Universidad Tecnológica Metropolitana

DIVISIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN

División Tecnologías de la Información y Comunicación

Coordinación de Servicios de Tecnologías de la Información y Comunicación

Universidad Tecnológica Metropolitana

4F

ACTIVIDAD # 10-11

Nombre alumno:
Fernando David Sanchez Sacnhez

Nombre del profesor(a): Ruth Dominguez

Fecha de entrega: Mérida, Yucatán a martes 31 de octubre de 2024

Practica 11.

Descripción General

Este script gestiona una cola de clientes en una aplicación web, asignando un turno único a cada cliente que se registra y permitiendo la atención secuencial de los clientes en el orden de llegada. Implementa una interfaz simple con opciones para agregar un cliente a la cola, atender al siguiente en turno y salir del sistema, asegurando que solo usuarios logueados puedan acceder.

Variables Globales

- cola: Arreglo que almacena los datos de cada cliente en la cola.
- turno: Número de turno actual, se incrementa con cada nuevo cliente.
- LIMITE_COLA: Límite máximo de clientes que puede tener la cola al mismo tiempo (10 clientes).

Elementos del DOM

- agregarBtn: Botón para agregar un cliente a la cola.
- atenderBtn: Botón para atender al primer cliente de la cola.
- salirBtn: Botón para cerrar la sesión y salir del sistema.
- colaClientes: Elemento <tbody> que muestra la lista de clientes en la tabla de cola.
- turnoInput: Campo de entrada para mostrar el número de turno actual.
- nombreInput: Campo de entrada para el nombre del cliente.
- movimientoSelect: Campo de selección para el tipo de movimiento del cliente.
- frenteSpan: Elemento para mostrar el turno del primer cliente en la cola.
- finalSpan: Elemento para mostrar el turno del último cliente en la cola.

actualizarTurno()

```
function actualizarTurno() {
  turnoInput.value = turno;
}
```

Actualiza el campo turnoInput con el valor actual del número de turno.

actualizarFrenteFinal()

```
function actualizarFrenteFinal() {
  frenteSpan.textContent = cola.length > 0 ? cola[0].turno : '-';
  finalSpan.textContent = cola.length > 0 ? cola[cola.length - 1].turno : '-';
}
```

Actualiza los elementos frenteSpan y finalSpan para mostrar el turno del primer y último cliente en la cola respectivamente. Si la cola está vacía, muestra un guion ("-").

agregarCliente()

```
function agregarCliente() {
```

1. Verifica si la cola ha alcanzado el LIMITE_COLA. Si es así, muestra una alerta y detiene la función.
2. Obtiene el nombre y el tipo de movimiento del cliente desde los campos nombreInput y movimientoSelect.
3. Si algún campo está vacío, muestra una alerta pidiendo que se complete toda la información.
4. Crea un objeto cliente con el turno, nombre, movimiento y hora de llegada.
5. Agrega el cliente al arreglo cola, incrementa el turno y actualiza el campo de turno y la información del frente y final de la cola.
6. Muestra una alerta confirmando la formación del cliente en la cola.
7. Limpia los campos de entrada.

agregarClienteTabla(cliente)

```
function agregarClienteTabla(cliente) {
  const fila = colaClientes.insertRow();
  fila.insertCell(0).textContent = cliente.turno;
  fila.insertCell(1).textContent = cliente.nombre;
  fila.insertCell(2).textContent = cliente.movimiento;
  fila.insertCell(3).textContent = cliente.horaLlegada;
}
```

Agrega una fila en la tabla colaClientes con los datos del cliente (turno, nombre, movimiento, y hora de llegada).

atenderCliente()

```
function atenderCliente() {
  if (cola.length === 0) {
    alert('No hay clientes en la cola.');
    return;
  }
```

1. Comprueba si la cola está vacía. Si es así, muestra una alerta y detiene la función.
2. Extrae al primer cliente de la cola (cola.shift()).
3. Calcula el tiempo de espera llamando a calcularTiempoEspera().
4. Muestra una alerta con los detalles del cliente atendido.
5. Elimina la primera fila de la tabla colaClientes.
6. Actualiza los datos de frente y final de la cola.

calcularTiempoEspera(horaLlegada, horaAtencion)

```
function calcularTiempoEspera(horaLlegada, horaAtencion) {
  const [horaLleg, minLleg] = horaLlegada.split(':').map(Number);
  const [horaAtenc, minAtenc] = horaAtencion.split(':').map(Number);
  return (horaAtenc - horaLleg) * 60 + (minAtenc - minLleg);
}
```

Salida del Sistema

```
salirBtn.addEventListener('click', () => {
  if (window.confirm("¿Estás seguro de que deseas salir del sistema?")) {
    localStorage.removeItem('isLoggedIn');
    window.location.href = 'login.html';
  }
});
```

Practica11.

Variables Globales

- carQueue: Arreglo que almacena los coches en la cola, cada uno con un ID y un color.
- carColors: Arreglo de colores posibles para los coches, usado para asignar colores aleatorios a los nuevos coches.
- carsPainted: Contador de coches pintados correctamente.
- timeElapsed: Tiempo transcurrido en segundos desde que empezó el juego.
- gameInterval: Intervalo que controla la frecuencia de llegada de nuevos coches a la cola.
- selectedColor: Color seleccionado por el jugador para pintar el coche actual.

Elementos del DOM

- carQueueContainer: Contenedor donde se muestran los coches en la cola.
- paintCarButton: Botón para pintar el primer coche de la cola.
- recordDisplay: Elemento que muestra el número de coches pintados y el tiempo transcurrido.
- gameOverMessage y gameOverOverlay: Mensajes que aparecen cuando el juego termina.
- restartButton: Botón para reiniciar el juego.
- colorOptions: Elementos de la paleta de colores que el jugador puede seleccionar.

Funcionalidades

Selección de Color

```
const colorOptions = document.querySelectorAll('.color-option');
colorOptions.forEach(option => {
  option.addEventListener('click', function() {
    selectedColor = this.getAttribute('data-color');
    colorOptions.forEach(opt => opt.style.border = '2px solid black');
    this.style.border = '4px solid #0000ff'; // Indicar que este color está seleccionado
  });
});
```

Permite al jugador seleccionar un color de la paleta. El color seleccionado se almacena en selectedColor, y se indica el color activo visualmente con un borde azul.

addCarToQueue()

```
function addCarToQueue() {
  if (carQueue.length >= 5) {
    endGame();
    return;
  }
```

- Verifica si la cola ha alcanzado el límite de cinco coches. Si es así, llama a `endGame()` y detiene la función.
- Crea un nuevo coche con un ID único y un color aleatorio del arreglo `carColors`.
- Agrega el coche a `carQueue` y actualiza la vista de la cola llamando a `renderCars()`.

```
// Función para renderizar los coches en la cola
function renderCars() {
  carQueueContainer.innerHTML = '';
  carQueue.forEach(car => {
    const carDiv = document.createElement('div');
    carDiv.classList.add('car');
    carDiv.setAttribute('data-id', car.id);
```

paintCar()

```
// Función para pintar un coche
function paintCar() {
  if (carQueue.length === 0) {
    alert("No hay coches en la cola.");
    return;
  }

  const firstCar = carQueue[0];
```

1. Verifica si hay coches en la cola. Si está vacía, muestra una alerta.
2. Obtiene el primer coche de la cola (carQueue[0]) y compara su color con selectedColor.
3. Si el color coincide, elimina el coche de la cola, incrementa carsPainted y actualiza la vista y el contador llamando a renderCars() y updateRecord().
4. Si el color no coincide, muestra una alerta para que el jugador seleccione el color correcto.
5. Si carsPainted es múltiplo de 3, aumenta la velocidad del juego a tres segundos (reduce el intervalo de llegada de coches).

updateRecord()

```
// Función para actualizar el registro de coches pintados
function updateRecord() {
    timeElapsed++;
    recordDisplay.textContent = `Carros pintados: ${carsPainted} | Tiempo: ${timeElapsed} segundos`;
}
```

Actualiza el texto de recordDisplay, mostrando el número de coches pintados y el tiempo transcurrido en segundos.

endGame()

```
function endGame() {
    clearInterval(gameInterval);
    paintCarButton.disabled = true;
    gameOverOverlay.style.display = 'flex';
}
```

Detiene el intervalo de llegada de nuevos coches, deshabilita el botón paintCarButton y muestra el mensaje de "Game Over" en gameOverOverlay.

restartGame()

```
// Reiniciar el juego
function restartGame() {
  carQueue = [];
  carsPainted = 0;
  timeElapsed = 0;
  paintCarButton.disabled = false;
  gameOverOverlay.style.display = 'none';
  updateRecord();
  renderCars();
  gameInterval = setInterval(addCarToQueue, 7000); // Reiniciar el intervalo de 7 segundos
}
```