Casos prácticos
Edición Web I

Edición Web I
Jonathan Faus

Caso práctico 1

Centrar contenido con Flex y Padding

Paso1:

Cremamos 2 contenedores a los que le aplicaremos diferentes fondos al que asignaremos las clases **content1**, **content2**. Creamos dentro de cada contenedor otro contenedor hijo al que le aplicaremos la classe **info1**, dentro de esta colocamos un <h2> con la clase **title1** y un <p> con la clase **text1**.

Replicamos las clases en el segundo contenedor, como podemos ver en la captura del código.



```
<div class="content1">
    <div class="info1">
        <h2 class="title1">Título ección</h2>
        <p class="text1">
            Lorem ipsum dolor sit amet, consectetur adipiscir
            ultrices urna, ac fringilla neque fermentum sed.
            Sed ultrices fringilla ipsum a dapibus. Donec nec
            Nullam iaculis arcu neque, vitae gravida libero t
            nec pellentesque erat. Quisque hendrerit massa qu
            Morbi elementum varius porta.
        </p>
        <a href="" class="button1">Botón</a>
    </div>
</div>

<div class="content2">
    <div class="info2">
        <h2 class="title2">Título ección</h2>
        <p class="text2">
            Lorem ipsum dolor sit amet, consectetur adipiscir
            ultrices urna, ac fringilla neque fermentum sed.
            Sed ultrices fringilla ipsum a dapibus. Donec nec
            Nullam iaculis arcu neque, vitae gravida libero t
            nec pellentesque erat. Quisque hendrerit massa qu
            Morbi elementum varius porta.
        </p>
        <a href="" class="button2">Botón</a>
    </div>
</div>
```

Paso2:

.content1:
añadimos una imagen de fondo y una altura de 800px.

.info1:
Anchura del 60%. Añadimos padding y un color de fondo blanco con transparencia.

.title1 y text1:
Asignamos color a ambas clases y un tamaño grande al título.

.button1:
Ceamos un botón con las siguientes características.

!!! Cuando hacemos hover queremos que el coor del botón sea transparente, vemos una nueva propiedad del color:
Background-color: transparent;

```css
/*contenido 1*/
.content1{
    background-image: url("img/blue-futre-room.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;

    height:800px;
}


.info1{
    background-color: rgba(255, 255, 255, 0.443);
    width:60%;
    padding:80px;
}

.title1{
    font-size: 32px;
    color: white;
}

.text1{
    color: rgb(8, 8, 228);
}

.button1{
    background-color: white;
    color: rgb(8, 8, 228);
    padding:8px 16px;
    border: 2px white solid;
    transition: 0.6s;
    text-decoration: none;
}

.button1:hover{
    background-color: transparent;
    color: white;
    transition: 0.6s;
}
```

```css
.info1{
    background-color: rgba(255, 255, 255, 0.443);
    width:60%;
    padding:80px;
}
```




```css
.content1{
    background-image: url("img/blue-futre-room.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;

    height:800px;
}
```

Paso 3 (centrado flex):

Centramos el contenido que hay dentro de content1 aplicando flex.

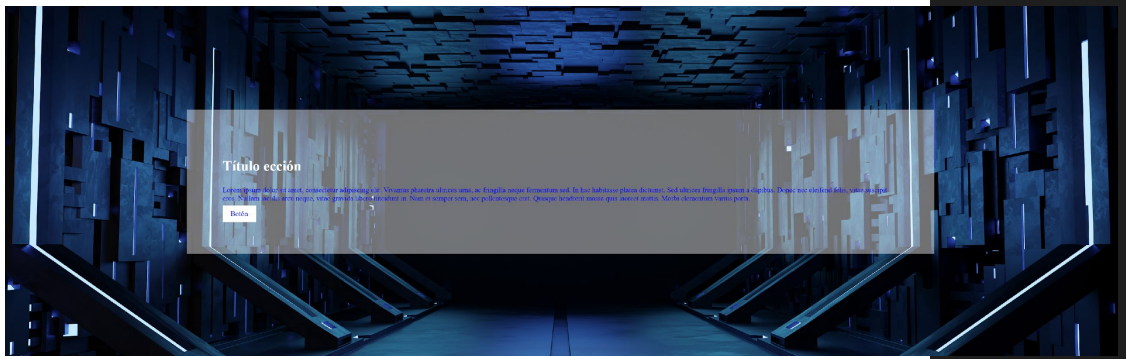
Ahora el div info1 que está dentro de content1 estará centrado en el eje principal (row) y en el eje secundario (column):

```css
/*contenido 1*/
.content1{
    background-image: url("img/blue-futre-room.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;

    height:800px;

    display: flex;
    justify-content: center;
    align-items: center;

}
```



Paso 4:

.content2:
añadimos una imagen de fondo, **no añadimos ninguna medida a nuestro contenedor padre.**

.info2:
Añadimos padding y un color de fondo blanco con transparencia. **no añadimos ninguna medida a nuestro contenedor hijo.**

Duplicamos las clases title, text y button y les asignamos un 2 al final de la clase.

```css
/*contenido 2*/
.content2{
    background-image: url("img/purple-future-room.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;

}

.info2{
    background-color: rgba(255, 255, 255, 0.443);
    padding:80px;
}

.title2{
    font-size: 32px;
    color: white;
}

.text2{
    color: rgb(8, 8, 228);
}

.button2{
    background-color: white;
    color: rgb(8, 8, 228);
    padding:8px 16px;
    border: 2px white solid;
    transition: 0.6s;
    text-decoration: none;
}

.button2:hover{
    background-color: transparent;
    color: white;
    transition: 0.6s;
}
```

Paso 4 centrado padding (explicación):

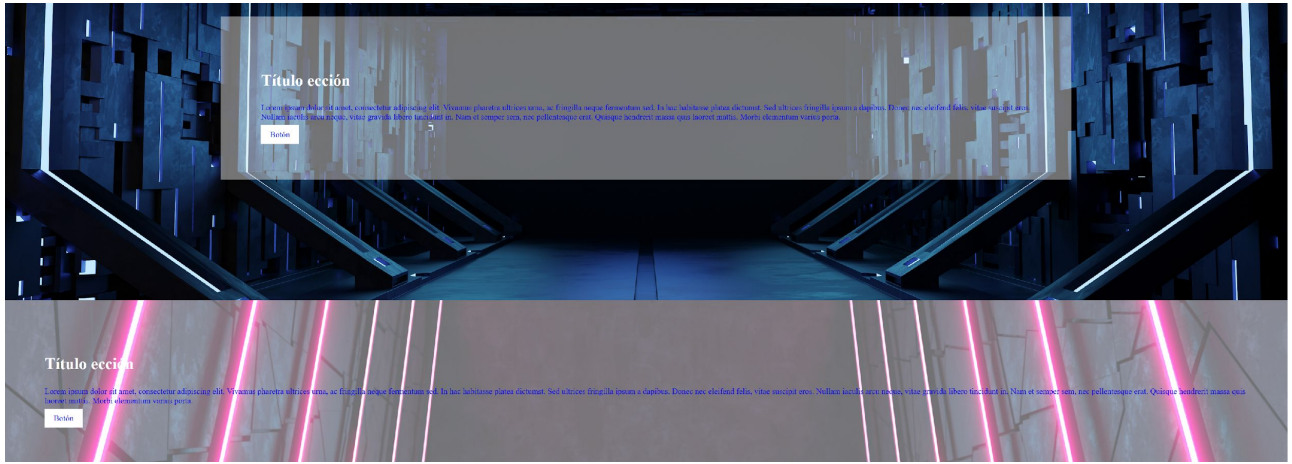
- Como el contenedor padre no tiene altura coge la altura del contenedor hijo, ahora el contenedor padre e hijo tienen las mismas dimensiones.

- Al contenedor hijo tampoco lo hemos dado medidas por eso coge toda la anchura que puede (100%) y la altura se la determina el padding.

- Ahora para centrar el contenedor hijo respecto al padre vamos a añadir padding al contenedor padre.

```css
/*contenido 2*/
.content2{
    background-image: url("img/purple-future-room.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;

    padding: 5% 16%;
}
```



Paddingo contenedor padre **.content2**

```css
.content2{
    background-image: url("img/purple-future-room.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;

    padding: 5% 16%;
}
```



Padding contenedor hijo **.info2**

```css
.info2{
    background-color: rgba(255, 255, 255, 0.443);
    padding:80px;
```

Resultado final

Contenido1:

- Definimos nosotros la medida de los contenedores
- Centramos mediante flex.

Contendio2:

- No definimos tamaño(la anchura siempre será la máxima, la altura la define lo que ocupen los elementos html)
- Centramos añadiendo padding al contenedor padre.




```css
/*contenido 1*/
.content1{
    background-image: url("img/blue-futre-room.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;

    height:800px;

    display: flex;
    justify-content: center;
    align-items: center;
}


.info1{
    background-color: rgba(255, 255, 255, 0.443);
    width:60%;
    padding:80px;
}


/*contenido 2*/
.content2{
    background-image: url("img/purple-future-room.jpg");
    background-size: cover;
    background-repeat: no-repeat;
    background-position: center;

    padding: 5% 16%;
}

.info2{
    background-color: rgba(255, 255, 255, 0.443);
    padding:80px;
}
```

Edición Web I
Jonathan Faus

Caso práctico 2

Distribuir elementos navbar (flex)



Paso 1:

Instalamos una fuente de Google fonts.

Fuente Host Grotesk:

Ir a Google fonts

Html
Pegamos el código de Google fonts en el head

```
    <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Host+Grotesk:ital,wght@0,300..800;1,300..
```

Css
Pegamos el código de Google en el css, cambiamos la clase que nos da por defecto por: h1,h2,h3,h4,h5,p,a

```
h1,h2,h3,h4,h5,p{
  font-family: "Host Grotesk", sans-serif;
  font-optical-sizing: auto;
  font-weight: 300;
  font-style: normal;
}
```



Paso 2:

Creamos nuestro contenedor **nav**, dentro de este:

H1 al que le asignamos la clase **logo**

Contenedor hijo al que le asignamos la clase **.menu**, dentro de este 4 enlaces que formarán nuestro menú, asignamos a cada enlace la clase **item**.

Creamos otro contenedor al que le añadimos la clase **rrss**, dentro de este colocamos 3 imagenes con un ancho de 40px.

```
<nav>

    <h1 class="logo">Tienda <span style="color: white;">online</span></h1>

    <div class="menu">
        <a href="" class="item">menu 1</a>
        <a href="" class="item">menu 2</a>
        <a href="" class="item">menu 3</a>
        <a href="" class="item">menu 4</a>
    </div>

    <div class="rrss">
        <img src="img/instagram-icon.png" width="40px">
        <img src="img/linkedin-icon.png" width="40px">
        <img src="img/twitter-icon.png" width="40px">
    </div>

</nav>
```



Paso 3:

- Añadimos color y padding al contendor nav
- Añadimos color, tamaño de fuente y grosor de fuente al logo
- Añadimos color, tamaño de fuente y grosor de fuente al logo y quitamos la decoración de los enlaces.

!!! Los elementos del menú y las imágenes de las redes sales en horizontal porque tanto el elemento <a> como el elemento <img> son elementos en línea y por defecto no salen apilados como el resto de elementos html.




```css
nav{
    background-color: rgb(141, 228, 199);
    padding:16px;

}

.logo{
    font-size: 28px;
    font-weight: 800;
    color: rgb(5, 193, 131);
}


.item{
    color: rgb(5, 193, 131);
    font-size:24px;
    font-weight: 400;
    text-decoration: none !important;
}
```

Tienda online menu 1 menu 2 menu 3 menu 4

Paso 4:

- Añadimos flex a nuestro contenedor principal **nav** para alienar nuestros 3 elementos:

-Titulo/logo
-Contenedor menú
-Contenedor RRSS

```css
nav{
    background-color: rgb(141, 228, 199);
    padding:16px;

    display: flex;
    justify-content: space-between;
    align-items: center;

}
```

Tienda online menu 1 menu 2 menu 3 menu 4



Paso 5:

Los elementos de menú y RRSS salen alineados horizontalmente sin usar flex (por ser elementos en línea)...

Pero podemos añadir diplay:flex; y utilizar la propiedad GAP para darle espaciado a estos elementos entre si:

```
.menu{
    display: flex;
    gap:16px;
  }

.rrss{
    display: flex;
    gap:16px;
}
```

Sin gap

