

# Introducción al Desarrollo web

## Curso: Desarrollo Frontend con React

# HTML5: Fundamentos del Lenguaje de Marcado

Html5

Courses Pricing

About



Unlock your  
web potential

Start learning

# Agenda



## 1 Fundamentos de HTML

Historia, evolución y papel en el desarrollo web



## 2 Estructura Básica

Elementos esenciales y sintaxis del documento HTML



## 3 Elementos Semánticos

Etiquetas con significado y estructura lógica



## 4 Tipos de Elementos

Diferencias entre elementos de bloque e en línea



## 5 Accesibilidad Web

Principios y prácticas para una web inclusiva



## 6 Desarrollo Práctico

Ejercicios guiados y mejores prácticas

# ¿Qué aprenderemos hoy?

Comprenderás la importancia de HTML5 como lenguaje fundamental de la web moderna

Aprenderás a estructurar correctamente documentos HTML siguiendo estándares actuales

Identificarás y aplicarás etiquetas semánticas para mejorar la estructura y significado

Distinguirás entre diferentes tipos de elementos y su uso adecuado en el desarrollo

Implementarás buenas prácticas de accesibilidad y optimización para buscadores

Diseñarás una página web básica aplicando todos los conceptos aprendidos



# Parte 1: Fundamentos de HTML

# ¿Qué es HTML?

HTML (HyperText Markup Language) es el lenguaje estándar para crear páginas web.

- Define la **estructura** y el **contenido** de las páginas web
- Utiliza **etiquetas** para marcar diferentes tipos de contenido
- Es interpretado por los navegadores para mostrar el contenido al usuario
- Es la **columna vertebral** de cualquier sitio web
- Trabaja junto con CSS (estilos) y JavaScript (interactividad)

HTML5 es la **quinta revisión importante** del lenguaje básico de la World Wide Web.



# HTML5: ¿Qué lo hace especial?

## Semántica Mejorada

Nuevas etiquetas con significado específico: `<header>`, `<nav>`,  
`<section>`, `<article>`, `<footer>`

## Multimedia Nativo

Soporte integrado para audio y vídeo sin plugins: `<audio>`,  
`<video>`, `<canvas>`

## Formularios Avanzados

Nuevos tipos de inputs y validación nativa: `email`, `date`, `range`,  
`search`, `tel`

## Almacenamiento Local

APIs para almacenar datos en el navegador: `localStorage`,  
`sessionStorage`, `IndexedDB`

## Conectividad

Comunicación bidireccional con el servidor: WebSockets,  
Server-Sent Events

## Rendimiento Optimizado

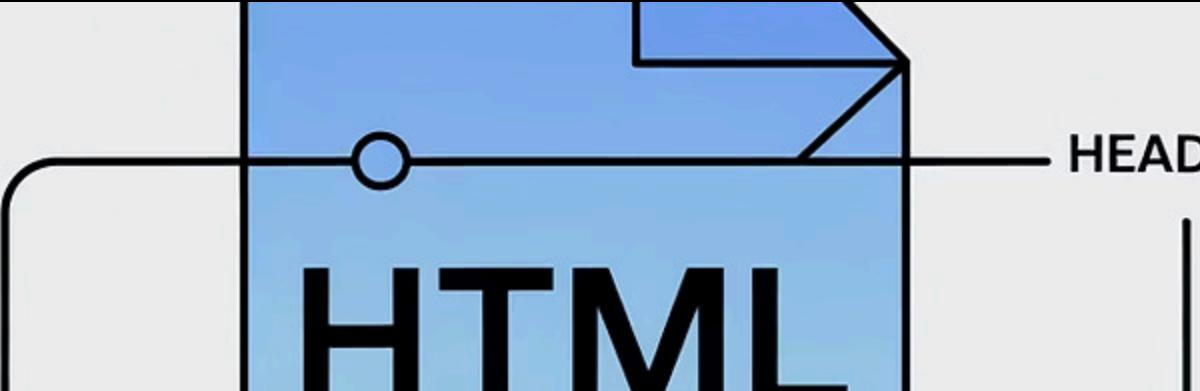
Web Workers, History API, navegación offline con Application Cache

HTML5 representa un [salto cualitativo](#) en las capacidades web, permitiendo aplicaciones mucho más ricas y funcionales.

# HTML y la Web Moderna

HTML5 forma parte de un **ecosistema tecnológico** que sustenta la web moderna:





## Parte 2: Estructura Básica del Documento HTML

# Estructura Básica de un Documento HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Título de la página</title>
  <link rel="stylesheet" href="estilos.css">
  <script src="script.js" defer></script>
</head>
<body>
  <!-- Contenido visible de la página -->
  <h1>Encabezado principal</h1>
  <p>Párrafo con contenido.</p>
</body>
</html>
```

Esta estructura representa el **esqueleto básico** que todo documento HTML5 debe tener.

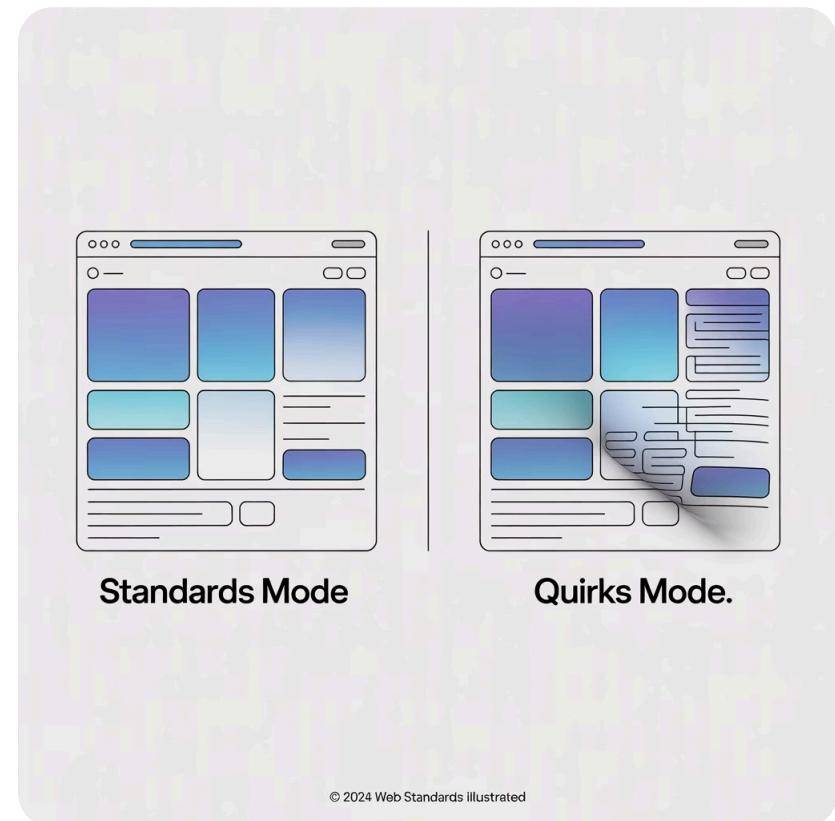
# Declaración de Tipo de Documento

## <!DOCTYPE html>

- Es la **primera línea** de todo documento HTML5
- Indica al navegador que el documento está escrito en HTML5
- A diferencia de versiones anteriores de HTML, es muy simple
- No es una etiqueta HTML, sino una **declaración**
- Es **obligatoria** para asegurar que el navegador interprete correctamente el documento

ⓘ En HTML4, el DOCTYPE era mucho más complejo e incluía referencias a DTD (Document Type Definition):

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
"http://www.w3.org/TR/html4/strict.dtd">
```



Sin DOCTYPE, los navegadores entran en "modo de compatibilidad" o "quirks mode", que puede causar inconsistencias en la visualización.

# Elemento HTML

```
<html lang="es">  
  <!-- Resto del contenido -->  
</html>
```

## Elemento Raíz

Es el **contenedor principal** de todos los demás elementos HTML (excepto DOCTYPE)

## Atributo lang

Especifica el idioma del contenido para mejorar:

- Accesibilidad (lectores de pantalla)
- SEO ( motores de búsqueda)
- Correcta interpretación de caracteres

## Buenas Prácticas

Siempre incluir el atributo lang con el código ISO correcto:

- es: español
- en: inglés
- es-ES: español de España
- es-MX: español de México

El elemento `<html>` establece el **ámbito del documento** y contiene todos los demás elementos HTML.

# Elemento HEAD

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Título de la página</title>
  <meta name="description" content="Descripción de la página">
  <link rel="stylesheet" href="estilos.css">
  <script src="script.js" defer></script>
</head>
```

Contiene **metadatos e información** sobre el documento que **no se muestra** directamente en la página.

1

## Título de la página

Se muestra en la pestaña del navegador y en los resultados de búsqueda

2

## Metadatos

Información sobre codificación, viewport, descripción, autor, etc.

3

## Enlaces a recursos

CSS, fuentes, favicons, etc.

4

## Scripts

JavaScript para funcionalidad y comportamiento

# Metadatos Esenciales

## Codificación de Caracteres

```
<meta charset="UTF-8">
```

Especifica la codificación de caracteres, permitiendo mostrar correctamente acentos, eñes y caracteres especiales. UTF-8 es el estándar recomendado actualmente.

## Viewport

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Controla cómo se muestra la página en dispositivos móviles. Esencial para el diseño responsive. Indica que el ancho de la página debe adaptarse al ancho del dispositivo.

## Título

```
<title>Título de la página</title>
```

Elemento **obligatorio** que define el título del documento mostrado en la pestaña del navegador. Es crucial para SEO y accesibilidad.

## Descripción

```
<meta name="description" content="Descripción concisa de la página">
```

Breve resumen del contenido de la página, utilizado por los motores de búsqueda para mostrar snippets en los resultados.

# Metadatos Adicionales

## Autor

```
<meta name="author" content="Nombre del Autor">
```

Identifica al creador del contenido

## Palabras Clave

```
<meta name="keywords" content="html5, web, desarrollo">
```

Aunque su relevancia para SEO ha disminuido, sigue siendo útil para categorización interna

## Robots

```
<meta name="robots" content="index, follow">
```

Controla cómo los motores de búsqueda indexan y siguen los enlaces de la página

## Open Graph

```
<meta property="og:title" content="Título">
<meta property="og:description" content="Descripción">
<meta property="og:image" content="imagen.jpg">
```

Para compartir mejor en redes sociales

## Twitter Card

```
<meta name="twitter:card" content="summary">
<meta name="twitter:title" content="Título">
```

Para optimizar la visualización en Twitter

## Tema de Color

```
<meta name="theme-color" content="#75bae6">
```

Define el color de la interfaz del navegador en móviles

Estos metadatos mejoran la [experiencia del usuario](#) y la [visibilidad](#) de tu sitio web.

# Enlaces a Recursos Externos

## Hojas de Estilo CSS

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

Vincula hojas de estilo externas para definir la presentación visual

## Favicon

```
<link rel="icon" href="favicon.ico" type="image/x-icon">
```

Icono que aparece en la pestaña del navegador

## Fuentes Web

```
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Roboto&display=swap">
```

Permite utilizar tipografías personalizadas

## Scripts

```
<script src="script.js" defer></script>
```

Enlaces a archivos JavaScript con atributos para optimizar la carga

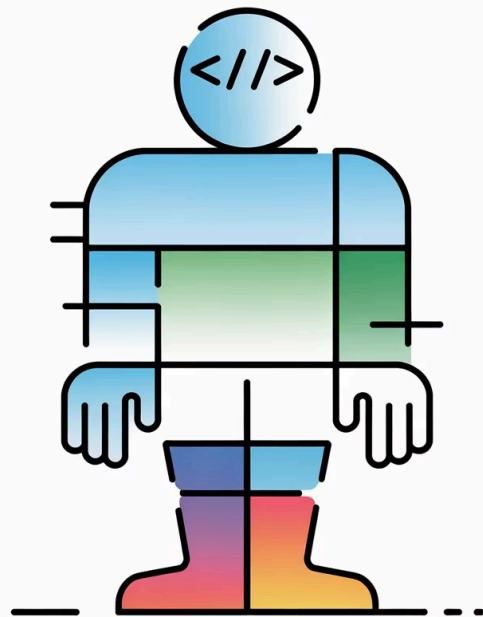
## Precargas

```
<link rel="preload" href="imagen-critica.jpg" as="image">
```

Permite cargar recursos críticos de forma anticipada

El elemento `<head>` permite **optimizar el rendimiento** mediante la gestión eficiente de recursos.

# Elemento BODY



## <body>

Contiene [todo el contenido visible](#) de la página web:

- Texto, imágenes y multimedia
- Enlaces y botones
- Formularios
- Estructuras de navegación
- Secciones y artículos
- Tablas y listas
- Cualquier otro elemento que el usuario pueda ver o con el que pueda interactuar

Es el "lienzo" donde se construye toda la experiencia de usuario. Su estructura interna debe ser [semánticamente correcta](#) para garantizar accesibilidad y SEO.

# Parte 3: Etiquetas Semánticas Clave

Percoking for finnarauns

Connect bank

Food Entertainment Rent Other

© 2024 Fintrack

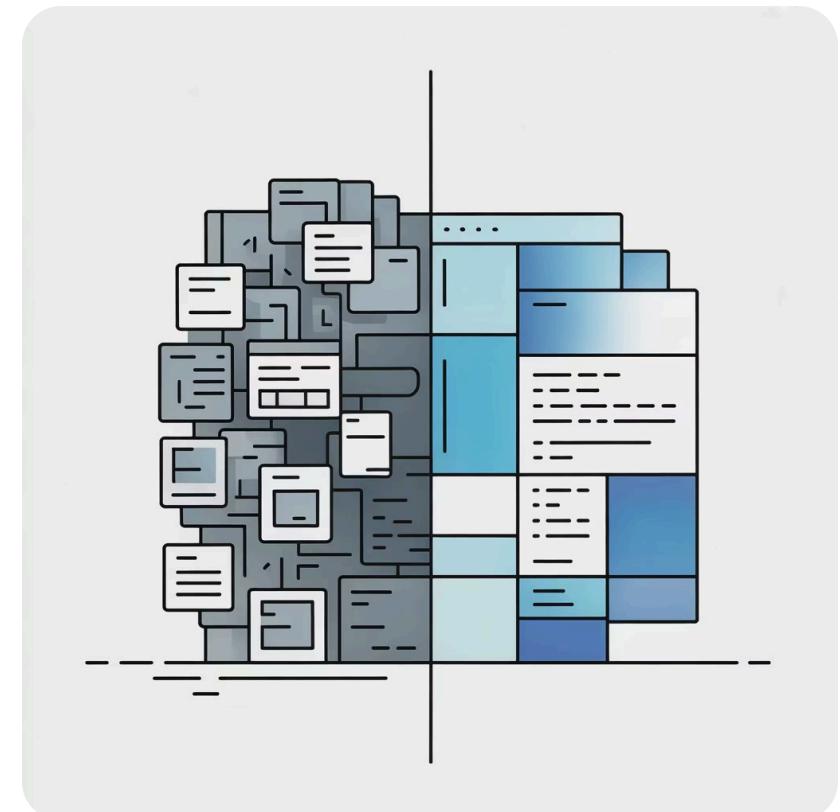
# ¿Qué es la Semántica en HTML?

La **semántica en HTML** se refiere al **significado** de los elementos, no solo a su apariencia visual.

Antes de HTML5, la mayoría de sitios web se construían principalmente con `<div>` y `<span>`, elementos que no aportan significado semántico:

```
<div id="header">...</div>
<div id="nav">...</div>
<div id="main">
  <div class="article">...</div>
  <div class="sidebar">...</div>
</div>
<div id="footer">...</div>
```

HTML5 introdujo elementos semánticos que describen explícitamente su propósito, mejorando la estructura y el significado del contenido.



El uso de etiquetas semánticas hace que el código sea más **legible, mantenible y accesible**.

# Beneficios de la Semántica en HTML5

## Accesibilidad

Los lectores de pantalla y tecnologías asistivas interpretan mejor el contenido, mejorando la experiencia para usuarios con discapacidades.

## SEO

Los motores de búsqueda comprenden mejor la estructura y el contenido de la página, lo que puede mejorar el posicionamiento.

## Mantenibilidad

El código es más legible y fácil de mantener, ya que la estructura refleja la intención del contenido.

## Reutilización

Facilita la implementación de diseños responsive y la adaptación a diferentes dispositivos y contextos.

## Futuro

Prepara el contenido para nuevas tecnologías y formas de consumo (asistentes de voz, realidad aumentada, etc.).

El HTML semántico crea una web más **estructurada, accesible e inteligente**.

# Estructura Semántica Básica

```
<body>
  <header>
    <h1>Título del sitio</h1>
    <nav>
      <ul>
        <li><a href="#">Inicio</a></li>
        <li><a href="#">Servicios</a></li>
        <li><a href="#">Contacto</a></li>
      </ul>
    </nav>
  </header>

  <main>
    <section>
      <h2>Sección principal</h2>
      <article>
        <h3>Artículo 1</h3>
        <p>Contenido...</p>
      </article>
    </section>

    <aside>
      <h2>Contenido relacionado</h2>
      <p>Enlaces, publicidad, etc.</p>
    </aside>
  </main>

  <footer>
    <p>Copyright 2023</p>
  </footer>
</body>
```



Esta estructura semántica define claramente las diferentes áreas de la página y su propósito, creando una **jerarquía lógica** del contenido.

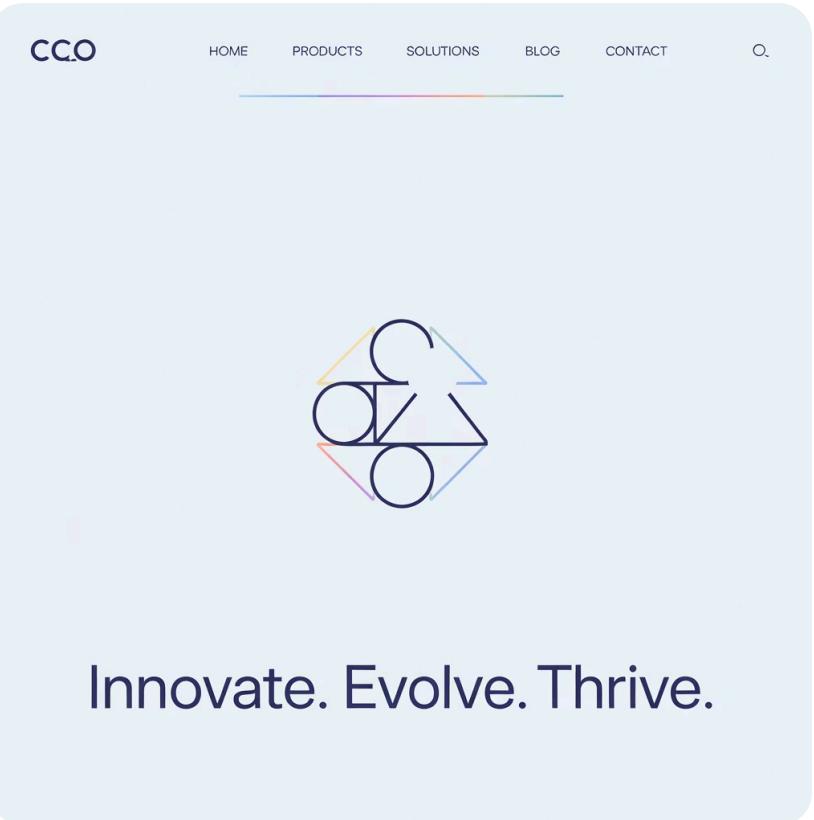
# Elemento HEADER

## <header>

Representa el [encabezado introductorio](#) de una página o sección.

- Típicamente contiene:
  - Título principal (h1-h6)
  - Logotipo
  - Navegación principal
  - Buscador
  - Información de la empresa/autor
- Puede existir más de un <header> en una página
- No confundir con <head>, que contiene metadatos

```
<header>
  
  <h1>Nombre del Sitio</h1>
  <nav>
    <!-- Menú de navegación -->
  </nav>
</header>
```



El <header> establece el contexto inicial y ayuda a los usuarios a orientarse en la página.

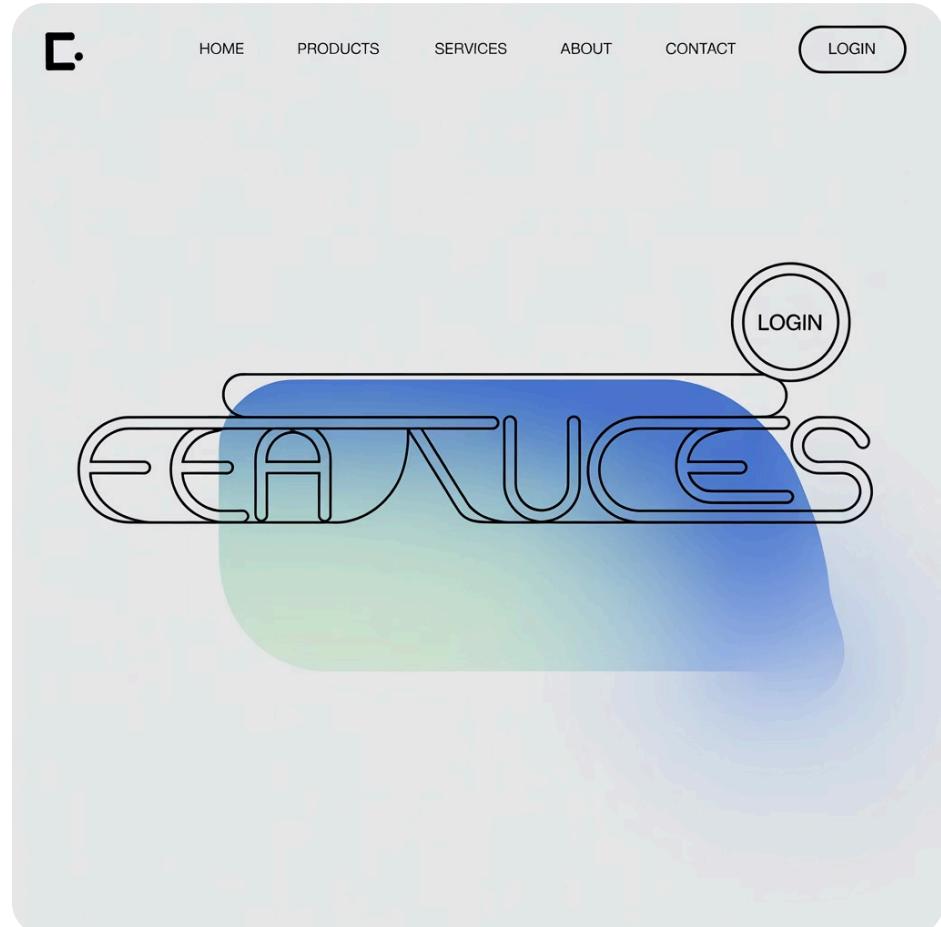
# Elemento NAV

## <nav>

Define una [sección de navegación](#) con enlaces a otras páginas o partes del sitio.

- Debe contener los enlaces principales de navegación
- No todos los grupos de enlaces necesitan estar en un <nav>
- Especialmente útil para tecnologías asistivas, que pueden identificarlo como área de navegación
- Puede haber múltiples elementos <nav> en una página (navegación principal, secundaria, de pie de página, etc.)

```
<nav>
  <ul>
    <li><a href="/">Inicio</a></li>
    <li><a href="/productos">Productos</a></li>
    <li><a href="/servicios">Servicios</a></li>
    <li><a href="/contacto">Contacto</a></li>
  </ul>
</nav>
```



Un buen <nav> mejora la [usabilidad](#) y la [accesibilidad](#) del sitio web.

# Elemento MAIN

## <main>

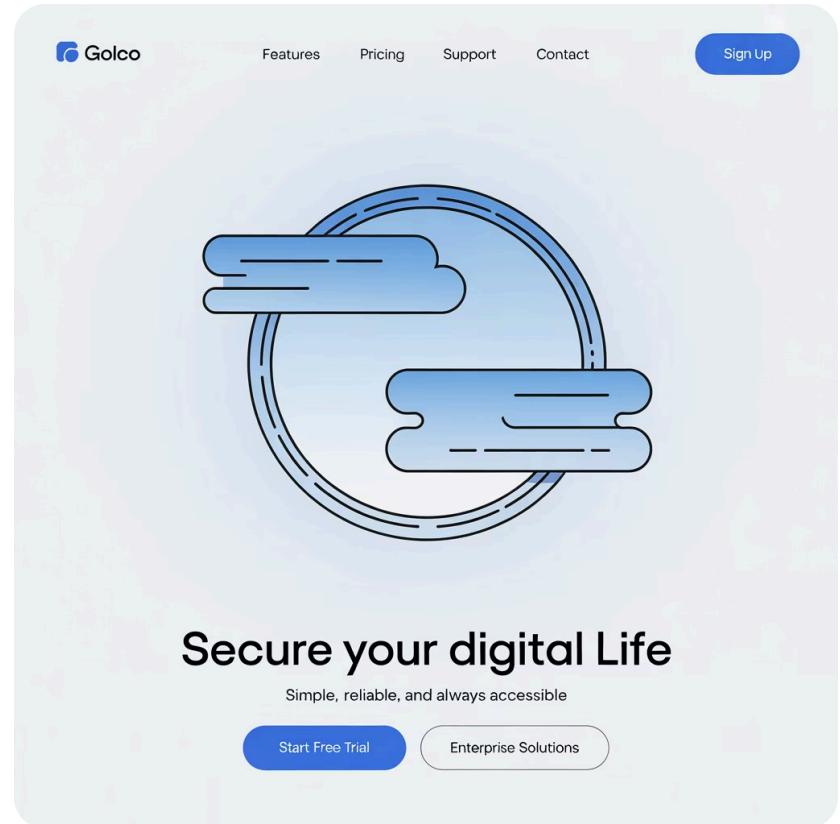
Representa el **contenido principal** y único de la página.

- Solo debe existir **un único elemento <main>** por página
- No debe estar contenido dentro de <header>, <footer>, <nav>, <aside> o <article>
- Excluye contenido que se repite en múltiples páginas (navegación, pie de página, etc.)
- Es el punto focal para tecnologías asistivas

```
<main>
  <h1>Nuestra empresa</h1>
  <p>Información sobre nuestra empresa...</p>

  <section>
    <h2>Nuestros servicios</h2>
    <!-- Contenido de servicios -->
  </section>

  <section>
    <h2>Testimonios</h2>
    <!-- Contenido de testimonios -->
  </section>
</main>
```



El elemento **<main>** ayuda a identificar rápidamente el **contenido central** de la página.

# Elemento SECTION

## <section>

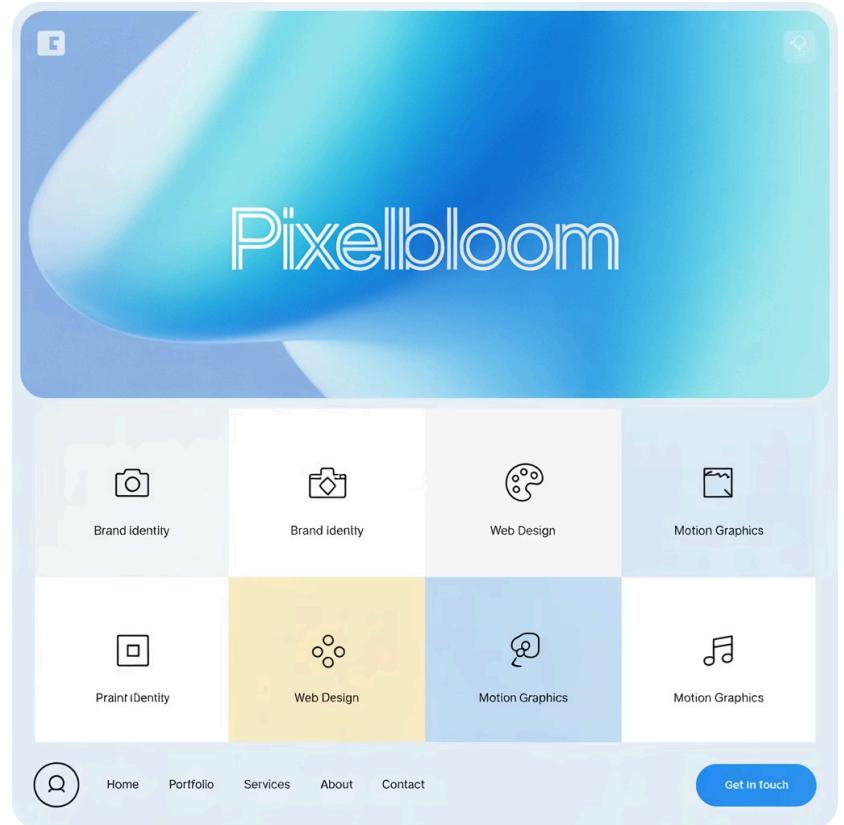
Representa una [sección genérica](#) de contenido temáticamente relacionado.

- Agrupa contenido relacionado temáticamente
- Generalmente tiene un encabezado (h1-h6)
- No usar solo para aplicar estilos (para eso está <div>)
- Puede contener múltiples <article>
- Pregunta clave: ¿Este contenido aparecería como una entrada en el índice de un libro?

```
<section>
  <h2>Características del producto</h2>
  <p>Descripción general de las características...</p>

  <article>
    <h3>Característica 1</h3>
    <p>Descripción detallada...</p>
  </article>

  <article>
    <h3>Característica 2</h3>
    <p>Descripción detallada...</p>
  </article>
</section>
```



Las secciones permiten [organizar](#) el contenido en unidades temáticas coherentes.

# Elemento ARTICLE

## <article>

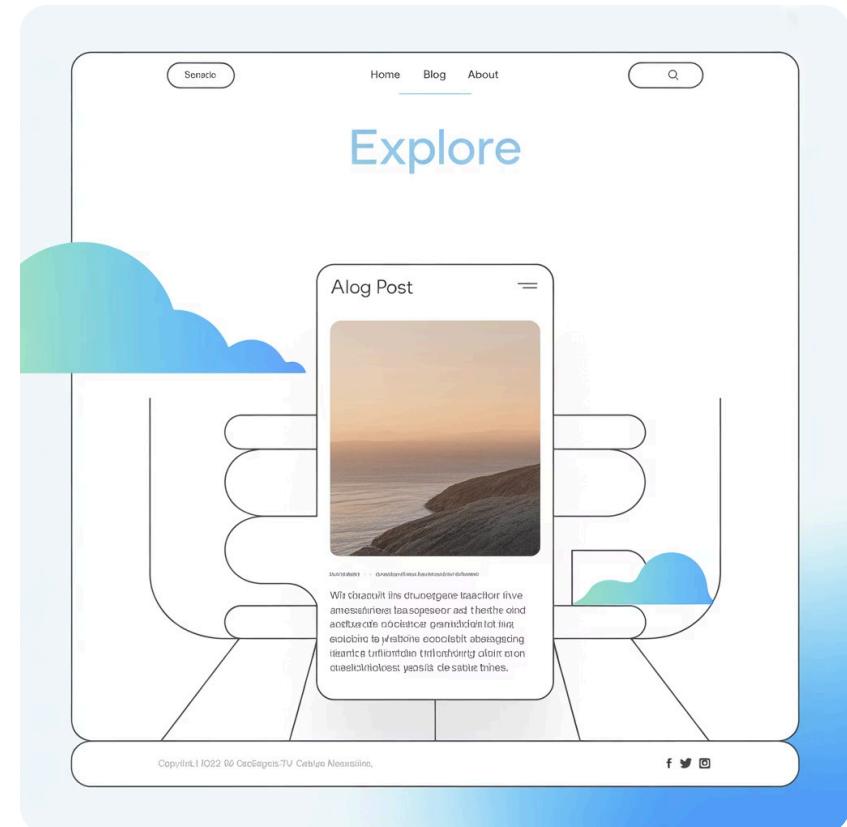
Representa una [composición autónoma](#) que puede distribuirse o reutilizarse independientemente.

- Debe tener sentido por sí mismo, fuera de contexto
- Ejemplos típicos:
  - Entradas de blog
  - Noticias
  - Comentarios
  - Productos en una tienda
  - Widgets interactivos
- Puede contener su propio <header>, <footer> y <section>

```
<article>
  <header>
    <h2>Título del artículo</h2>
    <p>Publicado el <time datetime="2023-10-15">15 de octubre de 2023</time> por <author>Autor</author></p>
  </header>

  <p>Contenido del artículo...</p>

  <footer>
    <p>Categorías: HTML5, Web</p>
  </footer>
</article>
```



Un <article> debe ser independiente y tener sentido completo por sí mismo.

# Diferencia entre SECTION y ARTICLE

## SECTION

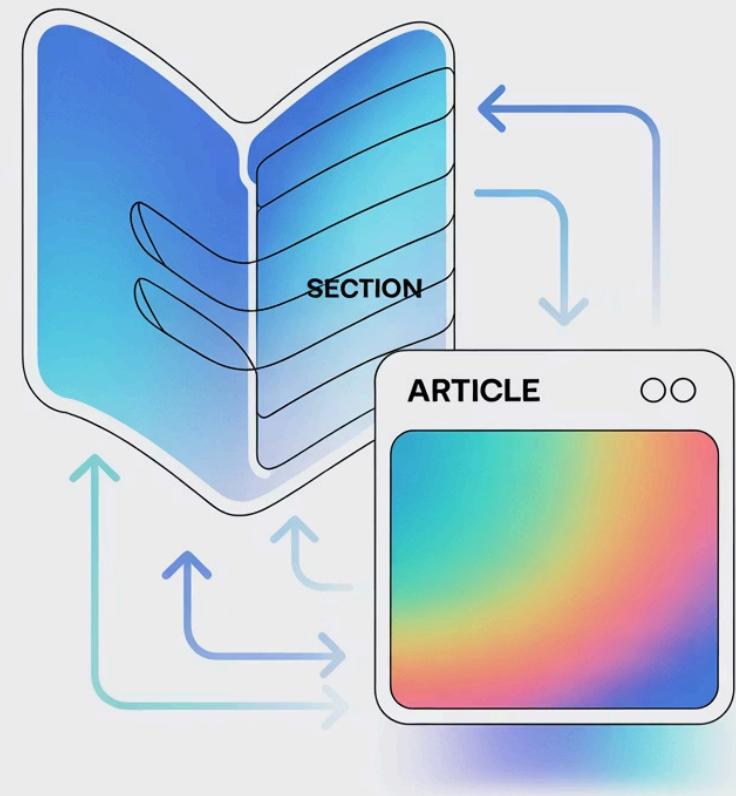
- Agrupa contenido **relacionado temáticamente**
- No necesariamente tiene sentido por sí solo
- Es como un capítulo de un libro
- Ejemplo: "Características del producto", "Testimonios"

## ARTICLE

- Contenido **independiente y autosuficiente**
- Debe tener sentido completo fuera de contexto
- Es como un artículo de revista que puede leerse por separado
- Ejemplo: "Entrada de blog", "Reseña de producto"

ⓘ Un `<article>` puede contener varias `<section>` y una `<section>` puede contener varios `<article>`, dependiendo de la estructura lógica del contenido.

# Structuring Your Webpage



# Elemento ASIDE

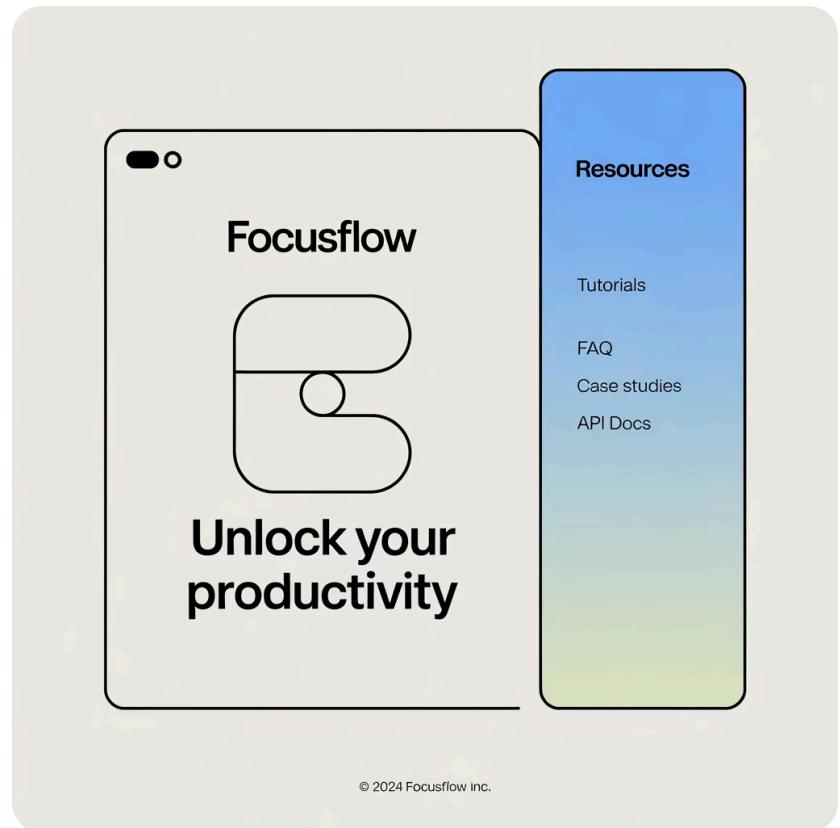
## <aside>

Representa contenido **tangencialmente relacionado** con el contenido principal.

- Contenido relacionado pero no esencial para el flujo principal
- Ejemplos típicos:
  - Barras laterales
  - Bloques de publicidad
  - Grupos de enlaces relacionados
  - Notas al margen
  - Información biográfica del autor
- Puede estar dentro o fuera del <main>

```
<aside>
  <h3>Artículos relacionados</h3>
  <ul>
    <li><a href="#">Introducción a CSS3</a></li>
    <li><a href="#">JavaScript para principiantes</a></li>
    <li><a href="#">Responsive Web Design</a></li>
  </ul>

  <div class="publicidad">
    <!-- Contenido publicitario -->
  </div>
</aside>
```



El <aside> contiene información **complementaria** que enriquece el contenido principal pero no es esencial.

# Elemento FOOTER

## <footer>

Representa un [pie de página](#) para la sección o documento.

- Típicamente contiene:
  - Información de copyright
  - Enlaces a políticas de privacidad/términos
  - Información de contacto
  - Mapa del sitio
  - Enlaces a redes sociales
  - Enlaces relacionados
- Puede haber múltiples <footer> en una página
- No necesariamente debe estar al final de la página

```
<footer>
  <p>© 2023 Mi Empresa. Todos los derechos reservados.</p>
  <nav>
    <ul>
      <li><a href="/privacidad">Política de privacidad</a></li>
      <li><a href="/terminos">Términos de uso</a></li>
      <li><a href="/contacto">Contacto</a></li>
    </ul>
  </nav>
  <div class="social-links">
    <!-- Enlaces a redes sociales -->
  </div>
</footer>
```



El <footer> proporciona [información contextual](#) y [navegación secundaria](#).

# Otros Elementos Semánticos Importantes

## FIGURE y FIGCAPTION

```
<figure>  
    
  <figcaption>Figura 1: Rendimiento anual</figcaption>  
</figure>
```

Para contenido ilustrativo (imágenes, diagramas, etc.) con su descripción

## TIME

```
<p>El evento comienza el <time datetime="2023-11-15T18:00">15 de noviembre a las 18:00</time>.</p>
```

Para representar fechas y horas en formato legible por máquinas

## MARK

```
<p>En el texto siguiente, <mark>esta parte es relevante</mark> para la búsqueda actual.</p>
```

Para resaltar texto de referencia o de interés especial

## DETAILS y SUMMARY

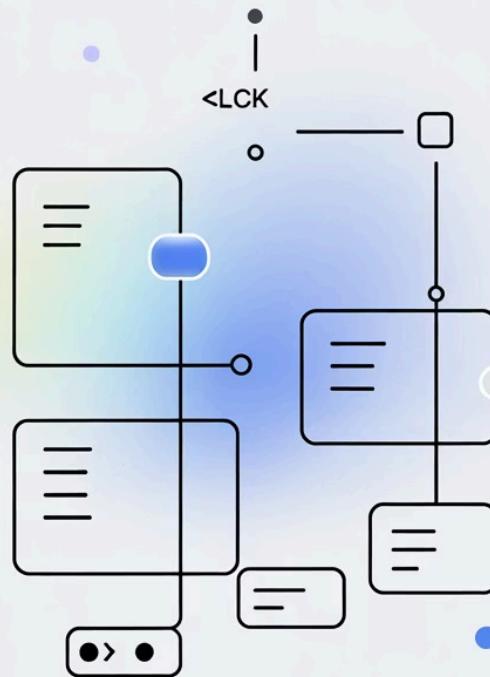
```
<details>  
  <summary>Haga clic para ver más información</summary>  
  <p>Contenido adicional que se muestra al expandir.</p>  
</details>
```

Para crear un widget de información expandible

Estos elementos proporcionan **significado específico** a diferentes tipos de contenido.

# Web Development

## HTML Tutoring



# Parte 4: Tipos de Elementos HTML

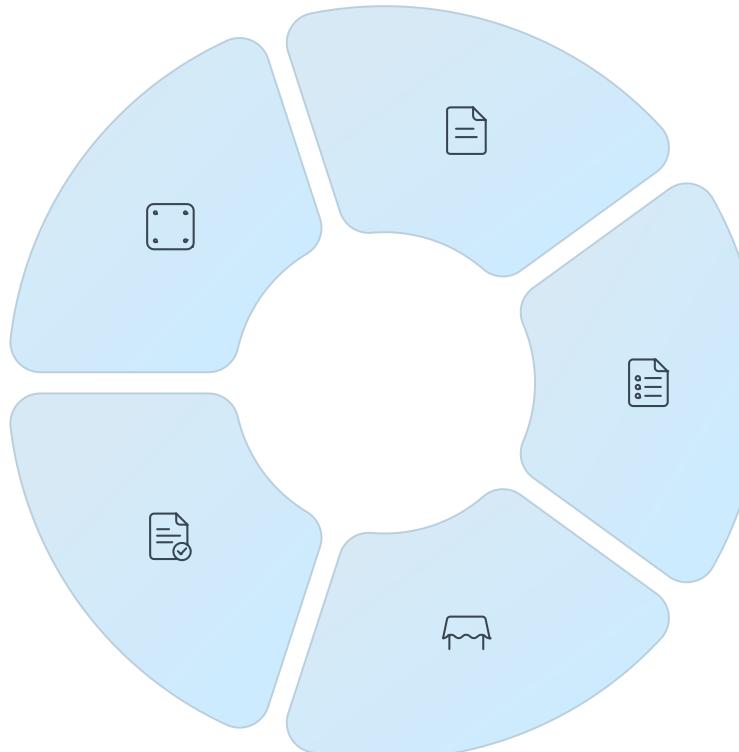
# Clasificación de Elementos HTML

## Elementos de Bloque

Ocupan todo el ancho disponible y crean nuevas líneas

## Elementos de Formulario

Para recopilar datos del usuario



## Elementos en Línea

Ocupan solo el espacio necesario y no crean nuevas líneas

## Elementos de Lista

Especializados en presentar información en forma de listas

## Elementos de Tabla

Para organizar datos en filas y columnas

Esta clasificación afecta al [flujo del documento](#) y a cómo se comportan los elementos en la página.

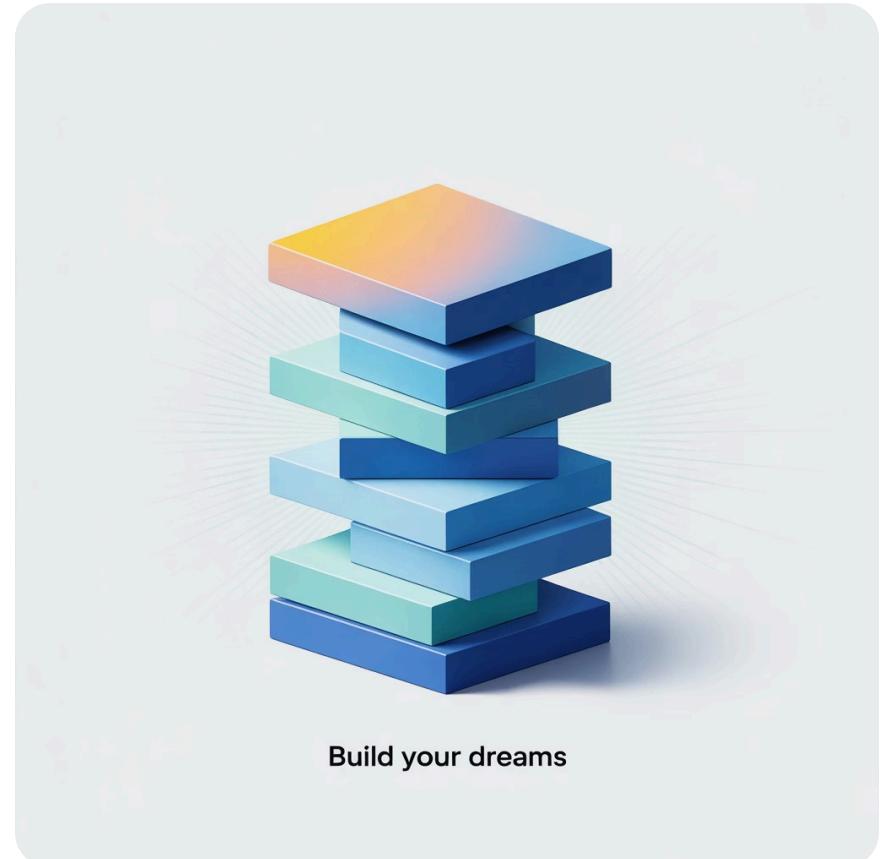
# Elementos de Bloque

Los **elementos de bloque** son aquellos que:

- Ocupan [todo el ancho disponible](#) de su contenedor
- Comienzan en una [nueva línea](#)
- Pueden contener otros elementos de bloque y elementos en línea
- Pueden tener margen y padding en todas direcciones
- Su altura se adapta automáticamente al contenido

Ejemplos de elementos de bloque:

- `<div>`: División genérica
- `<p>`: Párrafo
- `<h1>` a `<h6>`: Encabezados
- `<ul>`, `<ol>`: Listas
- Elementos semánticos: `<header>`, `<section>`, `<article>`, etc.



Los elementos de bloque crean la [estructura principal](#) de la página, dividiéndola en secciones claramente definidas.

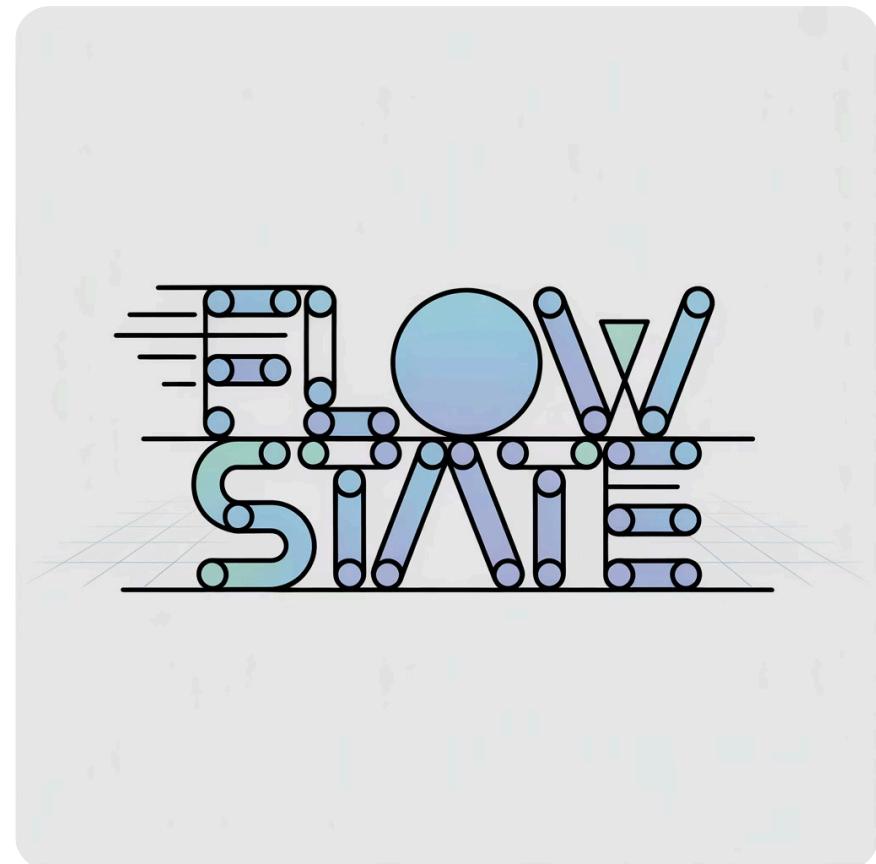
# Elementos en Línea

Los **elementos en línea** son aquellos que:

- Ocupan **solo el espacio necesario** para su contenido
- **No crean nuevas líneas** (fluyen dentro del texto)
- No pueden contener elementos de bloque
- Solo pueden tener margen y padding horizontal (no vertical)
- Se ven afectados por las propiedades de texto (font-size, line-height, etc.)

Ejemplos de elementos en línea:

- `<span>`: Contenedor genérico en línea
- `<a>`: Enlaces
- `<strong>`, `<em>`: Énfasis
- `<img>`: Imágenes (comportamiento especial)
- `<br>`: Salto de línea
- `<code>`, `<abbr>`, `<cite>`: Elementos semánticos en línea



Los elementos en línea permiten **formatear y enriquecer** el contenido de texto sin interrumpir su flujo natural.



## Ejemplos Visuales: Elementos de Bloque vs. En Línea

```
<div>Este es un elemento de bloque.</div>
<p>Este párrafo ocupa todo el ancho disponible.</p>
<h2>Este encabezado también es un elemento de bloque.</h2>
```

```
<p>Este texto contiene <strong>texto en negrita</strong>,
<em>texto en cursiva</em> y un <a href="#">enlace</a>
que son elementos en línea.</p>
```

Entender la diferencia entre elementos de bloque y en línea es **fundamental** para crear estructuras HTML correctas y aplicar estilos CSS adecuadamente.

# Elementos Especiales: Inline-Block

Los elementos **inline-block** son un [híbrido](#) que combina características de ambos tipos:

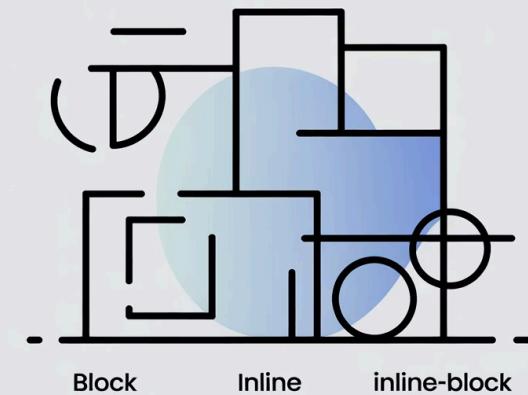
- Fluyen [en línea](#) como los elementos inline
- Pueden tener [dimensiones definidas](#) (ancho/alto) como los de bloque
- Respetan márgenes y padding en todas direcciones
- No fuerzan saltos de línea

Ejemplos de elementos que se comportan como inline-block por defecto:

- `<img>`: Imágenes
- `<button>`: Botones
- `<input>`: Campos de formulario
- `<select>`: Listas desplegables

Cualquier elemento puede convertirse en inline-block mediante CSS:

```
.mi-elemento {  
    display: inline-block;  
}
```



Los elementos inline-block son útiles para crear [layouts horizontales](#) donde necesitamos controlar dimensiones.

# Anidamiento de Elementos

## Reglas de Anidamiento

- Los elementos de **bloque** pueden contener:
  - Otros elementos de bloque
  - Elementos en línea
  - Texto
- Los elementos **en línea** solo pueden contener:
  - Otros elementos en línea
  - Texto
- ¡Los elementos en línea NO pueden contener elementos de bloque!

## Ejemplos

✓ Correcto:

```
<div>
  <p>Este es un <strong>párrafo</strong></p>
  <ul>
    <li>Un elemento de lista</li>
  </ul>
</div>

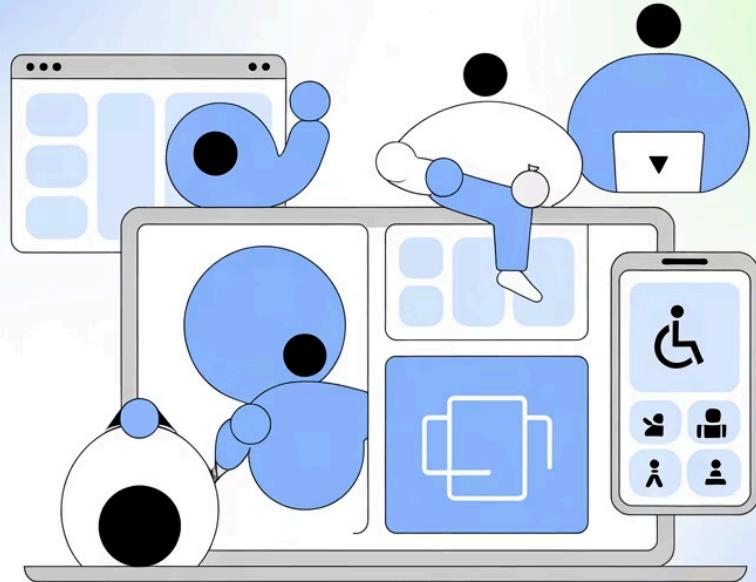
<p>Un párrafo con <a href="#">un enlace</a></p>
```

✗ Incorrecto:

```
<span>
  <p>Este párrafo no debería estar aquí</p>
</span>

<a href="#">
  <div>Este div no debería estar aquí</div>
</a>
```

Respetar las reglas de anidamiento es crucial para crear **HTML válido** y evitar comportamientos inesperados.



# Designing for Everyone

Empowering users of all abilities

## Parte 5: Accesibilidad y Uso Semántico

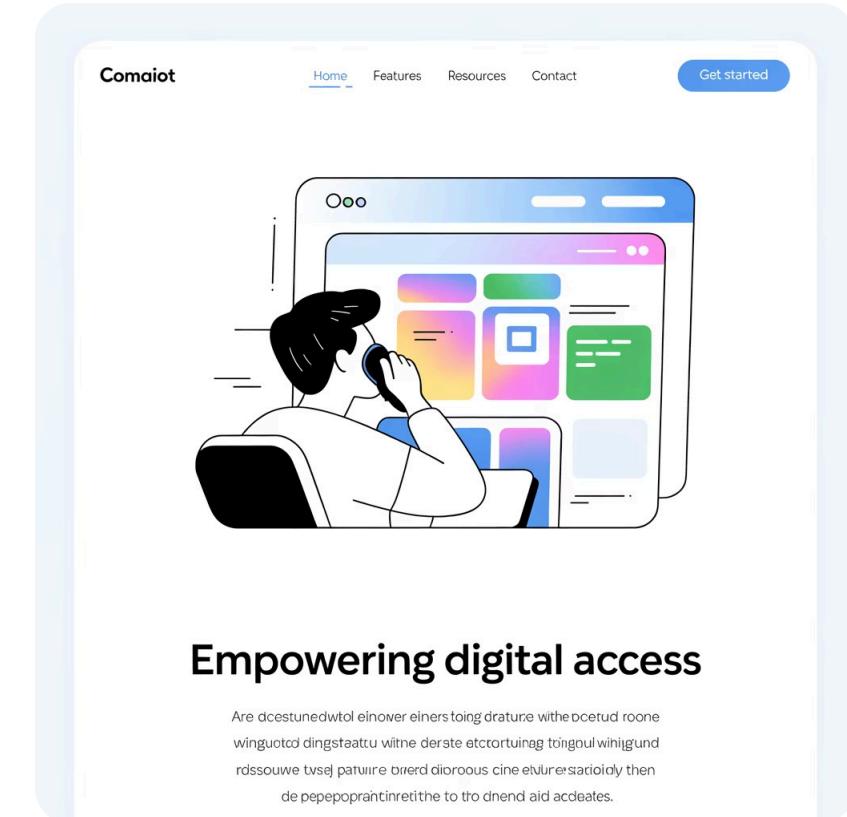
# ¿Qué es la Accesibilidad Web?

La **accesibilidad web** significa que los sitios web, herramientas y tecnologías están diseñados y desarrollados para que [todas las personas puedan usarlos](#), incluyendo aquellas con:

- Discapacidades visuales
- Discapacidades auditivas
- Discapacidades motoras
- Discapacidades cognitivas
- Discapacidades neurológicas
- Limitaciones temporales o situacionales

Según la OMS, aproximadamente el [15% de la población mundial vive con algún tipo de discapacidad](#).

En muchos países, la accesibilidad web es un [requisito legal](#) para sitios gubernamentales y servicios públicos.



## Empowering digital access

Are dcestunedwtol einover einerstoing drature withe ocetud roone  
winguotcd dingstaattu witne derste etccorturing toringulwhigund  
rdissouwe twsej patuire brered diorous cine elulure satoioiy then  
de pepepopratinretithe to tho dnend aid acdeates.

La accesibilidad no es solo para personas con discapacidades permanentes; beneficia a todos los usuarios en diferentes contextos y situaciones.

# Principios WCAG

Las **Web Content Accessibility Guidelines (WCAG)** son el estándar internacional para la accesibilidad web, desarrollado por el W3C.

## Perceptible

La información debe ser presentada de manera que los usuarios puedan percibirla, independientemente de sus capacidades sensoriales.

- Alternativas textuales para contenido no textual
- Alternativas para multimedia
- Contenido adaptable y distinguible

## Operable

Los usuarios deben poder interactuar con todos los controles y funcionalidades de la interfaz.

- Accesibilidad mediante teclado
- Tiempo suficiente para interactuar
- Evitar contenido que cause convulsiones
- Navegación fácil y eficiente

## Comprensible

La información y el funcionamiento de la interfaz deben ser claros y predecibles.

- Contenido legible y comprensible
- Funcionamiento predecible
- Ayuda para evitar y corregir errores

## Robusto

El contenido debe ser suficientemente robusto para funcionar con diferentes tecnologías, incluidas las asistivas.

- Compatibilidad con tecnologías actuales y futuras
- Código limpio y estándar

Estos principios se conocen por el acrónimo **POUR** (Perceptible, Operable, Understandable, Robust).

# HTML Semántico y Accesibilidad

El **HTML semántico** es la base de la accesibilidad web porque:

- Proporciona **significado implícito** a los elementos que ayuda a las tecnologías asistivas
- Crea una **estructura lógica** que facilita la navegación
- Define **roles predeterminados** que las tecnologías asistivas reconocen
- Mejora la **experiencia de usuario** para todos

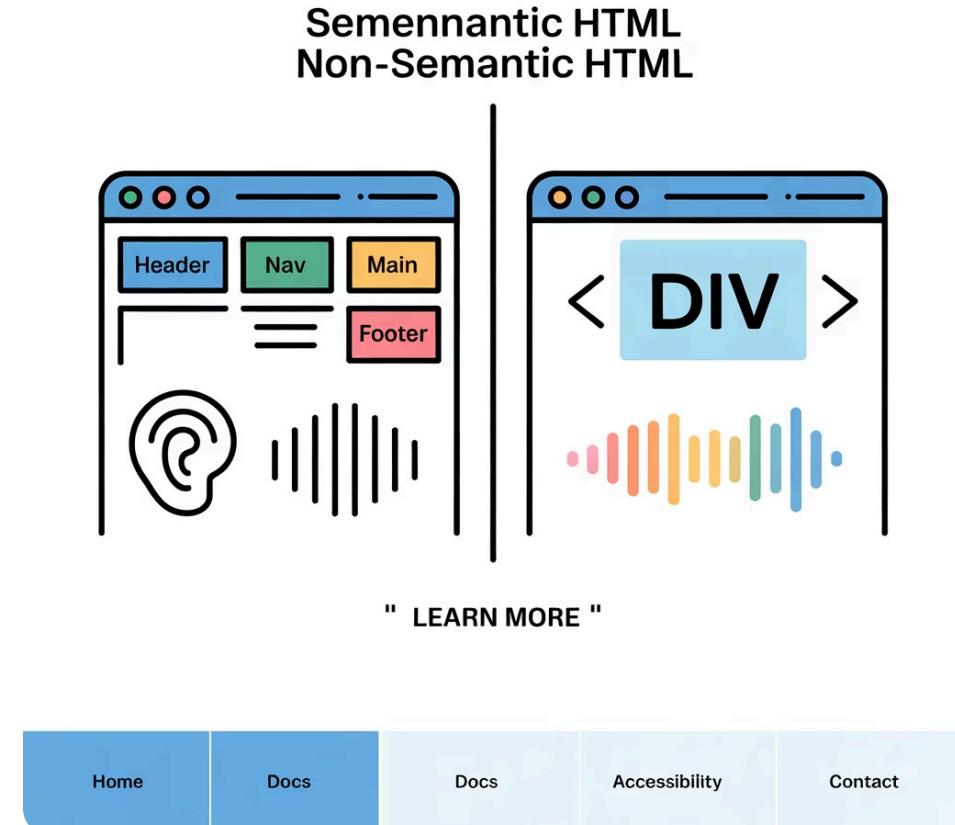
Comparación de enfoque no semántico vs. semántico:

**No Semántico**

```
<div class="header">...</div>
<div class="nav">...</div>
<div class="main">...</div>
<div class="footer">...</div>
```

**Semántico**

```
<header>...</header>
<nav>...</nav>
<main>...</main>
<footer>...</footer>
```



El HTML semántico transmite **significado**, no solo estructura visual, lo que es crucial para usuarios de tecnologías asistivas.

# Elementos y Atributos para Accesibilidad

## Texto Alternativo

```

```

El atributo `alt` proporciona una descripción textual de las imágenes para usuarios que no pueden verlas.

## ARIA Landmarks

```
<div role="search">  
  <form>...</form>  
</div>
```

ARIA (Accessible Rich Internet Applications) proporciona roles, estados y propiedades adicionales para mejorar la accesibilidad.

## Etiquetas de Formulario

```
<label for="nombre">Nombre:</label>  
<input type="text" id="nombre"  
  name="nombre">
```

Asociar etiquetas con controles de formulario mejora la usabilidad y accesibilidad.

## Títulos y Encabezados

```
<h1>Título Principal</h1>  
<h2>Subtítulo</h2>  
<h3>Sección</h3>
```

Una jerarquía clara de encabezados permite a los usuarios de lectores de pantalla navegar por el contenido eficientemente.

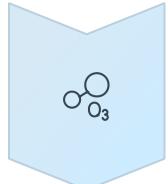
## TabIndex

```
<div tabindex="0">Contenido interactivo</div>
```

Controla el orden de navegación mediante teclado, permitiendo que elementos no estándar reciban el foco.

Estos elementos y atributos hacen que tu contenido sea [accesible para todos](#), independientemente de sus capacidades o dispositivos.

# Técnicas para Mejorar la Accesibilidad



## Estructura Lógica

Usa elementos semánticos y una jerarquía clara de encabezados (h1-h6) que reflejen la estructura del contenido.



## Imágenes Accesibles

Proporciona texto alternativo para todas las imágenes y usa imágenes decorativas con alt="" o CSS.



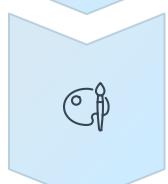
## Accesibilidad por Teclado

Asegúrate de que todas las funcionalidades sean accesibles mediante teclado, con indicadores visuales de enfoque.



## Formularios Claros

Etiqueta todos los campos, proporciona instrucciones y mensajes de error accesibles.



## Contraste Suficiente

Asegura un contraste adecuado entre el texto y el fondo (4.5:1 para texto normal, 3:1 para texto grande).

# Validación y Pruebas de Accesibilidad

## Herramientas de Validación

- **Validador W3C**: Verifica que el HTML sea válido y bien formado
- **WAVE**: Evaluación visual de accesibilidad que muestra errores directamente en la página
- **Lighthouse**: Integrado en Chrome DevTools, incluye auditorías de accesibilidad
- **axe**: Motor de pruebas de accesibilidad automatizadas
- **Screen readers**: NVDA (Windows), VoiceOver (Mac/iOS), JAWS, TalkBack (Android)

## Pruebas Manuales

- **Navegación por teclado**: Intenta usar el sitio solo con teclado (Tab, Enter, teclas de flecha)
- **Zoom**: Aumenta el zoom del navegador al 200% y verifica que el contenido siga siendo usable
- **Lectores de pantalla**: Experimenta con un lector de pantalla para entender cómo interpretan tu sitio
- **Simuladores de daltonismo**: Verifica cómo se ve tu sitio para personas con diferentes tipos de daltonismo
- **Deshabilitar estilos**: Verifica que el contenido tenga sentido sin CSS



# Parte 6: Estructura de una Página Simple

# Anatomía de una Página Web Simple

## **Header (Encabezado)**

Contiene el logo, título del sitio, navegación principal y posiblemente un buscador

## **Navigation (Navegación)**

Enlaces principales para moverse por el sitio

## **Main Content (Contenido Principal)**

El contenido único y específico de la página actual

## **Sidebar (Barra Lateral)**

Información complementaria, enlaces relacionados o publicidad

## **Footer (Pie de Página)**

Información de contacto, enlaces secundarios, copyright y legales

# Estructura HTML5 Básica

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Mi Página Web</title>
  <link rel="stylesheet" href="estilos.css">
</head>
<body>
  <header>
    <h1>Mi Sitio Web</h1>
    <nav>
      <ul>
        <li><a href="#">Inicio</a></li>
        <li><a href="#">Acerca de</a></li>
        <li><a href="#">Servicios</a></li>
        <li><a href="#">Contacto</a></li>
      </ul>
    </nav>
  </header>

  <main>
    <section>
      <h2>Bienvenidos a mi sitio</h2>
      <p>Contenido principal...</p>
    </section>
  </main>

  <footer>
    <p>© 2023 Mi Sitio Web. Todos los derechos reservados.</p>
  </footer>
</body>
</html>
```

Esta estructura proporciona un **esqueleto básico** para una página web simple y semánticamente correcta.

# Variaciones de Estructura Común

## One-Page Site

Una única página con secciones enlazadas mediante navegación de anclaje interna.

```
<nav>
  <a href="#section1">Sección 1</a>
</nav>
...
<section id="section1">...</section>
```

## Blog

Lista de artículos con fecha, autor y contenido, generalmente con comentarios.

```
<article>
  <header>
    <h2>Título del Post</h2>
    <time datetime="2023-10-15">15/10/2023</time>
  </header>
  <p>Contenido...</p>
</article>
```

## E-commerce

Catálogo de productos con fichas, carrito, proceso de compra.

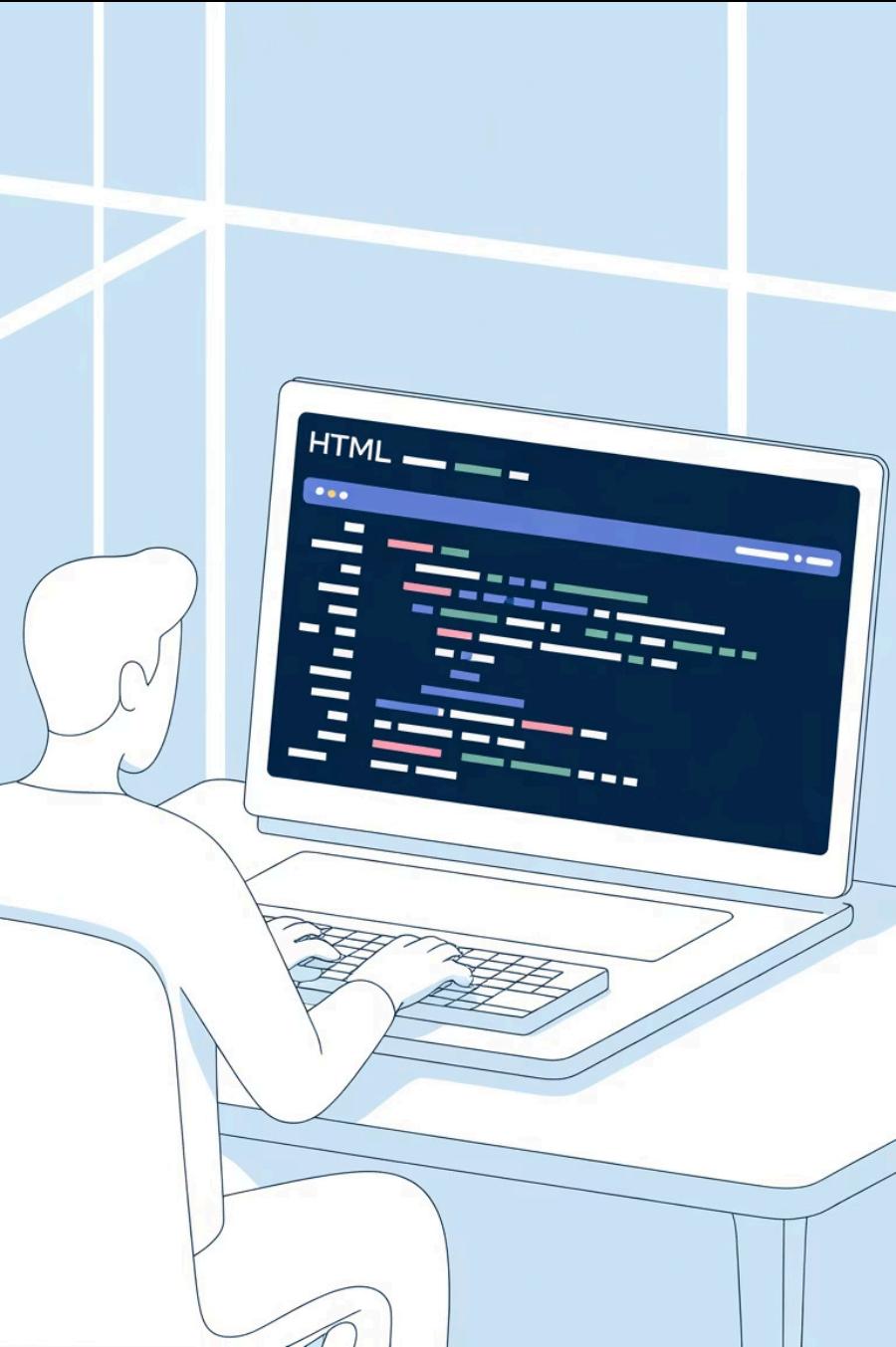
```
<section class="productos">
  <article class="producto">
    
    <h3>Nombre del producto</h3>
    <p class="precio">29,99 €</p>
    <button>Añadir al carrito</button>
  </article>
</section>
```

## Dashboard

Interfaz con paneles, widgets y controles para aplicaciones.

```
<main class="dashboard">
  <section class="widget">
    <h3>Estadísticas</h3>
    <div class="grafico">...</div>
  </section>
</main>
```

Cada tipo de sitio tiene sus propias [convenciones estructurales](#), pero todos deben seguir los principios semánticos básicos.



# Parte 7: Buenas Prácticas

# Indentación y Formato

## Código Bien Indentado

```
<section>
  <h2>Título de la sección</h2>
  <p>Primer párrafo con <a href="#">un enlace</a>
    dentro del texto.</p>
  <div class="contenedor">
    <ul>
      <li>Primer elemento</li>
      <li>Segundo elemento</li>
    </ul>
  </div>
</section>
```

Un código bien indentado es [más legible](#) y [fácil de mantener](#).

- Usa [2 o 4 espacios](#) para cada nivel de indentación (consistentemente)
- Indenta el contenido dentro de cada elemento
- Alinea las etiquetas de apertura y cierre
- Utiliza un editor con resaltado de sintaxis y autoindentación

## Código Mal Indentado

```
<section>
  <h2>Título de la sección</h2>
  <p>Primer párrafo con <a href="#">un enlace</a> dentro del texto.</p>
  <div class="contenedor">
    <ul>
      <li>Primer elemento</li>
      <li>Segundo elemento</li>
    </ul>
  </div>
</section>
```

Un código mal indentado dificulta la identificación de la estructura y la detección de errores.

# Anidamiento Correcto

## Anidamiento Correcto

```
<div>
  <p>Este es un <strong>párrafo</strong>
    con texto en negrita.</p>
  <ul>
    <li>Primer elemento</li>
    <li>Segundo <em>elemento</em></li>
  </ul>
</div>
```

## Anidamiento Incorrecto

```
<div>
  <p>Este es un <strong>párrafo</p>
  </strong> con texto en negrita.
  <ul>
    <li>Primer elemento
    <li>Segundo <em>elemento</li></em>
  </ul>
</div>
```

Las etiquetas se cierran en el **orden inverso** al que se abrieron.

Este código tiene **errores de anidamiento** que pueden causar problemas de renderizado y accesibilidad.

- Las etiquetas deben cerrarse en el **orden inverso** al que se abrieron
- No "cruzar" etiquetas (abrir A, abrir B, cerrar A, cerrar B)
- Cerrar todas las etiquetas (excepto las auto-cerradas como `<img>`)
- Utilizar un validador HTML para detectar errores de anidamiento

# Uso de Comentarios

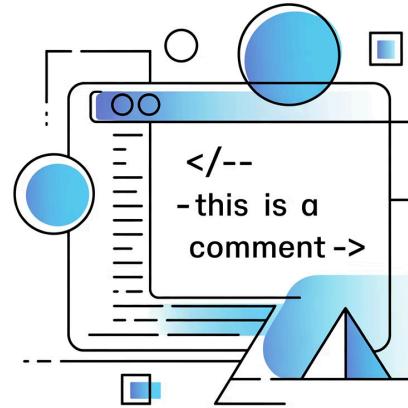
Los **comentarios en HTML** son útiles para:

- Explicar secciones complejas
- Marcar el inicio y fin de grandes bloques
- Documentar el propósito de ciertas decisiones
- Dejar notas para otros desarrolladores (o para ti mismo)
- Comentar temporalmente código que no queremos eliminar

Sintaxis: <!-- Texto del comentario -->

```
<!-- Header principal del sitio -->
<header>
  <h1>Título del sitio</h1>
  <!-- Navegación principal -->
  <nav>
    <ul>
      <!-- TO-DO: Añadir más enlaces -->
      <li><a href="#">Inicio</a></li>
      <li><a href="#">Productos</a></li>
    </ul>
  </nav>
</header>

<!-- Contenido principal -->
<main>
  <section id="productos">
    <!-- Lista de productos destacados -->
    <!-- Actualizado: 15/10/2023 por Juan -->
    ...
  </section>
</main>
```



Los comentarios **no se muestran** en el navegador, pero son visibles en el código fuente.

i No uses comentarios para información sensible, ya que cualquiera puede verlos al inspeccionar el código fuente.

En proyectos grandes, es común utilizar comentarios para marcar el inicio y fin de secciones importantes:

```
<!-- INICIO: Sección de testimonios -->
...
<!-- FIN: Sección de testimonios -->
```

# Convenciones de Nombres

## IDs y Clases

- Usa nombres descriptivos que indiquen el propósito, no la apariencia
- id="nav-principal", class="tarjeta-producto"
- id="div1", class="azul-grande"

## Formatos Comunes

- **kebab-case**: menu-principal, boton-enviar (recomendado para HTML)
- **camelCase**: menuPrincipal, botonEnviar
- **snake\_case**: menu\_principal, boton\_enviar

## Nombres de Archivos

- Usa minúsculas para evitar problemas en servidores case-sensitive
- Evita espacios y caracteres especiales
- pagina-contacto.html, imagen-principal.jpg
- Página Contacto.html, imagen#1.jpg

## Metodologías

- **BEM** (Block Element Modifier): bloque\_elemento--modificador
- **SMACSS** (Scalable and Modular Architecture for CSS)
- **OOCSS** (Object Oriented CSS)

Seguir convenciones consistentes hace que el código sea más **mantenible** y **comprendible** para todos los miembros del equipo.

# Optimización y Rendimiento

## Estructura Limpia

- Evita el "div soup" (abuso de divs sin propósito semántico)
- Reduce la profundidad de anidamiento innecesaria
- Elimina código HTML no utilizado

## Imágenes

- Usa atributos width y height para evitar cambios de diseño durante la carga
- Utiliza formatos modernos como WebP cuando sea posible
- Implementa imágenes responsive con srcset y sizes

## Carga de Recursos

- Usa lazy loading para imágenes fuera de la vista inicial
- Coloca los scripts al final del body o usa defer/async
- Prioriza el contenido crítico

## Optimización de Código

- Minifica HTML, CSS y JavaScript en producción
- Utiliza compresión GZIP o Brotli en el servidor
- Implementa una estrategia de caché efectiva

Un HTML bien optimizado mejora la [velocidad de carga](#), la [experiencia de usuario](#) y el [posicionamiento SEO](#).

# Validación de Código

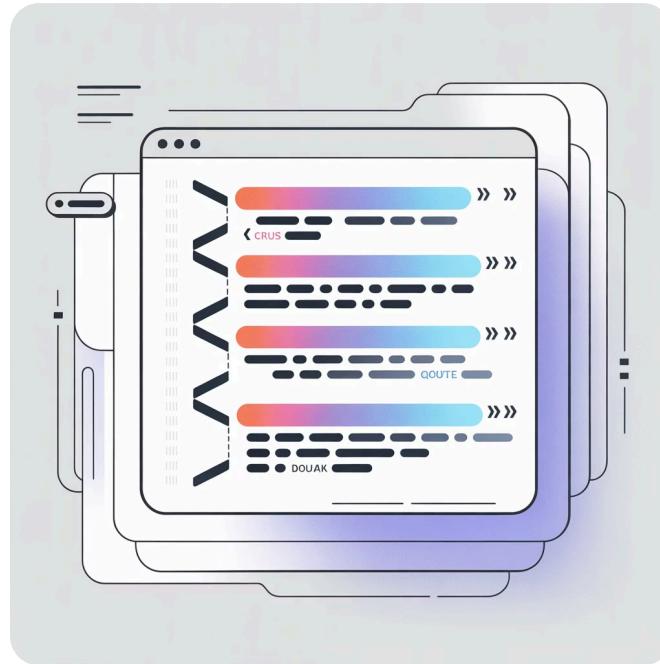
La **validación** consiste en verificar que tu HTML cumple con los estándares y no contiene errores sintácticos.

## Beneficios de la Validación

- Garantiza la compatibilidad entre navegadores
- Mejora la accesibilidad
- Facilita el mantenimiento futuro
- Previene comportamientos inesperados
- Puede mejorar el rendimiento
- Es una buena práctica profesional

## Herramientas de Validación

- **Validador W3C:** [validator.w3.org](https://validator.w3.org)
- **HTML5 Validator:** [html5.validator.nu](https://html5.validator.nu)
- **Extensiones para editores:** VS Code, Sublime Text, etc.
- **Herramientas integradas** en sistemas de integración continua



La validación debe ser un **paso regular** en tu flujo de trabajo de desarrollo.

```
<!-- Ejemplo de error común -->
<p>Texto con <strong>negrita</p></strong>
```

```
<!-- Error al validar -->
Error: Etiqueta "strong" no cerrada correctamente
```



MENU LOCATION ABOUT CONTACT

COFFEE NOW

# THE DAILY GRIND

LOCAL CROISSANT COFFEE AND CAKES  
UPFARNOHCEPEELLOFFE SLICE  
MISSITBOECROTINH  
OFFICE CUTLIBOSEE.

CONFETTI



INFORMACIÓN SOBRE

Acerca de Nosotros  
Localización  
Anotaciones  
Reservar

ACERCA DE NOSOTROS

Actualizaciones  
Localización  
Anotaciones  
Reservar

NOTICIAS  
Últimas actualizaciones  
Última actualización  
Reservar

CONTACTO  
Comunicarse & Preguntas  
Preguntas & Respuestas  
Contacto

# Proyecto Web

## Diseño HTML de un Sitio Web para Cafetería

Vamos a aplicar todos los conceptos aprendidos para crear un sitio web simple y semánticamente correcto para una cafetería local. Incluiremos elementos esenciales como el menú, la ubicación, los horarios de apertura y una galería de productos.

# Especificaciones del Sitio Web para Cafetería

## Estructura Básica

Crea un documento HTML5 completo con las secciones clave para una cafetería:

- Header con el logo/nombre de la cafetería y navegación principal (Inicio y Menú)
- Sección principal de bienvenida o "Acerca de Nosotros"
- Sección detallada del Menú de productos (bebidas, comidas, postres)
- Footer con información de copyright, ubicación y contacto

## Requisitos Semánticos

Utiliza correctamente las etiquetas semánticas para una estructura clara:

- Estructura lógica con `<header>`, `<nav>`, `<main>`, `<section>`, `<article>` (si aplica), `<aside>` (si aplica), `<footer>`
- Jerarquía adecuada de encabezados (`<h1>` a `<h6>`) para organizar el contenido
- Uso apropiado de elementos de bloque e en línea según su función
- Listas ordenadas o desordenadas para los elementos del menú o puntos clave

## Validación

Tu código debe ser válido y libre de errores:

- El código HTML debe pasar la validación del W3C sin errores ni advertencias

# Conclusiones y Reflexiones Finales

## El HTML Semántico como Base

HTML5 no es solo etiquetas, es un lenguaje que proporciona [significado y estructura](#) al contenido web. Usar correctamente las etiquetas semánticas mejora la accesibilidad, el SEO y la mantenibilidad.

## Accesibilidad Universal

El diseño web debe ser [inclusivo para todos](#). La estructura semántica correcta es el primer paso para crear contenido accesible que pueda ser interpretado por diferentes tecnologías y usuarios.

## Fundamento de Desarrollo Web

Dominar HTML es esencial antes de avanzar a CSS y JavaScript. Es el [esqueleto sobre el que se construye](#) toda la experiencia web, independientemente de los frameworks o tecnologías que se utilicen.

## Preguntas de Análisis

- ¿Por qué es importante el uso semántico en HTML más allá del aspecto visual?
- ¿Cómo contribuye una estructura HTML bien formada a la accesibilidad web?
- ¿Qué desafíos presenta la web actual para mantener un código HTML limpio y semántico?

El HTML es el lenguaje que da [forma y significado](#) a la web. Dominarlo es el primer paso para convertirse en un desarrollador web completo.

# #EDCOUNIANDES

<https://educacioncontinua.uniandes.edu.co/>

Contacto: [educacion.continua@uniandes.edu.co](mailto:educacion.continua@uniandes.edu.co)

© - Derechos Reservados: La presente obra, y en general todos sus contenidos, se encuentran protegidos por las normas internacionales y nacionales vigentes sobre propiedad Intelectual, por lo tanto su utilización parcial o total, reproducción, comunicación pública, transformación, distribución, alquiler, préstamo público e importación, total o parcial, en todo o en parte, en formato impreso o digital y en cualquier formato conocido o por conocer, se encuentran prohibidos, y solo serán lícitos en la medida en que se cuente con la autorización previa y expresa por escrito de la Universidad de los Andes.