首先要声明一点，所谓“作用域”就是“信息共享的范围”，也就是说一个信息能够在多大的范围内有效。4个JSP内置对象的作用域分别为：application、session、request、page 。JSP内置对象作用域表如下：

|  |  |
| --- | --- |
| 名称 | 作用域 |
| application | 在所有应用程序中有效 |
| session | 在当前会话中有效 |
| request | 在当前请求中有效 |
| page | 在当前页面有效 |

Web交互的最基本单位为HTTP请求。每个用户从进入网站到离开网站这段过程称为一个HTTP会话，一个服务器的运行过程中会有多个用户访问，就是多个HTTP会话。作用域解释如下。

### **application 作用域**

     如果把变量放到application里，就说明它的作用域是application，它的有效范围是整个应用。 整个应用是指从应用启动，到应用结束。我们没有说“从服务器启动，到服务器关闭”，是因为一个服务器可能部署多个应用，当然你关闭了服务器，就会把上面所有的应用都关闭了。 application作用域里的变量，它们的存活时间是最长的，如果不进行手工删除，它们就一直可以使用。

application作用域上的信息传递是通过ServletContext实现的，它提供的主要方法如下所示：

Object getAttribute（String name）  //从application中获取信息；

void setAttribute（String name, Object value）  //向application作用域中设置信息。

### **session作用域**

       session作用域比较容易理解，同一浏览器对服务器进行多次访问，在这多次访问之间传递信息，就是session作用域的体现。如果把变量放到session里，就说明它的作用域是session，它的有效范围是当前会话。所谓当前会话，就是指从用户打开浏览器开始，到用户关闭浏览器这中间的过程。这个过程可能包含多个请求响应。也就是说，只要用户不关浏览器，服务器就有办法知道这些请求是一个人发起的，整个过程被称为一个会话（session），而放到会话中的变量，就可以在当前会话的所有请求里使用。   
 session是通过HttpSession接口实现的，它提供的主要方法如下所示：

Object HttpSession.getAttribute（String name） //从session中获取信息。

void HttpSession.setAttribute（String name, Object value）//向session中保存信息。

HttpSession HttpServletRequest.getSessio() //获取当前请求所在的session的对象。

 session的开始时刻比较容易判断，它从浏览器发出第一个HTTP请求即可认为会话开始。但结束时刻就不好判断了，因为浏览器关闭时并不会通知服务器，所以只能通过如下这种方法判断：如果一定的时间内客户端没有反应，则认为会话结束。Tomcat的默认值为120分钟，但这个值也可以通过HttpSession的setMaxInactiveInterval()方法来设置：

void setMaxInactiveInterval(int interval)

如果想主动让会话结束，例如用户单击“注销”按钮的时候，可以使用 HttpSession 的 invalidate()方法，用于强制结束当前session：void invalidate()

### **request作用域**

     一个HTTP请求的处理可能需要多个Servlet合作，而这几个Servlet之间可以通过某种方式传递信息，但这个信息在请求结束后就无效了。request里的变量可以跨越forward前后的两页。但是只要刷新页面，它们就重新计算了。如果把变量放到request里，就说明它的作用域是request，它的有效范围是当前请求周期。 所谓请求周期，就是指从http请求发起，到服务器处理结束，返回响应的整个过程。在这个过程中可能使用forward的方式跳转了多个jsp页面，在这些页面里你都可以使用这个变量。

Servlet之间的信息共享是通过HttpServletRequest接口的两个方法来实现的：

void setAttribute（String name, Object value） //将对象value以name为名称保存到request作用域中。

Object getAttribute（String name）//从request作用域中取得指定名字的信息。

     JSP中的doGet()、doPost()方法的第一个参数就是HttpServletRequest对象，使用这个对象的 setAttribute()方法即可传递信息。那么在设置好信息之后，要通过何种方式将信息传给其他的Servlet呢？这就要用到RequestDispatcher接口的forward()方法，通过它将请求转发给其他Servlet。

 RequestDispatcher ServletContext.getRequestDispatcher(String path) //取得Dispatcher以便转发，path为转发的目的Servlet。

 void RequestDispatcher.forward(ServletRequest request, ServletResponse response)//将request和response转发

     因此，只需要在当前Servlet中先通过setAttribute()方法设置相应的属性，然后使用forward()方法进行跳转，最后在跳转到的Servlet中通过使用getAttribute()方法即可实现信息传递。

**需要注意两点：**

1、转发不是重定向，转发是在Web应用内部进行的。

2、转发对浏览器是透明的，也就是说，无论在服务器上如何转发，浏览器地址栏中显示的仍然是最初那个Servlet的地址。

### **page作用域**

page对象的作用范围仅限于用户请求的当前页面，对于page对象的引用将在响应返回给客户端之后被释放，或者在请求被转发到其他地方后被释放。page里的变量只要页面跳转了，它们就不见了。如果把变量放到pageContext里，就说明它的作用域是page，它的有效范围只在当前jsp页面里。从把变量放到pageContext开始，到jsp页面结束，你都可以使用这个变量。

以上介绍的作用范围越来越小，request和page的生命周期都是短暂的，它们之间的区别：一个request可以包含多个page页（include，forward及filter）。