# 修改Servlet模板



每次创建的Servlet中有很多的注释，每次删除好麻烦，可以修改生成这个Servlet的模板，方法如下：



注册在桌面右击图标打开的位置还不是MyEclipse的安装根目录，还需要往上一层，在这一层里进行搜索。打开模板文件后删除掉那些注释即可。

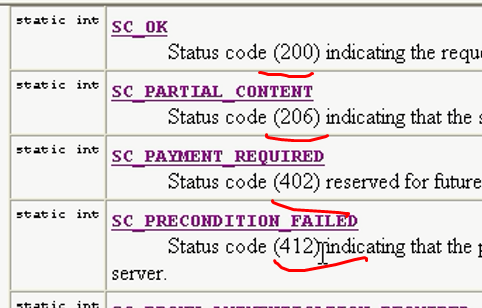
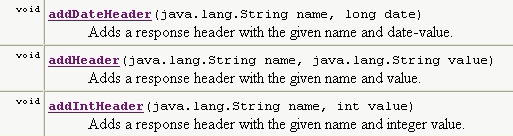
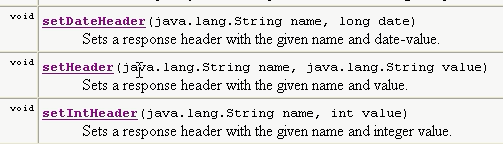
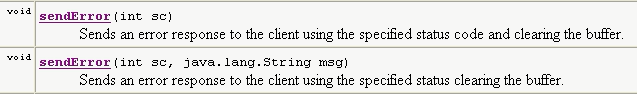
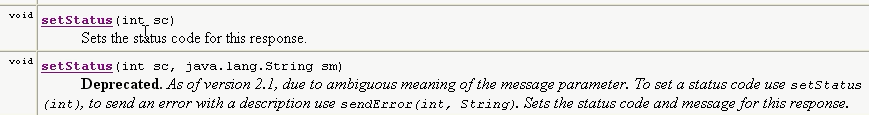
# 重命名Servlet名字的问题

当修改了XXXServlet.java的名字时，web.xml中的相关映射中的类名需要我们手动修改过来，系统并不会自动并我们改，即使我们改类名时使用的是重构也不行。

# 简介

* Web服务器收到客户端的http请求，会针对**每一次**请求，分别创建一个用于代表请求的request对象、和代表响应的response对象。
* request和response对象即然代表请求和响应，那我们要获取客户机提交过来的数据，只需要找request对象就行了。要向客户机输出数据，只需要找response对象就行了。

# HttpServletResponse

* HttpServletResponse对象代表服务器的**响应**。这个对象中封装了向客户端发送数据、发送响应头，发送响应状态码的方法。
* 可查看该类的API文档，该类有各种状态码的常量，如下：  
    
  这里只是列出了几个而已，还有很多很多的状态码常量  
    
  这3个方法中第2个是通用的，如有些头的参数值是一个数值，使用addIntHeader可以，使用addHeader传一个字符串形式的数字也可以  
    
  这3个方法在真实开发用的比较多一些，没有就添加头，有就修改值   
    
  传错误相关的响应码与消息  
    
    
  传没错的响应码与消息
* HttpServletResponse它的父类中的方法也需要看：  
    
  当调用此方法时，客户端就能立马看到结果，但是我们不需要调用这个方法，当给response写内容时，当缓存满了，它会自动调用此方法，或者doGet或doPost方法返回时Tomcat容器会帮我们调用。  
  默认缓存大小是8K  
    
  setContentLength，设置正文的长度，不设置也能知道  
  setContentType，，正文的类型，浏览器需要知道正文是什么类型，然后选择以什么样的方式处理，比如：如果正文是一个文件，则采用下载处理

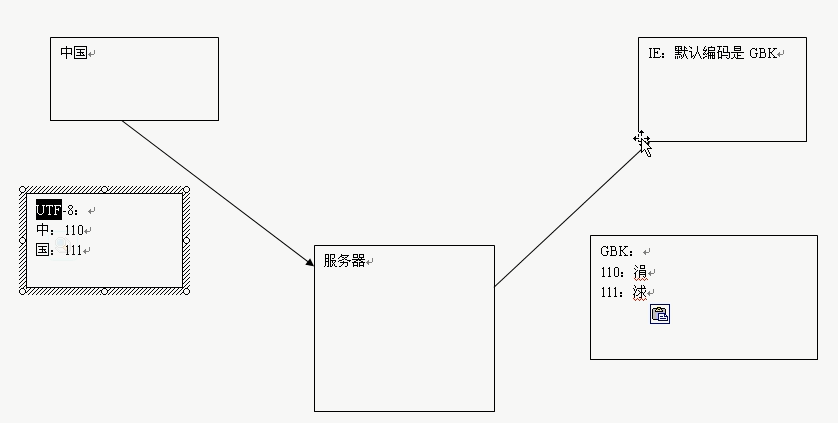
# response常见应用

## 向客户端输出中文数据

### 用OutputStream(字节流)发送数据：

1、response.getOutputStream().write(“中国”.getBytes());//以默认编码发送数据(GBK)，而浏览器在读取时默认也使用本地编码读，所以没有乱码问题

2、response.getOutputStream().write("中国".getBytes("UTF-8"));//以UTF-8编码发送数据，浏览器(默认用GB2312)会出现乱码

画图描述出现该问题的原因。  
 

解决办法：

2.1通过更改浏览器的编码方式：IE/”查看”/”编码”/”UTF-8”(不可取)

2.2通过设置响应头告知客户端编码方式：response.setHeader(“Content-type”, “text/html;charset=UTF-8”);//告知浏览器数据类型及编码  
 还有另一个更方便的方法：  
 内部原理也是添加头

2.3通过meta标签模拟请求头:out.write("<meta http-equiv='Content-Type' content='text/html; charset=utf-8' />".getBytes()); // 这是把内容写到了Html中，相当于在Html中设置头的效果，虽然不是响应头，功能类似响应头。  


2.4通过以下方法：response.setContentType("text/html;charset=UTF-8");

总结：程序以什么编码输出，就需要告知客户端以什么编码显示。

小细节：输出字符“1”用response.getOutputStream().write(1);出现的问题？ 写的时候是把1的字节码写出去了，但是IE显示时是把读取到的字节当字符来显示，所以会去查找GBK码表，把对应的字节的字符显示出来，而我们的字节是表示一个int值，并不是一个字符。解决办法把1变成String即可：”1”

### 用PrintWriter(字符流)发送数据：

示例：response.getWriter().write(“中国“); Servlet是老外发明的，写字符的编码默认使用的是ISO-8859-1编码，所以会发生乱码

解决办法：

setCharacterEncoding(“UTF-8”);// 告诉response使用Writter写字符的时候以”UTF-8”编码去写，这个方法一定要在获取response的流之前先设置，否则不生效。

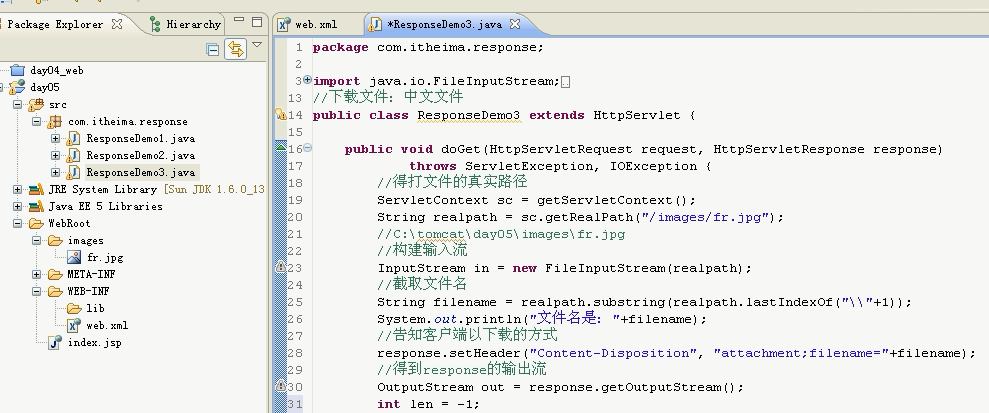
response.setHead(“Context-type”,”text/html;charset=UTF-8”);//告诉客户端以UTF-8编码读取

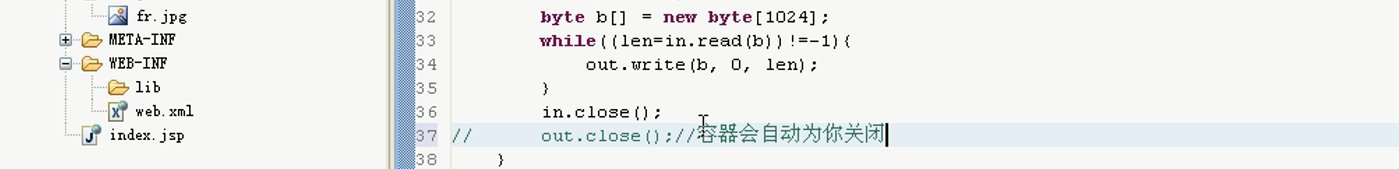
由于经常改动编码，response提供了一种更简单的方式：

response. setContentType(“text/html;charset=UTF-8”);其作用相当于以上两条代码。也就是这个方法内部会调用setCharacterEncoding来设置编码，同时还添加请求头。

实际开发中，使用Writer的方式写出内容比较常用，因为浏览器显示的内容一般就是字符。

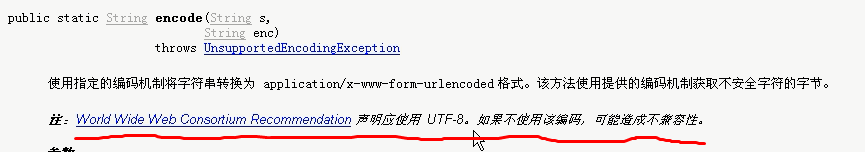
## 文件下载



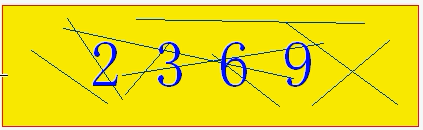
  
上面截取文件名的地方，“+1”需要写到括号外，表示最后一个反斜杠的位置的索引 + 1

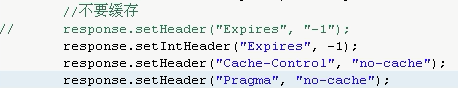
中文文件名的文件下载

* + 问题：文件名乱码或无法下载
  + 解决：将文件名进行编码  
    

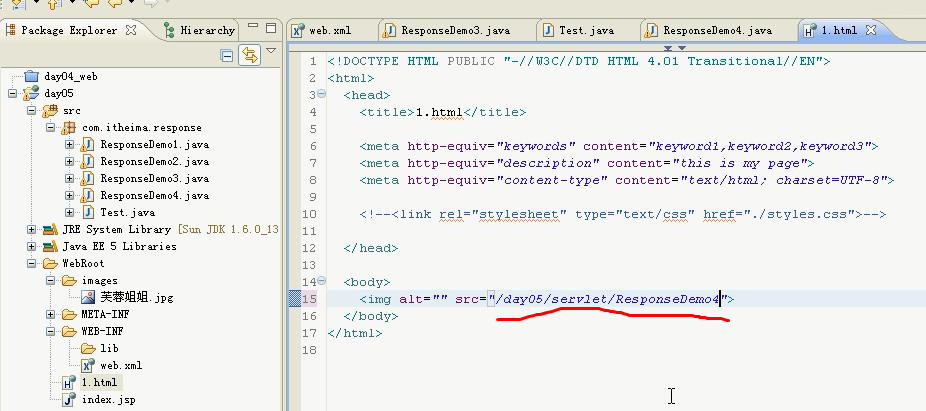
  
  
从API文档可知，W3C告诉我们声明编码时应该使用UTF-8。

## 生成验证码



* 随机图片(专业术语叫CAPTCHA图像):Completely Automated Public Turing Test to Tell Computers and Humans Apart (全自动区分计算机和人类的测试)
* 相关主要类(JDK 查看API)
  + BufferedImage:代表一幅内存图像
  + Graphics：代表画笔
  + ImageIO：用于输出图像
  + 放在html页面上<img src/>
* 注意：浏览器默认会缓存图片，所以当用户点击IE上的刷新按钮时，验证码图片并不会重新请求，而是使用缓存的图片，所以需要告诉IE不要缓存：
  + response.addHeader("Pragma", "no-cache");
  + response.setHeader("Cache-Control", "no-cache");
  + response.setHeader("Expires", "0");  
    

在html中使用到图片：



在html中有一个img标签，src指向了一个Servlet，不论是一个静态资源图片还是一个Servlet，都是向服务器发一个请求，当请求的是Servlet时，则可以通过Servlet动态的返回图片内容，可以返回一个静态的图片，也可以动态的创建一张图片并返回。  
Servlet代码如下：  


完善Html网页：

  
点击看不清时需要刷新验证码，就是重新请求图片的地址，但是如果图片地址不变，浏览器不会重新请求，所以需要动态的修改请求的url，可以在url后面通过?连接一些参数，这些参数就随便写了，如使用时间，这样每次发送的请求图片url都是不一样的，因为参数值不一样。  


## 通过response实现请求重定向

* 通过response实现请求重定向。
  + 请求重定向指：一个web资源收到客户端请求后，通知客户端去访问另外一个web资源，这称之为请求重定向。
* 地址栏会变，共发送2次请求，增加服务器负担
* 实现方式
  + response.sendRedirect()
  + 实现原理：
    - 302/307状态码和location头即可实现重定向