2. HTML y CSS

Introducción

> “¿Cómo creen que los sitios web que visitamos todos los días muestran contenido, estilos y se adaptan a diferentes dispositivos?”

HTML define la estructura y contenido, mientras que **CSS** define la presentación y estilos.

HTML: Estructura básica y semántica

Estructura básica

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Mi primera página</title>
</head>
<body>
    <h1>Hola mundo</h1>
    <p>Esta es mi primera página web.</p>
</body>
</html>
```

Etiquetas semánticas

- **header:** encabezado principal de la página
- **nav:** menú de navegación
- **main / article / section / aside:** organización del contenido
- **footer:** pie de página

Actividad: Pide a los estudiantes que identifiquen estas etiquetas en un sitio web conocido usando “Inspeccionar elemento”.

Formularios y accesibilidad

```
<form action="/submit" method="POST">
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre" name="nombre" required>
    <button type="submit">Enviar</button>
</form>
```

- Explica `label` y `for` para accesibilidad.
  - for hace que se enfoque elemento con el id dentro del for
- Tip: usa atributos como `required`, `placeholder` y `aria-*` para mejorar la accesibilidad.

CSS: Estilos y diseño (25 min)

Selectores básicos

```
h1 { color: blue; }
p { font-size: 16px; }
```

Box model

- Elementos tienen **content, padding, border y margin**
- Ejemplo visual: un cuadro con borde, relleno y margen
- Mini demo: inspeccionar elemento en navegador y modificar padding/margin en vivo

Flexbox

```
.container {
    display: flex;
    justify-content: space-between;
    align-items: cetner;
}
```

- Distribución de elementos de manera flexible
- Ejercicio: crear un menú horizontal con flexbox

Grid

```
.container {
    display:grid;
    grid-template-columns: repeat(3, 1fr);
```

```
    gap: 10px;
}
```

- Distribución en filas y columnas
- Mini actividad: crear galería de imágenes simple

Responsive design (media queries)

```
@media (max-width: 600px) {
    .container {
        flex-direction: column;
    }
}
```

- Ajusta el diseño según tamaño de pantalla
- Ejercicio: hacer que el menú horizontal se vuelva vertical en móviles

Animaciones básicas

```
@keyframes mover {
    from {transform: translateX(0); }
    to {transform: translateX(100px); }
}

.box {
    animation: mover 2s infinite alternate;
}
```

- Explica `@keyframes` y propiedades de animación
    - Permite controlar los pasos intermedios en una secuencia de animación CSS (como en edición de video)

Buenas prácticas

- Usar **BEM** para nombres de clases (`.header__menu--activo`)
    - Block Element Modifier es una metodología para mejorar mantenibilidad
    - Se basa en nombrar clases de manera descriptiva para dividir la interfaz en bloques independientes (Bloque), sus partes (Elemento) y sus variaciones (Modificador)
        - tarjeta__titulo (elemento)
        - tarjeta--roja (modificador)

- Organizar estilos: reset/normalize, variables, modularización

---

Vincular HTML y CSS

CSS Externo

- Buena práctica
- Se vincula en head

```
<link rel="stylesheet" href="estilos.css">
```

Ejemplo práctico (10–15 min)

- Crear una **landing page simple**:
  - Header con título y nav
  - 2–3 secciones con contenido y una imagen
  - Footer con copyright
- Aplicar **flexbox/grid** para diseño
- Agregar un **hover** simple o una animación básica

5️⃣ Cierre y preguntas (5 min)

- Recuerda los conceptos clave: estructura HTML, etiquetas semánticas, formularios, box model, flexbox, grid, responsive, animaciones y buenas prácticas.
- Pregunta final:

> “Si quiero que mi página se vea bien en móviles y desktop, ¿qué técnicas de CSS debo usar?”