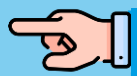
Funciones y Ciclos

Expresiones regulares

{desafío} latam_

Codificar un programa utilizando funciones para la reutilización de código acorde al lenguaje Javascript.

- Unidad 1:
  Introducción al lenguaje JavaScript
- Unidad 2:
  Funciones y Ciclos
- Unidad 3:
  Arrays y Objetos
- Unidad 4:
  APIs



Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Reconocer las principales características de las expresiones regulares para identificar sus ventajas y contextos de uso.*
- *Aplicar expresiones regulares básicas sobre cadenas de texto para validar la estructura de la información recibida.*

{desafío} latam_

De acuerdo a lo aprendido, ¿qué es el DOM?

{desafío} latam_

¿Qué almacena el atributo innerHTML de un elemento HTML?

/* ¿Qué son las Expresiones Regulares (RegExp)? */

Expresiones Regulares

¿Qué son las RegExp?

- Son objetos que describen patrones de caracteres y se utilizan como patrones de búsqueda.

- Son ampliamente utilizados a la hora de realizar validaciones de cadenas de texto, ya que suelen tener una sintaxis muy corta y su utilización es muy sencilla a la hora de validar.

```
/patrón/modificadores;
```

- Ejemplo:

```
/[a-zA-Z]/gim
/[A-Z0-9._%+-]+@[A-Z0-9-]+.[A-Z]/gim
```

/* Comprendiendo y validando datos con Expresiones Regulares */

{desafío}
latam_

Comprendiendo y validando datos

- Para validar datos con expresiones regulares, podemos hacerlo mediante el uso del método match.

- Este método retorna un arreglo en el caso de encontrar alguna coincidencia con la coincidencia hallada. De lo contrario, se retornará null.

- Por ejemplo, si quisiéramos validar que una cadena de texto que contenga la palabra gato, podríamos generar el siguiente patrón:

Comprendiendo y validando datos

Modificadores de Expresiones Regulares

i (insensitive)	g (global)	m (multi line)
Se utiliza para que el patrón de búsqueda no discrimine entre mayúsculas y minúsculas.	Se utiliza para buscar todas las coincidencias posibles dentro de una cadena de texto, en función de un patrón dado.	Permite encontrar coincidencias al principio de cada línea y al final de éstas, mediante los operadores ^ y $ respectivamente.
`/leopardo/i`	`/leopardo/ig`	`/leopardo$/igm`

Demostración - “Validar ingreso de datos”

Ejercicio guiado

Validar ingreso de datos

Validar el ingreso de datos en un buscador de animales, por lo que sólo debe aceptar las palabras “perro” y “gato”. De lo contrario, no puede permitir la búsqueda, generando un mensaje de error en un alert.

Para este ejercicio se cuenta con el documento HTML en Ejercicios - **Expresiones Regulares** dentro del LMS.

Ejercicio guiado

Validar ingreso de datos

- **Paso 1:** Lo primero que se necesita es acceder al botón de búsqueda agregando un listener, luego, agregar una función externa para poder capturar los datos ingresados por el usuario y aplicar el proceso de validación mediante las expresiones regulares.

```
var buscar = document.getElementById("buscar");
buscar.addEventListener('click',validar);
```

Ejercicio guiado

Validar ingreso de datos

- **Paso 2:** Una vez que el usuario haga click en el botón, debemos tomar los valores ingresados en el elemento input, para esto hacemos uso de la instrucción querySelector, igualmente creamos las dos expresiones regulares que utilizaremos dentro de una estructura condicional “if” en conjunto con el método match (), que busca un string para una coincidencia con una expresión regular definida.

```
function validar(){
   var animal = document.querySelector(".animal").value;
   var patron1 = /gato/i;
   var patron2 = /perro/i;
   if (animal.match(patron1) || animal.match(patron2)){
      alert("Palabra ingresada permitida");
   } else {
      alert("La palabra ingresada no es permitida");
   }
};
```

Ejercicio guiado

Validar ingreso de datos

- **Paso 3:** Ahora si unimos todo el código JavaScript.

```
var buscar = document.getElementById("buscar");
buscar.addEventListener('click',validar);

function validar(){
   var animal = document.querySelector(".animal").value;
   var patron1 = /gato/i;
   var patron2 = /perro/i;
   if (animal.match(patron1) || animal.match(patron2)){
      alert("Palabra ingresada permitida");
   } else {
      alert("La palabra ingresada no es permitida");
   }
};
```

Ejercicio guiado

Validar ingreso de datos

- **Paso 4:** Al ejecutar el código anterior en nuestro navegador, el resultado obtenido al ingresar perro o gato en el campo requerido.

- **Paso 5:** Al ingresar una palabra no permitida por el validar, el mensaje sería.

/* Corchetes en Expresiones Regulares */

Corchetes en Expresiones Regulares

[abc]

Los corchetes en expresiones regulares nos ayudan a encontrar un rango de caracteres que definamos dentro de ellos:

- **[abc]:** Sólo se obtienen los caracteres definidos entre corchetes.

```
const patron = /[abc]/gim;
const texto = 'a b c ';
console.log(texto.match(patron));
```

```
(3) ["a", "b", "c"]
```

{desafío} latam_

Corchetes en Expresiones Regulares

[^xyz]

- **[^xyz]:** Encuentra cualquier caracter que no coincida con los caracteres al interior de los corchetes:

```
const patron = /[^xyz]/gim;
const texto = 'a y c x ';
console.log(texto.match(patron));
```

```
(2) ["a", "c"]
```

Corchetes en Expresiones Regulares

[a-zA-Z]

- **[a-zA-Z]:** Encuentra cualquier carácter entre los rangos de la letra a hasta la z y desde A hasta Z. Es decir, válida cualquier carácter que sea una letra. Por ejemplo, con el siguiente código:

```
const patron = /[a-zA-Z]/gim;
const texto = 'todo a excepcion del numero 1';
console.log(texto.match(patron));
```

```
(23) [ "t", "o", "d", "o", "a", "e", "x", "c", "e",
"p", "c", "i", "o", "n", "d", "e", "l", "n", "u",
"m", "e", "r", "o" ]
```

Demostración - “Corchetes en la expresión regular”

Ejercicio guiado

Corchetes en la expresión regular

Validar el ingreso de datos en un campo de texto que permita el ingreso de caracteres alfanuméricos combinados, más no solo numéricos. De lo contrario, no puede permitir el envío de la información, generando un mensaje de error en un alert.

Para este ejercicio se cuenta con el documento HTML en **Ejercicios - Expresiones Regulares** dentro del LMS.

Ejercicio guiado

Corchetes en la expresión regular

- **Paso 1:** Lo primero que se necesita es acceder al botón de búsqueda agregando un listener, luego, agregar una función externa para poder capturar los datos ingresados por el usuario y aplicar el proceso de validación mediante las expresiones regulares.

```
var enviar = document.getElementById("enviar");
enviar.addEventListener('click',validar);
```

Ejercicio guiado

Corchetes en la expresión regular

- **Paso 2:** Ahora una vez que el usuario haga click en el botón, debemos tomar los valores ingresados en el elemento input, para esto hacemos uso de la instrucción querySelector, igualmente creamos las expresiones regulares que utilizaremos dentro de una estructura condicional “if” en conjunto con el método match (). En este caso la expresión regular permitirá texto alfanumérico entre mayúsculas y minúsculas, pero un texto con solo caracteres numéricos no lo permitirá.

```
function validar(){
   var lugar = document.querySelector(".lugar").value;
   var permitido = /[a-zA-Z]/gim;
   if (lugar.match(permitido)){
      alert("El texto ingresado es permitido");
   } else {
      alert("Solo debe ingresar caracteres alfabéticos");
   };
};
```

Ejercicio guiado

Corchetes en la expresión regular

- **Paso 3:** Ahora si unimos todo el código JavaScript, quedaría el script de esta manera:

```
var enviar = document.getElementById("enviar");
enviar.addEventListener('click',validar);

function validar(){
   var lugar = document.querySelector(".lugar").value;
   var permitido = /[a-zA-Z]/gim;
   if (lugar.match(permitido)){
      alert("El texto ingresado es permitido");
   } else {
      alert("El texto ingresado no es permitido");
   };
};
```

Utilizar regex para validar un campo de un formulario

- Las expresiones regulares son muy útiles a la hora de validar formularios, por ejemplo, validar si un campo es sólo números o sólo letras, formatos especiales como correos, entre otros.

- En este caso, implementaremos el objeto de JavaScript RegExp, el cual, es un objeto de expresión regular con propiedades y métodos predefinidos.

- Entre ellos se encuentra el método “test()”, quien compara una expresión dada con un patrón establecido y devuelve verdadero o falso, según el resultado.

Demostración - “Validando campos de un formulario”

Ejercicio guiado

Validando campos de un formulario

Validar los campos nombre, teléfono y correo de un formulario, desarrollando paso a paso cada una de las instrucciones.

Para este ejercicio se cuenta con el documento HTML en **Ejercicios - Expresiones Regulares** dentro del LMS.

Ejercicio guiado

Validando campos de un formulario

- **Paso 1:** En el script.js, agregaremos un listener al formulario con el evento del tipo submit, por lo que es necesario guardar el id del formulario en una variable y captemos los datos dentro de la función del escucha, implementado los querySelector.

```
let form = document.getElementById( "form" );

form.addEventListener( "submit", function ( event ) {
   event.preventDefault();
   limpiarErrores();
   let textNombre = document.querySelector(".textNombre").value;
   let textTelefono = document.querySelector(".textTelefono").value;
   let textEmail = document.querySelector(".textEmail").value;

   let resultado = validar(textNombre,textTelefono,textEmail);

   if(resultado == true) {
     exito();
   };
});
```

Ejercicio guiado

Validando campos de un formulario

- **Paso 2:** Realizamos las funciones externas de limpieza, validación y de éxito. Estas funciones tendrán sus procesos por separado y serán llamados desde el listener.

```
function limpiarErrores() {
document.querySelector(".resultado").innerHTML
= "";
document.querySelector(".errorNombre").innerHTM
L = "";
document.querySelector(".errorTelefono").innerH
TML = "";
document.querySelector(".errorEmail").innerHTML
= "";
};

function exito() {

document.querySelector(".resultado").innerHTML
= "Formulario pasó la validación";
};

function validar(nombre,telefono,email) {

};
```

Ejercicio guiado

Validando campos de un formulario

- **Paso 3:** trabajaremos dentro de la función validar(), recibiendo como parámetros los datos ingresados en el formulario. Primero agregaremos la validación del nombre, para ello usamos el patrón /[a-zA-Z]/gim en el que sólo permitimos letras, entonces validamos con el método test() si el texto cumple o no la condición, si no mediante javascript agregamos un mensaje para que el usuario corrija. Cambiamos nuestra variable **`pasamosLaValidacion`** a false para que no se muestre el mensaje de éxito,

```
function validar(nombre,telefono,email) {

    let pasamosLaValidacion = true;
    let validacionNombre = /[a-zA-Z]/gim;

    if (validacionNombre.test(nombre) == false) {
       document.querySelector(".errorNombre").innerHTML = "Ingrese un
nombre válido."
       pasamosLaValidacion = false;
   }
};
```

{desafío} latam_

Ejercicio guiado

Validando campos de un formulario

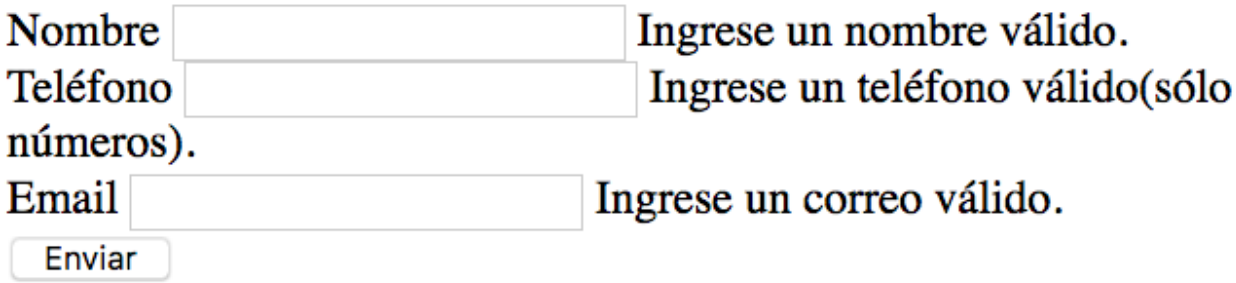
- **Paso 4:** Ahora aplicaremos la lógica para el teléfono y correo modificando las expresiones regulares para cada elemento.

```
function validar(nombre,telefono,email) {

            let pasamosLaValidacion = true;
            let validacionNombre = /[a-zA-Z]/gim;

            if (validacionNombre.test(nombre) == false) {
            document.querySelector(".errorNombre").innerHTML = "Ingrese un
nombre válido."
            pasamosLaValidacion = false;
            };

            let validacionTelefono = /\d/gim;

            if (validacionTelefono.test(telefono) == false) {
            document.querySelector(".errorTelefono").innerHTML = "Ingrese un
teléfono válido(sólo números)."
            pasamosLaValidacion = false;
            };

            let validaciónEmail = /[A-Z0-9._%+-]+@[A-Z0-9-]+.[A-Z]/gim;

            if (validaciónEmail.test(email) == false ) {
            document.querySelector(".errorEmail").innerHTML = "Ingrese un
correo válido."
            pasamosLaValidacion = false;
            };

            return pasamosLaValidacion;
};
```

Ejercicio guiado

Validando campos de un formulario

- **Paso 5:** Al unir todo el código y ejecutar el index.html en el navegador web, el resultado final debería ser el siguiente para el caso que el usuario no ingrese dato alguno:



¿Existe algún concepto que no hayas comprendido?

Volvamos a revisar los conceptos que más te hayan costado antes de seguir adelante

Próxima sesión...

- *Revisaremos y desarrollaremos el material sincrónico que corresponde a un **Desafío evaluado**, con el cual pondrás a prueba los conocimientos adquiridos en estas sesiones.*

{desafío}
latam_

{desafío}
latam_

Academia de
talentos digitales