path/to/weblog_entries_clean.txt

You'll get the following output:

Listening for transport dt_socket at address: 7272

Developing and testing MapReduce jobs running in local mode

6.



Developing and testing MapReduce jobs running in local mode

7. Create a new breakpoint and debug.


That's all for now...