Министерство образования Республики Беларусь

Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

ОТЧЕТ

По лабораторной работе №1

«**Работа с Web-ресурсами**»

Вариант 5

Выполнил: ст. гр. 220602

И. Г. Жарский

Проверила: Ю.О. Герман

Минск 2025

## **1 ПОСТАНОВКА ЗАДАЧИ**

Познакомиться с возможностями поиска информации в удаленных *URL*-ресурсах.

**2 ХОД РАБОТЫ**

Подготовим собственный набор *html*-документов, содержащий качества разных пород собак. Пример одного из таких документов представлен на рисунке 1.

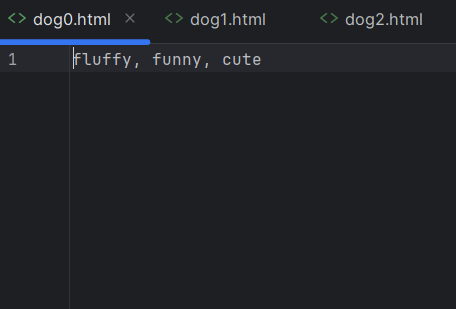


Рисунок 1 – Файл, содержащий качества собаки 0

Затем дорабатываем предоставленный код, добавляя вывод файла с наибольшим числом совпадений в браузер. Итоговый код:

package web\_res\_lab;  
  
import java.awt.\*;  
import java.awt.event.\*;  
import java.io.\*;  
import java.util.\*;  
import java.io.BufferedInputStream;  
import java.io.File;  
import java.io.IOException;  
import java.net.URL;  
import java.net.URLConnection;  
import java.awt.Desktop;  
import java.io.File;  
import java.io.IOException;  
import java.net.URI;  
import java.net.URISyntaxException;  
  
public class Web\_res\_lab extends Frame implements ActionListener{  
  
 Button exitButton = new Button("Exit");  
 Button searchButton = new Button("Search");  
 TextArea textArea = new TextArea();  
  
 public Web\_res\_lab() {  
 super("My Window");  
 setLayout(null);  
 setBackground(new Color(150, 200, 100));  
 setSize(450, 250);  
  
 exitButton.setBounds(110, 190, 100, 20);  
 exitButton.addActionListener(this);  
 add(exitButton);  
  
 searchButton.setBounds(110, 165, 100, 20);  
 searchButton.addActionListener(this);  
 add(searchButton);  
  
 textArea.setBounds(20, 50, 300, 100);  
 add(textArea);  
  
 setLocationRelativeTo(null);  
 setVisible(true);  
  
 addWindowListener(new WindowAdapter() {  
 @Override  
 public void windowClosing(WindowEvent we) {  
 System.*exit*(0);  
 }  
 });  
 }  
  
 @Override  
 public void actionPerformed(ActionEvent ae) {  
  
 if (ae.getSource() == exitButton) {  
 System.*exit*(0);  
  
 } else if (ae.getSource() == searchButton) {  
 int i=0;  
 String str = "";  
 String[] keywords = textArea.getText().split(",");  
 ArrayList<File> files = new ArrayList<>(Arrays.*asList*(new File("C:/java\_projects/lab1german/src/htmls").listFiles()));  
 textArea.setText("");  
 for (File file : files) {  
 int coincidenceCount = testUrl(file, keywords);  
 if (i < coincidenceCount) {  
 i = coincidenceCount;  
 str = file.getPath();  
 }  
 textArea.append("\n" + file + " :" + coincidenceCount);  
 }  
 openInBrowser(str);  
  
 }  
 }  
  
 private int testUrl(File file, String[] keywords) {  
 int results = 0;  
 try {  
 URL url = file.toURI().toURL();  
 URLConnection connection = url.openConnection();  
 StringBuilder htmlContent = new StringBuilder();  
  
 try (BufferedInputStream bis = new BufferedInputStream(connection.getInputStream());  
 BufferedReader reader = new BufferedReader(new InputStreamReader(bis))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 htmlContent.append(line).append("\n");  
 }  
 }  
 String content = htmlContent.toString().toLowerCase(); // file content in string  
  
 for (String keyword : keywords) {  
 if (content.contains(keyword.trim().toLowerCase())) {  
 results++;  
 }  
 }  
 } catch (IOException e) {  
 System.*out*.println("Error: " + e.getMessage());  
 return -1;  
 }  
 return results;  
 }  
  
 public static void main(String[] args) {  
 new Web\_res\_lab();  
 }  
  
  
  
 public void openInBrowser(String filePath) {  
 // Specify the path to your HTML file  
  
 File file = new File(filePath);  
  
 if (Desktop.*isDesktopSupported*()) { // Check if Desktop API is supported  
 Desktop desktop = Desktop.*getDesktop*();  
 if (desktop.isSupported(Desktop.Action.*BROWSE*)) { // Check if browsing action is supported  
 try {  
 // Convert the file path to a URI  
 URI uri = file.toURI();  
 desktop.browse(uri); // Open the URI in the default browser  
 System.*out*.println("File opened in browser successfully.");  
 } catch (IOException e) {  
 System.*err*.println("Error opening file in browser: " + e.getMessage());  
 }  
 } else {  
 System.*out*.println("Browsing action not supported on this platform.");  
 }  
 } else {  
 System.*out*.println("Desktop API not supported on this platform.");  
 }  
 }  
}

Пример работы приложения показан на рисунках 2-3.

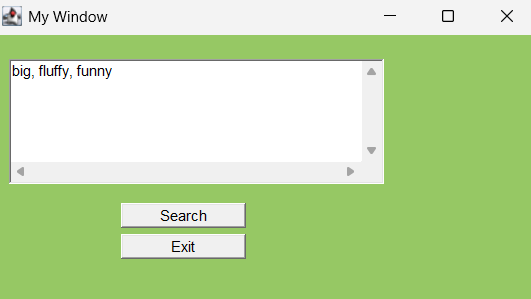


Рисунок 2 – Ввод качеств

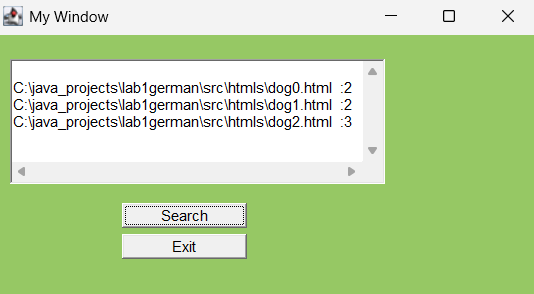


Рисунок 3 – Результат поиска

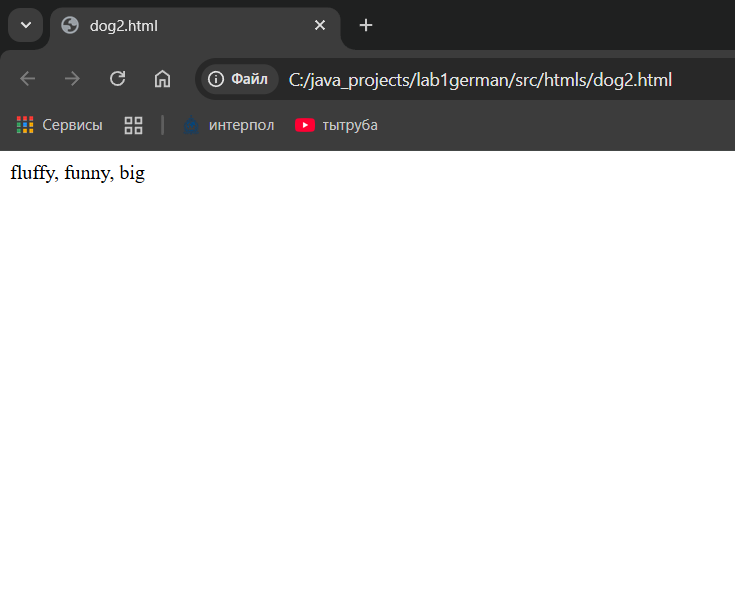


Рисунок 4 – Вывод файла в браузер

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения данной лабораторной работы были успешно достигнуты поставленные цели. Было изучено взаимодействие *Java*-приложений с удаленными *URL*-ресурсами, а также принципы поиска информации в локальной файловой системе, имитирующей веб-сервер.

Было разработано приложение, которое позволяет пользователю вводить ключевые слова, выполняет их поиск в содержимом *HTML*-документов, расположенных в указанной директории, и определяет наиболее релевантный документ по количеству совпадений. Ключевым улучшением исходной заготовки программы стала реализация функционала для автоматического открытия найденного документа с максимальным числом совпадений в браузере по умолчанию операционной системы.

В рамках индивидуального задания по варианту №5 программа была доработана для решения конкретной прикладной задачи: поиска породы собаки по набору из 1-3 характеристик. Для этого был создан специализированный набор *HTML*-документов, где каждая порода собак описывается своими ключевыми качествами. Алгоритм поиска был адаптирован для работы с этим структурированным набором данных.