

HTTP&Tomcat

今日内容介绍

- ◆ 访问 tomcat 下已经发布的 web 项目

今日内容学习目标

- ◆ 能够描述出浏览器和服务器交互过程
- ◆ 能够产生 HTTP 请求协议格式
- ◆ 能够产生 HTTP 响应协议格式
- ◆ 看得懂 WEB 项目的目录结构
- ◆ 使用 Tomcat 发布 web 项目，并成功访问
- ◆ 在 eclipse 下发布 web 项目

第1章 访问tomcat下已经发布的web项目

为了可以通过浏览器访问到自己 tomcat 下的 web 项目，我们需要先了解以下内容：

1.1 HTTP 协议的概述：

1.1.1 什么是 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。用于定义 WEB 浏览器与 WEB 服务器之间交换数据的过程。

http

 编辑

本词条由“[科普中国](#)”百科科学词条编写与应用工作项目 审核。

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。

1.1.2 HTTP 协议的作用及特点

HTTP 协议的作用

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 协议的特点

- 基于请求/响应模型的协议。请求和响应必须成对；先有请求后有响应。
- HTTP 协议默认的端口:80
 - 例如：<http://www.itheima.com:80>

1.1.3 HTTP 协议的版本：

- HTTP/1.0，发送请求，创建一次连接，获得一个 web 资源，连接断开。
- HTTP/1.1，发送请求，创建一次连接，获得多个 web 资源，连接断开。

1.1.4 HTTP 协议的组成：

HTTP 请求协议、HTTP 响应协议。

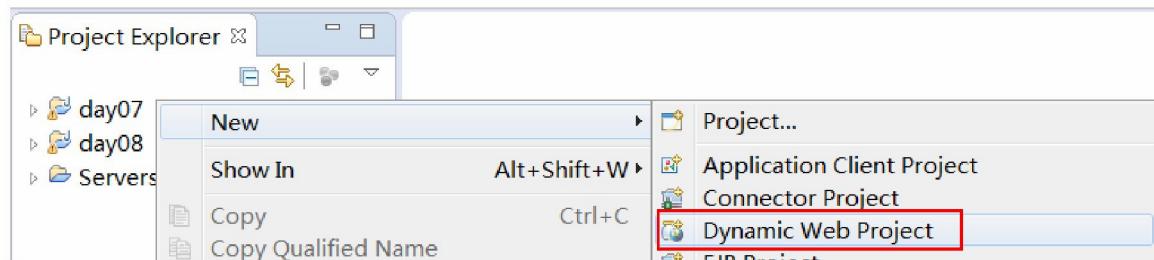
HTTP 请求包括：请求行、请求头、请求体

HTTP 响应包括：响应行、响应头、响应体

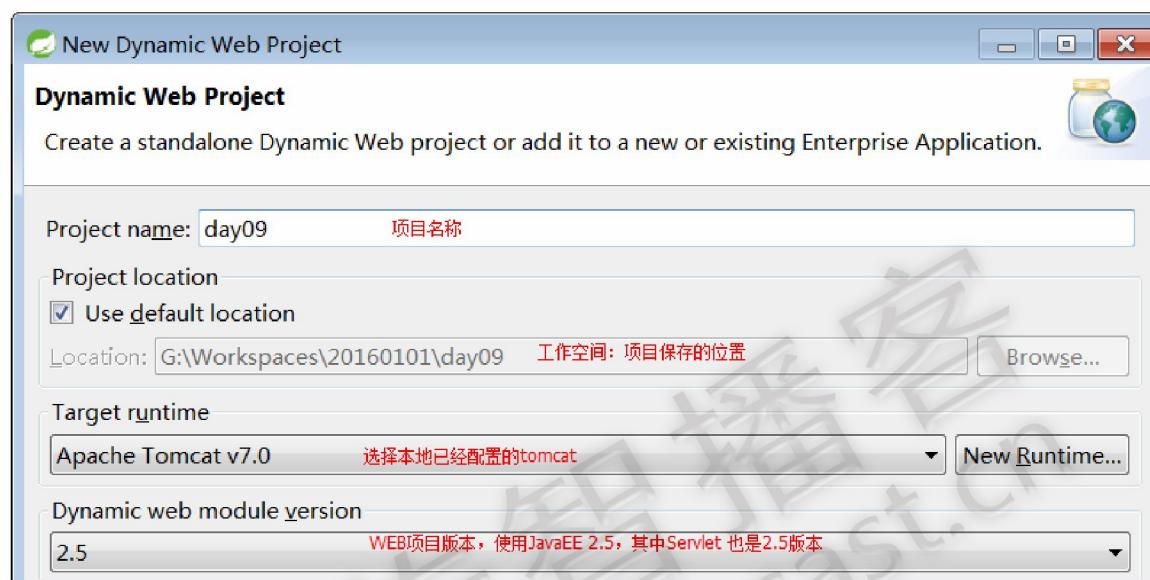
1.1.5 HTTP 协议入门

1.1.5.1 准备工作

1. 创建项目，JavaWeb 项目，选择版本为 2.5
- 步骤 1：在 STS 中 New/Dynamic Web Project



● 步骤 2:



2. 测试用例，编写“form.html”页面，并提供两个表单，分别设置表单的提交方式为：get 和 post。将表单提交位置设置成#，表示提交到当前表单。

▲ WebContent

 ▲ 01.http

 form.html

```
<form action="#" method="get">
    用户名: <input type="text" name="username" value="jack" /> <br/>
    密码: <input type="text" name="password" value="1234" /> <br/>
    <input type="submit" value="get 提交" />
</form>

<form action="#" method="post">
    用户名: <input type="text" name="username" value="jack" /> <br/>
    密码: <input type="text" name="password" value="1234" /> <br/>
    <input type="submit" value="post 提交" />
</form>
```

3. 安装 HttpWatch，用于抓取 HTTP 协议的数据包（抓包）



1.1.5.2 HTTP 请求的详解

HTTP 请求格式：请求行、请求头、请求体。

如下图，我们提供两种请求方式抓包结果：

GET 请求抓包数据：

```
GET /day09/01.http/form.jsp?username=jack&username=1234 HTTP/1.1 ← 请求行
Accept: application/x-ms-application, image/jpeg, application/xaml+xml, image/gif, image/pjpeg, applic
Referer: http://localhost:8080/day09/01.http/form.jsp
Accept-Language: ar-AE, ja-JP;q=0.8, ko-KR;q=0.5, zh-CN;q=0.3
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; qdesk 2.5.1277.202;
Accept-Encoding: gzip, deflate
Host: localhost:8080
Connection: Keep-Alive
Cookie: JSESSIONID=A151F9954997A6AF02AE9D1F1AF15ECC
```

POST 请求抓包数据：

```
POST /day09/01.http/form.html HTTP/1.1 ← 请求行
Accept: application/x-ms-application, image/jpeg, application/xml+xml, image/gif, image/pjpeg, applica
Referer: http://localhost:8080/day09/01.http/form.html
Accept-Language: ar-AE,ja-JP;q=0.8,ko-KR;q=0.5,zh-CN;q=0.3
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; qdesk 2.5.1277.202;
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
Host: localhost:8080 ← 请求头
Content-Length: 27
Connection: Keep-Alive
Cache-Control: no-cache
Cookie: JSESSIONID=A151F9954997A6AF02AE9D1F1AF15ECC

username=jack&username=1234 ← 请求体
```

● 请求行

例如：POST /day09/01.http/form.html HTTP/1.1

请求行必须在 HTTP 请求格式的第一行。

请求行格式：请求方式 资源路径 协议/版本

请求方式：协议规定 7 种，常用两种：**GET** 和 **POST**

GET 请求：

将请求参数追加在 URL 后面，不安全。例如：form.html?username=jack&username=1234
URL 长度限制 GET 请求方式的数据大小。

没有请求体

POST 请求

请求参数显示请求体处，较安全。

请求数据大小没有显示。

只有表单设置为 method="post" 才是 post 请求，其他的都是 get 请求。

常见 GET 请求：地址栏直接访问、``、`` 等

● 请求头

例如：Host: localhost:8080

请求头从第二行开始，到第一个空行结束。及请求头和请求体之间存在一个空行。

请求头通常以键值对 (key:value) 方式传递数据。

`key` 为规范规定的固定值

`value` 为 `key` 对应的取值，通常是一个值，可能是一组。

常见请求头	描述（红色掌握，其他了解）
Referer	浏览器通知服务器，当前请求来自何处。如果是直接访问，则不会有这个头。 常用于：防盗链
If-Modified-Since	浏览器通知服务器，本地缓存的最后变更时间。与另一个响应头组合控制浏览器页面的缓存。
Cookie	与会话有关技术，用于存放浏览器缓存的 cookie 信息。
User-Agent	浏览器通知服务器，客户端浏览器与操作系统相关信息
Connection	保持连接状态。Keep-Alive 连接中，close 已关闭
Host	请求的服务器主机名
Content-Length	请求体的长度
Content-Type	如果是 POST 请求，会有这个头，默认值为 application/x-www-form-urlencoded，表示请求体内容使用 url 编码
Accept:	浏览器可支持的 MIME 类型。文件类型的一种描述方式。 MIME 格式：大类型/小类型[;参数] 例如： text/html，html 文件 text/css，css 文件 text/javascript，js 文件 image/*，所有图片文件
Accept-Encoding	浏览器通知服务器，浏览器支持的数据压缩格式。如：GZIP 压缩
Accept-Language	浏览器通知服务器，浏览器支持的语言。各国语言（国际化 i18n）

● 请求体

通常情况下，只有 post 请求方式才会使用到请求体，请求体中都是用户表单提交的数据，每一项数据都使用键值对 (`k=v`)，多组值使用&相连。

例如：username=jack&password=1234

1.1.5.3 HTTP 响应的详解

HTTP 响应格式：响应行、响应头、响应体

如下图，我们提供的响应的抓包结果（HttpWatch 只支持 GBK 编码，否则中文会出现乱码）

```

HTTP/1.1 200 OK ← 响应行
Server: Apache-Coyote/1.1
Accept-Ranges: bytes
ETag: W/"567-1458657455925"
Last-Modified: Tue, 22 Mar 2016 14:37:35 GMT
Content-Type: text/html ← 响应头
Content-Length: 567
Date: Tue, 22 Mar 2016 14:39:35 GMT
      此处有空行
<!DOCTYPE html> ← 响应体(正文)
<html>
<head>
<meta charset="GBK">
<title>Insert title here</title>
</head>
<body>
    <form action="#" method="get">
        用户名: <input type="text" name="username" value="jack" /> <br/>
        密码: <input type="text" name="password" value="1234" /> <br/>
        <input type="submit" value="get提交" />
    </form>

    <form action="#" method="post">
        用户名: <input type="text" name="username" value="jack" /> <br/>
        密码: <input type="text" name="password" value="1234" /> <br/>
        <input type="submit" value="post提交" />
    </form>
</body>
</html>

```

● 响应行

例如: HTTP/1.1 200 OK

格式: 协议/版本 状态码 状态码描述

状态码: 服务器与浏览器用于确定状态的固定数字号码

200 : 请求成功。

302 : 请求重定向。

304 : 请求资源没有改变, 访问本地缓存。

404 : 请求资源不存在。通常是用户路径编写错误, 也可能是服务器资源已删除。

500 : 服务器内部错误。通常程序抛异常。

● 响应头

响应头也是用的键值对 k:v

服务器通过响应头来控制浏览器的行为, 不同的头浏览器操作不同。

常见请求头	描述
Location	指定响应的路径, 需要与状态码 302 配合使用, 完成跳转。
Content-Type	响应正文的类型 (MIME 类型) 取值: text/html;charset=UTF-8
Content-Disposition	通过浏览器以下载方式解析正文 取值: attachment;filename=xx.zip
Set-Cookie	与会话相关技术。服务器向浏览器写入 cookie
Content-Encoding	服务器使用的压缩格式 取值: gzip
Content-length	响应正文的长度
Refresh	定时刷新, 格式: 秒数;url=路径。url 可省略, 默认值为当前页。

	取值: 3;url=www.itcast.cn //三秒刷新页面到 www.itcast.cn
Server	指的是服务器名称，默认值: Apache-Coyote/1.1。可以通过 conf/server.xml 配置进行修改。<Connector port="8080" ... server="itcast"/>
Last-Modified	服务器通知浏览器，文件的最后修改时间。与 If-Modified-Since 一起使用。

- 响应体

响应体，就是服务器发送给浏览器的正文。

1.2 Web 开发概述

1.2.1 WEB 通信

WEB 采用 B/S 通信模式，通过超文本传送协议(HTTP, Hypertext transport protocol)进行通信。通过浏览器地址栏编写 URL，向服务器发送一个请求，服务器端根据请求进行相应的处理，处理完成之后，会向浏览器作出一个响应，及将服务器端资源发送给浏览器。



1.2.2 软件架构

- C/S 架构: Client/Server 客户端/服务器。要求客户端电脑安装一个客户端程序。
 - 常见应用: QQ, 迅雷, 360, 旺旺 等
 - 优点:
 1. 用户体验好，效果炫
 2. 对信息安全的控制较强
 3. 应用服务器运行数据负荷较轻，部分计算功能在客户端完成。
 - 缺点:
 1. 占用硬盘空间
 2. 维护麻烦
 3. 安装使用依赖其他条件
- B/S 架构: Browser/Server 浏览器/服务器。通过浏览器与服务器交互，不需要安装其他程序
 - 常见应用: 网银系统, 淘宝, 京东 12306 等
 - 优点:

1. 维护和升级简单，无缝升级。
2. 不用必须安装程序，操作系统内置了浏览器。

■ 缺点：

1. 动画效果受浏览器限制
2. 对信息安装控制较差。例如：网银就需要使用 U 盾，在浏览器端加密。
3. 应用服务器运行数据负荷较重。大部分计算都在服务器端，增加服务器压力。使用 Ajax 可以改善部分用户体验。

1.2.3 WEB 资源介绍

静态资源：指 web 页面中供人们浏览的数据始终是不变。比如：HTML、CSS、JS、图片、多媒体。

动态资源：指 web 页面中供人们浏览的数据是由程序产生的，不同时间点访问 web 页面看到的内容各不相同。比如：JSP/Servlet、ASP、PHP

1.2.4 WEB 服务器

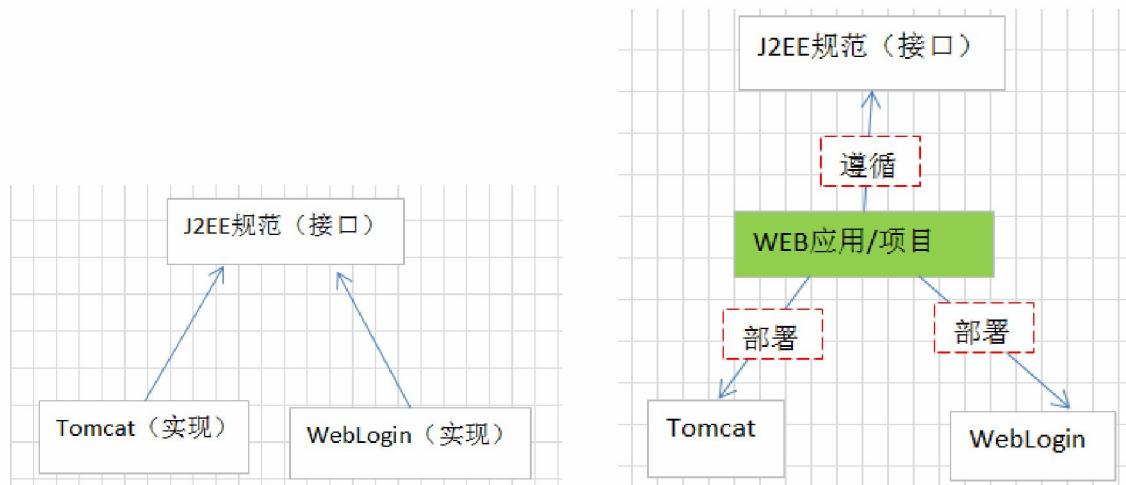
JCP (Java Community Process) Java 技术规范制定与更新的国际组织。主要维护规范包括：J2ME、J2SE、J2EE、XML 等。

J2EE 企业级开发 Java 规范。标准规范包括：servlet、jsp、jdbc、xml、jta、javamail 等。在 java 中规范就是接口。J2EE 又称为 JavaEE。

WEB 服务器对 JavaEE 规范部分或全部支持（实现），也就是 WEB 服务器实现部分或全部接口。

常见的 WEB 服务器：

1. Tomcat：Apache 组织提供一个免费的小型的服务器软件。支持 Servlet 和 JSP 规范。
2. WebLogic：Bea 公司的一个收费的大型的服务器软件，后被 Oracle 收购。支持 EE 的所有的规范。
3. WebSphere：IBM 公司的一个收费的大型的服务器软件，支持 EE 的所有的规范。
4. JBoss：是一个基于 J2EE 的开放源代码的应用服务器。JBoss 是一个管理 EJB 的容器和服务，JBoss 核心服务不包括支持 servlet/JSP 的 WEB 容器，一般与 Tomcat 或 Jetty 绑定使用。



1.2.5 URL 请求路径（待定）

URL (Uniform Resource Locator)，统一资源定位符是对互联网上资源位置的一种表示，互联网上的每个文件都有一个唯一的 URL。

完整格式如下

协议://用户名:密码@域名:端口号/资源位置?参数=值#标志

协议，http、https、ftp 等

用户名:密码，常用于 ftp 访问，路径直接编写账号（一般不写）

域名，域名或 IP 地址，都可以访问 WEB 资源

端口号，程序必须使用端口号，才可以让另一个计算机访问。http 协议的默认端：80

资源位置，用于描述 WEB 资源在服务器上的位置。

参数=值，浏览器和服务器交互传递的数据

#标志，锚点，用于指定页面的某一个位置。

例如：

常见路径

<http://www.itcast.cn:80/subject/javaeezly/index.shtml>

百度搜索“传智播客”

<https://www.baidu.com/s?cl=3&wd=%B4%AB%D6%C7%B2%A5%BF%CD>

使用锚点：

http://baike.baidu.com/link?url=fptQHWW2jmBg04BSDedBAQgF-H_VxNX56edrPCukpK6zcovyAqvHOGPwTdQb136O3VPm5IW39EbzyKpqtQrl0H_#3_1

1.3 Tomcat 介绍

1.3.1 概述

Tomcat 服务器是一个免费的开放源代码的 Web 应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试 JSP 程序的首选。

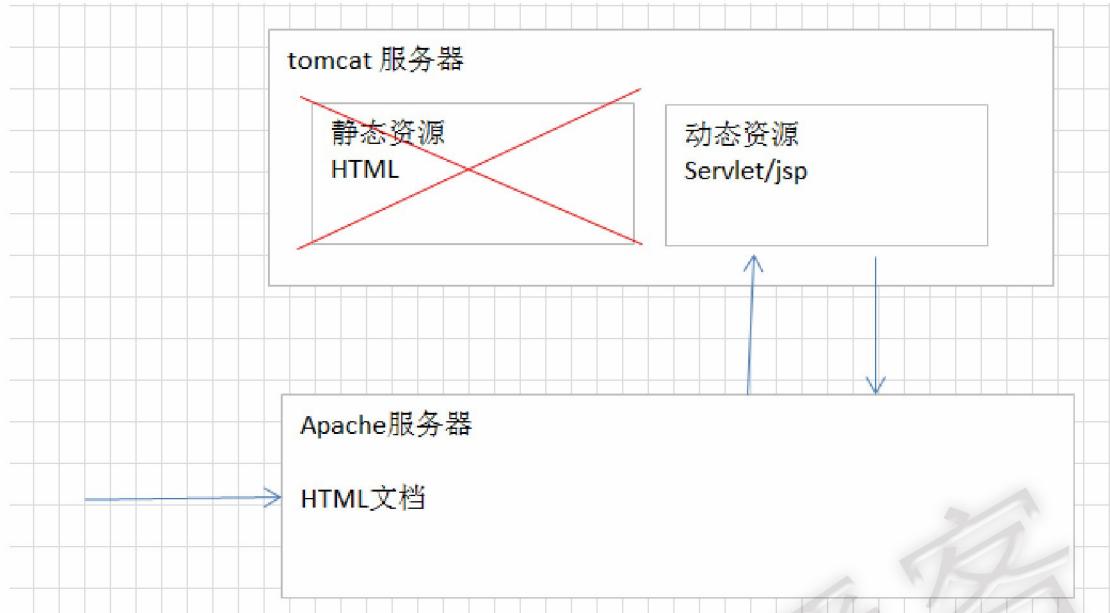
支持 Servlet 和 JSP 规范，且最新的 Servlet 和 JSP 规范总是能在 Tomcat 中得到体现。

Apache 软件基金会有两款常用软件：apache web 服务器 和 tomcat web 服务器。

- apache web 服务器专门处理 HTML 页面的。
- tomcat web 服务器，不仅可以处理 servlet 和 jsp，而且也能处理 html 页面，不过不如 apache web 服务器。
- 在开发中，一般使用 tomcat 处理 servlet 和 jsp，使用 apache 服务器处理 html 页面。及 apache

和 tomcat 被整合在一起使用。

- 学习阶段，我们使用 tomcat 所有的特性



1.3.2 Tomcat 版本（了解）

tomcat 目前存在很多版本，希望大家了解 tomcat 的版本，从而知道自己的创建的项目使用的是几版本规范，不同版本的规范技术可能不同。我们学习的 WEB5.0，Servlet 规范 2.5，tomcat 至少使用 6 版本。

官网地址：<http://tomcat.apache.org/whichversion.html>

Tomcat 版本	Servlet 版本	JSP 版本	EL 版本	JavaEE 版本	JDK 版本
9.0.x	4.0	2.4?	3.1?	?	8 (1.8)
8.0.x	3.1	2.3	3.0	7.0	7 (1.7)
7.0.x	3.0	2.2	2.2	6.0	6 (1.6)
6.0.x	2.5	2.1	N/A	5.0	5 (1.5)

最新版本 myeclipse 可选的 WEB 项目版本。



1.3.3 安装 Tomcat

- 步骤一：下载一个 tomcat 服务器软件.

<http://tomcat.apache.org/download-70.cgi>



- 步骤二：解压下载好的 zip 文件。

将解压后的文件 copy 到一个没有中文和空格的路径下即可。

例如：D:\java\tomcat\apache-tomcat-7.0.68

1.3.4 Tomcat 目录结构

bin: 脚本目录
启动脚本: startup.bat
停止脚本: shutdown.bat

conf: 配置文件目录 (config /configuration)
核心配置文件: server.xml
用户权限配置文件: tomcat-users.xml
所有 web 项目默认配置文件: web.xml

lib: 依赖库, tomcat 和 web 项目中需要使用的 jar 包

logs: 日志文件.
localhost_access_log.*.txt tomcat 记录用户访问信息, 星*表示时间。
例如: localhost_access_log.2016-02-28.txt

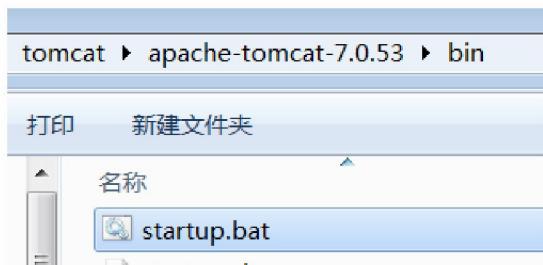
temp: 临时文件目录, 文件夹内内容可以任意删除。

webapps: 默认情况下发布 WEB 项目所存放的目录。

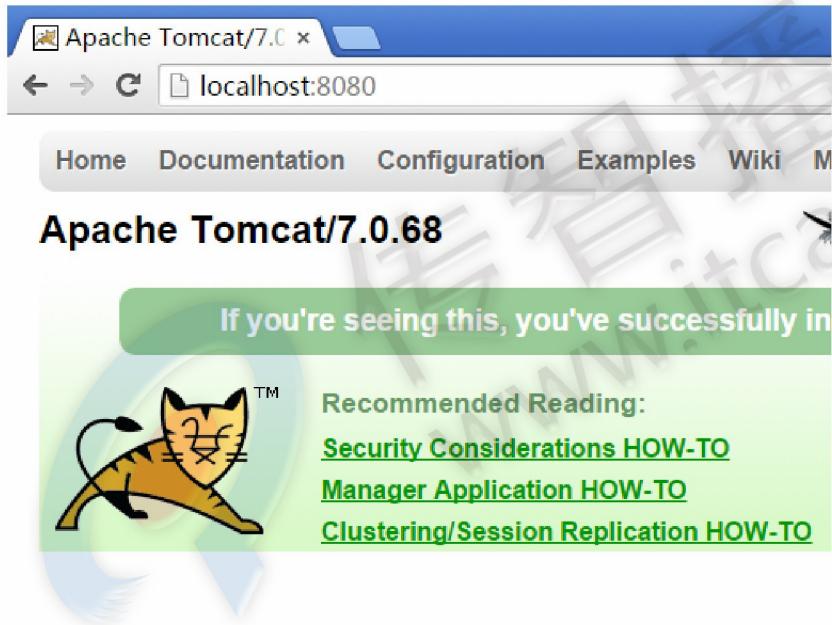
work: tomcat 处理 JSP 的工作目录。

1.3.5 Tomcat 启动和访问

- tomcat 解压目录/bin/startup.bat 双击运行启动 tomcat



- 访问路径: <http://localhost:8080/>



1.3.6 Tomcat 常见问题

1.3.6.1 JAVA_HOME 的配置

如果没有配置 JAVA_HOME 环境变量，在双击“startup.bat”文件运行 tomcat 时，将一闪立即关闭。且必须配置正确，及 JAVA_HOME 指向 JDK 的安装目录



当同一台计算机启动两个 tomcat 时，第二个 tomcat 将会在控制台抛异常，摘要信息如下：

1.3.6.2 端口号冲突

```
严重： Failed to initialize end point associated with ProtocolHandler ["http-bio-8080"]
java.net.BindException: Address already in use: JVM_Bind <null>:8080
...
Caused by: java.net.BindException: Address already in use: JVM_Bind
...
```

控制台将出现大量异常信息，描述的是 3 个端口被占用（8080、8009、8005）

通过 \$JAVA_HOME/conf/server.xml 修改口号。

```
21  L -->
22  日<Server port="8005" shutdown="SHUTDOWN">
23  白  <!-- Security listener. Documentation at /docs/config/listeners.html
    ...
70      <Connector port="8080" protocol="HTTP/1.1"
71          connectionTimeout="20000"
72          redirectPort="8443" />
    ...
91      <!-- Define an AJP 1.3 Connector on port 8009 -->
92      <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />
93  白
```

1.4 web 项目目录结构（重要）

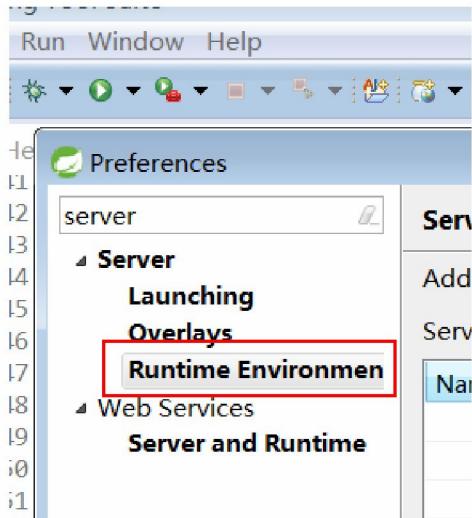
在 JavaEE 规范中，WEB 项目存在一定的目录结构，具体结构如下：

```
项目名称（webapps 文件夹）
|-----静态资源.HTML, CSS, JS
|-----WEB-INF（不能直接通过浏览器进行访问）
    |---web.xml 当前 WEB 项目的核心配置，Servlet2.5 必须有，3.0 可省略。
    |---lib 当前 WEB 项目所需要的第三方的 jar 的存放位置。
    |---classes Java 源码编译后生成 class 文件存放的位置。
```

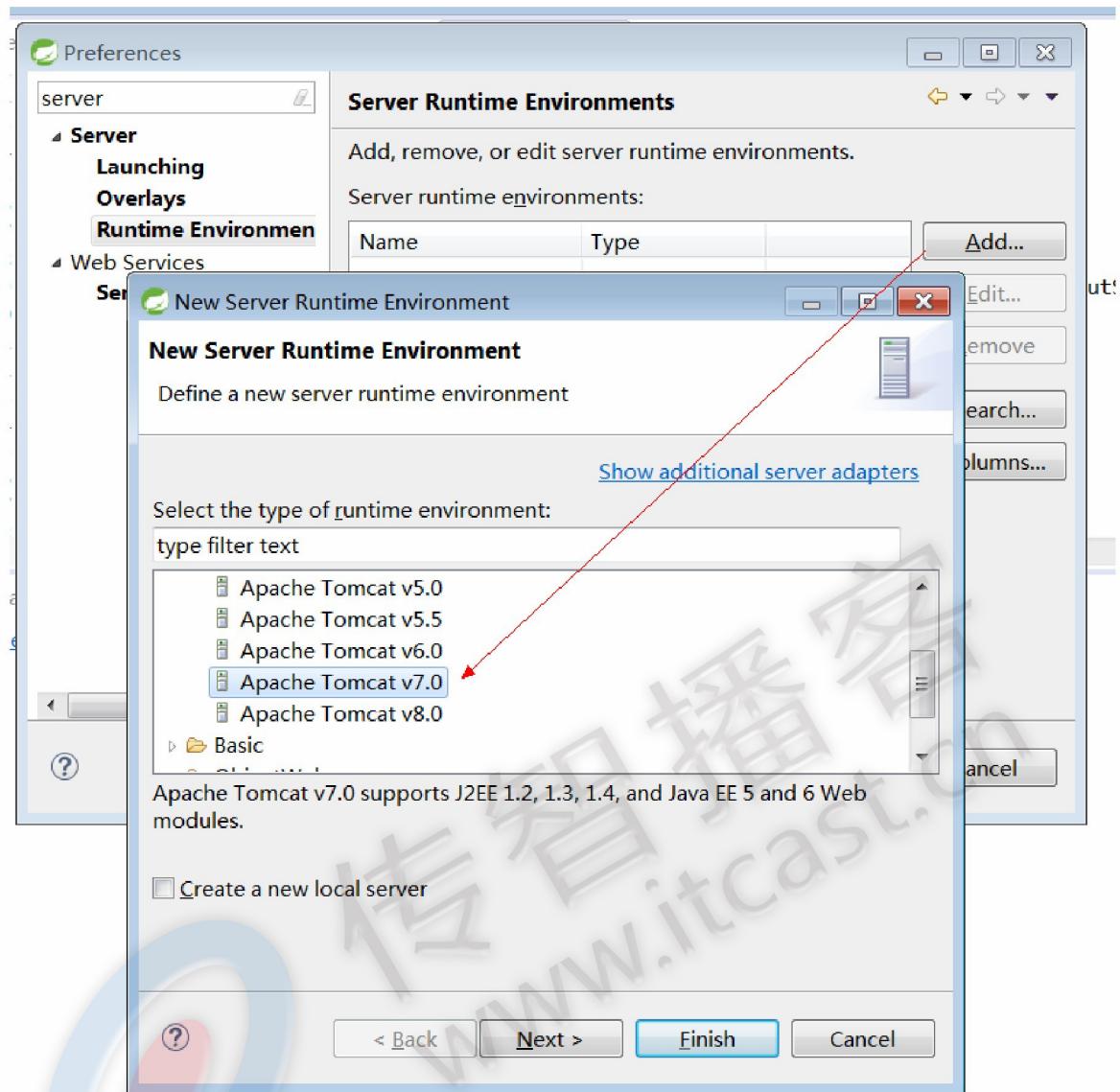
1.5 eclipse 发布 web 项目

1.5.1 配置 Tomcat

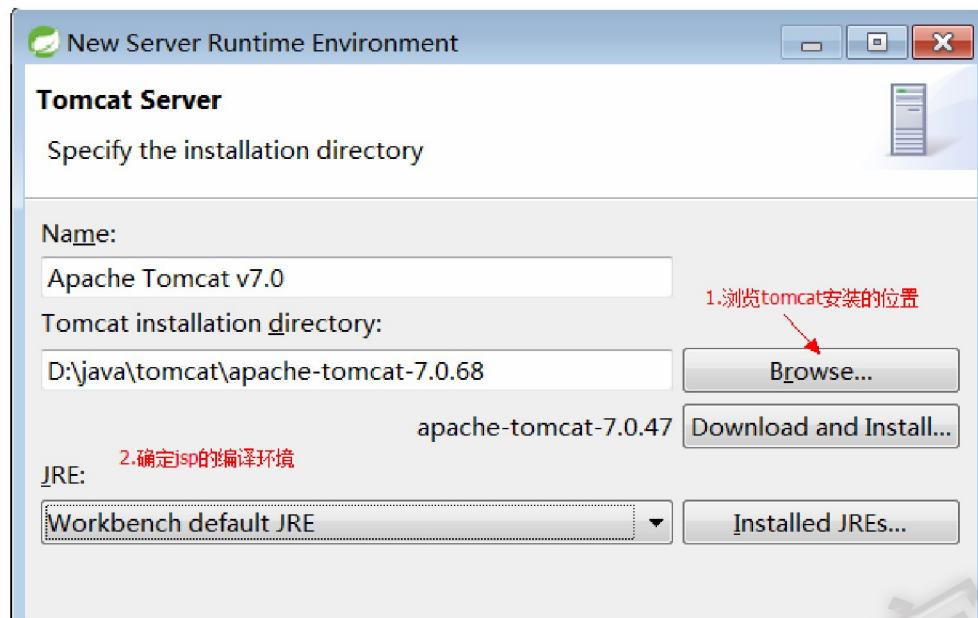
- 步骤 1：获得服务器运行环境配置，Window/Preferences/Server/Runtime Environment



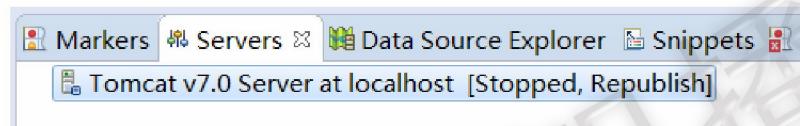
- 步骤 2：添加服务器



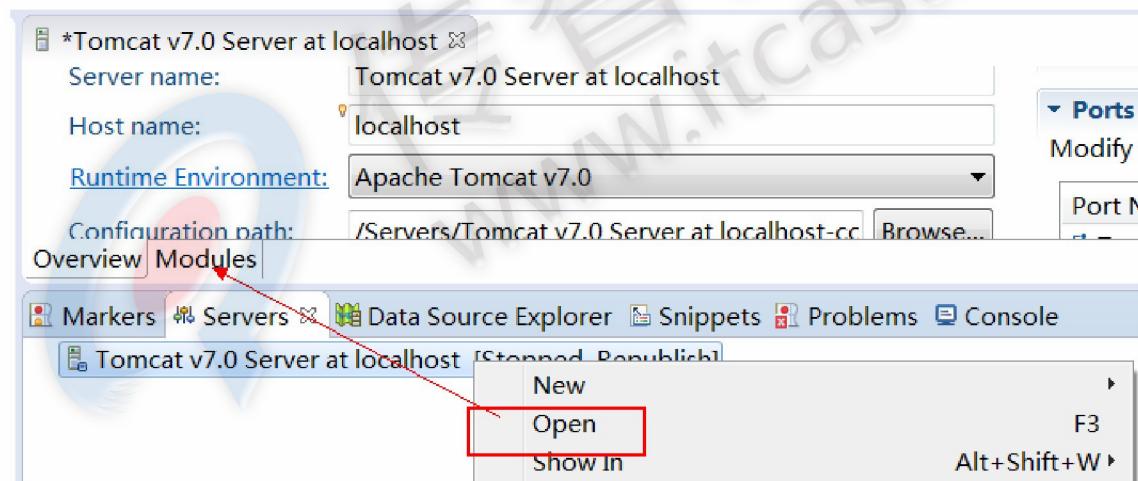
- 步骤 3：选择服务器在硬盘的地址，然后所有的都是确定/Next/Finish



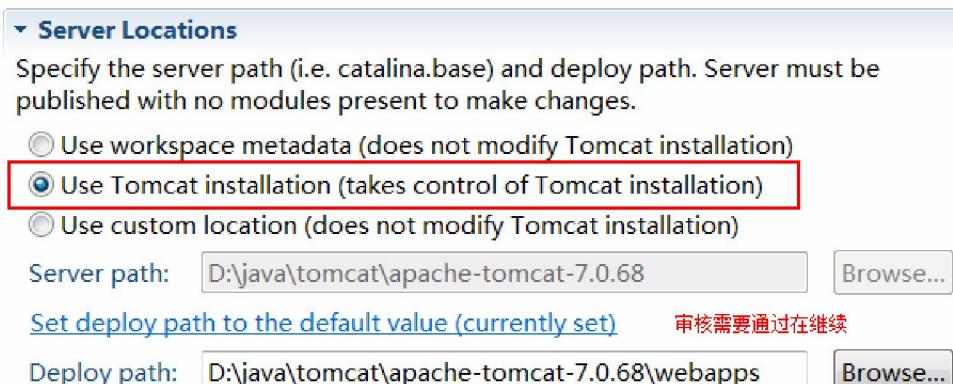
- 步骤 4：完成成功



- 步骤 5：设置发布位置

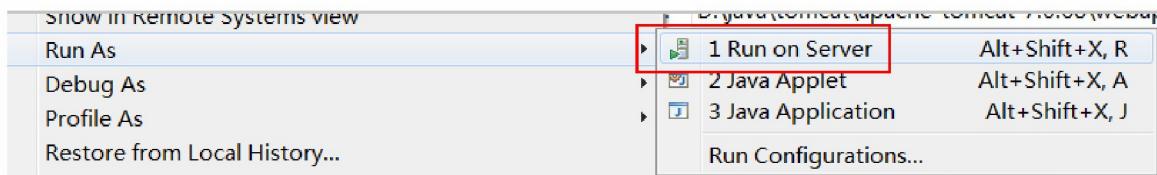


修改 tomcat 发布的位置

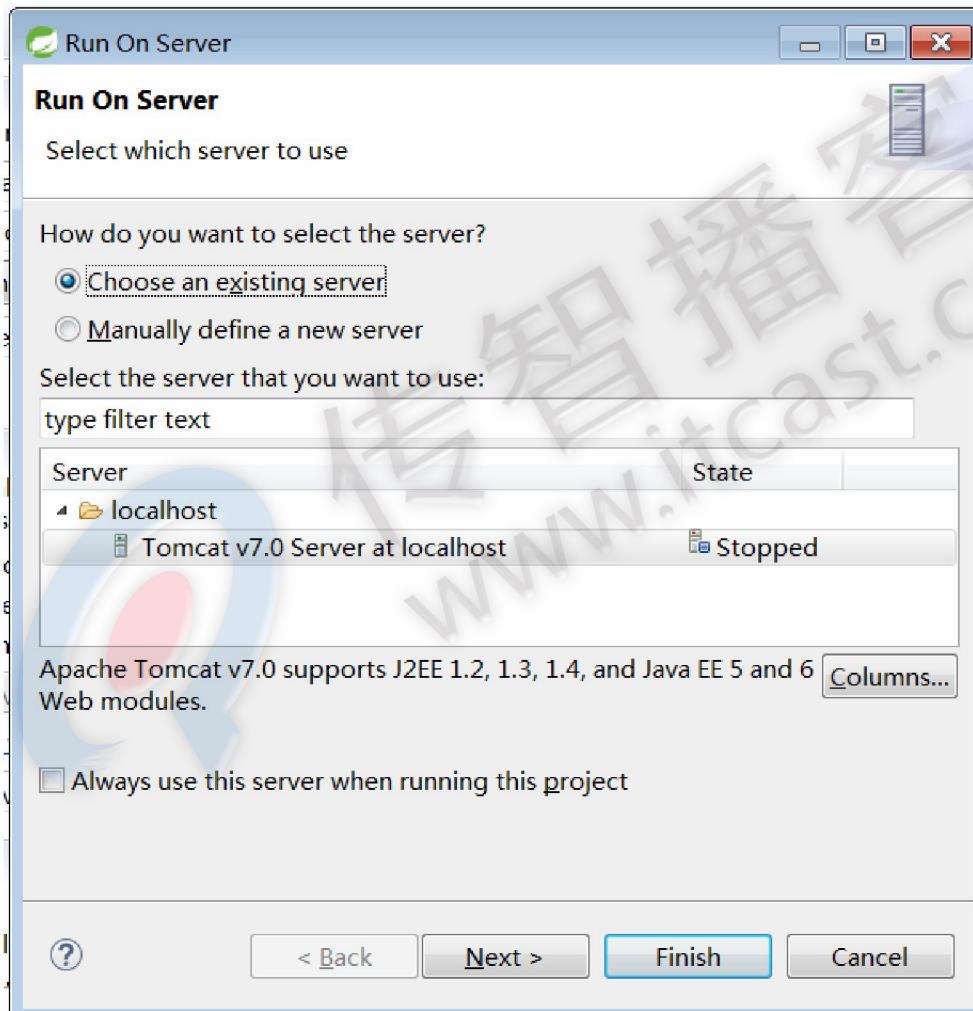


1.5.2 发布 web 项目

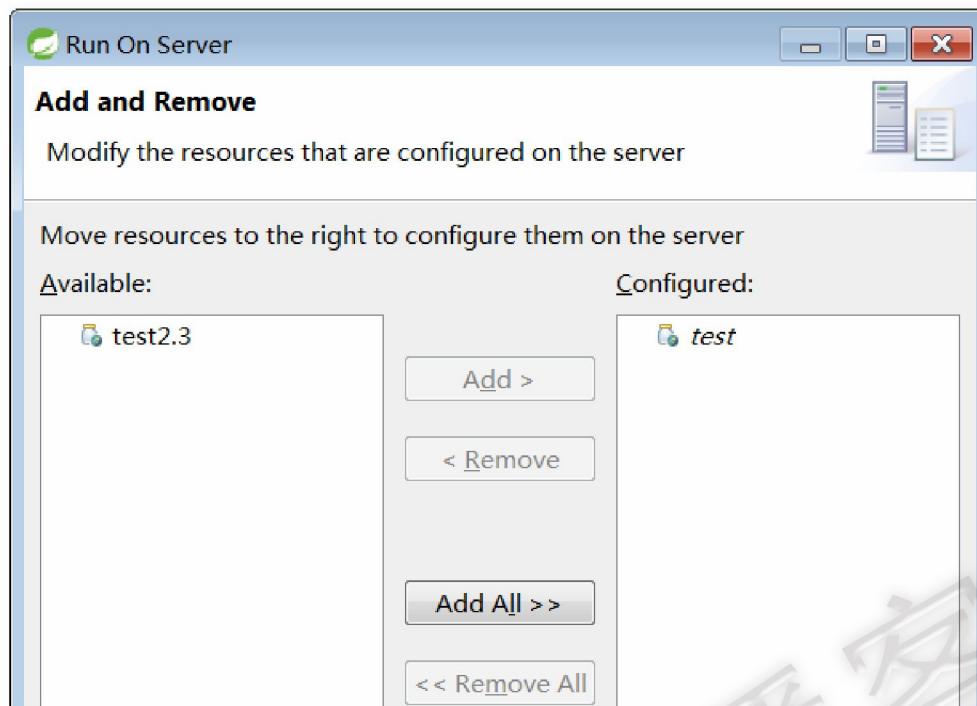
- 步骤 1：项目右键/Run As/Run on Server



- 步骤 2：选择 WEB 服务器执行程序。



- 步骤 3：选择或添加文本框



第2章 总结