**Лабораторная работа №9**

# СТАНДАРТНЫЕ ЭЛЕМЕНТЫ ACTION

Стандартные элементы action (действия) выглядят как обычные теги, название которых начинается с сочетания символов **jsp:** ,что согласно терминологии XML означает принадлежность данному пространству имен.

Синтаксис элементов action:

|  |
| --- |
| <jsp:action attributes /> |

Основные стандартные элементы action:

* jsp:useBean
* jsp:setProperty
* jsp:getProperty
* jsp:include
* jsp:forward
* jsp:plugin
* jsp:params
* jsp:param
* jsp:fallback

Рассмотрим встроенные объекты подробнее.

**jsp:useBean**

Элемент позволяет использовать экземпляр компонента JavaBeans. При его выполнении сервер приложений обеспечивает поиск экземпляра данного java-класса, пользуясь значениями, определенными в атрибутах. С помощью действия jsp:useBean можно:

* найти существующий компонент JavaBeans;
* создать новый экземпляр класса JavaBeans;

задать локальное имя объекту, определенному в другой странице JSP или сервлете.

Атрибуты тега jsp:useBean:

* id – идентификатор, применяемый для ссылки на компонент JavaBeans из страницы JSP;
* scope – атрибут, определяющий область видимости ссылки на экземпляр объекта JavaBeans (значения: page, request, session, application);
* class – параметр, определяющий полное имя класса реализации объекта;
* beanName – наименование класса реализации объекта. Данный параметр используется, если компонент JavaBeans еще не существует;
* type – тип специфицированного объекта. С помощью этого атрибута можно избежать автоматической инициализации компонента JavaBeans, если он еще не существует в данной области видимости.

Если имя класса (class) и наименование объекта (beanName) не определены, объект должен быть представлен в заданной области видимости. Значение идентификатора id должно быть уникально в текущем модуле трансляции (JSP-странице).

**jsp:setProperty**

Элемент позволяет устанавливать значения полей - свойств компонента JavaBeans, который должен быть предварительно создан действием jsp:useBean.

Атрибуты тега jsp:setProperty:

* name – параметр, идентифицирующий экземпляр объекта JavaBeans, свойство которого устанавливается текущим тегом;
* property – имя свойства, которому необходимо определить значение;
* param – имя параметра запроса, который передается свойству компонента JavaBeans;
* value – значение устанавливаемого свойства.

Значения свойств компонента JavaBeans устанавливается с учетом соответствия типов значения и свойства.

**jsp:getProperty**

Элемент делает свойства компонента JavaBeans видимыми, позволяет получить значения полей указанного объекта, преобразует его в строку и отправляет в выходной поток.

Атрибуты тега jsp:getProperty:

* name – параметр, идентифицирующий экземпляр объекта JavaBeans;
* property – имя свойства, значение которого необходимо получить.

**jsp:include**

Элемент позволяет включать статические и динамические ресурсы в контекст текущей страницы JSP. Ресурс определяется по его относительному URL-адресу, который интегрируется в контекст веб-сервера. В отличие от директивы include, которая включает файл на этапе трансляции страницы JSP, действие jsp:include производит вставку файла или ресурса при запросе страницы.

Атрибуты тега jsp:include:

* page – параметр, идентифицирующий URL-адрес ресурса (определяется относительно текущей страницы JSP);
* flush – атрибут, управляющий переполнением буфера (является необязательным). Если этот атрибут имеет значение true и выходной поток страницы JSP буферизуется, то буфер при переполнении освобождается, в противном случае – не освобождается.

**jsp:forward**

Элемент позволяет перенаправить запрос другой странице JSP, статическому ресурсу или классу Java-сервлета, находящемуся в том же контексте, что и текущая страница JSP. Атрибуты тега jsp:forward аналогичны атрибутам jsp:include.

Действие jsp:forward может включать элементы jsp:param, которые предоставляют значения для некоторых параметров запроса, используемые для перенаправления.

**jsp:plugin**

Элемент заменяет соответствующие клиентскому браузеру конструкции OBJECT или EMBED, которые приводят к выполнению компонентов Java-модуля (JavaBeans или апплета).

Атрибуты тега jsp:include:

* type – тип объекта (компонент JavaBeans или апплет);
* code – код (class-файл) объекта;
* codebase – расположение объекта;
* height и width – размер объекта по высоте и ширине;
* align – выравнивание объекта;
* archive – список архивов;

другие атрибуты в соответствии со спецификацией HTML.

**jsp:param**

Элемент используется для предоставления информации ключ/значение в элементах jsp:include, jsp:plugin, jsp:params и jsp:forward.

Атрибуты тега jsp:param:

* name – наименование параметра;
* value – значение параметра.

**jsp:params**

Элемент позволяет группировать параметры внутри стандартного тега jsp:plugin.

**jsp:fallback**

Элемент указывает содержимое, которое будет использоваться браузером клиента, если подключаемый модуль элемента jsp:plugin не сможет запуститься. Действие jsp:fallback, как и jsp:params, должно быть определено внутри тега jsp:plugin и вне него не применяется.

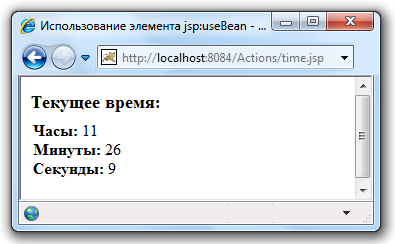
***Лабораторная работа***

***Цель работы:*** знакомство со стандартными элементами action, создание страницы JSP для описания процессов их реализации.

1. Создайте новый проект с названием **Actions**.
2. Создайте страницу **time.jsp**.

|  |
| --- |
| <%@ page language="java" contentType="text/html"  pageEncoding="UTF-8"%>  <html>  <head>  <title> Использование элемента jsp:useBean </title>  </head>  <body>  <jsp:useBean id="clock" class="java.util.Date" />  <h3> Текущее время: </h3>  <b> Часы: </b>  <jsp:getProperty name="clock" property="hours"/> <br>  <b> Минуты: </b>  <jsp:getProperty name="clock" property="minutes"/> <br>  <b> Секунды: </b>  <jsp:getProperty name="clock" property="seconds"/>  </body>  </html> |

1. Запустите страницу **time.jsp** на выполнение:



1. Создайте страницы **book1.html** и **book2.html** с информацией о книгах:
   1. сформируйте программный код страницы **book1.html**:

|  |
| --- |
| <html>  <head>  <title> Автор Перри Б. </title> |
| <meta http-equiv="Content-Type" content="text/html;  charset=windows-1251">  </head>  <body>  <b> Перри Б. </b> Java сервлеты и JSP: сборник рецептов. Изд. 3-е / Пер. с англ. – М.: КУДИЦ-ПРЕСС, 2009. – 768 с.  </body>  </html> |

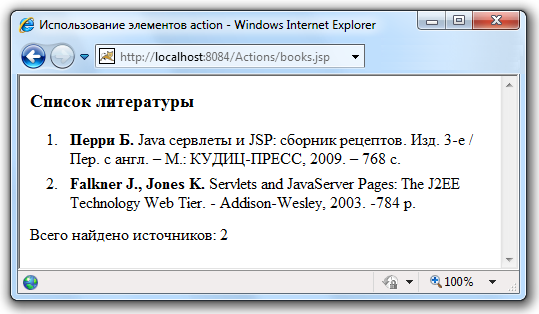
* 1. сформируйте программный код блока <**body**>**…**<**/body**> страницы **book2.html**:

|  |
| --- |
| <body>  <b> Falkner J., Jones K. </b> Servlets and JavaServer Pages: The J2EE Technology Web Tier. - Addison-Wesley, 2003. - 784 p.  </body> |

1. Создайте страницу **books.jsp**:

|  |
| --- |
| <%@ page language="java" contentType="text/html"  pageEncoding="UTF-8"%>  <html>  <head>  <title> Использование элементов action </title>  </head>  <body>  <h3> Список литературы </h3>  <ol>  <li> <jsp:include page="book1.html"/>  <li> <jsp:include page="book2.html"/>  </ol>  <jsp:declaration> int num = 2; </jsp:declaration>  <jsp:text> Всего найдено источников: </jsp:text>  <jsp:expression> num </jsp:expression>  </body>  </html> |

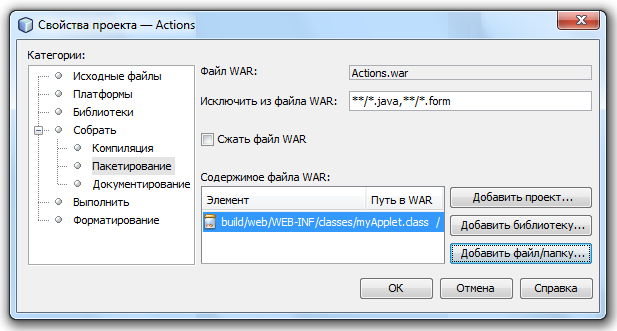
1. Запустите страницу **books.jsp** на выполнение и просмотрите результат работы в браузере:



1. Создайте Java-апплет с названием **myApplet:**
2. выполните команду **Файл → Создать файл**;
3. выберите в списке **Категории** пункт **Java**;
4. выберите в списке **Типы файлов** пункт **Апплет** и перейдите к следующему шагу;
5. введите имя **myApplet** в строку **Имя класса**;
6. нажмите на кнопку **Готово** для завершения создания апплета.
7. Сформируйте программный код апплета:

|  |
| --- |
| import java.awt.\*;  import java.awt.event.\*;  import java.applet.\*;  import java.util.Random;  public class myApplet extends Applet  implements ActionListener {  int num=0;  Button but=new Button("Прогноз погоды");  public void init() {  but.addActionListener(this);  add(but);  } |
| public void paint(Graphics g){  switch (num){  case 1: {  //солнце  g.setColor(Color.black);  g.drawString("Солнечно",10,20);  g.drawRect(20, 60, 100, 100);  g.setColor(new Color(247, 148, 29));  g.fillOval(45, 85, 50, 50);  g.drawLine(40, 80, 100, 140);  g.drawLine(40, 140, 100, 80);  g.drawLine(30, 110, 110, 110);  g.drawLine(70, 70, 70, 150);  break;  }  case 2: {  //облако  g.setColor(Color.black);  g.drawString("Облачно",10,20);  g.drawRect(20, 60, 100, 100);  g.setColor(new Color(109, 207, 246));  g.fillOval(30, 110, 83, 20);  g.fillArc(30, 105, 30, 30, 0, 180);  g.fillArc(45, 90, 50, 50, 0, 180);  g.fillArc(78, 103, 35, 35, 0, 180);  break;  }  case 3: {  g.setColor(Color.black);  g.drawString("Переменная",10,20);  g.drawString("облачность",20,40);  g.drawRect(20, 60, 100, 100);  g.setColor(new Color(247, 148, 29)); |
| g.fillOval(45, 85, 50, 50);  g.drawLine(40, 80, 100, 140);  g.drawLine(40, 140, 100, 80);  g.drawLine(30, 110, 110, 110);  g.drawLine(70, 70, 70, 150);  g.setColor(new Color(109, 207, 246));  g.fillOval(30, 120, 83, 20);  g.fillArc(30, 115, 30, 30, 0, 180);  g.fillArc(45, 100, 50, 50, 0, 180);  g.fillArc(78, 113, 35, 35, 0, 180);  break;  }  }  }  public void actionPerformed(ActionEvent e){  Random random = new Random();  num = random.nextInt(3)+ 1;  repaint();  }  } |

1. Скомпилируйте файл **myApplet.java**:
2. нажмите на название файла в окне **Проекты** правой кнопкой мыши;
3. выберите пункт контекстного меню **Компилировать файл**;
4. для того, чтобы просмотреть результат работы апплета выберите пункт контекстного меню **Выполнить файл**.
5. Подключите класс **myApplet** к проекту:
6. нажмите на название проекта в окне **Проекты** правой кнопкой мыши;
7. выберите пункт контекстного меню **Свойства**;
8. в списке **Категории** выберите **Пакетирование**;
9. нажмите на кнопку **Добавить файл/папку**;
10. перейдите к папке, которая содержит class-файл апплета **myApplet.class**, и выберите его:



1. Создайте страницу **appletJSP.jsp**:

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

<html>

<head>

<meta http-equiv="Content-Type"

content="text/html; charset=UTF-8">

<title> Внедрение мультимедиа в JSP </title>

</head>

<body>

<jsp:plugin type="applet" code="myApplet"

width="370" height="420">

</jsp:plugin>

</body>

</html>

1. Запустите страницу **appletJSP.jsp** на выполнение и просмотрите результат работы в браузере.

