PROGRAMACIÓN BÁSICA EN JAVASCRIPT

Módulo 2

OTEC PLATAFORMA 5 CHILE

MÓDULO 2

CLASE 6

Introducción

Objetos

Bracket Notation

Dot Notation

Modificación de Objetos

Objetos en Arreglos

For... in

Introducción

Tal vez haya escuchado que en JavaScript todo es un Objeto y es que los Objetos son tan importantes que surgió una rama entera de dedicada a ellos: la Programación Orientada a Objetos (POO, en inglés Object-Oriented Programming).

Cómo vió en la diapositiva anterior, en esta clase aprenderá sobre los Objetos y cómo acceder a ellos, cómo modificarlos, anidarlos y recorrerlos.

Sabemos que todo este Módulo es muy completo y desafiante, pero concéntrese en lo que debe aprender. No intente comprender más allá, ignore la complejidad.

Objetos

En JavaScript, un Objeto es una entidad independiente con Propiedades. A su vez, esas Propiedades tienen valores.

El concepto de Objetos puede compararse con entidades de la vida real. Por ejemplo, una persona representaría un Objeto con ciertas Propiedades: su color de pelo, cierta edad, la ciudad en la que vive, etc. Del mismo modo, los Objetos en JavaScript pueden tener Propiedades que definan sus características.

```
//Declaramos una variable que es
un Objeto porque tiene {}
let objeto = {
 //key:value,
 propiedad1: valor1,
 propiedad2: valor2,
};
```

La sintaxis está compuesta por:

- **Para declarar un Objeto, utilizamos llaves: { }.**
- **Los Objetos tienen Propiedades que se escriben en formato key-value pairs. Es decir, cada ítem viene de a pares. Y se separan por comas , .**

Las Propiedades de los Objetos suelen ser flexibles; por ejemplo, los valores incluyen números, palabras, otros Objetos, cadenas de caracteres e, incluso, Funciones. Pero, en caso de querer agregar números o dos (o más) palabras, deberás agregar un String.

Bracket Notation

Los valores guardados en los Objetos no tienen un orden (a diferencia de los Arreglos). Por lo tanto, no podemos acceder a ellos a través de un índice numérico.

Una de las formas de acceder es con Bracket Notation. Para ello, tiene que escribir el nombre de la variable y entre corchetes [], escribe como un string la propiedad a la cual desea acceder.

Importante: Si la propiedad a la que desea acceder es una función, para ejecutarla deberá colocar paréntesis () luego de los corchetes.

```
> let auto = {
    marca: "Volkswagen",
    modelo: "Gol",
    año: 2000,
    arranque: function () {
      console.log("El auto arrancó");
    },
  };
< undefined
> auto["marca"]
< 'Volkswagen'
> auto["modelo"]
< 'Gol'
> auto["año"]
< 2000
> auto["arranque"]()
  El auto arrancó
```

Dot Notation

Dot notation es una manera más fluida para acceder a los Objetos y da como resultado un código más simple de leer.

Su sintaxis está compuesta por la siguiente forma:

Primero debemos llamar a la variable seguida de un punto .

Luego escribiremos el nombre de la propiedad.

```
> auto
<· ▼ {marca: 'Volkswagen', modelo: 'Gol', año: 2000, arranque: f}
     ▶ arranque: f ()
       año: 2000
       marca: "Volkswagen"
       modelo: "Gol"
     ▶ [[Prototype]]: Object
> auto.marca
<· 'Volkswagen'
> auto.modelo
<· 'Gol'
> auto.año
<· 2000
> auto.arranque()
  El auto arrancó
```

VENTAJAS Y DESVENTAJAS

Aunque la forma Dot Notation es más simple y rápida, veremos que no es la mejor forma de acceder a las propiedades, ya que presenta limitaciones.

Modificación de propiedades de un Objeto

Las Propiedades de un Objeto se pueden modificar fácilmente. Para hacerlo, debemos acceder a esa Propiedad y asignarle un nuevo valor. Podemos utilizar tanto Bracket cómo Dot notation.

```
> misMascotas
< ▼ {perro: 'Federica', perro2: 'Benji', gato: 'Walter'} i
    gato: "Walter"
    perro: "Federica"
    perro2: "Benji"
  ▶ [[Prototype]]: Object
> misMascotas["perro2"]="Gorda";
< 'Gorda'
> misMascotas.gato="Luli";
< 'Luli'
> misMascotas
< ▼ {perro: 'Federica', perro2: 'Gorda', gato: 'Luli'} i
    gato: "Luli"
    perro: "Federica"
    perro2: "Gorda"
  ▶ [[Prototype]]: Object
```

Agregar propiedades a un Objeto

Además, si quisiéramos agregar Propiedades a un Objeto podemos hacerlo con ambos métodos. Debemos colocar la nueva Key y asignarle el valor que tendrá.

Algo a tener en cuenta, no se puede asignar más de 1 valor a cada propiedad. Pero si podemos crear una propiedad que como dato sea un objeto.

```
> misMascotas
<- ▼ {perro: 'Federica', perro2: 'Gorda', gato: 'Luli'} i
     gato: "Luli"
     perro: "Federica"
     perro2: "Gorda"
   ▶ [[Prototype]]: Object
> misMascotas.perro3="Benji";
<- 'Benji'
> misMascotas["dinosaurio"]="Rex";
<- 'Rex'
> misMascotas
<- ▼ {perro: 'Federica', perro2: 'Gorda', gato: 'Luli', perro3: 'Benji', dinosa
   urio: 'Rex'} i
     dinosaurio: "Rex"
     gato: "Luli"
     perro: "Federica"
     perro2: "Gorda"
     perro3: "Benji"
   ▶ [[Prototype]]: Object
>
```

Objetos dentro de Arreglos

Los Objetos al ser una forma de colección de propiedades, podemos almacenar datos relacionados entre sí que son propios de la característica del Objeto.

Para acceder a ellos usamos las formas ya conocidas. Para refrescar un poco la memoria...

Cuando necesitamos acceder a un elemento que está dentro de un arreglo, utilizamos el índice y para acceder al objeto utilizamos Dot Notation o Bracket Notation.

```
> let alumnos = [
    { nombre: "Juanito", apellido: "García" },
    { nombre: "Maribel", apellido: "Suarez" },
    { nombre: "Catalina", apellido: "López" },
  ];
<· undefined
> alumnos[0]
<· ▶ {nombre: 'Juanito', apellido: 'García'}
> alumnos[1].nombre
<· 'Maribel'
> alumnos[2]["apellido"]
<· 'López'
```

Recorrer un Objeto con el Método *for... in*

Como se mencionó anteriormente, un objeto no puede recorrerse con un For Loop como si fuera un Arreglo ya que no se puede asignar un índice a cada propiedad. Es por eso que se implementa el iterador For in.

¿Qué podemos obtener al recorrer un Objeto?

Podemos obtener el dato que necesitemos... Ya sea la propiedad o el valor de cada propiedad.

```
coloresMuebles
▼ {cama: 'blanca', ropero: 'marrón', escritorio: 'beige'}
    cama: "blanca"
    escritorio: "beige"
    ropero: "marrón"
  ▶ [[Prototype]]: Object
```

```
for (let mueble in coloresMuebles){
  console.log (mueble);
};
```

```
cama
ropero
escritorio
```

```
coloresMuebles
▼ {cama: 'blanca', ropero: 'marrón', escritorio: 'beige'}
    cama: "blanca"
    escritorio: "beige"
    ropero: "marrón"
  ▶ [[Prototype]]: Object
```

```
for (let mueble in coloresMuebles){
  console.log (coloresMuebles[mueble]);
};
```

```
blanca
marrón
beige
```

MUY IMPORTANTE

Para recorrer un objeto y acceder a sus propiedades utilizamos Bracket Notation, ya que Dot notation no acepta Variables. Intentará buscar el texto propiedad de forma literal y no lo encontrará (porque no existe).

MÓDULO 2

GRACIAS

Aquí finaliza la clase n°6 del módulo 2

OTEC PLATAFORMA 5 CHILE