INSTITUTO KHIPU

Semestre IV

Sesión 11

PROGRAMA DE ESTUDIOS

DESARROLLO DE SISTEMAS DE INFORMACIÓN

COMUNITY MANAGER Y MARKETING DIGITAL

Tema

UTILIZAR JAVASCRIPT PARA MODIFICAR EL DOM EN RESPUESTA A EVENTOS DEL USUARIO

Utilizar JavaScript para Modificar el DOM en Respuesta a Eventos del Usuario

1. ¿Qué es el DOM? Relación con HTML y el Navegador

El **DOM (Document Object Model)** es una representación estructurada en forma de árbol de los elementos HTML de una página web. JavaScript puede interactuar con el DOM para modificar contenido, estilos y estructura de manera dinámica.

📌 **Ejemplo de estructura del DOM:**

```
<!DOCTYPE html>
<html>
<head>
    <title>Ejemplo de DOM</title>
</head>
<body>
    <h1 id="titulo">Hola, Mundo</h1>
    <button id="boton">Haz clic</button>
</body>
</html>
```

- `<html>` es el nodo raíz.
- `<head>` y `<body>` son nodos hijos.
- `<h1>` y `<button>` son elementos dentro de `<body>`.
- Se pueden seleccionar y modificar usando JavaScript.

2. Métodos de Selección de Elementos

Para modificar elementos del DOM, primero debemos seleccionarlos.

2.1 `document.getElementById()`

Permite seleccionar un elemento por su atributo `id`.

📌 **Ejemplo:**

```
let titulo = document.getElementById("titulo");
titulo.textContent = "Nuevo Título";
```

2.2 `document.querySelector()`

Selecciona el primer elemento que coincida con un selector CSS.

📌 **Ejemplo:**

```
let boton = document.querySelector("#boton");
boton.style.backgroundColor = "blue";
```

2.3 `document.querySelectorAll()`

Selecciona todos los elementos que coincidan con un selector CSS, devolviendo un **NodeList** (similar a un array).

📌 **Ejemplo:**

```
let elementos = document.querySelectorAll("p");
elementos.forEach(el => el.style.color = "green");
```

3. Modificación de Elementos

3.1 Cambiar contenido con `innerHTML` y `textContent`

- **`innerHTML`**: Permite insertar contenido HTML dentro de un elemento.

- **`textContent`**: Solo cambia el texto, sin interpretar etiquetas HTML.

📌 **Ejemplo:**

```
let titulo = document.getElementById("titulo");
titulo.innerHTML = "<strong>Nuevo contenido con HTML</strong>";
let titulo = document.getElementById("titulo");
titulo.textContent = "Nuevo contenido sin HTML";
```

3.2 Modificar estilos con `style`

Podemos cambiar propiedades CSS directamente.

📌 **Ejemplo:**

```
let boton = document.getElementById("boton");
boton.style.backgroundColor = "red";
boton.style.color = "white";
```

3.3 Agregar y remover clases con `classList.add()` y `classList.remove()`

📌 **CSS:**

```
.destacado {
    font-size: 20px;
    color: blue;
    background-color: yellow;
}
```

📌 **JavaScript:**

```
let titulo = document.getElementById("titulo");
titulo.classList.add("destacado"); // Agrega la clase "destacado"
```

```
titulo.classList.remove("destacado"); // Elimina la clase "destacado"
```

4. Demostración Práctica

4.1 Creación de un botón en HTML y cambio de su color al hacer clic

📌 **HTML:**

```
<button id="cambiarColor">Cambiar Color</button>
```

📌 **JavaScript:**

```
document.getElementById("cambiarColor").addEventListener("click",
function() {
    this.style.backgroundColor = "purple";
});
```

4.2 Ejemplo de actualización de contenido dinámico con `innerHTML`

📌 **HTML:**

```
<p id="mensaje">Texto original</p>
<button id="actualizar">Actualizar Texto</button>
```

📌 **JavaScript:**

```
document.getElementById("actualizar").addEventListener("click",
function() {
    document.getElementById("mensaje").innerHTML = "<strong>Texto
actualizado dinámicamente</strong>";
});
```

4.3 Mostrar cómo ocultar y mostrar elementos con `display: none`

📌 **HTML:**

```
<p id="parrafo">Este es un párrafo visible</p>
<button id="ocultarMostrar">Ocultar/Mostrar</button>
```

📌 **JavaScript:**

```
document.getElementById("ocultarMostrar").addEventListener("click",
function() {
    let parrafo = document.getElementById("parrafo");
    if (parrafo.style.display === "none") {
        parrafo.style.display = "block";
    } else {
        parrafo.style.display = "none";
    }
});
```

Conclusión

- JavaScript permite modificar el DOM en respuesta a eventos del usuario.
- Se pueden seleccionar elementos con `getElementById()`, `querySelector()` y `querySelectorAll()`.
- Se puede modificar contenido con `innerHTML` y `textContent`, así como cambiar estilos y clases dinámicamente.
- Con eventos como `click`, es posible actualizar el contenido, cambiar colores y mostrar u ocultar elementos.

INSTITUTO

KHIPU