

***RECUERDA PONER A GRABAR LA
CLASE***





¿DUDAS DEL ON-BOARDING?

MIRALO AQUI



Clase 07. DESARROLLO WEB

GRIDS II





OBJETIVOS DE LA CLASE

- Conocer qué es un Diseño Responsive
- Conocer qué es un Diseño Mobile First
- Generar Diseños utilizando grids y flexbox

GLOSARIO:

Clase 6

CSS Grid: es el sistema de maquetación más potente que hay disponible. Se trata de un sistema en 2D que permite definir filas y columnas (a diferencia de, por ejemplo, Flexbox, el cual funciona en una única dimensión).

Diseño responsive: se refiere a la idea de que un sitio web debería mostrarse igual de bien en todo tipo de dispositivo, desde monitores de pantalla panorámica hasta teléfonos móviles. El diseño responsive se logra a través de "Media Queries" de CSS. Pensemos en las Media Queries como una forma de aplicar condicionales a las reglas de CSS.

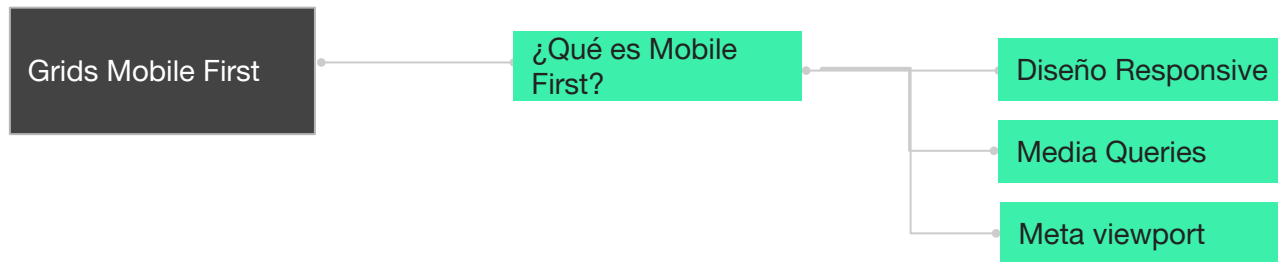
Mobile First: significa crear el código primero para los dispositivos más pequeños que los usuarios probablemente tengan, como teléfonos o tabletas. Implica trabajar en el dispositivo más pequeño, y luego acumular desde allí todo en el mismo código y el mismo proyecto, en lugar de hacer uno nuevo para cada tamaño de pantalla.

Meta viewport: una etiqueta <meta> viewport da al navegador las instrucciones sobre cómo controlar las dimensiones, y el ajuste a escala de la página.

MAPA DE CONCEPTOS

MAPA DE CONCEPTOS CLASE 7

¡Para
recordar!



CRONOGRAMA DEL CURSO

Clase 6



Grids



PRÁCTICAS DE LO
VISTO EN CLASE

Clase 7



Grids II



PRÁCTICAS DE LO
VISTO EN CLASE

Clase 8



Animaciones, transformaciones y transiciones



APLICANDO GRIDS



TRANSFORMACIONES Y
ANIMACIONES



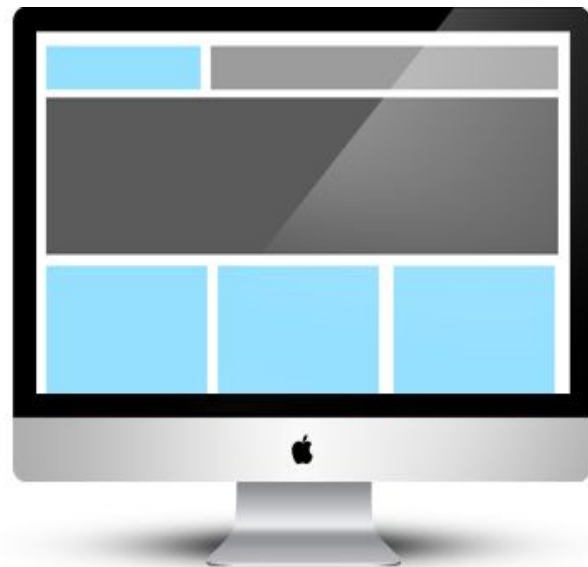
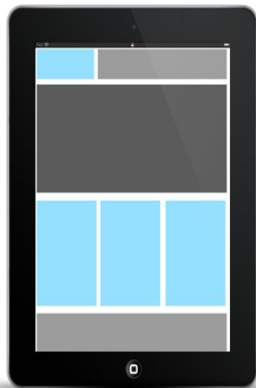
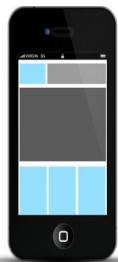
GUIÓN DE LA CLASE

Accede al material complementario [aquí](#).



GRIDS MOBILE FIRST

MOBILE FIRST





MOBILE FIRST

Antes de hablar de “Mobile-First” debemos hacer una referencia al llamado Diseño Responsive. Se refiere a la idea de que un sitio web debería mostrarse igual de bien en todo tipo de dispositivo, desde monitores de pantalla panorámica hasta teléfonos móviles.

Es un enfoque para el diseño y desarrollo web que elimina la distinción entre la versión amigable para dispositivos móviles de un sitio web y su contraparte de escritorio. Con un diseño responsive ambos son lo mismo.

MOBILE FIRST

"Mobile First" significa crear el código primero para los dispositivos más pequeños que los usuarios probablemente tengan, como teléfonos o tabletas.

Trabajar en el dispositivo más pequeño y luego acumular desde allí todo en el mismo código y el mismo proyecto, en lugar de uno nuevo para cada tamaño de pantalla.

MOBILE FIRST

CoderTips



Se recomienda:

- 👉 Primero trabajar el código para que se reproduzca perfectamente en un teléfono.
- 👉 Segundo, ajustar para que se ejecute en una tableta.
- 👉 Por último, trabajar en un dispositivo de escritorio.



CODER HOUSE

MOBILE FIRST



Cualquier estilo dentro del siguiente Media Query se ejecutará cuando el tamaño de la pantalla sea de al menos 768px de ancho -tablet portrait iPad Mini- pero no cuando el tamaño de la pantalla sea menor:

Estilos Mobile

```
@media screen and (min-width: 768px) {  
  .body {  
    background-color: #000000;  
  }  
}
```

MEDIA QUERIES

- 👉 El diseño responsive se logra a través de "Media Queries" de CSS.
- 👉 Pensemos en las Media Queries como una forma de aplicar condicionales a las reglas de CSS.
- 👉 Estas últimas le dicen al navegador qué reglas debe **ignorar o aplicar** dependiendo del dispositivo del usuario.

BREAK POINTS - 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

GRIDS MOBILE FIRST



Estructura HTML paso a paso

- 👉 Lo primero es asignarle a nuestro contenedor la propiedad de `display: grid;`
- 👉 Luego número de columnas y filas que tendrá nuestra grilla, y un espacio de separación.
- 👉 Definir el área que ocupará cada caja de nuestro contenedor, primero le asignaremos un nombre y un color característico.
- 👉 Definir cómo queremos que cada área sea acomodada en nuestro layout

TABLET



Siguiendo el ejemplo de las Grillas por áreas

👉 Para la versión tablet lo primero que hacemos es cambiar la disposición de las columnas de nuestro Grid.

👉 Luego cambiar la disposición de los ítems, este vez usando el recurso de *grid-row* y *grid-column*, que es el método corto de *grid-row-start/end* *grid-column-start/end*

```
#grilla { display: grid; }  
  
@media screen and (min-width: 768px) {  
    #grilla {  
        grid-template-columns: repeat(4, 1fr);  
    }  
    .border {  
        border: 4px solid black;  
        background-color: blue;  
    }  
}
```

DESKTOP



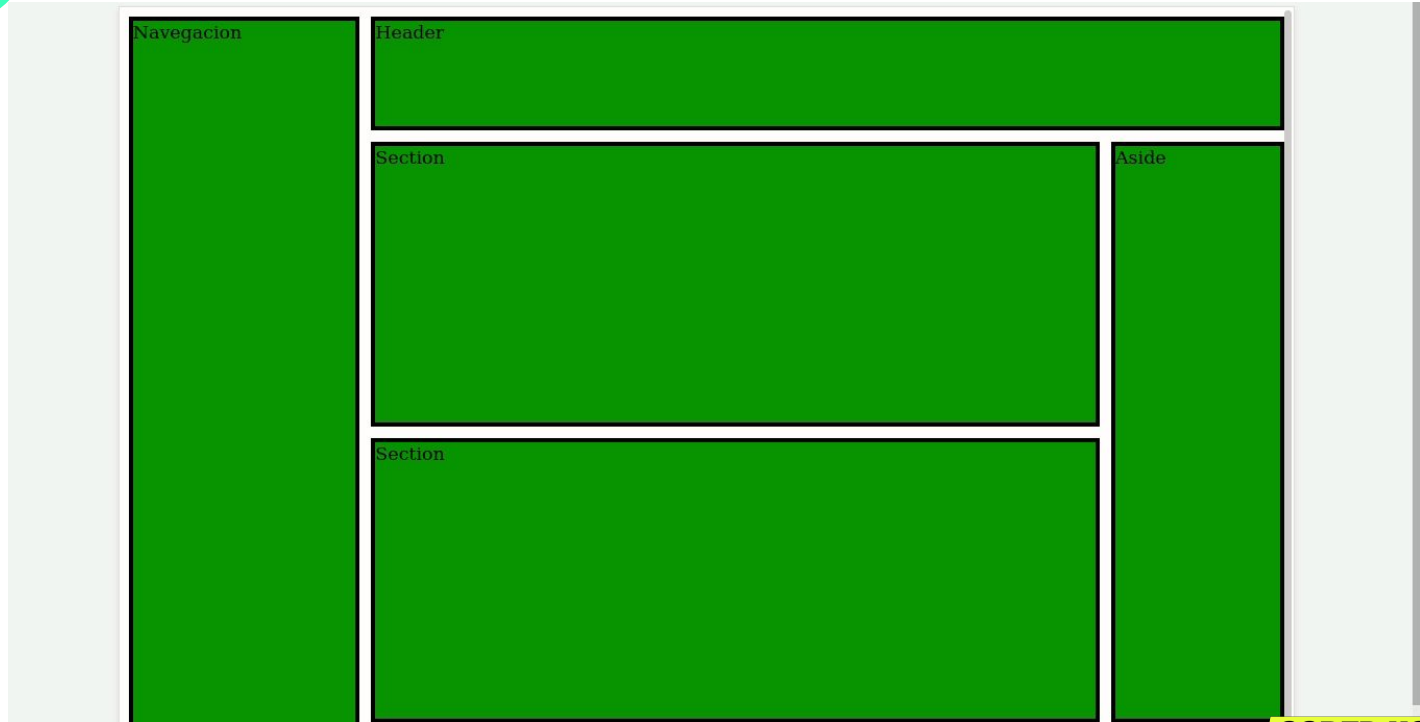
Siguiendo el ejemplo de las Grillas por áreas

👉 Cambiamos la disposición de la grilla.

👉 Y ahora una vez más cambiamos la disposición de los ítems

```
@media screen and (min-width: 1024px) {  
  #grilla {  
    grid-template-columns: repeat(3, 1fr);  
  }  
  .border {  
    border: 4px solid black;  
    background-color: green;  
  }  
}
```

RESULTADO



META VIEWPOINT

Las páginas optimizadas para diferentes dispositivos deben incluir la etiqueta `<meta> viewport` en el encabezado del documento HTML. Una etiqueta `<meta> viewport` da al navegador las instrucciones sobre cómo controlar las dimensiones y el ajuste a escala de la página.

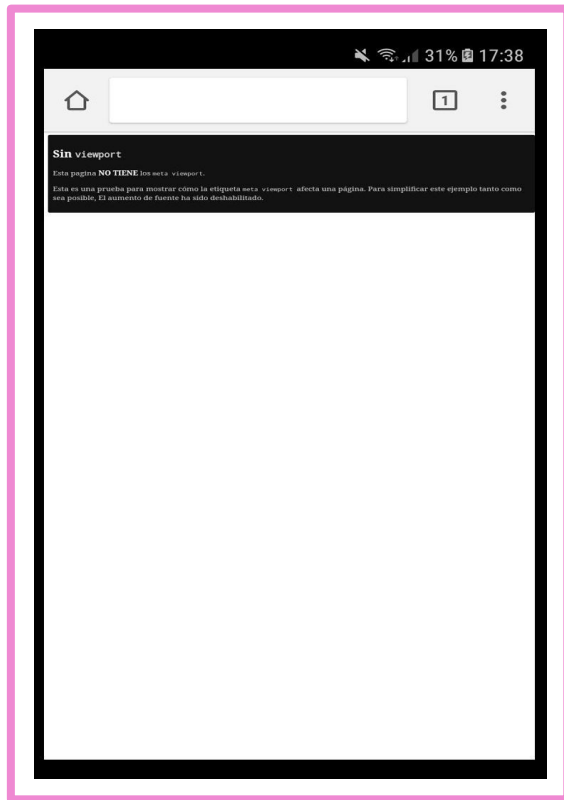
- Usa la etiqueta `<meta> viewport` para controlar el ancho y el ajuste de la ventana de visualización del navegador.
- Incluye `width=device-width` para hacer coincidir el ancho de la pantalla en píxeles independientes del dispositivo.
- Usa `initial-scale=1` para establecer una relación de 1:1 entre los píxeles CSS y los píxeles independientes del dispositivo.

META VIEWPOINT

El uso del valor de *width=device-width* indica a la página que debe hacer coincidir el ancho de la pantalla en píxeles independientes del dispositivo. Esto permite que la página realice el reprocesamiento del contenido para adaptarlo a diferentes tamaños de pantalla.

```
<meta name="viewport" content="width=device-width, initial-scale=1.0"/>
```

META VIEWPOINT



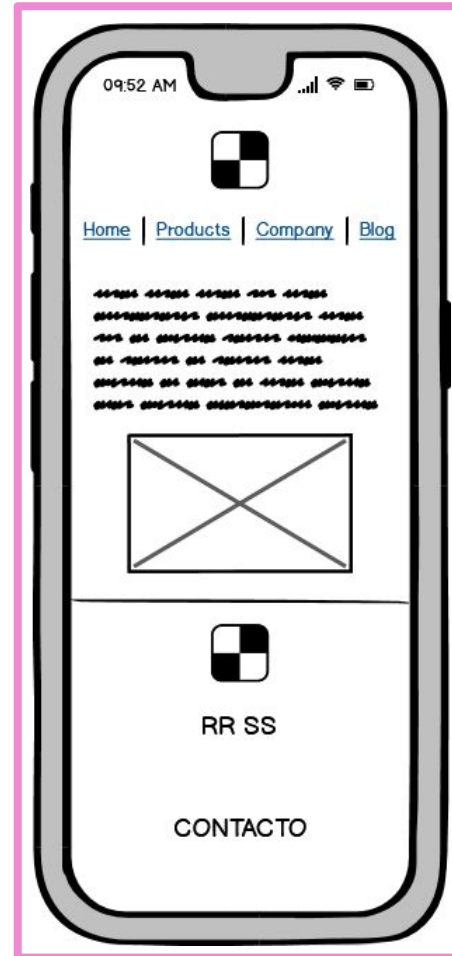
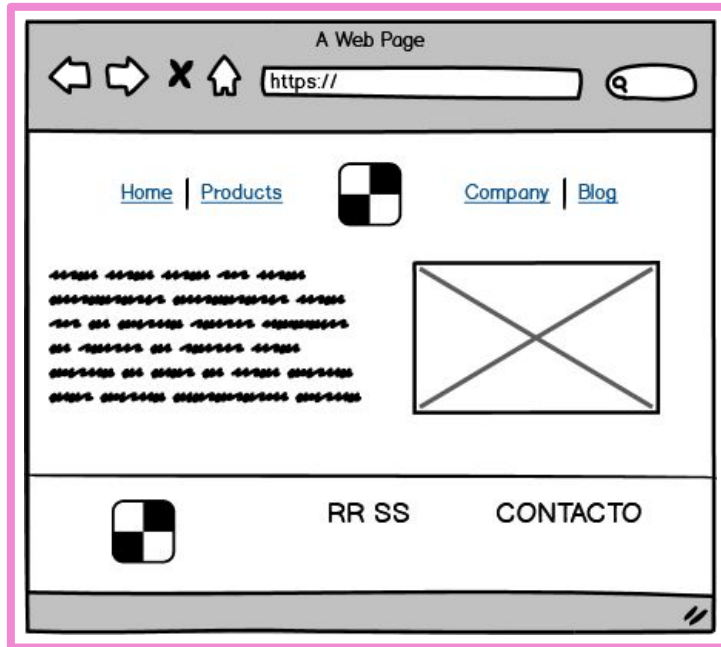
GRIDS + FLEX + @MEDIA

Vamos a combinar todo...

La idea es generar un diseño responsive haciendo uso de grids para el layout + flexbox para los componentes. Para eso, vamos a tener como referencia los siguientes diseños para vista mobile y desktop.



WIREFRAMES





BREAK

¡5/10 MINUTOS Y VOLVEMOS!



¿PREGUNTAS?

#CoderTip: Ingresa al [siguiente link](#) y revisa el material interactivo que preparamos sobre **Preguntas Frecuentes**, estamos seguros de que allí encontrarás algunas respuestas.

Ejemplo
en vivo



¡VAMOS A PRACTICAR LO VISTO!



Vamos a combinar todo...

HTML

```
<div class="gridContainer">
```

```
<header>
```

```

```

```
</header>
```

```
<div class="navList">
```

```
<ul>
```

```
<li><a href="#">Home</a></li>
```

```
<li><a href="#">About Us</a></li>
```

```
</ul>
```

```
</div>
```

```
<div class="navDes">
```

```
<ul>
```

```
<li><a href="#">Services</a></li>
```

```
<li><a href="#">Contact</a></li>
```

```
</ul>
```

```
</div>
```

```
<section class="parrafo">
```

```
<p>Lorem </p>
```

```
</section>
```

```
<section class="imgParrafo">
```

```

```

```
</section>
```

```
<footer>
```

```

```

```
<h3>REDES SOCIALES</h3>
```

```
<h3>CONTACTO</h3>
```

```
</footer>
```

```
</div>
```



Vamos a combinar todo...

CSS

```
/* css general */

*{margin: 0;padding: 0;}

/* vista mobile */

@media only screen and (min-width: 0px) and (max-width: 767px){

    .gridContainer{

        display: grid; grid-template-areas:

            "navlzq navlzq header header navDer navDer" "parrafo parrafo parrafo img img""footer
            footer footer footer footer"; grid-template-columns: 16.6% 16.6% 16.6% 16.6% 16.6% 16.6%;grid-template-rows: 100px auto 200px; }

        header{grid-area:header; text-align: center;}

        div.navlzq{grid-area: navlzq; }.navlzq ul, .navDer ul {display: flex; justify-content: space-around;}

        li{padding: 10px; list-style-type: none; position: relative; top: 20px;}

        div.navDer{grid-area: navDer;}

        section.parrafo{grid-area: parrafo;}

        section.imgParrafo{grid-area: img;}

        footer{grid-area: footer;display: flex; justify-content: space-around;}

    }

    header{grid-area:header;display: flex; justify-content: center;} div.navlzq{grid-area: navlzq; }.navlzq ul, .navDer ul {display: flex; justify-content:
    space-around;} li{padding: 10px; list-style-type: none;} div.navDer{grid-area: navDer;} section.parrafo{grid-area: parrafo;} section.imgParrafo{grid-area:
    img;} footer{grid-area: footer;display: flex; justify-content: space-around;padding-top: 20px; }

}
```

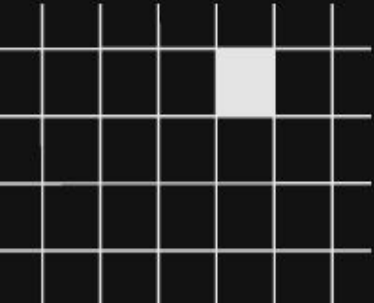

¿PREGUNTAS?





¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Grids + Flexbox + @media.
 - Nuevas formas de modelar Grids.
- 



OPINA Y VALORA ESTA CLASE

#DEMOCRATIZANDO LA EDUCACIÓN

CODER HOUSE