# JSP历年编程题库

[JSP历年编程题库](#header-n0)  
 [JDBC](#header-n3)  
 [1.jdbc\_001 按要求编写程序](#header-n4)  
 [2.jdbc\_002 按要求编写程序](#header-n9)  
 [3.sw](#header-n16)*[2019](#header-n16)*[jdbc\_003 编程实现对classroom表的删除操作](#header-n16)  
 [JSP基础](#header-n30)  
 [1.zj](#header-n31)*[Servlet](#header-n31)*[Program2 按要求编写程序](#header-n31)  
 [2.zj](#header-n41)*[Servlet](#header-n41)*[Program1 按要求编写程序](#header-n41)  
 [JSP语法](#header-n54)  
 [1.zj](#header-n55)*[Jsp](#header-n55)*[003 按要求编写javabean和jsp](#header-n55)  
 [2.zj](#header-n65)*[Jsp](#header-n65)*[002 按要求编写jsp](#header-n65)  
 [3.zj](#header-n68)*[Jsp](#header-n68)*[001 按要求编写jsp](#header-n68)  
 [JSTL&EL](#header-n72)  
 [1.2019-sw-jsp-04 按要求使用JSTL和EL，输出用户名列表](#header-n73)  
 [2.2019-sw-jsp-002 按要求创建JSP页面并用EL输出下面的信息](#header-n79)  
 [3.2019-sw-jsp-005 使用EL输出信息：请求的URL、服务器端口以及JESSIONID的值](#header-n86)  
 [Srvlet基础](#header-n94)  
 [1.2018](#header-n95)*[sw](#header-n95)*[swevlet\_005 猜数字程序2(√)](#header-n95)  
 [2.2018](#header-n108)*[sw](#header-n108)*[swevlet\_f001 定义Servlet，用Cookie记录访问时间(√)](#header-n108)  
 [3.期末](#header-n138)*[zj](#header-n138)*[2019](#header-n138)*[servlet](#header-n138)*[002 按要求编写servlet程序(√)](#header-n138)  
 [4.2018](#header-n145)*[sw](#header-n145)*[servlet\_004 按要求编写Servlet处理登录请求](#header-n145)  
 [5.期末](#header-n161)*[zj](#header-n161)*[2019](#header-n161)*[servlet](#header-n161)*[003 按要求编写servlet程序](#header-n161)  
 [6.期末](#header-n170)*[zj](#header-n170)*[2019](#header-n170)*[servlet](#header-n170)*[004 按要求编写servlet程序](#header-n170)  
 [7.期末](#header-n173)*[zj](#header-n173)*[2019](#header-n173)*[servlet](#header-n173)*[005 按要求编写servlet程序](#header-n173)  
 [8.期末](#header-n176)*[zj](#header-n176)*[2019](#header-n176)*[servlet](#header-n176)*[001 按要求编写servlet程序](#header-n176)  
 [内置对象](#header-n184)  
 [1.2019-sw-jsp-006 编写一个计算器页面](#header-n185)

## JDBC

### 1.jdbc\_001 按要求编写程序

编写一个servlet，实现下面的功能：

1.能够显示一个表单界面，用户可以输入图书编号，图书名称，出版社，价格以及作者等信息，表单中有提交按钮，可以将用户输入的信息提交给当前的servlet。

2.能够将用户提交的图书信息保存到数据库表中。

@WebServlet("/jdbc\_001")  
public class jdbc\_001 extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
 public jdbc\_001() {  
 super();  
 }  
  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

PrintWriter out = response.getWriter();  
 out.println("<!DOCTYPE html> \n"+   
 "<html>\n" +  
 "<head><title></title></head>\n" +  
 "<body>" +  
 "<form method=\"post\">" +  
 "<input type=\"text\" name=\"bookNum\">" +  
 "<input type=\"text\" name=\"bookName\">" +  
 "<input type=\"text\" name=\"publish\">" +  
 "<input type=\"text\" name=\"price\">" +  
 "<input type=\"text\" name=\"writer\">" +  
 "<button class=\"submit btn btn-primary\">提交</button>" +  
 "</form>" +  
 "</body>"  
 );  
 }  
  
   
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 String bookNum = request.getParameter("bookNum");  
 String bookName = request.getParameter("bookName");  
 String publish = request.getParameter("publish");  
 String price = request.getParameter("price");  
 String writer = request.getParameter("writer");  
   
 final String sql\_url = "jdbc:mysql://localhost:3306/jspcourse";  
 Connection connection = null;  
 Statement stmt = null;  
 try {  
 //1.实现并注册一个Driver对象  
 Class.forName("com.mysql.jdbc.Driver");//mysql的包  
 //2.创建一个连接到数据库的Connection对象  
 String url = sql\_url;  
 connection = DriverManager.getConnection(url);  
 //4.执行sql语句  
 String sql = "INSERT INTO t\_book(num,name,publish,price,writer) value("+bookNum+","+bookName+","+publish+","+price+","+writer+")";  
 //3.创建一个Statement对象（子接口preparedStatement）  
 stmt = connection.createStatement();  
 //5.处理结果  
 stmt.executeUpdate(sql);  
 /\*ResultSet rSet = stmt.executeQuery(sql);//执行sql语句，返回数据集合  
 \* resultset需要通过循环输出  
 \* while(rSet.next()){  
 System.out.println(rSet.getSting("name"));  
 }  
 \* \*/  
 } catch (ClassNotFoundException | SQLException e) {  
 // TODO Auto-generated catch block  
 e.printStackTrace();  
 }finally {  
 try {  
 //6.关闭相关JDBC对象  
 stmt.close();  
 connection.close();  
 } catch (SQLException e) {  
 // TODO Auto-generated catch block  
 e.printStackTrace();  
 }  
 }  
 }  
  
}

### 2.jdbc\_002 按要求编写程序

编写一个jsp和一个servlet，实现下面的功能：

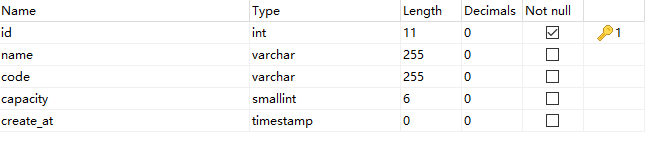
1.在jsp中，具有一个文本框和一个查询按钮，点击查询按钮后，能够根据文本框中用户输入的图书名称进行查询，查询结果在servlet中通过table显示。

2.在servlet中通过table显示查询结果，要求table具有删除的链接，在servlet中能够根据图书的编号进行删除。

jsp  
<%@ page language="java" contentType="text/html; charset=UTF-8"  
 pageEncoding="UTF-8"%>  
<!DOCTYPE html>  
<html>  
<head>  
<meta charset="UTF-8">  
<title>Insert title here</title>  
</head>  
<body>  
<form action="JDBC002.java" method="post">  
 <div>  
 图书名称：<input type="text" name="inputName" /><br>  
 </div>  
 <div>  
 <button>查询</button>  
 </div>  
</form>  
</body>  
</html>

### 3.sw*2019*jdbc\_003 编程实现对classroom表的删除操作

数据库中定义了一个表classroom保存教室信息，表的设计如下：



表说明：id：自增主键；name：教室名称；code教室编码；capacity：教室容量；create\_at：教室数据创建时间

在MySQL中创建classroom表的sql语句：

DROP TABLE IF EXISTS `classroom`;  
  
 CREATE TABLE `classroom` (  
  
 `id` int(11) NOT NULL AUTO\_INCREMENT,  
  
 `name` varchar(255) DEFAULT NULL,  
  
 `code` varchar(255) DEFAULT NULL,  
  
 `capacity` smallint(6) DEFAULT NULL,  
  
 `create\_at` timestamp NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,  
  
 PRIMARY KEY (`id`)  
  
 ) ENGINE=InnoDB AUTO\_INCREMENT=1015 DEFAULT CHARSET=utf8;

要求：

1.根据上表classroom定义实体类Classroom。

2。实现下面ClassroomDao中的deleteById(int id)方法，实现根据id从classroom表中删除一条数据，ClassroomDao中将用到定义的实体类Classroom。

//定义实体类Classroom  
/\*\*\*\*\*\*start\*\*\*\*\*\*/  
  
  
  
  
/\*\*\*\*\*\*end\*\*\*\*\*\*/  
public class ClassroomDao{  
/\*\*\*\*\*\*start\*\*\*\*\*\*/  
  
  
//实现根据id删除一条教室数据，返回删除成功的数据条数  
//该方法要求使用PreparedStatement实现，否则分数减半  
public int deleteById(int id){  
  
  
  
  
  
/\*\*\*\*\*\*end\*\*\*\*\*\*/  
}  
  
}

package servlets;  
  
import java.sql.\*;  
  
import com.sun.jmx.snmp.Timestamp;  
  
//定义实体类Classroom  
/\*\*\*\*\*\*start\*\*\*\*\*\*/  
class Classroom{  
 private int id;  
 private String name;  
 private String code;  
 private int capacity;  
 private Timestamp create\_at;  
 public int getId() {  
 return id;  
 }  
 public void setId(int id) {  
 this.id = id;  
 }  
 public String getName() {  
 return name;  
 }  
 public void setName(String name) {  
 this.name = name;  
 }  
 public String getCode() {  
 return code;  
 }  
 public void setCode(String code) {  
 this.code = code;  
 }  
 public int getCapacity() {  
 return capacity;  
 }  
 public void setCapacity(int capacity) {  
 this.capacity = capacity;  
 }  
 public Timestamp getCreate\_at() {  
 return create\_at;  
 }  
 public void setCreate\_at(Timestamp create\_at) {  
 this.create\_at = create\_at;  
 }  
   
}  
/\*\*\*\*\*\*end\*\*\*\*\*\*/  
public class ClassroomDao {  
 /\*\*\*\*\*\*start\*\*\*\*\*\*/  
 private final String sql\_url = "jdbc:mysql://localhost:3306/jspcourse";  
  
 //实现根据id删除一条教室数据，返回删除成功的数据条数  
 //该方法要求使用PreparedStatement实现，否则分数减半  
 public int deleteById(int id){  
 Connection connection = null;  
 PreparedStatement stmt = null;  
 try {  
 //1.实现并注册一个Driver对象  
 Class.forName("com.mysql.jdbc.Driver");//mysql的包  
 //2.创建一个连接到数据库的Connection对象  
 String url = sql\_url;  
 connection = DriverManager.getConnection(url);  
 //4.执行sql语句  
 String sql = "DELECT FROM classroom WHERE id = ?";  
 //3.创建一个Statement对象（子接口preparedStatement）  
 stmt = connection.prepareStatement(sql);  
 //5.处理结果  
 int row = stmt.executeUpdate(sql);  
 /\*ResultSet rSet = stmt.executeQuery(sql);//执行sql语句，返回数据集合  
 \* resultset需要通过循环输出  
 \* while(rSet.next()){  
 System.out.println(rSet.getSting("name"));  
 }  
 \* \*/  
 return row;  
 } catch (ClassNotFoundException | SQLException e) {  
 // TODO Auto-generated catch block  
 e.printStackTrace();  
 return 0;  
 }finally {  
 try {  
 //6.关闭相关JDBC对象  
 stmt.close();  
 connection.close();  
 } catch (SQLException e) {  
 // TODO Auto-generated catch block  
 e.printStackTrace();  
 }  
 }  
   
 }  
  
  
  
  
 /\*\*\*\*\*\*end\*\*\*\*\*\*/  
}

package dao;  
  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.PreparedStatement;  
import java.sql.SQLException;  
import java.sql.Timestamp;  
  
class Classroom{  
 private int id;  
 private String name;  
 private String code;  
 private int capacity;  
 private Timestamp create\_at;  
 public int getId() {  
 return id;  
 }  
 public void setId(int id) {  
 this.id = id;  
 }  
 public String getName() {  
 return name;  
 }  
 public void setName(String name) {  
 this.name = name;  
 }  
 public String getCode() {  
 return code;  
 }  
 public void setCode(String code) {  
 this.code = code;  
 }  
 public int getCapacity() {  
 return capacity;  
 }  
 public void setCapacity(int capacity) {  
 this.capacity = capacity;  
 }  
 public Timestamp getCreate\_at() {  
 return create\_at;  
 }  
 public void setCreate\_at(Timestamp create\_at) {  
 this.create\_at = create\_at;  
 }  
   
}  
public class ClassroomDao {  
 private final String URL = "jdbc:mysql://localhost:3306/jspcourse";  
 private final String DB\_USER = "root";  
 private final String DB\_PWD = "123456";  
 public int deleteById(int id) {  
 Connection connection = null;  
 PreparedStatement pstmt = null;  
 try {  
 Class.forName("com.mysql.jdbc.Driver");  
 connection = DriverManager.getConnection(URL, DB\_USER, DB\_PWD);  
 String sql = "DELETE FROM classroom WHERE id = ?";  
 pstmt = connection.prepareStatement(sql);  
 pstmt.setInt(1, id);  
 return pstmt.executeUpdate();  
 } catch (ClassNotFoundException e) {  
 // TODO Auto-generated catch block  
 e.printStackTrace();  
 return 0;  
 } catch (SQLException e) {  
 // TODO Auto-generated catch block  
 e.printStackTrace();  
 return 0;  
 }finally {  
 try {  
 connection.close();  
 pstmt.close();  
 } catch (SQLException e) {  
 // TODO Auto-generated catch block  
 e.printStackTrace();  
 }  
 }  
 }  
}

## JSP基础

### 1.zj*Servlet*Program2 按要求编写程序

编写两个Listener（ContextParam和OnLineUser）按要求实现以下的功能：

1. 在ContextParam中实现当ServletContext对象初始化之后，在ServletContext对象中设置一个属性UserCount，属性的值为0。（8分）
2. 在OnLineUser中实现当有新的会话对象产生后，对ServletContext对象中的UserCount实现加1操作。（6分）
3. 在OnLineUser中实现当有会话对象销毁时，对ServletContext对象中的UserCount实现减1操作。（6分）

ContextParam.java  
  
package servlets;  
  
import javax.servlet.ServletContext;  
import javax.servlet.ServletContextAttributeEvent;  
import javax.servlet.ServletContextAttributeListener;  
import javax.servlet.ServletContextEvent;  
import javax.servlet.ServletContextListener;  
import javax.servlet.annotation.WebListener;  
  
/\*\*  
 \* Application Lifecycle Listener implementation class ContextParam  
 \*  
 \*/  
@WebListener  
public class ContextParam implements ServletContextListener, ServletContextAttributeListener {  
  
 /\*\*  
 \* Default constructor.   
 \*/  
 public ContextParam() {  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see ServletContextAttributeListener#attributeAdded(ServletContextAttributeEvent)  
 \*/  
 public void attributeAdded(ServletContextAttributeEvent scae) {   
 // TODO Auto-generated method stub  
 }  
  
 /\*\*  
 \* @see ServletContextAttributeListener#attributeRemoved(ServletContextAttributeEvent)  
 \*/  
 public void attributeRemoved(ServletContextAttributeEvent scae) {   
 // TODO Auto-generated method stub  
 }  
  
 /\*\*  
 \* @see ServletContextListener#contextDestroyed(ServletContextEvent)  
 \*/  
 public void contextDestroyed(ServletContextEvent sce) {   
 // TODO Auto-generated method stub  
 }  
  
 /\*\*  
 \* @see ServletContextAttributeListener#attributeReplaced(ServletContextAttributeEvent)  
 \*/  
 public void attributeReplaced(ServletContextAttributeEvent scae) {   
 // TODO Auto-generated method stub  
 }  
  
 /\*\*  
 \* @see ServletContextListener#contextInitialized(ServletContextEvent)  
 \*/  
 public void contextInitialized(ServletContextEvent sce) {   
 // TODO Auto-generated method stub  
 ServletContext application = sce.getServletContext();  
 application.setAttribute("UserCount", 0);  
 }  
   
}  
  
  
OnLineUser.java  
  
package servlets;  
  
import javax.servlet.ServletContext;  
import javax.servlet.annotation.WebListener;  
import javax.servlet.http.HttpSession;  
import javax.servlet.http.HttpSessionEvent;  
import javax.servlet.http.HttpSessionListener;  
  
/\*\*  
 \* Application Lifecycle Listener implementation class OnLineUser  
 \*  
 \*/  
@WebListener  
public class OnLineUser implements HttpSessionListener {  
  
 /\*\*  
 \* Default constructor.   
 \*/  
 public OnLineUser() {  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see HttpSessionListener#sessionCreated(HttpSessionEvent)  
 \*/  
 public void sessionCreated(HttpSessionEvent se) {   
 // TODO Auto-generated method stub  
 // TODO Auto-generated method stub  
 HttpSession session = se.getSession();  
 ServletContext application = session.getServletContext();  
 int UserCount = (Integer)application.getAttribute("UserCount");  
   
 UserCount++;  
   
 application.setAttribute("UserCount", UserCount);  
 }  
  
 /\*\*  
 \* @see HttpSessionListener#sessionDestroyed(HttpSessionEvent)  
 \*/  
 public void sessionDestroyed(HttpSessionEvent se) {   
 // TODO Auto-generated method stub  
 HttpSession session = se.getSession();  
 ServletContext application = session.getServletContext();  
 int UserCount = (Integer)application.getAttribute("UserCount");  
   
 UserCount--;  
   
 application.setAttribute("UserCount", UserCount);  
 }  
   
}

### 2.zj*Servlet*Program1 按要求编写程序

编写一个Servlet（UserLogin）按要求实现以下的功能：

1. 通过注解的方式定义Servlet，Servlet映射的地址为/Login.do, Servlet的初始化参数为：UserName=admin,UserPwd=adminpassword.(5分)
2. doPost方法中进行请求参数（username和userpwd）的获取以及和初始化参数的比较。(8分)
3. 用户名和密码验证通过后，将username的值和请求时间（当前服务器时间）写入Session对象。（4分）
4. 比较后进行相应的跳转（success.view或error.view）（3分）。

package servlets;  
  
import java.io.IOException;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
import javax.servlet.ServletConfig;  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebInitParam;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import javax.servlet.http.HttpSession;  
  
/\*\*  
 \* Servlet implementation class UserLogin  
 \*/  
@WebServlet(  
 urlPatterns = "/UserLogin",  
 initParams = {  
 @WebInitParam(name = "UserName",value = "admin"),  
 @WebInitParam(name = "UserPwd",value = "adminpassword"),  
 })  
public class UserLogin extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
   
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public UserLogin() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 doPost(request, response);  
 }  
  
 /\*\*  
 \* @see #doPost(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 String inputName = request.getParameter("username");  
 String inputPwd = request.getParameter("userpwd");  
 String userName = getInitParameter("UserName");  
 String userPwd = getInitParameter("UserPwd");  
   
 if(inputName.equals(userName) && inputPwd.equals(userPwd)) {  
 HttpSession session = request.getSession();  
 //获取当前时间  
 SimpleDateFormat format = new SimpleDateFormat("hh:mm:ss");  
 String time = format.format(new Date());  
   
 session.setAttribute("username", userName);  
 session.setAttribute("time", time);  
   
 request.getRequestDispatcher("success.view").forward(request, response);  
 }else {  
 request.getRequestDispatcher("error.view").forward(request, response);  
 }  
 }  
  
}

## JSP语法

### 1.zj*Jsp*003 按要求编写javabean和jsp

1.编写一个javabean，具有username和userpwd两个属性。 2.编写一个jsp(login.jsp)，包含一个表单(要用户输入用户名了密码)，表单提交到logincheck.jsp.

1. 编写一个jsp(logincheck.jsp)使用动作元素接收用户提交的请求参数,并判断用户名和密码是否等于admin和admin123，如果判断为真，将用户名和密码保存到session，并跳转到welcom.jsp.
2. 编写一个jsp(welcom.jsp)，显示session中保存的用户信息。

javabean.java  
package javaBean;  
  
import java.io.Serializable;  
  
public class userBean implements Serializable{  
 private String username = null;  
 private String userpwd = null;  
   
 public userBean() {  
 // TODO Auto-generated constructor stub  
 super();  
 }  
  
 public String getUsername() {  
 return username;  
 }  
  
 public void setUsername(String username) {  
 this.username = username;  
 }  
  
 public String getUserpwd() {  
 return userpwd;  
 }  
  
 public void setUserpwd(String userpwd) {  
 this.userpwd = userpwd;  
 }  
   
   
}

login.jsp  
<%@ page language="java" contentType="text/html; charset=utf-8"  
 pageEncoding="utf-8"%>  
<!DOCTYPE html>  
<html>  
<head>  
<meta charset="utf-8">  
<title>Insert title here</title>  
</head>  
<body>  
<form action="logincheck.jsp" method="post">  
 <div>  
 用户名：<input type="text" name="name"><br>  
 </div>  
 <div>  
 密码：<input type="password" name="pwd"><br>  
 </div>  
 <div>  
 <button>登录</button>  
 </div>  
 </form>  
</body>  
</html>

logincheck.jsp  
  
<%@ page language="java" contentType="text/html; charset=utf-8"  
 pageEncoding="utf-8"%>  
<jsp:useBean id="user" class="javaBean.userBean"/>  
<jsp:setProperty name="user" property="username" param="name"/>  
<jsp:setProperty name="user" property="userpwd" param="pwd"/>  
<!-- 使用jsp动作元素获取参数时，通过建立JavaBean，给JavaBean赋值完成。  
需注意：赋值时，name为设置的javaBean的id，property是要设置的javaBean的属性，  
param="param\_name"用于接收参数为param\_name的值，并赋值给property，即param是将JavaBean实例对象的某个属性值设置为一个请求参数值  
value指定JavaBean对象的某个属性的值，value的值可以是字符串（自动转化为JavaBean属性匹配的类型），也可以是表达式（写入的是表达式的结果）  
-->  
<%  
 String name = user.getUsername();  
 String pwd = user.getUserpwd();  
 if(name.equals("admin") && pwd.equals("admin123")){  
 session.setAttribute("userName", name);  
 session.setAttribute("userPwd", pwd);  
 request.getRequestDispatcher("welcom.jsp").forward(request, response);  
 }  
 else{  
 out.println("error");  
 out.println(user.getUsername());  
 out.println(user.getUserpwd());  
 }  
%>

welcom.jsp  
<%@ page language="java" contentType="text/html; charset=utf-8"  
 pageEncoding="utf-8"%>  
<%  
 String name = String.valueOf(session.getAttribute("userName"));  
 String pwd = String.valueOf(session.getAttribute("userPwd"));  
 out.println(name);  
 out.println(pwd);  
%>

### 2.zj*Jsp*002 按要求编写jsp

1.定义一个长度为5的字符串数组，使用EL表达式输出每个元素。 2.使用EL表达式获取请求参数名为“UserName”的请求参数，并输出。 3.使用EL表达式获取session中名为“logtime”的字符串变量，并输出。 4.使用EL表达式获取ServeltContext的名为“admin”的初始参数，并输出。

<%@ page language="java" contentType="text/html; charset=utf-8"  
 pageEncoding="utf-8"%>  
<!-- 定义一个长度为5的字符串数组，使用EL表达式输出每个元素 -->  
<%  
 String[] nums = {"0","1","2","3","4"};  
 request.setAttribute("nums", nums);//保存变量  
%>  
${nums[0]}<br>  
${nums[1]}<br>  
${nums[2]}<br>  
${nums[3]}<br>  
${nums[4]}<br>  
  
<!-- 使用EL表达式获取请求参数名为“UserName”的请求参数，并输出 -->  
${param.UserName }<br>  
<!-- 使用EL表达式获取session中名为“logtime”的字符串变量，并输出 -->  
<% session.setAttribute("logtime","logtime1");%>  
${logtime }<br>  
<!-- 使用EL表达式获取ServeltContext的名为“admin”的初始参数，并输出 -->  
<% application.setAttribute("admin","aaa");%>  
${admin }<br>

### 3.zj*Jsp*001 按要求编写jsp

编写一个jsp网页，判断用户的请求中是否有名称为“autologin”的cookie信息，如果有该cookie，则请求跳转（不是请求重定向）到"welcom.view"，否则显示一个登陆界面。

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"  
 pageEncoding="ISO-8859-1"%>  
<%  
 Cookie[] cookies = request.getCookies();  
 for(Cookie cookie:cookies){  
 if(cookie.getName().equals("autologin")){  
 request.getRequestDispatcher("welcome.view").forward(request, response);  
 }  
 }  
 request.getRequestDispatcher("login.view");  
%>

## JSTL&EL

### 1.2019-sw-jsp-04 按要求使用JSTL和EL，输出用户名列表

有一个Servlet代码如下：

import java.io.IOException;  
  
import java.util.ArrayList;  
  
import java.util.List;  
  
import javax.servlet.ServletException;  
  
import javax.servlet.annotation.WebServlet;  
  
import javax.servlet.http.HttpServlet;  
  
import javax.servlet.http.HttpServletRequest;  
  
import javax.servlet.http.HttpServletResponse;  
  
@WebServlet("/users.view")  
  
public class UserServlets extends HttpServlet {  
  
 private List<String> userNames;  
  
 public UserServlets(){  
  
 userNames = new ArrayList<>();  
  
 userNames.add("Tomcat");  
  
 userNames.add("JSP");  
  
 userNames.add("Jerry");  
  
 }  
  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
  
 // TODO Auto-generated method stub  
  
 request.setAttribute("userNames", userNames);  
  
 request.getRequestDispatcher("users.jsp").forward(request, response);  
  
 }  
  
}

现要求你定义users.jsp页面，在users.jsp页面中输出该Servlet发过来的用户名列表，要求在users.jsp页面中使用JSTL和EL实现该功能。

<%@ page anguage="java" contentType="text/html; charset=utf-8"  
 pageEncoding="utf-8"%>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
<!DOCTYPE html>  
<html>  
<head>  
<meta charset="utf-8">  
<title>Insert title here</title>  
</head>  
<body>  
<c:forEach items="${userName}" var="username">  
${username }  
</c:forEach>  
</body>  
</html>

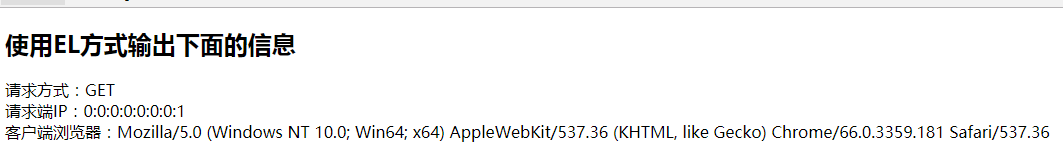
### 2.2019-sw-jsp-002 按要求创建JSP页面并用EL输出下面的信息

现有一个HTML代码如下：

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  
  
<html>  
  
<head>  
  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
  
<title>Insert title here</title>  
  
</head>  
  
<body>  
  
<h2>使用EL方式输出下面的信息</h2>  
  
请求方式：\*\*\*\*\*  
  
<br>  
  
请求端IP：\*\*\*\*\*\*  
  
<br>  
  
客户端浏览器：\*\*\*\*\*\*  
  
</body>  
  
</html>

现要求：

1、将该HTML页面改为JSP页面，设置page指令，支持页面显示中文；

2、在**\*\***处添加EL脚本，输出相应的信息，只能使用EL，并且填写EL代码时删除掉**\*\***，最终页面运行结果显示如下 ：

<%@ page language="java" contentType="text/html; charset=UTF-8"  
 pageEncoding="UTF-8"%>  
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  
  
<html>  
  
<head>  
  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
  
<title>Insert title here</title>  
  
</head>  
  
<body>  
  
<h2>使用EL方式输出下面的信息</h2>  
  
请求方式：${pageContext.request.method}  
<!-- 获取请求方式 -->  
<br>  
  
请求端IP：${pageContext.request.remoteAddr}  
<!-- 获取请求端IP地址 -->  
  
<br>  
  
客户端浏览器：${header["user-agent"]}  
<!--用户浏览器的版本-->  
  
</body>  
  
</html>

### 3.2019-sw-jsp-005 使用EL输出信息：请求的URL、服务器端口以及JESSIONID的值

现有一个HTML页面如下：

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  
  
<html>  
  
<head>  
  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
  
<title>EL</title>  
  
</head>  
  
<body>  
  
<h2>使用EL输出下面的信息</h2>  
  
请求的URI：  
  
<br>  
  
服务器响应端口：  
  
<br>  
  
名字为JSESSIONID的 Cookie的值：  
  
</body>  
  
</html>

要求将上述的HTML页面修改为JSP页面：

1、设置page指令，能支持页面的中文编码和输出；

2、按要求在指定位置使用EL输出响应的信息，输出结果参考下图，所有值都是以你的项目为准，下列信息仅供参考。

<%@ page language="java" contentType="text/html; charset=utf-8"  
 pageEncoding="utf-8"%>  
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  
  
<html>  
  
<head>  
  
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  
  
<title>EL</title>  
  
</head>  
  
<body>  
  
<h2>使用EL输出下面的信息</h2>  
  
请求的URI：${pageContext.request.requestURL}   
  
<br>  
  
服务器响应端口：${pageContext.request.serverPort}  
  
<br>  
  
名字为JSESSIONID的 Cookie的值：${cookie.JSESSIONID.value}  
  
</body>  
  
</html>

## Srvlet基础

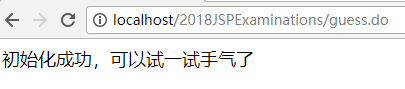
### 1.2018*sw*swevlet\_005 猜数字程序2(√)

实现一个网页猜数字程序，猜数字页面（guess.html）代码如下：

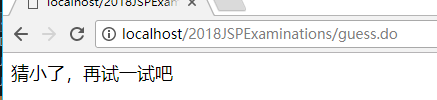
<html>  
  
<body>  
  
<h3>猜数字，猜测0~20之间的一个整数</h3>  
  
<form action="guess.do" method="post">  
  
 我猜是：<input type="text" name="number" value="0"><br>  
  
 <input type="submit" value="试一下手气">  
  
 </form>  
  
</body>  
  
</html>

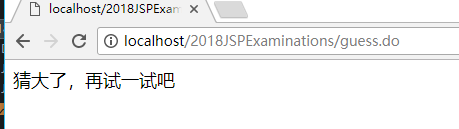
要求定义一个Servlet实现验证操作，Servlet的类名自定义，访问路径见guess.html，使用HttpSession对象实现；

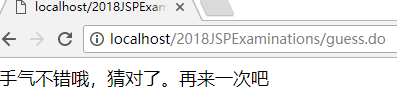
具体要求有：

1、用户第一次点击“试一下手气”时，在Servlet中初始化一个0~20之间的随机整数，并将此整数作为本次猜数字的答案并保存下来，再输出信息：“初始化成功，可以试一试手气了”，3秒后自动跳转到guess.html页面，开始猜测；

2、用户输入本次猜测的数字，点击“试一下手气”，Servlet收到本次猜测的数字后与答案对比，并将猜测结果显示出来：







如果没有猜对，在显示结果后的3秒钟后返回到猜测页面重写猜测；

如果猜对了，将保存的答案销毁，3秒后返回猜测页面开始新的一次猜测。

package servlets1;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.util.Random;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import javax.servlet.http.HttpSession;  
  
/\*\*  
 \* Servlet implementation class guessServlet  
 \*/  
@WebServlet("/guess.do")  
public class guessServlet1 extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
   
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public guessServlet1() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
   
 /\*\*  
 \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 response.setContentType("text/html;charset=UTF-8");  
 PrintWriter out = response.getWriter();  
   
 //获取输入的数字且保存为int类型  
 String input = request.getParameter("number");  
 int inputNum = Integer.parseInt(input);  
   
 //获取随机数  
 Random random = new Random();  
 int rightNum = random.nextInt(20);  
   
 //初始化  
 HttpSession session = request.getSession();  
 if(session.getAttribute("rightNum") == null) {  
 session.setAttribute("rightNum", rightNum);  
 out.println("初始化成功，可以试一试手气了");  
 response.setHeader("Refresh", "3;url='guess.html'");  
 }  
 //开始猜测  
 else {  
 if (inputNum != rightNum) {  
 if(inputNum < rightNum)  
 out.println("猜小了，再试一试吧");  
 else if(inputNum > rightNum)  
 out.println("猜大了，再试一试吧");  
 else {  
 out.println("error");  
 }  
 }  
 else {  
 out.println("手气不错哦，猜对了，再来一次吧");  
 session.invalidate();  
 }  
 response.setHeader("Refresh", "3;url='guess.html'");  
 }  
 }  
  
}

### 2.2018*sw*swevlet\_f001 定义Servlet，用Cookie记录访问时间(√)

* 定义一个Servlet，每次访问该Servlet时，使用Cookie保存用户此次的访问时间，并输出用户上一次访问时间。
* 输出示例：
* 如果是首次访问，输出：“您是第一次访问本页面”；
* 如果不是首次访问，则输出：“您上一次访问本页面的时间是：xxxx”，xxx为时间信息，格式是：小时：分：秒
* 具体要求：
* 1、上述要求使用Cookie实现；
* 2、该Servlet的访问路径是：visitTime.view，请用注解方式定义；
* 3、设置输出Servlet的响应类型为html，并支持中文输出；
* 提示：cookie中保存的信息不能有空格，可使用下面代码获取并格式化当前时间信息time：
  + SimpleDateFormat format = new SimpleDateFormat("hh:mm:ss");
  + String time = format.format(new Date());

/\*- 定义一个Servlet，每次访问该Servlet时，使用Cookie保存用户此次的访问时间，并输出用户上一次访问时间。  
- 输出示例：  
- 如果是首次访问，输出：“您是第一次访问本页面”；  
- 如果不是首次访问，则输出：“您上一次访问本页面的时间是：xxxx”，xxx为时间信息，格式是：小时：分：秒  
- 具体要求：  
- 1、上述要求使用Cookie实现；  
- 2、该Servlet的访问路径是：visitTime.view，请用注解方式定义；  
- 3、设置输出Servlet的响应类型为html，并支持中文输出；  
-   
- 提示：cookie中保存的信息不能有空格，可使用下面代码获取并格式化当前时间信息time：  
 - SimpleDateFormat format = new SimpleDateFormat("hh:mm:ss");  
 - String time = format.format(new Date());  
  
 \*/  
  
package servlets;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.Cookie;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
/\*\*  
 \* Servlet implementation class visitTime  
 \*/  
@WebServlet("/visitTime.view")  
public class visitTime extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
   
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public visitTime() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 response.setContentType("text/html;charset=UTF-8");  
 PrintWriter out = response.getWriter();  
   
 SimpleDateFormat format = new SimpleDateFormat("hh:mm:ss");  
 String time = format.format(new Date());  
   
 Cookie[] cookies = request.getCookies();  
 String lastTime = null;  
   
 for(Cookie cookie: cookies) {  
 if(cookie.getName().equals("lastTime")) {  
 lastTime = cookie.getValue();  
 break;  
 }  
 }  
   
 if(lastTime == null) {  
 out.println("您是第一次访问本页面");  
 }else {  
 out.println("您上一次访问本页面的时间是："+lastTime);  
 }  
   
 Cookie cookie = new Cookie("lastTime", time);  
 response.addCookie(cookie);  
 }  
  
 /\*\*  
 \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 doGet(request, response);  
 }  
  
}

import java.io.IOException;  
import java.io.PrintWriter;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.Cookie;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
/\*\*  
 \* Servlet implementation class Sw\_2018\_servlet\_f001  
 \*/  
@WebServlet("/visitTime.view")  
public class Sw\_2018\_servlet\_f001 extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
   
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public Sw\_2018\_servlet\_f001() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 response.setContentType("text/html;charset=utf-8");  
 PrintWriter out = response.getWriter();  
 Cookie[] cookies = request.getCookies();  
 String lastVisitTime = null;  
 for (Cookie cookie : cookies) {  
 if (cookie.getName().equals("lastVisitTime")) {  
 lastVisitTime = cookie.getValue();  
 break;  
 }  
 }  
 if (lastVisitTime == null) {  
 out.println("您是第一次访问本页面");  
 }else {  
 out.println("您上一次访问本页面的时间是:"+lastVisitTime);  
 }  
 SimpleDateFormat format = new SimpleDateFormat("hh:mm:ss");  
 String time = format.format(new Date());  
 Cookie cookie = new Cookie("lastVisitTime", time);  
 response.addCookie(cookie);  
 }  
  
}

### 3.期末*zj*2019*servlet*002 按要求编写servlet程序(√)

1. 通过注解的方式定义Servlet，Servlet映射的地址为/userLog.do, Servlet的初始化参数为：param1=“admin”,param2=”admin”.(4分)
2. Init方法中取出初始化参数param1和param2，并在doPost方法中输出。（6分）

package servlets1;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
  
import javax.servlet.ServletConfig;  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebInitParam;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
/\*\*  
 \* Servlet implementation class userLogServlet3  
 \*/  
@WebServlet(  
 name = "userLogServlet3",  
 initParams = {  
 @WebInitParam(name = "param1",value = "admin"),  
 @WebInitParam(name = "param2",value = "admin"),  
 },  
 urlPatterns = "/userLog.do")  
public class userLogServlet3 extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
 String param1 = null;  
 String param2 = null;  
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public userLogServlet3() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see Servlet#init(ServletConfig)  
 \*/  
 public void init(ServletConfig config) throws ServletException {  
 // TODO Auto-generated method stub  
 param1 = config.getInitParameter("param1");  
 param2 = config.getInitParameter("param2");  
 }  
  
 /\*\*  
 \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 PrintWriter out = response.getWriter();  
 out.println(param1);  
 out.println(param2);  
 }  
  
}

### 4.2018*sw*servlet\_004 按要求编写Servlet处理登录请求

有一个登录页面（login.html），其登录表单的HTML代码如下：

<form action="login.do" method="post">  
   
 用户名：<input type="text" name="userName"><br>  
 密码： <input type="password" name="pwd"><br>  
 <input type="submit" value="登录">  
  
 </form>

定义一个Servlet（类名自定义）实现登录验证，具体要求如下：

1、正确的用户名和密码保存在ServletContext的初始化参数中，ServletContext中的参数如下，请自行将下面的配置放置到web.xml文件中，再获取相应的参数值；

userName

JSP

pwd

JSP

2、验证登录用户提交的用户名和密码是否正确，如果正确，将用户名添加为HttpSession的一个属性，再将请求内部派发到home.html；

3、如果用户名和密码不正确，则将请求重定向到login.html

package servlets1;  
  
import java.io.IOException;  
import javax.servlet.ServletConfig;  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import javax.servlet.http.HttpSession;  
  
import com.sun.org.apache.xpath.internal.operations.And;  
  
/\*\*  
 \* Servlet implementation class loginServlet4  
 \*/  
@WebServlet("/login.do")  
public class loginServlet4 extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
 private String userName= null;  
 private String pwd = null;  
   
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public loginServlet4() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see Servlet#init(ServletConfig)  
 \*/  
 public void init(ServletConfig config) throws ServletException {  
 // TODO Auto-generated method stub  
 userName = config.getInitParameter("userName");  
 pwd = config.getInitParameter("pwd");  
 }  
  
 /\*\*  
 \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 response.getWriter().append("Served at: ").append(request.getContextPath());  
 }  
  
 /\*\*  
 \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 String inputUserName = request.getParameter("userName");  
 String inputPwd = request.getParameter("pwd");  
   
 if(inputUserName.equals(userName) && inputPwd.equals(pwd)) {  
 HttpSession session = request.getSession();  
 session.setAttribute("userName", inputUserName);  
 request.getRequestDispatcher("home.html").forward(request, response);  
 }else {  
 response.sendRedirect("login.html");  
 }  
 }  
  
}

### 5.期末*zj*2019*servlet*003 按要求编写servlet程序

1. 在doGet方法中获取客户端传输的cookie参数：c1。（2分）
2. 将c1和请求参数r1比较，如果相等，将r1的值通过cookie的方式传回客户端，cookie的名称为“flag”。（4分）
3. 如果不相等，请求重定向到test.do。（2分）

package servlets1;  
  
import java.io.IOException;  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.Cookie;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
  
/\*\*  
 \* Servlet implementation class servlet5  
 \*/  
@WebServlet("/servlet5")  
public class servlet5 extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
   
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public servlet5() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 String r1 = request.getParameter("r1");  
 String c1 = null;  
 String value = null;  
 boolean flag = true;  
 Cookie[] cookies = request.getCookies();  
   
 for(int i=0; i < cookies.length; i++) {  
 Cookie cookie = cookies[i];  
 c1 = cookie.getValue();  
 if(c1.equals(r1)) {  
 Cookie cookie2 = new Cookie("flag",r1);  
 response.addCookie(cookie2);  
 flag = false;  
 break;  
 }  
 }  
 if(flag = true)  
 response.sendRedirect("test.do");  
 }  
  
 /\*\*  
 \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 doGet(request, response);  
 }  
  
}

### 6.期末*zj*2019*servlet*004 按要求编写servlet程序

设计一个监听器，当有新的会话产生时，实现ServletContext对象中属性“LoginUserCount”的加1操作。

package servlets1;  
  
import javax.servlet.annotation.WebListener;  
import javax.servlet.http.HttpSessionAttributeListener;  
import javax.servlet.http.HttpSessionBindingEvent;  
import javax.servlet.http.HttpSessionEvent;  
import javax.servlet.http.HttpSessionListener;  
  
/\*\*  
 \* Application Lifecycle Listener implementation class Listener6  
 \*  
 \*/  
@WebListener  
public class Listener6 implements HttpSessionListener, HttpSessionAttributeListener {  
  
 /\*\*  
 \* Default constructor.   
 \*/  
 public Listener6() {  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see HttpSessionListener#sessionCreated(HttpSessionEvent)  
 \*/  
 public void sessionCreated(HttpSessionEvent se) {   
 // TODO Auto-generated method stub  
   
 }  
  
 /\*\*  
 \* @see HttpSessionListener#sessionDestroyed(HttpSessionEvent)  
 \*/  
 public void sessionDestroyed(HttpSessionEvent se) {   
 // TODO Auto-generated method stub  
 }  
  
 /\*\*  
 \* @see HttpSessionAttributeListener#attributeAdded(HttpSessionBindingEvent)  
 \*/  
 public void attributeAdded(HttpSessionBindingEvent event) {   
 // TODO Auto-generated method stub  
 }  
  
 /\*\*  
 \* @see HttpSessionAttributeListener#attributeRemoved(HttpSessionBindingEvent)  
 \*/  
 public void attributeRemoved(HttpSessionBindingEvent event) {   
 // TODO Auto-generated method stub  
 }  
  
 /\*\*  
 \* @see HttpSessionAttributeListener#attributeReplaced(HttpSessionBindingEvent)  
 \*/  
 public void attributeReplaced(HttpSessionBindingEvent event) {   
 // TODO Auto-generated method stub  
 }  
   
}

### 7.期末*zj*2019*servlet*005 按要求编写servlet程序

1、Servlet能够输出一个表单，表单中包括：文本框Question，文本框Answer，以及一个提交按钮。（4分） 2、Servlet能够处理提交按钮提交的请求，将Answer写入Session中，并通过cookie的方式将Answer发送到客户端。最后通过请求调派的方式转发到“Ques.view”.（6分）

package servlets1;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.Cookie;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import javax.servlet.http.HttpSession;  
import javax.websocket.Session;  
  
/\*\*  
 \* Servlet implementation class outPutFrom7  
 \*/  
@WebServlet("/outPutFrom7")  
public class outPutFrom7 extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
   
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public outPutFrom7() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 response.setContentType("text/html;charset=utf-8");  
 PrintWriter out =response.getWriter();  
 out.print("<!DOCTYPE html> \n"+   
 "<html>\n" +  
 "<head><title></title></head>\n" +  
 "<body>" +  
 "<form method=\"get\">" +  
 "<input type=\"text\" name=\"Question\">" +  
 "<input type=\"text\" name=\"Answer\">" +  
 "<button class=\"submit btn btn-primary\">提交</button>" +  
 "</form>" +  
 "</body>"  
 );  
 String inputQuestion = request.getParameter("Question");  
 String inputAnswer = request.getParameter("Answer");  
 //把answer写入session  
 HttpSession session = request.getSession();  
 session.setAttribute("Answer", inputAnswer);  
 //通过cookie方式将answer发送到客户端  
 Cookie cookie = new Cookie("Answer", inputAnswer);  
 response.addCookie(cookie);  
 //通过请求调派的方式转发  
 request.getRequestDispatcher("Ques.view");  
 }  
  
 /\*\*  
 \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 doGet(request, response);  
 }  
  
}

### 8.期末*zj*2019*servlet*001 按要求编写servlet程序

编写一个servlet，要求实现以下功能：

1、通过注解的方式定义Servlet，Servlet映射的地址为/Test.do。（2分）

2、在doGet方法中获取请求参数p1，并在网页上输出。（2分）

3、获取当前的系统时间，并写入Session。（4分）

4、将p1通过cookie的方式发送到客户端。（2分）

package servlets1;  
  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.text.SimpleDateFormat;  
import java.util.Date;  
  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.Cookie;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import javax.servlet.http.HttpSession;  
  
/\*\*  
 \* Servlet implementation class testServlet8  
 \*/  
@WebServlet("/Test.do")  
public class testServlet8 extends HttpServlet {  
 private static final long serialVersionUID = 1L;  
   
 /\*\*  
 \* @see HttpServlet#HttpServlet()  
 \*/  
 public testServlet8() {  
 super();  
 // TODO Auto-generated constructor stub  
 }  
  
 /\*\*  
 \* @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
 PrintWriter out = response.getWriter();  
 String p1 = request.getParameter("p1");  
 out.println(p1);  
   
 //获取当前系统时间  
 SimpleDateFormat format = new SimpleDateFormat("hh:mm:ss");  
 String time = format.format(new Date());  
 //将当前系统时间写入session  
 HttpSession session = request.getSession();  
 session.setAttribute("time", time);  
   
 //将p1通过cookie方式发送到客户端  
 Cookie cookie = new Cookie("p1", p1);  
 response.addCookie(cookie);  
 }  
  
 /\*\*  
 \* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  
 \*/  
 protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
 // TODO Auto-generated method stub  
   
 }  
  
}

## 内置对象

### 1.2019-sw-jsp-006 编写一个计算器页面

要求实现一个基于Web的简单计算器页面，现已有静态的计算器页面calculator.html，代码如下：

<!DOCTYPE html>  
  
<html>  
  
<head>  
  
<meta charset="UTF-8">  
  
<title>计算器</title>  
  
</head>  
  
<body>  
  
 <form action="calculator.jsp" method="post">  
  
 <input type="text" name="op1"> <select name="operator">  
 <option value="+">+</option>  
  
 <option value="-">-</option>  
  
 <option value="\*">\*</option>  
  
 <option value="/">/</option>  
  
 </select> <input type="text" name="op2"> <input type="submit"  
  
 value="计算">  
  
 </form>  
  
</body>  
  
</html>

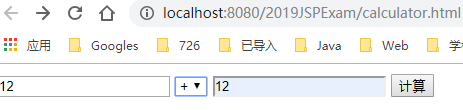
**要求：**

1、编写一个calculator.jsp页面，在该页面获取calculator.html提交的数据，输出计算结果；

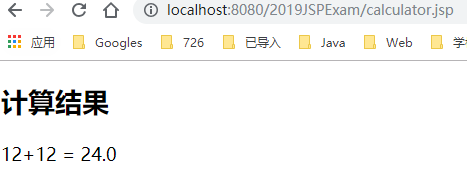
2、编写一个错误显示页面：error.jsp，当calculator.jsp页面在运行过程中抛出异常，将自动跳转到error.jsp页面，并在error.jsp页面输出异常信息，要求使用@page指令设置实现。

**运行示例：**

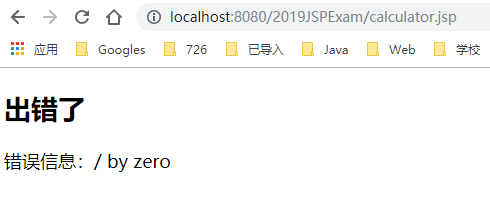
1、calculator.html页面



2、calculator.jsp页面显示运行结果：



3、错误信息显示页面：



calculator.jsp  
  
<%@ page language="java" contentType="text/html; charset=UTF-8"  
 pageEncoding="UTF-8" errorPage = "error.jsp"%>  
<h1>计算结果</h1>  
<%  
//这里必须用int类型才会跳转到error页面  
 int param1 = Integer.parseInt(request.getParameter("op1"));  
 int param2 = Integer.parseInt(request.getParameter("op2"));  
 String operator = request.getParameter("operator");  
 double result = 0;  
 switch(operator){  
 case "+":  
 result = param1 + param2;  
 break;  
 case "-":  
 result = param1 - param2;  
 break;  
 case "\*":  
 result = param1 \* param2;  
 break;  
 case "/":  
 result = param1 / param2;  
 break;  
 default:  
 break;  
 }  
 out.println(param1+operator+param2+"="+result);//用于在页面输出，若是System.out.print则是在控制台输出  
%>  
  
  
  
error.jsp  
<%@ page language="java" contentType="text/html; charset=utf-8"  
 pageEncoding="UTF-8" isErrorPage = "true"  
%>  
<!DOCTYPE html>  
<html>  
<head>  
<meta charset="utf-8">  
<title>Insert title here</title>  
</head>  
<body>  
<h1>出错了</h1>  
错误信息：<%=exception.getMessage() %>  
</body>  
</html>