

Java-Success.com

Prepare to fast-track, choose & go places with 800+ Java & Big Data Q&As with lots of code & diagrams.

[Home](#) [Why? ▾](#) [300+ Java FAQs ▾](#) [300+ Big Data FAQs ▾](#) [Courses ▾](#)[👤 Membership ▾](#) [Your Career ▾](#)[Home](#) › [bigdata-success.com](#) › [Tutorials - Big Data](#) › [TUT - Apache Storm](#) › 02: Simple

Apache Storm application running on a single node local cluster

02: Simple Apache Storm application running on a single node local cluster

 Posted on [January 18, 2018](#)

This extends [Installing & getting started with Apache Storm on Cloudera quickstart](#) and [Simple Apache Storm application running inside Eclipse in a local cluster](#) to run the same example in Storm cluster mode.

Step 1: Firstly modify the “storm.yaml” in the folder “/opt/storm/apache-storm-1.1.1/conf” after backing up file. Storm works with zookeeper, and we can specify Cloudera’s zookeeper host name by looking it

300+ Java Interview FAQs

300+ Java FAQs



16+ Java Key Areas Q&As



150+ Java Architect FAQs



80+ Java Code Quality Q&As



150+ Java Coding Q&As



300+ Big Data Interview FAQs

300+ Big Data FAQs



Tutorials - Big Data

TUT -  Starting Big Data

TUT - Starting Spark & Scala

up via the Cloudera manager configuration. Also create a folder “/app/storm” folder for Nimbus, Supervisor, and UI processes createfiles under thhis folder whilst running.

```

1
2 # limitations under the License.
3
4 ##### These MUST be filled in for a storm co
5 storm.zookeeper.servers:
6     - "quickstart.cloudera"
7
8 storm.zookeeper.port: 2181
9
10 storm.local.dir: "/app/storm/"
11
12 nimbus.seeds: ["localhost"]
13
14 supervisor.slots.ports:
15     - 6700
16     - 6701
17     - 6702
18     - 6703
19
20
21 #
22 # ##### These may optionally be filled in:
23 #
24 ## List of custom serializations
25 # topology.kryo.register:
26 #     - org.mycompany.MyType
27 #     - org.mycompany.MyType2: org.mycompany.MyType
28 #
29 ## List of custom kryo decorators
30 # topology.kryo.decorators:
31 #     - org.mycompany.MyDecorator
32 #
33 ## Locations of the drpc servers
34 # drpc.servers:
35 #     - "localhost"
36

```

The above “**storm.yaml**” configuration overrides the “**default.yaml**” file that is in the “storm-core-1.1.1.jar”.

Step 2: The pom.xml file will build the jar. The “storm-core-1.1.1.jar” should have the “provided” scope as it

TUT - Starting with Python

TUT - Kafka

TUT - Pig

TUT - Apache Storm

TUT - Spark Scala on Zeppelin

TUT - Cloudera

TUT - Cloudera on Docker

TUT - File Formats

TUT - Spark on Docker

TUT - Flume

TUT - Hadoop (HDFS)

TUT - HBase (NoSQL)

TUT - Hive (SQL)

TUT - Hadoop & Spark

TUT - MapReduce

TUT - Spark and Scala

TUT - Spark & Java

TUT - PySpark on Databricks

TUT - Zookeeper

800+ Java Interview Q&As

300+ Core Java Q&As



300+ Enterprise Java Q&As



150+ Java Frameworks Q&As



120+ Companion Tech Q&As



Tutorials - Enterprise Java



is present in the folder “/opt/storm/apache-storm-1.1.1/lib”.

```
1
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
4     <modelVersion>4.0.0</modelVersion>
5     <groupId>com.mytutorial</groupId>
6     <artifactId>simple-storm</artifactId>
7     <packaging>jar</packaging>
8     <version>1.0-SNAPSHOT</version>
9     <name>simple-storm</name>
10    <url>http://maven.apache.org</url>
11    <dependencies>
12      <dependency>
13        <groupId>junit</groupId>
14        <artifactId>junit</artifactId>
15        <version>3.8.1</version>
16        <scope>test</scope>
17      </dependency>
18
19      <dependency>
20        <groupId>org.apache.storm</groupId>
21        <artifactId>storm-core</artifactId>
22        <version>1.1.1</version>
23        <scope>provided</scope>
24      </dependency>
25    </dependencies>
26
27    <build>
28      <plugins>
29        <plugin>
30          <artifactId>maven-assembly-plugin</artifactId>
31          <configuration>
32            <archive>
33              <manifest>
34                <mainClass>com.mytutorial</mainClass>
35              </manifest>
36            </archive>
37            <descriptorRefs>
38              <descriptorRef>jar-with-dependencies</descriptorRef>
39            </descriptorRefs>
40          </configuration>
41        </plugin>
42      </plugins>
43    </build>
44
45 </project>
46
```

Step 3: The main method topology code will be slightly different to running in the local cluster.

```
1
2 package com.mytutorial;
3
4 import org.apache.storm.Config;
5 import org.apache.storm.StormSubmitter;
6 import org.apache.storm.generated.AlreadyAliveException;
7 import org.apache.storm.generated.AuthorizationException;
8 import org.apache.storm.generated.InvalidTopologyException;
9 import org.apache.storm.topology.TopologyBuilder;
10
11 public class SimpleTopology {
12
13     public static void main(String[] args) throws
14         TopologyBuilder builder = new TopologyBuilder();
15         builder.setSpout("simple-spout", new SimpleSpout(), 1);
16         builder.setBolt("isOdd", new SimpleIsOddBolt(), 1);
17
18         Config conf = new Config();
19
20         StormSubmitter.submitTopology(args[0], conf, builder.createTopology());
21     }
22 }
23
24
```

Step 4: Build the jar file.

```
1
2 bash-4.1$ cd /home/cloudera/projects/simple-storm
3 bash-4.1$ mvn clean package assembly:single
4
```

This will build “simple-storm-1.0-SNAPSHOT-jar-with-dependencies.jar”,

Step 5: Open three terminal windows and start Nimbus, Supervisor, and UI as shown below.

```
1
2 bash-4.1$ cd /opt/storm/apache-storm-1.1.1/bin
3 bash-4.1$ sudo ./storm nimbus
```

```
4 |
|
1 |
2 | bash-4.1$ cd /opt/storm/apache-storm-1.1.1/bin
3 | bash-4.1$ sudo ./storm supervisor
4 |
|
1 |
2 | bash-4.1$ cd /opt/storm/apache-storm-1.1.1/bin
3 | bash-4.1$ sudo ./storm ui
4 |
```

After the UI has started, open a browser and enter the URL “http://quickstart.cloudera:8080”.

Step 6: Submit the Topology to the above 1 node cluster with the following command by opening a new terminal window.

```
1 |
2 | bash-4.1$ cd /opt/storm/apache-storm-1.1.1/bin
3 | bash-4.1$ sudo ./storm jar /home/cloudera/projects/
4 |
```

Step 7: You can see the topology that you had submitted via the UI by refreshing it and also vi the coman-line

```
1 |
2 | bash-4.1$ cd /opt/storm/apache-storm-1.1.1/bin
3 | bash-4.1$ sudo ./storm list
4 |
|
1 |
2 | Topology_name      Status      Num_tasks  Num_worl
3 | -----
4 | simple-storm-tutorial ACTIVE      3          1
5 |
```

Step 8: You can remove the topology with the following command.

```
1  
2 bash-4.1$ cd /opt/storm/apache-storm-1.1.1/bin  
3 bash-4.1$ sudo ./storm kill simple-storm-tutorial  
4
```

If you list it again, the topology would not be running on the cluster.

◀ 02: Simple Apache Storm application running inside Eclipse in a local cluster

18: Q121 – Q124 Choosing between HDFS & AWS S3 for BigData projects interview Q&As ▶

Disclaimer

The contents in this Java-Success are copyrighted and from EmpoweringTech pty Ltd. The EmpoweringTech pty Ltd has the right to correct or enhance the current content without any prior notice. These are general advice only, and one needs to take his/her own circumstances into consideration. The EmpoweringTech pty Ltd will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. Links to external sites do not imply endorsement of the linked-to sites. [Privacy Policy](#).