

Arquitectura y Diseño de Sistemas WEB y C/S

Tema: Tecnologías Clientes



Grupo 6

Integrantes

Daniel Ferreiro Rodríguez

Bianca Marinela Lupu

Carlos Javier Hellín Asensio

Francisco Calles Esteban

Darius Dumitras Tamas

Grado de Ingeniería Informática

Curso: 2021-2022



Contenido

Introducción.....	3
Ejercicio JavaScript.....	3
HTML.....	3
JavaScript	3
Ejercicios con el Lenguaje HTML 5.....	5
Comprobación de funcionalidades en navegadores.....	5
Creación de página con elementos multimedia	6
Creación de página con formulario para su auto-validación	9
Ejercicio práctico de creación de páginas con otras funcionalidades.....	9
Drag & Drop	9
Geoposicionamiento.....	12
Canvas	15
LocalStorage.....	18
Extensión del Ejercicio JavaScript	19
Conclusiones	20



Introducción

La separación de la funcionalidad de JavaScript, el estilo y el HTML permite crear páginas web mucho más organizadas, potentes y sobre todo extensibles, poder modificar tan solo la funcionalidad, cambiar completamente el estilo de la página con la modificación de un parámetro y poder extender la estructura de la página web sin tener el JavaScript y el estilo de por medio son tan sólo unas pocas de las funcionalidades básicas que se pretenden conseguir con esta práctica.

Ejercicio JavaScript

HTML

Con HTML5 se ha creado la base, pudiendo implementar el formulario con sus botones, checkboxes, inputs...

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Restaurante v1</title>
</head>
<body>
  <form>
    <div>
      <input type="radio" id="primeros" name="radios">
      <label for="primeros">Primeros platos</label><br>
      <input type="radio" id="segundos" name="radios">
      <label for="segundos">Segundos platos</label><br>
      <input type="radio" id="postres" name="radios">
      <label for="postres">Postres</label><br>
    </div>
    <div>
      <select name="mesas" id="mesas">
        <option value="mesa1">Mesa1</option>
        <option value="mesa2">Mesa2</option>
        <option value="mesa3">Mesa3</option>
        <option value="mesa4">Mesa4</option>
        <option value="mesa5">Mesa5</option>
      </select>
    </div>
    <div>
      <select name="listaplatos" id="listaplatos" multiple></select>
      <select name="platosseleccion" id="platosseleccion" multiple></select>
    </div>
    <div>
      <p><input type="checkbox" id="cafeccheck" name="cafeccheck">
      <label for="cafeccheck">con café</label></p>
      <p><input type="checkbox" id="copaccheck" name="copaccheck">
      <label for="copaccheck">con copa</label></p>
    </div>
    <div>
      <button type="button" id="botonpagar">Pagar</button>
    </div>
    <div>
      <p>Total<input type="text" id="preciototal" value=0 readonly>€</p>
    </div>
  </form>
  <script src="libs/jquery-3.6.0.min.js"></script>
  <script src="js/restaurante.js"></script>
</body>
</html>
```

Utilizando javascript y css se ha podido crear el estilo y la funcionalidad de la página web.

JavaScript

La funcionalidad de la página web viene determinada con javascript, con funcionalidad se entienden: las acciones que deberían ocurrir al pulsar un botón, seleccionar un check, actualizar cajas de texto, inputs...

Utilizando jquery se ha accedido al DOM, como almacén de datos temporal se han utilizado arrays y hashmaps, la finalidad del almacén temporal de datos es el almacenaje de los estados de las mesas, checkboxes y que se mantengan los datos para poder recalcular los precios, la mantención del estado



de las mesas es para que se puedan mover de unas mesas y otras y que los pedidos se mantengan, con sus precios respectivos.

Ej: mesa 1 contiene puré, macarrones y trufas, con café de bebida.

Mesa1 ▾

Trufas de chocolate
Natillas
Helado de sabores
Sandía

Puré de patata
Macarrones con queso
Trufas de chocolate

☒ con café

☐ con copa

Cuando se crea una Mesa2:

Mesa2 ▾

Trufas de chocolate
Natillas
Helado de sabores
Sandía

Sandía

☐ con café

☒ con copa

Si se vuelve a seleccionar Mesa1:

Mesa1 ▾

Trufas de chocolate
Natillas
Helado de sabores
Sandía

Puré de patata
Macarrones con queso
Trufas de chocolate

☒ con café

☐ con copa

Sigue el estado almacenado.

Existen hashmaps que asocian los platos seleccionados con los precios que tienen cada uno y mediante un doble clic en la lista se añadirán los platos a la lista de la derecha, dándole doble clic a la lista de la derecha se eliminará un plato solicitado, actualizándose el precio en tiempo real, también, dependiendo del check seleccionado se pondrá su precio respectivo.

Al seleccionar del principio los primeros platos, segundos o postres, la lista de la izquierda cambiará, mostrando otros platos.



Costillas de cerdo
Merluza a la plancha
Escalope de ternera

Puré de patata
Macarrones con queso
Trufas de chocolate

Trufas de chocolate
Natillas
Helado de sabores
Sandía

Puré de patata
Macarrones con queso
Trufas de chocolate



Ejercicios con el Lenguaje HTML 5

Comprobación de funcionalidades en navegadores

En la siguiente tabla se realiza una comparativa en las principales características HTML5 entre algunos de los navegadores más populares en una versión en concreto. Los navegadores son:

- Internet Explorer 11
- Firefox 59
- Chrome 67
- Edge 18
- Samsung Internet 4.0

Se observa que Microsoft ha hecho una apuesta firme en el desarrollo del navegador Edge y ha dejado desactualizado el Internet Explorer en muchos aspectos.

El navegador Google Chrome, en la inmensa mayoría de los aspectos evaluados, ha tenido una puntuación máxima y ha ganado con cierta holgura a sus competidores. En la actualidad es el navegador más instalado con mucha diferencia, no se aprecia esa desigualdad si tenemos en cuenta las puntuaciones de esta tabla.

¿Cuál navegador utilizar diariamente? Todo dependerá de qué tarea se quiera hacer y de la experiencia de uso que se tenga.

<i>Navegadores</i>	Internet Explorer 11	Firefox 59	Chrome 67	Edge 18	Samsung Internet 4.0
<i>Puntuación / 555</i>	312	491	528	496	469
Analizando reglas	5/5	5/5	5/5	5/5	5/5
Elementos	15/30	26/30	27/30	23/30	25/30
Formularios	34/65	52/65	64/65	64/65	64/65
Componentes web	0/10	2/10	10/10	2/10	8/10
Localización y orientación	20/20	20/20	20/20	20/20	20/20
Salida	3/8	8/8	8/8	8/8	8/8
Entrada	5/10	10/10	10/10	10/10	5/10
Vídeo	31/33	29/33	29/33	33/33	29/33
Audio	20/30	27/30	29/30	27/30	27/30
Streaming	5/6	5/6	5/6	5/6	0/6
Imágenes receptivas	0/15	15/15	15/15	15/15	15/15
Gráficos 2D	14/25	24/25	24/25	24/25	24/25
3D y Realidad Virtual	15/23	18/23	20/23	20/23	20/23
Animación	5/8	8/8	8/8	5/8	8/8
Comunicación	27/40	40/40	40/40	35/40	40/40
Streams	0/6	0/6	4/6	4/6	4/6
Peer to Peer	0/45	40/45	40/45	33/45	23/45
Interacción del usuario	12/20	18/20	19/20	19/20	18/20
Rendimiento	10/12	12/12	12/12	10/12	11/12
Seguridad	13/32	24/32	29/32	29/32	24/32
Pagos	0/5	0/5	5/5	5/5	0/5
Aplicaciones web	3/17	17/17	16/17	15/17	17/17



Almacenamiento	35/40	35/40	35/40	35/40	35/40
Ficheros	15/15	15/15	15/15	15/15	15/15
Scripting	16/30	27/30	30/30	28/30	20/30
Otro	9/9	9/9	9/9	9/9	9/9

Fuente: <https://html5test.com/compare/browser/ie-11/firefox-59/chrome-67/edge-18/android.samsung-4.0.html>

Creación de página con elementos multimedia

En este apartado se ha creado una página web formada por un solo documento HTML5 en el que se incorporan elementos multimedia como son una imagen, un video y un audio.

Para poder incorporar los elementos multimedia se han utilizado diferentes etiquetas soportadas por HTML5.

Etiquetas para la imagen:

```

```

Etiquetas para el audio:

```
<figure class="audio">
  <figcaption>Banda sonora de la película</figcaption>
  <audio controls>
    <source src="media/sharknado_audio.mp3" type="audio/mpeg">
    <p>Your browser does not support the
      <code>audio</code> element.</p>
  </audio>
</figure>
```

Etiquetas para el video:

[OBJ]

Documento HTML completo:

```
<!doctype html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>Elementos Multimedia</title>
</head>

<body>

  <h1>Mi película favorita</h1>
  <h2>Sharknado</h2>
  <p class="argumento">Un salvaje huracán ha succionado gran parte del agua del océano
  y, con ella, a cientos de hambrientos tiburones que devoran todo
  a su paso por la aterrada ciudad de Los Ángeles. Los mortíferos
  escualos siembran el caos y el terror.</p>

  <div class="agrupar">
    
    <figure class = "video">
      <figcaption>Trailer de la película</figcaption>
      <video controls width="700">
        <source src="media/sharknado_video.mp4"
          type="video/mp4">
        <p>Sorry, your browser doesn't support embedded videos.</p>
      </video>
    </figure>
    <figure class="audio">
      <figcaption>Banda sonora de la película</figcaption>
      <audio controls>
        <source src="media/sharknado_audio.mp3" type="audio/mpeg">
        <p>Your browser does not support the
        <code>audio</code> element.</p>
      </audio>
    </figure>
  </div>

</body>

</html>
```

Si se carga el documento en un navegador web se visualizará lo siguiente:

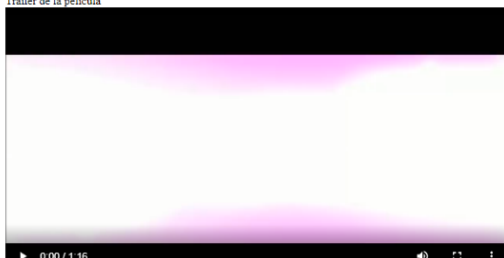
Mi película favorita

Sharknado

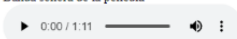
Un salvaje huracán ha succionado gran parte del agua del océano y, con ella, a cientos de hambrientos tiburones que devoran todo a su paso por la aterrada ciudad de Los Ángeles. Los mortíferos escualos siembran el caos y el terror.



Trailer de la película



Banda sonora de la película





También se ha realizado una segunda versión incorporando CSS para darle un diseño o estilo a la página.

Documento CSS:

```
h1 {
  font-size: 60px;
  font-family: sans-serif;
  font-weight: bold;
  text-align: center;
}

h2 {
  font-size: 40px;
  font-family: sans-serif;
  font-weight: bold;
  text-align: center;
}

h1, h2 {
  background: #90EE90;
}

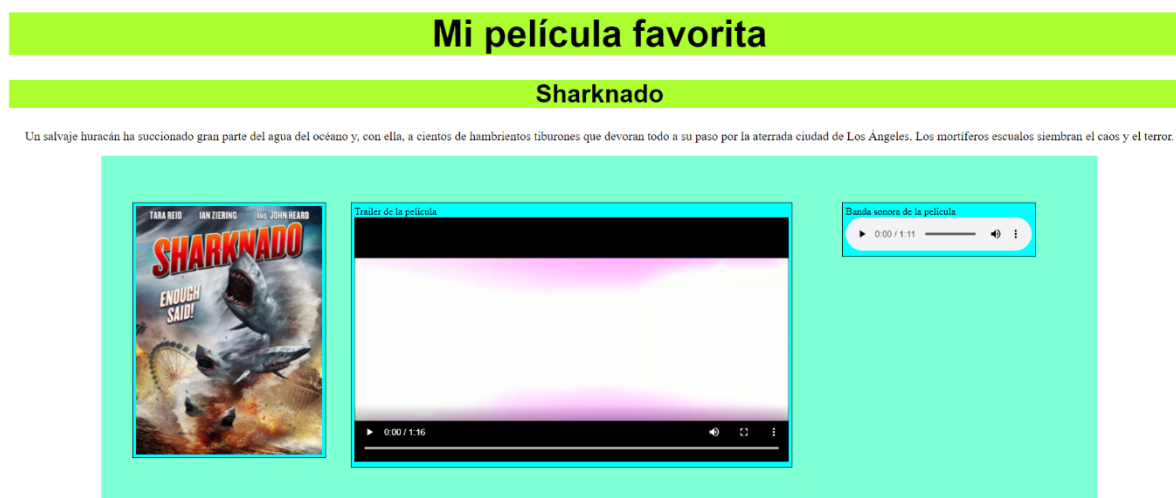
.agrupar {
  background: #4682B4;
  clear: both;
  display: table;
  margin-left: auto;
  margin-right: auto;
  padding: 50px;
}

.argumento {
  text-align: center;
  font-size: 20px;
}

.imagen, .audio, .video {
  background-color: #00FFFF;
  border: 1px solid black;
  float: left;
  padding: 5px;
  margin-top: 25px;
}

.audio {
  margin-right: 50px;
}
```

Página con CSS en el navegador web:





Creación de página con formulario para su auto-validación

En este apartado se ha hecho un formulario html5 con el uso de los nuevos **types** de la etiqueta **input**, permitiendo hacer el primer filtro de validación en el cliente, sin necesidad de interactuar con el servidor.

En este formulario, no solo se ha limitado el tamaño mínimo y máximo de los campos sino que también se ha forzado a que se rellenen con el atributo **required**. Con ayuda de la opción **invalid** de los inputs se ha cambiado el fondo de los campos a rojo (fallo) y a verde (éxito) si es **valid**. Cuando el usuario tenga todos los campos en verde, y en solo este caso, podría clicar en el botón ENVIAR y se enviaría al servidor los datos para su procesamiento.

Formulario HTML5

Nombre:

Apellidos:

Dirección:

Fecha de nacimiento:

Correo:

Elige un cine:

Número de Tarjeta:

Número de entradas:

Ejercicio práctico de creación de páginas con otras funcionalidades

Drag & Drop

Se trata de una API de HTML5 que implementa una característica común que permite al usuario agarrar un objeto de la página y modificar su ubicación con el simple hecho de arrastrarlo. Esto se realiza mediante la creación de una serie de funciones que permitan agarrar y soltar los objetos.

La función **drag(ev)** permite mediante el método **dataTransfer.setData()** establecer el tipo de dato y valor del elemento capturado, en este caso el valor del atributo id. Para permitir que los elementos sean soltados en otros se crea la función **allowDrop(ev)** que utiliza el método **event.preventDefault()** para poder permitir esas acciones. Y, por último, **drop(evento)** que se encarga de la funcionalidad de soltar un objeto ya arrastrado anteriormente, este hace uso de los métodos: **preventDefault()** para evitar la manipulación del navegador de los datos, **dataTransfer.getData()** para obtener los datos modificados por **.setData()** del objeto arrastrado, y un **appendChild** para agregar dicho objeto arrastrado al elemento en el que se produjo el evento **ondrop**.

Código del archivo **funcionalidades.js** con las funciones necesarias para la ejecución del script necesario de la página html.

```
//Para funcionalidad Drag & Drop
function allowDrop(ev) {
  ev.preventDefault();
}

function drag(ev) {
  ev.dataTransfer.setData("text", ev.target.id);
}

function drop(ev) {
  ev.preventDefault();
  var data = ev.dataTransfer.getData("text");
  ev.target.appendChild(document.getElementById(data));
}
```

Para la implementación de esta API se ha cogido como base el archivo html **comentarios.html** realizado en la práctica anterior, a lo que se ha modificado para que las imágenes con estrellas que permitían valorar, esta vez colocadas en la parte superior de la página, se pudieran arrastrar al cuadrado respectivo del usuario y así crear dicha relación entre el comentario y la valoración.

Esto se ha realizado empleando un **div** que contuviera todas las imágenes con posibilidad de ser arrastradas, y después cada usuario con sus comentarios y un cuadrado que permitiera capturar dicho elemento arrastrado, en otro **div**. El primer conjunto, aquel que contiene las imágenes, tiene la opción **draggable=true** para poder activar la acción de arrastrar, y mediante **ondragstart** puede llamar al método **drag(event)**. Por otro lado, cada **div** en cuestión relacionado con cada reseña de usuario utiliza **ondrop** para **drop(event)**, y **ondragover** para llamar a **allowdrop(event)**.

El código html implementado es el siguiente, donde se utiliza la etiqueta **<style>** para definir visualmente cada **div**.

```
<script src="js/funcionalidades.js"></script>
<title>Funcionalidad 1 - Drag & Drop</title>
<style>
  #div1, #div2, #div3, #div4, #div5, #div6 {
    float: left;
    width: 150px;
    height: 60px;
    margin: 10px;
    padding: 10px;
    border: 1px solid black;
  }
</style>
```

El **div** que contiene las imágenes sería de la siguiente manera:

```
<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)">
  
  
  
  
  
</div> </tr>
```

Mientras que el **div** con sus opciones para cada reseña, resultaría de la siguiente manera, repitiéndose para cada una de ellas.

```
<tr><td> You, 5 hours ago via PR #22 • Funcionalidad-1 Drag & Drop terminada
<div id="div2" ondrop="drop(event)" ondragover="allowDrop(event)"></div>
<p><label>Rubén Velasco: </label></p>
<p><label> Buen servicio, todo correcto y en orden. Sin duda lo recomiendo, ha sido una experiencia brutal. </label></p>
</td> </tr>
```

La página html inicialmente tendría el siguiente aspecto:

Comentarios



☐ Rubén Velasco:

Buen servicio, todo correcto y en orden. Sin duda lo recomiendo, ha sido una experiencia brutal.

☐ Darío Sánchez:

Todo por unos pocos peniques.

Buena conexión a wifi tanto en el avión como en el alojamiento, perfecto para pasar memes y animar el ambiente.

☐ Dani Ferreiro:

Decidí prepararle una sorpresa a mi jefa y creo que acerté.

Me habéis salvado la vida con vuestros paquetes y ofertas, después de las vacaciones he podido volver a dormir en la misma cama que ella.

☐ Carlos Hellín:

El viaje en avión no me ha gustado, todo muy alocado, nada mejor que un viaje en moto para disfrutar del entorno y las vistas.

En cambio muy bonito el destino, con lugares y gente muy agradable.

☐ Fran Esteban:

Ni tan mal. Está bien si quieres salir a tomar algo con los amigos, tienen unas terrazas de puta madre.

Y tienen zonas con buena conexión a internet donde te puedes echar unas buenas partidas al Lol.

Y tras arrastrar y soltar imágenes de la parte superior, quedaría:

Comentarios



Rubén Velasco:

Buen servicio, todo correcto y en orden. Sin duda lo recomiendo, ha sido una experiencia brutal.

☐ Darío Sánchez:

Todo por unos pocos peniques.

Buena conexión a wifi tanto en el avión como en el alojamiento, perfecto para pasar memes y animar el ambiente.



Dani Ferreiro:

Decidí prepararle una sorpresa a mi jefa y creo que acerté.

Me habéis salvado la vida con vuestros paquetes y ofertas, después de las vacaciones he podido volver a dormir en la misma cama que ella.



Carlos Hellín:

El viaje en avión no me ha gustado, todo muy alocado, nada mejor que un viaje en moto para disfrutar del entorno y las vistas.

En cambio muy bonito el destino, con lugares y gente muy agradable.

☐ Fran Esteban:

Ni tan mal. Está bien si quieres salir a tomar algo con los amigos, tienen unas terrazas de puta madre.

Y tienen zonas con buena conexión a internet donde te puedes echar unas buenas partidas al Lol.



Geoposicionamiento

HTML5 permite utilizar una API de geolocalización que ofrece funcionalidades relacionadas con la localización, uso de coordenadas y ubicación de los dispositivos. Esta API se puede utilizar en JavaScript mediante el objeto 'navigator.geolocation'; dicho objeto utiliza distintos métodos para obtener información de la localización del usuario o de otros sitios.

Para obtener la localización actual del dispositivo del usuario se utiliza el método 'getCurrentPosition()'.

La página web que se realiza tendrá la siguiente estructura y contenido:

```
1 <!doctype html>
2 <html lang="es">
3   <head>
4     <meta charset="utf-8">
5     <title>Geoposicionamiento</title>
6   </head>
7
8   <body>
9     <h1>Mi localización</h1>
10    <div>
11      <button id = "miLoc">Muestra mi localización</button><br/>
12      <p id = "estado"></p>
13      <a id = "enlaceMap" target="_blank"></a>
14    </div>
15
16    <script src="js/geoPos.js"></script>
17  </body>
18
19 </html>
```

Tras cargarla en un navegador web:

Mi localización

Muestra mi localización

En el HTML se crea un botón con id = "miLoc", una línea en la que se guardará el estado de conexión con los servicios de geoposicionamiento (mensaje de error, mensaje de espera de localización), y la línea en la que se mostrará el enlace con los datos de ubicación del usuario.

Para el correcto funcionamiento de la página se crea el archivo JavaScript 'geoPos.js' como se muestra a continuación:

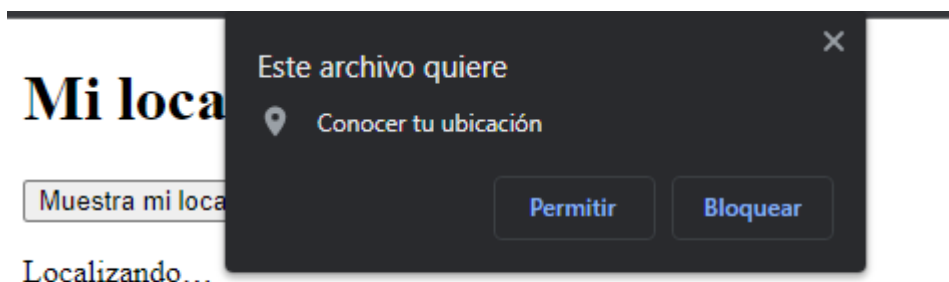
```

1 function geoFindMe() {
2
3     const estado = document.querySelector('#estado');
4     const enlaceMap = document.querySelector('#enlaceMap');
5
6     enlaceMap.href = '';
7     enlaceMap.textContent = '';
8
9     function success(position) {
10         const latitude = position.coords.latitude;
11         const longitude = position.coords.longitude;
12
13         estado.textContent = '';
14         enlaceMap.href = `https://www.google.com/maps/@${latitude},${longitude},17z?hl=es-ES`;
15         enlaceMap.textContent = `Latitud: ${latitude} °, Longitud: ${longitude} °`;
16     }
17
18     function error() {
19         estado.textContent = 'No se puede mostrar su ubicación';
20     }
21
22     if(!navigator.geolocation) {
23         estado.textContent = 'La geolocalización no está disponible para su navegador';
24     } else {
25         estado.textContent = 'Localizando...';
26         navigator.geolocation.getCurrentPosition(success, error);
27     }
28
29 }
30
31 document.querySelector('#miLoc').addEventListener('click', geoFindMe);

```

En éste se crea la función principal 'geoFindMe()' que se ejecutará cuando se pulse el botón "miLoc". En la función se almacena el estado de conexión y el enlace a los datos de ubicación. El estado se maneja mediante la función 'error()' y mediante la comprobación de si se puede usar el objeto 'navigator.geolocation' en el navegador. Las coordenadas de latitud y longitud se obtienen mediante el objeto 'position.coords' y estos se utilizan para ponerlos en el enlace a Google Maps que se almacena en 'enlaceMap'. Tras esto se llama al método 'getCurrentPosition()' cuyos argumentos son 'sucess' (la función donde se obtiene las coordenadas de localización) y 'error' (la función que actualizará el estado con un mensaje para cuando no se permita usar la ubicación).

Si se pulsa el botón se mostrará un aviso para permitir la ubicación mientras el mensaje de estado es 'Localizando...'.



En caso de se bloquee el estado se actualizará al siguiente:

Mi localización

Muestra mi localización

No se puede mostrar su ubicación

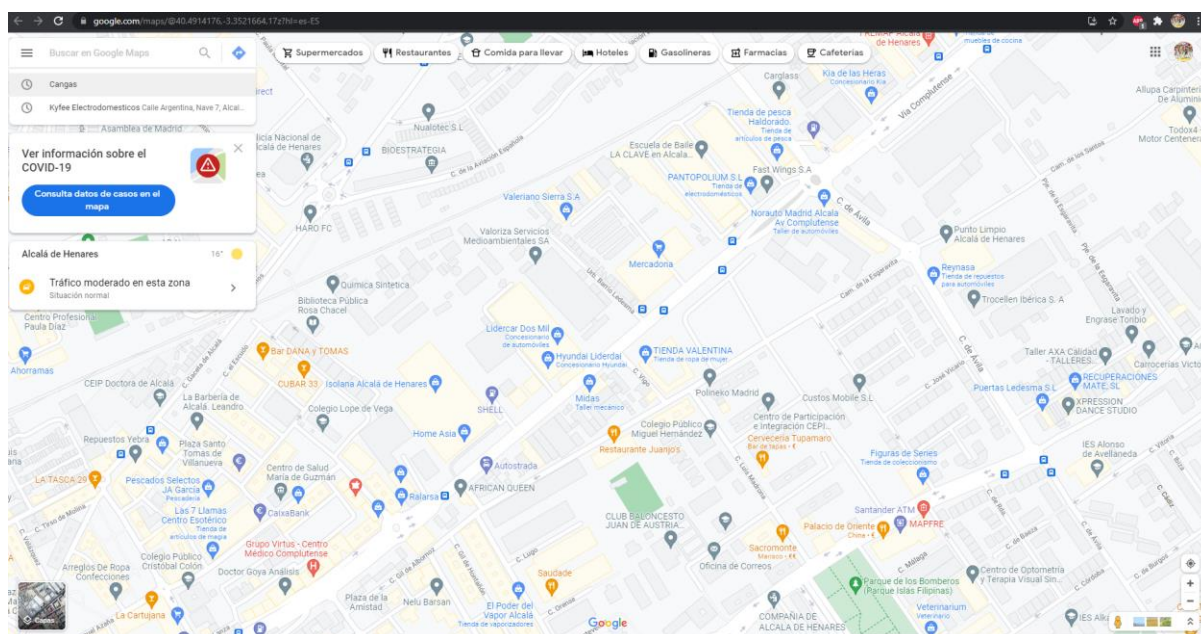
En caso de que se permita, el estado se borrará y se mostrará el enlace con las coordenadas de la ubicación del dispositivo del usuario:

Mi localización

Muestra mi localización

[Latitud: 40.4914176 °, Longitud: -3.3521664 °](#)

Si se pincha en el enlace se abrirá una nueva pestaña de Google Maps con la localización en el mapa:





Canvas

Se trata de una API de HTML5 que implementa una característica común que permite dibujar gráficas, imágenes, modificar texto, etc. En este caso se ha aplicado modificaciones al segundo título, se ha subido la imagen del mapa, y se ha aplicado un filtro a la imagen del avión.

Se utiliza **window.onload = function()** para evitar hacer las llamadas desde el código html, así al iniciar la página se aplican directamente. Dentro de esta se encuentran **iniTitle()**, que se encarga de la modificación del título **Bienvenid@! Tenemos tu vuelo ideal**, **uploadPic()** para subir la imagen correspondiente al mapa utilizando ciertas medidas aplicadas, y por último **degradePic()** que se encarga de todo el proceso de degradado de la imagen del avión, este crea una imagen de fondo a la que se le superpone la imagen original y posteriormente se degrada la parte inferior con tonos transparentes y blancos. Así, al iniciar la página se realizan todas las funciones resultando la página final.

Código del archivo **funcionalidades.js** con las funciones necesarias para la ejecución del script necesario de la página html.

```
//Para funcionalidad Canvas
window.onload = function(){
    iniTitle();
    uploadPic();
    degradePic();
}

function iniTitle(){
    var title = document.getElementById("titulo");
    var titletx = title.getContext("2d");
    titletx.font = "30px Arial";
    titletx.strokeStyle = "#FF0000";
    titletx.strokeText("Bienvenid@! Tenemos tu vuelo ideal", 20, 50);
}

function uploadPic(){
    var picMap = document.getElementById('picMap');
    var ctx = picMap.getContext('2d');
    var img = new Image();

    img.onload = function(){
        picMap.width = 500;
        picMap.height = 300;
        ctx.drawImage(img, 0, 0, img.width, img.height);
    }
    img.src = "img/mundo.gif";
}
```

```
function degradePic(){
    var picAvion = document.getElementById('picAvion');
    var ctxA = picAvion.getContext('2d');

    var lingrad = ctxA.createLinearGradient(0,0,0,picAvion.height);
    lingrad.addColorStop(0, 'orange');
    lingrad.addColorStop(.6, 'purple');
    lingrad.addColorStop(1, 'blue');

    ctxA.fillStyle = lingrad;
    ctxA.fillRect(10,10,200, 100);

    var img = new Image();
    img.onload = function(){
        var icvs = document.createElement('canvas');
        icvs.width = img.width;
        icvs.height = img.height;
        var ictx = icvs.getContext('2d');
        ictx.drawImage(img, 0, 0);

        ictx.globalCompositeOperation = 'destination-out';

        var gradient = ictx.createLinearGradient(0, 0, 0, icvs.height);
        gradient.addColorStop(.4, "transparent");
        gradient.addColorStop(1, "white");
        ictx.fillStyle = gradient;
        ictx.fillRect(0, 0, icvs.width, icvs.height);

        picAvion.width = 700;
        picAvion.height = 300;
        ctxA.drawImage(icvs, 0, 0, img.width, img.height, 0, 0, picAvion.width, picAvion.height);
    }
    img.src = "img/avion.jpg";
}
```

El código html implementado es el siguiente, donde se utiliza la etiqueta **<canvas>** tanto para el título, y las dos imágenes. No necesita de más parámetros o llamadas a funciones ya que se realiza dentro de **funcionalidades.js** toda la ejecución correspondiente.

```
<td colspan="1" rowspan="28" style="vertical-align: top;">
<canvas id="titulo" width="578" height="100"></canvas>
<canvas id="picMap"></canvas>
<br>
<p>Desde Asia hasta Rumanía, pasando por España, todos los clientes que hemos tenido, han viajado cómodos, seguros
y con las mejores condiciones que una compañía de vuelos low cost puede ofrecer</p>
<p>¿No te convence? echa un vistazo a las opiniones de nuestros clientes en el apartado Comentarios</p>
<p>Si te sientes afortunad@, mira a ver si encuentras tu viaje ideal en ofertas!!</p>
<canvas id="picAvion"></canvas>
<h2>Los mejores aviones!</h2>
<p>La última tecnología es lo más, puedes volar, y eso ya es decir mucho, Galileo soñaba con dichas hazañas,
¿no te sientes afortunado?</p>
<p>para cualquier duda, no seas tímido, pregunta, ve a contacto y envianos un mensaje... o... llámanos!</p>
</td>
```

La página html tendría el siguiente aspecto, donde se puede ver el título modificado, la imagen del mapa y por último la aplicación de degradado a la imagen del avión:



Entrada



Bienvenid@! Tenemos tu vuelo ideal

Desde Asia hasta Rumania, pasando por España, todos los clientes que hemos tenido, han viajado cómodos, seguros y con las mejores condiciones que una compañía de vuelos low cost puede ofrecer

¿No te convence? echa un vistazo a las opiniones de nuestros clientes en el apartado Comentarios

Si te sientes afortunad@, mira a ver si encuentras tu viaje ideal en ofertas!!



Los mejores aviones!

La última tecnología es lo más, puedes volar, y eso ya es decir mucho, Galileo soñaba con dichas hazañas, ¿no te sientes afortunado?

para cualquier duda, no seas tímido@, pregunta, ve a contacto y envianos un mensaje... o... llámanos!

LocalStorage


Se ha hecho una miniaplicación en la que se ha podido ver todas las opciones del LocalStorage.

Alimentos


Alimento: Cantidad: Comprado

Lista de alimentos comprados

No hay alimentos comprados



Eliminar el primer elemento de la lista

 **Borrar LocalStorage**

Funcionamiento: Primeramente, se debe especificar el nombre del alimento y la cantidad que se ha comprado para poder darle al botón **Comprado**. Esta información se guarda en el localStorage, la clave es el alimento y el valor la cantidad especificada. Al añadirse un elemento a la lista, el mensaje “No hay alimentos comprados” y la imagen desaparecen.

El botón “Eliminar el primer elemento de la lista”, visualmente hace lo que bien dice su nombre, pero también lo elimina de la estructura.

El botón “Borrar LocalStorage” elimina todo el contenido de la estructura y muestra el mensaje “No hay alimentos comprados” y la imagen de las verduras vuelve aparecer.

Alimentos

Alimento: Cantidad: Comprado

Lista de alimentos comprados

- tomate 3
- lechuga 4

Eliminar el primer elemento de la lista

 **Borrar LocalStorage**

Importante: Si se le da al botón “Comprado” y del alimento ha sido comprado anteriormente, se le sobrescribe la cantidad por el último valor especificado.



Extensión del Ejercicio JavaScript

Al documento HTML del restaurante se le ha añadido una hoja de estilos CSS llamada 'style.css' que se ha creado de la siguiente forma:

```
1 body {  
2   background-image: url("https://static.vecteezy.com/system/resources/previews/001/229/268/non_2x/white-brick-wall-for-background-free-photo.jpg");  
3 }  
4  
5  
6 .formulario {  
7  
8 }  
9  
10 .tipoPlatos {  
11   font-weight: bold;  
12   margin-right: 80px;  
13   float: left;  
14 }  
15  
16 .mesas {  
17   margin-top: 20px;  
18   font-weight: bold;  
19 }  
20  
21 .listas {  
22   clear: both;  
23   margin-top: 50px;  
24 }  
25  
26  
27 .listaPlatos {  
28   width: 200px;  
29   height: 125px;  
30 }  
31  
32 .listaSeleccion {  
33   width: 200px;  
34   height: 125px;  
35 }  
36  
37 .checkC {  
38   font-weight: bold;  
39 }  
40 .checkC, .pagar {  
41   margin-left: 200px;  
42 }  
43  
44 .pagar {  
45   background-color: blue;  
46   color: white;  
47 }  
48  
49 .precio {  
50   margin-left: 165px;  
51 }
```

La interfaz de la página web con el CSS implementado quedaría como se muestra a continuación:

☒ **Primeros platos**
☐ **Segundos platos**
☐ **Postres**

Mesa1 ▼

Puré de patata

Macarrones con queso

Garbanzos con chorizo

Judías verdes

Sopa de pescado

☐ con café
☐ con copa

Total 0 €

Pagado



Conclusiones

En esta práctica se ha logrado consolidar los conocimientos de HTML5, CSS3 y Javascript. Se ha repasado las nuevas etiquetas de HTML5. Todos los apartados se han terminado. Se ha tardado más de lo esperado en la resolución de los ejercicios teniendo en cuenta que para muchos es la primera vez que se hace programación web.