Servlet面试：<https://blog.csdn.net/Jeff_Seid/article/details/80761076>

JSP面试：<https://blog.csdn.net/qq_21439971/article/details/60955533>

## Servlet

## Servlet生命周期

Servlet生命周期包括三部分：

初始化：Web容器加载servlet，调用init()方法

处理请求：当请求到达时，运行其service()方法。service()自动派遣运行与请求相对应的doXXX（doGet或者doPost）方法。

销毁：服务结束，web容器会调用servlet的distroy()方法销毁servlet。

### Servlet生命周期可分为5个步骤

加载Servlet。当Tomcat第一次访问Servlet的时候，Tomcat会负责创建Servlet的实例

初始化。当Servlet被实例化后，Tomcat会调用init()方法初始化这个对象

处理服务。当浏览器访问Servlet的时候，Servlet 会调用service()方法处理请求

销毁。当Tomcat关闭时或者检测到Servlet要从Tomcat删除的时候会自动调用destroy()方法，让该实例释放掉所占的资源。一个Servlet如果长时间不被使用的话，也会被Tomcat自动销毁

卸载。当Servlet调用完destroy()方法后，等待垃圾回收。如果有需要再次使用这个Servlet，会重新调用init()方法进行初始化操作。

简单总结：只要访问Servlet，service()就会被调用。init()只有第一次访问Servlet的时候才会被调用。destroy()只有在Tomcat关闭的时候才会被调用。

## get提交和post提交有何区别

get参数传递通过url，post放在resquest body中；

get请求参数有长度限制1k，post没有

get不安全，因为参数暴露在url中。

get请求只能进行url编码。post支持多种编码

get请求参数会被完整保留在浏览历史记录，而post不会被保留。

get和post本质就是tcp链接，并无差别。但是由于HTTP的规定和浏览器/服务器的限制，导致他们在应用过程中体现出一些不同。

get产生一个tcp数据包；post产生两个tcp数据包。

## JSP与Servlet有什么区别

jsp在本质上就是servlet，但是两者的创建方式不同，servlet都是java程序代码构成，用于流程控制和事务处理,而jsp由html和jsp标签构成，可以方便地编写动态网页。jsp和servlet主要有两方面不同：编译：jsp修改后就可立刻看到效果，不需要编译，而servlet需要编译。转换：jsp是动态网页开发技术是运行在服务端的脚本语言，而servlet是web服务器的编程技术，所以jsp运行时就是转换为servlet。

## doGet与doPost方法的两个参数是什么

HttpServletRequest：封装了与请求相关的信息

HttpServletResponse：封装了与响应相关的信息

## forward和redirect的区别

转发与重定向

（1）从地址栏显示来说

forward是服务器请求资源,服务器直接访问目标地址的URL,把那个URL的响应内容读取过来,然后把这些内容再发给浏览器.浏览器根本不知道服务器发送的内容从哪里来的,所以它的地址栏还是原来的地址.

redirect是服务端根据逻辑,发送一个状态码,告诉浏览器重新去请求那个地址.所以地址栏显示的是新的URL.

（2）从数据共享来说

forward:转发页面和转发到的页面可以共享request里面的数据.

redirect:不能共享数据.

（3）从运用地方来说

forward:一般用于用户登陆的时候,根据角色转发到相应的模块.

redirect:一般用于用户注销登陆时返回主页面和跳转到其它的网站等.

（4）从效率来说

forward:高.

redirect:低.

## 获取页面的元素的值有几种方式，分别说一下

request.getParameter() 返回客户端的请求参数的值

request.getParameterNames() 返回所有可用属性名的枚举

request.getParameterValues() 返回包含参数的所有值的数组

## request.getAttribute()和request.getParameter

（1）有setAttribute，没有setParameter方法

（2）getParameter获取到的值只能是字符串，不可以是对象，而getAttribute获取到的值是Object类型的。

（3）通过form表单或者url来向另一个页面或者servlet传递参数的时候需要用getParameter获取值；getAttribute只能获取setAttribute的值

（4）setAttribute是应用服务器把这个对象放到该页面所对应的一块内存当中，当你的页面服务器重定向到另一个页面的时候，应用服务器

会把这块内存拷贝到另一个页面对应的内存当中。通过getAttribute可以取得你存下的值，当然这种方法可以用来传对象。

用session也是一样的道理，这是说request和session的生命周期不一样而已。

## Servlet的基本架构

public class ServletName extends HttpServlet {

  public void doPost(HttpServletRequest request, HttpServletResponse response) throws

      ServletException, IOException  {

      }

  public void doGet(HttpServletRequest request, HttpServletResponse response) throws

      ServletException, IOException  {

      }

}

## 什么情况下调用doGet()和doPost()？

JSP页面中的form标签里的method属性为get时调用doGet()，为post时调用doPost()；超链接跳转页面时调用doGet()

## Servlet执行时一般实现哪几个方法？

public void init(ServletConfig config)

public ServletConfig getServletConfig()

public String getServletInfo()

public void service(ServletRequest request,ServletResponse response)

public void destroy()

## jsp有九大内置对象

request ：包含用户端请求的信息

response：包含服务器传回客户端的响应信息

session：与请求有关的会话期

pageContext：管理网页属性

application：服务器启动创建，服务器关闭停止，为多个应用程序保存信息

out：向客户端传输数据

config：servlet的架构组件

page：指网页本身

exception：针对错误页面才可使用

## 四大作用域

page：一个页面

request：一次请求

session：一次会话

application：服务器从启动到停止

## JSP中动态INCLUDE和静态INCLUDE有什么区别

include指令用于把另一个页面包含到当前页面中，在什么时候包含的？再转换成servlet的时候包含进去的。

动态INCLUDE用jsp:include动作实现 <jsp:include page="included.jsp" flush="true" />它总是会检查所含文件中的变化,适合用于包含动态页面,并且可以带参数.

静态INCLUDE用include伪码实现,定不会检查所含文件的变化,适用于包含静态页面<%@ include file="included.htm" %>

## 什么是HTTP隧道？

HTTP隧道是一种利用HTTP或者是HTTPS把多种网络协议封装起来进行通信的技术。因此，HTTP协议扮演了一个打通用于通信的网络协议的管道的包装器的角色。把其他协议的请求掩盖成HTTP的请求就是HTTP隧道

## 什么是cookie？

网页之间的交互是通过HTTP协议传输数据的，而Http协议是无状态的协议。无状态的协议是什么意思呢？一旦数据提交完后，浏览器和服务器的连接就会关闭，再次交互的时候需要重新建立新的连接。

服务器无法确认用户的信息，于是乎，W3C就提出了：给每一个用户都发一个通行证，无论谁访问的时候都需要携带通行证，这样服务器就可以从通行证上确认用户的信息。通行证就是Cookie

## Session和cookie有什么区别？

从存储方式上比较

Cookie只能存储字符串，如果要存储非ASCII字符串还要对其编码。

Session可以存储任何类型的数据，可以把Session看成是一个容器

从隐私安全上比较

Cookie存储在浏览器中，对客户端是可见的。信息容易泄露出去。如果使用Cookie，最好将Cookie加密

Session存储在服务器上，对客户端是透明的。不存在敏感信息泄露问题。

从有效期上比较

Cookie保存在硬盘中，只需要设置maxAge属性为比较大的正整数，即使关闭浏览器，Cookie还是存在的

Session的保存在服务器中，设置maxInactiveInterval属性值来确定Session的有效期。并且Session依赖于名为JSESSIONID的Cookie，该Cookie默认的maxAge属性为-1。如果关闭了浏览器，该Session虽然没有从服务器中消亡，但也就失效了。

从对服务器的负担比较

Session是保存在服务器的，每个用户都会产生一个Session，如果是并发访问的用户非常多，是不能使用Session的，Session会消耗大量的内存。

Cookie是保存在客户端的。不占用服务器的资源。像baidu、Sina这样的大型网站，一般都是使用Cookie来进行会话跟踪。

从浏览器的支持上比较

如果浏览器禁用了Cookie，那么Cookie是无用的了！

如果浏览器禁用了Cookie，Session可以通过URL地址重写来进行会话跟踪。

从跨域名上比较

Cookie可以设置domain属性来实现跨域名

Session只在当前的域名内有效，不可夸域名

## Servlet安全性问题

由于Servlet是单例的，当多个用户访问Servlet的时候，服务器会为每个用户创建一个线程。当多个用户并发访问Servlet共享资源的时候就会出现线程安全问题。

原则：如果一个变量需要多个用户共享，则应当在访问该变量的时候，加同步机制synchronized (对象){}

如果一个变量不需要共享，则直接在 doGet() 或者 doPost()定义.这样不会存在线程安全问题

## HTTP响应的结构是怎么样的？

HTTP响应由三个部分组成：

状态码(Status Code)：描述了响应的状态。可以用来检查是否成功的完成了请求。请求失败的情况下，状态码可用来找出失败的原因。如果Servlet没有返回状态码，默认会返回成功的状态码HttpServletResponse.SC\_OK。

HTTP头部(HTTP Header)：它们包含了更多关于响应的信息。比如：头部可以指定认为响应过期的过期日期，或者是指定用来给用户安全的传输实体内容的编码格式。如何在Serlet中检索HTTP的头部看这里。

主体(Body)：它包含了响应的内容。它可以包含HTML代码，图片，等等。主体是由传输在HTTP消息中紧跟在头部后面的数据字节组成的。

## 一个 JSP 的生命周期包含以下步骤：

编译：当浏览器请求一个 JSP 时，JSP 引擎首先检查是否需要编译这个页面。如果这个页面从来没有被编译过，或者 JSP 在最后一次被编译后已经被修改了，那么 JSP 引擎会编译这个页面。

编译过程包括三个步骤：

解析 JSP。

将 JSP 转换成一个 servlet。

编译这个 servlet。

初始化：当一个容器加载一个 JSP 时，在修改任何请求之前会调用 jsInit() 方法。

执行：无论何时一个浏览器请求一个 JSP 时，这个页面都会被加载并且初始化，JSP 引擎在 JSP 中调用 \_jspService() 方法。一个 JSP 的 \_jspService() 方法在每一次请求时都会被调用，并且为该请求负责生成响应，该方法还负责生成所有的7个 HTTP 方法的响应，也就是 GET， POST，DELETE 等等。

清除：当一个 JSP 被一个容器从应用中移除时是 JSP 生命周期的破坏阶段。jspDestroy() 方法相当于 servlets 中的销毁方法。