# 1.得到web的 输出流文件(这个可以写下载文件)

OutputStream oStream = response.getOutputStream();

# 2 web的文件资源(images)的路径（可以写下载文件的路径）

String path1 = **this**.getServletContext().getRealPath("/images/1.png");

# 二、软件开发中基本就是两种文件，数据中有关系的，就是用xml文件（一个生一个），库信息没有关系就是properties，

## 1.现在我们写一个库信息（properties文件）db.

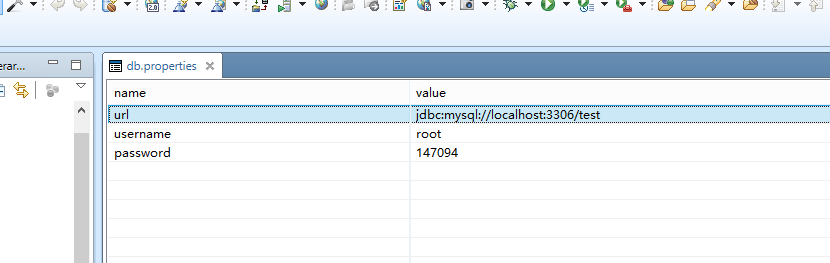
## 2.开始新建一个包---》db.properities ，然后在这个包中建立这个db.properities文件，文件中写入下面的内容

url=jdbc:mysql://localhost:3306/test

username=root

password=147094

## 3.点击properties就会出现下面的东西



### 4.这样我们的库的连接信息就搭建了

## 5.我们开始使用servlet读取这个资源文件，读取资源文件我们用到了



## 6.这里我们要使用servlet来读取这个资源文件，所以先新建一个servlet文件，

### 6.1其对于我们在jsp中的servlet文件是在我们WEB-INF文件夹下面的一个classes文件夹下面的，通过getServletContext().getResourceAsStream，得到资源文件作为流，

#### 6.1.1为了在system中打印出来，是不需要包的名字的，但是如果要是在网页上打印出来就是一定需包的名字的

### 6.2然后为了得到这个properties文件值专门有一个针对他的一个类，名字就是自己本身，properties，实例化了之后加载下输入流中的文件即可

response.setContentType("text/html");

PrintWriter out = response.getWriter();

//因为要读取文件，所以我们首先要的得带的就是文件的输入流

InputStream isInputStream = **this**.getServletContext().getResourceAsStream

(("/WEB-INF/classes/db/properities/db.properties"));

//读取 properities 文件专门有它的方法

Properties properties = **new** Properties();

properties.load(isInputStream);

out.println(properties.getProperty("url"));

out.println(properties.getProperty("username"));

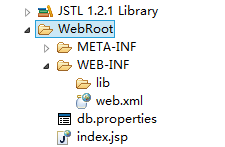
out.println(properties.getProperty("password"));

out.println(**this**.getServletContext().getResourceAsStream("/WEB-INF/classes/db.properties"));

}

# 6.3重新学习下，Android中的抽取方法，当我们有很多的方法中有重复的语句时候，可以将这些语句抽取出来制作成一个方法进行直接调用就可以了

------》先用鼠标选中部分然后 -----》右键----》Refactor -----》抽取发方法即可---》然后给他一个方法的名字7.有的时候我们的文件不在上面的classes中，而是放到了webroot中



**private** **void** test2(PrintWriter out) **throws** IOException {

//因为要读取文件，所以我们首先要的得带的就是文件的输入流

InputStream isInputStream = **this**.getServletContext().getResourceAsStream

("/db.properties");

//读取 properities 文件专门有它的方法

Properties properties = **new** Properties();

properties.load(isInputStream);

out.println((properties.getProperty("url")).toString());

}

# 三、使用文件输入流读取资源文件

## 1.FileInputStream 读取资源的时候是在虚拟机的启动目录下面来读取的。如果要是使用了下面的东西就会出现了错误

//使用fileInpuStream,这文件输入流是在虚拟机的目录下面的，我们一般用这个来使用

**private** **void** test3(PrintWriter out) **throws** IOException {

//因为要读取文件，所以我们首先要的得带的就是文件的输入流

FileInputStream isInputStream = **new** FileInputStream("classes/db.perities");

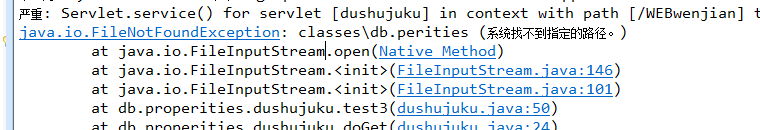
//读取 properities 文件专门有它的方法

Properties properties = **new** Properties();

properties.load(isInputStream);

out.println((properties.getProperty("url")).toString());

}



## 2.在1中已经有了 ，它是虚拟机的启动目录，也就是我们的的tomcat的bin目录下的classes文件夹下面，出现的文件

## 3.因此我们一般不适用这个输入流的方式，而是使用servletContext来进行读取文件的路径

# 四、我们还可结合以上的两种来进行操作

//结合servletContext和FileInputStream 两中方法来进行的得到资源文件

**private** **void** test4(PrintWriter out) **throws** IOException {

//因为要读取文件，所以我们首先要的得带的就是文件的输入流

String isString = (String)(**this**.getServletContext().getRealPath

("/WEB-INF/classes/db/properities/db.properties"));

FileInputStream isInputStream = **new** FileInputStream(isString);

//读取 properities 文件专门有它的方法

Properties properties = **new** Properties();

properties.load(isInputStream);

out.println((properties.getProperty("url")).toString());

}

2.针对这种方式我们可以得到资源的名称，但是使用servletContext不可以得到资源文件的名称

# 五、java文件读取web资源文件，src其实就是我们的classes文件夹，输入流中的资源文件直接就是这个文件夹

**public** **class** java {

**public** **void** update() **throws** IOException{

InputStream isInputStream = **this**.getClass().getClassLoader().getResourceAsStream

("/db/properities/db.properities");

//读取 properities 文件专门有它的方法

Properties properties = **new** Properties();

properties.load(isInputStream);

System.*out*.println((properties.getProperty("url")).toString());

}

}

## 1.用这个类就会使用类加载器

**public** **class** java {

**static**{

**try** { InputStream isInputStream = java.**class**.getClassLoader().getResourceAsStream("db/properities/db.properties"); //读取 properities 文件专门有它的方法

Properties properties = **new** Properties();

properties.load(isInputStream);

System.*out*.println((properties.getProperty("url")).toString());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

## 2.然后我们使用其中的类就可以完成了

**public** **class** java {

**static** Properties *properties* = **new** Properties();

**static**{

**try** {

InputStream isInputStream = java.**class**.getClassLoader().getResourceAsStream("db/properities/db.properties");

//读取 properities 文件专门有它的方法

*properties*.load(isInputStream);

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**public** **void** update (){

System.*out*.println((*properties*.getProperty("url")).toString());

}

}

## 3.需要注意的是，类装载器，只会装载一次，所以我们如果调用了上面的方法之后就会调用这个类中的一次如果我们改变了这个资源文件之后，再次使用不会重新加载我们改变后的资源文件，而是之前的，所以我么就会需要使用下面的得到路径的方式来进行解决了

**public** **void** update () **throws** IOException{

String pathString = **this**.getClass().getClassLoader()

.getResource("db/properities/db.properties").getPath();

FileInputStream inputStream = **new** FileInputStream(pathString) ;

*properties*.load(inputStream);

}