Introducción

**¿Qué es una expresión regular?**

Las regex son las unidades de descripción de los lenguajes regulares, que se incluyen en los denominados lenguajes formales. Son un instrumento clave de la informática teórica, la cual, entre otras cosas, establece las bases para el desarrollo y la ejecución de programas informáticos, así como para la construcción del compilador necesario para ello. Es por esto que las expresiones regulares, también denominadas regex y basadas en reglas sintácticas claramente definidas, se utilizan principalmente en el ámbito del desarrollo de software.

Para cada regex existe un denominado autómata finito (también conocido como máquina de estado finito) que acepta el lenguaje especificado por la expresión y que, con ayuda de la construcción de Thompson, se desarrolla a partir de una expresión regular. Por otro lado, para cada autómata finito también hay una expresión regular que describe el lenguaje aceptado por el autómata. Este puede generarse bien con el algoritmo de Kleene o bien con la eliminación de estados.

Desarrollo:

A continuación, listamos algunos de los **caracteres especiales de sintaxis** que pueden ampliar los patrones con opciones específicas:

| **Caracteres especiales regex de sintaxis** | **Función** |
| --- | --- |
| [] | Los corchetes identifican a una clase de caracteres que siempre representa a un único carácter en un patrón de búsqueda. |
| () | Los paréntesis identifican un grupo de caracteres formado por uno o varios caracteres y que pueden operarse unos dentro de los otros. |
| - | Funciona a modo de especificación del área (de […] hasta […]) cuando se sitúa entre dos caracteres normales. |
| ^ | Limita la búsqueda al inicio de una línea (otra función: elemento de negación en clases de caracteres). |
| $ | Limita la búsqueda al final de una línea. |
| . |  |
| \* | El número del carácter, de la clase o del grupo situado antes del asterisco puede ser aleatorio (cero incluido). |
| + | El carácter, la clase o el grupo antes de un signo más debe aparecer como mínimo una vez. |
| ? | El carácter, la clase o el grupo antes del signo de interrogación es opcional y puede aparecer como máximo una vez. |
| {n} | El carácter, la clase o el grupo anteriores aparecen exactamente n veces. |
| {n,m} | El carácter, la clase o el grupo anteriores aparecen como mínimo n veces y como máximo m veces. |
| {n,} | El carácter, la clase o el grupo anteriores aparecen como mínimo n veces o con frecuencia. |
| \b | Tiene en cuenta el límite de palabra durante la búsqueda. |
| \B | Ignora el límite de palabra durante la búsqueda. |
| \d | Cualquier dígito; abreviatura para la clase de caracteres [0-9]. |
| \D | Cualquier no dígito; abreviatura para la clase de caracteres [^0-9]. |
| \w | Cualquier carácter alfanumérico; abreviatura para la clase de caracteres [a-zA-Z\_0-9]. |
| \W | Cualquier carácter no alfanumérico; abreviatura para la clase de caracteres [^\w]. |

A partir de esta expresión se puede verificar si un correo electrónico es válido:

**^[a-z0-9\_-]+(?:\.[a-z0-9\_-]+)\*@(?:[a-z0-9](?:[a-z0-9-]\*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]\*[a-z0-9])?$**

**^** Limita la búsqueda al inicio de una línea

**[a-z0-9\_-]+** acepta los caracteres de la a la z y del 0 al 9, incluyendo \_ o -, y todos estos símbolos se pueden repetir una o más veces.

**?:\.[a-z0-9\_-]+** ? El carácter, la clase o el grupo antes del signo de interrogación es opcional y puede aparecer como máximo una vez. \. Se coloca cuando se escapa la función del carácter “.” Y pasa a ser el carácter “.”. Este punto tiene que continuar con uno de los símbolos de la a a la z, del 0 al 9, \_ o -, todo esto una o más veces.

**(?:\.[a-z0-9\_-]+)\*** Toda la expresión se puede usar 0 a más veces.

**@** Continua con el carácter “@”

**[a-zA-Z0-9-]+** De la a a la z, de la A a la Z, del 0 al 9 se aceptan una o más veces.

**(?:[a-z0-9](?:[a-z0-9-]\*[a-z0-9])?\.)+** ? El carácter, la clase o el grupo antes del signo de interrogación es opcional y puede aparecer como máximo una vez. \. Se coloca cuando se escapa la función del carácter “.” Y pasa a ser el carácter “.”. Este punto tiene que continuar con uno de los símbolos de la a a la z, de la A a Z, del 0 al 9, todo esto una o más veces.

**[a-z0-9](?:[a-z0-9-]\*[a-z0-9])?$** Todo el conjunto repetido 0 o más veces. $Limita la búsqueda al final de una línea.

Conclusión:

Víctor Hugo Vázquez Gómez

Las expresiones regulares funcionan como un patrón de búsqueda a la hora de tener lenguajes formales, tales como un lenguaje de programación, o en sí una cadena de texto como un correo electrónico. El uso de esta nos ayudó a definir si un correo electrónico es válido o no gracias a su característica principal que es la búsqueda de patrones. Con ella nos permite tener una gran velocidad a la hora de validar su certeza.

Nicole Rodríguez González

Las expresiones reculares son conjuntos de descripciones para formar un lenguaje con reglas en este, en base a este se pueden realizar programas informáticos. Usamos este para definir las especificaciones que se deben cumplir para que un correo electrónico sea valido.