Servlet的编码问题

一、JSP和Servlet的乱码问题：

1）请求转发过程

记住一点：第一个HTTP请求是由浏览器生成的，使用的**默认字符编码为ISO-8859-1，而请求中含有中文等ASCII码之外的字符，就使用下面的规则。一般请求URL或请求体包含的大多是参数和参数值，这一个键值对**

一般来说，URL只能使用英文字母、阿拉伯数字和某些标点符号（ASCII码范围），不能使用其他文字和符号。

但是URL 常常会包含 ASCII 码范围之外的字符，所以URL 必须转换为有效的 ASCII 格式才能正确使用。URL使用"**%"其后跟随两位的十六进制数**来替换非 ASCII 字符。URL 不能包含空格，URL 编码通常使用 **+ 来替换空格**。

但是这个跟在%后面的**十六进制数是多少，是由编码时使用的编码方式决定的**。（也就是前端页面的字符编码）

另外HTTP 请求传入servlet对象中，servlet根据HTTP请求首部字段中的Accept-Charset字段是浏览器声明自己准备得到的响应的字符编码，可以有几种，而且有优先级选择，这部分不需要怎么理会；

2）容器

Tomcat的server.xml设置URIEncoding="UTF-8"

和useBodyEncodingForURI="true"，具体作用可查阅书签收藏

**3）响应生成过程**

在servlet中处理生成的响应时，再记住一点：servlet对HTTP响应的默认编码处理是西欧编码，所以要指定响应的文本类型（MIME+charset）

Response.setContentType(“text/html; **charset=utf-8**”);

**！！！**如果漏了charset部分，则响应默认使用西欧编码iso-8859-1，这样除了西欧字符，都会变成乱码或者问号

二、 [Http响应和请求的编码问题](http://blog.csdn.net/tianmaxingkong_/article/details/45935671)

**1.HttpServletRequest请求中的编码一致**

    通过Html的表单<form>向服务器提交数据，在Servlet中通过request.getParameter(key)来获得参数值，由于**Http请求的默认字符编码为ISO-8859-1**，需要在获得参数值之后进行字符编码转换。E.g.

**Html表单：**

<form action="/TestPro/servlet/CheckServlet">

    输入：<input type="text" name="name" />

    <input type="submit"/>

</form>

**CheckServlet：**

String name = request.getParameter("name");

String name\_u = new String(name.getBytes("ISO-8859-1"), "utf-8");

**【注意】**此处之所以是转化为utf-8的编码格式，是因为**<form>表单的编码格式是utf-8**。在html的<head>中:

<meta http-equiv="content-type" content="text/html; charset=utf-8">

当然此处的charset也可以指定为GB2312，同时在request中获得参数后，转化后的编码格式也得是GB2312，否则会出现乱码。

**2.HttpServletResponse响应中的编码**

HttpServletResponse**有两种方式响应Brower**，一种是OutputStream字节流，二种是PrintWriter字符流。

**2.1  OutputStream字节流**(字节流的编码格式与ContentType的编码格式要一致)

/\*

Utf-8编码的字符转字节

\*/

response.getOutputStream().write("天马行空".getBytes("utf-8"));

/\*

下面开始设置html的头部信息

\*/

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

//也可以通过以下两种方式

response.setHeader("Content-Type","text/html;charset=utf-8");

//或者

response.setContentType("text/html");

response.setCharacterEncoding("utf-8");

**2.2 PrintWriter字符流**（只需指定ContentType的编码格式）

response.setContentType("text/html;charset=GB2312");

/\*

字符流使用了默认的字符编码，直接把一块数据套上去

\*/

PrintWriter out = response.getWriter();

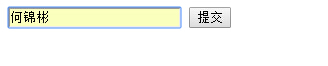
Out.println("天马行空");

Out.write("天马行空"); //两种输出方式效果一样

三、[网页提交中文到WEB容器的经历了些什么过程....](http://www.cnblogs.com/springsource/p/4892803.html)

先准备一个网页

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | <html><br><meta http-equiv="Content-Type" content="text/html; charset=gbk" />  <title>测试编码</title>  <body>  <form id="form1" name="form1" method="post" action="http://localhost:8080/TestServer/Receive">    <label>    <input type="text" name="name" />    </label>    <label>    <input type="submit" name="SBbt" value="提交" />    </label>  </form>  </body><br></html> |



输入中文字，名字   “何锦彬”  进行提交

提交到servelt ， 用servelt输出收到的内容

[复制代码](javascript:void(0);)

protected void doPost(HttpServletRequest request,

HttpServletResponse response) throws ServletException, IOException {

InputStream input = request.getInputStream();

BufferedInputStream inputBf = new BufferedInputStream(input);

byte[] buf = new byte[1024];

int length = inputBf.read(buf);

byte[] receviveData = new byte[length];

System.arraycopy(buf, 0, receviveData, 0, length);

System.out.println("收到:" + new String(receviveData));

// System.out.println(request.getParameter("name"));

}

[复制代码](javascript:void(0);)

输出如下：

收到:name=%BA%CE%BD%F5%B1%F2&SBbt=%CC%E1%BD%BB

尝试把"何锦彬" 按GBK转成 16进制输出

ffffffba ffffffce ffffffbd fffffff5 ffffffb1 fffffff2

忽略前面的'f', 会发现，servelt接受到的内容就是： 中文经过GBK编码转码后， 用16进制标识， 并在前面加上"%"以作区分

继续修改修改网页的meta信息，把"GBK" 换成"utf-8"

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

输出如下：

收到:name=%E4%BD%95%E9%94%A6%E5%BD%AC&SBbt=%E6%8F%90%E4%BA%A4

明显 name 后面是9个字符，是用UTF-8转码后16进制表示了字节后再加了'%' 组成

整个过程如下

1， **把中文根据meta 里的字符编码转换成字节**

2， **把字节加上百分号并用16进制表示**

（1，2部相当于java.net.URLEncoder.encode(content,"utf-8")）

3. 进行传输

4， WEB容器进行解码，相当于调用

TOMCAT中相关代码

org.apache.tomcat.lite.http.HttpRequest    672行

try {

parameters.add(urlDecode(tmpName, enc),

urlDecode(tmpValue, enc));

} catch (IOException e) {

// ignored

}