**ЛАБОРАТОРНАЯ РАБОТА №3**. Работа с серверными классами

**Цель работы.** Познакомиться с техникой использования сервлетов – классов Java на стороне сервера.

**Краткое теоретическое содержание**. В теоретическом материале курса представлены сведения по работе с сервлетами. Нас интересует поиск различных информационных объектов на стороне сервера. Клиентская часть обычно представляется сайтом (документом Html). В качестве примера возьмем следующий сайт:

<!DOCTYPE html>

<html>

<head>

<title></title>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

</head>

<body bgcolor="#aaccff">

<Font color="green" size="10">

Форма для работы со словарем

</Font>

<br>

<br>

<form name="frm" method="Get" action="MyServlet">

<Font color="blue" size="6"> Введите русское слово:</Font><Input type="Text" name="txt" value=""/>

<br>

<br>

<Font color="blue" size="6">Перевод: </Font><input type="text" name ="trans" value=" "/><br>

<h4>Кликни здесь для получения перевода :<Input type="submit" value="Перевести"/>

</h4>

</form>

</body>

</html>Конец формы

Отметим, что воспроизведение русского текста обеспечивается строкой

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">.

В тексте сайта выполняется вызов сервлета с именем MyServlet. Это имя указывается в атрибуте action="MyServlet". Вызов сервлета выполняется по нажатию на кнопку с типом type="submit". При открытии сайт выглядит следующим образом (рисунок 1):

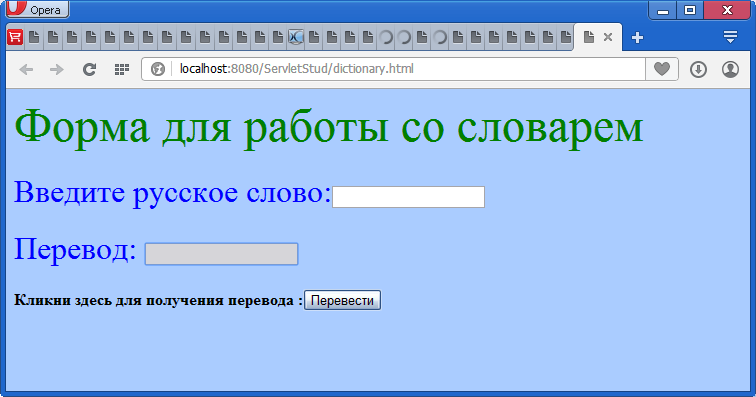


Рисунок 1. Страница сайта

Нужно создать сервлет, обрабатывающий данную форму. Для построения приложения подобного типа следует использовать шаблон Java Web. Структура такого проекта состоит из следующих узлов (пример на рисунке 2)

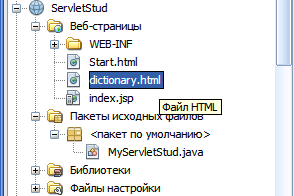


Рисунок 2. Структура проекта

В узле WEB-INF помещают файлы jsp и html. В узле Пакеты исходных файлов помещают сервлеты и классы Java. В узел Библиотеки добавляют jar-файлы и библиотеки с требуемыми классами Java (если необходимо). Нам также понадобится изменить конфигурационный файл в узле Файлы и настройки.

Будем считать, что документ dictionary.html (текст которого приведен выше) создан. Создаем теперь класс сервлета. Для этого открываем контекстное меню щелчком правой кнопки мыши на узле <пакет по умолчанию>, выбираем пункт Создать и опцию Сервлет. Указываем имя класса сервлета.

Класс сервлета содержит два важнейших метода – doGet и doPost. Оба предназначены для обработки данных от клиентской формы. Эти методы вызываются в зависимости от атрибута method тега form документа html:

<form name="frm" method="Get" action="MyServlet">

В методе Get данные клиентской формы передаются одним сообщением вместе со служебной информацией (заголовком). В методе Post данные разбиваются на несколько пакетов. Оба этих метода «перенаправлены» на один метод processRequest. Именно в последнем методе мы и «сосредоточим» всю необходимую обработку:

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException

{

response.setContentType("text/html;charset=UTF-8");

PrintWriter out = response.getWriter();

String rus\_word=""+request.getParameter("txt");

**. . . . . . . . . . .**

Здесь приведена стартовая часть метода processRequest. У метода есть два встроенных объекта − request,response, с помощью которых можно получить значения элементов формы клиентского сайта и направить их обратно в клиентский сайт. Видим, как читается значение параметра

String rus\_word=""+request.getParameter("txt");

Здесь txt − это имя элемента клиентской формы

<Input type="Text" name="txt" value=""/>

Отправка значений обратно клиенту поясняется следующим фрагментом

PrintWriter out = response.getWriter();

out.println("<html>");

out.println("<head>");

out.println("<title>Servlet MyServletStud</title>");

out.println("</head>");

out.println("<body bgcolor='#aaccff'");

out.println("<form>");

out.println("<h2> Привет клиенту!!!</h2><br><br>");

out.println("</form>");

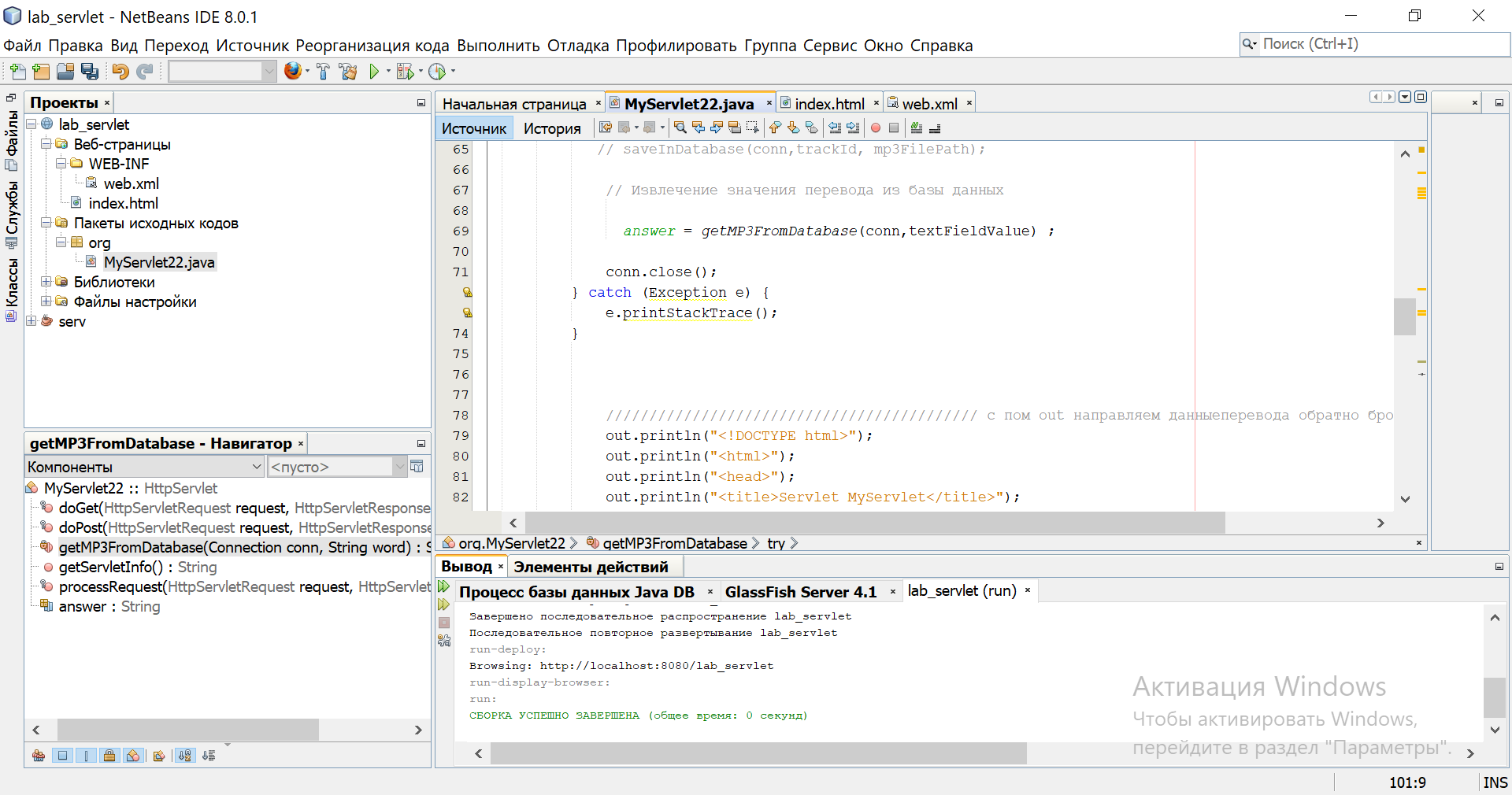
out.println("</body>");

out.println("</html>");

. . . . . . . . .

Создаем потоковый объект PrintWriter out и используем его метод println.

Теперь скорректируем конфигурационный файл web.xml



<?xml version="1.0" encoding="UTF-8"?>

<web-app version="3.0" xmlns="http://java.sun.com/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd">

<servlet>

<servlet-name>MyServlet</servlet-name>

<servlet-class>MyServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>MyServlet</servlet-name>

<url-pattern>/MyServlet</url-pattern>

</servlet-mapping>

<welcome-file-list>

<welcome-file>

dictionary.html

</welcome-file>

</welcome-file-list>

<session-config>

<session-timeout>

30

</session-timeout>

</session-config>

</web-app>

Чтобы этот файл появился в проекте, нужно открыть контекстное меню на имени проекта и выполнить последовательно пункты Очистить и Построить и Развернуть. В документе прописан один или несколько классов сервлетов, а в тегах <welcome-file> указан стартовый сайт приложения (с которого данное приложение запускается).

### Задание (общее)

1. Завершите приложение работоспособной программой, в которой используется база данных, содержащая переводы слов.
2. Добавьте в программу обратный перевод.
3. Дополнительно!

Используйте для клиента и сервера два разных проекта. Для этого нужно вызвать сервлет из клиентского обычного java-приложения, используя url .

//java-client

package apacheclient;

import org.apache.http.HttpResponse;

import org.apache.http.client.methods.HttpGet;

import org.apache.http.impl.client.CloseableHttpClient;

import org.apache.http.impl.client.HttpClients;

import org.apache.http.util.EntityUtils;

import javax.swing.\*;

import java.awt.\*;

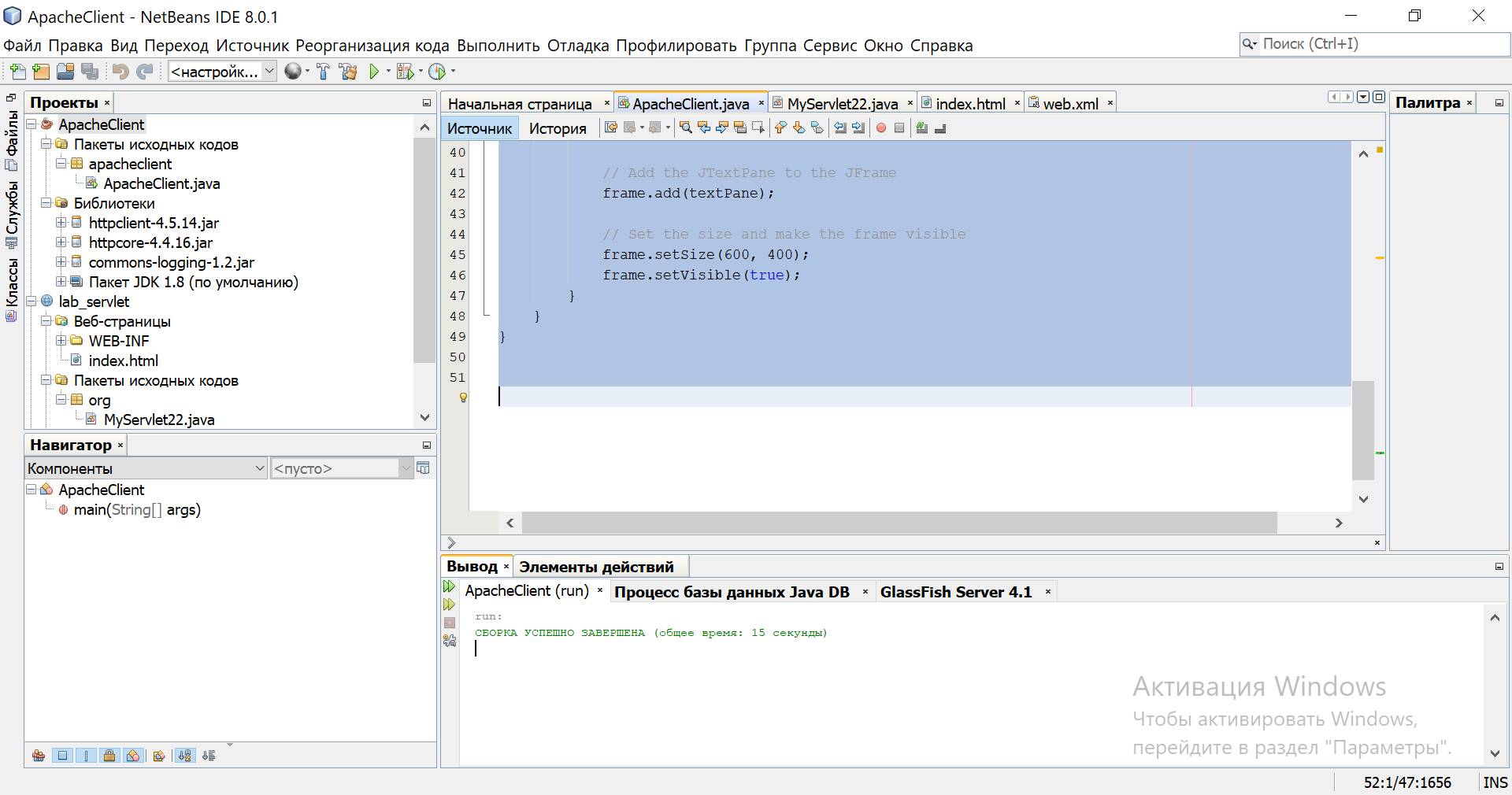
public class ApacheClient {

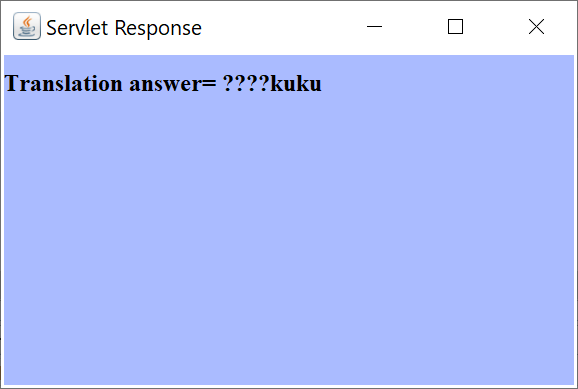
public static void main(String[] args) throws Exception {

String url = "http://localhost:8080/lab\_servlet/MyServlet22"; // Replace with your servlet's URL

….

Download jar-files:





Доделать словарь, используя различные проекты.

***Порядок выполнения работы***

1. Изучить теоретическую часть.
2. Получить индивидуальное задание у преподавателя из прилагаемого списка.
3. Создать и отладить приложение .
4. Написать отчет.
5. Защитить работу.

***Индивидуальные задания***

**Вариант 1.**

Написать сервлет, возвращающий на сторону клиента презентацию Power Point с автоматическим открытием на стороне клиента

**Вариант 2.**

Написать сервлет, возвращающий на сторону клиента файл с рисунком

**Вариант 3.**

Написать сервлет, возвращающий на сторону клиента музыкальный файл с автоматическим проигрыванием на стороне клиента

**Вариант 4.**

Написать сервлет, возвращающий на сторону клиента документ Word с автоматическим открытием на стороне клиента

**Вариант 5.**

Написать сервлет, возвращающий на сторону клиента документ Excel с автоматическим открытием на стороне клиента

***Контрольные вопросы***

1. Что такое сервлет, какие основные объекты и методы он предоставляет?
2. Как сервлет определяется в клиентском сайте?
3. Для чего служит конфигурационный файл web.xml, поясните его структуру.
4. Как вернуть результаты, полученные в сервлете, обратно на сторону клиента?