МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«Харківський Політехнічний Інститут»

Кафедра «стратегічне управління»

ЗВІТ

з лабораторної роботи №2

з дисципліни

«ПРОГРАМУВАННЯ ВЕБ-ЗАСТОСУВАНЬ java ee»

ВИКОНАВ

Студент групи КН-317А

ПЕРЕВІРИла

ас. каф. СУ

Луценко С. Ю.

Харків 2020

ДОСЛІДЖЕННЯ ПРИНЦИПІВ ЗАСТОСУВАННЯ СЕРВЛЕТІВ ДЛЯ СТВОРЕННЯ ВЕБ-СТОРІНОК

Мета

1. Закріпити знання про структуру веб-застосувань.
2. Закріпити навички створення веб-застосувань засобами обраної IDE.
3. Вивчити питання побудови та використання сервлетів та їх створення, а також навички створення сервлетів та їх використання в ході розробки веб-застосувань.
4. Набути навички розробки веб-сторінок із застосуванням сервлетів.
5. Набути навички налаштування параметрів для запуску веб-застосувань та їх розгортання без IDE.

Завдання роботи

1. Виконати ре факторинг веб-застосування AboutAuthor, що складається з двох сторінок:

*index*, на який э надпис про студента, та кнопка About, при натисканні на яку відображається сторінка з інформацією про студента;

*about*, на якій відображається інформація, в якій студент описує себе як кандидата на посаду розробника програмного забезпечення:

* прізвище;
* ім’я;
* по батькові;
* фото;
* вік;
* мови програмування (рівень вивчення або термін вивчення);
* досвід використання СКБД (термін роботи або назва програмного рішення);
* володіння мовами;
* сертифікати, що підтверджуютькваліфікацію;
* контакти.

В новій версії перехід на сторінку *about* організувати двома способами:

* використовуючи тег гіперпосилання;
* шляхом розробки сервлету, що здійснює перенаправлення на готову сторінку.

1. Розробити веб-застосування *MyList*, що складається з двох сторінок:

* *index*, на якій представлені компоненти, що дозволяють ввести логін та пароль, а також кнопка, при натисканні на яку перевіряється коректність вводу та перехід на сторінку mylist;
* *mylist*, на якій відображається таблиця із 3-5 стовпчиків та 10-15 рядків, заповнених інформацією. Таблиця має містити шапку, що відрізняється стилем оформлення.

Для заповнення таблиці має бути створена колекція об’єктів класу. Доцільно створити окремий клас для представлення інформації та клас, що агрегує об’єкти цього класу. Наприклад, клас User и клас UserList. В класі UserList мають бути представлені метод, що забезпечує початкове формування колекції, та метод, що повертає колекцію.

Формування сторінки здійснюється сервлетом (jsp не використовувати).

1. Підготувати war-файли для розгортання на сторонньому веб-сервері (наприклад, на комп’ютері викладача або сервері мережі). Запустити обидва проекти на іншому комп’ютері.

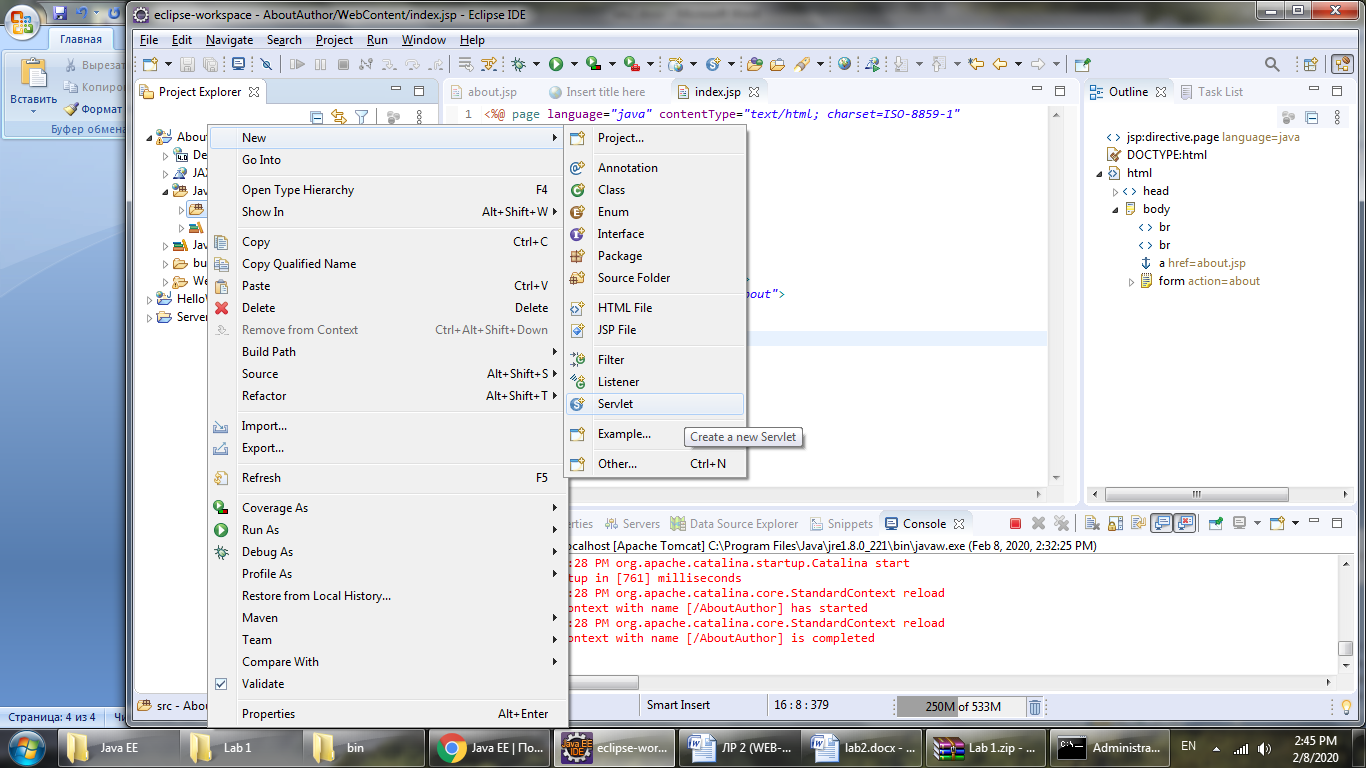
Хід виконання

1. Рефакторинг веб-застосування AboutAuthor, що складається з двох сторінок

Додамо гіперпосилання до вже створеного у поереддній лабораторній роботі index.jsp, а також змінимо атрибут action форми з *about.jsp* на *about* для подальшої обробки сервером.

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*  pageEncoding=*"ISO-8859-1"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"ISO-8859-1"*>  <title>Insert title here</title>  </head>  <body>  Garbuz S. I. <br><br>  <a href = *"about.jsp"*>About</a>  <form action = *"about"* method = *"GET"*>  <input type = *"submit"* value = *"About"*>  </form>  </body>  </html> |

Створимо новий сервлет AboutServlet – рисунок 1.



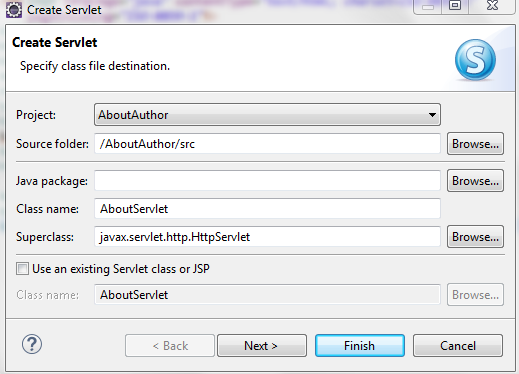


Рисунок 1 – Створення сервлету

Змінемо код методу doGet на наступний

|  |
| --- |
| **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws** ServletException, IOException {  // **TODO** Auto-generated method stub  //response.getWriter().append("Served at: ").append(request.getContextPath());  RequestDispatcher view = request.getRequestDispatcher("/about.jsp");  view.forward(request, response);  } |

Тепер маємо два види посилання на сторінку about – рисунок 2.

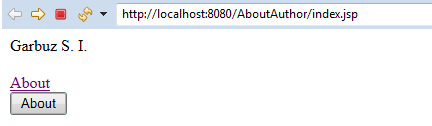


Рисунок 2 – index.jsp

Створимо новий проект MyList з файлом index.jsp.

Вміст файлу index.jsp

|  |
| --- |
| <%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"* pageEncoding=*"UTF-8"*%>  <!DOCTYPE html>  <html>  <head>  <meta charset=*"ISO-8859-1"*>  <title>Insert title here</title>  </head>  <body>  <form action = *"CountriesServlet"* class=*"login100-form validate-form p-l-55 p-r-55 p-t-178"*>  <span class=*"login100-form-title"*>  Sign In  </span>  <div class=*"wrap-input100 validate-input m-b-16"* data-validate=*"Please enter username"*>  <input class=*"input100"* type=*"text"* name=*"username"* placeholder=*"Username"*>  <span class=*"focus-input100"*></span>  </div>  <div class=*"wrap-input100 validate-input"* data-validate=*"Please enter password"*>  <input class=*"input100"* type=*"password"* name=*"pass"* placeholder=*"Password"*>  <span class=*"focus-input100"*></span>  </div>  <div class=*"container-login100-form-btn"*>  <input type = *"submit"* value = *"Sign In"*>  </div>    </form>  </body>  </html> |

Також створимо нові класи Country, CountryList, Language, LanguageList, User, UserList.

Country.java

|  |
| --- |
| import java.util.ArrayList;  import java.util.List;  public class Country {  private int id;  private String name;  private LanguageList languages;    public Country(int id, String name) {  super();  this.id = id;  this.name = name;  }    public Country(int id, String name, LanguageList languages) {  super();  this.id = id;  this.name = name;  this.languages = languages;  }    public LanguageList getLanguages() {  return languages;  }  public void setLanguages(LanguageList languages) {  this.languages = languages;  }  public Country() {    }    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;  }    } |

CountryList.java

|  |
| --- |
| import java.util.ArrayList;  import java.util.List;  public class CountryList {  private List<Country> countryList;    public CountryList () {  this.countryList = new ArrayList<>();  }  public CountryList(ArrayList<Country> countryList) {  this.countryList = countryList;  }    public List<Country> getCountryList() {  return countryList;  }  public void setCountryList(List<Country> countryList) {  this.countryList = countryList;  }    } |

Language.java

|  |
| --- |
| **public** **class** Language {  **private** **int** id;  **private** String name;    **public** Language(**int** id, String name) {  **super**();  **this**.id = id;  **this**.name = name;  }  **public** Language() {    }    **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;  }  } |

LanguageList.java

|  |
| --- |
| import java.util.ArrayList;  import java.util.List;  public class LanguageList {  private List<Language> languageList;    public LanguageList () {  this.languageList = new ArrayList<>();  }  public LanguageList(List<Language> languageList) {  this.languageList = languageList;  }    public List<Language> getLanguageList() {  return languageList;  }  public void setLanguageList(List<Language> languageList) {  this.languageList = languageList;  }  } |

User.java

|  |
| --- |
| **public** **class** User {  **private** **int** id;  **private** String login;  **private** String password;    **public** User(**int** id, String login, String password) {  **super**();  **this**.id = id;  **this**.login = login;  **this**.password = password;  }  **public** User() {    }  **public** **int** getId() {  **return** id;  }  **public** **void** setId(**int** id) {  **this**.id = id;  }  **public** String getLogin() {  **return** login;  }  **public** **void** setLogin(String login) {  **this**.login = login;  }  **public** String getPassword() {  **return** password;  }  **public** **void** setPassword(String password) {  **this**.password = password;  }  } |

UserList.java

|  |
| --- |
| import java.util.ArrayList;  import java.util.List;  public class UserList {  private List<User> userList;    public UserList () {  this.userList = new ArrayList<>();  }  public UserList(ArrayList<User> userList) {  this.userList = userList;  }    public List<User> getUserList() {  return userList;  }  public void setUserList(List<User> userList) {  this.userList = userList;  }  } |

Створимо клас Dummy.java для ініціалізації декількох об’єктів та колекцій, що будуть імітувати інформацію з бази даних.

Dummy,java

|  |
| --- |
| **import** java.util.Arrays;  **public** **class** Dummy {  **public** **static** UserList *users* = **new** UserList();  **public** **static** CountryList *countries* = **new** CountryList();    **public** **static** Language *lang1* = **new** Language(0, "Украинский");  **public** **static** Language *lang2* = **new** Language(1, "Русский");  **public** **static** Language *lang3* = **new** Language(2, "Английский");  **public** **static** Language *lang4* = **new** Language(3, "Французский");  **public** **static** Language *lang5* = **new** Language(4, "Испанский");  **public** **static** Language *lang6* = **new** Language(5, "Арабский");  **public** **static** Language *lang7* = **new** Language(6, "Беларусский");  **public** **static** Language *lang8* = **new** Language(7, "Португальский");    **static** {  *users*.setUserList(Arrays.*asList*(  **new** User(0, "qwerty", "123456"),  **new** User(0, "user", "user"),  **new** User(0, "admin", "admin")  ));  *countries*.setCountryList(Arrays.*asList*(  **new** Country(0, "Украина", **new** LanguageList(Arrays.*asList*(*lang1*))),  **new** Country(1, "Россия", **new** LanguageList(Arrays.*asList*(*lang2*))),  **new** Country(3, "США", **new** LanguageList(Arrays.*asList*(*lang3*))),  **new** Country(4, "Аргентина", **new** LanguageList(Arrays.*asList*(*lang5*))),  **new** Country(5, "Барбадос", **new** LanguageList(Arrays.*asList*(*lang3*))),  **new** Country(6, "Беларусь", **new** LanguageList(Arrays.*asList*(*lang7*, *lang2*))),  **new** Country(7, "Канада", **new** LanguageList(Arrays.*asList*(*lang3*, *lang4*))),  **new** Country(8, "ЦАР", **new** LanguageList(Arrays.*asList*(*lang4*))),  **new** Country(9, "Чили", **new** LanguageList(Arrays.*asList*(*lang5*))),  **new** Country(10, "Конго", **new** LanguageList(Arrays.*asList*(*lang4*))),  **new** Country(11, "Коста Рика", **new** LanguageList(Arrays.*asList*(*lang5*))),  **new** Country(12, "Джибути", **new** LanguageList(Arrays.*asList*(*lang4*, *lang6*))),  **new** Country(13, "Доминиканская Республика", **new** LanguageList(Arrays.*asList*(*lang5*))),  **new** Country(14, "Экваториальная Гвинея", **new** LanguageList(Arrays.*asList*(*lang4*, *lang5*, *lang8*)))  ));  }  } |

Створимо сервлет CountriesServlet для формування логіки сайту

|  |
| --- |
| **import** java.io.IOException;  **import** java.io.PrintWriter;  **import** javax.servlet.ServletException;  **import** javax.servlet.ServletOutputStream;  **import** javax.servlet.annotation.WebServlet;  **import** javax.servlet.http.HttpServlet;  **import** javax.servlet.http.HttpServletRequest;  **import** javax.servlet.http.HttpServletResponse;  /\*\*  \* Servlet implementation class CountriesServlet  \*/  @WebServlet("/CountriesServlet")  **public** **class** CountriesServlet **extends** HttpServlet {  **private** **static** **final** **long** ***serialVersionUID*** = 1L;    /\*\*  \* **@see** HttpServlet#HttpServlet()  \*/  **public** CountriesServlet() {  **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 login = request.getParameter("username");  String password = request.getParameter("pass");  System.***out***.println(login + ' ' + password);  **boolean** isAuthenticated = **false**;  **for** (**int** i = 0; i < Dummy.*users*.getUserList().size(); i++) {  User user = Dummy.*users*.getUserList().get(i);  **if** (login.equals(user.getLogin()) && password.equals(user.getPassword())) {  isAuthenticated = **true**;  **break**;  }  }  **if** (!isAuthenticated)  response.sendRedirect(request.getContextPath() + "/index.jsp");    response.setContentType("text/html; charset=UTF-8");  response.setCharacterEncoding("UTF-8");  PrintWriter out = response.getWriter();    out.println("<html>");  out.println("<head><title>Hello Servlet</title></head>");    out.println("<body>");    out.println("<table style = 'border: 1px solid black'>");  out.println("<tr style = 'border: 1px solid black'>");  out.println("<th style = 'border: 2px solid black'>ID</th>");  out.println("<th style = 'border: 2px solid black'>Страна</th>");  out.println("<th style = 'border: 2px solid black'>Язык #1</th>");  out.println("<th style = 'border: 2px solid black'>Язык #2</th>");  out.println("<th style = 'border: 2px solid black'>Язык #3</th>");  out.println("</tr>");  **for** (Country country : Dummy.*countries*.getCountryList()) {  out.println("<tr style = 'border: 1px solid black'>");  out.println("<td style = 'border: 1px solid black'>" + country.getId() + "</td>");  out.println("<td style = 'border: 1px solid black'>" + country.getName() + "</td>");  **for** (Language language : country.getLanguages().getLanguageList()) {  out.println("<td style = 'border: 1px solid black'>" + language.getName() + "</td>");  }  **for** (**int** i = 0; i < 3-country.getLanguages().getLanguageList().size(); i++) {  out.println("<td style = 'border: 1px solid black'> </td>");  }  out.println("</tr>");  }  out.println("</table>");    out.println("</body>");  out.println("<html>");  }  /\*\*  \* **@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);  }  } |

Подивитися інформацію стосовно країн та їхніх офіційних мов можна ввівши коректні значення у поля – рисунок 3.

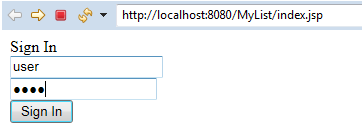


Рисунок 3 – index.jsp

Сторінка з таблицею – рисунок 4

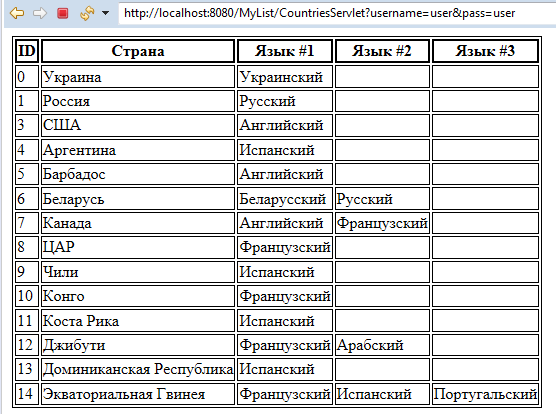


Рисунок 4 – index.jsp з url параметрами

Збережемо проект у вигляді .war файлу до директорії з TomCat – рисунок 5.

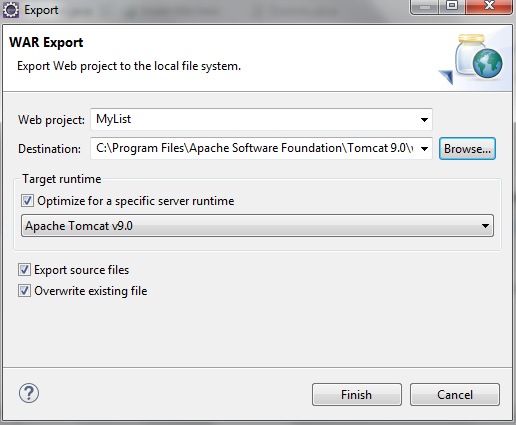


Рисунок 5

Надалі, якщо запустити програму Monitor Tomcat та потім запустити сервер – ми зможемо заходити на сторінку за межами Eclipse – рисунок 6.

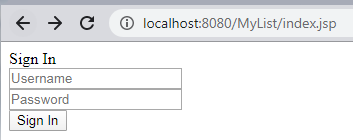


Рисунок 6 – створена jsp у браузері Google Chrome

Висновки

При виконанні цієї лабораторної роботи був створений новий web проект з простою логікою для відображення згенерованої HTML розмітки з даними. Створення логіки веб серверу є одною з складових частин розробки і саме в цьому етапі проявляється уся доцільність використовувати ту чи іншу мову програмування. Тут було застосовано Java – схожу на C#, але складнішу аніж Node JS або Python Django framework. Було здійсненно запуск проекту на іншому комп’ютері. Застосування було екпортровано у war файл задля зручного використання на інших комп’ютерах.