Capacitarte

Material Imprimible

Curso de JavaScript

Módulo 1

JavaScript

JavaScript es un robusto lenguaje de programación que se puede aplicar a un documento HTML y usarse para crear interactividad dinámica en los sitios web. Fue inventado por Brendan Eich, cofundador del proyecto Mozilla, Mozilla Foundation y la Corporación Mozilla. Puedes hacer casi cualquier cosa con JavaScript. Puedes empezar con pequeñas cosas como carruseles, galerías de imágenes, diseños fluctuantes, y respuestas a las pulsaciones de botones. Con más experiencia, serás capaz de crear juegos, animaciones 2D y gráficos 3D, aplicaciones integradas basadas en bases de datos ¡y mucho más!

JavaScript por sí solo es bastante compacto, aunque muy flexible, y los desarrolladores han escrito gran cantidad de herramientas encima del núcleo del lenguaje JavaScript, desbloqueando una gran cantidad de funcionalidad adicional con un mínimo esfuerzo. Esto incluye:

- Interfaces de Programación de Aplicaciones del Navegador (APIs) — APIs construidas dentro de los navegadores que ofrecen funcionalidades como crear dinámicamente contenido HTML y establecer estilos CSS, hasta capturar y manipular un vídeo desde la cámara web del usuario, o generar gráficos 3D y muestras de sonido.
- APIs de terceros, que permiten a los desarrolladores incorporar funcionalidades en sus sitios de otros proveedores de contenidos como Twitter o Facebook.
- Marcos de trabajo y librerías de terceros que puedes aplicar a tu HTML para que puedas construir y publicar rápidamente sitios y aplicaciones.

¿Por qué estudiar JavaScript?

JavaScript es uno de los **3 idiomas que** todos los desarrolladores web **deben** aprender:

1. **HTML** para definir el contenido de las páginas web

2. **CSS** para especificar el diseño de las páginas web
3. **JavaScript** para programar el comportamiento de las páginas web

Las páginas web no son el único lugar donde se usa JavaScript. Muchos programas de escritorio y servidor usan JavaScript. Node.js es el más conocido (Node.js es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor basado en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google). Algunas bases de datos, como MongoDB y CouchDB, también usan JavaScript como lenguaje de programación.

Introducción a JavaScript

JavaScript puede cambiar el contenido HTML

Uno de los muchos métodos HTML JavaScript es getElementById().

Este ejemplo utiliza el método para "encontrar" un elemento HTML (con id = "demo") y cambia el contenido del elemento (innerHTML) a "Hola JavaScript":

```
<!DOCTYPE html>
<html>
<body>

<h2> ¿Qué puede hacer JavaScript?</h2>

<p id="demo">
JavaScript puede cambiar el contenido HTML.</p>

<button type="button" onclick='document.getElementById("demo").innerHTML = "Hola JavaScript!"'>Click Me!</button>

</body>
</html>
```

JavaScript acepta comillas dobles y simples

```
<!DOCTYPE html>
<html>
```

```
<body>

<h2>What Can JavaScript Do?</h2>

<p id="demo">JavaScript can change HTML content.</p>

<button type="button" onclick="document.getElementById('demo').innerHTML = 'Hello
JavaScript!'">Click Me!</button>

</body>
</html>
```

JavaScript puede cambiar los valores de los atributos HTML

En este ejemplo, JavaScript cambia el valor del `src`atributo (fuente) de una `<img>`etiqueta:

```
<!DOCTYPE html>
<html>
<body>

<h2>¿Que puede hacer JavaScript?</h2>

<p>
JavaScript puede cambiar los valores de los atributos HTML.</p>

<p>
En este caso, JavaScript cambia el valor del atributo src (fuente) de una imagen.</p>

<button onclick="document.getElementById('myImage').src='apagada.jpg'">
Apaga la luz</button>

<img id="myImage" src="apagada.jpg" style="width:100px">
```

<button onclick="document.getElementById('myImage').src='encendida.jpg'">
Enciend la luz</button>

</body>
</html>

JavaScript puede cambiar los estilos HTML (CSS)

Cambiar el estilo de un elemento HTML es una variante de cambiar un atributo HTML:

<! DOCTYPE html>
<html>
<body>

<h2> ¿Qué puede hacer JavaScript? </h2>

 JavaScript puede cambiar el estilo de un elemento HTML.

<button type = "button" onclick = "document.getElementById ('demo'). style.fontSize = '35px'">
¡Haga clic aqui! </button>

</body>
</html>

JavaScript puede ocultar elementos HTML

Es posible ocultar elementos HTML cambiando el displayestilo:

<! DOCTYPE html>
<html>
<body>

<h2> ¿Qué puede hacer JavaScript? </h2>

 JavaScript puede ocultar elementos HTML.

<button type = "button" onclick = "document.getElementById ('demo'). style.display = 'none'">
¡Haga clic aqui! </button>

</body>
</html>

JavaScript puede mostrar elementos HTML

También se puede mostrar elementos HTML ocultos cambiando el displayestilo:

<! DOCTYPE html>
<html>
<body>

<h2> ¿Qué puede hacer JavaScript? </h2>

 JavaScript puede mostrar elementos HTML ocultos.

 ¡Hola, JavaScript!

<button type = "button" onclick = "document.getElementById ('demo'). style.display = 'block'">
¡Haga clic aqui! </button>

</body>
</html>

JavaScript a dónde

La etiqueta <script>

En HTML, el código JavaScript se inserta entre <script>y las </script>etiquetas.

<! DOCTYPE html>
<html>
<body>

```
<h2> JavaScript en el cuerpo </h2>

<p id = "demo"> </p>

<script>
document.getElementById ("demo"). innerHTML = "Mi primer JavaScript";
</script>

</body>
</html>
```

Funciones y eventos de JavaScript

Un JavaScript function es un bloque de código JavaScript, que se puede ejecutar cuando se le "solicita".

Por ejemplo, se puede llamar a una función cuando ocurre un **evento**, como cuando el usuario hace clic en un botón.

JavaScript en <head> o <body>

Puede colocar cualquier número de scripts en un documento HTML.

Los scripts se pueden colocar en <body>, o en la <head>sección de una página HTML, o en ambos.

JavaScript en <head>

En este ejemplo, functionse coloca un JavaScript en la <head>sección de una página HTML.

La función se invoca (se llama) cuando se hace clic en un botón:

```
<! DOCTYPE html>
<html>
<head>
<script>
function myFunction () {
 document.getElementById ("demo"). innerHTML = "Párrafo cambiado";
```

```
}
</script>
</head>
<body>

<h2> JavaScript en Head </h2>

<p id = "demo"> Un párrafo. </p>

<button type = "button" onclick = "myFunction ()"> Pruébelo </button>

</body>
</html>
```

JavaScript en <body>

En este ejemplo, functionse coloca un JavaScript en la <body>sección de una página HTML.

La función se invoca (se llama) cuando se hace clic en un botón:

```
<! DOCTYPE html>
<html>
<body>

<h2> JavaScript en el cuerpo </h2>

<p id = "demo"> Un párrafo. </p>

<button type = "button" onclick = "myFunction ()"> Pruébelo </button>

<script>
function myFunction () {
  document.getElementById ("demo"). innerHTML = "Párrafo cambiado";
}
```

```
</script>

</body>
</html>
```

JavaScript externo

Las secuencias de comandos también se pueden colocar en archivos externos: Crear el js ahora!!

Archivo externo: myScript.js

```
function myFunction() {
 document.getElementById("demo").innerHTML = "Paragraph changed.";
}
```

Los scripts externos son prácticos cuando se usa el mismo código en muchas páginas web diferentes.

Los archivos JavaScript tienen la extensión de archivo **.js** .

Para usar un script externo, coloque el nombre del archivo de script en el `src`atributo (fuente) de una `<script>`etiqueta:

```
<script src="myScript.js"></script>
<! DOCTYPE html>
<html>
<body>

<h2> JavaScript externo </h2>

<p id = "demo"> Un párrafo. </p>

<button type = "button" onclick = "myFunction ()"> Pruébelo </button>

<p> (myFunction se almacena en un archivo externo llamado "myScript.js") </p>

<script src = "myScript.js"> </script>
```

```
</body>
</html>
```

Puede colocar una referencia de script externo en <head>o <body>como desee.

El script se comportará como si estuviera ubicado exactamente donde <script>se encuentra la etiqueta.

Los scripts externos no pueden contener <script>etiquetas.

Ventajas externas de JavaScript

Colocar scripts en archivos externos tiene algunas ventajas:

- Separa HTML y código
- Hace que HTML y JavaScript sean más fáciles de leer y mantener
- Los archivos JavaScript en caché pueden acelerar las cargas de página

Para agregar varios archivos de script a una página, use varias etiquetas de script:

```
<script src="myScript1.js"></script>
<script src="myScript2.js"></script>
```

Referencias externas

Se puede hacer referencia a los scripts externos con una URL completa o con una ruta relativa a la página web actual.

```
<script src="https://www.ejemplojs.com/js/myScript1.js"></script>
```

Este ejemplo utiliza un script ubicado en una carpeta especificada en el sitio web actual:

```
<! DOCTYPE html>
<html>
<body>

<h2> JavaScript externo </h2>

<p id = "demo"> Un párrafo. </p>
```

<button type = "button" onclick = "myFunction ()"> Pruébelo </button>

 (myFunction se almacena en un archivo externo llamado "myScript.js")

<script src = "/ js / myScript.js"> </script>

</body>
</html>
Salida de JavaScript

Posibilidades de visualización de JavaScript

JavaScript puede "mostrar" datos de diferentes maneras:

- Escribir en un elemento HTML, usando innerHTML.
- Escribir en la salida HTML usando document.write().
- Escribir en un cuadro de alerta, usando window.alert().
- Escribir en la consola del navegador, usando console.log().

Usando innerHTML

Para acceder a un elemento HTML, JavaScript puede usar el
document.getElementById(id)método.
El idatributo define el elemento HTML. La innerHTML propiedad define el contenido HTML:
<! DOCTYPE html>
<html>
<body>

<h2> Mi primera página web </h2>
 Mi primer párrafo.

```
<script>
document.getElementById ("demo"). innerHTML = 5 + 6;
</script>

</body>
</html>
```

Cambiar la propiedad innerHTML de un elemento HTML es una forma común de mostrar datos en HTML.

Usando document.write ()

Para fines de prueba, es conveniente usar document.write():

```
<! DOCTYPE html>
<html>
<body>

<h2> Mi primera página web </h2>
<p> Mi primer párrafo. </p>

<p> Nunca llame a document.write después de que el documento haya terminado de cargarse. Sobreescribirá todo el documento. </p>

<script>
document.write (5 + 6);
</script>

</body>
</html>
```

Si usa document.write () después de cargar un documento HTML, **eliminará todo el HTML existente** :

```
<! DOCTYPE html>
<html>
<body>

<h2> Mi primera página web </h2>
<p> Mi primer párrafo. </p>

<button type = "button" onclick = "document.write (5 + 6)"> Pruébelo </button>

</body>
</html>
```

El método document.write () solo debe usarse para pruebas.

Usando window.alert ()

Puede usar un cuadro de alerta para mostrar datos:

```
<! DOCTYPE html>
<html>
<body>

<h2> Mi primera página web </h2>
<p> Mi primer párrafo. </p>

<script>
window.alert (5 + 6);
</script>

</body>
</html>
```

Puedes saltarte la windowpalabra clave.

En JavaScript, el objeto de ventana es el objeto de alcance global, lo que significa que las variables, propiedades y métodos pertenecen por defecto al objeto de ventana. Esto también significa que especificar la windowpalabra clave es opcional:

Usando console.log ()

Para fines de depuración, puede llamar al console.log()método en el navegador para mostrar datos.

```
<! DOCTYPE html>
<html>
<body>

<h2> Activar depuración </h2>

<p> F12 en su teclado activará la depuración. </p>
<p> Luego seleccione "Consola" en el menú del depurador. </p>
<p> Luego haga clic en actualizar nuevamente. </p>

<script>
console.log (5 + 6);
</script>

</body>
</html>
```

JavaScript Imprimir

JavaScript no tiene ningún objeto de impresión o métodos de impresión.

No puede acceder a dispositivos de salida desde JavaScript.

La única excepción es que puede llamar al window.print()método en el navegador para imprimir el contenido de la ventana actual.

```
<! DOCTYPE html>
<html>
<body>

<h2> El método window.print () </h2>

<p> Haga clic en el botón para imprimir la página actual. </p>

<button onclick = "window.print ()"> Imprimir esta página </button>

</body>
</html>
```

Declaraciones JavaScript

```
<! DOCTYPE html>
<html>
<body>

<h2> Declaraciones de JavaScript </h2>

<p> Un <b> programa de JavaScript </b> es una lista de <b> declaraciones </b> que debe ejecutar una computadora. </p>

<p id = "demo"> </p>

<script>
var x, y, z; // Declaración 1
x = 5; // Declaración 2
y = 6; // Declaración 3
```

```
z = x + y; // Declaración 4

document.getElementById ("demo"). innerHTML =
"El valor de z es" + z + ".";
</script>

</body>
</html>
```

Programas JavaScript

Un programa de computadora es una lista de "instrucciones" para ser "ejecutadas" por una computadora.

En un lenguaje de programación, estas instrucciones de programación se denominan declaraciones.

Un programa de JavaScript es una lista de declaraciones de programación.

En HTML, los programas de JavaScript son ejecutados por el navegador web.

Declaraciones JavaScript

Las declaraciones de JavaScript se componen de:

Valores, operadores, expresiones, palabras clave y comentarios.

Esta declaración le dice al navegador que escriba "Hola Dolly". dentro de un elemento HTML con id = "demo":

```
<! DOCTYPE html>
<html>
<body>

<h2> Declaraciones de JavaScript </h2>

<p> En HTML, el navegador ejecuta las declaraciones de JavaScript. </p>

<p id = "demo"> </p>
```

```
<script>
document.getElementById ("demo"). innerHTML = "Hola Dolly";
</script>

</body>
</html>
```

La mayoría de los programas de JavaScript contienen muchas declaraciones de JavaScript.

Las declaraciones se ejecutan, una por una, en el mismo orden en que se escriben.

Los programas de JavaScript (y las declaraciones de JavaScript) a menudo se denominan código JavaScript.

Punto y coma;

Los puntos y coma separan las declaraciones de JavaScript.

Agregue un punto y coma al final de cada instrucción ejecutable:

```
<! DOCTYPE html>
<html>
<body>

<h2> Declaraciones de JavaScript </h2>

<p> Las declaraciones de JavaScript están separadas por punto y coma. </p>

<p id = "demo1"> </p>

<script>
var a, b, c;
a = 5;
b = 6;
c = a + b;
document.getElementById ("demo1"). innerHTML = c;
```

```
</script>

</body>
</html>
```

Cuando están separados por punto y coma, se permiten múltiples declaraciones en una línea:

```
a = 5; b = 6; c = a + b;
```

En la web, puede ver ejemplos sin punto y coma.

No se requieren declaraciones finales con punto y coma, pero es muy recomendable.

Fuentes: https://www.w3bai.com/es/js/js_intro.html

https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/JavaScript_basics

https://www.hostinger.com.ar/tutoriales/insertar-javascript-en-html/

https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript