Javascript en front-end

Carrera Programador full-stack

¿Qué es un objeto?

Un objeto es una colección de datos relacionados y/o funcionalidad.

Posee variables y funciones, que se denominan propiedades y métodos.

Cada propiedad tiene un nombre y un valor.

Definir un objeto

```
let Persona = {
    nombre: "Lucas",
    apellido: "Martinez",
    edad: 47,
    sexo: "masculino",
    dni: 23.305.379
    casado: true,
    hijos:['Ana Martinez', 'Marcos Martinez']
}
```

PROPIEDAD

NOMBRE

VALOR

Acceder a propiedades

Para acceder a las propiedades de un objeto tenemos 2 opciones:

- Objeto.propiedad
- Objeto[propiedad]

Siguiendo el ejemplo anterior obtendremos lo siguiente:

console.log(Persona.nombre) ---> "Lucas"

console.log(Persona['dni']) ---> 23.305.379

Mutabilidad

Por defecto en JS los objetos son mutables, esto significa que además de obtener el valor de una propiedad de un objeto también podemos modificarlo:

Persona.nombre = 'Fernando'

console.log(Persona.nombre) ---> 'Fernando'

Referencia

Cuando definimos objetos en JS la variable que lo declara hace referencia al lugar de memoria donde se encuentra almacenado ese objeto.

Por lo tanto:

```
let Persona1 = {nombre: 'Fernando'};
let Persona2 = persona1;
let Persona3 = {nombre: 'Fernando'};

console.log(Persona1 === Persona2) ---> true
console.log(Persona1 === Persona3) ---> false

Persona1.nombre = 'Lucas';
console.log(Persona2.nombre) ---> 'Lucas'
```

Destructuring

En JS existe una expresión que nos permite desempaquetar valores de arrays u objetos en grupos de variables, permitiéndonos simplificar y crear código más legible.

Teniendo en cuenta el ejemplo anterior:

```
let { nombre, apellido, edad, sexo, dni, casado, hijos } = Persona;

console.log(nombre) ---> 'Lucas'
console.log(apellido) ---> 'Martinez'
console.log(edad) ---> 47
console.log(sexo) ---> 'masculino'
console.log(dni) ---> 23.305.379
console.log(casado) ---> true
console.log(hijos) ---> ['Ana Martinez', 'Marcos Martinez']
```

JSON

Carrera Programador full-stack

¿Qué es JSON?

Corresponde a las siglas de Java Script Object Notation.

Es un formato ligero para almacenar y transportar datos.

Se usa a menudo cuando los datos se envían desde un servidor a una página web.

Es "autodescriptivo" y fácil de entender.

Reglas de sintaxis

Su sintaxis es muy similar a la de los objetos en JS, la principal diferencia es que los nombres de las propiedades también van entre comillas:

```
{
    "empleados": [
        {"nombre": "Juan", "apellido": "Perez"},
        {"nombre": "Karina", "apellido": "Gonzalez"},
        {"nombre": "Luz", "apellido": "López"},
        {"nombre": "Marcos", "apellido": "Villalba"}
    ]
}
```

Convertir un JSON a un objeto JS

Un uso común de JSON es leer datos de un servidor web y mostrar los datos en una página web.

El formato JSON es sintácticamente idéntico al código para crear objetos JavaScript, gracias a esta similitud, es muy sencillo convertir datos JSON en objetos JavaScript nativos utilizando el método JSON.parse()

Convertir un JSON a un objeto JS

```
let data = `{
    "empleados": [
        {"nombre": "Juan", "apellido": "Perez"},
        {"nombre": "Karina", "apellido": "Gonzalez"},
        {"nombre": "Luz", "apellido": "López"},
        {"nombre": "Marcos", "apellido": "Villalba"}
    ]
}`

let processedData = JSON.parse(data);

console.log(processedData.empleados[0]);
 //{nombre: "Juan", apellido: "Perez"}
```

Convertir un objeto JS a JSON

De igual manera podemos convertir un objeto JS a formato JSON utilizando JSON.stringify()

```
let data = {
      empleados: [
            {nombre: "Juan", apellido: "Perez"},
            {nombre: "Karina", apellido: "Gonzalez"},
            {nombre: "Luz", apellido: "López"},
            {nombre: "Marcos", apellido: "Villalba"}
       ]
}

let dataJson = JSON.stringify(data);
```

Aplicación

Carrera Programador full-stack

Cards de personas

Vamos a aplicar lo aprendido, crearemos una página que muestre una card por cada persona que tengamos almacenada en un arreglo de objetos, mostrando dinámicamente su nombre, apellido, edad, sexo y correo electrónico.

En nuestro html solo crearemos un título y un div principal donde se crearán dinámicamente las cards correspondientes.

Además linkearemos el archivo css y el archivo js.

Cards de personas

Comenzaremos creando nuestro arreglo de objetos que contendrá a cada persona y sus propiedades:

```html
<body>
    <h1>Personas</h1>
    <div id="mainContainer"></div>
    <script type="text/javascript" src="main.js"></script>
</body>
```

```javascript
let personas = [
  {
    nombre: "Juan",
    apellido: "Perez",
    edad: 21,
    sexo: "masculino",
    email: "juan_perez@gmail.com",
  },
  {
    nombre: "Karina",
    apellido: "Gonzalez",
    edad: 35,
    sexo: "femenino",
    email: "karina_gonzalez@gmail.com",
  },
  {
    nombre: "Luz",
    apellido: "López",
    edad: 19,
    sexo: "femenino",
    email: "luz_lopez@gmail.com",
  },
  {
    nombre: "Marcos",
    apellido: "Villalba",
    edad: 47,
    sexo: "masculino",
    email: "marcos_villalba@gmail.com",
  },
];
```

Cards de personas

Declaramos un método donde crearemos cada card, en este crearemos un div que servirá a modo de card, le agregaremos una clase y crearemos también el título con el nombre y apellido de la persona.

```
let crearCard = (persona) => {
  let card = document.createElement("div");
  card.classList.add("card-persona");
  let title = document.createElement("h2");
  title.innerHTML = `${persona.nombre} ${persona.apellido}`;
  card.appendChild(title);
};
```

Cards de personas

Ahora crearemos de forma dinámica una lista con cada una de las propiedades de la persona en cuestión, con excepción del nombre y el apellido, que ya se encuentran en el título. Para esto emplearemos la instrucción for(... in ...) la cual itera sobre todas las propiedades enumerables de un objeto.

```
let listaDatos = document.createElement("ul");
for (key in persona) {
  if (key !== "nombre" && key !== "apellido") {
    let listItem = document.createElement("li");
    listItem.innerHTML = `${key}: ${persona[key]}`;
    listaDatos.appendChild(listItem);
  }
}
card.appendChild(listaDatos);
```

Cards de personas

El último paso es ejecutar la función crearCard pasando por parámetro cada una de las personas en nuestro arreglo, para ello agregaremos un eventListener al elemento window para el evento load.

```
window.addEventListener("load", () => {
  for (let i = 0; i < personas.length; i++) {
    crearCard(personas[i]);
  }
});
```

Cards de personas

¿Qué pasaría si nuestros datos estuvieran en formato JSON?

```
let data = `
    {
        "personas": [
          {
            "nombre": "Juan",
            "apellido": "Perez",
            "edad": 21,
            "sexo": "masculino",
            "email": "juan_perez@gmail.com"
          },
          {
            "nombre": "Karina",
            "apellido": "Gonzalez",
            "edad": 35,
            "sexo": "femenino",
            "email": "karina_gonzalez@gmail.com"
          },
          {
            "nombre": "Luz",
            "apellido": "López",
            "edad": 19,
            "sexo": "femenino",
            "email": "luz_lopez@gmail.com"
          },
          {
            "nombre": "Marcos",
            "apellido": "Villalba",
            "edad": 47,
            "sexo": "masculino",
            "email": "marcos_villalba@gmail.com"
          }
        ]
    }
`;
```

Cards de personas

Tendríamos que convertirlo a un objeto JS nativo antes de poder utilizarlo.

```
window.addEventListener("load", () => {
  let processedData = JSON.parse(data);
  for (let i = 0; i < processedData.personas.length; i++) {
    crearCard(processedData.personas[i]);
  }
});
```