Programador JavaScript Avanzado

Unidad 1: Conceptos principales de JS

Contactos: tecnologia@elearning-total.com
Web: www.elearning-total.com

Indice

Unidad 1: Conceptos principales de JS

Objetivos

Que el alumno logre:

- Crear y aplicar funciones desarrolladas con Javascript

DOM

DOM significa Document Object Model (Modelo de Objetos del Documento, en español) y es una interfaz de programación para documentos HTML y XML.

En JavaScript, el DOM se refiere a la estructura de objetos que representa el contenido de un documento web y cómo los objetos pueden interactuar entre sí para manipular ese contenido. Con el DOM, los desarrolladores web pueden escribir scripts en JavaScript que accedan y modifiquen el contenido de una página web en tiempo real.

El DOM proporciona una serie de objetos predefinidos que representan elementos HTML, como botones, formularios, imágenes, etc. Estos objetos también tienen propiedades y métodos que permiten a los desarrolladores interactuar con ellos y manipular su contenido. Por ejemplo, mediante el uso del DOM, es posible agregar o eliminar elementos HTML, cambiar el estilo de un elemento, mover elementos en una página web, etc.

En resumen, el DOM es una herramienta fundamental en el desarrollo de aplicaciones web interactivas, ya que permite que el contenido de una página web sea dinámico y responda a la interacción del usuario.

Podemos utilizar DOM en JavaScript para interactuar con el contenido de una página web:

Cambiar el texto de un elemento HTML:

```
<!-- HTML -->
<p id="miTexto">Este es un párrafo dentro de mi archivo HTML.</p>

// JavaScript
const texto = document.getElementById("miTexto");
texto.textContent = "Este es un nuevo texto para el párrafo.";
```

Este código selecciona el elemento p con el id "miTexto" del documento HTML y cambia su texto a "Este es un nuevo texto para el párrafo." utilizando la propiedad textContent.

También podemos utilizar DOM para agregar un nuevo elemento HTML:

```
<!-- HTML -->
<ul id="miLista">
  <li>Primer Item</li>
  <li>Segundo Item</li>
  <li>Tercer Item</li>
</ul>
```

```
// JavaScript
const lista = document.getElementById("miLista");
const nuevoElemento = document.createElement("li");
nuevoElemento.textContent = "Cuarto Item";
lista.appendChild(nuevoElemento);
```

Este código selecciona la lista ul con el id "miLista" del documento HTML y crea un nuevo elemento li con el texto "Elemento 4". Luego, utiliza el método appendChild para agregar el nuevo elemento a la lista.

También podemos cambiar o agregar cualquier propiedad de CSS de un elemento HTML:

```
<!-- HTML -->
<a id="miBoton">Clickeame</a>

// JavaScript
const boton = document.getElementById("miBoton");
boton.style.backgroundColor = "red";
boton.style.color = "white";
```

Este código selecciona el botón con el id "miBoton" del documento HTML y cambia su color de fondo a rojo y su color de texto a blanco utilizando la propiedad style.

BOM

BOM significa Browser Object Model (Modelo de Objetos del Navegador, en español) y es una interfaz de programación para interactuar con el navegador web.

En JavaScript, el BOM se refiere a la estructura de objetos que representan el navegador y su ventana, incluyendo objetos como la ventana del navegador, la barra de direcciones, el historial de navegación, las cookies, entre otros.

El BOM proporciona objetos y métodos para interactuar con la ventana del navegador, como abrir y cerrar ventanas, cambiar la ubicación de la ventana, establecer y leer cookies, etc. También proporciona información sobre el usuario, como la resolución de pantalla, el navegador utilizado y su versión, entre otros.

A diferencia del DOM, el BOM no es un estándar y puede variar entre diferentes navegadores web. Algunos objetos y métodos del BOM pueden no ser compatibles con todos los navegadores, por lo que es importante tener en cuenta las diferencias entre ellos al escribir scripts en JavaScript.

En resumen, el BOM es una herramienta importante para interactuar con el navegador web y obtener información sobre el usuario, lo que permite crear aplicaciones web más interactivas y personalizadas.

Usos

Podemos usar el BOM en JavaScript para interactuar con el navegador web.

Por ejemplo, para obtener una URL:

```
// JavaScript
const urlActual = window.location.href;
console.log(urlActual);
```

Este código utiliza el objeto window del BOM para obtener la URL actual de la página web y la muestra en la consola.

También podemos identificar y manipular las ventanas del navegador:

```
// JavaScript
const nuevaVentana = window.open("https://www.google.com", "_blank",
"width=500,height=500");
```

Este código utiliza el método open del objeto window para abrir una nueva ventana del navegador con la URL "https://www.google.com" y las dimensiones de ancho y alto especificadas. La opción "_blank" indica que la ventana debe abrirse en una nueva pestaña.

Obtener el ancho y alto de la ventana del navegador:

```
// JavaScript
const anchoVentana = window.innerWidth;
const altoVentana = window.innerHeight;
console.log("Ancho de la ventana: " + anchoVentana);
console.log("Alto de la ventana: " + altoVentana);
```

Este código utiliza las propiedades innerWidth e innerHeight del objeto window para obtener el ancho y alto de la ventana del navegador y los muestra en la consola.

Objeto Window

El objeto **window** representa la ventana completa del navegador. Mediante este objeto, es posible mover, redimensionar y manipular la ventana actual del navegador. Incluso es posible abrir y cerrar nuevas ventanas de navegador.

Si una página emplea frames, cada uno de ellos se almacena en el array frames, que puede ser accedido numéricamente (window.frames[0]) o, si se ha indicado un nombre al frame, mediante su nombre (window.frames["nombre del frame"]).

Como todos los demás objetos heredan directa o indirectamente del objeto window, no es necesario indicarlo de forma explícita en el código JavaScript. En otras palabras:

```
window.frames[0] == frames[0]
window.document == document
```

BOM define cuatro métodos para manipular el tamaño y la posición de la ventana:

- *moveBy(x, y)* desplaza la posición de la ventana x píxel hacia la derecha y y píxel hacia abajo. Se permiten desplazamientos negativos para mover la ventana hacia la izquierda o hacia arriba.
- *moveTo(x, y)* desplaza la ventana del navegador hasta que la esquina superior izquierda se encuentre en la posición (x, y) de la pantalla del usuario. Se permiten desplazamientos negativos, aunque ello suponga que parte de la ventana no se visualiza en la pantalla.

- *resizeBy(x, y)* redimensiona la ventana del navegador de forma que su nueva anchura sea igual a (anchura_anterior + x) y su nueva altura sea igual a (altura_anterior + y). Se pueden emplear valores negativos para reducir la anchura y/o altura de la ventana.
- *resizeTo(x, y)* redimensiona la ventana del navegador hasta que su anchura sea igual a x y su altura sea igual a y. No se permiten valores negativos.

Los navegadores son cada vez menos permisivos con la modificación mediante JavaScript de las propiedades de sus ventanas. De hecho, la mayoría de navegadores permite a los usuarios bloquear el uso de JavaScript para realizar cambios de este tipo. De esta forma, una aplicación nunca debe suponer que este tipo de funciones están disponibles y funcionan de forma correta.

A continuación se muestran algunos ejemplos de uso de estas funciones:

```
// Mover la ventana 20 píxel hacia la derecha y 30 píxel hacia abajo
window.moveBy(20, 30);

// Redimensionar la ventana hasta un tamaño de 250 x 250
window.resizeTo(250, 250);

// Agrandar la altura de la ventana en 50 píxel
window.resizeBy(0, 50);

// Colocar la ventana en la esquina izquierda superior de la ventana
window.moveTo(0, 0);
```

Objeto document

El objeto document es el único que pertenece tanto al DOM (como se vio en el capítulo anterior) como al BOM. Desde el punto de vista del BOM, el objeto document proporciona información sobre la propia página HTML.

Objeto location

El objeto location es uno de los objetos más útiles del BOM. Debido a la falta de estandarización, locationes una propiedad tanto del objeto window como del objeto document.

El objeto location representa la URL de la página HTML que se muestra en la ventana del navegador y proporciona varias propiedades útiles para el manejo de la URL.

De todas las propiedades, la más utilizada es location.href, que permite obtener o establecer la dirección de la página que se muestra en la ventana del navegador.

Además de las propiedades de la tabla anterior, el objeto location contiene numerosos métodos y funciones. Algunos de los métodos más útiles son los siguientes:

```
// Método assign()
location.assign("http://www.ejemplo.com");  // Equivalente a location.href =
"http://www.ejemplo.com"

// Método replace()
location.replace("http://www.ejemplo.com");
// Similar a assign(), salvo que se borra la página actual del array history del navegador

// Método reload()
location.reload(true);
/* Recarga la página. Si el argumento es true, se carga la página desde el servidor.
  Si es false, se carga desde la cache del navegador */
```

Objeto Navigator

El objeto navigator es uno de los primeros objetos que incluyó el BOM y permite obtener información sobre el propio navegador. En Internet Explorer, el objeto navigator también se puede acceder a través del objeto clientInformation.

Aunque es uno de los objetos menos estandarizados, algunas de sus propiedades son comunes en casi todos los navegadores.

El objeto **navigator** se emplea habitualmente para detectar el tipo y/o versión del navegador en las aplicaciones cuyo código difiere para cada navegador. Además, se emplea para detectar si el navegador tiene habilitadas las cookies y Java y también para comprobar los plugins disponibles en el navegador.

Objeto screen

El objeto screen se utiliza para obtener información sobre la pantalla del usuario. Uno de los datos más importantes que proporciona el objeto screen es la resolución del monitor en el que se están visualizando las páginas. Los diseñadores de páginas web necesitan conocer las resoluciones más utilizadas por los usuarios para adaptar sus diseños a esas resoluciones.

La altura/anchura de pantalla disponible para las ventanas es menor que la altura/anchura total de la pantalla, ya que se tiene en cuenta el tamaño de los elementos del sistema operativo como por ejemplo la barra de tareas y los bordes de las ventanas del navegador.

Además de la elaboración de estadísticas de los equipos de los usuarios, las propiedades del objeto screense utilizan por ejemplo para determinar cómo y cuánto se puede redimensionar una ventana y para colocar una ventana centrada en la pantalla del usuario.

El siguiente ejemplo redimensiona una nueva ventana al tamaño máximo posible según la pantalla del usuario:

```
window.moveTo(0, 0);
window.resizeTo(screen.availWidth, screen.availHeight);
```

Resumen

En esta Unidad…

Trabajamos con la Introducción a JavaScript

En la próxima Unidad…

Trabajaremos con variables y operadores