



inicio

Temario

03

Flexbox

- ✓ Definición
- ✓ Propiedades

04

Grid

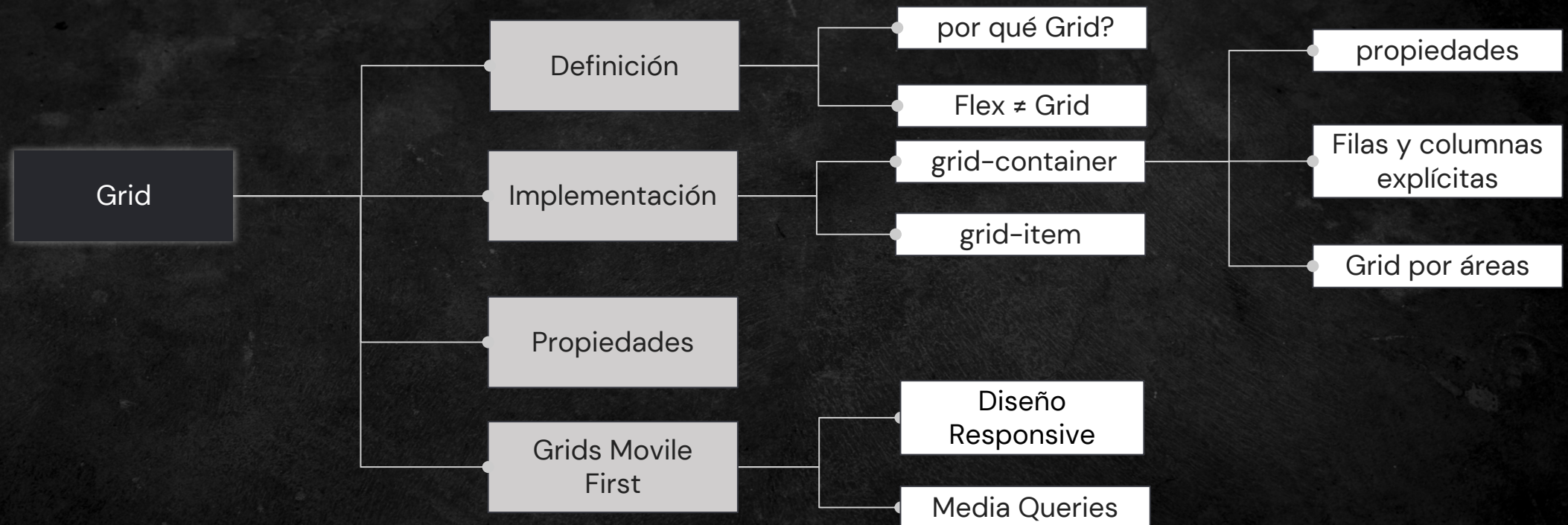
- ✓ Grid
- ✓ Grid + flexbox
- ✓ Implementación
- ✓ Diseño responsive
- ✓ Media queries
- ✓ Mobile first
- ✓ Grids+Flex+@Media

05

Pseudoclasas BEM -GIT

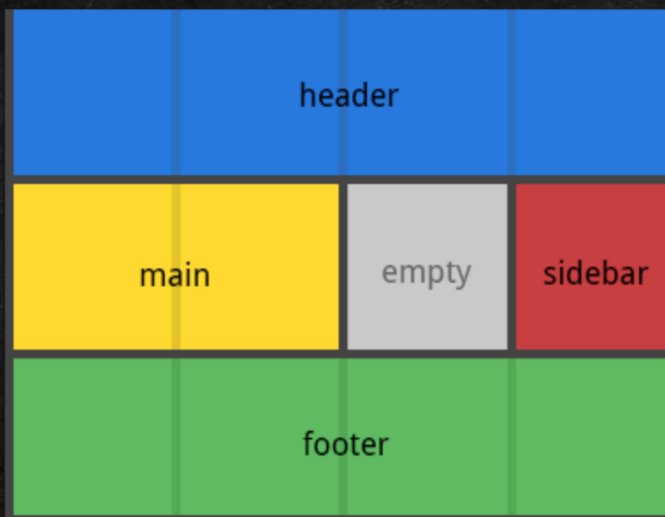
- ✓ Pseudoclasas
- ✓ Pseudoelementos
- ✓ BEM
- ✓ Git
- ✓ Comandos básicos
- ✓ Repositorios
- ✓ Ramas

Mapa de conceptos



Definición

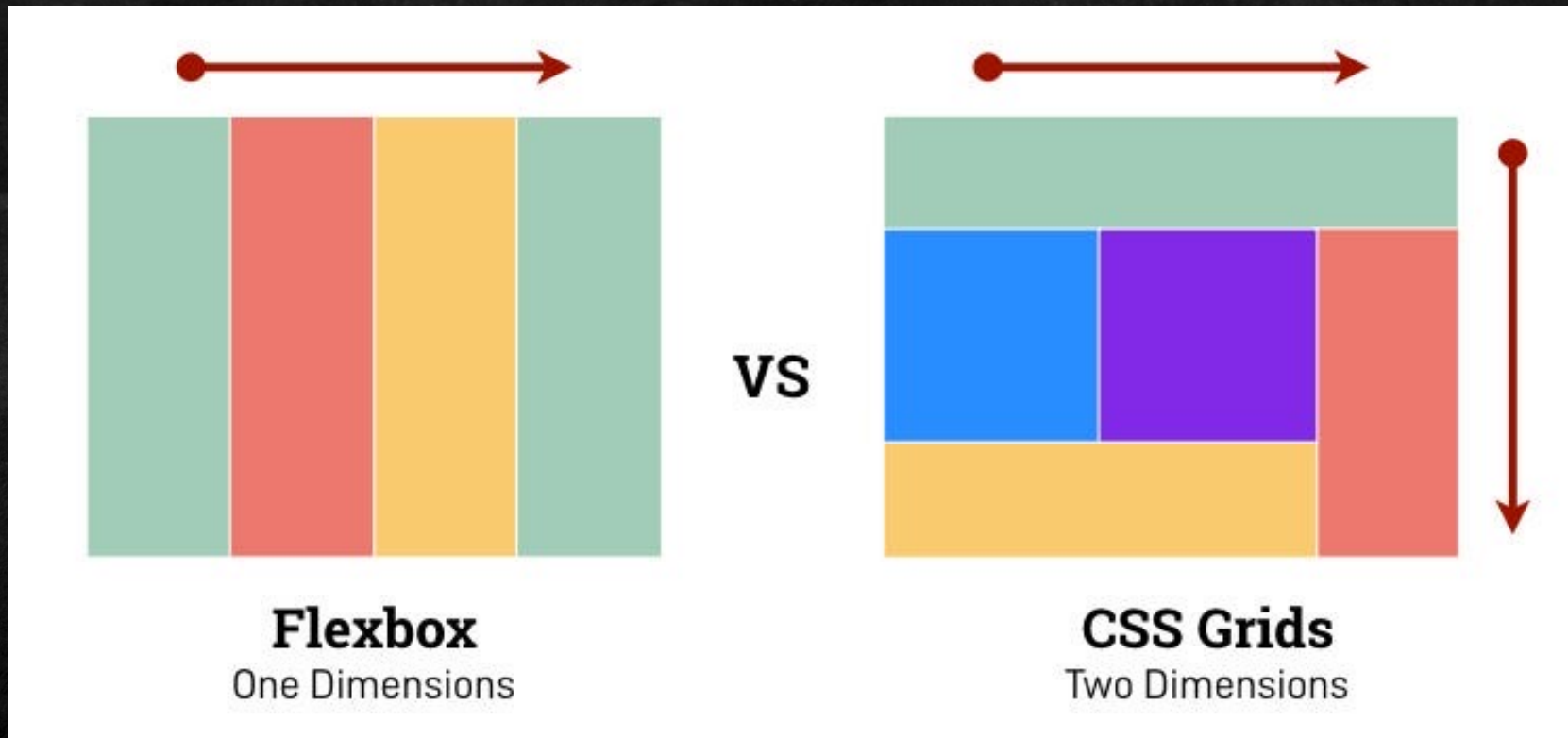
Por qué: Grid es un sistema de maquetación muy versátil que está disponible para CSS.



- ✓ El grid layout es un sistema de dos dimensiones [filas y columnas] que permite alinear los elementos de forma simultánea en ambos ejes.
- ≠ (Flexbox funciona en una dimensión).

Diferencias

FLEX Vs GRID



Construyamos el layout de la derecha con Flex

grid-container (padre)

Filas & Columnas Explícitas



```
1 <section class="grid-container">
2   <div class="grid-item">Item 1</div>
3   <div class="grid-item">Item 2</div>
4   <div class="grid-item">Item 3</div>
5 </section>
```



```
1 .grid-container {
2   display: grid;
3 }
```

Propiedad	Función
grid-template-columns	Establece el tamaño de cada columna (col 1, col 2...).
grid-template-rows	Establece el tamaño de cada fila (fila 1, fila 2...).
grid-template-areas	Indica la disposición de las áreas en el grid. Cada texto entre comillas simboliza una fila.
column-gap	Establece el tamaño de los huecos entre columnas (líneas verticales).
row-gap	Establece el tamaño de los huecos entre filas (líneas horizontales).

grid-container (padre)

Filas & Columnas Explícitas



```
1 <section class="grid-container">
2     <div class="grid-item">Item 1</div>
3     <div class="grid-item">Item 2</div>
4     <div class="grid-item">Item 3</div>
5 </section>
```



```
1 .grid-container {
2     display: grid;
3     grid-template-columns: 300px 100px; /* 2 columnas */
4     grid-template-rows: 40px 100px; /* 2 filas */
5 }
```

- Creamos una grilla especificando cuántas columnas y filas queremos -

Propiedad	Función
grid-template-columns	Establece el tamaño de cada columna (eje horizontal).
grid-template-rows	Establece el tamaño de cada fila (eje vertical).

grid-container (padre)

Filas & Columnas Explícitas



```
1 <section class="grid-container">
2     <div class="grid-item">Item 1</div>
3     <div class="grid-item">Item 2</div>
4     <div class="grid-item">Item 3</div>
5 </section>
```



```
1 .grid-container {
2     display: grid;
3     grid-template-columns: 300px 100px; /* 2 columnas */
4     grid-template-rows: 40px 100px; /* 2 filas */
5 }
```

- Creamos una grilla especificando cuántas columnas y filas queremos -

Propiedad	Función
grid-template-columns	Establece el tamaño de cada columna (eje horizontal).
grid-template-rows	Establece el tamaño de cada fila (eje vertical).

grid-container (padre)

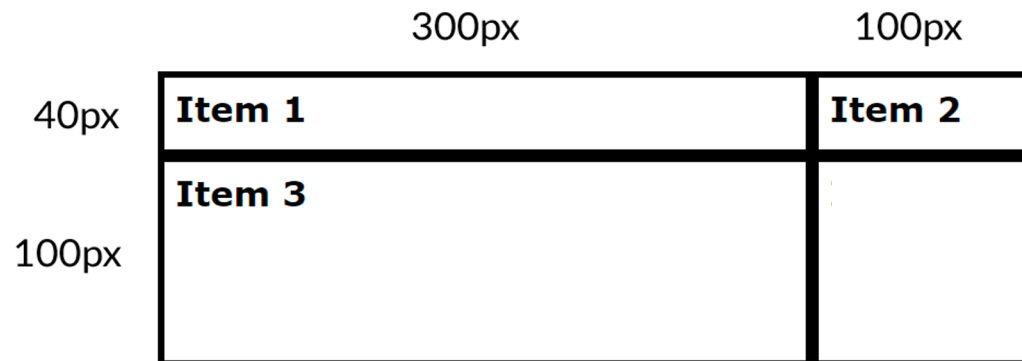
Filas & Columnas Explícitas



```
1 <section class="grid-container">
2     <div class="grid-item">Item 1</div>
3     <div class="grid-item">Item 2</div>
4     <div class="grid-item">Item 3</div>
5 </section>
```



```
1 .grid-container {
2     display: grid;
3     grid-template-columns: 300px 100px; /* 2 columnas */
4     grid-template-rows: 40px 100px; /* 2 filas */
5 }
```



Nota: Podemos utilizar otras unidades y combinarlas, (**%** , **fr** , **auto** [que obtiene el tamaño restante]).

grid-container (padre)

Filas & Columnas Explícitas



```
1 <section class="grid-container">
2     <div class="grid-item">Item 1</div>
3     <div class="grid-item">Item 2</div>
4     <div class="grid-item">Item 3</div>
5 </section>
```



```
1 .grid-container {
2     display: grid;
3     grid-template-columns: repeat(6, 1fr);
4     grid-template-rows: repeat(6, 1fr);
5     /* repeat([cantidad],[tamaño]) */
6 }
```

Item 1	Item 2	Item 3			

grid-container (padre)

Grid por Áreas

```
1 <div id="grilla">
2   <header>Encabezado</header>
3   <section id="productos">Seccion_Productos</section>
4   <section id="servicios">Seccion_Servicios</section>
5   <nav>Menu</nav>
6   <aside>Enlaces</aside>
7   <footer>Pie de página</footer>
8 </div>
```

Navegación	Encabezado	
	Seccion_Productos	Enlaces
	Seccion_Servicios	
	Pie de página	

Nota:

none →

Indica que no se colocará ninguna celda.

. [uno o más puntos] →

Indica una celda vacía.

```
1 #grilla {
2   display: grid;
3   grid-template-areas:
4     "nav header header"
5     "nav productos publicidad"
6     "nav servicios publicidad"
7     "nav footer footer";
8   grid-template-rows: 100px 1fr 1fr 75px;
9   grid-template-columns: 20% auto 15%;
10 }
11 header {
12   grid-area: header;
13 }
14 footer {
15   grid-area: footer;
16 }
17 section#productos {
18   grid-area: productos;
19 }
20 section#servicios {
21   grid-area: servicios;
22 }
23 nav {
24   grid-area: nav;
25 }
26 aside {
27   grid-area: publicidad;
28 }
```

grid-item (hijos)



```
1 <section id="padre">
2   <div>item</div>
3   <div class="hijo">item</div>
4   <div>item</div>
5   <div>item</div>
6 </section>
```



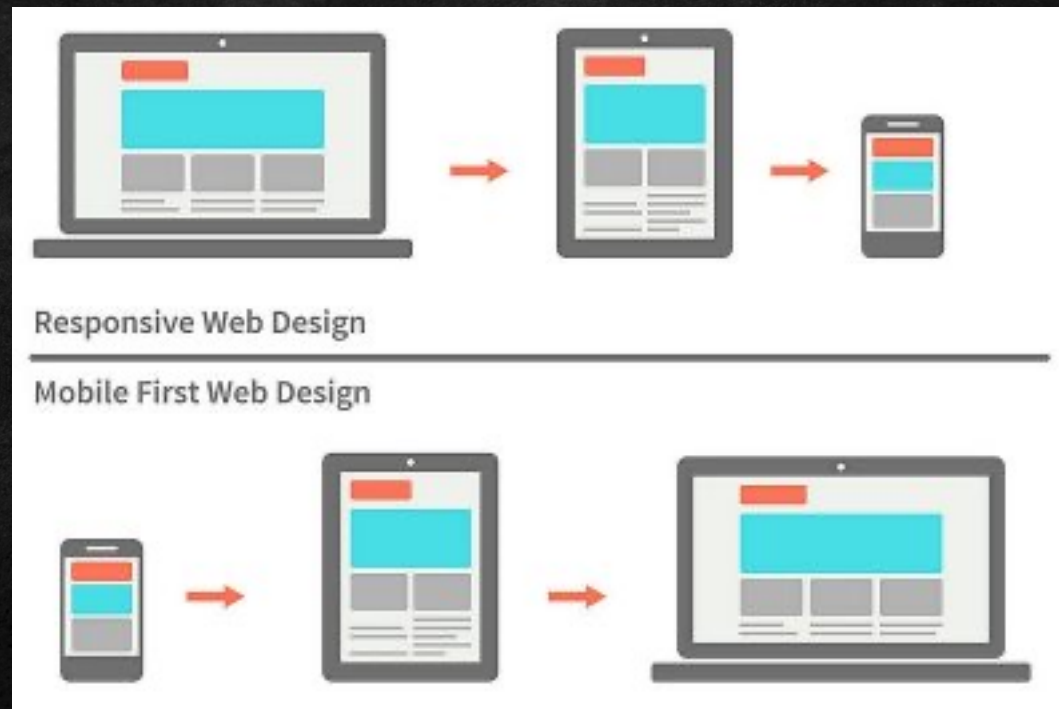
```
1 .hijo { /* un solo hijo */
2     /* --- probando propiedades --- */
3     justify-self: start; /* start | end | center | stretch */
4     align-self: start; /* start | end | center | stretch */
5 }
6
7 #padre div { /* cada hijo */
8     border: solid 1px;
9     background-color: yellow;
10    font-size: 21px;
11    padding: 5px;
12 }
```

Propiedad	Función
justify-self	Altera la justificación del ítem hijo en el eje horizontal.
align-self	Altera la alineación del ítem hijo en el eje vertical.
grid-area	Indica un nombre al área especificada, para su utilización con grid-template-areas.

se aplican a cada ítem hijo de la cuadrícula, para alterar o cambiar el comportamiento específico de dicho elemento. **Prueba sus propiedades [aquí](#).**

Grid mobile first (Responsive desing)

El diseño responsive se refiere a la idea de que un sitio web debería mostrarse igualmente bien en todo tipo de dispositivo, desde monitores de pantalla panorámica hasta teléfonos móviles.



Media Queries (Responsive desing)

Son una forma de aplicar **condicionales** al css.

MOBILE FIRST

```
@media (min-width:...) { ... }
```



Mobile First Desktop last

Ejemplo


```
1  .miestilo {  
2      background-color: green;  
3  }  
4  
5  @media (max-width: 480px) {  
6      .miestilo {  
7          background-color: red;  
8      }  
9  }
```

Pantallas extra pequeñas (xs) fondo rojo sino verde

Media Queries (Responsive desing)


Operador **AND** : Permite sumar diferentes indicaciones

Ejemplos



```
1 @media (min-width: 400px) and (max-width: 700px) {
2     .miestilo {
3         text-align: left;
4     }
5 }
```

Estilo aplicado a pantallas que están desde 400px a 700px



```
1 @media (min-width: 700px) and (orientation: landscape){
2     .miestilo {
3         text-align: left;
4     }
5 }
```

Estilo aplicado a pantallas superiores a 700px y en formato horizontal

Media Queries (Responsive desing)

Breakpoints : Categorías

Tamaño	Dispositivo
320px	Para dispositivos con pantallas pequeñas, como los teléfonos en modo vertical
480px	Para dispositivos con pantallas pequeñas, como los teléfonos, en modo horizontal
600px	Tabletas pequeñas, como el Amazon Kindle (600×800) y Barnes & Noble Nook (600×1024), en modo vertical
768px y 1023px	Tabletas de diez pulgadas como el iPad (768×1024), en modo vertical
1024px	Tabletas como el iPad (1024×768), en modo horizontal, así como algunas pantallas de ordenador portátil, netbook, y de escritorio
1200px	Para pantallas panorámicas, principalmente portátiles y de escritorio

Grid mobile first (Responsive desing)

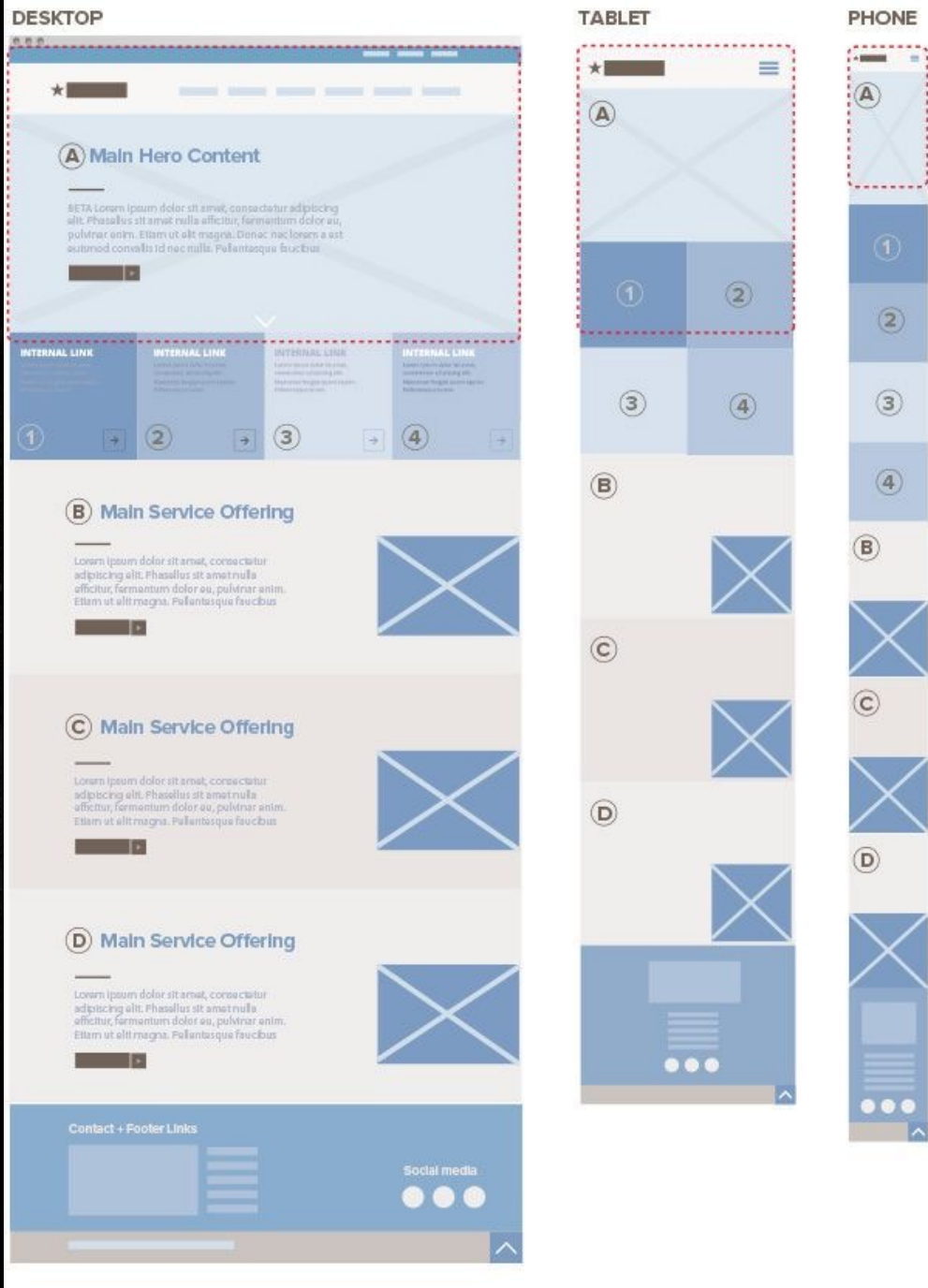


Ejercicios en Clase

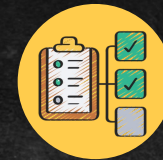
Encabezado
Navegación
Seccion_Productos
Seccion_Servicios
Enlaces
Pie de página

Encabezado	Nav
Seccion_Productos	
Seccion_Servicios	
Pie de página	Enlaces

Navegación	Encabezado	
	Seccion_Productos	Enlaces
	Seccion_Servicios	
	Pie de página	



Grids + Flex + @Media



Ejercicios Extraclase

