

«Talento Tech»

Front-End JS

Clase 06



Clase 06 | CSS3 - Modelo de caja, posicionamiento y Flexbox

Temario:

1. Modelado de caja y propiedades

- Concepto del Modelo de Caja
- Contenido, Relleno, Borde y Margen
- Propiedades Clave: width, height, padding, border, margin

2. Posicionamiento y Visualización

- Tipos de Posicionamiento
 - Static
 - Relative
 - Absolute
 - Fixed
 - Sticky

3. Selectores Avanzados

- Selector de Descendientes
- Selector de Hijos Directos
- Selector de Hermano Adyacente
- Selector General de Hermanos

4. ¿Qué es Flexbox?

- Conceptos Básicos de Flexbox
- Contenedor Flex y Ejes

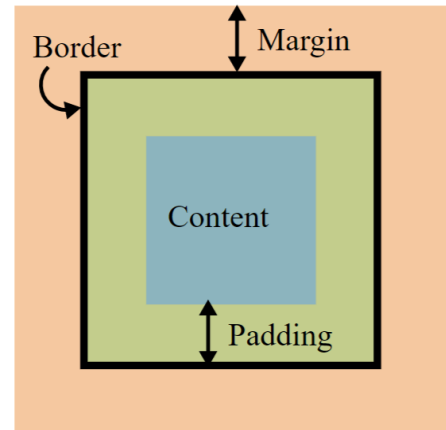
5. Propiedades para el Contenedor Flex y los Flex Items

- Propiedades para el Contenedor
 - flex-direction
 - flex-wrap
 - justify-content
 - align-items
- Propiedades para los Flex Items
 - flex-grow
 - flex-shrink
 - flex-basis

1. Modelo de caja y propiedades

El **modelo de caja** es una de las características más importantes de CSS, ya que determina cómo se estructuran y se muestran los elementos en una página web. Cada elemento de HTML que hemos aprendido es representado por una caja que incluye:

- **Contenido (content):** El área donde se muestra el contenido, como texto, imágenes, etc.
- **Relleno (padding):** Espacio entre el contenido y el borde. Es transparente y permite que el fondo del contenido se muestre.
- **Borde (border):** El límite que rodea el relleno y el contenido. Puede ser estilizado con diferentes colores, grosores y estilos.
- **Margen (margin):** Espacio exterior al borde. También es transparente y permite separar un elemento de otro.



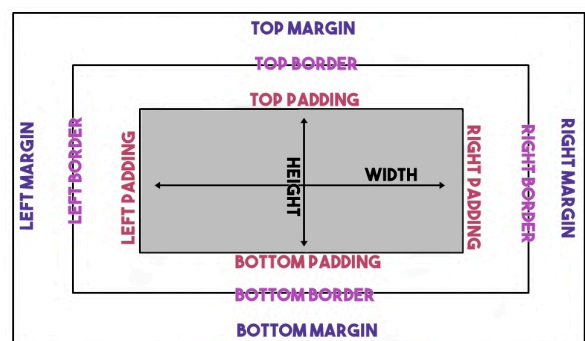
Ejemplo:

```
.box {  
  width: 300px;  
  height: 150px;  
  padding: 10px;  
  border: 2px solid #333;  
  margin: 20px;  
}
```

En este ejemplo, `.box` es un contenedor con un área de contenido de 300x150 píxeles. Tiene un relleno de 10 píxeles, un borde sólido de 2 píxeles y un margen de 20 píxeles alrededor.

Propiedades clave del modelo de caja:

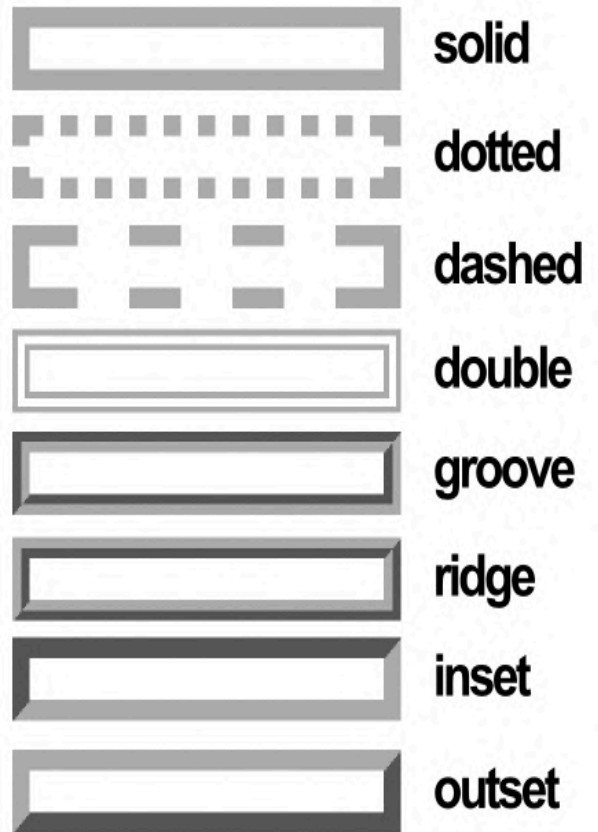
- **width y height:** Definen el ancho y alto del contenido.
- **padding:** Controla el espacio entre el contenido y el borde.
- **border:** Define el borde alrededor del relleno.
- **margin:** Establece el espacio exterior alrededor del borde.



Nota: Los márgenes y rellenos pueden colapsar bajo ciertas circunstancias, especialmente en elementos adyacentes, lo que puede afectar la disposición visual.

Border:

- **hidden:** Oculto. Idéntico a none, salvo para conflictos con tablas.
- **dotted:** Borde basado en puntos.
- **dashed:** Borde basado en rayas (línea discontinua).
- **solid:** Borde sólido (línea continua).
- **double:** Borde doble (dos líneas continuas).
- **groove:** Borde biselado con luz desde arriba.
- **ridge:** Borde biselado con luz desde abajo. Opuesto a groove.
- **inset:** Borde con profundidad «hacia dentro».
- **outset:** Borde con profundidad «hacia fuera». Opuesto a inset.



2. Posicionamiento y visualización

El **Posicionamiento** en CSS es una herramienta poderosa para controlar la ubicación de los elementos en la página. Existen diferentes modos de posicionamiento:

- **static:** Valor por defecto. Los elementos se colocan en su posición natural dentro del flujo del documento.
- **relative:** Permite desplazar un elemento respecto a su posición original sin alterar el flujo del documento.
- **absolute:** Posiciona el elemento relativo a su contenedor posicionado más cercano o al viewport si no hay un contenedor posicionado.
- **fixed:** El elemento se fija en una posición relativa al viewport y no se mueve al hacer scroll.
- **sticky:** Combina características de **relative** y **fixed**. El elemento se comporta como **relative** hasta que alcanza un umbral, momento en que se "pega" y se comporta como **fixed**.

Ejemplo de posicionamiento relativo y absoluto:

```
.relative-box {  
  position: relative;  
  left: 30px;  
  top: 20px;  
  background-color: #f0f0f0;  
  width: 200px;  
  height: 100px;  
}  
  
.absolute-box {  
  position: absolute;  
  right: 10px;  
  bottom: 15px;  
  background-color: #ccc;  
  width: 150px;  
  height: 75px;  
}
```

En este ejemplo, **.relative-box** se desplaza **30px hacia la derecha** y **20px hacia abajo** desde su posición original. **.absolute-box**, en cambio, se coloca **10px desde el borde derecho** y **15px desde el borde inferior** del contenedor posicionado.

3. Selectores avanzados

Los **selectores avanzados** en CSS permiten aplicar estilos a elementos específicos con mayor precisión. Algunos de los selectores avanzados más comunes incluyen:

Selector de descendientes (): Aplica estilos a elementos que están dentro de otro elemento.

```
div p {  
  color: blue;  
}
```

- Aplica el color azul a todos los párrafos dentro de cualquier **div**.

Selector de hijos directos (>): Aplica estilos solo a los elementos que son hijos directos de un contenedor.

```
div > p {  
  color: green;  
}
```

- Aplica el color verde a los párrafos que son hijos directos de un **div**.

Selector de hermano adyacente (+): Aplica estilos al elemento que sigue inmediatamente a otro.

```
h1 + p {  
  font-size: 1.2em;  
}
```

- Aplica un tamaño de fuente diferente al párrafo que sigue inmediatamente después de un `h1`.

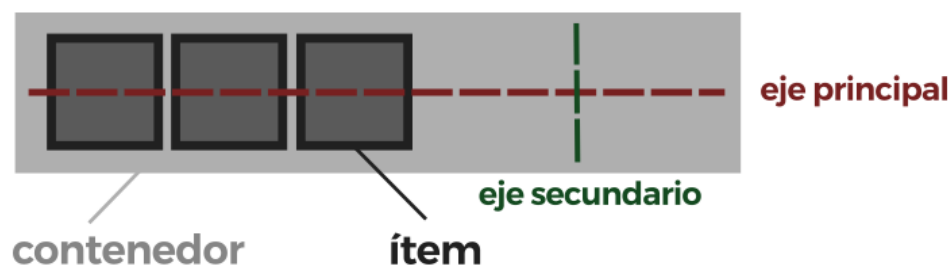
Selector general de hermanos (~): Aplica estilos a todos los elementos que son hermanos de un elemento especificado.

```
h2 ~ p {  
  color: red;  
}
```

- Aplica el color rojo a todos los párrafos que son hermanos de un `h2`.

4. ¿Qué es Flexbox?

Flexbox es un modelo de diseño que facilita la creación de layouts dinámicos y flexibles, especialmente cuando se trata de ajustar elementos dentro de un contenedor. Flexbox es ideal para crear diseños de una sola dimensión, ya sea en un eje horizontal (filas) o vertical (columnas).



Conceptos básicos de Flexbox:

- **Contenedor Flex (`display: flex`):** El elemento padre que contiene los ítems flexibles.
- **Eje principal:** Por defecto, es el eje horizontal donde los ítems se alinean.
- **Eje secundario:** Es el eje perpendicular al principal, utilizado para la alineación secundaria.

Ejemplo de contenedor flex:

```
.container {  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  height: 300px;  
  background-color: #eaeaea;  
}  
  
.item {  
  background-color: #4caf50;  
  color: white;  
  padding: 20px;  
  margin: 10px;  
}
```

Este código crea un contenedor flex que centra los elementos `.item` tanto horizontal como verticalmente.

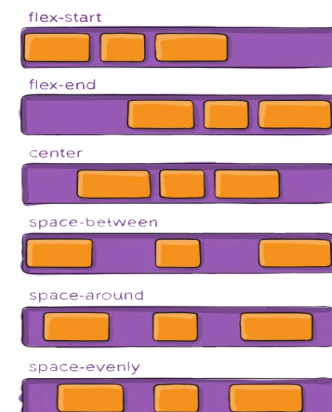
5. Propiedades para el contenedor flex y los flex items

En Flexbox, hay varias propiedades que permiten controlar la disposición de los ítems flexibles dentro del contenedor:

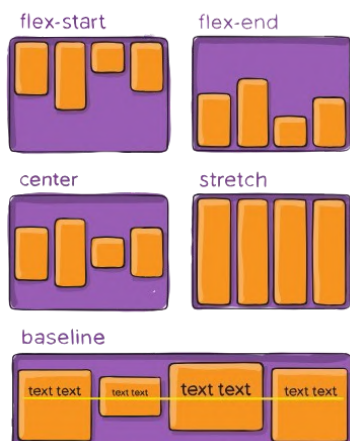
justify-content: Alinea los ítems a lo largo del eje principal.

Ejemplos: `flex-start`, `flex-end`, `center`, `space-between`, `space-around`, `space-evenly`.

justify-content



align-items

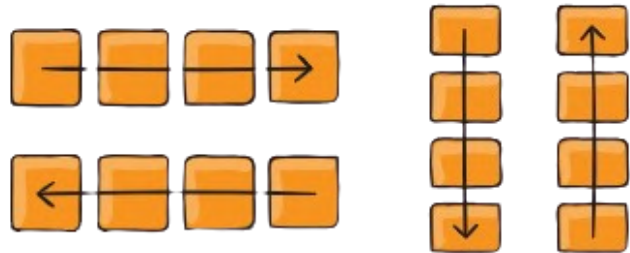


align-items: Alinea los ítems a lo largo del eje secundario.

Ejemplos: `stretch` (por defecto), `flex-start`, `flex-end`, `center`, `baseline`.

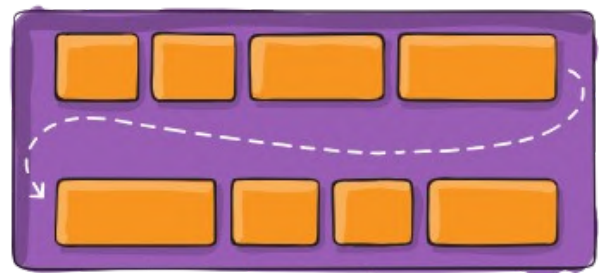
flex-direction

flex-direction: Establece la dirección del eje principal. Valores comunes incluyen **row** (por defecto), **column**, **row-reverse**, y **column-reverse**.



flex-wrap

flex-wrap: Permite a los ítems flexibles pasar a la siguiente línea si no caben en el contenedor. Valores comunes incluyen **nowrap** (por defecto), **wrap**, y **wrap-reverse**.



Propiedades para los ítems flex:

- **flex-grow:** Permite que un ítem crezca para ocupar el espacio disponible. Un valor de **1** significa que el ítem puede crecer para llenar el espacio.
- **flex-shrink:** Permite que un ítem se encoja si es necesario para evitar desbordamiento. Un valor de **0** significa que el ítem no se encogerá.
- **flex-basis:** Establece el tamaño inicial del ítem antes de que el espacio se distribuya. Puede ser un valor fijo (como **px**, **em**, **%**) o **auto**.
-

Ejemplo completo de Flexbox:

```
.container {  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
  justify-content: space-between;  
  align-items: center;  
}
```



```
.item {  
  flex: 1 1 200px;  
  margin: 10px;  
  background-color: #2196F3;  
  color: white;  
  padding: 20px;  
}
```

En este ejemplo, los ítems `.item` se distribuyen en filas, se envuelven cuando es necesario, y se alinean al centro del eje secundario. Cada ítem ocupa un mínimo de 200px y puede crecer si hay espacio disponible.

Ejercicio práctico #1:

Modelado de caja

Objetivo: Definir las propiedades del modelo de caja para los elementos principales (`header`, `main`, `section`, `footer`). Ajustar los márgenes, rellenos y bordes de estos elementos.

Ejercicio práctico #2

Sección productos con Flexbox

Objetivo: Crear una disposición de productos en la sección `Productos` utilizando Flexbox. Asegúrate de que los productos se alineen correctamente y se adapten a diferentes tamaños de pantalla.

A large, stylized wireframe dome structure, resembling a geodesic dome, is positioned on the left side of the page. It is composed of numerous interconnected lines forming a triangular mesh pattern. The dome is rendered in a light gray color against a dark blue background.

Buenos Aires
aprende
Agencia de Habilidades para el futuro

BA Buenos
Aires
Ciudad