1、servlet登录流程

<https://www.pianshen.com/article/6466312324/>

1）Web客户向Servlet容器（tomcat）发出Http请求；

2）Servlet容器解析Web客户的Http请求；

3）Servlet容器创建一个HttpRequest对象，在这个对象中封装Http请求信息；

4）Servlet容器创建一个HttpResponse对象；

5）Servlet容器调用HttpServlet的service方法，把HttpRequest和HttpResponse对象作为service方法的参数传给HttpServlet对象；

HttpServlet事实上是servlet的一种子类实例也是最一般的实例。当编写一个servlet时,必须直接或间接实现servlet接口,最可能实现的方法就是扩展javax.servlet.genericservlet或javax.servlet.http.httpservlet，其中genericservlet类提供了servlet接口的基本实现,httpservlet类扩展了genericservlet并且提供了servlet接口中具体于http的实现。

6）HttpServlet调用HttpRequest的有关方法，获取HTTP请求信息；

7）HttpServlet调用HttpResponse的有关方法，生成响应数据；

一般通过HttpServletRequest和HttpServletResponse获取HTTP请求信息和返回响应。事实上servlet理论上可以处理多种形式的请求响应形式 http只是其中之一 所以HttpServletRequest HttpServletResponse分别是ServletRequest和ServletResponse的子类。一般，HttpServlet对应HttpServletRequest和HttpServletResponse。

8）Servlet容器把HttpServlet的响应结果传给Web客户。

2、会话跟踪:cookie+session

https://blog.csdn.net/vv0606/article/details/112826423

1、Cookie数据保存在客户端，Session数据保存在服务器端。

2、Session是由应用服务器维持的一个服务器端的存储空间，用户在连接服务器时，会由服务器生成一个唯一的SessionID,用该SessionID 为标识符来存取服务器端的Session存储空间。而SessionID这一数据则是保存到客户端，用Cookie保存的，用户提交页面时，会将这一SessionID提交到服务器端，来存取Session数据。这一过程，是 不用开发人员干预的。所以一旦客户端禁用Cookie，那么Session也会失效。

3、Cookies是属于Session对象的一种。但有不同，Cookies不会占服务器资源，是存在客服端内存或者一个 Cookie的文本文件中；而Session则会占用服务器资源。所以，尽量不要使用Session，而使用Cookies。但是我们一般认为Cookie是不可靠的，Cookies是保存在本机上的，但是其信息的完全可见性且易于本地编辑性，往往可以 引起很多的安全问题Session是可靠地。但是目前很多著名的站点也都用Cookie。

4、session相对于cookie更安全，cookie一般存储少量不太敏感的数据。

客户端请求服务器，如果服务器需要记录该用户状态，就使用response向客户端浏览器颁发一个Cookie。客户端浏览器会把Cookie保存起来。当浏览器再请求该网站时，浏览器把请求的网址连同该Cookie一同提交给服务器。服务器检查该Cookie，以此来辨认用户状态

3、转发与重定向

https://blog.csdn.net/gaibian0823/article/details/51020543

转发是服务器行为，重定向是客户端行为。

转发：

1.转发不会改变浏览器的地址栏

2.转发共享同一个request

3.转发只能在同一个web应用中使用（不可以跨域）

重定向：

1.重定向会改变浏览器的地址栏

2.重定向不共享request

3.可以重定向到任意URL（可以跨域）

4、过滤器与跟踪器

web 监听器是Servlet 中一种的特殊的类，能帮助开发者监听 web 中的特定事件， 比如 ServletContext，HttpSession，ServletRequest 的创建和销毁；变量的创建、销毁和修改等。 可以在 某些动作前后增加处理，实现监控。例如可以用来统计在线人数等。

监听器有三类8种：

⑴ 监听生命周期：

ServletRequestListener

HttpSessionListener

ServletContextListener

⑵ 监听值的变化：

ServletRequestAttributeListener

HttpSessionAttributeListener

ServletContextAttributeListener

⑶ 针对 session 中的对象:

监听 session 中的 java 对象(javaBean) ，是 javaBean 直接实现监听器 的接口。

Filter 即为过滤，用于在 Servlet 之外对 Request 或者 Response 进行修改。它主要用于对用户请求进行预处理，也可以对 HttpServletResponse 进行后处理。使用 Filter 的完整流程： Filter 对用户请求进行预处理，接着将请求交给 Servlet 进行处理并生成响应，最后 Filter 再 对服务器响应进行后处理。在一个 web 应用中，可以开发编写多个 Filter，这些 Filter 组合 起来称之为一个 Filter 链。 可以通过实现一个叫做javax.servlet.Fileter的接口来实现一个过滤器，其中定义了 三个方法，init(),doFilter(), destroy()分别在相应的时机执行。后期观察生命周期。Filter 的实现只需要两步：

Step1: 编写 java 类实现 Filter 接口，并实现其 doFilter 方法。

Step2: 通过@WebFilter注解设置它所能拦截的资源。 filter接口中有一个 doFilter 方法，当开发人员编写好 Filter，并配置对哪个 web 资源进行拦截后，Web 服务器每次在调用 web 资源的 service 方法之前，都会先调用一下 filter 的 doFilter 方法。因此可以达到如下效果：调用目标资源之前，让一段代码执行。是否调用目标资源（即是否让用户访问 web 资源）。

web 服务器在调用 doFilter 方法时，会传递一个 filterChain 对象进来，filterChain 对象是 filter 接口中最重要的一个对象，它提供了一个 doFilter 方法，开发人员可以根据需求决定 是否调用此方法，调用该方法，则 web 服务器就会调用 web 资源的 service 方法，即 web 资源就会被访问，否则 web 资源不会被访问。（本质是放行，调用doFilter方法后，即请求可以到达资源）