

storm脚本：是一个python脚本

调用storm脚本时，执行其中的main方法，main方法源代码如下：

```
1. def main():
2.     if len(sys.argv) <= 1:
3.         print_usage()
4.         sys.exit(-1)
5.     global CONFIG_OPTS
6.     config_list, args = parse_config_opts(sys.argv[1:])
7.     parse_config(config_list)
8.     COMMAND = args[0]
9.     ARGS = args[1:]
10.    (COMMANDS.get(COMMAND, unknown_command))(*ARGS)
```

main方法中有两个参数：

COMMAND = args[0]

ARGS = args[1:]

所有的COMMAND：

```
COMMANDS = {
    "jar": jar, "kill": kill, "shell": shell, "nimbus": nimbus, "ui": ui, "logviewer": logviewer,
    "drpc": drpc, "supervisor": supervisor, "localconfvalue": print_localconfvalue,
    "remoteconfvalue": print_remoteconfvalue, "repl": repl, "classpath": print_classpath,
    "activate": activate, "deactivate": deactivate, "rebalance": rebalance, "help": print_usage,
    "list": listtopos, "dev-zookeeper": dev_zookeeper, "version": version, "monitor": monitor
}
```

执行命令，提交Topology任务到集群：

命令如下：

storm jar WordCount.jar com.stone.WordCountMain wordcount

脚本的执行流程如下：

- (1) 获得COMMAND = args[0]参数，COMMAND = "jar"
- (2) 在COMMANDS 中找到对应的命令jar
- (3) 调用def jar 方法

```
1. def jar(jarfile, klass, *args):
2.     exec_storm_class(
3.         klass,
4.         jvmtype="-client",
5.         extrajars=[jarfile, USER_CONF_DIR, STORM_DIR + "/bin"],
6.         args=args,
7.         jvmopts=JAR_JVM_OPTS + ["-Dstorm.jar=" + jarfile])
```

- (4) 在def jar方法中，调用exec_storm_class

```
1. def exec_storm_class(klass, jvmtype="-server", jvmopts=[], extrajars=[], args=[],
2.     fork=False):
3.     global CONFFILE
4.     storm_log_dir = confvalue("storm.log.dir", [CLUSTER_CONF_DIR])
```

```

4.     if(storm_log_dir == None or storm_log_dir == "nil"):
5.         storm_log_dir = STORM_DIR+"/logs"
6.     all_args = [
7.         JAVA_CMD, jvmtype, get_config_opts(),
8.         "-Dstorm.home=" + STORM_DIR,
9.         "-Dstorm.log.dir=" + storm_log_dir,
10.        "-Djava.library.path=" + confvalue("java.library.path", extrajars),
11.        "-Dstorm.conf.file=" + CONFFILE,
12.        "-cp", get_classpath(extrajars),
13.    ] + jvmopts + [klass] + list(args)
14.    print("Running: " + " ".join(all_args))
15.    if fork:
16.        os.spawnvp(os.P_WAIT, JAVA_CMD, all_args)
17.    else:
18.        os.execvp(JAVA_CMD, all_args) # replaces the current process and
19.        # never returns

```

(5) 在exec_storm_class中最终操作系统调用os.spawnvp(os.P_WAIT, JAVA_CMD, all_args)或者os.execvp(JAVA_CMD, all_args)完成命令的执行。

也就是执行：

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

实际上是调用：

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