

G-174. Programación de Servidores Web

Tema 3: Programación del lado del cliente

Por Andres Gorostidi - Andres.Gorostidi@cunef.edu

Programación del lado del cliente

Programación del lado del cliente se refiere al desarrollo de código que se ejecuta directamente en el navegador del usuario, en lugar de en el servidor.

Implica crear la interfaz de usuario y gestionar la interacción del usuario con la aplicación web.

01

Fundamentos de HTML y CSS

Fundamentos de HTML y CSS: HTML

HyperText Markup Language es el lenguaje de marcado de hipertexto. Define la estructura, semántica y contenido de las páginas Web. El navegador (cliente) interpreta el HTML y lo representan en pantalla. Permite agregar texto, imágenes, enlaces, tablas, listas, formularios, etc.

Su última versión del estándar: HTML 5.

Hipertexto: Sistema que permite enlazar fragmentos de textos entre sí. Permite que la lectura no sea lineal, sino que el usuario acceda a la información a través de los ítems relacionados.

Fundamentos de HTML y CSS: HTML

¿Por qué HTML?

HTML es un lenguaje de etiquetas. Estas etiquetas (tag) HTML comunican al navegador cual es la información a mostrar por pantalla, además del formato de dicha información.

- **Etiquetas** : Definen que función cumple cada elemento dentro de la página.

Fundamentos de HTML y CSS: HTML

Un documento HTML comienza con la declaración `<!doctype HTML>`, seguido por las etiquetas `<html>`, en su interior estarán las secciones `<head>` y `<body>`. La sección `<head>` contiene metadatos mientras que la sección `<body>` tiene el contenido real que se muestra al usuario.

Cada sección tiene un propósito específico, facilitando la **organización** del contenido.



Fundamentos de HTML y CSS: HTML

Etiquetas

Estructura básica, tienen una apertura y un cierre que describen la información contenida entre ellas, aunque algunos casos solamente tienen apertura, como la etiqueta `
` (line break o salto de línea).

```
<b>Texto de ejemplo</b>
```

 Texto de ejemplo

```
<p>Hola, estamos en el párrafo 1</p>  
<p>Ahora hemos cambiado de párrafo</p>
```

Hola, estamos en el párrafo 1

Ahora hemos cambiado de párrafo

```
<p>Este es un párrafo <br> que continúa  
en este renglón</p>
```

Este es un párrafo
que continúa en este renglón

Fundamentos de HTML y CSS: HTML

Etiquetas

< etiqueta



Html, body,
img, title,
head,
...

atributo =



Charset, alt,
src, id, class,
href, target,
...

"valor" >



Utf-8,
nombre de
clases, id, url,
...

Contenido

</etiqueta>

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


Fundamentos de HTML y CSS: HTML

Etiquetas básicas

`<h1>` , `<h2>`, `<h3>`... `<h6>`: Encabezados, numerados del 1 al 6 por orden de relevancia. Es importante respetar ese orden para que el navegador entienda la estructura de la página.

```
<h1>Encabezado en H1</h1>
<h2>Encabezado en H2</h2>
<h3>Encabezado en H3</h3>
<h4>Encabezado en H4</h4>
<h5>Encabezado en H5</h5>
<h6>Encabezado en H6</h6>
```

Encabezado en H1

Encabezado en H2

Encabezado en H3

Encabezado en H4

Encabezado en H5

Encabezado en H6

Tips: En VSC si escribimos h1 y presionamos TAB la etiqueta de cierre se escriben automáticamente.

Fundamentos de HTML y CSS: HTML

Etiquetas básicas

- `<div>`: Representa una división.
- `<p>`: Representa un párrafo.
- `<spam>`: Agrupar elementos en línea.
- `<u>`: Representa un subrayado.
- `<sub>`: Representa un subíndice.
- `
`: Representa el salto de línea.

```
<p>Lorem ipsum dolor sit amet consectetur adipisicing elit.  
Facilis labore eius nesciunt, voluptate modi quis tempora.  
Odit nisi voluptatem praesentium unde commodi optio,  
voluptate minima minus porro provident veritatis tempore  
voluptatum harum vero.</p>  
<p>Lorem ipsum dolor sit amet consectetur adipisicing  
elit.</p>
```

Fundamentos de HTML y CSS: HTML

Atributos

Son valores adiciones que agregan a una etiqueta para configurarla o definir su comportamiento.

Se añaden a la etiqueta de apertura

Ejemplo con una imagen.

```

```

```

```

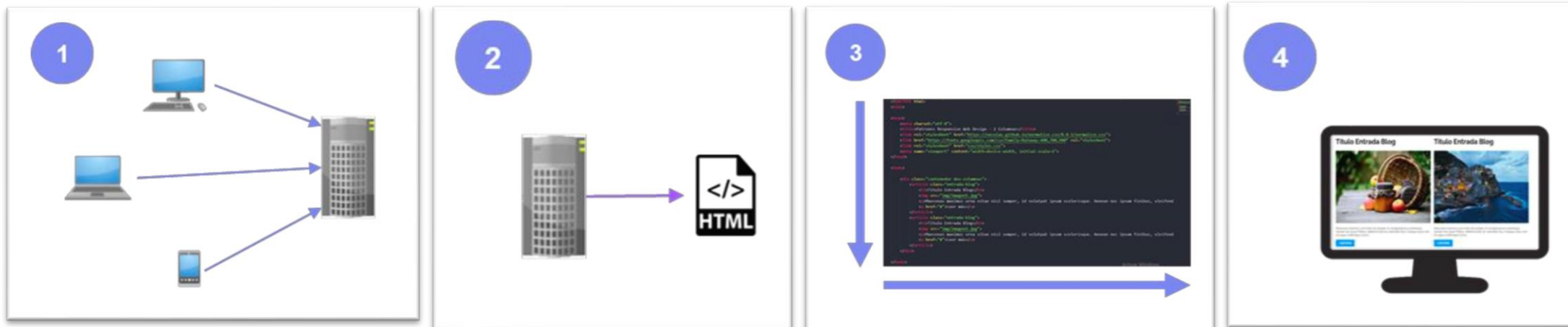
Fundamentos de HTML y CSS: HTML

¿Cómo funciona HTML?

1. El navegador (cliente) le pide información al servidor.
2. El servidor devuelve la información al cliente en un archivo HTML.
3. El navegador en el cliente lee el archivo de arriba hacia abajo y de izquierda a derecha para interpretar la información.
4. Tiene en cuenta las etiquetas que tiene el documento y las va renderizando en la pantalla (lo que se ve en el navegador).

Fundamentos de HTML y CSS: HTML

¿Cómo funciona HTML?



Fundamentos de HTML y CSS: HTML

Estructura de una página web

Estas etiquetas ayudan a definir la clase de contenido tendrá una página Web. Describen su significado tanto para el navegador como para el desarrollador. A través de ellas los navegadores y buscadores reconocen patrones y una estructura determinada. Debemos respetarlas porque ayudan al navegador a entender su significado para mostrarlo en pantalla y ayudan a los buscadores a reconocer el contenido y la estructura del sitio.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>

</body>
</html>
```

Fundamentos de HTML y CSS: HTML

Estructura de una página web

- `<!DOCTYPE html>` Indica que la versión corresponde a HTML5.
- `<html lang= "es">` Es la etiqueta principal que engloba al resto de las etiquetas, el atributo lang define el tipo de lenguaje.
- `<head>` Es la cabeza del documento que contiene los metadatos de la página web.
- `<meta charset="utf-8">` Indica al navegador qué tipo de caracteres contiene la página, con el atributo charset vamos a indicar el conjunto de caracteres que vamos a usar y con el valor "utf-8" abarcamos a la mayoría de los sistemas de escritura.
- `<title>` Indica el título de la página Web, que se visualiza en la barra de título del navegador.
- `<body>` Es el cuerpo del documento donde va a estar todo el contenido que vamos a mostrar.

ETIQUETAS HTML COMUNES: HEAD

- `<!DOCTYPE html>` : Declara el tipo de documento y la versión de HTML utilizada. En este caso, se está utilizando HTML5.
- `<html lang="es">` : Define el inicio del documento HTML y el idioma del contenido (en este caso, español).
- `<head>` : Contiene metadatos y enlaces relacionados con el documento HTML.
- `<meta charset="UTF-8">` : Especifica la codificación de caracteres utilizada en el documento. En este caso UTF-8.
- `<meta name="viewport" content="width=device-width, initial-scale=1.0">` : Configura la vista del sitio web en dispositivos móviles, asegurando que el contenido se ajuste al ancho de la pantalla y que el zoom inicial sea 1.

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Página de prueba</title>
    <link rel="stylesheet" href="styles.css" />
    <script src="script.js" defer></script>
    <link rel="icon" href="favicon.ico" type="image/x-icon" />
  </head>
  <body>
    <!-- Contenido de la página -->
  </body>
</html>
```


ETIQUETAS HTML COMUNES: HEAD

- `<title>Página de prueba</title>` : Define el título del documento que aparece en la pestaña del navegador.
- `<link rel="stylesheet" href="styles.css">` : Vincula una hoja de estilo externa al documento para aplicar estilos CSS. El atributo href especifica la ubicación del archivo CSS.
- `<script src="script.js" defer></script>` : Vincula un archivo JavaScript externo al documento. El atributo defer indica que el script debe ser ejecutado después de que el documento haya cargado por completo.
- `<link rel="icon" href="favicon.ico" type="image/x-icon">` : Define el icono de la pestaña del navegador (favicon). El atributo href especifica la ubicación del archivo del icono.

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Página de prueba</title>
    <link rel="stylesheet" href="styles.css" />
    <script src="script.js" defer></script>
    <link rel="icon" href="favicon.ico" type="image/x-icon" />
  </head>
  <body>
    <!-- Contenido de la página -->
  </body>
</html>
```

ETIQUETAS HTML COMUNES: TEXTO

- `<h1>`, `<h2>`, ..., `<h6>` : Etiquetas de encabezado que definen títulos y subtítulos. `<h1>` es el más importante, mientras que `<h6>` es el menos importante.
- `<p>` : Define un párrafo de texto.
- `` : Resalta el texto en negrita, indicando importancia.
- `` : Resalta el texto en cursiva, indicando énfasis.
- `
` : Inserta un salto de línea.
- `<hr>` : Inserta una línea horizontal para dividir contenido.

```
<h1>Título principal</h1>
<h2>Subtítulo 1</h2>
<h3>Subtítulo 2</h3>
<h4>Subtítulo 3</h4>
<h5>Subtítulo 4</h5>
<h6>Subtítulo 5</h6>
<p>
  Lorem ipsum dolor sit amet, consectetur adipisicing elit. Possimus itaque
  ullam delectus, nisi illum perspiciatis fugit, natus amet exercitationem
  aperiam reprehenderit neque optio, inventore voluptatem quos non rerum. Alias,
  at.
</p>
<strong>lorem</strong>
<em>texto en cursiva</em>
<p>Primera línea</p>
<br />
<p>Segunda línea</p>
<hr />
```

ETIQUETAS HTML COMUNES: ENLACES Y LISTAS



- `<a>` : Define un hipervínculo.
- `` : Define una lista desordenada (con viñetas).
- `` : Define una lista ordenada (numerada).
- `` : Define un ítem en la lista.

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

<ul>
  <li>Item 1</li>
  <li>Item 2</li>
</ul>

<ol>
  <li>Item 1</li>
  <li>Item 2</li>
</ol>
```

ETIQUETAS HTML COMUNES: TABLA

- `<table>` : Define una tabla.
- `<tr>` : Define una fila en la tabla.
- `<th>` : Define una celda de encabezado en la tabla.
- `<td>` : Define una celda de datos en la tabla.

```
<table>
  <tr>
    <th>ID</th>
    <th>Nombre</th>
  </tr>
  <tr>
    <td>1</td>
    <td>Daniel</td>
  </tr>
</table>
```

ETIQUETAS HTML COMUNES: CONTENEDORES

- `<div>` : Es un contenedor de bloque utilizado para agrupar secciones de elementos html.
- `` : Es un contenedor en línea utilizado para aplicar estilos o comportamientos a un fragmento de texto.



```
<div>
  <h2>Section 1</h2>
  <p>lorem ipsum</p>
</div>

<p>Este es un párrafo con
<span>texto</span> dentro de él.</p>
```

Fundamentos de HTML y CSS: CSS

Es un lenguaje de diseño que nos permite darle estilos a los componentes de un documento en función de una jerarquía. Se ocupa de la estética, el aspecto.

CSS es una sigla que proviene de Cascading StyleSheets (Hojas de Estilo en Cascada, en español). La palabra cascada hace referencia a una propiedad muy importante de CSS, y es la forma en que se comporta cuando entran en conflicto dos o más reglas de estilo.

Cuando diseñamos un sitio web profesional, con un equipo de trabajo, mantener los estilos separados de la estructura y contenido (HTML) facilita la división de tareas entre los desarrolladores.

Fundamentos de HTML y CSS: CSS

¿Cómo incorporamos CSS?

- **CSS en Línea:** Dentro del atributo style="" incorporamos los estilos que se van a aplicar solo en esa misma etiqueta. Opción no recomendable.

```
<p style="color: white; background-color: violet">Esto es un párrafo</p>
```

Esto es un párrafo

- Utilizando el atributo style dentro de la etiqueta le proporcionamos estilo al párrafo. Se pueden utilizar a la vez varias parejas de: propiedad: valor.

Fundamentos de HTML y CSS: CSS

¿Cómo incorporamos CSS?

- **CSS interno:** Incluimos la etiqueta `<style>` dentro del `<head>` en nuestro documento. Opción menos recomendable:

```
<!-- Estilo interno -->
<style>
  h1{
    color: white;
    background-color: red;
  }
</style>
```

Título H1

En el ejemplo anterior todas las etiquetas `<h1>` tendrán color de fuente blanco y fondo de color rojo.

Fundamentos de HTML y CSS: CSS

¿Cómo incorporamos CSS?

- **CSS Externo:** En el head del documento HTML tenemos que incluir una referencia al archivo .css dentro del elemento <link>.

Es la forma más recomendada.

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Ejemplo con CSS Externo</title>
  <!-- Aquí enlazamos el archivo CSS externo -->
  <link rel="stylesheet" href="styles.css">
</head>
```

- La referencia al archivo externo debe incluir la ruta completa, el nombre del archivo y su extensión si se encuentra en alguna subcarpeta dentro del proyecto.
- En caso de que el archivo de estilos se encuentre en la misma carpeta que el documento HTML, únicamente se debe incluir el nombre del archivo y su extensión. Recordemos que es aconsejable mantener estos archivos (CSS y HTML) en carpetas separadas.

Fundamentos de HTML y CSS: CSS

Reglas de CSS

➤ **Propiedades:** Definen qué aspecto o característica del elemento se va a modificar.

Ej: color, background-color, font-size, margin, padding.

➤ **Valores:** Asignan un valor a la propiedad.

Ej: color: red;, font-size: 14px;, margin: 10px;

Cascada: Si varios estilos aplican a un elemento, se siguen las reglas de la cascada (último definido es el que prevalece).

Especificidad: Determina qué estilo se aplica cuando hay conflicto. Las reglas más específicas tienen mayor prioridad.

Ej: #mi-id (ID) > .mi-clase (Clase) > p (Elemento).

Herencia: Algunas propiedades se heredan automáticamente de un elemento padre a sus hijos.

Ej: font-family, color se heredan; margin, border no se heredan.

Fundamentos de HTML y CSS: CSS

Reglas de CSS: Unidades de medida

- **Unidades absolutas:** Medidas fijas. Ej: px (píxeles), cm (centímetros).
- **Unidades relativas:** Medidas relativas a otro valor. Ej: em, rem, %.

em: Relativo al tamaño de fuente del elemento padre.

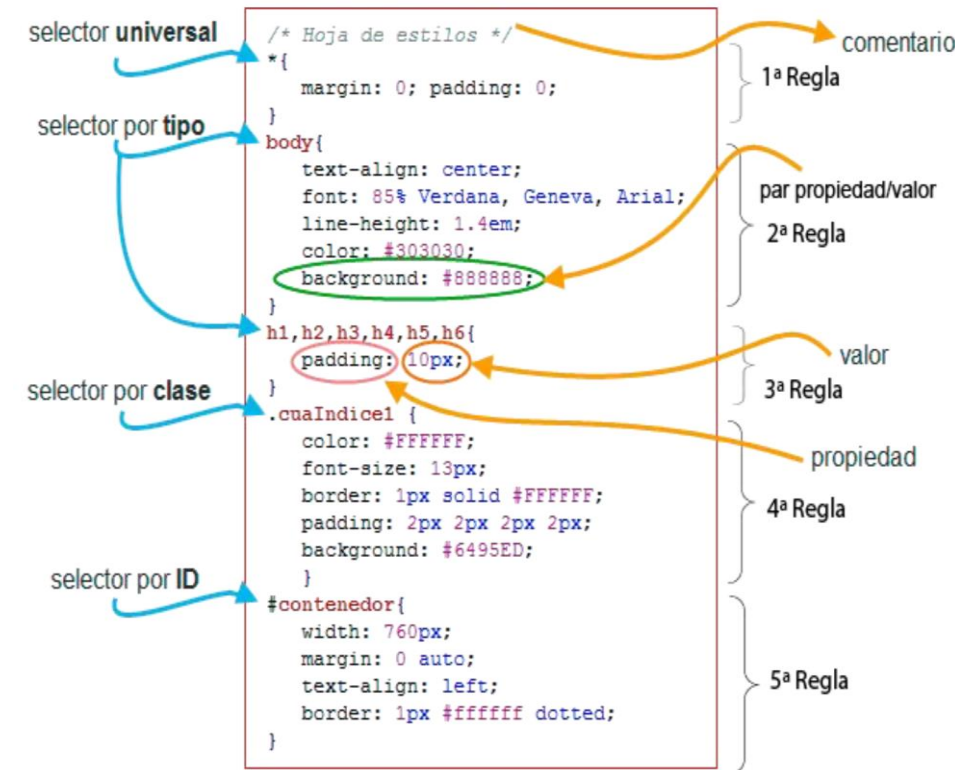
rem: Relativo al tamaño de fuente raíz (html).

Fundamentos de HTML y CSS: CSS

Selectores

Indican el elemento al que se debe aplicar el estilo. La declaración indica "qué hay que hacer" y el selector indica "a quién hay que aplicarlo". Hay cuatro selectores básicos:

- **selector universal:** Selecciona todos los elementos de HTML.
- **selector de etiqueta o tipo:** Se utiliza para seleccionar una etiqueta específica.
- **selector de clase:** Se utiliza agregando el atributo class a los elementos que queremos aplicarles estilos.
- **selector de identificador (id):** Similar a .class pero solo se aplica a una etiqueta individual.



PROPIEDADES BÁSICAS: COLORES

- **Color:** Se utiliza la propiedad color para establecer el color del texto.
- **Fondos:** La propiedad background permite definir el color o la imagen de fondo de un elemento.

Se pueden usar valores como nombres de colores, hexadecimales, RGB, RGBA, HSL, y HSLA.

```
/* Color del texto */  
p {  
    color: blue; /* Color del texto en azul */  
}  
  
/* Fondo del elemento */  
div {  
    background-color: lightgray; /* Color de  
fondo gris claro */  
}
```

PROPIEDADES BÁSICAS: FUENTES Y TIPOGRAFÍA

- **Fuente:** La propiedad `font-family` especifica la fuente de un texto.
- **Tamaño de fuente:** La propiedad `font-size` define el tamaño del texto.
- **Estilo de fuente:** Las propiedades `font-style` y `font-weight` permiten definir el estilo (normal, italic) y el grosor (normal, bold) del texto.

```
/* Fuente y tamaño del texto */  
h1 {  
    font-family: 'Arial', sans-serif; /*  
Fuente Arial con alternativa sans-serif */  
    font-size: 24px; /* Tamaño de fuente de 24  
píxeles */  
    font-style: italic; /* Estilo de fuente en  
cursiva */  
    font-weight: bold; /* Peso de fuente en  
negrita */  
}
```

PROPIEDADES BÁSICAS: TAMAÑO Y ESPACIADO

- **Tamaño:** La propiedad width y height se utilizan para definir el tamaño de los elementos.
- **Espaciado:** Las propiedades margin y padding controlan el espacio alrededor y dentro de los elementos, respectivamente.

```
/* Tamaño del elemento */
div {
  width: 300px; /* Ancho de 300 píxeles */
  height: 200px; /* Alto de 200 píxeles */
}

/* Espaciado alrededor y dentro del elemento */
p {
  margin: 20px; /* Margen de 20 píxeles
alrededor del elemento */
  padding: 10px; /* Relleno de 10 píxeles
dentro del elemento */
}
```

MODELO DE CAJA (BOX MODEL)

El Modelo de Caja es una técnica de CSS que describe la forma en que los elementos son renderizados en la página.

Cada elemento se representa como una caja rectangular que consta de cuatro partes:

- Contenido
- Relleno
- Borde
- Margen

```
div {  
  width: 200px; /* Contenido */  
  height: 100px; /* Contenido */  
  padding: 20px; /* Relleno */  
  border: 2px solid black; /* Borde */  
  margin: 10px; /* Margen */  
}
```


CONTENIDO

El área de contenido es donde se muestra el texto o las imágenes dentro de un elemento. Se define con las propiedades width y height.

Es el área principal del elemento, sobre la que se aplican el relleno, el borde y el margen.

```
div {  
  width: 200px; /* Contenido */  
  height: 100px; /* Contenido */  
  padding: 20px; /* Relleno */  
  border: 2px solid black; /* Borde */  
  margin: 10px; /* Margen */  
}
```

RELLENO

El relleno es el espacio entre el contenido del elemento y su borde. Se utiliza la propiedad `padding` para definir el relleno, que puede ser aplicado de forma uniforme o individual a cada lado.

Aumenta el tamaño del área ocupada por el elemento, sin afectar el tamaño del borde o del margen.

```
div {  
  width: 200px; /* Contenido */  
  height: 100px; /* Contenido */  
  padding: 20px; /* Relleno */  
  border: 2px solid black; /* Borde */  
  margin: 10px; /* Margen */  
}
```

BORDE



El borde rodea el relleno y el contenido del elemento. Se define con la propiedad `border`, que incluye el ancho, el estilo y el color del borde.

Puede ser sólido, punteado, rayado, entre otros estilos.



```
div {  
  width: 200px; /* Contenido */  
  height: 100px; /* Contenido */  
  padding: 20px; /* Relleno */  
  border: 2px solid black; /* Borde */  
  margin: 10px; /* Margen */  
}
```

MARGEN

El margen es el espacio entre el borde del elemento y los elementos adyacentes. Se utiliza la propiedad `margin` para definir el margen, que puede ser aplicado de forma uniforme o individual a cada lado.

El margen no afecta el tamaño del elemento, pero crea espacio alrededor de él.



```
div {  
  width: 200px; /* Contenido */  
  height: 100px; /* Contenido */  
  padding: 20px; /* Relleno */  
  border: 2px solid black; /* Borde */  
  margin: 10px; /* Margen */  
}
```

DISEÑO Y POSICIONAMIENTO

El diseño y posicionamiento en CSS permite controlar cómo se disponen y alinean los elementos en una página web.

Incluye conceptos básicos de layout, diferentes tipos de posicionamiento, y técnicas avanzadas como Flexbox y Grid.

LAYOUT BÁSICO: BLOQUE E INLINE

- **Elementos de bloque:** Ocupan todo el ancho disponible y comienzan en una nueva línea. Ejemplos: div, p, h1.
- **Elementos inline:** Solo ocupan el espacio necesario para su contenido y no inician una nueva línea. Ejemplos: span, a, img.

```
div {  
  display: block;  
}  
  
span {  
  display: inline;  
}
```

POSICIONAMIENTO

- **Estático:** Valor por defecto. Los elementos se posicionan en el flujo normal del documento.
- **Relativo:** Los elementos se posicionan en relación con su posición original. Usa las propiedades top, right, bottom, left.
- **Absoluto:** Los elementos se posicionan en relación con el contenedor más cercano con posición distinta de static. Usa las propiedades top, right, bottom, left.
- **Fijo:** Los elementos se posicionan en relación con la ventana del navegador y permanecen en la misma posición al hacer scroll. Usa las propiedades top, right, bottom, left.

```
.static {  
  position: static;  
}  
  
.relative {  
  position: relative;  
  top: 10px;  
  left: 20px;  
}  
  
.absolute {  
  position: absolute;  
  top: 10px;  
  right: 20px;  
}  
  
.fixed {  
  position: fixed;  
  bottom: 10px;  
  left: 20px;  
}
```

- Flexbox es un modelo de diseño unidimensional que permite distribuir espacio y alinear elementos en una fila o columna.
- Utiliza las propiedades `display: flex`, `justify-content`, `align-items`, y `flex-direction` para controlar el diseño.

```
/* Contenedor Flexbox */  
.container {  
  display: flex; /* Establece un  
  contenedor flexbox */  
  justify-content: center; /* Alinea  
  elementos horizontalmente en el centro  
  */  
  align-items: center; /* Alinea  
  elementos verticalmente en el centro  
  */  
  flex-direction: row; /* Dispone los  
  elementos en una fila */  
}
```


GRID

- Grid es un modelo de diseño bidimensional que permite crear layouts complejos con filas y columnas.
- Utiliza las propiedades `display: grid`, `grid-template-columns`, `grid-template-rows`, `grid-area`, y `grid-column` para definir la estructura del layout.

```
/* Contenedor Grid */
.grid-container {
  display: grid; /* Establece un
contenedor grid */
  grid-template-columns: repeat(3, 1fr);
/* Crea 3 columnas de igual tamaño */
  grid-template-rows: repeat(
    2,
    100px
  ); /* Crea 2 filas de 100 píxeles de
alto */
  gap: 10px; /* Espacio entre las celdas
del grid */
}

/* Elemento Grid */
.grid-item {
  grid-column: span 2; /* Ocupa 2 columnas
del grid */
  grid-row: span 1; /* Ocupa 1 fila del
grid */
}
```

TRANSICIÓN Y ANIMACIONES



Las transiciones y animaciones permiten agregar efectos dinámicos a los elementos en una página web.

Las transiciones permiten cambiar de estado suavemente, mientras que las animaciones ofrecen un control más detallado sobre los cambios de estado.

PROPIEDADES DE TRANSICIÓN

Las transiciones permiten que las propiedades CSS cambien gradualmente de un valor a otro.

Propiedad transition: Especifica la duración, el tipo de transición, y el retraso.

- **transition-property:** Define qué propiedad se va a animar.
- **transition-duration:** Define la duración de la transición.
- **transition-timing-function:** Define el ritmo de la transición (por ejemplo, ease, linear).
- **transition-delay:** Define el retraso antes de que la transición comience.

```
/* Transición en color y tamaño */
.button {
  background-color: blue; /* Color inicial */
  width: 100px; /* Ancho inicial */
  height: 50px; /* Alto inicial */
  transition-property: background-color,
width; /* Propiedades a animar */
  transition-duration: 0.5s; /* Duración de
la transición */
  transition-timing-function: ease-in-out;
/* Ritmo de la transición */
  transition-delay: 0s; /* Retraso antes de
iniciar la transición */
}

/* Transición al pasar el ratón */
.button:hover {
  background-color: red; /* Nuevo color */
  width: 150px; /* Nuevo ancho */
}
```

ANIMACIONES BÁSICAS CON @KEYFRAMES

Las animaciones se definen con la regla `@keyframes`, que describe cómo cambian los estilos a lo largo del tiempo.

Propiedad `animation`: Controla la animación aplicada a un elemento.

- **`animation-name`:** Especifica el nombre de la animación definida con `@keyframes`.
- **`animation-duration`:** Define la duración de la animación.
- **`animation-timing-function`:** Define el ritmo de la animación (por ejemplo, `ease`, `linear`).
- **`animation-delay`:** Define el retraso antes de que comience la animación.
- **`animation-iteration-count`:** Define cuántas veces se repetirá la animación.

```
/* Definición de la animación */
@keyframes slideIn {
  from {
    transform: translateX(-100%); /*
    Comienza fuera de la pantalla */
  }
  to {
    transform: translateX(0); /* Termina en
    su posición original */
  }
}

/* Aplicación de la animación */
.slide {
  animation-name: slideIn; /* Nombre de la
  animación */
  animation-duration: 1s; /* Duración de la
  animación */
  animation-timing-function: ease-in-out; /*
  Ritmo de la animación */
  animation-delay: 0s; /* Retraso antes de
  iniciar la animación */
  animation-iteration-count: 1; /* Número de
  repeticiones */
}
```

EJEMPLO COMBINADO

```
/* Transición en un botón */
.button {
  background-color: blue; /* Color inicial */
  width: 100px; /* Ancho inicial */
  height: 50px; /* Alto inicial */
  transition: background-color 0.5s ease-in-out,
width 0.5s ease-in-out; /* Transiciones */
}
/* Animación en un cuadro */
.box {
  width: 100px;
  height: 100px;
  background-color: green;
  animation: expand 2s infinite; /* Aplicación de
la animación */
}
@keyframes expand {
  0% {
    width: 100px;
    height: 100px;
  }
  50% {
    width: 200px;
    height: 200px;
  }
  100% {
    width: 100px;
    height: 100px;
  }
}
```

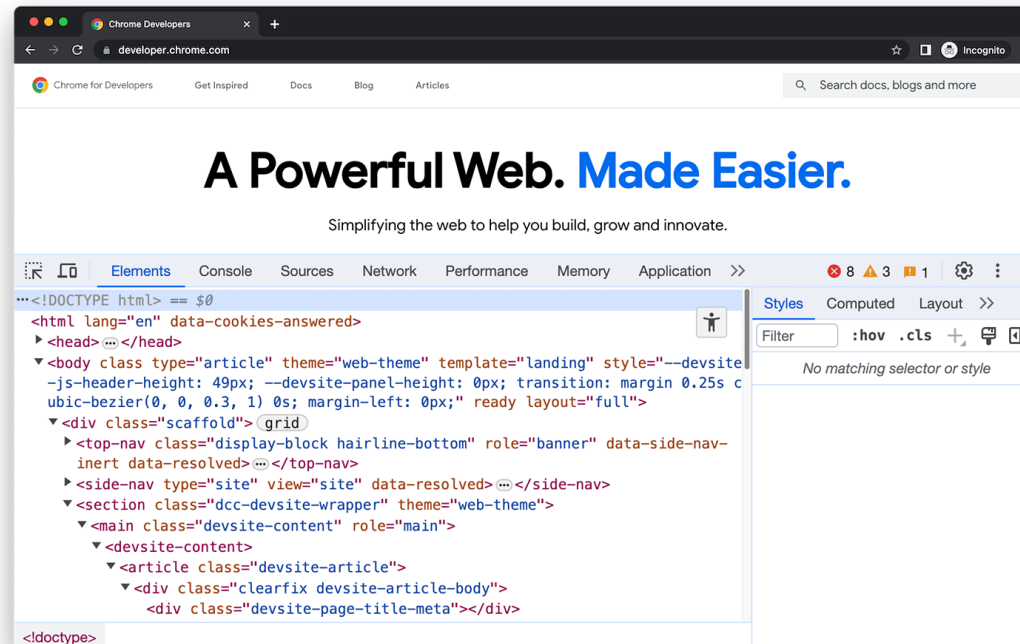
02

Uso de la consola del Developer en Chrome

Consola en Chrome

La consola del desarrollador es una herramienta poderosa en Chrome que permite inspeccionar y depurar código directamente en el navegador.

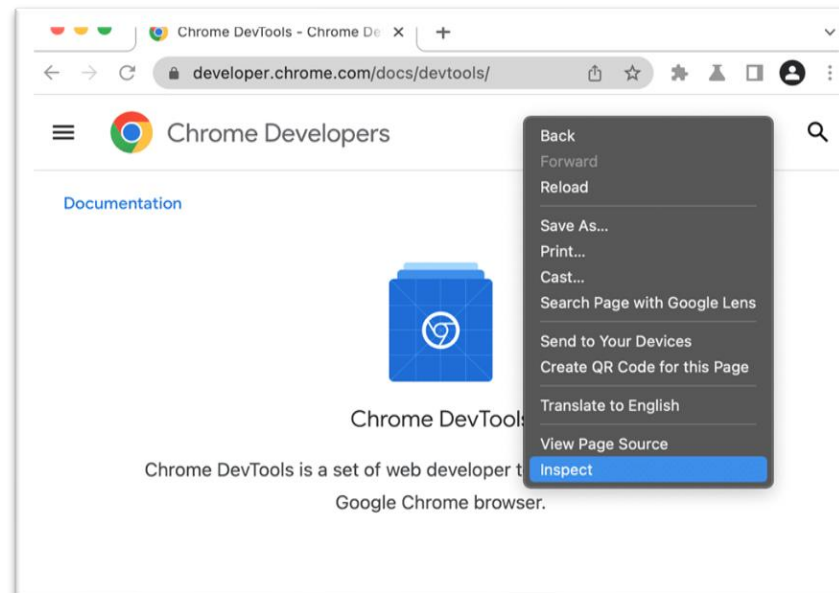
Acceso a elementos HTML, CSS y JavaScript en tiempo real.



Consola en Chrome

Abrir la consola en Chrome

Para abrir la consola de herramientas para desarrolladores en Chrome, puedes hacer clic derecho en la página y seleccionar "Inspeccionar". También puedes usar el atajo de teclado: en Mac, presiona Comando + Opción + C; en Windows o Linux, presiona Control + Shift + J.

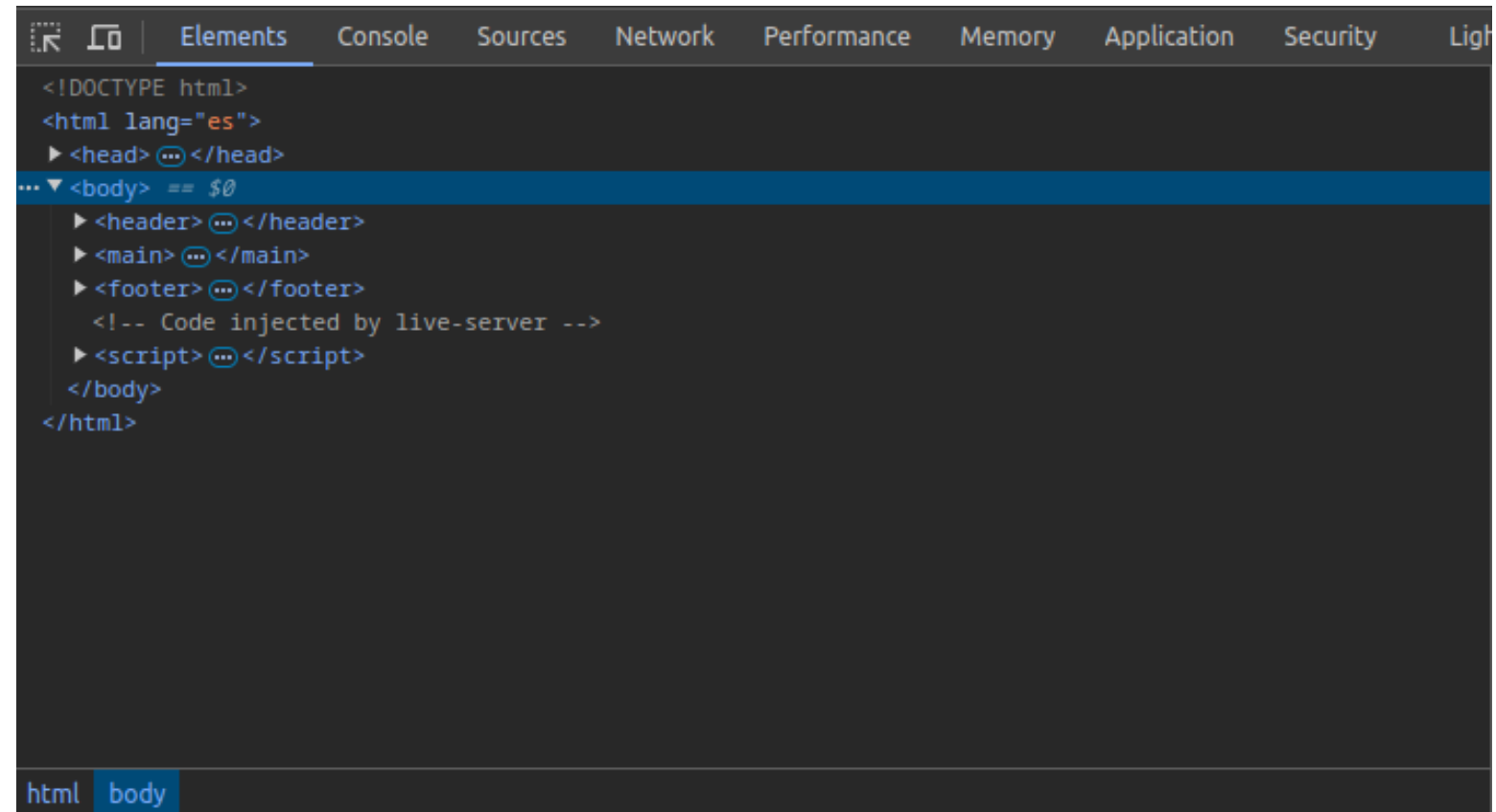


Consola en Chrome

Funciones claves de la consola

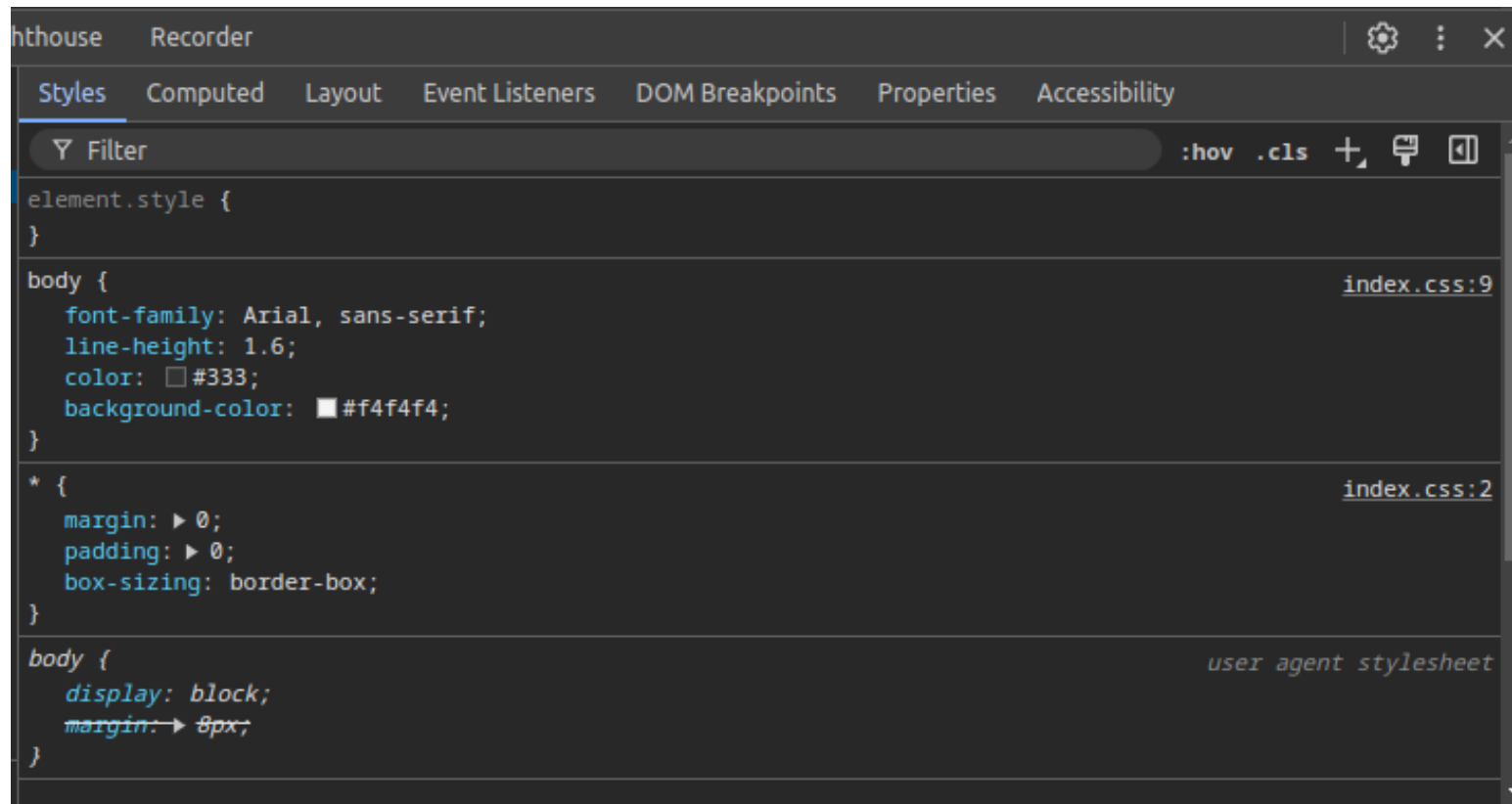
- **Ejecución de JavaScript:** Ejecutar y probar código en tiempo real.
- **Depuración:** Identificar y resolver errores en el JavaScript.
- **Inspección del DOM:** Modificar y analizar elementos HTML.
- **Monitoreo de red:** Analizar solicitudes y tiempos de carga.
- **Registro de información:** Usar `console.log()`, etc., para registrar datos.
- **Medición de rendimiento:** Evaluar el rendimiento de scripts y carga de páginas.
- **Almacenamiento:** Acceder y modificar `localStorage`, `sessionStorage` y cookies.
- **Depuración de CSS:** Inspeccionar y cambiar estilos CSS.
- **Simulación de dispositivos:** Probar vistas en diferentes dispositivos y resoluciones.
- **Acceso a APIs de Chrome:** Interactuar con APIs específicas del navegador.

USO DE LA CONSOLA DE DEVELOPER EN CHROME



```
<!DOCTYPE html>
<html lang="es">
  <head> </head>
  <body>
    <header> </header>
    <main> </main>
    <footer> </footer>
    <!-- Code injected by live-server -->
    <script> </script>
  </body>
</html>
```

USO DE LA CONSOLA DE DEVELOPER EN CHROME



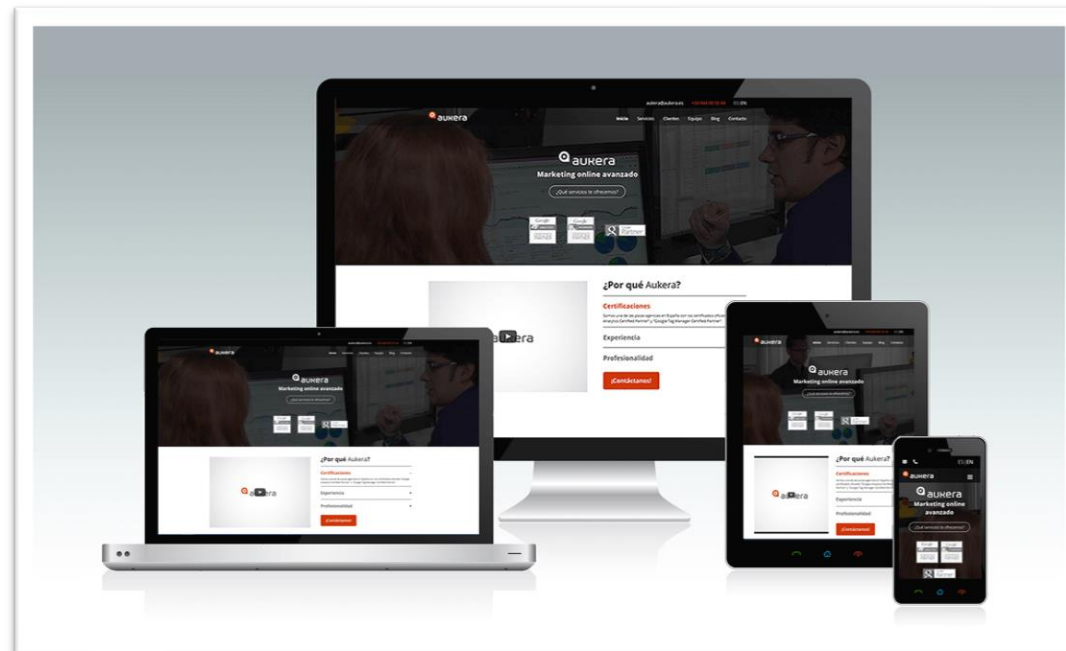
03

Principios de diseño responsivo

Diseño responsivo

El diseño responsivo asegura que tu sitio web se vea y funcione bien en cualquier dispositivo, desde móviles hasta monitores de escritorio.

Adapta el contenido y el diseño para mejorar la experiencia del usuario.



Diseño responsivo

Principios del diseño responsivo:

- **Flexibilidad:** El sitio web debe ser flexible y adaptarse a diferentes tamaños de pantalla.
- **Grid:** El sitio web debe utilizar un grid para organizar el contenido.
- **Imágenes:** Las imágenes deben ser escalables y adaptarse a diferentes tamaños de pantalla.
- **Media queries:** Las media queries nos permiten aplicar estilos diferentes según el tamaño de pantalla.
- **Mobile-first:** El sitio web debe ser diseñado para móviles primero y luego adaptarse a otros dispositivos.

Diseño responsivo



Principios del diseño responsivo: Flexibilidad

La flexibilidad es fundamental en el diseño responsivo. El sitio web debe ser capaz de adaptarse a diferentes tamaños de pantalla sin perder su estructura y diseño.

Utilizar porcentajes en lugar de píxeles para definir el ancho y alto de los elementos.

Utilizar unidades relativas (como em o rem) para definir el tamaño de la fuente.

Utilizar flexbox o grid para crear diseños flexibles.

Diseño responsivo

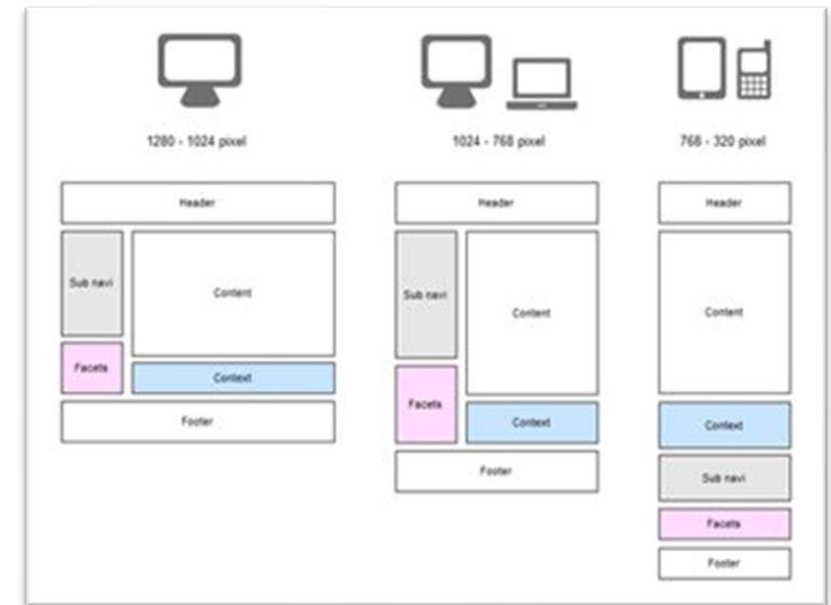
Principios del diseño responsivo: Grid

Un grid es una estructura de diseño que nos permite organizar el contenido de manera efectiva y consistente.

Utilizar un grid para organizar el contenido.

Definir el ancho y alto de las columnas y filas.

Utilizar margin y padding para crear espacios entre los elementos.



Diseño responsivo

Principios del diseño responsivo: Imágenes

Las imágenes deben ser escalables y adaptarse a diferentes tamaños de pantalla.

Utilizar imágenes vectoriales (como SVG) en lugar de imágenes raster (como JPEG o PNG).

Utilizar la propiedad max-width para definir el ancho máximo de las imágenes.

Utilizar la propiedad height para definir el alto de las imágenes.

Diseño responsivo

Principios del diseño responsivo: Media queries

Las media queries nos permiten aplicar estilos diferentes según el tamaño de pantalla.

Utilizar la propiedad @media para definir las media queries.

Definir las condiciones para aplicar los estilos (como el ancho o alto de la pantalla).

Utilizar las media queries para aplicar estilos diferentes según el dispositivo.

Diseño responsivo

Principios del diseño responsivo: Mobile-first

El sitio web debe ser diseñado para móviles primero y luego adaptarse a otros dispositivos

Mejora la experiencia del usuario: Mejora la experiencia del usuario en dispositivos móviles.

Incrementa la conversión: Incrementa la conversión y la retención de usuarios.

Mejora el rendimiento: Mejora el rendimiento y la velocidad de carga del sitio web.

UNIDADES RELATIVAS Y ABSOLUTAS

Unidades absolutas: Fijas y no cambian con el tamaño del viewport.

- **px (píxeles):** Unidades fijas que no dependen del tamaño del viewport.
- **pt (puntos):** Utilizado en impresión, 1pt = 1/72 de pulgada.

Unidades relativas: Cambian en función de otros elementos o el viewport.

- **em:** Relativo al tamaño de fuente del elemento padre.
- **rem:** Relativo al tamaño de fuente del elemento raíz (<html>).
- **%:** Relativo al tamaño del contenedor padre o al viewport.
- **vw/vh:** Relativo al viewport (viewport width/height).

```
/* Unidades absolutas */
.fixed-width {
  width: 300px; /* Ancho fijo en píxeles */
}

/* Unidades relativas */
.relative-width {
  width: 50%; /* Ancho relativo al contenedor padre */
}

.relative-font {
  font-size: 1.5em; /* Tamaño de fuente relativo al
elemento padre */
}

.relative-root {
  font-size: 1rem; /* Tamaño de fuente relativo al
elemento raíz */
}

.viewport-width {
  width: 50vw; /* 50% del ancho del viewport */
}
```

MEDIDAS DE DISEÑO RESPONSIVO

- **Diseño fluido:** Usa unidades relativas (% , em, rem) para que los elementos se ajusten al tamaño del contenedor.
- **Imágenes flexibles:** Utiliza max-width: 100%; para que las imágenes se adapten al contenedor y no desborden.
- **Layouts adaptativos:** Usa flexbox o grid para crear diseños que se ajusten a diferentes tamaños de pantalla.

```
/* Imágenes flexibles */
img {
  max-width: 100%; /* La imagen no excederá el tamaño del
  contenedor */
  height: auto; /* Mantiene la proporción de la imagen */
}

/* Layout adaptativo con Flexbox */
.container {
  display: flex;
  flex-wrap: wrap; /* Permite que los elementos se
  ajusten en múltiples líneas */
}

.item {
  flex: 1 1 100%; /* Los elementos ocuparán el 100% del
  ancho del contenedor en pantallas pequeñas */
}

/* Layout adaptativo con Grid */
.grid-container {
  display: grid;
  grid-template-columns: repeat(
    auto-fill,
    minmax(200px, 1fr)
  ); /* Ajusta automáticamente el número de columnas */
  gap: 10px; /* Espacio entre las celdas */
}
```

MEDIA QUERIES

```
/* Estilos para pantallas pequeñas */
@media (max-width: 600px) {
  .responsive-text {
    font-size: 14px; /* Tamaño de fuente más pequeño en
pantallas pequeñas */
  }

  .responsive-layout {
    display: block; /* Cambia el diseño a bloque en
pantallas pequeñas */
  }
}

/* Estilos para pantallas grandes */
@media (min-width: 768px) {
  .responsive-text {
    font-size: 18px; /* Tamaño de fuente más grande en
pantallas grandes */
  }

  .responsive-layout {
    display: flex; /* Usa flexbox en pantallas grandes */
    flex-direction: row; /* Elementos en fila */
  }
}
```



GRACIAS POR VUESTRA ATENCIÓN