Departamento de Lenguajes y Sistemas Informáticos

Universitat d'Alacant
Universidad de Alicante

JavaScript: Expresiones regulares

Programación en Internet
Curso 2009-2010

Algunas personas cuando se enfrentan a un problema piensan "Ya sé, ¡usaré expresiones regulares!"

Ahora tienen dos problemas.

Jamie Zawinski, programador de Netscape Navigator 1.1

Programación en Internet – Curso 2009-2010

Índice

- Introducción
- Expresiones regulares en JavaScript
- Ejemplos

Programación en Internet – Curso 2009-2010

Introducción

- Una expresión regular es un patrón que se emplea para compararlo con un grupo de caracteres
- Según la Wikipedia:
  - Una expresión regular, a menudo llamada también patrón, es una expresión que describe un conjunto de cadenas sin enumerar sus elementos. Por ejemplo, el grupo formado por las cadenas Handel, Händel y Haendel se describe mediante el patrón "H(a|ä|ae)ndel".

Programación en Internet – Curso 2009-2010

Introducción

- Las expresiones regulares se pueden emplear en:
  - Comandos de sistemas operativos, como sed y grep en Linux
  - Editores de texto como emacs
  - Lenguajes de programación, de forma nativa como JavaScript, PHP, awk y Perl, o a través de librerías como Java o .NET
- Básicamente, existen dos estilos de expresiones regulares que en algunos aspectos son iguales

Programación en Internet – Curso 2009-2010

Introducción

- Una expresión regular es un patrón que puede estar formado por un conjunto de **caracteres** (letras, números o signos) y por un conjunto de **metacaracteres** que representan otros caracteres o que indican la forma de combinar los caracteres
- Los metacaracteres reciben este nombre porque no se representan a ellos mismos, sino que son interpretados de una manera especial

Programación en Internet – Curso 2009-2010

Introducción

- Los metacaracteres más usados son:

. * ? + [ ] ( ) { } ^ $ |

- **^**: Sirve para indicar que el patrón que lo acompaña esta al principio de la cadena.
- **$**: Indica que el patrón esta al final de una cadena.
- **.**: Representa cualquier carácter.
- ***:** El patrón que lo precede se repite 0 o mas veces.
- **?**: El patrón se repite 0 o 1 vez.
- **+**: El patrón se repite 1 o mas veces.
- **{x,y}:** El patrón se repite un mínimo de x veces y un máximo de y.
- **|**: Sirve para alternar expresiones.

Programación en Internet – Curso 2009-2010

Introducción

- Los corchetes [] incluidos en un patrón permiten especificar el rango de caracteres válidos a comparar.

[abc] // El patrón coincide con la cadena si en esta hay
// cualquiera de estos tres carácteres: a, b, c

[a-c] // Coincide si existe una letra en el rango ("a", "b" o "c")

c[ao]sa // Coincide con casa y con cosa

[^abc] // El patrón coincide con la cadena si en esta NO hay ninguno
// de estos tres carácteres: a, b, c.
// Nota que el signo ^ aquí tiene un valor excluyente

[0-9] // Coincide con una cadena que contenga cualquier
// número entre el 0 y el 9

Programación en Internet – Curso 2009-2010

Introducción

(): Los paréntesis sirven para agrupar expresiones regulares.

- **|**: Sirve para alternar expresiones. Por ejemplo: **(la|el):** coincide si esta presente la o el.

- Para escapar caracteres especiales debemos usar la clásica barra invertida '\'. Por ejemplo si buscamos 100$, pondríamos 100\$, ya que si pusiéramos 100$ buscaría un 100 a final de cadena.

Programación en Internet – Curso 2009-2010

Patrón	Significado
.	cualquier carácter (excepto \n y \r)
^c	empezar por el carácter c
c$	terminar por el carácter c
c+	1 o más caracteres c
c*	0 o más caracteres c
c?	0 o 1 caracteres c
\n	nueva línea
\t	tabulador
\	escape, para escribir delante de caracteres especiales: ^ . [ ] % ( ) | * ? { } \
(cd)	caracteres c y d agrupados
c|d	carácter c o d
c{n}	n veces el carácter c
c{n,}	n o más caracteres c
c{n,m}	desde n hasta m caracteres c

Programación en Internet – Curso 2009-2010

Patrón	Significado
[a-z]	cualquier letra minúscula
[A-Z]	cualquier letra mayúscula
[0-9]	cualquier dígito
[cde]	cualquiera de los caracteres c, d o e
[c-f]	cualquier letra entre c y f (es decir, c, d, e o f)
[^c]	que no esté el carácter c
\w	cualquier letra o dígito o subrayado (pero no vocales acentuadas, ñ, ç, etc.)
\W	lo contrario de \w
\d	cualquier dígito
\D	lo contrario de \d
\s	cualquier espacio en blanco
\S	lo contrario de \s
\b	busca un emparejamiento a partir de un límite de palabra
\B	busca un emparejamiento cuando **no** es un límite de palabra

Programación en Internet – Curso 2009-2010

Expresiones regulares en JavaScript

- Podemos crear una ER de dos formas:

```
/* Permite crear expresiones regulares en tiempo
   de ejecución */
var txt=new RegExp(pattern,attributes);

/* Crea expresiones regulares estáticas, no se
   pueden modificar durante la ejecución */
var txt=/pattern/attributes;
```

Programación en Internet – Curso 2009-2010

Expresiones regulares en JavaScript

- Los atributos son:
  - `m`: Si nuestra cadena contiene varias líneas físicas (\n) respeta esos saltos de línea, lo que significa, por ejemplo, que las anclas ^ $ no se aplican al principio y final de la cadena, sino al principio y final de cada línea.
  - `i`: Se confronta el patrón con la cadena ignorando mayusculas y minúsculas.
  - `g`: Realiza una búsqueda global, no se detiene en la primera ocurrencia que encuentra.

Programación en Internet – Curso 2009-2010

Expresiones regulares en JavaScript

- Métodos del objeto `RegExp`:
  - `compile()`: cambia la expresión regular del objeto
  - `exec()`: busca la ER, devuelve el valor encontrado y recuerda la posición
  - `test()`: busca la ER, devuelve `true` o `false`

Expresiones regulares en JavaScript

- Además, el objeto `String` tiene unos métodos que admiten ER:
  - `search()`: busca en una cadena la ER, devuelve la posición
  - `match()`: busca en una cadena la ER, devuelve un array con los valores emparejados
  - `replace()`: sustituye unos caracteres por otros caracteres
  - `split()`: divide una cadena en un array de cadenas

Ejemplos

- Valida una matrícula moderna:

```
function validaMatricula() {
  var mat = document.getElementById("matricula").value;

  var ex1 = new RegExp("^[0-9]{4} [A-Z]{3}$");
  var ex2 = /^[0-9]{4} [A-Z]{3}$/;

  if(ex1.test(mat))
    alert("Ok");
  else
    alert("Error");

  if(ex2.test(mat))
    alert("Ok");
  else
    alert("Error");
}
```

Programación en Internet – Curso 2009-2010

Ejemplos

- Valida una fecha (sólo el formato):

```
function validaFecha() {
  var fec = document.getElementById("fecha").value;

  var ex1 = new RegExp("^(0?[1-9]|[12][0-9]|3[01])\/(0?[1-
   9]|1[012])\/[0-9]{1,2}$");
  var ex2 = /^(0?[1-9]|[12][0-9]|3[01])\/(0?[1-9]|1[012])\/[0-
   9]{1,2}$/;

  if(ex1.test(fec))
    alert("Ok");
  else
    alert("Error");

  if(ex2.test(fec))
    alert("Ok");
  else
    alert("Error");
}
```