

Bootcamp

Globant + Pioneras

Clase #5 - Abril 16





Itinerario

5:30pm Daily

5:45pm CSS

6:50pm Break

7:00pm Más CSS



Daily



Parte I

CSS



```
.class {  
    number: 5;  
}
```



CSS Styling Links

`:link`, link normal sin visitar

`:visited`, link que ha sido visitado

`:hover`, link con el mouse sobre él

`:active`, link al momento de ser clickeado

```
/* unvisited link */
a:link {
  color: red;
}

/* visited link */
a:visited {
  color: green;
}

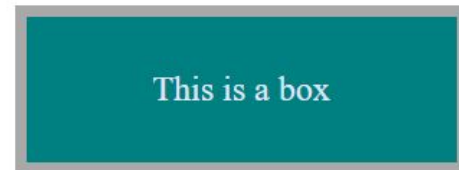
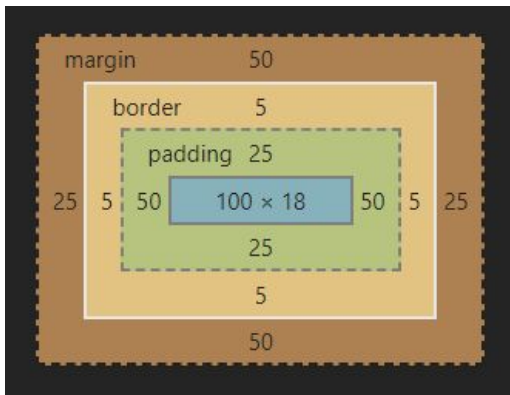
/* mouse over link */
a:hover {
  color: hotpink;
}

/* selected link */
a:active {
  color: blue;
}
```

Modelo de caja - Borders

Los *borders* hacen referencia, como su nombre lo indica, a los bordes que rodean a la caja (elemento HTML), estos bordes se encuentran entre el relleno (*padding*) y los márgenes (*margins*)

```
div {  
  color: ■ lavender;  
  background: ■ teal;  
  width: 100px;  
  padding: 25px 50px;  
  text-align: center;  
  margin: 50px 25px;  
  border: 5px solid ■ darkgray;  
}
```





Display property

Esta propiedad especifica el tipo de presentación (rendering) de un elemento HTML.

- **inline**: Renderiza el elemento en línea, *height* y *width* no tienen efecto en este elemento.
- **block**: Renderiza el elemento como un bloque, empieza en una nueva línea y ocupa todo el espacio posible.
- **flex**: Igual que *block* pero adicionalmente convierte al elemento en un contenedor flex.
- **table**: Hace que el elemento se comporte como una tabla `<table>`.
- **table-row**: Hace que el elemento se comporte como una fila de una tabla `<tr>`.
- **table-column**: Hace que el elemento se comporte como una columna de una tabla `<col>`.
- **table-cell**: Hace que el elemento se comporte como una celda de una tabla `<td>`.
- **inline-block**: Igual que *block* pero en línea.
- **inline-flex**: Igual que *flex* pero en línea.
- **inline-table**: Igual que *table* pero en línea.
- **none**: Elimina completamente al elemento.



Posicionamiento en CSS

1. **Static**: Es la posición por defecto de un elemento HTML, este tipo de posicionamiento significa que el elemento será estático y no se verá afectado por las propiedades *top*, *right*, *bottom*, y/o *left*.
2. **Relative**: El elemento se posiciona relativo a su posición inicial, usar las propiedades *top*, *right*, *bottom*, y/o *left* ajustan al elemento desde su posición inicial. El resto del contenido no se ajusta.
3. **Fixed**: Este posicionamiento ajusta al elemento respecto a la ventana, este elemento queda fijo en la posición aún después de hacer scroll, las propiedades *top*, *right*, *bottom*, y/o *left* son usadas para posicionar al elemento.
4. **Absolute**: Funciona igual que **fixed** pero el elemento se posiciona relativo a su elemento ancestro posicionado más cercano, no a la ventana. Un elemento posicionado es un elemento que tiene cualquier tipo de posicionamiento excepto **static**



Break

Parte II



CSS



```
.class {  
    part: 2;  
}
```

z-index

Esta propiedad define la posición del elemento en el eje z (eje que “sale” de la pantalla”), es decir, en casos de superposición de elementos, la propiedad *z-index* define la prioridad de cada elemento en la pantalla, siendo a mayor valor el elemento que se sobrepone a otros elementos de menor valor.

```
.back {  
  z-index: 0;  
  background:  red;  
}  
  
.front {  
  z-index: 1;  
  background:  blue;  
}
```

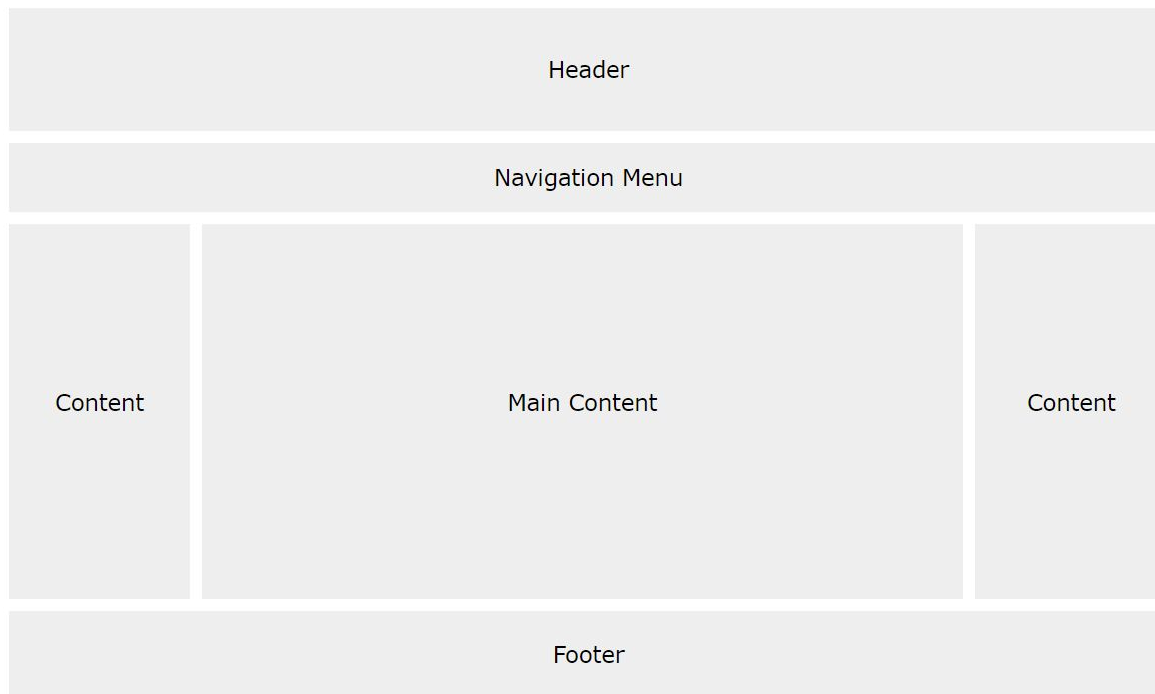




Layout (1/2)

Se entiende como layout o diseño la forma en cómo está construido un website, es decir, cómo se posicionan los elementos, el tipo de elementos que tiene, y el orden de estos.

Si bien existen layouts base para empezar a diseñar, no es obligatorio seguir ninguno, queda a la imaginación y creatividad de cada diseñador hacer diferentes layouts según se ajuste a gustos y necesidades.





Layout (2/2)

Para crear los layouts con elementos HTML se utilizan modos de layouts, a veces abreviados simplemente como layouts, estos los layouts que existen:

- *Normal “flow”*: Todos los elementos por defecto pertenecen a este modo a menos que se especifique lo contrario, a este “flow” pertenecen el *block layout* y el *inline layout*.
- *Table layout*: Diseñado para el funcionamiento de las tablas.
- *Float layout*: Diseñado para que un elemento se posicione a sí mismo a la derecha o izquierda con el resto de elementos que lo rodean.
- *Positioned layout*: Diseñador para posicionar elementos que no requieren mucha o ninguna interacción con otros elementos.
- *Flexible box layout*: Diseñado para crear diseños complejos que se acomodan y redimensionan suavemente sin necesidad de código complejo.
- *Grid layout*: Diseñado para modularizar tu página en una cuadrícula fija y colocar tus elementos relativos a esta.



Flexbox (1/2)

Este layout nos permite organizar nuestros elementos de manera sencilla y que estos se acomoden cuando sea necesario de manera automática y suave.

Las propiedades más importantes de flexbox son:

- **display: flex:** Define un contenedor flex, es decir que crea un contexto flex para sus hijos.
- **flex-direction: row | row-reverse | column | column-reverse:** Define el eje principal en el cual se acomodan los elementos.
- **flex-wrap: nowrap | wrap | wrap-reverse:** Define cómo se deben envolver los elementos cuando no haya más espacio horizontal para acomodarlos.



Flexbox (2/2)

- **align-items:** stretch | flex-start | flex-end | center | baseline | first baseline | last baseline | start | end | self-start | self-end: Define cómo organizar verticalmente los elementos.
- **align-content:** flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | start | end | baseline | first baseline | last baseline: Define cómo organizar verticalmente múltiples líneas de elementos, esta propiedad no tiene efecto si sólo existe una línea de elementos.



Responsive CSS

Se le considera responsive CSS a las propiedades que este lenguaje nos provee para modificar los diseños y propiedades de los elementos HTML basados en los tipos de pantalla, tipos de dispositivos, y diferentes tamaños de pantallas.

Se utiliza la regla `@media` para definir el espacio dónde harán efecto las propiedades CSS que definamos, este espacio puede ser muchas cosas: Un rango de ancho de pantalla, un tipo de dispositivo, una característica de la pantalla, entre muchas otras opciones.

```
@media screen and (max-width: 600px) {  
  div.example {  
    display: none;  
  }  
}
```

```
@media not print and (orientation: landscape) {  
  div {  
    background: ■rgb(100, 200, 100);  
  }  
}
```




Ejercicio!

Crear la página de una revista imaginaria utilizando los conceptos aprendidos hoy.

Requerimientos:

1. Utilizar diferentes formas de estilos para los links.
2. Utilizar diferentes valores de la propiedad `display`.
3. Utilizar diferentes valores de la propiedad `position`.
4. Utilizar diferentes valores de la propiedad `z-index`.
5. Utilizar el flexbox layout para construir varias secciones de tu website.
6. La página debe contener diferentes media-queries para verse bien en diferentes tamaños de pantalla y dispositivos variados.



¿Preguntas?

Feedback



<https://forms.gle/mGzEToMYzC2n2i6c9>

A wide-angle landscape photograph of a mountain range. In the foreground, a grassy ridge slopes down from the left. A small, bright red cabin with a white roof and a small white porch is situated on the ridge. To the left of the cabin is a small white structure. A dirt path leads up to the cabin. In the background, a vast mountain range stretches across the horizon under a cloudy sky. A river is visible in the valley to the right.

Gracias

... Y cuídense mucho!