## CMD学习笔记

### 查看端口占用情况

查看端口的占用情况：Cmd 🡪netstat –ano

查看某个端口（例如：8080）被谁占用 netstat -aon|findstr "8080"

根据PID查看该PID（例如：2720）是哪个程式名称：tasklist|findstr "2720"

## Java Web 学习笔记

### tomcat 解压后的目录：

bin:可执行文件（startup.dat , shutdown.bat）

conf:配置文件（server.xml）

lib:tomcat依赖的jar包

log：放日志文件（记录出错等信息）

temp：放临时文件

webapps：可执行的项目（将我们开发的项目，放入该目录）

work：存放由jsp翻译成的java，以及编译成class的文件，（jsp—）java—》class）

### 配置tomcat

1. 配置jdk（必须配置JAVA\_HOME）JAVA\_HOME 、classpath、path
2. 配置catalina\_home,配置tomcat的根目录

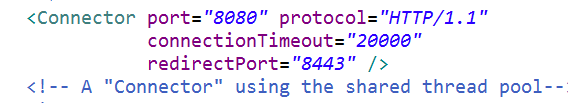
双击bin/startup.bat启动tomcat，

常见错误：可能与其他服务的端口号冲突，

修改tomcat的默认端口号

1. 修改端口号方式:

Server.xml 中的connector



### 访问tomcat

<http://localhost:8080>

常见状态码：

200：一切正常

404：资源不存在

403：权限不足（如果访问a目录，但是a目录设置为不可见）

300/301：代表页面重定向（跳转）

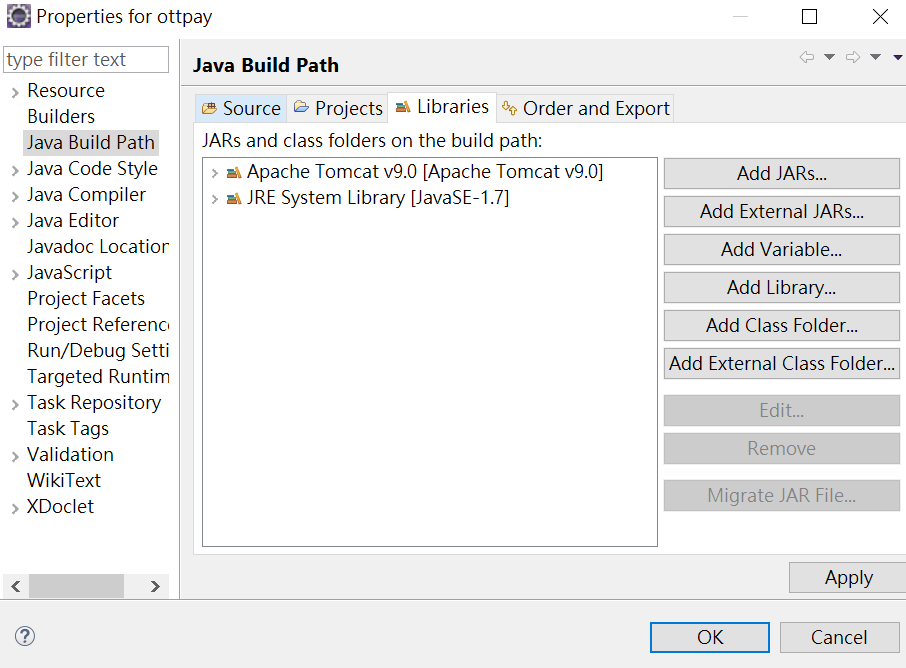
500：服务器内部错误（代码有误）

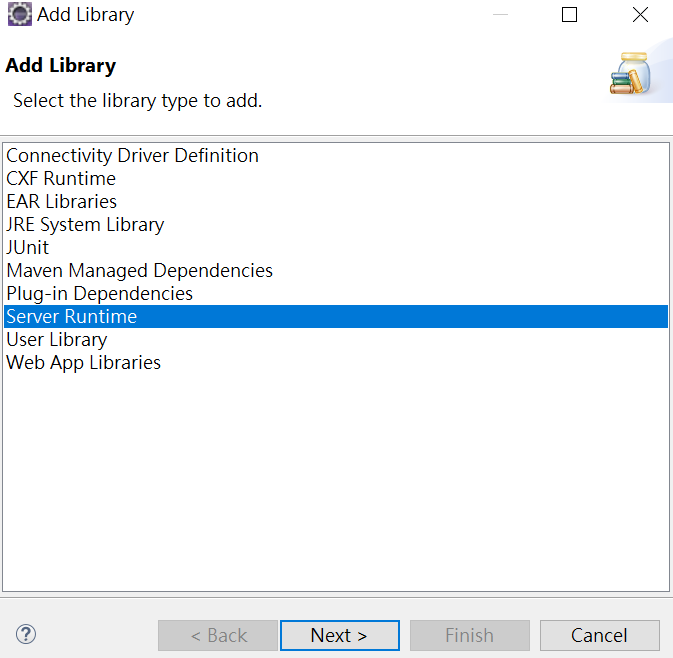
### 将tomcat server加入build path

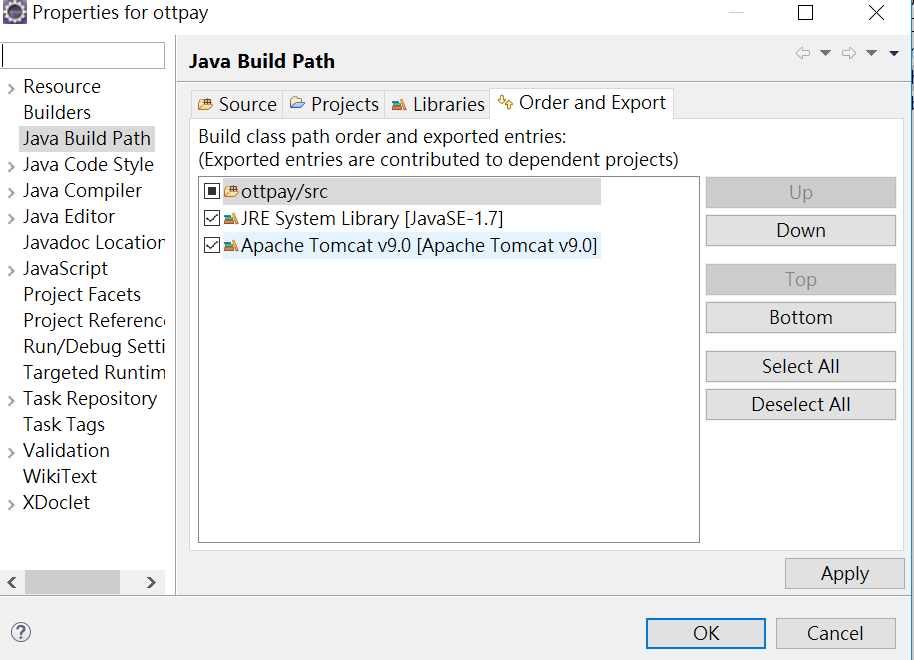
项目右键🡪build path🡪configure build path 🡪libraries🡪add library -🡪

Server Runtime 🡪 Apache tomcat 🡪 finish 🡪

Order and Export🡪勾选Apache Tomcat🡪 apply







### 开发自己的项目

jspProject

在web.xml中设定默认首页：

<welcome-file-list>

<welcome-file>index.html</welcome-file>

<welcome-file>index.htm</welcome-file>

<welcome-file>index.jsp</welcome-file>

</welcome-file-list>

### 虚拟路径

方式一：

将web项目配置在webapps以外的目录，

在tomcat/conf/server.xml中找Host标签，配置Context标签，

docBase：实际路径

path：虚拟路径（绝对路径、相对路径（相对于webapps））

然后重启tomcat

<Host name=*"localhost"* appBase=*"webapps"*

unpackWARs=*"true"* autoDeploy=*"true"*>

<!-- Access log processes all example.

Documentation at: /docs/config/valve.html

Note: The pattern used is equivalent to using pattern="common" -->

<Valve className=*"org.apache.catalina.valves.AccessLogValve"* directory=*"logs"*

prefix=*"localhost\_access\_log"* suffix=*".txt"*

pattern=*"%h %l %u %t &quot;%r&quot; %s %b"* />

<Context docBase=*"E:\Dev\Java\iportal-robot\WebRoot"* path=*"/robot"*></Context>

</Host>

方式二：

在tomcat\conf\Catalina\localhost

中新建 “项目名.xml”,并新增一行

<Context docBase=*"E:\Dev\Java\iportal-robot\WebRoot"* path=*"/robot"*></Context>

### 虚拟路径（配置网站域名）

目标：配置一个网站的域名

第1步.修改Server.xml中的Engine标签

<Engine name="*Catalina*" defaultHost="www.test.com">

<Host name=*"* *www.test.com"* appBase=*"webapps"*

unpackWARs=*"true"* autoDeploy=*"true"*>

<Context docBase=*"E:\Dev\Java\iportal-robot\WebRoot"* path=*"/robot"*></Context>

</ Host >

第2步.修改本机Host文件：

C:\Windows\System32\drivers\etc\Host

增加 127.0.0.1 www.test.com

流程：

www.test.com-->host找映射关系--->server.xml找Engine的defaultHost-->通过“/”映射到“*E:\Dev\Java\iportal-robot\WebRoot*”

### 使用Eclipse开发Web项目

#### 7.1. **配置tomcat运行时环境**

1. 将tomcat/lib中的servlet-api.jar加入项目的构建路径
2. 右键项目—>Build Path🡪Add Library 🡪 Server Runtime

#### 7.2. 部署tomcat

在servers面板上，新建一个tomcat实例，再在该实例中部署项目（右键-add）之后运行

#### 7.3. 统一字符集编

a.编码分类：

设置jsp文件的编码（jsp文件中的pageEncoding的属性）：jsp—>Java

设置浏览器读取jsp文件的编码（jsp文件中的content属性）

一般将上述设置成一致的编码，推荐使用utf-8

文本编码：

1. 将整个eclipse中的文件 统一设置（推荐）
2. 设置某一个项目
3. 设置单独的文件

#### 7.4.Jsp的页面元素：

a.脚本Scriptlet

1）<% 定义局部变量、Java语句%>

2）<%! 全局变量、定义方法%>

3）<%=输出表达式%>

一般而言，修改web.xml、配置文件、java，需要重启tomcat服务，

但是如果修改Jsp、html、css、js，不需要重启

. 指令

Page指令

<%@ page …. %>

Page: 指定的属性：

Language：jsp页面使用的脚本语言（java）

Import：导入类

pageEncoding：jsp文件自身的编码 jsp—java

contentType:浏览器解析jsp的编码

<%@ page language = “java” contentType=”text/html; charset = utf-8” pageEncoding=”UTF-8” import = “java.util.Date”%

1. 注释

Html 注释<!-- //-->

Java注释 // /\*\*/

Jsp注释：<%-- --%>

### 在eclipse中启动tomcat,可是单独打开浏览器不能访问

问题：

tomcat在eclipse里面能正常启动，而在浏览器中访问http://localhost:8080/不能访问，且报404错误。同时其他项目页面也不能访问。

关闭eclipse里面的tomcat，在tomcat安装目录下双击startup.bat手动启动tomcat服务器。访问htt://localhost:8080/能正常访问tomcat管理页面。

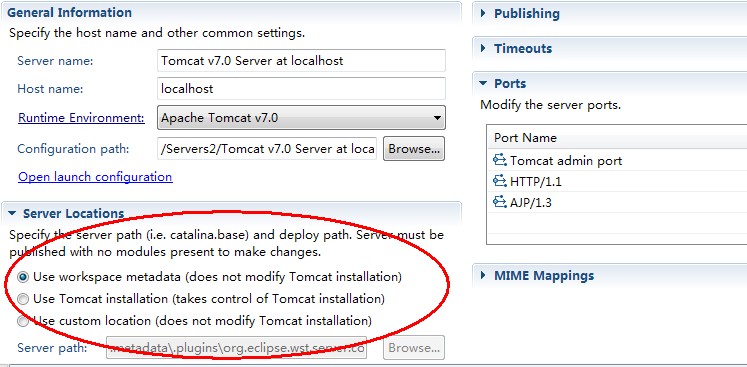
原因：

eclipse将tomcat的项目发布目录（tomcat 目录中的webapp）重定向了,所以你会发现在tomcat安装目录下的webapp目录里面找不到你的项目文件。

解决办法：

重新配置下tomcat服务器：

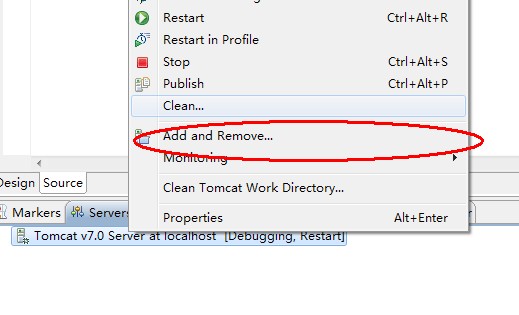
在eclipse中的server页面，双击tomcat服务，会看到如图所示的配置页面：



可以看到红圈中选择的是 Use workspace metadata(does not modify Tomcat installion)

如果该tomcat中部署了项目的话，这红圈中的选项会灰掉不能修改，要修改必须得先把tomcat中的部署的服务都移除。

如图：



通过右键单击tomcat服务器选择 Add and Remove，在弹出的对话框中移除已部署的项目。移除完确定后，将看到上面的选项面板部分可编辑了。

选择Use tomcat installation(Task control of Tomcat installation) 即选择tomcat的安装目录来作为项目的发布目录。

然后,下来四行,看到"Deploy Path"了没?它后面的值默认是"wtpwebapps",把它改成"webapps",也就是 tomcat

中发布项目所在的文件夹名字。

修改后关掉该页面，保存配置。这样就将项目部署到了tomcat安装目录下的webapp

重启tomcat服务器，访问http://localhost:8080则能正常访问了，自己部署的项目也能正常访问了。

### jsp九大内置对象（自带的，不需要new也能使用的对象）

out: 输出对象，想客户端输出内容

pageContext

request： 请求对象，存储“客户端向服务端发送的请求”。

Request对象的常见方法：

String getParameter(String Name): 根据请求的字段名key，返回字段值value

String[] getParameter(String Name): 根据请求的字段名key，返回字段值value（checkbox）

Void setCharacterEncoding(“编码格式utf-8”): 设置请求编码（tomcat7以前默认iso-8859-1,tomcat8以后默认utf-8）

getRequstDispatcher(“b.jsp”).forward(request,response): 请求转发的方式跳转页面A🡪B

getServerContext(): 获取当前项目的ServletContext对象

乱码解决

1. Get方式 ：通过new String(para.getBytes(“iso-8859-1”),”utf-8”);

修改Server.xml,统一get的编码：URIEncoding=”utf-8”

Post方式：通过request.setCharacterEncoding(“utf-8”);

name

response

session

application

config

page

exception

## Spring MVC

**1. Spring MVC中的重要组件**

1.1 DispatcherServlet:前端控制器，接收所有请求（如果配置@WebServlet("/"),则代表不包含jsp）

1.2 HandlerMapping:解析请求格式，判断希望要执行哪个具体的方法

## cc-backed问题解决笔记

1. Access-Control-Allow-Origin 问题

## 

解决方式：(没有解决)

1. 把下面这一段，增加到tomcat server 里面的web.xml中

<filter>

<filter-name>CORS</filter-name>

<filter-class>com.thetransactioncompany.cors.CORSFilter</filter-class>

<init-param>

<param-name>cors.allowOrigin</param-name>

<param-value>\*</param-value>

</init-param>

<init-param>

<param-name>cors.supportedMethods</param-name>

<param-value>GET, POST, HEAD, PUT, DELETE</param-value>

</init-param>

<init-param>

<param-name>cors.supportedHeaders</param-name>

<param-value>Accept, Origin, X-Requested-With, Content-Type, Last-Modified</param-value>

</init-param>

<init-param>

<param-name>cors.exposedHeaders</param-name>

<param-value>Set-Cookie</param-value>

</init-param>

<init-param>

<param-name>cors.supportsCredentials</param-name>

<param-value>true</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>CORS</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

## Angular学习笔记

### Node.js和 npm 包管理器

Node地址：<https://nodejs.org/en/>

检查的版本： node –v

npm检查版本：npm –v

Exception

Description Resource Path Location Type

The container 'Maven Dependencies' references non existing library 'E:\eclipse\apache-tomcat-9.0.12\maven\repository\com\caucho\hessian\4.0.37\hessian-4.0.37.jar' icardpay-settle Build path Build Path Problem

Description Resource Path Location Type

The container 'Maven Dependencies' references non existing library 'E:\eclipse\apache-tomcat-9.0.12\maven\repository\com\icardpay\util\CommonUtil\1.1.3\CommonUtil-1.1.3.jar' icardpay-merchant Build path Build Path Problem