1. **Response概述**

代表Http响应的对象.用来封装服务器将要发送给浏览器的响应信息.

* 1. **response继承结构**

ServletResponse接口 – 提供了一个response对象应该具有的功能

|

|-- HttpServletResponse接口，继承了ServletResponse，并添加了很多和Http协议相关的方法

* 1. **response提供的常用方法**

状态行

HTTP/1.1 200 OK

若干响应头

xxx : xxx

响应实体内容

xxx

**提供的常用方法有:**

（1）设置状态码的方法

setStatus(int status)

（2）设置响应头的方法

setHeader(String name，String value)

（3）设置响应实体内容的方法

getOutputStream()

getWriter()

# response对象提供的功能

## 向客户端发送数据

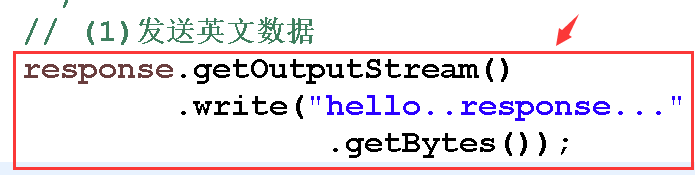
getOutputStream()

getWriter()

### getOutputStream（）方法

**1、使用字节流(getOutputStream())向客户端发送数据**

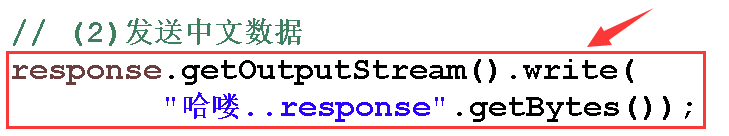
（1）使用字节流发送英文数据



在浏览器中:



（2）使用字节流发送中文数据



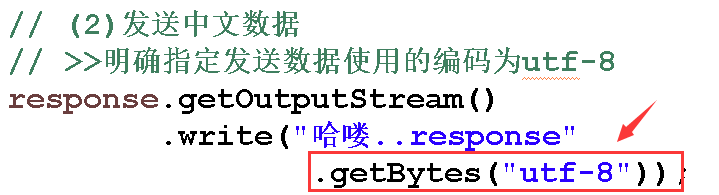
**2、在发送中文数据时出现了乱码，那么乱码产生的原因:**

（1）服务端在发送数据，数据使用的编码是GBK(字符串转成二进制数组调用的是getBytes()，这个方法如果不指定，将会使用默认的平台码GBK)

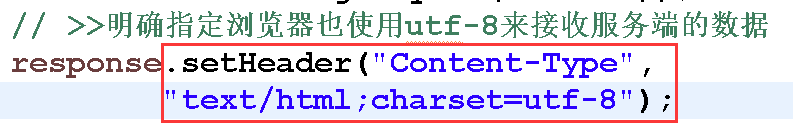
（2）浏览器自己有默认的编码(比如谷歌)，如果不指定，默认使用utf-8来接收数据. 发送数据和接收数据使用的码表不一致造成乱码。

3、**解决方案:**

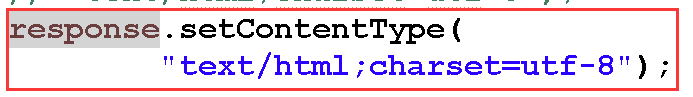
第一步:设置服务器端发送的数据使用的编码是UTF-8



第二步:设置浏览器接收数据时使用的编码为UTF-8



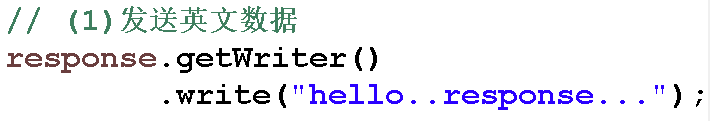
等价于<====>



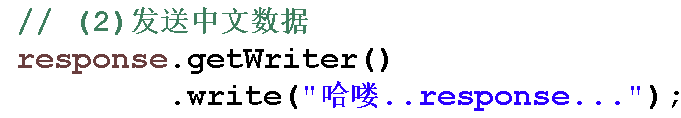
### getWriter（）

**1、使用字符流(getWriter())向客户端发送数据**

（1）使用字符流发送英文数据



（2）使用字符流发送中文数据



在浏览器中:



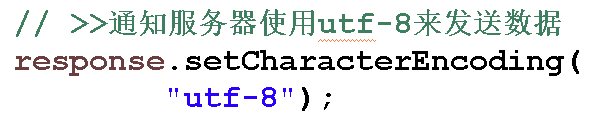
**2、乱码产生的原因:**

（1）服务器端使用字符流发送数据，底层需要将字符转成字节，这时需要使用一个码表，如果不指定，服务器默认使用iso8859-1编码，而这个编码中没有中文汉字，所以必然会出现乱码!

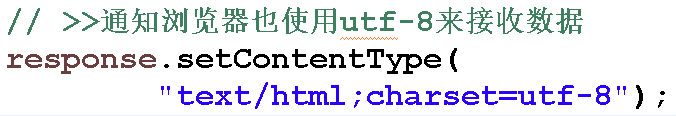
（2）无论客户端使用什么编码，接收到的数据都是乱码!

**3、解决的方案:**

第一步: 通知服务端使用utf-8来发送数据



第二步: 通知浏览器也使用utf-8来接收数据



**4、总结:** 其实在解决字符流发送中文乱码问题的过程中，

response.setCharacterEncoding("utf-8");

-- 这行代码可以省略不写，原因是因为

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

-- 这行代码有两个作用，既会通知服务器用utf-8来发送数据，也会通知浏览器使用utf-8来接收数据!

### 两个方法的细节问题

（1）在一个请求中，getWriter()和getOutputStream()不能同时使用，因为这两个方法是互斥的! 如果两个方法同时使用，会抛出异常

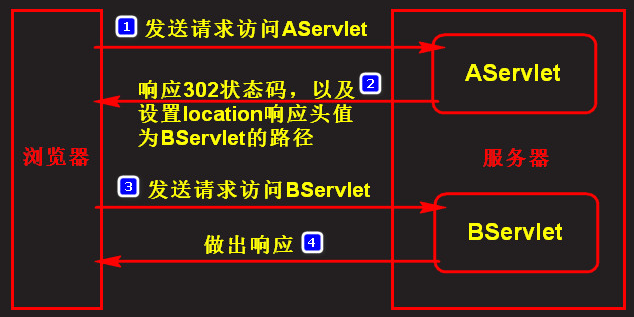


（2）通过getWriter()和getOutputStream()这两个方法获取到的流，使用完后不需要关闭，因为服务器会自动帮我们关闭!!

（3）通过getWriter()和getOutputStream()这两个方法获取到的流，是指向response缓冲区，而不是直接指向了浏览器!

## 实现请求重定向

### 什么是请求重定向



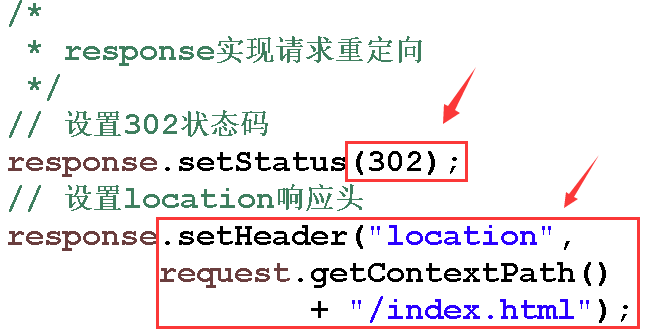
请求重定向: 可以实现资源的跳转

### 实现请求重定向

通过302状态码和location响应头可以实现重定向

response.setStatus(302);

response.setHeader(“location”，“/index.html”);



等价于 <=========>

response.sendRedirect(“/day15/index.html”); 这行代码也可以实现重定向操作

### 重定向的特点:

（1）两次请求，两次响应(request对象不是同一个)

（2）地址栏地址会发生变化

（3）既可以在同一个WEB应用内部的资源之间进行跳转，也可以在不同的WEB应用或者是不同的虚拟主机之间进行跳转

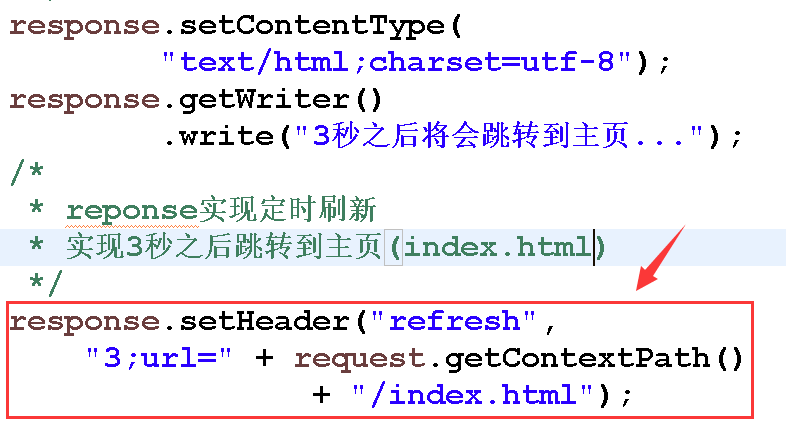
## 实现定时刷新

### 什么是定时刷新

通过refresh头可以实现在多少秒之后跳转到指定的资源

### 实现定时刷新

实现定时刷新的代码如下:



### 定时刷新的特点

（1）两次请求，两次响应

（2）request对象不是同一个

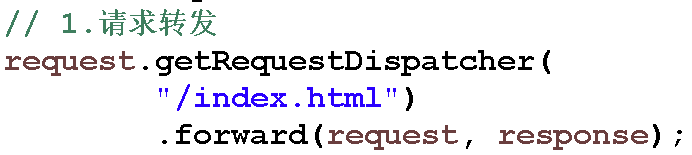
（3）地址栏地址会发生变化

（4）和重定向的区别在于，重定向是立即发生跳转，而定时刷新是在指定多少秒之后立即跳转，在跳转之前，可以向浏览器发送响应，并维系一段时间.

## 三种资源跳转方式的总结

### 请求转发、重定向、定时刷新三种资源跳转方式的区别

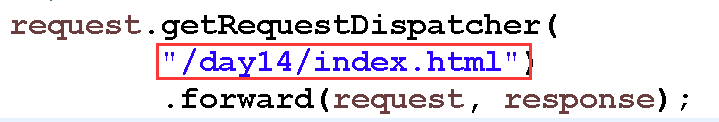
**1、请求转发**

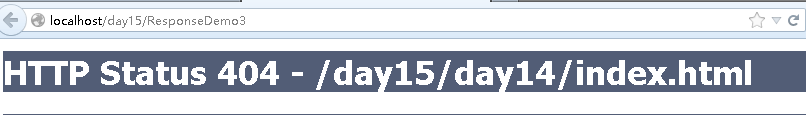


（1）一次请求，一次响应（request对象是同一个）

（2）地址栏地址不会发生变化

（3）转发只能在同一个WEB应用内部的资源之间进行跳转，不能在不同的WEB应用或者是不同的虚拟主机之间进行跳转。否则将会提示找不到资源.



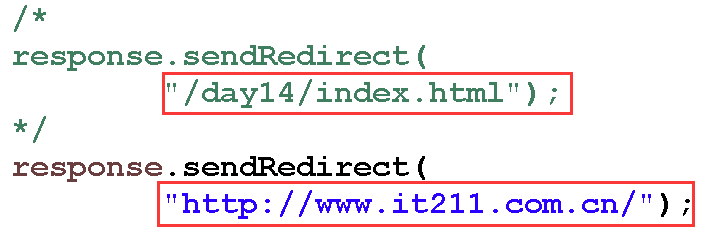


**2、重定向**

（1）两次请求，两次响应(request对象不是同一个)

（2）地址栏地址会发生变化

（3）既可以在同一个WEB应用内部的资源之间进行跳转，也可以在不同的WEB应用或者是不同的虚拟主机之间进行跳转，如下:

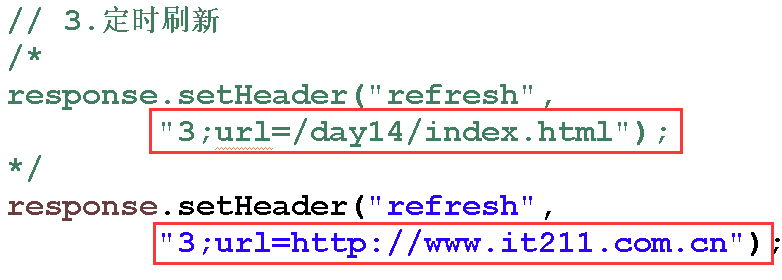


**3、定时刷新**

（1）两次请求，两次响应(request对象不是同一个)

（2）地址栏地址会发生变化

（3）既可以在同一个WEB应用内部的资源之间进行跳转，也可以在不同的WEB应用或者是不同的虚拟主机之间进行跳转，如下:



（4）定时刷新和重定向主要的区别在于，重定向是立即跳转,而定时刷新是在指定多少秒之后立即跳转，并且在跳转之前,可以向浏览器发送响应数据并维系一段时间.

### 请求转发、重定向、定时刷新分别在什么情况下使用?

**1、如果是同一个WEB应用内部资源的跳转，三种方式都可以，具体使用哪一种:**

（1）如果希望在跳转之后地址栏地址不会发生变化，这里只能使用请求转发

（2）如果希望在跳转之后地址栏地址会发生变化，这里可以使用重定向或定时刷新

（3）如果在跳转的过程中，希望带数据到目的地，这里只能使用请求转发

（4）如果没有什么特别的需求，仅仅是做一个跳转，推荐使用转发. 因为转发是一次请求，可以减少访问服务器的次数，减少服务器的压力!

**2、如果是不同的WEB应用之间资源的跳转，只能使用重定向和定时刷新，具体使用哪一种:**

重定向是立即跳转，中间没有间隔，而定时刷新可以指定多少秒之后再进行跳转，在跳转之前，还可以发送响应到客户端，并维系一段时间.

（1）因此如果需要指定多少时间之后再跳转可以使用定时刷新，如果在跳转之前，需要提示用户，可以使用定时刷新.

（2）如果没有什么特殊需要，两种方式都可以.

1. **ServletContext对象**
   1. **ServletContext概述**
      1. **ServletContext对象简介**

代表整个WEB应用的对象

在服务器启动并加载WEB应用之后，服务器会立即创建代表当前WEB应用的ServletContext对象，该对象会一直驻留在服务器的内存中，唯一的代表当前WEB应用. 在服务器关闭或者WEB应用被移出容器，随着WEB应用的销毁，ServletContext对象也跟着销毁.

* + 1. **获取ServletContext对象**

ServletContext context = this.getServletContext();

* 1. **作为域对象使用**
     1. **作为域对象使用**

1、域对象: 如果一个对象具有可以被看见的范围，利用该对象上的map可以在整个范围内，实现数据的共享.

2、常用方法:

setAttribute(String name，Object value);

getAttribute(String name);

removeAttribute(String name);

getAttributeNames();

3、ServletContext对象的特征

（1）生命周期: 在WEB应用被加载后，服务器立即创建代表当前WEB应用的ServletContext对象，在WEB应用被移出容器时，随着WEB应用的销毁，对象也跟着销毁!

（2）作用范围: 整个WEB应用

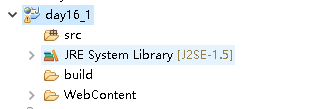
（3）功能: 在整个WEB应用中实现数据的共享

* + 1. **案例：统计网站访问人数.**

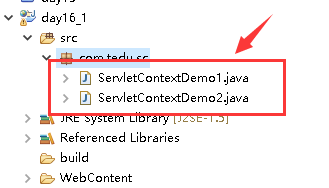
需求: 统计本网站访问的人数，当用户访问本网站时，响应该用户是本网站今天第几位访客!

实现步骤:

1. 创建一个工程(day16\_1)



1. 创建两个Servlet: ServletContextDemo1和ServletContextDemo2



1. 在两个Servlet中分别实现:当用户访问时，响应该用户是本网站的第几位访客

