@Description：Report of Servlet Basic, Part 2

@Date：2018.6.1

@Auther：Sujin Guo

@Reference: [1] 《Servlet & Jsp &Spring MVC 初学指南》

[2] https://www.cnblogs.com/raozihao/p/7711582.html

JSP

JSP 基础

1、概论

在 jsp 出现前，Servlet 必须在 Writer 中拼接 html，使得控制层和显示层混在一起。jsp 则实现了控制层和显示层的分离，让 Servlet 只关注业务逻辑，jsp 控制页面展示。

JSP 的生命周期：

JSP 页面本质上是一个 Servlet。当一个JSP页面第一次被请求时，Servlet/JSP 容器主要做以下两件事情：

1. 转换JSP页面到JSP页面实现类，该实现类是一个实现 javax.servlet.jsp.JspPage 接口或子接口 javax.servlet.jsp.HttpJspPage 的Java类。JspPage是 javax.servlet.Servlet 的子接口，这使得每一个JSP页面都是一个Servlet。该实现类的类名由Servlet/JSP容器生成。如果出现转换错误，则相关错误信息将被发送到客户端。
2. 如果转换成功，Servlet/JSP 容器随后编译该 Servlet 类，并装载和实例化该类，像其他正常的 Servlet 一样执行生命周期操作。

JSP 页面也可以被配置为启动时加载。当JSP文件修改时，容器会自动重新编译，无需重启tomcat。

注释：

java 注释，不会发送给客户端。

```
<%-- retrieve products to display --%>
```

html 注释，会发送给客户端

```
<!-- [comments here] -->
```

2、JSP 中九大隐式对象

Servlet 容器会传递几个对象给它运行的 Servlet。例如，可以通过 Servlet 的 service 方法拿到 HttpServletRequest 和 HttpServletResponse 对象，以及可以通过init方法访问到 ServletConfig 对象。此外，可以通过调用 HttpServletRequest 对象的 getSession 方法访问到 HttpSession 对象。

在 JSP 中，可以通过使用隐式对象来访问上述对象。表3.1所示为 JSP 隐式对象。

对象	类型
request	javax.servlet.http.HttpServletRequest
response	javax.servlet.http.HttpServletResponse
out	javax.servlet.jsp.JspWriter
session	javax.servlet.http.HttpSession
application	javax.servlet.ServletContext
config	javax.servlet.ServletConfig
pageContext	javax.servlet.jsp.PageContext
page	javax.servlet.jsp.HttpJspPage
exception	java.lang.Throwable

可以直接在JSP内嵌java语句中使用这些对象，如：

```
<% String userName = request.getParameter("userName"); %>
```

PageContext：

PageContext 有设置和获取属性的方法。属性值可被存储在4个范围之一：页面、请求、会话和应用程序（PAGE_SCOPE、REQUEST_ SCOPE、SESSION_SCOPE和APPLICATION_SCOPE）。页面范围是最小范围，这里存储的属性只在同一个JSP页面可用。请求范围是指当前的ServletRequest中。会话范围指当前的HttpSession中。应用程序范围指应用的ServletContext中。

```
public abstract void setAttribute(String name, Object value, int scope)
```

案例：jsp 调用隐式对象页面

（1）request 对象

request 对象是 javax.servlet.http.HttpServletRequest 对象的一个实例。每当客户端请求页面时，JSP引擎将创建一个新对象来表示该请求。

request对象提供了获取包括表单数据，Cookie，HTTP方法等HTTP头信息的方法。

（2）response 对象

response对象是javax.servlet.http.HttpServletResponse对象的一个实例。就像服务器创建request对象一样，它还创建一个对象来表示对客户端的响应。

response对象还定义了处理创建新HTTP头的接口。通过此对象，JSP程序员可以添加新的Cookie或日期戳，HTTP状态代码等。

（3）out 对象

out隐式对象是javax.servlet.jsp.JspWriter对象的一个实例，用于在响应中发送内容。可以动态输出页面内容，如动态打印表格。

初始化JspWriter对象根据页面是否缓存而不同地实例化。缓冲可以通过使用page指令的buffered ='false'属性来关闭。

JspWriter对象包含与java.io.PrintWriter类大部分相同的方法。但是，JspWriter还有一些额外的方法用来处理缓冲。与PrintWriter对象不同，JspWriter会抛出IOExceptions异常。

（4）session 对象

session对象是javax.servlet.http.HttpSession的一个实例，其行为与Java Servlet下的会话对象行为完全相同。session对象用于跟踪客户端请求之间的客户端会话。

（5）application 对象

application 对象是生成的 Servlet 的 ServletContext 对象的直接包装，实际上是javax.servlet.ServletContext对象的一个实例。

application对象是JSP页面在其整个生命周期中的表示。 当JSP页面被初始化时，将创建此对象，并且在JSP页面被jspDestroy()方法删除时 application 对象也将被删除。

通过向 application 对象添加属性值，可以确保组成 Web 应用程序的所有 JSP 文件都可以访问它。

（6）config 对象

config 对象是 javax.servlet.ServletConfig 的实例化，是生成的 servlet 的 ServletConfig 对象周围的直接包装。

（7）pageContext 对象

pageContext 对象是 javax.servlet.jsp.PageContext 对象的一个实例。pageContext 对象用于表示整个JSP页面。

（8）page 对象

page对象是对该页面实例的实际引用。可以认为它是表示整个JSP页面的对象。

page对象是this对象的直接同义词。

（9）exception 对象

exception对象是一个包含上一页抛出的异常的包装器。它通常用于生成对错误条件的适当响应。

3、JSP 中的四个作用域

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。JSP中九个内置对象及其相应所属作用域如下表：

对象	所属作用域	作用域描述
request	**request**	在当前请求中有效
response	**page**	在当前页面有效
out	**page**	在当前页面有效
session	**session**	在当前会话中有效
application	**application**	在所有应用程序中有效
config	**page**	在当前页面有效
pageContext	**page**	在当前页面有效
page	**page**	在当前页面有效
Exception	**page**	在当前页面有效

由上图可知，这九个内置对象都有相应的作用域，作用域在这里的作用就是限定对象的生命周期。如果跳出了当前对象的作用域，该对象的信息就不能再被访问。

JSP指令

指令是 JSP 语法元素的第一种类型。它们指示 JSP 转换器如何翻译 JSP 页面为 Servlet。

1、page指令

```
<%@page attribute1="value1" attribute2="value2" ... %>
```

page 指令有如下属性：

属性	说明
import	定义一个或多个本页面中将被导入和使用的**java**类型。
session	本页面是否加入会话管理。默认值为True。
buffer	定义隐式对象out的缓冲大小。必须以KB后缀结尾。默认大小为8KB或更大（取决于JSP容器）。该值可以为none，这意味着没有缓冲，所有数据将直接写入PrintWriter。
autoFlush	默认值为True。若值为True，则当输出缓冲满时会自写入输出流。而值为False，则仅当调用隐式对象的flush方法时，才会写入输出流。因此，若缓冲溢出，则会抛出异常。
isThreadSafe	定义该页面的线程安全级别。不推荐使用 JSP 参数，因为使用该参数后，会生成一些Servlet容器已过期的代码。
info	返回调用容器生成的 Servlet 类的 getServletInfo 方法的结果
errorPage	定义当出错时用来处理错误的页面。
isErrorPage	标识本页是一个错误处理页面。
contentType	定义本页面隐式对象**response**的内容类型，默认是**text/html**。
pageEncoding	定义本页面的字符编码，默认是**ISO-8859-1**。
isELIgnored	配置是否忽略EL表达式。EL是Expression Language的缩写。
language	定义本页面的脚本语言类型，默认是 Java，这在 JSP 2.2 中是唯一的合法值。
extends	定义JSP实现类要继承的父类。这个属性的使用场景非常罕见，仅在特殊理由下使用。
deferredSyntax AllowedAsLiteral	定义是否解析字符串中出现“#{”符号，默认是False。“{# ”是一个表达式语言的起始符号。
trimDirective Whitespaces	定义是否不输出多余的空格/空行，默认是False。

2、include 指令

include 指令的语法如下：

```
<%@include file="url"%>
```

include 指令引入 copyright.jspf 文件，include所在的行被copyright.jspf 文件内容代替。按照惯例，以JSPF为扩展名的文件代表JSP fragement 或 JSP segment。include指令也可以包含静态HTML文件。

```
<html>
  <head><title>Including a file</title></head>
  <body>
    This is the included content: <hr/>
    <%@ include file="copyright.jspf"%>
  </body>
</html>
```

脚本元素

一个脚本程序是一个Java代码块，以<%符号开始，以%>符号结束。

1、表达式

每个表达式都会被JSP容器执行，并使用隐式对象out的打印方法输出结果。表达式以“<%=”开始，并以“%>”结束。

```
Today is <%=java.util.Calendar.getInstance().getTime()%>
<!-- 上下两段代码等效 -->
Today is <% out.print(java.util.Calendar.getInstance().getTime()); %>
```

2、声明

可以声明能在JSP页面中使用的变量和方法。声明以“<%！”开始，并以“%>”结束。

```
<%!
public String getTodaysDate() {
return new java.util.Date();
}%>
<html>
  <head><title>Declarations</title></head>
  <body>
  Today is <%=getTodaysDate()%>
  </body>
</html>
```

可以使用声明来重写JSP页面，实现类的init和destroy方法。通过声明jspInit方法，来重写init方法。通过声明jspDestroy方法，来重写destory方法。在创建和销毁JSP页面时会分别调用jspInit和jspDestroy方法。

3、禁用脚本元素

随着JSP 2.0对表达式语言的加强，推荐的实践是：在JSP页面中用EL访问服务器端对象且不写Java代码。因此，从JSP 2.0起，可以通过在部署描述符中的定义一个scripting-invalid元素，来禁用脚本元素。

```
<jsp-property-group>
  <url-pattern>*.jsp</url-pattern>
  <scripting-invalid>true</scripting-invalid>
</jsp-property-group>
```

动作

动作是第三种类型的语法元素，它们被转换成Java代码来执行操作，如访问一个Java对象或调用方法。下面讨论所有JSP容器支持的标准动作。除标准外，还可以创建自定义标签执行某些操作。

1、useBean

useBean将创建一个关联Java对象的脚本变量。这是早期分离的表示层和业务逻辑的手段。随着其他技术的发展，如自定义标签和表达语言，现在很少使用useBean方式。

它创建一个java.util.Date实例，并赋值给名为today的脚本变量，然后在表达式中使用。访问这个页面会显示系统当前时间：

```
<jsp:useBean id="today" class="java.util.Date"/>
<%=today%>
```

2、setProperty 和 getProperty

setProperty动作可对一个Java对象设置属性，而getProperty则会输出Java对象的一个属性。

```
//Employee 实体
public class Employee {
  private String id;
  private String firstName;
  private String lastName;
  ... getter and setter 函数 ...
}
```

```
<!-- 用setProperty设置实体属性，getProperty获取实体属性 -->
<body>
  <jsp:useBean id="employee" class="app03a.Employee"/>
  <jsp:setProperty name="employee" property="firstName"value="Abigail"/>
  First Name: <jsp:getProperty name="employee" property="firstName"/>
</body>
```

3、include

include动作用来动态地引入另一个资源。可以引入另一个JSP页面，也可以引入一个Servlet或一个静态的HTML页面。

```
<body>
  <jsp:include page="jspf/menu.jsp">
    <jsp:param name="text" value="How are you?"/>
  </jsp:include>
</body>
```

include 指令和 include 动作的不同：

1. 对于include指令，资源引入发生在 JSP 容器将页面转换为生成的 Servlet 时。而对于include动作，资源引入发生在请求页面时。因此，使用include动作是可以传递参数的，而include指令不支持。
2. 第二个不同是，include指令对引入的文件扩展名不做特殊要求。但对于include动作，若引入的文件需以JSP页面处理，则其文件扩展名必须是JSP。若使用.jspf为扩展名，则该页面被当作静态文件。

4、forward

forward将当前页面转向到其他资源。下面代码将从当前页转向到 login.jsp 页面：

```
<jsp:forward page="jspf/login.jsp">
  <jsp:param name="text" value="Please login"/>
</jsp:forward>
```

错误处理

当 jsp 页面抛出异常时，用户将看到一个精心设计的网页解释发生了什么，而不是一个用户无法理解的错误信息。

page指令定义错误页面：

```
<%@page isErrorPage="true"%>
<html>
<head><title>Error</title></head>
  <body>
  An error has occurred. <br/>
  Error message:
  <% out.println(exception.toString()); %>
  </body>
</html>
```

其他页面指向错误页面，也用page指令定义：

```
<%@page errorPage="errorHandler.jsp"%>
Deliberately throw an exception
<% Integer.parseInt("Throw me"); %>
```

JSP：表达式语言

1、语法

JSP 2.0最重要的特性之一就是表达式语言（EL），JSP用户可以用它来访问应用程序数据。

表达式格式：

EL表达式以 ${ 开头，并以 } 结束。EL表达式的结构如下：

```
格式：${expression}
如：${x+y}
```

获取对象属性：

为了获取对象属性，可以用下列两种方式：

```
${object["propertyName"]}
${object.propertyName}
```

嵌套使用的情况：

```
${pageContext["request"]["servletPath"]}
${pageContext.request["servletPath"]}
${pageContext.request.servletPath}
${pageContext["request"].servletPath}
```

2、EL隐式对象

在JSP页面中，可以利用JSP脚本来访问JSP隐式对象。但是，在免脚本的JSP页面中，则不可能访问这些隐式对象。EL允许通过提供一组它自己的隐式对象来访问不同的对象。EL隐式对象如下表所示：

对象	描述
pageContext	这是当前JSP的javax.servlet.jsp.PageContext
initParam	这是一个包含所有环境初始化参数，并用参数名作为key的Map
param	这是一个Map，可以获取请求提交的参数，如登录时的 username、password。
paramValues	和param类似，但Map中key对应的value是一个字符串数组，可返回同一个param名称对应的多个属性。
header	这是一个Map，可以获取请求头的属性，如accept、connect。
headerValues	和header类似，但Map中key对应的value是一个字符串数组，可返回同一个header名称对应的多个属性。
cookie	这是一个包含了当前请求对象中所有Cookie对象的Map。Cookie名称就是key名称，并且每个key都映射到一个Cookie对象
applicationScope	这是一个包含了ServletContext对象中所有属性的Map，并用属性名称作为key
sessionScope	这是一个包含了HttpSession对象中所有属性的Map，并用属性名称作为key
requestScope	这是一个Map，其中包含了当前HttpServletRequest对象中的所有属性，并用属性名称作为key
pageScope	这是一个Map，其中包含了全页面范围内的所有属性。属性名称就是Map的key

在servlet/JSP编程中，有界对象是指在以下对象中作为属性的对象：PageContext、ServletRequest、HttpSession或者ServletContext。隐式对象sessionScope、requestScope和pageScope与applicationScope相似。但是，其范围分别为session、request和page。

3、配置 EL

有了EL、JavaBeans和定制标签，就可以编写免脚本的JSP页面了。JSP 2.0及其更高的版本中还提供了一个开关，可以使所有的JSP页面都禁用脚本。

```
<jsp-config>
  <jsp-property-group>
    <url-pattern>*.jsp</url-pattern>
    <scripting-invalid>true</scripting-invalid>
  </jsp-property-group>
</jsp-config>
```

另一方面，在有些情况下，可能还会需要在应用程序中取消EL。例如，正在使用与JSP 2.0兼容的容器，却尚未准备升级到JSP 2.0，那么就需要这么做。在这种情况下，可以关闭EL表达式的计算。

```
<jsp-config>
  <jsp-property-group>
    <url-pattern>*.jsp</url-pattern>
    <el-ignored>true</el-ignored>
  </jsp-property-group>
</jsp-config>
```

JSTL

JSP 标准标签库（JavaServer Pages Standard Tag Library，JSTL）是一个定制标签库的集合，用来解决像遍历Map或集合、条件测试、XML处理，甚至数据库访问和数据操作等常见的问题。

Demo

- 通过 Servlet 向浏览器传递文件和下载文件
- jsp 隐式对象的作用域