# WEB开发概述

## WEB资源介绍

WEB：网页

我们在浏览器上能够查看的一切信息，都称之为**WEB资源**。

作用：用来展示信息

分类：

* 静态WEB资源：代码不变，展示内容就不变。比如：HTML、CSS、JS、图片、声音、视频等。

例如：<img src=”001.jpg”/>，代码不换，效果就不会换

* 动态WEB资源：展示内容的改变，由程序控制。比如：JSP/Servlet、ASP、PHP、Ruby、Python等

例如：网站有统计计数器。

15:00登录网站，显示在线会员10人

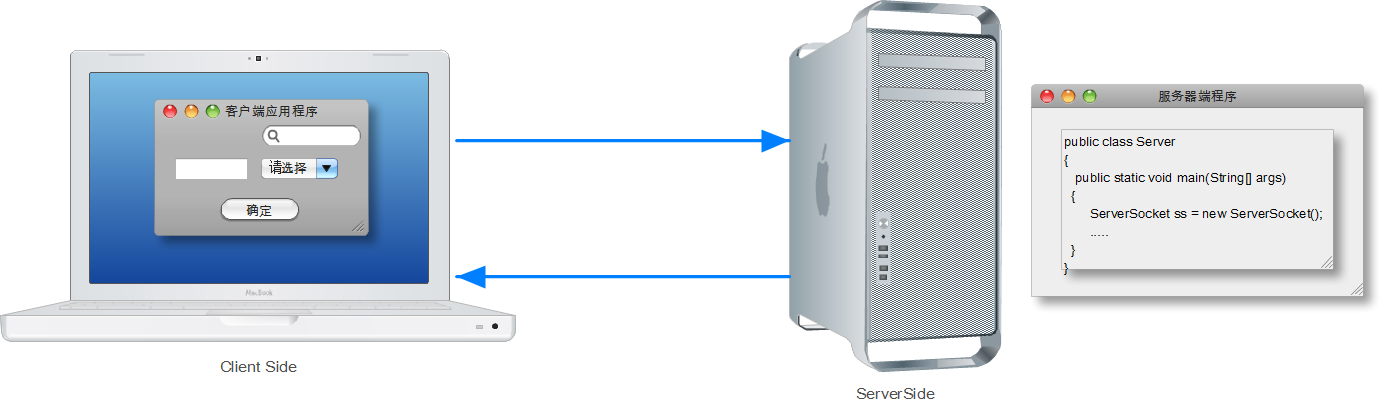
17:00登录网站，显示在线会员100人。

这种效果展示由程序控制

## 1.2、软件架构

目前市场上有两种流行的软件架构：

* **C/S架构**：Client/Server 客户端/服务器。要求客户端用户在 电脑安装一个客户端程序。
  + 常见应用：QQ，欢乐斗地主，消消乐，王者荣耀 等
  + 优点：（体验好、效果炫、服务器压力小）
    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：（客户端程序上）
    1. 占用硬盘空间
    2. 维护麻烦
    3. 安装使用依赖其他条件



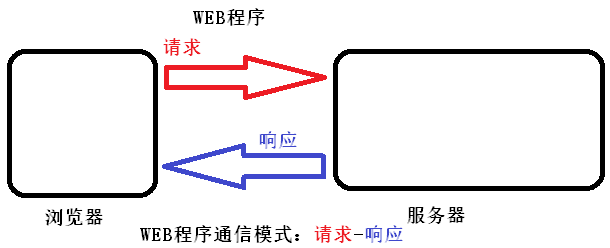
* **B/S架构**：Browser/Server 浏览器/服务器 。通过浏览器与服务器交互，不需要安装其他程序
  + 常见应用：网银系统，淘宝，京东 12306 等
  + 优点：（没有客户端，直接使用浏览器即可）
    1. 维护和升级简单，无缝升级。
    2. 不用必须安装程序，操作系统内置了浏览器。大大减少了使用成本
  + 缺点：（效果一般、体验一般、服务器压力极大）
    1. 动画效果受浏览器限制
    2. 对信息安全控制较差。例如：网银就需要使用U盾，在浏览器端加密。
    3. 应用服务器运行数据负荷较重。大部分计算都在服务器端，增加服务器压力。使用Ajax可以改善部分用户体验。



## WEB通信模式(重点)

客户端：浏览器 乞丐

服务端：tomcat（服务器） 皇帝



1、浏览器需要查看服务器某个资源，就向服务器发送请求；

2、服务器接收到浏览器请求，处理请求；

3、服务器处理完毕，将处理结果响应回浏览器

4、浏览器根据响应，展示资源信息

注： **必须先浏览器的请求， 才会有服务器的响应**

**请求和响应必须成对出现**

请求响应模式----相当于一问一答的模式

## 1.4、URL请求路径

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。

用于标识互联网上某个资源的具体位置，相当于内网路径中的绝对路径

完整格式如下

**协议://域名:端口号/资源位置?参数=值**

协议，http、https、ftp等 【必须写】

域名，域名或IP地址，都可以访问WEB资源【必须写】

端口号，程序必须使用端口号，才可以让另一个计算机访问。8080

资源位置，用于描述WEB资源再服务器上的位置。

参数=值，浏览器和服务器交互传递的数据

例如：

常见路径

https://www.taobao.com/?spm=a2e15.8261149.1581860521.1.5c6e29b4nROeKy

百度搜索“中公教育”

http://bj.offcn.com/

## 1.5、WEB服务器软件

### 1.5.1 JavaEE的概念

JavaEE 企业级开发Java规范 （ Java Enterprice Edtion——Java企业版）。标准规范包括：servlet、jsp、jdbc、xml、jta、javamail等13个技术规范（参见文档）。在java中规范就是接口。JavaEE又称为J2EE。

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。(要用到的)
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

运算能力较强、稳定性较高

1. WebSphere：IBM公司的一个收费的超大型的服务器软件，支持EE的所有的规范。

数据库使用DB2，机器使用IBM系列

银行、金融、证券

运算能力无人出其左右，

稳定性极强

社区版：Geronimo

1. 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的容器和服务器，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。

# 2、TOMCAT（重点）

### 2.1、概述

Tomcat 服务器是Apache旗下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。

### 2.2、Tomcat版本（了解）

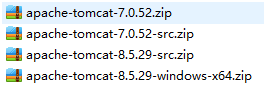
tomcat 目前存在很多版本，希望大家了解tomcat的版本，从而知道自己的创建的项目使用的是几版本规范，不同版本的规范技术可能不同。我们学习的WEB5.0，Servlet规范2.5，tomcat至少使用6版本。

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

### 2.3、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.

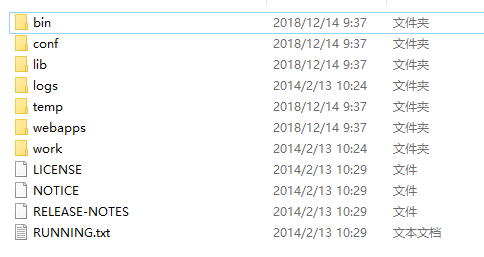
将解压后的文件copy到一个没有中文和空格的路径下即可.

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

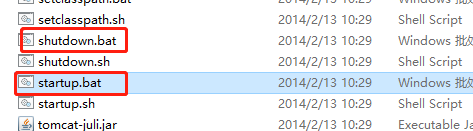
要求：

* + - * 1. **目录层次不要深**
        2. **程序目录的名称不能包含中文、特殊符号**

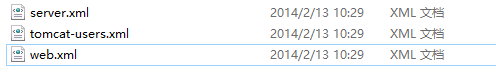
### 2.4、Tomcat目录结构



Bin：脚本目录(存放启动、关闭这些命令)



Conf:存放配置文件的目录



Lib:存放jar包

Logs: 存放日志文件

Temp： 临时文件

Webapps： 项目发布目录（以后自己编写的项目就是发布到这个目录中）

Work: jsp文件编译后生成的.java文件和.class文件存放的目录

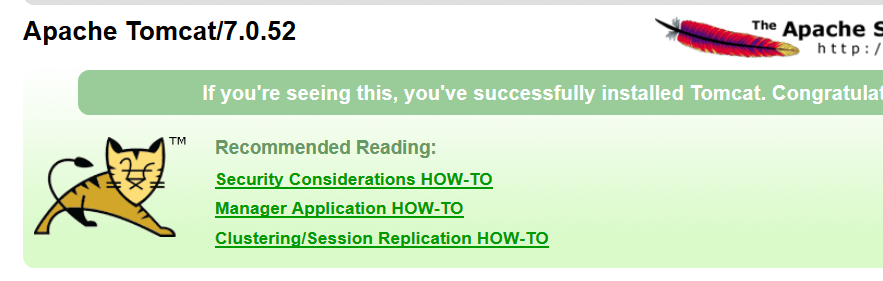
### 2.5、Tomcat启动和访问、关闭

启动：



访问：http://localhost:8080/

如果看到如下页面，代表访问成功



关闭：

正常关闭：

 除了这种方式，其它方式都是异常关闭

异常关闭：

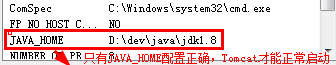
关闭窗口

Ctrl+c 强制退出

### 2.6、Tomcat常见问题

#### 2.6.1、JAVA\_HOME的配置

启动Tomcat时，闪一下就关闭了。



#### 2.6.2、端口号冲突

|  |
| --- |
| java.net.BindException: Address already in use: JVM\_Bind |

处理方式：

1. 关闭正使用该端口的程序

|  |
| --- |
| **cmd>netstat -ano**  查看哪些程序占用了哪些端口，找到占用端口的程序，进入任务管理器结束该进程。 |

1. 更换端口号

|  |
| --- |
| <Connector port="8080" protocol="HTTP/1.1"  connectionTimeout="20000"  redirectPort="8443" />  **注意：修改时，不要使用1024以前的端口号（一般被操作系统占用）** |

### 2.7、web项目目录结构

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

|-----静态资源.HTML，CSS，JS、图片、视频.....

|-----**WEB-INF**

|----**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。Serlvet,service,dao,utils.**相当于src目录**

**一个项目只要有classes才可能有动态WEB资源**

**只要WEB项目符合规范，放置在webapps目录中，服务器就会将其顺利发布网络**

**注：WEB-INF是受保护目录，从浏览器不能直接访问，只有通过请求转发（后面学习）**

### 2.8 发布项目到tomcat中

方式一：

拷贝项目文件夹到tomcat的webapps目录中

方式二：

修改tomcat的server.xml

|  |
| --- |
| <Host>  <Context path="/访问路径" docBase="E:/bbb"></Context>  **path** 访问的虚拟路径  **docBase** 项目所在的真实路径  </Host> |

**这种方式Tomcat不建议使用（Server.xml是tomcat的核心配置文件，一旦改错了，tomcat有可能启动不了）**

方式三：

在tomcat的 config/catalina/localhost/目录下创建 \*\*\*.xml

在 xml文件中：

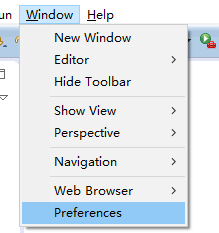
|  |
| --- |
| <Context docBase=E:\ccc"></Context>  **docBase** 项目所在的真实路径  **path不要了**  **虚拟路径就是这个xml文件的文件名**  **xml文件名可以用来分级**  **例如：xyz.xml 路径就是 xyz/**  **xyz#aaa.xml 路径就是 xyz/aaa/**  **#--------用来做分级** |

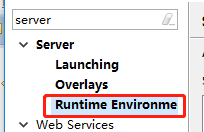
### 2.9、eclipse和tomcat整合

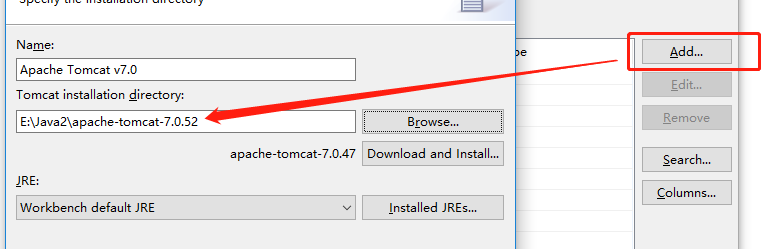
在将来的开发中，程序员都是使用eclipse和tomcat整合使用，方便开发测试。



1. 配置tomcat

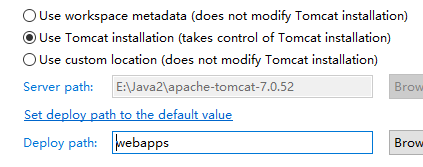




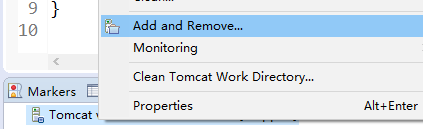


1. 修改配置文件

双击打开



1. 将项目部署到服务器



# HTTP协议（重点）



## 3.1、HTTP协议的概述:

### 3.1.1、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。

HTTP协议：专门用来进行超文本信息传输的 规定和规范，可以使得超文本传递更为快捷高效

协议：规定、规则、规范

该协议是为了使得浏览器和服务器之间 请求-响应传递的 超文本信息更加快捷高效。

### 3.1.2、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TCP协议

基于请求/响应模型。请求和响应必须成对；先有请求后有响应。

默认端口是80端口

### 3.1.3、HTTP协议的版本

* HTTP/1.0，发送请求，创建一次连接(Connection )，获得一个web资源，连接断开。

<img src=”001.jpg” /> 建立一次连接，断开

<img src=”002.jpg” />

<img src=”003.jpg” />

<img src=”004.jpg” />

* HTTP/**1.1**，发送请求，创建一次连接，获得多个web资源，连接断开。（效率更高）——广泛应用

### 3.1.4、HTTP协议的组成：

无论是请求还是响应，信息都是二进制信息

HTTP请求协议、HTTP响应协议。

* HTTP请求协议，是浏览器把信息发送给服务器的方式。
* HTTP响应协议，是服务器把信息返回给浏览器的方式。

## 3.2、HTTP协议入门

### 3.2.1、准备工作

发布一个WEB项目到TOMCAT，WEB项目中准备一个HTML页面

<form 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>

<hr />

<form 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>

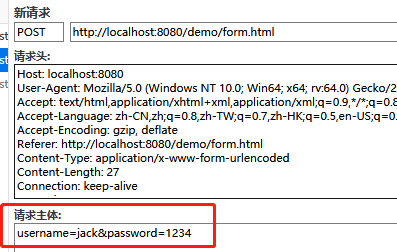
### 3.2.2、HTTP请求的详解

请求行：

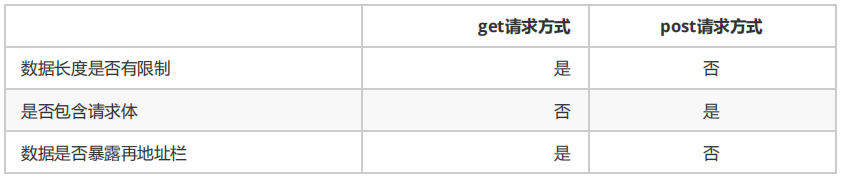
Get方式：



Post方式：



Get请求和post请求的区别？



请求头：

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（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  请求的来源。 |
| Cookie | 与会话有关技术，用于存放浏览器缓存的cookie信息。（后面课程主讲） |
| 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  用户-代理 |
| 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。  MIME类型：互联网上为了具体传递文件类型的一种方式。  扩展名类型：硬盘上为了保存数据的一种文件类型方式，例如：\*.html,\*.jpg  MIME格式：大类型/小类型[;参数]  例如：  text/html ，\*.html，html格式的文本  text/css，\*.css， css格式的文本  image/JPEG，\*.jpg ， JPEG格式的图片 |
| 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） |
| 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。（目前使用频率不高） |

* 请求体，用于**传递表单数据**，**数据量无限**

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。



### 3.2.3、HTTP响应的详解



* 响应行，用于返回响应结果。（极为少量）

例如：HTTP/1.1 200 OK

HTTP/1.1 协议版本

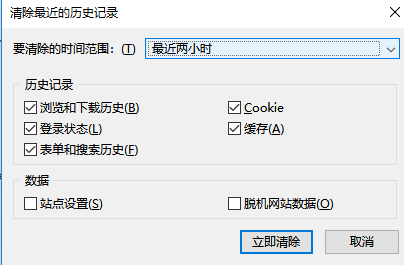
状态码：

200 响应成功 （200——299）

404 路径找不到

302 重定向

304 访问的是浏览器缓存 如何清除浏览器缓存 快捷键：ctrl+shift+delete



400 方法请求有问题

500 服务器程序出现异常

状态码描述：

* 响应头，用于存放预定义好的key/value少量信息

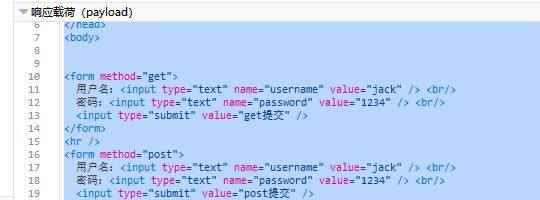
响应头也是用的键值对k:v

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。

|  |  |
| --- | --- |
| 常见响应头 | 描述(了解) |
| **Location** | 重定向的路径  指定响应的路径，需要与状态码302配合使用，完成跳转。（后面课程主讲） |
| **Content-Type** | 响应正文的类型（MIME类型）  取值：text/html;charset=UTF-8 |
| **Content-Disposition** | 正文-配置。 在文件下载时才会使用  通过浏览器以下载方式解析正文（后面课程主讲）  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（后面课程主讲）  服务器和浏览器之间传递的一张小纸条 |
| Content-Encoding | 服务器使用的压缩格式  取值：gzip |
| Content-length | 响应正文的长度 |
| Server | 指的是服务器名称，默认值：Apache-Coyote/1.1。可以通过conf/server.xml |
| *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。（目前不用）* |

* 响应体，传递的是请求需要查看的资源数据，数据量无限

响应体，就是服务器发送给浏览器的资源数据。



即：若浏览器想看html网页，响应体里就是html页面代码

若浏览器想看jpeg图片，响应体里就是JPEG图片的二进制字节流数据

若浏览器想看AVI视频，响应体里就是AVI视频的二进制字节流数据

要求：使用Servlet写一个入门案例

1. 创建一个Java类
2. 继承HttpServlet
3. 重写doGet doPost方法

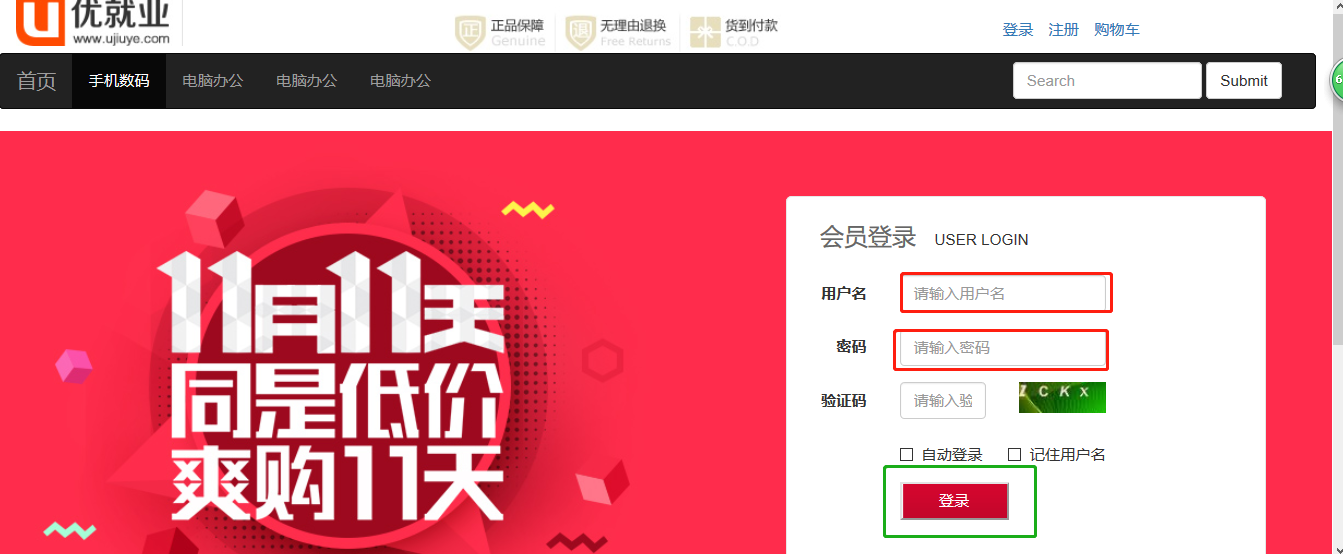


1. 在web.xml编写配置



# 4、案例：使用Servlet技术完成一个用户登录（必须做 2遍）

## 4.1、需求说明



* 若用户名和密码正确，判定为登录成功，则控制台输出登录成功；
* 若用户名或密码错误，判定为登录失败，则控制台输出登录失败。

## 4.2、需求分析

1、我们需要服务器Java程序能够从提交的表单数据中获取用户名和密码(Servlet)

2、我们需要服务器有Java程序能够处理用户名和密码，并去数据库查询校验

3、对查询结果进行判断

## 4.3、知识讲解-Servlet

### 4.3.1、什么是Servlet

Servlet=server+applet（服务器端小程序）

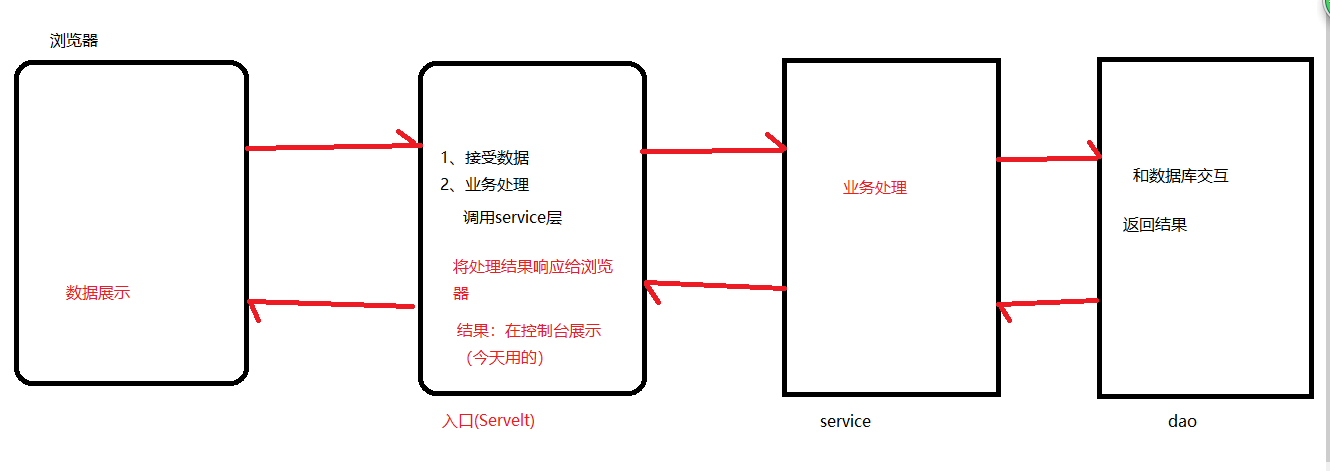
Servlet是sun公司提供的一门用于开发动态web资源的技术。

运行在服务器端，一个小的Java程序。它可以接收从web浏览器发送过来的请求，也可以给浏览器产生相应。通常使用超文本传输协议(HTTP)

**JavaEE、Servlet、JSP 版本以及tomcat服务器的支持情况：**

|  |
| --- |
| J2EE Servlet JSP Tomcat  1.3 2.3 1.2 4  1.4 2.4 2.0(EL JSTL) 5  JavaEE5 2.5 2.1 6  JavaEE6 3.0 2.2 7  JavaEE7 3.1 ？？ 8 |

### 4.3.2、Servlet的作用 （MVC）



用来处理从客户端发送过来的请求,并对该请求作出响应。

Servlet的任务有：入口

1. 接收浏览器数据
2. 调用Service处理请求
3. 向浏览器发出响应（结果展示）

Servlet中的规范代码结构：

**try{**

**//1、准备数据**

**//2、调用service处理**

**//3、结果展示**

**}catch(Exception e){**

**}**

### 4.3.3、Servlet的入门

#### 4.3.3.1、准备工作

Servlet规范要求：Servlet程序需要编写实现类，并在web.xml进行配置。

* 实现类：通常继承javax.servlet.http.HttpServlet类，并复写doGet和doPost方法。
  + doGet()方法用于处理get请求。
  + doPost()方法用于处理post请求。

配置信息：在web.xml进行配置。

#### 4.3.3.2、编写步骤

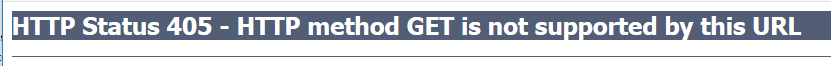
1. 创建类，继承HttpServlet，复写doGet和doPost方法
2. 编写配置文件
3. 浏览器访问
4. 服务器执行

以下三步缺一不可，否则Servlet无法正常运行。

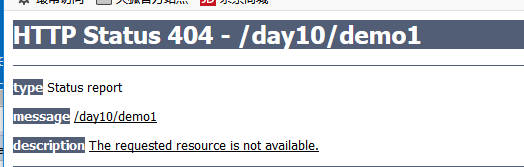
1. **java程序必须继承HttpServlet**



1. **java程序必须重写doGet() doPost()**

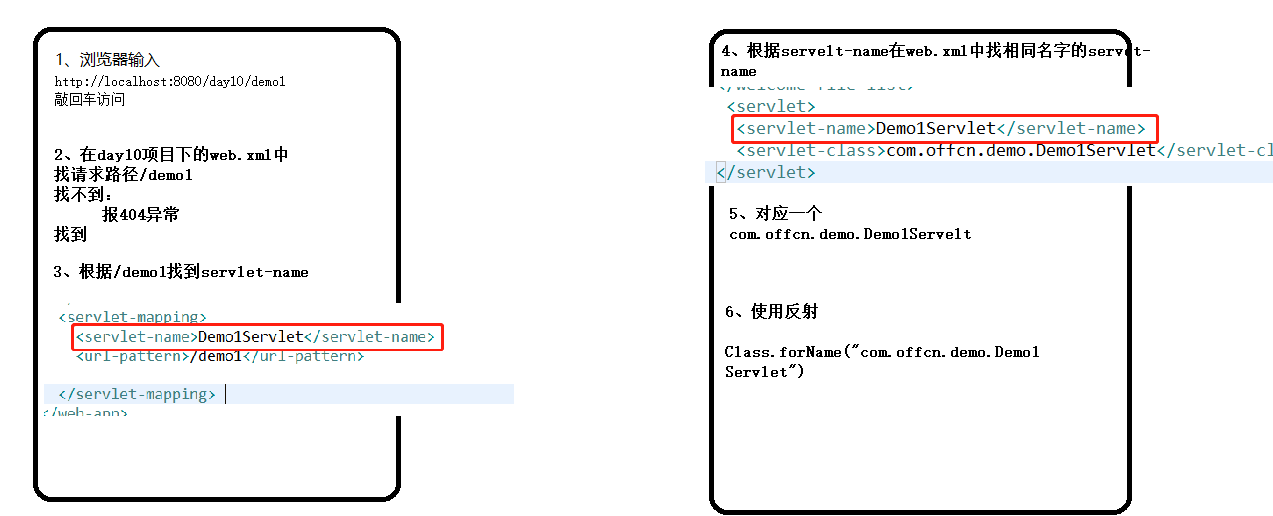


1. **java程序必须在web.xml中配置 <servlet> <servlet-mapping>**



#### 4.3.3.3、Servlet的执行的流程详解:

问题：为什么在浏览器输入http://localhost:8080/day10/demo1就能够执行doGet()方法



### 4.3.4、Request接收请求参数

#### 1.3.4.1、Request接收请求参数的概述

请求参数

Request对象的方法

|  |  |
| --- | --- |
| 方法名 | 描述 |
| **String getParameter(String name)** | 获得指定参数名对应的值。  返回值：String  方法不区分get/POST  如果参数列表上没有对应的参数名，方法会返回null。  如果参数名对应多个参数值，该方法只能获得第一个。 |

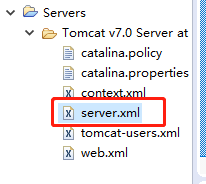
获取数据的乱码问题：

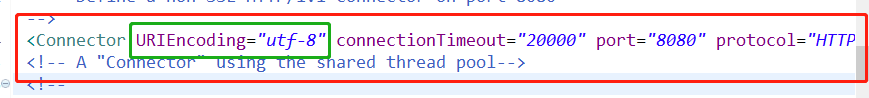
Get乱码：

1. 手动解决乱码（缺点：需要一个一个参数进行解码）

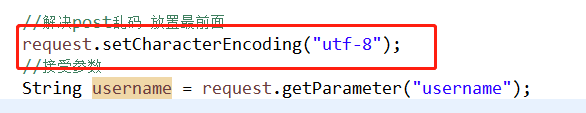


2、一劳永逸的解决方法





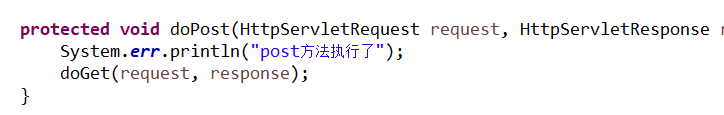
Post乱码：



服务器对请求行的固定解码为：Iso8859-1

服务器对请求体的默认解码为：Iso8859-1

doGet和doPost调用



doGet和doPost相互调用

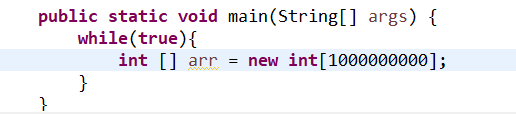


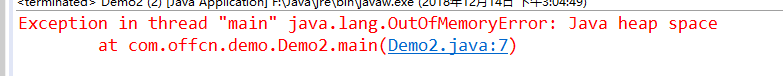


递归：得有一个条件结束循环

方法运行在栈内存：

堆内存溢出：

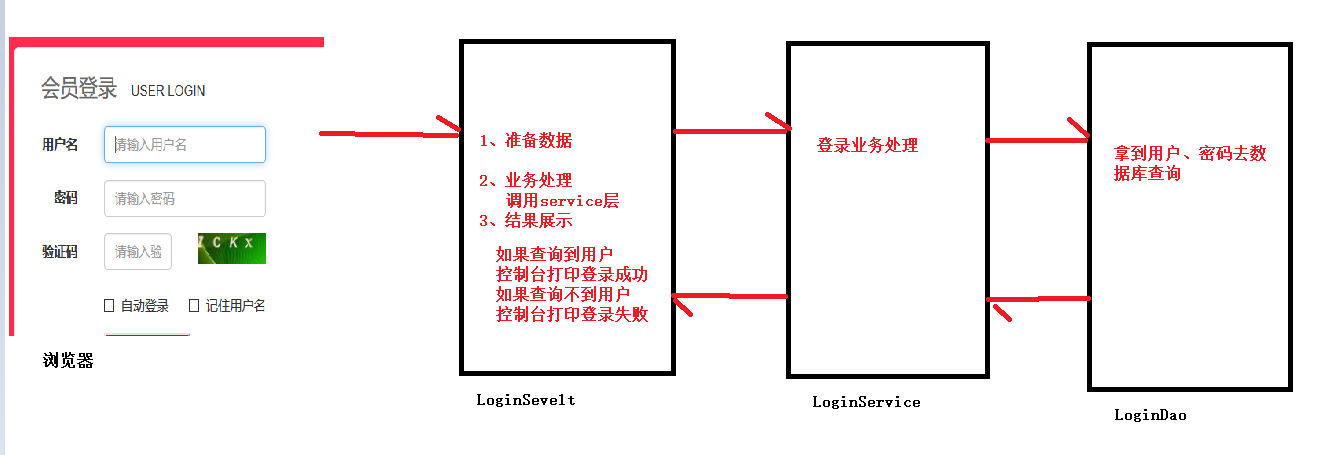




注意：

不要关闭

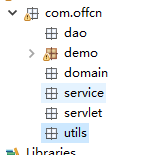
## 4.4、登录流程分析



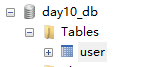
## 4.5、案例原型准备及代码实现

案例准备：

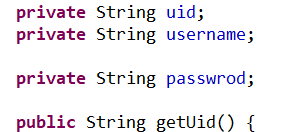
1. 创建各层的包



1. 创建数据库和表



1. 创建实体类



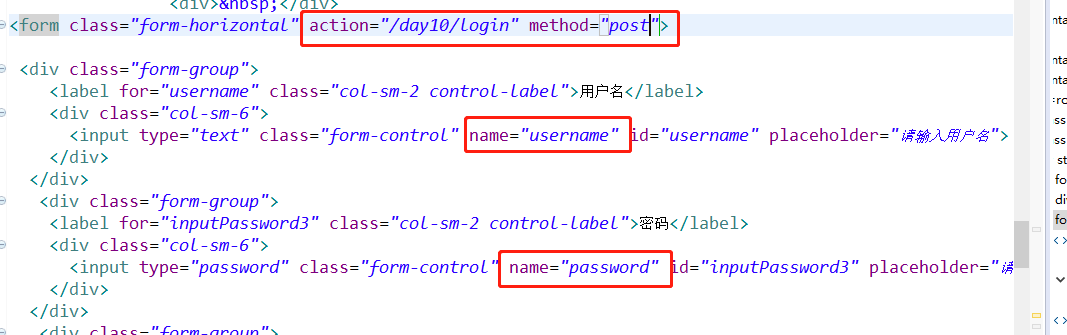
1. 导入jar包、连接池、工具类





案例实现：

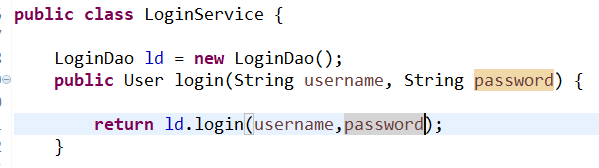
Login.html



LoginServlet：



LoginService



LoginDao：、

