BOM - Objetos del navegador. Location

- 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 **location**.

Este objeto, representa la ubicación, o **URL**, del objeto al que está vinculado. En otras palabras, básicamente **almacena la información de la URL** (o dirección de Internet) de la página que se está ejecutando en el momento.

No hay un estándar público pero la mayoría de los navegadores soportan **screen**.

CARACTERÍSTICAS

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

- ***href***: que indica cuál es el HREF (URL) de la página.
- ***hostname***: almacena el host de la página.
- ***pathname***: almacena la ruta (lo que hay después del hostname) de la página.
- ***protocol***: indica cuál es el protocolo de la página (http, https, etc.)
- ***hash***: en caso de que haya un hash o ancla de página, la muestra (Ej. #quienes-somos)
- ***host***: a diferencia de hostname, en este caso almacena el nombre del hostname y el puerto.
- ***origin***: incluye el nombre del protocolo, el hostname y el puerto.
- ***search***: almacena el quer ystring de la página (Ej. www.web.com/index.html?user=carmelo).

Ver más propiedades y métodos de **location**.

CARACTERÍSTICAS
¿Qué es el query string?

Dentro del protocolo HTTP podemos encontrar diferentes métodos (verbos) con los cuales podemos realizar diferentes tipos de peticiones. El método más utilizado sin duda es el método GET, método cual nos permite obtener un recurso por parte del servidor, ya sea una página web, un archivo txt, un gif, etc.

Siempre que nosotros ingresamos a una sitio web, por ejemplo, `www.web.com` utilizando nuestro navegador la petición al servidor se hará a través del método GET.

Algo interesante de este método es que podemos enviar información al servidor de tal forma que seamos más precisos en el recurso que deseamos obtener. La información la enviaremos a través de la URL, en una sección denominada **Query String**.

Veamos un ejemplo:

CARACTERÍSTICAS
¿Qué es el query string?

Si nosotros realizamos una petición a la ruta: `/users`

Esperamos que la respuesta (ya sea una página web, un objeto JSON o XML) tenga relación con usuarios. Si queremos (y el servidor lo permite) podemos enviar información extra: `/users?order=true`

En este caso realizamos la petición indicando que el servidor nos debe retornar los usuarios de forma ordenada. Todo lo que se encuentre después del signo de interrogación (`?`) lo conoceremos como **Query String** y posee la siguiente estructura:

```
llave (nombre del parametro) , signo igual (=) y valor
```

CARACTERÍSTICAS

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

- ***assign(<url>):*** permite asignar un nuevo documento a la página.
- ***reload():*** recarga la página (como si pulsáramos F5 o las flechas circulares).
- ***replace(<url>):*** sustituye una página por otra haciendo desaparecer su historial.

Todas las partes que forman la URL de una página pueden ser extraídas utilizando Javascript con el objeto **location**

Ver todas las partes que forman la URL

PROPIEDADES DEL OBJETO *location*

A continuación, mediante un ejemplo guiado se pone en práctica las propiedades citadas para facilitar su comprensión.

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

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

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

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

PROPIEDADES DE *location*

href

```
    //href: HREF (URL) de la página

texto += "<br/>Href: "+location.href;
```

PROPIEDADES DE *location*

hostname

```
    //hostname: nombre del host de la página

texto += "<br/>Hostname: "+location.hostname;
```

PROPIEDADES DE *location*

pathname

```
//pathname: pathname de la página

texto += "<br/>Pathname: "+location.pathname;
```

PROPIEDADES DE *location*

protocol

```
//protocol: protocolo de la página

texto += "<br/>Protocol: "+location.protocol;
```

PROPIEDADES DE *location*

hash

```
//hash: hash o ancla de la página (Ej. www.web.com/index.html#indice)
//Cuando tenemos una página con mucho texto, un ancla nos permite indicar
una posicion de esa página y desplazarnos a través de ella.
```

Ver UT2.4.13 SCROLL SNAP en CSS

```
texto += "<br/>Hash: "+location.hash;
```

PROPIEDADES DE *location*

host

Las siguientes propiedades resultan de la combinación de las anteriores.

```
//host: nombre del hostname y el puerto

texto += "<br/>Host: "+location.host;
```

PROPIEDADES DE *location*

origin

```
//origin: nombre del protocolo, hostname y el puerto

    texto += "<br/>Origin: "+location.origin;
```

PROPIEDADES DE *location*

search

```
//search: extraer la query string de la página (Ej. www.web.com/index.html?user=carmelo)

texto += "<br/>Search: "+location.search;
```

Si ejecutamos una página web en local, la mayoría de propiedades devolverán `null` debido a que no tienen un servidor donde se aloje la página, etc.

MÉTODOS DE *location*

assign(<url>)

```
//assign(<url>): asigna un nuevo documento a la página
//Al usar este método se mantiene el historial de la ventana
//El historial de la ventana es independiente del historial general del
//navegador

        function nuevoDocumento(){
            location.assign("http://www.google.com")
        }
```

MÉTODOS DE *location*

reload()

```
//reload(): recarga la página

function recarga(){
    location.reload()
}
```

MÉTODOS DE *location*

replace(<url>)

```
//replace(<url>): sustituye la página por otra. DESAPARECE SU HISTORIAL.

        function sustituye(){
            location.replace("http://www.google.com");
        }
```

MÉTODOS DE location

Por último, mediante tres botones vamos a testear el comportamiento de los métodos anteriores.

```
<button onclick="nuevoDocumento()">Carga un nuevo documento</button>
<button onclick="recarga()">Recarga la página</button>
<button onclick="sustituye()">Sustituye la página</button>
```