BOM - Objetos del navegador. Navigator

- C.F.G.S. DAW
- 6 horas semanales
- Mes aprox. de impartición: Nov

Carmelo José Jaén Díaz

- iPasen - cjaedia071@g.educaand.es

Índice

OBJETIVO

- Identificar el concepto de BOM con los objetos que lo componen y saber utilizarlo en el desarrollo de aplicaciones web.

GLOSARIO

BOM (Browser Object Model). Objeto en el que se representa el navegador, no solo el documento. Mediante el BOM se puede acceder a datos como el historial de navegación, dimensiones de la ventana, etcétera.

DOM. Plataforma e interfaz de un documento que permite a los scripts y programas acceder y modificar dinámicamente su contenido, estructura y estilo.

Tag. Término en inglés que significa “etiqueta” y hace referencia a una palabra clave que sirve para describir un documento.

INTRODUCCIÓN

En esta lección vamos a ver el objeto del navegador **navigator**.

Este objeto, almacena información sobre el navegador que está ejecutando nuestra aplicación web. Esta información puede ser muy valiosa para saber ciertas cosas antes de ejecutar una u otra instrucción, como saber si están activadas las cookies, si estamos ejecutando la aplicación de manera online u offline, cuál es el idioma del navegador, etc.

No hay un estándar público pero la mayoría de los navegadores soportan este objeto.

Nota: *Debido a los constantes problemas de seguridad y actualizaciones de los navegadores, resulta de vital importancia conocer si las propiedades o métodos de este objeto se encuentran desaconsejados/obsoletos. Puedes consultar dicha información en MDN/Navigator*

INTRODUCCIÓN

El objeto **navigator** tiene una serie de propiedades, tales como:

- ***onLine***: nos indica si el navegador está actualmente online u offline.
- ***language***: muestra el idioma elegido por el usuario para la interfaz del navegador.
- ***cookieEnabled***: devuelve un booleano que indica si el navegador tiene activadas las cookies.
- ***userAgent***: devuelve la cadena de agente usuario del navegador actual.
- ***userAgentData***: devuelve un objeto **NavigatorUAData**, que da acceso a información sobre el navegador y el sistema operativo del usuario.
- ***geolocation***: devuelve un objeto **geolocation** que puede servir para conocer la ubicación del dispositivo que ejecuta la aplicación.

MÉTODOS DEL OBJETO *navigator*

A continuación, mediante un ejemplo guiado se pone en práctica los métodos citados para facilitar su comprensión.

En primer lugar, vamos a crear un documento HTML que contenga:

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

A continuación, en un fichero JS asociado a dicho documento HTML iremos modificando el valor de esa etiqueta (inicialmente vacía) mediante:

```
document.getElementById("navegador").innerHTML = texto;
```

PROPIEDADES DE *navigator*

onLine

```
//onLine: devuelve true si el navegador está online / false si offline

        texto += "<br/>OnLine: "+navigator.onLine;
```

PROPIEDADES DE *navigator*
language

```
//language: idioma del navegador

        texto += "<br/>Idioma: "+navigator.language;
```

PROPIEDADES DE *navigator*
cookieEnabled

```
//cookieEnabled: devuelve true si las cookies están activadas
//                          false en caso contrario

        texto += "<br/>Cookies: "+navigator.cookieEnabled;
```

PROPIEDADES DE *navigator*

userAgent

```
//userAgent: cabecera user-agent devuelta por el navegador al servidor

        texto += "<br/>UserAgent: "+navigator.userAgent;
```

Importante: *los usuarios pueden cambiar el valor de esta cabecera para evitar restricciones/ataques UA spoofing.*

PROPIEDADES DE *navigator*
userAgentData

```
//userAgentData: devuelve un objeto NavigatorUAData, que da acceso a información sobre el navegador y el sistema operativo del usuario.
```

El objeto **navigatorUAData** tiene una serie de propiedades, tales como:

- ***NavigatorUAData.brands***: devuelve un array con información de marcas que contiene el nombre y la versión del navegador.
- ***NavigatorUAData.mobile***: devuelve `true` si el agente de usuario se está ejecutando en un dispositivo móvil.
- ***NavigatorUAData.platform:*** devuelve la marca de la plataforma en la que se está ejecutando el agente de usuario.

PROPIEDADES DE *navigator*
userAgentData

```
//userAgentData: devuelve un objeto NavigatorUAData, que da acceso a información sobre el navegador y el sistema operativo del usuario.

//userAgentData.mobile: determinar si el navegador se ejecuta en un dispositivo móvil

texto += "<br/>Dispositivo móvil: "+navigator.userAgentData.mobile;
```

PROPIEDADES DE *navigator*
userAgentData

```
//userAgentData: devuelve un objeto NavigatorUAData, que da acceso a
información sobre el navegador y el sistema operativo del usuario.

//userAgentData.brands: determinar el navegador y version
let brands = navigator.userAgentData.brands;

if (brands && brands.length > 0) {
    texto += "<br/>Navegador: "+ brands[0].brand; // Primera marca
    texto += "<br/>Version Navegador: "+ brands[0].version; // Versión de la
primera marca
} else {
  console.log("No se encontraron marcas referentes al navegador");
}
```

PROPIEDADES DE *navigator*
getHighEntropyValues()

```
//userAgentData: obtener valores de alta entropia

   navigator.userAgentData.getHighEntropyValues([
       "architecture",
       "model",
       "platform",
       "platformVersion",
       "fullVersionList",])
     .then(highEntropy => {
// Contendrá valores de alta entropía como 'platform', 'architecture', etc
// Puedes incluir menos si lo deseas
//al devolver un objeto tendríamos que formatearlo para capturar el dato deseado
       console.log(highEntropy);
     });
```

PROPIEDADES DE *navigator*
NavigatorUAData.toJSON()

CONCEPTO DE SERIALIZADOR

En el contexto del objeto **navigatorUAData**, un serializador es un mecanismo que convierte las propiedades de baja entropía de este objeto en un formato que puede ser fácilmente almacenado o transmitido, como **JSON**.

Simplificando, el serializador toma los datos del objeto **navigatorUAData** (específicamente las propiedades de baja entropía, que son menos cambiantes y más predecibles) y los convierte a un formato estándar como **JSON**. Esto es útil cuando se necesita compartir o almacenar estos datos de forma estructurada.

PROPIEDADES DE *navigator*
NavigatorUAData.toJSON()

EJEMPLO DE FUNCIONAMIENTO

Si tienes un objeto **navigatorUAData** con propiedades como *la plataforma* o *el nombre del navegador*, el serializador convierte esas propiedades en un formato **JSON**, que se puede enviar a un servidor o almacenar en una base de datos.

CONCEPTO DE BAJA ENTROPÍA

Las propiedades de **baja entropía** son aquellas que no cambian con frecuencia y son bastante constantes, como el *nombre del navegador* o el *sistema operativo*. En contraste, las propiedades de **alta entropía** (como las relacionadas con el comportamiento del usuario) son más dinámicas y cambiantes.

PROPIEDADES DE *navigator*
NavigatorUAData.toJSON()

EJEMPLO DE USO

```
let lowEntropyData = navigator.userAgentData.toJSON();

// Esto devuelve una representación JSON de las propiedades de baja entropía
console.log(lowEntropyData);
```

PROPIEDADES DE *navigator*

geolocation

```
//geolocation: devuelve un objeto geolocation que puede servir para localizar la posición del usuario.
```

El objeto **geolocation** tiene una serie de métodos, tales como:

- ***getCurrentPosition()***: determina la ubicación actual del dispositivo y devuelve un objeto **position** con los datos.
- ***watchPosition()***: devuelve un tipo de dato `long` que representa la función de devolución de llamada de reciente creación que se invoca cada vez que cambia la ubicación del dispositivo.
- ***clearWatch()***: elimina el controlador especial instalado previamente utilizando ***watchPosition()***.

PROPIEDADES DE *navigator.geolocation*

El objeto **position** tiene una serie de propiedades, tales como:

- ***Position.coords***: devuelve un objeto **coordinates** que indica la posición actual.
    - ***Coordinates.latitude:*** devuelve un double que representa la latitud de la posición (grados).
    - ***Coordinates.longitude:*** devuelve un double que representa la longitud de la posición (grados).
    - ***Coordinates.altitude:*** devuelve un double que representa la altitud de la posición (m s. n. m.).
    - ***Coordinates.speed:*** devuelve un double que representa la velocidad del dispositivo (m/s).
    - Ver más propiedades del objeto **coordinates**

- ***Position.timestamp***: devuelve un `DOMTimeStamp` indicando el momento en que la localización ha sido recuperada.

PROPIEDADES DE *navigator*
geolocation

```
//geolocation: devuelve un objeto geolocation que puede servir para localizar la posición del usuario.

//Primero, utilizamos el método getCurrentPosition para obtener el objeto position con los datos

function obtenerUbicacion() {
    navigator.geolocation.getCurrentPosition(exito, error);
  }
```

PROPIEDADES DE *navigator*
geolocation

```
//Si el navegador tiene acceso a la ubicación (usuario da permiso, etc), formateamos el objeto position para obtener la latitud

  function exito(position) {
    let latitude = position.coords.latitude;
    texto += "<br/>Latitud: " + latitude;
    // Actualizar el contenido del elemento HTML dentro del callback ya que getCurrentPosition es asíncrona.
    document.getElementById("navegador").innerHTML = texto;
  }
```

PROPIEDADES DE *navigator*
geolocation

```
  //Si el navegador no tiene acceso a la ubicación (usuario no da permiso, etc), mostramos mensaje de error.

  function error(err) {
    texto += "<br/>No se pudo obtener la ubicación.";
  // Actualizar el contenido del elemento HTML dentro del callback de error ya que getCurrentPosition es asíncrona.
  document.getElementById("navegador").innerHTML = texto;
  }

  //Llamar a la función principal

  obtenerUbicacion();
```