附录 习题答案

第 1 章　JSP 简介

1. 安装 Tomcat 引擎的计算机需要事先安装 JDK 吗?
答：需要。
2. 怎样启动和关闭 Tomcat 服务器?
答：bin 目录下使用 startup.bat 启动 Tomcat 服务器。bin 目录下使用 shutdown.bat 关闭 Tomcat 服务器。
3. Boy.jsp 和 boy.jsp 是否是相同的 JSP 文件名字?
答：不是
4. 请在 D:\下建立一个名字为 water 的目录，并将该目录设置成一个 Web 服务目录，然后编写一个简单 JSP 页面保存到该目录中，让用户使用虚拟目录 fish 来访问该 JSP 页面。
答：设置方法:
① 建立 D:\ water 目录;
② 修改 server.xml 文件,在</host>上一行添加:
<Context path="/fish" docBase="D:\ water" debug="0" reloadable="true"/>
③ 使用 http://localhost:8080/ fish /example1_1.jsp 访问 example1_1.jsp 页面.
5. 假设 Dalian 是一个 Web 服务目录，其虚拟目录为 moon。A.jsp 保存在 Dalian 的子目录 sea 中。那么在 Tomcat 服务器 (端口号 8080) 所在计算机的浏览器键入下列哪种方式是访问 A.jsp 的正确方式?
A．http://127.0.0.1:8080/A.jsp　　B. http://127.0.0.1:8080/Dalian/A.jsp
C. http://127.0.0.1:8080/moon/A.jsp　　D. http://127.0.0.1:8080/moon/sea/A.jsp
答：D
6. 如果想修改 Tomcat 服务器的端口号，应当修改哪个文件？能否将端口号修改为 80?
答：修改 Tomcat 服务器的 conf 目录下的主配置文件 server.xml 可以更改端口号。若 Tomcat 服务器上没有其它占有 80 端口号的程序，可以将其修改为 80，否则不能。
7. 在 Tomcat 服务器的 webapps 目录下新建一个名字是 letter 的 Web 服务目录。编写 JSP 页面 letter.jsp，保存在 letter 的 WEB 服务目录中，该 JSP 页面可以显示希腊字母表。
答：参看 1.6.1 的输出英文字母表。

第 2 章　JSP 语法

1. " <%! " 和 " %> " 之间声明的变量与 " <% " 和 " %> " 声明的变量有何不同?

答：" <%! " 和 " %> " 声明的变量为类的成员变量,其所占的内存直到 Tomcat 服务器关闭才释放。" <% " 和 " %> " 为类方法中声明的局部变量,仅在 JSP 页面后继的程序片及表达式中有效。

2. 如果有 2 个用户访问一个 JSP 页面,该页面中的 Java 程序片将被执行几次?

答：2 次（当有用户访问 JSP 页面，Java 程序片就被执行一次）。

3.是否允许 JSP 页面同时含有如下两条 page 指令：

```
<%@ page contentType="text/html;charset=gb2312" %>
<%@ page contentType="application/msword" %>
```

答：不允许。

4. 是否允许 JSP 页面同时含有如下两条 page 指令：

```
<%@ page import="java.util.*"%>
<%@ page import="java.sql.*" %>
```

答：允许。

5. 假设有两个不同用户访问下列 JSP 页面 hello.jsp,请问第一个访问和第二个访问 hello.jsp 页面的用户看到的页面的效果有何不同?

hello.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<%@ page isThreadSafe="false" %>
<HTML><body>
<%!  int sum=1;
    void add(int m){
       sum = sum +m;
    }
%>
<%  int n =100;
    add(n);
%>
<%=sum%>
</body></HTML>
```

答：第一个用户看到结果是 101，第二个用户看到结果是 201。

6. 请编写一个简单的 JSP 页面，显示英文字母表。

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<HTML><body>
<%
   for(char c='α';c<='ω';c++){
       out.println(" "+c);
```

```
    }
  %>
  </body></HTML>
```

7. 请简单叙述 include 指令标记和 include 动作标记的不同.

答:

include 指令标记:是把被包含的文件的内容放于包含文件中,组成一个文件后编译运行。

include 动作标记: 是把被包含的文件的运行结果放于包含文件运行产生的结果中,这 2 个文件各自编译运行。

8. 编写三个 JSP 页面:main.jsp、circle.jsp、ladder.jsp，将三个 JSP 页面保存在同一 web 服务目录中。main.jsp 使用 include 动作标记加载 circle.jsp 和 ladder.jsp 页面。circle.jsp 页面可以计算并显示圆的面积。ladder.jsp 页面可以计算并显示梯形的面积。当 circle.jsp 和 ladder.jsp 被加载时获取 main.jsp 页面 include 动作标记的 param 子标记提供的圆的半径以及梯形的上底、下底和高的值。

main.jsp:

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<html><body>
  <jsp:include page="circle.jsp">
    <jsp:param name="R" value="4"/>
  </jsp:include>
  <jsp:include page="ladder.jsp">
    <jsp:param name="A" value="5"/>
    <jsp:param name="B" value="6"/>
    <jsp:param name="C" value="10"/>
  </jsp:include>
</body></html>
```

circle.jsp:

```
<%@ page contentType="text/html;charset=GB2312" %>
<html><body>
<%
 String strR=request.getParameter("R");
 double R=Double.parseDouble(strR);
 double area=3.14*R*R;
%>
<p>圆形的面积是：<%= area %>
</body></html>
```

ladder.jsp:

```
<%@ page contentType="text/html;charset=GB2312" %>
<html><body>
<%
 String strA=request.getParameter("A");
 String strB=request.getParameter("B");
 String strC=request.getParameter("C");
```

```
  double a=Double.parseDouble(strA);
  double b=Double.parseDouble(strB);
  double c=Double.parseDouble(strC);
  double area=(a+b)*c/2;
%>
<p>梯形的面积是：<%= area %>
</body></html>
```

第 3 章　Tag 文件与 Tag 标记

1. 用户可以使用浏览器直接访问一个 Tag 文件吗?
答：不可以。
2. Tag 文件应当存放在怎样的目录中?
答：如果某个 Web 服务目录下的 JSP 页面准备调用一个 Tag 文件，那么必须在该 Web 服务目录下建立目录结构：Web 服务目录\WEB-INF\tags，其中，WEB-INF 和 tags 都是固定的子目录名称，而 tags 下的子目录名字可由用户给定。一个 Tag 文件必须保存到 tags 目录或其下的子目录中。
3. Tag 文件中的 tag 指令可以设置哪些属性的值?
答：body-content、language、import、pageEncoding 。
4. Tag 文件中的 attribute 指令有怎样的作用?
答：使用 attribute 指令可以动态地向该 Tag 文件传递对象的引用。
5. Tag 文件中的 variable 指令有怎样的作用?
答：使用 variable 指令可以将 Tag 文件中的对象返回给调用该 Tag 文件的 JSP 页面。
6. 编写两个 Tag 文件 Rect.tag 和 Circle.tag。Rect.tag 负责计算并显示矩形的面积，Circle.tag 负责计算并显示圆的面积。编写一个 JSP 页面 lianxi6.jsp，该 JSP 页面使用 Tag 标记调用 Rect.tag 和 Circle.tag。调用 Rect.tag 时，向其传递矩形的两个边的长度；调用 Circle.tag 时，向其传递圆的半径。

lianxi6.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<%@ taglib tagdir="/WEB-INF/tags" prefix="computer"%>
<HTML><BODY>
   <H3>以下是调用 Tag 文件的效果：</H3>
   <computer:Rect sideA="5" sideB="6"/>
    <H3>以下是调用 Tag 文件的效果：</H3>
   <computer:Circle radius="16"/>
</BODY></HTML>
```

Rect.tag

```
<%@ tag pageEncoding="utf-8" %>
<h4>这是一个 Tag 文件，负责计算矩形的面积。
<%@ attribute name="sideA" required="true" %>
<%@ attribute name="sideB" required="true" %>
  <%!
       public String getArea(double a,double b) {
       if(a>0&&b>0)  {
            double area=a*b ;
             return "<BR>矩形的面积:"+area;
          }
       else {
             return("<BR>"+a+","+b+"不能构成一个矩形,无法计算面积");
```

```
        }
      }
  %>
  <%  out.println("<BR>JSP 页面传递过来的两条边："+sideA+","+sideB);
      double a=Double.parseDouble(sideA);
      double b=Double.parseDouble(sideB);
      out.println(getArea(a,b));
  %>
```

Circle.tag

```
<%@ tag pageEncoding="utf-8" %>
<h4>这是一个 Tag 文件，负责计算园的面积。
<%@ attribute name="radius" required="true" %>
  <%!
       public String getArea(double r) {
          if(r>0) {
             double area=Math.PI*r*r ;
             return "<BR>圆的面积:"+area;
          }
         else{
           return("<BR>"+r+"不能构成一个圆,无法计算面积");
          }
       }
  %>
  <%  out.println("<BR>JSP 页面传递过来的半径："+radius);
      double r=Double.parseDouble(radius);
      out.println(getArea(r));
  %>
```

7. 编写一个 Tag 文件：GetArea.tag 负责求出三角形的面积，并使用 variable 指令返回三角形的面积给调用该 Tag 文件的 JSP 页面。JSP 页面负责显示 Tag 文件返回的三角形的面积。JSP 在调用 Tag 文件时，使用 attribute 指令将三角形三边的长度传递给 Tag 文件。one.jsp 和 two.jsp 都使用 Tag 标记调用 GetArea.tag。one.jsp 将返回的三角形的面积保留最多 3 位小数、two.jsp 将返回的三角形的面积保留最多 6 位小数。

one.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<%@ taglib tagdir="/WEB-INF/tags" prefix="computer"%>
<HTML><body bgcolor=cyan>
  <computer:GetArea sideA="3" sideB="6" sideC="5"/>
  <h4> 面积保留 3 位小数点：
<%  double result=area.doubleValue();
      String str=String.format("%10.3f",result);
      out.println(str);
%>
</body></HTML>
```

two.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
```

```
<%@ taglib tagdir="/WEB-INF/tags" prefix="computer"%>
<HTML><body bgcolor=cyan>
  <computer:GetArea sideA="3.2" sideB="6.1" sideC="5.5"/>
  <h4> 面积保留 6 位小数点:
<%  double result=area.doubleValue();
      String str=String.format("%10.6f",result);
      out.println(str);
%>
</body></HTML>>
```

GetArea.tag

```
<%@ tag pageEncoding="utf-8" %>
<%@ attribute name="sideA" required="true" %>
<%@ attribute name="sideB" required="true" %>
<%@ attribute name="sideC" required="true" %>
<%@ variable name-given="area"
    variable-class="java.lang.Double" scope="AT_END" %>
<%
   double a=Double.parseDouble(sideA);
   double b=Double.parseDouble(sideB);
   double c=Double.parseDouble(sideC);
   if(a+b>c&&a+c>b&&c+b>a){
      double p=(a+b+c)/2.0;
      double result=Math.sqrt(p*(p-a)*(p-b)*(p-c)) ;
      jspContext.setAttribute("area",new Double(result));
   }
   else{ jspContext.setAttribute("area",new Double(-1));
   }
 %>
```

第 4 章　JSP 内置对象

1.假设 JSP 使用的表单中有如下的 GUI(复选框)

<input type="checkbox" name="item" value="bird">鸟

<input type="checkbox" name="item" value="apple">苹果

<input type="checkbox" name="item" value="cat">猫

<input type="checkbox" name="item" value="moon">月亮

该表单所请求的 JSP 可以使用内置对象 request 获取该表单提交的数据,那么,下列哪些是 request 获取该表单提交的值的正确语句?

A. String　a=request.getParameter("item");

B. String　b=request.getParameter("checkbox");

C. String　c[]=request.getParameterValues("item");

D. String　d[]=request.getParameterValues("checkbox");

答:　C.

2. 如果表单提交的信息中有汉字，接收该信息的页面应做怎样的处理?

答：JSP 页面文件的编码为 utf-8 编码，因此只要让内置对象 request 在获取信息之前调用 setCharacterEncoding 方法设置编码为 utf-8（默认是 iso-8859-1）就可以避免乱码现象，代码如下：

```
request.setCharacterEncoding("utf-8");
```

3.编写两个 JSP 页面 inputString.jsp 和 computer.jsp，用户可以使用 inputString.jsp 提供的表单输入一个字符串，并提交给 computer.jsp 页面，该页面通过内置对象获取 inputString.jsp 页面提交的字符串，计算并显示该字符串的长度。

inputString.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<HTML><body>
   <form action="computer.jsp" method=post >
      <BR>请输入字符串:<input type="text" name="string" value=""/></BR>
      <input type="submit" value="提交" name="submit"/>
      <input type="reset" value="重置" >
  </form>
</body></HTML>
```

computer.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<HTML><body>
<%  String yourString=request.getParameter("string"); %>
<p> 您输入的字符串是:<%=yourString %></P>
<p> 字符串的长度是:<%=yourString.length() %></p>
```

```
<a href = "inputString.jsp">返回</a>
</body></HTML>
```

4. response 调用 sendRedirect(URL: url)方法的作用是什么?

答：从一个页面跳转到 sendRedirect(URL: url)中 url 指定的页面，并且这种跳转是客户端跳转。

5. 对例子 1 的代码进行改动，如果用户在 example4_1.jsp 页面提供的表单中输入了非数字字符，computer.jsp 就将用户重新定向到 example4_1.jsp

在例子 1 的 example4_1_computer.jsp 页面的 try-catch 语句的 catch 部分加入重定向代码，即将 catch 部分修改为如下：

```
    catch(NumberFormatException ee){
         out.println("<BR>请输入数字字符");
         response.sendRedirect("example4_1.jsp");
    }
```

6. 一个用户在不同 Web 服务目录中的 session 对象相同吗?

答：不相同.

7. 一个用户在同一 Web 服务目录的不同子目录的 session 对象相同吗?

答：相同.

8. 编写一个 JSP 页面 selectMusic.jsp，该页面使用 select（下拉列表）提供一些歌曲名，用户选择一个一个歌曲名, 单击提交键提交给当前页面, 然后当前页面播放用户选择的音乐 (音频文件保存的 Web 服务目录的 \music 子目录中)。

selectMusic.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<style>
   #tom{
      font-family:宋体;font-size:26;color:blue
   }
</style>
<%   String music = request.getParameter("music");
     if(music==null) music = "back1.mp3";
%>
<HTML><body id=tom >
<form action=""  method=post >
   <b>选择音乐:<br>
   <select id=tom name="music" >
      <Option selected value="back1.mp3">绿岛小夜曲
      <Option value="back2.mp3">我是一片云</option>
      <Option value="back3.mp3">红河谷</option>
   </select>
   <input id=tom type="submit"  name="submit" value="提交"/>
</form>
<br><embed src="music/<%= music %>" height=50 /></body></HTML>
```

第 5 章　JSP 与 Javabean

1. 假设 Web 服务目录 mymoon 中的 JSP 页面要使用一个 bean，该 bean 的包名为 blue.sky。请说明，应当怎样保存 bean 的字节码文件?

答:

(1) 在当前 Web 服务目录下建立如下目录结构:

```
Web 服务目录\WEB-INF\classes
```

(2) 根据类的包名，在目录 classes 下建立相应的子目录，即:

```
Web 服务目录\WEB-INF\classes\blue\sky
```

将获得的字节码文件保存在其中。

2. 假设 Web 服务目录是 mymoon，star 是 mymoon 的一个子目录， JSP 页面 a.jsp 保存在 star 中，并准备使用一个 bean，该 bean 的包名为 tom.jiafei。下列哪个叙述是正确的?

A. 创建 bean 的字节码文件保存在\mymoon\WEB-INF\classes\tom\jiafei 中。

B. 创建 bean 的字节码文件保存在\mymoon\star\WEB-INF\classes\tom\jiafei 中。

C. 创建 bean 的字节码文件保存在\mymoon\WEB-INF\star\classes\tom\jiafei 中。

D. 创建 bean 的字节码文件保存在\mymoon\WEB-INF\classes\start\tom\jiafei 中。

答：A

3. tom.jiafei.Circle 是创建 bean 的类，下列哪个标记是正确创建 session 周期 bean 的标记?

```
A. <jsp:useBean id="circle" class="tom.jiafei.Circle" scope="page"/>
B. <jsp:useBean id="circle" class="tom.jiafei.Circle" scope="request"/>
C. <jsp:useBean id="circle" class="tom.jiafei.Circle" scope="session"/>
D. <jsp:useBean id="circle" type="tom.jiafei.Circle" scope="session"/>
```

答： C

4. 假设创建 bean 的类有一个 int 型的属性 number，下列哪个方法是设置该属性值的正确方法?

A.
```
public void setNumber(int n) {
   number=n;
}
```

B.
```
void setNumber(int n){
   number =n;
}
```

C.
```
public void SetNumber(int n) {
   number =n;
}
```

D.
```
public void Setnumber(int n){
   number =n;
}
```

答： A.

5. 不可以。

6. 编写一个 JSP 页面，该页面提供一个表单，用户可以通过表单输入梯形的上底、下底和高的值，并提交给本 JSP 页面，该 JSP 页面将计算梯形的面积之任务交给一个 page bean 去完成。JSP 页面使用 getProperty 动作标记显示 page bean 中的数据，比如梯形的面积。

ladder.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<jsp:useBean id="ladder" class="geng.Ladder" scope="page"/>
<HTML><body bgcolor=cyan>
<form action="" Method="post" >
   输入梯形上底、下底和高:
   上底:<input type=text name="top" value=0>
   下底:<input type=text name="bottom" value=0>
   高:<input type=text name="height" value=0>
  <input type=submit value="提交">
</form>
<jsp:setProperty name="ladder" property="*"/>
  上底:<jsp:getProperty name="ladder" property="top"/>,
  下底: <jsp:getProperty name="ladder" property="bottom"/>,
   高: <jsp:getProperty name="ladder" property="height"/>.
<br>面积: <jsp:getProperty name="ladder" property="area"/>
</body></HTML>
```

Ladder.java

```
package geng;
public class Ladder {
   double top=0,bottom=0,height=0,area=-1;
   public double getArea(){
       area=(top+bottom)*height/2;
       return area;
   }
   public double getBottom() {
     return bottom;
   }
   public void setBottom(double bottom) {
     this.bottom = bottom;
   }
   public double getHeight() {
     return height;
   }
   public void setHeight(double height) {
     this.height = height;
   }
   public double getTop() {
     return top;
   }
   public void setTop(double top) {
     this.top = top;
```

```
   }
}
```

7. 编写两个 JSP 页面 a.jsp 和 b.jsp，a.jsp 页面提供一个表单，用户可以通过表单输入矩形的两个边长提交给 b.jsp 页面，b.jsp 调用一个 request bean 去完成计算矩形面积的任务。b.jsp 页面使用 getProperty 动作标记显示矩形的面积。

a.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<HTML><body bgcolor=cyan>
<form action="b.jsp" method="post" >
   输入矩形的长和宽:
   长:<input type=text name="length" value=0>
   宽:<input type=text name="width" value=0>
   <Input type=submit value="提交">
</form>
</body></HTML>
```

b.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<jsp:useBean id="rectangle" class="geng.Rectangle" scope="request"/>
<HTML><BODY bgcolor=pink>
<jsp:setProperty name="rectangle" property="*"/>
    矩形的长和宽:
    长:<jsp:getProperty name="rectangle" property="length"/>,
    宽:<jsp:getProperty name="rectangle" property="width"/>,
   <BR>面积是: <jsp:getProperty name="rectangle" property="area"/>
</body></HTML>
```

Rectangle.java:

```
package geng;
public class Rectangle {
   double length=0,width=0,area=-1;
   public double getArea() {
       area=length*width;
       return area;
   }
   public double getLength() {
    return length;
   }
   public void setLength(double length) {
    this.length = length;
   }
   public double getWidth() {
```

```
    return width;
  }
  public void setWidth(double width) {
    this.width = width;
  }
}
```

第 6 章　Java Servlet 基础

1.假设 Web 服务目录 mymoon 中的 JSP 页面要使用一个 servlet，该 servlet 的包名为 blue.sky。请说明，应当怎样保存 servlet 的字节码文件。

答：

1） 在当前 Web 服务目录下建立如下目录结构：

Web 服务目录\WEB-INF\classes

2） 根据类的包名，在目录 classes 下建立相应的子目录，即：

Web 服务目录\WEB-INF\classes\blue\sky

将字节码文件保存在其中。

2.假设 Web 服务目录是 mymoon，star 是 mymoon 的一个子目录， JSP 页面 a.jsp 保存在 star 中，a.jsp 准备请求一个 servlet，该 servlet 的包名为 tom.jiafei。下列哪个叙述是正确的?

A.创建 servlet 的字节码文件保存在\mymoon\WEB-INF\classes\tom\jiafei 中。

B.创建 servlet 的字节码文件保存在\mymoon\star\WEB-INF\classes\tom\jiafei 中。

C.创建 servlet 的字节码文件保存在\mymoon\WEB-INF\star\classes\tom\jiafei 中。

D.创建 servlet 的字节码文件保存在\mymoon\WEB-INF\classes\start\tom\jiafei 中。

答：A

3. 假设 Web 服务目录是 mymoon，star 是 mymoon 的一个子目录， JSP 页面 a.jsp 保存在 star 中，a.jsp 准备请求一个 servlet，该 servlet 的包名为 tom.jiafei。下列哪个叙述是正确的?

A.web.xml 文件保存在\mymoon\WEB-INF\classes 中。

B.web.xml 文件保存在\mymoon\WEB-INF\中。

C.web.xml 文件保存在\mymoon\WEB-INF\star\中。

D.web.xml 文件保存在\mymoon\star\WEB-INF\中。

答：B

4. servlet 对象是在服务器端，还是在客户端创建的?

答：服务器端.

5. servlet 对象被创建后首选调用 init 方法还是 service 方法?

答：init 方法.

6. “servlet 第一次被请求加载时调用 init 方法，当后续的客户请求 servlet 对象时, servlet 对象不再调用 init 方法”， 这样的说法是否正确?

答：正确.

7.servlet 第一次被请求加载后，当后续的客户请求 servlet 对象时，下列哪个叙述是正确的?

A.servlet 调用 service 方法。

B.servlet 调用 init 方法。

C.servlet 调用 doPost 方法。

D.servlet 调用 doGet 方法。

答：A.

8. 假设创建 servlet 的类是 tom.jiafei.Dalian,创建的 servlet 对象的名字是 myservlet，应当怎样配置 web.xml 文件?

```
<?xml version=“1.0” e ncoding="utf-8"?>
```

```
<web-app>
 <servlet> <!--创建一个 servlet 对象  -->
        <servlet-name>myservlet</servlet-name> <!--对象名  -->
          <servlet-class>tom.jiafei.Dalian</servlet-class><!--指定创建对象的 servlet -->
 </servlet>
  <servlet-mapping>
          <servlet-name>myservlet</servlet-name><!--具体的映射路径，前面必须有一个/ -->
            <url-pattern>/lookHello</url-pattern>
  </servlet-mapping>
</web-app>
```

9.　如果 Servlet 类不重写 service 方法，那么应当重写哪两个方法?

答：doGet 方法或 doPost 方法。

10. HttpServletResponse 类的 sendRedirect 方法和 RequestDispatcher 类的 forward 方法有何不同?

答：　javax.servlet.http .HttpServletResponse 提供的重定向方法

　　void sendRedirect(String location) throws IOException

实现让客户端跳转，但跳转的目标页面无法用 request 对象获取用户提交的参数的值。

　　javax.servlet .RequestDispatcher 接口提供的转发方法

　　void　forward(ServletRequest request, ServletResponse response)

使得目标页面可用 request 对象获取用户提交的参数的值。

11. servlet 对象怎样获得用户的会话对象?

答：HttpSession session=request.getSession(true);访问某个 Web 服务目录的用户，在不同的 servlet 中获取的 session 对象是完全相同的，不同的用户的 session 对象互不相同。

12. 编写 inputCircle.jsp，页面提供 form 表单，该 form 表单提供 2 个 text 文本框，用于用户输入圆的圆心（例如（12,34））和圆的半径，用户单击 submit 提交键请求名字是 drawCircle 的 servlet。编写创建 servlet 的 Servlet 类，该类创建的 servlet 可以绘制圆。

（1）JSP 页面

inputCircle.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<style>
   #tom{
      font-family:宋体;font-size:26;color:blue
   }
</style>
<%
String s ="120,120";
%>
<HTML><body id=tom bgcolor=#ffccff>
<form action="drawCircle" id= tom method=post>
输入圆心坐标格式是 m,n( 0<=m<=800,0<=n<=600)<br>
<input type=text name = 'center' id=tom value='<%=s%>'/>
```

```
<input type=text name = 'radius' id=tom value='100'/>
<br><input type=submit id=tom value="提交"/>
</form>
</body></HTML>
```

（2）Servlet

DrawCircle_Servlet.java

```
package handle.data;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.awt.image.BufferedImage;
import java.awt.*;
import java.awt.geom.*;
import javax.imageio.ImageIO;
public class DrawCircle_Servlet extends HttpServlet{
   HttpServletRequest request;
   HttpServletResponse response;
   public void init(ServletConfig config) throws ServletException{
      super.init(config);
   }
   public void service(HttpServletRequest request,
             HttpServletResponse response) throws IOException{
      request.setCharacterEncoding("utf-8");
      String center = request.getParameter("center");
      String radius = request.getParameter("radius");
      if(center == null||center.length()==0){
          response.sendRedirect("inputCircle.jsp");//重定向到输入数据页面。
          return;
      }
      String []str =center.split("[,, ]+");
      double x=0,y=0,r=0;
      try{   x = Double.parseDouble(str[0]);
             y = Double.parseDouble(str[1]);
             r = Double.parseDouble(radius);
      }
      catch(NumberFormatException exp){
         response.sendRedirect("inputCircle.jsp");
         return;
      }
      response.setContentType("image/jpeg");
      Ellipse2D ellipse= new Ellipse2D.Double(x-r,y-r,2*r,2*r);
      BufferedImage image = getImage(ellipse);
      OutputStream outClient= response.getOutputStream();
```

```
      boolean boo =ImageIO.write(image,"jpeg",outClient);
   }
   BufferedImage getImage(Shape shape){ //得到图形的图像。
       int width=1000, height=800;
       BufferedImage image =
       new BufferedImage(width,height,BufferedImage.TYPE_INT_RGB);//图像。
       Graphics g = image.getGraphics();
       g.fillRect(0, 0, width, height);
       Graphics2D g_2d=(Graphics2D)g;
       g_2d.setColor(Color.blue);
       g_2d.draw(shape); //向图像上绘制图形。
       return image;
   }
}
```

（3）web.xml 文件

web.xml

```
<?xml version="1.0" encoding="utf-8"?>
<web-app>
       <servlet>
       <servlet-name>drawCircle</servlet-name>
       <servlet-class>handle.data.DrawCircle_Servlet</servlet-class>
    </servlet>
    <servlet-mapping>
       <servlet-name>drawCircle</servlet-name>
       <url-pattern>/drawCircle</url-pattern>
    </servlet-mapping>
</web-app>;
```

第 7 章　MVC 模式

1. 在 JSP 中,MVC 模式中的数据模型之角色由谁担当?
答：由 Java Bean 充当.
2. 在 JSP 中，MVC 模式中的控制器之角色由谁担当?
答：一个或多个 servlet 对象充当.
3. 在 JSP 中,MVC 模式中的视图之角色由谁担当?
答：由一个或多个 JSP 页面充当.
4. MVC 的好处是什么?
答：MVC 模式的核心思想是有效地组合“视图”、“模型”和“控制器”。在 JSP 技术中，视图是一个或多个 JSP 页面，其作用主要是向控制器提交必要的数据和为模型提供数据显示；模型是一个或多个 Javabean 对象，用于存储数据；控制器是一个或多个 servlet 对象，根据视图提交的要求进行数据处理操作，并将有关的结果存储到 Javabean 中，然后 servlet 使用重定向方式请求视图中的某个 JSP 页面更新显示。
5. MVC 模式中用到的 Javabean 是由 JSP 页面还是 servlet 负责创建?
答：是 servlet 负责。
6. 参照 7.5.1 中的实验设计一个 Web 应用。用户可以通过 JSP 页面输入三角形的三边或梯形的上底、下底和高给一个 servlet 控制器，控制器负责计算三角形和梯形的面积，并将结果存储到数据模型中，然后请求 JSP 页面显示数据模型中的数据。
（1）bean（模型）

```
package mybean.data;
public class Area{
   double a,b,c,area;
   String mess;
   public void setMess(String mess){
      this.mess=mess;
   }
   public String getMess(){
      return mess;
   }
   public void setA(double a){
      this.a=a;
   }
   public void setB(double b){
      this.b=b;
   }
   public void setC(double c){
      this.c=c;
   }
   public void setArea(double s){
```

```
      area=s;
   }
   public double getArea(){
      return area;
   }
}
```

（2）JSP 页面（视图）

inputData.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<HTML><body bgcolor=cyan>
<form action="lookArea" method="post" >
   三角形:
  <br>输入边 A:<input type=text name="a" size=4>
      输入边 B:<input type=text name="b" size=4>
      输入边 C:<input type=text name="c" size=4>
  <input type=submit value="提交">
</form>
<form action="lookArea" method="get" >
   梯形:
  <br>输入上底:<input type=text name="a" size=4>
      输入下底:<input type=text name="b" size=4>
      输入高:  <input type=text name="c" size=4>
  <input type=submit value="提交">
</form>
</body></HTML>
```

showResult.jsp

```
<%@ page contentType="text/html" %>
<%@ page pageEncoding = "utf-8" %>
<%@ page import="mybean.data.Area"%>
<jsp:useBean id="data" type="mybean.data.Area" scope="request"/>
<HTML><body bgcolor=yellow>
   <jsp:getProperty name="data" property="mess"/>:
   <jsp:getProperty name="data" property="area"/>
</body></HTML>
```

（2）servlet（控制器）

HandleArea.java

```
package myservlet.control;
import mybean.data.Area;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
```

```
public class HandleArea extends HttpServlet{
  public void init(ServletConfig config) throws ServletException{
    super.init(config);
  }
  public void doPost(HttpServletRequest request,
        HttpServletResponse response)throws ServletException,IOException{
    Area dataBean=new Area();            //创建 Javabean 对象
    request.setAttribute("data",dataBean);//将 dataBean 存储到 request 对象中
    try{  double a=Double.parseDouble(request.getParameter("a"));
          double b=Double.parseDouble(request.getParameter("b"));
          double c=Double.parseDouble(request.getParameter("c"));
          dataBean.setA(a);          //将数据存储在 dataBean 中
          dataBean.setB(b);
          dataBean.setC(c);
          double s=-1;
          double p=(a+b+c)/2.0;
          if(a+b>c&&a+c>b&&b+c>a)
            s=Math.sqrt(p*(p-a)*(p-b)*(p-c));
          dataBean.setArea(s);        //将数据存储在 dataBean 中
          dataBean.setMess("三角形面积");
    }
    catch(Exception e){
          dataBean.setArea(-1);
          dataBean.setMess(""+e);
    }
    RequestDispatcher
    dispatcher=request.getRequestDispatcher("showResult.jsp");
     //请求 showResult.jsp 显示 dataBean 中的数据:
     dispatcher.forward(request,response);
   }
  public void doGet(HttpServletRequest request,HttpServletResponse response)
                   throws ServletException,IOException{
    Area dataBean=new Area();           //创建 Javabean 对象
    request.setAttribute("data",dataBean);//将 dataBean 存储到 request 对象中
    try{  double a=Double.parseDouble(request.getParameter("a"));
          double b=Double.parseDouble(request.getParameter("b"));
          double c=Double.parseDouble(request.getParameter("c"));
          dataBean.setA(a);          //将数据存储在 dataBean 中
          dataBean.setB(b);
          dataBean.setC(c);
          double s=-1;
          s=(a+b)*c/2.0;
          dataBean.setArea(s);        //将数据存储在 dataBean 中
```

```
            dataBean.setMess("梯形面积");
        }
        catch(Exception e){
            dataBean.setArea(-1);
            dataBean.setMess(""+e);
        }
        RequestDispatcher
        dispatcher=request.getRequestDispatcher("showResult.jsp");
        //请求 showResult.jsp 显示 dataBean 中的数据 :
        dispatcher.forward(request,response);
    }
}
```

第 8 章 JSP 中使用数据库

1. 启动 MySQL 数据库服务器的命令是哪个?
答:进入 MySQL 安装目录的 bin 子目录下键入 mysqld 或 mysqld -nt ,回车确认启动 MySQL 数据库服务器。
2. 启动 MySQL 数据库服务器命令行客户端的命令是哪个?
答:进入 MySQL 安装目录下的 bin 子目录。执行 mysql.exe,即启动命令行客户端。执行格式为:

```
mysql -h ip -u root -p
```

对于本机调试(即客户端和数据库服务器同机),执行格式为:

```
mysql  -u root -p
```

3. 操作数据库之前可以不和数据库建立连接吗。
答:不可以。
4. 查询表中记录的 SQL 语句的基本语句格式是怎样的?
答:select 字段 from 表名
5. 更新表中记录的 SQL 语句的基本语句格式是怎样的?
答:update 表 set 字段 = 新值 where <条件子句>
6. 删除表中记录的 SQL 语句的基本语句格式是怎样的?
答:delete from 表名 where <条件子句>
7. 向表中插入(添加)记录的 SQL 语句的基本语句格式是怎样的?
答:insert into 表(字段列表) values (对应的具体的记录)
8. 加载 MySQL 的 JDBC-数据库连接器的代码是什么?
答:

```
try{  Class.forName("com.mysql.cj.jdbc.Driver ");
}
catch(Exception e){}
```

字符序列和 8.0 版本之前的 com.mysql.jdbc.Driver 不同。
9. 加载 SQL Server 的 JDBC-数据库连接器的代码是什么?
答:

```
try{  Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver ");
}
catch(Exception e){}
```

10. 使用预处理语句的好处是什么?
答:提高效率,代码更加方便灵活。
11. 建立连接池的配置文件的名字是什么?应到保存在哪个目录中。
答:名字必须是 context.xml,保存在 Web 服务目录的 META-INF 子目录中。
12. 使用连接池修改例子 8_2。

答：参见例子 8_2 和例子 8_10。

13. 参照例子 8_2，编写一个查询 Access 数据库的 JSP 页面。

答：参见例子 8_2 和例子 8_9。

14. 使用 MVC 结构，设计一个用户注册的 Web 应用程序。

答：参见 8.12.3 中的注册模块。

第 9 章 JSP 中的文件操作

1. File 对象能读写文件吗?
答：不能。
2. File 对象怎样获取文件的长度?
答：调用 public long length()方法。
3. RandomAccessFile 类创建的流在读写文件时有什么特点?
答：即可以读文件，也可以写文件。
4. 参考 9.6.2 的代码编写一个播放视频的 Web 应用程序。用户通过一个下列列表选择视频的名字，单击提交键可以看到视频的文本介绍，以及播放视频的 GUI 控件。
答：和 9.6.2 的代码及其类似。只不过此处使用的文本文件的内容是电影介绍，多媒体文件是 mp4、avi 等视频格式而已。