EXPRESIONES REGULARES

EXPRESIÓN REGULAR

Las expresiones regulares son patrones utilizados para encontrar una determinada combinación de caracteres dentro de una cadena de texto.

Las expresiones regulares proporcionan una manera muy flexible de buscar o reconocer cadenas de texto. Por ejemplo, el grupo formado por las cadenas Handel, Händel y Haendel se describe con el patrón "H(a|ä|ae)ndel".

La mayoría de las formalizaciones proporcionan los siguientes constructores: una expresión regular es una forma de representar los lenguajes regulares (finitos o infinitos) y se construye utilizando caracteres del alfabeto sobre el cual se define el lenguaje.

Construcción de una expresión regular

Construcción de una expresión regular

Específicamente, las expresiones regulares se construyen utilizando los operadores unión, concatenación y clausura de Kleene. Toda expresión regular tiene algún autómata finito asociado.

Alternación

Una barra vertical separa las alternativas, las cuales son evaluadas de izquierda a derecha. Por ejemplo, "amarillo|azul" se corresponde con amarillo o azul

Construcción de una expresión regular

Cuantificación

Un cuantificador tras un carácter especifica la frecuencia con la que este puede ocurrir. Los cuantificadores más comunes son "?", "+" y "*":

- El signo de interrogación ? indica que el carácter que le precede puede aparecer como mucho una vez. Por ejemplo, "ob?scuro" se corresponde con oscuro y obscuro.
- El signo más + indica que el carácter que le precede debe aparecer al menos una vez. Por ejemplo, "ho+la" describe el conjunto infinito hola, hoola, hooola, hoooola, etcétera.
- El asterisco * indica que el carácter que le precede puede aparecer cero, una, o más veces. Por ejemplo, "0*42" se corresponde con 42, 042, 0042, 00042, etcétera.

Construcción de una expresión regular

Agrupación

Los paréntesis pueden usarse para definir el ámbito y precedencia de los demás operadores. Por ejemplo, "(p|m)adre" es lo mismo que "padre|madre", "(des)?amor" se corresponde con amor y con desamor.

Los constructores pueden combinarse libremente dentro de la misma expresión, por lo que "H(ae?|ä)ndel" equivale a "H(a|ae|ä)ndel". La sintaxis precisa de las expresiones regulares cambia según las herramientas y aplicaciones consideradas.

Aplicaciones

Aplicaciones

Su utilidad más obvia es la de describir un conjunto de cadenas para una determinada función, resultando de utilidad en editores de texto y demás aplicaciones informáticas para buscar y manipular textos.

Numerosos editores de texto y otras herramientas utilizan expresiones regulares para buscar y reemplazar patrones en un texto. Por ejemplo, las herramientas proporcionadas por las distribuciones de Unix (incluyendo el editor sed y el filtro grep) popularizaron el concepto de expresión regular entre usuarios no programadores, aunque ya era familiar entre los programadores.

Inicialmente, este reconocimiento de cadenas se programaba para cada aplicación sin mecanismo alguno inherente al lenguaje de programación pero, con el tiempo, se ha ido incorporando el uso de expresiones regulares para facilitar programar la detección de ciertas cadenas. Por ejemplo, Perl tiene un potente motor de expresiones regulares directamente incluido en su sintaxis. Otros lenguajes lo han incorporado como funciones específicas sin incorporarlo a su sintaxis.

Las expresiones regulares en programación

Las expresiones regulares en programación

En el área de la programación, las expresiones regulares son un método por medio del cual se pueden realizar búsquedas dentro de cadenas de caracteres. Sin importar la amplitud de la búsqueda requerida de un patrón definido de caracteres, las expresiones regulares proporcionan una solución práctica al problema. Adicionalmente, un uso derivado de la búsqueda de patrones es la validación de un formato específico en una cadena de caracteres dada, como por ejemplo fechas o identificadores.

Para poder utilizar las expresiones regulares al programar es necesario tener acceso a un motor de búsqueda con la capacidad de utilizarlas. Es posible clasificar los motores disponibles en dos tipos según su uso: motores para el programador y motores para el usuario final.

Motores para el usuario final

Son programas que permiten realizar búsquedas sobre el contenido de un archivo con un texto o expresión regular dado en el programa. Están diseñados para permitir al usuario realizar búsquedas avanzadas usando este mecanismo, sin embargo es necesario aprender a redactar expresiones regulares adecuadas para poder utilizarlos eficientemente. Algunos programas disponibles de este tipo son:

- **grep**: programa de los sistemas operativos Unix/Linux.
- **PowerGrep**: versión de grep para los sistemas operativos Windows.
- **Sublime Text**: permite realizar búsquedas/reemplazos con expresiones regulares sobre archivos (gratuito).

Motores para el programador

Permiten automatizar el proceso de búsqueda de modo que sea posible utilizarlo muchas veces para un propósito específico. Estas son algunas de las herramientas de programación disponibles que ofrecen motores de búsqueda con soporte a expresiones regulares:

C++: Desde su versión C++11 es posible utilizar expresiones regulares mediante la biblioteca estándar, usando la cabecera <regex>.

JavaScript: a partir de la versión 1.2 (ie4+, ns4+) JavaScript tiene soporte integrado para expresiones regulares.

PHP: tiene dos tipos diferentes de expresiones regulares disponibles para el programador, aunque la variante POSIX (ereg) va a ser desechada en PHP 6.

Python: lenguaje de scripting con soporte de expresiones regulares mediante su biblioteca re.

.Net Framework: provee un conjunto de clases mediante las cuales es posible utilizar expresiones regulares para hacer búsquedas, reemplazar cadenas y validar patrones.

Descripción de las expresiones regulares

Descripción de las expresiones regulares

El punto ".".

El punto se interpreta por el motor de búsqueda como "cualquier carácter", es decir, busca cualquier carácter SIN incluir los saltos de línea. Los motores de expresiones regulares tienen una opción de configuración que permite modificar este comportamiento. En .Net Framework se utiliza la opción RegexOptions.Singleline para especificar la opción de que busque todos los caracteres incluidos el salto de línea (\n).

El signo de admiración "!"

Se utiliza para realizar una "búsqueda anticipada negativa". La construcción de la expresión regular es con el par de paréntesis, el paréntesis de apertura seguido de un signo de interrogación y un signo de exclamación. Dentro de la búsqueda tenemos la expresión regular. Por ejemplo, para excluir exactamente una palabra, habrá que utilizar "^(palabra.+|(?!palabra).*)$"

Descripción de las expresiones regulares

La barra inversa o contrabarra "\"

La barra inversa se utiliza para escapar el siguiente carácter de las expresiones de búsqueda de forma que este adquiera un significado especial o deje de tenerlo. O sea, la barra inversa no se utiliza nunca por sí sola, sino en combinación con otros caracteres. Al utilizarlo por ejemplo en combinación con el punto "\." este deja de tener su significado normal y se comporta como un carácter literal.

Los corchetes "[ ]"

La función de los corchetes en el lenguaje de las expresiones regulares es representar "clases de caracteres", o sea, agrupar caracteres en grupos o clases. Son útiles cuando es necesario buscar uno de un grupo de caracteres. Dentro de los corchetes es posible utilizar el guion "-" para especificar rangos de caracteres. Adicionalmente, los metacaracteres pierden su significado y se convierten en literales cuando se encuentran dentro de los corchetes.

Descripción de las expresiones regulares

La barra "|“

Sirve para indicar una de varias opciones. Por ejemplo, la expresión regular "a|e" encontrará cualquier "a" o "e" dentro del texto. La expresión regular "este|oeste|norte|sur" permitirá encontrar cualquiera de los nombres de los puntos cardinales. La barra se utiliza comúnmente en conjunto con otros caracteres especiales.

El signo de dólar "$"

Representa el final de la cadena de caracteres o el final de la línea, si se utiliza el modo multi-línea. No representa un carácter en especial sino una posición. Si se utiliza la expresión regular "\.$" el motor encontrará todos los lugares donde un punto finalice la línea, lo que es útil para avanzar entre párrafos.

Descripción de las expresiones regulares

El acento circunflejo "^"

Este carácter tiene una doble funcionalidad y dependiendo de si se utiliza individualmente y cuando se utiliza en conjunto con otros caracteres especiales. En primer lugar su funcionalidad como carácter individual: el carácter "^" representa el inicio de la cadena (de la misma forma que el signo de dólar "$" representa el final de la cadena). Por tanto, si se utiliza la expresión regular "^[a-z]" el motor encontrará todos los párrafos que den inicio con una letra minúscula.

Los paréntesis "()“

De forma similar que los corchetes, los paréntesis sirven para agrupar caracteres, sin embargo existen varias diferencias fundamentales entre los grupos establecidos por medio de corchetes y los grupos establecidos por paréntesis:

- Los caracteres especiales conservan su significado dentro de los paréntesis.
- Utilizados en conjunto con la barra "|" permite hacer búsquedas opcionales.

Descripción de las expresiones regulares

El signo de interrogación "?"

El signo de interrogación tiene varias funciones derivadas del lenguaje de las expresiones regulares. La primera de ellas es especificar que una parte de la búsqueda es opcional. Por ejemplo, la expresión regular "ob?scuridad" permite encontrar tanto "oscuridad" como "obscuridad". En conjunto con los paréntesis redondos permite especificar que un conjunto mayor de caracteres es opcional; por ejemplo "Nov(\.|iembre|ember)?" permite encontrar tanto "Nov" como "Nov.", "Noviembre" y "November".

Las llaves "{}"

Comúnmente las llaves son caracteres literales cuando se utilizan por separado en una expresión regular. Para que adquieran su función de metacaracteres es necesario que encierren uno o varios números separados por coma y que estén colocados a la derecha de otra expresión regular de la siguiente forma: "\d{2}" Esta expresión le dice al motor de búsqueda que encuentre dos dígitos contiguos. Utilizando esta fórmula podríamos convertir el ejemplo "^\d\d/\d\d/\d\d\d\d$" que servía para validar un formato de fecha en "^\d{2}/\d{2}/\d{4}$" para una mayor claridad en la lectura de la expresión.

Descripción de las expresiones regulares

El asterisco "*“

El asterisco sirve para encontrar algo que sea precedido a repetido n veces. Por ejemplo, utilizando la expresión "[a-zA-Z]\d*" será posible encontrar tanto "H" como "H1", "H01", "H100" y "H1000", es decir, una letra seguida de un número indefinido de dígitos. Es necesario tener cuidado con el comportamiento del asterisco, ya que este, por defecto, trata de encontrar la mayor cantidad posible de caracteres que correspondan con el patrón que se busca.

El signo de suma "+“

Se utiliza para encontrar una cadena que se encuentre repetida una o más veces. A diferencia del asterisco, la expresión "[a-zA-Z]\d+" encontrará "H1" pero no encontrará "H". También es posible utilizar este metacarácter en conjunto con el signo de interrogación para limitar hasta donde se efectúa la repetición.