Lingüística computacional: trabajando con expresiones regulares

Rogelio Nazar

Pontificia Universidad Católica de Valparaíso

Trabajando con expresiones regulares

¿Qué son las expresiones regulares?

- No son un producto de software
- Son un constructo teórico
- Fueron creadas en los años cincuenta por Stephen Kleene
- Hoy forma parte de muchos productos informáticos como procesadores de texto y lenguajes de programación

Trabajando con expresiones regulares

¿Qué son las expresiones regulares?

Una definición corta:

Son un lenguaje formal utilizado para detectar patrones en cadenas de caracteres.

Una definición más larga:

Son un mecanismo utilizado para caracterizar un conjunto de cadenas de caracteres de una manera económica, para buscar con una sola expresión un conjunto potencialmente grande de cadenas distintas.

Trabajando con expresiones regulares

¿Para qué las queremos?

- Para encontrar patrones en un corpus.
- Para automatizar procedimientos cuando tenemos demasiada cantidad de datos como para procesarla manualmente
- Para realizar operaciones *de búsqueda y reemplazo*.

Trabajando con expresiones regulares

Tipos de operaciones:

Alternancia

En general, cuando hacemos uso de la función de búsqueda de un procesador de texto, estamos buscando expresiones literalmente, como *casa* o *departamento*. Cuando usamos un editor de texto que tiene incorporadas las expresiones regulares, en cambio, podemos buscar contextos de ocurrencia de ambas expresiones si utilizamos un símbolo especial (la línea vertical, |).

Si formulamos la interrogación de la siguiente manera:
(casa|departamento)
el resultado será instancias de textos donde cualquiera de las dos expresiones aparece.
De la misma manera, si buscamos alsgo como lo siguiente:
(p|m)adre
Encontraremos contextos de cualqueira de los dos, *padre* o *madre*.

Trabajando con expresiones regulares

Tipos de operaciones:

Cuantificación

Los símbolos de cuantificación se utilizan en expresiones regulares para especificar al cantidad de veces en que algo debe aparecer. Los símbolos más comunes son: ?, + y *.

El primero, ?, significa que el símbolo precedente es opcional.
El segundo, +, significa que el símbolo precedente puede aparecer una o más veces.
Finalmente, *, significa que carácter puede aparecer cero o más veces.

Tipos de operaciones:

Cuantificacion: ?

El signo de interrogación, ?, significa que el carácter anterior es opcional.

Si buscamos algo como:
buenas?
Encontrará instancias en el texto tanto de la forma *buena* como de la forma *buenas*.
Lo mismo sucede en un caso como:
ob?scuro
Encontrará apariciones tanto de la forma *oscuro* como *obscuro*.

Trabajando con expresiones regulares

Tipos de operaciones:

Cuantificacion: +

El signo más, +, significa que el símbolo precedente puede aparecer una o más veces.

Si buscamos una expresión como:
buena+s
Encontrará formas como:
buenas, *buenaas*, *buenaaas*, *buenaaaas*,etc.

Trabajando con expresiones regulares

Tipos de operaciones:

Cuantificación: *

El asterisco, *, es similar al signo más con la diferencia de que en lugar de encontrar algo que aparece una o más veces, encuentra algo que ocurre cero o más veces.

Si buscamos algo como:
0*34
Encontrará:
34, 034, 0034, 00034, etc.

Tipos de operaciones:

Agrupamiento

Existen distintas formas de hacer agrupamiento con expresiones regulares. Lo más común es utilizar corchetes, [y], que especifican qué caracteres pueden aparecer en una determinada posición.

Por ejemplo: [a-z], cualquier letra de la a a la z puede ocurrir. Pero también podemos limitar este conjunto de manera arbitraria, como por ejemplo de la siguiente manera: [abcd], lo que sinificaría que en esa posición solo pueden aparecer las letras a, b, c y d.

Trabajando con expresiones regulares

Tipos de operaciones:

Agrupamiento

Otra forma de hacer agrupamientos es mediante el uso de paréntesis. Esta manera nos permite buscar secuencias de letras y almacenar lo capturado para utilizarlo más tarde, por ejemplo en una operación de reemplazo.

Considérese, por ejemplo, una lista de nombres en los que el primer nombre viene seguido del apellido, como en este ejemplo:
Fred Flintstone
pero lo que nosotros necesitamos es que aparezca primero el apellido, una coma, y recién entonces el nombre de pila, tal como sigue:
Flintstone, Fred
En ese caso tendríamos una expresión de búsqueda como la siguiente:
([a-zA-Z]+) ([a-zA-Z]+)
y una expresión de reemplazo como la que sigue:
\2, \1
Estos números nombran variables, que a su vez refieren las secuencias obtenidas entre paréntesis.

Tipos de operaciones: resumiendo
Operadores de alternancia:

REGEX	Descripción
.	Cualquier caracter (Excepto \n, en algunas implementaciones).
|	Disyunción. e.g. "a|b" corresponde a "a" o "b".
[ ]	Cualquier caracter incluido en el par de corchetes, como "[ab]" corresponde a "a" y "b"; "[0-9]" es cualquier dígito y [a-z] cualquier letra. Los dos se pueden combinar: [a-z0-9]

Tipos de operaciones: resumiendo cuantificación

REGEX	Descripción
?	El carácter precedente aparece 0 o 1 vez.
*	El carácter precedente aparece 0 o más veces.
+	El carácter precedente aparece 1 o más veces.

Tipos de operaciones: resumiendo agrupamiento

REGEX	Descripción
()	Se utiliza para almacenar el contenido de lo que se encuentra. Los valores se recuperan por medio de números que refieren al orden de los paréntesis: \1, \2, etc.

Tipos de operaciones: resumiendo negación

REGEX	Descripción
[^]	Encuentra cualquier caracter excepto los que aparecen dentro de los corchetes

Tipos de operaciones: resumiendo posición

REGEX	Descripción
^	Encuentra solo cosas que están al principio de una línea.
$	Encuentra solo cosas que están al final de la línea.

Tipos de operaciones: resumiendo otros

REGEX	Descripción
\t	tabulador
\n	carácter de final de línea
\r	En los archivos de textos creados con Windows, el carácter de final de línea viene precedido por el este otro símbolo (conocido como "retorno de carro")
\	Se utiliza para "escapar" caracteres, lo cual significa usar un símbolo de manera literal por oposición a su valor especial (por ejemplo, para buscar en el texto un asterisco o un signo más).

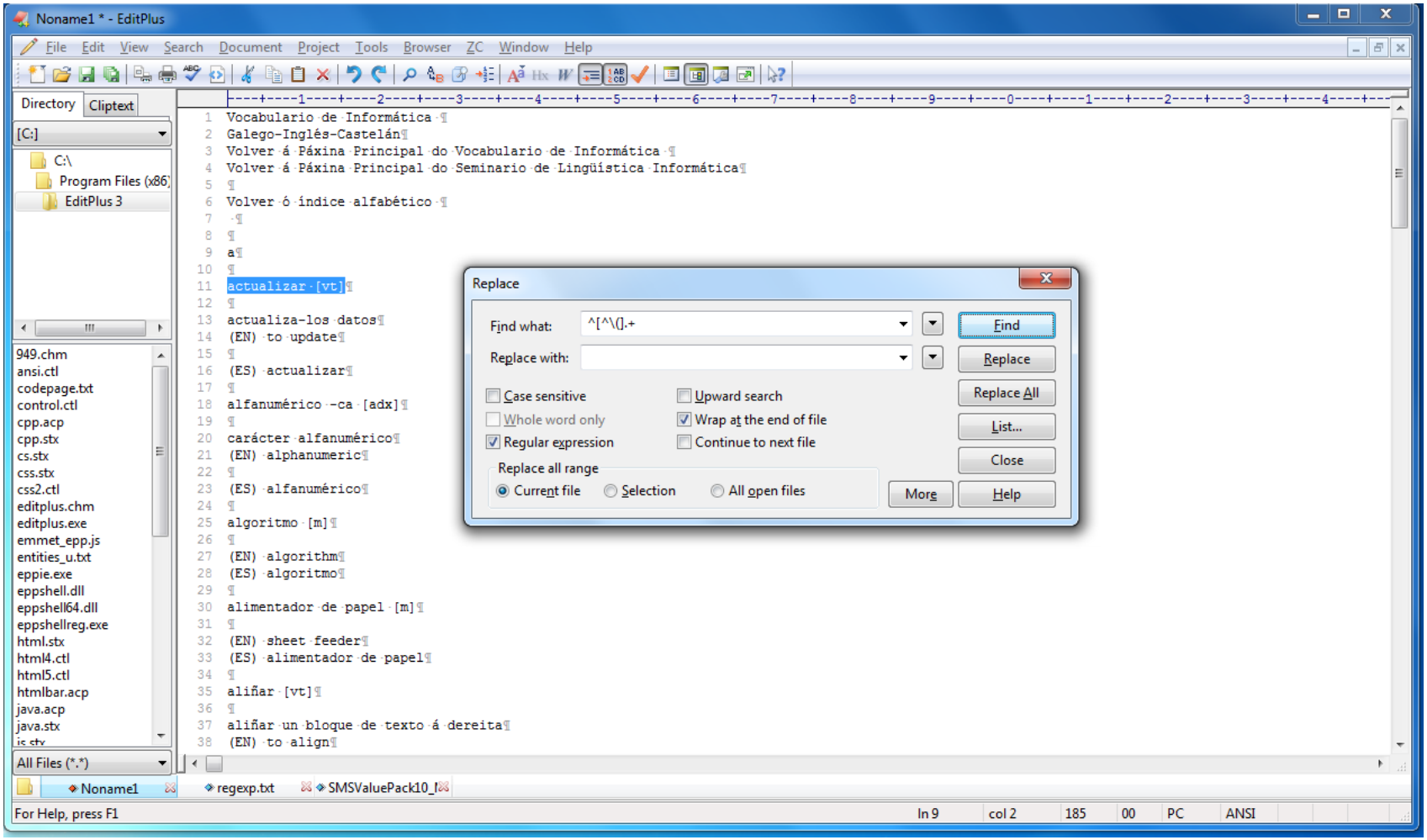
Trabajando con expresiones regulares

¿Qué herramientas utilizamos para trabajar con expresiones regulares?

- Editores de texto
- Aplicaciones Web
- Herramientas de línea de comando
- Lenguajes de programación

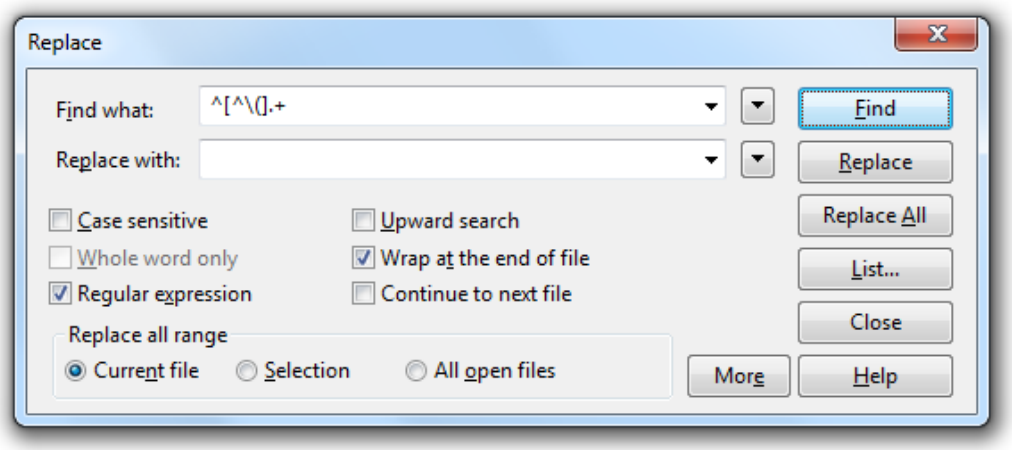
Trabajando con expresiones regulares

Editores de texto: EditPlus
http://www.editplus.com/



Trabajando con expresiones regulares

Editores de texto: EditPlus
http://www.editplus.com/



Editores de texto: jEdit
http://www.jedit.org/

www.jedit.org

Last Site Update: 28 July 2013 | Stable Version: 5.1.0

jEdit
Programmer's Text Editor

Download

jEdit
Follow +1
+ 1,802

jEdit is a mature programmer's text editor with hundreds (counting the time developing plugins) of person-years of development behind it. To download, install, and set up jEdit as quickly and painlessly as possible, go to the Quick Start page.

While jEdit beats many expensive development tools for features and ease of use, it is released as free software with full source code, provided under the terms of the GPL 2.0.

The jEdit core, together with a large collection of plugins is maintained by a world-wide developer team.

Some of jEdit's features include:

- Written in Java, so it runs on Mac OS X, OS/2, Unix, VMS and Windows.
- Built-in macro language; extensible plugin architecture. Hundreds of macros and plugins available.
- Plugins can be downloaded and installed from within jEdit using the "plugin manager" feature.
- Auto indent, and syntax highlighting for more than 200 languages.
- Supports a large number of character encodings including UTF8 and Unicode.
- Folding for selectively hiding regions of text.
- Word wrap.
- Highly configurable and customizable.
- Every other feature, both basic and advanced, you would expect to find in a text editor. See the Features page for a full list.

About
- Main Site
- Features
- Compatibility
- Screenshots
- Icons and Images
- Reviews
- Download
- Plugins

Community
- jEdit Community
- jEdit Wiki

Help
- Quick Start Guide
- Online Documentation
- Feedback and Support

Development Links
- Development
- SourceForge Project

Tell us what you think

Editores de texto: NotePad Plus
http://notepad-plus-plus.org/

more languages

Notepad++

home
download
news
features
resources
contribute
donate
contributors
links

download

Current Version: 6.6.7

News

Notepad++ 6.6.7 released
Jun 24 2014

Notepad++ 666 Friday the 13th edition
Jun 12 2014

Response and Apology for Sexist Jokes
Jun 07 2014

Notepad++ 6.6.4 Tiananmen June Fourth Incident Edition
Jun 04 2014

Notepad++ 6.6.3 released
May 18 2014

Notepad++ 6.6.2 released
May 08 2014

v6.6.1 - May the 4th be with you
May 04 2014

Notepad++ Workers' Day Edition (v6.6)
May 01 2014

Notepad++ 6.5.5 released
Mar 06 2014

Notepad++ 6.5.4 released
Feb 18 2014

more news »

About

Notepad++ is a free (as in "free speech" and also as in "free beer") source code editor and Notepad replacement that supports several languages. Running in the MS Windows environment, its use is governed by GPL License.

Based on the powerful editing component Scintilla, Notepad++ is written in C++ and uses pure Win32 API and STL which ensures a higher execution speed and smaller program size. By optimizing as many routines as possible without losing user friendliness, Notepad++ is trying to reduce the world carbon dioxide emissions. When using less CPU power, the PC can throttle down and reduce power consumption, resulting in a greener environment.

*D:\source\notepad4ever.cpp - Notepad++

Notepad_plus.cpp | notepad4ever.cpp

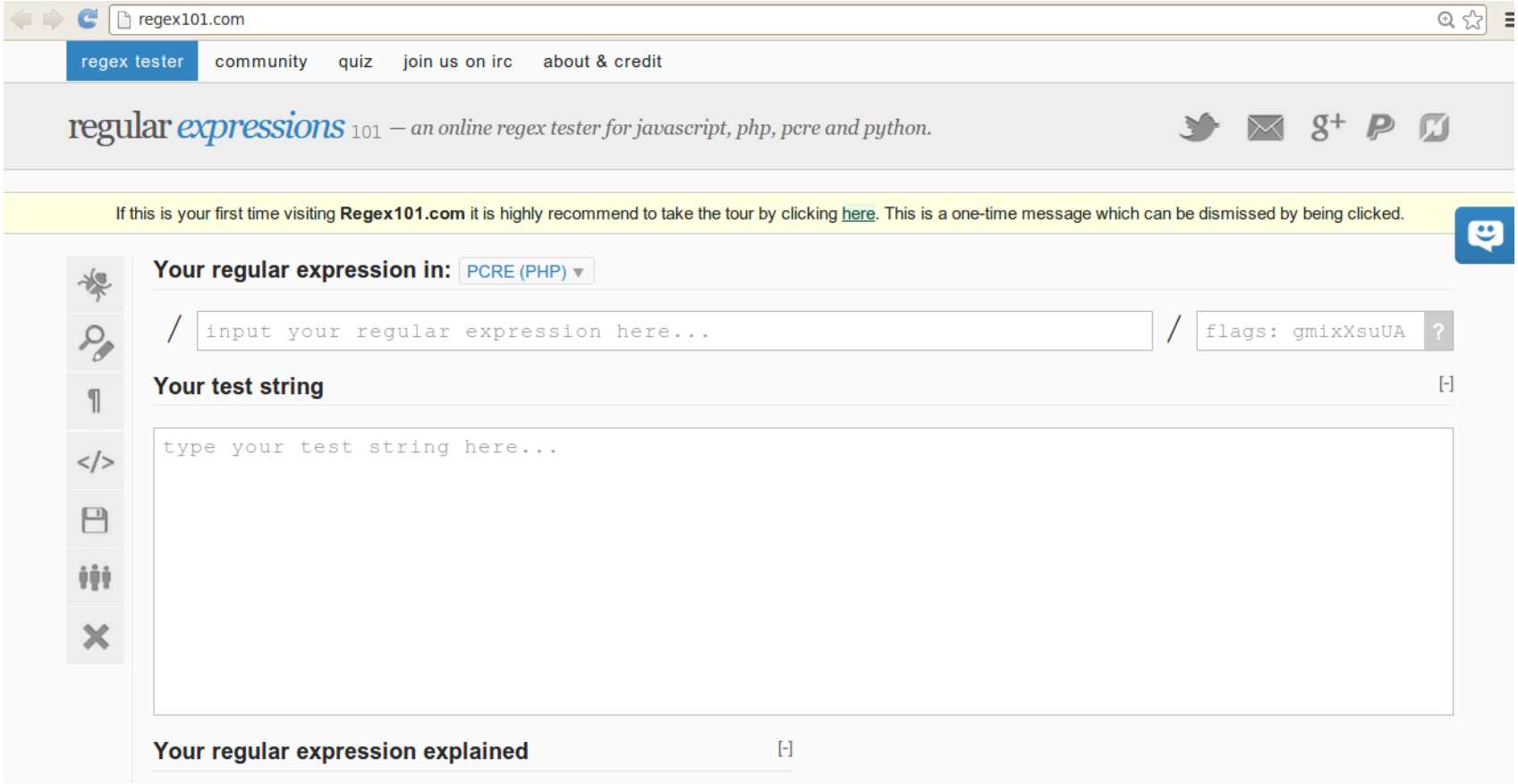
```
1  #include <GPL.h>
2  #include <free_software.h>
3
4  void notepad4ever()
5  {
6      while (true)
7      {
8          Notepad++;
9      }
10 }
11
```

You're encouraged to translate Notepad++ into your native language if there's not already a translation present in the Binary Translations page.

I hope you enjoy Notepad++ as much as I enjoy coding it.

Trabajando con expresiones regulares

Aplicaciones Web:
http://www.regex101.com/



Trabajando con expresiones regulares

Aplicaciones Web:
http://www.regular-expressions.info/

www.regular-expressions.info/javascriptexample.html

Regex Tools

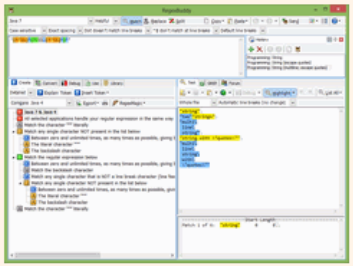
- grep
- PowerGREP
- RegexBuddy
- RegexMagic

General Applications

- EditPad Lite
- EditPad Pro

Languages & Libraries

- Delphi
- GNU (Linux)
- Groovy
- Java
- JavaScript
- .NET
- PCRE (C/C++)
- Perl
- PHP
- POSIX
- PowerShell



Interactively create and test regular expressions with RegexBuddy.
Create and analyze regex patterns with RegexBuddy's intuitive regex building blocks. Quickly **test** regular expressions on sample data and files in a safe sandbox. **Debug** regexes easily with real-time highlighting and informative regex match details. Get your own copy of RegexBuddy now.

JavaScript RegExp Example: Regular Expression Tester

Feel free to test JavaScript's RegExp support right here in your browser. Obviously, JavaScript (or Microsoft's variant JScript) will need to be enabled in your browser for this to work. Since this tester is implemented in JavaScript, it will reflect the features and limitations of your web browser's JavaScript implementation. If you're looking for a general-purpose regular expression tester supporting a variety of regex flavors, grab yourself a copy of RegexBuddy.

Learn how to use the JavaScript RegExp object.

Regexp: \bt[a-z]+\b

Subject string: This is a test of the JavaScript RegExp object

Test Match | Show Match

Replacement text: replaced

Result: This is a replaced of replaced JavaScript RegExp ob

Replace

www.regexbuddy.com/test.html

Trabajando con expresiones regulares

Aplicaciones Web:
http://www.regexplanet.com/

www.regexplanet.com/advanced/perl/index.html

RegexPlanet | Testing | Cookbook | Support

Regular expression:

Options:
- multiline (m)
- single line (s)
- case-insensitive (i)
- Extended with whitespace and comments (x)
- Preserve matched strings (p)
- Global match (g)
- Keep current position (c)
- Use ASCII charset rules (a)
- Use default charset rules (d)
- Use locale charset rules (l)
- Use Unicode charset rules (u)

Replacement:

Test | More Inputs

Input 1:

Input 2:

Input 3:

Trabajando con expresiones regulares

Aplicaciones Web:
http://www.gskinner.com/RegExr/

gskinner.com/RegExr/

Match | Replace

\be(\w*)s\b

global | ignoreCase | extended | dotall | multiline | Share Link

Welcome to RegExr 0.3b, an intuitive tool for learning, writing, and testing Regular Expressions. Key features include:

* real time results: shows results as you type
* code hinting: roll over your expression to see info on specific elements
* detailed results: roll over a match to see details & view group info below
* built in regex guide: double click entries to insert them into your expression
* online & desktop: regexr.com or download the desktop version for Mac, Windows, or Linux
* save your expressions: My Saved expressions are saved locally
* search Community expressions and add your own
* create Share Links to send your expressions to co-workers or link to them on Twitter or your blog [ex. http://RegExr.com?2rjl6]

Built by gskinner.com with Flex 3 [adobe.com/go/flex] and Spelling Plus Library for text highlighting [gskinner.com/products/spl].

Welcome to RegExr 0.3b, an intuitive tool for learning, writing, and testing Regular Expressions. Key features include:

* real time results: shows results as you type
* code hinting: roll over your expression to see info on specific
* detailed results: roll over a match to see details & view group info below
* built in regex guide: double click to insert them into your expression
* online & desktop: regexr.com or download the desktop version for Mac, Windows, or Linux
* save your : My Saved are saved locally
* search Community and add your own
* create Share Links to send your to co-workers or link to them on Twitter or your blog [ex. http://RegExr.com?2rjl6]

Built by gskinner.com with Flex 3 [adobe.com/go/flex] and Spelling Plus Library for text highlighting [gskinner.com/products/spl].

RegExp: /\be(\w*)s\b/g
pattern: \be(\w*)s\b
flags: g
1 capturing groups:
group 1: (\w*)

Samples | My Saved | Community

show all

.
\w
\W
\d
\D
\s
\S
[ABC]
[^ABC]
[a-z]
[a-zA-Z]
[^f-m]
[0-9]

Matches any character, except for line breaks if dotall is false.

0.3.1b built by gskinner.com

About RegExr | Desktop Version

Trabajando con expresiones regulares

Aplicaciones Web:
http://regexpal.com/

Trabajando con expresiones regulares

Herramientas en línea de comando: grep (para Linux y Mac)

```
$ grep --color t[a-z] midsummer
I know a bank where the wild thyme blows,
Where oxlips and the nodding violet grows,
Quite over-canopied with luscious woodbine,
With sweet musk-roses and with eglantine.
$
```

Trabajando con expresiones regulares

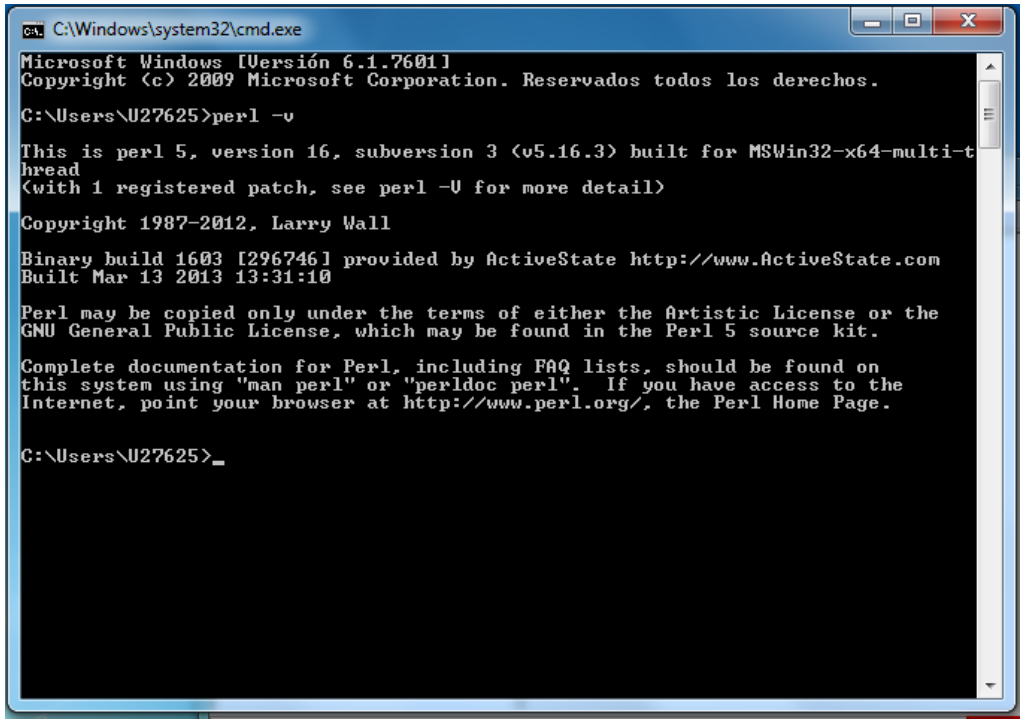
Herramientas en línea de comando: findstr (para Windows)

```
C:\Windows\system32\cmd.exe
  UDP    [::]:49220             *:*
  UDP    [::1]:1900             *:*
  UDP    [::1]:49561            *:*
  UDP    [fe80::c68:3cf8:ab1a:be66%12]:1900   *:*
  UDP    [fe80::c68:3cf8:ab1a:be66%12]:49562   *:*
  UDP    [fe80::689e:6f47:5330:cb90%8]:1900   *:*
  UDP    [fe80::689e:6f47:5330:cb90%8]:49557   *:*
  UDP    [fe80::6c31:9411:f388:954a%17]:1900   *:*
  UDP    [fe80::6c31:9411:f388:954a%17]:49555   *:*
  UDP    [fe80::7484:3849:5254:28e5%9]:1900   *:*
  UDP    [fe80::7484:3849:5254:28e5%9]:49556   *:*
  UDP    [fe80::b0e0:6c44:8f3c:99e2%19]:53   *:*
  UDP    [fe80::b0e0:6c44:8f3c:99e2%19]:1900   *:*
  UDP    [fe80::b0e0:6c44:8f3c:99e2%19]:49560   *:*
  UDP    [fe80::c8b7:9912:ee43:52a1%15]:1900   *:*
  UDP    [fe80::c8b7:9912:ee43:52a1%15]:49559   *:*
  UDP    [fe80::d13c:1690:22a6:2f3%13]:1900   *:*
  UDP    [fe80::d13c:1690:22a6:2f3%13]:49558   *:*

C:\Users\Dracula>netstat -an | findstr 445
  TCP    10.0.0.1:445           10.0.0.200:1131        ESTABLISHED
  TCP    192.168.1.135:445      192.168.1.137:1235     ESTABLISHED
  TCP    [::]:445               [::]:0                 LISTENING

C:\Users\Dracula>
```

Lenguajes de programación (como Perl)



Trabajando con expresiones regulares

Un caso de estudio: extraer información de un glosario

- Hay una gran diversidad de recursos en la web
- pero todos tienen formatos distintos
- y no existe un único programa informático para entenderlos a todos.
- Así que estudiaremos un simple caso de conversión de información no estructurada en información estructurada

Trabajando con expresiones regulares

Un caso de estudio: extraer información de un glosario
Ejemplo tomado de: http://webs.uvigo.es/sli/lexico/a.html
Tomaremos de aquí los equivalentes de inglés y castellano.

alimentador de papel [m]

(EN) sheet feeder

(ES) alimentador de papel

---

aliñar [vt]

aliñar un bloque de texto á dereita

(EN) to align

(ES) alinear

---

almacenaxe óptica [f]

(EN) optical storage

(ES) almacenamiento óptico

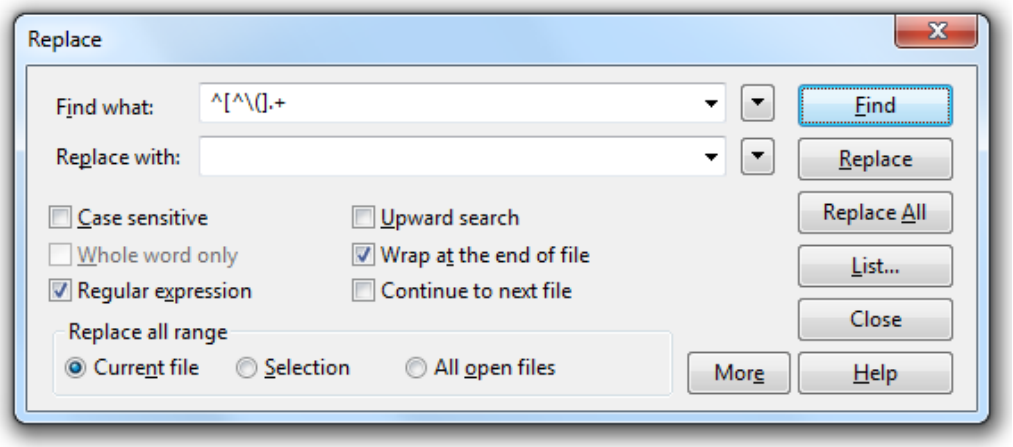
Trabajando con expresiones regulares

Para este ejercicio, vamos a copiar y pegar el contenido de esta web en un fichero de texto:

```
---+----1----+----2----+----3----+----4----+---
1  actualizar [vt]
2
3   (EN) to update
4
5   (ES) actualizar
6
7  alfanumérico -ca [adx]
8
9   (EN) alphanumeric
10
11  (ES) alfanumérico
12
13 algoritmo [m]
14  (EN) algorithm
15
16  (ES) algoritmo
17
18 alimentador de papel [m]
19  (EN) sheet feeder
20
21  (ES) alimentador de papel
```

Trabajando con expresiones regulares

Con la función de búsqueda y reemplazo de nuestro editor de texto,



vamos a buscar `^[^\(].+` y lo reemplazamos por nada. Lo que estamos haciendo aquí es eliminar todas las líneas que no empiezan por paréntesis, porque esas no nos interesan.

Trabajando con expresiones regulares

Lo siguiente es eliminar los signos (EN) y (ES) y poner los equivalentes en una misma línea. Podemos hacer todo esto con un solo movimiento buscando esto:

```
\n+\(EN\) *([^\n]+)\n\(ES\) *
```

y reemplazando por esto:

```
\1\t
```

Lo que estamos haciendo es primero eliminar los símbolos de lengua –(EN) y (ES)– y el carácter de final de línea que aparece al final del término en inglés(\n). Guardamos el término en inglés por medio del uso de los paréntesis, para recuperarlo luego con la variable \1.

El resultado es el término en inglés seguido de su equivalente en castellano en cada una de las líneas separado por un caracter tabulador.

to update	actualizar
alphanumeric	alfanumérico
algorithm	algoritmo
sheet feeder	alimentador de papel
to align	alinear

Fin del trabajo con expresiones regulares.
¡Pasamos al siguiente tema!