# Tomcat的使用（详细流程）

# 1、JavaWeb概念

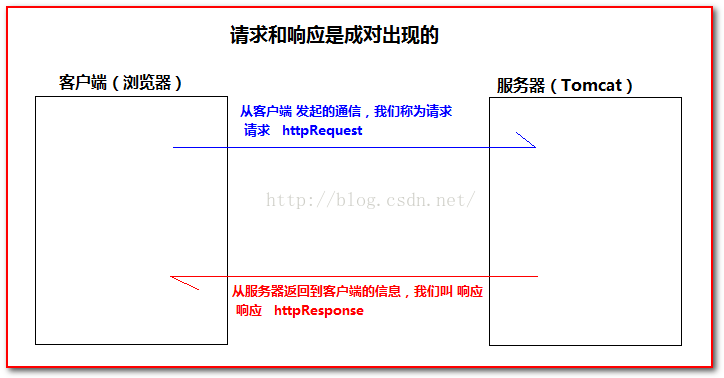
Java web，是用java技术来解决相关web互联网领域的技术的总称。web包括：web服务器和web客户端两部分。

java在最早web客户端的应用有java applet程序，不过这种技术在很久之前就已经被淘汰了。

java在服务器端的应用非常丰富，比如Servlet，jsp和第三方框架等等。java技术对web领域的发展注入了强大的动力  
  
简单的说，就是使用java语言实现浏览器可以访问的程序内容。称之为Java Web。  
  
javaweb开发是基于请求和响应的：

**请求**：浏览器（客户端）向服务器发送信息  
**响应**：服务器向（客户端）浏览器回送信息

请求和响应是成对出现的。

****

# 2、web资源分类：

所谓web资源即放在Internet网上供外界访问的文件或程序，又根据它们呈现的效果及原理不同，将它们划分为静态资源和动态资源。  
  
**静态web资源**：固定不变数据文件（静态网页 HTML、CSS文件、文本、音频、视频）  
**静态web技术**：HTML+CSS+JavaScript  
  
**动态web资源**：一段服务程序，运行后，生成的数据文件

**动态web技术**：servlet，jsp，php， .net ,ruby、python等等

# 3、常见的web服务器

web服务器简介：

Tomcat：由Apache组织提供的一种Web服务器，提供对jsp和Servlet的支持。它是一种轻量级的javaWeb容器（服务器），也是当前应用最广的JavaWeb服务器（免费）。  
Jboss：是一个遵从JavaEE规范的、开放源代码的、纯Java的EJB服务器，它支持所有的JavaEE规范（免费）。  
GlassFish： 由Oracle公司开发的一款JavaWeb服务器，是一款强健的商业服务器，达到产品级质量（应用很少，收费）。  
Resin：是CAUCHO公司的产品，是一个非常流行的应用服务器，对servlet和JSP提供了良好的支持，性能也比较优良，resin自身采用JAVA语言开发（收费，应用比较多）。  
WebLogic：是Oracle公司的产品，是目前应用最广泛的Web服务器，支持JavaEE规范，而且不断的完善以适应新的开发要求，适合大型项目（收费，用的不多，适合大公司）。

## 3.1、Tomcat服务器

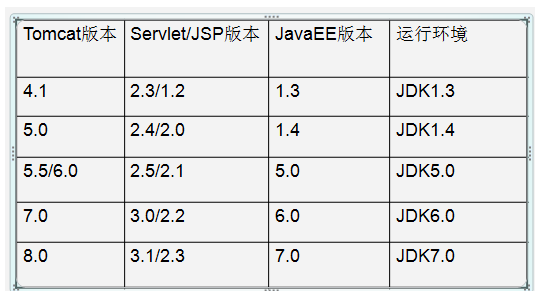
开源小型web服务器 ，完全免费，主要用于中小型web项目，只支持Servlet和JSP 等少量javaee规范（就是JavaWeb编程接口）



## 3.2、tomcat服务器与servlet版本的关系

servlet：sun公司提供的用于开发动态web资源的技术。  
jsp：（java server page），java提供的一门开发web网页的技术。

tomcat软件：java开发的。java软件运行的时候需要jdk。



向下兼容。tomcat7也支持servlet3.0/jsp2.2规范，可以支持javaee6.0当前企业常用的版本 6.\* / 7.\*/8.\*

## 3.3、tomcat下载和安装说明

到http://tomcat.apache.org  下载  
  
**1） Tomcat首页**



**2）Tomcat下载**



**3）下载后的包**

C:\Users\ray\Desktop\20161106225903968.png

4）安装：解压

C:\Users\ray\Desktop\20161106230003259.png

5）tomcat的安装目录介绍：

bin：可以执行文件。  
conf：tomcat服务器的配置文件  
lib：tomcat启动后需要依赖的jar包  
logs：tomcat工作之后的日志文件  
webapps：是tomcat布暑工程的目录。

work：jsp文件在被翻译之后，保存在当前这个目录下，session对象被序列化之后保存的位置

## 3.4、Tomcat服务器启动（\*\*重点）

注意事项：

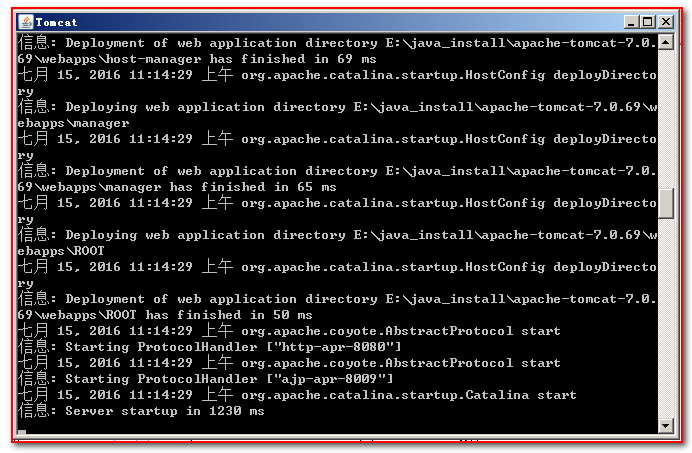
**1、JAVA\_HOME:环境变量。并且配置到jdk的目录，其完整过程如下：**



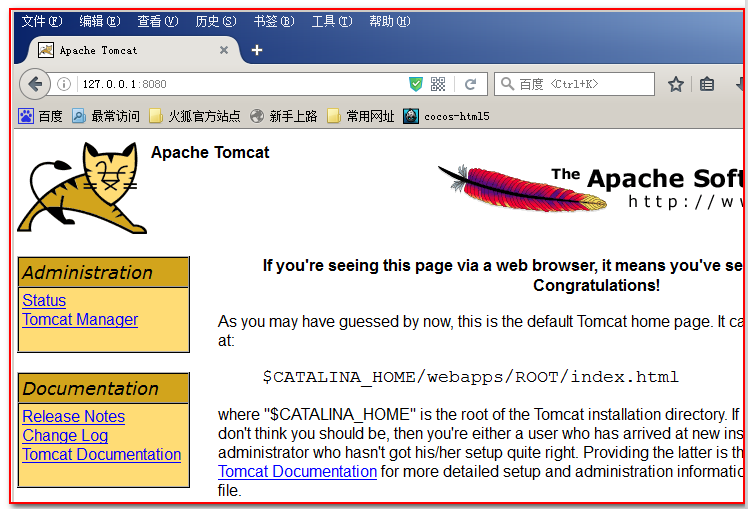
打开控制台（cmd命令打开窗口）。输入java -version测试



2、启动tomcat目录。 tomcat目录/bin/startup.bat(window启动文件) 找到startup.bat 双击运行。会有一个黑窗口，黑窗口不要关闭。（如果关闭，相当于把tomcat停止了。）



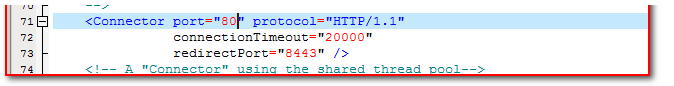
3、在浏览器地址栏中输入：http://localhost:8080 或者 <http://127.0.0.1:8080> localhost ，如果看到如下页面，证明启动成功



## 3.5、配置tomcat的端口（\*\*\*\*重点）

tomcat默认的端口是8080（访问端口）

http的默认端口是80，如果访问的时候输入http://www.baidu.com相当于http://www.baidu.com:80。当真正在项目上线之后，通常采用80，修改方法如下：  
**1）找到tomcat目录/conf/server.xml  
2）修改port的值，将port端口的值修改为80**



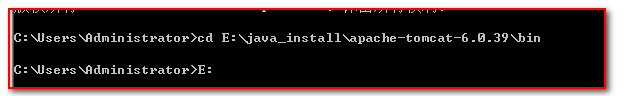
**3）然后在浏览器中输入 http://127.0.0.1:80 或 http://127.0.0.1 访问测试**



访问成功！！！

## 3.6、catalina run 启动Tomcat

Tomcat启动，还有一种启动的方法就是在命令行中，先把你的当前目录切换 到你tomcat目录\bin目录下，如下是我的位置



再执行catalina run 这个命令启动Tomcat。这个命令有什么好处。当Tomcat启动失败的时候，会有一闪而过的情况，

当我们使用catalina run 这个命令启动Tomcat的时候，哪怕有错误，我们也可以清楚的看到tomcat失败的原因。不会一闪而过。

## 3.7、tomcat关闭

有三种方法。

第一种：Ctrl+C键 关闭Tomcat服务器  
第二种：点击Tomcat窗口的右上角关闭按钮 （暴力停止服务器）  
第三种：找到tomcat目录/bin/shutdown.bat文件，双击执行关闭Tomcat。

# 4、常用的布署工程到Tomcat中的两种方式

把我们自己书写的html，servlet这些信息，部署到tomcat的方式。

## 4.1、第一种方法：在tomcat目录/conf/server.xml 配置---了解

在conf/server.xml文件的host元素中配置，例如：  
在host标签内书写如下内容

<Context  path="/atguigu"  docBase="D:\atguigu"/>  
<Context  path=”浏览器要访问的目录---虚拟目录”  docBase=”网站所在磁盘目录”/>

配置好之后，要重启服务器。

缺点（Tomcat7.0之后）：如果配置错误：tomcat会启动失败。（如果tomcat里面存放的其他的网站），其他网站也会停机。

## 4.2、第二种方式：将网站目录复制到tomcat/webapps目录（常用，必须掌握）

有一个网站（一个文件夹），把文件夹复制到tomcat的webapps目录下。

文件夹的名字，就是网站或者工程的访问目录.相当于之前配置 <Context path=”” 的配置

## 4.3、把网站目录压缩成war包部署到tomcat中

war包：就是一个压缩文件 zip格式的压缩文件。 只不过扩展名不是.zip 而是.war  
  
把我们的项目进行压缩zip，改成war，把war文件拷贝到tomcat/webapps目录下

步骤1、把文件夹中的内容压缩成zip的格式，点击一个要部署的文件夹下面，全选 然后压缩  
步骤2、修改文件的后缀名为.war  
步骤3.把war文件复制到webapps目录下。tomcat会自己把war的文件进行解压

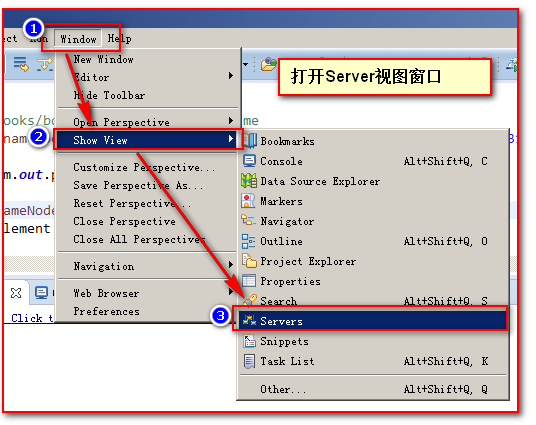
## 4.4、webapps目录下/ROOT工程的访问

当我们在浏览器中直接输入http://ip地址:端口号   那么 默认访问的是Tomcat目录/webapps/ROOT目录  
如果webapps下面有一个ROOT的项目。那么在访问的时候，直接可以省略项目的名字/ 表示找到root目录

# 5、整合Tomcat和Eclipse开发工具中（\*\*\*常用必须掌握）

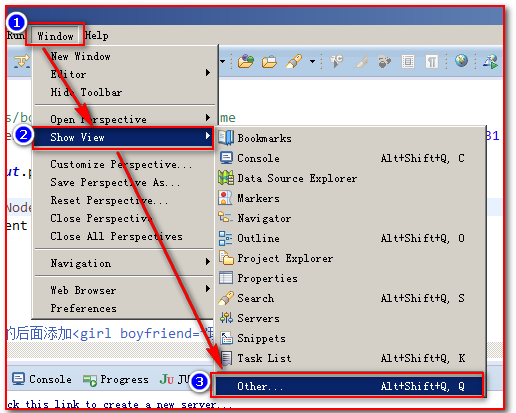
## 5.1、打开Eclipse的Server视图窗口

**第一种情况，直接打开Servers窗口**

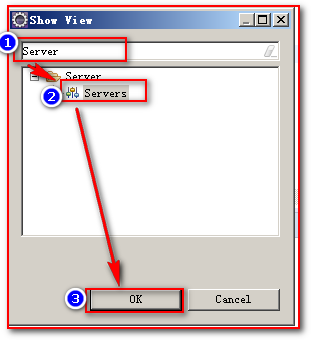


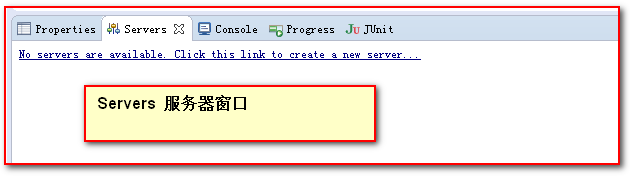
**第二种情况，搜索Servers窗口打开**

图一，打开总的eclipse视图



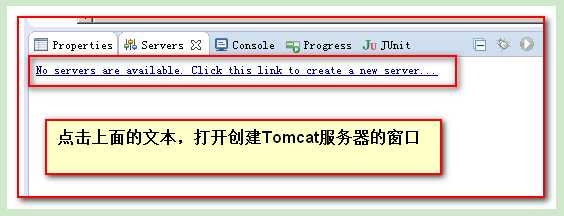
图二：输入Server过滤出服务器窗口选项



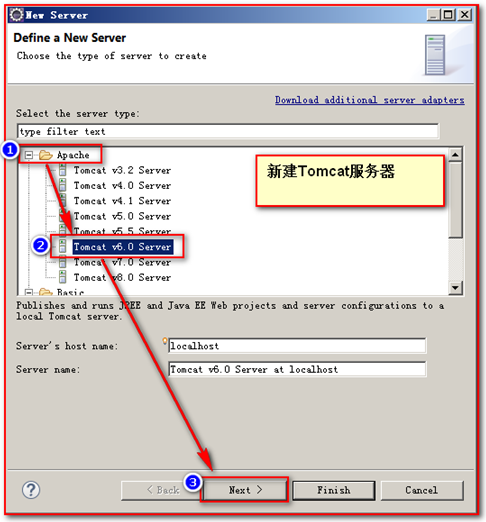
Servers服务器窗口已成功打开，如下图：  


## 5.2、创建Tomcat 服务器

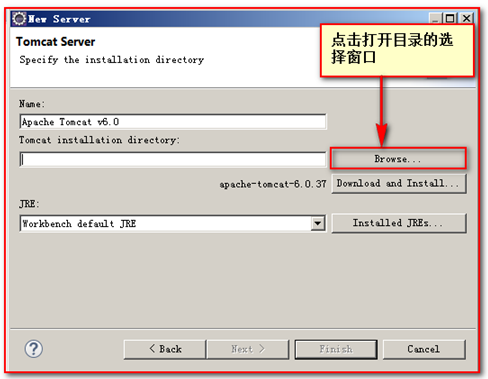
**1）在Servers窗口中，点击 创建 server 的文字提示连接。如下图：**



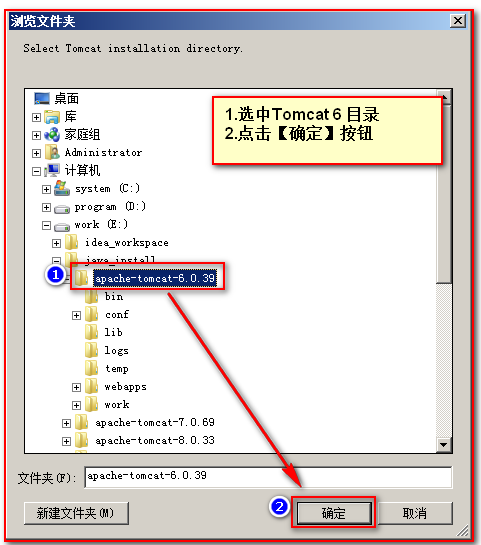
**2）创建一个新的Tomcat服务器实例**



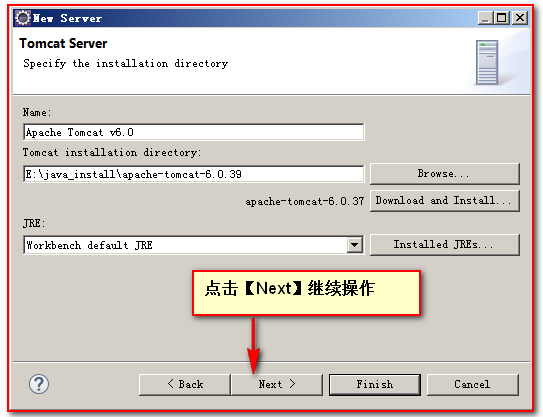
**3）点击Browse按钮，打开目录选择窗口。选择Tomcat 解压目录**



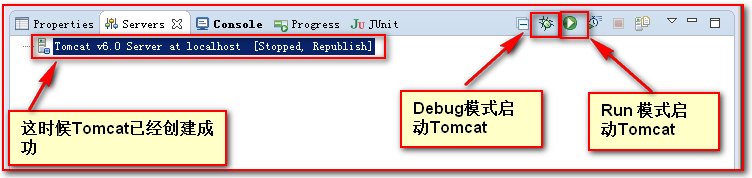
**4）选择Tomcat目录，然后点击确定按钮**



**5）Tomcat目录选择好之后，点击 【Next】按钮继续操作**

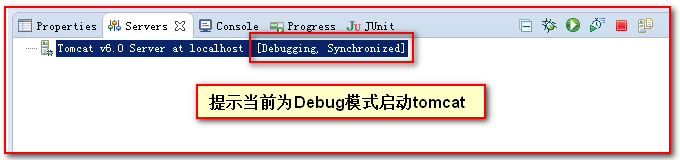


**6）点击 【Finish】按钮结束操作  
  
7）Tomcat 服务器创建成功！！！**

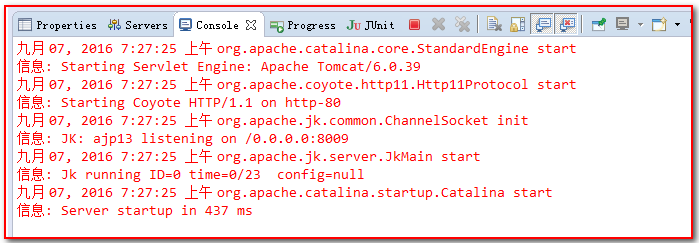


## 5.3、启动Eclipse中的Tomcat服务器

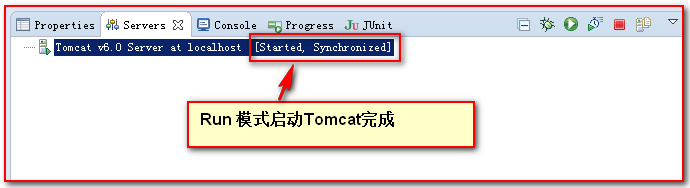
**1）Debug模式启动Tomcat服务器**  
  
提示当前为Debug模式启动！！！



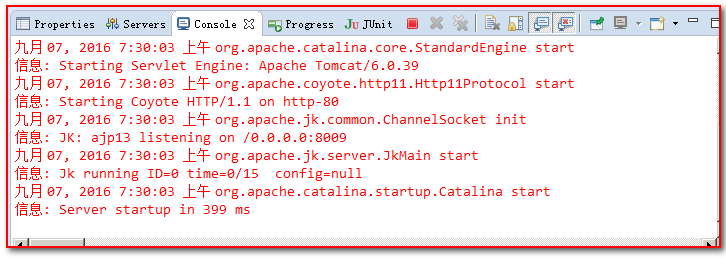
Tomcat启动成功的控制台提示！！！



**2）Run模式启动Tomcat服务器**  
  
Run模式启动显示



Tomcat启动成功的控制台提示！！！

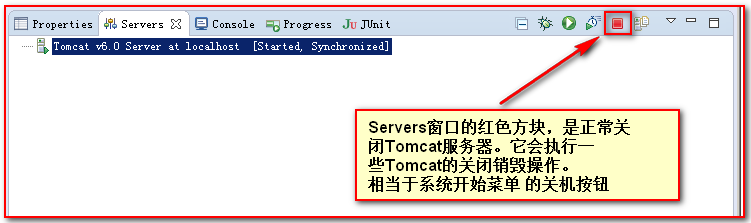


## 5.4、停止 Eclipse 中的Tomcat 服务器

暴力停止 Tomcat （相当于电脑被拔掉电源一样。没有执行关机的准备操作。）

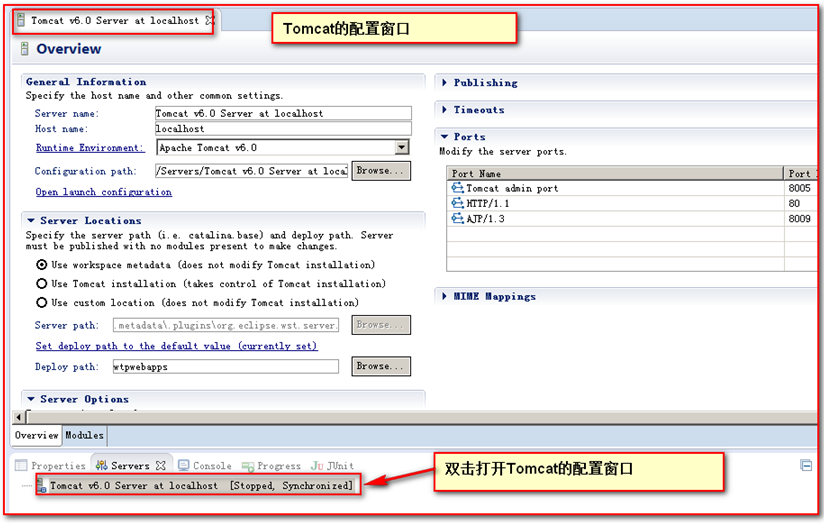


正常停止 Tomcat （相当于点击操作系统中的关机按钮，执行关机保存操作，然后关机）

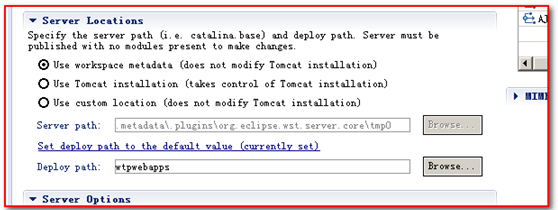


## 5.5、配置Eclipse 中的Tomcat 布暑的Web工程路径

**1）打开Servers窗口，双击Tomcat v6.0 Server 服务器打开 Tomcat的配置窗口**



这里是Eclipse把工程发布后的三种不同的选项。



**2）Tomcat 位置-选项介绍说明：**

2.1、User workspance metadata (does not modify Tomcat installation)

将在eclipse的工作区间目录下eclipse的工作空间目录\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\ 有一份tomcat的拷贝所有布暑的web工程都会布暑到eclipse的工作空间目录\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps目录中  
2.2、 User Tomcat installation (takes control of Tomcat installation)

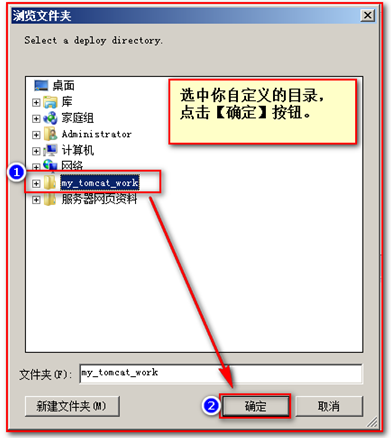
在原Tomcat目录下做更改操作所有布暑的web工程都会发布到Tomcat目录下的\wtpwebapps目录中  
2.3、 User custom location ( does not modify Tomcat installtion )

自定义一个目录去布暑Web工程

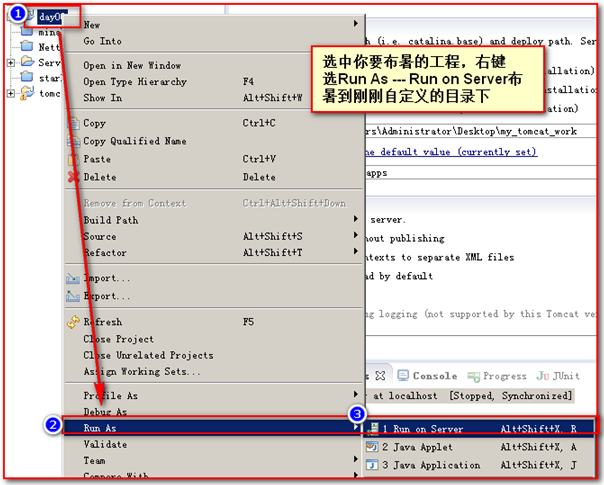
比如操作如下：  
**一：选择自定义的布暑目录**



**二：选择你自定义的目录，然后点击确定。之后Tomcat的一些目录会被拷贝过来。布暑的工程也在这个目录下的wtpwebapps目录下**

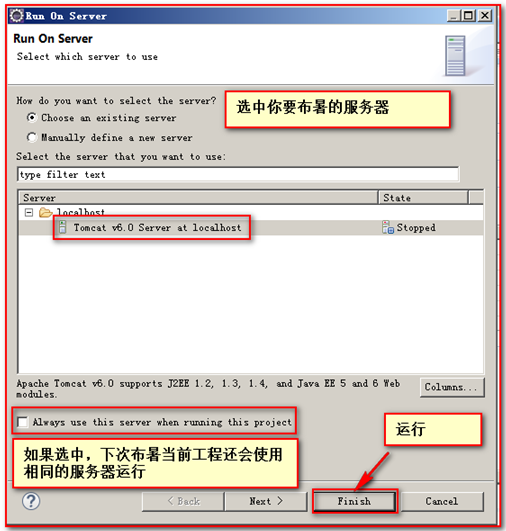


**三：选中你要布暑的工程，右键选择菜单 Run As --->>> Run on Server 将工程布暑到刚刚自定义的目录下。**

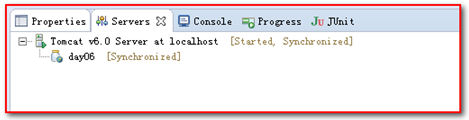


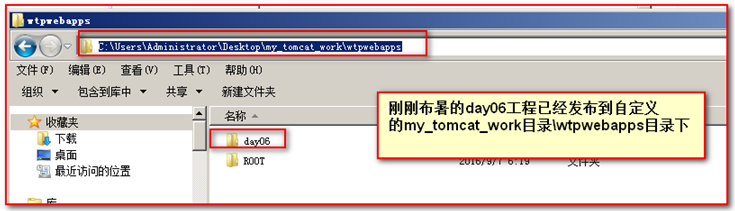
**四：选择你当前的web工程需要使用的哪个服务器运行。**

如果勾选中 Always use this server when running this project 表示下次运行服务器的时候默认使用当勾时时候选择的服务器运行。

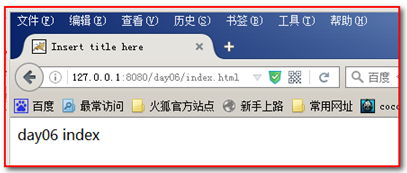


**五：等待Tomcat启动，web工程布暑成功后。查看**





**六：在浏览器中输入http://127.0.0.1:8080/day06/index.html 测试查看**



# 6 HTTP协议介绍

## 6.1、 HTTP协议

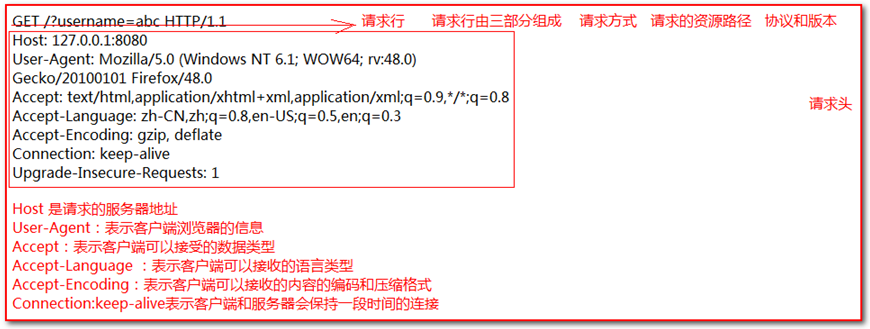
1.HTTP（hypertext transport protocol），即超文本传输协议。这个协议详细规定了浏览器和万维网服务器之间互相通信的规则。  
2.客户端与服务端通信时传输的内容我们称之为报文。  
3.HTTP就是一个通信规则，这个规则规定了客户端发送给服务器的报文格式，也规定了服务器发送给客户端的报文格式。实际我们要学习的就是这两种报文。客户端发送给服务器的称为”请求报文“，服务器发送给客户端的称为”响应报文“。  
  
大白话说，什么是协议。是双方相互约定好的规则；比如：租房协议：租房协议就是租客和房东之间相互约定好的租房规则 

## 6.2、请求的协议格式

请求的HTTP协议格式：

请求行     
请求头  
空行  
请求体

GET请求协议格式  （get请求没有请求体）



POST请求协议格式



## 6.3、常见请求头的说明



GET /Hello/index.jsp HTTP/1.1：GET请求，请求服务器资源的路径 Hello/index.jsp，  协议为http   版本为1.1；  
Host:localhost：请求的主机名为localhost；  
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0…：与浏览器和OS相关的信息。有些网站会显示用户的系统版本和浏览器版本信息，这都是通过获取User-Agent头信息而来的；  
Accept: \*/\*：告诉服务器，当前客户端可以接收的数据类型， \*/\*，就表示什么都可以接收；  
Accept-Language: zh-CN：当前客户端支持的语言，可以在浏览器的工具选项中找到语言相关信息；  
Accept-Encoding: gzip, deflate：支持的压缩格式。数据在网络上传递时，可能服务器会把数据压缩后再发送；  
Connection: keep-alive：客户端支持的链接方式，保持一段时间链接，默认为3000ms；

## 6.4、get请求和post请求都分别是哪些？

GET请求 

1）、输入浏览器地址栏输入地址，直接按回车    
2）、点击<a>超链接     
3）、GET请求 表单提交    
4）、script src=””，引入外部文件   
5）、img src=”路径”,引入图片  
6）、引入外部css。。。

POST请求

1）只有表单提交的时候method=post,提交表单就是发post请求

## 6.5、响应的协议格式

响应的HTTP协议格式

响应首行  
响应头信息  
空行  
响应体

## 6.6、常见的响应码

响应码对浏览器来说很重要，它告诉浏览器响应的结果；  
  
200：请求成功，浏览器会把响应体内容（通常是html）显示在浏览器中；  
404：请求的资源没有找到，说明客户端错误的请求了不存在的资源；  
500：请求资源找到了，但服务器内部出现了错误；  
302：请求重定向，当响应码为302时，表示服务器要求浏览器重新再发一个请求，服务器会发送一个响应头Location，它指定了新请求的URL地址；

# 7、servlet（重点\*\*\*\*\*）

## 7.1、servlet简介

servlet 是运行在 Web 服务器中的小型 Java 程序。servlet 通常通过 HTTP（超文本传输协议）接收和响应来自 Web 客户端的请求。   
要实现此接口，可以编写一个扩展 javax.servlet.GenericServlet 的一般 servlet，或者编写一个扩展 javax.servlet.http.HttpServlet 的 HTTP servlet。   
此接口定义了初始化 servlet 的方法、为请求提供服务的方法和从服务器移除 servlet 的方法。这些方法称为生命周期方法，它们是按以下顺序调用的： 

1.构造 servlet，然后使用 init 方法将其初始化。   
2.处理来自客户端的对 service 方法的所有调用。   
3.从服务中取出 servlet，然后使用 destroy 方法销毁它，最后进行垃圾回收并终止它。

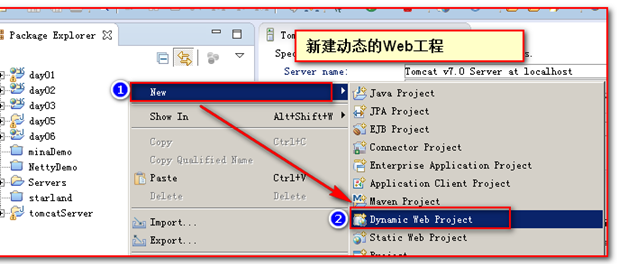
Servlet：

1、接受浏览器发送过来的消息。  
2、给浏览器返回消息。浏览器认识html。可以动态去输出html

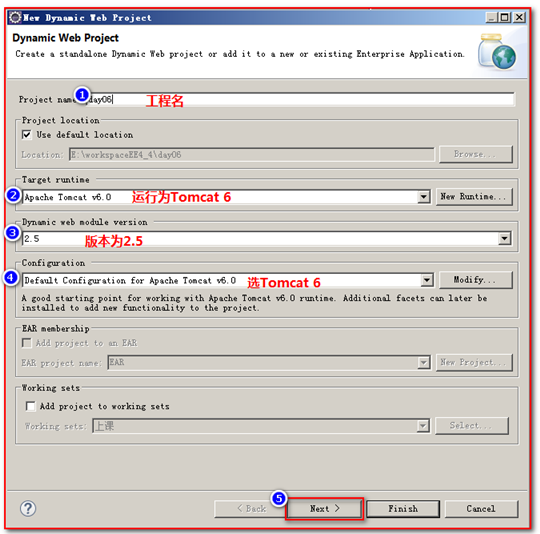
## 7.2、servlet快速入门

### 7.2.1、如何创建动态的Web工程

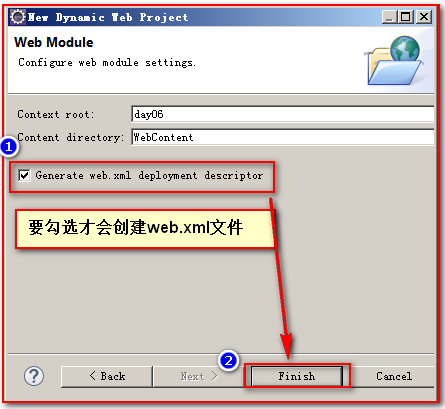
**1）先创建 动态的Web工程**



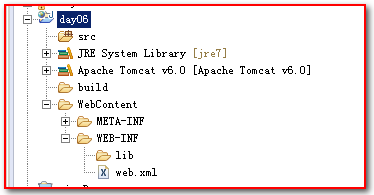
**2)  配置工程的选项**



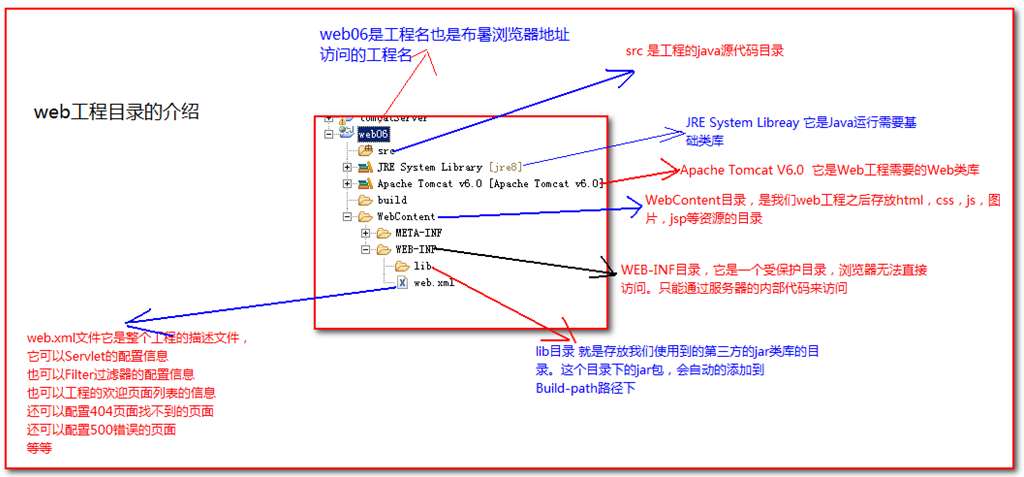
**3）勾选生成web.xml配置文件**

****

**4）动态web工程创建完成！！！**

****

**5）web工程的介绍和说明**

****

### 7.2.2、手动编写servlet实现

写servlet做两件事

1、实现servlet接口。 由sun公司定义的一个接口。(定义一个规范)  
2、把类部署到web服务器中（tomcat）。

sun公司定义一个servlet的规范。定义了servlet应该有哪些方法，以及方法需要的参数。

**1、实现servlet接口（javax.servlet.Servlet）  
2、重写service方法(service方法每次请求都会调用一次)**

1. package com.atguigu.web;
3. import java.io.IOException;
5. import javax.servlet.Servlet;
6. import javax.servlet.ServletConfig;
7. import javax.servlet.ServletException;
8. import javax.servlet.ServletRequest;
9. import javax.servlet.ServletResponse;
11. public class Hello implements Servlet{
13. @Override
14. public void destroy() {
15. *// TODO Auto-generated method stub*
16. System.out.println("Servlet销毁了！");
17. }
19. @Override
20. public ServletConfig getServletConfig() {
21. *// TODO Auto-generated method stub*
22. return null;
23. }

26. @Override
27. public String getServletInfo() {
28. *// TODO Auto-generated method stub*
29. return null;
30. }

33. @Override
34. public void init(ServletConfig arg0) throws ServletException {
35. *// TODO Auto-generated method stub*
36. System.out.println("ServerConfig 初始化了");
37. }

40. @Override
41. public void service(ServletRequest arg0, ServletResponse arg1)
42. throws ServletException, IOException {
43. *// TODO Auto-generated method stub*
44. System.out.println("hello servlet service方法被调用");
45. }
46. }

当浏览器输入地址，访问servlet的时候，servlet会执行servcie方法。

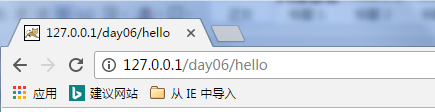
**3、在WebContent/WEB-INF/web.xml中配置servlet的访问路径 。浏览器访问servlet的路径**

web.xml（新建web工程的时候，eclipse自动创建出来的）的位置：  
  
在web.xml的根标签下，直接书写如下内容。

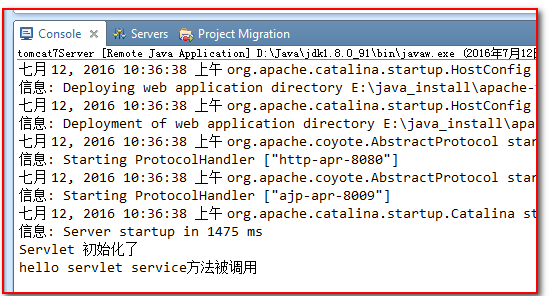
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd" id="WebApp\_ID" version="2.5">
3. <display-name>day06</display-name>
5. <servlet>
6. <servlet-name>Hello</servlet-name>
7. <servlet-class>com.atguigu.web.Hello</servlet-class>
8. </servlet>
9. <servlet-mapping>
10. <servlet-name>Hello</servlet-name>
11. <url-pattern>/hello</url-pattern>
12. </servlet-mapping>


16. <welcome-file-list>
17. <welcome-file>index.html</welcome-file>
18. <welcome-file>index.htm</welcome-file>
19. <welcome-file>index.jsp</welcome-file>
20. <welcome-file>default.html</welcome-file>
21. <welcome-file>default.htm</welcome-file>
22. <welcome-file>default.jsp</welcome-file>
23. </welcome-file-list>
25. </web-app>

**4、把项目部署到tomcat中，去启动tomcat。在地址栏中输入信息，访问servlet**

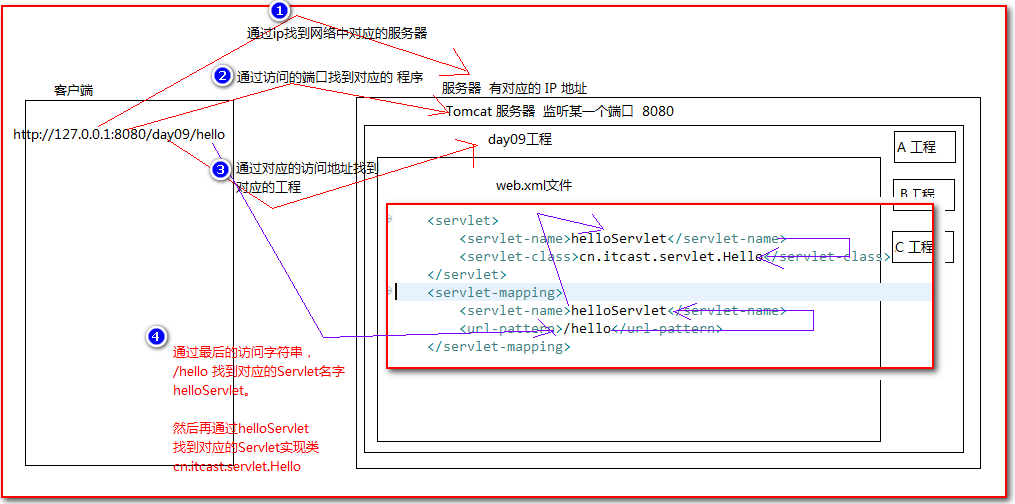
****

控制台打印：



### 7.2.3、访问servlet的细节

浏览器地址栏中输入：http://localhost:8080/day06/hello  
访问过程分析：



### 7.2.4、servlet生命周期

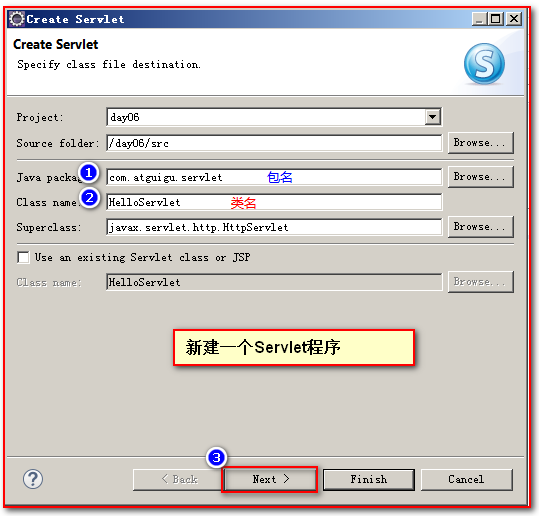
Servlet的生命周期

1.调用 init 方法 初始化Servlet  
2.调用 Servlet中的service方法 处理请求操作  
3.调用 destory方法 执行Servlet销毁的操作

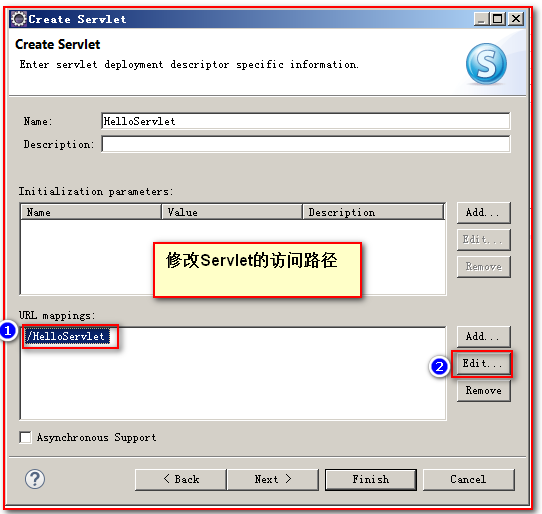
**init方法**：当服务器创建一个serlvet的时候，会去调用init方法。当我们第一次去访问一个servlet的时候，会去创建这个servlet对象。并且只会创建一次。如果配置了load-on-startup 表示服务器启动的时候就创建servlet实例。  
**service方法**：客户端每一次请求，tomcat都会去调用servcie方法。处理用户的请求。并且给其响应。每一次请求都会调用servcie方法。  
**estroy 方法**：当服务器销毁一个servlet的时候，会调用里面的destory方法。当我们的web服务器，正常关闭的时候，会去调用destroy方法。否则不会调用destroy的方法。

## 7.3、使用Eclipse创建Servlet程序（重点\*\*\*\*\*）

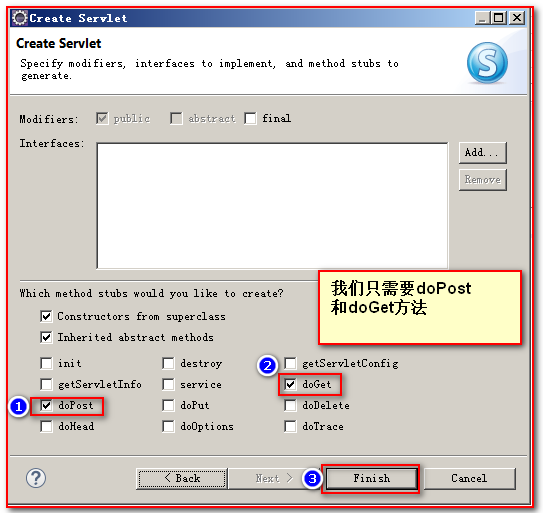
**1）通过Eclipse自动新建一个Servlet程序**



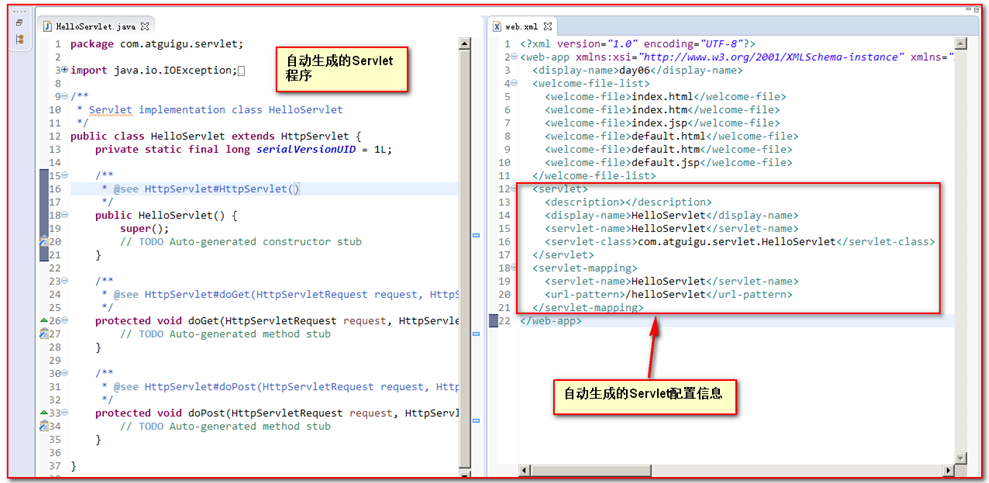
**2）修改Servlet的访问url地址**



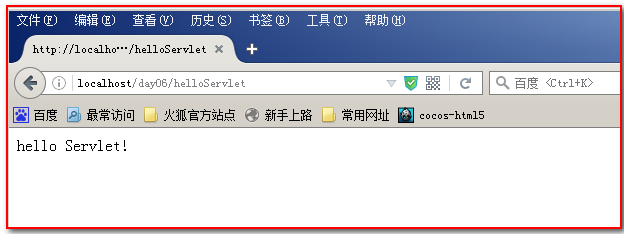
**3）勾选需要生成的Servlet方法**



**4) 查看自动生成的结果内容！！！**



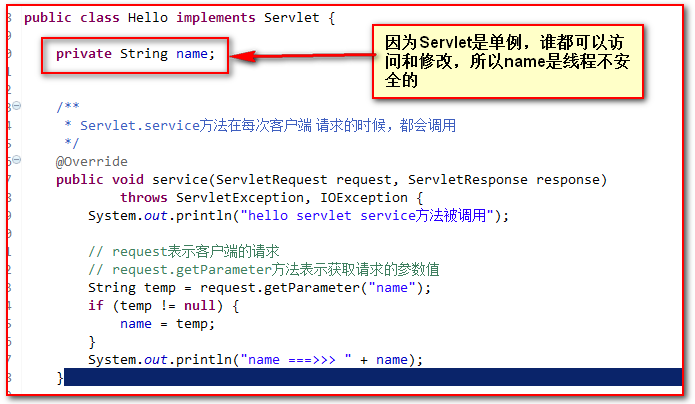
**5) 在浏览器中输出http://127.0.0.1:80/day06/helloServlet 访问测试**



访问成功

## 7.4、Servlet是单例的。Servlet中的变量，它有线程安全问题。

### 7.4.1、全局变量，数据不安全。



### 7.4.2、方法内的局部变量，数据安全

