CARRERA

"TECNOLOGÍA SUPERIOR EN
REDES Y TELECOMUNICACIONES"

TEMA: "Manejo de Arrays - Objetos"

Materia: **Programación Web**

Estudiante: **Edison Alejandro Guachán Iñiguez**

Aula: **B-302**

Docente: **Ing. Cristian Muñoz**

D.M. Quito, 21 de julio 2022

A) Introducción

JavaScript (**JS**) es un lenguaje de programación ligero, interpretado, o compilado justo-a-tiempo (just-in-time) con funciones de primera clase. Si bien es más conocido como un lenguaje de scripting (secuencias de comandos) para páginas web, y es usado en muchos entornos fuera del navegador, tal como Node.js, Apache CouchDB y Adobe Acrobat JavaScript es un lenguaje de programación basada en prototipos, multiparadigma, de un solo hilo, dinámico, con soporte para programación orientada a objetos, imperativa y declarativa (por ejemplo programación funcional).

Una página web es un documento HTML que es interpretado por los navegadores en forma gráfica, pero también permiten el acceso al código. El Modelo de Objetos del Documento (DOM) permite ver el mismo documento de otra manera, describiendo el contenido del documento como un conjunto de objetos que un programa Javascript puede actuar sobre ellos.

b) Cuerpo

¿Qué son las arrays en JavaScript?

En JavaScript, un arreglo es una variable que se utiliza para almacenar diferentes tipos de datos. Básicamente almacena diferentes elementos en una caja y luego se puede evaluar con la variable.

Declarando un arreglo:

```
let miCaja = [];     // Declaración inicial de arreglo en JS
```

Los arreglos pueden contener múltiples tipos de datos

```
let miCaja = ['hola', 1, 2, 3, true, 'hey'];
```

Los arreglos se pueden manipular usando varias acciones conocidas como **métodos**. Algunos de estos métodos nos permiten agregar, eliminar, modificar y hacer mucho más en los arreglos.

Es muy habitual el trabajo con Arreglos o Arrays en Javascript, en los desarrollos se tienen que acceder a dichos valores de estos, pero se necesita conocer los métodos que nos brinda Javascript, a continuación, se detalla algunos ejemplos como acceder e interactuar con Arrays en Javascript.

Método: .map()

Este método nos crea un nuevo array de datos por cada elemento que compone nuestro array.

Ejemplo:

Si tenemos ciertos kilos de azúcar para hacer torta de chocolate para cierta cantidad de personas, pero se nos pide que solo usemos la mitad de azúcar, tenemos que sacar cual es la mitad de azúcar que debemos emplear para preparar la torta, leemos los elementos del array con el método .map() y luego le pasamos la función Dividir() que contiene una división que debe realizar

```
var kilosazucar = [3, 6, 9, 12];
var cantidadmitad = kilosazucar.map(Dividir);

document.getElementById("contenedor").innerHTML = cantidadmitad;

function Dividir(valor) {
    return " " + valor / 2 +" Kg" ;
}

// Obtenemos la mitad de Azucar en Kgs
1.5 Kg, 3 Kg, 4.5 Kg, 6 Kg
```

Método: .filter()

Este método nos sirve para analizar y filtrar los datos de un array, estos nuevos datos que obtenemos al ser filtrados, los colocamos en un nuevo array,

Ejemplo:

Si en una empresa tenemos 7 días en donde se llevarán reuniones importantes, pero ya llego la quincena y entonces solo necesitamos calcular los días que habrá reuniones a partir del día 15 en adelante.

```
var elementos = [30, 48, 2 ];
var reducir = elementos.reduce(Calcular);

document.getElementById("contenedor").innerHTML = "La cantidad de personas que no entran a la cine son: " + reducir;

function Calcular(total, valor) {
    return total - valor;
}

// Obtenemos
La cantidad de personas que no entran a la sala del cine son: -20
```

Método: .every()

Con este método verificamos si todos los valores de un array cumplen una condición o función que le hayamos asignado.

Ejemplo:

si tenemos un stock de 4 postres en una dulcería, 23 tortas de chocolate, 31 gelatinas de fresa, 15 pie de manzanas y 18 flans y queremos que nuestro sistema o aplicación nos envíe una notificación cuando el stock este totalmente agotado, es decir con 0 productos.

```
var stock = [23, 31, 15, 18];
var revisar = stock.every(Verificar);

document.getElementById("contenedor").innerHTML = "El Stock de productos esta agotado: " + revisar;

function Verificar(valor) {
    return valor < 1;
}

// Obtenemos
El Stock de productos esta agotado: false

// Pero si nuestro array de stocks de productos es var stock = [0, 0, 0, 0] obtenemos
El Stock de productos esta agotado: true
```

Método: .some()

Este método nos devuelve un resultado si el array cumple alguna de las condiciones solamente.

Ejemplo:

Si el personal a cargo de realizar las entrevistas para el puesto de supervisor de ventas en una empresa necesita obtener al instante quienes tienen una enfermedad grave luego de que se hicieran el examen médico, usaremos los valores 0 y 1, es decir para una persona sana le asignaremos 0 y para una persona enferma le asignamos el valor 1

```
var donantesdesangre = [0, 0, 0, 0, 0];
var verificar = donantesdesangre.some(Revisar);

document.getElementById("contenedor").innerHTML = "Se encontro personas con enfermedad contagiosa: " + verificar;

function Revisar(valor) {
    return valor > 0;
}

// Obtenemos
Se encontro personas con enfermedad contagiosa: false

// Pero si en nuestro array tenemos al menos 1 persona con enfermedad contagiosa donantesdesangre = [0, 0, 1, 0, 0]; entonces Obtenemos
Se encontro personas con enfermedad contagiosa: true
```

Objetos en Javascript

Los objetos de JavaScript son grupos de pares **clave-valor**. Los valores pueden consistir en **propiedades** y **métodos**, y pueden contener todos los demás tipos de datos de JavaScript, como cadenas, números y booleanos.

Todos los objetos de JavaScript derivan del constructor del Objectprincipal.Object tiene muchos métodos incorporados útiles a los cuales podemos acceder y recurrir para

simplificar el trabajo con objetos individuales. A diferencia de los métodos Array.prototype, como sort() y reverse() que se usan en la instancia de Array, los métodos Object se usan directamente en el constructor Object y emplean la instancia del objeto como parámetro. Esto se conoce como método estático.

Métodos de objetos en JavaScript

Object.create()

El método Object.create() se usa para crear un nuevo objeto y vincularlo al prototipo de un objeto existente.

Ejemplo:

Podemos crear una instancia de objeto job y ampliarla para obtener un objeto más específico.

```
// Initialize an object with properties and methods
const job = {
    position: 'cashier',
    type: 'hourly',
    isAvailable: true,
    showDetails() {
        const accepting = this.isAvailable ? 'is accepting applications' : "is not currently acc

        console.log(`The ${this.position} position is ${this.type} and ${accepting}.`);
    }
};

// Use Object.create to pass properties
const barista = Object.create(job);

barista.position = "barista";
barista.showDetails();
```

Copy

```
Output
The barista position is hourly and is accepting applications.
```

Object.keys()

Object.keys() crea una matriz que contiene las claves de un objeto.

Podemos crear un objeto e imprimir la matriz de claves.

```
1  // Initialize an object
2  const employees = {
3      boss: 'Michael',
4      secretary: 'Pam',
5      sales: 'Jim',
6      accountant: 'Oscar'
7  };
8
9  // Get the keys of the object
10 const keys = Object.keys(employees);
11
12 console.log(keys);  [ 'boss', 'secretary', 'sales', 'accountant' ]

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
Quokka 'Untitled-1.js' (node: v16.17.1 - Electron)

Reveal in value explorer
[ 'boss', 'secretary', 'sales', 'accountant' ]
  at keys quokka.js:12:1
```

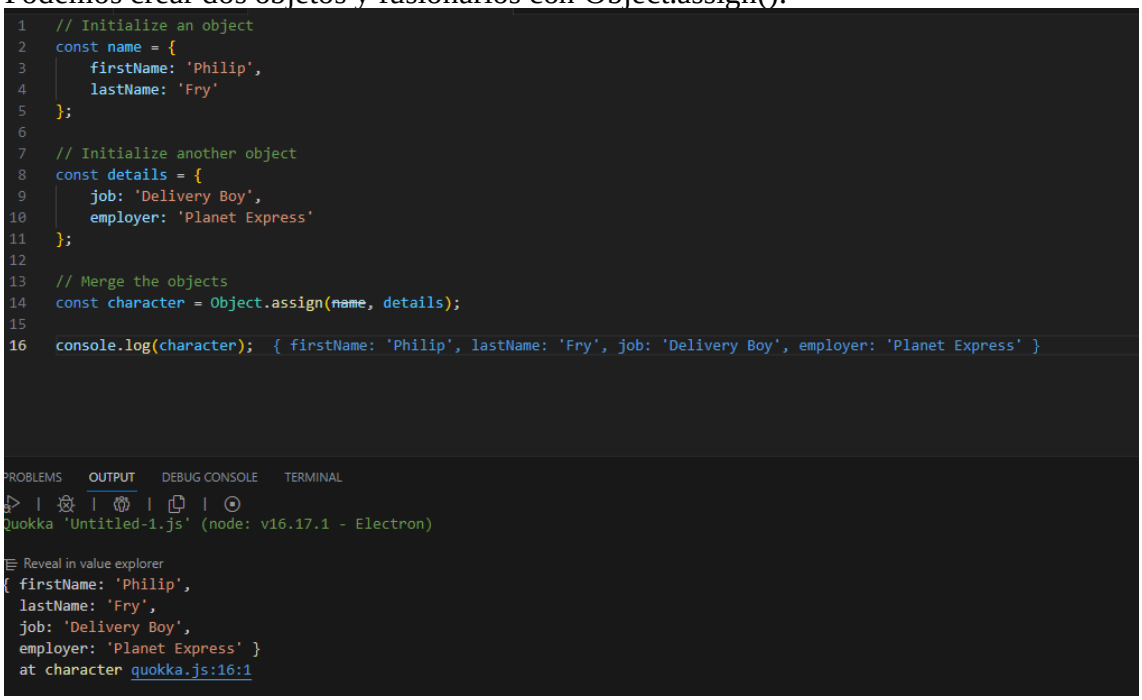
Object.entries()

Object.entries()crea una matriz anidada con los pares clave-valor de un objeto.

Object.assign()

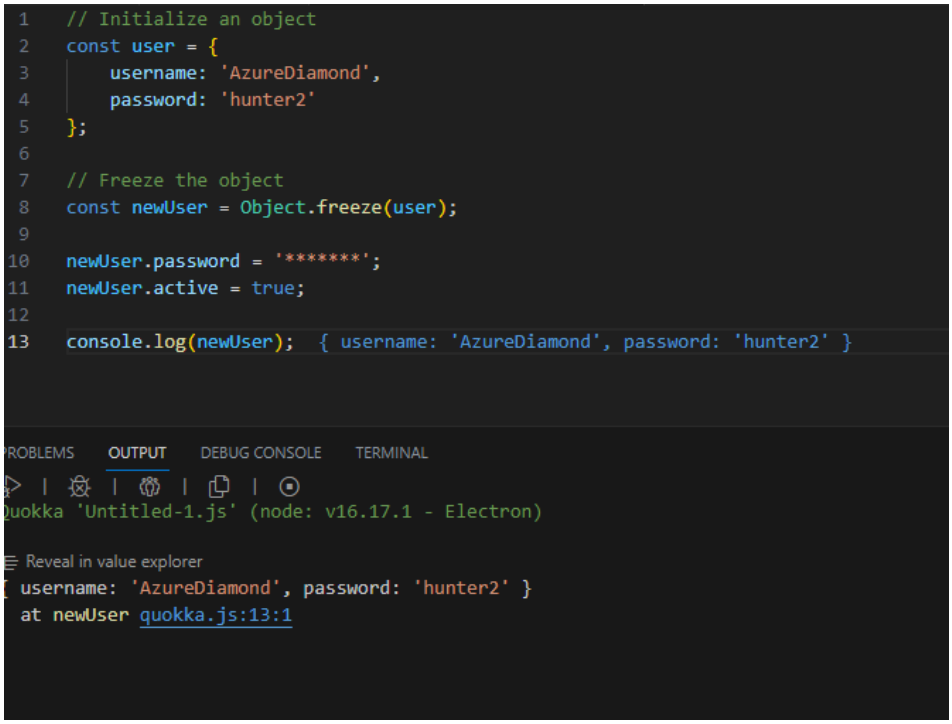
Object.assign() se usa para copiar valores de un objeto a otro.

Podemos crear dos objetos y fusionarlos con Object.assign().



Object.freeze()

Object.freeze() impide la modificación de propiedades y valores de un objeto, y evita que se agreguen propiedades a un objeto o que se eliminen de él.



Object.seal()

Object.seal() impide la adición de nuevas propiedades a un objeto, pero permite la modificación de propiedades existentes. Este método es similar a Object.freeze().

c) Conclusiones

- Un array es una forma ordenada de guardar un conjunto de elementos en una sola variable que nos permiten acceder a esos datos

- Los objetos tienen muchos métodos útiles que nos ayudan a modificarlos, protegerlos y generar repeticiones de ellos.

d) Referencias Bibliográficas

- *Arrays - Aprende Desarrollo Web | MDN*. (2023, 13 marzo). https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/Arrays
- Colectiva, N. (2018, 7 agosto). 5 Situaciones en donde los Métodos para Manipular Arrays en Javascript nos Ayudan. *nubecolectiva*. https://blog.nubecolectiva.com/5-situaciones-en-donde-los-metodos-para-manipular-arrays-en-javascript-nos-ayudan/
- Rascia, T. (2019). Cómo usar métodos de objetos en JavaScript. *DigitalOcean*. https://www.digitalocean.com/community/tutorials/como-usar-metodos-de-objetos-en-javascript-es