**一、重定向与请求转发的区别**

1.重定向时浏览器上的网址会发生改变；转发时浏览器上的网址不变。

转发过程：客户浏览器发送http请求->web服务器接受此请求->调用内部的一个方法在容器内部完成请求处理和转发动作->将目标资源发送给客户；在这里，转发的路径必须是同一个web容器下的url，其不能转向到其他web路径上去，中途传递的是自己容器内的request。在客户浏览器路径栏显示的仍然是第一次访问的路径，也就是说客户是感觉不到服务器做了转发的，转发行为是浏览器只做了一次访问请求。

重定向过程：客户浏览器发送http请求->web服务器接受后发送302状态码响应及对应新的location给客户浏览器->客户浏览器发现时302响应，则自动再发送一个新的http请求，请求url是新的location地址->服务器根据此请求寻找资源并发送给客户浏览器。在这里location可以重定向到任意url，既然是浏览器重新发出了请求，则就没有什么request传递的概念了。在客户浏览器路径栏显示的是其重定向的路径，客户可以观察到地址的变化的。重定向行为是浏览器做了至少两次的访问请求的。

2.重定向实际上产生了两次请求，转发只有一次请求。

重定向的两次请求：

第一次：客户端请求，A服务器响应，并response回来，告诉浏览器，你应该去找B。这个时候浏览器可以看到地址变了，而且历史的回退按钮也亮了。重定向可以访问自己web应用以外的资源。在重定向的过程中，传输的资源会被丢失。

Eg：response.sendRedirect("loginsuccess.jsp");

请求转发：

请求转发是服务器内部把对一个request/response的处理权，移交给另外一个

对于客户端而言，它只知道自己最早请求的那个A，而不知道中间的B，甚至C、D。传输的消息不会丢失。

Eg: RequestDispatcher dis=request.getRequestDispatcher(“loginsuccess.jsp”);

       dis.forward(request,response);

3.重定向的网址可以使任何网址，转发的网址必须是本站点的网址。

4.重定向与请求转发的使用：

前后两个页面有数据传递用请求转发，没有则用重定向。

比如servlet查询了数据需要在页面显示，就用请求转发。

比如servlet做了update操作跳转到其他页面就用重定向。

1. **JavaEE中的中文乱码处理**

**浏览器正常显示**

法一：

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

法二：

response.setCharacterEncoding("utf-8");

respnse.setHeader(“Context-Type”,”text/html;charset=utf-8”);

**服务器端正常接收**

中文乱码有三种情况

1）表单提交(get方式，post方式)：input表单输入为中文

2）超链接(get):链接？参数名=值（中文）

3）sendRedict(get): 链接?参数名=值（中文）

乱码产生的根本原因

请求发送时：浏览器有默认的编码，经上述设置之后，是utf-8。

web服务器接收时候，也有一种编码，由于服务器一般是国外开发的，因此默认的编码方式一般是iso-8859-1

浏览器与服务器的编码方式不一致，因而产生乱码

问题解决方法

1，如果是表单提交，方法是post（表单提交应当尽量选择post）

此时通过http请求体传递参数，因为服务器会将对应的http请求头解析并进一步包装为request对象，因此利用request.setCharacterEncoding方法，指定web服务器接受端的编码方式，如request.setCharacterEncoding("utf-8")

2，如果是表单提交，方法是get

此时通过链接传递参数，针对乱码情况，可以利用

String value=new String(request.getParameter（"参数名").getBytes("iso-8859-1"),"utf-8")来解决

这种情况可以封装成一个工具类

 3，超链接乱码，sendRedict的乱码本质上也是get请求，解决方法同2

4，如果是ie6及以下版本，上述方法中涉及到get方法时依然有可能出现乱码，此时的解决方法是：

对要发送的中文进行编码

String info=java.net.URLEncoder.encode("你好","utf-8");

response.sendRedict("url路径?参数名="+info);

第三个文件接收

String info=new String(request.getParameter("参数名").getBytes("iso-8859-1"),"utf-8");

**补充关于乱码的相关知识点：**

一、当我们下载文件的时候，可能提示框是中文乱码，此时的解决方法是：

String temp=java.net.URLEncoding.encode("中文文件名","utf-8")

response.setHeader("Content-Disposition","attachment;filename="+temp);

二、request.getRequestDispatcher()之后出现乱码

对于某一个页面，在单独访问的时候是正常的，可是讲过上述语句转向之后出现乱码

 在Servlet中，一般有参数传递的话，会设置页面接收参数和传递参数的编码。即下面两句：

       request.setCharacterEncoding("UTF-8");  
       response.setCharacterEncoding("utf-8");

  一般情况下，大部分都会想到使用这个，但是这两句代码的位置有时却容易被忽视，正确的写法是，request.setCharacterEncoding("UTF-8");    response.setCharacterEncoding("utf-8");要放在PrintWriter out = response.getWriter();的前面。因为out对象初始化之后，再设置编码已经没有任何意义了！所以必须在out对象初始化之前进行编码的设置。