

Hoja de ruta: HTML y CSS

Construye un portfolio personal partiendo de cero. Seguirás una estrategia donde cada ejercicio se construye sobre el anterior, iterando sobre el mismo proyecto hasta tener un portfolio completo.

La estrategia: Empezarás con HTML básico, agregarás elementos paso a paso, y luego aplicarás CSS para estilizar.

Usaremos principalmente recursos gratuitos, principalmente MDN Web Docs (uno de los recursos mas completos) y web.dev. Recurso de Google para aprender desarrollo web.

Los recursos propuestos tienen siempre opción para español. Si les sale en ingles revisar la página en la parte de arriba para cambiar el idioma

Instrucciones Generales

- El taller debe entregarse en un repositorio en Github enviando el link
- Cada ejercicio debe hacerse en 1 solo commit

Ejercicio 1: Tu estructura HTML básica

HTML es lo que le dice al navegador qué mostrar—títulos, párrafos, enlaces, imágenes.

Empieza con la guía de MDN que está hecha para principiantes:

- **Recurso:** [Creating the Content](#) — MDN Web Docs
- **Qué aprenderás:** La estructura básica (`<!DOCTYPE html>`, `<html>`, `<head>`, `<body>`), cómo usar títulos (`<h1>`-`<h6>`), párrafos (`<p>`), y cómo el navegador lee todo esto

Tu tarea: Crea un archivo `index.html` con la estructura básica. Este será tu portfolio. Incluye:

- Un título con tu nombre (`<h1>`)
 - Una sección "Sobre mí" con un párrafo
 - Una sección "Proyectos" (vacía por ahora)
-

Ejercicio 2: Agrega enlaces e imágenes

Dos elementos que hacen que la web sea web: conexiones y cosas visuales.

- **Enlaces:** [Guía de Enlaces de web.dev](#)
- **Imágenes:** [Guía de Imágenes de web.dev](#)

Lo que tienes que memorizar:

```
<a href="https://ejemplo.com">Haz clic aquí</a>
```

```

```

Tu tarea: Abre tu `index.html` del Ejercicio 1 y agrega:

- Una foto tuya en la parte superior
 - Links a tu LinkedIn, GitHub, Twitter o lo que uses
 - En la sección "Proyectos", agrega 2 proyectos (pueden ser ficticios) con links a ellos
-

Ejercicio 3: Estructura semántica y listas

HTML semántico le dice al navegador y a Google qué significa cada parte de tu página.

- **Elementos semánticos:** [HTML Semántico de web.dev](#)
- **Estructura:** [Guía de MDN](#)
- **Listas:** [Tutorial de Listas de MDN](#)

Los elementos que necesitas: `<header>`, `<nav>`, `<main>`, `<section>`, `<article>`, `<footer>`. Para listas: ``, ``, ``.

Tu tarea: Refactoriza tu `index.html` usando estructura semántica:

- Envuelve todo en `<header>`, `<main>`, `<footer>`

- Agrega `<nav>` con links internos (Inicio, Proyectos, Contacto)
 - Convierte los proyectos en una lista (``) con cada uno en un ``
 - En cada proyecto, usa `` para el nombre y `` para la descripción
-

Ejercicio 4: Conecta CSS a tu proyecto

Tu HTML está hecho. Ahora vamos con la apariencia.

Aprende CSS básico:

- ¿Qué es CSS?: [Primer pasos con CSS de MDN](#)
- Selectores: [Selectores básicos de MDN](#)

Los tres selectores que necesitas:

```
h1 { }           /* Selector por tipo - afecta todos los h1 */  
  
.classname { }   /* Selector por clase - afecta elementos con class="classname" */  
  
#uniqueid { }    /* Selector por ID - afecta un solo elemento */
```

Tu tarea:

1. Crea un archivo `styles.css` en la misma carpeta que `index.html`
2. En tu HTML, agrega en `<head>`: `<link rel="stylesheet" href="styles.css">`

3. En `styles.css`, estiliza:
 - El `<h1>` con un color y tamaño más grandes
 - Los párrafos con un color y tamaño específicos
 - Los enlaces con un color diferente
-

Ejercicio 5: El modelo de caja (Box Model)

Cada elemento en CSS es una caja. Entiende esto y explicarás el 80% de cómo funciona CSS.

- La guía principal: [Box Model de MDN](#)
- Con dibujos: [Box Model de web.dev](#)

Lo importante: Contenido → Padding (dentro) → Border (borde) → Margin (afuera). Y agrega `box-sizing: border-box` al inicio.

Tu tarea: Abre `styles.css` y agrega:

1. Al inicio del archivo: `* { box-sizing: border-box; }`
2. Agrega clases a tus secciones en HTML: `class="section"`
3. En CSS, estiliza `.section` con:
 - `padding: 20px;`
 - `margin: 20px 0;`
 - `border: 1px solid #ccc;`
 - `border-radius: 8px;`

Experimenta aumentando y disminuyendo los valores.

Ejercicio 6: Colores, fondos y tipografía

- Colores y fondos: [Fondos y bordes de MDN](#)
- Tipografía: [Fundamentos de texto de MDN](#)

Propiedades clave: `color`, `background-color`, `font-family`, `font-size`, `font-weight`, `line-height`, `text-align`.

Tu tarea: Agrega a `styles.css`:

1. Estilo para `body`:

```
body {  
  
    font-family: Arial, sans-serif;  
  
    background-color: #f5f5f5;  
  
    color: #333;  
  
    line-height: 1.6;  
  
}
```

1. Estilo para `header`:

```
header {  
  
background-color: #2c3e50;  
  
color: white;  
  
padding: 30px;  
  
}
```

1. Estilo para footer:

```
footer {  
  
background-color: #2c3e50;  
  
color: white;  
  
padding: 20px;  
  
text-align: center;  
  
}
```

Ejercicio 7: Flexbox para layout

Flexbox es la herramienta moderna para acomodar elementos en filas o columnas.

- **Tutorial completo:** [Flexbox de MDN](#)
- **Práctica interactiva:** [Flexbox Froggy](#) — Un juego que enseña flexbox con 24 niveles (juega primero!)

Lo esencial:

```
.container {  
  
  display: flex;  
  
  justify-content: center;    /* Alineación horizontal */  
  
  align-items: center;       /* Alineación vertical */  
  
  gap: 20px;                 /* Espacio entre elementos */  
  
}
```

Tu tarea:

1. En tu HTML, agrega `class="nav-links"` a tu `<nav>`

1. En CSS, estiliza:

```
.nav-links {  
  
  display: flex;  
  
  gap: 20px;  
  
  justify-content: center;  
  
}
```

1. Estiliza también la lista de proyectos con flexbox si lo crees necesario

1. Experimenta con diferentes valores de `justify-content` y `gap`

Ejercicio 8: Responsive (para que se vea bien en móvil)

Haz que tu sitio funcione en cualquier tamaño de pantalla:

- Resumen rápido: [Responsive Design Basics de web.dev](#)
- Para referencia: [Responsive Design de MDN](#)

Dos cosas que tienes que agregar:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

(Agrega esto en `<head>`)

```
@media (max-width: 600px) {  
  
    /* Estilos para pantallas chicas */  
  
}
```

Tu tarea:

1. Asegúrate de que tu `index.html` tiene `<meta name="viewport">` en `<head>`

1. En `styles.css`, agrega un media query:

```
@media (max-width: 600px) {  
  
  body {  
  
    padding: 10px;  
  
  }  
  
  header {  
  
    padding: 15px;  
  
  }  
  
  .nav-links {  
  
    flex-direction: column;  
  
    gap: 10px;  
  
  }  
  
  img {  
  
    max-width: 100%;  
  
    height: auto;  
  
  }  
}
```

1. Abre tu página en el navegador y redimensiona la ventana—verás que se adapta automáticamente



Ejercicio 9: Pulido final

Haz que tu portfolio se vea profesional con los últimos detalles.

Tu tarea: Agrega los toques finales a tu portfolio:

1. Revisa los colores—¿se ven bien juntos?
1. Agrega `color: #3498db;` a los enlaces en tu CSS (o el color que prefieras)
1. Agrega transiciones suaves a los enlaces:

```
a {  
  transition: color 0.3s ease;  
}
```

```
a:hover {  
  color: #2980b9;  
}
```

1. Asegúrate de que la foto tenga estilos bonitos:

```
img {  
  max-width: 100%;
```

```
height: auto;

border-radius: 10px;

box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

}
```

Recursos completos

Tema	Recurso Principal
Estructura HTML	MDN: Creating the Content
Enlaces	web.dev: Links
Imágenes	web.dev: Images
Listas	MDN: Lists
HTML Semántico	web.dev: Semantic HTML
Intro a CSS	MDN: Getting Started
Selectores	MDN: Basic Selectors
Modelo de caja	MDN: Box Model
Colores/fondos	MDN: Backgrounds and Borders
Tipografía	MDN: Text Fundamentals
Flexbox	MDN: Flexbox
Responsive	web.dev: Responsive Basics