



Manual Modernizr



Arquitectura Web

Profesor Antonio Cardeña

Luna Maya, Gustavo Antonio

Primavera 2021



ÍNDICE

INTRODUCCIÓN	3
Teoría	4
¿Qué es Modernizr?	4
¿Qué son los Polyfills?	4
Descarga la biblioteca	6
Implementación	8
CONCLUSIONES	11
BIBLIOGRAFÍA/REFERENCIAS	12



INTRODUCCIÓN

El objetivo de este reto y manual es mostrar la funcionalidad de la biblioteca Modernizr y que quien lea este manual sepa en qué consiste y aprenda sobre las funcionalidades básicas para su implementación en un desarrollo JavaScript. Mediante este manual de uso e implementación de Modernizr se espera que sean capaces de implementar la biblioteca con sus funcionalidades básicas.

En este manual encontrarán la información necesaria paso a paso para implementar y probar la biblioteca de Modernizr con ayuda de imágenes y ejemplos para su mejor entendimiento y se verá porque la importancia sobre esta biblioteca.



DESARROLLO

Teoría

¿Qué es Modernizr?

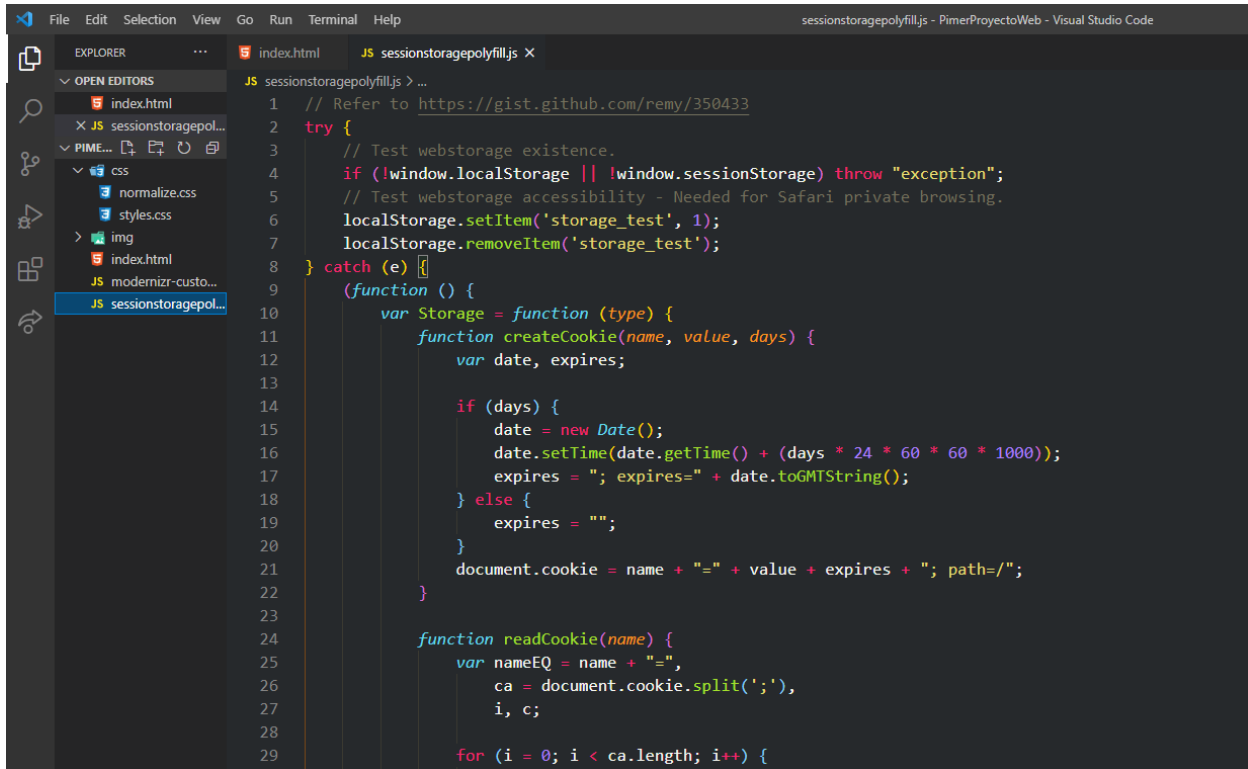
Modernizr es una biblioteca o un fragmento de código JavaScript el cual detecta de forma automática las tecnologías web o funcionalidades de próxima generación ya sean utilizadas en HTML5 o CCS3 que estén disponibles en los navegadores de nuestros usuarios. Ahora ya no es necesario desarrollar código para hacer listas de las tecnologías y funcionalidades que no estén disponibles en nuestros navegadores ya que Modernizr utiliza la detección de funciones para adaptar la experiencia de nuestros usuarios dependiendo de las capacidades de sus respectivos navegadores.

Esta herramienta es sumamente útil hoy en día ya que en los últimos años la tecnología y desarrollo web ha ido creciendo y se han desarrollado nuevas funcionalidades, al igual que cada vez más navegadores se han desarrollado, aunque esto no deja por detrás a los navegadores que ya conocemos como Chrome, Safari, Opera, Edge, etc... , por lo tanto esta herramienta nos permite saber qué tecnologías podemos o no implementar en nuestros proyectos dependiendo el navegador usado por los usuarios, esto nos permite ahorrar varias líneas de código.

¿Qué son los Polyfills?

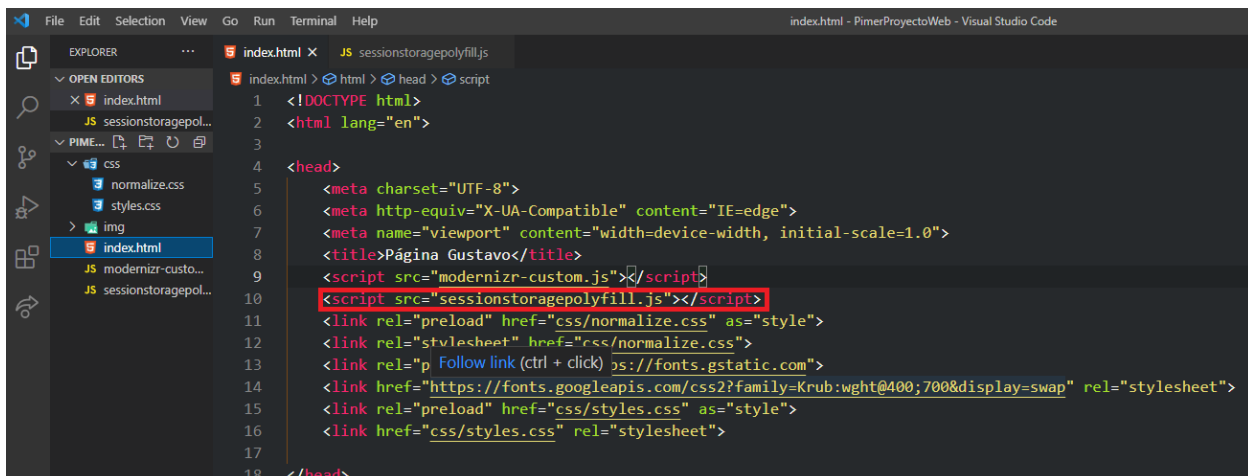
Un Polyfill es un fragmento de código JavaScript que principalmente se usa para dar una funcionalidad o proporcionar alguna tecnología moderna a un navegador antiguo o simplemente añadir X funcionalidad a un navegador que no la soporte de forma nativa, por ejemplo, si un navegador no soporta la herramienta de Session Storage por ejemplo podemos agregarla con ayuda de los Polyfills. Hoy ya podemos encontrar el código para implementar este Polyfill el cual se encuentra en el siguiente link: <https://gist.github.com/jarrodirwin/0ce4c0888336b533b2c4>

Para hacer uso de este como ejemplo tenemos que agregar el código en un nuevo archivo de tipo JavaScript en la raíz de nuestro proyecto de la siguiente forma.



```
1 // Refer to https://gist.github.com/remy/350433
2 try {
3   // Test webstorage existence.
4   if (!window.localStorage || !window.sessionStorage) throw "exception";
5   // Test webstorage accessibility - Needed for Safari private browsing.
6   localStorage.setItem('storage_test', 1);
7   localStorage.removeItem('storage_test');
8 } catch (e) {
9   (function () {
10     var Storage = function (type) {
11       function createCookie(name, value, days) {
12         var date, expires;
13
14         if (days) {
15           date = new Date();
16           date.setTime(date.getTime() + (days * 24 * 60 * 60 * 1000));
17           expires = "; expires=" + date.toGMTString();
18         } else {
19           expires = "";
20         }
21         document.cookie = name + "=" + value + expires + "; path=/";
22       }
23
24       function readCookie(name) {
25         var nameEQ = name + "=",
26             ca = document.cookie.split(';'),
27             i, c;
28
29         for (i = 0; i < ca.length; i++) {
```

Agregamos un nuevo archivo a nuestro proyecto con un nombre que nos ayude a identificar el Polyfill y copiamos el código para este caso, y por último referenciamos el archivo en nuestro index.html.



```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <title>Página Gustavo</title>
9   <script src="modernizr-custom.js"></script>
10  <script src="sessionstoragepolyfill.js"></script>
11  <link rel="preload" href="css/normalize.css" as="style">
12  <link rel="stylesheet" href="css/normalize.css">
13  <link rel="p Follow link (ctrl + click)" href="https://fonts.gstatic.com">
14  <link href="https://fonts.googleapis.com/css2?family=Krub:wght@400;700&display=swap" rel="stylesheet">
15  <link rel="preload" href="css/styles.css" as="style">
16  <link href="css/styles.css" rel="stylesheet">
17
18 </head>
```

De esta forma ya tendríamos implementado un Polyfill el cual nos permitirá contar con la herramienta de SessionStorage de ser necesaria en un navegador la cual no la soporte. Este es uno de los ejemplos de lo que podríamos hacer con Polyfills.



Pasemos a descargar e implementar la biblioteca de Modernizr.

Descarga la biblioteca

Modernizr nos ofrece una lista extremadamente amplia de funciones o tecnologías las cuales podemos seleccionar dependiendo de nuestra página web o proyecto, lo ideal en un ambiente de producción es descargar las herramientas específicas que deseemos comprobar en nuestro navegador. Para descargar la herramienta nos dirigimos a la siguiente página: <https://modernizr.com/download?setclasses>

1. El primer paso después de dirigirse a la página es seleccionar las tecnologías que deseamos detectar en nuestros navegadores. En este caso como ejemplo utilizaremos las siguientes.

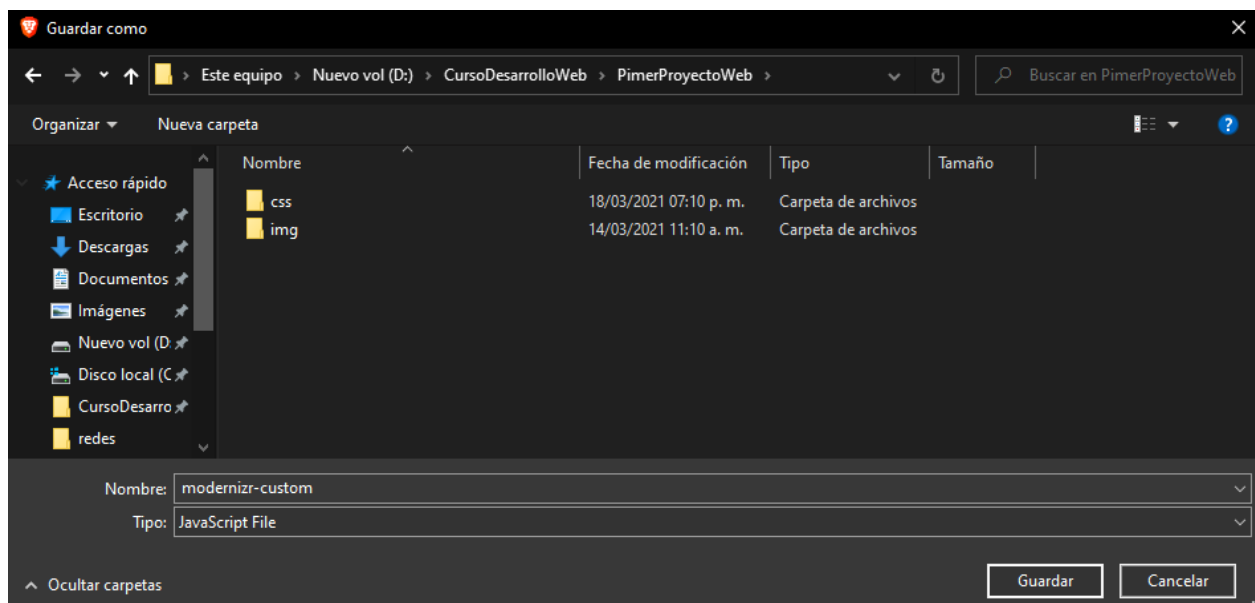
<input type="checkbox"/> Ambient Light Events	<input type="checkbox"/> Application Cache	<input type="checkbox"/> HTML5 Audio Element
<input type="checkbox"/> Battery API	<input type="checkbox"/> Blob constructor	<input type="checkbox"/> Canvas
<input type="checkbox"/> Canvas text	<input type="checkbox"/> Content Editable	<input type="checkbox"/> Context menus
<input checked="" type="checkbox"/> Cookies	<input type="checkbox"/> Cross-Origin Resource Sharing	<input type="checkbox"/> Web Cryptography
<input type="checkbox"/> Custom Elements API	<input type="checkbox"/> Custom protocol handler	<input type="checkbox"/> CustomEvent
<input type="checkbox"/> Dart	<input type="checkbox"/> DataView	<input type="checkbox"/> Emoji
<input type="checkbox"/> Event Listener	<input type="checkbox"/> EXIF Orientation	<input checked="" type="checkbox"/> Flash
<input type="checkbox"/> Force Touch Events	<input type="checkbox"/> Fullscreen API	<input type="checkbox"/> GamePad API
<input type="checkbox"/> Geolocation API	<input type="checkbox"/> Hashchange event	<input type="checkbox"/> Hidden Scrollbar

2. Construimos nuestro archivo o biblioteca personalizada ya que para un proyecto ya en producción se debe usar el archivo más ligero posible, esto para que nuestra página esté lo más optimizada posible.



3. Descargamos el archivo con la opción build y lo instalamos o guardamos en la carpeta raíz de nuestro proyecto.

Build	Copy to Clipboard	Download
Command Line Config	Copy to Clipboard	Download
Grunt Config	Copy to Clipboard	Download
Open build on codepen.io		





Implementación

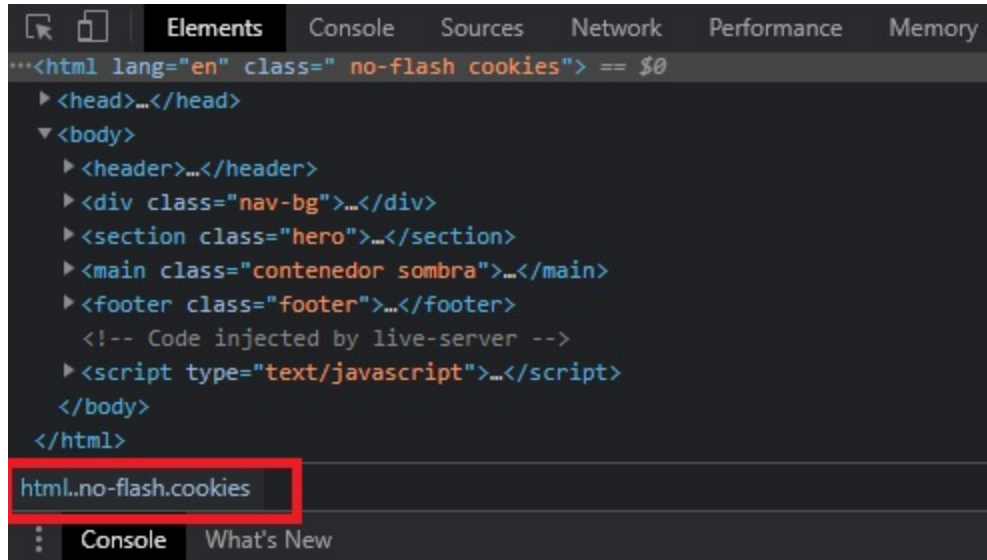
- Después de guardar el archivo en la carpeta raíz donde se encuentra nuestro proyecto, incluimos la librería en nuestro archivo index.html como se muestra en la imagen. En este caso se utilizó un pequeño proyecto como ejemplo, pero la implementación es la misma en cualquier archivo index.html.

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Página Gustavo</title>
  <script src="modernizr-custom.js"></script>
  <link rel="preload" href="css/normalize.css" as="style">
  <link rel="stylesheet" href="css/normalize.css">
  <link rel="preconnect" href="https://fonts.gstatic.com">
  <link href="https://fonts.googleapis.com/css2?family=Krub:wght@400;700&display=swap" rel="stylesheet">
  <link rel="preload" href="css/styles.css" as="style">
  <link href="css/styles.css" rel="stylesheet">
</head>

<body>
  <header>
    <h1>Gustavo Luna Maya <span>Freelancer</span></h1>
  </header>
  <div class="nav-bg">
    <nav class="navegacion-principal contenedor">
      <a href="#">Inicio</a>
      <a href="#">Sobre Mi</a>
      <a href="#">Clientes</a>
      <a href="#">Contacto</a>
    </nav>
  </div>
```

- Al abrir nuestra página en el navegador deseado accedemos a las herramientas de desarrollador y en la pestaña de elementos nos podemos dar cuenta de la disponibilidad de las tecnologías que previamente seleccionamos en la página de Modernizr.



Podemos observar que en este caso al utilizar el navegador Brave la herramienta o tecnología flash no está disponible, esto lo sabemos ya que el prefijo no- antes de la herramienta indica que no está disponible, en caso contrario las cookies si están disponibles ya que no presenta un prefijo no-.

6. Como lo que queremos es que nos indique esto mediante código JavaScript implementaremos un pequeño fragmento de código el cual nos indicará su disponibilidad.

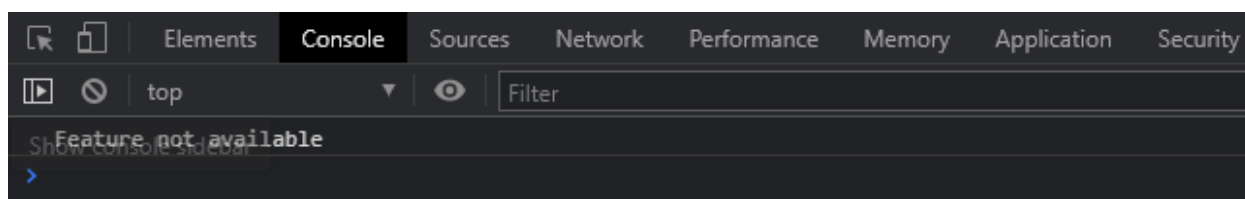
```
<body>
  <script>
    if (Modernizr.flash) {
      console.log("Feature available");
    }

    else {
      console.log("Feature not available");
    }
  </script>
```



Este script puede estar en el archivo index.html o en un archivo diferente debidamente enlazado con el index.html. En este código probamos la disponibilidad de la herramienta flash, si está disponible la tecnología nos arroja como respuesta que la herramienta está disponible en el navegador y en el caso contrario nos dirá que no lo está.

7. Para visualizar esto en nuestro navegador nos dirigimos a nuestra página web y accedemos a las herramientas de desarrollador nuevamente y no ubicamos en la pestaña de Console o Consola.



Como podemos observar la tecnología de flash no está disponible en el navegador que se utilizó en este caso Brave que está basado en Google Chrome. De esta forma podemos visualizar la disponibilidad de las distintas herramientas que nos ofrece la biblioteca Modernizr para saber la disponibilidad de estas en los distintos navegadores que hoy existen.



CONCLUSIONES

La importancia de esta herramienta junto con sus grandes ventajas la hacen una librería que todo proyecto de desarrollo web debería implementar, sabemos que en los últimos años el número de navegadores ha aumentado al igual que la diversidad de dispositivos que se conectan a la red por ello la importancia de contar con una herramienta que nos permita saber la disponibilidad de las tecnologías de desarrollo que están disponibles en los distintos navegadores y sobre todo las distintas características de los dispositivos en su mayoría hoy en día móviles que acceden a la red al igual que sus limitaciones.



BIBLIOGRAFÍA/REFERENCIAS

Modernizr. What is Modernizr. Recuperado de:

<https://modernizr.com/docs>

Modernizr.js. Detección de características con JavaScript. Recuperado de:

<https://www.youtube.com/watch?v=zgb74ii9t1w>

Archivo. sessionStorage Polyfill. Recuperado de:

<https://gist.github.com/jarrodirwin/0ce4c0888336b533b2c4>

Polyfill. ¿Qué es un Polyfill?. Recuperado de:

https://developer.mozilla.org/es/docs/Glossary/Polyfill#conocimientos_generales