

# w3.unpo<code>todo

PUBLICIDAD:

&amp;NBSP;

## RegEx (Expresiones Regulares)



**No te olvides:** por defecto las expresiones regulares son codiciosas (*greedy*). Esto significa que una expresión regular devuelve la cadena de texto más larga que coincida con ella. Las expresiones regulares son también ansiosas (*eager*) por devolver un resultado.

Para verificar tus regEx utiliza [RegexPal](#)

### Modificadores

Modificadores	Descripción
i	Insensible a las mayúsculas y minúsculas <a href="#">W</a> ( <i>case insensitive</i> )
g	Busqueda global ( <i>global match</i> )
m	Busqueda en multiples líneas de texto. ( <i>Multiple lines</i> )
s	Incluye saltos de línea. Sin él, las nuevas líneas son excluidas.

### Clases de Carácteres (Ranges)

Expression	Descripción
[abc]	Encuentra uno de los caracteres entre corchetes
[^abc]	Encuentra cualquier carácter que <b>NO</b> esté entre corchetes
[0-9]	Encuentra un dígito de <b>0</b> a <b>9</b>
[^0-9]	Encuentra cualquier carácter que <b>NO</b> sea un dígito de 0 a 9
[A-Z]	Encuentra cualquier carácter de <b>A</b> mayuscula a <b>Z</b> mayuscula

[A-z]	Encuentra cualquier carácter de <b>A</b> mayúscula a <b>z</b> minúscula
[adgk]	Encuentra uno de los caracteres entre corchetes
[^adgk]	Encuentra cualquier carácter que <b>NO</b> esté entre corchetes
(a b)	<b>a</b> o <b>b</b>
(...)	Se utilizan para agrupar partes de una expresión.

## Cuantificadores

El metacarácter **?** (detrás de otro metacarácter) hace que una expresión regular, habitualmente codiciosa (*greedy*), se convierta en perezosa (*lazy*), y resulte en la cadena más corta posible que coincida con ella.

Greedy	Lazy	Descripción
*	*?	<b>0</b> o <b>más</b> veces
+	+?	<b>1</b> o <b>más</b> veces
?	??	<b>0</b> o <b>1</b> veces
{ n }	{ n }?	<b>n</b> veces
{ n , }	{ n , }?	<b>n</b> o <b>más</b> veces
{ n , m }	{ n , m }?	De <b>n</b> a <b>m</b> veces

## Anclas

Metacarácter	Descripción
^	Comienzo de una línea
\$	Final de una línea
\A	Comienzo de una <u>cadena de texto</u> ( <i>string</i> ). Nunca final de línea. (Soporte: Java, .NET, Perl, PHP, Python, Ruby)
\Z	Fin de <u>cadena de texto</u> ( <i>string</i> ). Nunca final de línea. (Soporte: Java, .NET, Perl, PHP, Python, Ruby)
\b	<u>Principio o final de palabra</u> ( <i>Word boundary</i> )
\B	<b>NO</b> al principio o al final de una palabra



## Retroreferencias (*backreference*)

Para designar una retroreferencia (*backreference*), a veces utilizamos la barra inversa ( **\** ), otras veces el dólar ( **\$** ), dependiendo del lenguaje que se utilice.

\$2	\2	"xyz" en /^ (abc)(xyz) \$/
\$2	\2	"xyz" en /^ (abc(xyz)) \$/
?		cambia el significado del grupo
:		el significado del grupo es: <b>pasivo</b>
?:		especifica un <u>grupo pasivo</u> ( <i>passive group / non-capturing group</i> )
\$1	\1	"xyz" en /^ (? :abc)(xyz) \$/ ( <i>porque el primer grupo es pasivo</i> )

Más sobre retroreferencias [↗](#)

**POSIX**  
(*Portable Operating System Interface*)

Manera correcta de utilizarlos:

`[[:alpha:]]` or `[^[:alpha:]]`.

**Soporte:**

**SI:** Perl, PHP, Ruby, Unix

**NO:** Java, JavaScript, .NET, Python

POSIX	Descripción
[[:alpha:]]	Caracteres alfabeticos [a-zA-Z]
[[:digit:]]	Dígitos [0-9]
[[:alnum:]]	Caracteres alfanuméricos [a-zA-Z0-9]
[[:lower:]]	Letras minúsculas [a-z]
[[:upper:]]	Letras mayúsculas [A-Z]
[[:word:]]	Letras, números y el guion bajo [A-Za-z0-9_]
[[:punct:]]	Puntuacion y símbolos. [!"#\$%&'()*+,-./:;<=>@[\\]^_`{ }~]
[[:space:]]	Espacios en blanco, incluido   [ \t\r\n\v\f]
[[:blank:]]	Espacio y tabulador [ \t]

[.graph:]	Caracteres visibles (i.e: excepto espacios, caracteres de control, etc.) [\x21-\x7E]
[.xdigit:]	Dígitos hexadecimales [A-Fa-f0-9]
[.ascii:]	Caracteres ASCII [\x00-\x7F]
[.cntrl:]	Caracteres de control [\x00-\x1F\x7F]

Metacaracteres basicos

Los metacaracteres deben ser escapados cuando se utilicen como caracteres normales.

Metacarácter	Descripción
.	Cualquier carácter excepto salto de linea
*	Indica que el carácter precedente puede ocurri <b>0</b> o <b>más</b> veces.
+	Indica que el carácter precedente puede ocurri <b>1</b> o <b>más</b> veces.
?	Indica que el carácter precedente puede ocurrir <b>0</b> o <b>1</b> vez. Hace que el metacaracter anterior sea <u>perezoso</u> ( <i>lazy</i> ).
[	Abre un set de caracteres
]	Cierra un set de caracteres
^	Niega un set de caracteres ( como en [^0-9] )
-	Define un set de caracteres ( de 0 a 9: [0-9] )
{	Comienza la repetición cuantificada del carácter precedente {min,max}
}	Acaba la repetición cuantificada del carácter precedente {min,max}
(	El comienzo de un grupo de caracteres.
)	El fin de un grupo de caracteres.
	<b>Altern</b> a entre uno y otro carácter
\	El carácter de escape ( <i>Escape character</i> )



Characters específicos

Carácter	Descripción	Equivalente
\w	Encuentra un carácter alfanumérico, incluido el guión bajo ( _ )	[a-z A-Z0-9_]

\W	Encuentra cualquier carácter <b>NO</b> alfanumérico	[ a-zA-Z0-9_]
\d	Encuentra un dígito	[0-9]
\D	Encuentra cualquier carácter que <b>NO</b> es un dígito.	[^0-9]
\s	Encuentra un espacio en blanco	[\t\r\n]
\S	Encuentra cualquier carácter que <b>NO</b> es un espacio en blanco.	[^\t\r\n]
\b	Encuentra una coincidencia al inicio o al final de una palabra.	
\B	Encuentra una coincidencia que <b>NO</b> está al inicio o al final de una palabra.	
\0	Encuentra un carácter NUL	
\n	Salto de línea ( <i>new line</i> )	
\f	Salto de página ( <i>feed</i> )	
\r	Retorno de carro ( <i>return</i> )	
\t	Tabulador	
\v	Tabulador vertical	
\xxx	Representa un carácter especificado por un número <b>octal</b> xxx	
\xdd	Representa un carácter especificado por un número <b>hexadecimal</b> dd	
\uxxxx	Representa un carácter <b>Unicode</b> especificado por un número hexadecimal xxxx	

## Declaraciones (Assertions)

	Descripción
?=	declaración positiva de búsqueda hacia delante ( <i>positive lookahead assertion</i> )
	<b>/(?=premature)pre/</b> encuentra <b>pre</b> de premature pero no pre de precavido
	<b>/pre(=mature)/</b> encuentra <b>pre</b> de premature pero no pre de precavido
?!	declaración negativa de búsqueda hacia delante ( <i>negative lookahead assertion</i> )
	<b>/(!premature)pre/</b> encuentra <b>pre</b> pero <b>no</b> de premature
	<b>/pre(!mature)/</b> encuentra <b>pre</b> pero <b>no</b> de premature
?<=	declaración positiva de búsqueda hacia atrás ( <i>positivee lookbehind assertion</i> )



	<del>...de antemano, pero no de</del>
	<b>/(?&lt;=balon)mano/</b> encuentra <b>-mano</b> de balonmano pero no de antemano
?<!	declaración negativa de búsqueda hacia atras (negative lookbehind assertion) <b>Soporte:</b> <b>Si:</b> .NET, Java, Perl, PHP, Python, Ruby 1.9 <b>No:</b> JavaScript, Ruby 1.8, Unix
	<b>/(?!balon)mano/</b> encuentra <b>-mano</b> pero <b>no</b> de balonmano

Más sobre declaraciones [↗](#)


Comodines UNICODE

**Soporte:**  
**Si:** Java, .NET, Perl, PHP, Ruby  
**No:** JavaScript, Python, Unix

	Descripción
\X	Carácter comodín Unicode. Encuentra cualquier carácter incluso salto de línea.
\p{xx}	Un carácter <b>con</b> la propiedad xx
\P{xx}	Un carácter <b>sin</b> la propiedad xx
M	<b>\p{M}</b> Marca (acentos, tilde...etc)
L	<b>\p{L}</b> Letra (incluye las letras accentuadas, la ñ...etc)
N	<b>\p{N}</b> Numero
S	<b>\p{S}</b> Símbolo
C	<b>\p{C}</b> Otros

Para más Información:  
[Propiedades de los caracteres Unicode ↗](#)

RegEx útiles

	Descripción	
(\bnegro\b)(?!.*\1)	ultima ocurencia de "negro" en un texto	
\b\w+\b(?:\.\.?)	palabras seguidas de un punto ( . ), pero no el punto.	
^#(?:[0-9a-fA-F]{3}){1,2}\$	un color hex	

## Artículos relacionados

- *regEx - la chuleta*
- *RegEx - una introducción* [↗](#)
- *RegEx en JavaScript* [↗](#)
- *regEx para fechas* [↗](#)
- *regEx para horas* [↗](#)
- *regEx para emails* [↗](#)
- *regEx para contraseñas* [↗](#)
- *regEx para IPs y URLs* [↗](#)
- *regEx para tarjetas de crédito* [↗](#)

## Enlaces útiles

- Para verificar tus regEx: *RegexPal* [↗](#)
- *PHP.net: Propiedades de los caracteres Unicode* [↗](#)
- Más sobre *retroreferencias RegEx* [↗](#) (*backreferences*)
- Más sobre *declaraciones RegEx* [↗](#) (*assertions*)

PUBLICIDAD:

w3.unpo<code>todo.info utiliza una estructura generada con foundation

