

Storm任务提交流程：

1.Client端提交Topology到nimbus

调用命令：

```
storm jar WordCount.jar com.stone.WordCountMain wordcount
```

实际上是调用：

```
java -client WordCount.jar com.stone.WordCountMain wordcount
```

2.通过TopologyBuilder将Spout、Bolt按照一定的逻辑顺序构建Topology程序。

```
1. TopologyBuilder builder = new TopologyBuilder();
2. //RandomSentenceSpout类，在已知的英文句子中，随机发送一条句子出去。
3. builder.setSpout("spout1", new RandomSentenceSpout(), 3);
4. // SplitSentenceBolt类，主要是将一行一行的文本内容切割成单词
5. builder.setBolt("split1", new SplitSentenceBolt(), 9).shuffleGrouping("spout1");
6. // WordCountBolt类，对单词出现的次数进行统计
7. builder.setBolt("count2", new WordCountBolt(), 3).shuffleGrouping("split1");
```

3.调用TopologyBuilder的createTopology()方法，获取StormTopology实例对象。源码如下：

```
1. // STONE_NOTE 调用TopologyBuilder的此方法，创建StormTopology的实例对象
2. public StormTopology createTopology() {
3.     Map<String, Bolt> boltSpecs = new HashMap<String, Bolt>();
4.     Map<String, SpoutSpec> spoutSpecs = new HashMap<String, SpoutSpec>();
5.     for (String boltId : _bolts.keySet()) {
6.         IRichBolt bolt = _bolts.get(boltId);
7.         ComponentCommon common = getComponentCommon(boltId, bolt);
8.         boltSpecs.put(boltId, new Bolt(ComponentObject.serialized_java(Utils.javaS
erialize(bolt)), common));
9.     }
10.    for (String spoutId : _spouts.keySet()) {
11.        IRichSpout spout = _spouts.get(spoutId);
12.        ComponentCommon common = getComponentCommon(spoutId, spout);
13.        spoutSpecs.put(spoutId, new SpoutSpec(ComponentObject.serialized_java(Util
s.javaSerialize(spout)), common));
14.    }
15.    // STONE_NOTE 将Spout和Bolt的相关信息都封装在对应的map中，然后获取StormTopolog
y实例对象
17.    return new StormTopology(spoutSpecs, boltSpecs, new HashMap<String, StateSpout
Spec>());
18. }
```

4.开始提交任务，具体过程如下：

(1) 调用StormSubmitter.submitTopologyWithProgressBar("WordCount", conf, builder.createTopology())提交任务。

submitTopologyWithProgressBar的源码如下：

```

1. // STONE_NOTE 调用此方法提交任务
2. public static void submitTopologyWithProgressBar(String name, Map stormConf, Storm
3. Topology topology, SubmitOptions opts) throws AlreadyAliveException,
4. InvalidTopologyException {
5.
6.     /**
7.      * remove progress bar in jstorm
8.      */
9.     // STONE_NOTE 调用submitTopology方法，传入Topology的名称、配置参数、实例对象
10.    submitTopology(name, stormConf, topology, opts);
11. }

```

(2) 在`submitTopologyWithProgressBar`方法中，调用了`StormSubmitter`的`submitTopology(name, stormConf, topology, opts)`方法。

`submitTopology`方法的源码如下：

```

1. public static void submitTopology(String name, Map stormConf, StormTopology topolo
2. gy, SubmitOptions opts) throws AlreadyAliveException,
3. InvalidTopologyException {
4.     // STONE_NOTE 检验Stormconf，必须是json-serializable Json的序列化对象
5.     if (!Utils.isValidConf(stormConf)) {
6.         throw new IllegalArgumentException("Storm conf is not valid. Must be json-
7. serializable");
8.     }
9.     // STONE_NOTE 利用stormConf创建一个hashmap的实例，并传给stormConf
10.    stormConf = new HashMap(stormConf);
11.    // STONE_NOTE 获得命令行参数，并放入stormConf中
12.    stormConf.putAll(Utils.readCommandLineOpts());
13.    Map conf = Utils.readStormConfig();
14.    conf.putAll(stormConf);
15.    putUserInfo(conf, stormConf);
16.    try {
17.        String serConf = Utils.to_json(stormConf);
18.        if (localNimbus != null) {
19.            LOG.info("Submitting topology " + name + " in local mode");
20.            // STONE_NOTE 如果localNimbus不为空的话，调用本地模式运行
21.            localNimbus.submitTopology(name, null, serConf, topology);
22.        } else {
23.            // STONE_NOTE 通过Topology的配置信息，获取到NimbusClient
24.            NimbusClient client = NimbusClient.getConfiguredClient(conf);
25.            try {
26.                // STONE_NOTE 检测Topology的名称在集群上是否存在
27.                if (topologyNameExists(client, conf, name)) {
28.                    // STONE_NOTE 如果已经存在，抛出异常；提示Topology的名称已存在
29.                    throw new RuntimeException("Topology with name `" + name + "`
30. already exists on cluster");
31.                }
32.                // STONE_NOTE 调用submitJar方法，提交jar文件
33.                submitJar(client, conf);
34.                LOG.info("Submitting topology " + name + " in distributed mode wit

```

```

h conf " + serConf);
32.         // STONE_NOTE 否则的话，调用分布式集群模式
33.         if (opts != null) {
34.             // STONE_NOTE 新的提交方式，携带opts参数 提交Topology任务
35.             client.getClient().submitTopologyWithOpts(name, path, serConf,
topology, opts);
36.         } else {
37.             // this is for backwards compatibility
38.             // STONE_NOTE 这个是为了兼容之前的版本 默认将opts设置为ACTIVE
39.             client.getClient().submitTopology(name, path, serConf, topolog
y);
40.         }
41.     } finally {
42.         client.close();
43.     }
44. }
45. LOG.info("Finished submitting topology: " + name);
46. } catch (InvalidTopologyException e) {
47.     LOG.warn("Topology submission exception", e);
48.     throw e;
49. } catch (AlreadyAliveException e) {
50.     LOG.warn("Topology already alive exception", e);
51.     throw e;
52. } catch (TopologyAssignException e) {
53.     LOG.warn("Failed to assign " + e.get_msg(), e);
54.     throw new RuntimeException(e);
55. } catch (TException e) {
56.     LOG.warn("Failed to assign ", e);
57.     throw new RuntimeException(e);
58. }
59. }

```

在submitTopology()方法中,做了一下工作:

1) 检验Stormconf, 必须是json-serializable Json的序列化对象

```
Utils.isValidConf(stormConf)
```

2) 判断Topology的运行模式

```
// STONE_NOTE 如果localNimbus不为空的话，调用本地模式运行
```

```
localNimbus.submitTopology(name, null, serConf, topology);
```

3) 如果为分布式集群模式运行

```
// STONE_NOTE 检测Topology的名称在集群上是否存在
```

```
topologyNameExists(client, conf, name)
```

```
// STONE_NOTE 调用submitJar方法，提交jar文件
```

```
submitJar(client, conf);
```

```
// STONE_NOTE 新的提交方式，携带opts参数 提交Topology任务
```

```
client.getClient().submitTopologyWithOpts(name, path, serConf, topology, opts);
```

最终任务提交完成！