Aplicaciones Web - Práctica1: Introducción a HTML

Preparación del Entorno

Vamos a utilizar **Visual Studio Code (VSCode)** para crear nuestras páginas web.

¿Qué es VSCode?

- Es un **editor de código** creado por Microsoft.
- Sirve para escribir y organizar programas, páginas web y otros proyectos.
- Aunque está pensado para programar, también se puede usar para editar texto normal.
- Es gratuito y se puede descargar en Windows, Linux y macOS.

¿Qué es una extensión?

- Una extensión es como una **"aplicación extra"** que añade nuevas funciones a VSCode.
- Ejemplos:
  - Soporte para nuevos lenguajes.
  - Herramientas para depuración.
  - Servidores web locales como *Live Server*.

¿Cómo instalar una extensión en VSCode?

1. Abrir VSCode.
2. En la barra lateral izquierda, pulsar el icono de **Extensiones** (cuatro cuadraditos).
3. Buscar el nombre de la extensión (por ejemplo, *Live Server*).
4. Pulsar **Install** para añadirla.
5. Una vez instalada, aparecerán nuevas opciones o botones en VSCode.

Extensiones que usaremos

- **Live Server**
  - Monta un **servidor HTTP en local**.
  - Abre la página en tu **navegador real** (Chrome, Edge, Firefox...).
  - Dirección típica: `http://127.0.0.1:5500`.
  - Es la forma más parecida a cómo se vería tu web publicada en Internet.
- **Live Preview**
  - También monta un **servidor HTTP en local**, pero muestra la página dentro de una **pestaña integrada en VSCode**.

  - Permite ver la web sin salir del editor.

  - Es útil para trabajar de forma rápida sin cambiar de ventana.

En ambos casos:
- Un **servidor HTTP** es una aplicación que entrega páginas web al navegador cuando este las solicita.
- En Internet, los servidores HTTP están en ordenadores remotos; en nuestro caso, estas extensiones hacen ese trabajo **en tu propio ordenador**.
- Gracias a ello podemos **desplegar** (mostrar) la página mientras la editamos.
- Cada vez que guardes el archivo, la página se actualizará sola.

¿Cómo abrir la página con estas extensiones?

¿Qué es una IP?

- Una dirección **IP (Internet Protocol)** es como la "dirección" de un ordenador en la red.

- La **IP local** más común es `127.0.0.1` o `localhost`.
  - Representa **tu propio ordenador**.

  - Cuando accedes a `http://127.0.0.1` **a través del navegador**, estás haciendo una conexión HTTP hacia tu propio ordenador.

¿Qué es un puerto?

- Un **puerto** es como una "puerta de entrada" a un servicio en el ordenador.

- Sirve para identificar qué aplicación está usando la red en tu ordenador.

- Por ejemplo: `http://127.0.0.1:5500`
  - `127.0.0.1` → tu ordenador (IP local).

  - `5500` → el puerto donde está escuchando Live Server.

- Así el navegador sabe a qué servicio conectarse.

Introducción: ¿Qué es HTML?

- HTML significa **HyperText Markup Language** (lenguaje de marcado de hipertexto).
- **Hipertexto**: texto que incluye enlaces que permiten saltar a otras páginas o recursos (imágenes, videos, audios...).

- No es un lenguaje de programación, sino un **lenguaje de marcado** que organiza el contenido de una página web.

- Funciona dentro del **navegador**.

- Cada página web que visitas está hecha con HTML en su base.

Relación con HTTP

- **HTTP** es el protocolo de comunicación: es como el "vehículo" que lleva la página desde el servidor hasta tu navegador.
- El navegador recibe el HTML mediante HTTP y lo interpreta para mostrar la web.

Relación con CSS

- **CSS (Cascading Style Sheets)** se encarga de la **apariencia**: colores, tamaños, posiciones, fondos, estilos...
- HTML estructura, CSS estilo.

Relación con JavaScript

- **JavaScript** da **interactividad y lógica**: botones que reaccionan, menús que se despliegan, validaciones, juegos en el navegador, etc.
- HTTP = transporte, HTML = estructura, CSS = estilo, JS = interacción.

Estructura Básica de una Página HTML

Antes de escribir una página completa, vamos a entender cómo funciona HTML.

¿Qué es un elemento HTML?

- Un **elemento** HTML está formado por:
  - Una **etiqueta de apertura** (tag de apertura).
  - Un **contenido** (texto u otros elementos).
  - Una **etiqueta de cierre** (tag de cierre).
- Ejemplo: `<p>Hola mundo</p>`
  - `<p>` → etiqueta de apertura.
  - `Hola mundo` → contenido.
  - `</p>` → etiqueta de cierre.

¿Qué es un tag?

- Un **tag** es la parte entre `< >` que le dice al navegador cómo debe interpretar el contenido.
- Puede ser de dos tipos:
  - **Con apertura y cierre**: `<h1> ... </h1>`, `<p> ... </p>`.
  - **Tag vacío (sin cierre)**: no tiene contenido y tampoco necesita etiqueta de cierre.
    - Ejemplo: `<br>` (salto de línea).

¿Qué son los atributos?

- Los **atributos** son información extra que se puede añadir a un tag para modificar su comportamiento o dar más detalles.

- Siempre van en la **etiqueta de apertura**.

- Se escriben como `nombre="valor"`.

Ejemplo sencillo:
- El atributo **`title`** sirve para añadir una descripción o información extra sobre un elemento.
- Normalmente, el navegador muestra ese texto como un **tooltip** (cuadro emergente) al pasar el ratón por encima. - Del atributo **`align`** ya hablaremos en la siguiente práctica.

```
<h1 title="Este es un encabezado principal">Bienvenidos</h1>
<p align="center">Este texto está centrado</p>
```

Etiquetas dentro de otras

- Los elementos HTML pueden estar **anidados**, es decir, uno dentro de otro.

Estructura mínima de un documento HTML

Todos los archivos HTML empiezan con un "esqueleto" común:

1. **DOCTYPE**
   - `<!DOCTYPE html>` indica que el documento usa HTML5.
2. **Etiqueta `<html>`**
   - Es el contenedor principal. Todo el contenido de la página va dentro de `<html> ... </html>`.
3. **Etiqueta `<head>`**
   - Contiene información **sobre la página**, no visible directamente.

   - Ejemplo: el título que aparece en la pestaña del navegador (`<title>`).
4. **Etiqueta `<body>`**
   - Contiene el **contenido visible** de la página: texto, imágenes, enlaces, etc.

Etiquetas básicas en el `<body>`

- **Encabezados (`<h1>` a `<h6>`)**
  - Sirven para títulos y subtítulos.

  - `<h1>` es el más importante, `<h6>` el menos.
- **Párrafos (`<p>`)**
  - Sirven para escribir texto normal.
- **Header y Footer**
  - `<header>` → cabecera de la página o sección (normalmente el título principal o menú).

  - `<footer>` → pie de la página o sección (información al final, como copyright o contacto).

Comentarios

- Los comentarios sirven para escribir notas en el código que no se muestran en la página web.
- Se escriben entre `<!--` y `-->`.

¿Qué es `index.html`?

- Cuando creamos una página web, normalmente el archivo principal se llama **`index.html`**.

- La palabra *index* significa **índice** o **página de inicio**.

- Los servidores web, por defecto, buscan un archivo llamado `index.html` en una carpeta para mostrarlo como **punto de entrada** de la web.

Ejemplo:
- Si subes tu web a un servidor y entras en `http://midominio.com/`, el servidor abrirá automáticamente el archivo `index.html`.
- Si quieres otra página distinta, tendrás que poner su nombre en la dirección, por ejemplo:
- `http://midominio.com/contacto.html`
- `http://midominio.com/galeria.html`

Hiperenlaces

Un **hiperenlace** (enlace o *link*) es un elemento que permite **navegar de una página a otra** o abrir recursos externos.
En HTML los hiperenlaces se crean con la etiqueta `<a>` (*anchor*, que significa "ancla").

Atributos principales de `<a>`

- `href` → (*Hypertext REFerence*) indica la dirección a la que llevará el enlace.
  - Puede ser una dirección web externa (ej: `https://www.google.com`) o un archivo interno de tu proyecto (ej: `contacto.html`).

- `target` → define **dónde** se abrirá el enlace.
  - Por defecto abre en la **misma pestaña**.

  - Con `target="_blank"` se abre en **una pestaña nueva**.

```
<a href="https://www.google.com" target="_blank">Abrir Google en otra pestaña</a>
```

- `title` → añade **información extra** que aparece como un **tooltip** (cuadro emergente) al pasar el ratón sobre el enlace.

Ejemplo básico

```
<a href="https://www.google.com">Ir a Google</a>
```

- `href="https://www.google.com"`: dirección de destino.
- `Ir a Google`: texto visible en la página.

Enlaces internos y externos

- **Enlace externo**: lleva a otra página en Internet.

```
<a href="https://www.wikipedia.org">Wikipedia</a>
```

- **Enlace interno**: lleva a otra página dentro de nuesro proyecto.

```
<a href="contacto.html">Ir a la página de contacto</a>
```

Enlaces vacíos

Si aún no tienes el destino del enlace, puedes usar `#` como marcador temporal:

```
<a href="#">Enlace pendiente</a>
```

Enlaces a correo

- Los enlaces con `mailto:` abren el cliente de correo del usuario para redactar un email.
- Formato básico:

```
<a href="mailto:soporte@miweb.com">Escríbenos</a>
```

Imagenes

- Las **imágenes** se insertan con la etiqueta `<img>`.

- `<img>` es un **tag vacío (sin cierre)**: no contiene contenido dentro y no tiene etiqueta de cierre.

Atributos principales de `<img>`

- `src`: (*source*) ruta de la imagen que quieres mostrar.
  - Puede ser **externa** (URL completa) o **interna** (archivo dentro de tu proyecto).
- `alt`: texto alternativo que **describe la imagen**.
  - Es importante para **accesibilidad** (lectores de pantalla) y aparece si la imagen no carga.

  - Si la imagen es **decorativa**, usa `alt=""` (vacío).
- `width` y `height`: tamaño en **píxeles** (por defecto).
  - Mejor definir **solo uno** para mantener la proporción.

  - Si pones ambos con valores que no coinciden con la proporción original, la imagen se **deformará**.
- *(Opcional)* `title`: texto informativo que aparece como **tooltip** al pasar el ratón.

Rutas de imágenes (`src`)

- **Ruta absoluta (externa)**: apunta a Internet.
  - `src="https://misitio.com/imagenes/logo.png"`
- **Ruta relativa (interna)**: apunta a un archivo de tu proyecto.
  - Estructura típica del proyecto:

    ```
    proyecto/
      index.html
      img/
        foto.jpg
        logo.png
    ```
  - Ejemplos:
    - `src="img/foto.jpg"` (desde `index.html`)
    - `src="../assets/logo.png"` (sube un nivel y entra en `assets`)

Formatos comunes

- **JPG/JPEG**: fotos, buena compresión con pérdida.

- **PNG**: gráficos con **transparencia**, sin pérdida.

- **GIF**: animaciones simples, pocos colores.

- **SVG**: gráficos **vectoriales** (iconos, logotipos), escalan sin perder calidad.
- **WebP**: moderno, buena compresión y puede tener transparencia.

Audio

El elemento `<audio>` permite **reproducir sonido** en una página web.

Atributos clave

- `controls`: muestra los controles del reproductor (play/pausa, volumen, barra de progreso).
- `src`: ruta del archivo de audio (puede usarse directamente en `<audio>` o dentro de `<source>`).
- `type`: tipo MIME del archivo (ej. `audio/mpeg`, `audio/ogg`). MIME es un estándar que identifica el tipo de contenido.
- `autoplay`: intenta reproducir automáticamente al cargar la página. Por políticas de los navegadores, el autoplay suele **solo funcionar si el audio está `muted`** o tras interacción del usuario.
- `muted`: inicia el audio silenciado.
- `loop`: al terminar, vuelve a empezar automáticamente.

Se recomienda **varias fuentes** (`<source>`) con formatos distintos para mayor compatibilidad, y dejar un **mensaje de respaldo** por si el navegador no soporta `<audio>`.

Formatos comunes

- **MP3 (audio/mpeg)**: el más compatible; buena calidad con archivos pequeños.
- **AAC (audio/aac / audio/mp4)**: muy usado en móviles/streaming, buena calidad.
- **WAV (audio/wav)**: sin compresión; máxima calidad y tamaño grande.

Ejemplo de audio básico

```
<audio controls>
    <source src="audio/cancion.mp3" type="audio/mpeg">
    <source src="audio/cancion.ogg" type="audio/ogg">
    Tu navegador no soporta el elemento <code>audio</code>.
    Puedes <a href="audio/cancion.mp3">descargar el archivo MP3</a>.
</audio>
```

Vídeo

El elemento `<video>` permite **reproducir vídeo** en una página web.

Atributos principales de `<video>`

- `controls`: muestra los controles del reproductor (play/pausa, volumen, barra de progreso).
- `src`: ruta del archivo de vídeo (también puedes usar varias fuentes con `<source>`).
- `type`: tipo MIME del archivo (p. ej., `video/mp4`, `video/webm`, `video/ogg`).
- `autoplay`: intenta reproducir automáticamente al cargar la página.
  > Por políticas de los navegadores, suele **solo funcionar si el vídeo está `muted`**.
- `muted`: inicia el vídeo silenciado (útil en algunos casos/normal para `autoplay`).
- `loop`: al terminar, vuelve a empezar automáticamente.
- `poster`: imagen que se muestra antes de reproducir (miniatura/portada).

- `width` / `height`: tamaño del reproductor. Define **solo uno** para no deformar.

  Recomendación: usa **varias fuentes** (`<source>`) en diferentes formatos para compatibilidad y deja un **mensaje de respaldo** (con enlace de descarga) por si el navegador no soporta `<video>`.

Formatos comunes

- **MP4 (video/mp4)**: el más compatible; buena calidad y compresión.
- **WebM (video/webm)**
- **OGG (video/ogg)**

Ejemplo

```
<video src="video/demo.mp4" controls>
  Tu navegador no soporta el elemento <code>video</code>.
</video>
```

Formateado de texto

Estas etiquetas sirven para **dar formato** al texto. Recuerda: algunas tienen **significado** (semántica) y otras solo **apariencia**.

Salto de linea y separador

- `<br>` → salto de línea.
- `<hr>` → línea horizontal (separador).

```
<p>
    Primera línea<br>
    Segunda línea (después de &lt;br&gt;)
</p>
<hr>
<p>Este texto aparece tras una línea separadora (&lt;hr&gt;).</p>
```

Negrita y cursiva semánticas

- `<strong>` → importancia (negrita por defecto).
- `<em>` → énfasis (cursiva por defecto).

```
<p>Este es un <strong>mensaje importante</strong> con <em>énfasis</em>.</p>
<p>Se pueden combinar: <strong><em>muy importante</em></strong>.</p>
```

Negrita y cursiva visual

- `<b>` → negrita (sin importancia).
- `<i>` → cursiva (sin énfasis).

```
<p>Este es un <b>negrita</b> con <i>cursiva</i>.</p>
<p>Se pueden combinar: <b><i>negrita en cursiva</i></b>.</p>
```

Otros formateados útiles

- **<mark>** → resaltar texto.
- **<small>** → texto pequeño.
- **<del>** → texto eliminado (tachado).
- **<ins>** → texto insertado (subrayado).
- **<sub>** → texto subíndice.
- **<sup>** → texto superíndice.
- **<code>** → texto en formato de código.

```
<p>Este es un <mark>texto resaltado</mark>.</p>
<p>Este es un <small>texto pequeño</small>.</p>
<p>Este es un <del>texto eliminado</del>.</p>
<p>Este es un <ins>texto insertado</ins>.</p>
```

Listas

Las listas sirven para **organizar información** de manera ordenada o desordenada.

Tipos de listas

- **Lista desordenada (<ul>)**
  Los elementos aparecen con viñetas.
  Cada elemento va dentro de <li> (*list item*).

```
<ul>
  <li>Manzana</li>
  <li>Plátano</li>
  <li>Naranja</li>
</ul>
```

- **Lista ordenada (<ol>)**
  Los elementos aparecen numerados.
  Cada elemento va dentro de <li> (*list item*).

```
<ol>
  <li>Primero</li>
  <li>Segundo</li>
  <li>Tercero</li>
</ol>
```

- **Lista de definición (<dl>)**
  Los elementos aparecen como definiciones.
  Cada elemento va dentro de <dt> (término) y <dd> (definición).

```
<dl>
  <dt>HTML</dt>
  <dd>Lenguaje de marcado para la web.</dd>
  <dt>CSS</dt>
  <dd>Lenguaje de estilos para la web.</dd>
</dl>
```

- **Listas alfabéticas (<ol type="A">)**
  Los elementos aparecen con letras mayúsculas.
  Cada elemento va dentro de <li> (*list item*).

```
<ol type="A">
  <li>Primero</li>
  <li>Segundo</li>
  <li>Tercero</li>
</ol>
```

- **Listas anidadas**
  Se pueden anidar listas dentro de otras listas.

```
<ul>
  <li>Frutas
    <ul>
      <li>Manzana</li>
      <li>Plátano</li>
    </ul>
  </li>
  <li>Verduras
    <ul>
      <li>Lechuga</li>
      <li>Espinaca</li>
    </ul>
  </li>
</ul>
```

Tablas

Botones

Formularios

Metadatos

Contenido Embebido