Módulo 1 - Introducción - Laboratorio

1. Instalación de Visual Studio Code

- https://code.visualstudio.com/download

2. Instalación de Node.js

- https://nodejs.org/en/

3. Instalación de lite-server

- `npm install --global lite-server` o `npm i -g lite-server`

> Desinstalar `lite-server`: `npm uninstall -g lite-server`

4. Interacción con el DOM / JavaScript

1. Crear una estructura básica de HTML

- Desde dentro de **VS Code** creamos un nuevo fichero HTML *./index.html*
- Agregamos una estructura HTML básica (recordad, escribimos **html:5** y presionamos intro).

./index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Datos de usuario</title>
  </head>

  <body></body>
</html>
```

- Agregamos los contenedores genéricos para estructurar nuestra página.

./index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Datos de usuario</title>
  </head>

  <body>
    <!-- diff -->
    <div class="container">
      <div class="img"></div>
      <div class="data"></div>
    </div>
    <!-- diff -->
  </body>
</html>
```

- Ahora vamos a crear dos cajas de texto con etiquetas para rellenarlas.

./index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Datos de usuario</title>
  </head>

  <body>
    <div class="container">
      <div class="img"></div>
      <div class="data">
        <!-- diff -->
        <label for="name"><b>Name</b></label>
        <input type="text" id="name" name="name" />

        <label for="lastName"><b>Last name</b></label>
        <input type="text" id="lastName" name="lastName" />
        <!-- diff -->
      </div>
    </div>
  </body>
</html>
```

2. Creamos los estilos de nuestro página

- Vamos a estructurar nuestros ficheros, así que primero creamos en el raíz una carpeta `css` para los estilos.
- Creamos un nuevo fichero de estilos *site.css*

./css/site.css

```
body {
  font-family: Arial, Helvetica, sans-serif;
}

input {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0 32px 0;
  display: inline-block;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

.container {
  margin: auto;
  width: 50%;
}

.data {
  padding: 16px;
}
```

- Enlazamos la hoja de estilos creada al `index.html`

./index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Datos de usuario</title>
```

```
    <!-- diff -->
    <link rel="stylesheet" href="css/site.css" />
    <!-- diff -->
  </head>

  <body>
    <div class="container">
      <div class="img"></div>
      <div class="data">
        <label for="name"><b>Name</b></label>
        <input type="text" id="name" name="name" />

        <label for="lastName"><b>Last name</b></label>
        <input type="text" id="lastName" name="lastName" />
      </div>
    </div>
  </body>
</html>
```

3. Agregando interacción al HTML con JavaScript

- Vamos a introducir nuestros datos en el formulario que hemos creado. Para ello, al ser un elemento de tipo `input`, una vez tenemos el elemento, debemos hacer uso del método `value`
- Primero, creamos una carpeta `js`
- Creamos un fichero JavaScript llamado *demo.js*

./js/demo.js

```
document.getElementById("name").value = "Mi nombre";
document.getElementById("lastName").value = "Mi apellido";
```

- Enlazamos el fichero `demo.js` previamente creado al HTML mediante una etiqueta `script`

./index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Datos de usuario</title>
    <link rel="stylesheet" href="css/site.css" />
  </head>

  <body>
    <div class="container">
      <div class="img"></div>
      <div class="data">
        <label for="name"><b>Name</b></label>
        <input type="text" id="name" name="name" />

        <label for="lastName"><b>Last name</b></label>
        <input type="text" id="lastName" name="lastName" />
      </div>
    </div>
    <!-- diff -->
    <script src="js/demo.js"></script>
    <!-- diff -->
  </body>
</html>
```

- Comprobamos que todo funciona como esperamos.

4. Insertar una imagen en nuestro formulario

- Primero, buscamos una imagen apropiada, por ejemplo en https://undraw.co/search podemos buscar una imagen de perfil mediante la palabra `avatar`

- La guardamos dentro de una carpeta para las imágenes que llamaremos `img`.
- Una vez tenemos nuestra imagen, vamos a editar el `index.html` para poder insertar una etiqueta `img`

./index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Datos de usuario</title>
    <link rel="stylesheet" href="css/site.css" />
  </head>

  <body>
    <div class="container">
      <div class="img">
        <!-- diff -->
        <img src="" id="avatar" alt="Avatar" class="avatar" />
        <!-- diff -->
      </div>
      <div class="data">
        <label for="name"><b>Name</b></label>
        <input type="text" id="name" name="name" />

        <label for="lastName"><b>Last name</b></label>
        <input type="text" id="lastName" name="lastName" />
      </div>
    </div>
    <script src="js/demo.js"></script>
  </body>
</html>
```

- Actualizamos el fichero de estilos para darle forma a la imagen.

./css/site.css

```
body {
  font-family: Arial, Helvetica, sans-serif;
}

input {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0 32px 0;
  display: inline-block;
  border: 1px solid #ccc;
  box-sizing: border-box;
}

.container {
  margin: auto;
  width: 50%;
}
/** diff */
.img {
  text-align: center;
  margin: 24px 0 12px 0;
  position: relative;
}

img.avatar {
  width: 20%;
  border-radius: 50%;
}
/** diff */
.data {
  padding: 16px;
}
```

- Ahora debemos actualizar el script para agregar el código necesario.

./js/demo.js

```
document.getElementById("name").value = "Mi nombre";
document.getElementById("lastName").value = "Mi apellido";
// diff
document.getElementById("avatar").src = "ruta_imagen.png";
```

> Recuerda que debes poner la ruta a tu imagen con la extensión y la carpeta.

- Comprobamos los cambios.

Opcional

El objetivo es mostrar por consola el contenido de uno de los input, pulsando un botón.

- Vamos a editar el *index.html* para agregar el botón.

./index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Datos de usuario</title>
    <link rel="stylesheet" href="css/site.css" />
  </head>

  <body>
    <div class="container">
      <div class="img">
        <!-- diff -->
        <img src="" id="avatar" alt="Avatar" class="avatar" />
      </div>
      <div class="data">
        <label for="name"><b>Name</b></label>
        <input type="text" id="name" name="name" />

        <label for="lastName"><b>Last name</b></label>
        <input type="text" id="lastName" name="lastName" />
      </div>

      <!-- diff -->
      <div id="actions">
        <button type="button">
          Update
        </button>
      </div>
      <!-- diff -->
    </div>
    <script src="js/demo.js"></script>
  </body>
</html>
```

- Actualizamos el estilado para tener en cuenta este nuevo elemento.

./css/site.css

```
body {
  font-family: Arial, Helvetica, sans-serif;
}

input {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0 32px 0;
```

```
  border: 1px solid #ccc;
  box-sizing: border-box;
}

.container {
  margin: auto;
  width: 50%;
}

.img {
  text-align: center;
  margin: 24px 0 12px 0;
  position: relative;
}

img.avatar {
  width: 20%;
  border-radius: 50%;
}

.data,
.actions {
  padding: 16px;
}

button {
  background-color: #c7b93e;
  border: none;
  color: white;
  padding: 10px 24px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
}
```

- Agregamos en el *index.html* el código necesario para mostrar por consola nuestro input con el id `name`.

./index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Datos de usuario</title>
    <link rel="stylesheet" href="css/site.css" />
  </head>

  <body>
    <div class="container">
      <div class="img">
        <!-- diff -->
        <img src="" id="avatar" alt="Avatar" class="avatar" />
      </div>
      <div class="data">
        <label for="name"><b>Name</b></label>
        <input type="text" id="name" name="name" />

        <label for="lastName"><b>Last name</b></label>
        <input type="text" id="lastName" name="lastName" />
      </div>

      <div id="actions">
        <!-- diff -->
        <button
          type="button"
          onclick="console.log(document.getElementById('lastName').value)"
        >
          Update
        </button>
        <!-- diff -->
```

```
      </div>
    </div>
    <script src="js/demo.js"></script>
  </body>
</html>
```

Lo que hemos hecho es obtener el `value` del input con `id` name y utilizarlo como parámetro de entrada para alimentar la función de `console.log`.

5. Depurar con VS Code y Chrome

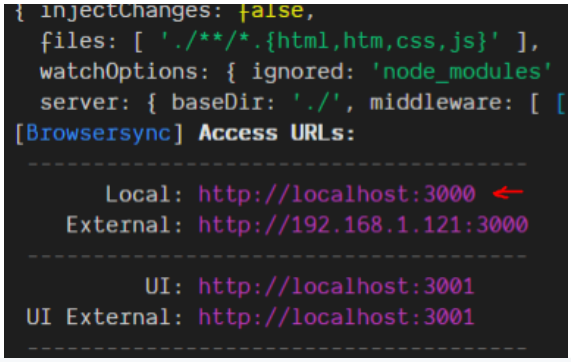
VS Code te permite instalar extensiones muy útiles para desarrollar. Una de ellas es `Debugger for Chrome`.

Esta extensión te permite depurar tu código directamente desde el editor. Vamos a ir paso a paso mostrando cómo se instala, configura y se usa con nuestro código.

- Primero, navegamos en el panel lateral izquierdo hasta la sección de extensiones.
- En el buscador, introducimos `Debugger for Chrome`. Debería aparecer la primera de la lista (identificador: msjsdiag.debugger-for-chrome).
- Para instalarla, pulsamos el botón verde de instalar y esperamos a que finalice. En algunas ocasiones nos pide reiniciar VSCode. Yo recomiendo cerrarlo y volverlo a abrir.
- Configuramos la extensión yendo a la sección de depuración que se encuentra en el lateral izquierdo (el icono del bichito).
- En el desplegable de la izquierda, seleccionamos `Agregar configuración`. También podemos seleccionarlo desde el menú de herramientas (arriba del todo), en >Depuración >Agregar configuración
- Seleccionamos como entorno `Chrome` y se nos creará un fichero con una configuración parecida a la siguiente

```
{
  // Use IntelliSense to learn about possible attributes.
  // Hover to view descriptions of existing attributes.
  // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
  "version": "0.2.0",
  "configurations": [
    {
      "type": "chrome",
      "request": "launch",
      "name": "Launch Chrome against localhost",
      "url": "http://localhost:8080",
      "webRoot": "${workspaceFolder}"
    }
  ]
}
```

- Editamos la configuración para que depure nuestro `localhost`. Debemos apuntar al **puerto** sobre el que se está ejecutando `lite-server`




```
{
  // Use IntelliSense to learn about possible attributes.
  // Hover to view descriptions of existing attributes.
  // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
  "version": "0.2.0",
  "configurations": [
    {
```

```
      "type": "chrome",
      "request": "launch",
      "name": "Launch Chrome against localhost",
-      "url": "http://localhost:8080",
+      "url": "http://localhost:3000",
      "webRoot": "${workspaceFolder}"
    }
  ]
}
```

- Levantamos `lite-server` si no lo teníamos ya y comprobamos que el puerto coincide.
- Volvemos al fichero `demo.js` y ponemos un punto de interrupción haciendo clic en el lateral derecho de la línea que queremos depurar, a la derecha de los números de línea.
- **Depuramos nuestro código pulsando F5**. También puedes lanzar la depuración desde el menú >Depuración >Comenzar la depuración. O desde el desplegable que tienes en la sección de depuración, haciendo clic en el triángulo verde de comienzo.