# Ejercicios de Javascript con variables, funciones y clases

Aprende a programar en Javascript con ejercicios prácticos

Empezar

# Descripción general

En este curso aprenderás los conceptos fundamentales de Javascript a través de ejercicios que te ayudarán a practicar con variables, funciones y clases. Conviértete en un experto en el lenguaje de programación más popular del desarrollo web.

# Introducción a las variables en JavaScript

01 | Introducción a las variables en JavaScript

## Variables en JavaScript

Una variable en JavaScript es un contenedor que nos permite almacenar y manipular datos. Pueden almacenar diferentes tipos de datos, como números, texto, booleanos y objetos. Las variables son fundamentales en la programación, ya que nos permiten almacenar y acceder a datos de manera eficiente.

En JavaScript, las variables se definen utilizando la palabra clave `var`, seguida del nombre de la variable. Por ejemplo, para crear una variable llamada `edad` que almacena un número, podemos utilizar la siguiente sintaxis:

```
var edad = 25;
```

También es posible declarar múltiples variables en una sola línea utilizando comas:

```
var nombre = "Juan", apellido = "Pérez", edad = 30;
```

# Tipos de datos

En JavaScript, existen varios tipos de datos que podemos asignar a una variable:

- **Números**: se utilizan para representar valores numéricos, ya sean enteros o decimales. Por ejemplo: `var edad = 25;`
- **Cadenas de texto**: se utilizan para representar texto. Deben estar encerradas entre comillas simples `'` o dobles `"`. Por ejemplo: `var nombre = "Juan";`
- **Booleanos**: se utilizan para representar los valores de verdadero (`true`) o falso (`false`). Por ejemplo: `var esMayorDeEdad = true;`
- **Objetos**: se utilizan para representar una colección de datos relacionados, organizados en pares clave-valor. Por ejemplo: `var persona = { nombre: "Juan", edad: 25 };`

# Ámbito de las variables

Las variables en JavaScript tienen un ámbito, que es el contexto en el cual pueden ser accedidas. Existen dos tipos de ámbito: global y local.

- **Ámbito global**: Cuando una variable se declara fuera de cualquier función, se considera global y puede ser accedida desde cualquier parte del código.
- **Ámbito local**: Cuando una variable se declara dentro de una función, se considera local y solo puede ser accedida desde dentro de esa función.

```
 var nombre = "Juan"; // variable global

function imprimirNombre() {
    var apellido = "Pérez"; // variable local
    console.log(nombre); // Acceso a variable global
    console.log(apellido); // Acceso a variable local
}
```

```
imprimirNombre();
console.log(nombre); // Acceso a variable global fuera de la función
console.log(apellido); // ERROR: Variable local no está disponible fuera de la
```

Es importante tener en cuenta el ámbito de las variables para evitar conflictos y errores en nuestro código.

## Conclusiones

En resumen, las variables son una parte fundamental de JavaScript ya que nos permiten almacenar y manipular datos. Aprendimos cómo declarar variables utilizando la palabra clave `var` y vimos los diferentes tipos de datos que podemos asignar a una variable. También discutimos el ámbito de las variables y cómo afecta el alcance de las mismas. ¡Ahora que conocemos las variables en JavaScript, podemos usarlas para crear programas más complejos y sofisticados!

Conclusión - Introducción a las variables en JavaScript

En este curso, aprendiste sobre las variables en JavaScript y cómo pueden almacenar diferentes tipos de datos. También vimos cómo declarar y usar variables en tus programas. ¡Sigue practicando para solidificar tus conocimientos en el uso de variables en JavaScript!

# Creación y uso de funciones en JavaScript

02 | Creación y uso de funciones en JavaScript

## ¿Qué son las funciones en JavaScript?

En JavaScript, una función es un bloque de código que se puede reutilizar para realizar una tarea específica. Las funciones se utilizan para dividir el código en partes más pequeñas y organizadas, lo que facilita su mantenimiento y reutilización.

En JavaScript, hay dos formas de crear funciones: mediante la declaración de función y mediante la expresión de función.

### Declaración de función

La declaración de función se realiza utilizando la palabra clave `function`, seguida del nombre de la función, paréntesis y llaves. Veamos un ejemplo:

```
 function saludar() {
  console.log("¡Hola!");
}
```

En este ejemplo, hemos creado una función llamada `saludar` que imprime "¡Hola!" en la consola.

### Expresión de función

La expresión de función se utiliza cuando asignamos una función a una variable. Veamos un ejemplo:

```
 var sumar = function(a, b) {
  return a + b;
};
```

En este ejemplo, hemos creado una función anónima y la hemos asignado a la variable `sumar`. Esta función toma dos parámetros (`a` y `b`) y devuelve la suma de los mismos.

## Parámetros y argumentos de una función

Las funciones en JavaScript pueden aceptar parámetros, que son valores que se pasan a la función cuando se invoca. Estos parámetros se definen dentro de los paréntesis en la declaración de la función.

Por ejemplo, vamos a crear una función que tome dos parámetros y los multiplique:

```
 function multiplicar(num1, num2) {
  return num1 * num2;
}
```

En este caso, `num1` y `num2` son los parámetros de la función `multiplicar`. Cuando invoquemos esta función, deberemos proporcionar los argumentos correspondientes, que son los valores reales que se utilizan en la función:

```
 var resultado = multiplicar(4, 6);
console.log(resultado); // Salida: 24
```

En este ejemplo, `4` y `6` son los argumentos que se pasan a la función `multiplicar`. La función devuelve el resultado, que se almacena en la variable `resultado`.

## Retorno de valor de una función

Una función en JavaScript también puede tener un valor de retorno, que es el valor que se devuelve cuando la función se ejecuta. Este valor de retorno se especifica utilizando la palabra clave `return`.

Vamos a modificar nuestra función `multiplicar` para que devuelva el resultado en lugar de imprimirlo en la consola:

```
 function multiplicar(num1, num2) {
  return num1 * num2;
}

var resultado = multiplicar(4, 6);
console.log(resultado); // Salida: 24
```

En este caso, la función `multiplicar` devuelve el resultado de la multiplicación de `num1` y `num2`. Este resultado se asigna a la variable `resultado` y se muestra en la consola.

# Uso de funciones en JavaScript

Una vez que hemos creado una función en JavaScript, podemos invocarla en cualquier momento para ejecutar su código. Para invocar una función, simplemente escribimos su nombre seguido de paréntesis.

Vamos a crear una función de ejemplo y a invocarla:

```
 function saludar(nombre) {
  console.log("¡Hola, " + nombre + "!");
}

saludar("Juan"); // Salida: ¡Hola, Juan!
saludar("María"); // Salida: ¡Hola, María!
```

En este ejemplo, hemos creado la función `saludar`, que toma un parámetro `nombre` y muestra un saludo en la consola. Hemos invocado esta función dos veces, pasando diferentes argumentos.

Conclusión - Creación y uso de funciones en JavaScript

En este curso, exploramos en detalle las funciones en JavaScript y cómo pueden realizar tareas específicas. Aprendiste a declarar funciones, asignar parámetros y devolver valores. ¡Continúa practicando para convertirte en un experto en el uso de funciones en JavaScript!

# Implementación de clases en JavaScript



En la programación orientada a objetos, las clases son una herramienta fundamental para estructurar y organizar el código. En JavaScript, también podemos utilizarlas para crear objetos y definir su comportamiento. En esta sección, exploraremos cómo implementar clases en JavaScript y cómo aprovechar todas sus capacidades.

## ¿Qué es una clase en JavaScript?

Una clase en JavaScript es un plano o plantilla que define las propiedades y métodos de un objeto. Actúa como un modelo para crear nuevos objetos que comparten características y comportamientos similares. Al definir una clase, especificamos qué atributos tendrá el objeto y qué acciones podrá realizar.

## Sintaxis básica de una clase en JavaScript

La sintaxis básica para definir una clase en JavaScript es la siguiente:

```
class NombreDeLaClase {
 constructor(parametros) {
   // inicializar propiedades
 }

 metodo() {
   // definir comportamiento del método
 }
}
```

- **class** es la palabra reservada para declarar la clase seguida del nombre que le daremos a la misma.
- **constructor** es un método especial que se invoca al crear un nuevo objeto a partir de la clase. Aquí se inicializan las propiedades del objeto.
- **metodo** es una función dentro de la clase que define un comportamiento específico del objeto.

## Creación de objetos a partir de una clase

Una vez que hemos definido una clase, podemos crear instancias u objetos a partir de ella utilizando el siguiente formato:

```
const objeto = new NombreDeLaClase(parametros);
```

El operador **new** se encarga de crear un nuevo objeto basado en la clase especificada. Los parámetros se pasan al constructor de la clase, si es que lo tiene.

## Atributos y métodos de una clase

Los atributos de una clase son las variables que contienen los valores específicos de cada objeto creado a partir de la misma. Se definen dentro del constructor y se acceden utilizando la palabra **this** seguida del nombre del atributo.

Los métodos son las funciones que definen el comportamiento de los objetos en la clase. Se definen utilizando la sintaxis de una función y se pueden acceder utilizando el nombre del objeto seguido de un punto y el nombre del método.

## Herencia en JavaScript

Una de las características más poderosas de las clases en JavaScript es la capacidad de heredar propiedades y métodos de una clase padre. Esto permite crear una jerarquía de clases y reutilizar código de manera efectiva.

Para lograr la herencia en JavaScript, utilizamos la palabra reservada **extends** seguida del nombre de la clase padre. De esta manera, la clase hija hereda todas las propiedades y métodos de la clase padre.

```
 class ClaseHija extends ClasePadre {
  constructor(parametros) {
    super(parametros); // invocar el constructor de la clase padre
  }

  // definir métodos y propiedades adicionales
}
```

Podemos sobrescribir métodos y propiedades de la clase padre en la clase hija si necesitamos modificar su comportamiento.

# Conclusiones

Las clases en JavaScript nos permiten organizar y estructurar nuestro código de una manera más eficiente y fácil de mantener. A través de la sintaxis y las funcionalidades mencionadas, podemos crear objetos con propiedades y comportamientos específicos, así como también aprovechar la herencia para reutilizar código y crear relaciones entre las clases.

Conclusión - Implementación de clases en JavaScript

En este curso, te introdujimos al mundo de las clases en JavaScript y cómo te permiten crear objetos con características y comportamientos específicos. Aprendiste a declarar clases, crear objetos a partir de ellas y acceder a sus propiedades y métodos. ¡Sigue practicando para dominar el uso de clases en JavaScript!

# Ejercicios Practicos

Pongamos en práctica tus conocimientos

04 | Ejercicios Practicos

En esta lección, pondremos la teoría en práctica a través de actividades prácticas. Haga clic en los elementos a continuación para verificar cada ejercicio y desarrollar habilidades prácticas que lo ayudarán a tener éxito en el tema.

## Declaración de variables

En este ejercicio, debes declarar diferentes tipos de variables en JavaScript, como variables numéricas, de texto y booleanas.

## Funciones de suma y resta

En este ejercicio, debes crear funciones en JavaScript que realicen operaciones de suma y resta con dos números ingresados como parámetros.

## Creación de una clase de coche

En este ejercicio, debes implementar una clase en JavaScript que represente un coche. La clase debe tener propiedades como marca, modelo y año, y métodos para obtener y establecer estas propiedades.

# Resumen

Repasemos lo que acabamos de ver hasta ahora

05 | Resumen

- ✓ En este curso, aprendiste sobre las variables en JavaScript y cómo pueden almacenar diferentes tipos de datos. También vimos cómo declarar y usar variables en tus programas. ¡Sigue practicando para solidificar tus conocimientos en el uso de variables en JavaScript!
- ✓ En este curso, exploramos en detalle las funciones en JavaScript y cómo pueden realizar tareas específicas. Aprendiste a declarar funciones, asignar parámetros y devolver valores. ¡Continúa practicando para convertirte en un experto en el uso de funciones en JavaScript!
- ✓ En este curso, te introdujimos al mundo de las clases en JavaScript y cómo te permiten crear objetos con características y comportamientos específicos. Aprendiste a declarar clases, crear objetos a partir de ellas y acceder a sus propiedades y métodos. ¡Sigue practicando para dominar el uso de clases en JavaScript!

# Prueba

Comprueba tus conocimientos respondiendo unas preguntas

06 | Prueba

Pregunta 1/6

¿Cuál es la forma correcta de declarar una variable en JavaScript?

- let x;
- var x;
- const x;

Pregunta 2/6

¿Cuál es la forma correcta de concatenar dos strings en JavaScript?

- str1 + str2;
- str1 .concat(str2);
- str1 .add(str2);

Pregunta 3/6

¿Cuál es la forma correcta de definir una función en JavaScript?

- function myFunction() {}
- myFunction() = function() {};
- myFunction() ⇒ {};

Pregunta 4/6

¿Qué palabra clave se utiliza para declarar una clase en JavaScript?

- class
- object
- prototype

Pregunta 5/6

¿Cómo se llama la propiedad que almacena el número de elementos en un array en JavaScript?

- length
- size
- count

Pregunta 6/6

¿Cuál es el operador lógico que representa la negación en JavaScript?

- ! 
- &&
- ||

Entregar

Conclusión

# Felicidades!

¡Felicitaciones por completar este curso! Has dado un paso importante para desbloquear todo tu potencial. Completar este curso no se trata solo de adquirir conocimientos; se trata de poner ese conocimiento en práctica y tener un impacto positivo en el mundo que te rodea.

Comparte este curso