# JSP九大内置对象

JSP的九大内置对象分别为：**request、response、session、application、out、pagecontext、config、page、exception。**

======================================================

Request（javax.servlet.http.HttpServletRequest）：顾名思义 请求的意思 该对象表示的是由客户端（浏览器端）发送的请求封装。 每当请求发生后我们就可以通过request对象获取客户端传递的参数以及获取请求头内容等等所有有关请求的内容信息都可以通过request对象来进行获取这些。

Request生命周期：仅存在与当次请求中请求相应结束后该request销毁（包括其中所含参数等信息一起被销毁），属于“一次性”的对象。所以对于我们之前使用的request.getRequestDispatcher("/userlist.jsp").forward(request, response)这段代码，在servlet中并没有进行直接的响应而是将该请求对象进行了一次传递，传递给我们的jsp，jsp此时才可以获取的到我们通过request.setAttribute(“”,””)放置在request对象中的值，而你使用response.sendriirect方法直接跳转页面的话是获取不到**请求**传递的参数的。

所以基本上 action也好 servlet也罢对于传递参数这块我们只能用request或者其他session对象进行传递不可以使用response对象进行操作。一旦执行response则说明当前请求结束响应完成，request对象被销毁。

=======================================================

Response（javax.servlet.http.HttpServletResponse）：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（浏览器）。根据HTTp协议 每一次请求都跟随一次响应，请求与响应总是成对出现的，除非代码或者服务器出现异常才会出现只有请求没有响应。

Response对象也具有作用域，它只在JSP页面（或servlet）内有效。与request相同，当本次响应结束后response对象被销毁。

=======================================================

Session（javax.servlet.http.HttpSession）：session（会话）对象是一个JSP内置对象，它在第一个JSP页面被装载时自动创建（访问这个项目的第一个页面开始 此时 session对象被创建），完成会话期管理。从一个用户打开浏览器并连接到服务器开始，到用户关闭浏览器离开这个服务器结束，被称为一个会话。（ 有关当前项目的所有页面被关闭后 session被销毁 ）

Session对象一旦被创建它的作用域 作用于所有这个项目下的打开的 所有 相关这个项目的jsp页面中。

当一个客户访问一个服务器时，可能会在这个服务器的几个页面之间切换，服务器应当通过某种办法知道这是一个客户，就需要Session对象。

当一个客户首次访问服务器上的一个JSP页面时，JSP引擎产生一个Session对象，同时分配一个String类型的ID号，JSP引擎同时将这换个ID号发送到客户端，存放在Cookie中，这样Session对象，直到客户关闭浏览器后，服务器端该客户的Session对象才取消，并且和客户的会话对应关系消失。当客户重新打开浏览器再连接到该服务器时，服务器为该客户再创建一个新的Session对象。

总结session的生命周期主要是存在于在通过访问http地址访问到某jsp（注意一定是jsp）页面开始session就可以开始生成了，生成session后可以在servlet或者action中获取session对象从而使用session对象为我们服务，当前session对象存活于当前客户端（电脑打开浏览器）中的所有页面中（当然是当前项目的页面），当关闭掉所有页面后 该session被销毁。

在servlet中获取session方法：request.getSession

在action中获取session方法：

方法一:ActionContext.getContext().getSession();

方法二：ServletActionContext.getRequest().getSession();

方法一获取的session是struts封装过的一个Map类型的session，只能调用put()方法缓存数据。

方法二获取的session是原生的session，可以调用setAttribute()方法进行传参。

一个使用session的经典案例：

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，因此保存数据的格式为 “Key/value”。 session对象的value可以使复杂的对象类型，而不仅仅局限于字符串类型。

public String getId()：获取Session对象编号。

public void setAttribute(String key,Object obj)：将参数Object指定的对象obj添加到Session对象中，并为添加的对象指定一个索引关键字。

public Object getAttribute(String key)：获取Session对象中含有关键字的对象。

public Boolean isNew()：判断是否是一个新的客户。

HttpSession session = request.getSession();

=======================================================

Application（javax.servlet.ServletContext）：

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”。

**服务器启动后就产生了这个Application对象**，当客户再访问的网站的各个页面之间浏览时，这个Application对象都是同一个，直到服务器关闭。但是与Session对象不同的时，所有客户的Application对象都时同一个，即所有客户共享这个内置的Application对象。

setAttribute(String key,Object obj)：将参数Object指定的对象obj添加到Application对象中，并为添加的对象指定一个索引关键字。

getAttribute(String key)：获取Application对象中含有关键字的对象。

在servlet中获取Application：

ServletContext application=this.conf.getServletContext();

ServletContext application=this.getServletContext();

在action中获取Application：

ActionContext ctx = ActionContext.getContext();

也有通过配置文件配置后获取Application的写法，有点奇葩不建议使用 知道有这么回事即可。

=======================================================

Out（javax.servlet.jsp.jspWriter）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。

Out对象时一个输出流，用来向客户端输出数据。Out对象用于各种数据的输出。其常用方法如下。

out.print()：输出各种类型数据。

out.newLine()：输出一个换行符。

out.close()：关闭流。

====================================================

PageContext（javax.servlet.jsp.PageContext）

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、reponse、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。

pagecontext 对象代表JSP本身，只有在JSP页面内才是合法的。 pagecontext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。

在servlet里获取pagecontext对象：

JspFactory fac=JspFactory.getDefaultFactory();

PageContext pageContext=fac.getPageContext(this, request,response, null,false,JspWriter.DEFAULT\_BUFFER, true);

Struts2的Action中获取PageContext方法；

ServletActionContext.getPageContext()；

注意：pagecontext对象主要作用范围在于jsp页面中 尽量不在servlet或Action中使用。

=======================================================

config（javax.servlet.ServletConfig）对象的主要作用是取得服务器的配置信息。通过 pageConext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。总结web.xml中的配置内容可以在config对象中获取。

=======================================================

cookie对象

Cookie是Web服务器保存在用户硬盘上的一段文本。Cookie允许一个Web站点在用户电脑上保存信息并且随后再取回它。举例来说，一个Web站点可能会为每一个访问者产生一个唯一的ID，然后以Cookie文件的形式保存在每个用户的机器上。

创建一个Cookie对象 调用Cookie对象的构造函数就可以创建Cookie对象。Cookie对象的构造函数有两个字符串参数：Cookie名字和Cookie值。

例如：Cookie c = new Cookie(username",john"); 将Cookie对象传送到客户端。

JSP中，如果要将封装好的Cookie对象传送到客户端，可使用Response对象的addCookie()方法。

例如：response.addCookie(c)，读取保存到客户端的Cookie。

使用Request对象的getCookie()方法，执行时将所有客户端传来的Cookie对象以数组的形式排列，如果要取出符合需要的Cookie对象，就需要循环比较数组内每个对象的关键字。设置Cookie对象的有效时间，用Cookie对象的setMaxAge()方法便可以设置Cookie对象的有效时间，

例如：Cookie c = newCookie(username","john");c.setMaxAge(3600);------

Cookie对象的典型应用时用来统计网站的访问人数。由于代理服务器、缓存等的使用，唯一能帮助网站精确统计来访人数的方法就是为每个访问者建立一个唯一ID。使用Cookie，网站可以完成以下工作:

测定多少人访问过。测定访问者有多少是新用户（即第一次来访），多少是老用户。

测定一个用户多久访问一次网站。当一个用户第一次访问时，网站在数据库中建立一个新的ID，并把ID通过Cookie传送给用户。用户再次来访时，网站把该用户ID对应的计数器加1，得到用户的来访次数。

总结 Cookie是存储在客户端的，向cookie中放置值使用response.addCookie(new Cookie（“key”，“value”）)(这种结构很像map的 key-value键值对形式)；cookie可以设置在客户端存活时间寿命长短；cookie其实是存放在 客户端硬盘中的，不是存放在内存中。

最常见的cookie应用场景：勾选保存用户登陆信息x天（这个应用场景使用cookie再合适不过了，老师会带领大家做一遍这个流程）。

**前端cookie的相关知识：**

cookie是浏览器提供的一种机制，它将document 对象的cookie属性提供给JavaScript。可以由JavaScript对其进行控制，而并不是JavaScript本身的性质。cookie是存于用户硬盘的一个文件，这个文件通常对应于一个域名，当浏览器再次访问这个域名时，便使这个cookie可用。因此，cookie可以跨越一个域名下的多个网页，但不能跨越多个域名使用。

不同的浏览器对cookie的实现也不一样，但其性质是相同的。例如在Windows 2000以及Windows xp中，cookie文件存储于documents and settings\userName\cookie\文件夹下。通常的命名格式为：userName@domain.txt。

cookie机制将信息存储于用户硬盘，因此可以作为全局变量，这是它最大的一个优点。它可以用于以下几种场合。

（1）保存用户登录状态。例如将用户id存储于一个cookie内，这样当用户下次访问该页面时就不需要重新登录了，现在很多论坛和社区都提供这样的功能。 cookie还可以设置过期时间，当超过时间期限后，cookie就会自动消失。因此，系统往往可以提示用户保持登录状态的时间：常见选项有一个月、三个 月、一年等。

（2）跟踪用户行为。例如一个天气预报网站，能够根据用户选择的地区显示当地的天气情况。如果每次都需要选择所在地是烦琐的，当利用了 cookie后就会显得很人性化了，系统能够记住上一次访问的地区，当下次再打开该页面时，它就会自动显示上次用户所在地区的天气情况。因为一切都是在后 台完成，所以这样的页面就像为某个用户所定制的一样，使用起来非常方便。

（3）定制页面。如果网站提供了换肤或更换布局的功能，那么可以使用cookie来记录用户的选项，例如：背景色、分辨率等。当用户下次访问时，仍然可以保存上一次访问的界面风格。

（4）创建购物车。正如在前面的例子中使用cookie来记录用户需要购买的商品一样，在结账的时候可以统一提交。例如淘宝网就使用cookie记录了用户曾经浏览过的商品，方便随时进行比较。

当然，上述应用仅仅是cookie能完成的部分应用，还有更多的功能需要全局变量。cookie的缺点主要集中于安全性和隐私保护。主要包括以下几种：

（1）**cookie可能被禁用**。当用户非常注重个人隐私保护时，他很可能禁用浏览器的cookie功能；

（2）cookie是与浏览器相关的。这意味着即使访问的是同一个页面，不同浏览器之间所保存的cookie也是不能互相访问的；

（3）cookie可能被删除。因为每个cookie都是硬盘上的一个文件，因此很有可能被用户删除；

（4）cookie安全性不够高。所有的cookie都是以纯文本的形式记录于文件中，因此如果要保存用户名密码等信息时，最好事先经过加密处理。（MD5加密算法）

=======================================================

exception （java.lang.Throwable）

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。

excepation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。

exception对象了解即可，知道jsp对exception是提供支持的即可。值得一提的是 我们的Struts2框架就是利用了exception对象 有时候将一些原本后台的错误信息全部打印到前端方便开发人员查看使用，了解即可。

如若想要使用类似对象获取其他对象如session或request(话说servlet中doget与dopost本身方法参数里面带着request呢)可以通过如下方式获取：

|  |  |  |
| --- | --- | --- |
| **要获取内容** | **Servlet中** | **Action中** |
| Session | HttpSession session = request.getSession(); | ServletActionContext.getActionContext(ServletActionContext.getRequest()).getSession() |
| Request  Response同理 | doget dopost（作为方法入参供使用） | ServletActionContext.getRequest |