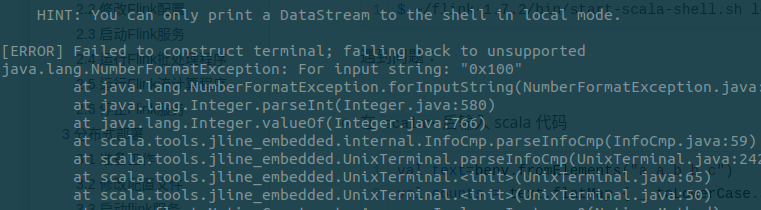
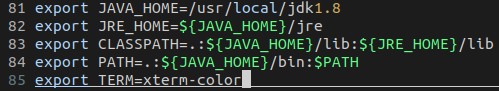
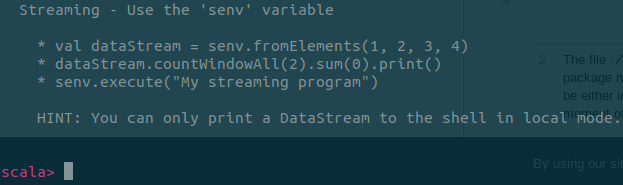
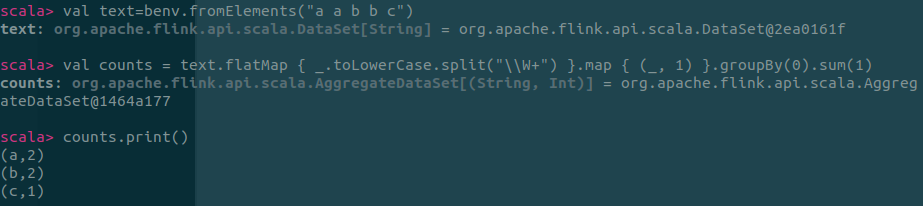
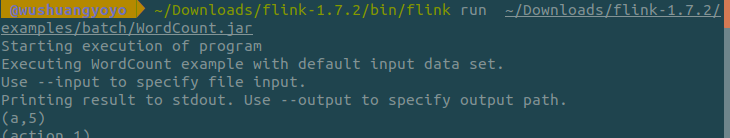
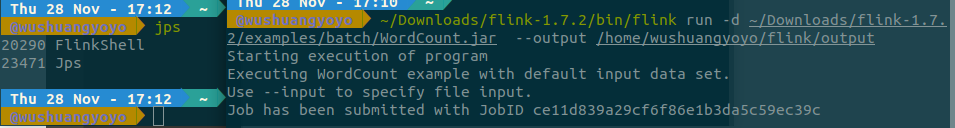
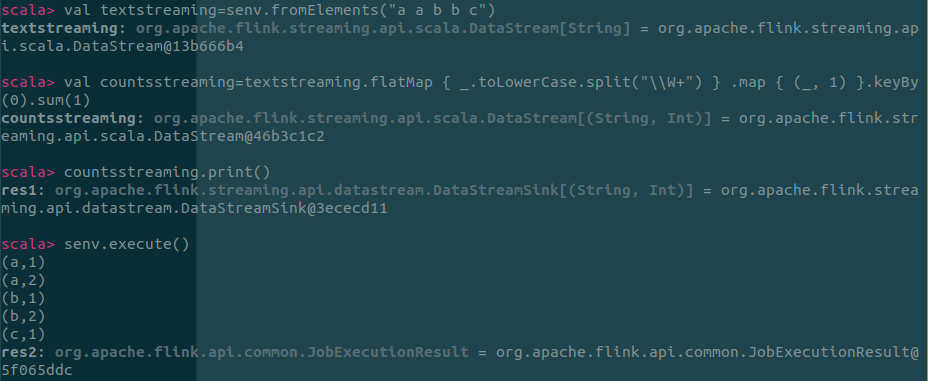
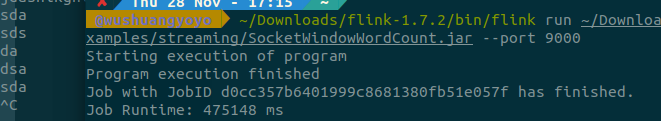
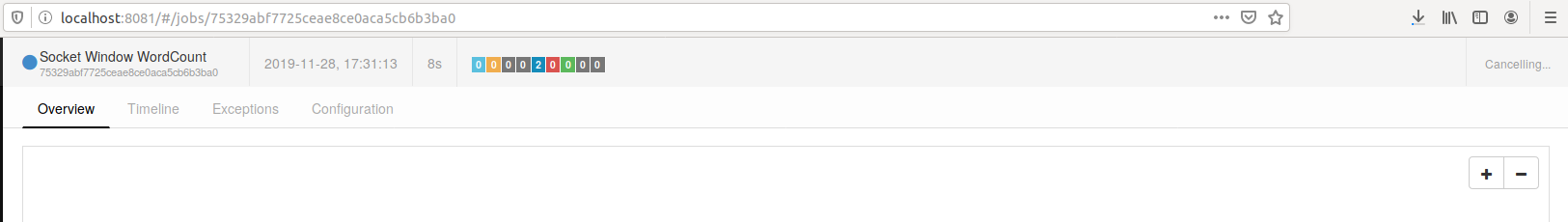
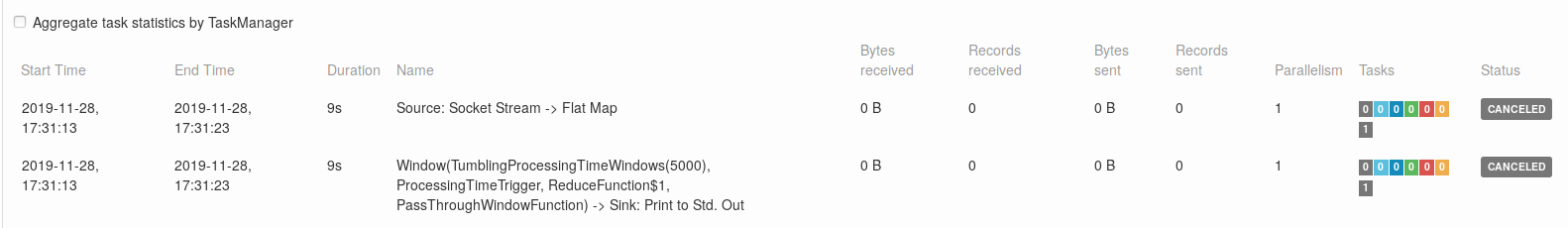
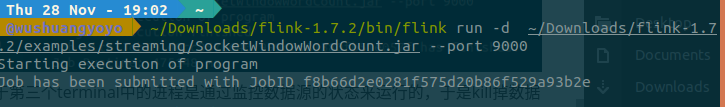
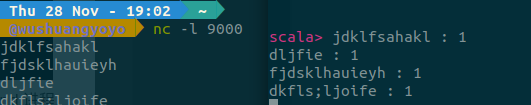
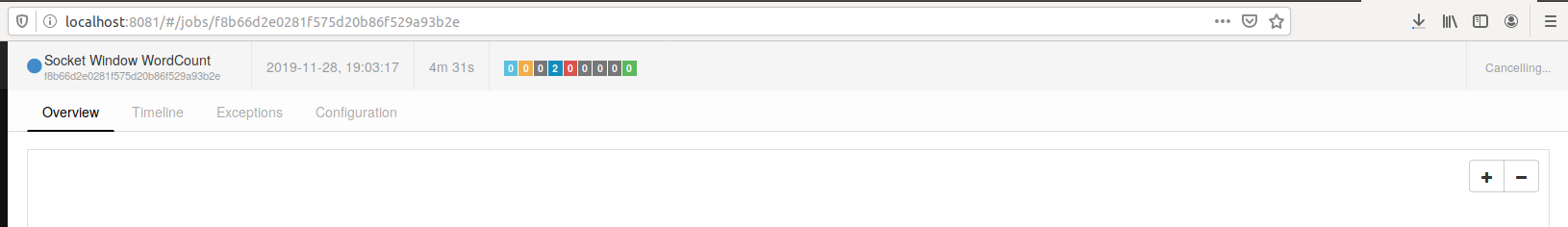
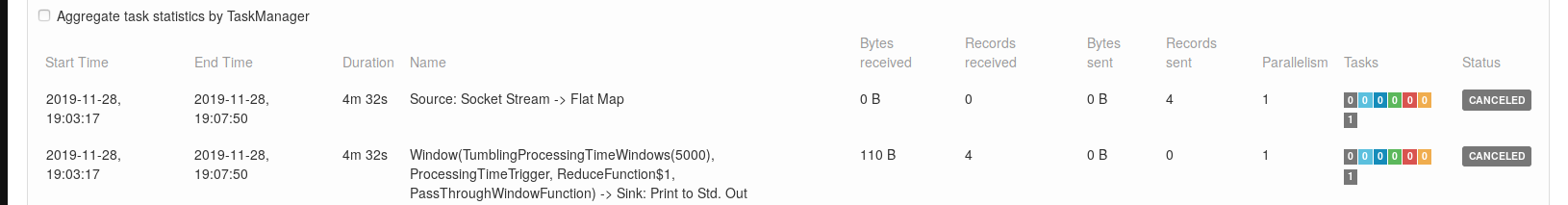
# Flink 部署

## 1 单机集中式部署

### 1.1 准备工作并运行Flink批处理程序

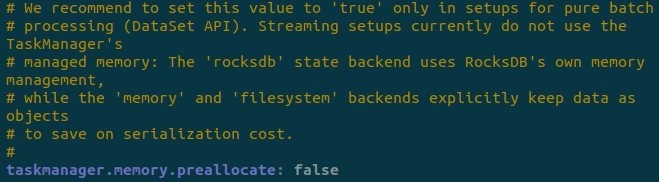
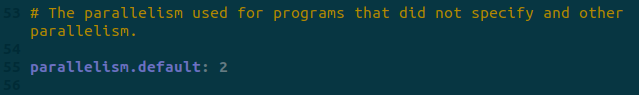
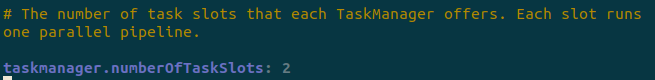
* 使用shell运行批处理程序
  + 本地模式启动Scala-Shell
  + $ ~/flink-1.7.2/bin/start-scala-shell.sh local
  + 遇到问题：
  + 
  + 查询[stackoverflow](https://issues.sonatype.org/browse/NEXUS-17638)得知，此问题出现的原因是因为在Ubuntu LTS18.04 中，系统会尝试将 Scala 放进一个 tmux session中运行，而 tmux 会将 TERM 参数设置为 xterm-256color。
  + 所以解决方法为在/etc/profile 或者 shell 的 zsh 启动脚本 ~/.zshrc 中设置一个 export 的命令，将 TERM 设置成 xterm-color（这里使用了后者，即在shell启动脚本中设置）：
  + 
  + 重新启动 shell，在运行 scala-shell，成功。结果如下：
  + 
  + 在 scala> 后输入 scala 代码。运行结果如下：
  + 
* 通过提交 jar 包运行批处理程序
* flink不可以直接提交jar包运行，首先仍需要在终端 1本地模式启动Scala-Shell。
* 另起一个终端，提交jar包
  + 默认模式提交
  + 
  + 在**运行过程中**另起一个终端执行 jps 查看进程：
* 
  + detached模式提交
* 运行结果与运行时进程情况如下：
* 
* 查看打印的结果如下:
* fig:

### 1.3 运行Flink流计算程序

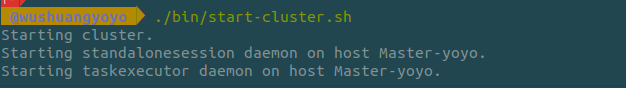
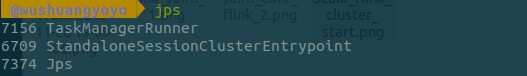
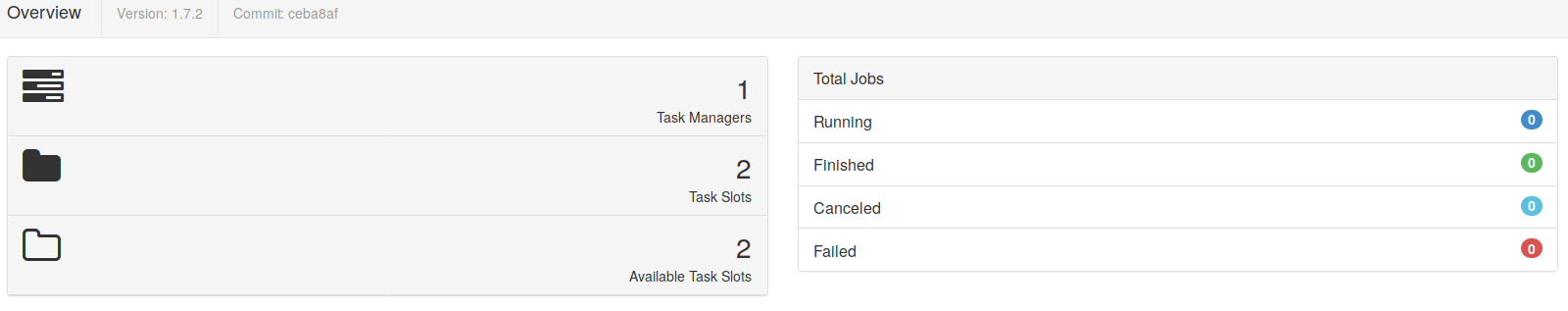
* 使用shell运行流计算程序
  + 本地模式启动Scala-Shell，同上。
  + 在 scala> 后输入 scala 代码
  + 
* 通过提交 jar 包运行流计算程序
* 同上，需要在终端本地模式启动Scala-Shell。
* 之后提交jar包
  + 默认模式提交
  + 在另一个终端中启动socket服务作为数据源。显示如下：
  + 
  + 在第三个终端中提交jar包
  + 
  + 向第二个终端中输入数据进行wordcount计算，在第一个终端运行结果如下：
  + 
* 在运行过程中另起一个终端执行 jps 查看进程：
* 
* 流计算任务的终止
* 在作为数据源的终端中，用键盘 ctrl + c杀掉socket服务：
* 
* 由于第三个terminal中的进程是通过监控数据源的状态来运行的，于是kill掉数据源的进程就可以kill掉wordcount运算，而scala-shell的进程不依赖于数据源，所以不会受影响。
  + 或者在[WebUI](localhost:8081])上cancel：
  + 
  + cansel后的任务显示canceled：
  + 
  + detached模式提交
* 终端2中启动本地服务
* 
* 终端3中提交任务jar包
* 
* 程序提交完毕后退出客户端，不再打印作业进度等信息
* 向终端2中输入数据进行wordcount计算，在终端1运行结果如下图所示:
* 
* 在运行过程中另起一个终端执行 jps 查看进程，此时不会出现 CliFrontend 进程
* 
* 流计算任务的终止
  + 在终端2, ctrl + c杀掉socket服务，同上。
  + 或者在[WebUI](localhost:8081])上cancel
  + 
  + cancel后的任务显示canceled：
  + 

## 2 伪分布式部署

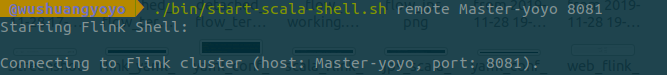
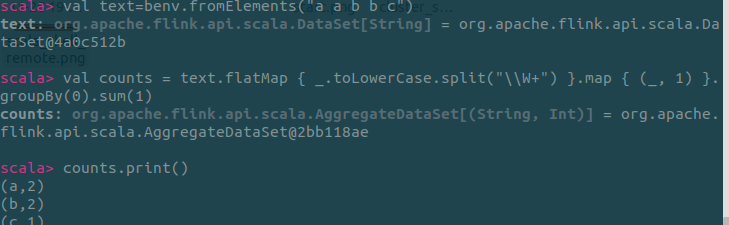
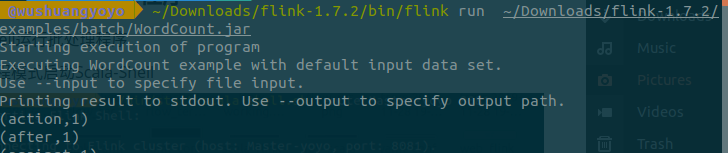
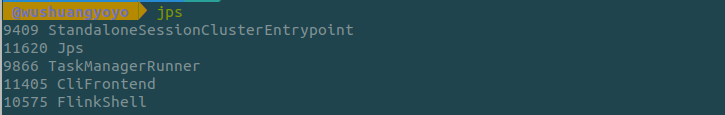
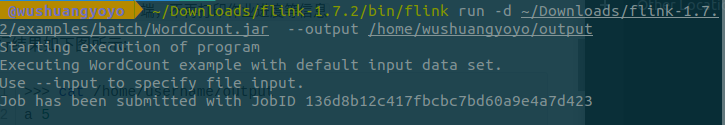
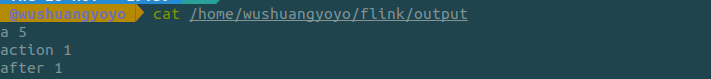
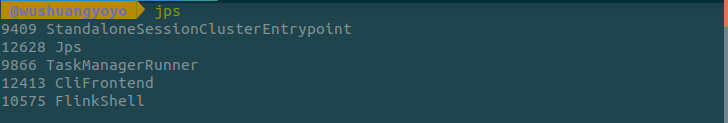
### 2.1 修改Flink配置

* 更改配置文件flink-conf.yaml
* vim ~/Downloads/flink-1.7.2/conf/flink-conf.yaml
* 其他都是按照默认的设定使用，只修改其中的两个参数：
  1. 配置是否在Flink集群启动时候给TaskManager分配内存，默认不进行预分配，这样在我们不适用flink集群时候不会占用集群资源：
  + 
  1. 配置程序默认并行计算的个数：
  + 
  1. 配置TaskManager 提供的任务 slots 数量大小：
  + 
* 以下还有一些非常重要的配置值（需要调节时更改，本例中不做更改）：
* 每个JobManager（jobmanager.heap.mb）的可用内存量
* 每个TaskManager（taskmanager.heap.mb）的可用内存量
* 每台机器的可用CPU数量（taskmanager.numberOfTaskSlots）
* 集群中的CPU总数（parallelism.default）
* 临时目录（taskmanager.tmp.dirs） #内存不够用时，写入到taskmanager.tmp.dirs指定的目录中。如果未显式指定参数，Flink会将临时数据写入操作系统的临时目录。
* 更改配置文件slaves
* 文件中默认内容为localhost，本例中不做修改。

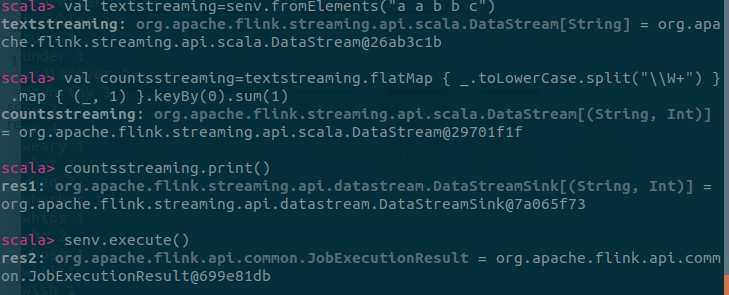
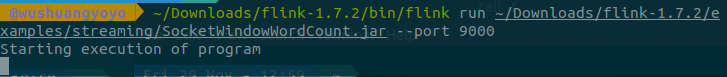
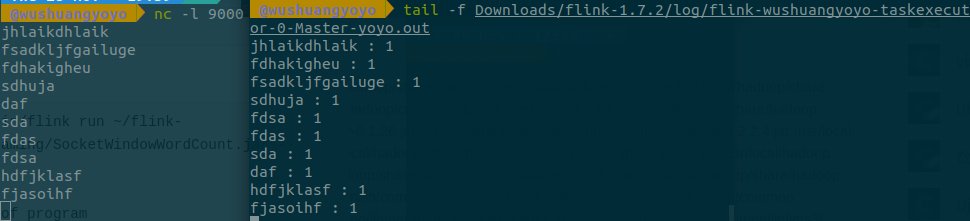
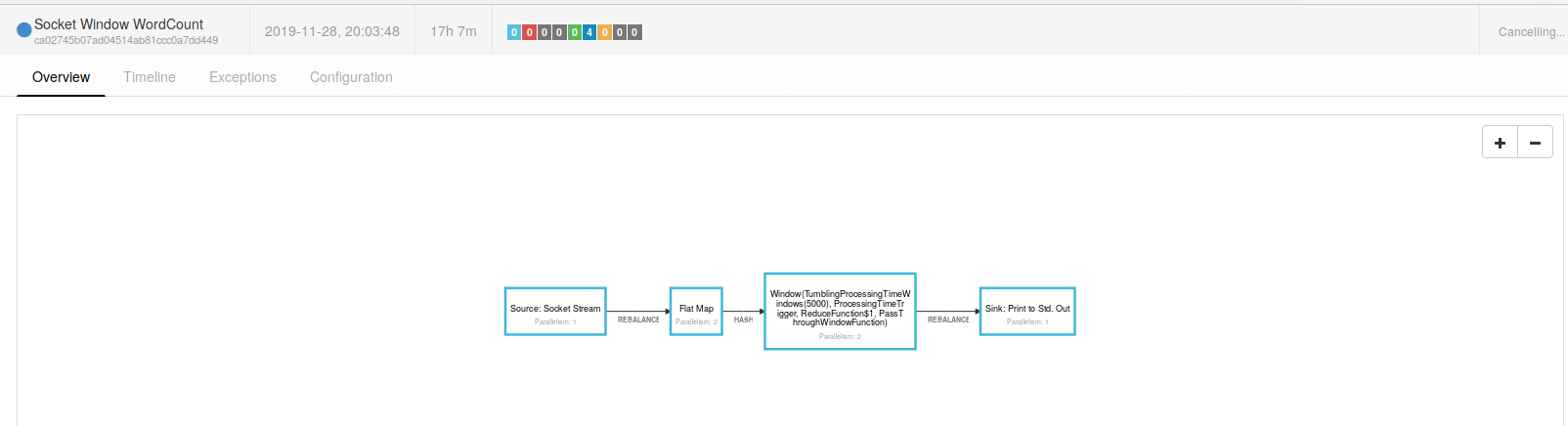
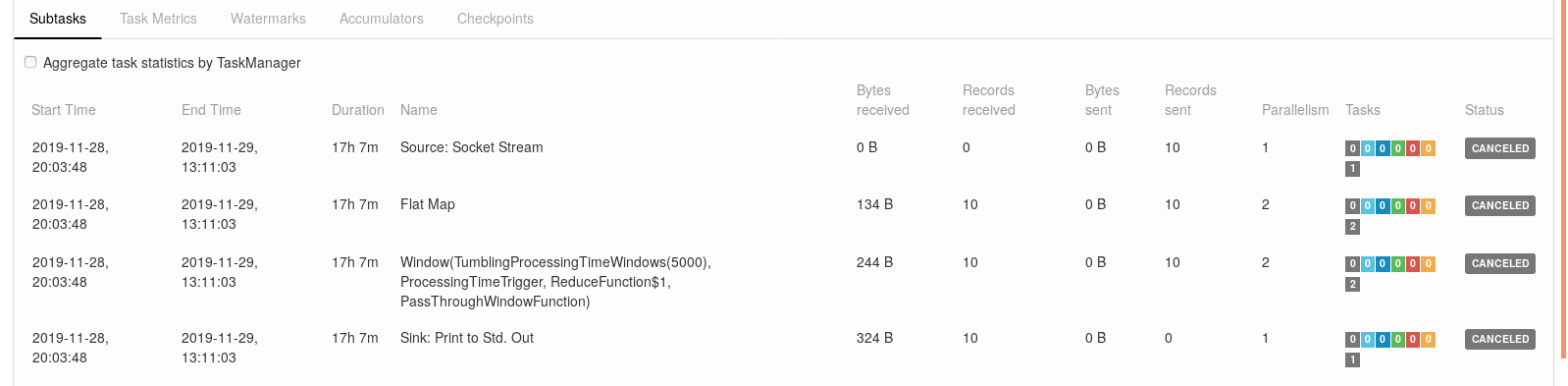
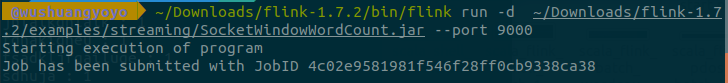
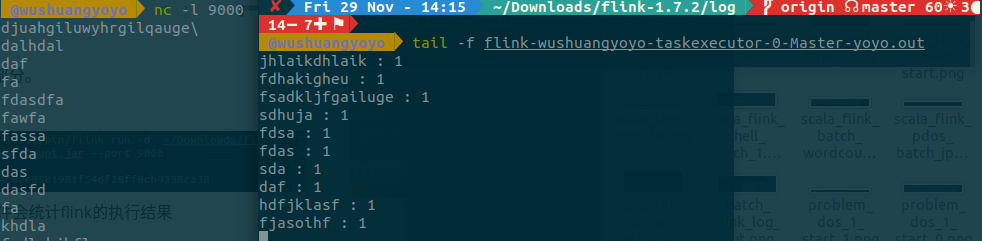
### 2.3 启动Flink服务

* 启动命令
* 
* 查看进程，验证是否成功启动服务
  + 使用jps命令，因为在此单机伪分布式部署模式下，该节点既充当JobManager角色，又充当TaskManager角色，故该节点上会有两个进程：一个JobManager进程和一个TaskManager进程。其中，在standalone模式下，Jobmanager的进程名为StandaloneSessionClusterEntrypoint。结果如下：
  + 
* 在 [Web](http://localhost:8081) 中查看flink状态：
* 

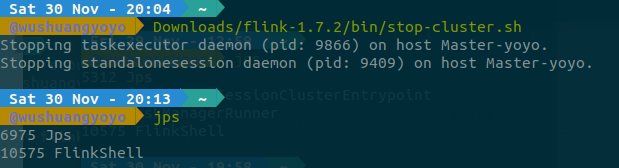
### 2.4 运行Flink批处理程序

* 使用shell运行批处理程序
  + 远程模式启动Scala-Shell
  + 
  + 在 scala> 后运行wordcount程序，结果如下图所示：
  + 
* 通过提交 jar 包运行批处理程序
  + 默认模式提交。运行结果如下：
  + 
  + 在**运行过程中**另起一个终端执行 jps 查看进程  
    
  + detached模式提交
  + 
  + 运行结果如下图所示:
  + 
  + 另起一个终端执行 jps 查看进程
  + 

### 2.5 运行Flink流计算程序

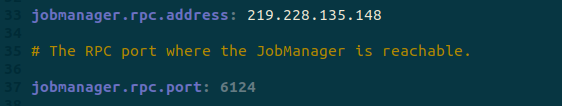
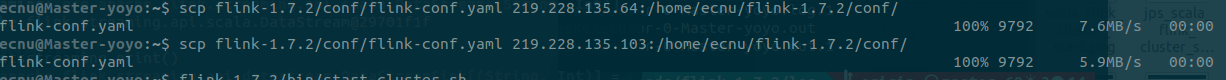
* 使用shell运行流计算程序
  + 远程模式启动Scala-Shell，同上。
  + 在 scala> 后输入 scala 代码
  + 
* 通过提交 jar 包运行流计算程序
  + 默认模式提交
  + 终端1中启动本地服务，同上。
  + 终端2中提交任务jar包
  + 
  + 终端3 中打开log目录下的out文件会统计flink的执行结果。结果如下：
  + 
  + 在运行过程中另起一个终端执行 jps 查看进程。结果如下：
  + 
  + 流计算任务的终止
    - 在终端2, ctrl + c杀掉socket服务，同上一部分。
    - 或者在[WebUI](localhost:8081])上cancel：
    - 
    - canceled后的任务状态显示为canceled：
    - 
* detached模式提交
* 终端1中启动本地服务，同上一部分。
* 终端2中提交任务jar包
* 
* 终端3 中打开log目录下的out文件会统计flink的执行结果
* 
* 在运行过程中另起一个终端执行 jps 查看进程  
   此时不会出现 CliFrontend 进程
* >>>jps  
   19138 StandaloneSessionClusterEntrypoint  
  19604 TaskManagerRunner  
   9098 Jps
* 流计算任务的终止，同上一部分。

### 2.6 停止Flink服务

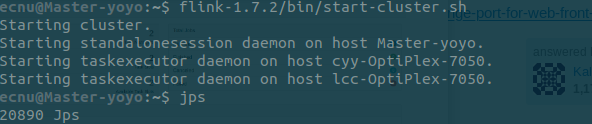
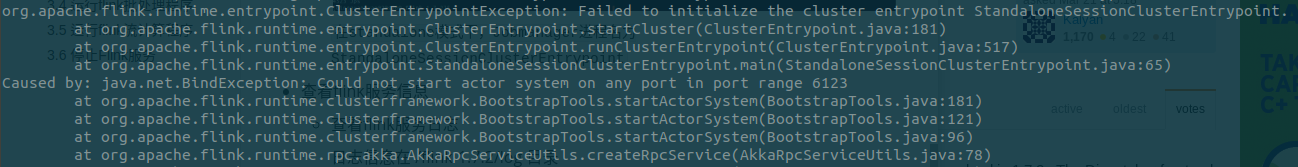
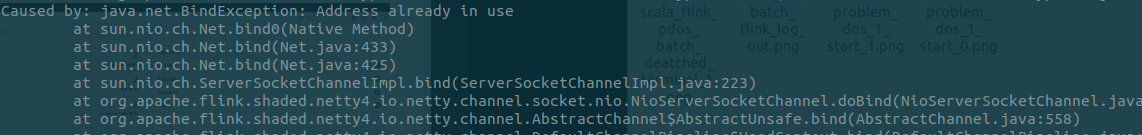
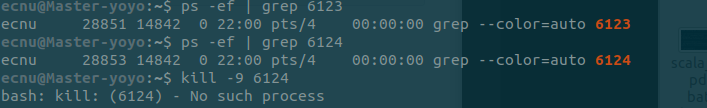
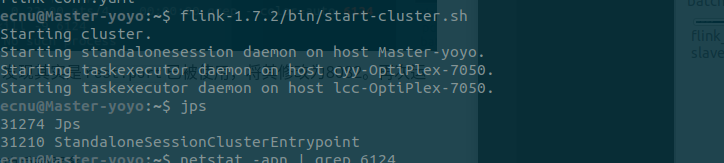
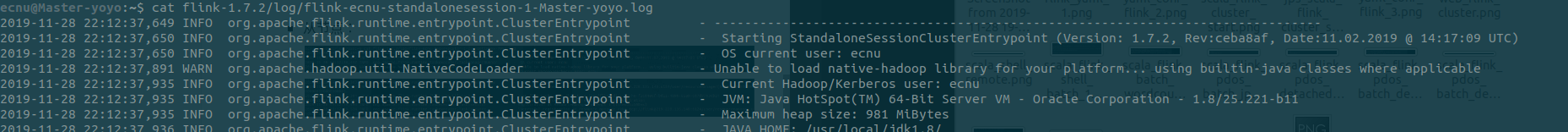
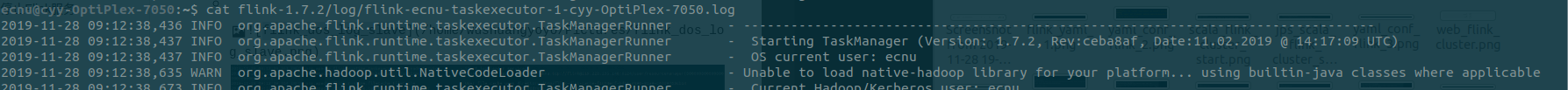
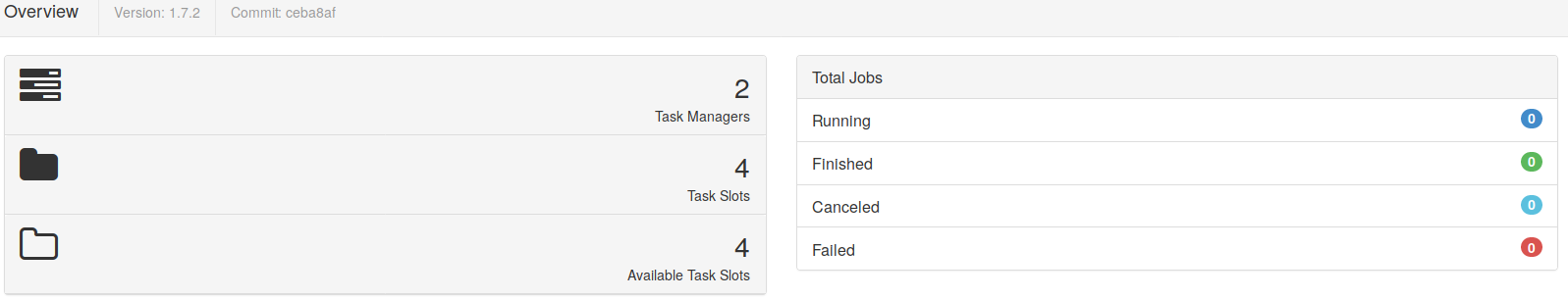
* 停止命令 并通过jps来判断进否关闭成功。
* 

## 3 分布式部署

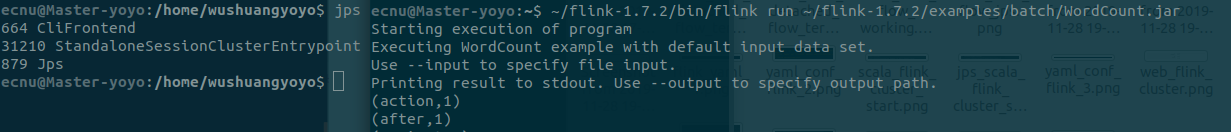
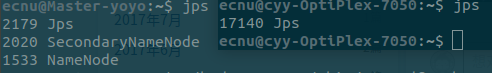
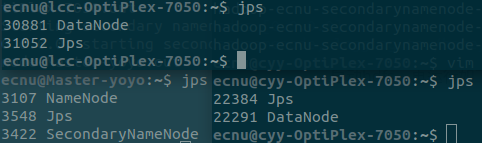
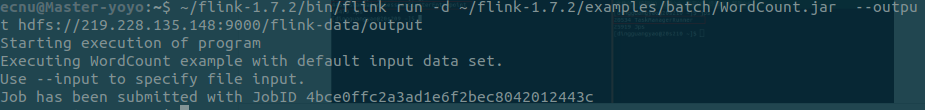
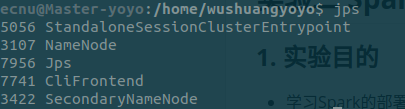
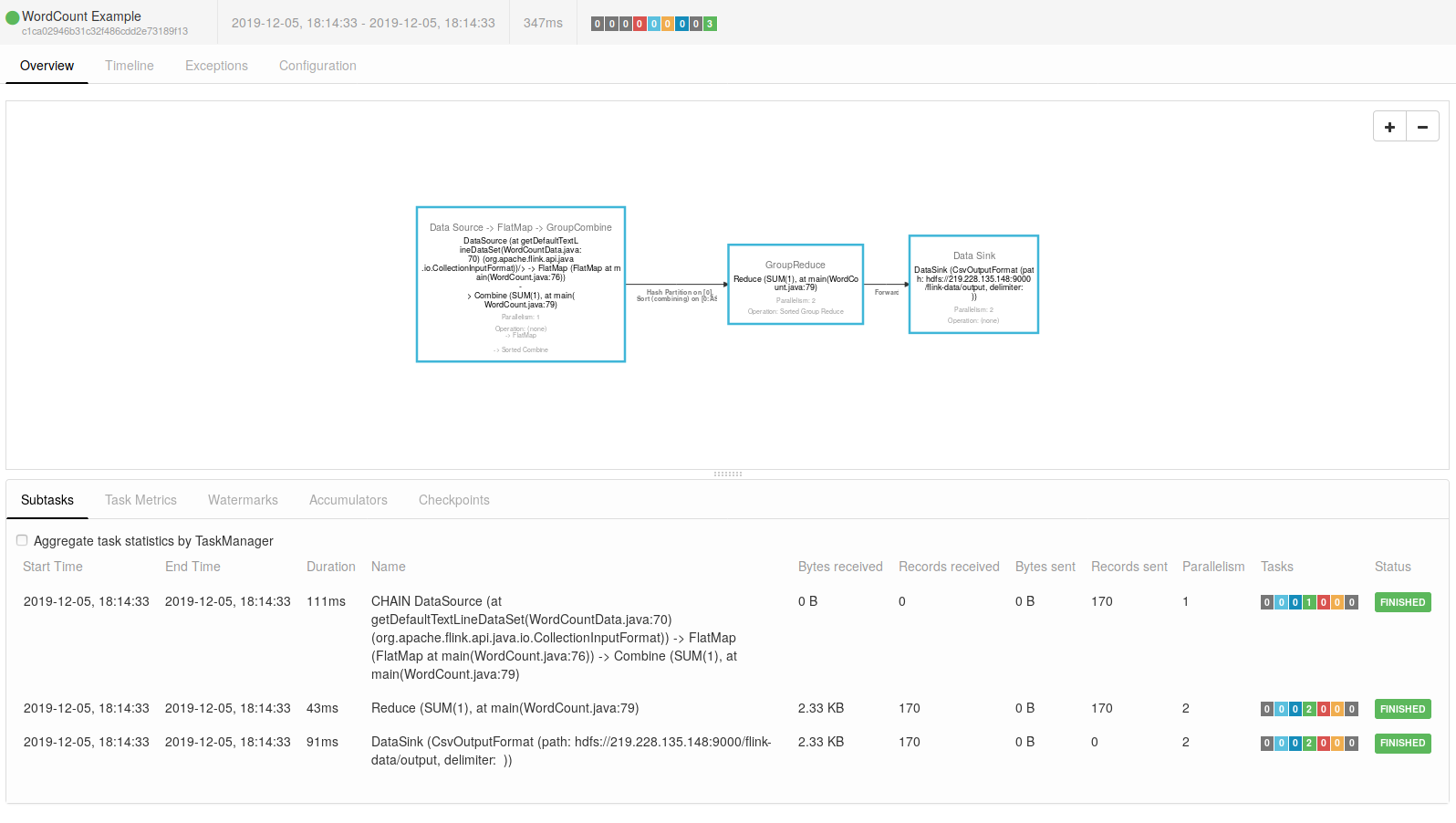
### 3.1 修改配置文件

* 更改配置文件flink-conf.yaml：
* 
* 更改配置文件slaves
* 
* 将配置好的Flink同步到其他节点
* 

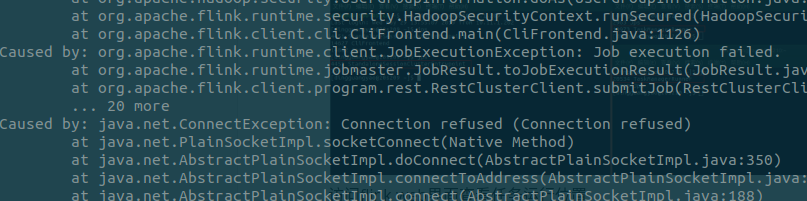
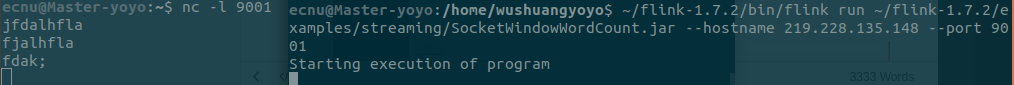
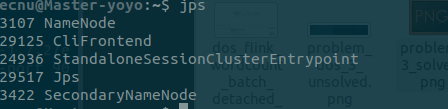
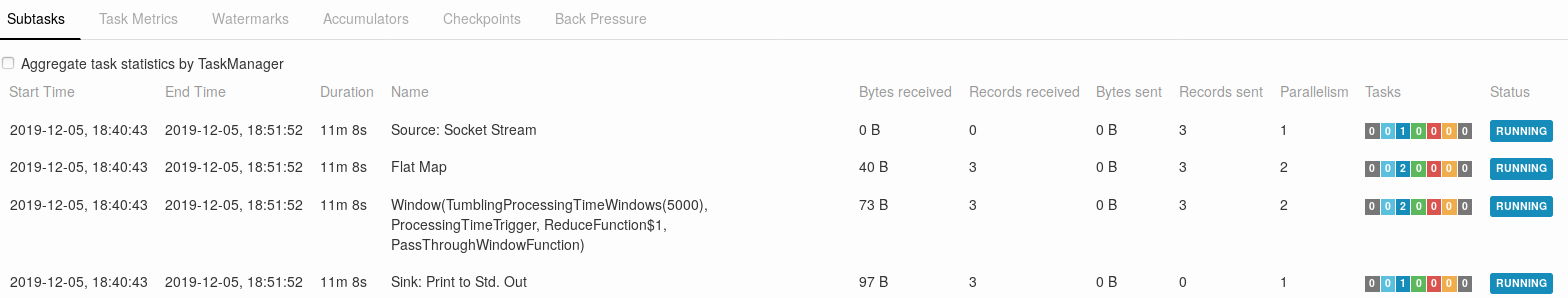
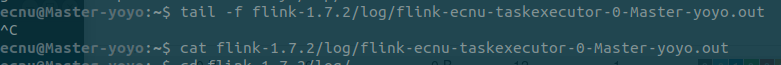
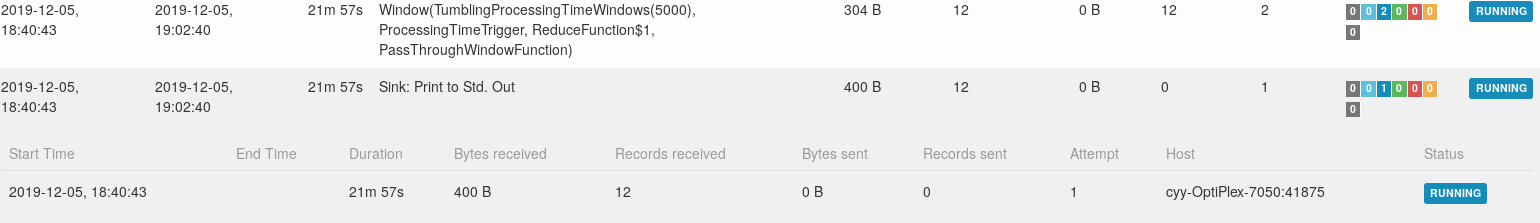
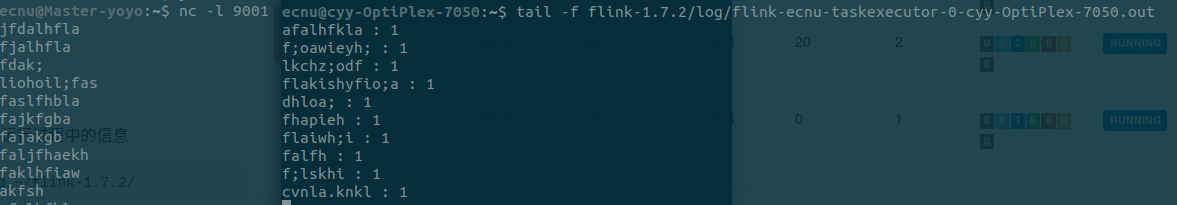
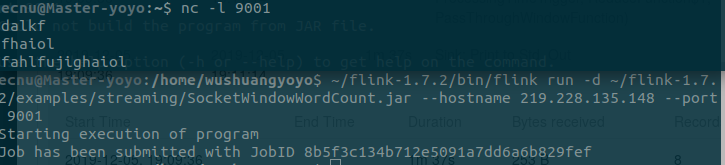
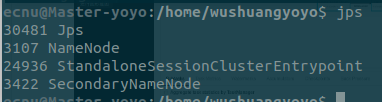
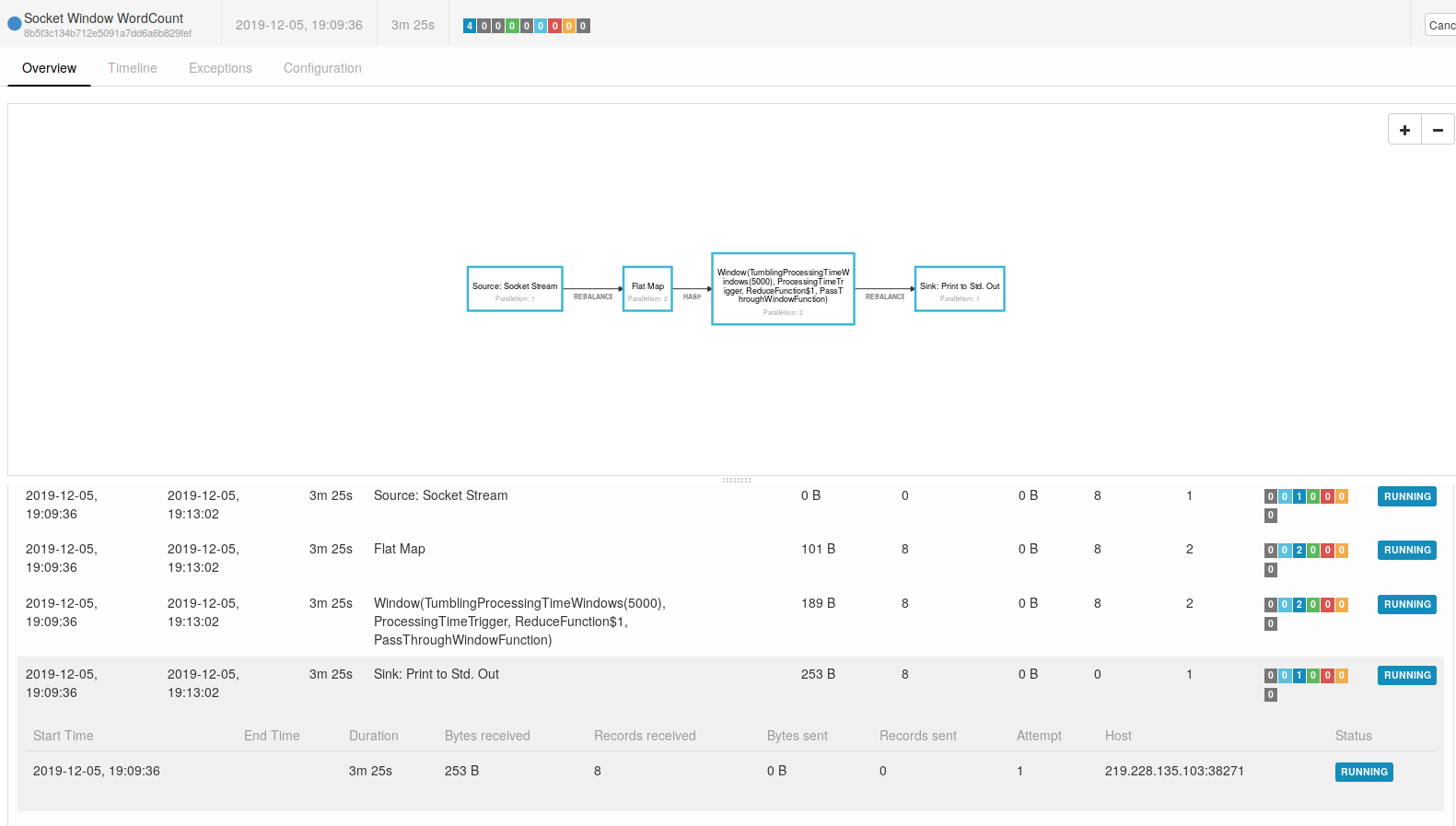
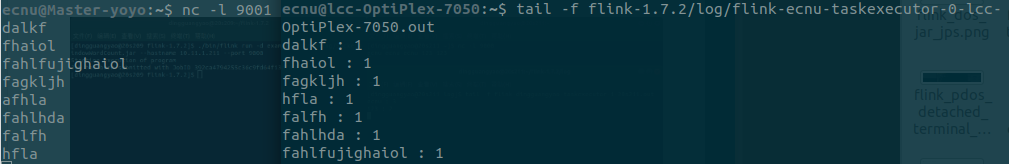
### 3.3 启动flink服务

* 启动命令
* >>> ~/flink-1.7.2/flink-1.7.2/bin/start-cluster.sh
* 遇到问题：
* 
* 运行了分布式的启动程序，但是jps却没有显示进程正在运行，查看log文件，错误信息为：
* 
* 不可以使用6123这个端口，查看网络信息，发现校园网并不可以使用6123这样的ipv6的端口。于是将taskmanager.rpc.port设置为6124，再次运行，依然遇到问题：
* 
* 显示问题为地址已被使用，一开始以为是6124端口被占用，于是尝试了kill -9：
* 
* 结果发现没有在使用。发现其实是rest.port已被使用，将其修改为8082。再次运行，成功。并运行jps检查进程情况，结果如下：
* 
* 从节点进程情况如下：
* 
* 查看flink服务信息
  + 查看flink服务日志(./log/flink-ecnu-standalonesession-1-Master-yoyo.log / ./log/flink-ecnu-taskexecutor-1-cyy-OptiPlex-7050.log)，其中1表示运行成功。
    - 主节点：
    - 
    - 从节点：
    - 
  + 访问flink web界面
  + 

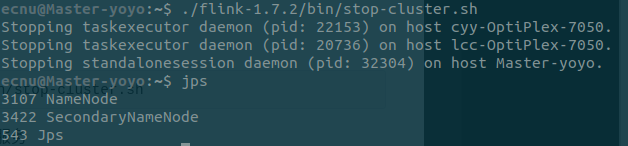
### 3.4 运行flink批处理程序

* 运行jar文件并查看结果
  + 默认模式提交，可以在客户端看到应用程序运行过程中的信息
  + >>> ~/flink-1.7.2/bin/flink run ~/flink-1.7.2/examples/batch/WordCount.jar
* 运行并使用jps命令。结果如下：
* 
  + detached模式提交，在客户端看不到应用程序运行过程中的信息
  + 由于有一个从节点的ip地址发生了改变，于是修改hadoop-2.9.2/etc/hadoop/slaves中的ip地址至与现实一致。
  + 同样的问题依然存在：
* 
* 查看从节点的datanode的logs日志。发现问题出现的原因是All specified directories are failed to load.即datanode和namenode相关设置不一致，只需要修改namenode下的./hadoop-2.9.2/tmp/dfs/current/name/VERSION和datanode下的./hadoop-2.9.2/tmp/dfs/current/data/VERSION至一致便可以解决问题。
* 于是删除tmp文件并重新格式化namenode，重新启动hadoop，成功。结果如下：
* 
* >>> ~/flink-1.7.2/bin/flink run -d ~/flink-1.7.2/examples/batch/WordCount.jar --output hdfs://20s209:9001/flink-data/output
* 运行效果如下图：
* 
* 同时运行jps来显示此时正在运行的进程：
* 
* detached模式提交并不是如文档所说的没有CliFrontend进程。感谢陈启航学长细心指导得知其中原因（事实上，除了个别看了源码解决的问题，基本上所有的网上解决不了的问题，都是陈启航学长帮忙解决的）：CliFrontend进程在detached模式下的准备阶段运行，当任务提交以后，便自动退出。所以在submitted的情况下是没有CliFrontend进程的。
* 查看hdfs中的运行结果：
* 
* 访问flink web 界面
* 

### 3.5 运行flink流计算程序

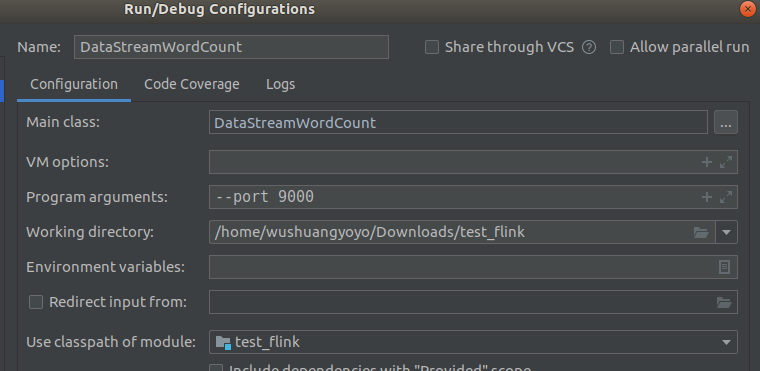
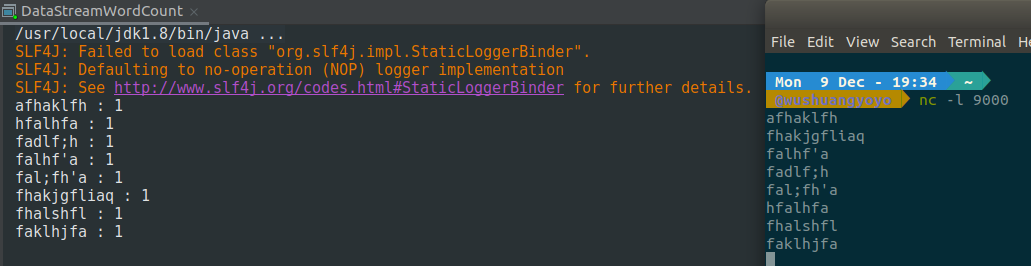
* 运行jar文件并查看结果
  + 默认模式提交，可以在客户端看到应用程序运行过程中的信息
  + >>> ~/flink-1.7.2/bin/flink run ~/flink-1.7.2/examples/streaming/SocketWindowWordCount.jar --hostname 219.228.135.148 --port 9001
  + 程序提交完毕后退出客户端，不再打印作业进度等信息。
* 遇到问题：
* 
* 一开始以为是端口被占用造成的原因，所以更换多次端口，未果。以为是ipv6端口不支持，再次选取ipv4端口在运行，未果。经徐老师提醒，应该先打开netcat数据源端口，再打开dataflow的程序。
* 更换运行顺序以后再次启动，成功。结果如下：
* 
* 使用jps命令查看进程状况：
* 
* 访问flink web界面查看任务运行位置
* 
* 在任务运行位置的flink log目录下输入命令：tail -f ~/flink-1.7.2/log/flink-ecnu-taskexecutor-0-Master-yoyo.out，遇到问题：
* 
* out文件没有输出。经老师提醒，发现输出节点不在本台机器：
* 
* 查看相应的机器中的存储数据，结果如下：
* 
* 流计算任务的终止
  + 在终端2, ctrl + c杀掉socket服务，同上面的步骤。
    - 或者在[WebUI](localhost:8081])上cancel，同上面的步骤。
* **注：杀掉客户端进程是无法停止任务的**
  + detached模式提交，在客户端看不到应用程序运行过程中的信息
  + >>> ~/flink-1.7.2/bin/flink run -d ~/flink-1.7.2/examples/streaming/SocketWindowWordCount.jar --hostname 219.228.135.148 --port 9001
* 
* 使用jps观察进程运行状况：
* 
* 访问flink web界面查看任务运行位置，并且查看数据保存的文件所处位置：
* 
* 在任务运行位置的flink log目录下输入命令：tail -f ~/flink-1.7.2/log/flink-ecnu-taskexecutor-0-lcc-OptiPlex-7050.out，查看运行结果：
* 
* 停止flink正在运行中的任务
  + 在web UI 上cancel，同上面的步骤。

### 3.6 停止Flink服务

* 停止命令
* >>> ~/flink-1.7.2/bin/stop-cluster.sh
* 查看进程，验证是否成功停止服务
  + 若成功停止，JobManager进程和TaskManager进程应消失，如下图所示：
  + 

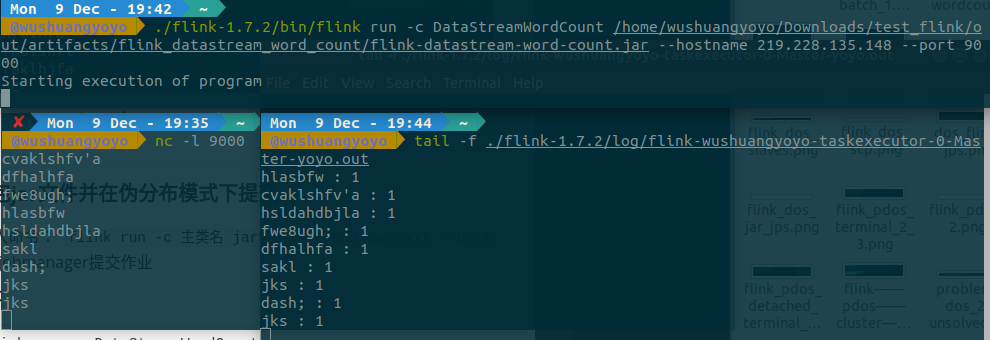
# 1. 编写并调试Flink程序

#### IDE中直接运行

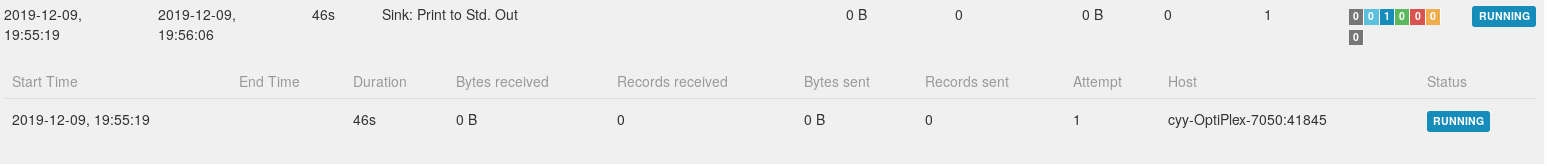
* + 配置运行环境，并进行本地调试。在IntelliJ菜单栏中选择Run->Edit Configuration，在弹出对话框中新建Application配置，配置Main Class为DataStreamWordCount，Program arguments为hostname port，分别为主机名和端口号，默认主机名为localhost。如下图所示：
  + 
  + 配置完成后，右键->Run'DataStreamWordCount'
  + 运行结果如下：
  + 

# 2. 运行Flink程序

#### 利用IDE打包jar文件并在伪分布模式下提交

* + 在终端输入命令，向jobmanager提交作业。并另起终端输入如下命令查看运行结果。
  + 如下所示：
  + 

#### 分布式模式下提交flink程序

* + 在client中输入命令，向地址为host:port的jobmanager提交作业
  + 由于通过flink 的 webUI可以看出输出的节点为cyy-OptiPlex-7050。
  + 如下所示：
  + 
  + 于是在cyy-OptiPlex-7050节点中另起终端输入如下命令查看运行结果：
  + 