## Tomcat

找进程杀进程：netstat -ano|findstr "80"

taskkill /f /pid 1828

1. 修改8080到web默认端口号80：



在其配置文件中修改8080到80后就不用使用端口号就可以进行访问了：



J2ee在Tomcat上的部署：

<https://how2j.cn/k/tomcat/tomcat-deploy/1143.html>

## Serverlet

### 从0开始配置一个web服务java平台

1.首先要有一个最基本的java project模子和一个Servlet

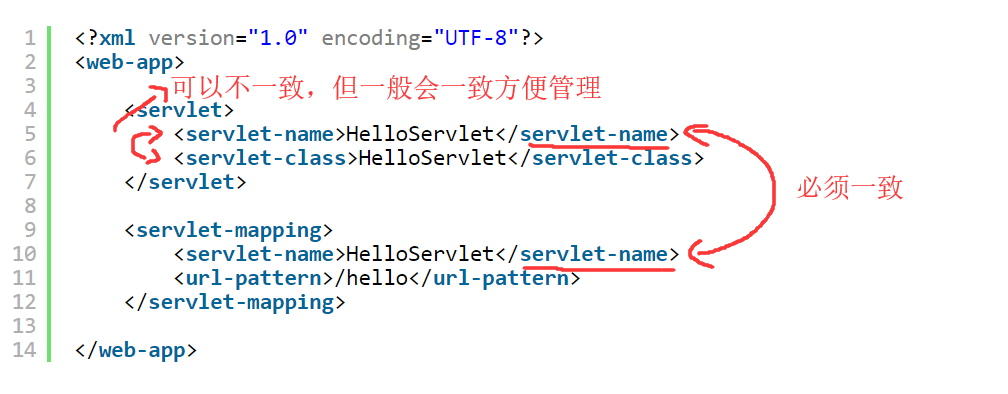
2.然后创建web文件夹：

web文件下有WEB-INF，里面有web.xml来给servlet文件和路径提供对应。

比如使 HelloServlet和/hello 这个路径相对应。



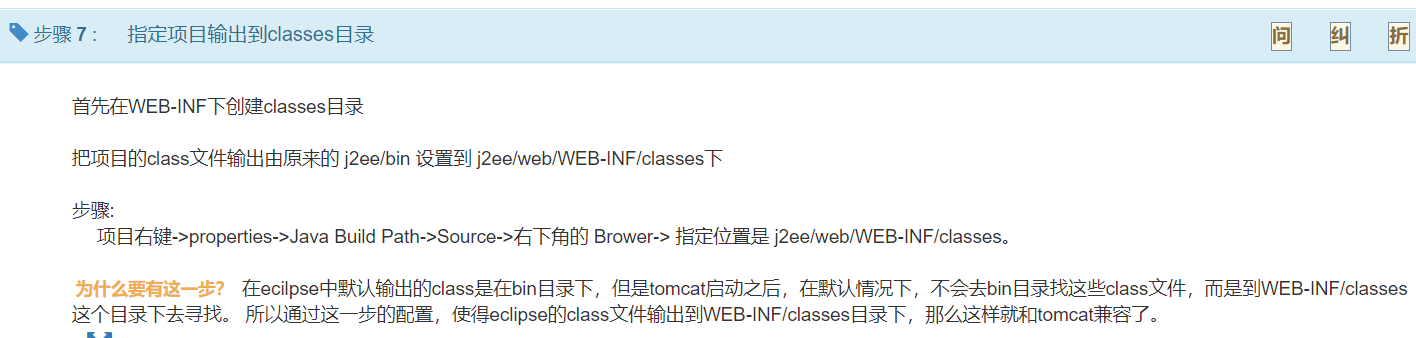
格式对应：



原文：

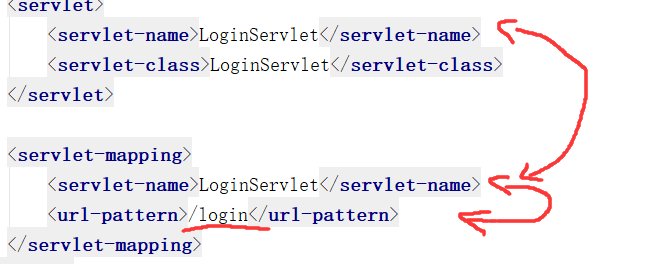


1. 将eclipse的输出目录改为WEB-INF底下的classes文件夹（要自己创建），因为在tomcat中寻找的输出目录是WEB-INF下的classes，而默认eclipse的输出目录是bin，所以需要修改。



### 2.一个Serlevt的地址就是对应一个接口的地址。





相互对应 url-parttern对应Servlet-name对应servlet-class

### doPost和doGet属于service ()

一个类继承了HttpServlet的时候，同时也继承了一个service方法。

在执行doPost和doGet之前都会执行service()方法，所以service其实包含了doPost和doGet方法，所以可以直接代替他们两直接写service().

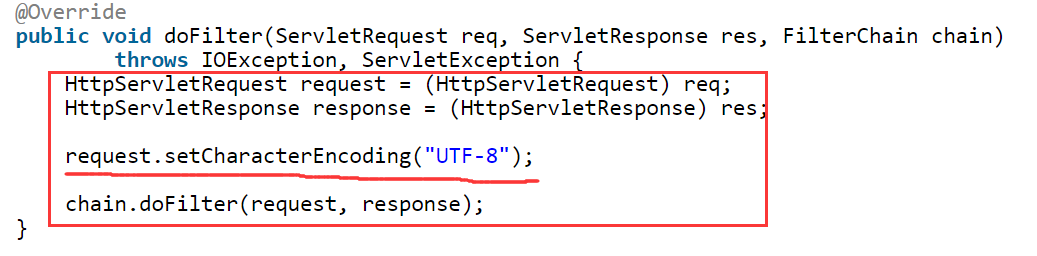




### 处理中文问题：

正确的方式是在过滤器中添加中文处理，然后在进入到Servlet中，

因为如果按下面的方式去处理中文，那么每一个Servlet中都需要设置中文处理了，就很麻烦。



传统不方便的做法：

在servlet中设置request和response的编码格式支持中文：

request.setCharacterEncoding(**"UTF-8"**);  
response.setContentType(**"text/html; charset=UTF-8"**);

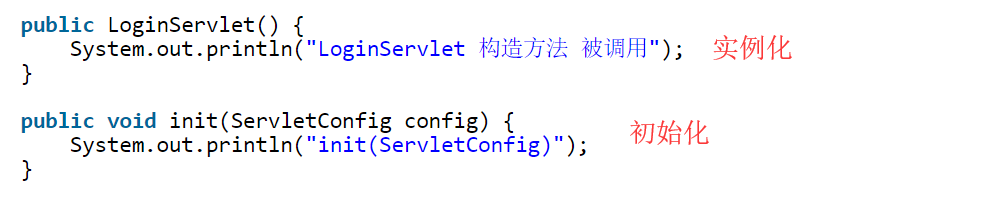
在浏览器头部中通知浏览器使用UTF-8编码向服务器发送数据：

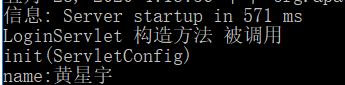
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">



5.生命周期：







### Servlet跳转：

跳转可以分为服务端跳转和客户端跳转：

原文：

页面跳转是开发一个web应用经常会发生的事情。  
  
比如登录成功或是失败后，分别会跳转到不同的页面。  
  
跳转的方式有两种，**服务端跳转**和客户端跳转

1. 服务端跳转：

request.getRequestDispatcher("success.html").forward(request, response);



一步到位

跳转到success.html且包含将req和resp也打包发过去。

1. 客户端跳转：

客户端跳转就比较煞笔（虽然自己以前也是这么做的。）

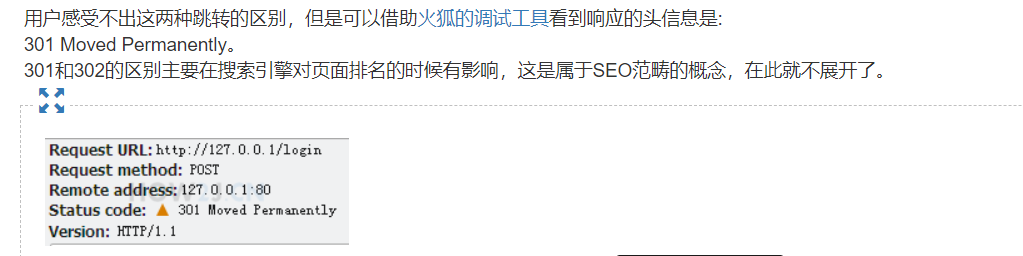
发送请求后台处理处理之后返回状态令牌，然后前端靠状态令牌再去向后端访问某个页面。（这是最傻逼的）

response.sendRedirect("fail.html");

二是response.setStatus(301);

response.setHeader("Location", "fail.html");

区别：



这样是让浏览器去另一个页面，是浏览器客户端跳转。

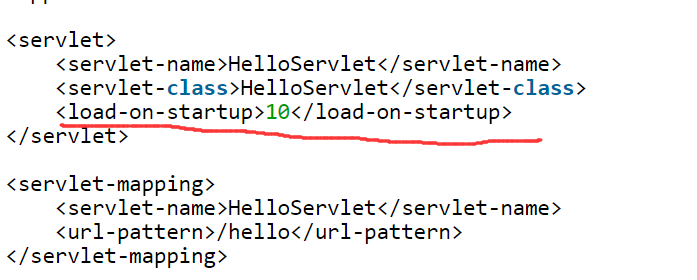
6.

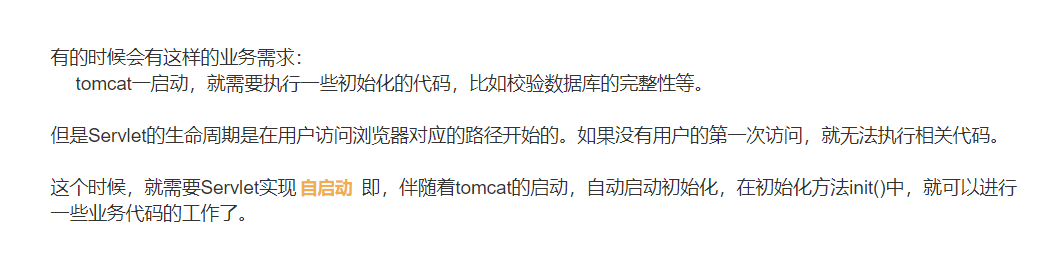
### 6.Servlet预启动：

没用到过，先码着：

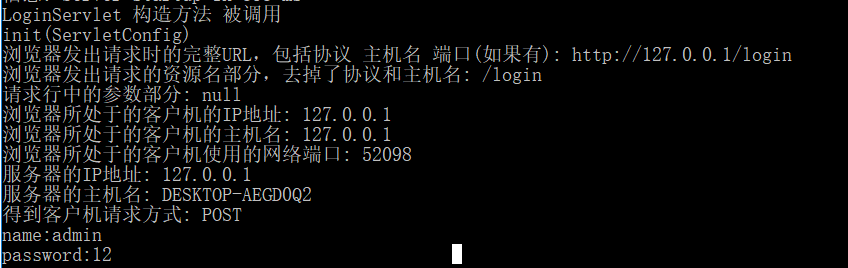
在web.xml中对应的Servlet中配置：

<load-on-startup>10</load-on-startup>

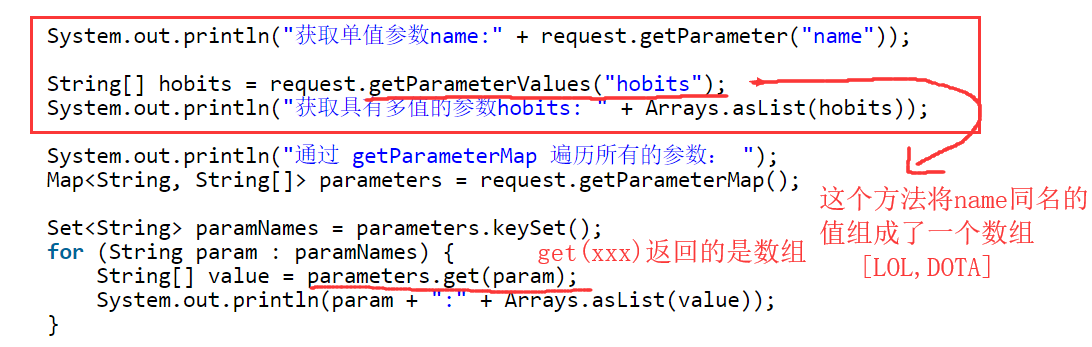




### 7.request的一些实用方法：

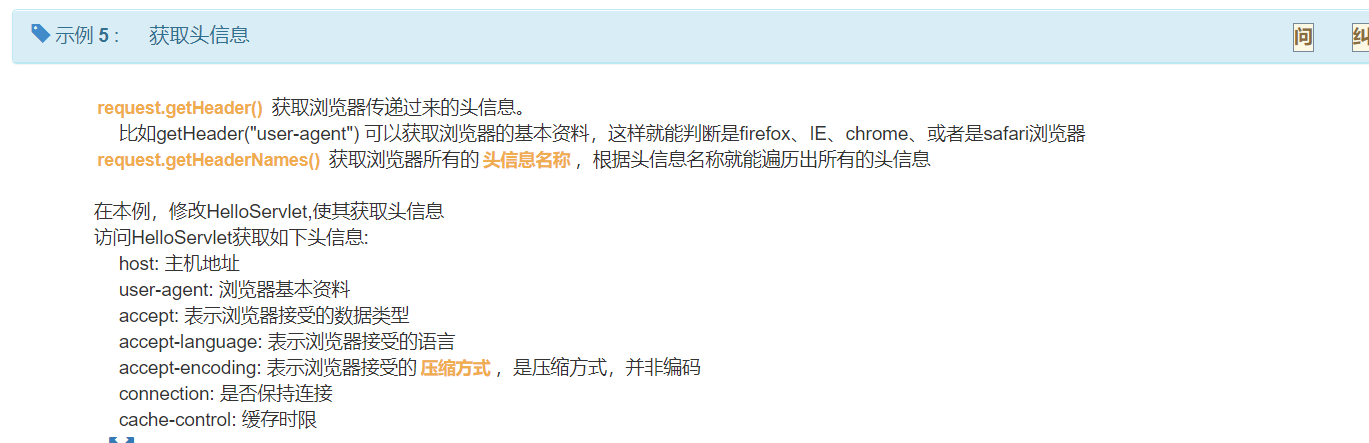


单参数和多参数的key传过来的处理方法：



这个方法直接将同名的值打成了数组。

获取头部信息：



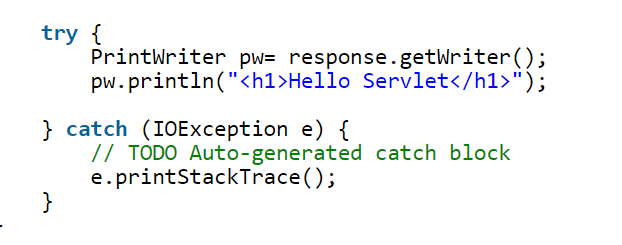




### 8.response的用法：

1. 常规用法：

New出一个printwrite对象进行写入：



同时可以设置相应格式：

response.setContentType("text/html");

1. 设置编码：

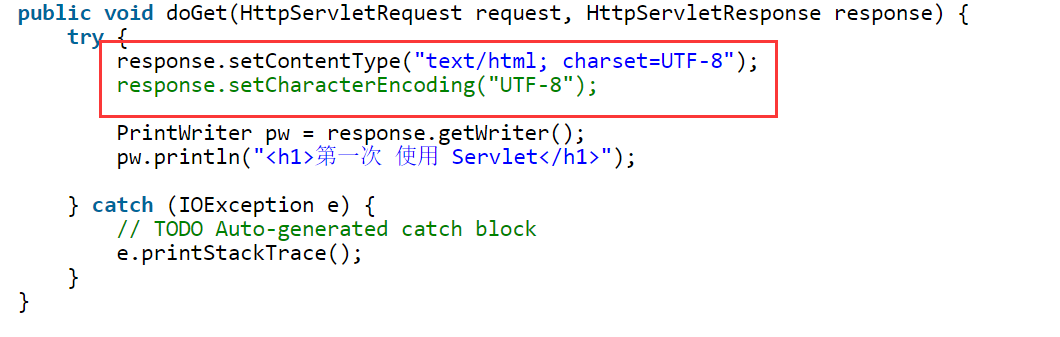
有两种方式：

1. response.setContentType("text/html; charset=UTF-8");

2. response.setCharacterEncoding("UTF-8");

第一种是返回使用UTF-8格式，并且通知浏览器使用UTF-8来读取文件。

第二种是只是使用UTF-8返回数据，而浏览器使用什么编码管不着，如果浏览器不是UTF-8中文就会乱码。



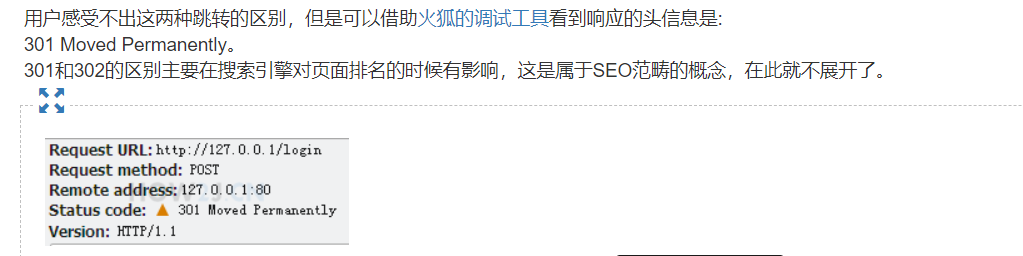
1. 客户端跳转的两种方式：

一种就是最普通的response..sendRedirect()

二是response.setStatus(301);

response.setHeader("Location", "fail.html");

区别：



### 9.Servlet文件上传到本地

标准流程参见

UploadPhotoServlet.class

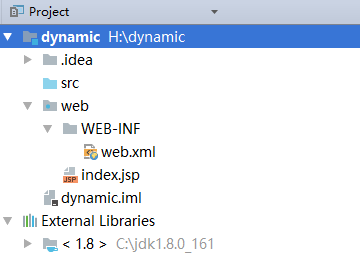




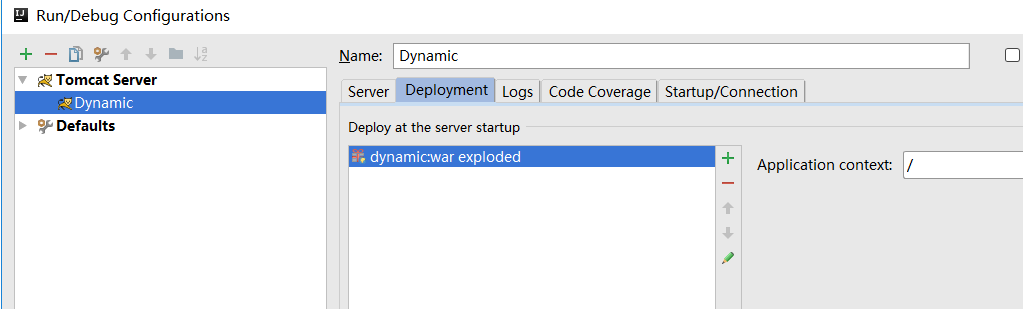
## 3.Dynamic Web Project:

创建一个dynamic web project，idea会自动把支持tomcat的东西都配置（包括输出文件夹Build（不是classes），web文件夹->WEB-INF->lib->web.xml





并且添加了tomcat后要在它的配置中的deployment中添加这个项目：



## Filter（登录，中文编码）

顾名思义 过滤关卡，也是用来节省资源调用的好方法。

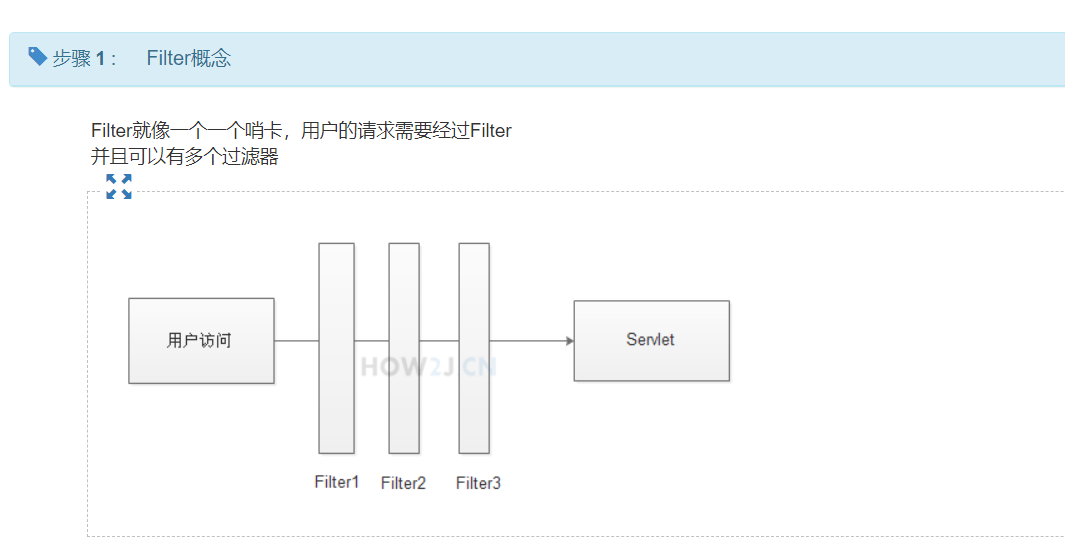
与Servlet需要配置[自启动](https://how2j.cn/k/servlet/servlet-startup/553.html)才会随着tomcat的启动而执行init()方法不一样。  
  
Filter****一定****会随着tomcat的启动自启动。

Filter是web应用非常重要的一个环节，如果Filter启动失败，或者本身有编译错误，****不仅这个Filter不能使用****，****整个web应用会启动失败****，导致用户无法访问页面

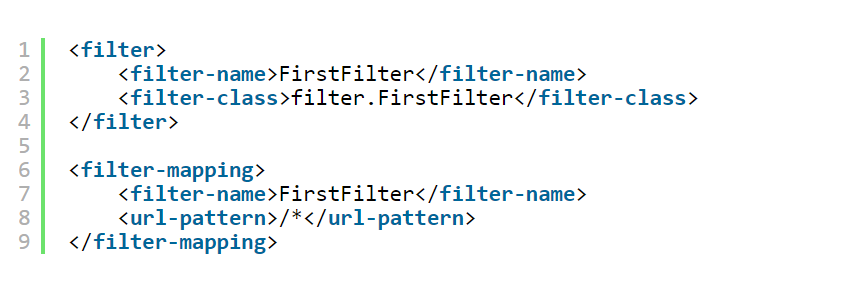
doFilter方法的第一个参数req,是ServletRequest 类型的，不支持setCharacterEncoding，所以要先强制转换为HttpServletRequest（用来处理中文编码）

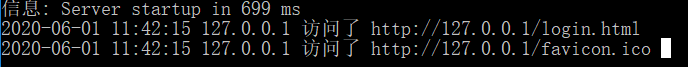
对于用户验证的好处：

在[用户是否登陆的验证](https://how2j.cn/k/mvc/mvc-authentication/591.html)中，我们可以通过在HeroListServlet中增加对session的判断代码来做到登陆验证。  
  
但是按照这样的做法，所有的Servlet都要加上一样的代码，就会显得比较累赘。  
  
与通过[Filter处理中文问题](https://how2j.cn/k/filter/filter-encoding/589.html)一样，也可以通过Filter一次性解决所有的登陆验证问题

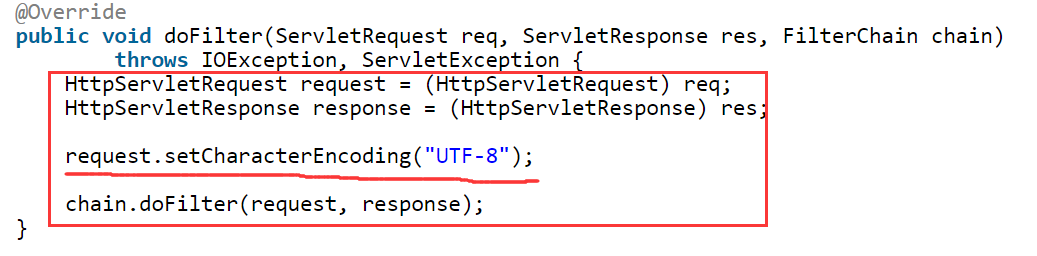








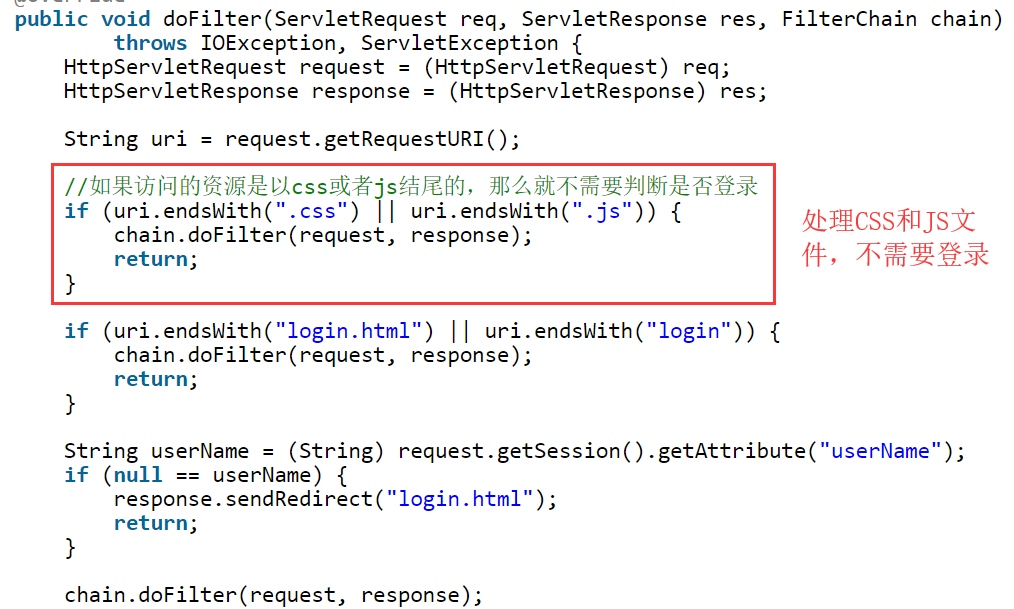
对中文的过滤：



登录过滤：



还有CSS和JS文件是不需要登录就可以访问的，加上img



## Listener 监听

### Cookie（钥匙）：

由服务器创建而保存在客户端当中的凭证。可以使用cookie来找到放东西的session

Cookie是一种浏览器和服务器交互数据的方式。  
  
Cookie是由服务器端创建，但是****不会保存在服务器****。  
  
创建好之后，发送给浏览器。浏览器****保存在用户本地****。  
  
下一次访问网站的时候，就会把该Cookie发送给服务器。

其他描述：

Set-Cookie响应头字段（Response header）是服务器发送到浏览器或者其他客户端的一些信息，一般用于登陆成功的情况下返回给客户端的凭证信息，然后下次请求时会带上这个cookie，这样服务器端就能知道是来自哪个用户的请求了。

Cookie请求头字段是客户端发送请求到服务器端时发送的信息（满足一定条件下浏览器自动完成，无需前端代码辅助）。

http请求带上cookie的条件：

下表为Set-Cookie响应头可以设置的属性



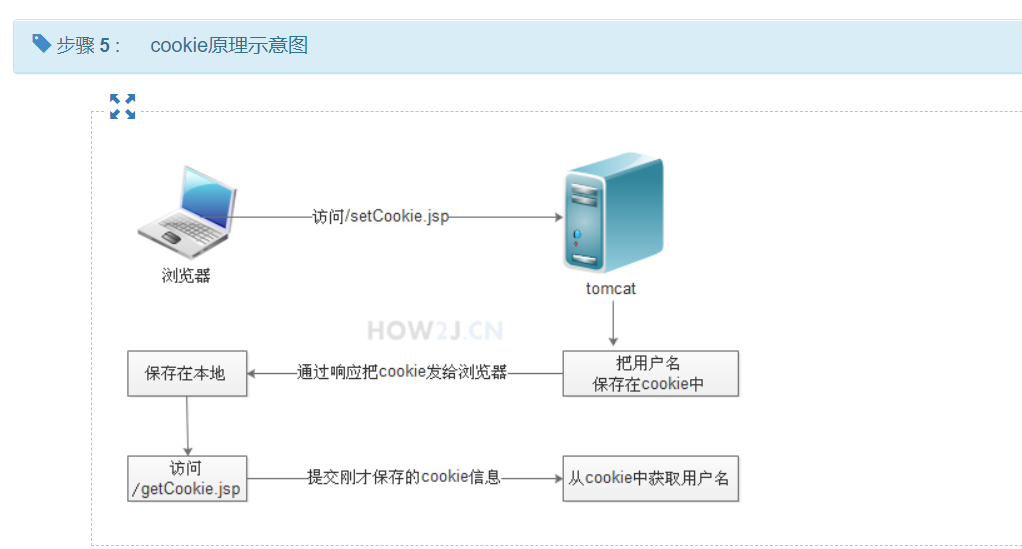
请看上面标红的三个属性，拿一个Http POST请求来说  http://aaa.www.com/xxxxx/list

如果满足下面几个条件：

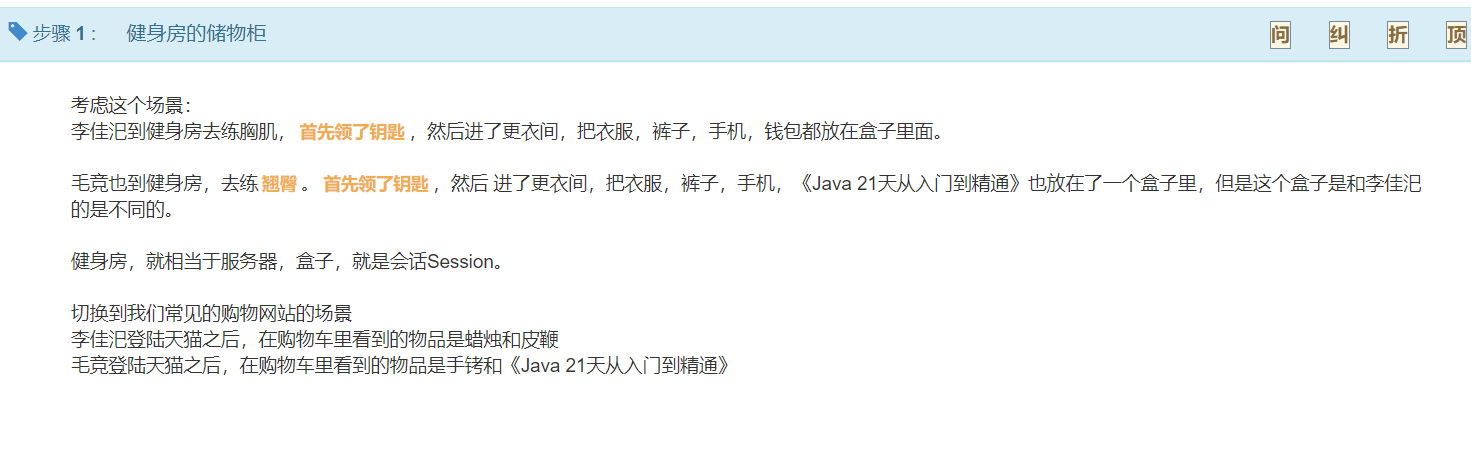
1、浏览器端某个Cookie的domain字段等于aaa.www.com或者www.com

2、都是http或者https，或者不同的情况下Secure属性为false

3、要发送请求的路径，即上面的xxxxx跟浏览器端Cookie的path属性必须一致，或者是浏览器端Cookie的path的子目录，比如浏览器端Cookie的path为/test，那么xxxxxxx必须为/test或者/test/xxxx等子目录才可以



### Session（柜子）:

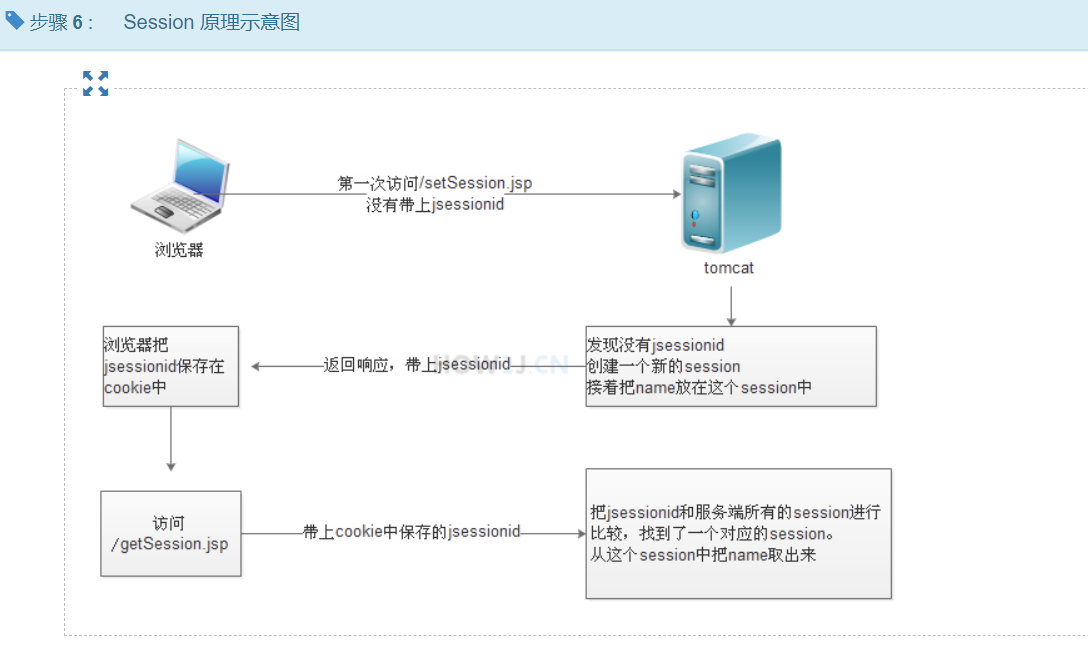


session.setAttribute("name", "teemo");

session.getAttribute("name");

对于cookie和Session的关系：

回到[健身房的储物柜](https://how2j.cn/k/jsp/jsp-session/583.html#step3042)这一段：  
  
李佳汜和毛竞都有自己的盒子，那么他们怎么知道哪个盒子是自己的呢？  
通过钥匙就能找到自己的盒子了。  
  
盒子对应服务器上的Session。  
钥匙对应浏览器上的Cookie。



比如登录一个网站，登录后，在短时间内，依然可以继续访问而不用重新登录。  
  
但是较长时间不登录，依然会要求重新登录，这是因为服务端的session在一段时间不使用后，就失效了。  
  
这个时间，在Tomcat默认配置下，是30分钟。  
  
可以通过 d:/tomcat/conf/web.xml 中的session-config 配置进行调整

### 利用Session实现用户登录统计

利用监听Session的多少来实现用户的统计。

