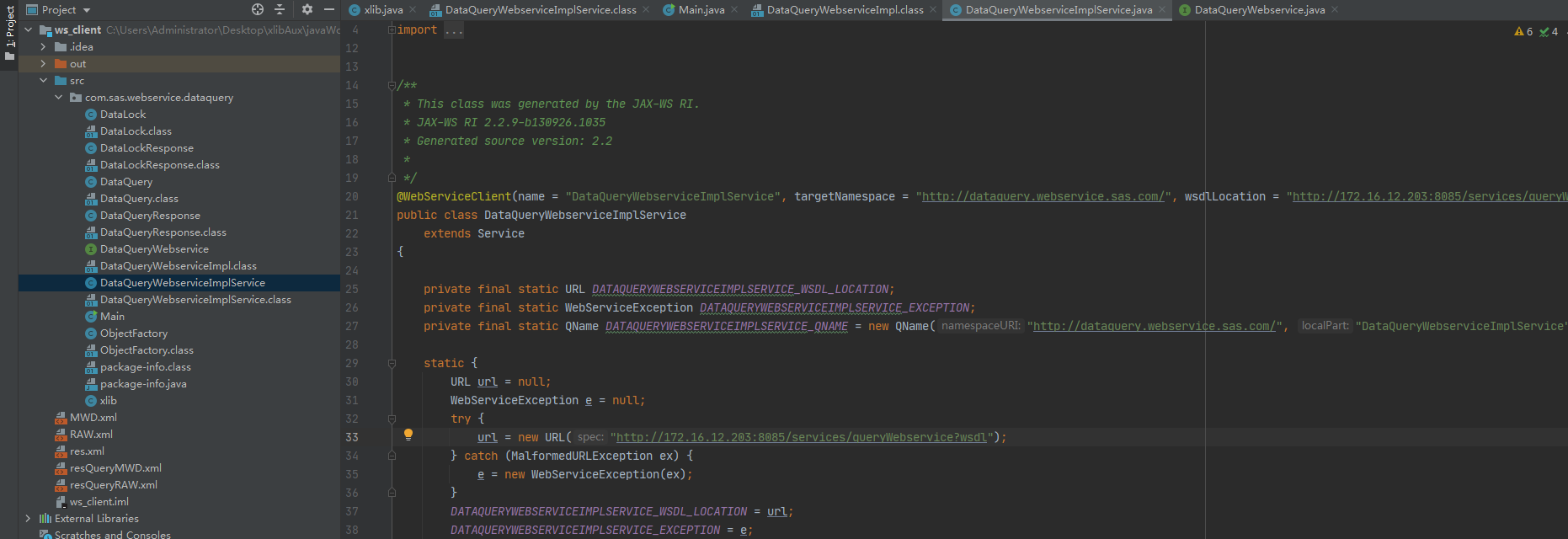
# Webservice

wsimport -keep http://192.168.31.13:8085/services/queryWebservice?wsdl

wsimport -keep http://192.168.31.13:8085/ws\_server/weather?wsdl

日志是应用软件中不可缺少的部分，Apache的开源项目[log4j](http://www.codeceo.com/article/log4j-usage.html)是一个功能强大的日志组件,提供方便的日志记录。在apache网站：[jakarta.apache.org/log4j](http://jakarta.apache.org/log4j) 可以免费下载到Log4j最新版本的软件包。

## 结构没变不用重新导入代码,改地址就行



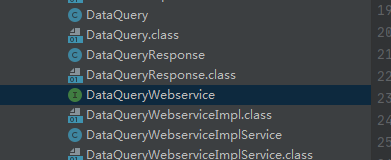
原来是

@WebServiceClient(name = "DataQueryWebserviceImplService", targetNamespace = "http://dataquery.webservice.sas.com/", wsdlLocation = "http://192.168.31.38:8091/services/queryWebservice?wsdl")

注意端口

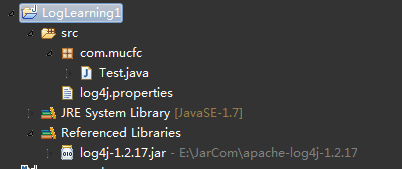
## 接口生成和真实的不一样名字,改名就行,内容不变就行

生成的带impl,直接改



# log4j

1.新建一个JAva工程，导入包log4j-1.2.17.jar，整个工程最终目录如下



2、src同级创建并设置log4j.properties

### 设置###

log4j.rootLogger = debug,stdout,D,E

### 输出信息到控制抬 ###

log4j.appender.stdout = org.apache.log4j.ConsoleAppender

log4j.appender.stdout.Target = System.out

log4j.appender.stdout.layout = org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern = [%-5p] %d{yyyy-MM-dd HH:mm:ss,SSS} method:%l%n%m%n

### 输出DEBUG 级别以上的日志到=E://logs/error.log ###

log4j.appender.D = org.apache.log4j.DailyRollingFileAppender

log4j.appender.D.File = E://logs/log.log

log4j.appender.D.Append = true

log4j.appender.D.Threshold = DEBUG

log4j.appender.D.layout = org.apache.log4j.PatternLayout

log4j.appender.D.layout.ConversionPattern = %-d{yyyy-MM-dd HH:mm:ss} [ %t:%r ] - [ %p ] %m%n

### 输出ERROR 级别以上的日志到=E://logs/error.log ###

log4j.appender.E = org.apache.log4j.DailyRollingFileAppender

log4j.appender.E.File =E://logs/error.log

log4j.appender.E.Append = true

log4j.appender.E.Threshold = ERROR

log4j.appender.E.layout = org.apache.log4j.PatternLayout

log4j.appender.E.layout.ConversionPattern = %-d{yyyy-MM-dd HH:mm:ss} [ %t:%r ] - [ %p ] %m%n

3、设置日志内容

**package** com.mucfc;

**import** org.apache.log4j.Logger;

/\*\*

\*@author linbingwen

\*@2015年5月18日9:14:21

\*/

**public** **class** **Test** {

**private** **static** Logger logger = Logger.getLogger(Test.class);

/\*\*

\* @param args

\*/

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

// System.out.println("This is println message.");

// 记录debug级别的信息

logger.debug("This is debug message.");

// 记录info级别的信息

logger.info("This is info message.");

// 记录error级别的信息

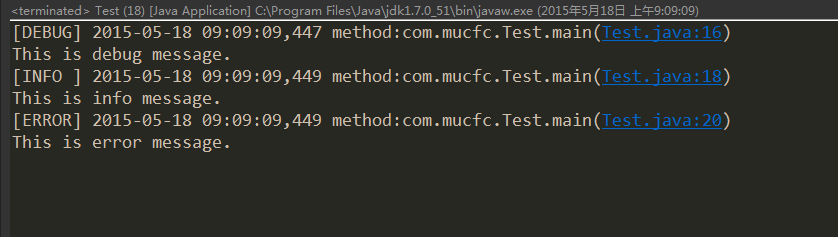
logger.error("This is error message.");

}

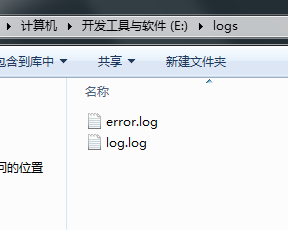
}

4、输出结果

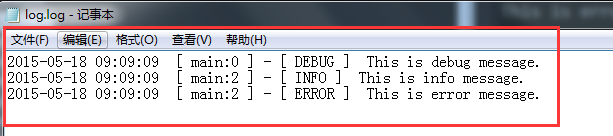
（1）首先是控制台的信息

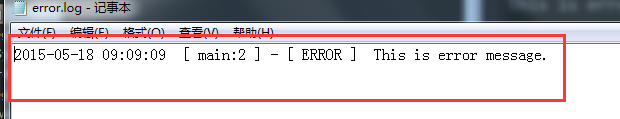


（2）再来看输出的文件



内容如下，发现已按照要求输出到对应的文档中去了。





# intell idea

折叠代码

//region 折叠后的描述

your code....

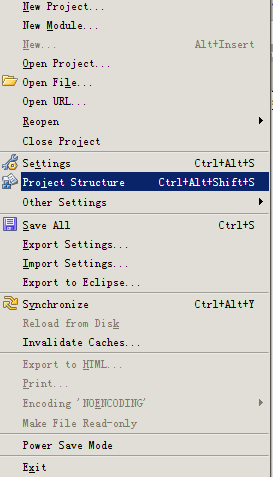
//endregion

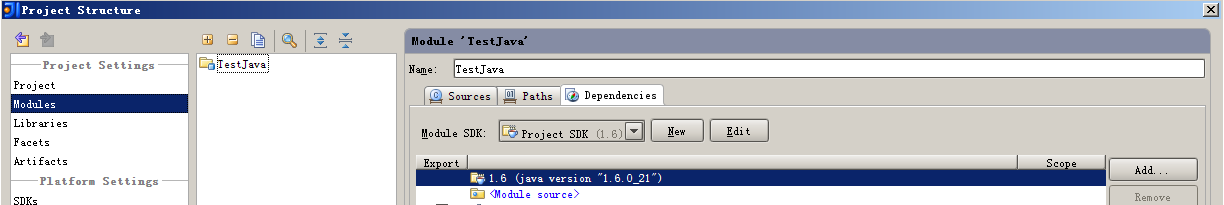
## 大小写不敏感提示

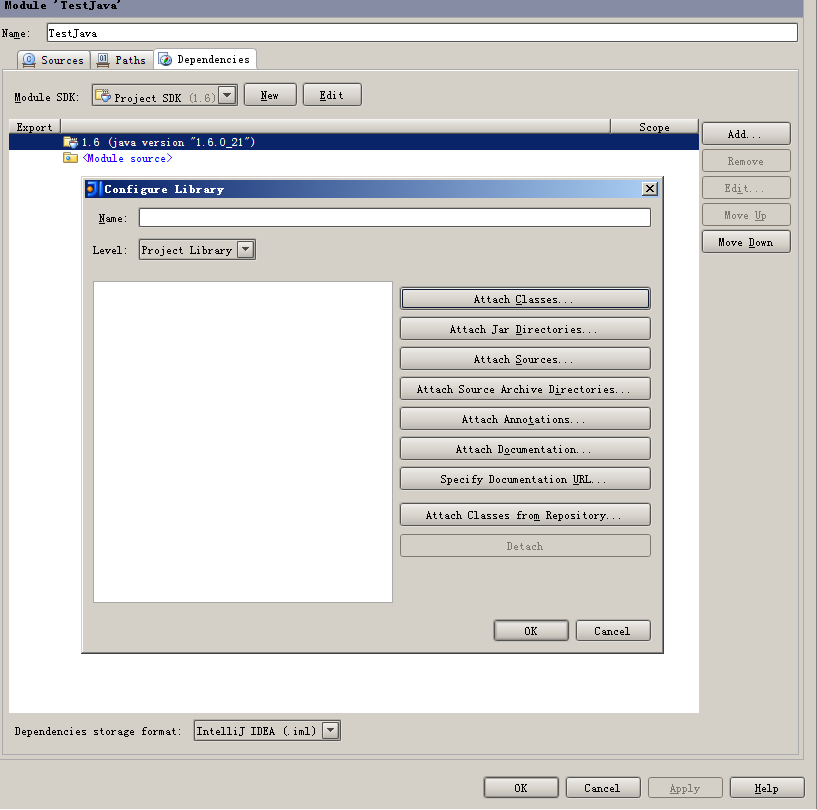
## [IntelliJ IDEA 10.5.1 引用外部Jar包](http://www.cnblogs.com/lonelyxmas/archive/2011/07/17/2108895.html)(如mysql驱动)

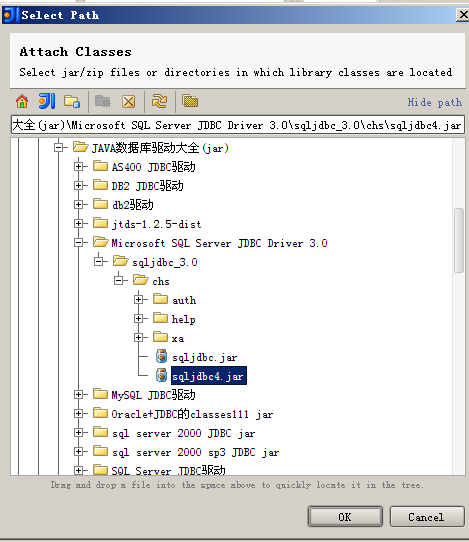
具体步骤：

File -> Project Structure (ctrl + shift + alt + s ) -> Module -> Dependencies -> Add... -> Library... -> Attach Classes.

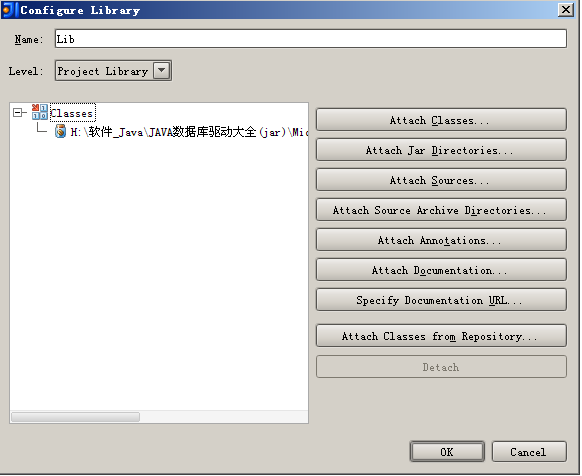


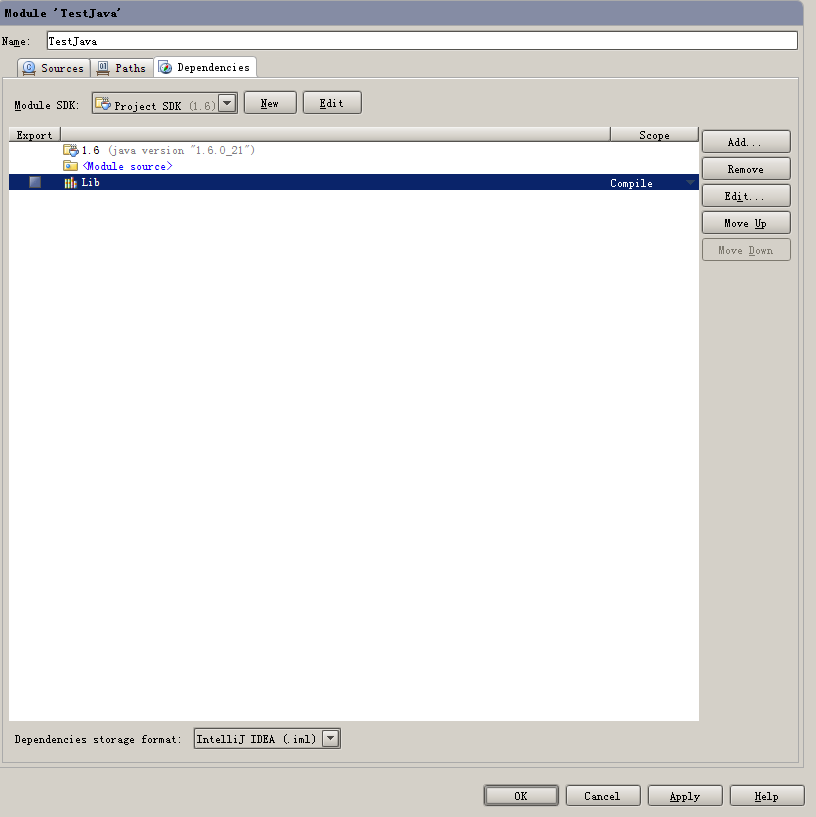


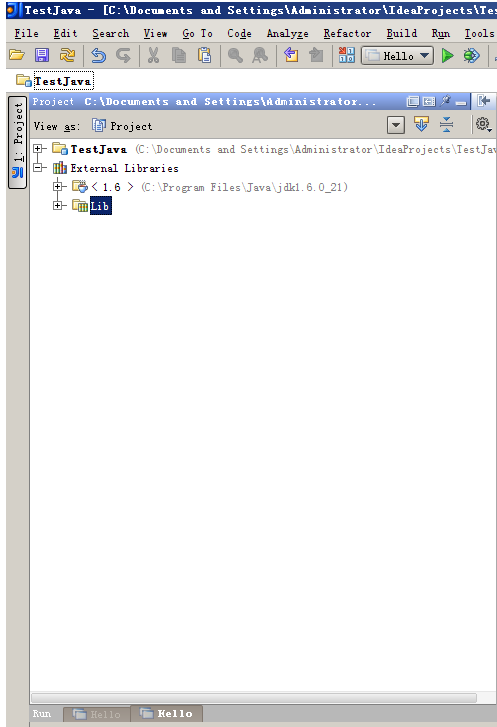




输入 Name：Lib （自定义）->OK







## 命令行运行java程序

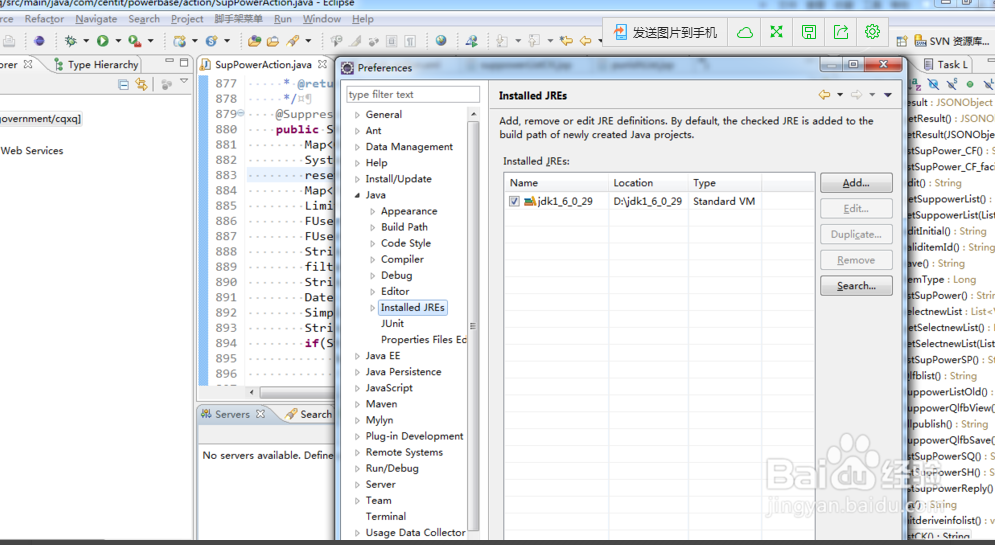
workspace-项目-out-production-项目-main.class是编译出来的

直接java main即可,如果有其它的调用,如mysql驱动,则把jar包放到main.class同级目录

java -Djava.ext.dirs=./ main

## 如果不行的话,改用jar包方式

# Eclipse更换jkd



# 连接mysql

这几天一直在弄mysql和JDBC之间连接的问题。

下面具体说下。大家少走弯路。

第一步：

先下载：mysql和JDBC的驱动:

下载地址：<http://mysql.ntu.edu.tw/Downloads/Connector-J/mysql-connector-java-5.0.8.zip>

第二步:

**解压文件，把里面的mysql-connector-java-5.0.8-bin.jar放到\Java\jdk1.6.0\_22\jre\lib\ext目录下。基本就完成了。**

下面写个测试程序：

import java.sql.Connection;

import java.sql.DriverManager;

public class Test {

public static void main(String[] args){

String url = "jdbc:mysql://127.0.0.1:3306/FACTORY";

        try{

              Class.forName("com.mysql.jdbc.Driver").newInstance();

              boolean jdbc;

Connection conn=DriverManager.getConnection(url,你数据库的用户名,你数据库的密码);

              System.out.println("connect mysql success");

 //             return conn;

        }catch(Exception e){

               e.printStackTrace();

 //              return null;

        }

}

}

# Java socket 发送类

接口包名必须一样

## 接口

package com.test.test;

import java.io.Serializable;

public class Packets implements Serializable{

public int id;

public String value;

public int ballot\_pid;

public int ballot\_num;

public int position;

public Packets(int id, String value, int ballot\_pid, int ballot\_num, int position){

this.id= id;

this.value= value;

this.ballot\_pid= ballot\_pid;

this.ballot\_num= ballot\_num;

this.position= position;

}

}

## 服务端

import java.io.IOException;

import java.io.ObjectInputStream;

import java.net.ServerSocket;

import java.net.Socket;

import com.test.test.Packets;

public class Server extends Thread{

private int port= 2013;

/\*flag used to stop this thread\*/

private boolean stop= false;

/\*pass addresses of clients from outside\*/

public Server(int port){

this.port= port;

}

public void run(){

Socket socket= null;

try{

ServerSocket serverSocket= new ServerSocket(port);

while(!stop){

socket= serverSocket.accept();

handleSocket(socket);

}

}

catch(Exception e){

e.printStackTrace();

}

super.run();

}

private void handleSocket(Socket socket) throws IOException, ClassNotFoundException{

ObjectInputStream ois= new ObjectInputStream(socket.getInputStream());

Packets packets= (Packets)ois.readObject();

/\*handle packets start\*/

/\*if broadcast, use 'Blog.broadcast(Packets packets)'\*/

System.out.println("id : "+ packets.id);

System.out.println("val: "+ packets.value);

System.out.println("pid: "+ packets.ballot\_pid);

System.out.println("num: "+ packets.ballot\_num);

System.out.println("pos: "+ packets.position);

/\*handle packets end\*/

socket.close();

}

/\*function used to fail this server\*/

public void fail(){

this.stop= true;

}

/\*function used to restart this server\*/

public void unfail(){

this.stop= false;

}

}

## 客户端

package javaClient;

import java.io.IOException;

import java.io.ObjectOutputStream;

import java.net.Socket;

import java.net.UnknownHostException;

import com.test.test.Packets;

public class Client{

private int port= 2013;

private String host= "192.168.1.124";

public Client(String host, int port){

this.host= host;

this.port= port;

}

public void send(Packets packets) throws UnknownHostException, IOException{

Socket socket= new Socket(host, port);

ObjectOutputStream os= new ObjectOutputStream(socket.getOutputStream());

os.writeObject(packets);

socket.getOutputStream().flush();

socket.close();

}

}

# 读取配置文件

2012-12-28 16:58 1701人阅读 [评论](http://blog.csdn.net/tender001/article/details/8448348#comments)(1) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/tender001/article/details/8448348#report)

http://static.blog.csdn.net/images/category_icon.jpg 分类：

Java（54） http://static.blog.csdn.net/images/arrow_triangle%20_down.jpg

xml文件：

　　Xml代码

|  |
| --- |
| <?xml version="1.0" encoding="GB2312"?>   　　<RESULT>  　　<VALUE>  　　<NO>A1234</NO>  　　<ADDR>河南省郑州市</ADDR>  　　</VALUE>  　　<VALUE>  　　<NO>B1234</NO>  　　<ADDR>河南省郑州市二七区</ADDR>  　　</VALUE>  　　</RESULT> |

　　第一种 DOM 实现方法：

　　Java代码

|  |
| --- |
| import java.io.File;  　　import javax.xml.parsers.DocumentBuilder;  　　import javax.xml.parsers.DocumentBuilderFactory;  　　import org.w3c.dom.Document;  　　import org.w3c.dom.NodeList;  　　public class MyXMLReader2DOM {  　　public static void main(String arge[]) {  　　long lasting = System.currentTimeMillis();  　　try {  　　File f = new File("data\_10k.xml");  　　DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();  　　DocumentBuilder builder = factory.newDocumentBuilder();  　　Document doc = builder.parse(f);  　　NodeList nl = doc.getElementsByTagName("VALUE");  　　for (int i = 0; i < nl.getLength(); i++) {  　　System.out.print("车牌号码:"+ doc.getElementsByTagName("NO").item(i).getFirstChild().getNodeValue());  　　System.out.println("车主地址:"+ doc.getElementsByTagName("ADDR").item(i).getFirstChild().getNodeValue());  　　System.out.println("运行时间：" + (System.currentTimeMillis() - lasting)  　　+ "毫秒");  　　}  　　}  　　} catch (Exception e) {  　　e.printStackTrace();  　　}  　　}  　　} |

　　第二种，DOM4J实现方法：

　　Java代码

|  |
| --- |
| import java.io.\*;  　　import java.util.\*;  　　import org.dom4j.\*;  　　import org.dom4j.io.\*;  　　public class MyXMLReader2DOM4J {  　　public static void main(String arge[]) {  　　long lasting = System.currentTimeMillis();  　　try {  　　File f = new File("data\_10k.xml");  　　SAXReader reader = new SAXReader();  　　Document doc = reader.read(f);  　　Element root = doc.getRootElement();  　　Element foo;  　　for (Iterator i = root.elementIterator("VALUE"); i.hasNext();) {  　　foo = (Element) i.next();  　　System.out.print("车牌号码:" + foo.elementText("NO"));  　　System.out.println("车主地址:" + foo.elementText("ADDR"));  　　}  　　System.out.println("运行时间：" + (System.currentTimeMillis() - lasting)  　　+ "毫秒");  　　}  　　} catch (Exception e) {  　　e.printStackTrace();  　　}  　　}  　　} |

　　第三种 JDOM实现方法：

　　Java代码

|  |
| --- |
| import java.io.\*;  　　import java.util.\*;  　　import org.jdom.\*;  　　import org.jdom.input.\*;  　　public class MyXMLReader2JDOM {  　　public static void main(String arge[]) {  　　long lasting = System.currentTimeMillis();  　　try {  　　SAXBuilder builder = new SAXBuilder();  　　Document doc = builder.build(new File("data\_10k.xml"));  　　Element foo = doc.getRootElement();  　　List allChildren = foo.getChildren();  　　for (int i = 0; i < allChildren.size(); i++) {  　　System.out.print("车牌号码:"+ ((Element) allChildren.get(i)).getChild("NO").getText());  　　System.out.println("车主地址:"+ ((Element) allChildren.get(i)).getChild("ADDR").getText());  　　}  　　System.out.println("运行时间：" + (System.currentTimeMillis() - lasting)  　　+ "毫秒");  　　}  　　} catch (Exception e) {  　　e.printStackTrace();  　　}  　　}  　　} |

　　第四种SAX实现方法：

　　Java代码

|  |
| --- |
| import javax.xml.parsers.SAXParser;  　　import javax.xml.parsers.SAXParserFactory;  　　import org.xml.sax.Attributes;  　　import org.xml.sax.InputSource;  　　import org.xml.sax.SAXException;  　　import org.xml.sax.helpers.DefaultHandler;  　　public class MyXMLReader2SAX extends DefaultHandler {  　　java.util.Stack tags = new java.util.Stack();  　　public MyXMLReader2SAX() {  　　super();  　　}  　　public static void main(String args[]) {  　　long lasting = System.currentTimeMillis();  　　try {  　　SAXParserFactory sf = SAXParserFactory.newInstance();  　　SAXParser sp = sf.newSAXParser();  　　MyXMLReader2SAX reader = new MyXMLReader2SAX();  　　sp.parse(new InputSource("data\_10k.xml"), reader);  　　} catch (Exception e) {  　　e.printStackTrace();  　　}  　　System.out.println("运行时间：" + (System.currentTimeMillis() - lasting)  　　+ "毫秒");  　　}  　　public void characters(char ch[], int start, int length)  　　throws SAXException {  　　String tag = (String) tags.peek();  　　if (tag.equals("NO")) {  　　System.out.print("车牌号码：" + new String(ch, start, length));  　　}  　　if (tag.equals("ADDR")) {  　　System.out.println("地址:" + new String(ch, start, length));  　　}  　　}  　　public void startElement(String uri, String localName, String qName,  　　Attributes attrs) {  　　tags.push(qName);  　　}  　　} |

# 时间加减

//时间加减

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");

String str = "2011-08-23 00:00:00";

Date dt;

try {

dt = sdf.parse(str);

Calendar rightNow = Calendar.*getInstance*();

rightNow.setTime(dt);

// rightNow.add(Calendar.YEAR, -1);// 日期减1年

// rightNow.add(Calendar.MONTH, 3);// 日期加3个月

// rightNow.add(Calendar.DAY\_OF\_YEAR, 10);// 日期加10天

// rightNow.add(Calendar.HOUR, -3);

rightNow.add(Calendar.*MINUTE*, -30);

Date dt1 = rightNow.getTime();

String reStr = sdf.format(dt1);

System.*out*.println(reStr);

} catch (ParseException e) {

e.printStackTrace();

}

# 获取当前时间

## Date型

Date now=new Date();

## String型

public String getNowTime()

{

Date dt=new Date();//如果不需要格式,可直接用dt,dt就是当前系统时间

DateFormat df = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");//设置显示格式

String nowTime="";

nowTime= df.format(dt);//用DateFormat的format()方法在dt中获取并以yyyy/MM/dd HH:mm:ss格式显示

return nowTime;

//System.out.println(nowTime);

}

# 下载从url

/\*\*

\* 从网络Url中下载文件

\* @param urlStr

\* @param fileName

\* @param savePath

\* @throws IOException

\*/

public static void downLoadFromUrl(String urlStr,String fileName,String savePath) throws IOException{

URL url = new URL(urlStr);

HttpURLConnection conn = (HttpURLConnection)url.openConnection();

//设置超时间为3秒

conn.setConnectTimeout(3\*1000);

//防止屏蔽程序抓取而返回403错误

conn.setRequestProperty("User-Agent", "Mozilla/4.0 (compatible; MSIE 5.0; Windows NT; DigExt)");

//得到输入流

InputStream inputStream = conn.getInputStream();

//获取自己数组

byte[] getData = *readInputStream*(inputStream);

//文件保存位置

File saveDir = new File(savePath);

if(!saveDir.exists()){

saveDir.mkdir();

}

File file = new File(saveDir+File.*separator*+fileName);

FileOutputStream fos = new FileOutputStream(file);

fos.write(getData);

if(fos!=null){

fos.close();

}

if(inputStream!=null){

inputStream.close();

}

System.*out*.println("info: "+url+" download success");

}

/\*\*

\* 从输入流中获取字节数组

\* @param inputStream

\* @return

\* @throws IOException

\*/

public static byte[] readInputStream(InputStream inputStream) throws IOException {

byte[] buffer = new byte[1024];

int len = 0;

ByteArrayOutputStream bos = new ByteArrayOutputStream();

while((len = inputStream.read(buffer)) != -1) {

bos.write(buffer, 0, len);

}

bos.close();

return bos.toByteArray();

}

# 小数点后两位

DecimalFormat df = new DecimalFormat("0.00");

// add by xuzhenwei 添加京东申请售后按钮 end

model.addAttribute("jdResultOrderItemVOList", jdResultOrderItemVOList);

model.addAttribute("compareJDprice", df.format(compareJDprice));

compareJDprice是bigdecimal的

# 转换

## String –转BigDecimal

String str = “2.30”;

BigDecimal bd = new BigDecimal(str);

# JDK环境变量

系统变量→新建 JAVA\_HOME 变量 。变量值填写jdk的安装目录（本人是 C:\Program Files\Java\jdk1.7.0\_45)

系统变量→寻找 Path 变量→编辑

在变量值最后输入 %JAVA\_HOME%\bin;%JAVA\_HOME%\jre\bin;

（注意原来Path的变量值末尾有没有;号，如果没有，先输入；号再输入上面的代码）

系统变量→新建 CLASSPATH 变量

变量值填写   .;%JAVA\_HOME%\lib;%JAVA\_HOME%\lib\tools.jar（注意最前面有一点）

系统变量配置完毕

Javac测试

# 基本语法

## 比较字符串

int res=str1.compareTo(str2) res=0相等,否则不等

一般地，当需要使用数字的时候，我们通常使用内置数据类型，如：byte、int、long、double等。

1. int i = 5000;
2. float gpa = 13.65;
3. byte mask = 0xaf;

然而，在实际开发过程中，我们经常会遇到需要使用对象，而不是内置数据类型的情形。为了解决这个问题，Java语言为每一个内置数据类型提供了对应的包装类。  
所有的包装类（Integer、Long、Byte、Double、Float、Short）都是抽象类Number的子类。

这种由编译器特别支持的包装称为装箱，所以当内置数据类型被当作对象使用的时候，编译器会把内置类型装箱为包装类。相似的，编译器也可以把一个对象拆箱为内置类型。Number类属于java.lang包。  
下面是一个装箱与拆箱的例子：

1. **public** **class** Test{
2. **public** **static** void main(String args[]){
3. Integer x = 5; // boxes int to an Integer object
4. x = x + 10; // unboxes the Integer to a int
5. System.out.println(x);
6. }
7. }

运行结果如下：  
15  
  
当x被赋为整型值时，由于x是一个对象，所以编译器要对x进行装箱。然后，为了使x能进行加运算，所以要对x进行拆箱。

## **Number类的成员方法：**

下面的表中列出的是Number类的方法：

|  |  |
| --- | --- |
| **序号** | **方法与描述** |
| 1 | xxxValue() 将number对象转换为xxx数据类型的值并返回。 |
| 2 | compareTo() 将number对象与参数比较。 |
| 3 | equals() 判断number对象是否与参数相等。 |
| 4 | valueOf() 返回一个Integer对象指定的内置数据类型 |
| 5 | toString() 以字符串形式返回值。 |
| 6 | parseInt() 将字符串解析为int类型。 |
| 7 | abs() 返回参数的绝对值。 |
| 8 | ceil() 对整形变量向左取整，返回类型为double型。 |
| 9 | floor() 对整型变量向右取整。返回类型为double类型。 |
| 10 | rint() 返回与参数最接近的整数。返回类型为double。 |
| 11 | round() 返回一个最接近的int、long型值。 |
| 12 | min() 返回两个参数中的最小值。 |
| 13 | max() 返回两个参数中的最大值。 |
| 14 | exp() 返回自然数底数e的参数次方。 |
| 15 | log() 返回参数的自然数底数的对数值。 |
| 16 | pow() 返回第一个参数的第二个参数次方。 |
| 17 | sqrt() 求参数的算术平方根。 |
| 18 | sin() 求指定double类型参数的正弦值。 |
| 19 | cos() 求指定double类型参数的余弦值。 |
| 20 | tan() 求指定double类型参数的正切值。 |
| 21 | asin() 求指定double类型参数的反正弦值。 |
| 22 | acos() 求指定double类型参数的反余弦值。 |
| 23 | atan() 求指定double类型参数的反正切值。 |
| 24 | atan2() 将笛卡尔坐标转换为极坐标，并返回极坐标的角度值。 |
| 25 | toDegrees() 将参数转化为角度。 |
| 26 | toRadians() 将角度转换为弧度。 |
| 27 | random() 返回一个随机数。 |

# 网站web开发环境(java)

## MySql

安装包安装,不要设置root密码,即不要安全设置不然出错装不上,navicat连接也不用密码,(默认密码空)

若导入sql文件不成功,换换编码试试,一般为gbk或者默认utf-8

往数据库写入中文为乱码的问题:安装选择语言的时候选最下面,语言也选的gbk

### 连接

下载jdbcformysql,把里面的.jar直接拷贝到项目目录web-inf lib中,加入import java.sql.\*;

body中加入

<%

try{

Class.forName("com.mysql.jdbc.Driver");

String url="jdbc:mysql://localhost:3306/test";

String username="root";

String password="";

Connection conn=DriverManager.getConnection(url,username,password);

if(conn!=null)

{

System.out.println("数据库连接成功");

}

else

{

System.out.println("数据库连接失败");

}

}catch(ClassNotFoundException e)

{

e.printStackTrace();

}catch(SQLException e){

e.printStackTrace();

}

%>

## MyEclipse

要装jdk,然后装tomcat,然后再me中添加tomcat路径:window-preference-myeclipse-server,然后再server中启动tomcat,然后点deploy添加tomcat服务器,然后deploy,最后在浏览器中输入地址<http://localhost:8080/tomcatTest/> 最后参数是项目名称.

因为64位的me9没有设计拖控件啥的,所以装32位的,jdk也要装32位的,否则无法激活,导致不能自动计算systemid啥的,但是激活之后貌似用64jdk也不影响使用

workspace目录下一定要直接放项目文件夹,中间不要隔着东西,否则影响发布

改启动页面在web-inf下面的web.xml中改welcome那项

### 激活

MyEclipse9下载、安装、激活

注意：一定要按照下列方法，顺序不能乱！！！！！

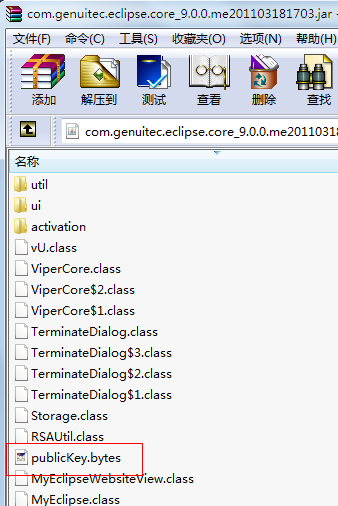
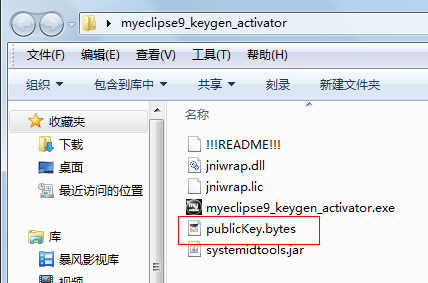
MyEclipse9迅雷下载地址：

ed2k://|file|%5Bmyeclipse.9.0.%E5%8F%91%E5%B8%83%5D.myeclipse-9.0-offline-installer-windows.exe|971679120|86377fc318829a84ba7915f3544752f2|h=bbclsdgclihngyo6fzknpa6b7o4pwojr|/

下载注册机

注册机下载地址：[myeclipse9\_keygen\_activator](http://www.chinasb.org/wp-content/uploads/2011/03/myeclipse9_keygen_activator.zip)

解压后得到（图一）

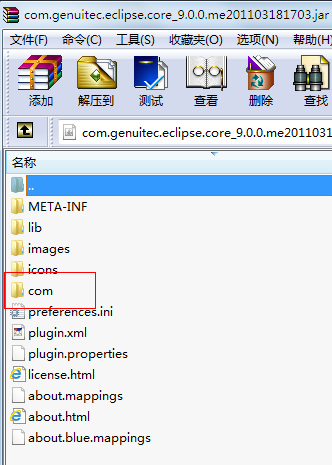


图一 图二

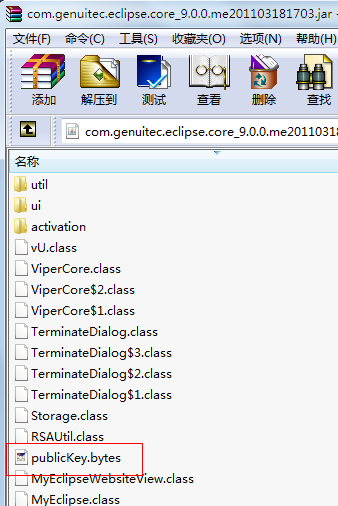
替换publicKey.bytes文件（将图二中的publicKey.bytes用图一中的publicKey.bytes文件替换）替换方法如下：

打开MyEclipse9安装目录，找到

com.genuitec.eclipse.core\_9.0.0.me201103181703.jar文件（通常文件路径为：安装盘根目录未加上AppData\Local\MyEclipse\Common\plugins），用打开此com.genuitec.eclipse.core\_9.0.0.me201103181703.jar文件，得到如图。



打开com文件夹，到com\genuitec\eclipse\core目录下（图二）。

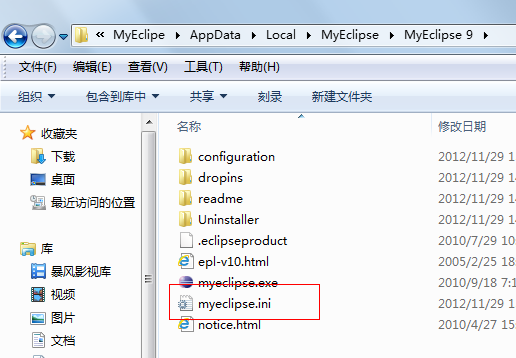


图二

复制图一中的publicKey.bytes文件，在图二中粘贴就替换了。

修改myeclipse.ini文件。

打开myeclipse9安装目录，找到myeclipse.ini文件（通常路径：安装盘根目录未加上AppData\Local\MyEclipse\MyEclipse 9）如图所示：

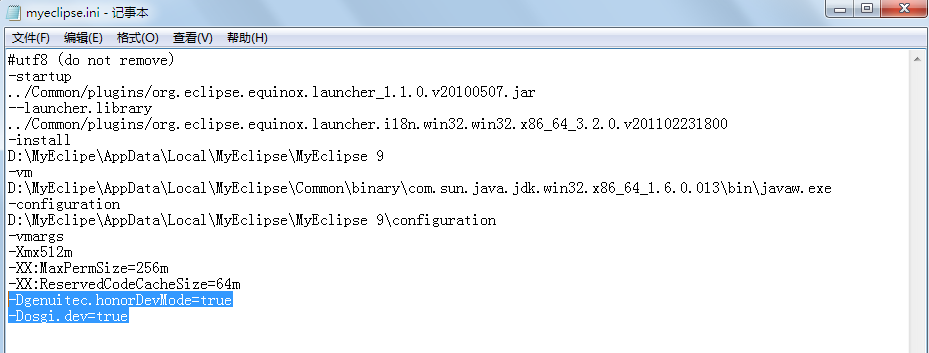


双击打开myeclipse.ini文件，编辑在最后加入

-Dgenuitec.honorDevMode=true

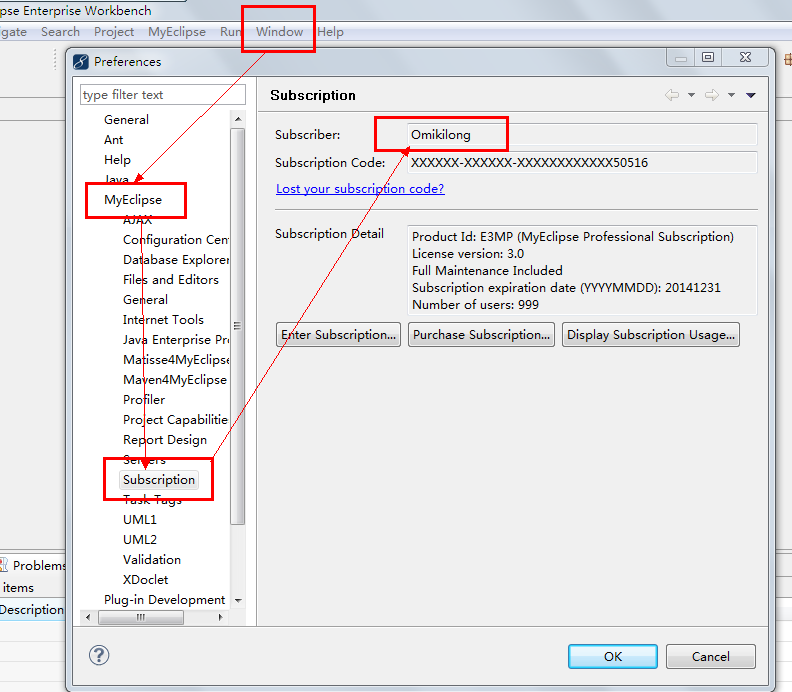
-Dosgi.dev=true

两句代码，保存完成修改。



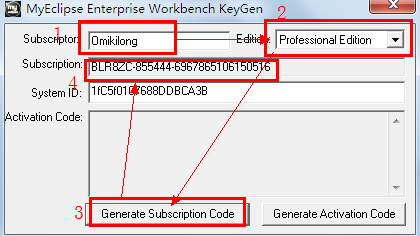
启动MyEclipse9。注意：自动弹出要求注册直接取消，不要注册，手动注册。

启动后，按如下图操作：记住Subscriber名，不必关闭此对话框。

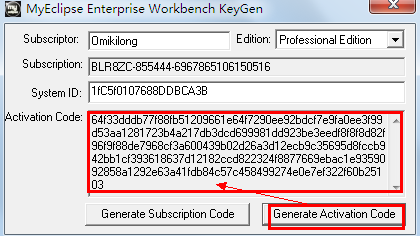


打开注册机。

打开图一中的myeclipse9\_keygen\_activator.exe文件，输入上图中得到的Subscriber名到Subscriptor中，Edition选择ProfessionalEdition，点击GenerateSubscriptionCode按钮，得到Subscription号，点击GenerateActivationCode按钮，得到ActcvationCode号。如图操作。



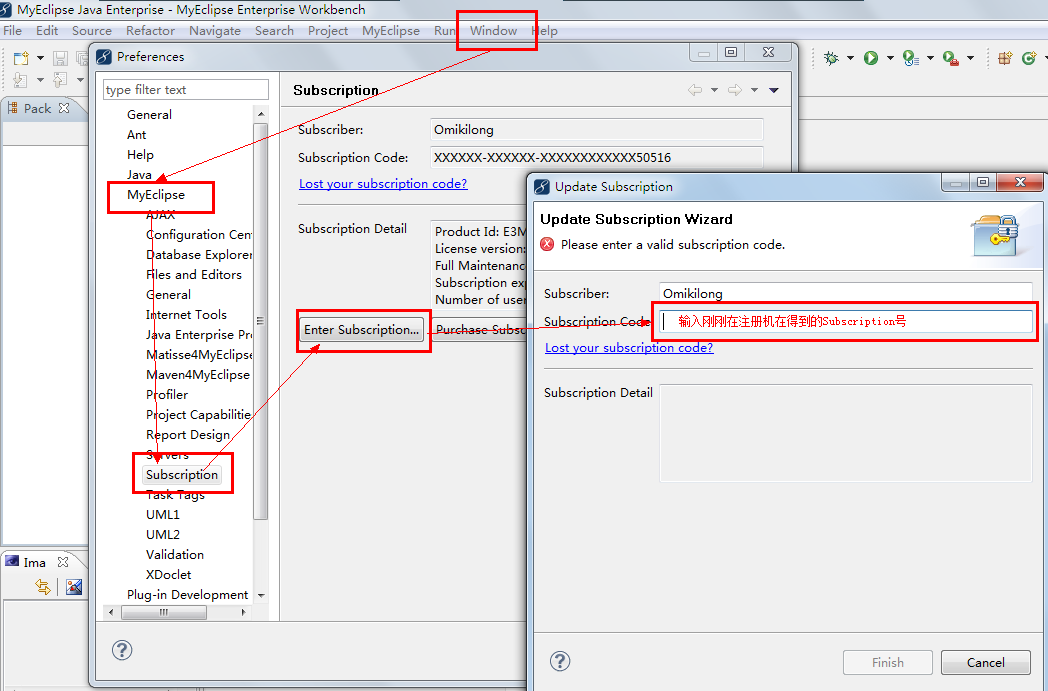
获得Subscription号



获得ActivationCode号

不要关闭注册机，回到MyEclipse9界面。

修改Subscription号，如图方法修改。



修改完成后，输入注册码，完成激活。

输入在注册机在得到的ActivationCode号，完成激活。

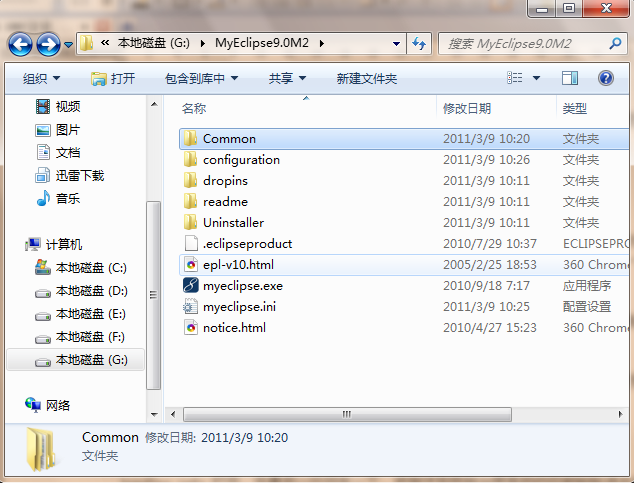
### 汉化(未测试)

MyEclipse 9.0正式版中文汉化教程

有了汉化包我们开始汉化吧，需要照我的步骤来。

切记：安装目录不要出现带空格的文件夹

1：把下载好的汉化包里的language的文件夹全粘贴到到你安装的*MyEclipse 9.0*里面的Common文件夹(如果没有,就创建一个)。

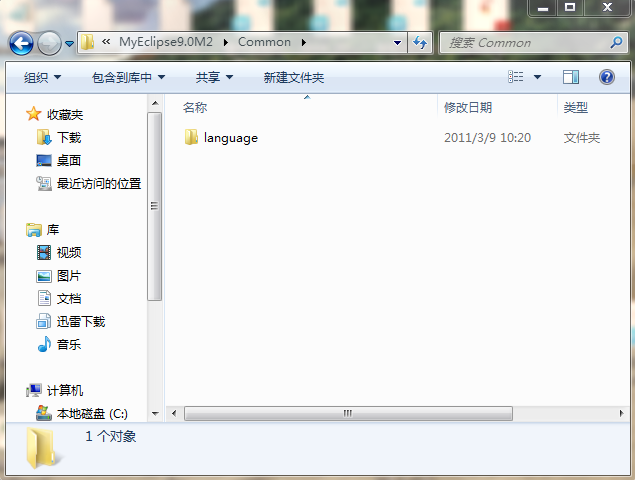


2：把压缩包里面的CreatePluginsConfig.java里面的代码复制如下面一样。  
注意上面C:\\Program Files\\Genuitec\\Common\\language\\plugins的路径要改成你电脑安装*MyEclipse 9.0*的路径，把红色的路径地址替换成你的，注意在每个"\"多加一个"\"，比如我的安装路径是c:\soft\就要改成c:\\soft\\。

3：创建一个项目，在项目里面建立一个CreatePluginsConfig类，把上面的代码复制进去，记得修改你的插件路径。

4：运行你的CreatePluginsConfig类，然后在控制台下面会输出一大串英文。这时要全部把控制台输出的英文都复*制到*

*C:\Program Files\Genuitec\MyEclipse 9.0\configuration\org.eclipse.equinox.simpleconfigurator* ,用记事本把bundles.info 打开，在最后一行回车一下。把刚才在控制台里面的内容都粘贴进去保存。*MyEclipse 9.0*的文件夹是你*MyEclipse 9.0*运行路径看图1。

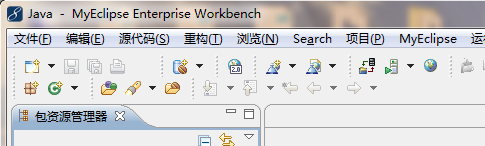
  
图片1

如果你的MyEclipse运行的文件夹不是*MyEclipse 9.0*的话就看一下你的运行路径是什么按照上面的路径打开bundles.info 文件就行了。这个很重要因为有2个configuration文件夹件，怕大家进错了，所以要注意是MyEclipse运行文件夹里的configuration。

5：找到*MyEclipse 9.0*的目录下找到myeclipse.ini文件，文件就图1的里面的位置。用记事本打开在最后一行加多一条语句

-Duser.language=zh

6：保存，启动你的MyEclipse吧，这里你就可以看到中文的界面了，祝你成功。



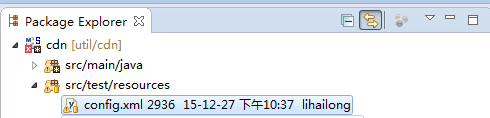
# cdn

## 注意

日志下载和资源监控已设置定时任务.

### 日志保存路径

下载日志时要指定日志保存路径,配置文件:



内容为

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<result>

<parent>

<path>C:\\desktop\\logs</path>

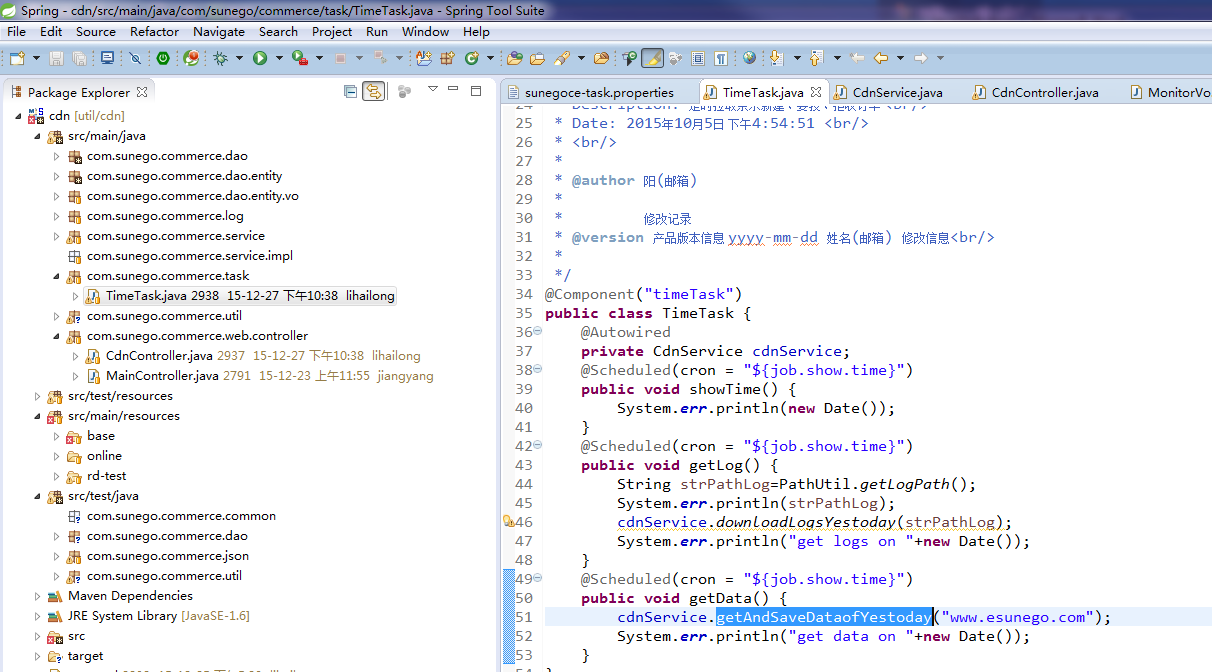
</parent>

</result>

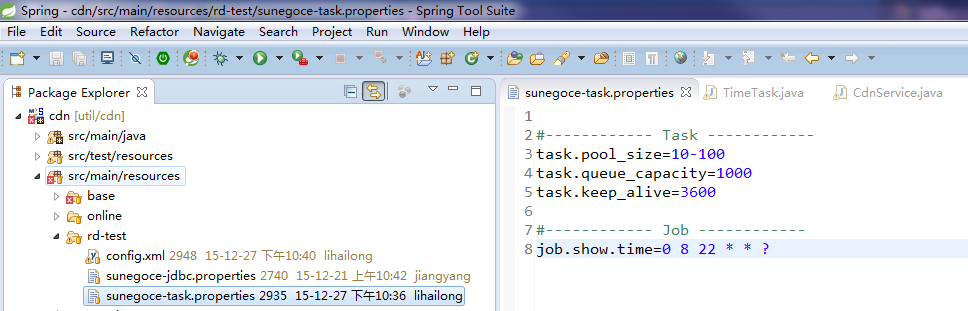
### 定时任务

#### 所在类

TimeTask.java



#### 时间设置



## 刷新指定缓存

### 所在类名

CdnService

### 描述

/\*\*

\* 刷新指定地址的缓存

\*

\* **@author** xlfd

\* **@param** strCachePath

\* 例子www.esunego.com/product\_category.jhtml

\* **@return**

\*/

**public** **static** String refreshOne(String strCachePath)

## 下载昨天所有日志

### 所在类名

CdnService

### 描述

/\*\*

\* 根据保存路径下载昨天的日志

\*

\* **@author** xlfd

\* **@param** strSavePath

\* 保存路径

\*/

**public** **static** **void** downloadLogsYestoday(String strSavePath)

## 下载昨天所有资源监控并存到数据库

### 所在类名

CdnService

/\*\*

\* 获取data并存储

\*

\* **@author** xlfd

\* **@param** strDomainName

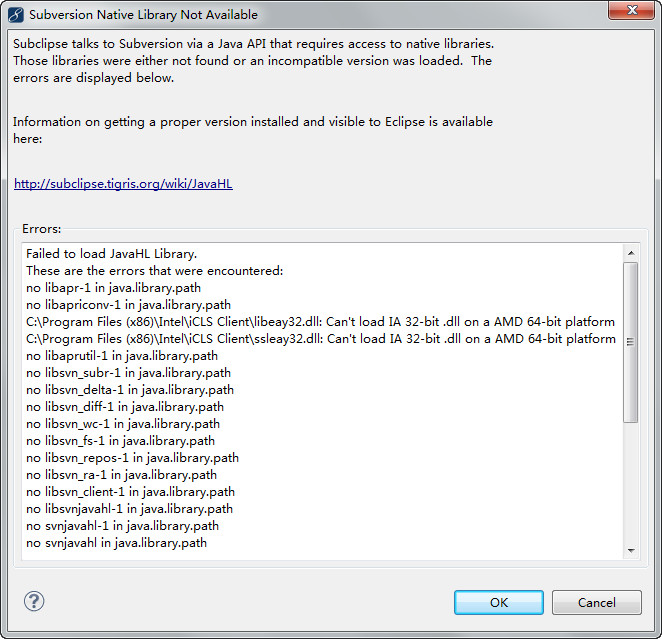
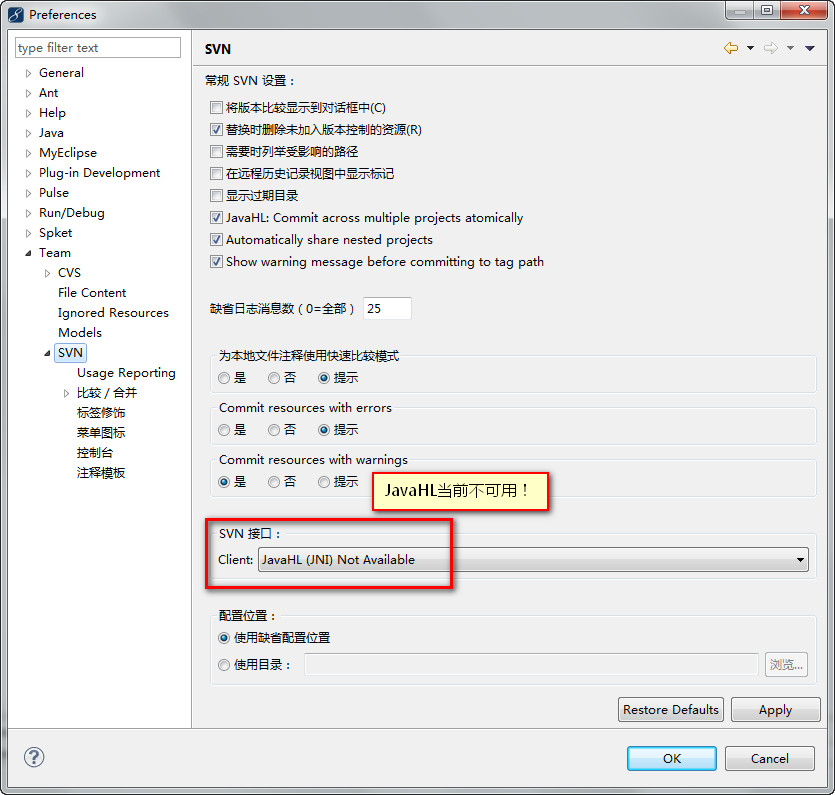
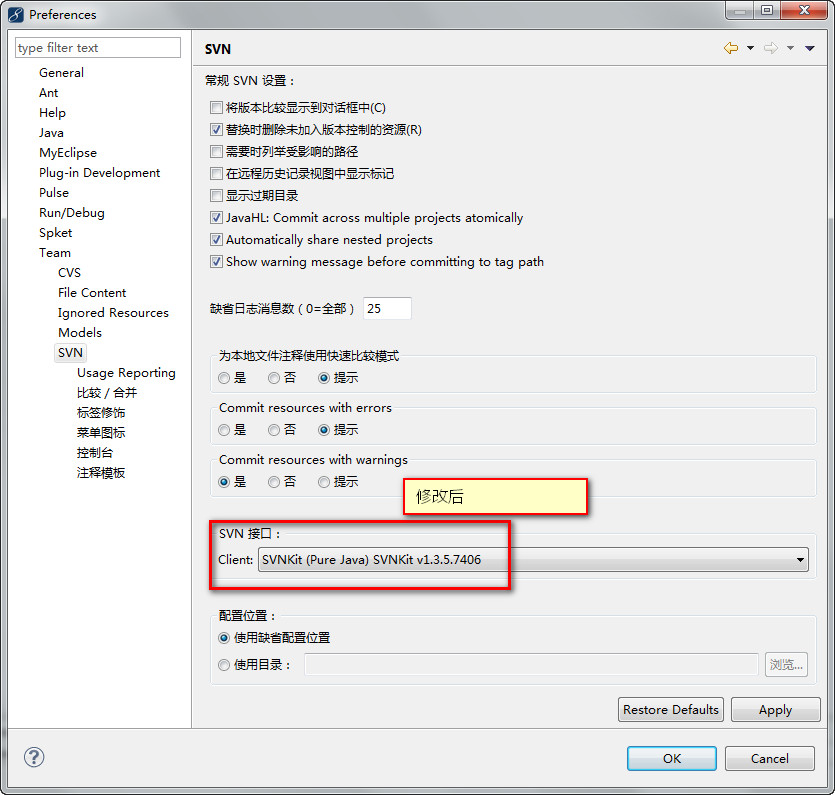
\* 域名"www.esunego.com"

\* **@return**

\*/

**public** **void** getAndSaveDataofYestoday(String strDomainName)

# [Failed to load JavaHL Library解决方法](http://www.blogjava.net/icewee/archive/2013/03/18/396594.html)

进来的看官使用的是win7吧？！是64位的吧！？安装了eclipse的subclipse插件了吧！每次用到SVN插件时都会弹出如下的对话框，虽然不影响使用但是很不爽是不是啊？LZ也是一个有丁点儿强迫症的人，我想干掉这个弹出框！  
  
稍后上图，服务器估计挂了！！  
  
解决方法：  
Window-Preferences-Team-SVN，在SVN接口的下拉框可以看到，默认选择的是JavaHL(JNI) Not Available，手动更改为SVNKit(Pure Java) SVNKit v1.3.5.7406，OK，enjoy it！  
  
错误提示框：  
  
  
修改前：  
  
  
修改后：  


# [关于实体类对象与XML的转换](http://blog.csdn.net/sdsky1987/article/details/7230461)

使用 XStream不用任何映射就能实现多数 Java 对象的序列化，却不关心序列化/逆序列化的类的字段的可见性，序列化/逆序列化类的字段也不需要 getter 和 setter 方法，序列化/逆序列化的类不需要有默认构造函数。

使用 XStream 序列化的类不需要实现 Serializable 接口。

在生成的 XML中对象名变成了元素名，类中的字符串组成了 XML中的元素内容。

XStream 是一种序列化工具而不是数据绑定工具，不能从XML 或者 XML Schema Definition (XSD) 文件生成类。

本例实现了以下4种基本功能：

1. object类型转换为xml类型，在控制台打印

2. xml类型转换为object类型，在控制台打印

3. 将object类型转换为xml类型，并写入XML文件

4. 读取XML文件，加载进相应Object类型

=================================================================================================

1。加载 XStream 需要的jar包 xstream-1.4.2.jar

xpp3\_min-1.1.4c.jar

xmlpull-1.1.3.1.jar

2。创建实体类User.java

**[java]** [view plaincopyprint?](http://blog.csdn.net/sdsky1987/article/details/7230461)

1. **package** com.entity;
3. /\*\*
4. \*
5. \* 实体类
6. \*
7. \* @author 莫小哆\_ly 2012-2-3
8. \*/
9. **public** **class** User {
11. **private** String userName;
13. **private** String passWord;
15. **private** String registeredTime;
17. **public** User() {
19. }
21. **public** User(String userName, String passWord, String registeredTime) {
22. **this**.userName = userName;
23. **this**.passWord = passWord;
24. **this**.registeredTime = registeredTime;
25. }
27. **public** String getUserName() {
28. **return** userName;
29. }
31. **public** **void** setUserName(String userName) {
32. **this**.userName = userName;
33. }
35. **public** String getPassWord() {
36. **return** passWord;
37. }
39. **public** **void** setPassWord(String passWord) {
40. **this**.passWord = passWord;
41. }
43. **public** String getRegisteredTime() {
44. **return** registeredTime;
45. }
47. **public** **void** setRegisteredTime(String registeredTime) {
48. **this**.registeredTime = registeredTime;
49. }
50. }

3。编写功能实现类

**[java]** [view plaincopyprint?](http://blog.csdn.net/sdsky1987/article/details/7230461)

1. **package** com.switcher;
3. **import** java.io.FileInputStream;
4. **import** java.io.FileNotFoundException;
5. **import** java.io.FileOutputStream;
6. **import** java.text.SimpleDateFormat;
7. **import** java.util.Date;
9. **import** com.entity.User;
11. **import** com.thoughtworks.xstream.XStream;
13. /\*\*
14. \*
15. \* 使用 XStream不用任何映射就能实现多数 Java 对象的序列化。
16. \*
17. \* 在生成的 XML中对象名变成了元素名，类中的字符串组成了 XML中的元素内容。
18. \*
19. \* 使用 XStream 序列化的类不需要实现 Serializable 接口。
20. \*
21. \* XStream 是一种序列化工具而不是数据绑定工具
22. \*
23. \* 不能从XML 或者 XML Schema Definition (XSD) 文件生成类。
24. \*
25. \* @author 莫小哆\_ly 2012-2-3
26. \*/
27. **public** **class** SwitchXML {
29. /\*
30. \* XStream 不关心序列化/逆序列化的类的字段的可见性。
31. \*
32. \* 序列化/逆序列化类的字段不需要 getter 和 setter 方法。
33. \*
34. \* 序列化/逆序列化的类不需要有默认构造函数。
35. \*/
36. **private** XStream xStream = **new** XStream();
38. **private** String  xmlPath = "D:/userdata.xml";
40. // 1. obj -> xml(object类型转换为xml类型)
41. **public** String printXML(User user) {
42. System.out.println("obj -> xml");
43. String xml = xStream.toXML(user);
44. System.out.println(xml);
45. **return** xml;
46. }
48. // 2. xml->obj(xml类型转换为object类型，并打印)
49. **public** **void** printObj(String xml) {
50. System.out.println("xml -> obj");
51. User u = (User) xStream.fromXML(xml);
52. System.out.println(u.getUserName() + " " + u.getPassWord() + " " + u.getRegisteredTime());
53. }
55. // 3. 将object类型转换为xml类型，并写入XML文件(其他格式也可以，比如txt文件)
56. **public** **void** writerXML(User user) {
57. **try** {
58. FileOutputStream fs = **new** FileOutputStream(xmlPath);
59. xStream.toXML(user, fs);
60. } **catch** (FileNotFoundException e) {
61. e.printStackTrace();
62. }
63. }
65. // 4. 读取XML文件，加载进相应Object类型
66. **public** **void** readerXML() {
67. User user = **new** User();
68. FileInputStream fis = **null**;
69. **try** {
70. fis = **new** FileInputStream(xmlPath);
71. } **catch** (FileNotFoundException e) {
72. e.printStackTrace();
73. }
74. xStream.fromXML(fis, user);
75. // 打印对象信息
76. System.out.println(user.toString());
77. // 打印对象的属性值
78. System.out.println(user.getUserName() + "-" + user.getPassWord() + "-"
79. + user.getRegisteredTime());
80. }
82. **public** **static** **void** main(String[] args) {
83. // 用户名
84. String username = "admin";
85. // 密码
86. String password = "admin";
87. // 注册时间
88. Date now = **new** Date();
89. SimpleDateFormat format = **new** SimpleDateFormat("yyyyMMddHHmmss");
90. String registeredtime = format.format(now);
92. User user = **new** User(username, password, registeredtime);
93. SwitchXML switchXML = **new** SwitchXML();
95. // 1. object类型转换为xml类型，在控制台打印
96. String xml = switchXML.printXML(user);
97. System.out.println("---------------------");
98. // 2. xml类型转换为object类型，在控制台打印
99. switchXML.printObj(xml);
100. System.out.println("---------------------");
101. // 3. 将object类型转换为xml类型，并写入XML文件
102. switchXML.writerXML(user);
103. System.out.println("---------------------");
104. // 4. 读取XML文件，加载进相应Object类型
105. switchXML.readerXML();
107. }
108. }

运行即可

====================================================================

完整源码 <http://download.csdn.net/detail/sdsky1987/4042378>

附带jar包

# String date转换

## String-date

1. String dateString = "2012-12-06 ";
2. **try**
3. {
4. SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd ");
5. Date date = sdf.parse(dateString);
6. }
7. **catch** (ParseException e)
8. {
9. System.out.println(e.getMessage());
10. }

## date-string

DateFormat df = **new** SimpleDateFormat("yyyy-MM-dd HH:mm:ss");// 设置显示格式

String nowTime= df.format(date);

# 关于定时器

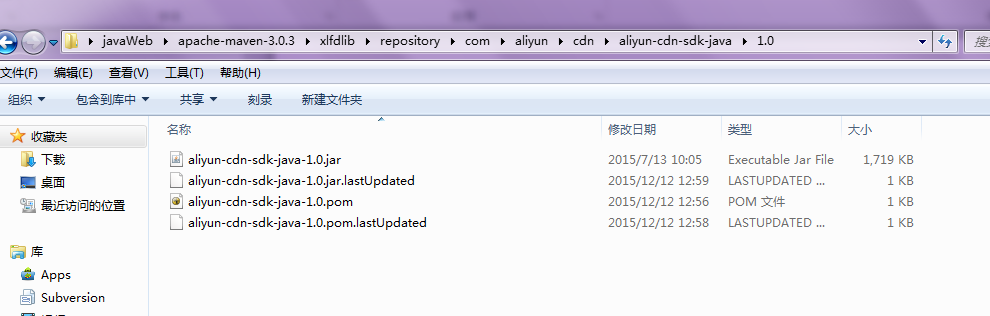
1. /\*\*
2. \* 定时计算。每天凌晨 01:00 执行一次
3. \*/
4. @Scheduled(cron = "0 0 1 \* \* \*")

# 添加maven库

## 命令

mvn install:install-file -Dfile=D:/lib/CCP\_REST\_SDK\_JAVA\_v2.7r.jar -DgroupId=com.cloopen.rest -DartifactId=CCP\_REST\_SDK\_JAVA -Dversion=2.7 -Dpackaging=jar

## 自己创建也行



<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>com.aliyun.cdn</groupId>

<artifactId>aliyun-cdn-sdk-java</artifactId>

<version>1.0</version>

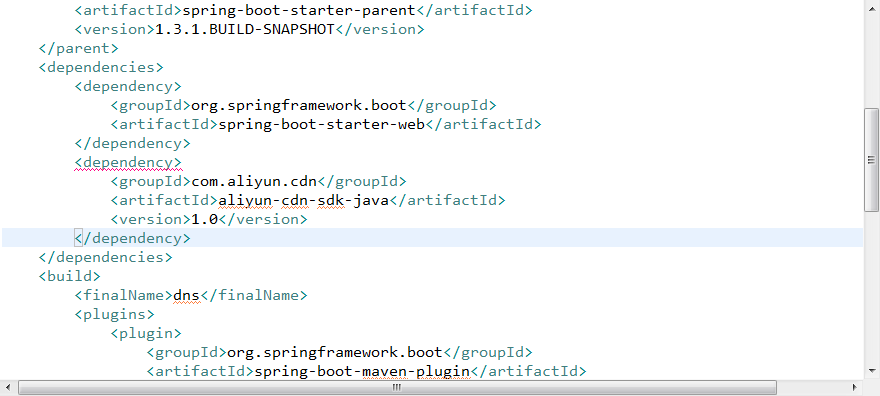
<packaging>jar</packaging>

<name>Aliyun Open Services SDK for Java</name>

<url>http://www.aliyun.com</url>

</project>

在pom.xml中



## 加入源以便查看接口

右键maven sdk属性加入

# STS

编译时用clean package,启动服务器用run on application(在main函数上);

# CDN

|  |
| --- |
| CDN   1. 刷新cdn缓存   两个域名下的所有路径 |
| 2、下载日志  访问日志,oss里找,随便放file中 |
| 1. 下载监控数据   cdn api中的接口描述与最新的java sdk cdn20141111驴唇不对马嘴,有无最新版的手册?如果有,请给出地址,  如果没有,请解释response接口中的DescribeCdnMonitorDataResponse中为何只有一个getRequestID而没有得到资源信息的函数?  如果使用DescribeOneMinuteDataRequest接口,请解释必须给的request参数有哪些 |

## 问题

把资源监控数据存到数据库

日志监控增加新路径和关键字

yigou项目的bug

RefreshObjectCaches刷新节点上的文件内容。刷新制定URL内容指

Cache节点，每次只能提交一个URL

## 调用方式

对CDN API接口调用是通过向CDN API的服务端地址发送HTTP GET请求，并按照接口说明在请求中加入相应

请求参数来完成的；根据请求的处理情况，系统会返回处理结果。

## 服务地址

CDN API的服务接入地址为：cdn.aliyuncs.com

支持通过HTTP或HTTPS通道进行请求通信。为了获得更高的安全性，推荐您使用HTTPS通道发送请求。

请求方法

支持HTTP GET方法发送请求，这种方式下请求参数需要包含在请求的URL中。

请求参数

每个请求都需要指定要执行的操作，即Action参数（例如CreateCDNServer），以及每个操作都需要包含的公

共请求参数和指定操作所特有的请求参数。

# 腾讯云服务器密码等

## 阿里云服务器终端管理密码

无线密码

ygyg1344

**230**

3f0f5abbef8d3debf1e4a18b71646831

**148**

204e2dd75209399eda26397d36f99fc5

内网地址  
yangguang.mysql.rds.aliyuncs.com  
yigou  
517a86375\_548826ef2  
readyigou  
f046cb518\_4f4

**610012**



root   
517a86375548826ef23dAB8d63d147

姜阳 18:32:17

别名

ip地址或域名

配置

用途

用户名密码

demo服务器

115.159.123.222

2c4g

华东区-上海

root cetc15suo!

sunego-jd

119.29.93.64

2c4g1m

华南区-广州 拉取京东数据的工程，从数据库

root

6f030700ecd23885e36038e60f563109

mysql nh6QAew1Wr82d 从数据库

yigou 132653d34a8bec25845dc4988eb0abe7

新门户网站

119.29.2.54

4c8g5m

华南区-广州 门户网站，主数据库

root

e3fbec2ead9cc8f69ef64fcadbb2028f

mysql nh6QAew1Wr82d 主数据库

yigou ec67911eae9b81ad1040891238395237

B2C交易系统-

119.29.115.119

4c8g5m

华南区-广州 两个tomcat，一个供前端使用，一个供生成静态页面使用

root

2a0fbaf046cb5184f4d0a3b71f4bbf10

yigou

517a86375548826ef23d188d63d147b3

论坛测试服务器

182.254.217.12

root

C52T3K4kkFqMR

阿里云账号：zhangman@sunnybs.com 密码sunnyego17

数据库账号密码yigou\_1

云服务器密码Aliyunaliyun1234567890

119.28.93.64: xlfd460089744$

地区                         主机名                     外网ip                        密码  
华东区-上海            demo服务器           115.159.123.222         root cetc15suo!

mysql npants0703\*\*\*  
华南区-广州            sunego-jd-2c4g      119.29.93.64        10.104.24.221

  root 96e79218965eb72c92a549dd5a330112  mysql nh6QAew1Wr82d yigou   132653d34a8bec25845dc4988eb0abe7从服务器  
华南区-广州           新门户网站-4c8g5m      119.29.2.54            10.104.21.57

  root 98dd0448054d7142fe6511defb1d5a6a  mysql nh6QAew1Wr82d yigou ec67911eae9b81ad1040891238395237  主服务器  
华南区-广州            B2C交易系统-4c8g5m      119.29.115.119

root f99069f4a376a0c56be243c59ca17bd8 yigou 517a86375548826ef23d188d63d147b3    两个tomcat所在

admin管理页面 jiangyang jy@5610660 论坛密码

dao.entity新建一个类,会自动在数据库创建表

# 阿里云

|  |
| --- |
| [https://buy.aliyun.com/rds#/prepay](qq://txfile/) |
| 鸿钧 9:03:27  [https://buy.aliyun.com/rds#/postpay](qq://txfile/) |
| 鸿钧 9:04:21  [http://help.aliyun.com/knowledge\_detail.htm?knowledgeId=5975111](qq://txfile/) |

# 各种需要注意的

**登陆** 数据库admin改密码后,需要前台登陆一下刷下,后台才能登陆.

提交代码,改了哪个提交哪个

更新的时候全部更新

写完代码记得ctrl+shift+o;ctrl+shift+f

函数注释 /\*\*回车会自动出模板

提交svn代码时

target  
settings  
.project

过滤,不提交

代码都是全量更新  
部分提交  
改哪个文件就提交哪个  
而且要频繁提交  
改一部分测试通过就提交一部分

# 各种问题

## 包冲突各种找不到符号

ctrl+shift+o 智能导入包

然后yigou-project-clean

## [ERROR] Failed to execute goal on project yigou: Could not resolve dependencies for project com.sunego.commerce:yigou:war:1.0.0-SNAPSHOT: Could not find artifact com.cloopen.rest:CCP\_REST\_SMS\_SDK:jar:v2.7r in central (http://repo1.maven.org/maven2) -> [Help 1]

mvn install:install-file -Dfile=D:/lib/CCP\_REST\_SDK\_JAVA\_v2.7r.jar -DgroupId=com.cloopen.rest -DartifactId=CCP\_REST\_SDK\_JAVA -Dversion=2.7 -Dpackaging=jar

如果还不行,maven配置里的settings要用yiogu项目里的,不能用maven自带的

不行,还是得用自己的

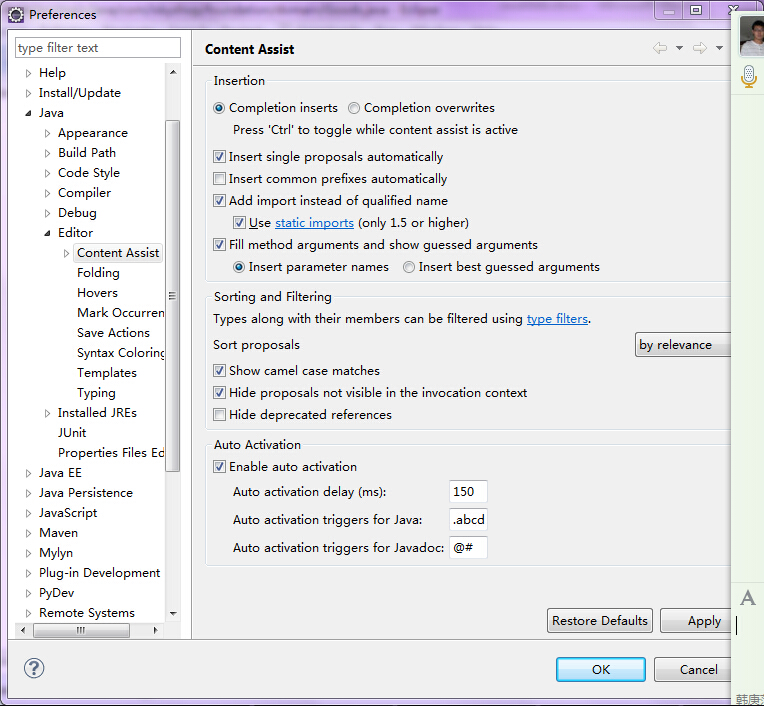
## 乱码

要改字体general-workspace改成utf-8

## 不自动提示

要么按alt+?

腾讯云监控



Auto activation triggers for java 改成.abcdef….xyz

# 项目

## 关于数据库

### 配置哪个数据库

要看打包后的properties,来确定是用的哪个数据库

D:\desktop\programs\workspace\yigou\target\xxproperties

### clean了之后,要重新考一遍

## yigou项目

admin 用户名是admin,密码是yangguang

## Maven项目

### 基础

#### 什么是Maven仓库

在不用Maven的时候，比如说以前我们用Ant构建项目，在项目目录下，往往会看到一个名为/lib的子目录，那里存放着各类第三方依赖jar文件，如log4j.jar，junit.jar等等。每建立一个项目，你都需要建立这样的一个/lib目录，然后复制一对jar文件，这是很明显的重复。重复永远是噩梦的起点，多个项目不共用相同的jar文件，不仅会造成磁盘资源的浪费，也使得版本的一致性管理变得困难。此外，如果你使用版本管理工具，如SVN（你没有使用版本管理工具？马上试试SVN吧，它能帮你解决很多头疼的问题），你需要将大量的jar文件提交到代码库里，可是版本管理工具在处理二进制文件方面并不出色。

Maven仓库就是放置所有JAR文件（WAR，ZIP，POM等等）的地方，所有Maven项目可以从同一个Maven仓库中获取自己所需要的依赖JAR，这节省了磁盘资源。此外，由于Maven仓库中所有的JAR都有其自己的坐标，该坐标告诉Maven它的组ID，构件ID，版本，打包方式等等，因此Maven项目可以方便的进行依赖版本管理。你也不在需要提交JAR文件到SCM仓库中，你可以建立一个组织层次的Maven仓库，供所有成员使用。

简言之，Maven仓库能帮助我们管理构件（主要是JAR）。

### Eclipse导入maven项目及配置

当自己越来越多的接触到开源项目时，发现大多数的开源项目都是用maven来够建的。并且在开发应用时，也越来越意识到maven的确会解决很多问题，如果你要了解maven，可以参考：[Maven入门指南（一）](http://ifeve.com/maven-1/)和 [Maven入门指南（二）](http://ifeve.com/maven-2/)。所以自己也学着使用maven。以下均参考的互联网上的内容，有的做了一些小的修改，并注明了出处。

第一部分：[原文链接](http://www.blogjava.net/fancydeepin/archive/2012/07/13/382965.html)

### maven3 安装：

    安装 Maven 之前要求先确定你的 JDK 已经安装配置完成。Maven是 Apache 下的一个项目，目前最新版本是 3.0.4，我用的也是这个。

    首先去官网下载 Maven：<http://www.apache.org/dyn/closer.cgi/maven/binaries/apache-maven-3.0.4-bin.tar.gz>

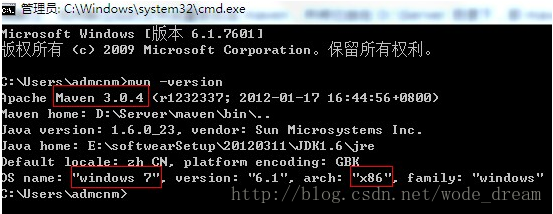
    下载完成之后将其解压，我将解压后的文件夹重命名成 maven，并将它放在 D:\Server 目录下，即 maven 最终的路径是：D:\Server\maven

配置 maven 环境变量：

    系统变量：MAVEN\_HOME = D:\Server\maven

    用户变量：path = %MAVEN\_HOME%\bin

    相信大家都有配过环境变量的，详细步骤就不说了，对着把属性名和属性值配上的OK了。

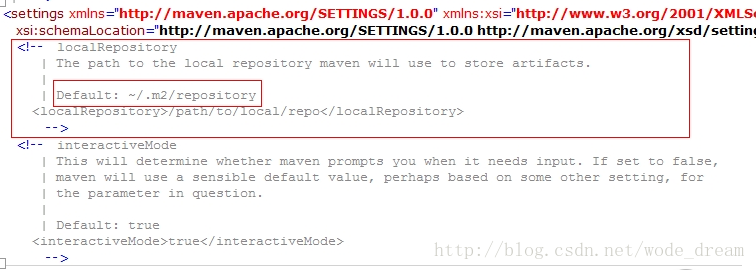
    打开 cmd，在里面敲：mvn -version  
  
 

    如果能打印如上信息，说明到此 Maven3 已经在你的电脑上安装完成。  
    mvn 是 mavn 的一个指令，mvn -version 是查看版本信息，我的操作系统是 32位的 WIN7，安装的 maven 是 3.0.4

    如果能打印如上信息，说明到此 Maven3 已经在你的电脑上安装完成。

修改 maven 仓库存放位置：

    找到 maven 下的 conf 下的 settings.xml 配置文件，我的是在 D:\Server\maven\conf\settings.xml

  
  
    maven 的仓库默认是放在本地用户的临时文件夹下面的 .m2 文件夹下的 repository 下，我的是在 C:\Users\admcnm\.m2\repository 目录下，

    现在我们来修改将它指定到我们自己的路径下，我现在要将仓库指定到 D:\Repositories\Maven 目录下，只需要将上面注销的本地仓库打开，

    然后把相应的路径值写到里面去就行了：



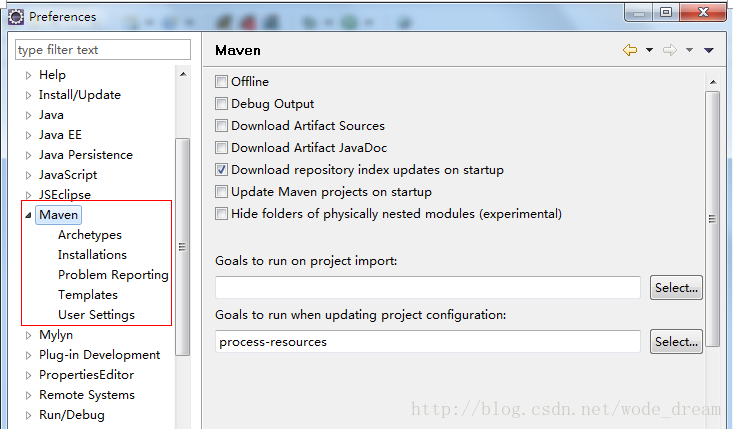
OK，先来体会一下 maven，在 cmd 中敲并回车执行：mvn help:system

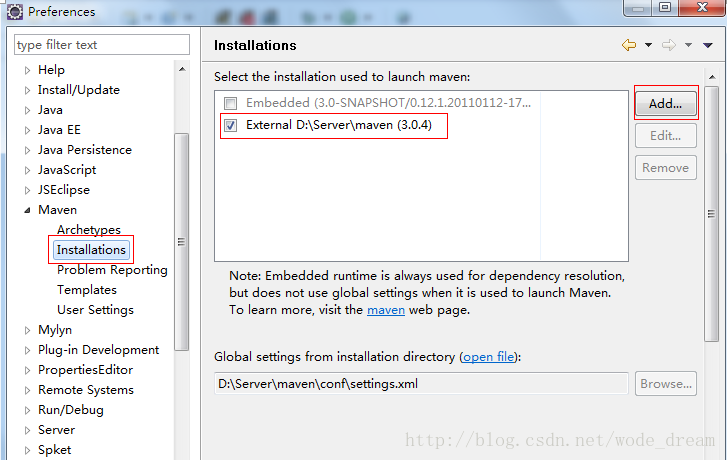
    这时候 maven 就会从远程仓库开始下载一大堆的东西，没事，让它下载着，迟早都要下载的，接下来是在 eclipse 中安装 maven 插件。

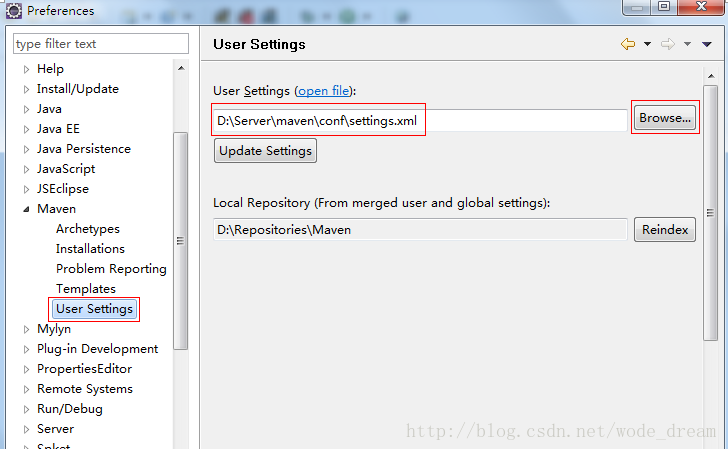
然后在线安装eclipse插件：：Help  -->  Install New Software，然后输入 HTTP 地址来安装（http://download.eclipse.org

），可以在此链接中找相对应的版本。

最后：

检查 eclipse 的 maven 插件是否安装成功：Window  -->  Preferences  
  


配置 maven：  
    1. 点击 Add 按钮，选到你本机安装 maven 的路径值  
  


    2. 点击 Browse 按钮，选到你 maven 的 setting.xml 配置文件，然后点击 OK，这样就完成了 eclipse maven 插件的配置  


### 二、在eclipse安装maven插件时，可能会遇到的错误：[原文链接1](http://zhidao.baidu.com/link?url=spA2z2ILX924pms0uEgLx9cq73aa7yfRDkUKHSnD8xJ2_-T7QuFpqZeYxyFaGlsLlfg6iNmS8HolVJCVeZxI5_) ，[原文链接2](http://bbs.csdn.net/topics/390824532)

1.安装m2eclipse的时候，遇见这种错误如何解决？错误请见详细信息。

问题：Cannot complete the install because one or more required items could not be found.

Software being installed: Maven Integration for Eclipse (Required) 0.12.1.20110112-1712 (or

g.maven.ide.eclipse.feature.feature.group 0.12.1.20110112-1712)

Missing requirement: Maven Project Model Edit Bundle 0.12.1.20110112-1712 (org.maven.id

e.eclipse.maven\_model\_edit 0.12.1.20110112-1712) requires 'bundle org.eclipse.emf.ecore 0.

0.0' but it could not be found

Cannot satisfy dependency:

From: Maven Integration for Eclipse (Required) 0.12.1.20110112-1712 (org.maven.ide.eclip

se.feature.feature.group 0.12.1.20110112-1712)

To: org.maven.ide.eclipse.maven\_model\_edit [0.12.1.20110112-1712] 他说有个文件找不到

。但是到Maven的m2e的链接中，相关文件又是存在的。试了很多次了。都有这个问题。

**解决：**先连接Helios Update site (Helios - <http://download.eclipse.org/releases/helios>)，安装

插件Graphical Editing Framework Zest Visualization Toolkit SDK。然后再安装 M2Eclipse plug-ins 。

2**：如果装的插件版本太高会报**

Cannot complete the install because one or more required items could not be

 found.

  Software being installed: m2e - Maven Integration for Eclipse (includes I

ncubating components) 1.5.0.20140606-0033 (org.eclipse.m2e.feature.feature.

group 1.5.0.20140606-0033)

  Missing requirement: Maven Integration for Eclipse 1.5.0.20140606-0033 (o

rg.eclipse.m2e.core 1.5.0.20140606-0033) requires 'bundle com.google.guava

 [14.0.1,16.0.0)' but it could not be found

  Cannot satisfy dependency:

    From: Maven Integration for Eclipse 1.5.0.20140606-0033 (org.eclipse.m2

e.core.ui 1.5.0.20140606-0033)

    To: bundle org.eclipse.m2e.core [1.5.0,1.6.0)

  Cannot satisfy dependency:

    From: m2e - Maven Integration for Eclipse (includes Incubating componen

ts) 1.5.0.20140606-0033 (org.eclipse.m2e.feature.feature.group 1.5.0.201406

06-0033)

答案：这种情况换一个低版本的安装，如

<http://download.eclipse.org/technology/m2e/milestones/1.4>

### 用maven创建新项目

在要创建项目的目录执行

D:\desktop\programs\workspace\springmvc>mvn archetype:generate -DgroupId=imooc-a

rthur -DartifactId=**spring-mvc-study** -DarchetypeArtifactId=maven-archetype-webapp

其中spring-mvc-study是名称

### 跳过单元测试

clean install -Dmaven.test.skip=true

## tomcat

### 配置注意事项

配环境变量是为了启动服务,如果不需要服务只调试,那么什么都不用配置

window-preference-servers-run environment—新建

### Server等待启动超时时间

D:\Program Files\eclipse\workspace\.metadata\.plugins\org.eclipse.wst.server.core\server.xml

minIdle=1

或者双击server那个也可以直接界面改

## eclipse

runat server用tomcat时,实际上是将工程下边target下的web工程考到

workspace\.metadata\.plugins\org.eclipse.wst.server.core去

D:\Program Files\eclipse\workspace\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps

不行的话各种删除target什么的clean一下重新来过,server中要加入yigou才行

### ctrl+m,最大话窗口,搜索用file

### 格式化

ctrl+i

## Linux服务器(系统centos6.4 final x64)

### 路由器密码

13441344

### 外网地址(重启路由器要变)

sunego.org或www.sunego.org

### 帐号密码

root yigou密码均为yigou1344 vnc两个端口:1,:2,密码一样

svn lihailong = 2fde2fr41

路由器密码 npants0703\*\*\*

无线密码T\_TSoS110<\_>

服务器Root密码 npants0703\*\*\*

Yigou用户密码 npants0703\*\*\*

服务器Mysql root密码 npants0703\*\*\*

### 网址

外网地址改为域名[sunego.com.cn](http://sunego.org:8889/)

jira地址：

<http://sunego.com.cn:18889>

Confluence地址：

<http://sunego.com.cn:18090/>

Jenkins

<http://sunego.com.cn:18380/jenkins>

sonar

<http://sunego.com.cn:19000>

nexus maven私服

<http://sunego.com.cn:18044>

ftp

<ftp://sunego.com.cn:121>

ssh

sunego.com.cn:8322

其它机器如110,107等在登上此机器之后使用ssh即可登陆

mstsc

sunego.com.cn:13389

svn地址：

svn://460089744xlfd.xicp.net:3690

sonar

登陆用户名：yigou，密码：sonar1344

http:// sunego.org:9000/

Jenkins

高级管理员：admin  密码：yi#$gou@1915

http://sunego.org:8180/jenkins/

ftp地址：

ftp://sunego.org

### 店铺等级记录

表名: iskyshop\_storegrade\_log 店铺等级记录

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 空否 | 字段描述 |
| **id** | bigint(20) | N | 主键，自动加1 |
| addTime | datetime | Y | 添加日期 |
| deleteStatus | bit(1) | N | 删除标记  0 不删除(默认)  1 删除 |
| log\_edit | bit | 1 | 编辑记录 |
| store\_grade\_info | varchar | 255 | 审核信息 |
| store\_grade\_status | int | 11 | 审核状态 |
| store\_id | bigint | 20 | 外键—店铺id  对应表---iskyshop\_store |

### Jira启动路径

/opt/atlassian/jira/bin 用root启动

### Confluence启动路径

/opt/atlassian/confluence/bin用root启动

### sonar启动路径

### Jenkins启动路径

/home/yigou/app/apache-tomcat-7.0.54-jenkins/bin/startup.sh 用yigou启动

### 已开放端口

#### ftp-21

#### ssh-22

#### rdp-3389

#### mysql-3306

#### vnc-5901 5902对应两个帐号yigou yigou2

# 源码view

分 manage和 view两部分

## Manage-管理

分admin,buyer,seller三部分

### Admin-管理员

#### Action

##### ActivityManageAction-活动管理

##### AdminManageAction-管理员管理

##### AdvertManageAction-广告管理

##### AreaManageAction-地区管理

##### ArticleClassManageAction-文章分类管理

##### ArticleManageAction-文章管理

##### BargainManageAction-特价管理

##### BaseManageAction-基础管理

###### 用户登录

###### …

##### CacheManageAction-缓存管理

##### CombinManageAction-组合销售管理

##### ComplaintManageAction-投诉管理

##### ComplaintSubjectManageAction-投诉主题管理

##### ConsultManageAction-咨询管理

##### CouponManageAction-优惠券管理

##### DatabaseManageAction-数据库管理

##### DeliveryManageAction-买就送管理

##### DocumentManageAction-系统文章管理

##### EvaluateManageAction-商品评价管理

##### ExpressCompanyManageAction-快递管理

##### GoldRecordManageAction-金币购买记录管理

##### GoodsBrandManageAction-商品品牌管理

##### GoodsClassManageAction-商品分类管理

##### GoodsFloorManageAction-楼层分类管理

##### GoodsManageAction-商品管理

##### GoodsSpecificationManageAction-商品规格管理

##### GoodsTypeManageAction-商品类型管理

##### GroupAreaManageAction-团购区域管理

##### GroupClassManageAction-团购分类管理

##### GroupManageAction-团购管理

##### GroupPriceRangeManageAction-团购价格区间表管理

##### ImageManageAction-会员相册管理

##### IntegralGoodsManageAction-积分礼品管理

##### IntegralLogManageAction-积分明细管理

##### LuceneManageAction-全文检索设置管理

##### NavigationManageAction-页面导航管理

##### OperationManageAction-基本设置管理

##### OrderManageAction-订单设置管理

##### PartnerManageAction-合作伙伴管理

##### PaymentManageAction-支付方式管理

##### PredepositCashManageAction-提现申请管理

##### PredepositLogManageAction-预存款明细管理

##### PredepositManageAction-预存款管理

##### ReportManageAction-未处理举报管理

##### ReportSubjectManageAction-举报主题管理

##### ReportTypeManageAction-举报类型管理

##### SpareGoodsFloorManageAction-闲置商品楼层管理

##### SpareGoodsManageAction-闲置商品管理

##### StoreClassManageAction-店铺分类管理

##### StoreGradeManageAction-店铺等级管理

##### StoreManageAction-店铺管理

##### TemplateManageAction-模板管理

##### TransAreaManageAction-运费地图管理

##### UcenterManageAction-UC配置管理

##### UserManageAction-会员管理

##### ZtcManageAction-直通车管理

#### Tools

##### AreaManageTools-区域管理工具

##### BargainManageTools-交易管理工具

##### GoodsFloorTools-商品管理工具

##### MsgTools-信息(通知)管理工具

##### PaymentTools-支付管理工具

##### StatTools-信息收集管理工具

##### StoreTools-存储管理工具

##### UserTools-用户管理工具

### Buyer-买家

#### Ations

##### AccountBuyerAction-个人信息管理

##### AddressBuyerAction-买家收货地址管理

##### BaseBuyerAction-买家中心管理

##### ComplaintBuyerAction-买家投诉管理

##### ConsultBuyerAction-买家咨询管理

##### CouponBuyerAction-优惠券管理

##### FavoriteBuyerAction-商品收藏管理

##### HomePageBuyerAction-个人主页管理

##### IntegralOrderBuyerAction-买家订单管理

##### MessageBuyerAction-站内短信管理

##### OrderBuyerAction-买家订单管理

##### PredepositBuyerAction-会员充值管理

##### PredepositCashBuyerAction-提现管理

##### ReportBuyerAction-买家举报管理

### Seller-卖家

#### Ations

##### ActivitySellerAction-活动管理

##### AdvertSellerAction-广告管理

##### AlbumSellerAction-相册管理

##### BargainSellerAction-今日特价管理

##### BaseSellerAction-卖家中心管理

##### CombinSellerAction-组合销售管理

##### ComplaintSellerAction-卖家投诉管理

##### ConsultSellerAction-卖家咨询管理

##### DeliverySellerAction-买就送管理

##### GoldSellerAction-金币兑换管理

##### GoodsBrandSellerAction-卖家品牌管理

##### GoodsClassSellerAction-卖家商品分类管理

##### GoodsReturnSellerAction-卖家退货管理

##### GoodsSellerAction-卖家商品管理

##### GroupSellerAction-卖家团购管理

##### OrderSellerAction-卖家订单管理

##### PaymentSellerAction-支付方式管理

##### RefundSellerAction-卖家退款管理

##### SpareGoodsSellerAction-闲置商品管理

##### StoreNavSellerAction-卖家导航管理

##### StorePartnerManageAction-卖家合作伙伴管理

##### StoreSellerAction-店铺管理

##### SubAccountSellerAction-子账户管理

##### TaobaoSellerAction-淘宝相关

##### TransportSellerAction-卖家邮寄管理

##### WaterMarkSellerAction-图片水印管理

##### ZtcSellerAction-卖家直通车

#### Tools

##### BargainSellerTools-卖家交易工具

##### MenuTools-菜单工具

##### TransportTools-邮寄管理

页面都在web-inf-template-zh-cn中

# 逻辑

前端-----接口-----业务-----数据库表

# Java积累

## 常用

**java.lang.Long.parseLong(String s)**方法解析的字符串参数s作为一个符号的十进制长

## 基础

**Action**是管理业务（Service）调度和管理跳转的。

**Service**是管理**具体的功能**的。

**Action**只负责管理，而**Service负责实施**。

**DAO**只完成增删改查，虽然可以1-n，n-n，1-1关联，模糊、动态、子查询都可以。但是无论多么复杂的查询，dao只是封装增删改查。至于增删查改如何去实现一个功能，dao是不管的。

**Spring**负责业务层管理，即Service（或Manager）。

Service为action提供统计的调用接口，封装持久层的DAO；

可以写一些自己的业务方法；

统一的Javabean管理方法；

声明式事务管理；

集成Hibernate。

# 数据库

1. 范围,是否只跟商品相关的? 是
2. 需要整理出什么形式的东西?文档? 商品相关的表的属性等
3. 如果要文档,文档具体内容,表关系? 表关系在代码中如何实现, 一对多,多对一,如多个商品对应一个店铺等等
4. 其它要求

## 数据库表对应的类位置:

Yigou\_base中

com.iskyshop.foundation.domain;

## 数据库配置

D:\desktop\yigou-shop\yigou-web\target\yigou-web\WEB-INF\classes

yigou-web\WEB-INF\classes\jdbc.properties

D:\svn\yigou-conf\src\main\resources\online\jdbc.properties

手动把web文件夹考到

D:\Program Files\eclipse\workspace\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps

时,要检查这个

F:\eclipse\workspace-yigou\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps\yigou-office-web\WEB-INF\classes里面的jdbc有没有正确

run as 是自动考的

### 本地内容

driverClassName=org.gjt.mm.mysql.Driver

url=jdbc:mysql://localhost:3306/yigou?createDatabaseIfNotExist=true&amp;useUnicode=true&amp;characterEncoding=utf-8

username=root

password=

initialSize=1

maxActive=100

maxIdle=8

minIdle=1

### 远程内容(很慢)

driverClassName=org.gjt.mm.mysql.Driver

url=jdbc:mysql://222.129.235.12:3306/yigou?createDatabaseIfNotExist=true&amp;useUnicode=true&amp;characterEncoding=utf-8

username=root

password=yigou1344

initialSize=1

maxActive=100

maxIdle=8

## sql文件目录(整个数据库的)

D:\svn\yigou-web\src\main\webapp\resources\data\test.sql

# 编译:

右键-runas-maven build之后会在target下边打出一个包

右键-runas-maven build2 在里面输入 clean install 是清除

切换workspace

# js等页面请求在 webapp下

# 腾讯云监控

腾讯云119.29.115.241  
用户名：root  
密码：cetc15suo!  
  
MySQL : root  
Password : yangguang  
FTP User : admin  
Password : npants\*\*\*

2.54 root mysqlbk npants0703\*\*\*

mysql nh6QAew1Wr82d

93.64 root xlfd460089744$ mysql nh6QAew1Wr82d yigou密码也一样

119 root: dsp11259375!

地区                         主机名                     外网ip                        密码

华东区-上海            demo服务器           115.159.123.222         root cetc15suo!

华南区-广州            sunego-jd-2c4g      119.29.93.64              root 96e79218965eb72c92a549dd5a330112  mysql nh6QAew1Wr82d yigou  132653d34a8bec25845dc4988eb0abe7从服务器

华南区-广州           新门户网站-4c8g5m      119.29.2.54              root 1c0e10f6d94cdb3284a7840252e05648  mysql nh6QAew1Wr82d yigou ec67911eae9b81ad1040891238395237  主服务器

华南区-广州            B2C交易系统-4c8g5m      119.29.115.119              root f99069f4a376a0c56be243c59ca17bd8 yigou 517a86375548826ef23d188d63d147b3    两个tomcat所在

**更新工程使用的maven库的jar包**

最近在左侧工程名点yigou-office上点击右键  
maven  update projecy

然后再maven install

右键 run as 选9 maven install

# maven-usersetting

用的是代码目录下的xml

先看商品品牌管理

然后看短信模块,msgtools 换一个模块 融联通讯

监控 cpu 内存 mysql apache

# 问题:

编译时整个编译,运行时只运行web

手动测试时,将target下边web工程文件夹拷贝到D:\Program Files\eclipse\workspace\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps下,启动tomcat浏览器访问

如果yigou-office-web不行,试试yigou-web

# 更改workspace,

window-preference搜workspace,把其它的删除,选中prompt那个是为了重启eclipse时选workspace,这样重启之后才有.metadata

# eclipse字体大小设置

在左边的菜单栏中找到general，appearance在general展开后的第一行，点击展开appearance

# eclipse设置编码

preference-workspace-textcode

右键工程,属性-resources改

# svn更新代码

window-showview-others-svn

右键项目-检出为-选作为已存在项目,第二个,否则会弹出选则wizard框

# eclipse提示设置

window-preference-java-editor-contentasistent-autoactivation triiger,改成.abcdefghijklmnopqrstuvwxyz

# tomcat日志设置

为朝阳大悦城线上O2O平台提供数据分析工具，是朝阳大悦城大数据项目的主要工作内容之一。针对线上O2O平台的网站数据分析，可以长期有效的指导O2O平台的运营、优化。信柏科技团队计划为朝阳大悦城提供在线分析平台——信柏SAE平台，并整理了该平台的相关分析业务需求、技术实现要求（详见《朝阳大悦城大数据体系建设整体解决方案》）。信柏SAE平台将采用页面抓取的方式，实时抓取用户页面访问信息，提供相应的分析工作和数据分析结果。

在实际实施过程中，信柏团队SAE系统构建、上线时间晚于朝阳大悦城O2O平台的上线时间。为确保在O2O平台稳定运行时，平台管理和运营团队可以获取相关的网站关键指标分析，信柏团队特设计本过渡方案，计划结合相关分析工具，通过手工日志分析的方式，为朝阳大悦城提供数据分析报告。

本方案作为过渡计划，在设计上存在一定局限性（如实时性、自动化等方面的不足），但在业务上旨在满足网站运营数据分析的基本需求。

方案原理

web服务器日志记录了Web服务器接收处理请求及运行时错误等各种原始信息。通过对日志进行统计、分析和综合，就能有效地掌握服务器的运行状况、发现和排除错误原因、了解客户访问分布等，更好地加强系统的维护和管理。Web服务模式主要有三个步骤：

 1）服务请求：包含用户端的众多基本信息，如IP地址、浏览器类型、目标URL等。

 2）服务响应：Web服务器接收到请求后，按照用户要求运行相应的功能，并将信息返回给用户。如果出现错误，将返回错误代码。

 3）追加日志：服务器将对用户访问过程中的相关信息以追加的方式保存到日志文件中。如图1 。

　图1 Web服务模式主要有三个步骤

Tomcat日志配置方式

默认情况下tomcat访问日志没有打开，配置的方式如下：

编辑 ${catalina}/conf/server.xml文件（注:${catalina}是tomcat的安装目录），在server.xml里的<host>标签下，把以下的注释(<!-- -->)去掉即可。

<!-- <Valve className="org.apache.catalina.valves.AccessLogValve"

directory="logs"  prefix="localhost\_access\_log." suffix=".txt" pattern="common" resolveHosts="false"/> -->

1）其中 directory是产生的目录 tomcat安装${catalina}作为当前目录

2）pattern表示日志生产的格式，common是tomcat提供的一个标准设置格式“ %h %l %u %t "%r" %s %b”。本次项目建议采用以下具体的配置“%h %l %u %t "%r" %s %b %T ”，因为标准配置有一些重要的日志数据无法生。

Tomcat日志标准

从业务角度，web server可以分为两部分日志：

1）运行日志：它主要记录运行的一些信息，尤其是一些异常错误日志信息。

2）访问日志：访问日志，它记录的访问的时间，IP，访问的资料等相关信息。

具体的日志产生样式说明如下(从官方文档中摘录)：

    \* %a - Remote IP address

    \* %A - Local IP address

    \* %b - Bytes sent, excluding HTTP headers, or '-' if zero

    \* %B - Bytes sent, excluding HTTP headers

    \* %h - Remote host name (or IP address if resolveHosts is false)

    \* %H - Request protocol

    \* %l - Remote logical username from identd (always returns '-')

    \* %m - Request method (GET, POST, etc.)

    \* %p - Local port on which this request was received

    \* %q - Query string (prepended with a '?' if it exists)

    \* %r - First line of the request (method and request URI)

    \* %s - HTTP status code of the response

    \* %S - User session ID

    \* %t - Date and time, in Common Log Format

    \* %u - Remote user that was authenticated (if any), else '-'

    \* %U - Requested URL path

    \* %v - Local server name

    \* %D - Time taken to process the request, in millis

    \* %T - Time taken to process the request, in seconds

There is also support to write information from the cookie, incoming header, the Session or something else in the ServletRequest. It is modeled after the apache syntax:

    \* %{xxx}i for incoming headers

    \* %{xxx}c for a specific cookie

    \* %{xxx}r xxx is an attribute in the ServletRequest

    \* %{xxx}s xxx is an attribute in the HttpSession

Tomcat日志解析

1）原始日志数据：

根据前面配置要求“ %h %l %u %t "%r" %s %b %T”，生产的访问日志数据的业务意义如下：

    \* %h 访问的用户IP地址

    \* %l 访问逻辑用户名，通常返回'-'

    \* %u 访问验证用户名，通常返回'-'

    \* %t 访问日时

    \* %r 访问的方式(post或者是get)，访问的资源和使用的http协议版本

    \* %s 访问返回的http状态

    \* %b 访问资源返回的流量

    \* %T 访问所使用的时间

2）日志数据分析：

根据上述原始日志数据，我们可以根据时间段做以下的分析处理：

  \* 独立IP数统计（IP去重，独立访客数）

  \* 访问请求数统计

  \* 访问资料文件数统计

  \* 访问流量统计

  \* 访问处理响应时间统计

  \* 统计所有404错误页面

  \* 统计所有500错误的页面

  \* 统计访问最频繁页面（页面访问量列表）

  \* 统计访问处理时间最久页面（页面平均相应时间列表）

  \* 统计并发访问频率最高的页面（页面并发访问频次列表）

日志传输方案

1）传输格式

日志按excel格式，用邮件传输，表头字段为

IP地址

逻辑用户名

验证用户名

访问日时

访问方式

返回状态

返回的流量

访问时长

2）传输频率

日志传输频率为24小时一次，每一次传输24小时内的全部日志。

jenkins 账号：admin 密码：yi#$gou@1915

账号：yigou 密码：jenkins1344

sonar 账号：admin 密码：yi#$gou@1915

账号：yigou 密码：sonar1344

svn 账号：fanliwei 密码:fasdfw2d

jira 账号：fanliwei 密码：fanliwei

前台用户名：jy5610660 admin8

后台用户名：admin admin8

路由器密码 ZxAsQw!2?>\_<

无线密码T\_TSoS110<\_>

服务器yigou用户密码 npants0703\*\*\*

FTP： http://123.123.110.193:21

yigou yigou1344

Jl5610660

# 任务

拉取预存款,

## 数据库操作流程

JdOrderApi查询用户余额接口

要写 conf, task,dao,service

dao要建一个表 预存款表 按照dao.entity下任一个格式建java base entity

表字段

日期

金额

dao 和 dao.impl都要建 一个空的即可.:继承.

service层 同上,建

task 在AccountTask里面写 照着checkJdAccount写

定时任务在 suneogoce-task.properties.

Rollback(false):不回滚,即数据库操作如插入之后,就是插入了,提交时要去掉,默认是回滚.

@Test

// @Rollback(false)

**public** **void** saveBalanceTest() {

accountTask.saveBalance();

}

数据库的float,double型要用BigDecimal

转换用BigDecimal d1TobigDe = new BigDecimal(balance);

serviceimpl公共业务逻辑实现类

/\*\*

\* 每天凌晨 12点查询京东余额接口，并保存余额到jd\_balance表中

\*

\*/

@Scheduled(cron = "${job.jd.select.balance}")

**public** **void** saveBalance() {

**double** jdOrderCount=-1;

String token = jdTokenService.getAccessToken();

**try** {

jdOrderCount = JdOrderApi.*selectBalance*(token);

} **catch** (JdSalException e) {

*LOGGER*.info("called selectBalance() 查询京东余额失败" + e.getMessage());

}

Balance balance = **new** Balance();

balance.setBalance(String.*valueOf*(jdOrderCount));

balance.setDate(**new** Date());

jdBalanceService.save(balance);

}

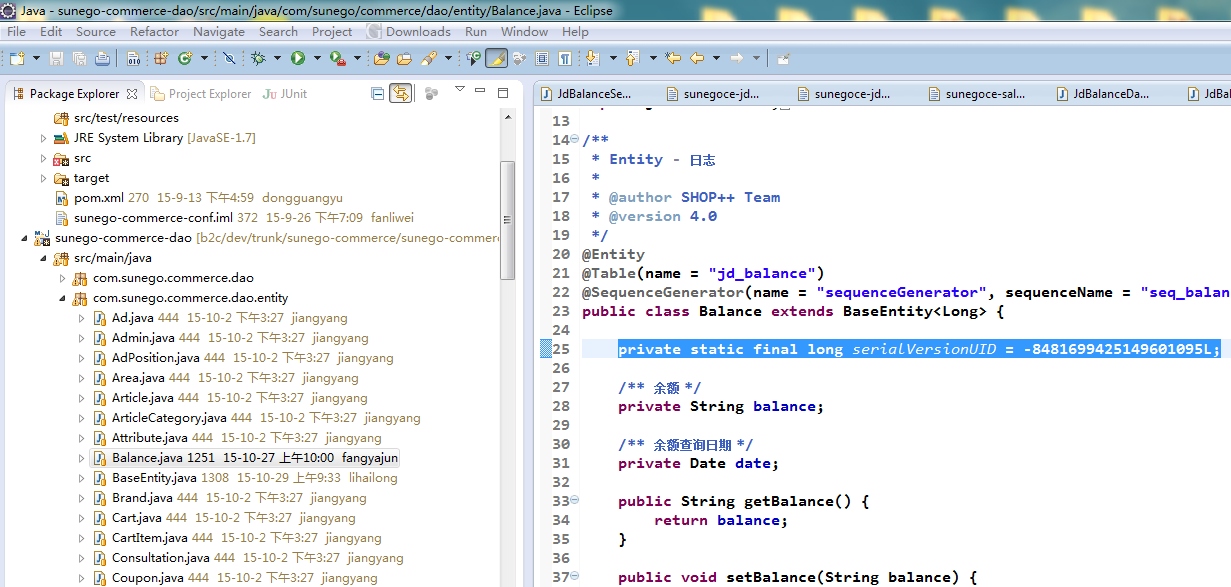
@Autowired

**private** JdBalanceService jdBalanceService;

在类开头定义;

## 问题

### 1 新建实体类的问题



dao实体层如何新建一个类,如果复制的话这个ID不行

,还是复制

### 2 初始化的问题

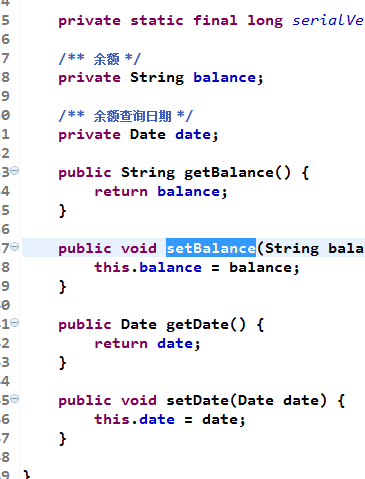
Balance balance = **new** Balance();

实体类这需要new初始化

但jdBalanceService这不需要初始化,为何?什么情况下要new,何时又不需要?service 是注入方式,不需要初始化.类初始化交给工厂.

### 3 字段类型问题

预存款余额为何不用BigDecimal而用String?



什么时候用String,什么时候用BigDemical,跟钱有关的用B.

### 4 maven install 的问题

关于clean install,是否是执行一次就可以,然后test的代码随便改(其它地方代码可以随便改吗)?

一般一次就行

### sql语句中lower的含义,转小写orders别名

**public** JdResultOrder findByOrderId(String orderId) {

**if** (orderId == **null** || orderId.toString() == "") {

**return** **null**;

}

String jpql = "select orders from JdResultOrder orders where lower(orders.orderId) = lower(:orderId)";

**try** {

**return** entityManager.createQuery(jpql, JdResultOrder.**class**).setParameter("orderId", orderId).getSingleResult();

} **catch** (NoResultException e) {

**return** **null**;

}

}

### entity是实体类,里面只放get,set,dao里面是啥?是放一些查询条件什么的么

dao封装一些实体类的操作

### 多条记录如何返回给页面

支付宝查出的记录有多条,如何传给前台 Page类

### @Controller后边的名字是什么意思

@Controller("adminLocalStorageController")

### List和Page有何区别

list是表,Page是分页,用于显示需要翻页的 page就是封装分页信息的list

### sql啥意思

**public** JdResultOrder findByJdOrderId(String jdOrderId) {

**if** (jdOrderId == **null** || jdOrderId.toString() == "") {

**return** **null**;

}

String jpql = "select orders from JdResultOrder orders where lower(orders.jdOrderId) = lower(:jdOrderId)";

**try** {

**return** entityManager.createQuery(jpql, JdResultOrder.**class**).setParameter("jdOrderId", jdOrderId).getSingleResult();

} **catch** (NoResultException e) {

**return** **null**;

}

}

### orderid,jdorderid,sku的关系

### 如何确定是否使用优惠券以及余额

## 显示支付宝信息

### 指示

|  |
| --- |
| 好 |
| 你说的那个问题原理就是set和get |
| **先new一个新page** |
| **把旧page的属性赋值给新page** |
| **旧page的list通过set和get转成新的list** |
| **然后新的list再set给新的page** |

jdresultorder,完成的显示

问题

### 在哪个类中写?或者新建?

service.acount.impl中直接写.

### 查询到了数据放在哪,如何跟后台交互?

在account里改

### @ResponseBody

作用：

      该注解用于将Controller的方法返回的对象，通过适当的HttpMessageConverter转换为指定格式后，写入到Response对象的body数据区。

使用时机：

      返回的数据不是html标签的页面，而是其他某种格式的数据时（如json、xml等）使用；

service.acount.impl中直接写.

dao层的不用写了?只写controller中的一个函数即可?是,找一个dao层的查找.

请求信息是一个网址?是

查询到的是好几个参数,函数如何返回值,是return list吗?返回一个string值

**public** String setting(ModelMap model) {

PluginConfig pluginConfig = localStoragePlugin.getPluginConfig();

model.addAttribute("pluginConfig", pluginConfig);

**return** "/net/shopxx/plugin/localStorage/setting";

}

加值,找有列表的.

也是要写一个account一个.impl? 只在impl中写,

如何测试?

提交svn代码时

target  
settings  
.project

过滤,不提交

/\*\*

\* 列表

\*/

@RequestMapping(value = "/list", method = RequestMethod.*GET*)

**public** String list(Pageable pageable, ModelMap model) {

model.addAttribute("page", adService.findPage(pageable));

**return** "/admin/ad/list";

}

select count(\*) from Member members where lower(members.username) = lower(:username)";

@SuppressWarnings("unchecked")

@Override

**public** List<Order> findList(Date beginDate, Date endDate) {

**if** (**null** == beginDate || **null** == endDate) {

**return** **null**;

}

// 查询从00：00：00到23：59：59时间段的审核通过(订单状态不为“连接超时拒绝”、订单操作日志类型为“审核”)的订单

**try** {

String jsql = "select orders from Order orders,OrderLog orderLogs where orders.status !=8 "

+ " and orders.id = orderLogs.order and orderLogs.createDate>=:beginDate"

+ " and orderLogs.createDate<=:endDate and orderLogs.type = 3";

**return** entityManager.createQuery(jsql).setParameter("beginDate", beginDate)

.setParameter("endDate", endDate).getResultList();

} **catch** (NoResultException e) {

**return** **null**;

}

}

引用接口时要加上这个resource,不然不行,找不到代码

@Resource(name = "jdResultOrderServiceImpl")

**private** JdResultOrderService jdResultOrderService;

page移植时必须用它自己的查询一次

page中的查询，判断为空用isNUll

set是有序不重复的

list是无序且可以重复

failed to lazily initialize a collection of role: net.shopxx.entity.order.Order.payments, could not initialize proxy - no Session

懒加载问题,只会在test中出现,要加注解

@OneToMany(mappedBy = "order", fetch = FetchType.*EAGER*, cascade = CascadeType.*REMOVE*)

@OrderBy("createDate asc")

**public** Set<Payment> getPayments() {

**return** payments;

}

如果是D:\desktop\用户目录\我的文档\Tencent Files\1415532825\Image\C2C\U[33)V5HSRX3C_1L3KK~B4V.png

则

@OneToMany(mappedBy = "order", fetch = FetchType.*LAZY*, cascade = CascadeType.*REMOVE*)