# Servlet常用API

## HttpServletRequest 请求对象

setAttribute(String key, Object value) // 设置key对应值value

getAttribute(String key) // 通过key获取请求对象的值

setCharacterEncoding() // 设置请求对象的编码格式

getInputStream() // 获取在servlet中的输入流对象

getParameter(String key)

key 对应前端页面提交的元素的name属性的值

eg: <input name=”aaa” /> key == aaa

获取文本框的值: getParameter(“aaa”);

URL地址传参数方式: http://www.baidu.com?id=EYEDHDH3434&name=Kkdfj\_343k4jk

获取id的值: getParameter(“id”);

获取name的值: getParameter(“nmae”);

getParameterNames() // 获取到有参数名称组成一个枚举类型对象

getParameterValues(String key) // 获取页面中相同元素名称的值组成的数组

getParameterMap() // 获取由元素的名称和值组成的Map集合

getRequestDispatcher(String url) // 获取到服务内部的转发器对象【url 转发方式访问路径】

getRequestURI() // 获取请求的资源标识符

getMethod(); // 获取请求提交的方式: POST | GET

getCookies(); // 获取本地中所有cookie对象组成数组

## HttpServerResponse 响应 对象

addCookie(Cookie var1); // 把服务端生成cookie对象添加到客户响应

getCharacterEncoding() // 获取响应对象设置的编码格式

setCharacterEncoding(String valu) // 设置响应对象的编码格式

setContentType(“text/html;charset=编码格式[UTF-8,GBK,GB2312]”) // 设置响应对象在网页中的编码格式

getWriter() // 字符输出流

getOutputStream()// 获取输出流

sendRedirect(String url); // 跳转页面，重定向页面

## 重定向（Redirect）和转发(forward)的区别?

1、重定向地址栏发生改变，转发地址栏不会变化。

2、重定向中request中存储的数据会丢失，而转发request对象中的数据不会出现丢失问题。

3、重定向不能直接重定向到服务器内部资源，但是转发可以访问服务器内部的资源

4、重定向发出的网络请求都是2次及以上，转发在网络请求只会有一次

req.getRequestDispatcher(**"ui"**).forward(req,resp);

resp.sendRedirect(**"ui"**);

## 乱码处理问题解决方案

1、如果表单提交出现乱码问题。

a、表单未设置method属性为post

b、servlet中没有设置编码格式: setCharacterEncoding(“UTF-8”)

c、你创建项目的时候使用编码格式不是UTF-8， 但是你的网页编码格式设置了UTF-8， 在serlvet中setCharacterEncoding(“UTF-8”)设置编码， 乱码仍然存在。

解决方案: 重写建立项目.

通过notepad++把文件的编码格式设置为UTF-8 格式文件。

创建项目初期，一定需要统一项目编码格式（页面，java文件，servlet，过滤器中设置的编码格式必须完全一致）

2、设置服务器解析出来的数据的编码(不推荐)【最有效】

3、编码格式的处理原则：

如果是POST请求的编码格式可以批量处理。

只需要在获取请求的数据之前，设置编码格式就可以了

-------------------------------------------------------------------------------

如果使用GET请求，编码格式只能一个一个的处理。

String account = request.getParameter(“account”);

account = new String(account.getBytes(“ISO-8859-1”),UTF-8);

4、在servlet中正确设置编码格式的方式:

*// 设置请求对象的编码格式*req.setCharacterEncoding(**"UTF-8"**);  
*// 设置响应对象的编码格式  
// resp.setCharacterEncoding("UTF-8");*resp.setContentType(**"text/html;charset=UTF-8"**);

# Web会话

## Cookie会话

Cookie是一个文件，文件必然涉及到完整性、大小限制、安全问题、容易被截取。

Cookie不同的浏览器存储方法不同，IE存储cookie,在其他浏览器上是无法获取的

Cookie主要作用是：是主要用来保存Session会话的

key value

JSESSIONID = ACAD89074E40B0DE5EE07589409D1D1F

## Session会话

（Session跨越问题）

http协议：无状态协议=>服务端是无法知道客户端的状态的，

会话的结束，不代表Session对象销毁。

Cookie被禁用：

如果解决Cookie被禁用，导致回话不一致问题：

答：可以通过重写URL的方式，解决Cookie被禁用的问题

表单和超链接：encodeURL

重定向连接：encodeRedirectURL

Session中比较常用的方法：  
一 设置session过期时间（默认时间为30分钟）

在web.xml里设置

<**session-config**>  
 *<!--以分钟为单位-->* <**session-timeout**>120</**session-timeout**>  
</**session-config**>

二 调用session.setMaxInactiveInterval(120);//以秒为单位

三 让整个会话中的session全部失效

Session.invalidate();//就是 安全退出 效果

四 移除会话中指定key 的对象

Session.removeAttribute(“account”)

Session 使用的原则：session中保存的数据只是用户在服务器端的凭证而已

Session就是在服务器端记录用户的一个状态而已。

# Servlet线程安全

1 在servlet中不要定义成员变量（最推荐的方式）

2 加锁synchronized/lock(),是可以解决线程安全的问题，但是对于系统开销非常大

3 实现SingleThreadModel接口，这个每次访问的时候都会开启一个新的Servlet线程来访问

# JSP(java server pages)[面试问题]

## JSP基本结构（静态化）

实质：就是一个Servlet

页面指令：告诉转义后的servlet的编码格式，指定转义语言

<%@ **page contentType**="**text/html;charset=UTF-8**" **language**="**java**" %>

<%! 成员变量/方法 %>声明

<% %>小脚本，可以把里面逻辑放入方法里

<%= %>表达式（使用方法和变量的）

Servlet 处理业务，JSP和用户交互

为什么首次访问JSP页面的时候比较慢，而第二次后就非常快？

答：因为第一次JSP需要转义为java源文件，编译为.class，在jvm上面执行

## JSP中九大内置对象

四大域对象：application/session/request/pageContext(由强到弱)

Out,config,exception,page,response

# EL表达式

。。。。

# JSTL

…

# Filter

…

# Listener(事件)

….

# MVC

…

# Aajx

…

# 封装框架SpringMVC框架（把javaweb学习的内容进行综合的应用）