* 这个文档是从第八天开始记录的，资料放置在 **资料&工具02** 中。第七天是Java基础加强，暂时先不记录，后面再写。

## day08-Web简介与Tomcat

### 课程回顾

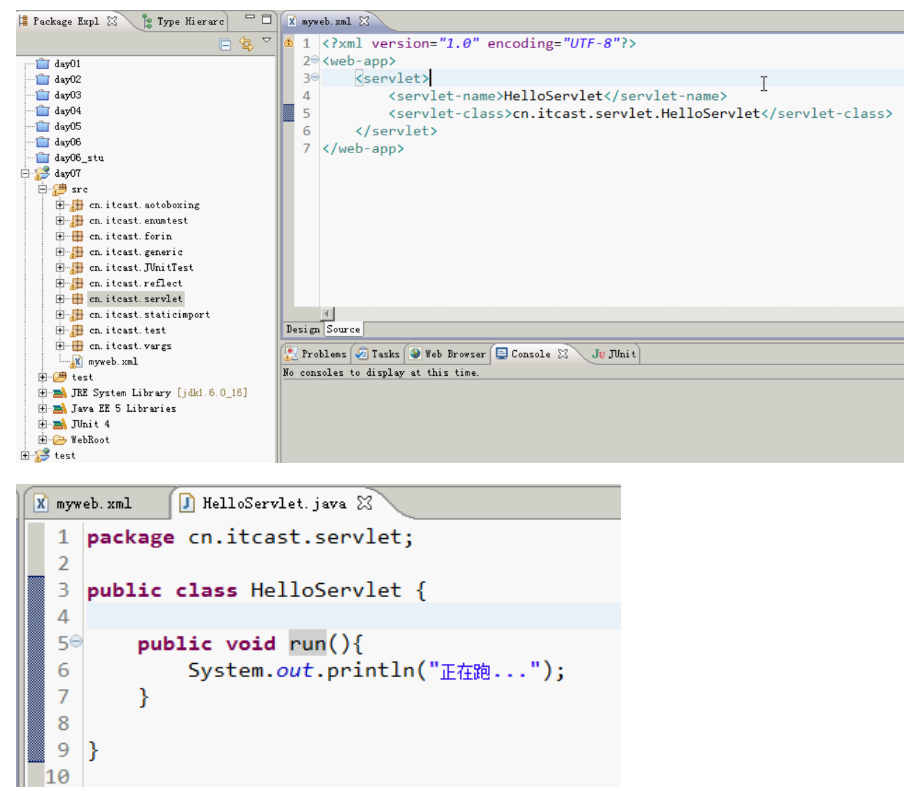
* Java基础加强（注：这一小节内容需要配合day07视频来回顾）
* MyEclipse的简介
  + cn.itcast.xxx创建的包
  + 创建类 User
  + 方法 runDemo()
* DeBug调试（断点调试）
  + 打断点。选择DeBug As
  + Step Over 跳过 F6
  + Resume 执行下一个断点，如果没有断点，执行完成。F8
  + Step Into 跳入 F5
  + Step Return 跳出 F7
  + Drop To Frame 返回到最开始位置
* JUnit 单元测试
  + 测试一个方法。
  + 使用的JUnit4的版本。导入jar包。myeclipse直接引入即可。
  + @Test第一次导包。
  + @Test 测试
* MyEclipse快捷建
* JDK5.0的特性
  + 泛型
    - 应用在集合上。
    - 编写通用性较强的代码。
      * 自定义泛型方法和类 public void run(T t){ T t }
  + 枚举 enum 类名称{ BOSS("红色"),HR; }
  + 静态导入（鸡肋）
  + 增强for循环 for(数据类型 变量 ：要循环的内容){

}

* + 可变参数
    - int...nums
    - 可以nums当成数组。
    - 只能有一个可变参数，可变参数放在末尾。
  + 反射
    - 通过.class文件获取类中的所有的内容。
    - 先获取Class对象
      * 类名 User.class;
      * 实例对象 new User().getClass();
      * Class.forName("类的包名+类名");
    - 构造器、属性、方法。获取到都是对象。

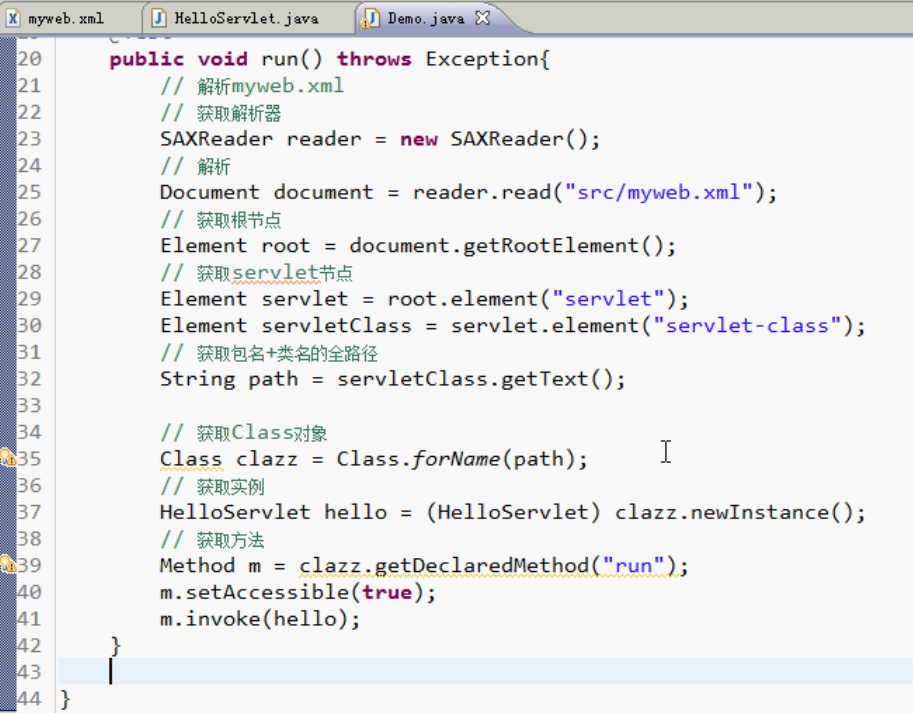
### 作业（XML解析+反射）

自己写一个xml文件，myweb.xml，用到的知识点是xml解析和反射。



步骤：

1. 导包 dom4j-1.6.1.jar，放在webroot目录的web-inf的lib目录下面
2. 写一个单元测试 在方法上面写一个@Test注解，不需要导包
3. 写代码逻辑



### 网络架构

WEB的学习

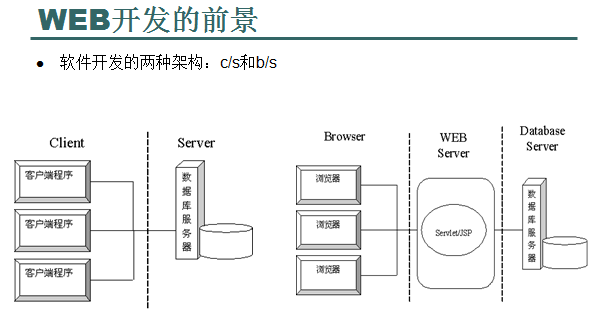
* 服务器
* HTTP协议

**服务器**

* 网络的架构（面试题）
  + C/S client/server 客户端/服务器端 例子：QQ 快播 暴风影音
    - 优点：交互性好，服务器压力小。
    - 缺点：客户端更新了，下载。
  + B/S browser/server 浏览器/服务器端 例子：购物网站 12306
    - 缺点：服务器压力大。
    - 优点：服务器更新就ok。

### WEB的简介

Web开发前景



主要是写 WEB server这一层的东西。

* WEB🡪网页 javaweb 🡪使用java开发网页。
* 静态的web资源
  + HTML、CSS、JAVASCRIPT
* 动态的web资源
  + 数据发生变化。例子：并不是页面动态，它就是动态web
  + Servlet/JSP
  + 了解一下同行的技术
    - ASP .net这是微软开发
    - PHP 开源（开发网站那速度真快），提供模板。
    - RUBY 小日本开发的。

JAVA做什么？后台的管理系统，处理大数据。

### 常见的服务器及tomcat服务器安装等

引入：

比如在自己C盘，新建了一个1.html文件，如何让别人可以访问到呢？

Java基础的时候可以通过socket方式，端口，流的知识，来进行文件的读取。这个时候就引入到服务器概念了，服务器会帮我们做这一系列工作。

服务器的原理就是**网络编程**。

概念：

* 硬件：就是一台主机。
* 软件：安装了服务器的软件（tomcat）。
* 安装软件后，称为WEB服务器。
* 启动服务器，访问资源。
* 访问：http://+ip+端口号 找到主机。如果资源的文件，就可以访问了。
* 访问百度：<http://www.baidu.com/>
* HTTP协议默认端口号是80，可以不写。
* ping [www.baidu.com](http://www.baidu.com/) 得到的是61.135.169.121
  + 完整的浏览器访问地址应该是 http://61.135.168.121:80
* 如果想访问本机的服务器（扩展）
  + [http://localhost:80](http://localhost/)
  + [http://127.0.0.1:80](http://127.0.0.1/)

常见服务器

* WebLogic是 BEA公司开发的（后来被Oracle收购了） 收费的 支持JAVAEE所有的规范（EJB servlet/jsp规范）
  + （JAVA+MySql(Oracle)+WebLogic）都是Oracle公司的了，一条龙。
* WebSphere是 IBM公司开发的 收费的 支持JAVAEE所有的规范（EJB servlet/jsp规范）
  + 银行喜欢用
* Tomcat 是apache（开源的组织，非常的伟大）只支持Servlet/JSP规范。免费的。

学习使用tomcat 6.0就可以了，以后熟练了可以使用新的。

Tomcat默认端口8080，不用修改

Tomcat的安装和注意实现

* （前提条件）安装JDK，配置java的环境变量。
* 安装版（给客户用）
* 解压版：解压当前文件夹，目录不要有中文。
* 启动服务器：tomcat/bin/startup.bat（启动服务器）
* 访问： http://192.168.11.100:8080
* 关闭服务器：1点关闭按钮，2. shutdown.bat

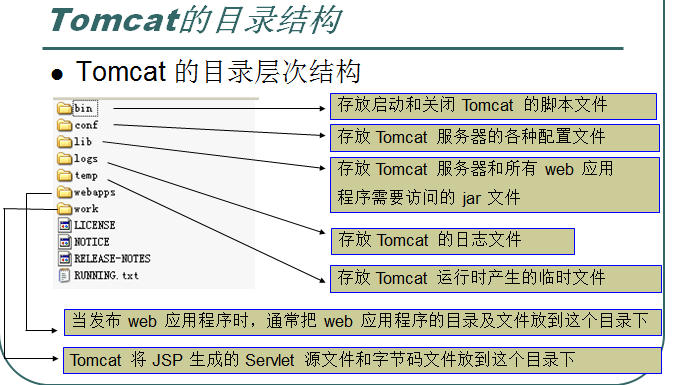
注：自己写的web文件，放到webapps/下面

### tomcat服务器常见的启动问题



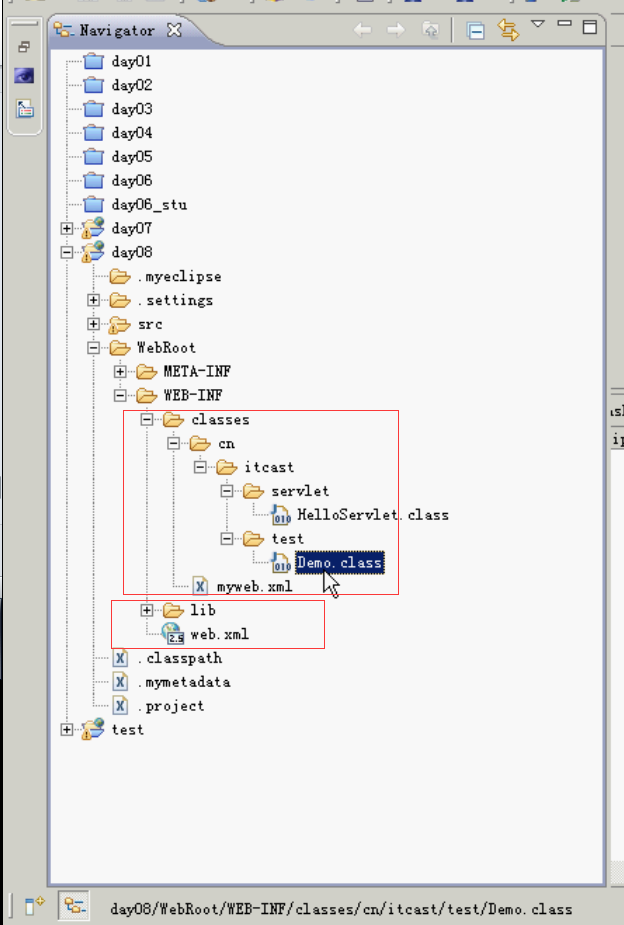
概率非常小，可以参考占用80端口解决的html文件，里面有解决方案。

### tomcat的包结构和动态的web资源目录结构



Tomcat目录结构

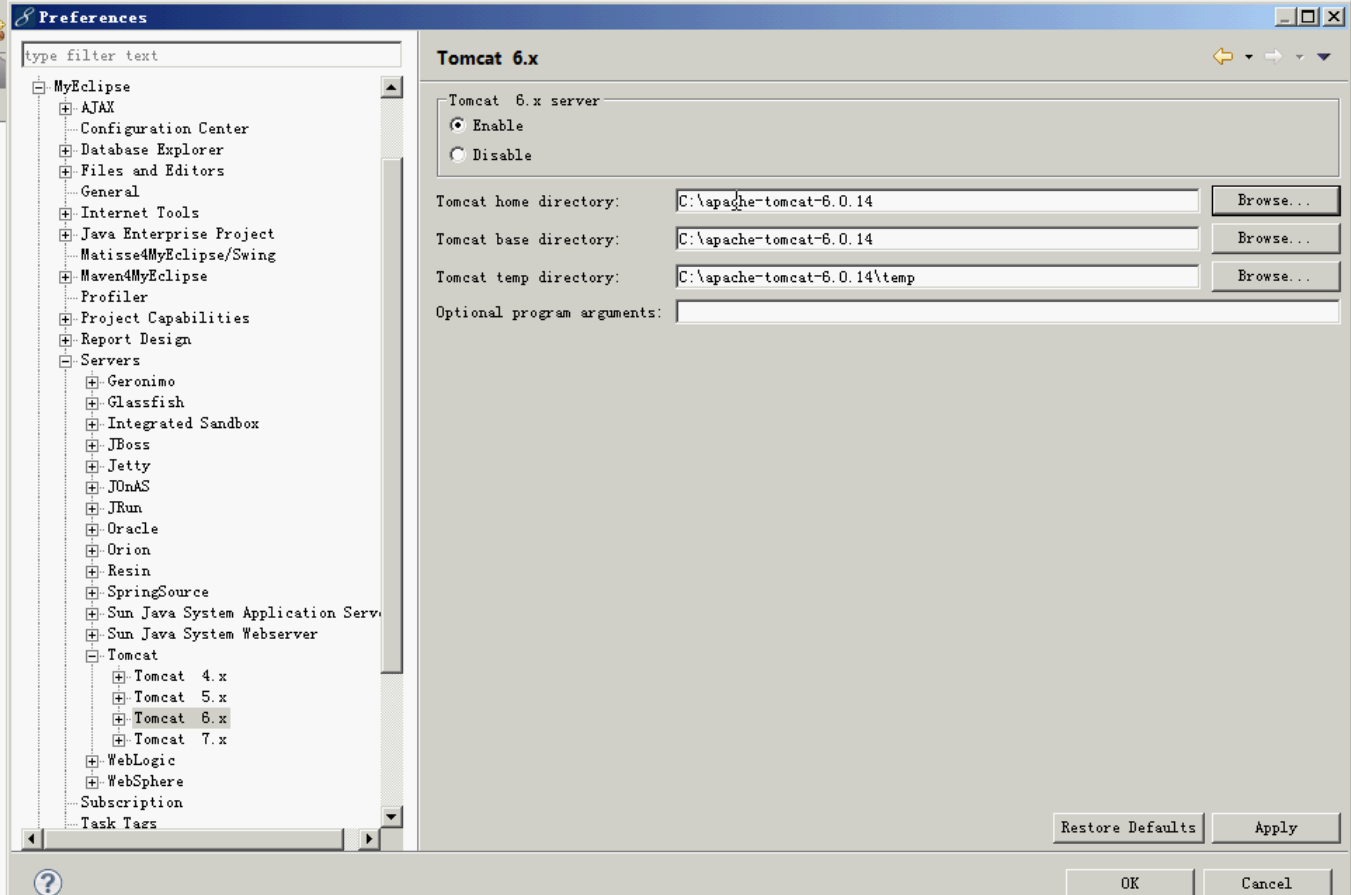
* bin（\*\*） 启动项，关闭项。
* conf（\*\*\*\*\*） 里面Tomcat配置文件
* lib（\*\*） 服务器运行使用的jar包
* logs（\*\*\*） 日志文件，运行时产生的日志。
* temp（\*\*） 运行时临时文件
* webapps（\*\*\*\*\*） web applications（web的应用们）
* work（\*\*\*\*\*） JSP翻译成Servlet程序



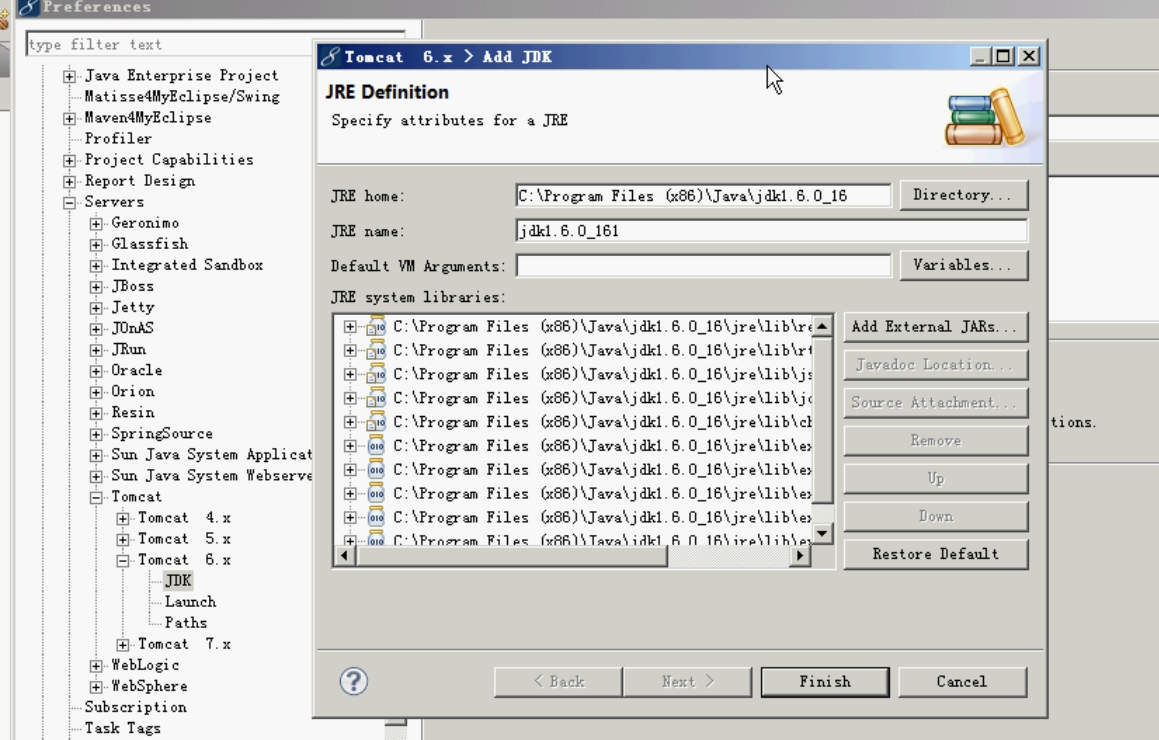
### Tomcat与MyEclipse集成和部署项目

Tomcat和MyEclipse的集成

* window -- 选项 -- MyEclipse -- servers -- 选择tomcat -- 选择Enable -- 选择tomcat根目录
* 配置JDK -- 选择JDK。
* 启动服务器。



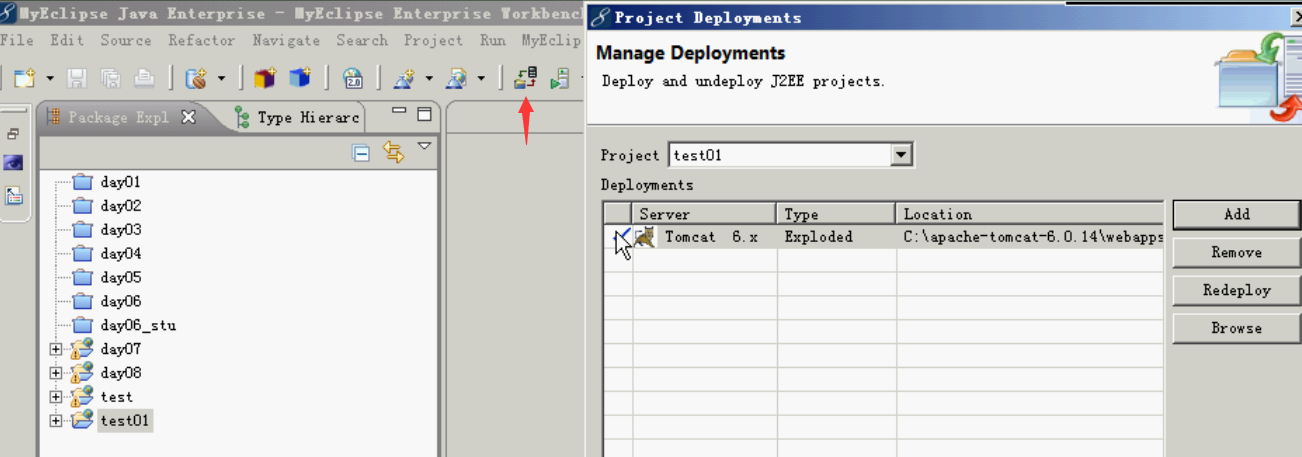
配置JDK：

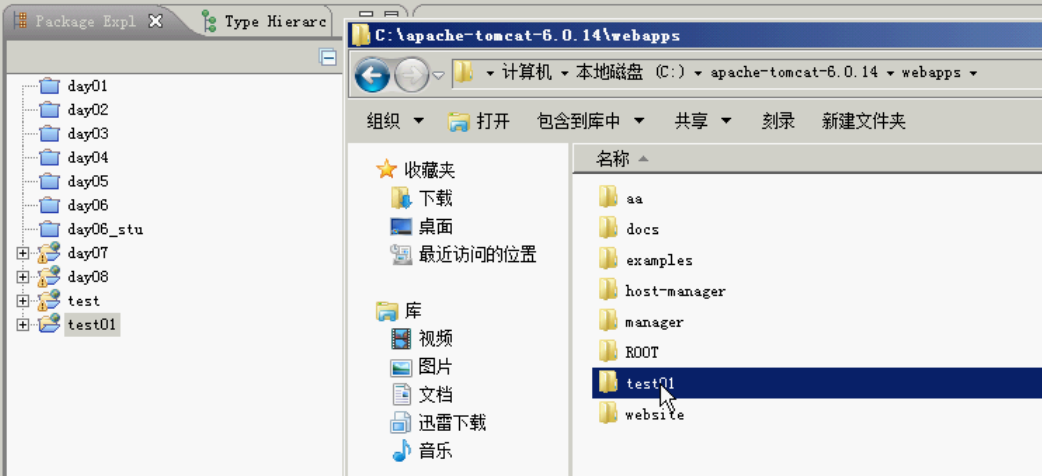


启动：



如何发布？





这个和手动写的一样。

### 虚拟路径问题

虚拟路径（访问路径）

* 默认和项目名称是相同（不要去修改）。
* WebRoot变成了虚拟路径（访问的路径）
* ## 以下是下午内容

### 虚拟路径修改和tomcat管理员的配置

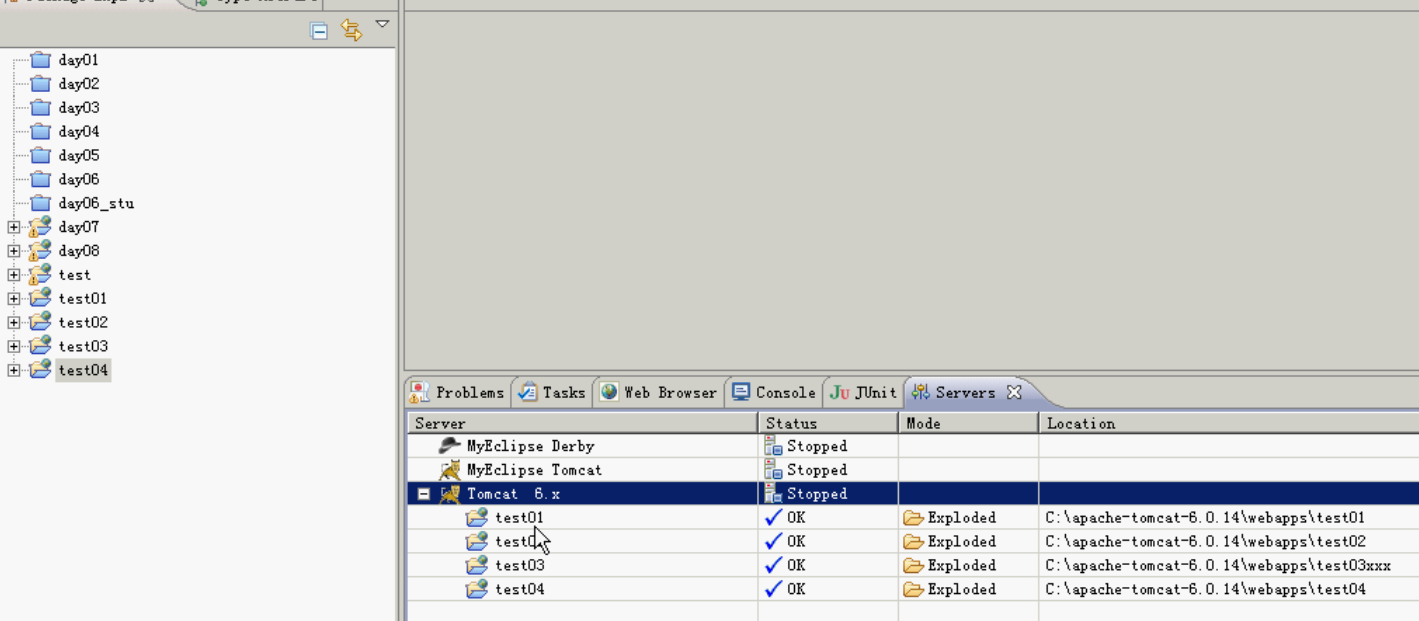
Web项目直接复制项目的时候，需要修改虚拟路径。

项目上右键 -- 选择属性 -- MyEclipse -- web -- 修改虚拟路径—发布项目。



扩展一个东西：

window -- show view – servers



Tomcat管理员的配置（了解）

* 在tomcat/conf/tomcat-user.xml

<role rolename="manager"/>

<user username="admin" password="admin" roles="manager"/>

修改完配置文件之后，需要重启tomcat。

### 部署项目的三种方式

1）项目复制到webapps目录下

2）通过配置虚拟路径的方式

A. 直接修改配置文件

B. 自己编写一个配置文件（格式）（推荐使用）

* 直接修改配置文件
* 写到tomcat/conf/server.xml
* 找到标签，配置到Host标签的中间
* 目的：通过配置，配置访问路径，准确找到c:\bb的文件
  + <Context docBase="文件夹的真实目录" path="虚拟路径（访问路径）" ><Context>
  + <Context docBase="C:\bb" path="/itcast" ></Context>
  + 访问：<http://localhost:80/itcast/bb.html>
* 自己编写一个配置文件（格式）（推荐使用）

\* 自定义xxx.xml结尾文件，在$CATALINA\_HOME/conf/[enginename]/[hostname]/ directory.目录下。

\* 把xxx当成虚拟（访问）路径。

\* 在xml的文件中编写。

\* 在哪个目录下：

\* $CATALINA\_HOME/conf/[enginename]/[hostname]/ directory.

\* 如果找引擎的名称和主机的名称，在server.xml中找。

\* tomcat/conf/Catalina/localhost/ccc.xml

\* ccc.xml的文件编写什么内容？

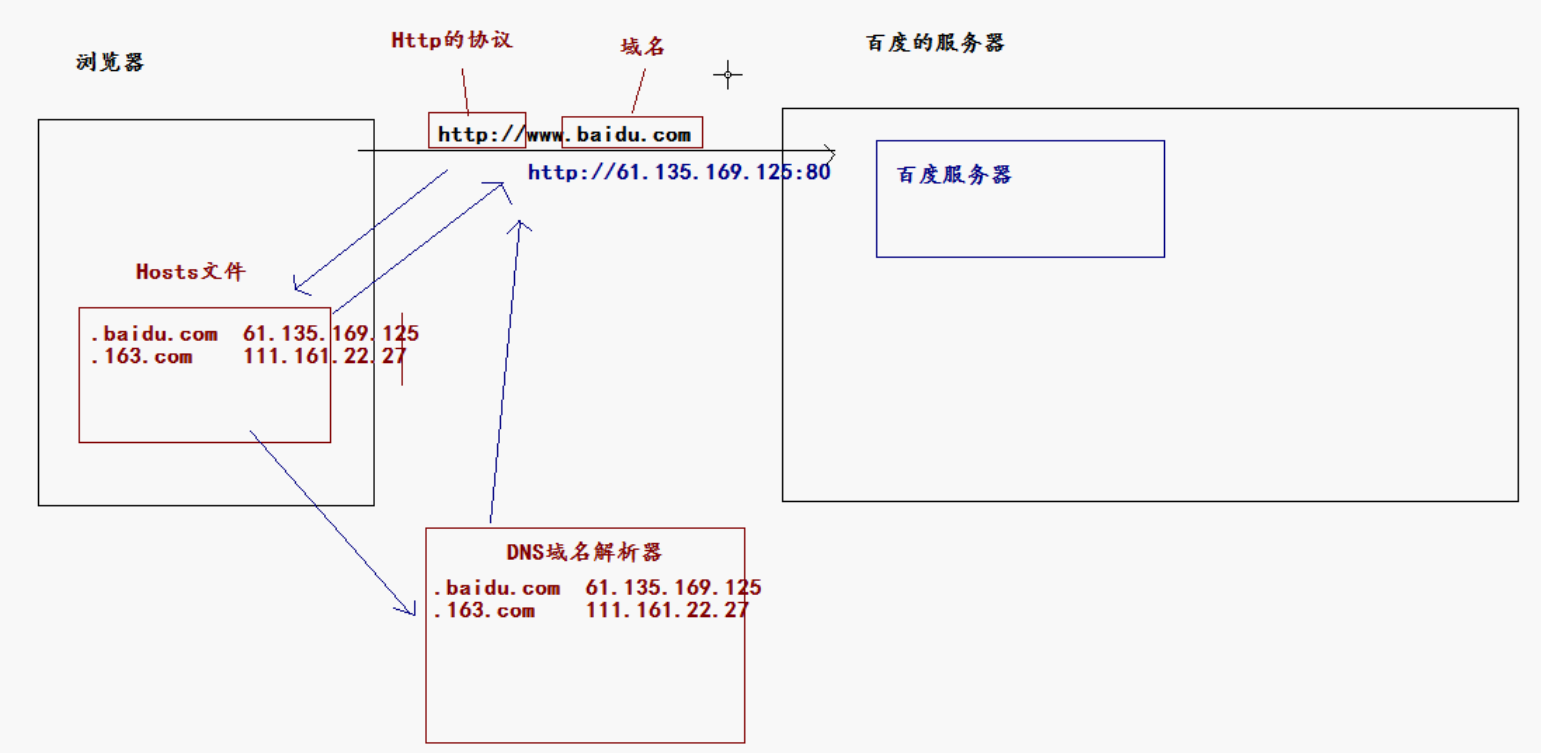
\* <Context docBase="C:\cc"></Context>

\* 访问：<http://localhost:80/ccc/abc.xml>

操作步骤：

1.在tomcat/conf/ 目录下面，创建一个Catalina文件夹，在这个文件夹下面创建一个localhost文件夹，然后创建ccc.xml文件

### web通信原理



### HTTP协议的简介

配置虚拟主机（了解），先没讲。

HTTP的协议

* 啥是HTTP的协议：超文本传输协议。
* 基于TCP链接的传输协议。
* HTTP协议默认是80
* 基于请求和响应的模式。（先有请求，后有响应）
  + www.baidu.com （访问百度的服务器）
  + 百度的首页返回给我
* 客户端连上web服务器后，若想获得web服务器中的某个web资源， 需遵守一定的通讯格式，HTTP协议用于定义客户端与web服务器通迅的格式

telnet命令：

使用telnet命令模拟HTTP请求，win7系统需要开启telnet指令，参看 **win7下开启telnet命令.doc**

启动服务器. cmd>telnet localhost 80 回车

telnet命令有一个乱码问题 解决乱码. 解决 ctrl+] 回车

注意：命令行中输入的内容不能写错，一旦写错了，不支持删除。 请求：

GET /aa/1.html HTTP/1.1

host:localhost

输入两次回车

响应：

HTTP/1.1 200 OK

Server: Apache-Coyote/1.1

ETag: W/"21-1416624196782"

Last-Modified: Sat, 22 Nov 2014 02:43:16 GMT+

Content-Type: text/html

Content-Length: 21

Date: Sat, 22 Nov 2014 07:33:53 GMT

<h1>hello java!!</h1>

HTTP协议的版本：

* HTTP/1.0
  + 链接后，只能获取一个web资源。
  + 链接后，发送请求，服务器做出响应，链接立即断开。

GET /aa/1.html HTTP/1.0

host:localhost

* HTTP/1.1（使用）
  + 链接后，可以获取多个web资源。
  + 链接后，发送请求，服务器做出响应，链接不会立即断开。 再次发送请求，直接有一段时间没操作，自动断开。

GET /aa/1.html HTTP/1.1

host:localhost

可以使用telnet 的dos窗口来演示http1.0和http1.1的区别

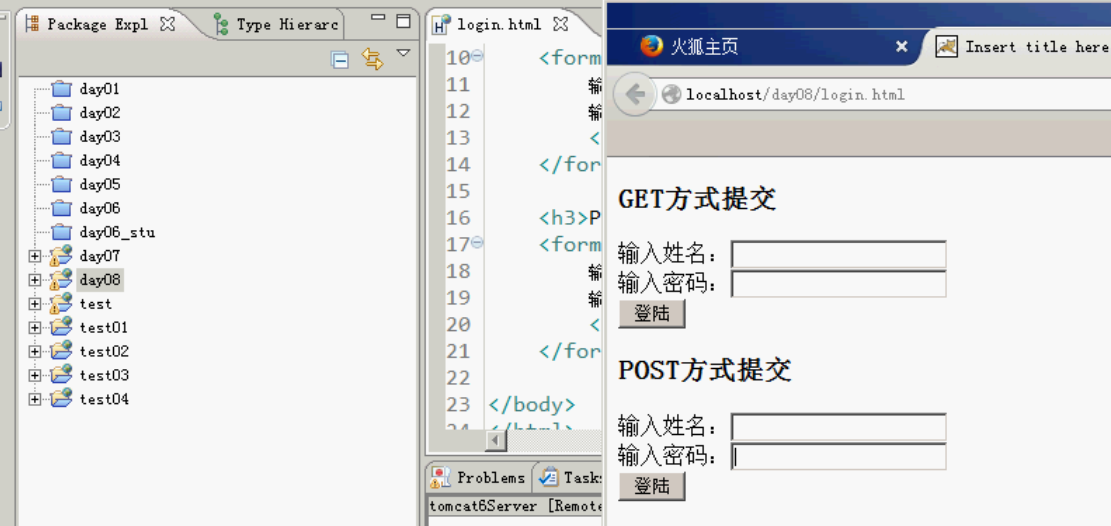
HTTP是hypertext transfer protocol（超文本传输协议）的简写，它是TCP/IP协议的一个应用层协议，用于定义WEB浏览器与WEB服务器之间交换数据的过程。

HTTP协议是学习JavaWEB开发的基石，不深入了解HTTP协议，就不能说掌握了WEB开发，更无法管理和维护一些复杂的WEB站点。

HTTP协议的版本：HTTP/1.0、HTTP/1.1

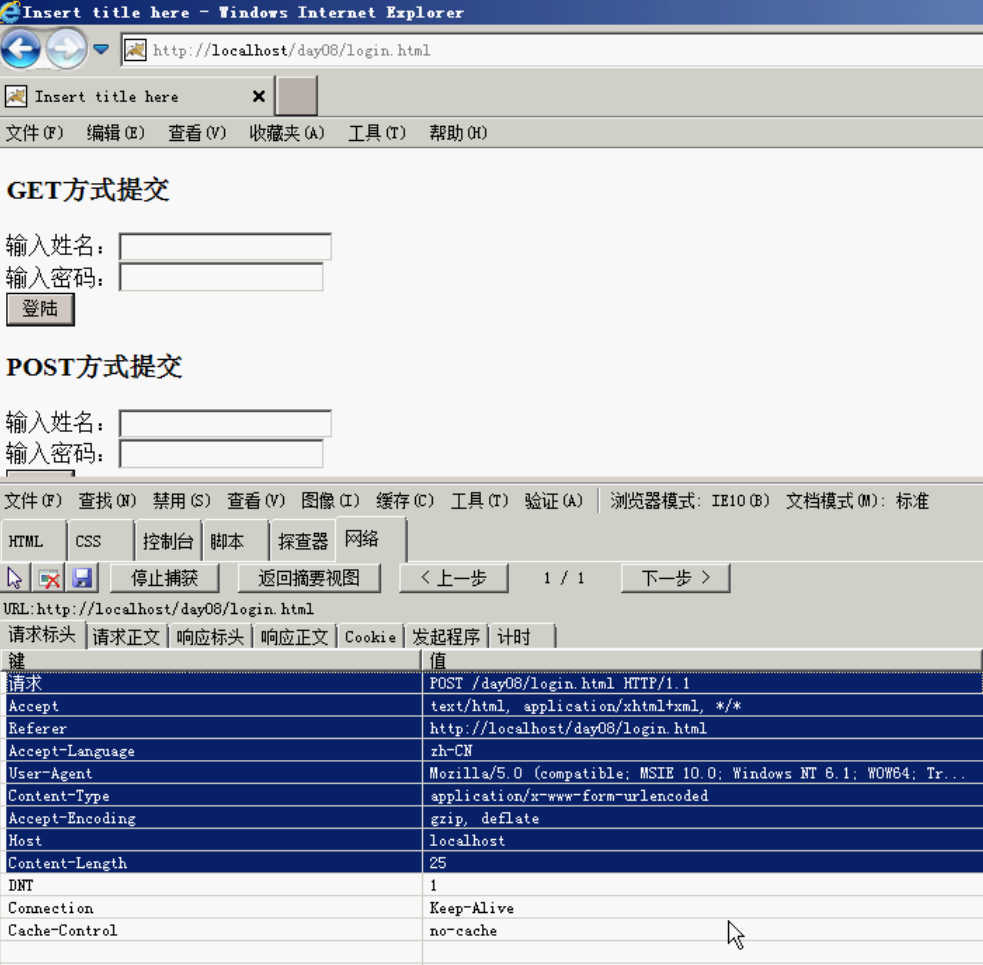
### HTTP协议之请求

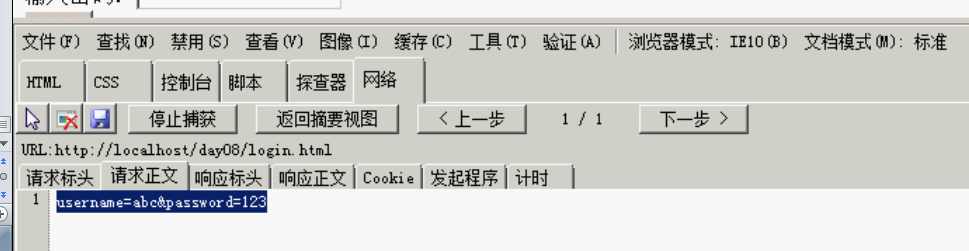
练习HTTP请求，写一个login.html 页面，两个文本框用户名密码登陆，form表单，action不填。



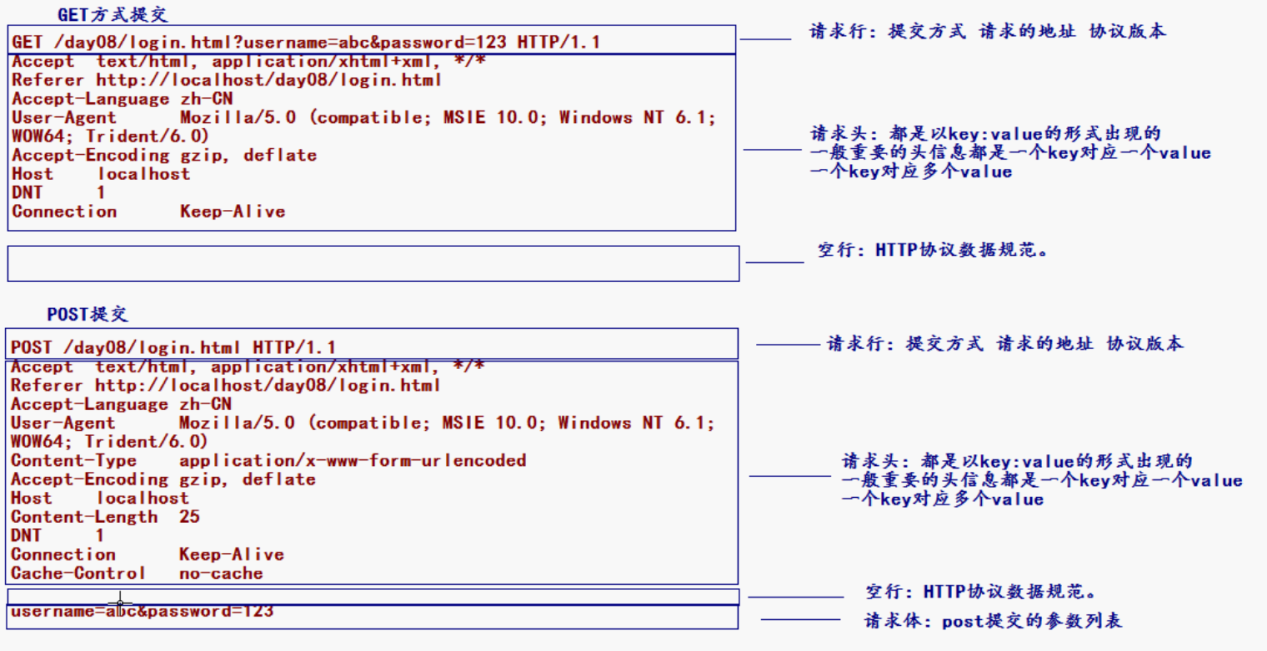


POST请求：





总结：



### 请求行&请求头

**请求行**

请求方式：

POST、GET、HEAD、OPTIONS、DELETE、TRACE、PUT、CONNECT

常用post和get

区别：

get把参数显示在地址栏上，安全级别低，不支持大数据。

post把参数封装请求体中，安全级别高，支持大数据。

请求地址：

请求资源

协议版本：

HTTP/1.1

**请求头**

Accept: text/html,image/\*

Accept-Charset: ISO-8859-1

Accept-Encoding: gzip

Accept-Language:zh-cn

Host: www.itcast.com:80

If-Modified-Since: Tue, 11 Jul 2000 18:23:51 GMT

Referer: http://www.itcast.com/index.jsp

User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0)

Connection: close/Keep-Alive

Date: Tue, 11 Jul 2000 18:23:51 GMT

重要的头：

If-Modified-Since 必须和响应头信息一起来完成控制本地的缓存。

Referer 当前的网页的来源。（防止盗链）

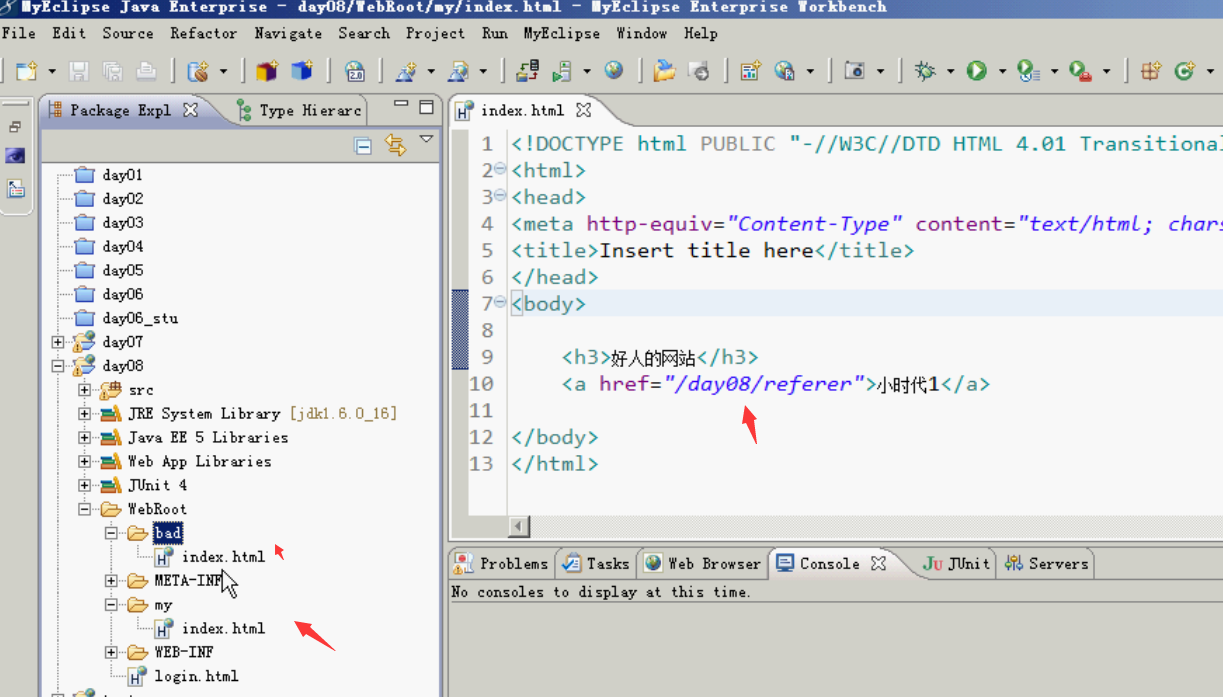
User-Agent 判断浏览器的版本（文件下载的时候）

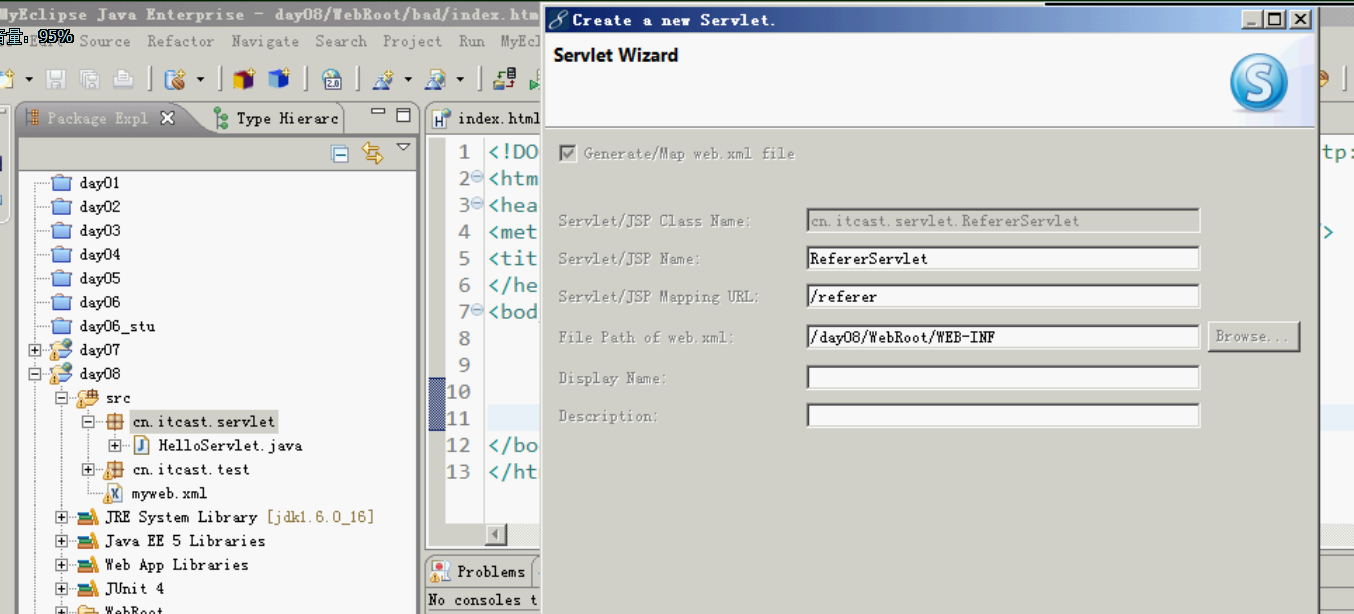
**空行**

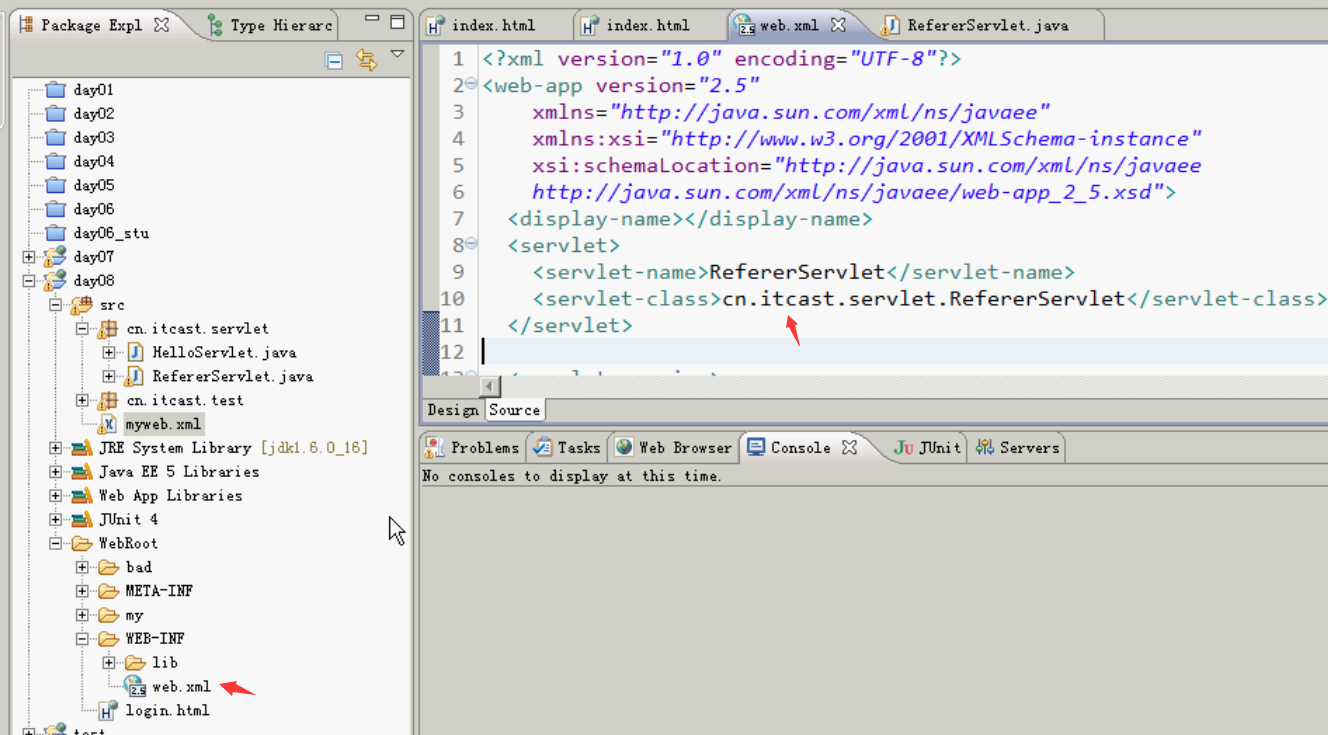
**请求体**

封装post参数列表。

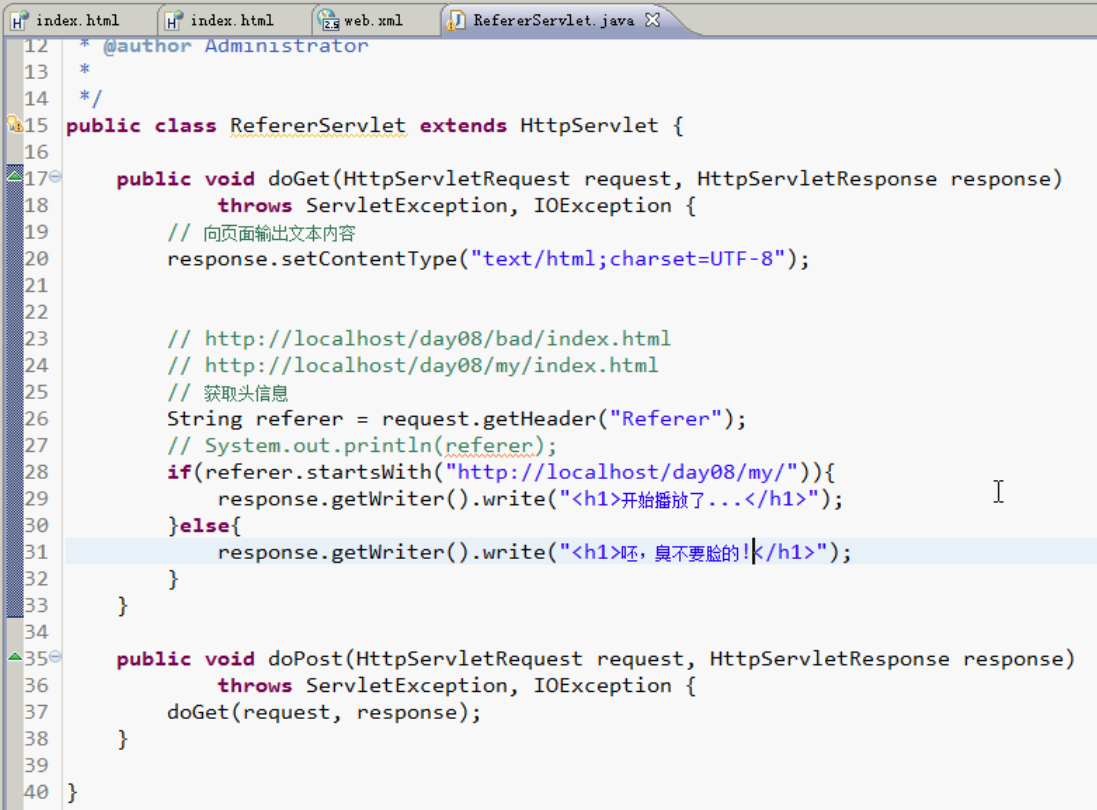
防盗链案例：





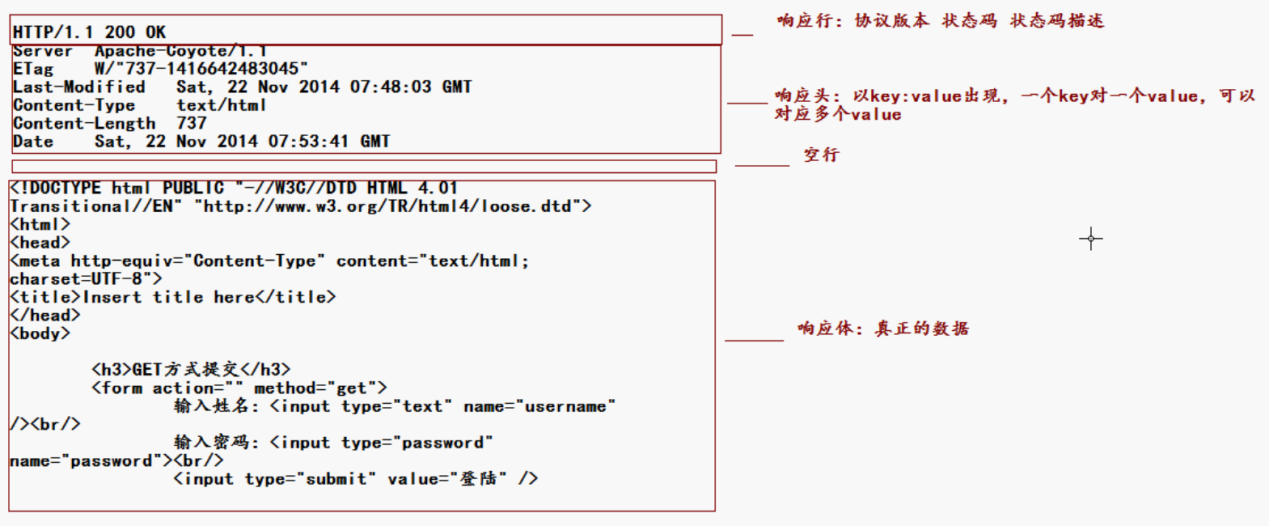


通过反射来调用，因为java文件会生成二进制class文件之后放在WebRoot的目录下面。

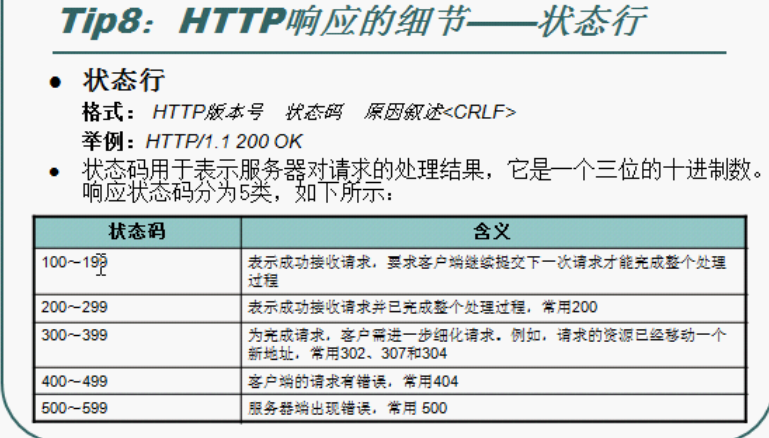


User-Agent用在下载文件，中文文件名，不同的浏览器编码不同，要先判断浏览器类型。

### 响应内容



状态码：



200 ：请求成功处理，一切OK

302 ：请求重定向

304 ：服务器端资源没有改动，通知客户端查找本地缓存

404 ：客户端访问资源不存在

500 ：服务器内部出错

转发和重定向的区别？

转发：找班长借钱，**他自己找**副班长借钱，

重定向：（找班长借钱，发送一次请求，回了我没钱，**返回状态码**302，同时也**返回副班长地址**，再去找富班长借钱，又发送了一次）

响应头：

Location: http://www.it315.org/index.jsp

Server:apache tomcat

Content-Encoding: gzip

Content-Length: 80

Content-Language: zh-cn

Content-Type: text/html; charset=GB2312

Last-Modified: Tue, 11 Jul 2000 18:23:51 GMT

Refresh: 1;url=http://www.it315.org

Content-Disposition: attachment; filename=aaa.zip

Expires: -1

Cache-Control: no-cache

Pragma: no-cache

Connection: close/Keep-Alive

Date: Tue, 11 Jul 2000 18:23:51 GMT

重要的头

* Location 和302一起完成重定向。
* Last-Modified 和请求头If-Modified-Since一起控制缓存。和状态码304
* Refresh 完成页面的定时跳转
* Content-Disposition 设置文件是以附件打开

\* 禁用缓存（网银系统）

Expires: -1

Cache-Control: no-cache

Pragma: no-cache

**Day 08 End。**

## day09-Servlet与ServletContext

### Servlet的快速入门

要写一个动态的web资源

什么是Servlet？

不能看javase文档了，要看javaee文档了。Java\_ee\_api

步骤是，继承Servlet，然后重写里面的5个方法。

Servlet是一个小的java程序，运行在服务器中，接收和响应从客户端（浏览器）发送过来的请求。

**快速入门**

编写一个类，实现Servlet接口，重写5个方法。

简单：编写一个类，继承GenericServlet类，重写一个方法service(ServletRequest req,ServletResponse res)

配置Servlet信息（这个必须会）

回顾动态web资源的目录结构

Tomcat/webapps/

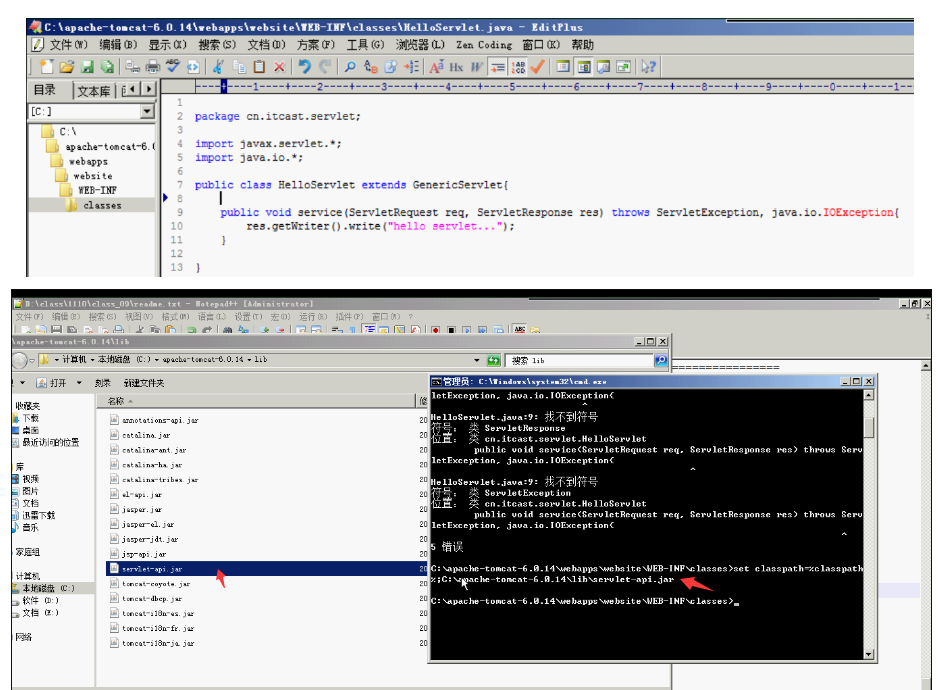
--WEB-INF

--classes文件夹

-- 放置编译后的class二进制文件

--lib文件夹

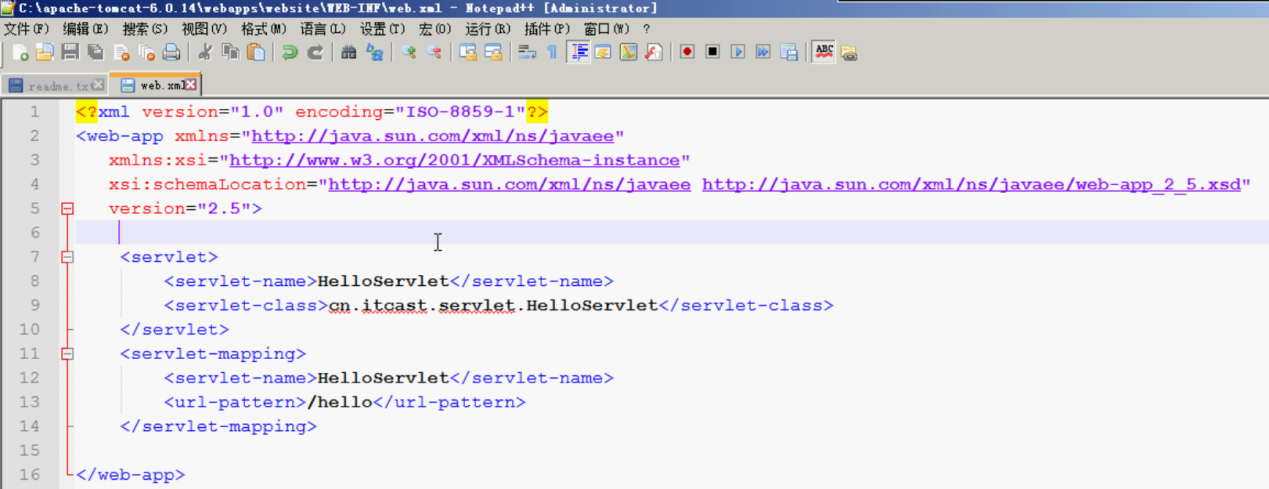
--web.xml 文件

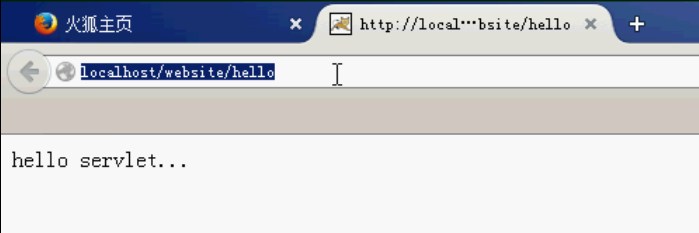


编译注意问题：

* javac -d . HelloServlet.java 带包编译
* 编译时报错信息：HelloServlet.java:4: 原因是软件包 javax.servlet 不存在
* 解决办法：set classpath=%classpath%;servlet-api.jar 设置临时的环境变量，只对当前的窗口有效。

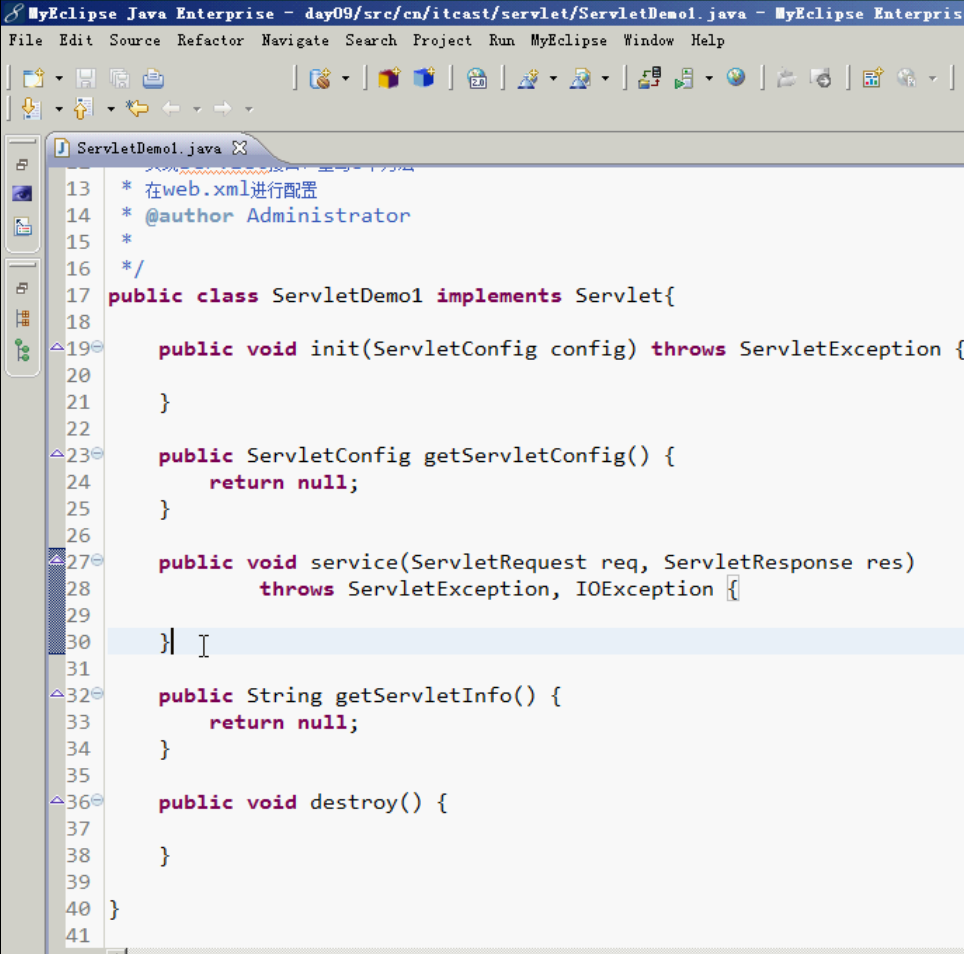
最后一步是配置Servlet信息。配置到WEB-INF/web.xml 这个文件里面





### 使用MyEclipse开发Servlet和执行过程

1.编写Java文件，写一个类继承Servlet接口，实现其中5个方法，然后在service方法，对请求做出返回数据。



2.web.xml文件的配置



1. 发布，启动服务器

### Servlet的生命周期

Servlet的生命周期（面试题）

* 生命周期：实例被创建，对外提供服务，销毁。
  + Servlet被创建后，然后调用init方法进行初始化，第一次访问的时候被调用，只会被调用一次 void init(ServletConfig config)
  + 从客户端发送所有的请求是service方法进行处理的。可以调用多次 void service(ServletRequest req, ServletResponse res)
  + 从服务器中移除服务，调用destroy方法。手动停止服务的时候调用 void destroy()
* Servlet的生命周期：第一次请求的时候，Servlet实例被创建，立即调用init方法进行初始化。实例通过service方法提供服务。服务器关闭或者移除服务时，调用destroy方法进行销毁。

代码测试：



配置web.xml（以后就可以不用手动配置了，使用模板）

几点注意：

1.启动tomcat的时候不会调用servlet的创建方法，而是第一次访问的时候，才会创建这个实例。（注，也可以通过配置文件实现，服务器一启动就会调用实例化方法）

2．Init方法只会在被调用的时候触发，而且只会调用一次。之后的请求会调用service方法

3，destory方法是在服务器关闭或者移除的时候调用。

可以结合ppt来巩固。

### 开发细节



Servlet的关系：

Servlet接口

|

GenericServlet（重写5个方法）

|

HttpServlet（继承GenericServlet实现了Servlet接口）

以后编写新的Servlet类的时候就可以直接继承HttpServlet了。

为什么会有GenericServlet？为什么有HttpServlet？

因为HttpServlet处理的和Http请求相关的，所以有相关的请求。

关联源码可以参看**servlet\_src.zip**和**apache\_tomcat\_6.0.36-src**文件夹。

注意：如果想重写init方法，重写无参数的init方法。

**重点**：

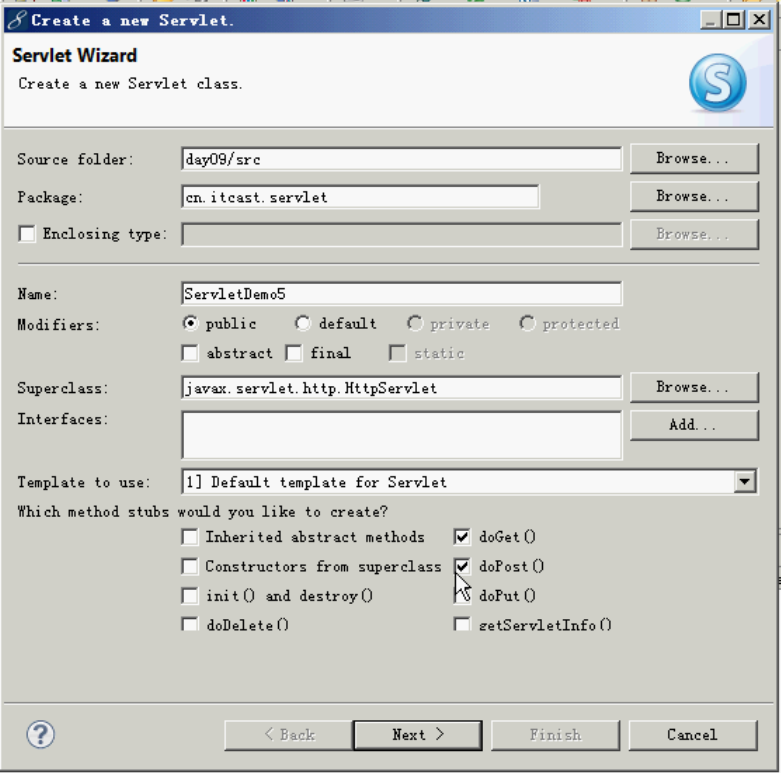
1.继承HttpServlet

2.重写方法，继承HttpServlet的时候，只需要重写doGet和doPost方法，因为service方法会按照父类的方法调用执行。

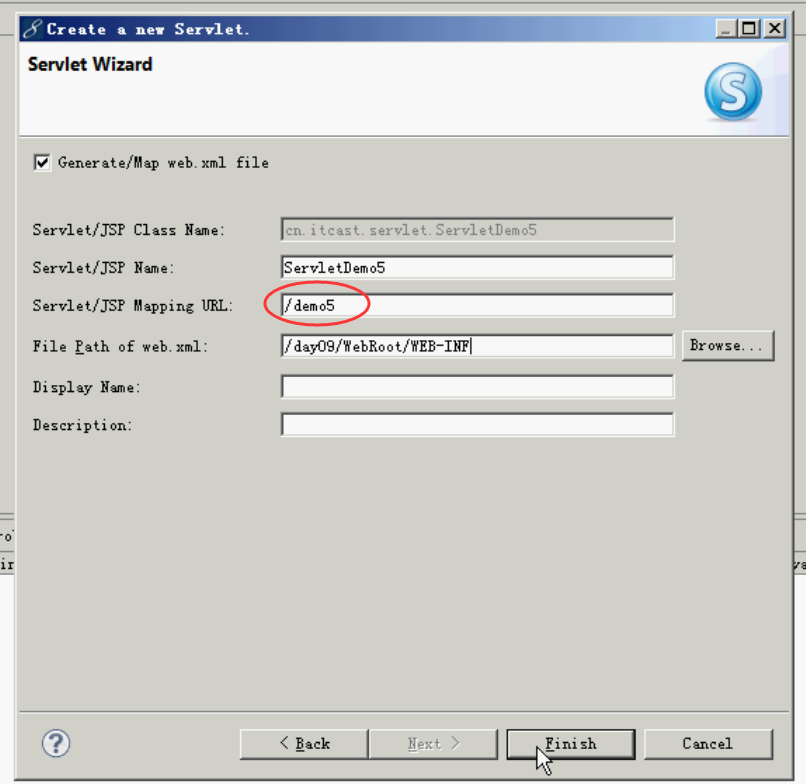
3.在web.xml配置

### Servlet模板的修改

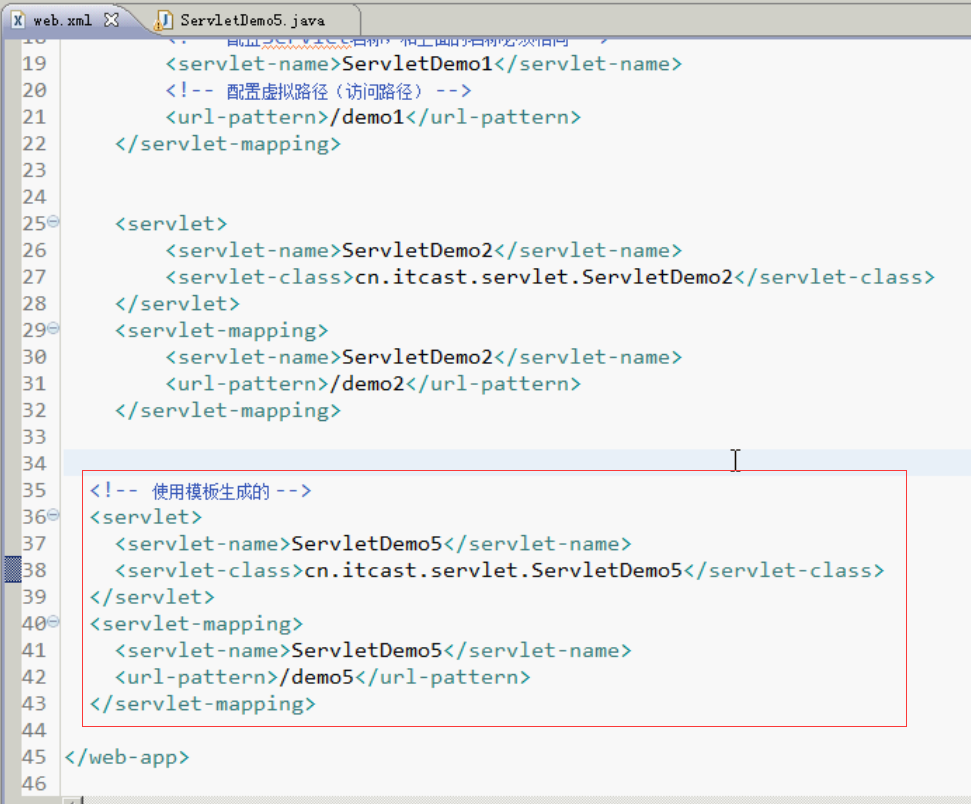
右键创建Servlet：



配置web.xml

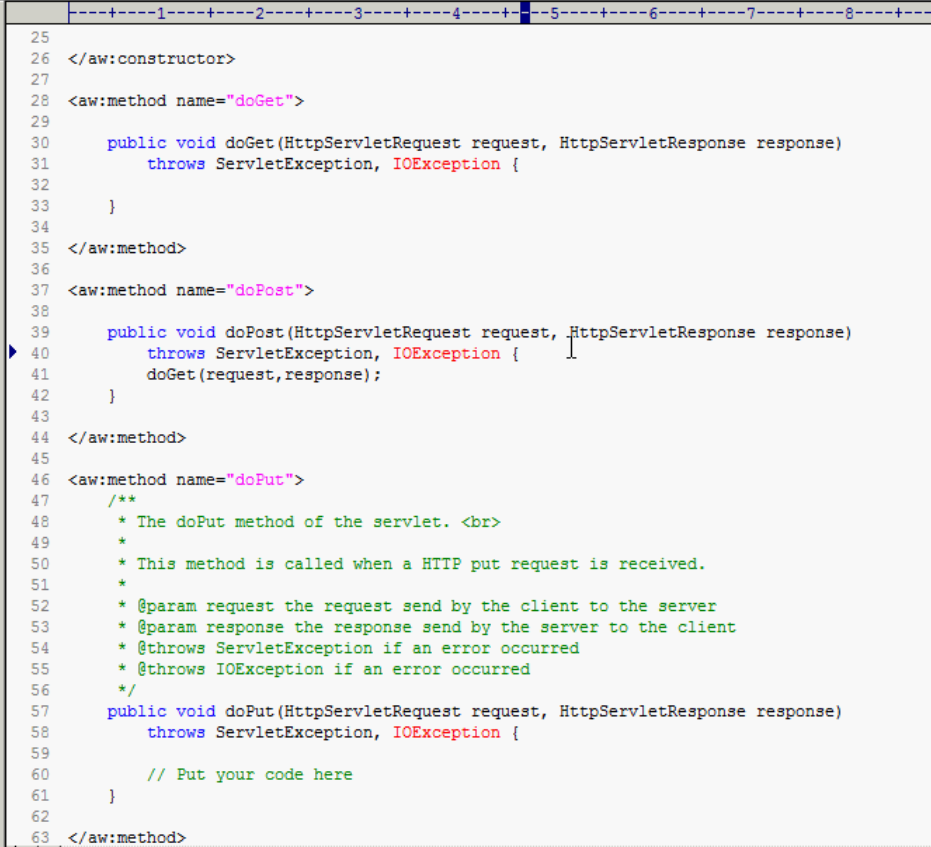


效果：



推荐修改Servlet模板：

* 先找到MyEclipse的安装路径。
* \myeclipse10.7\Common\plugins\com.genuitec.eclipse.wizards.xxxx.jar
* 千万别解压，右键--选择压缩工具打开--templates--Servlet.java，修改Servlet模板
* 拖回去之前，先MyEclipse关闭。



### Servlet自动加载

* Servlet默认是第一次访问时候创建实例。通过配置，服务器启动，创建实例。
* 为什么要自动加载？因为是init做初始化的操作，如果是非常消耗时间的操作就需要自动加载了。如初始化数据库链接
* 怎么配置？ 在标签<servlet>下 加入<load-on-startup>3</load-on-startup>
* 值是正整数
  + 值越小，优先级越高。
* ## 以下是下午内容

### 配置servlet的映射和优先级

配置虚拟路径（访问路径）

是在web.xml文件配置，如下：

<servlet-mapping>

<url-pattern>/demo5</url-pattern>

</servlet-mapping>

可以有3种配置方式：

1. 完全路径匹配

\* 以/开头的 如 /demo5，/servlet/demo5

\* 访问：http://localhost/day09/demo5 ， http://localhost/day09/servlet/demo5

2. 目录匹配

\* 以/开头的 这样的样式 /\*

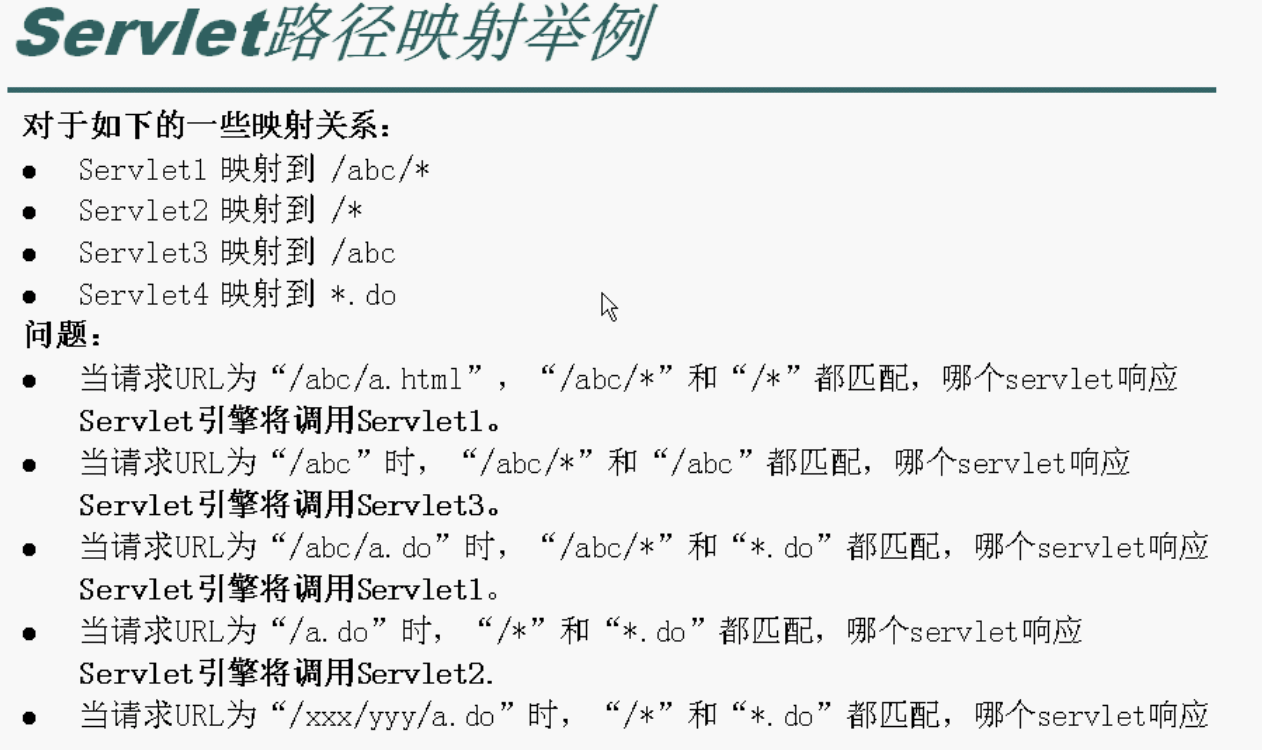
\* 访问：http://localhost/day09/demo5 , http://localhost/day09/xxx

3. 扩展名匹配

\* 不能以/开头的 \*.do \*.action

\* 访问：http://localhost/day09/demo5.do

\* 优先级：完全路径匹配 > 目录匹配 > 扩展名匹配（\*\*\*\*\*）



### WEB开发中路径的问题