Министерство образования и науки РФ

ФГБОУ ВПО «Тамбовский государственный технический университет»

Кафедра «САПР»

Лабораторная работа №7

по дисциплине «Интернет технологии»

«Сервлеты»

Выполнил: студент гр. Д. Ю.

Проверил: А. А.

Тамбов,

**Цель работы:** понимание принципов работы серверных программ языка Java, настройка и запуск тестового примера.

**Задание:**

Разработать сервлет (серверное java-приложение), реализующий определенную вычислительную задачу (выбор произвольный, рекомендуются вычислительные задачи из других дисциплин).

Основные условия:

* использование html-форм для инициализации запроса к серверному сценарию и передачи параметров для решения задачи (реализация связи html-страница → сервлет);
* работа с массивами;
* работа с файлами (чтение, запись).

Решение:

Запуск сервлета – это некий квест, к решению которого можно подходить по разному. У меня вышли следующие шаги:

1. Скачать последний сервер Tomcat (<http://tomcat.apache.org/download-80.cgi>);
2. Скачать последний пакет Java Dev Kit (<http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>);
3. Скачать IDE Intellij IDEA ([~~http://www.filetie.com/2014/11/jetbrains-intellij-idea-14-with-keygen.html~~](http://www.filetie.com/2014/11/jetbrains-intellij-idea-14-with-keygen.html));
4. Установить все это и выполнить hello\_word (<http://devcolibri.com/1043>);

После этих шагов принимаемся за что покруче. Как некий пункт отправки, можно выделить то, что сервлет – это тот же php по принципу работы, обрабатываем данные методами Get, Post, которые переопределяем под свои нужды, а логика для программы (сценария) пишется как обычная java, разница лишь в методах снятия\отправки данных.

Общий вид программы изображен на рисунке 1. Схема будет регистрировать нового студента в системе дистанционного образования, его данные вносятся в базу студентов, по завершении этапа регистрации предоставляется выбор доступных для изучения курсов, они подгружаются с базы курсов. При выборе идет показ вводной картинки того или иного курса.

Код index.html в таблице 1.

Рисунок Схема работы

|  |
| --- |
| Таблица 1. Index.html |
| <!DOCTYPE **html**> <**html**> <**head**>  <**meta charset="windows-1251" http-equiv="content-type" content="text/html"**>  <**title**>Регистрация</**title**> </**head**> <**body**>  <**h2 align="center"**>СДО</**h2**>  <**p**>Для регистрации занесите сведения о себе в следующие поля:</**p**>  <**br**>  <**form method="post" action="/s"**>  <**pre**>  Фамилия:<**input type="text" size="40" name="surname"**>  Имя:<**input type="text" size="40" name="name"**>  Отчество:<**input type="text" size="40" name="secname"**>  E-mail:<**input type="text" size="40" name="addr"**>   <**input type="submit" value="Зарегистрировать"**>  </**pre**>  </**form**> </**body**> </**html**> |

Форма посылает сетвлету s\_servlet четыре параметра: surname, name, secname и addr по HTTP-методу post. Сервлет должен принять их, обработать и послать клиенту результаты запроса или замечания по регистрации. Код сервлета приведен в таблице 2. Страница регистрации показана на рисунке 2.

|  |
| --- |
| Таблица 2. Сервлет регистрации клиента СДО |
| **package** servlets;  **import** javax.servlet.annotation.WebServlet; **import** javax.servlet.http.HttpServlet; **import** javax.servlet.http.HttpServletRequest; **import** javax.servlet.http.HttpServletResponse; **import** java.io.PrintWriter; **import** java.sql.\*; **import** java.util.StringTokenizer;  @WebServlet(**"/s"**) **public class** MyServlet **extends** HttpServlet {  **private** String **driver** = **"com.mysql.jdbc.Driver"**;  **private** Connection **con**;  **private** PreparedStatement **pst**;   @Override  **public void** init()  {  **try** {  *//Регистрируем драйвер с помощью статического метода registerDriver класса DriverManager:* DriverManager.*registerDriver*(**new** com.mysql.jdbc.Driver());  Class.*forName*(**driver**);   **con** = DriverManager.*getConnection*(  **"jdbc:mysql://localhost:3306/test"**, **"root"**, **""**);   **pst** = **con**.prepareStatement(**"INSERT INTO students (id, name, address) "** +  **"VALUES(id, ?, ?)"**);   }**catch** (Exception e)  {  System.***err***.println(**"\*\*\*\*\*\*\*\*\*\*\*From init(): "**+ e);  }  }   @Override  **protected void** doGet(HttpServletRequest req, HttpServletResponse resp)  {  doPost(req, resp);  }   @Override  **protected void** doPost(HttpServletRequest req, HttpServletResponse resp)  {  **try** {  req.setCharacterEncoding(**"Cp1251"**);  String surname = req.getParameter(**"surname"**);  String name = req.getParameter(**"name"**);  String secname = req.getParameter(**"secname"**);  String addr = req.getParameter(**"addr"**);  resp.setContentType(**"text/html;charset=windows-1251"**);  PrintWriter pw = resp.getWriter();  **if**(surname.length() \* name.length() \* secname.length() \* addr.length() == 0)  {  pw.println(**"<html><head>"**);  pw.println(**"<title>Продолжение регистрации</title>"**);  pw.println(**"</head><body><h2 align=center>"**+**"Дистанционная система обучения СДО</h2>"**);  pw.println(**"<h3>Замечание:</h3>"**);  pw.println(**"Заполните пожалуйста все поля.<br>"**);  pw.println(**"</body></html>"**);  pw.flush();  pw.close();  **return**;  }  String fullname = surname.trim() + **" "** + name.trim() + **" "** + secname.trim();  **pst**.setString(1, fullname);  **pst**.setString(2, addr);  **int** count = **pst**.executeUpdate();  java.sql.Statement st = **con**.createStatement();  ResultSet rs = st.executeQuery(**"SELECT id, name FROM students ORDER BY id DESC"**);  rs.next();  **int** id = rs.getInt(1);  fullname = rs.getString(2);  rs.close();   StringTokenizer sttok = **new** StringTokenizer(fullname);  sttok.nextToken();  name = sttok.nextToken();  secname = sttok.nextToken();   pw.println(**"<html><head>"**);  pw.println(**"<title>Регистрация</title>"**);  pw.println(**"</head><body><h2 align=center>"**+**"Дистанционная система обучения СДО</h2>"**);  pw.println(**"Добро пожаловать, "**+ name +**" "**+ secname +**"!<br>"**);  pw.println(**"Вы зарегистрированы в СДО.<br>"**);  pw.println(**"Ваш регистрационный номер "**+ id + **"<br>"**);  pw.println(**"Выберите учебный курс:<br>"**);   rs = st.executeQuery(**"SELECT course\_name FROM courses"**);  pw.println(**"<form method=get action="** + **"\"/CoursesServlet\">"**);  pw.println(**"<select size=5 name=courses>"**);  temp = **"get-->"**;**int** index = 0;   **while** (rs.next())  {  pw.println(**"<option value="** + index + **">"** + rs.getString(1) + **"</option>"**);  index++;  }   pw.println(**"</select>"**);  pw.println(**"<input type=submit value=\"Подписаться\" >"**);  pw.println(**"</form></body></html>"**);   pw.flush();  pw.close();  rs.close();  }**catch** (Exception e)  {  System.err.println(**"++++++++++++error: "**+e);  }  }   **public static** String temp = **""**;   @Override  **public void** destroy()  {  **try** {  pst.close();  con.close();  } **catch** (SQLException e)  {  e.printStackTrace();  }  } } |

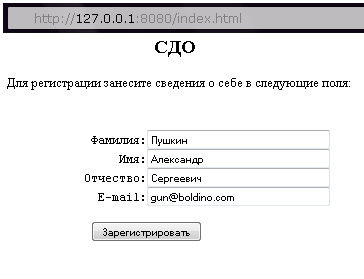
В таблице 2 все запросы пользуются одним и тем же соединением с базой данных. При большом количестве одновременных запросов это может снизить производительность системы и даже превысить допустимое число соединений. В таком случае при инициализации сервлета надо в методе init() создать пул соединений с тем, чтобы запросы брали соединения из этого пула и возвращали соединение при своем завершении.

Рисунок Страница регистрации

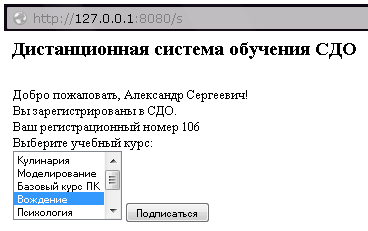
По коду из таблицы 2 можно пояснить следующие моменты:

Рисунок Страница подтверждения регистрации

req.setCharacterEncoding(**"Cp1251"**);

*По умолчанию кодировкой отклика является utf-8. Метод* ***setCharacterEncoding()*** *позволит определить кодировку HTTP-отклика. Для того, чтобы использовать в исходном HTML-коде кириллицу, достаточно вызвать* ***response.setCharacterEncoding("Cp1251")****.*

String surname = req.getParameter(**"surname"**);

*Данный метод получает параметр из HTTP-запроса.*

resp.setContentType(**"text/html;charset=windows-1251"**);

*Метод* ***setContentType()*** *определяет тип MIME для HTTP-отклика. Здесь типом MIME является* ***text/html****. Он показывает, что браузеру необходимо интерпретировать содержимое в качестве исходного HTML-кода.*

PrintWriter pw = resp.getWriter();

*Метод* ***getWriter()*** *получает* ***PrintWriter****. Все, что будет передаваться в поток, будет посылаться клиенту как часть HTTP-отклика. Метод* ***println()*** *используется для записи некоторого простого исходного HTML-кода в качестве HTTP-отклика.*

Последним описывается сервлет (таблица 3) вывода изображения курса, на который подписался обучаемый.

|  |
| --- |
| Таблица 3. Сервлет, отправляющий изображение клиенту |
| **package** servlets;  **import** javax.servlet.ServletContext; **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** java.io.\*; **import** java.util.Vector;  @WebServlet(**"/CoursesServlet"**) **public class** specCourses **extends** HttpServlet {  Vector **imFiles** = **new** Vector();  **private int curIndex**;   @Override  **public void** init()  {  File imDir = **null**;  String imDirName = **"D:\\Java Proj\\MyFirstWebProject\\src\\main\\webapp\\res"**;*//getInitParameter("res");//imagedir* **if**(imDirName != **null**) imDir = **new** File(imDirName);  System.***err***.println(**"Имя ресурса: "**+imDirName+**" имя папки:"**+imDir);   **if**((imDir != **null**)&& imDir.exists()&&imDir.isDirectory())  {  String[] files = imDir.list();   **for** (**int** i = 0; i < files.**length**; i++)  {  **if**(files[i].endsWith(**".jpg"**)||files[i].endsWith(**".gif"**))  {  File curFile = **new** File(imDir, files[i]);  **imFiles**.addElement(curFile);  }  }  }  **else** {  log(**"Не найдено изображений в каталоге: "**+imDirName);  }  }   @Override  **protected void** doGet(HttpServletRequest req, HttpServletResponse resp)  {  **try** {  System.***err***.println(**"вход в пост"**);  doPost(req, resp);  } **catch** (ServletException e) {  e.printStackTrace();  } **catch** (IOException e) {  e.printStackTrace();  }  }   @Override  **protected void** doPost(HttpServletRequest req, HttpServletResponse resp)  **throws** ServletException, IOException  {  System.***err***.println(**"вернуло"**+MyServlet.*temp*);  String surname = **new** String(req.getParameter(**"courses"**));   System.***err***.println(**"Выбрано из списка - "**+ surname);   **curIndex** = Integer.*parseInt*(surname);   **int** len = **imFiles**.size();  **if**(len>0)  {  *//Какой первыым:* File curFile = (File)**imFiles**.elementAt(**curIndex**);  String fileName = curFile.getName();  ServletContext ctxt = getServletConfig().getServletContext();  String ctype = ctxt.getMimeType(fileName);   **if**(ctype==**null**)  ctype = fileName.endsWith(**".jpg"**)?**"image/jpeg"**:**"image/gif"**;   resp.setContentType(ctype);  **try** {  BufferedInputStream bis = **new** BufferedInputStream(**new** FileInputStream(curFile));  OutputStream os = resp.getOutputStream();   **int** cur = 0;   **while** ((cur = bis.read()) != -1)  os.write(cur);   os.close();  bis.close();  }**catch** (FileNotFoundException e)  {  resp.sendError(HttpServletResponse.***SC\_NOT\_FOUND***);  System.***err***.println(**"++++++++++++error: "**+e);  }**catch** (Exception e)  {  resp.sendError(HttpServletResponse.***SC\_SERVICE\_UNAVAILABLE***);  System.***err***.println(**"++++++++++++error : "**+e);  }  **curIndex** = (**curIndex**+1)%len;  }  **else** {  resp.sendError(HttpServletResponse.***SC\_SERVICE\_UNAVAILABLE***);  System.***err***.println(**"++++++++++++error else"**);  resp.setContentType(**"text/html"**);  }  }   **public long** getLastModified()  {  **return** System.*currentTimeMillis*();  } } |

String surname = **new** String(req.getParameter(**"courses"**));

Цепляемся за значение параметра выбранного курса,

**curIndex** = Integer.*parseInt*(surname);

переводим в целое

String imDirName = **"D:\\Java Proj\\MyFirstWebProject\\src\\main\\webapp\\res"**

указываем где искать файлы изображений. Имена файлов такие же как и у атрибута courses, цифра от 0 до 6, примитивно, но работает.

Поздравляю, вы успешно подписаны на курс вождения!