# HTTP协议和tomcat服务器

**【学习目标】理解、了解、应用、记忆**

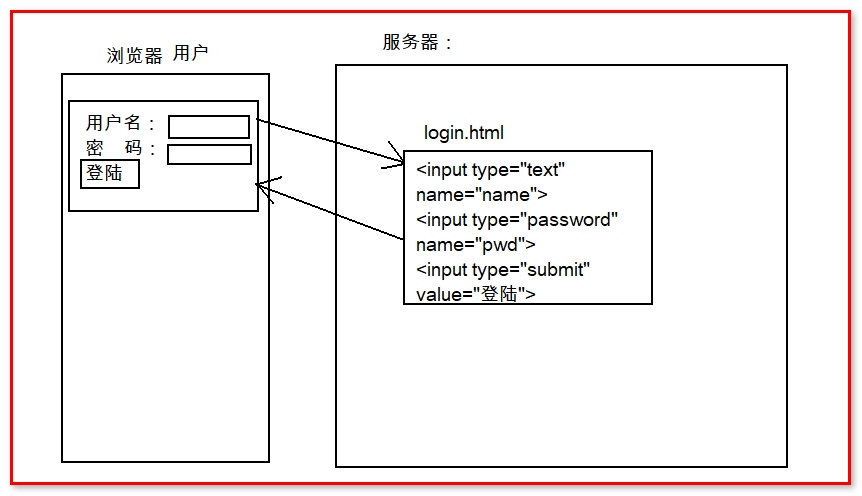
通过今天的学习，参训学员能够：（解释的时候说出二级目标的掌握程度）

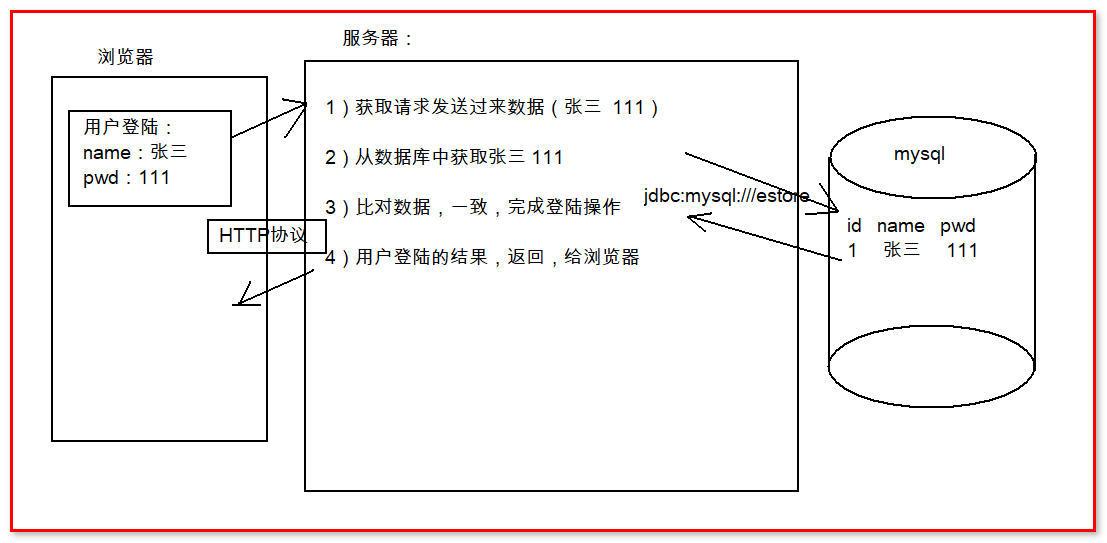
1. **【理解】能够说出WEB服务器和HTTP协议的由来**
2. 能够说出WEB服务器的概念
3. **【应用】能够独立下载安装tomcat**
4. 运用网络下载Tomcat服务器的安装文件
5. 说出Tomcat服务器核心的安装文件夹的作用
6. 运用bin目录下的startup.bat文件启动Tomcat服务器
7. 写出要部署的WEB项目的目录结构
8. 运用Tomcat服务器部署已经编写成功的WEB项目
9. **【应用】能够修改端口和发布web资源（项目）**
10. 修改Tomcat服务器的配置文件
11. 查询启动服务器时出现的异常信息
12. **【应用】能够使用Eclipse集成tomcat和开发web项目**
13. **【理解】能够理解HTTP协议**
14. 说出HTTP协议的概念
15. 能够理解HTTP协议的请求部分
16. 能够理解HTTP协议的响应部分

# WEB服务器和HTTP协议的由来

我们学会制作简单页面，我们学会简单操作数据MySQL，那么如何让在用户这一端的浏览器页面和我们这一端的数据库联系起来？

我们需要一些新事物的加入：





总结：在链接浏览器和数据库的时候需要服务器和http协议作为桥梁

今天我们要学习服务器是Tomcat服务器和http协议

先来看我们的服务器：

Tomcat的介绍：

Tomcat是Apache 软件基金会（Apache Software Foundation）的Jakarta 项目中的一个核心项目，由[Apache](https://baike.baidu.com/item/Apache/6265" \t "_blank)、Sun 和其他一些公司及个人共同开发而成。由于有了Sun 的参与和支持，最新的Servlet 和JSP 规范总是能在Tomcat 中得到体现，Tomcat 5支持最新的Servlet 2.4 和JSP 2.0 规范。因为Tomcat 技术先进、性能稳定，而且免费，因而深受Java 爱好者的喜爱并得到了部分软件开发商的认可，成为目前比较流行的Web 应用服务器。

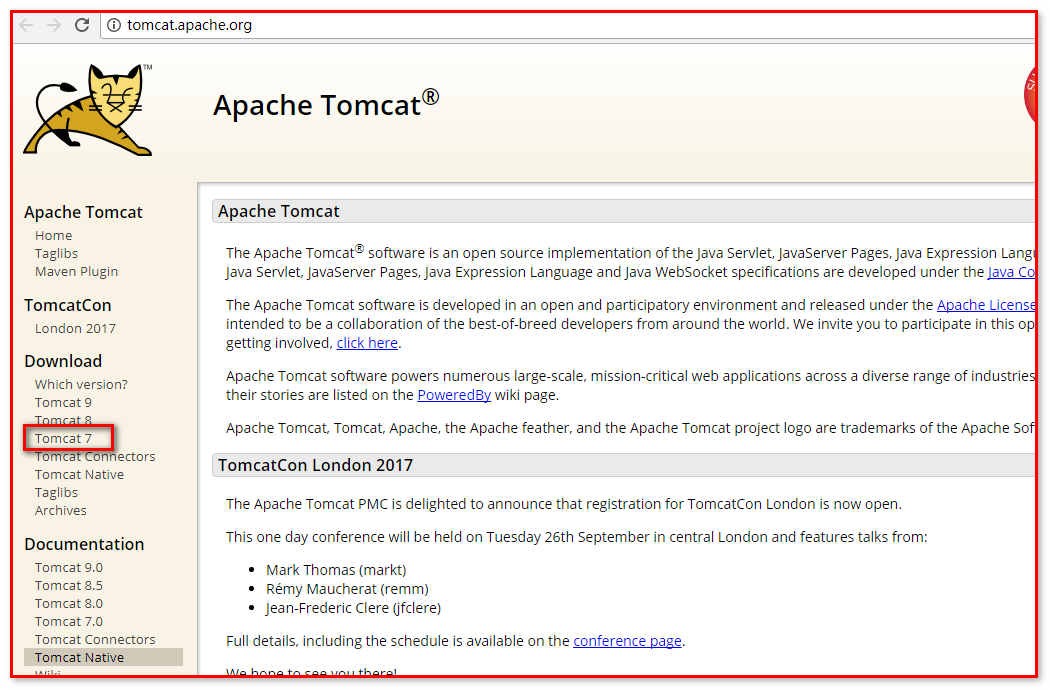
接下来，我们需要使用这个Tomcat服务器，必须要先从它的官网获取它！！！！

# 下载安装tomcat：

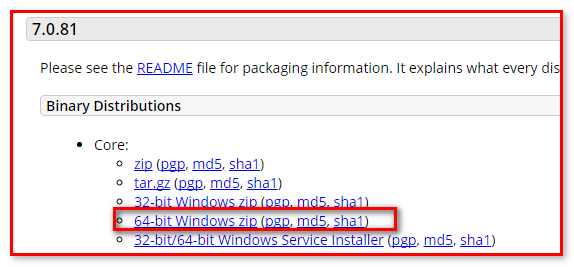
## 如何下载和安装tomcat服务器

### 下载安装

官网：http://tomcat.apache.org/



下载：

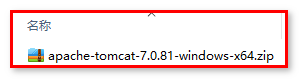


tar.gz 文件 是linux操作系统下的安装版本

exe文件是window操作系统下的安装版本

zip文件是window操作系统下压缩版本

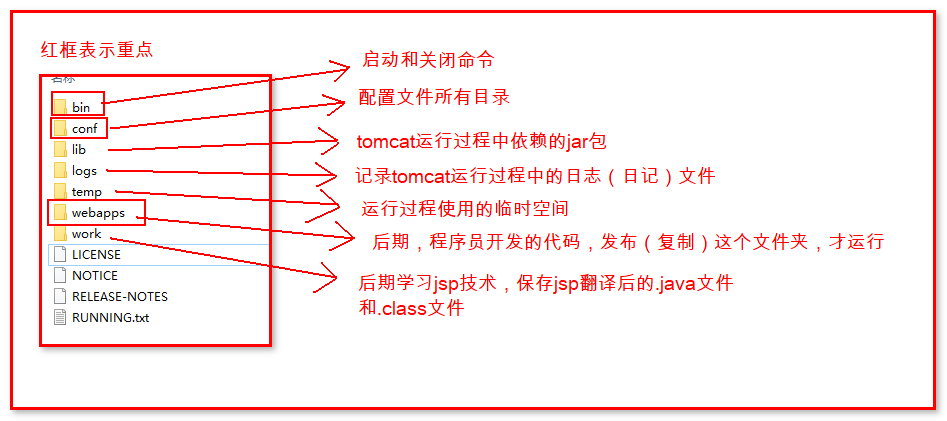
下载后：



### 安装和目录介绍

安装介绍：

直接解压当前这个tomcat压缩包。



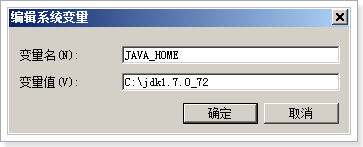
日志：相当于日记，程序记录自己运行过程的文件。

### 配置环境变量

Tomcat软件运行它需要依赖Java的运行环境。

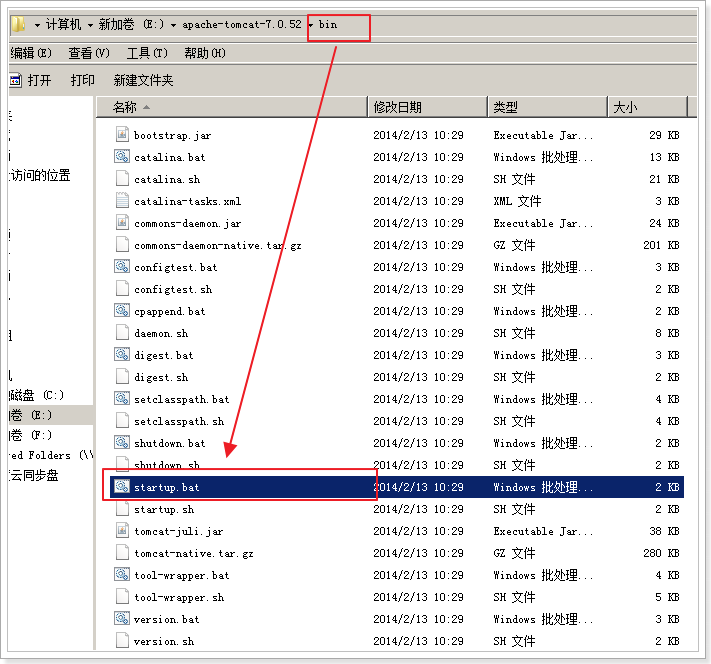
需要在电脑中配置JAVA\_HOME环境变量。

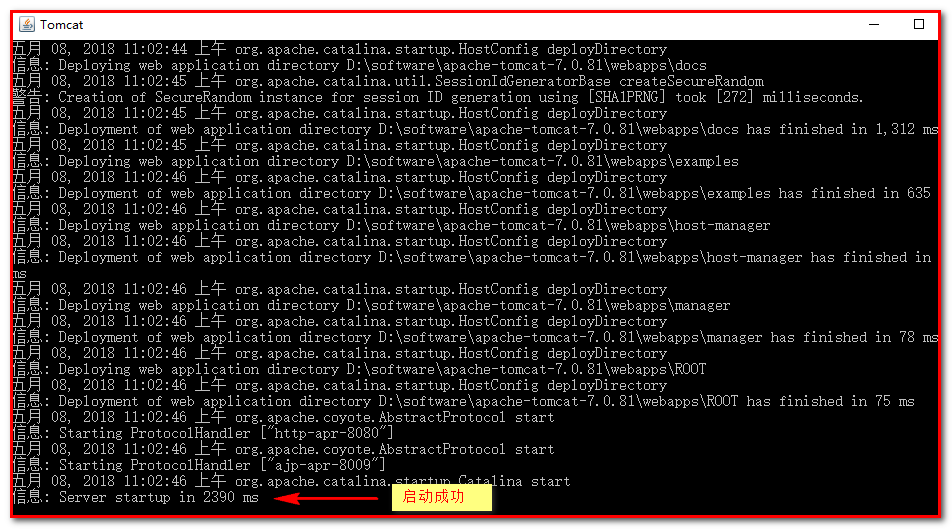
JAVA\_HOME环境变量中配置的JDK的安装目录，不能包含bin目录



### 启动服务器

进入tomcat的安装目录中，在bin目录下有startup.bat文件，双击运行。



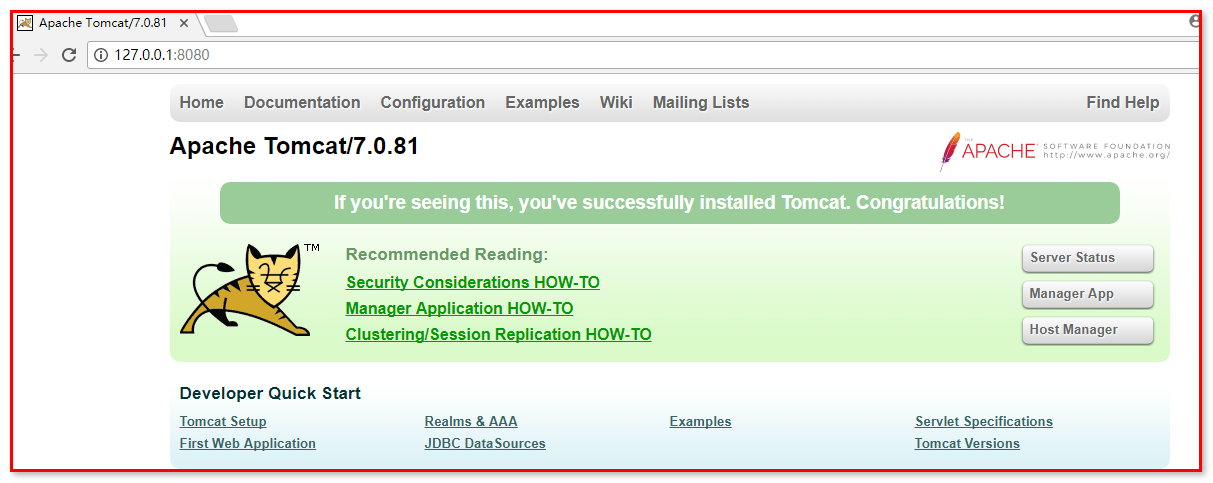


### 测试

打开浏览器在，在浏览器的地址栏中输入：

<http://127.0.0.1:8080>

<http://localhost:8080>



注意： localhost相当于127.0.0.1

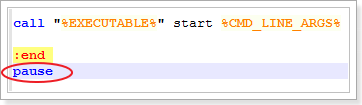
## tomcat服务器使用常见问题

### 无法启动（闪退）

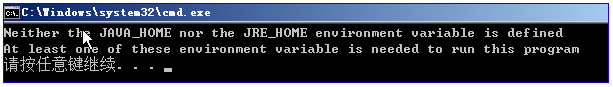
主要原因：没有配置JAVA\_HOME环境变量。

JAVA\_HOME 环境变量 中配置的是JDK的安装目录，不包含bin目录，不是tomcat的安装目录。

闪退的原因查看：可以在startup.bat文件中书写pause命令。让运行的窗口暂停。



无法找到java环境，无法启动tomcat。



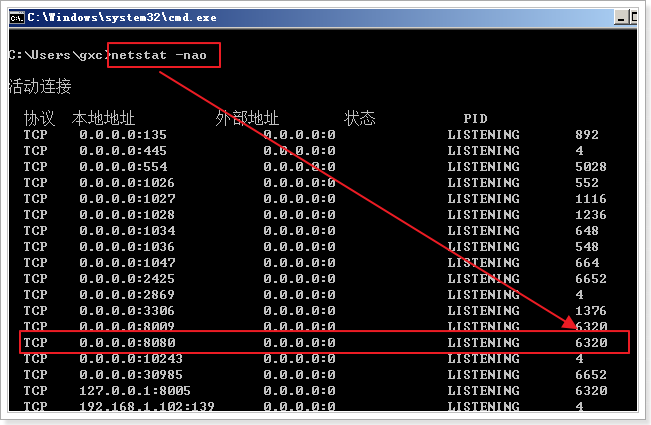
### 端口被占用启动失败的问题

如果启动的时候，发生异常问题，这时有可能是端口被占用。

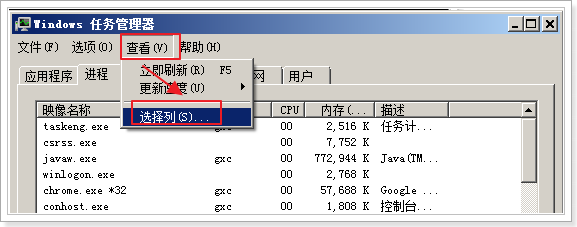
Tomcat服务器在启动的时候默认占用本地的8080端口，如果这个端口被占用，启动的时候就会报错。

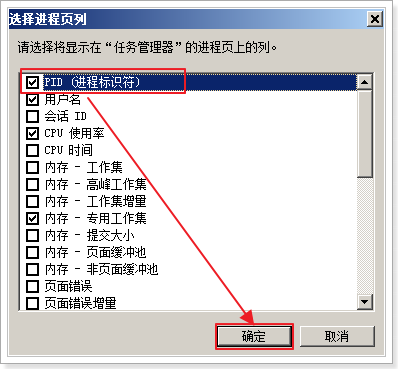
查看本地端口使用情况：

在dos窗口中输入 netstat –nao 就可以查看当前端口的占用情况



查看当前端口占用的pid，然后打开任务管理器，找到进行，在进行查看当前pid对应的进程，把进程干掉。







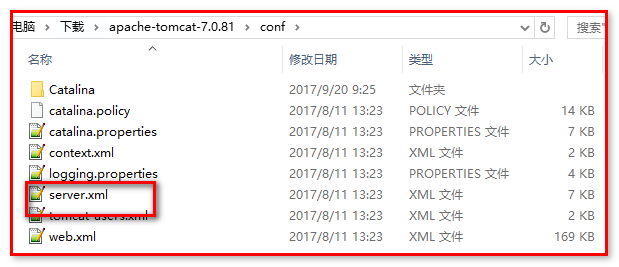
注意：如果这个进程是操作系统的任务进程，这时一般是不能停止这个进程。

如果是系统进程端口被占用了我们只能换一个端口，下面看如何修改端口

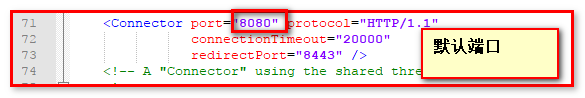
# 修改端口和web资源（项目）发布

## 修改端口

Tomcat服务器的配置，全部都需要在tomcat的安装目录下conf目录下完成：



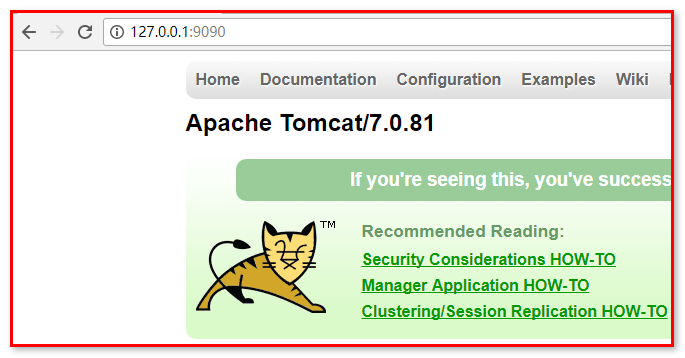
tomcat的默认端口为8080：



修改tomcat的端口为9090：

修改完server.xml文件必须重启服务器才能有效。

通过浏览器的地址栏访问：http://localhost:9090



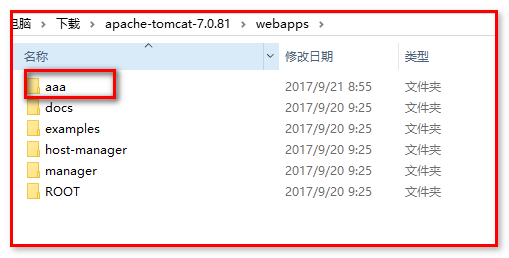
## 项目(web资源)发布

### 在webapps下面直接发布：(重点)

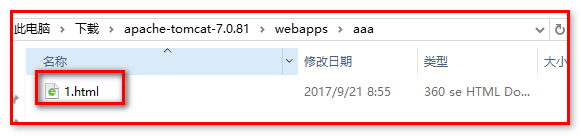
只需要把开发好的项目复制到webapps下面即可。这时不用重启tomcat服务器，tomcat服务器会自动的加载复制到webapps下面的所有项目资源。

#### 在aaa文件夹下新建01.html

创建aaa文件夹：



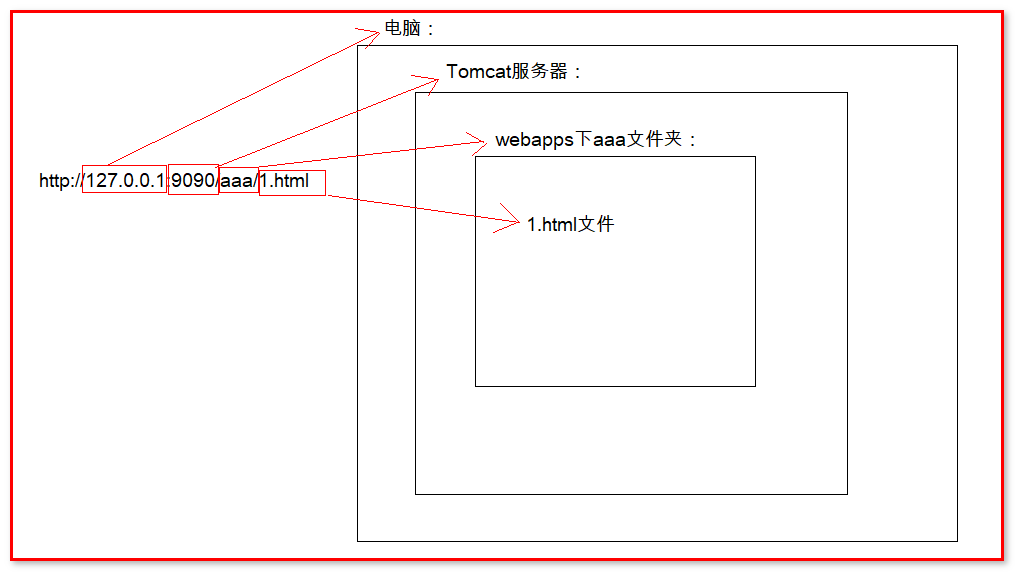
定义一个html文件

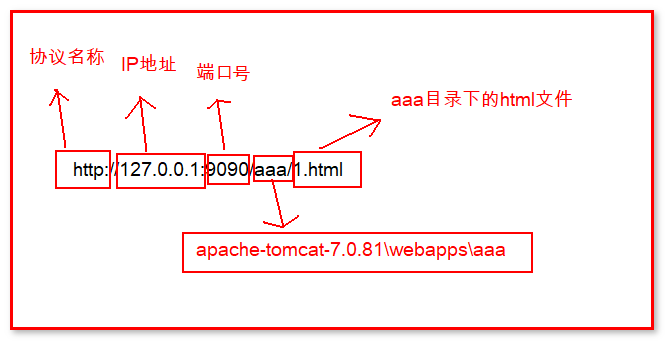


通过浏览器地址栏访问：http://localhost:9090/aaa/1.html



访问过程：



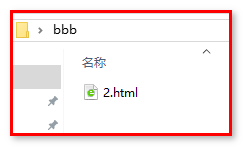


#### 使用压缩包发布项目，把压缩包发布到webapps下面：（重点）

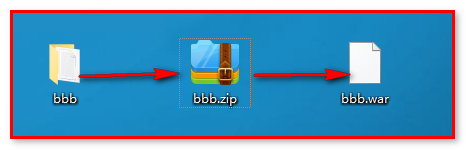
原因：直接复制，数据量大，文件多，管理不方便

使用.war发布好处： 文件在一个文件夹统一管理，数据进行压缩，数据量小，.war文件可以被tomcat自动解压，直接访问就可以获取资源。

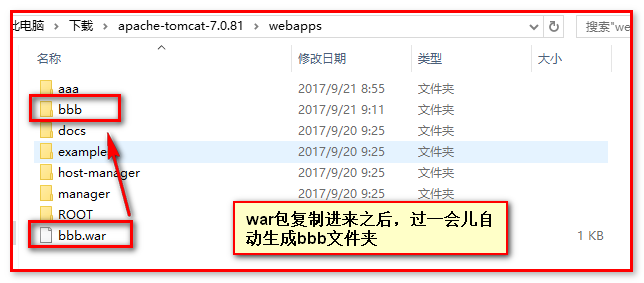
在新目录bbb创建02.html



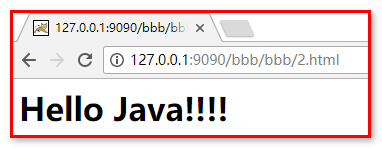
将工程使用zip格式压缩，然后更改后缀名为 war。



把war包复制到wabapps下面：如果不能自动生成，请重启服务器



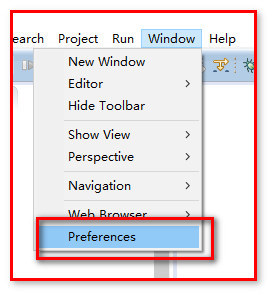
通过浏览器地址直接访问：http://localhost:9090/bbb/bbb/02.html

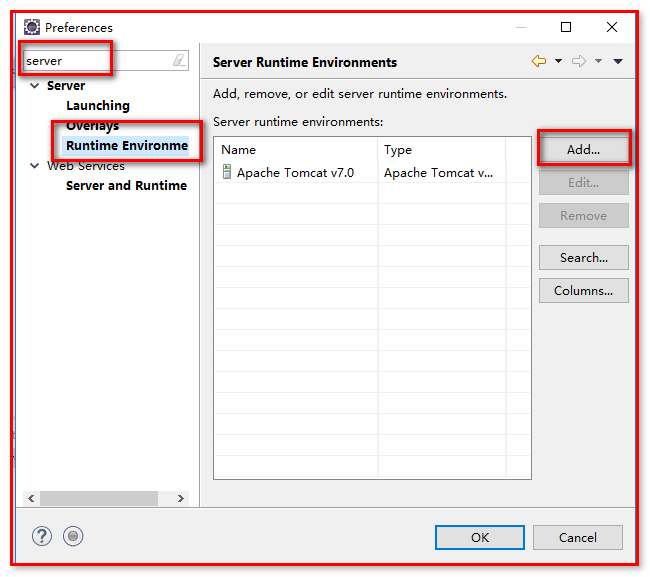


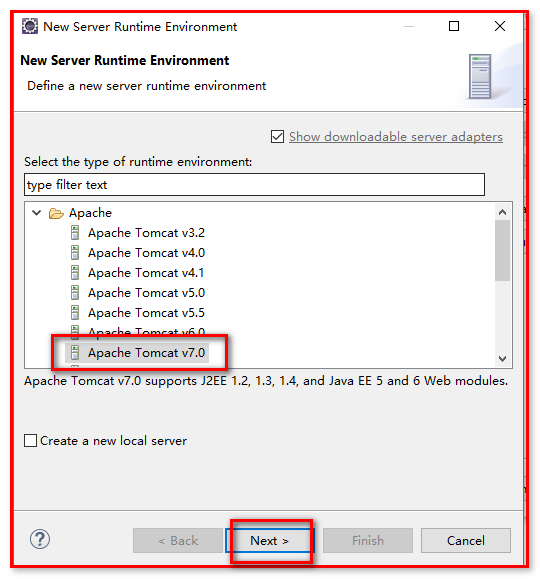
# Eclipse集成tomcat和开发web项目

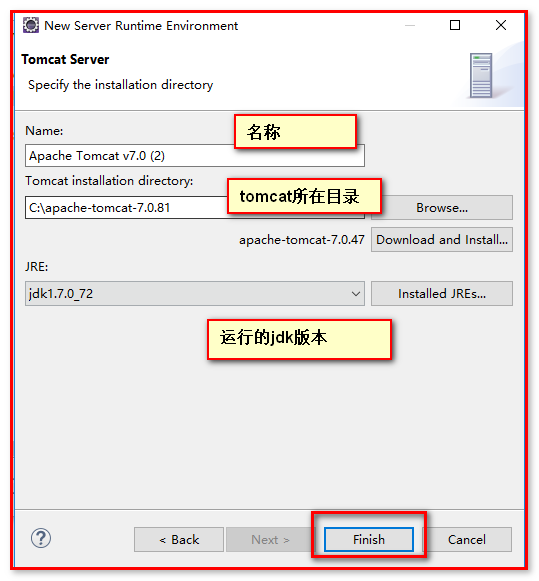
集成：使用eclipse操作tomcat服务器。

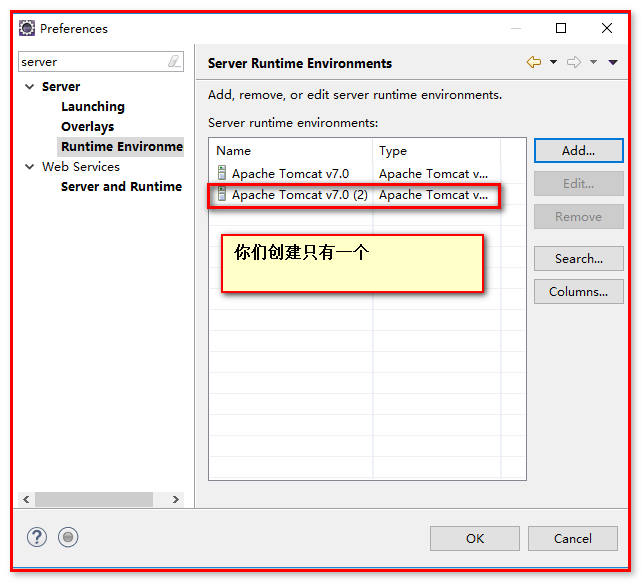
## Eclipse集成tomcat(重点)



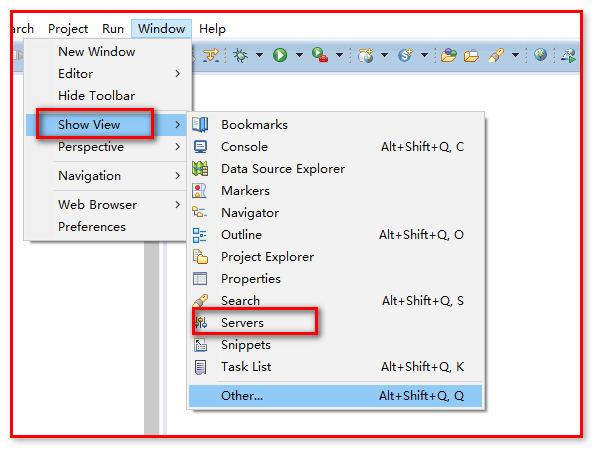


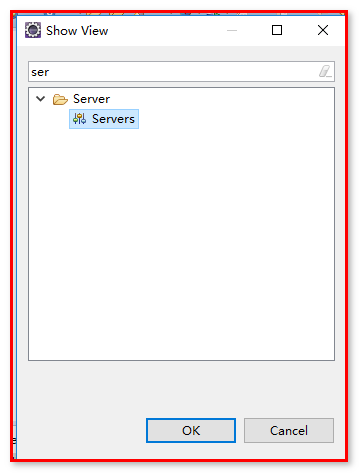


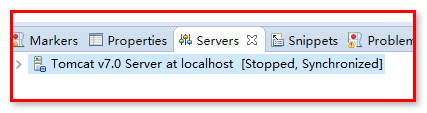




### 调出server 视图

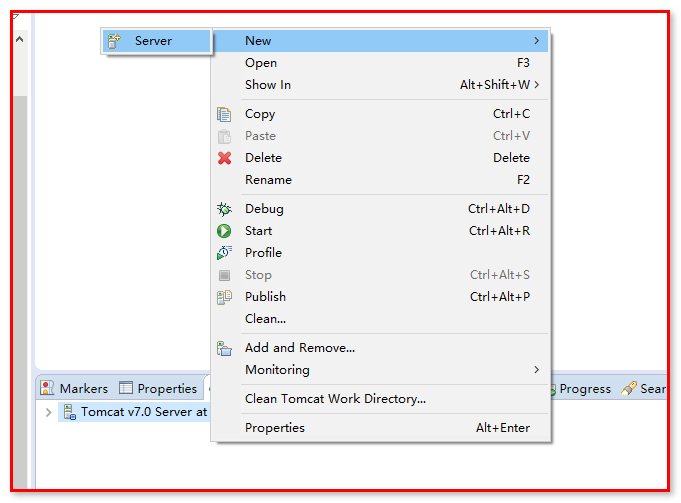


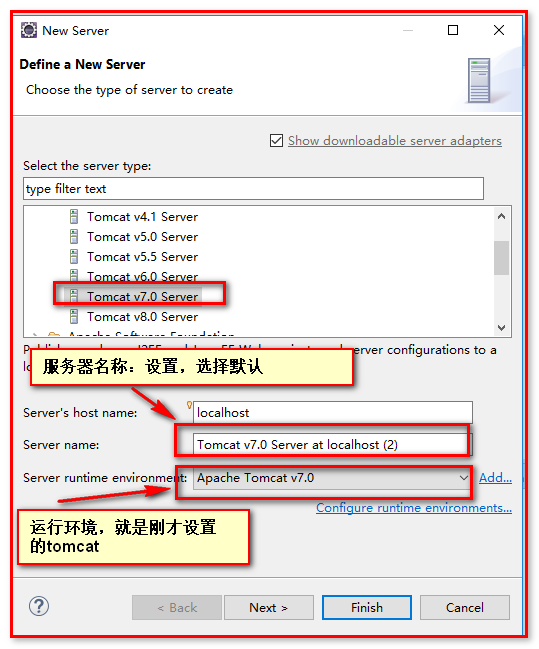




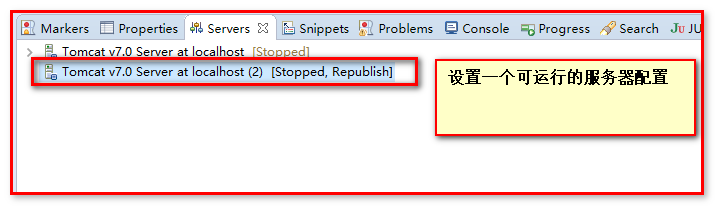
如果没出现，tomcat服务器，那么：

右键 new Server:





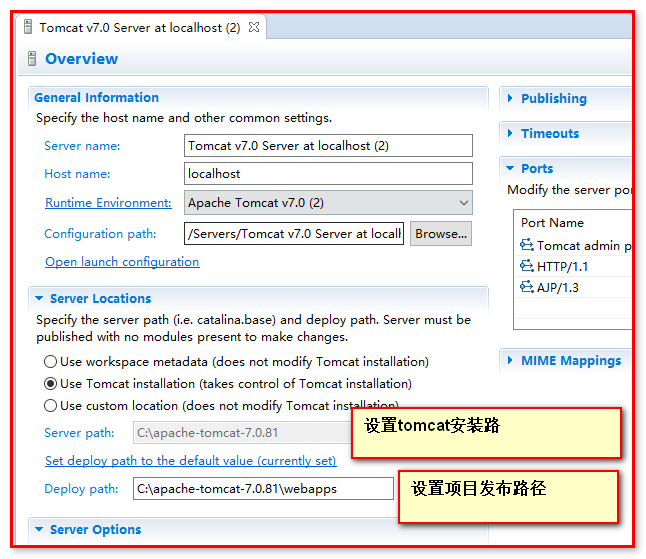
点击finished完成



### 配置tomcat信息

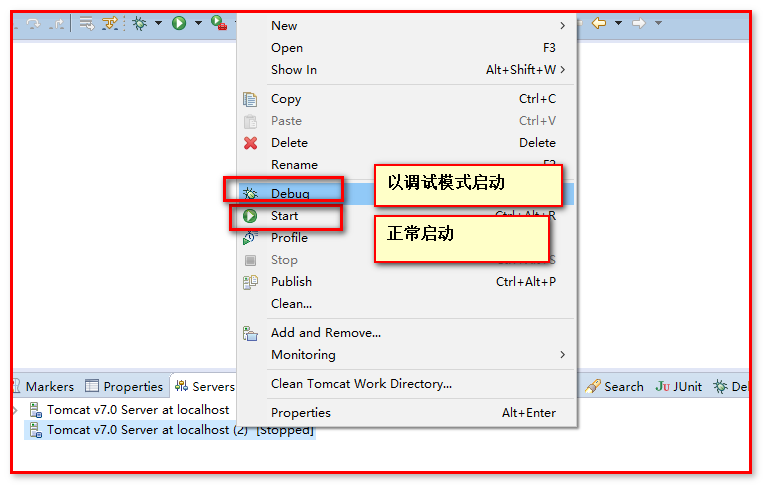
双击打开，修改配置：



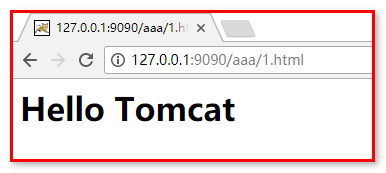


一定别忘了 ctrl + s 保存

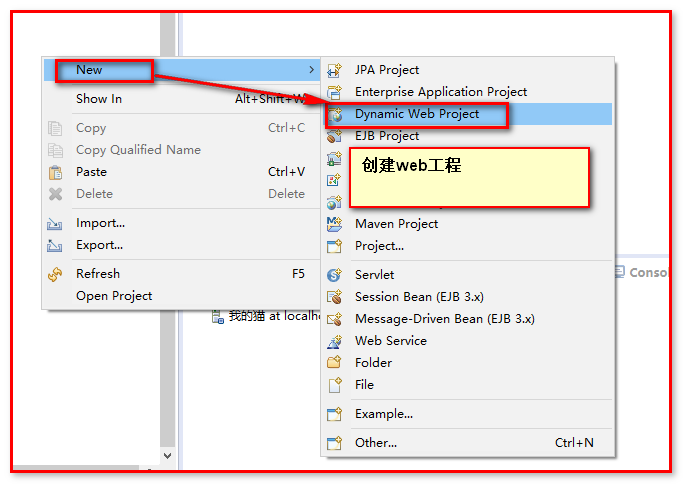
启动服务器：

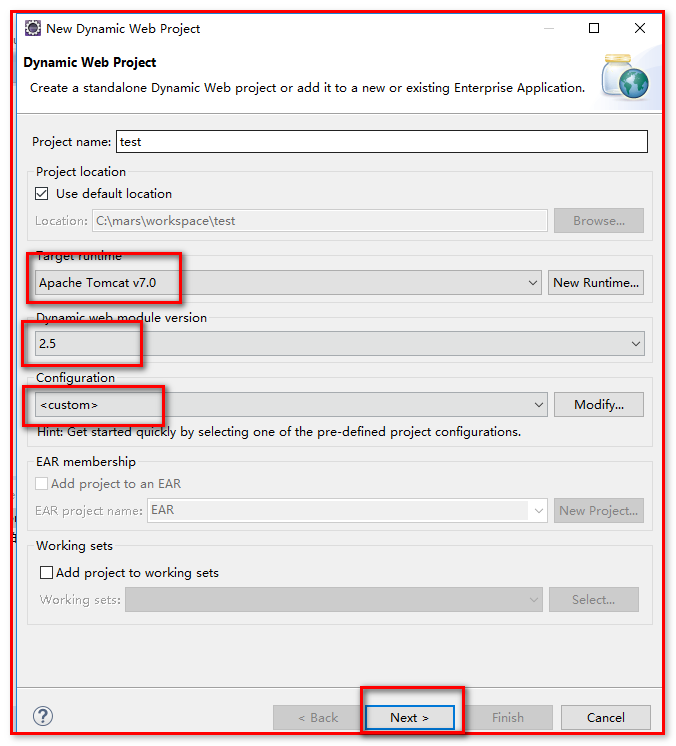


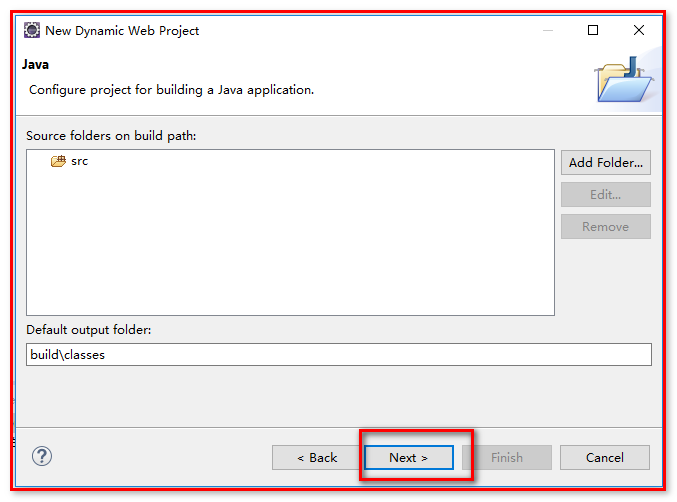
访问：

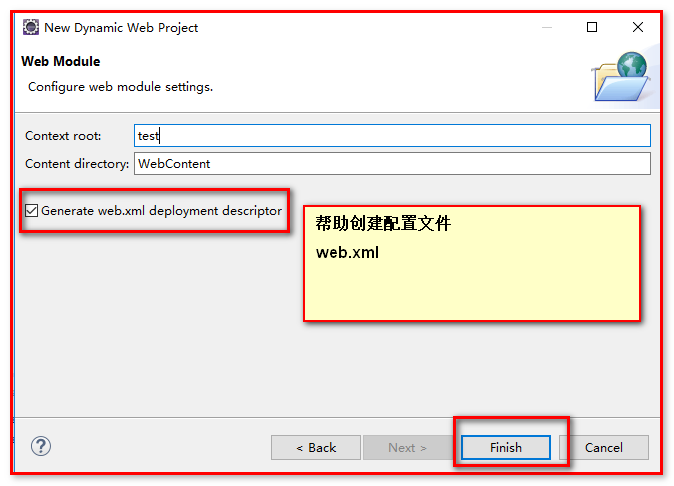


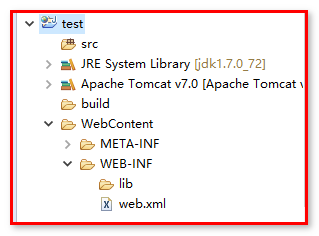
## Eclipse创建web项目（写代码，在服务器可以执行）



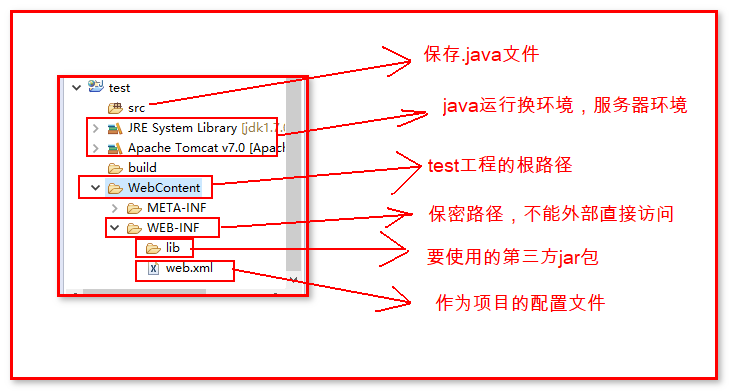




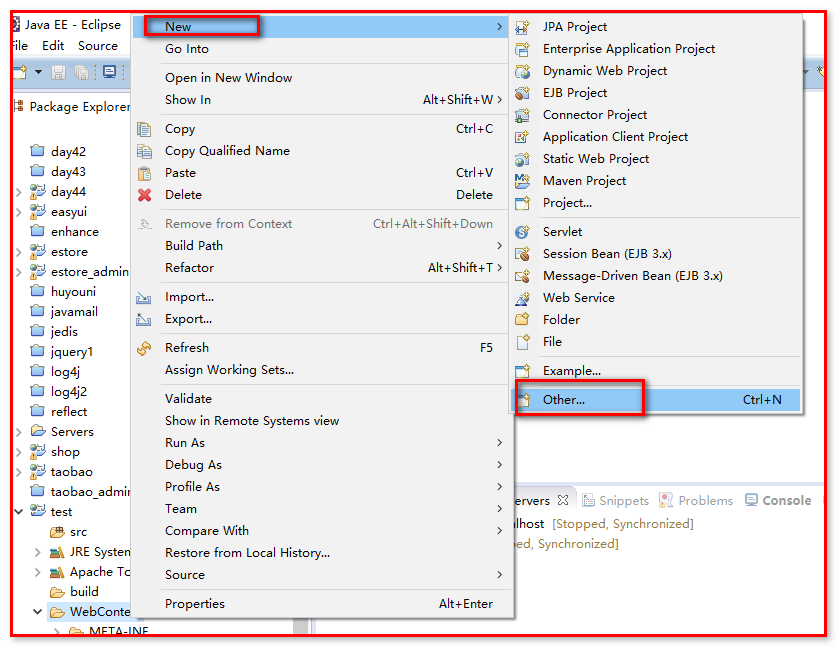


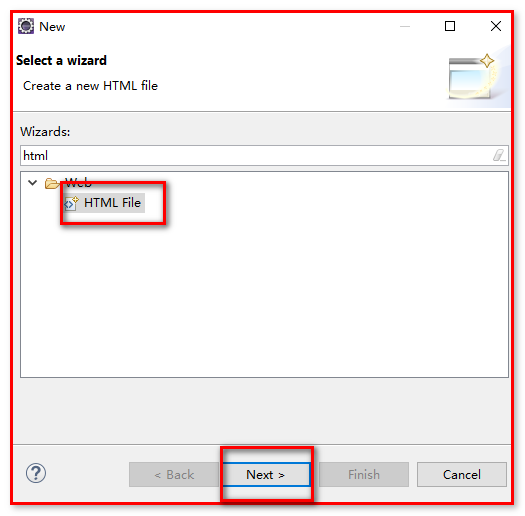


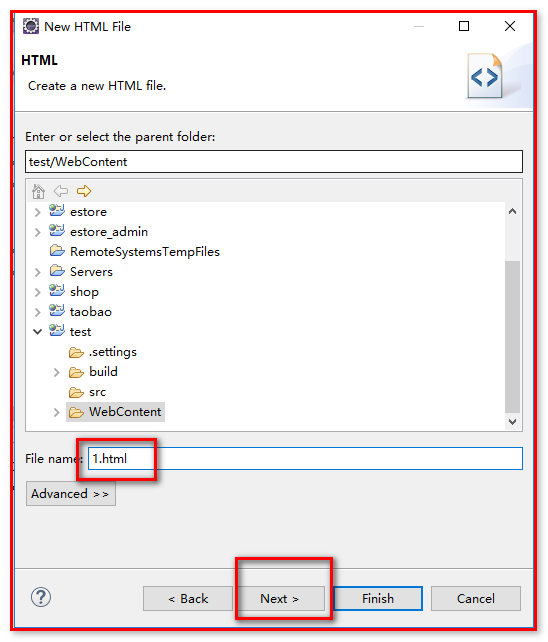
## 当前项目的目录介绍

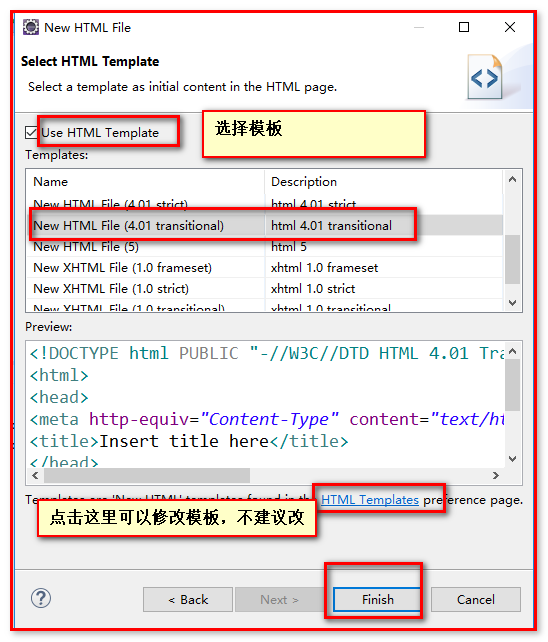


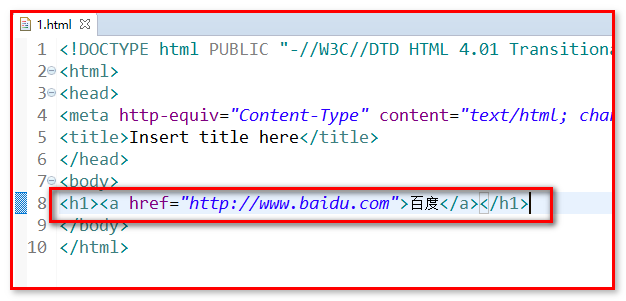
### 将web项目发布到tomcat上,对外发布资源,访问资源



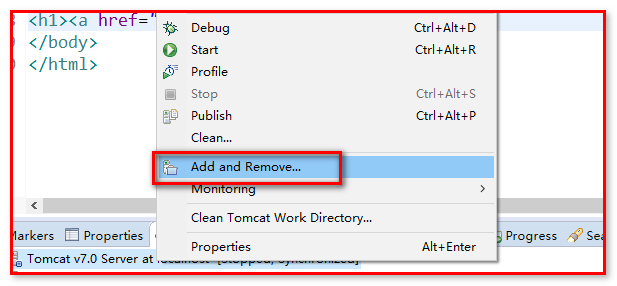


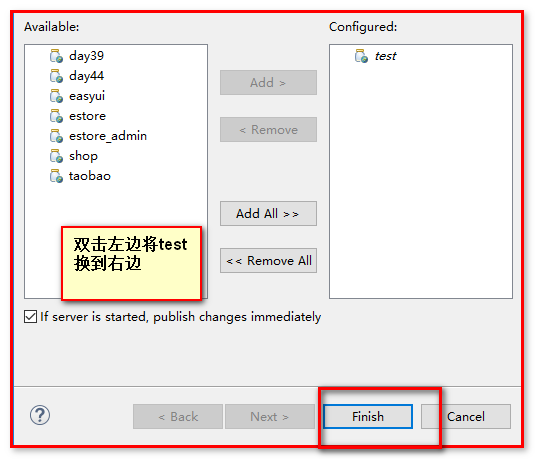




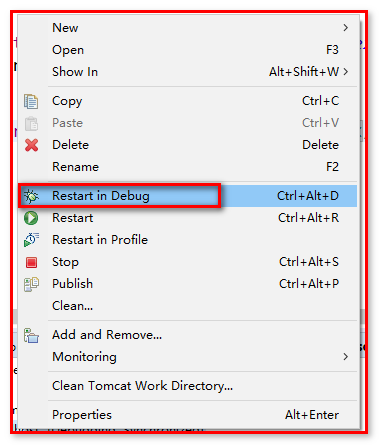


右键：





再次启动：



效果：



## 使用eclipse将项目发布到tomcat都做了那些事呢？

步骤1: elipse 会把src中java文件编译成class文件, 放到 WebRoot/WEB-INF/classes 目录下.

步骤2: eclipse将WebRoot 复制粘贴到tomcat/webapps目录下, 并且将 WebRoot 改名为 项目名(如test).

# HTTP协议

## 什么是协议？

协议：是通信双方应该遵守的一个共同的规则。相当于两个人交谈要使用同一种语言。

那么，两台计算机（两个人）按照指定的规则，一方发送数据（一个人说），另外一方才能接受到数据（另一个人听）。

让互联网中的计算机可以相互收发数据的这个规则，就是HTTP协议。

http协议：

超文本传输协议（HTTP，HyperText Transfer Protocol)是互联网上应用最为广泛的一种网络协议。所有的WWW文件都必须遵守这个标准。设计HTTP最初的目的是为了提供一种发布和接收HTML页面的方法。1960年美国人Ted Nelson构思了一种通过计算机处理文本信息的方法，并称之为超文本（hypertext）,这成为了HTTP超文本传输协议标准架构的发展根基。Ted Nelson组织协调万维网协会（World Wide Web Consortium）和互联网工程工作小组（Internet Engineering Task Force ）共同合作研究，最终发布了一系列的RFC，其中著名的RFC 2616定义了HTTP 1.1。

http的相关概念： http://127.0.0.1:9090/test/1.html

1、域名代替IP，默认端口是80（如果端口号是80，可以省略不写）

<http://www.baidu.com:80> 相当于 http://www.baidu.com

[http:// www.baidu.com](http://localhost)  代替 IP: 115.239.211.112

2、http协议永远是客户端（浏览器或者手机等）发送请求，服务器回送响应



客户端访问服务端称为客户端的请求，服务器给浏览器回送数据称为服务器的响应。

3、http协议是基于TCP协议。

注意：基础班学习网络编程的时候学习过：UDP、TCP他们都是传输层的协议。

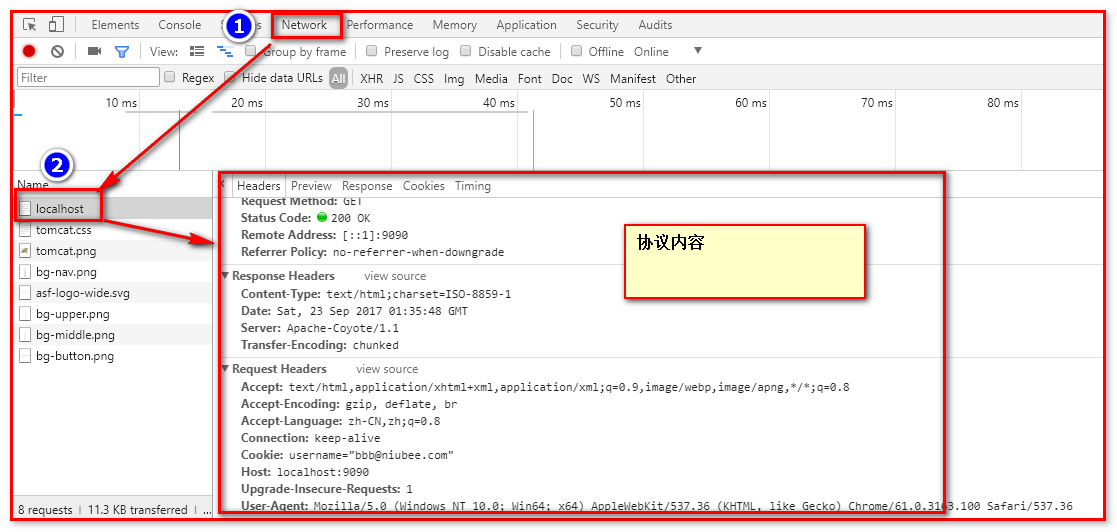
就业班学习的协议基本都是应用层的协议：主要是针对的是应用程序。（什么是应用层协议，参考资料文件夹中的《网络七层协议介绍》，该内容做了了解，不要求掌握）

那么之前安装好tomcat之后，我们也在使用浏览器获取tomcat服务器的资源，使用的也是http协议，如何查看http协议呢？

答：chrome(谷歌)：内置 f12 firefox(火狐)：安装firebug

访问：tomcat效果：

chrome(谷歌)：内置 f12：

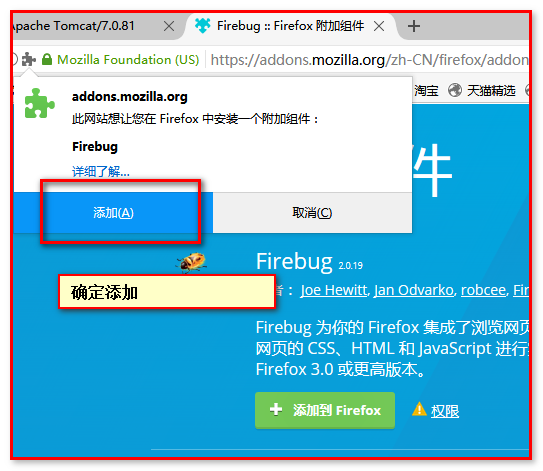


firefox(火狐)：安装firebug

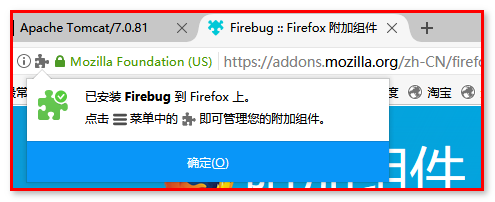
安装地址：<https://addons.mozilla.org/zh-CN/firefox/addon/firebug/>

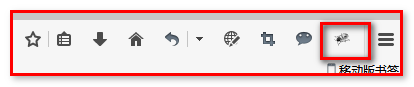
使用上面这个地址，访问后页面出现：



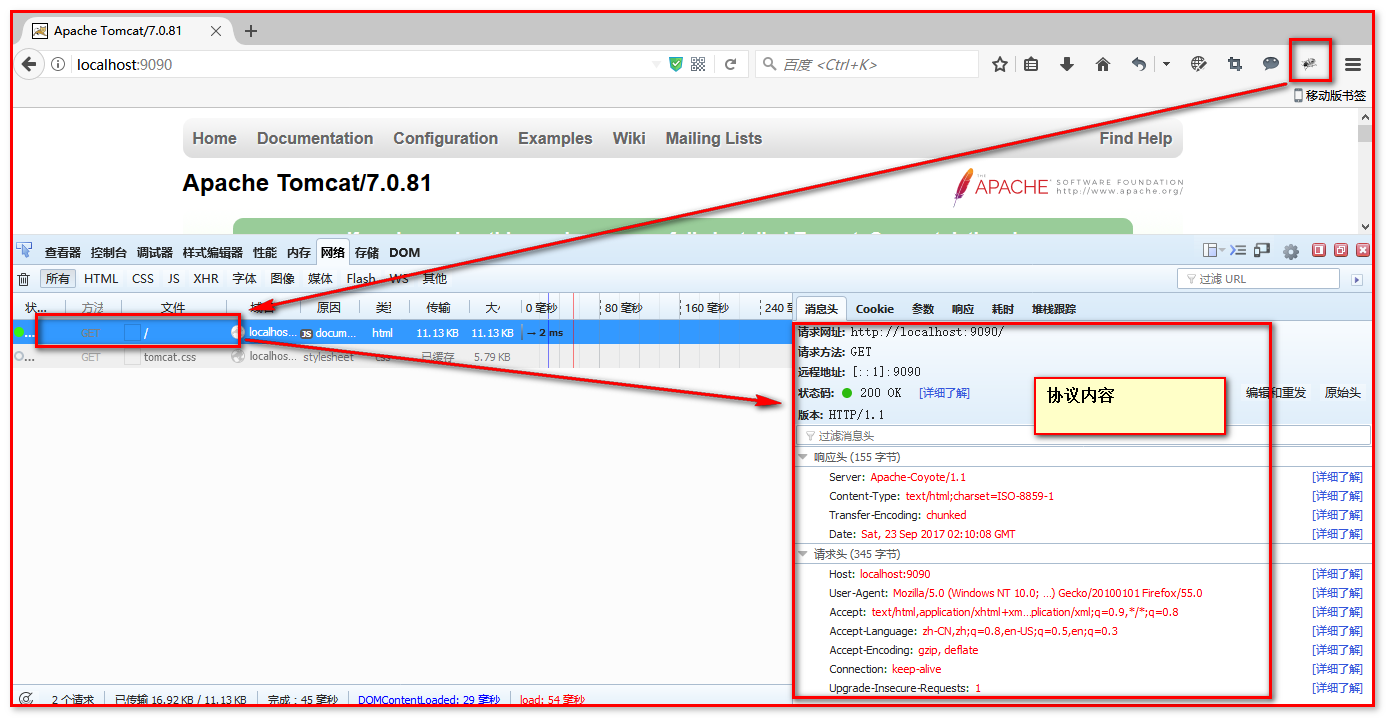


完成：





使用：



使用两个浏览器的原因：用户使用那种浏览器，程序员并不知道，但是，市面上主要的浏览器，就是谷歌和火狐，国产浏览器内核也是谷歌或者火狐，因此我们需要掌握两种浏览器的使用。

## HTTP协议中的请求格式（方向：浏览器 -》 服务器）

目的1: 知道浏览器把什么样的数据发送给服务器

目的2: 知道发送的数据格式

准备知识：

注意：HTTP请求方法有很多，我们目前要了解和使用的是get和post，所以在演示和解析请求的时候，我们演示get和post请求

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| HTTP请求方法 根据HTTP标准，HTTP请求可以使用多种请求方法。  HTTP1.0定义了三种请求方法： GET, POST 和 HEAD方法。  HTTP1.1新增了五种请求方法：OPTIONS, PUT, DELETE, TRACE 和 CONNECT 方法。   |  |  |  | | --- | --- | --- | | **序号** | **方法** | **描述** | | 1 | GET | 请求指定的页面信息，并返回实体主体。 | | 2 | HEAD | 类似于get请求，只不过返回的响应中没有具体的内容，用于获取报头 | | 3 | POST | 向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST请求可能会导致新的资源的建立和/或已有资源的修改。 | | 4 | PUT | 从客户端向服务器传送的数据取代指定的文档的内容。 | | 5 | DELETE | 请求服务器删除指定的页面。 | | 6 | CONNECT | HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。 | | 7 | OPTIONS | 允许客户端查看服务器的性能。 | | 8 | TRACE | 回显服务器收到的请求，主要用于测试或诊断。 | |

 准备测试html：

|  |
| --- |
| <form action=*"http://localhost:9090/test/1.html"* method=*"post"*>  <input type=*"text"* name=*"username"*><br>  <input type=*"submit"* value=*"提交"*>  </form> |

 介绍HTTP协议中规定的客户端请求服务器数据时的请求协议格式：

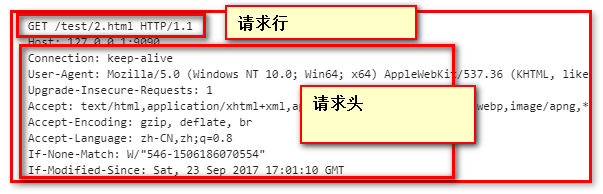
http规定：请求的数据必须分成三部分：

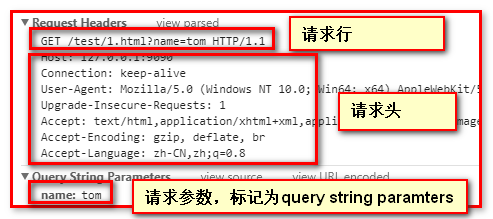
1、 请求行

2、 请求头

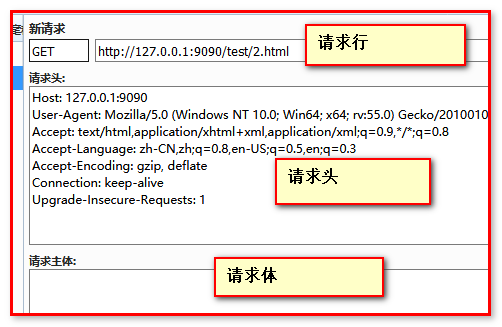
3、 请求体

这是get请求





火狐：



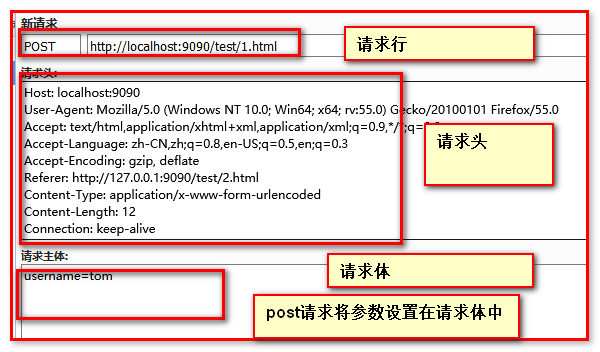
注意:get请求没有请求体，请求参数在请求行中

这是post请求:

谷歌：



火狐：



注意：post请求，请求参数在请求体中

### 请求行：分成3部分：

请求行：GET /test/2.html HTTP/1.1

GET： 请求方式

/test/1.html?name=tom ：请求地址 ？后面是请求参数（用户名密码等等。。。）

HTTP/1.1：HTTP版本

什么是请求方式？

答：请求方式就相当于回家的开门方式：get方式相当于拿钥匙开门 post方式相当于电子密码锁开门

问：那么使用浏览器如何发送get请求？

答：A标签和直接地址栏输入都是get请求

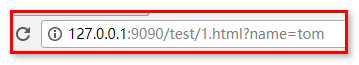
问：那么使用浏览器如何发送post请求？

答：使用form表单就是post请求

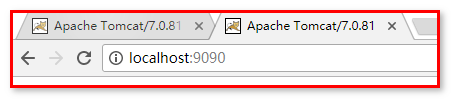
GET和POST的区别？

第一：

Get请求：暴露请求参数在地址栏不安全



POST：不会暴露参数





第二：

Get请求方式数据有长度限制，所有浏览器，都不允许，在地址无限的书写内容

Post请求方式，长度没有限制，可用户文件上传这样的功能。

什么是请求地址？

答：相当于你家的门牌号码，通过这个找到你家

请求地址说明：

请求地址：/test/1.html?name=tom

“/” 当前服务器根路径——http://localhost:9090

“test”服务器中资源名称

“？”开启拼接请求参数

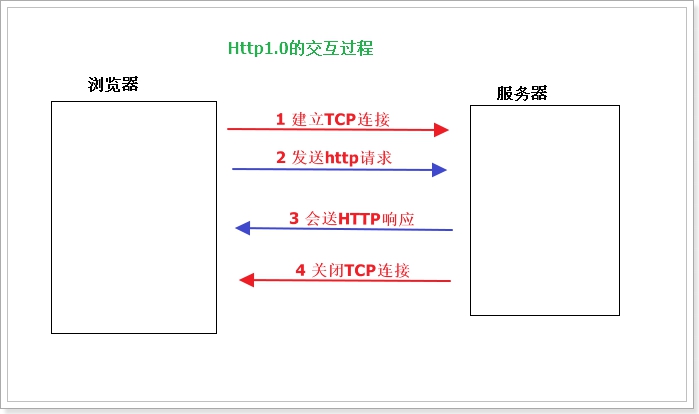
“name=tom”请求参数 post方式请求参数的请求体中

什么是HTTP版本？（了解）

答：跟软件一样，协议，过一段时间要适合互联网发展，做更新

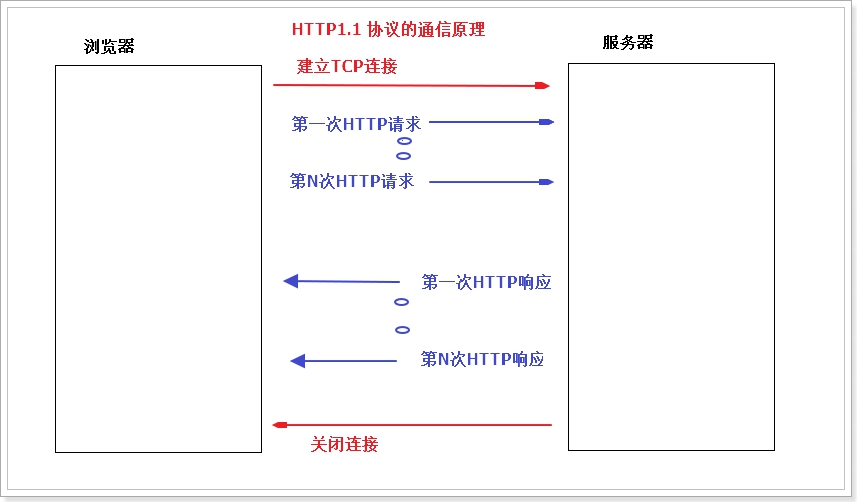
HTTP现在使用的版本：HTTP/1.1

早期1.0版本运行图解：一次请求，获取一次资源（响应），效率慢



明显缺陷：浏览器和服务器建立连接后，每次只能处理一次请求。

为了克服上述1.0的缺陷，http1.1 应运而生。它支持持久连接，也就是说在一个TCP连接上可以传送多个http请求和响应，从而减少建立和关闭连接的消耗延时。



### 请求头：

它是有key和value组成的数据，key和value之间必须使用冒号隔开。一个key可以对应多个value值。

Host localhost:9090 服务器的域名

User-Agent Mozilla/5.0 (Windows NT 10.0; …) Gecko/20100101 Firefox/55.0 首部包含了一个特征字符串，用来让网络协议的对端来识别发起请求的用户代理软件的应用类型、操作系统、软件开发商以及版本号。

Accept text/html,application/xhtml+xm…plication/xml;q=0.9,\*/\*;q=0.8 客户端可以处理的内容类型（text/html，超文本）

Accept-Language zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3 使用的语言

Accept-Encoding gzip, deflate 压缩格式

Referer <http://127.0.0.1:9090/test/1.html?name=tom> 当前页面的来源页面的地址

Content-Type application/x-www-form-urlencoded 发送前对所有字符进行编码

Content-Length 12 数据长度

Connection keep-alive 保持网络连接

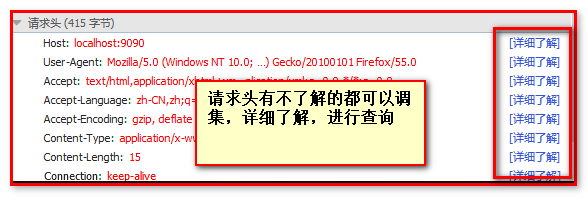
Upgrade-Insecure-Requests 1客户端优先选择加密及带有身份验证的响应

后期需要使用到的是：

Referer：可以用在用户登陆之后，跳转之前浏览的页面功能上

User-Agent：处理浏览器兼容性问题

有不明白的消息头，可以按一下方式查询：



请求体：post请求，会将请求要发送的参数，设置到请求体中

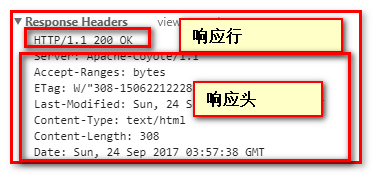
## HTTP协议中的响应格式（服务器 --》 浏览器）

http协议中在定义响应格式：分成3部分：

1、 响应行

2、 响应头

3、 响应体



响应体：

|  |
| --- |
| <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>Insert title here</title>  </head>  <body>  <h1><a href="http://www.baidu.com">百度</a></h1>  </body>  </html> |

### 响应行：分成三部分：

响应行：HTTP/1.1 200 OK

http/1.1 表示http协议和版本

200 响应状态码

OK 状态码代表意义 ，请求成功，响应完成

Xml技术？

-1 888 20170924xxxxx

<msg>

<status>-1</status>

<time>888</time>

<orderid>20170924xxxxx </orderid>

</msg>

### 响应行中的状态码：

常见状态码：

200 - 请求成功

302 – 请求重定向 （response）

404 - 请求的资源（网页等）不存在

500 - 内部服务器错误（如果遇到500的情况，表示的是java代码出错误）

状态码分类：



详细状态码：







### 常见的响应头信息参数说明

Location: http://www.it315.org/index.jsp   //  通常告知浏览器 马上向该地址发送请求 通常 和 302 一起使用!!

（重点） 结合302完成重定向 操作 Location重定向后地址

Server:apache tomcat 服务器类型

Content-Encoding: gzip  响应编码类型 gzip压缩

Content-Length: 800  响应体的长度

Content-Language: zh-cn 响应语言

Content-Type: text/html; charset=GB2312 响应字符集   决定浏览器打开文件格式类型 以及编码

Last-Modified: Tue, 11 Jul 2000 18:23:51 GMT 和If-Modified-Since 一起使用，实现服务器缓存策略

Refresh: 3;url=http://www.jd.com （页面自动刷新）

 3 表示是的时间，3秒

url 地址

到3秒后，页面刷新url

### 响应体——页面要展示的内容

服务器输出给浏览器显示的内容.

响应体：

|  |
| --- |
| <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  <html>  <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>Insert title here</title>  </head>  <body>  <!-- <h1><a href="http://www.baidu.com">百度</a></h1> -->  <form action="http://localhost:9090/test/1.html" method="post">  <input type="text" name="username"><br>  <input type="submit" value="提交">  </form>  </body>  </html> |

总结：

