Московский Государственный Технический Университет

имени Н.Э. Баумана

Факультет ИУ «Информатика и системы управления»

Кафедра ИУ-3 «Информационные системы и телекоммуникации»

Отчет к лабораторной работе №7
по курсу «Технологии программирования»
“Поисковый робот”

Продолжительность работы: 4 ак. часа

Сдал Ситниченко С.А.

Принял Руденкова Ю.С.

Москва, 2022

Лабораторная работа №7

Тема: Поисковый робот

Цель работы: В этой лабораторной работе, требуется написать простейший поисковый робот. Робот должен автоматически загружать веб страницы из сети Интернет, искать в них новые ссылки, и повторять эту операцию для каждой найденной ссылки. Он будет просто просматривать новые URL (указывающее на расположение других веб страниц) на каждой странице, сохранять эти ссылки и печатать их в конце работы программы.

Для хранения пары URL и глубины поиска мы используем класс URLDepthPair.

```
public static class URLDepthPair {
    String URL;
    int searchDepth;

    URLDepthPair(String URL, int searchDepth) {
        this.URL = URL;
        this.searchDepth = searchDepth;
    }

    @Override
    public String toString() {
        return (URL + " " + searchDepth);
    }
}
```

Метод сокета, который управляет соединением с сервером HTTP.

```
public static void socketProcess(String URL) throws IOException {
    URL url = new URL(URL);
    HttpURLConnection connection = (HttpURLConnection)url.openConnection();
    connection.setConnectTimeout(10000);
    connection.setReadTimeout(10000);
    connection.setRequestMethod("GET");

    bufferedReader = new BufferedReader(new InputStreamReader(connection.getInputStream()));

    inputStreamHandler(bufferedReader);

    bufferedReader.close();
    connection.disconnect();
}
```

Для приведения заголовка функции обработки полученного через “GET” контента.

```
public static void inputStreamHandler(BufferedReader bufferedReader) throws IOException {
    String line = bufferedReader.readLine();

    while (line != null) {
        StringBuffer bufferOfURL;
        bufferOfURL = hrefHandler(line);

        if (bufferOfURL.isEmpty()) {
            line = bufferedReader.readLine();
            continue;
        }

        URLSelection(bufferOfURL);
        line = bufferedReader.readLine();
    }
}
```

Метод hrefHandler вернет StringBuffer со всеми ссылками в строке в виде строк с разделителем.

```
public static StringBuffer hrefHandler(String line) {
    Matcher hrefPatternMatcher = hrefPattern.matcher(line);

    StringBuffer buffer = new StringBuffer();
    while (hrefPatternMatcher.find()) {
        buffer.append(hrefPatternMatcher.group()).append("-");
    }

    return buffer;
}
```

Функция URLSelection выполняет проверку ссылки в формате гипертекста и переводит её в обычный URL.

```
public static void URLSelection(StringBuffer URLBuffer) {
    URLBuffer.insert(0, "-");

    for (int i = 0; i < URLBuffer.length(); ++i) {
        if (URLBuffer.charAt(i) == '-' && i + 1 < URLBuffer.length()) {
            while (URLBuffer.charAt(i) != '\"')
                ++i;

            int intIndex = i + 1;
            ++i;

            while (URLBuffer.charAt(i) != '\"')
                ++i;

            if (prefixChecker(URLBuffer.substring(intIndex, i ))) {
                String URL = URLBuffer.substring(intIndex, i );

                if (rawURLList.size() == 0)
                    URLGenerator(URL, URLList.getLast().searchDepth + 1);
                else
                    URLGenerator(URL, rawURLList.getLast().searchDepth + 1);
            }
        }
    }
}
```

Функция записи ссылок URLGenerator.

```
public static void URLGenerator(String URL, int currentDepth) {
    if (prefixChecker(URL)) {
        rawURLList.add(new URLDepthPair(URL, currentDepth));
    }
}
```

Метод для переноса всех URL из сохраненных ранее rawURLList в список готовых URLList.

```
public static boolean replaceURL() {
    if (URLList.getLast().searchDepth > MAX_DEPTH)
        return true;

    else if (rawURLList.isEmpty())
        return false;

    URLDepthPair currentPair = rawURLList.getFirst();
    while (currentPair != null && currentPair.searchDepth <= MAX_DEPTH) {
        URLList.add(currentPair);
        rawURLList.removeFirst();
        if (rawURLList.isEmpty())
            break;

        currentPair = rawURLList.getFirst();
    }

    assert currentPair != null;
    return currentPair.searchDepth > MAX_DEPTH;
}
```

Output

Вывод ссылок при входной статье google.com и глубине 3.

```
C:\Users\днс\OneDrive\Документы\NetBeansProjects\Crawler\src\crawler>java Crawler https://google.com 3
All sites:
https://google.com 0
https://www.google.com/setprefdomain?prefdom=RU&amp;prev=https://www.google.ru/&amp;sig=K_L486608DO3q6ccVVBke9vLgiGtE%3D 1
https://www.google.com/setprefdomain?prefdom=RU&amp;prev=https://www.google.ru/&amp;sig=K_ag_26GMIj7TZzRlS083wsdjVTXU%3D 2
https://www.google.com/setprefdomain?prefdom=RU&amp;prev=https://www.google.ru/&amp;sig=K_0J44H0aRAnvdkPxljw5wNSzYLlU%3D 3
```

Вывод:

В результате данной лабораторной работы мы создали поискового робота, который ищет ссылки на интернет-страницы и выводит их. Также научились использовать подключение к серверу с помощью HTTP.