Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра «Информатики»

Отчет по лабораторной работе № 8

по дисциплине

«Кроссплатформенные технологии программирования»

на тему:

«Модифицированный веб-сканер»

Выполнил: студент группы БСТ2001

Багдасарян А. С.

Москва 2022

Оглавление

[**1.Цель работы:** 3](#_Toc102565611)

[**2.Задание:** 3](#_Toc102565612)

[**3.Ход работы** 3](#_Toc102565613)

[3.1Код программы 3](#_Toc102565614)

[3.2Результат работы программы 6](#_Toc102565615)

[3.3Программа на Git hub 6](#_Toc102565616)

[**4.Вывод** 7](#_Toc102565617)

# **Цель работы:**

Расширить сканер для использования поточной обработки Java так, чтобы несколько веб-страниц можно было сканировать параллельно. В свою очередь это должно привести к значительному повышению производительности.

# **2. Задание:**

Для данной лабораторной работы можно выделить следующие задания:

1. Реализовать класс с именем URLPool;
2. Создание класса CrawlerTask;
3. Синхронизация объектов пула URL-адресов;
4. Ожидание веб-сканера.

# **3. Ход работы**

3.1 Код программы

Ниже представлен код Crawler.java

package com.company;  
  
import java.net.HttpURLConnection;  
import java.net.URL;  
import java.util.HashMap;  
import java.util.LinkedList;  
import java.util.Scanner;  
import java.util.regex.Pattern;  
  
class Crawler {  
 private String URL;  
 private static int *maxDepth*;  
 public static int *CountThreads*;  
  
 public static int *WaitingThreads* = 0;  
 public static int *CountURLs* = 0;  
  
 public static int getMaxDepth() { return *maxDepth*; }  
  
 public Crawler(String URL, int maxDepth, int countThreads){  
 this.URL = URL;  
 Crawler.*maxDepth* = maxDepth;  
 Crawler.*CountThreads* = countThreads;  
 }  
  
 public void run() {  
 CrawlerTask task = new CrawlerTask(new URLdepthPair(URL,0));  
 task.start();  
 }  
  
 private static void printResult(){  
 System.*out*.println();  
 System.*out*.println("Всего ссылок: " + *CountURLs*);  
 }  
  
 public static void showHelp() {  
 System.*out*.println("usage: java Crawler <URL> <depth>");  
 System.*exit*(1);  
 }  
  
 public static void main(String[] args){  
 Scanner scan=new Scanner(System.*in*);  
 String[] argg = new String[2];  
 System.*out*.println("Enter URL: ");  
 argg[0]=scan.nextLine();  
 System.*out*.println("Enter depth: ");  
 argg[1]=scan.nextLine();  
 if (argg.length !=2) *showHelp*();  
 int maxDepth = 0;  
 String word=argg[0];  
 try {  
 maxDepth = Integer.*parseInt*(argg[1]);  
 } catch (Exception e) {  
 *showHelp*();  
 }  
 Crawler crawler = new Crawler(word, maxDepth,10);  
 crawler.run();  
  
 Runtime.*getRuntime*().addShutdownHook(new Thread(()->*printResult*()));  
 }  
}

Ниже представлен код URLdepthPair.java

package com.company;  
  
public class URLdepthPair {  
 private String URL;  
 private int depth;  
  
 public int getDepth() { return depth; }  
 public String getURL() { return URL; }  
  
 public URLdepthPair(String URL, int dep) {  
 this.URL = URL;  
 this.depth = dep;  
 }  
  
  
 @Override  
 public String toString() {  
 return "depth: " + depth + " URL: ["+ URL + "]";  
 }  
}

Ниже представлен код URLPool.java

package com.company;  
import java.util.\*;  
public class URLPool {  
 private HashMap<String, URLdepthPair> visited;  
 private LinkedList<URLdepthPair> pool;  
  
 public URLPool(){  
 visited = new HashMap<>();  
 pool = new LinkedList<>();  
 }  
  
 public synchronized URLdepthPair getLink(){  
 boolean isWaiting = false;  
 if(pool.size() == 0) {  
 try {  
 Crawler.*WaitingThreads*++;  
 isWaiting = true;  
 if(Crawler.*WaitingThreads* == Thread.*activeCount*()) {  
 System.*err*.println("Все потоки заняты");  
 System.*exit*(0);  
 };  
 this.wait();  
 }  
 catch (Exception e) { return null; }  
 }  
 if(isWaiting) Crawler.*WaitingThreads*--;  
 URLdepthPair link = pool.pop();  
 visited.put(link.getURL(),link);  
 return link;  
 }  
  
 public synchronized void addLink(URLdepthPair link){  
 if(!visited.containsKey(link.getURL())) {  
 pool.add(link);  
 this.notify();  
 }  
 }  
}

Ниже представлен код CrawlerTask.java

package com.company;  
  
import java.net.\*;  
import java.util.Scanner;  
  
public class CrawlerTask extends Thread {  
 private URLPool pool;  
  
 public CrawlerTask(URLdepthPair link) {  
 pool = new URLPool();  
 pool.addLink(link);  
 }  
  
 @Override  
 public void run() {  
 URLdepthPair link = pool.getLink();  
 System.*out*.println(link.toString());  
 System.*out*.println(Thread.*activeCount*());  
 Crawler.*CountURLs*++;  
 if(link.getDepth() == Crawler.*getMaxDepth*()) return;  
  
 findLinks(link);  
 }  
  
 private void findLinks(URLdepthPair link)  
 {  
 try {  
 URL url = new URL(link.getURL());  
  
 HttpURLConnection connection = (HttpURLConnection) url.openConnection();  
 connection.setRequestMethod("GET");  
  
 Scanner scanner = new Scanner(connection.getInputStream());  
  
 while (scanner.findWithinHorizon("<a\\s+(?:[^>]\*?\\s+)?href=([\"'])(.\*?)\\1", 0) != null) {  
 String newURL = scanner.match().group(2);  
 URLdepthPair newLink = createNewLink(newURL, link);  
 if (newLink == null) continue;  
 CreateNewThread(newLink);  
 }  
 }  
 catch (Exception e) {  
 System.*err*.println("Error: " + e.getMessage());  
 }  
 }  
  
 private URLdepthPair createNewLink(String newURL, URLdepthPair link){  
 if (newURL.startsWith("/")) {  
 newURL = link.getURL() + newURL;  
 }  
 else if (!newURL.startsWith("https")) return null;  
  
 return new URLdepthPair(newURL, link.getDepth() + 1);  
 }  
  
 private void CreateNewThread(URLdepthPair link) {  
 CrawlerTask task = new CrawlerTask(link);  
 task.start();  
 }  
}

* 1. **Результат работы программы**

Результат работы программы показан на рисунке 1

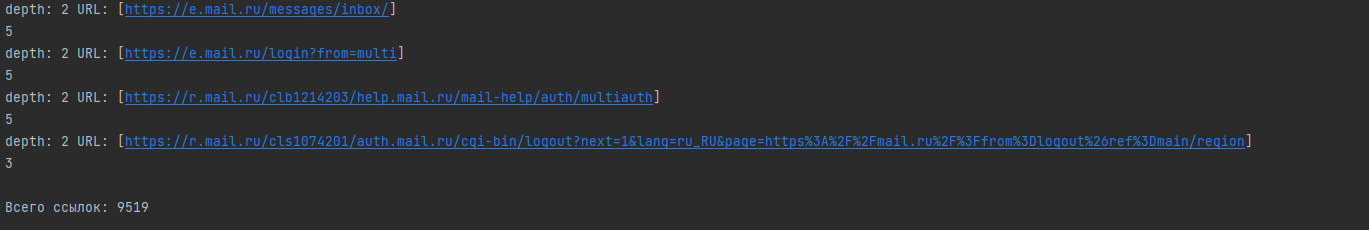


Рисунок 1 – Результат работы программы

* 1. **Программа на Git hub**

Ссылка на репозиторий на Git hub: https://github.com/Artem2406/KTP

# **4. Вывод**

В итоге следует отметить, что в процессе выполнения данной лабораторной работы все вышеперечисленные задания были выполнены и таким образом был модернизирован элементарный веб-сканер, сделанный в прошлой лабораторной работе.

Список использованных источников

1. ГОСТ 7.32-2017 СИБИД. Отчет о научно-исследовательской работе. Структура и правила оформления (с Поправками). [Правовой информационный ресурс]. – 2017. – URL: <http://www.consultant.ru/document/cons_doc_LAW_292293/> (дата обращения 07.02.2022)