Expresiones regulares ("Regular Expresion" o "Regex")

Ya antes hablamos brevemente (o más bien utilizamos para nuestros propósitos) la expresiones regulares. En este artículo explicaremos de la manera más sencilla posible este quebradero de cabeza para muchos de nosotros.

¿Qué son las expresiones regulares?

> Una expresión regular es una secuencia de caracteres que definen (o más bien *conforman*) un patrón de búsqueda.

Ejemplo:

Si nuestro patrón de búsqueda es "perro" y *lo aplicamos* a un archivo de texto con la siguiente lista:

```
1.-Tortuga.
2.-Autobús.
3.-Perro.
4.-Nutria gigante o perro de agua.
5.-Zamuro.
6.-Can (perro).
7.-Cinco céntimos de peseta (perra chica).
```

obtendremos el siguiente resultado (filtrado de líneas):

```
4.-Nutria gigante o perro de agua.
6.-Can (perro).
```

Obvio que se trata de devolver y mostrar el contexto donde se encuentre la cadena de búsqueda

(línea u oración e incluso párrafo completo) sin embargo veremos que es sumamente útil para localizar y sustituir rápidamente por otra cadena de texto.

En nuestro ejemplo no resulta seleccionada la línea número 3 ni 7, veamos por qué.

Trabajando con letras, no con palabras

Fíjense que empleamos las comillas para denotar que es exactamente eso lo que buscamos, *todo en minúsculas* y la línea seis no cumple a cabalidad con lo que ordenamos buscar. Incluso las siguientes palabras:

```
p erro
perro-s
p e r r o
```

tampoco son las que buscamos porque ya bien tienen espacios o guiones, sin embargo:

```
-perro, perro albarraniego.
```

sí que cumplen pues los guiones o espacios están fuera y no "en el medio". Es decir, estamos buscando una letra pe seguida de una letra e, luego dos erres y la vocal o, *todo junto ("sin espacios")*, en ese mismo orden, en minúsculas todos los caracteres.

El punto y el asterisco

Como vimos, nosotros los seres humanos tenemos una inteligencia superior y cuando buscamos algo también aceptamos sus equivalentes, así no estén explícitamente en la orden que dimos. **Esto resulta totalmente desconcertante a nuestras computadoras** y muchos científicos, matemáticos y programadores luchan todos los días para que las máquinas entiendan si quiera algo de lo que les pedimos (no hablemos ya de que piensen por sí mismas, todas los ordenadores del mundo juntos no tienen más inteligencia que un insecto cualquiera).

Queremos decir: que para nosotros buscar "perro" y que en el listado se encuentre la palabra "perra" y no haya sido seleccionada... pues a los humanos nos llama la atención esto.

Como dijimos estamos buscando carácter por carácter, letra por letra y "perra" no se corresponde exactamente y acá es donde comenzamos a utilizar un *comodín (por favor no confundir con* ***metacarácter,*** *no)*.

Para ello utilizaremos el punto para indicar que allí queremos buscar **una sola letra o carácter, sin importar cual sea.** Por eso ahora nuestra búsqueda sería "perr." y nuestro resultado sería el siguiente:

```
4.-Nutria gigante o perro de agua.
5.-Zamuro.
6.-Can (perro).
7.-Cinco céntimos de peseta (perra chica).
```

Ahora bien, *si tuviéramos en la* lista, palabras como "**perri**" o **"perru"** (del todo ilógicas) -o incluso **"perr_"-** también las devolvería como resultado... ¿Para qué nos serviría esto? Imaginen que programamos un corrector ortográfico: este toma palabra por palabra *(cadena de búsqueda)* en una lista de palabras "bien escritas" y si no la consigue asume que es un posible error ortográfico (decimos *posible* porque puede ser una palabra nueva o muy especializada que no tengamos en nuestro diccionario). **Pero vamos más allá**, si vamos de derecha a izquierda sustituyendo una letra por un punto podremos sugerir al usuario un candidato a corrección. Evidentemente que ir letra por letra puede ser tedioso, incluso para las máquinas (más ciclos de reloj consumidos), así que veremos el segundo comodín.

Veremos que el asterisco funciona de forma parecida al punto *pero en este caso no importa la longitud de cadena:* si queremos que devuelva también resultados como "perrera" o "perreda" tendremos que buscar por "perr*a"; o si queremos resultados como "perraje" o "perrengue" deberemos usar "perr*e".

> El asterisco funciona sin importar su longitud, la cual incluso **puede ser cero.** Exacto: cuando buscamos "perr*a" también obtendríamos "perra", ¡sorprendente! ¿cierto?

Incluso podremos combinar dos comodines y les proponemos que imaginen la lista de resultados posibles (de palabras registradas en cualquier diccionario) si buscamos por «perr*.», según estas

normas que estamos usando para propósitos didácticos.

Mantengamos la mente abierta porque en realidad las expresiones regulares funcionan de una manera un tanto más complicada

¿Cómo funciona?

Vamos a utilizar una noticia cualquiera del diario nacional venezolano «*Últimas Noticias*» y extraemos una línea que contiene la palabra "oro":

```
Fortalezcamos el bolívar, resp
aldémoslo en oro
, deslindémoslo del
 dólar, aumentemos las reservas en oro
, que al estar en las bóvedas del BCV no podrán ni robarlas, ni bloquearl
as.
```

Buscar la *expresión regular* "oro" va de la siguiente manera:

1. Tomamos la primera letra de la expresión regular, "o".
2. De izquierda a derecha, tomamos la primera letra de la línea, y la comparamos.
3. Si no coincide tomamos la siguiente letra de la línea.
4. Si coincide tomamos *la siguiente letra de la línea* y la comparamos con la segunda letra de la expresión regular.
5. Si no coincide (y vemos claro que no porque: "Fortalezcamos...") continuamos de nuevo con el paso número 3.
6. Como sospechamos, llegará un momento que coincidirán las tres letras de la expresión regular contra las tres letras en análisis de la línea, lo cual devuelve positivo, pero esperen, aún hay más.
7. Se debe seguir revisando ¿hasta el final de la línea? Pues no porque la expresión regular tiene tres caracteres, así que deberemos restar (en este ejemplo, 180 - 3 =177, hasta ese número de carácter).
8. Lo del paso anterior es importante (ver si hay más coincidencias) si estamos por sustituir texto, generalmente las expresiones regulares son muy utilizadas en ese propósito.

Caracteres, comodines, juegos de caracteres y anclas

Antes de tocar este punto, definiremos que "devolver un resultado" lo vamos a tomar como el concepto positivo de haber encontrado *la expresión regular en una cadena de caracteres (texto, archivo, línea, párrafo, etc.)*

Ahora de manera muy básica y con fines didácticos podemos decir que las expresiones regulares constan de al menos uno o más de los siguientes componentes, de menor a mayor jerarquía:

- Caracteres en sí mismos.
- Comodines, caracteres que se comportan como comodines, ya vimos el punto y el asterisco.
- Juegos de caracteres:
    - Implícitos:
        - Un solo carácter es el mínimo juego de caracteres posible y que devuelve como resultado un carácter (el mismo carácter, obvio).
        - Un comodín por sí solo es un juego de caracteres mínimo que puede representar un solo carácter como resultado (el caso del punto, por ejemplo) o varios caracteres como resultado (el caso del asterisco, por ejemplo).
    - Explícitos:
        - Van encerrados o delimitados entre dos *metacaracteres:* para ello utilizamos el par de corchete rectos "**[]**" y dentro colocaremos uno o varios de los

Fuentes consultadas

En idioma castellano

-

En idioma inglés

- «Regex For Noobs (like me!) - An Illustrated Guide».