TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TPHCM

**KHOA CÔNG NGHỆ THÔNG TIN**

**Extract Information on the website by Regular Expression**



<https://github.com/DoSonAn/DoAnTopic4>

**ĐỒ ÁN MÔN HỌC**

**PHÁT TRIỂN ỨNG DỤNG**

SVTH: Nguyễn Bình AN 15063451

Nguyễn Anh Đô 15066241

Lê Tây Sơn 15082961

GVHD: Huỳnh Thái Học

Võ Công Minh

TPHCM- Năm 2017

Mục lục

[1. Giới thiệu đề tài 3](#_Toc494756629)

[1.1. Đề tài: Extract Information on the website by Regular Expression và nhiệm vụ 3](#_Toc494756630)

[1.1.1. Giới thiệu đề tài 3](#_Toc494756631)

[1.1.2. Nhiệm vụ 3](#_Toc494756632)

[1.2. Tiến độ thực hiện 4](#_Toc494756633)

[2. Phân tích lý thuyết 4](#_Toc494756634)

[2.1. Regular Expression 4](#_Toc494756637)

[2.1.1. Regular Expression là gì? 4](#_Toc494756641)

[2.1.2. Tác dụng của Regular Expression 5](#_Toc494756642)

[2.1.3. Cú pháp cho Regular trong java 5](#_Toc494756643)

[2.2. URL 8](#_Toc494756644)

[2.2.1. URL là gì? 8](#_Toc494756649)

[2.2.2. Tạo một URL 8](#_Toc494756650)

[2.2.3. Đọc trực tiếp một URL 9](#_Toc494756651)

[2.2.4. Kết nối đến một URL 9](#_Toc494756652)

[3. Demo 11](#_Toc494756653)

[3.1. Mô hình của phần mềm 11](#_Toc494756657)

[3.2. Các chức năng của phần mềm 11](#_Toc494756658)

[3.2.1. Lấy email 11](#_Toc494756664)

[3.2.2. Lấy số điện thoại 11](#_Toc494756665)

[3.2.3. Lấy comment 11](#_Toc494756666)

[3.2.5. Reset 12](#_Toc494756671)

[3.3. Kết quả 12](#_Toc494756672)

[TÀI LIỆU THAM KHẢO: 15](#_Toc494756673)

# Giới thiệu đề tài

* 1. **Đề tài: Extract Information on the website by Regular Expression và nhiệm vụ**
     1. Giới thiệu đề tài

-Ngày nay, với nhu cầu gửi mail, gửi tin nhắn hàng loạt để PR cho sản phẩm hay chuyên mục quảng cáo nào đó thì nhu cầu lấy lọc địa chỉ mail hay số điện thoại số lượng lớn trên một website là rất cần thiết dẫn dến sự ra đời của nhiều công cụ hữu ích để thực hiện lấy địa chỉ mail, số diện thoại và nhiều thông tin cần thiết đáp ứng nhu cầu của con người. Với khá nhiều công cụ trên mạng để có thể bốc tách những comment, mail hay số điện thoại điển hình hiện nay là công cụ lọc comment trên mạng xã hội facebook hoặc một số phần mềm liên quan đến mảng marketing của iClick như SMS Marketing, Email Marketing... Nhìn chung mỗi công cụ chỉ phục vụ cho mỗi việc khác, vì thế với đề tài này, nhóm quyết định làm ra một công cụ bao quát nằm gọn trong một giao diện bao gồm bốc tách mail, số điện thoại và comment để đáp ứng được nhu cầu sử dụng của con người.

* + 1. Nhiệm vụ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tuần | Công việc | Kết quả | Tiến độ |
| 1 | 15/8 – 21/8 | - Tìm hiểu về Regular Expression  - Tìm hiểu tổng quan về đề tài  - Tìm hiểu và thực hiện Java cơ bản |  | -An, Đô  -An, Đô, Sơn  -Sơn |
| 2 | 22/8 – 28/8 | - Tìm hiểu về GUI trong Java  - Tìm hiểu cách đọc URL |  | -Sơn  -An, Đô |
| 3 | 29/8 – 4/9 | - Xây dựng GUI cho đề tài  - Tìm hiểu cách đọc URL |  | -An, Đô, Sơn  -An, Đô |
| 4 | 5/9 – 11/9 | - Hoàn thành GUI và đọc được url  -Tạo tài khoản github |  | -Sơn  -An, Đô |
| 5 | 12/9 – 18/9 | - Sử dụng Regular Expression để tiến hành lọc email từ website, phân tách theo dạng đuôi email : gmail, yahoo ... |  | -An, Đô, Sơn |
| 6 | 19/9 – 25/9 | - Sử dụng Regular Expression để tiến hành lọc số điện thoại từ website, phân tách được số điện thoại, nâng cao hơn sẽ phân tách được số di động thuộc theo chuẩn 10 số hoặc 11 số. |  | -An, Đô, Sơn |
| 7 | 26/9 – 2/10 | - Tìm hiểu cách lọc comment trong trang thegioididong.com  - Lọc email, sđt trùng… |  | -An, Đô  -Sơn |
| 8 | 3/10 – 9/10 | - Lọc comment của trang thegioididong.com  - Chức năng lưu file txt |  | -An, Đô  -Sơn |
| 9 | 10/10 – 16/10 | - Đóng gói ứng dụng và test lần cuối. |  | -An, Đô, Sơn |

* 1. **Tiến độ thực hiện**

Link github: <https://github.com/DoSonAn/DoAnTopic4>

# Phân tích lý thuyết

2. 1. **Regular Expression**
4. 1. 1. Regular Expression là gì?

* Regex (Regular Expression) có nghĩa là “Biểu thức chính quy” dùng để mô tả một mẫu chuỗi (string pattern) được sắp xếp theo một quy luật nhất định nào đó, ví dụ số điện thoại, ngày tháng, địa chỉ email…
* Khi bạn tìm 1 đoạn text, bạn có thể sử dụng một mẫu chuỗi để mô tả cái cần tìm.
* Một mẫu đơn giản có thể là một ký tự đơn.
* Mẫu phức tạp hơn có thể bao gồm nhiều ký tự, và có thể sử dụng để phân tích ngữ pháp, kiểm tra định dạng, thay thế và nhiều việc khác nữa.
  + 1. Tác dụng của Regular Expression
* Trong các kỹ thuật xử lý chuỗi (Ví dụ : Tìm kiếm, thay thế, sửa đổi …), thông thường chúng ta phải xác định các chuỗi tường minh để tìm kiếm. Các chuỗi này được gọi là từ khóa.
* Tuy nhiên không phải lúc nào chúng ta cũng có thể chỉ rõ chuỗi từ khóa đó mà chỉ có thể mô tả được quy luật để sinh ra chuỗi từ khóa đó, chẳng hạn làm thế nào để lấy được toàn bộ địa chỉ email trong một văn bản?
* Regex giúp ta dễ dàng giải quyết vấn đề này. Đơn giản bạn chỉ cần thiết lập ra quy luật của một địa chỉ email và yêu cầu chương trình tìm kiếm theo luật đó.
* Tóm lại, Regex có các tác dụng sau :
* Regex được sử dụng trong việc xử lý chuỗi : tìm kiếm, thay thế, bóc tách dữ liệu …
* Regex được sử dụng nhiều trong việc kiểm tra tính hợp lệ đầu vào của dữ liệu, ví dụ : địa chỉ email có hợp lệ hay không? Định dạng ngày tháng có đúng không?
  + 1. Cú pháp cho Regular trong java

|  |  |  |
| --- | --- | --- |
| STT | Biểu thức chính quy | Mô tả |
| 1 | . | Khớp (match) với bất kỳ ký tự nào |
| 2 | ^regex | Biểu thức chính quy phải khớp tại điểm bắt đầu |
| 3 | regex$ | Biểu thức chính quy phải khớp ở cuối dòng. |
| 4 | [abc] | Thiết lập định nghĩa, có thể khớp với a hoặc b hoặc c. |
| 5 | [abc][vz] | Thiết lập định nghĩa, có thể khớp với a hoặc b hoặc c theo sau là v hay z. |
| 6 | [^abc] | Khi dấu ^ xuất hiện như là nhân vật đầu tiên trong dấu ngoặc vuông, nó phủ nhận mô hình. Điều này có thể khớp với bất kỳ ký tự nào ngoại trừ a hoặc b hoặc c. |
| 7 | [a-d1-7] | Phạm vi: phù hợp với một chuỗi giữa a và điểm d và con số từ 1 đến 7. |
| 8 | X|Z | Tìm X hoặc Z. |
| 9 | XZ | Tìm X và theo sau là Z. |
| 10 | $ | Kiểm tra kết thúc dòng. |
| 11 | \d | Số bất kỳ, viết ngắn gọn cho [0-9] |
| 12 | \D | Ký tự không phải là số, viết ngắn gon cho [^0-9] |
| 13 | \s | Ký tự khoảng trắng, viết ngắn gọn cho [ \t\n\x0b\r\f] |
| 14 | \S | Ký tự không phải khoản trắng, viết ngắn gọn cho [^\s] |
| 15 | \w | Ký tự chữ, viết ngắn gọn cho [a-zA-Z\_0-9] |
| 16 | \W | Ký tự không phải chữ, viết ngắn gọn cho [^\w] |
| 17 | \S+ | Một số ký tự không phải khoảng trắng (Một hoặc nhiều) |
| 18 | \b | Ký tự thuộc a-z hoặc A-Z hoặc 0-9 hoặc \_, viết ngắn gọn cho [a-zA-Z0-9\_]. |
| 19 | \* | Xuất hiện 0 hoặc nhiều lần, viết ngắn gọn cho {0,} |
| 20 | + | Xuất hiện 1 hoặc nhiều lần, viết ngắn gọn cho {1,} |
| 21 | ? | Xuất hiện 0 hoặc 1 lần, ? viết ngắn gọn cho {0,1}. |
| 22 | {X} | Xuất hiện X lần, {} |
| 23 | {X,Y} | Xuất hiện trong khoảng X tới Y lần. |
| 24 | \*? | \* có nghĩa là xuất hiện 0 hoặc nhiều lần, thêm ? phía sau nghĩa là tìm kiếm khớp nhỏ nhất. |

VD : Muốn kiểm tra địa chỉ email có hợp lệ:

**package** test;

**public** **class** Email {

**private** **static** **boolean** CheckEmail(String s)

{

**boolean** a = s.matches("^[\_A-Za-z0-9-\\+]+(\\.[\_A-Za-z0-9-]+)\*@"

+ "[A-Za-z0-9-]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$");

**return** a;

}

**public** **static** **void** main(String[] args) {

String email = "nguyenan@gmail.com";

**if**(*CheckEmail*(email))

System.***out***.println("Email đúng định dạng");

**else**

System.***out***.println("Email sai định dạng");

}

}

Cùng phân tích: ^[\_A-Za-z0-9-\\+]+(\\.[\_A-Za-z0-9-]+)\*@"+ "[A-Za-z0-9-]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$

1. Đầu tiên ta kiểm tra kí tự đầu tiên của phần tên email bao gồm kí tự thường, in hoa, số, dấu \_, dấu – và dấu +. Vì vậy ta phải gom tất cả vào 1 nhóm ^[\_A-Za-z0-9-\\+]
2. Tiếp theo, ta kiểm tra các kí tự tiếp theo của phần tên email bao gồm kí tự thường, in hoa, số, dấu \_ và .. Vì vậy nên ta phải gom tất cả vào 1 nhóm [\_A-Za-z0-9-] (bởi vì giữa kí tự thường và in hoa có các kí tự đặc biệt nên ta không thể dùng A-z mà phải tách ra a-zA-Z).
3. Kí tự @
4. Phần tên miền thì bao gồm các kí tự thường, in hoa, số, dấu \_, dấu -, dấu .. Sau kí tự @ thì không có dấu chấm, dấu\_ nên biểu thức là [A-Za-z0-9-]. Tiếp theo là dấu chấm và các ký tự. Cuối cùng, sau dấu chấm phải có 2 kí tự trở lên nên biểu thức là (\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$

VD: Muốn kiểm tra 1 số điện thoại di động hợp lệ ở Việt Nam (không có khoảng trắng nhé)

**package** test;

**public** **class** Phone {

**private** **static** **boolean** CheckPhone(String s)

{

**boolean** a = s.matches("(\\+84|0\\d{9,10})");

**return** a;

}

**public** **static** **void** main(String[] args) {

String phone = "0961797005";

**if**(*CheckPhone*(phone))

System.***out***.println("Phone đúng định dạng");

**else**

System.***out***.println("Phone sai định dạng");

}

}

Cùng phân tích: **(\\+84|0)\\d{9,10}**

1. Đầu tiên, do số điện thoại có phần mở đầu có thể là +84 (ở Việt Nam) hoặc là 0, nên ta cần đặt vào trong group và thêm dấu |
2. Kế tiếp, 1 số điện thoại bao gồm 10 hoặc 11 chữ số, nhưng ta không tính phần đầu của số điện thoại nên chỉ còn khoảng 9 – 10 chữ số
   1. **URL**
4. 2. 1. URL là gì?

* URL có dạng một chuỗi mô tả cách tìm một tài nguyên trên Internet. URL có hai thành phần chính: giao thức cần thiết để truy cập tài nguyên và vị trí của tài nguyên.
* URL là từ viết tắt của Uniform Resource Locator và là một tài liệu tham khảo (một địa chỉ) cho một tài nguyên trên Internet.
  + 1. Tạo một URL
* Trong các chương trình Java của bạn, bạn có thể tạo ra một đối tượng URL đại diện cho một địa chỉ URL. Đối tượng URL luôn đề cập đến một URL tuyệt đối nhưng có thể được xây dựng từ một URL tuyệt đối, một URL tương đối, hoặc từ các thành phần URL.
* Cách dễ nhất để tạo ra một URLđối tượng là từ một mẫu Stringđại diện cho dạng địa chỉ URL có thể đọc được của con người. Đây thường là hình thức mà một người khác sẽ sử dụng cho một URL. Trong chương trình Java của bạn, bạn có thể sử dụng một Stringvăn bản chứa văn bản này để tạo một URLđối tượng: URL myURL = new URL("http://example.com/");
  + 1. Đọc trực tiếp một URL
* Phần này cho thấy các chương trình Java của bạn có thể đọc được từ URL sử dụng openStream()phương pháp như thế nào.
* Sau khi tạo thành công URL, bạn có thể gọi phương thức URL's openStream()để lấy một luồng mà bạn có thể đọc nội dung của URL. Các openStream()phương thức trả về một [java.io.InputStream](https://docs.oracle.com/javase/8/docs/api/java/io/InputStream.html)đối tượng, vì vậy đọc từ một URL có dễ dàng như đọc từ một input stream.
  + 1. Kết nối đến một URL
* Nếu bạn muốn làm nhiều hơn là chỉ đọc từ một URL, bạn có thể kết nối với nó bằng cách gọi openConnection() vào URL. Các openConnection() phương thức trả về một đối tượng URLConnection mà bạn có thể sử dụng cho truyền thông tổng quát hơn với URL, chẳng hạn như đọc từ nó, bằng văn bản cho nó, hoặc truy vấn nó cho nội dung và thông tin khác.
* Sau khi bạn tạo một URLđối tượng thành công , bạn có thể gọi phương thức URLcủa đối tượng openConnection để lấy một URLConnection đối tượng, hoặc một trong những lớp con của nó, ví dụ: [java.net.HttpURLConnection](https://docs.oracle.com/javase/8/docs/api/java/net/HttpURLConnection.html)
* Bạn có thể sử dụng URLConnectionđối tượng này để thiết lập các tham số và các thuộc tính yêu cầu chung mà bạn có thể cần trước khi kết nối. Kết nối đến đối tượng từ xa được đại diện bởi URL chỉ được khởi tạo khi URLConnection.connectphương thức được gọi. Khi thực hiện việc này, bạn sẽ khởi tạo một liên kết truyền thông giữa chương trình Java và URL qua mạng. Ví dụ, mã sau sẽ mở ra một kết nối đến trang web example.com:

**package** test;

**import** java.io.BufferedReader;

**import** java.io.InputStream;

**import** java.io.InputStreamReader;

**import** java.net.URL;

**import** java.net.URLConnection;

**import** java.util.ArrayList;

**public** **class** Url {

**public** **static** **void** main(String[] args) {

String link =" example.com";

URL url;

String content="";

**try** {

url = **new** URL(link);

URLConnection connection = url.openConnection();

InputStream inputstream = connection.getInputStream();

BufferedReader br = **new** BufferedReader(**new** InputStreamReader(inputstream));

String line = br.readLine();

**while**(line!=**null**)

{

content = content + line;

line = br.readLine();

}

br.close();

System.***out***.println(content);

} **catch** (Exception e) {

// **TODO**: handle exception

e.printStackTrace();

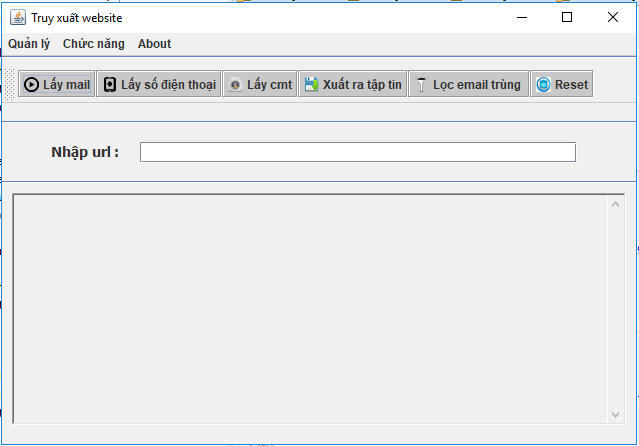
}

}

}

# Demo

3. 1. **Mô hình của phần mềm**



* 1. **Các chức năng của phần mềm**

3. 2. 1. **Lấy email**

Khi nhập một link trang web thì phần mềm lấy tất cả các email của trang về và hiển thị lên màn hình

* + 1. **Lấy số điện thoại**

Khi nhập một link trang web thì phần mềm lấy tất cả các số điện thoại của trang về và hiển thị lên màn hình

* + 1. **Lấy comment**

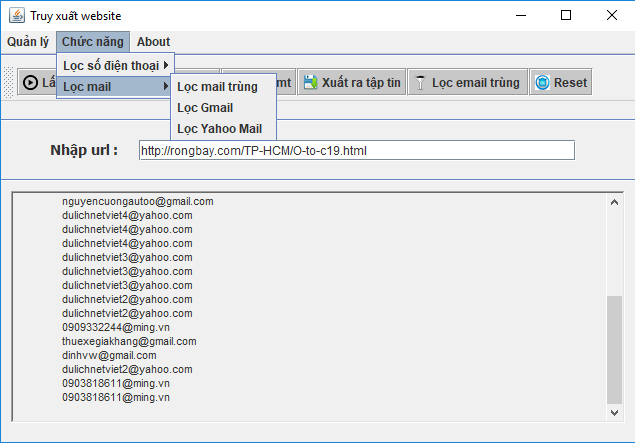
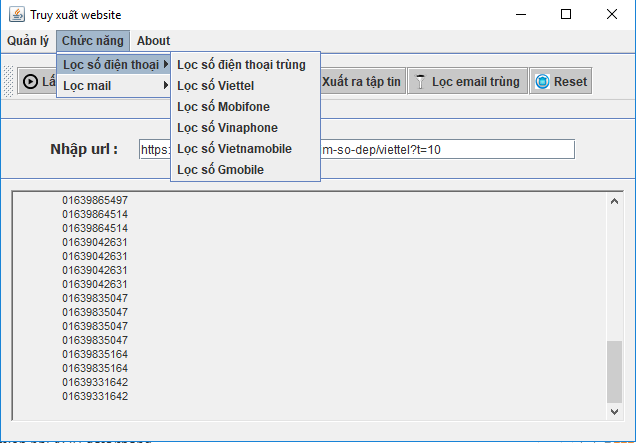
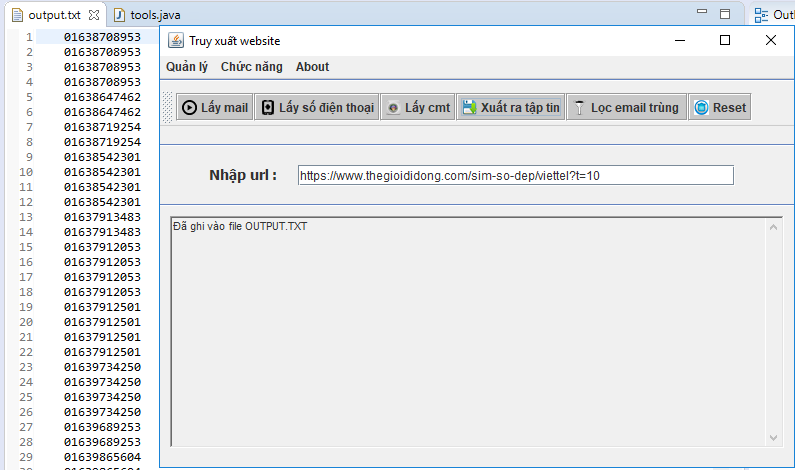
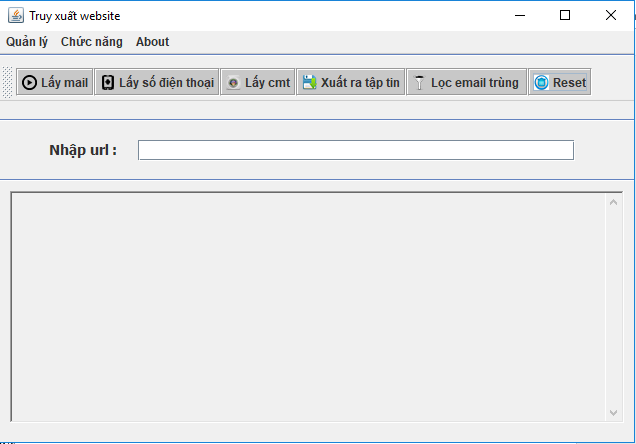
Lấy tất cả các đánh giá các sản phẩm của thế giới di động

* + 1. **Xuất ra tập tin**

Lưu tất cả các kết quả lọc được trên màn hình vào file output.txt

2. 2. 1. **Reset**

Màn hình trở về như lúc mới khởi động

* 1. **Kết quả**
     1. **Lấy email**
     2. **Lấy số điện thoại**
     3. **Xuất ra tập tin**
     4. **Reset**

# TÀI LIỆU THAM KHẢO:

Tìm hiểu về URL: <https://docs.oracle.com/javase/tutorial/networking/urls/index.html>

Tìm hiểu về Regular Expression:

<http://vietjack.com/java/regular_expression_trong_java.jsp>

Một vài biểu thức chính quy nên biết:

<https://kipalog.com/posts/30-doan-bieu-thuc-chinh-quy-ma-lap-trinh-vien-web-nen-biet>

Tham khảo code tra số điện thoại:

<https://github.com/datnq/Vietnamese-mobile-carrier>

Tìm hiểu về biểu thức chính quy trong java:

<http://o7planning.org/vi/10175/huong-dan-su-dung-bieu-thuc-chinh-quy-trong-java>

Tham khảo code lấy thông tin HTML trên web dân trí:

https://www.youtube.com/watch?v=EkhHx43Gkv8