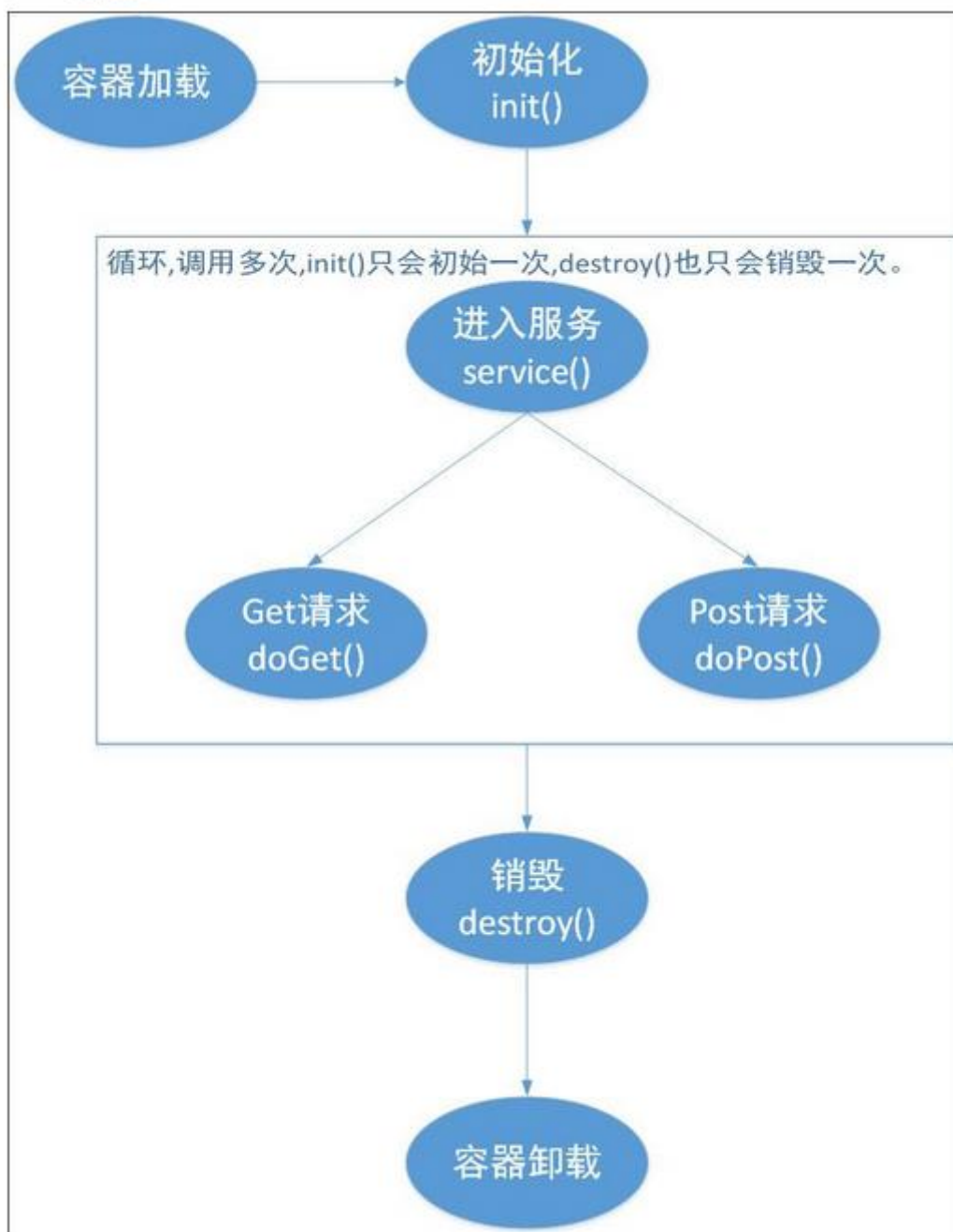


1. 描述Servlet 生命周期

Servlet生命周期



1.加载Servlet

Web容器负责加载Servlet,当web容器启动时或者是在第一次使用这个Servlet时,容器会负责创建Servlet实例,但是必须通过部署描述符(web.xml)指定Servlet的位置,也就是Servlet所在的类名称,成功加载后,web容器会通过反射的方式对Servlet进行实例化。

2. 初始化 (init())

当一个Servlet初始化后,容器将调用init()方法初始化这个对象,初始化的目的是为了let Servlet在处理客户端请求前完成一些初始化的工作,如建立数据库连接,读取资源文件信息等,如果初始化失败,则次Servlet将被直接卸载。

3. 进入服务 (Service---->doGet() && doPost())

当有请求提交时，Servlet将调用service()方法进行处理，常用的是service根据请求类型调用doGet()或者doPost()方法进行处理；在service()方法中，Servlet可以通过ServletRequest接受客户的请求，也可以利用ServletResponse设置响应信息。

4. 销毁 (destroy())

当web容器关闭或者检测到一个Servlet 要从容器中被删除时，会自动调用destroy() 方法，以便让该实例释放掉所占用的资源。

5. 卸载

当一个Servlet调用完destroy()方法后，该实例将等待被垃圾收集器所回收，如果需要再次使用此Servlet时，会重新调用init()方法初始化。

注意：

在正常情况下，Servlet只会初始化一次，而处理服务会调用多次，销毁也只会调用一次；但是如果一个Servlet长时间不使用的話，也会被容器自动销毁，而如果需要再次使用时会重新进行初始化的操作，即在特殊情况下初始化可能会进行多次，销毁也可能进行多次。

对于service()方法，一般来说这个方法是不需要重写的，因为在HttpServlet中已经有了很好的实现，它会根据请求的方式，调用doGet()，doPost()方法，也就是说service()是用来转向的，所以我们一般写一个Servlet，只需要重写doGet()或者doPost()就可以了。如果重写了service()方法，那么Servlet容器就会把请求交给这个方法来处理，倘若你重写的service()方法没有调用doXXX()，即使你在Servlet中又重写了其它doGet()，doPost()等也是不会被调用的，因为Servlet的service()被自动调用（就像init()和destroy()方法一样），所以如果你由于某种需要，需要重写service()方法，并且根据不同的method调用doPost()，doGet()方法时，就要在末尾加上一句super.service()，这样就可以解决问题了。

2.Get Post 的区别

(1) Http定义了与服务器交互的不同方法，最基本的方法有4种：Get、Post、Put、Delete。

(2) URL全称是资源描述符，我们认为：一个URL地址，它用于描述一个网络上的资源，而HTTP中的Get、Post、Put、Delete、就对应着这个资源的查，改，增，删 4个操作。

(3) Get 用于信息的获取，获取资源；URL跳转的方式来请求；doGet方法提交表单的时候会在URL后边显示提交的的内容，所以不安全。

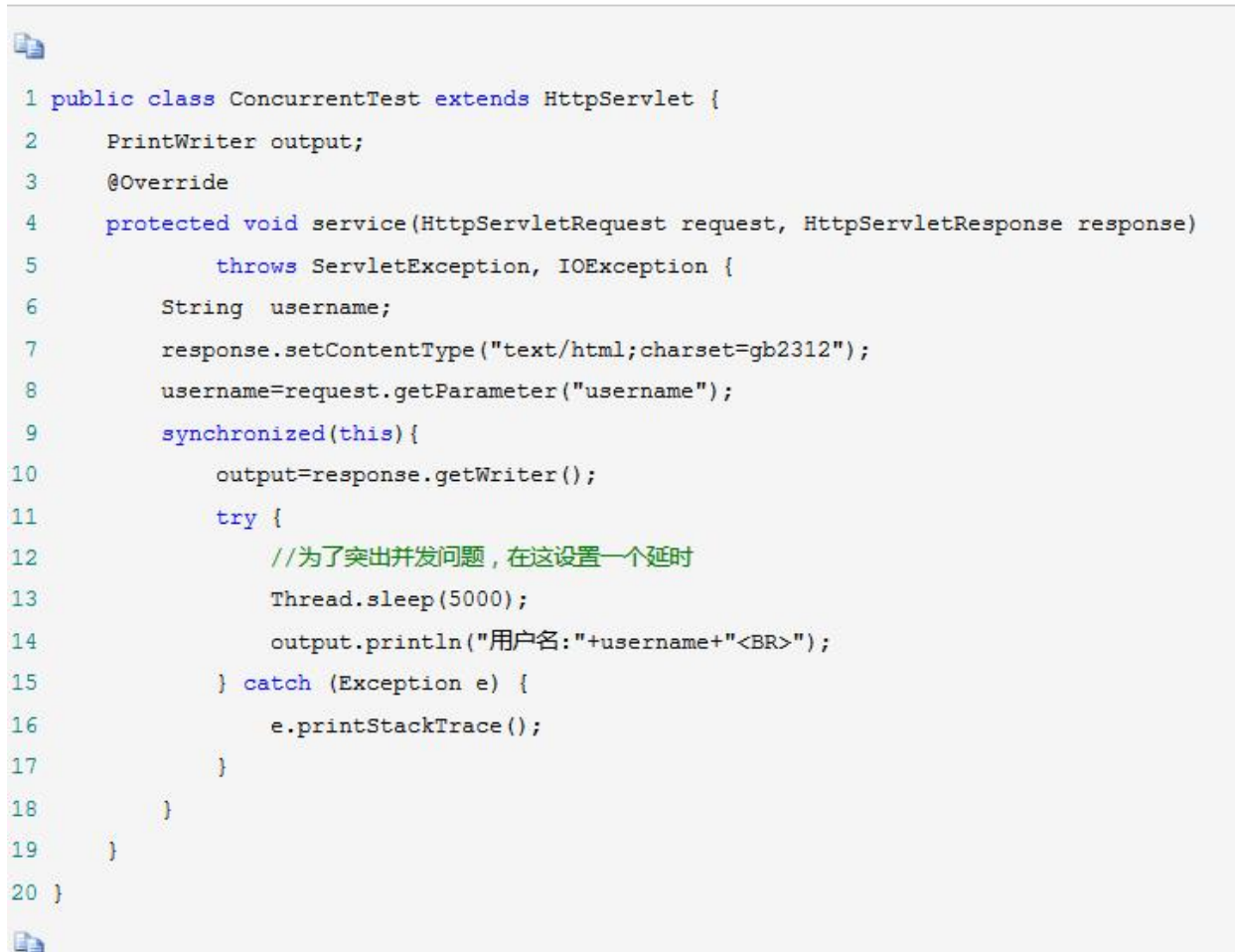
(4) Post用于保存于更新资源；采用表单提交方式来请求。

3.如何处理Servlet线程不安全问题?

答：为什么不安全？是因为Servlet共享了一个实例变量，所以在多线程的环境下容易产生线程不安全问题。加一个线程锁。

(1) 同步对共享数据的操作 (synchronized)

使用synchronized关键字能保证一次只有一个线程可以访问被保护的区段



```
1 public class ConcurrentTest extends HttpServlet {
2     PrintWriter output;
3     @Override
4     protected void service(HttpServletRequest request, HttpServletResponse response)
5         throws ServletException, IOException {
6         String username;
7         response.setContentType("text/html;charset=gb2312");
8         username=request.getParameter("username");
9         synchronized(this){
10             output=response.getWriter();
11             try {
12                 //为了突出并发问题，在这设置一个延时
13                 Thread.sleep(5000);
14                 output.println("用户名:"+username+"<BR>");
15             } catch (Exception e) {
16                 e.printStackTrace();
17             }
18         }
19     }
20 }
```

(2) 避免使用实例变量

将实例变量改为局部变量

```
1 public class ConcurrentTest extends HttpServlet {
2     @Override
3     protected void service(HttpServletRequest request, HttpServletResponse response)
4         throws ServletException, IOException {
5         PrintWriter output;
6         String username;
7         response.setContentType("text/html;charset=gb2312");
8         username=request.getParameter("username");
9         synchronized(this){
10             output=response.getWriter();
11             try {
12                 //为了突出并发问题,在这设置一个延时
13                 Thread.sleep(5000);
14                 output.println("用户名:"+username+"<BR>");
15             } catch (Exception e) {
16                 e.printStackTrace();
17             }
18         }
19     }
20 }
```

4.描述一下cookie和session及其区别

(1) session是在服务器保存的一个数据结构,用来跟踪用户的状态,这个数据可以保存在集群、数据库、文件中。

(2) cookie是客户端保存用户信息的一种机制,用来记录用户的一些信息,也是实现session的一种方式。

5.如何进行Session的会话跟踪

传统的3种会话跟踪: Cookie、隐藏表单域和URL重写

(1) Cookie: Cookie是由服务器端生成,发送User-Agent (一般是浏览器),浏览器会将Cookie的key/value保存到某个目录下的文件内,下次请求同一网站时就发送该Cookie给服务器。

6.描述一下Servlet中的request对象

是Servlet中继承HttpServlet 里面实现doGet、doPost (HttpServletRequest , HttpServletResponse)

HttpServletRequest和HttpServletResponse接口的类从哪里来，是从某段代码中实例化中来的，**Servlet容器中**。

7.描述转发和重定向 (sendRedirect) 的区别

(1) RequestDispatcher.forward()方法只能将请求转发给同一个WEB应用中的组件；而 HttpServletResponse.sendRedirect方法还可以重定向到同一个站点上的其他应用程序中的资源，甚至是使用绝对URL重定向到其他站点的资源。

(2) 重定向地址栏会改变，而转发地址栏不改变。

8.JSP中内置对象

Jsp中有九个这样的内置对象：**request、response、page、pageContext、session、application、out、config、exception**

1、request对象

request对象是HttpServletRequest类型对象，该对象代表了客户端的请求信息，主要用于接受通过HTTP协议传送到服务器的数据（包括头信息、系统信息、请求方式以及请求参数等）。request对象的作用域为一次请求。

2、response对象

response代表的是对客户端的响应，主要是将JSP容器处理过的对象传回到客户端。response对象也具有作用域，它只在JSP页面内有效。

3、Session对象

session对象是由服务器自动创建的与用户请求相关的对象。服务器为每个用户都生成一个session对象，用于保存该用户的信息，跟踪用户的操作状态。session对象内部使用Map类来保存数据。

4、Application对象

application 对象可将信息保存在服务器中，直到服务器关闭，否则application对象中保存的信息会在整个应用中都有效。与session对象相比，application对象生命周期更长，类似于系统的“全局变量”。

5、out对象

out 对象用于在Web浏览器内输出信息，并且管理应用服务器上的输出缓冲区。在使用out 对象输出数据时，可以对数据缓冲区进行操作，及时清除缓冲区中的残余数据，为其他的输出让出缓冲空间。待数据输出完毕后，要及时关闭输出流。

6、pageContext对象

pageContext 对象的作用是取得任何范围的参数，通过它可以获取 JSP页面的out、request、response、session、application 等对象。pageContext对象的创建和初始化都是由容器来完成的，在JSP页面中可以直接使用 pageContext对象。

7、config对象

config 对象的主要作用是取得服务器的配置信息。通过 pageContext对象的 getServletConfig() 方法可以获取一个config对象。当一个Servlet 初始化时，容器把某些信息通过 config对象传递给这个 Servlet。开发者可以在web.xml 文件中为应用程序环境中的Servlet程序和JSP页面提供初始化参数。

8、page对象

page 对象代表JSP本身，只有在JSP页面内才是合法的。page隐含对象本质上包含当前Servlet接口引用的变量，类似于Java编程中的 this 指针。

9、exception对象

exception 对象的作用是显示异常信息，只有在包含 isErrorPage="true" 的页面中才可以被使用，在一般的JSP页面中使用该对象将无法编译JSP文件。exception对象和Java的所有对象一样，都具有系统提供的继承结构。exception 对象几乎定义了所有异常情况。在Java程序中，可以使用try/catch关键字来处理异常情况；如果在JSP页面中出现没有捕获到的异常，就会生成 exception 对象，并把 exception 对象传送到在page指令中设定的错误页面中，然后在错误页面中处理相应的 exception 对象。

9、简述JSP中四大保存作用域

page:当页面，也就是只要跳到别的页面就失效了

request: 一次会话，简单的理解就是一次请求范围内有效

session: 浏览器进程，只要当页面没有被关闭（没有被程序强制删除），不管怎么跳转都是有效的

application: 服务器，只要服务器没有重启，数据就有效

10、WEB开发中的数据共享方式有哪些？

request、session、application

