实验四 用Java拷贝文件到HDFS

**·实验目标：**

通过该实验后，可以熟练使用eclipse创建Maven项目，并能使用java代码开发读写HDFS系统文件的程序。

1. 使用eclipse建立Maven项目
2. 使用java代码开发程序访问HDFS系统里文件
3. 使用java代码开发程序将本地文件copy到HDFS系统里

**·实验环境：**

HDFS集群环境、可以使用hdfs shell命令、eclipse开发工具

**·实验步骤：**

1.1、启动hadoop服务

使用ssh工具登录主服务器（server-01），到hadoop安装的目录下格式化namenode，再重新启动hadoop集群。（启动成功后查看服务是否启动成功）

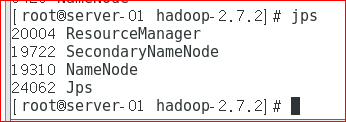
|  |
| --- |
| cd /usr/local/zhitu/hadoop-2.7.2  bin/hdfs namenode -format  Sbin/./start-all.sh |

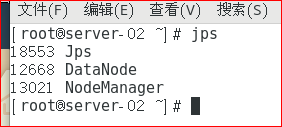
00

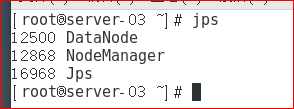
02

00

查看服务是否启动成功：







2、使用eclipse新建Maven项目

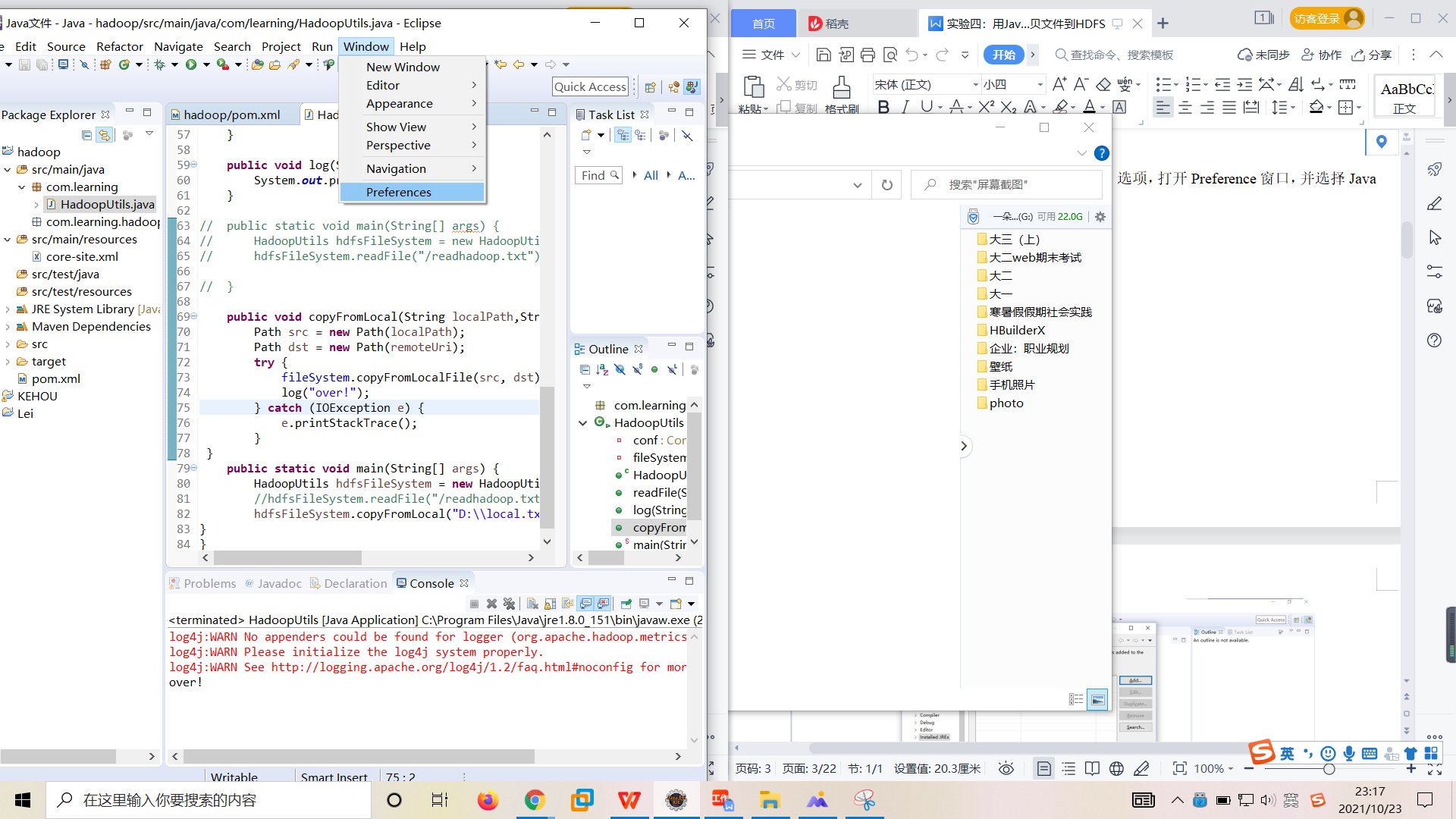
2.1、打开eclipse开发工具

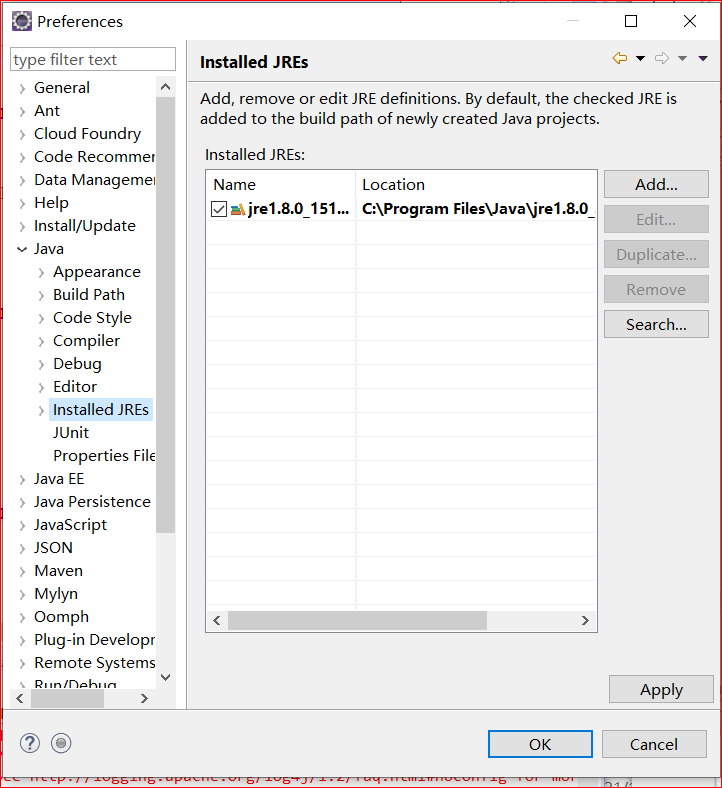
在分配的develop-pc上打开Eclipse开发工具(第一次打开的时候会要求选择一个工作目录，按默认目录即可)

2.2、设置JDK

如果Eclipse已经设置过jdk可省略此步。

选择Eclipse菜单栏windows菜单下的Preference 选项，打开Preference窗口，并选择Java下面的Installed JRE

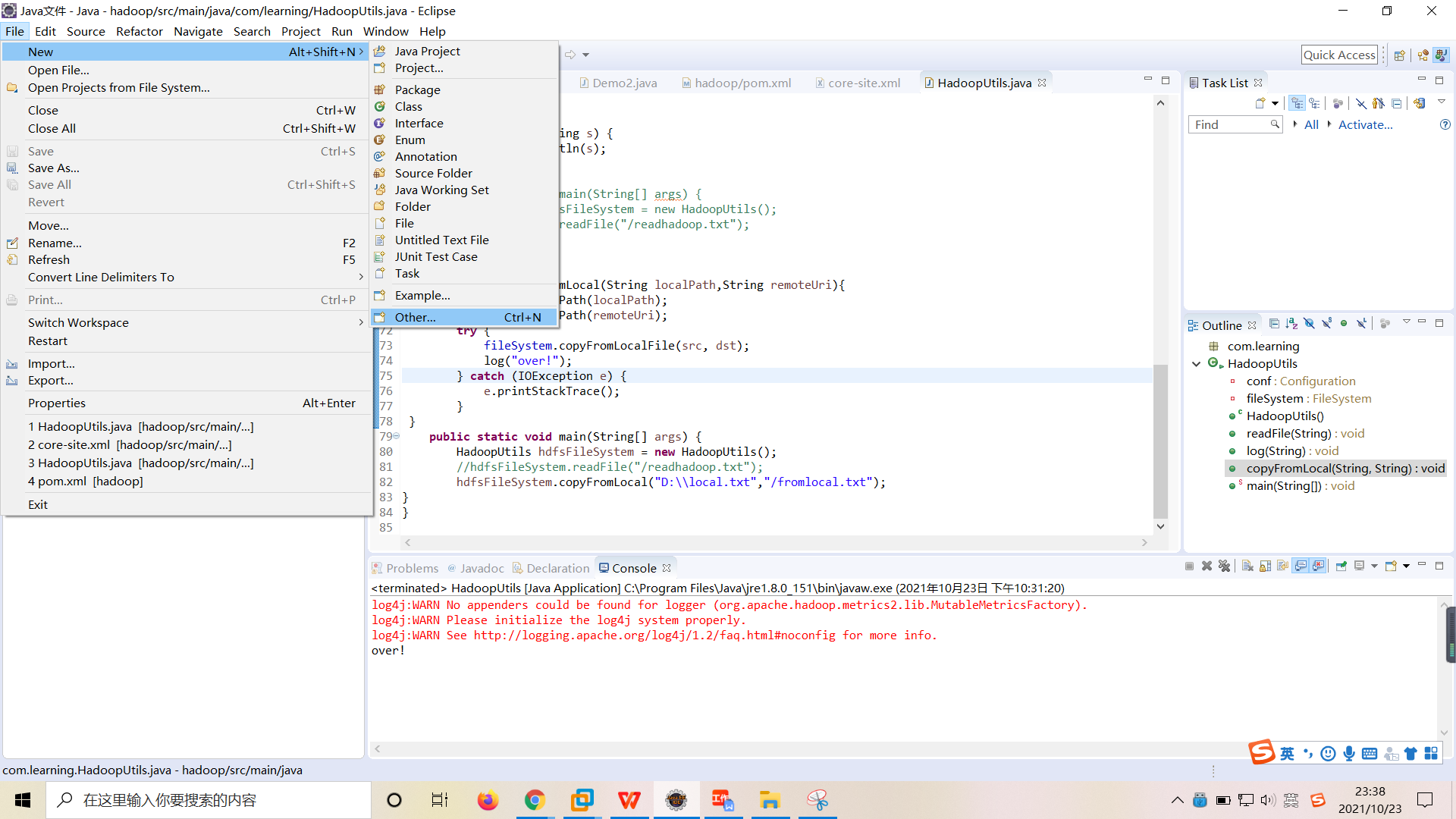


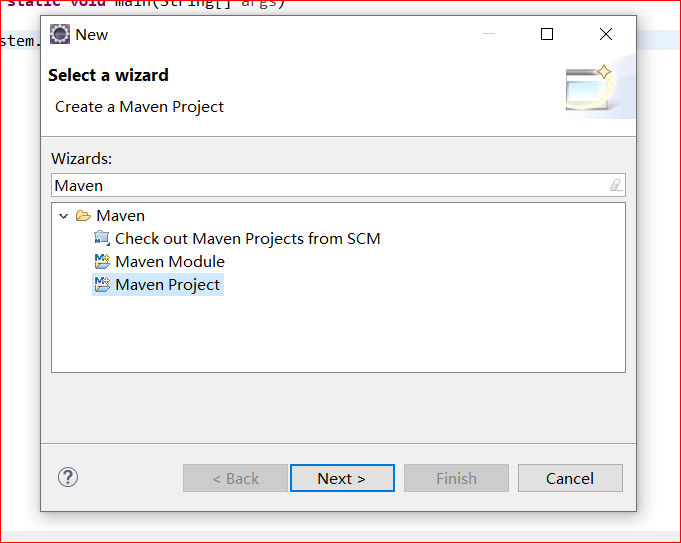


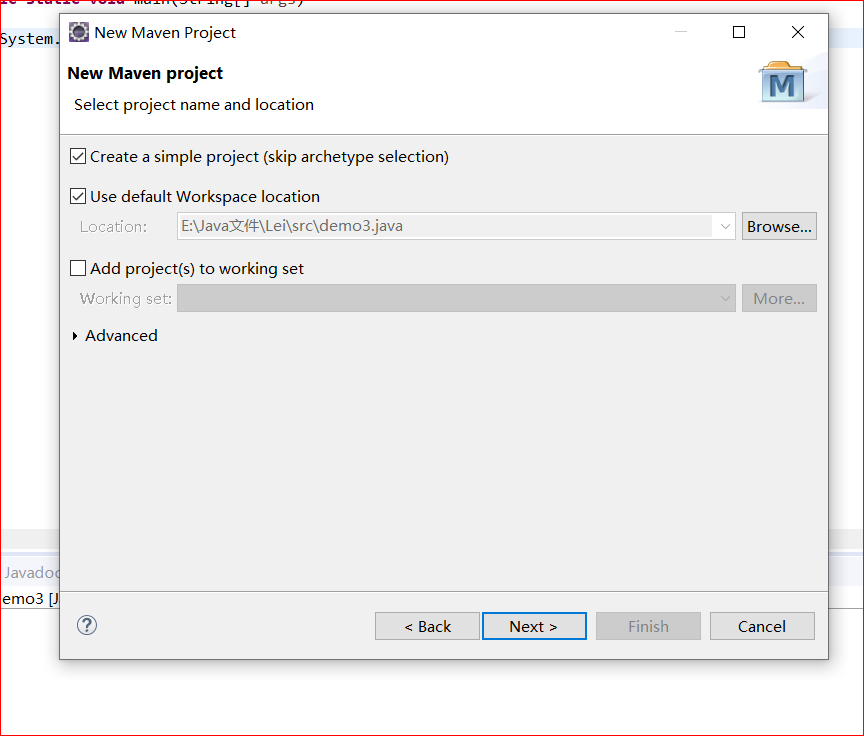
2.3、新建Maven项目

选择Eclipse菜单栏 File菜单里的New 下的Maven Project 选项

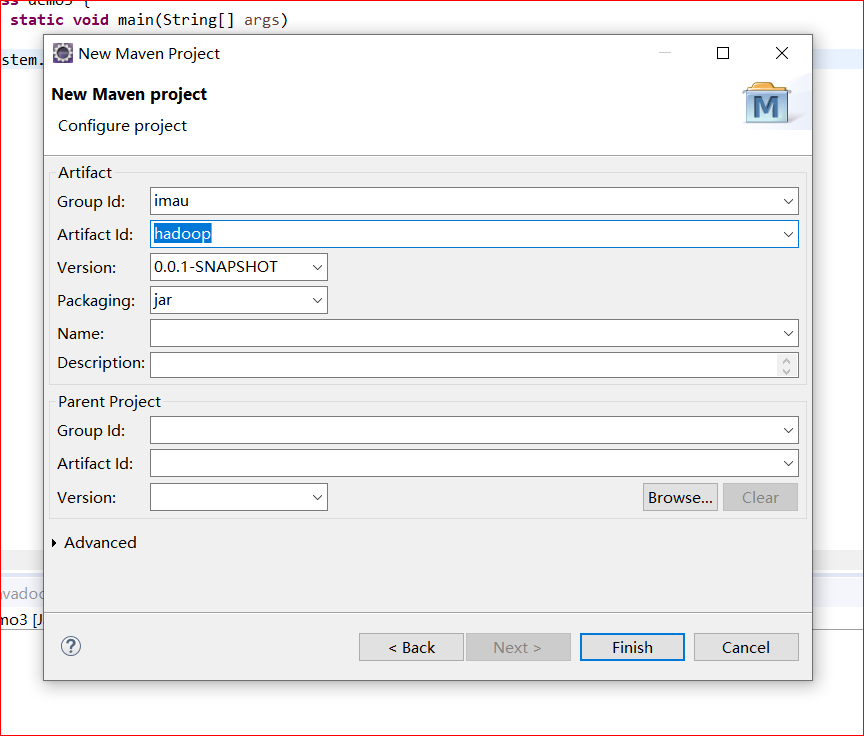
点击file然后打开other，创建Maven项目：



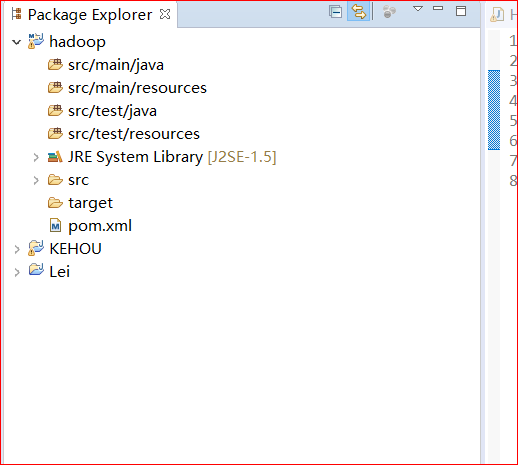




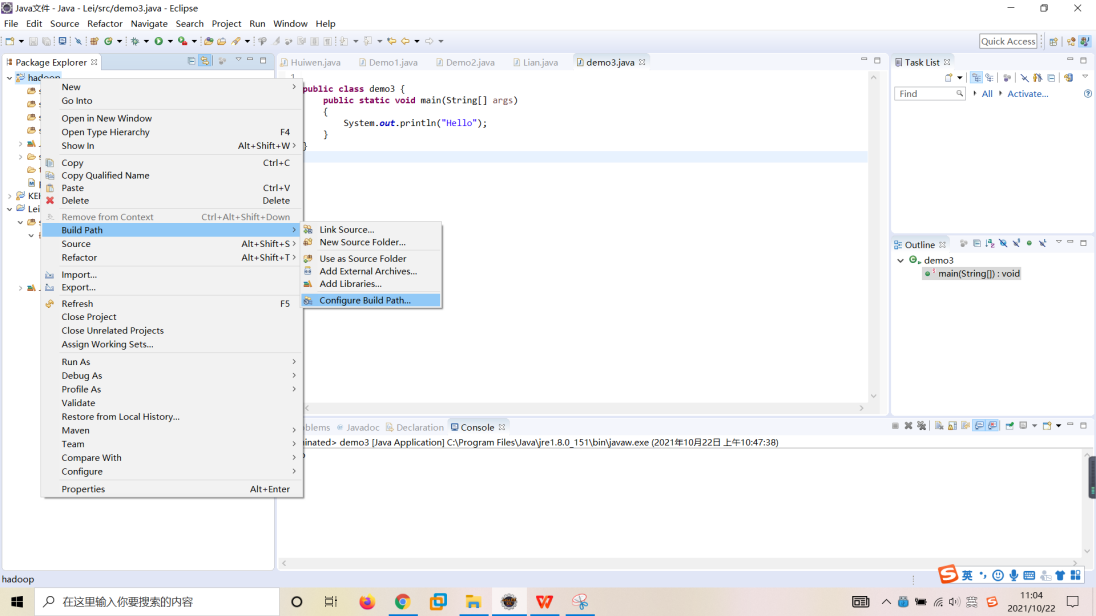
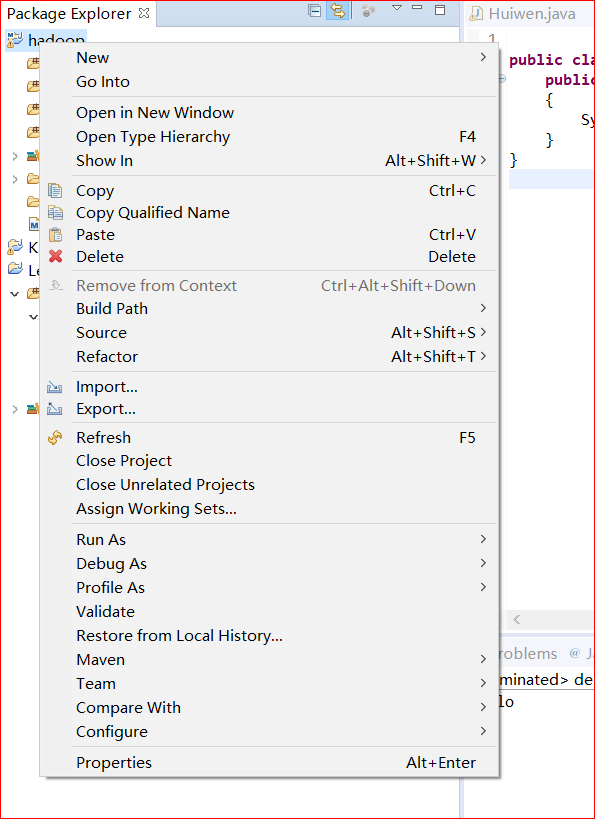
点击next按钮，在弹出来的窗口中，Group id （一般为公司组织名称）填写 :imau，Artifact Id(项目名称) 填写:hadoop ，然后点击finish按钮。



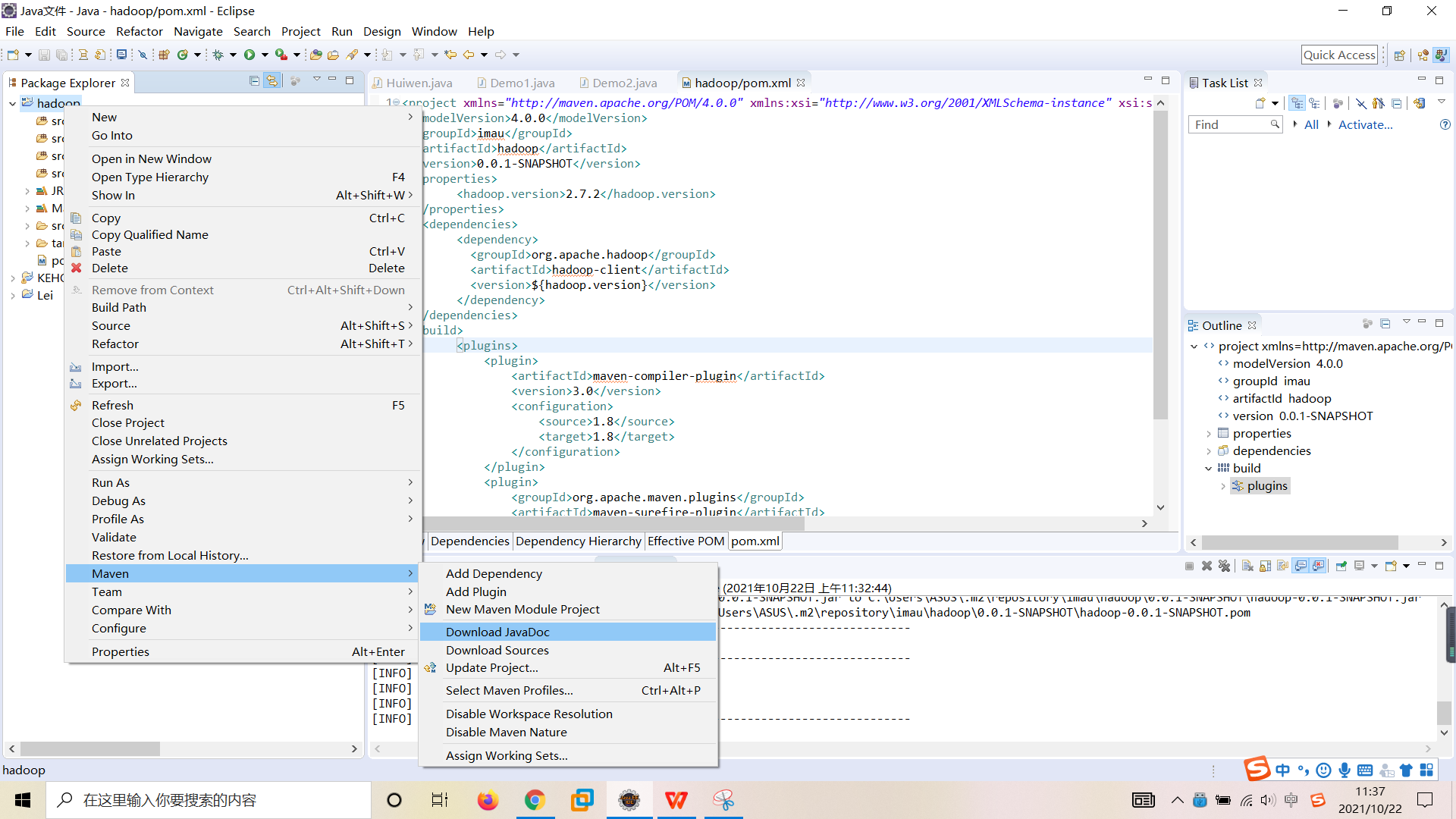
创建好的项目如图：



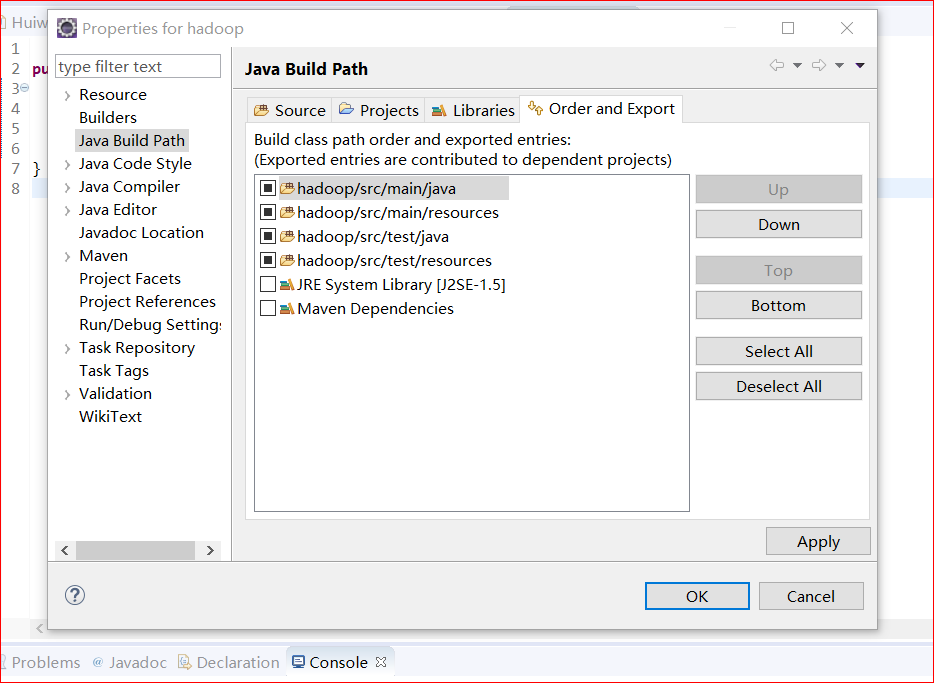
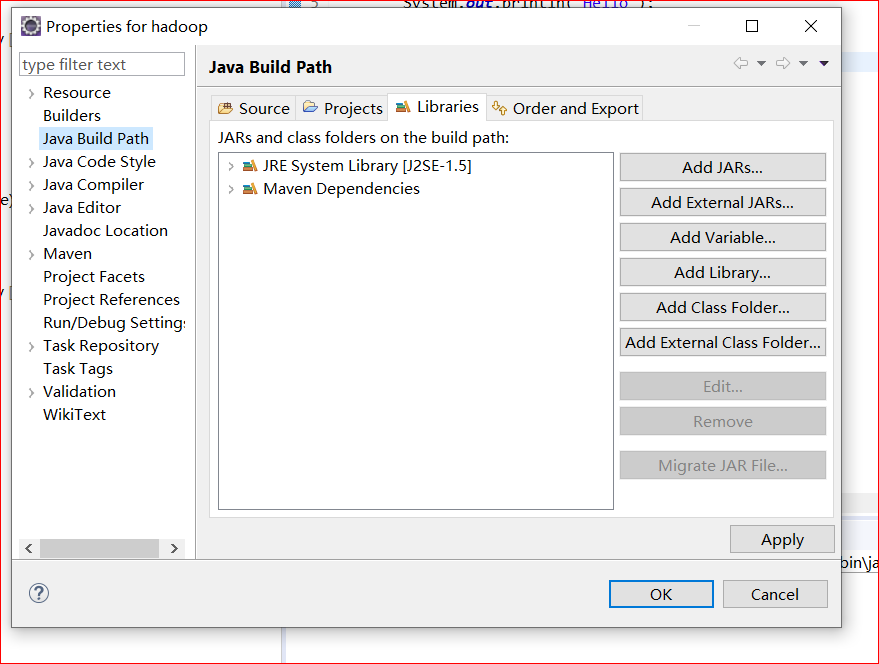
在创建好的hadoop项目上单击右键，选择选择Build Path 🡪 Configure Build Path...

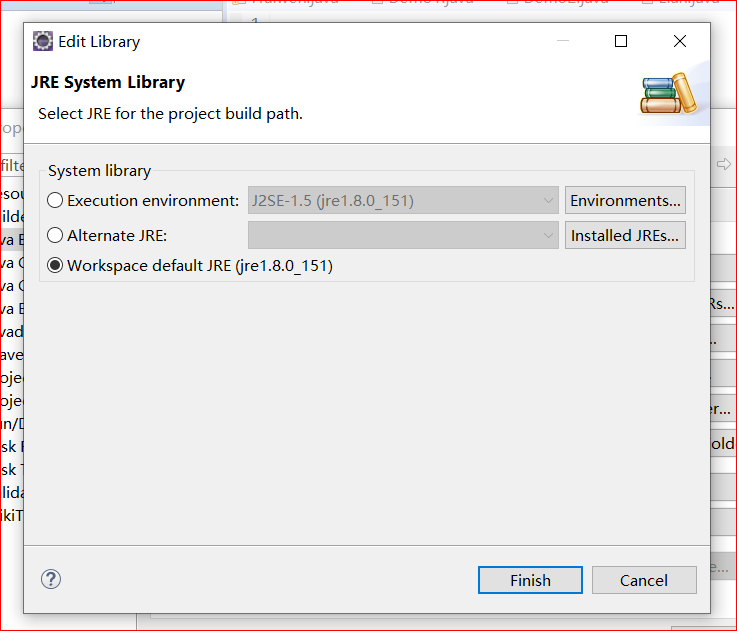


（\*如果没有Build Path ,需要在hadoop右键，选择Maven 🡪 Update Project\*）

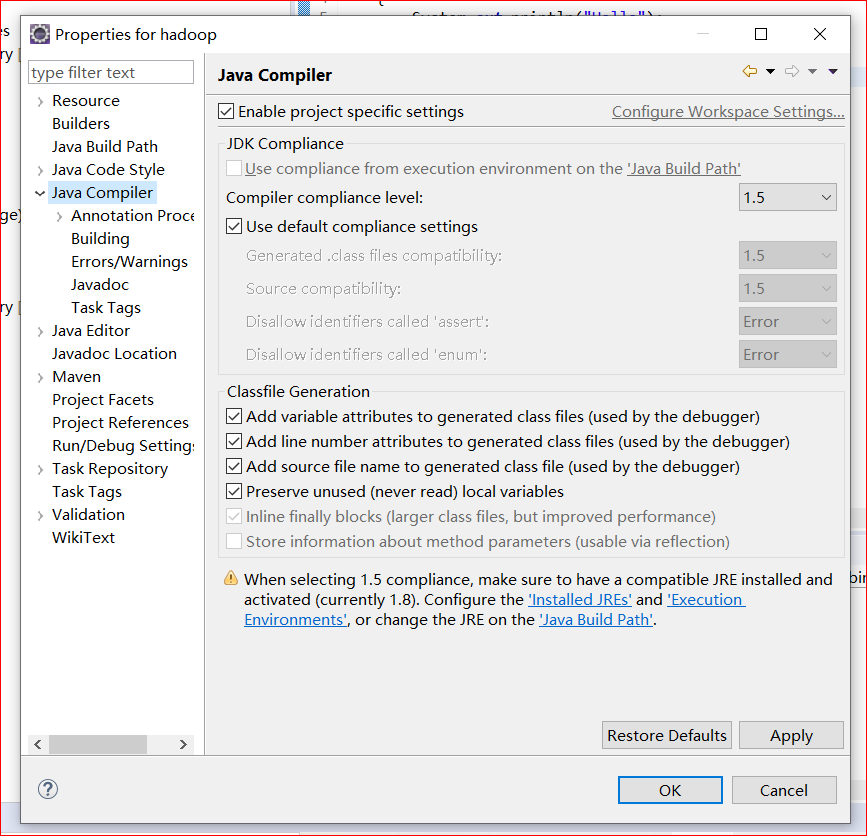
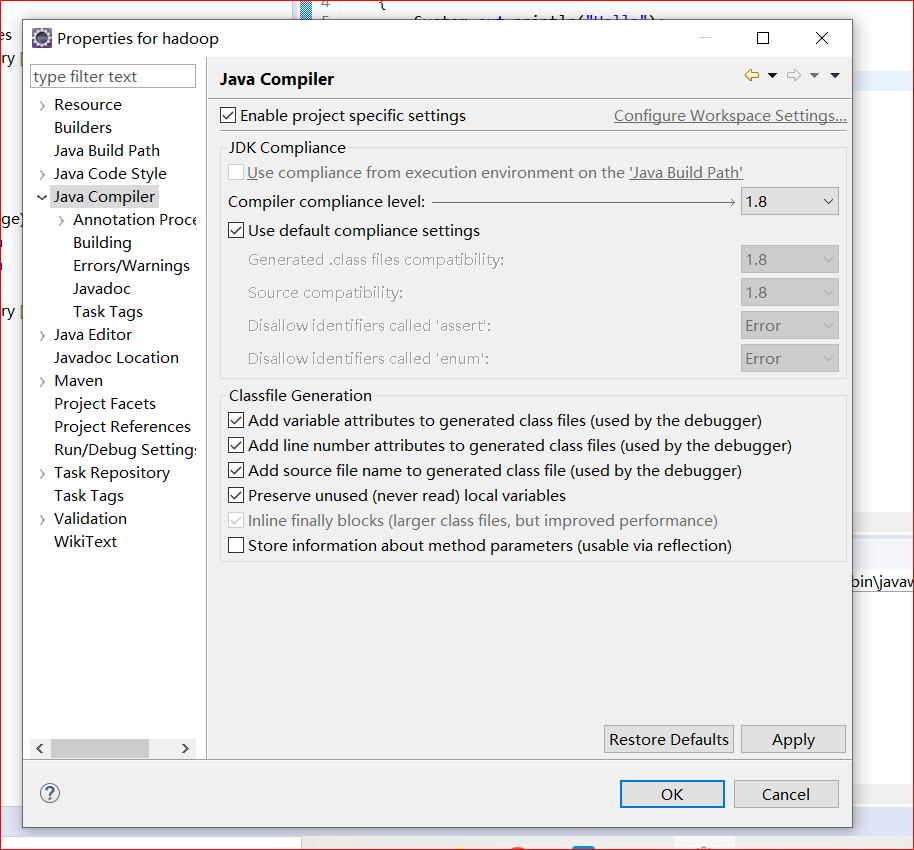


在弹出的窗口中左边窗口选择Java Build Path,右边选择Libraries 下的JRE System Libray选项后，点击 Edit 按钮。

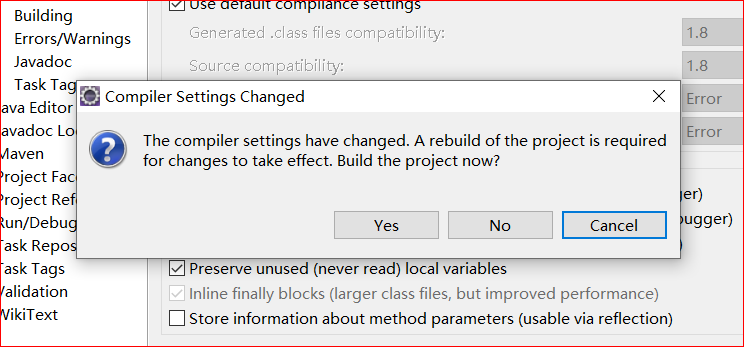
 

****

点击Finish按钮，然后将设置保存。再选择左边Java Compiler选项，将右边的Compiler compliance level: 设置为1.8。

点击OK按钮，并点击弹窗的yes按钮。

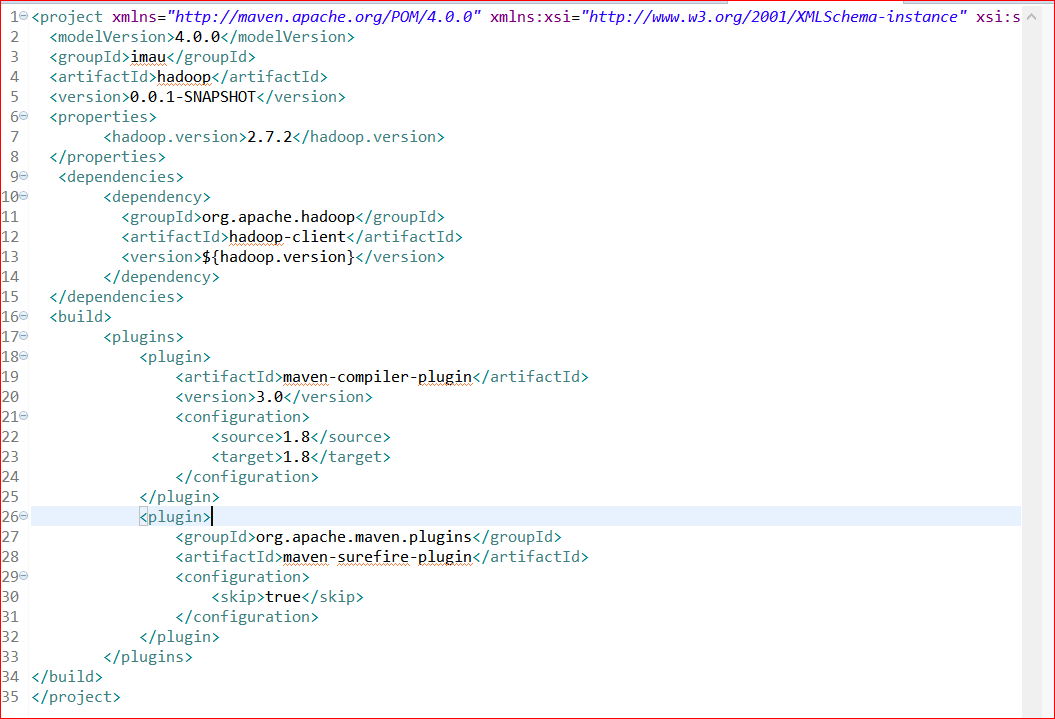


3、编写pom.xml文件

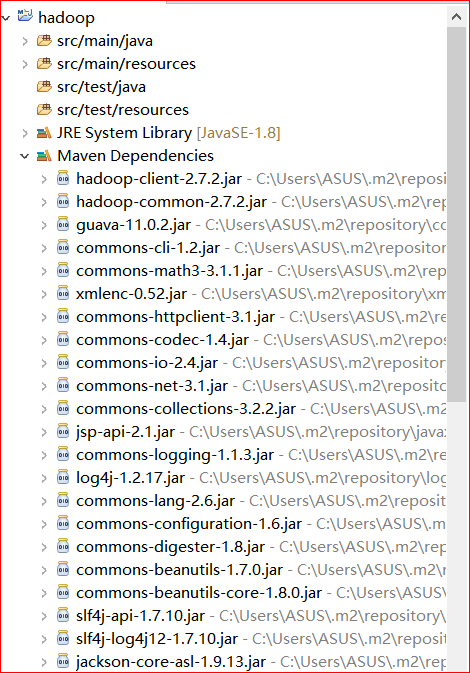
3.1、pom.xml文件主要设置项目的编译环境和依赖的jar包。

打开pom.xml文件，增加如下内容：

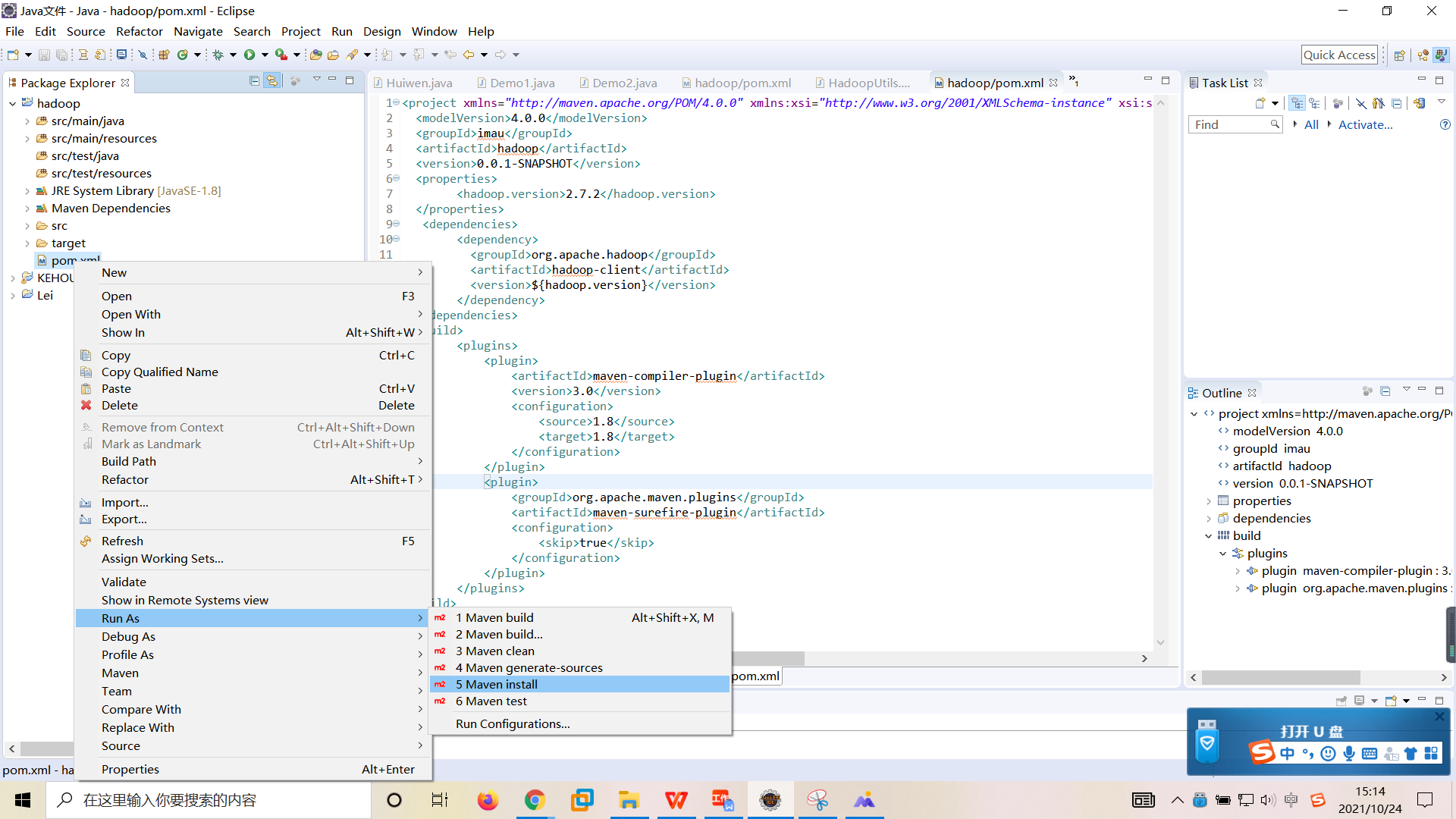
|  |
| --- |
| <properties>  <hadoop.version>2.7.2</hadoop.version>  </properties>  <dependencies>  <dependency>  <groupId>org.apache.hadoop</groupId>  <artifactId>hadoop-client</artifactId>  <version>${hadoop.version}</version>  </dependency>  </dependencies>  <build>  <plugins>  <plugin>  <artifactId>maven-compiler-plugin</artifactId>  <version>3.0</version>  <configuration>  <source>1.8</source>  <target>1.8</target>  </configuration>  </plugin>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-surefire-plugin</artifactId>  <configuration>  <skip>true</skip>  </configuration>  </plugin>  </plugins>  </build>  </project> |



保存之后，eclipse会自动下载依赖jar包，如下图：

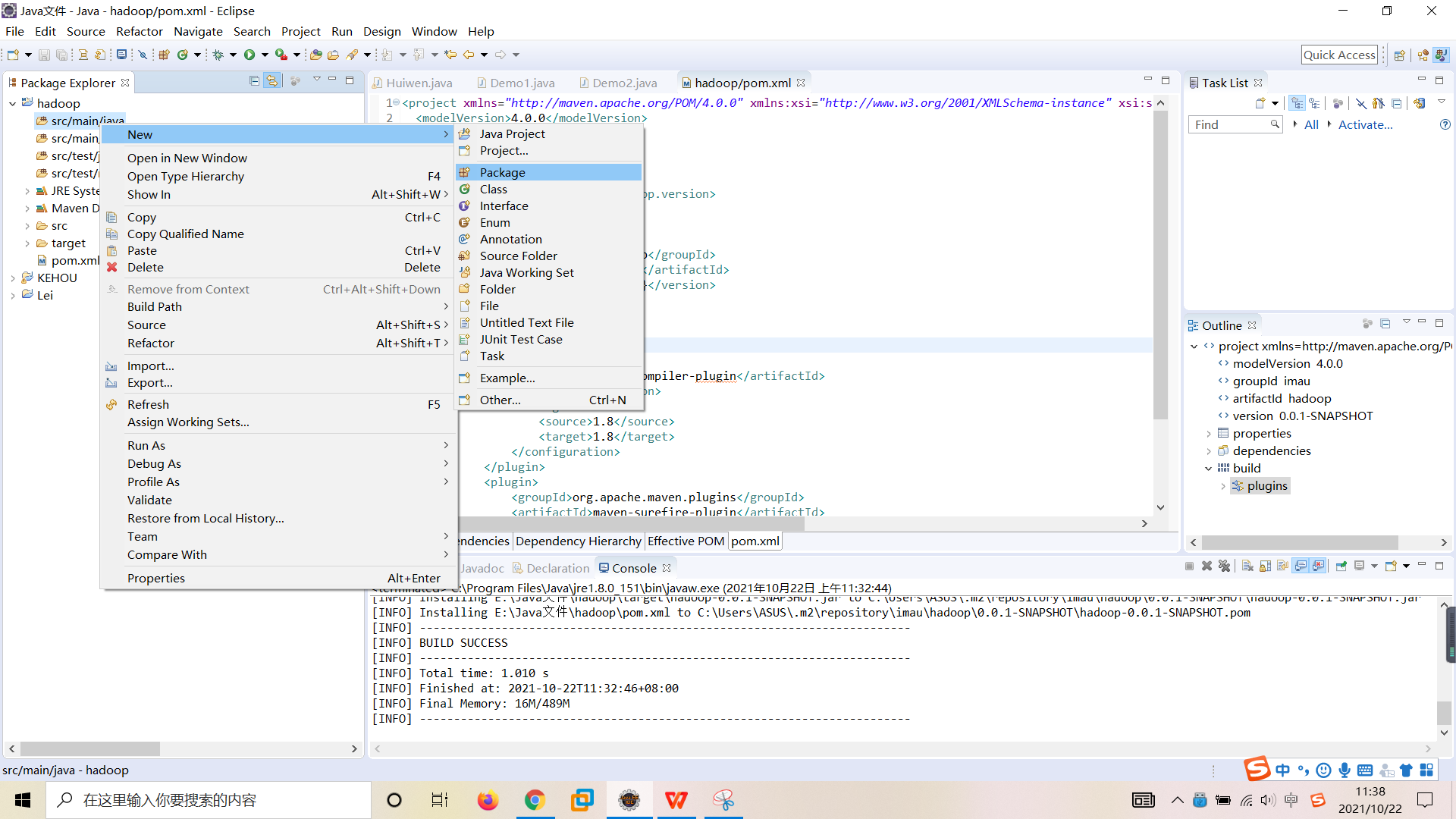


如果没有自动下载，可以在pom.xml文件上单击右键选择Run As 🡪Maven install来下载依赖jar包。

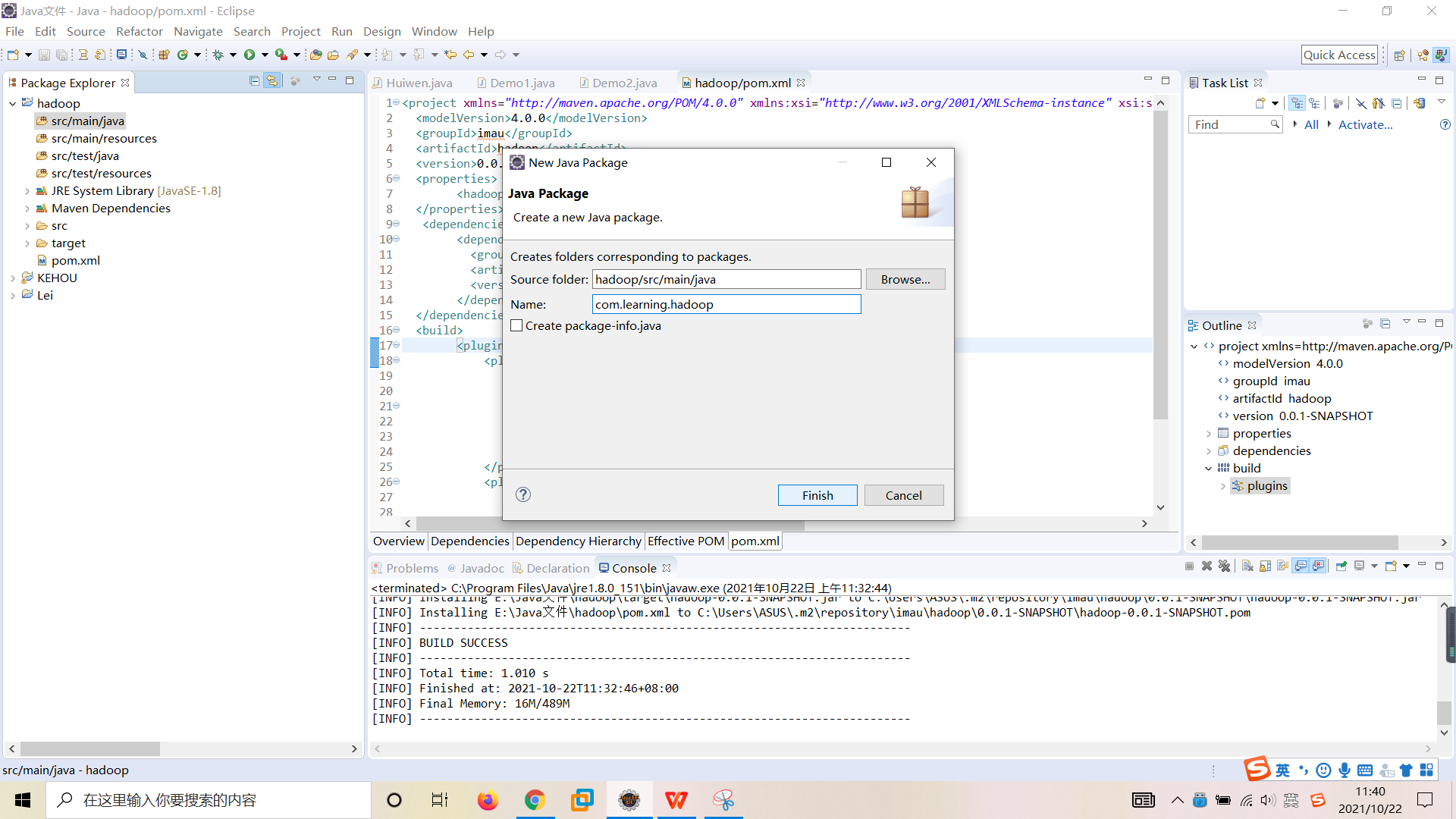


1. 开发读取hdfs系统里的文件代码

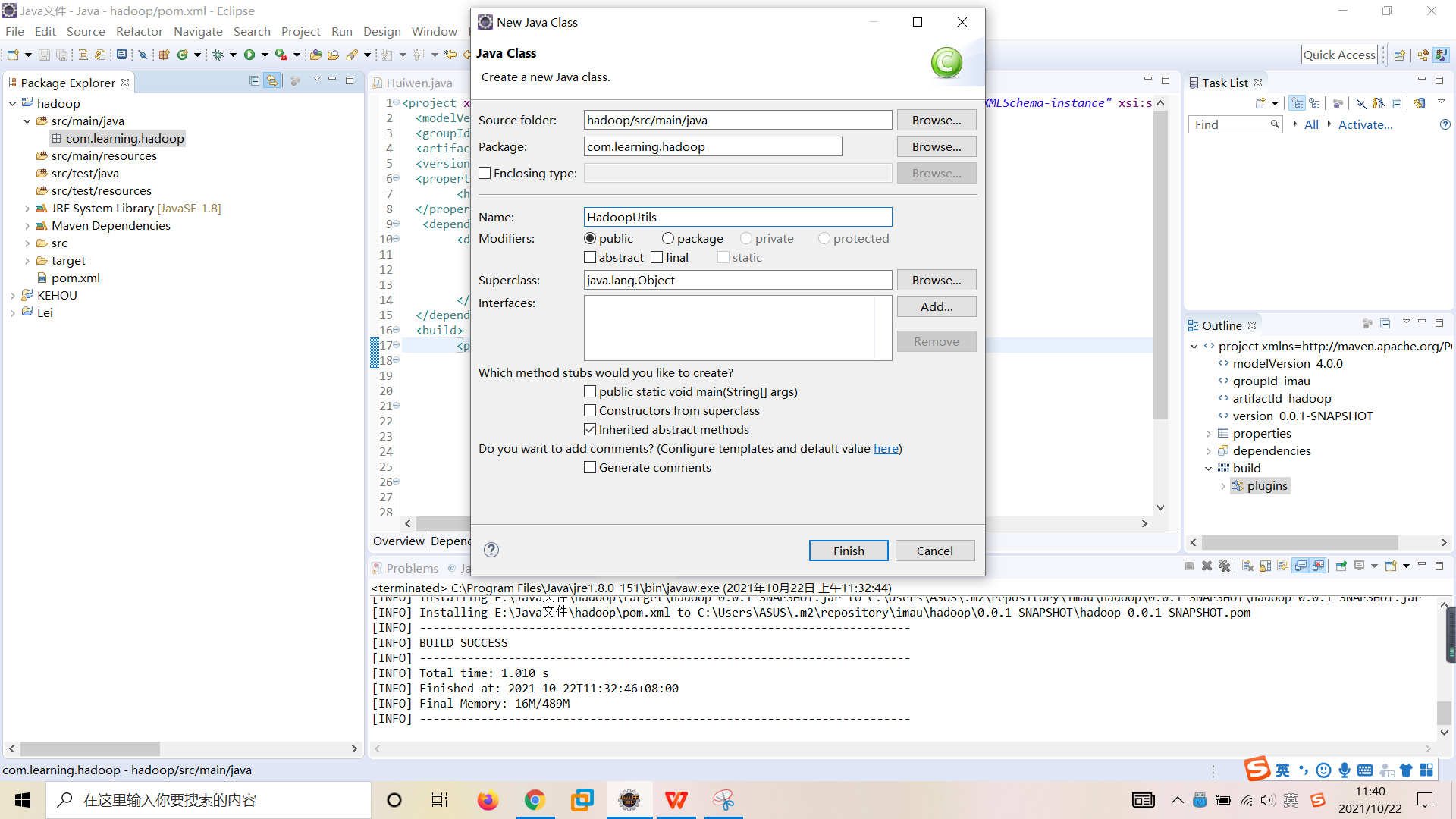
4.1、在src/main/java上单击右键新建package：



在弹窗口中“name：”里填写com.learning.hadoop：



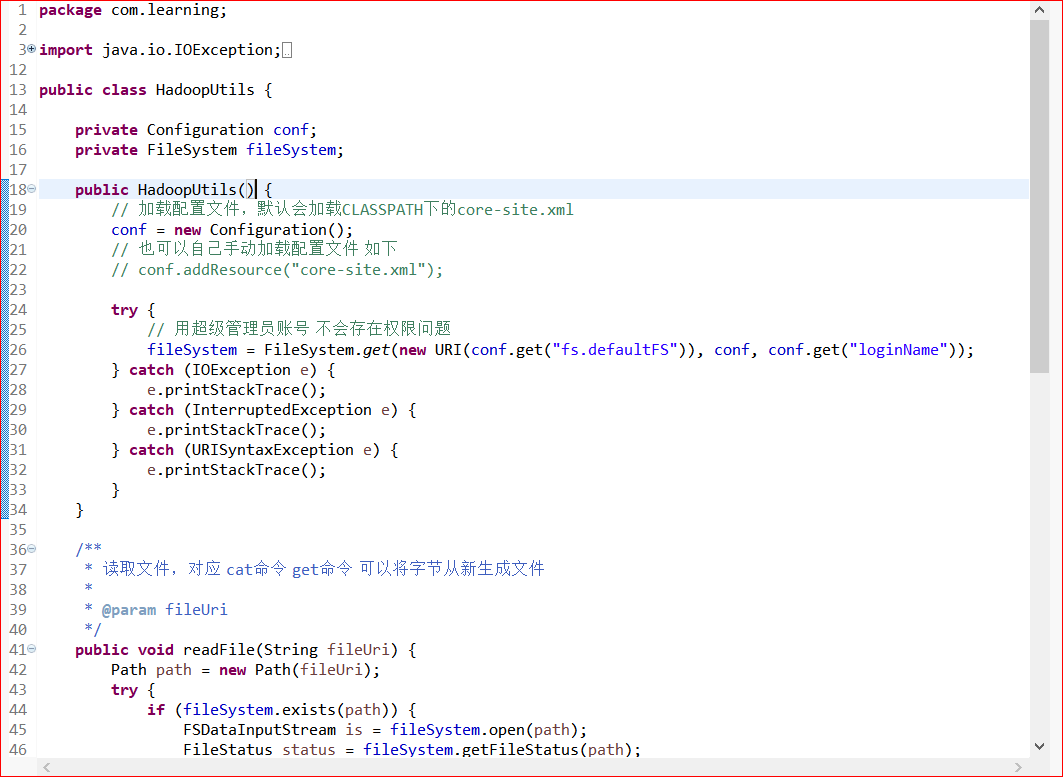
点击finish按钮，在创建的点击Finish按钮。再在创建的Package com.learning.hadoop上点击右键选择New 🡪 Class 创建一个类，类名叫HadoopUtils，并勾选创建 main方法选项。

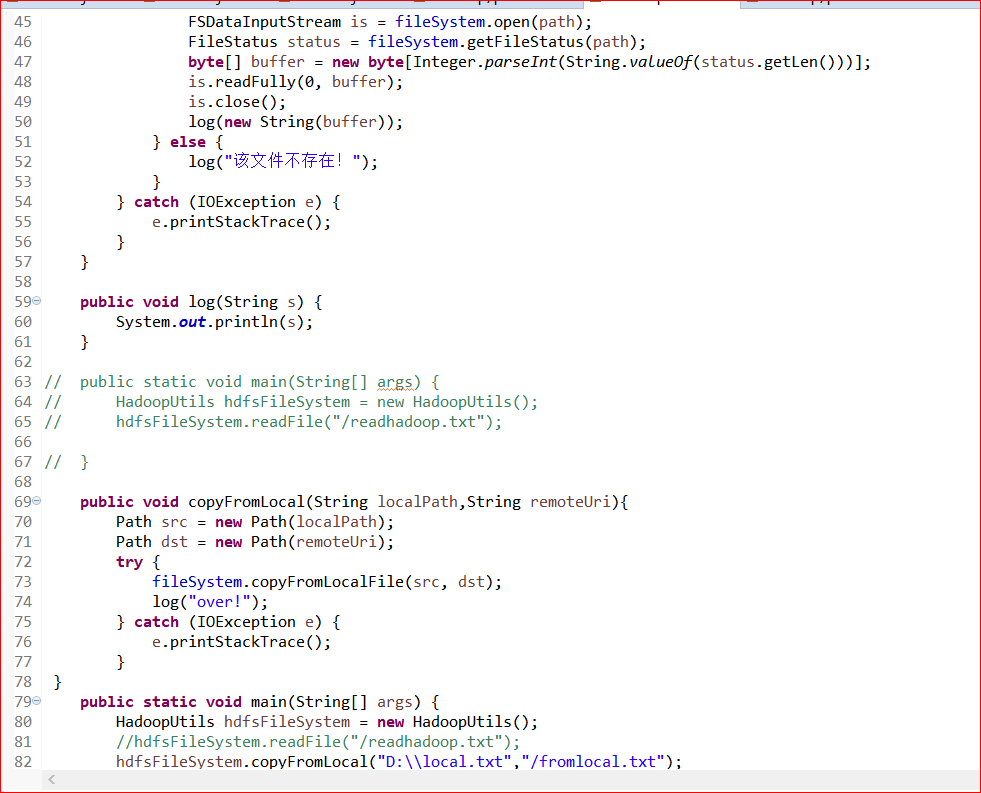


勾选

点击finish按钮，然后编辑HadoopUtils.java，参考代码如下：

|  |
| --- |
| package com.learning;  import java.io.IOException;  import java.net.URI;  import java.net.URISyntaxException;  import org.apache.hadoop.conf.Configuration;  import org.apache.hadoop.fs.FSDataInputStream;  import org.apache.hadoop.fs.FileStatus;  import org.apache.hadoop.fs.FileSystem;  import org.apache.hadoop.fs.Path;  public class HadoopUtils {  private Configuration conf;  private FileSystem fileSystem;  public HadoopUtils() {  // 加载配置文件，默认会加载CLASSPATH下的core-site.xml  conf = new Configuration();  // 也可以自己手动加载配置文件 如下  // conf.addResource("core-site.xml");  try {  // 用超级管理员账号 不会存在权限问题  fileSystem = FileSystem.get(new URI(conf.get("fs.defaultFS")), conf, conf.get("loginName"));  } catch (IOException e) {  e.printStackTrace();  } catch (InterruptedException e) {  e.printStackTrace();  } catch (URISyntaxException e) {  e.printStackTrace();  }  }  /\*\*  \* 读取文件，对应 cat命令 get命令 可以将字节从新生成文件  \*  \* @param fileUri  \*/  public void readFile(String fileUri) {  Path path = new Path(fileUri);  try {  if (fileSystem.exists(path)) {  FSDataInputStream is = fileSystem.open(path);  FileStatus status = fileSystem.getFileStatus(path);  byte[] buffer = new byte[Integer.parseInt(String.valueOf(status.getLen()))];  is.readFully(0, buffer);  is.close();  log(new String(buffer));  } else {  log("该文件不存在！");  }  } catch (IOException e) {  e.printStackTrace();  }  }  public void log(String s) {  System.out.println(s);  }  // public static void main(String[] args) {  // HadoopUtils hdfsFileSystem = new HadoopUtils();  // hdfsFileSystem.readFile("/readhadoop.txt");  // }    public void copyFromLocal(String localPath,String remoteUri){  Path src = new Path(localPath);  Path dst = new Path(remoteUri);  try {  fileSystem.copyFromLocalFile(src, dst);  log("over!");  } catch (IOException e) {  e.printStackTrace();  }  }  public static void main(String[] args) {  HadoopUtils hdfsFileSystem = new HadoopUtils();  //hdfsFileSystem.readFile("/readhadoop.txt");  hdfsFileSystem.copyFromLocal("D:\\local.txt","/fromlocal.txt");  }  } |

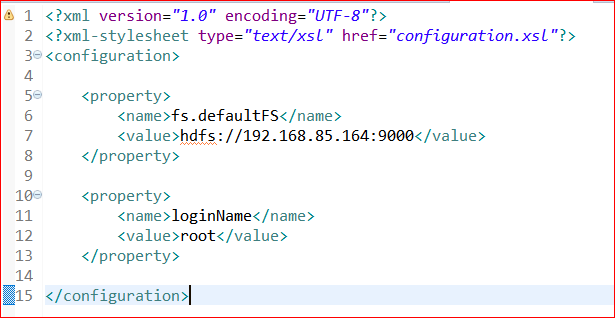




（\*如果 程序报错，在项目上右键，选择Maven -> Update project...\*）

在src/main/resources/ 目录下新建core-site.xml文件，内容如下(192.168.26.201为hadoop集群master服务器的ip，root为启动hadoop服务的用户，请根据自己实际情况修改)：

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <?xml-stylesheet type=*"text/xsl"* href=*"configuration.xsl"*?>  <configuration>    <property>  <name>fs.defaultFS</name>  <value>hdfs://192.168.85.164:9000</value>  </property>  <property>  <name>loginName</name>  <value>root</value>  </property>  </configuration> |

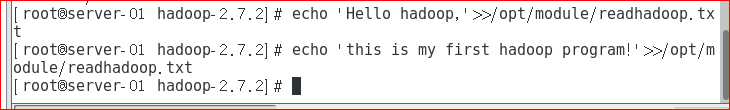


此时，代码基本开发完毕。我们在Hadoop集群上建立一个文本文档来测试：

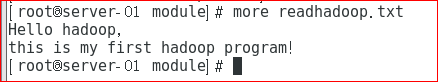
使用ssh工具登陆到hadoop集群的master服务器上，在目录/usr/local/zhitu/ 目录下创建一个readhadoop.txt文件，并输入内容为：hello hadoop，

This is my first hadoop program!

|  |
| --- |
| echo ‘hello hadoop,’>>/opt/module/readhadoop.txt  echo ‘This is my first hadoop program!’>>/opt/module/readhadoop.txt |

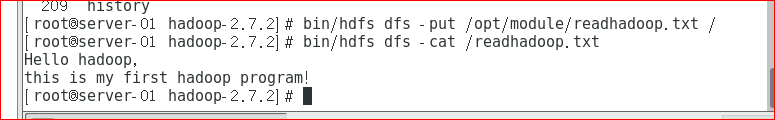


使用more命令查看readhadoop.txt文件内容：



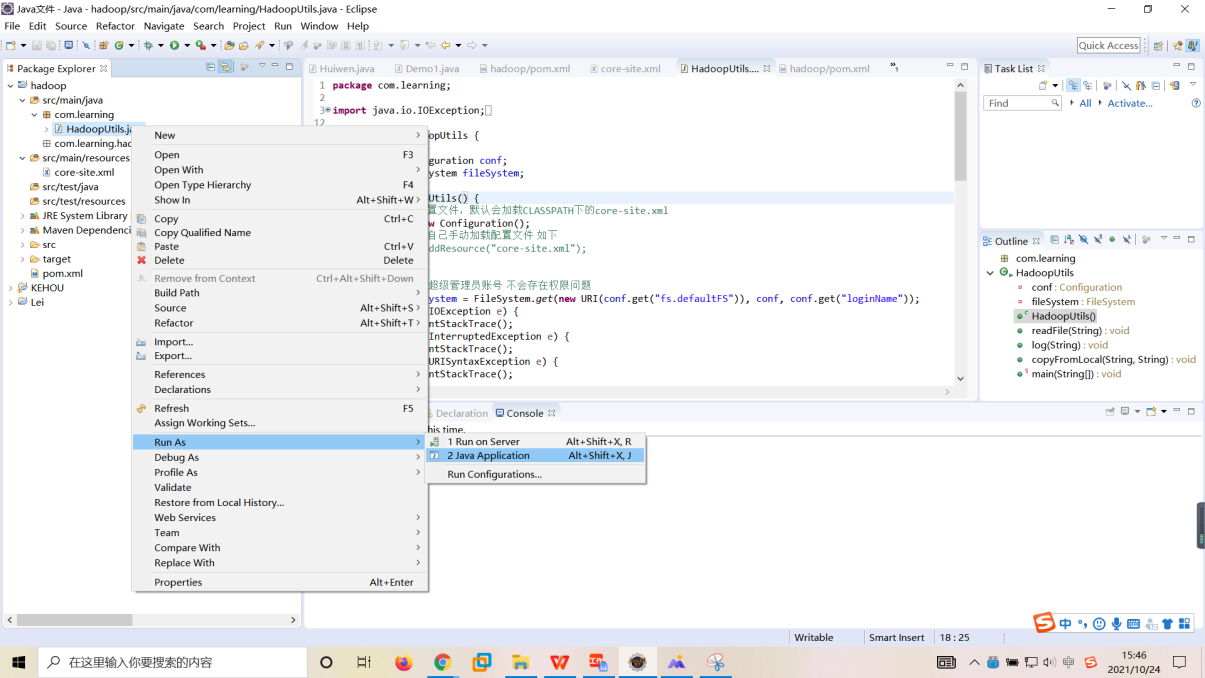
再使用命令bin/hdfs dfs -put 将文件上传到hdfs系统的根目录下，并使用bin/hdfs dfs -cat命令查看文件内容，命令参考如下：

|  |
| --- |
| bin/hdfs dfs -put /opt/module/readhadoop.txt /  bin/hdfs dfs -cat /readhadoop.txt |

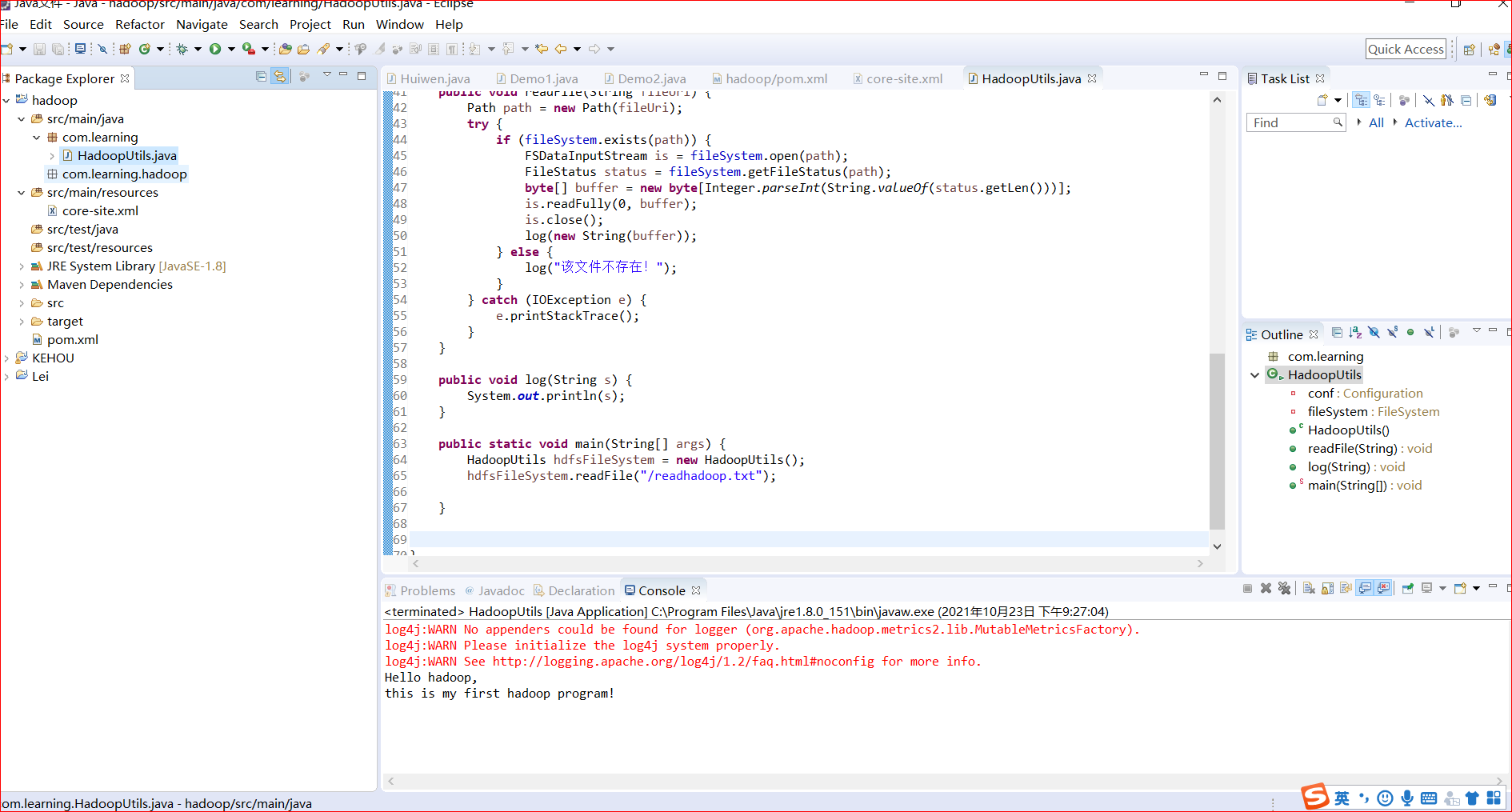


（\*如上图文件已上传成功\*）

现在可以运行刚刚编写的HadoopUtils.java，在HadoopUtils.java文件上点击右键，选择Run As 🡪 Java Application,如下图：



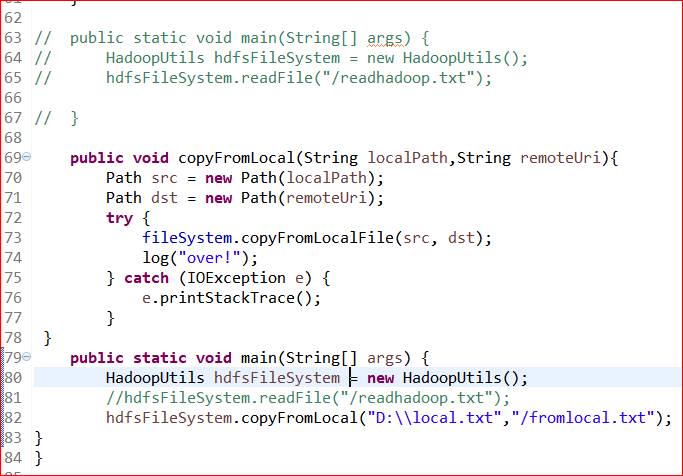
查看运行结果，可以看到已经成功将hdfs系统里的文件读取出来了：



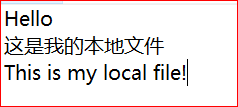
1. 开发上传本地文件到hdfs系统里的代码

5.1、在上一步中的HadoopUtils.java文件里 新增copyFromLocal(String localPath,String remoteUri) 方法

|  |
| --- |
| **public** **void** copyFromLocal(String localPath,String remoteUri){  Path src = **new** Path(localPath);  Path dst = **new** Path(remoteUri);  **try** {  fileSystem.copyFromLocalFile(src, dst);  log("over!");  } **catch** (IOException e) {  e.printStackTrace();  }  }  //并将main方法修改为如下：  **public** **static** **void** main(String[] args) {  HadoopUtils hdfsFileSystem = **new** HadoopUtils();  //hdfsFileSystem.readFile("/readhadoop.txt");  hdfsFileSystem.copyFromLocal("D:\\local.txt","/fromlocal.txt");  } |



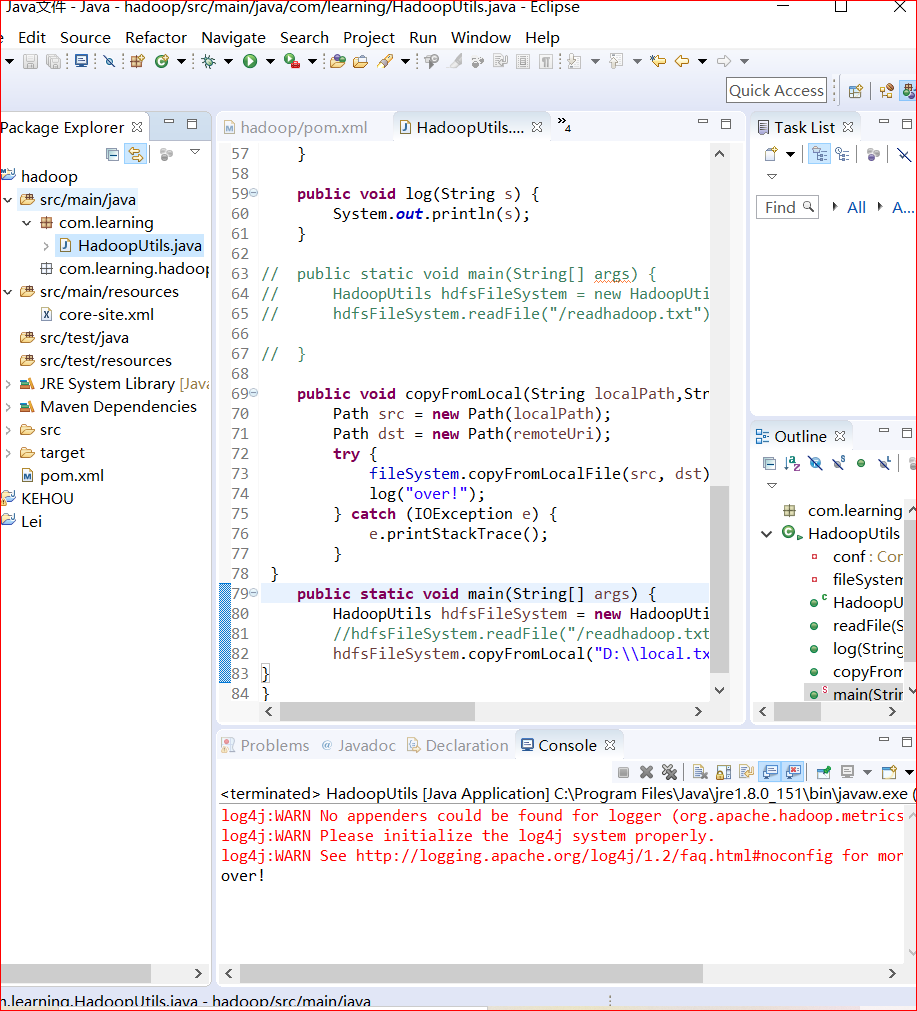
在自己的电脑D盘下创建一个文件，输入相关以下内容：



未上传到hdfs系统时：



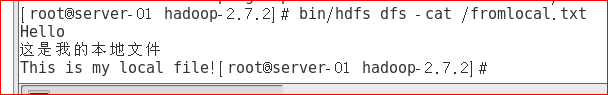
运行HadoopUtils.java：



再次去到hdfs系统里查看目录，并查看文件内容：

|  |
| --- |
| bin/hdfs dfs -ls -R /  bin/hdfs dfs -cat /fromlocal.txt |





**·实验心得：**

学会了如何在eclipse中建立Maven项目，如何在文件外写入内容，如何将文件上传到hdfs系统中，如何在eclipse中查看到hdfs系统中文件内容，如何通过eclipse中的java代码将本地磁盘的文件传入hdfs系统中，并进行查看文件内容。