Taller Práctico: Estructuras de Control en JavaScript

Julian F. Latorre
Bootcamp Desarrollo Web Full Stack

Introducción

¡Bienvenidos al taller práctico de Estructuras de Control en JavaScript! En este taller, aplicaremos los conceptos aprendidos para crear proyectos divertidos y útiles. Cada ejercicio incluye un código base para guiarte, pero no dudes en ser creativo y expandir las funcionalidades.

Consejo

Recuerda: La práctica es la clave para dominar la programación. ¡Diviértete experimentando con el código!

1. Ejercicio 1: El Juego de la Adivinanza

Objetivo

Crear un juego interactivo donde el usuario debe adivinar un número aleatorio entre 1 y 100.

Código Base

```
function juegoAdivinanza() {
    const numeroSecreto = Math.floor(Math.random() * 100) + 1;
    let intentos = 0;
    let adivinado = false;

    while (!adivinado) {
        // Tu c digo aqu
    }
}

juegoAdivinanza();
```

Tareas

1. Implementa la lógica para solicitar al usuario que adivine el número.
2. Usa una estructura condicional para comprobar si el número es correcto, mayor o menor.
3. Incrementa el contador de intentos en cada intento.
4. Muestra un mensaje de felicitación cuando se adivine el número, incluyendo el número de intentos.

2. Ejercicio 2: Generador de Patrones

Objetivo

Crear un programa que genere patrones de asteriscos basados en la entrada del usuario.

Código Base

```
function generadorPatrones(altura) {
    for (let i = 1; i <= altura; i++) {
        let linea = '';
        // Tu c digo aqu
        console.log(linea);
    }
}

generadorPatrones(5);
```

Tareas

1. Implementa la lógica para generar un triángulo de asteriscos.
2. Añade una opción para generar diferentes tipos de patrones (por ejemplo, cuadrado, pirámide invertida).
3. Usa estructuras de control anidadas para crear patrones más complejos.

3. Ejercicio 3: Calculadora de Estadísticas

Objetivo

Desarrollar una calculadora que analice un conjunto de números y proporcione estadísticas básicas.

Código Base

```
function calculadoraEstadisticas(numeros) {
    let suma = 0;
    let maximo = numeros[0];
    let minimo = numeros[0];

    for (let i = 0; i < numeros.length; i++) {
        // Tu c digo  aqu
    }

    // Calcular promedio, etc.

    return {
        suma: suma,
        promedio: promedio,
        maximo: maximo,
        minimo: minimo
    };
}

const numeros = [23, 45, 12, 67, 89, 34];
console.log(calculadoraEstadisticas(numeros));
```

Tareas

1. Implementa la lógica para calcular la suma, el promedio, el máximo y el mínimo.
2. Usa estructuras condicionales para determinar el máximo y el mínimo.
3. Agrega una función para calcular la mediana del conjunto de números.
4. Implementa una opción para filtrar números pares o impares antes de calcular las estadísticas.

4. Ejercicio 4: Conversor de Temperatura

Objetivo

Crear un programa que convierta temperaturas entre Celsius, Fahrenheit y Kelvin.

Código Base

```
function convertirTemperatura(valor, unidadOrigen, unidadDestino) {
    let resultado;

    switch (unidadOrigen) {
        case 'C':
```

```
            // Convertir desde Celsius
            break;
        case 'F':
            // Convertir desde Fahrenheit
            break;
        case 'K':
            // Convertir desde Kelvin
            break;
        default:
            return "Unidad de origen no v lida";
    }

    return resultado;
}

console.log(convertirTemperatura(25, 'C', 'F'));
console.log(convertirTemperatura(98.6, 'F', 'C'));
console.log(convertirTemperatura(300, 'K', 'C'));
```

Tareas

1. Implementa la lógica de conversión para cada caso en la estructura switch.
2. Agrega manejo de errores para unidades no válidas.
3. Crea una interfaz de usuario simple que permita al usuario ingresar el valor y seleccionar las unidades.
4. Implementa una función para redondear los resultados a dos decimales.

5. Ejercicio 5: Generador de Contraseñas

Objetivo

Desarrollar un generador de contraseñas que permita al usuario especificar la longitud y los tipos de caracteres a incluir.

Código Base

```
function generarContrasena(longitud, incluirMayusculas,
    incluirNumeros, incluirSimbolos) {
    const letrasMinusculas = 'abcdefghijklmnopqrstuvwxyz';
    const letrasMayusculas = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
    const numeros = '0123456789';
    const simbolos = '!@#$%^&*()_+-=[]{}|;:,.<>?';

    let caracteresPermitidos = letrasMinusculas;
    let contrasena = '';

    if (incluirMayusculas) caracteresPermitidos += letrasMayusculas
    ;
```

```
    if (incluirNumeros) caracteresPermitidos += numeros;
    if (incluirSimbolos) caracteresPermitidos += simbolos;

    for (let i = 0; i < longitud; i++) {
        // Generar car cter aleatorio
    }

    return contrasena;
}

console.log(generarContrasena(12, true, true, true));
```

Tareas

1. Implementa la lógica para generar caracteres aleatorios y construir la contraseña.
2. Agrega una verificación para asegurar que la contraseña incluya al menos un carácter de cada tipo solicitado.
3. Crea una función que evalúe la fortaleza de la contraseña generada.
4. Implementa una interfaz de usuario que permita al usuario especificar los parámetros de la contraseña.

6. Desafío Final: Mini Juego de Rol

Objetivo

Crear un mini juego de rol por turnos que utilice varias estructuras de control.

Código Base

```
const jugador = {
    nombre: "H roe",
    salud: 100,
    ataque: 10,
    defensa: 5
};

const enemigo = {
    nombre: "Drag n",
    salud: 150,
    ataque: 15,
    defensa: 8
};

function turno(atacante, defensor) {
    // L gica del turno
```

```
}

function juego() {
    while (jugador.salud > 0 && enemigo.salud > 0) {
        // L gica del juego
    }

    // Determinar ganador
}

juego();
```

Tareas

1. Implementa la lógica de los turnos, incluyendo cálculo de daño y probabilidad de acierto.
2. Agrega diferentes tipos de acciones (ataque, defensa, usar poción) utilizando una estructura switch.
3. Implementa un sistema de niveles y experiencia utilizando bucles y condicionales.
4. Crea una interfaz de usuario simple que muestre el estado del juego y permita al jugador elegir acciones.
5. Agrega elementos aleatorios (críticos, evasiones) utilizando Math.random() y estructuras condicionales.

7. Conclusión

¡Felicidades por completar el taller práctico de Estructuras de Control en JavaScript! Estos ejercicios te han permitido aplicar y reforzar tus conocimientos sobre condicionales y bucles en situaciones prácticas y divertidas.

Próximos Pasos

- Experimenta modificando y expandiendo estos ejercicios.
- Intenta combinar diferentes estructuras de control en proyectos más complejos.
- Comparte tus soluciones con tus compañeros y aprende de sus enfoques.

Recuerda, la clave para dominar la programación es la práctica constante y la curiosidad por aprender. ¡Sigue codificando y divirtiéndote!