Sintaxis de expresiones regulares

Clases de carácteres

Caracteres	Significado
.	Tiene uno de los siguientes significados: • Encuentra cualquier carácter único *excepto* terminadores de línea: \n, \r, \u2028 o \u2029. Por ejemplo, /.y/ reconoce "my" y "ay", pero no "yes", en "yes make my day". • Dentro de un juego de caracteres, el punto pierde su significado especial y concuerda con un punto literal. El indicador multilínea m no cambia el comportamiento del punto.
\d	Busca cualquier dígito numérico. Equivalente a [0-9]. Por ejemplo, /\d/ o /[0-9]/ encuentra el "2" en "B2 es el número de suite".
\D	Busca cualquier carácter que no sea un dígito numérico. Equivalente a [^0-9]. Por ejemplo, /\D/ o /[^0-9]/ encuentra la "B" en "B2 es el número de suite".
\w	Busca cualquier carácter alfanumérico del alfabeto latino básico, incluido el carácter de subrayado. Equivalente a [A-Za-z0-9_]. Por ejemplo, /\w/ encuentra la "m" en "manzana", el "5" en "$5.28" y el "3" en "3D".
\W	Busca cualquier carácter que no sea un carácter de palabra del alfabeto latino básico. Equivalente a [^A-Za-z0-9_]. Por ejemplo, /\W/ o /[^A-Za-z0-9_]/ encuentra el carácter "%" en "50%".
\s	Busca un solo carácter de espacio en blanco, incluido el espacio, tabulación, avance de página, avance de línea y otros espacios Unicode. Equivalente a [\f\n\r\t\v\u00a0\u1680\u2000-\u200a\u2028\u2029\u202f\u205f\u3000\ufeff]. Por ejemplo, /\s\w*/ reconoce " bar" en "foo bar".
\S	Busca un solo carácter que no sea un espacio en blanco. Equivalente a [^ \f\n\r\t\v\u00a0\u1680\u2000-\u200a\u2028\u2029\u202f\u205f\u3000\ufeff]. Por ejemplo, /\S\w*/ encuentra "foo" en "foo bar".
\t	Coincide con una tabulación horizontal.
\r	Coincide con un retorno de carro.
\n	Coincide con un salto de línea.
\v	Coincide con una tabulación vertical.
\f	Coincide con un carácter de avance de página.
[\b]	Coincide con un carácter de retroceso.
\0	Coincide con un carácter NUL. No sigue a este con otro dígito.
\c*X*	Coincide con un carácter de control usando notación de acento circunflejo, donde "X" es una letra de la A a la Z (correspondiente a los puntos de código U+0001-U+001F). Por ejemplo, /\cM/ reconoce el carácter "\r" en "\r\n".
\x*hh*	Busca el carácter con el código *hh* (dos dígitos hexadecimales).
\u*hhhh*	Busca una unidad de código UTF-16 con el valor *hhhh* (cuatro dígitos hexadecimales).
\u*{hhhh}* o \u*{hhhhh}*	(Solo cuando se establece el indicador u). Busca el carácter con el valor Unicode U+*hhhh* o U+*hhhhh* (dígitos hexadecimales).
\	Indica que el siguiente carácter se debe tratar de manera especial o "escaparse". Se comporta de dos formas. • Para los caracteres que generalmente se tratan literalmente, indica que el siguiente carácter es especial y no se debe interpretar literalmente. Por

Caracteres	Significado
	ejemplo, /b/ reconoce el carácter "b". Al colocar una barra invertida delante de "b", es decir, usando /\b/, el carácter se vuelve especial para significar que concuerda con el límite de una palabra. • Para los caracteres que generalmente se tratan de manera especial, indica que el siguiente carácter no es especial y se debe interpretar literalmente. Por ejemplo, "*" es un carácter especial que significa que deben reconocer 0 o más ocurrencias del carácter anterior; por ejemplo, /a*/ significa reconocer 0 o más "a"s. Para emparejar el * literal, precédelo con una barra invertida; por ejemplo, /a*/ concuerda con "a*". Algunos caracteres como :, -, @, etc. no tienen un significado especial cuando se escapan ni cuando no se escapan. En las expresiones regulares con indicador Unicode, esto provocará un error de *escape de identidad no válido*. Para reconocer este carácter literalmente, escápalo consigo mismo. En otras palabras, para buscar \ usa /\\/.

Aserciones

Aserciones de tipo límite

Caracteres	Significado
^	Coincide con el comienzo de la entrada. Si el indicador multilínea se establece en true, también busca inmediatamente después de un carácter de salto de línea. Por ejemplo, /^A/ no reconoce la "A" en "an A", pero encuentra la primera "A" en "An A". Este carácter tiene un significado diferente cuando aparece al comienzo de un grupo.
$	Coincide con el final de la entrada. Si el indicador multilínea se establece en true, también busca hasta inmediatamente antes de un carácter de salto de línea. Por ejemplo, /a$/ no reconoce la "t" en "eater", pero sí en "eat".
\b	Marca el límite de una palabra. Esta es la posición en la que un carácter de palabra no va seguido o precedido por otro carácter de palabra, por ejemplo, entre una letra y un espacio. El límite de una palabra encontrada no se incluye en el resultado, es decir, la longitud de un límite de palabra encontrada es cero. Ejemplos: • /\bm/ reconoce la "m" en "moon". • /oo\b/ no reconoce "oo" en "moon", porque "oo" va seguido de "n", que es un carácter de palabra. • /oon\b/ encuentra "oon" en "moon", porque "oon" es el final de la cadena, por lo que no va seguido de un carácter de palabra. • /\w\b\w/ nunca encontrará nada, porque un carácter de palabra nunca puede ir seguido de un carácter que no sea de palabra y otro de palabra.
\B	Coincide con un límite sin palabra. Esta es una posición en la que el carácter anterior y siguiente son del mismo tipo: ambos deben ser palabras o ambos deben ser no palabras, por ejemplo, entre dos letras o entre dos espacios. El principio y el final de una cadena se consideran no palabras. Igual que el límite de palabras encontradas, el límite sin palabras reconocidas tampoco se incluye en el resultado. Por ejemplo, /\Bon/ reconoce "on" en "at noon", y /ye\B/ encuentra "ye" en "possibly yesterday".

Otras aserciones

Caracteres	Significado
x(?=y)	**Aserción anticipada**: Coincide con "x" solo si "x" va seguida de "y". Por ejemplo, /Jack(?=Sprat)/ reconocerá a "Jack" solo si va seguida de "Sprat". /Jack(?=Sprat|Frost)/ encontrará a "Jack" solo si va seguida de "Sprat" o "Frost". Sin embargo, ni "Sprat" ni "Frost" forman parte del resultado.
x(?!y)	**Aserción de búsqueda anticipada negativa**: reconoce la "x" solo si la "x" no va seguida de "y". Por ejemplo, /\d+(?!\.)/ reconoce un número solo si no va seguido de un punto decimal. /\d+(?!\.)/.exec('3.141') halla el "141" pero no el "3".
(?<=y)x	**Aserción de búsqueda inversa**: encontrará "x" solo si "x" está precedida por "y". Por ejemplo, /(?<=Jack)Sprat/ reconoce a "Sprat" solo si está precedido por "Jack". /(?<=Jack|Tom)Sprat/ empareja "Sprat" solo si está precedido por "Jack" o "Tom". Sin embargo, ni "Jack" ni "Tom" forman parte del resultado.
(?<!y)x	**Aserción de búsqueda inversa negativa**: Reconoce la "x" solo si "x" no está precedida por "y". Por ejemplo, /(?<!-)\d+/ encuentra un número solo si no está precedido por un signo menos. /(?<!-)\d+/.exec('3') encuentra el "3". /(?<!-)\d+/.exec('-3') no lo reconoce porque el número está precedido por el signo menos.

Grupos y rangos

Caracteres	Significado
x|*y*	Coincide con "x" o "y". Por ejemplo, /verde|roja/ reconoce el "verde" en "manzana verde" y "roja" en "manzana roja".
[xyz] [a-c]	Coincide con cualquiera de los caracteres incluidos. Se puede especificar un rango de caracteres mediante el uso de un guion, pero si el guion aparece como el primero o último carácter entre corchetes, se toma como un guion literal para incluirse en el juego de caracteres como un carácter normal. También es posible incluir una clase de caracteres en un juego de caracteres. Por ejemplo, • [abcd] es lo mismo que [a-d]. Coincide con la "b" en "brisket" y la "c" en "chop". • [abcd-] y [-abcd] reconoce la "b" en "brisket", la "c" en "chop" y el "-" (guión) en "non-profit". • [\w-] es lo mismo que [A-Za-z0-9_-]. Ambos reconocen la "b" en "brisket", la "c" en "chop" y la "n" en "non-profit".
[^xyz] [^a-c]	Un juego de caracteres negado, cualquier cosa que no esté encerrada entre corchetes. Se puede especificar un rango de caracteres mediante el uso de un guion, pero si el guion aparece como el primero o último carácter entre corchetes, se toma como un guion literal para incluirse en el juego de caracteres como un carácter normal. Por ejemplo, [^abc] es lo mismo que [^a-c]. Inicialmente halla la "o" en "bacon" y la "h" en "chuleta". El carácter ^ además puede indicar el comienzo de la entrada.
(*x*)	**Grupo de captura**: Encuentra la *x* y la recuerda. Por ejemplo, /(foo)/ encuentra y recuerda "foo" en "foo bar". Una expresión regular puede tener varios grupos de captura. En los resultados, coincide con los grupos capturados normalmente en un arreglo cuyos miembros están en el mismo orden que los paréntesis de la izquierda en el grupo capturado. Este suele ser solo el orden de los propios grupos capturados. Esto se vuelve importante cuando los grupos capturados están anidados. Se accede a las coincidencias utilizando el

Caracteres	Significado
	índice de los elementos del resultado ([1], ..., [n]) o desde las propiedades predefinidas del objeto RegExp ($1, ..., $9). Los grupos de captura tienen una penalización de rendimiento. Si no necesitas que se recupere la subcadena coincidente, prefiere los paréntesis que no capturen (ve más abajo). String.match() no devolverá grupos si el indicador /.../g está configurado. Sin embargo, aún puedes usar String.matchAll() para obtener todas los encontrados.
\n	Donde "n" es un número entero positivo. Una referencia posterior a la última subcadena que coincide con el paréntesis n en la expresión regular (contando los paréntesis izquierdos). Por ejemplo, /apple(,)\sorange\1/ coincide con "apple, orange" en "apple, orange, cherry, peach".
\k<Name>	Una referencia inversa a la última subcadena encontrada con el grupo de captura **Nombrado** especificado por <Name>. Por ejemplo, /(?<title>\w+), yes \k<title>/ concuerda con "Sir, yes Sir" en "Do you copy? Sir, yes Sir!". \k aquí se usa literalmente para indicar el comienzo de una referencia a un grupo de captura nombrado.
(?<Name>x)	**Grupo de captura nombrado**: reconoce la "x" y la almacena en la propiedad group del resultado devuelto bajo el nombre especificado por <Name>. Los corchetes angulares (< y >) son obligatorios para el nombre del grupo. Por ejemplo, para extraer el código de área de Estados Unidos de un número de teléfono, podríamos usar /\((?<area>\d\d\d)\)/. El número resultante debería aparecer en matches.groups.area.
(?:*x*)	**Grupo sin captura**: reconoce la "x" pero no recuerda el resultado. La subcadena encontrada no se puede recuperar de los elementos del arreglo resultante ([1], ..., [n]) o de las propiedades predefinidas del objeto RegExp ($1, ..., $9).

Cuantificadores

Caracteres	Significado
*x**	Concuerda 0 o más veces con el elemento "x" anterior. Por ejemplo, /bo*/ reconoce a "boooo" en "Un fantasma booooed" y "b" en "A bird warbled", pero nada en "Una cabra gruñó".
x+	Encuentra 1 o más veces el elemento "x" anterior Equivalente a {1,}. Por ejemplo, /a+/ encuentra la "a" en "candy" y todas las "a"es en "caaaaaaandy".
	Halla 0 o 1 vez el elemento "x" anterior. Por ejemplo, /e?Le?/ reconoce a "el" en "ángel" y a "le" en "angle".
x?	Si se usa inmediatamente después de cualquiera de los cuantificadores *, +, ?o {}, hace que el cuantificador no codicioso (que reconoce el número mínimo de veces), a diferencia del predeterminado, que es codicioso (que reconoce el número máximo de veces).
x{*n*}	Donde "n" es un número entero positivo, concuerda exactamente con "n" apariciones del elemento "x" anterior. Por ejemplo, /a{2}/ no reconoce la "a" en "candy", pero reconoce todas las "a"s en "caandy" y las dos primeras "a"s en "caaandy ".
x{*n*,}	Donde "n" es un número entero positivo, concuerda con al menos "n" apariciones del elemento "x". Por ejemplo, /a{2,}/ no reconoce la "a" en "candy", pero reconoce todas

Caracteres	Significado
	las "a" en "caandy" y en "caaaaaaandy".
x{*n*,*m*}	Donde "n" es 0 o un número entero positivo, "m" es un número entero positivo y *m* > *n*, reconoce por lo menos "n" y como máximo "m" apariciones del elemento "x" anterior. Por ejemplo, /a{1,3}/ no reconoce nada en "cndy", la "a" en "caramelo", las dos "a" en "caandy" y las tres primeras "a" está en "caaaaaaandy". Observa que al comparar "caaaaaaandy", las "aaa" encontradas, aunque la cadena original tenía más "a" s.
*x**? *x*+? *x*?? *x*{n}? *x*{n,}? *x*{n,m}?	De manera predeterminada, los cuantificadores como * y + son "codiciosos", lo cual significa que intentan hacer coincidir la mayor cantidad de cadena posible. El carácter ? después del cuantificador hace que este sea "no codicioso": lo cual significa que se detendrá tan pronto como encuentre una concordancia. Por ejemplo, dada una cadena "algo como <foo> <bar> new </bar> </foo>": • /<.*>/ reconocerá "<foo> <bar> nuevo </bar> </foo>" • /<.*?>/ encajará "<foo>"