**WordCount编程**

实验手册

**OracleWDP·华育兴业**

# 实验目录

## 实验前HDFS平台的准备工作

## IDEA下建立Maven项目project

## WordCount编程

## 程序调度

## 运行程序与结果验证

# 实验内容

1. 实验前HDFS平台的准备工作

知识点

1. 启动Hadoop平台（平台需要处于启动状态）
2. 实验前数据路径/root/experiment/datas的准备（路径下不能有文件）
3. 实验程序每次执行前，HDFS上输出路径的检查（输出目录不能存在）

实验目的

1. 掌握实验前平台环境的准备工作的方法

实验环境

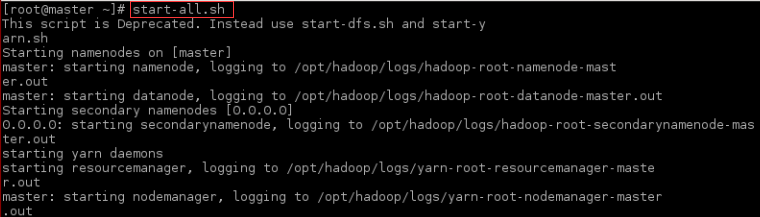
1. Oracle Linux 7.4
2. Hadoop2.7.4

实验步骤

1. 启动Hadoop平台
   * + 1. 启动Hadoop平台。具体的启动方法：鼠标双击桌面上的“Konsole”图标，打开终端，在终端命令窗口中输入Hadoop启动命令。由于配置不同，start-all.sh命令后显示的界面与下面可能有出入，也不要紧，只要保证第2步的jps后，守护进程对就可以。

[Command 001]:

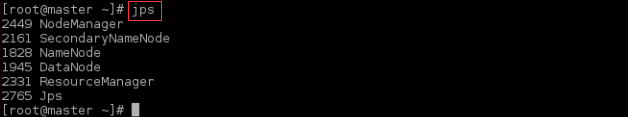
*start-all.sh*



* + - 1. 输入jps命令，查看Hadoop启动的守护进程。如果系统有其它程序同时运行，守护进程的界面可能与下面稍有不同。但只要保证“NodeManager、SecondaryNameNode、NameNode、DataNode、ResourceManager”进程在界面上有显示即可，不用关注是否有其它进程，或者进程前的数字。

[Command 002]:

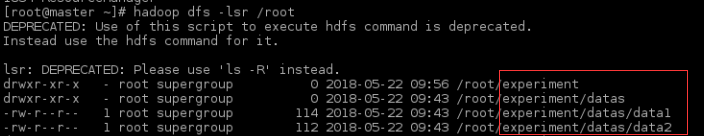
*jps*



1. 实验前数据路径/root/experiment/datas的准备
   * + 1. 数据路径如果已经存在，删除路径下的所有数据文件。命令格式：hadoop dfs -rmr /root/experiment/datas/要删除的文件名。如果要删除的文件名为file.txt，命令为hadoop dfs -rmr /root/experiment/datas/file.txt。
       2. 如果一次要删除多个文件，可用多个文件相同部分的内容+“\*”代替未知内容，例如要删除的文件有二个，名分别为file1.txt、file2.txt，则删除命令可以为命令为hadoop dfs -rmr /root/experiment/datas/file\*或者命令为hadoop dfs -rmr /root/experiment/datas/\*.txt。
       3. 删除已经存在的文件data1、data2

[Command 003]:

*hadoop dfs -lsr /root*

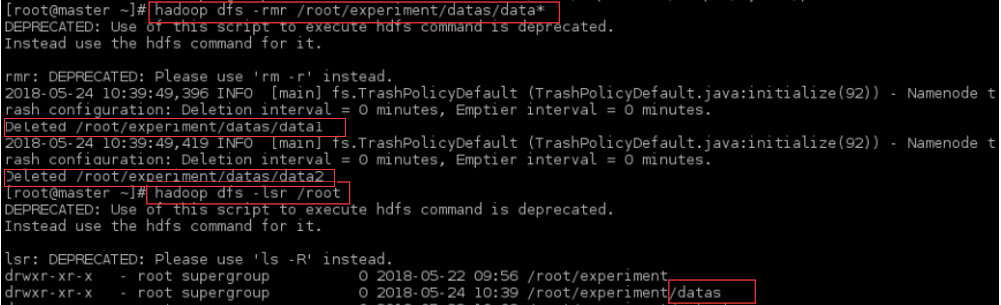


[Command 004]:

*hadoop dfs -rmr /root/experiment/datas/data\**

[Command 005]:

*hadoop dfs -lsr /root*



* + - 1. 如果数据存储路径不存在，在HDFS上建立/root/experiment/datas目录，用于记录程序运行时需要的实验数据。

[Command 006]:

*hadoop dfs -lsr /*

[Command 007]:

*hadoop dfs -mkdir /root*

[Command 008]:

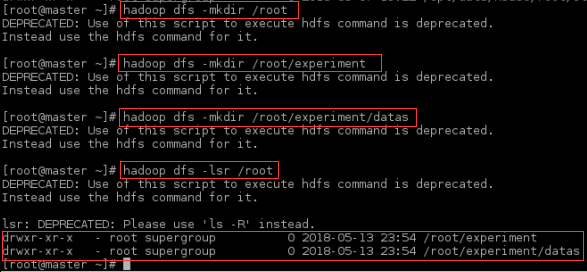
*hadoop dfs -mkdir /root/experiment*

[Command 009]:

*hadoop dfs -mkdir /root/experiment/datas*

[Command 010]:

*hadoop dfs -lsr /root*

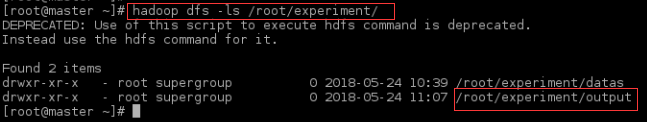


* + - 1. 至此，用于实验数据的HDFS下文件目录准备完成。

1. 实验程序每次执行前，运行结果输出路径的准备
   * + 1. 每次实验程序运行结果会存储在HDFS平台的/root/experiment/output/路径下，所以在程序执行前，该路径不能存在，如果存在，需要进行删除。
       2. 查看/root/experiment/output存在时，查看结果会有显示（见红框的内容）。如果平台有了其它应用，查询结果可能与下图有所不同，仅以红框内容为准备。

[Command 011]:

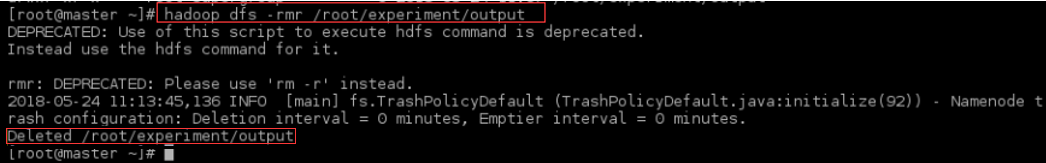
*hadoop dfs -ls /root/experiment/*



* + - 1. 删除已经存在的输出目录。

[Command 012]:

*hadoop dfs -rmr /root/experiment/output*



1. IDEA下建立Maven项目project

知识点

1. IDEA下Maven项目建立过程

实验目的

1. 掌握IDEA下Maven项目建立过程

实验环境

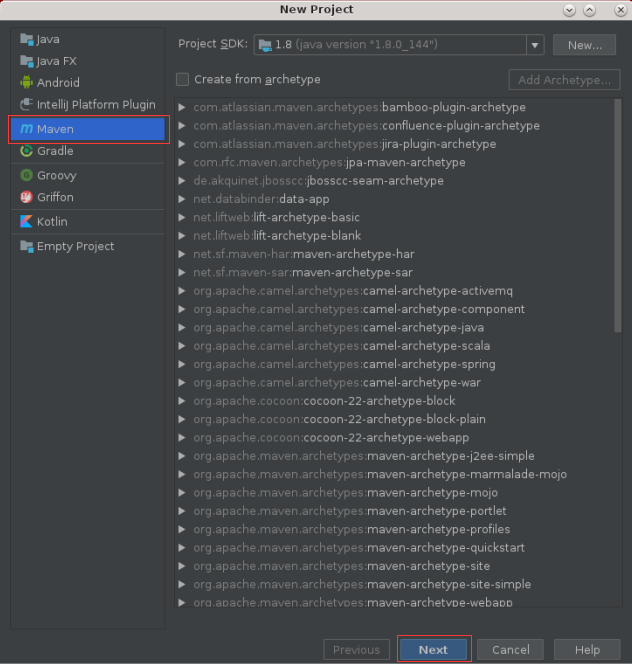
1. Oracle Linux 7.4
2. Hadoop2.7.4

实验步骤

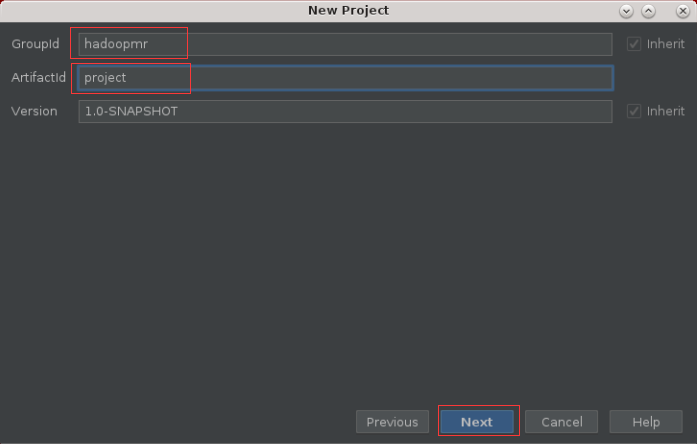
1. 建立新项目
   * + 1. 打开IDEA工具。鼠标双击桌面“IDEA”图标，如果是第一次使用，会弹出“Welcome IntelliJ IDEA”窗口，选择建立新项目条目“Create New Project”，如图所示。



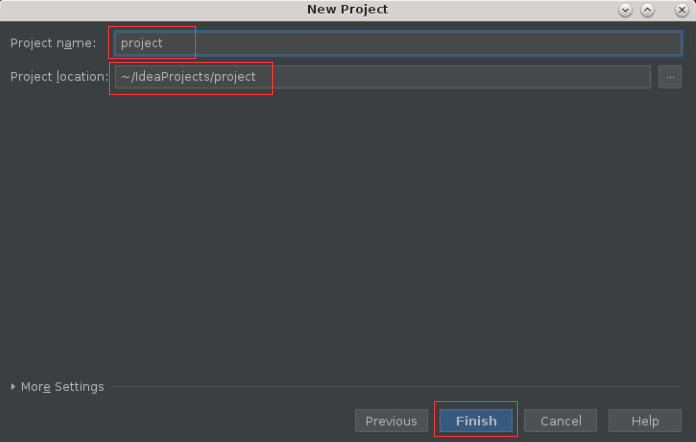
* + - 1. 确定要建立的项目类型。弹出新项目“New Project”窗口，选择要建立的项目类型“Maven”，点击下一步“Next”按钮。



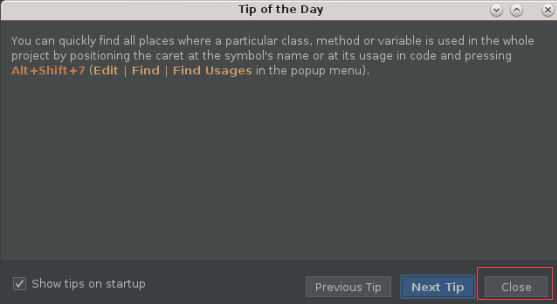
* + - 1. 填入工程信息。在弹出“New Project”窗口中，在GroupId对应文本框中输入“hadoopmr”，在ArtifactId对应文本框中输入“project”，然后按“Next”按钮。



* + - 1. 确认项目信息，完成项目创建。在弹出“New Project”窗口中，会显示新建立的项目的名称（name），及项目的存储位置(location)。点击“Finish”按钮，完成项目的初始建立。



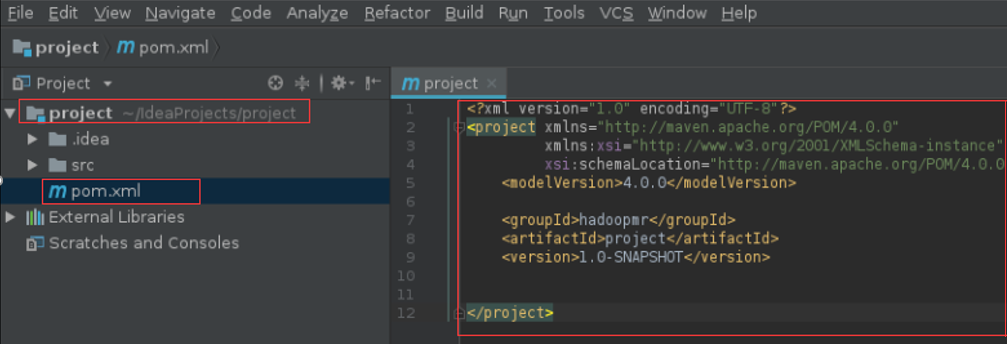
* + - 1. 此时进入IDEA的开发界面。如果在开发界面的上方弹出“Tip of the Day”窗口，点出”Close”按钮，关闭该窗口即可。



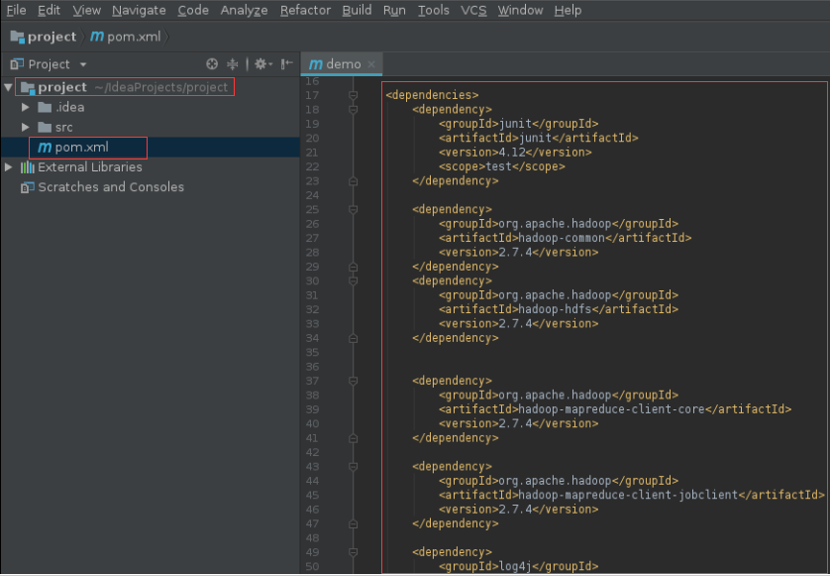
* + - 1. 在右下角弹出的对话框中，选择Enable Auto-Import（如未弹出该对话框请忽略此步骤）。

C:\Users\LENOVO\AppData\Local\Temp\ksohtml\wpsD8BF.tmp.jpg

* + - 1. 显示IDEA开发环境的主窗口。并在窗口左边可以看到新建立的“project”项目。其中pom.xml里记录Maven项目的依赖等。



1. 配置pom.xml文件。如下图。



* + - 1. 其中pom.xml文件更改后所有的文件内容是：

[Code 001]：

*<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">*

*<modelVersion>4.0.0</modelVersion>*

*<groupId>demo</groupId>*

*<artifactId>demo</artifactId>*

*<version>0.0.1-SNAPSHOT</version>*

*<packaging>jar</packaging>*

*<name>demo</name>*

*<url>http://maven.apache.org</url>*

*<properties>*

*<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>*

*</properties>*

*<dependencies>*

*<dependency>*

*<groupId>junit</groupId>*

*<artifactId>junit</artifactId>*

*<version>4.12</version>*

*<scope>test</scope>*

*</dependency>*

*<dependency>*

*<groupId>org.apache.hadoop</groupId>*

*<artifactId>hadoop-common</artifactId>*

*<version>2.7.4</version>*

*</dependency>*

*<dependency>*

*<groupId>org.apache.hadoop</groupId>*

*<artifactId>hadoop-hdfs</artifactId>*

*<version>2.7.4</version>*

*</dependency>*

*<dependency>*

*<groupId>org.apache.hadoop</groupId>*

*<artifactId>hadoop-mapreduce-client-core</artifactId>*

*<version>2.7.4</version>*

*</dependency>*

*<dependency>*

*<groupId>org.apache.hadoop</groupId>*

*<artifactId>hadoop-mapreduce-client-jobclient</artifactId>*

*<version>2.7.4</version>*

*</dependency>*

*<dependency>*

*<groupId>log4j</groupId>*

*<artifactId>log4j</artifactId>*

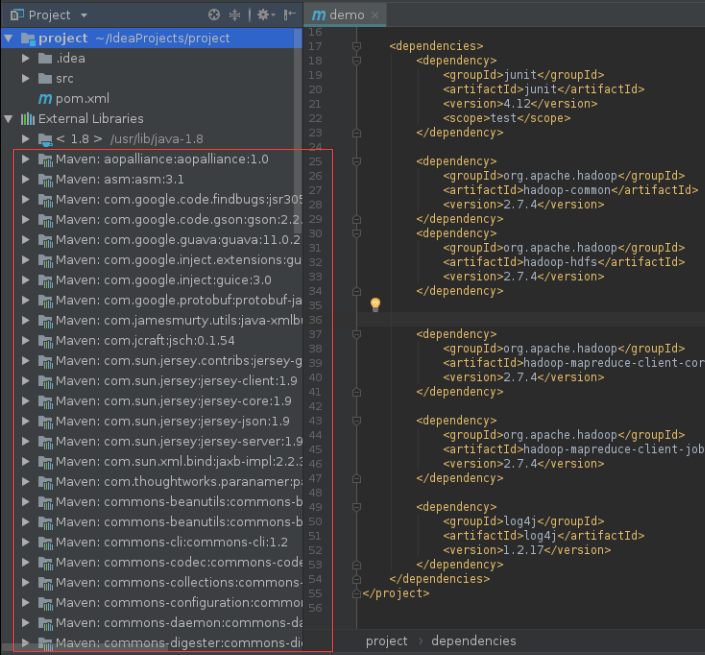
*<version>1.2.17</version>*

*</dependency>*

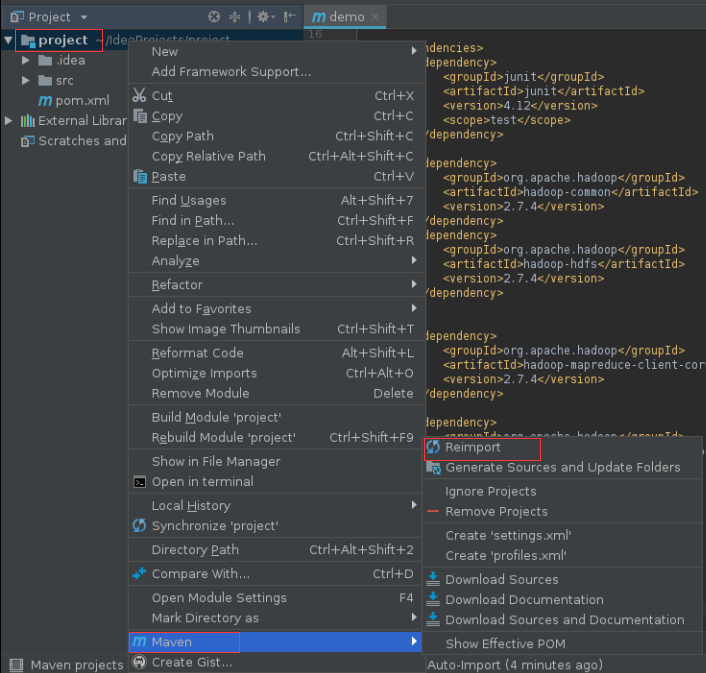
*</dependencies>*

*</project>*

1. 查看Hadoop工程的Maven依赖包
   * + 1. 查看导入依赖包。依赖包导入完成后，会在IDEA左侧窗口中看到新导入的依赖包（下图中显示部分依赖包内容）。



* + - 1. 如果Maven项目依赖包上面方法导入失败。也可以在配置好pom.xml文件内容后，选中项目名“project”,点击鼠标右键，选择“Maven”下的“ReImport”项目，开始Maven项目依赖包的导入。（注意，如果在上一步即1中，如果Maven依赖包导入成功，此步不需要做）



至此，Hadoop开发环境的Maven项目建立完成。

1. WordCount编程

知识点

1. 通过统计文件中每个单词出现的次数，理解Mapper、Reducer编程过程
2. 掌握Job作业调度方法

实验目的

1. 通过WordCount实验理解MapReduce编程过程。
2. 掌握常规Mapper和reducer调度方法。

实验环境

1. Oracle Linux 7.4
2. Hadoop2.7.4
3. IDEA

实验步骤

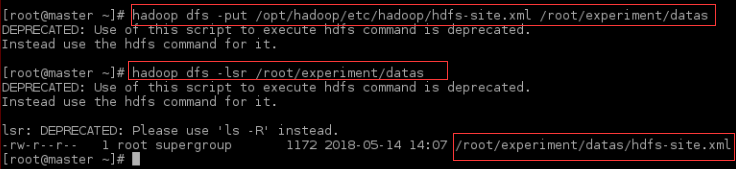
1. 实验数据准备
   * + 1. 将Hadoop安装包下/etc/hadoop/hdfs-site.xml文件复制到HDFS上的/root/experiment/datas目录下。（这里需要注意的是，如果作为统计的数据选择过大的话，要考虑系统的运行环境情况，例如堆的大小，如果堆的大小不能容纳数据量运行，程序运行时会出现不正常现象）。

[Command 013]:

*hadoop dfs -put /opt/hadoop/etc/hadoop/hdfs-site.xml /root/experiment/datas*

[Command 014]:

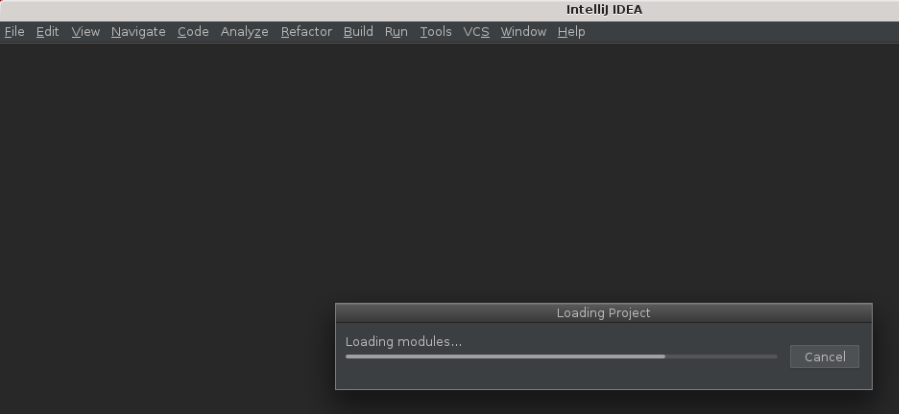
*hadoop dfs -lsr /root*

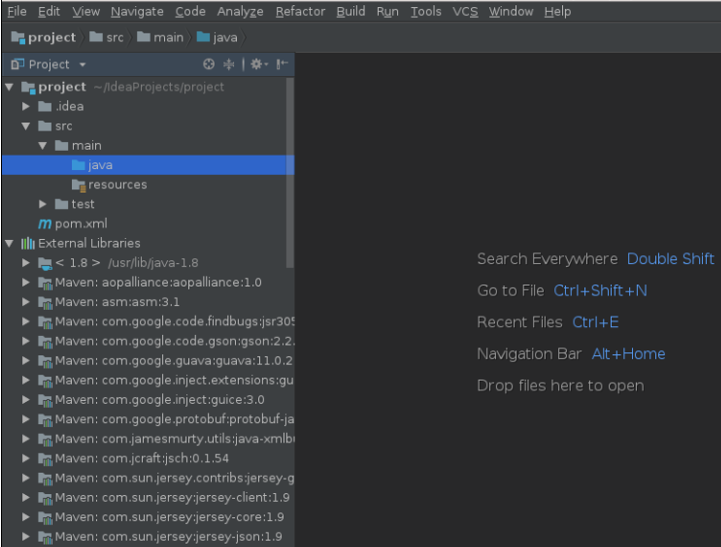


* + - 1. 至此，用于WordCount计算的实验数据准备完成。

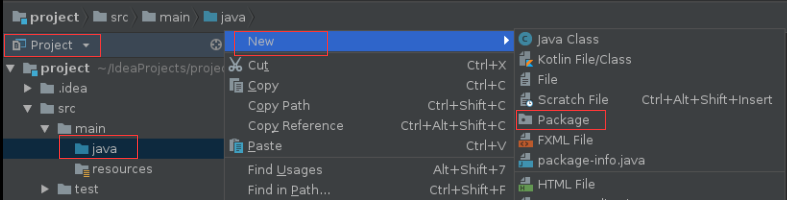
1. 打开IDEA，建立包名，用于在该包下编写WordCount程序
   * + 1. 鼠标双击桌面“IDEA”图标，打开IDEA开发环境界面。MapReduce程序将会写在开发项目中的Java文件夹下。



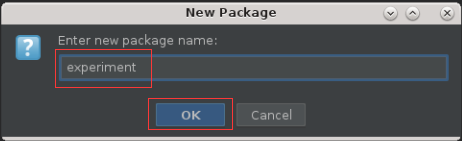




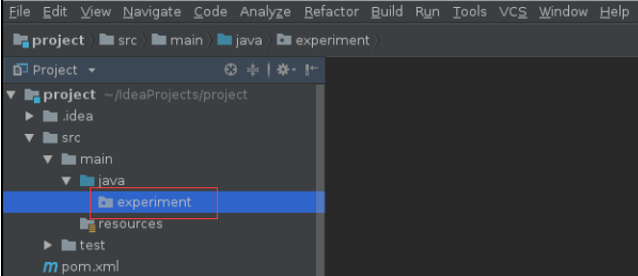
* + - 1. 选中“Project”，找到它下面的”Java”,鼠标右键选择它的子项“New”下的“Package”。



* + - 1. 弹出“New Package”窗口，在文本框中写上包名“experiment”，点击“OK”按钮，完成包名创建。

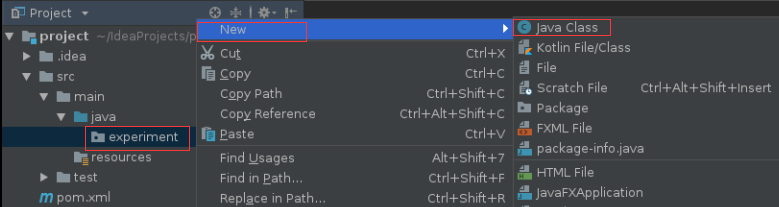


* + - 1. 此时，在IDEA工具的左侧窗口中的project项目下，可以看见新创建的包名。

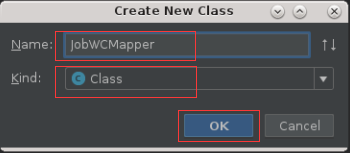


* + - 1. 至此，包名建立完毕。下面可以开始进行MapReduce程序的编写工作了。

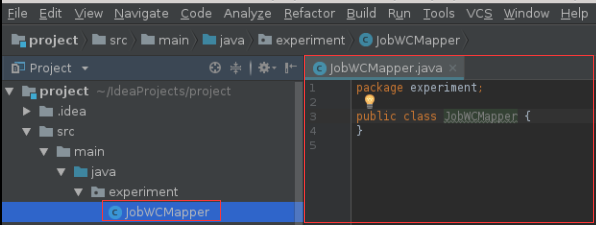
1. 建立JobWCMapper类文件，编写Mapper类。
   * + 1. 选中新创建的包（Package）名“experiment”,鼠标右键选择它的子项“New”下的“Java Class”选项。



* + - 1. 在弹出的窗口“Create New Class”中输入要建立的文件名字“JobWCMapper”和类型“Class”，然后点击“OK”按钮，完成类文件的创建。



* + - 1. 此时在IDEA工具左侧窗口中的项目Project中包“experiment”下会看到新建立的类文件JobWCMapper，并且在IDEA工具的中央窗口中会看见新建立的类文件的内容，我们就在该窗口中编写JobWCMapper类的程序代码。



* + - 1. 具体JobWCMapper类的程序代码如下：

[Code 002]：

*package experiment;*

*import java.io.IOException;*

*import java.util.StringTokenizer;*

*import org.apache.hadoop.io.IntWritable;*

*import org.apache.hadoop.io.LongWritable;*

*import org.apache.hadoop.io.Text;*

*import org.apache.hadoop.mapreduce.Mapper;*

*public class JobWCMapper extends Mapper<LongWritable, Text, Text, IntWritable> {*

*private final static IntWritable one = new IntWritable(1);*

*private Text word = new Text();*

*public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {*

*StringTokenizer itr = new StringTokenizer(value.toString());*

*while (itr.hasMoreTokens()) {*

*word.set(itr.nextToken());*

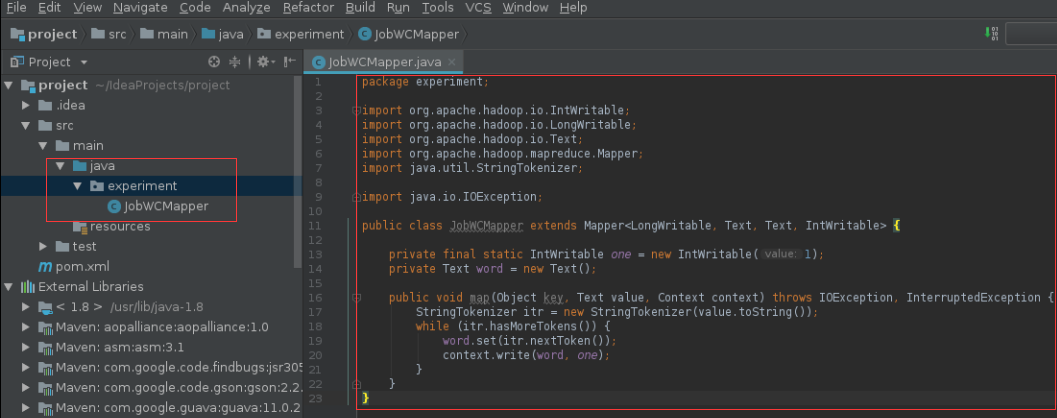
*context.write(word, one);*

*}*

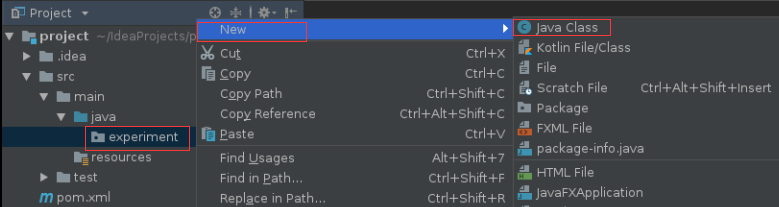
*}*

*}*

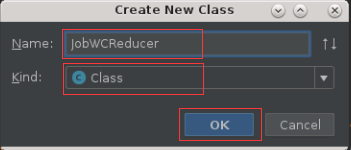
* + - 1. 写完代码后的窗口界面如下：



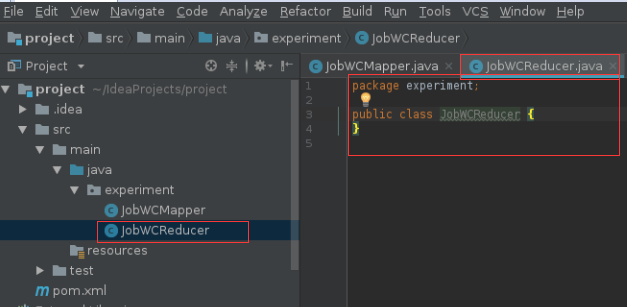
1. 建立JobWCReducer类文件，编写Reducer类。
   * + 1. 编写过程与Mapper类文件相似。首先，选中新创建的包（Package）名“experiment”,鼠标右键选择它的子项“New”下的“Java Class”。



* + - 1. 在弹出的窗口“Create New Class”中输入要建立的文件名字“JobWCReducer”和类型“Class”，然后点击“OK”按钮，完成Reducer类文件的创建。



* + - 1. 此时在IDEA工具左侧窗口项目Project中包“experiment”下会看到新建立的类文件，并且在IDEA工具的中央窗口中会看见新建立类文件“JobWCReducer”的内容，我们在该窗口中编写JobWCReducer类的程序代码。



* + - 1. 具体JobWCReducer类的程序代码如下：

[Code 003]：

*package experiment;*

*import java.io.IOException;*

*import org.apache.hadoop.io.IntWritable;*

*import org.apache.hadoop.io.Text;*

*import org.apache.hadoop.mapreduce.Reducer;*

*public class JobWCReducer extends Reducer<Text, IntWritable, Text, IntWritable> {*

*protected void reduce(Text key, Iterable<IntWritable> values,Context context) throws IOException, InterruptedException {*

*int sum = 0;*

*for (IntWritable val : values) {*

*sum += val.get();*

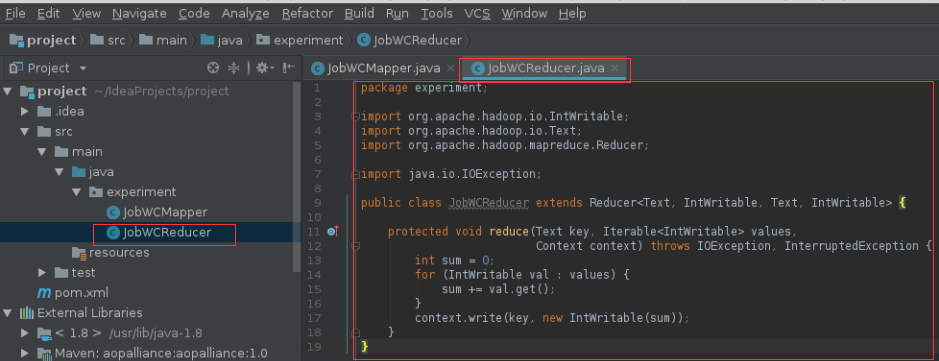
*}*

*context.write(key, new IntWritable(sum));*

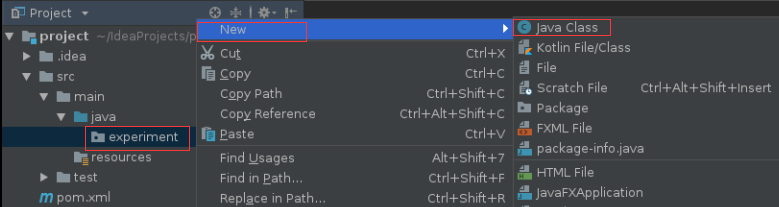
*}*

*}*

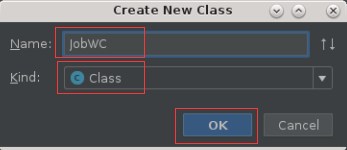
* + - 1. 写完代码后的窗口界面如下：



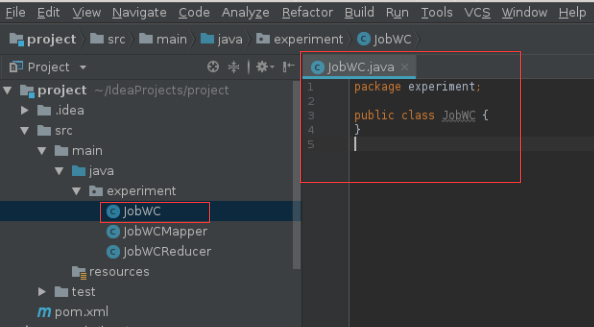
1. 编写主函数，通过Job方法启动Mapper、Reducer作业。
   * + 1. 选中新创建的包（Package）名“experiment”，鼠标右键选择它的子目录“New”下的“Java Class”。



* + - 1. 在弹出的窗口“Create New Class”中输入要建立的文件的名字“JobWC”和类型“Class”，然后点击“OK”按钮，完成类文件的创建。该类文件用于调度MapperReducer的作业程序。常规的方法Hadoop老版本中可通过JobConf去调度，新版本中它仍然可用，但更推荐采用Job调度。对于MapReduce程序的调度也可能通过Tool工具完成。本次实验只介绍常用的Job调度的写法。



* + - 1. 此时在IDEA工具左侧窗口项目Project中包“experiment”下会看到新建立的类文件“JobWC”，并且在IDEA工具的中央窗口中会看见新建立的类文件“JobWC”的类文件内容，我们在该窗口中编写JobWC类的程序代码。



* + - 1. 具体JobWC类的程序代码如下：

[Code 004]：

*package experiment;*

*import org.apache.hadoop.conf.Configuration;*

*import org.apache.hadoop.fs.Path;*

*import org.apache.hadoop.io.IntWritable;*

*import org.apache.hadoop.io.Text;*

*import org.apache.hadoop.mapreduce.Job;*

*import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;*

*import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;*

*import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;*

*import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;*

*public class JobWC {*

*public static void main(String[] args) throws Exception {*

*Path file = new Path("hdfs://master:9000/root/experiment/datas");*

*Path outfile = new Path("hdfs://master:9000/root/experiment/output ");*

*Configuration conf = new Configuration();*

*Job job = Job.getInstance(conf);*

*job.setJarByClass(JobWC.class);*

*job.setJobName("wordcount1");*

*job.setOutputKeyClass(Text.class);*

*job.setOutputValueClass(IntWritable.class);*

*job.setMapperClass(JobWCMapper.class);*

*job.setReducerClass(JobWCReducer.class);*

*job.setInputFormatClass(TextInputFormat.class);*

*job.setOutputFormatClass(TextOutputFormat.class);*

*FileInputFormat.addInputPath(job, file);*

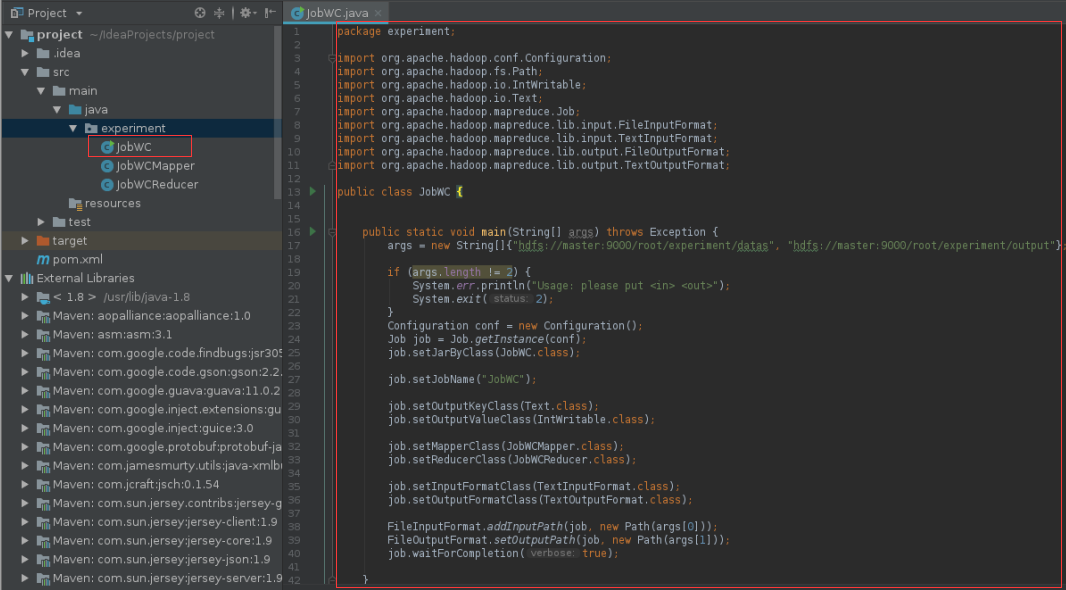
*FileOutputFormat.setOutputPath(job, outfile);*

*job.waitForCompletion(true);*

*}*

*}*

* + - 1. 写完代码后的窗口界面如下：



1. 运行程序与结果验证

知识点

1. IDEA下Maven项目中MapReduce程序运行过程
2. HDFS平台上查验结果

实验目的

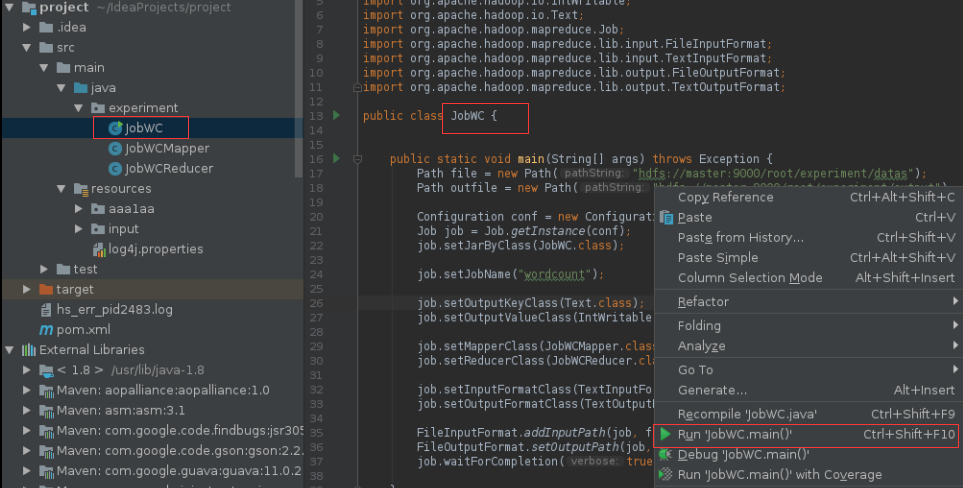
1. 掌握MapReduce程序的运行方法

实验环境

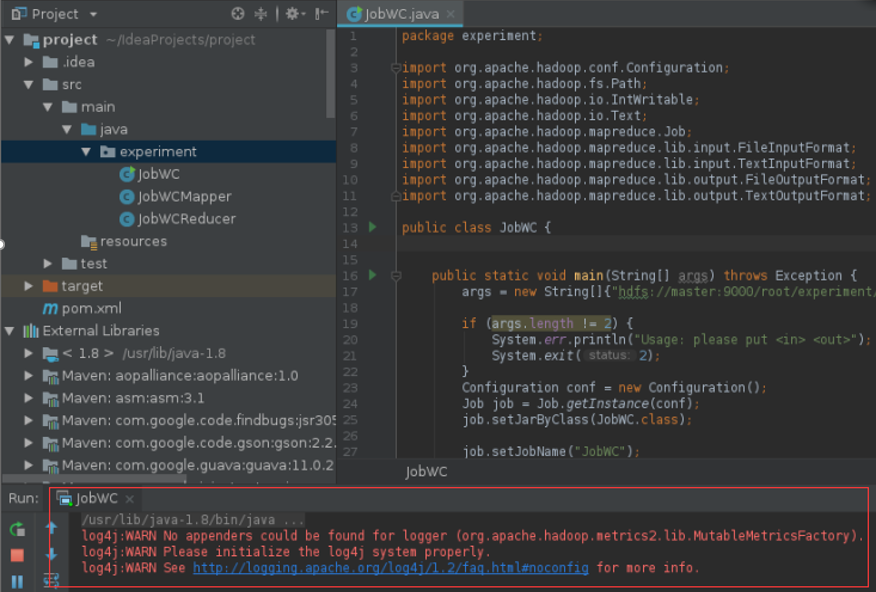
1. Oracle Linux 7.4
2. Hadoop2.7.4

实验步骤

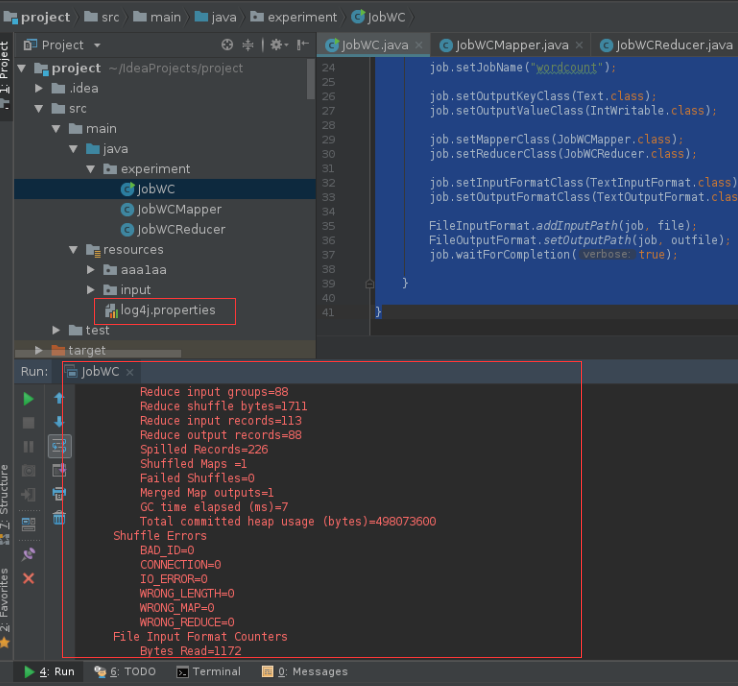
1. IDEA下Maven项目中MapReduce程序运行过程
   * + 1. 在IDEA工具下刚建立完的项目project下打开要运行的程序“JobWC”，并在窗口中点击鼠标右键，在弹出的窗口中点击“Run jobWC.main()”选项，开始运行JobWC中main方法中调用的MapReduce程序。（注意，如果多次运行，每次运行前请将输出目录"hdfs://master:9000/root/experiment/output"，删除后运行，或者将输出目录更名为HDFS上不存在的目录）



* + - 1. 运行时会在IDEA工具下的控制台显示运行的结果。



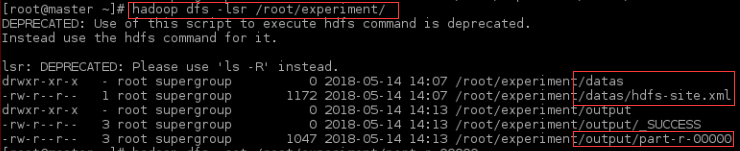
* + - 1. 如果需要观察程序的运行过程，可将Linux平台下/opt/hadoop/etc/hadoop/路径中的log4j.properties文件拷贝至IDEA工具下项目project中的resources下。运行后，会在控制台中显示MapReduce过程的描述。例如启动的JobID，一共启动了多个少Mapper等等信息。



1. HDFS中验证运行结果
   * + 1. 查询运行结果。其中datas下是计算用的数据，而output下的part-r-00000是运行结果。

[Command 015]:

*hadoop dfs -lsr /root/experiment/*



* + - 1. 查询运行结果中的内容（注意：图中显示的是部分结果）。

[Command 016]:

*hadoop dfs -cat /root/experiment/output/part-r-00000*

