МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ

УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)» (СГАУ)

Отчёт по лабораторной работе №2  
по курсу «Сети ЭВМ и телекоммуникации»

Выполнил:  
Никитин А.А

гр.6303

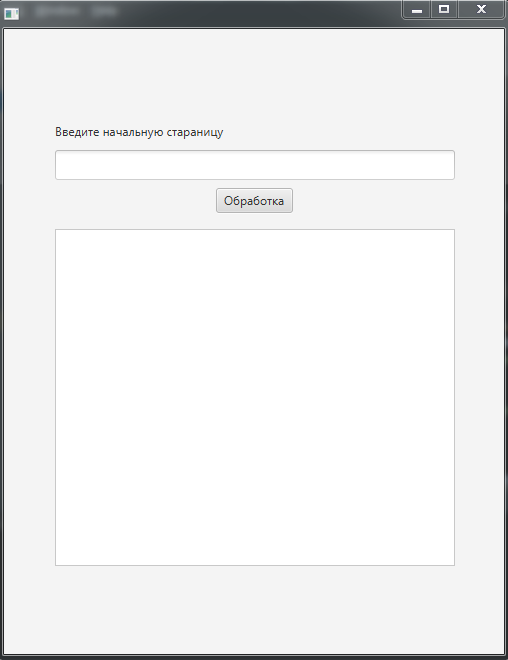
Проверил:  
Пигусов А.С.

Самара 2016

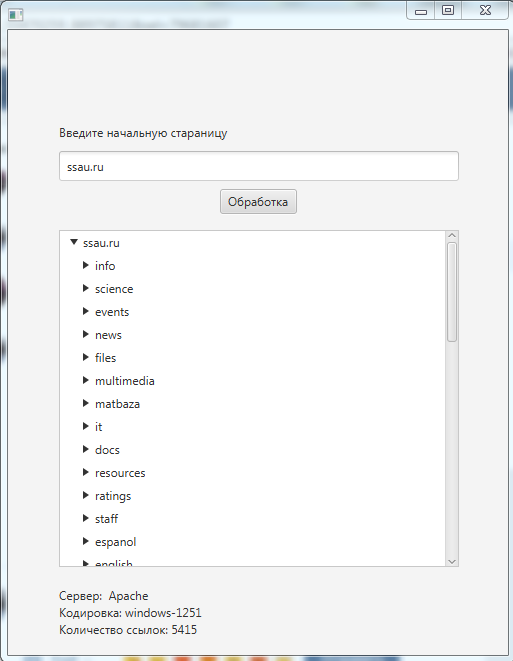
**Задание**

Вариант 1: Составить и вывести дерево каталогов www-сервера. Вывести информацию о сервере (кодировку страниц, версию программного обеспечения)

**Интерфейс**



Графический интерфейс представляет собой поле ввода доменного имени, кнопку начала обработки и поле вывода, представляющее собой дерево.



Выше представлен результат работы программы. В поле вывода было построено дерево каталогов в том виде, в котором оно хранится на сервере.

В процессе обработки не учитываются ссылки типа «mailto:», «javascript:», «//\*», «http://», «https://» и т.п.

В нижней части программы находится информация о сервере и количестве обработанных ссылок. Данные о сервере берутся из заголовка ответа сервера при запросе по адресу «hostname/».

Обработка ссылок происходит с глубиной, равной 3.

**Код программы**

**Отправка запроса на сервер:**

private static void sendHeader(OutputStream os, String host, String path, String method) throws IOException {  
 os.write((method + " " + path + " HTTP/1.1\r\n").getBytes());  
 os.write(("Host: " + host + "\r\n\r\n").getBytes());  
 System.*out*.println("Header have been sent. Path:" + path);  
}

**Получение данных с сервера:**

private static String getResponse(InputStream is) throws IOException, InterruptedException, TimeoutException, ExecutionException {  
 InputStreamReader isr = new InputStreamReader(is);  
 BufferedReader bfr = new BufferedReader(isr);  
 StringBuffer sbf = new StringBuffer();  
 try {  
 int readByte = bfr.read();  
 sbf.append((char) readByte);  
 while (readByte > 0) {  
 readByte = bfr.read();  
 sbf.append((char) readByte);  
 }  
  
 } catch (SocketTimeoutException t) {  
 System.*out*.println("Loading interrupted. Timeout.\n");  
 }  
 return sbf.toString();  
}

**Обработка html кода:**

public static Set<String> parseHtml(String html, String parentPath){  
 Set<String> links = new HashSet<>();  
 Document doc = Jsoup.*parse*(html);  
 Elements elements = doc.getElementsByTag("a");  
 elements.forEach(element -> {  
 String link = element.attr("href");  
 *parseLink*(link, links, parentPath);  
 });  
 return links;  
}

**Алгоритм запросов:**

private Collection<String> getLinks(String host, int depth) throws Exception {  
 Collection<String> resultLinks;  
 if (depth > 0) {  
 String hostBody = HttpService.*sendRequest*(host, "GET");  
 resultLinks = ParseService.*parseHtml*(hostBody);  
 depth--;  
  
 Collection<String> currentLinks = resultLinks;  
 for (int i = 0; i < depth; i++) {  
 Collection<String> tempCollection = new HashSet<>();  
 for (String tempLink : currentLinks) {  
 String body = HttpService.*sendRequest*(host, tempLink, "GET");  
 tempCollection.addAll(ParseService.*parseHtml*(body, tempLink));  
 }  
  
 tempCollection.removeAll(resultLinks);  
 resultLinks.addAll(tempCollection);  
 currentLinks = tempCollection;  
 }  
 return resultLinks;  
 }  
 return Collections.*emptyList*();  
}

**Конвертация ссылок в дерево:**

private TreeItem<String> getTreeRootItem(Collection<String> links, String rootName) {  
 TreeItem<String> root = new TreeItem<>(rootName);  
 Iterator<String> linksIterator = links.iterator();  
 while (linksIterator.hasNext()) {  
 String link = linksIterator.next();  
 String[] elements = link.split("/");  
 TreeItem<String> currentRoot = root;  
 for (String element : elements) {  
 if (!element.isEmpty()) {  
 TreeItem<String> container = contains(currentRoot, element);  
 if (container != null) {  
 currentRoot = container;  
 } else {  
 TreeItem<String> newChild = new TreeItem<>(element);  
 currentRoot.getChildren().add(newChild);  
 currentRoot = newChild;  
 }  
 }  
 }  
 linksIterator.remove();  
 }  
 return root;  
}

**Выводы**

В результате работы посредством сокетов были напрямую получены и обработаны ответы от www-сервера. Производилась обработка и анализ данных, полученных от сервера. Кроме того, анализировалась информация, передающаяся от сервера в качестве заголовка. Работа была написана на языке Java, графический интерфейс основан на платформе javaFX, обработка html данных производилась средствами библиотеки jsoup.