

# Practica 3 Layout CSS y Responsive Design

## Código Html

[Header de las páginas](#)

[Footer de las páginas](#)

[Main index.html](#)

[Main aplicaciones.html](#)

[Main en-el-cine.html](#)

[Main despacho-42.html](#)

## Código CSS

[Detalles generales](#)

[Header de las páginas](#)

[Footer de las páginas](#)

[Resto del código CSS](#)

[Responsive Design mediante @media](#)

[Main index.html](#)

## Validación

## GitHub

[Acceso al repositorio](#)

[Acceso a GitHub Pages](#)

## Código Html

### Header de las páginas

```
<header class="menu-navegacion">
  <div class="container-big">
    <p>
      <a href="index.html"> <strong>IA</strong> al día<
    </p>
    <nav>
      <ul>
        <li><a href="aplicaciones.html">APLICACIONES<
        <li><a href="despacho-42.html">NOTICIAS</a></li>
      </ul>
    </nav>
  </div>
</header>
```

```

        <li><a href="en-el-cine.html">EN EL CINE</a></li>
    </ul>
</nav>
</div>
</header>

```

Este es el código que tenemos para el header , el header es de la clase menu-navegacion , para poder seleccionarlo específicamente en el CSS .

Para facilitar la posición del contenido , tenemos todo dentro de un div , para así orientarlo mediante flex , según la necesidad que tengamos.

Tenemos una lista , que mediante css , utilizando flex , cumplirá con la ubicación deseada.

Este es el código que hemos creado para el encabezado (header). El encabezado pertenece a la clase "menu-navegacion", lo cual nos permite seleccionarlo de manera específica en el archivo CSS.

Con el objetivo de facilitar la disposición del contenido, hemos encapsulado todo dentro de un elemento div. De esta manera, podemos orientarlo utilizando el modelo flex según nuestras necesidades.

Dentro de este div, hemos incluido una lista que, mediante el uso de CSS y la propiedad flex, se ajustará a la ubicación deseada

## Footer de las páginas

```

<footer>
    <div class="container-footer">
        <ul>
            <li> <a href=""> Sobre nosotros</a></li>
            <li>&middot;</li>
            <li><a href="">Aviso legal</a></li>
            <li>&middot;</li>
            <li><a href="">Política de privacidad</a></li>
            <li>&middot;</li>
            <li><a href="">Política de cookies</a></li>
        </ul>
        <div class="container-logos">
            <a href="">
    <p><time datetime="2023">2023</time> &copy; IA al
    <p>Actividad realizada por Álvaro Ruiz</p>
</div>
</footer>

```

En el footer tenemos otro div que engloba todo para la ubicación principal .

Dentro de la lista podemos encontrar `&middot;` , que mostrará un carácter especial de un punto.

En la versión para escritorio lo necesitamos , pero para las demás versiones los ocultaremos mediante CSS

Aquí está la redacción corregida:

En el pie de página (footer), hemos incluido otro div que abarca todo para su ubicación principal.

Dentro de la lista, se encuentra el carácter especial `"."` ( `&middot;`) , que representará un punto. Este elemento es necesario para la versión de escritorio, sin embargo, para las otras versiones, lo ocultaremos utilizando reglas CSS.

## Main index.html

Para el main , tenemos una primera parte que necesitamos una imagen de fondo , la cual hemos añadido mediante CSS

La segunda parte tenemos un article , el cual tiene un div y un section .

El primer div contiene un primer texto descriptivo , y la section contiene articles , los cuales mediante CSS , los diferenciaremos .

El section queda definido mediante flex , para ubicar cada article según queramos.

Dentro de cada article tenemos un figure que contiene una imagen y su correspondiente figcaption con una breve descripción de la imagen.

Aquí tienes la redacción corregida:

En la sección principal (main), hemos dividido la estructura en dos partes. En la primera parte, necesitamos una imagen de fondo, la cual hemos incorporado mediante CSS.

La segunda parte consta de un artículo (article) que contiene un div y una sección. El primer div alberga un texto descriptivo inicial, mientras que la sección contiene varios artículos. Utilizando CSS, diferenciaremos visualmente cada artículo.

La sección está definida mediante flexbox, lo que nos permite posicionar cada artículo según nuestras necesidades.

Dentro de cada artículo, hemos incluido un elemento figure que contiene una imagen y su correspondiente figcaption con una breve descripción de la imagen.

## **Main aplicaciones.html**

En el main teníamos una estructura principal compuesta de un header y un section , pero el section no tenía ningún título , a la hora de validar nos recomienda utilizar div , por ello pasó a ser un div.

Dentro del div tenemos article , con un header y texto.

Mediante css , haremos que cada article , sea como una especie de tarjeta , para diferenciar así cada article.

mediante CSS y flex , en el div principal , ubicaremos según queramos cada tarjeta , para así conseguir la organización deseada.

En la sección principal del elemento main, inicialmente teníamos una estructura compuesta por un encabezado (header) y una sección (section). Sin embargo, la validación sugirió utilizar un elemento div en lugar de section, por lo que se cambió.

Dentro de este div, hemos creado elementos article, cada uno con su propio encabezado (header) y contenido textual. Para mejorar la presentación visual y distinguir cada artículo, hemos aplicado estilos mediante CSS para darles una apariencia similar a tarjetas.

Utilizando las propiedades de CSS, especialmente la propiedad flex, hemos organizado los artículos dentro del div principal. Esta disposición flexible nos permite ajustar la posición de cada "tarjeta" según nuestras preferencias y lograr la organización deseada.

## **Main en-el-cine.html**

En este main podemos ver que contiene un h1 y un div .

Dentro del div , tenemos 4 div a modo de contenedor y todos comparten la misma estructura.

Dentro de cada uno , encontramos un iframe , el cual contiene un video de youtube .

También tenemos un texto , que seria una breve descripción de cada video.

Aquí tienes la redacción corregida:

En la sección principal (main), observamos la presencia de un elemento h1 y un div.

Dentro de este div, se han creado cuatro contenedores adicionales, todos con la misma estructura. Cada uno de estos contenedores incluye un iframe que tiene un video de YouTube. Asimismo, se ha incorporado un texto que proporciona una breve descripción para cada video.

## **Main despacho-42.html**

Dentro del main , podemos encontrar una lista < ul > y div .

La lista , contiene nuestra ubicación dentro de la página en la que nos encontramos.

En el div , encontramos dos section.

En el primer section , primero un figure con su correspondiente figcaption , y tras esto un texto descriptivo. Tras esto encontramos diversos article , que tiene como estructura un header y un texto. Entre todos estos article , encontramos un div , ya que por su estructura y la necesidades que teníamos era mas viable.

En el segundo section , encontramos h2 para describir este bloque , y un div , el cual contiene divs en su interior , cada div interior seria para referenciarse a otros articulos.

El div que engloba a los dos section nos serviría para orientar estos dos section según nuestras necesidades , ya que el segundo section en escritorio pasa a estar a la derecha y en versión móvil , lo encontramos el la parte inferior.

Aquí tienes la redacción corregida:

Dentro del elemento main, se encuentra una lista <ul> y un div.

La lista contiene información sobre nuestra ubicación dentro de la página.

En el div, se han incluido dos secciones. En la primera sección, se encuentra un figure con su correspondiente figcaption, seguido de un texto descriptivo. Luego,

encontramos varios elementos article, cada uno con un encabezado (header) y un texto descriptivo. Entre estos articles, se ha introducido un div, que fue considerado más adecuado dada la estructura y las necesidades específicas.

En la segunda sección, se utiliza un h2 para describir este bloque y un div que contiene divs en su interior. Cada div interior sirve como referencia a otros artículos. El div que engloba a ambas secciones permite ajustar la orientación según nuestras necesidades, posicionando el segundo section a la derecha en la versión de escritorio y en la parte inferior en la versión móvil.

## Código CSS

### Detalles generales

```
/*Importamos la fuente desde Google Fonts */

@import url("https://fonts.googleapis.com/css2?family=Source+Sans+3:ital,wght@0,400;0,700;1,400;1,700&family=Work+Sans:ital,wght@0,400;0,700;1,400;1,700&display=swap");

/*Reset de los valores */
* {
  margin: 0;
  padding: 0;
  list-style: none;
  text-decoration: none;
  border: none;
  outline: none;
}

/* Aplicar la fuente a todo el cuerpo del documento */
* {
  font-family: "Source Sans 3", sans-serif;
}
```

Aquí podemos ver el inicio del CSS , donde importamos la fuente deseada , hacemos un reset de los valores predefinidos y aplicamos la fuente importada a todo el

documento.

## Header de las páginas

Para el encabezado (header) en CSS, hemos definido el contenedor con un ancho del 70%, aplicando un margen automático para centrarlo. Además, lo hemos convertido en un contenedor flexible (flex) con dirección de columna.

Los enlaces han sido predefinidos según la estructura establecida, a los cuales les hemos añadido una transición para suavizar el cambio al pasar el cursor sobre ellos. También hemos incorporado características como el tamaño y el peso de la fuente, así como la separación entre líneas.

En cuanto a la lista, hemos utilizado flex para posicionarla según nuestras necesidades.

## Footer de las páginas

En el CSS del pie de página (footer), hemos establecido un margen superior de 25px, un fondo negro con texto blanco. El contenedor (container-footer) tiene relleno superior e inferior de 50px, ancho del 69%, centrado con margen automático.

Los enlaces en el pie de página tienen color blanco, la lista se muestra en fila con alineación izquierda y márgenes en los elementos de lista.

Dentro del contenedor del pie de página, los enlaces de la lista tienen una transición de color suave al pasar el cursor, cambiando a rojo. Los logos (container-logos) tienen márgenes y las imágenes tienen una transición de opacidad al pasar el cursor, disminuyendo a un 70%. El párrafo dentro del contenedor tiene un tamaño de fuente pequeño y espaciado entre líneas de 1.3.

## Resto del código CSS

Todas las páginas quedan estructuradas basándose en lo mismo .

Tenemos un contenedor principal , al cual le hemos puesto una class para que sea mas fácil seleccionarlo.

Dentro del contenedor principal mediante la propiedad flex , vamos jugando con los contenedores que tienen en su interior.

Para conseguir una buena estructura mediante flex , podemos ir cambiando el flexdirection , o el tamaño de los bloques hijos para así conseguir la estructura deseada .

He empleado la propiedad flex , ya que para el resultado que estábamos buscando creo que era la propiedad mas adecuada , ya que ajustando los tamaños es relativamente fácil conseguir el resultado esperado

Aquí tienes la redacción corregida:

Todas las páginas siguen una estructura común. Se ha establecido un contenedor principal al que se le ha asignado una clase para facilitar su selección en el CSS.

Dentro de este contenedor principal, se utiliza la propiedad flex para manipular los contenedores internos. Ajustando la dirección del flex (flex-direction) o el tamaño de los bloques hijos, logramos la estructura deseada.

La elección de la propiedad flex se considera apropiada para el resultado buscado, ya que ajustando los tamaños, es relativamente sencillo obtener el diseño esperado.

## Responsive Design mediante @media

### Main index.html

Para la sección principal ( `main` ), hemos definido un primer selector para el elemento con la clase `foto-fondo` . Este selector establece la propiedad `background-image` para asignar una imagen de fondo, utilizando la ruta `../img/bg-hero-home.jpg` . Además, hemos aplicado la propiedad `display: flex` para centrar el contenido vertical y horizontalmente en dicho bloque.

También se realizaron modificaciones en el tamaño de fuente y en la intensidad de la misma para mejorar la legibilidad y el diseño visual de la sección.

El código se organiza en tres bloques distintos, cada uno identificado por un selector específico: `bloque-primer` , `bloque-segundo` , y `bloque-tercero` . Cada bloque se trabaja individualmente para lograr la ubicación deseada mediante el uso de propiedades de flexbox.

Dentro de cada bloque, se emplea la propiedad `display: flex` para organizar y posicionar los elementos de manera adecuada. Se realizan ajustes en la justificación



del contenido, los márgenes y el tamaño de fuente para obtener el diseño deseado.

Estos cambios contribuyen a una presentación más estructurada y atractiva del contenido en la sección principal del sitio web.

En el CSS de la sección "foto-fondo", hemos definido un contenedor con altura de 30em y ancho del 100%. Utilizamos una imagen de fondo, color de texto #f4f6f7, y el contenido esta centrado mediante un contenedor flex.

El "bloque-primer" tiene un ancho máximo del 55%, centrado con margen automático, y es un contenedor flex con dirección de columna, alineación y justificación central, y un relleno en la parte inferior de 30px.

Para los elementos h2, p, y el segundo bloque "bloque-segundo", se han establecido tamaños de fuente y márgenes específicos.

En el bloque "bloque-tercero", los elementos de artículo tienen un ancho máximo del 30%, y las imágenes dentro de este bloque ocupan el 100% de su contenedor.

También se ha aplicado un estilo a los encabezados h3 con un cambio de color y subrayado al pasar el cursor. Los enlaces dentro de este bloque tienen una transición de color y subrayado al pasar el cursor.

Para el main , tenemos un primer selector para foto-fondo , background-image: url(..img/bg-hero-home.jpg); Tambien lo hace flex , para centrar el contenido en el centro de dicho bloque.

También cambiamos algun que otro estilo como es el tamaño de la fuente y su intensidad.

Mediante selectores tenemos el código partido en tres bloques , que tiene su correspondiente selector , bloque-primer , bloque-segundo y bloque-tercero.

Dentro de cada bloque , mediante Flex , vamos trabajando para dejarlo en la ubicación deseada.

Podemos ver otros cambios como son la justificación del contenido , los margin o el tamaño de la fuente.

Para el main , tenemos un primer selector para foto-fondo , background-image: url(..img/bg-hero-home.jpg); Tambien lo hace flex , para centrar el contenido en el centro de dicho bloque.

También cambiamos algun que otro estilo como es el tamaño de la fuente y su intensidad.

Mediante selectores tenemos el código partido en tres bloques, que tiene su correspondiente selector, `bloque-primer`, `bloque-segundo` y `bloque-tercero`.

Dentro de cada bloque, mediante Flex, vamos trabajando para dejarlo en la ubicación deseada.

Podemos ver otros cambios como son la justificación del contenido, los márgenes o el tamaño de la fuente.

Para la sección principal (`main`), hemos definido un primer selector para el elemento con la clase `foto-fondo`. Este selector establece la propiedad `background-image` para asignar una imagen de fondo, utilizando la ruta `../img/bg-hero-home.jpg`. Además, hemos aplicado la propiedad `display: flex` para centrar el contenido vertical y horizontalmente en dicho bloque.

También se realizaron modificaciones en el tamaño de fuente y en la intensidad de la misma para mejorar la legibilidad y el diseño visual de la sección.

El código se organiza en tres bloques distintos, cada uno identificado por un selector específico: `bloque-primer`, `bloque-segundo`, y `bloque-tercero`. Cada bloque se trabaja individualmente para lograr la ubicación deseada mediante el uso de propiedades de flexbox.

Dentro de cada bloque, se emplea la propiedad `display: flex` para organizar y posicionar los elementos de manera adecuada. Se realizan ajustes en la justificación del contenido, los márgenes y el tamaño de fuente para obtener el diseño deseado.

Estos cambios contribuyen a una presentación más estructurada y atractiva del contenido en la sección principal del sitio web.

Para la sección principal (`main`), hemos definido un primer selector para el elemento con la clase `foto-fondo`. Este selector establece la propiedad `background-image` para asignar una imagen de fondo, utilizando la ruta `../img/bg-hero-home.jpg`. Además, hemos aplicado la propiedad `display: flex` para centrar el contenido vertical y horizontalmente en dicho bloque.

También se realizaron modificaciones en el tamaño de fuente y en la intensidad de la misma para mejorar la legibilidad y el diseño visual de la sección.

El código se organiza en tres bloques distintos, cada uno identificado por un selector específico: `bloque-primer`, `bloque-segundo`, y `bloque-tercero`. Cada bloque se

trabaja individualmente para lograr la ubicación deseada mediante el uso de propiedades de flexbox.

Dentro de cada bloque, se emplea la propiedad `display: flex` para organizar y posicionar los elementos de manera adecuada. Se realizan ajustes en la justificación del contenido, los márgenes y el tamaño de fuente para obtener el diseño deseado.

Estos cambios contribuyen a una presentación más estructurada y atractiva del contenido en la sección principal del sitio web.

Para conseguir el responsive design deseado tanto para escritorio , tablets y moviles , hemos recurrido a la propiedad `media`.

En este caso hemos partido de la versión escritorio y hemos ido modificandola para adaptarla al tamaño de pantalla de otros dispositivos.

Para las tablets , hemos utilizado →

```
@media only screen and (max-width: 991px) {  
}
```

Donde tan solo modificando alguno de los tamaños , gracias a utilizar `flex` , hemos conseguido adaptarla segun las necesidades que teniamos para tablets.

Para moviles , hemos utilizado →

```
@media only screen and (max-width: 575px) {  
}
```

Donde , igual que en el caso anterior , tan solo modificando algunos tamaños y el `flex-direction` , hemos conseguido , el resultado final esperado.

En la implementación del Responsive Design, hemos empleado la propiedad `media` para lograr la adaptabilidad deseada en distintos dispositivos, como escritorio, tablets y móviles.

Comenzamos estableciendo la versión para escritorio como punto de partida. A partir de ahí, hemos utilizado `media query` para ajustar en función del tamaño de pantalla.

Para las tablets, hemos aplicado la siguiente regla:

```
@media only screen and (max-width: 991px) {  
  /* Modificaciones y ajustes para tablets */  
}
```

```
}
```

Gracias a la propiedad flex y modificar algunos tamaños, hemos logrado adaptar la presentación según las necesidades específicas para tablets.

Para los dispositivos móviles, hemos empleado la siguiente regla:

```
@media only screen and (max-width: 575px) {  
  /* Modificaciones y ajustes para móviles */  
}
```

Similar al caso anterior, mediante la modificación de tamaños y ajustes en la dirección de flex, hemos obtenido el resultado final esperado para la versión móvil.

## Validación

Tanto las páginas HTML , como CSS , han sido validadas y cumplen las reglas establecidas.

## GitHub


### Acceso al repositorio

<https://github.com/Alvaret/Practica-3-Layout-CSS-y-Responsive-Design>

### Acceso a GitHub Pages

Inicio

Página de Inicio para Practica-3

 <https://alvaret.github.io/Practica-3-Layout-CSS-y-Responsive-Design/>

---

Práctica realizada por Álvaro Ruiz Poveda — 2 DAW Semipresencial