Servlet生命周期

1. 加载和实例化：默认情况下，当Servlet第一次被访问时，由容器创建servlet对象
2. 初始化：在servlet实例化之后，容器将调用servlet的init（）方法初始化这个对象，完成一些如加载配置文件、创建连接等初始化工作。该方法只调用一次
3. 请求处理：每次请求访问servlet时，servlet容器都会调用service方法对请求进行处理
4. 服务终止：当需要释放内存或者容器关闭时，容器就会调用servlet实例的destroy方法完成资源的释放，在destroy方法调用之后，容器会释放这个servlet实例，该实例随后会被java的垃圾回收机制回收

转发：一次请求一次响应，不修改路径

重定向：两次请求两次响应，修改路径，底层第一次响应返回302状态码，请求头key：location value：重定向路径

会话：用户打开浏览器，访问web访问web服务器的资源，会话建立，直到有一方断开连接，会话结束。再一次会话中可以包含多次请求和响应

会话跟踪：一种维护浏览器状态的方法，服务器需要识别多次请求是否来自于同一个浏览器，以便在同一会话的多次请求间共享数据

HTTP协议是无状态的，每次浏览器向服务器请求时，服务器都会将该请求是为新的请求，因此我们需要会话跟踪技术来实现

实现方式：

客户端会话跟踪技术：cookie

服务端会话跟踪技术：session

Cookie

Cookie：客户端会话技术，将数据保存到客户端，以后每次请求都携带Cookie数据进行访问

Cookie基本使用：

创建Cookie对象

Cookie cookie = new Cookie (key,value);

发送cookie

Response.addCookie(cookie);

获取cookie

Request.getCookies();

Cookie是基于http协议

响应头：set-cookie

请求头：cookie

Cookie使用细节：

Cookie存活时间：

默认情况下cookie存在浏览器内存中，当浏览器关闭，内存释放，则cookie被销毁

SetMaxAge（int second）：设置Cookie存活时间

1. 正数：将Cookie写入浏览器所在电脑的硬盘，持久化存储。到时间自动删除
2. 负数：默认值，Cookie在当前浏览器内存中，当历览器关闭，Cookie销毁
3. 0，删除Cookie

Session：

服务端会话跟踪技术：将数据保存到服务端

JavaEE提供了HttpSession接口，来实现一次会话的多次请求间数据共享功能

使用：

获取session对象

HttpSession session = request.getSession();

Session对象功能

setAttribute(key,value);存储数据到session域中

GetAttribute(key)根据key获取值

removeAttribute(key)根据key删除该键值对

Session使用原理：

Session是基于Cookie实现的

当第一次请求时，tomcat会自动向客户端set-cookie，键为jsessionId

当后续同一会话再次请求时，会获取cookie中jsessionid。

使用细节：

session钝化、活化：

服务器重启后，Session中的数据是否还在。

还在

钝化：在服务器正常关闭后，Tomcat会自动将session数据写入硬盘文件中

活化：再次启动服务器后，从文件中加载数据到Session中

※当客户端关闭时，session一定丢失

Session销毁：

默认情况下，无操作，30分钟自动销毁

手动销毁Session.invalidate()

小结：

Cookie和Session都是来完成一次会话内多次请求间数据共享的：

区别：

存储位置：Cookie是将数据存储在客户端，Session是将数据存储在服务端

安全性：cookie不安全，session安全

数据大小：cookie最大3KB，Session无大小限制

存储时间：cookie可以长期存储，session默认30分钟

服务器性能：cookie不占服务器资源，session占用服务器性能