

Ejemplos de Expresiones Regulares en Java

Una expresión regular define un patrón de búsqueda para cadenas de caracteres.

La podemos utilizar para comprobar si una cadena contiene o coincide con el patrón. El contenido de la cadena de caracteres puede coincidir con el patrón 0, 1 o más veces.

Algunos ejemplos de uso de expresiones regulares pueden ser:

- para comprobar que la fecha leída cumple el patrón dd/mm/aaaa
- para comprobar que un NIF está formado por 8 cifras, un guión y una letra
- para comprobar que una dirección de correo electrónico es una dirección válida.
- para comprobar que una contraseña cumple unas determinadas condiciones.
- Para comprobar que una URL es válida.
- Para comprobar cuántas veces se repite dentro de la cadena una secuencia de caracteres determinada.
- Etc. Etc.

El patrón se busca en el String de izquierda a derecha. Cuando se determina que un carácter cumple con el patrón este carácter ya no vuelve a intervenir en la comprobación.

Ejemplo:

La expresión regular "01010" la encontraremos dentro del String "010101010" solo dos veces: "010101010"

Símbolos comunes en expresiones regulares

Expresión	Descripción
.	Un punto indica cualquier carácter
^expresión	El símbolo ^ indica el principio del String. En este caso el String debe contener la expresión al principio.
expresión\$	El símbolo \$ indica el final del String. En este caso el String debe contener la expresión al final.
[abc]	Los corchetes representan una definición de conjunto. En este ejemplo el String debe contener las letras a ó b ó c.
[abc][12]	El String debe contener las letras a ó b ó c seguidas de 1 ó 2
[^abc]	El símbolo ^ dentro de los corchetes indica negación. En este caso el String debe contener cualquier carácter excepto a ó b ó c.
[a-z1-9]	Rango. Indica las letras minúsculas desde la a hasta la z (ambas incluidas) y los dígitos desde el 1 hasta el 9 (ambos incluidos)
A B	El carácter es un OR. A ó B
AB	Concatenación. A seguida de B

Meta caracteres

Expresión	Descripción
\d	Dígito. Equivale a [0-9]
\D	No dígito. Equivale a [^0-9]
\s	Espacio en blanco. Equivale a [\t\n\r\b\f]
\S	No espacio en blanco. Equivale a [^\s]
\w	Una letra mayúscula o minúscula, un dígito o el carácter _ Equivale a [a-zA-Z0-9_]
\W	Equivale a [^\w]
\b	Límite de una palabra.

En Java debemos usar una doble barra invertida \

Por ejemplo para utilizar \w tendremos que escribir \\w. Si queremos indicar que la barra invertida en un carácter de la expresión regular tendremos que escribir \\\w.

Cuantificadores

Expresión	Descripción
{X}	Indica que lo que va justo antes de las llaves se repite X veces
{X,Y}	Indica que lo que va justo antes de las llaves se repite mínimo X veces y máximo Y veces. También podemos poner {X,} indicando que se repite un mínimo de X veces sin límite máximo.
*	Indica 0 ó más veces. Equivale a {0,}
+	Indica 1 ó más veces. Equivale a {1,}
?	Indica 0 ó 1 veces. Equivale a {0,1}

Para usar expresiones regulares en Java se usa el package **java.util.regex**

Contiene las clases **Pattern** y **Matcher** y la excepción **PatternSyntaxException**.

Clase **Pattern**: Un objeto de esta clase representa la expresión regular. Contiene el método **compile(String regex)** que recibe como parámetro la expresión regular y devuelve un objeto de la clase Pattern.

La clase **Matcher**: Esta clase compara el String y la expresión regular. Contienen el método **matches(CharSequence input)** que recibe como parámetro el String a validar y devuelve true si coincide con el patrón. El método **find()** indica si el String contienen el patrón.

Ejemplos de Expresiones Regulares en Java:

1. Comprobar si el String *cadena* contiene exactamente el patrón (matches) "abc"

```
Pattern pat = Pattern.compile("abc");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("Si");
} else {
    System.out.println("No");
}
```

2. Comprobar si el String *cadena* contiene "abc"

```
Pattern pat = Pattern.compile(".*abc.*");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("Si");
} else {
    System.out.println("No");
}
```

También lo podemos escribir usando el método find:

```
Pattern pat = Pattern.compile("abc");
Matcher mat = pat.matcher(cadena);
if (mat.find()) {
    System.out.println("Válido");
} else {
    System.out.println("No Válido");
}
```

3. Comprobar si el String *cadena* empieza por "abc"

```
Pattern pat = Pattern.compile("^abc.*");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("Válido");
} else {
    System.out.println("No Válido");
}
```

4. Comprobar si el String *cadena* empieza por "abc" ó "Abc"

ENTRADAS POPULARES

2	Matrices en Java
3	Actualizado 2022
6	Un array en Java
9	puede tener más de una dimensión.
	El caso más general son los arrays bidimensionales también llamados m...

Java Ejercicios Básicos Resueltos 1

Relación Nº 1: Ejercicios 1, 2 y 3 Empezaremos por unos ejercicios básicos de programas Java con estructura secuencial, es decir, en es...

Java Scanner para lectura de datos

Actualizado 2022 La clase Scanner está disponible a partir de Java 5 y facilita la lectura de datos en los programas Java. Primero vere...

Java Ejercicios Básicos Resueltos 2

Java Ejercicios Básicos estructura secuencial Actualizado 2020 Relación Nº 2: Ejercicios 4, 5, 6 y 7 Ejercicio 4: Programa que lea una ca...

Java Ejercicios Básicos de Arrays Resueltos 1

Relación Nº 1: Ejercicios 1 y 2 1. Calcular la media de una serie de números que se leen por teclado. Programa Java que lea por teclado ...

Algoritmos de ordenación. Metodo de la Burbuja

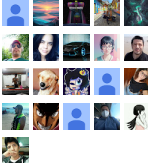
Actualizado 2020 ALGORITMO DE LA BURBUJA El algoritmo de la burbuja es uno de los métodos de ordenación más conocidos y uno de los primeros que aprenden l...



Programación Java Enrique García Hernández

SEGUIDORES

Abonnés (264) Suiv



S'abonner

TRANSLATE

Sélectionner une langue

LENGUAJE C++

Programacion C++
Números amigos en C++

```
Pattern pat = Pattern.compile("(^[a][bc].*)");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("SI");
} else {
    System.out.println("NO");
}
```

5. Comprobar si el String *cadena* está formado por un mínimo de 5 letras mayúsculas o minúsculas y un máximo de 10.

```
Pattern pat = Pattern.compile("[a-zA-Z]{5,10}");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("SI");
} else {
    System.out.println("NO");
}
```

6. Comprobar si el String *cadena* no empieza por un dígito

```
Pattern pat = Pattern.compile("^[^\\d].*");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("SI");
} else {
    System.out.println("NO");
}
```

7. Comprobar si el String *cadena* no acaba con un dígito

```
Pattern pat = Pattern.compile(".*[^\\d]$");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("SI");
} else {
    System.out.println("NO");
}
```

8. Comprobar si el String *cadena* solo contienen los caracteres a ó b

```
Pattern pat = Pattern.compile("(a|b)+");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("SI");
} else {
    System.out.println("NO");
}
```

9. Comprobar si el String *cadena* contiene un 1 y ese 1 no está seguido por un 2

```
Pattern pat = Pattern.compile(".*1(?:2).*");
Matcher mat = pat.matcher(cadena);
if (mat.matches()) {
    System.out.println("SI");
} else {
    System.out.println("NO");
}
```

Ejemplo: expresión regular para comprobar si un email es válido

```
package ejemplo1;
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Ejemplo1 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        String email;
        System.out.print("Introduce email: ");
        email = sc.nextLine();
        Pattern pat = Pattern.compile("(^[\\w-]+(\\.[\\w-]+)*@[A-Za-z0-9]+(\\.[A-Za-z0-9]+)*\\.([A-Za-z]{2,})$)");
        Matcher mat = pat.matcher(email);
        if(mat.find()){
            System.out.println("Correo Válido");
        }else{
            System.out.println("Correo No Válido");
        }
    }
}
```

Hemos usado la siguiente expresión regular para comprobar si un email es válido:

```
"^[\\w-]+(\\.[\\w-]+)*@[A-Za-z0-9]+(\\.[A-Za-z0-9]+)*\\.([A-Za-z]{2,})$"
```

La explicación de cada parte de la expresión regular es la siguiente:

[\\w-]+	Inicio del email El signo + indica que debe aparecer uno o más de los caracteres entre corchetes: \\w indica caracteres de la A a la Z tanto mayúsculas como minúsculas, dígitos del 0 al 9 y el carácter _ Carácter - En lugar de usar \\w podemos escribir el rango de caracteres con lo que esta expresión quedaría así: [A-Za-z0-9_]+
(\\.[\\w-]+)*	A continuación: El * indica que este grupo puede aparecer cero o más veces. El email puede contener de forma opcional un punto seguido de uno o más de los caracteres entre corchetes.
@	A continuación debe contener el carácter @
[A-Za-z0-9]+	Después de la @ el email debe contener uno o más de los caracteres que aparecen entre los corchetes
(\\.[A-Za-z0-9]+)*	Seguido (opcional, 0 ó más veces) de un punto y 1 ó más de los caracteres entre corchetes
(\\.[A-Za-z]{2,})	Seguido de un punto y al menos 2 de los caracteres que aparecen entre corchetes (final del email)

Usar expresiones regulares con la clase String. Métodos matches y splits.

String.matches(regex)

Podemos comprobar si una cadena de caracteres cumple con un patrón usando el método matches de la clase String. Este método recibe como parámetro la expresión regular.

```
if (cadena.matches(".*1(?:2).*")) {
    System.out.println("SI");
} else {
    System.out.println("NO");
}
```

String.split(regex)

El método split de la clase String es la alternativa a usar StringTokenizer para separar cadenas. Este método divide el String en cadenas según la expresión regular que recibe. La expresión regular no forma parte del array resultante.

Ejemplo 1:

Ce site utilise des cookies provenant de Google pour fournir ses services et analyser le trafic. Votre adresse IP et votre user-agent, ainsi que des statistiques relatives aux performances et à la sécurité, sont transmis à Google afin d'assurer un service de qualité, de générer des statistiques d'utilisation, et de détecter et de résoudre les problèmes d'abus.

EN SAVOIR PLUS OK

```
String str = "blanco-rojo;amarillo.verde_azul";
String [] cadenas = str.split("[~:~.~]");
for(int i = 0; i<cadenas.length; i++){
    System.out.println(cadenas[i]);
}
```

Muestra por pantalla:

```
blanco
rojo
amarillo
verde
azul
```

Ejemplo 2:

```
String str = "esto es un ejemplo de como funciona split";
String [] cadenas = str.split("(e[s|m])|(p|)");
for(int i = 0; i<cadenas.length; i++){
    System.out.println(cadenas[i]);
}
```

Salida:

```
to
un ej
o de como funciona s
it
```

Si te ha sido útil compártelo



53 comentarios:

- Anónimo

21 de agosto de 2013, 7:32

Excelente, agrego un aporte <http://usandojava.blogspot.com/2013/08/expresiones-regulares-usando-java-parte.html>

Responder
- Anónimo

23 de enero de 2014, 14:23

Me podriais ayudar diciendo que hace esta expresión regular?

(?<!{^.{0,10}})(!{^.{0,10}})(/^(^.{0,10}})))(&w+)(?!{^.{0,10}})(/^(^.{0,10}}))

Gracias

Responder

Respuestas

Anónimo

27 de enero de 2014, 20:43

Puedes probar tus expresiones regulares en:
<http://www.regexper.com/>

Responder
- Unknown

6 de marzo de 2014, 7:18

cual sería una expresion valida en el ejemplo del correo porque le introduzco correos y no los acepta

Responder

Respuestas

Enrique

6 de marzo de 2014, 17:26

En la expresión se ha colado un espacio en blanco al final entre el \$ y las comillas finales
"^[\\w;]+(\\[\\w-\\])*@[A-Za-z0-9]+(\\[\\[A-Za-z0-9\\])*"^(\\[A-Za-z\\]{2,})\$"
ese espacio sobra. Si lo quitas funciona correctamente. De todas formas ya lo he corregido en la entrada.

Gracias por avisar Jesus!!

Responder

mauricioramirezsen

20 de abril de 2014, 8:23

como haria una expresion regular que me valide la direccion de una casa
ejemplo avenida 11 # 6-27 el llano

Responder

Daniel Saavedra

28 de abril de 2014, 17:22

Como valido que un String tenga la siguiente estructura
El sistema debe validar que el valor para el campo "variables" sea igual a id=valorId[anio=Año]nombre=Nombre Cliente MPI; Un ejemplo del valor del campo variable es: id=16940075[anio=2013]nombre=HAROLD

Responder

Juan José salas

9 de mayo de 2014, 20:38

muy buen blog, gracias

Responder

Enrique

14 de mayo de 2014, 8:49

Gracias por el comentario Juan Jose. Espero que lo sigas visitando y que te sea útil. Saludos

Responder

Anónimo

1 de junio de 2014, 8:51

Como harias una expresion para validar while(condicion){sentencia}; donde donde sentencia se puede repetir n veces

Responder

Anónimo

24 de junio de 2014, 22:54

Hola me parece excelente este blog. Necesito ayuda con este programa que tengo que hacer en Netbeans 8.0 con una interfaz gráfica.
Ejercicio 2:
La empresa "Juegos Felices" lo contrató para realizar un juego de ahorcado. Usted debe realizar dos módulos para este sistema.
El primer módulo es el ingreso de palabras para realizar el juego. Estas deberán almacenarse en un archivo de texto. El sistema debe controlar que la palabra a ingresar no exista previamente. También debe tener la posibilidad de borrar el contenido del archivo y dejarlo en blanco para que el usuario comience una nueva inserción de palabras.
El segundo módulo es el módulo de juego en el cual, el sistema seleccionará una palabra al azar y le irá preguntando al usuario letras, y verificará si la letra digitada se encuentra en la palabra, en caso contrario irá sumando la cantidad de intentos fallidos antes de perder. El usuario dispone de 7 oportunidades antes de perder.
Se recomienda que el archivo de texto sea un archivo de palabras separadas por espacios; y la elección de cual palabra preguntarle al usuario se realizará mediante un método aleatorio donde se especifique un número m que estará en el rango 1 a n. Donde n representa la cantidad de palabras en el archivo.

Responder

Anónimo

27 de junio de 2014, 9:36

No entendi el punto 9 ".*1(?!2).*" me lo podrian explicar?

Responder

Anónimo

12 de septiembre de 2014, 4:03

Como demonios vas a referenciar en el punto 5 a la variable "mat" en el if???

Responder

Rutilio López

30 de enero de 2015, 12:15

Genial para los que nos estamos iniciando en las expresiones regulares. Me ha sacado de varios apuros. Saludos y gracias.

Responder

Resuestas

- Anónimo

30 de enero de 2015, 23:24

Gran aporte, de lo mejor en Expresiones Regulares en Java, gracias por tú tiempo y dedicación.

Responder
- Anónimo

4 de febrero de 2015, 13:36

Este blog es increíble. Gracias por el esfuerzo, Enrique.

Responder
- Juan Martín

4 de marzo de 2015, 13:42

alguien podría decirme que regex utiliza para validar nombres españoles? con acentos, ñ, guiones para nombres compuestos, ¢ para abreviaciones de María ... etc.


Gracias!

Responder
- Anónimo

25 de mayo de 2015, 9:15

Hola como podría validar una contraseña de al menos 5 caracteres y debe estar compuesta por letras y numeros es decir ambos y sin caracteres especiales

Responder



Yeinier Animalista

30 de mayo de 2015, 12:13

Aquí tienes unos ejemplos de validación:
<http://www.contadordecaracteres.info/prueba-expresiones-regulares.html>

esta herramienta que yo utilizo para comprobar las expresiones regulares y probar reemplazos de texto

Responder

Anónimo

3 de junio de 2015, 23:15

y si quiero poner solo letras y el espacio???

Responder

▼ Respuestas



Unknown

1 de septiembre de 2015, 19:23

"\D"

Responder

Anónimo

8 de agosto de 2015, 14:06

Muy buen aporte. Muchas gracias!

Responder



Unknown

26 de agosto de 2015, 4:04

por favor me ayudan que comience con la letra a y no termine en b

Responder

▼ Respuestas




Unknown

1 de septiembre de 2015, 19:23

"a.*(?!b)"

Responder




Unknown

24 de septiembre de 2015, 23:31

Cordial Saludo
Y si quiero que no haya mas de un espacio entre palabras
Es decir:
(ola k ase) es valido
(ola k ase) no es valido

Responder

▼ Respuestas




Unknown

24 de septiembre de 2015, 23:36

aunque no es una validación lo que siempre hago es:

cadena.replace(" ","");

Responder




Unknown

28 de octubre de 2015, 21:36

Saludos a todos
si es posible que alguien me pueda ayudar al validar una cadena
que no tenga espacios iniciales(que inicien con alguna palabra) y que no se puedan usar dobles espacios o mas

Responder



Buck3Heat

16 de noviembre de 2015, 23:25

Como saber si toda una cadena tiene por lo menos una Mayuscula?

Responder

Anónimo

9 de febrero de 2016, 23:10

Saludos es muy buena informacion pero me perdi U.u
Ayuda como hago para que me acepte funciones cubicas. Ejemplo
Que no me lea x^5 o que tenga cualquier numero antes de x^5 como 5x^5,2x^5 ayuda!!

Responder

Anónimo

8 de marzo de 2016, 19:42

Hola. Podrían ayudarme con el siguiente problema:

¿cual seria la expresión regular para validar solo 10 números o 10 dígitos de un numero de teléfono?.

Responder



Unknown

1 de mayo de 2016, 7:57

Me podrian ayudar necesito una expresion regular que me valide un operador OR en java, osea que me acepte ||

Responder

Anónimo

13 de junio de 2016, 2:08

MU CHIFO

Responder



Jenson Garrido

13 de junio de 2016, 4:53

Pattern patronNumero = Pattern.compile("[^0-4]");

como puedo hacer para que solo me eliga un num de ese rango :(

Responder



Jenson Garrido

13 de junio de 2016, 5:00

Pattern patronNumero = Pattern.compile("[^0-4]");

como puedo hacer para que solo me eliga un num de ese rango :(

Responder

▼ Respuestas



Aso Guerreros

25 de junio de 2016, 2:49

^[0-4]{1}

Responder

christian 14 de septiembre de 2016, 18:32
Un ID que está formado por 8 cifras, un guión y una letra
quien me ayuda a validar esta expresion regular
[Responder](#)

▼ Respuestas

jose luis 20 de mayo de 2021, 21:18
 [0-8][-_][a-z]
[Responder](#)

christian 14 de septiembre de 2016, 18:34
Un ID que está formado por 8 cifras, un guión y una letra
quien me ayuda a validar esta expresion regular
[Responder](#)

Unknown 20 de octubre de 2016, 18:38
hay forma de simplificar este patron, compara los rangos de el valor de INT

```
(^(((0-[1,9]))((1[0-9])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(20[0-9])|(0-[0-9])|(0-[0-9])
+ "^(0-[0-9])|(0-[0-9])|(21[0-3])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(214[0-6])|(0-[0-9])|(0-[0-9])
+ "[0-9])|(2147[0-3])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(21474[0-7])|(0-[0-9])
+ "^(0-[0-9])|(0-[0-9])|(214748[0-2])|(0-[0-9])|(0-[0-9])|(2147483[0-5])"
+ "(0-[0-9])|)((21474836(0-3)|(0-[0-9])|(214748364(0-8)))|(((0-[0-9])(1,9))|((1[0-9])|(0-[0-9])|(0-[0-9])
+ "(0-[0-9])|(0-[0-9])|(20[0-9])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(21[0-3])|(0-[0-9])|(0-[0-9])
+ "[0-9])|(0-[0-9])|(0-[0-9])|(214[0-6])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(0-[0-9])|(2147[0-3])|(0-[0-9])
+ "(0-[0-9])|(0-[0-9])|(21474[0-7])|(0-[0-9])
+ "^(0-[0-9])|(0-[0-9])|((214748[0-2])|(0-[0-9])|(0-[0-9])|(2147483[0-5])
+ "(0-[0-9])|)((21474836(0-3)|(0-[0-9])|(214748364(0-7))))");
```

[Responder](#)

Unknown 9 de noviembre de 2016, 23:00
una pregunta, estoy haciendo un software que reemplaza palabras segun la que se ingrese.
como no debe de distinguir entre mayusculas y minusculas, poder modificar por ejemplo todas las palabra Uno, UNO,uno,uNo etc. por
otra. use la expresion [\\W-]+ pero no se logra, espero resuelvan mi duda. gracias
[Responder](#)

Unknown 7 de febrero de 2017, 23:09
Como hago para que me obligue a introducir una letra y un numero
[Responder](#)

Unknown 8 de abril de 2017, 21:30
Muy Util, Gracias!
[Responder](#)

Unknown 2 de septiembre de 2017, 3:53
Hola, cómo haría para verificar todos los numeros menores a 17? Gracias
[Responder](#)

Anónimo 10 de agosto de 2018, 14:50
Hola como haria para colocar formato de fecha DOMMAA
[Responder](#)

Jose Gelimer Gomez 10 de agosto de 2018, 18:00
Como hago para convalidar el caso que la expresion use &???
[Responder](#)

Unknown 5 de enero de 2019, 22:38
Hola amigos me pueden ayudar como hago para validar un formato de fecha MM/DD/AA en el mes no pase de 12 y en los dias que no
pase de 31 .
he conseguido buscado en blocks el siguiente código \\d{2}\\/(d{2}\\/(d{2})pero si diguito 18/54/89 lo recibe como si no hubiera un
error
[Responder](#)

Unknown 4 de octubre de 2019, 9:23
gracias bro, buen aporte
[Responder](#)

Sel2411 9 de julio de 2020, 2:16
Hola amigos necesito realizar un programa en Java que pueda a través de expresiones regulares determinar en una cadena de
entrada, a través de métodos es un dígito, es un número entero, número decimal, es un carácter, un identificador, un operador
matemático, operador lógico, delimitador, un carácter de puntuación
[Responder](#)

Unknown 22 de agosto de 2020, 21:23
como puedo hacer una expresión regular para una notación científica, tengo esta, pero no he podido agregar la potencia

```
(([-+]?[0-9]*\\.?[0-9]+\\\\([eE]([-+]?[0-9]+)?)*)
```

[Responder](#)

Mendez Ulises 12 de octubre de 2020, 23:17
Hola me podrían ayudar a realizar una expresion con lo siguiente?
Nombre (Deberá permitir ingresar el nombre completo de una persona con 1 o
nombres y sus 2 apellidos
[Responder](#)

Unknown 28 de noviembre de 2020, 5:57
Alguien tiene los códigos para un alaiizador lexico para el idioma español?
[Responder](#)



Programación Java by Enrique García Hernández

Esta obra está bajo una licencia [Creative Commons Reconocimiento-NoComercial-CompartirIgual 3.0 España License](http://creativecommons.org/licenses/by-nc-sa/3.0/). Para reconocer la autoría debes poner el enlace <http://puntocomnoesunlenguaje.blogspot.com.es>

Con la tecnología de [Blogger](#).