Utilizar Almacenamiento Local Con HTML5 En En Sitio Web

Existen ciertos nuevos elementos de HTML5, que nos permiten almacenar los datos sin la necesidad de tener una conexión. En este artículo vamos a discutir específicamente sobre **sessionStorage** y **localStorage**. El almacenamiento local, nos permite guardar los datos en el navegador del usuario y hace que nuestras aplicaciones web o juegos funcionen sin una conexión (por cierto período de tiempo).

En un ejemplo del mundo real, los desarrolladores pueden tener la ventaja del almacenamiento local, como una copia de seguridad en caso de que la conexión a Internet no está disponible. Una vez se haga conexión con el servicio de internet, se puede enviar los datos al servidor en línea.

Si deseas saber cómo utilizar esta función del navegador en su sitio web, te invitamos a seguir leyendo ;)

SessionStorage HTML5

Este elemento almacena los datos de forma temporal en el navegador. Los datos de **sessionStorage** se usan con una estructura clave y valor,ademas los datos son exclusivos de la ventana o pestaña del navegador. Siempre y cuando el navegador o la pestaña siga abierta, los datos aún estarían allí, a menos que los borremos intencionalmente o que salgamos del navegador.

Para almacenar un dato en sessionStorage, podemos usar **.setItem()**. Aquí está un pequeño ejemplo en el que almacenamos «Hello World».

[html]sessionStorage.setItem("keyExample", "Hello World");
[/html]

Por otra parte, también podemos hacer lo siguiente. Lo siguiente creará una entrada de datos con **anotherKeyName** como la clave y '**Hello Too**' como valor.

[html]sessionStorage.anotherKeyExample = "Hello Too"; [/html]

En los navegadores basados en WebKit, como Safari, Chrome y Opera, se puede ver los datos en la pestaña Recursos ó Resources. En Firefox, usted puede buscar los datos que están en la pestaña DOM de Firebug.

Vale la pena señalar que **sessionStorage** sólo puede almacenar una **cadena o texto plano**. Un entero será traducido a cadena.

Si tiene datos JSON que tendrá que darle formato a cadena mediante **JSON.stringify()** y recuperarla usando **JSON.parse()** para convertir la cadena en JSON. A continuación se presentan algunos ejemplos de código:

[js]

```
var json = JSON.stringify({[1, 2, 3]});
sessionStorage.anotherKeyExample = json;
```

[/js]

Recuperando Datos De SesssionStorage

También tenemos dos formas de recuperar los datos de **sessionStorage**. En primer lugar, podemos utilizar el **.getItem()** o apuntando directamente el nombre de clave, de la siguiente manera.

[js]

```
var a = sessionStorage.getItem("keyExample");
var b = sessionStorage.anotherKeyExample;
```

[/js]

Borrando Los Datos En sessionStorage

Como se mencionó anteriormente, se eliminarán los datos de sessionStorage cuando el usuario cierra la ventana ó pestaña del navegador. Pero también podemos borrar intencionalmente. Podemos usar el método **.removeItem()** ó eliminar la clave:

[js]

```
sessionStorage.removeItem("keyExample");
delete sessionStorage.anotherKeyExample;
```

[/js]

Almacenar Datos Localmente

También podemos almacenar datos en el navegador de una forma de local. A diferencia de **sessionStorage**, los datos almacenados localmente son persistentes; los datos permanecerán en el navegador, siempre y cuando nosotros no los eliminemos intencionalmente.

El almacenamiento de los datos localmente es tan fácil como hacerlo con **sessionStorage**. De hecho, los aspectos técnicos son iguales, excepto que ahora utilizamos un objeto **localStorage**. Podemos introducir una entrada de datos, con el método **.setItem()** o establecer directamente el nombre de la clave, así:

[js]

```
localStorage.setItem ("keyName", "Hola, Almacenamiento local");
localStorage.anotherKeyName = 1;
```

[/js]

Recuperamos los datos con el método **.getItem()**.

[js]

```
var c = localStorage.getItem ("keyName");
var d = localStorage.anotherKeyName
```

[/js]

De igual manera podemos eliminar los datos desde **localStroge** con el método **.removeItem()** y eliminando la clave.

Limite De Tamaño En El Almacenamiento Local

Tanto **sessionStorage** y **localStorage** tienen límites en cuanto a la capacidad máxima, cada navegador tiene su propio límite. Firefox, Chrome y Opera límite es de **5 MB por dominio**. **Internet Explorer** ofrece más espacio con **10 MB por dominio**. Así que asegúrese de que sus datos no podrán superar el límite. Si los datos excede el límite, es posible que desee considerar la otra alternativa, como **SQLite**.

Finalmente

El almacenamiento local es realmente una gran característica que hace posible trabajar sin conexión en las aplicaciones web y juegos. Espero que este breve artículo puede ayudarle a empezar con el almacenamiento local en su sitio o aplicación web.