5.A. Gestión de eventos y formularios en JavaScript

5.- Expresiones regulares y objetos RegExp.

Para validar si determinada cadena de caracteres concuerda con un patrón esperado podemos comprobar mediante un programa que la cadena se adapta al formato definido en el patrón, sin embargo este método es bastante farragoso cuando el patrón es complejo. Podemos escribir ese patrón de búsqueda y que la lógica contenida en JavaScript realice el trabajo por nosotros.

Las expresiones regulares son patrones de búsqueda, que se pueden utilizar para encontrar texto que coincida con el patrón especificado.

Ejemplo de búsqueda de una cadena de texto sin usar expresiones regulares:

```
var texto = "La linea de alta velocidad llegará pronto a toda España,";

var subcadena = "velocidad";

var i = texto.indexOf(subcadena); // devuelve 17, índice de donde se encuentra la subcadena


if (i != -1) // correcto, se ha encontrado la subcadena
```

Este código funciona porque estamos buscando una subcadena de texto exacta. ¿Pero qué pasaría si hiciéramos una búsqueda más general? Por ejemplo si quisiéramos buscar la cadena "car" en textos como "cartón", "bicarbonato", "practicar", ...?

Cuando estamos buscando cadenas que cumplen un patrón en lugar de una cadena exacta, necesitaremos usar expresiones regulares. Podrías intentar hacerlo con funciones de `String`, pero al final, es mucho más sencillo hacerlo con expresiones regulares, aunque la sintaxis de las mismas es un poco extraña y no necesariamente muy amigable.

En JavaScript las expresiones regulares se gestionan a través del objeto `RegExp`.

Para crear un literal del tipo `RegExp`, tendrás que usar la siguiente sintaxis:

```
var  expresion = /expresión regular/;
```

La expresión regular está contenida entre la barras `/`, y fíjate que no lleva comillas. Las comillas sólo se pondrán en la expresión regular, cuando formen parte del patrón en si mismo.

Las expresiones regulares están hechas de caracteres, solos o en combinación con caracteres especiales, que se proporcionarán para búsquedas más complejas. Por ejemplo, lo siguiente es una expresión regular que realiza una búsqueda que contenga las palabras *Aloe Vera*, en ese orden y separadas por uno o más espacios en medio:

```
var expresion=/Aloe\s+Vera/;
```

Los caracteres especiales en este ejemplo son, la barra invertida (`\`), que tiene dos efectos: o bien se utiliza con un carácter regular, para indicar que se trata de un carácter especial, o se usa con un carácter especial, tales como el signo más (`+`), para indicar que el carácter debe ser tratado literalmente. En este caso, la barra invertida se utiliza con "`s`", que transforma la letra s en un carácter especial indicando un espacio en blanco, un tabulador, un salto de línea, etc. El símbolo `+` indica que el carácter anterior puede aparecer una o más veces.

Créditos de la imagen

Autoría: Chris Radcliff.