



HTML



HTML5 y CSS3 La Plantilla

Centro público integrado de
formación profesional
Alan Turing






CSS





Índice

¿Qué veremos en esta sección?

-  Paso 1: Estructura básica del documento HTML5
HTML5 introduce semántica
-  Paso 2: La etiqueta `<head>`
-  Paso 3: Declaración `<!DOCTYPE>` y apertura del `<html>`
-  Paso 4: El archivo `styles.css`
-  Paso 5: Contenido principal, tarjetas y pie de página



Paso 1: Estructura básica del documento HTML5

Paso 1: Estructura básica del documento HTML5



Paso 1: Estructura básica del documento HTML5

Etiquetas semánticas:

```
<header>...</header>
```

```
<nav>...</nav>
```

```
<main>...</main>
```

```
<footer>...</footer>
```



Paso 1: Estructura básica del documento HTML5

HTML5 introduce semántica

HTML5 introduce semántica para mejorar la accesibilidad, el SEO y la comprensión del código.

Etiquetas como `<header>`, `<nav>`, `<main>`, `<section>`, `<footer>` indican el propósito del contenido al navegador y a los lectores de pantalla.

Es más fácil de leer y mantener, especialmente para quienes están empezando.y CSS, dado que estas materias o se han visto o se verán en otros módulos.

A continuación, dos tutoriales

[HTML Tutorial](#). HTML es fácil de aprender: ¡lo disfrutarás!

[CSS Tutorial](#). Este tutorial le enseñará CSS desde lo básico hasta lo avanzado.



Paso 2: etiqueta `<head>`

Paso 2: etiqueta `<head>`



Paso 2: etiqueta <head>

```
<head>

  <!-- Codificación de caracteres -->
  <meta charset="UTF-8">

  <!-- Escala correctamente en dispositivos móviles -->
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- Título de la pestaña del navegador -->
  <title>Fundamentos de JavaScript</title>

  <!-- Icono que aparece en la pestaña del navegador -->
  <link rel="icon" type="image/x-icon" href="./img/favicon.ico">

  <!-- Hoja de estilos principal -->
  <link rel="stylesheet" href="./css/styles.css">

  <!-- Descripción breve del contenido de la página -->
  <meta name="description" content="La Plantilla en HTML5">

</head>
```



Paso 2: etiqueta `<head>`

Refuerza la idea de que el HTML5 es más que estructura visual, también tiene metainformación.

Introduce la idea de accesibilidad básica desde el principio (etiquetas meta).



Paso 3: Declaración `<!DOCTYPE>` y apertura del `<html>`

Paso 3: Declaración `<!DOCTYPE>` y apertura del `<html>`



Paso 3: Declaración `<!DOCTYPE>` y apertura del `<html>`

El `<!DOCTYPE html>` no es opcional: permite que el navegador interprete el HTML en modo estándar, no en modo quirúrgico o antiguo (quirks mode).

`lang="es"` ayuda a los lectores de pantalla y motores de búsqueda a entender el idioma del contenido, mejorando accesibilidad y SEO.

Ya tenemos cubierta la parte estructural de HTML hasta `<body>`.

Ahora pasamos a la parte que pediste: los estilos.



Paso 3: Declaración `<!DOCTYPE>` y apertura del `<html>`

```
<!DOCTYPE html> <!-- Indica que se trata de un documento HTML5 -->
```

```
<!-- Raíz del documento con el idioma especificado -->
```

```
<html lang="es">
```



Paso 4: Mejora del archivo `styles.css`

Paso 4: El archivo `styles.css`



Paso 4: El archivo `styles.css`

Selectores para que funcionen con etiquetas semánticas (header, nav, etc.) en lugar de clases (`.header`, `.topnav...`).

```
<header>
```

en vez de

```
<div class="header">
```

Ordenar el CSS con comentarios y bloques lógicos (cuerpo, estructura, componentes).

Revisar buenas prácticas: opacidad, colores, y accesibilidad.

Eliminar posibles redundancias.



Paso 4: El archivo styles.css

```
/* -----  
    Estilo general del documento  
----- */  
body {  
    background: url(../img/bgblanco.png) repeat;  
    text-align: justify;  
    font-family: Tahoma, Geneva, sans-serif;  
    font-size: 14px;  
    margin: 0; /* Eliminar márgenes por defecto */  
    padding: 0;  
    box-sizing: border-box;  
}
```



Paso 4: El archivo `styles.css`

Explicación:

- `background`: fondo visual básico.
- `text-align: justify`: fuerza la alineación del texto, aunque se puede debatir su legibilidad.
- `font-family`: uso de fuentes web seguras.
- `margin: 0;` y `padding: 0;` eliminan espacios por defecto.
- `box-sizing: border-box;` facilita el control de anchos y márgenes.



Paso 4: El archivo styles.css

```
/* -----  
    Cabecera principal  
----- */  
header {  
    background: url(../img/bgblue2.png) repeat;  
    opacity: 0.5;  
    text-align: center;  
    padding: 5px;  
    color: rgba(255, 250, 250, 0.9);  
    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.75);  
}
```



Paso 4: El archivo `styles.css`

Explicación:

- Ahora se usa `header` en lugar de `.header`.
- `rgba()` permite controlar la transparencia del color (color + opacidad).
- `text-shadow` da efecto visual, pero hay que hablar de legibilidad.



Paso 4: El archivo styles.css

Menú de navegación

```
/* -----  
   Menú de navegación  
----- */  
nav {  
  overflow: hidden;  
  background-color: rgba(210, 105, 30,  
0.5); /* chocolate semitransparente */  
  opacity: 0.85;  
}
```

```
nav a {  
  float: left;  
  display: block;  
  color: rgba(255, 250, 250, 0.9);  
  text-shadow: 2px 2px 4px rgba(0, 0, 0,  
0.75);  
  text-align: center;  
  padding: 14px 16px;  
  text-decoration: none;  
}  
  
nav a:hover {  
  background-color: rgba(128, 128, 128,  
0.25);  
  color: rgba(255, 215, 0, 0.75);  
}
```




Paso 4: El archivo `styles.css`

Explicación:

- `float: left;` se puede mejorar con Flexbox más adelante.
- `a:hover`: buen punto para enseñar pseudoclases.
- Puede debatirse el uso de opacidad y contraste en accesibilidad.



Paso 5: Contenido principal, tarjetas y pie de página

Paso 5: Contenido principal, tarjetas y pie de página



Paso 5: Contenido principal, tarjetas y pie de página

```
/* -----  
    Contenedor principal  
----- */  
main {  
  display: flex;  
  justify-content: center;  
  flex-wrap: wrap;  
  gap: 16px;  
  padding: 20px;  
}
```



Paso 5: Contenido principal, tarjetas y pie de página

Explicación:

- Flexbox facilita la maquetación responsive.
- gap controla la separación entre elementos sin márgenes manuales.
- padding mejora la separación del contenido respecto al borde.



Paso 5: Contenido principal, tarjetas y pie de página

```
/* -----  
   Tarjeta de ejercicio  
----- */  
.card {  
  display: flex;  
  flex-direction: column;  
  justify-content: space-between;  
  width: auto;  
  max-width: 300px;  
  border: 1px solid rgba(211, 211, 211, 0.5);  
  box-shadow: 2px 2px 8px 4px #d3d3d3d1;  
  border-radius: 15px;  
  font-family: sans-serif;  
  background-color: white;  
}
```




Paso 5: Contenido principal, tarjetas y pie de página

Explicación:

- max-width añade control y evita tarjetas excesivamente grandes.
- box-shadow introduce efectos visuales avanzados.
- border-radius suaviza los bordes → se puede explicar que da aspecto “moderno”.



Paso 5: Contenido principal, tarjetas y pie de página

```
.cardTitle {  
  font-size: 24px;  
  padding: 10px 10px 0 10px;  
}
```

```
.cardBody {  
  padding: 10px;  
  flex: 1 1 100%;  
}
```

```
.cardFooter {  
  background: rgba(50, 205, 50, 0.25); /*  
verde claro semitransparente */  
  border-radius: 0 0 15px 15px;  
  padding: 10px;  
  text-align: center;  
}
```

```
.cardFooter a {  
  text-decoration: none;  
  color: rgba(0, 0, 255, 0.8);  
  text-shadow: 2px 2px 4px rgba(0, 0, 0,  
0.25);  
}
```



Paso 5: Contenido principal, tarjetas y pie de página

Explicación:

- Refuerza separación de responsabilidades: título, cuerpo, pie.
- Ideal para introducir modularidad visual y CSS orientado a componentes.



Paso 5: Contenido principal, tarjetas y pie de página

```
/* -----  
    Pie de página  
----- */  
footer {  
    background: url(.../img/bgblue2.png) repeat;  
    opacity: 0.5;  
    text-align: center;  
    padding: 20px;  
    color: rgba(255, 250, 250, 0.9);  
    text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.75);  
}
```



Paso 5: Contenido principal, tarjetas y pie de página

Explicación:

- Como en header, se usa footer por claridad semántica.
- Buen lugar para introducir nociones de copyright o metadatos de autor.



Paso 5: Contenido principal, tarjetas y pie de página

```
/* -----  
    Estilo responsive (pantallas pequeñas)  
----- */  
@media screen and (max-width: 600px) {  
    main {  
        width: 100%;  
        flex-direction: column;  
        align-items: center;  
    }  
}
```



Paso 5: Contenido principal, tarjetas y pie de página

Explicación:

- Primer contacto con media queries.
- Les muestra cómo hacer que su sitio se vea bien en móviles.
- Buen momento para introducir el concepto de diseño responsive.



Ejercicios

JavaScript – ES6 – Plantillas



Ejercicios



Ejercicios – ¿Cómo vincular con GitHub?

Pasos para clonar un repositorio y empezar a trabajar:

1. Clona el repositorio de GitHub en tu máquina local:

- Usa el siguiente comando para clonar el repositorio de GitHub:
- `git clone https://github.com/jgarmay674/Plantilla03.git`
- Esto descargará todo el contenido del repositorio remoto a tu máquina y creará un directorio llamado Plantilla03 con los archivos que ya estén en el repositorio.

2. Accede a la carpeta del proyecto clonado:

- `cd Plantilla03`

3. Agrega los archivos de tu proyecto:

- Si ya tienes archivos en una carpeta aparte (por ejemplo, los de la **Plantilla03**), puedes copiarlos o moverlos a la carpeta del repositorio que acabas de clonar.



Ejercicios – ¿Cómo vincular con GitHub?

Pasos para clonar un repositorio y empezar a trabajar:

4. Haz un commit de los cambios:

- Una vez que hayas agregado o modificado archivos, puedes hacer un commit de los cambios:
- `git add . git commit -m "Añadiendo la Plantilla03"`

5. Sube los cambios al repositorio remoto (GitHub):

- Finalmente, haz push para subir los cambios al repositorio remoto en GitHub:
- `git push origin main`

Beneficios de esta opción:

- Comienzas con un repositorio sincronizado desde el principio.
- Evitas conflictos y problemas de divergencia.
- Facilita el control de versiones y el manejo de cambios a largo plazo.