Expresiones regulares con el comando grep

https://es.wikipedia.org/wiki/Expresi%C3%B3n_regular

Ángel González M.

Recordatorio:
El comando grep y las tuberías

Ángel González M.

Comando grep uso básico

1- Uso de grep donde la entrada estandar es el teclado

grep an <enter>

hola soy angel

> hola soy angel

2- Uso de grep donde la entrada estandar es un fichero

grep <busqueda> fichero.txt

Buscará cosas dentro de los ficheros

Ejemplo **grep angel fichero.txt**

esto mostrará todas las lineas del fichero.txt que contengan la palabra angel

Ángel González M.

Comando grep uso con tuberías

Como ya sabes el comando ls muestra el contenido de un directorio

También sabes que el comando grep busca cosas ejemplo:

Las tuberías o pipes nos permiten redireccionar la salida de un comando a la entrada de otro, por lo tanto:

ls | grep an

Buscará todos los ficheros que contengan la palabra an

Ángel González M.

Otra forma de usar grep

cat texto.txt | grep an

Ángel González M.

Las expresiones regulares

Ángel González M.

. Significa cualquier caracter.

^Indica el principio de una línea.

$ Indica el final de una línea.

* Indica cero o más repeticiones del caracter anterior.

+ Indica una o más repeticiones del caracter anterior.

\< Indica el comienzo de una palabra.

\> Indica el final de una palabra.

\ Caracter de escape. Da significado literal a un metacaracter.

[ ] Uno cualquiera de los caracteres entre los corchetes. Ej: [A-Z] (desde A hasta Z).

[^] Cualquier caracter distinto de los que figuran entre corchetes: Ej: [^A-Z].

{ } Nos permiten indicar el número de repeticiones del patrón anterior que deben darse.

| Nos permite indicar caracteres alternativos: Ej: (^.|[?&])

( ) Nos permiten agrupar patrones. Ej: ([0-9A-F]+:)+

Ángel González M.

El punto "."

Representa cualquier caracter (un solo caracter)

El punto se interpreta por el motor de búsqueda como "cualquier carácter", es decir, busca cualquier carácter SIN incluir los saltos de línea (un caracter y solo uno).

Los motores de Expresiones regulares tienen una opción de configuración que permite modificar este comportamiento. Singleline para especificar la opción de que busque todos los caracteres incluidos el salto de línea (\n).

El punto se utiliza de la siguiente forma: Si se le dice al motor de RegEx que busque "g.t" en la cadena "el gato de piedra en la gótica puerta de getisboro goot" el motor de búsqueda encontrará "gat", "gót" y por último "get". Nótese que el motor de búsqueda no encuentra "goot"; esto es porque el punto representa un solo carácter y únicamente uno. Si es necesario que el motor encuentre también la expresión "goot", será necesario utilizar repeticiones, las cuales se explican más adelante.

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 seguida 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).*)$"

La barra inversa o antibarra "\"

Se utiliza para escapar el siguiente carácter de la expresión 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 (el punto).

De la misma forma, cuando se coloca la barra inversa seguida de cualquiera de los caracteres especiales que discutiremos a continuación, estos dejan de tener su significado especial y se convierten en caracteres de búsqueda literal.

Como ya se mencionó con anterioridad, la barra inversa también puede darle significado especial a caracteres que no lo tienen. A continuación hay una lista de algunas de estas combinaciones:

\t — Representa un tabulador.

\r — Representa el "retorno de carro" o "regreso al inicio" o sea el lugar en que la línea vuelve a iniciar.

\n — Representa la "nueva línea" el carácter por medio del cual una línea da inicio. Es necesario recordar que en Windows es necesaria una combinación de \r\n para comenzar una nueva línea, mientras que en Unix solamente se usa \n y en Mac_OS clásico se usa solamente \r.

\a — Representa una "campana" o "beep" que se produce al imprimir este carácter.

\e — Representa la tecla "Esc" o "Escape"

\f — Representa un salto de página

\v — Representa un tabulador vertical

\x — Se utiliza para representar caracteres ASCII o ANSI si conoce su código. De esta forma, si se busca el símbolo de derechos de autor y la fuente en la que se busca utiliza el conjunto de caracteres Latin-1 es posible encontrarlo utilizando "\xA9". Ejemplo echo -e "\xE2\x99\xA5"

\u — Se utiliza para representar caracteres Unicode si se conoce su código. "\u00A2" representa el símbolo de centavos. No todos los motores de Expresiones Regulares soportan Unicode. El .Net Framework lo hace, pero el EditPad Pro no, por ejemplo.

\d — Representa un dígito del 0 al 9.

\w — Representa cualquier carácter alfanumérico.

\s — Representa un espacio en blanco.

\D — Representa cualquier carácter que no sea un dígito del 0 al 9.

\W — Representa cualquier carácter no alfanumérico.

\S — Representa cualquier carácter que no sea un espacio en blanco.

\A — Representa el inicio de la cadena. No un carácter sino una posición.

\Z — Representa el final de la cadena. No un carácter sino una posición.

\b — Marca la posición de una palabra limitada por espacios en blanco, puntuación o el inicio/final de una cadena.

NOTA: Para usar ciertos símbolos en grep hay que usar el modificador -P
> grep -p '\d$'

Ángel González M.

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. Por ejemplo, como vimos en la entrega anterior "\d" nos es útil para buscar cualquier carácter que represente un dígito. Sin embargo esta denominación no incluye el punto "." que divide la parte decimal de un número. Para buscar cualquier carácter que representa un dígito o un punto podemos utilizar la expresión regular "[\d.]". Como se hizo notar anteriormente, dentro de los corchetes, el punto representa un carácter literal y no un metacarácter, por lo que no es necesario antecederlo con la barra inversa. El único carácter que es necesario anteceder con la barra inversa dentro de los corchetes es la propia barra inversa. La expresión regular "[\dA-Fa-f]" nos permite encontrar dígitos hexadecimales. Los corchetes nos permiten también encontrar palabras aún si están escritas de forma errónea, por ejemplo, la expresión regular "expresi[oó]n" permite encontrar en un texto la palabra "expresión" aunque se haya escrito con o sin tilde. Es necesario aclarar que sin importar cuantos caracteres se introduzcan dentro del grupo por medio de los corchetes, el grupo sólo le dice al motor de búsqueda que encuentre un solo carácter a la vez, es decir, que "expresi[oó]n" no encontrará "expresion" o "expresión"

Ángel González M.

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.

El acento circunflexo "^"

Este carácter tiene una doble funcionalidad, que difiere cuando 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. Cuando se utiliza en conjunto con los corchetes de la siguiente forma "[^\w]" permite encontrar cualquier carácter que NO se encuentre dentro del grupo indicado. La expresión indicada permite encontrar, por ejemplo, cualquier carácter que no sea alfanumérico o un espacio, es decir, busca todos los símbolos de puntuación y demás caracteres especiales.

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.
- Los grupos establecidos con paréntesis establecen una "etiqueta" o "punto de referencia" para el motor de búsqueda que puede ser utilizada posteriormente como se denota más adelante.
- Utilizados en conjunto con la barra "|" permite hacer búsquedas opcionales. Por ejemplo la expresión regular "al (este|oeste|norte|sur) de" permite buscar textos que den indicaciones por medio de puntos cardinales, mientras que la expresión regular "este|oeste|norte|sur" encontraría "este" en la palabra "esteban", no pudiendo cumplir con este propósito.
- Utilizados en conjunto con otros caracteres especiales que se detallan posteriormente, ofrece funcionalidad adicional.

El signo de interrogación "?"

El signo de interrogación tiene varias funciones dentro 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". Como se mencionó anteriormente, los paréntesis nos permiten establecer un "punto de referencia" para el motor de búsqueda. Sin embargo, algunas veces, no se desea utilizarlos con este propósito, como en el ejemplo anterior "Nov(\.|iembre|ember)?". En este caso el establecimiento de este punto de referencia (que se detalla más adelante) representa una inversión inútil de recursos por parte del motor de búsqueda. Para evitarlo se puede utilizar el signo de pregunta de la siguiente forma: "Nov(?:\.|iembre|ember)?". Aunque el resultado obtenido será el mismo, el motor de búsqueda no realizará una inversión inútil de recursos en este grupo, sino que lo ignorará. Cuando no sea necesario reutilizar el grupo, es aconsejable utilizar este formato. De forma similar, es posible utilizar el signo de pregunta con otro significado: Los paréntesis definen grupos "anónimos", sin embargo el signo de pregunta en conjunto con los paréntesis triangulares "<>" permite "nombrar" estos grupos de la siguiente forma: "^(?<Día>\d\d)\/(?<Mes>\d\d)\/(?<Año>\d\d\d\d)$"; Con lo cual se le especifica al motor de búsqueda que los primeros dos dígitos encontrados llevarán la etiqueta "Día", los segundos la etiqueta "Mes" y los últimos cuatro dígitos llevarán la etiqueta "Año".

Ángel González M.

El asterisco "*"

El asterisco sirve para encontrar algo que se encuentra repetido 0 o más 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 éste, por defecto, trata de encontrar la mayor cantidad posible de caracteres que correspondan con el patrón que se busca. De esta forma si se utiliza "\(.*\)" para encontrar cualquier cadena que se encuentre entre paréntesis y se lo aplica sobre el texto "Ver (Fig. 1) y (Fig. 2)" se esperaría que el motor de búsqueda encuentre los textos "(Fig. 1)" y "(Fig. 2)", sin embargo, debido a esta característica, en su lugar encontrará el texto "(Fig. 1) y (Fig. 2)". Esto sucede porque el asterisco le dice al motor de búsqueda que llene todos los espacios posibles entre los dos paréntesis. Para obtener el resultado deseado se debe utilizar el asterisco en conjunto con el signo de interrogación de la siguiente forma: "\(.*?\)" Esto es equivalente a decirle al motor de búsqueda que "Encuentre un paréntesis de apertura y luego encuentre cualquier secuencia de caracteres hasta que encuentre un paréntesis de cierre".

Ángel González M.

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.

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.

"\d{2,4}" Esta forma añade un segundo número separado por una coma, el cuál indica al motor de búsqueda que como máximo debe aparecer 4 veces la expresión regular \d. Los posibles valores son:

- "^\d\d$" (mínimo 2 repeticiones)
- "^\d\d\d$"(tiene 3 repeticiones, por lo tanto entra en el rango 2-4)
- "^\d\d\d\d$" (máximo 4 repeticiones)

Webs para probar expresiones regulares online

- https://regex101.com/
- https://regexr.com/

Preparando los ejemplos: uso de grep y expresiones regulares

Copia el poema El cuervo de Edgar Allan Poe de la siguiente página dentro de un fichero de texto llamado cuervo.txt.

http://ciudadseva.com/texto/el-cuervo/

Sitúa este fichero dentro de una carpeta llamada regex1

También puedes descargar el fichero de esta forma:

$ wget https://raw.githubusercontent.com/kant003/MemesInformatica/master/cuervo.txt

Cuidado distingue entre mayúsculas y minúsculas

Mostrar las líneas que contengan en algún lugar las letras **Es**

#grep 'Es' cuervo.txt

No es lo mismo que:

#grep 'es' cuervo.txt

El comando anterior nos devuelve todas las líneas del fichero que comienzan por La.

 # grep '^La' cuervo.txt

Que terminen en "do." (termina en punto)

grep 'do\.$' cuervo.txt

~~grep 'do.$' cuervo.txt~~

Que tenga una vocal seguida de l

grep '[aeiouAEIOU]l' cuervo.txt

Qué comience por cualquier cosa y termine por un punto.

 # grep '^.*\.$' cuervo.txt

Que empiecen por una a o una e

#grep '^[ae]' cuervo.txt

El comando anterior nos devuelve la lista de ficheros que comienzan por un espacio seguido de un punto y cualquier número de caracteres, es decir, la lista de ficheros ocultos.

 # ls -la | grep ' \..*'

El comando anterior nos devuelve la lista de ficheros que comienzan por d, es decir, la lista de directorios.

 # ls -l | grep '^d'