INSTITUTO KHIPU

Semestre IV

Sesión 04

PROGRAMA DE ESTUDIOS

DESARROLLO DE SISTEMAS DE INFORMACIÓN

COMUNITY MANAGER Y MARKETING DIGITAL

Tema

MANEJO DEL MODELO DE CAJA Y POSICIONAMIENTO EN CSS

Manejo del Modelo de Caja y Posicionamiento en CSS

1. Explicación del Modelo de Caja en CSS

En CSS, todos los elementos HTML son representados como cajas rectangulares. El **Modelo de Caja** define cómo se distribuye el espacio dentro y alrededor de los elementos en una página web. Se compone de cuatro partes principales:

1.1. Partes del Modelo de Caja

- **Contenido (content)**: Es el área donde se muestra el texto, imágenes u otros elementos internos del elemento. Su tamaño está determinado por las propiedades `width` y `height`.
- **Relleno (padding)**: Es el espacio entre el contenido y el borde del elemento. Se controla con la propiedad `padding`, que puede tener valores individuales para cada lado o un solo valor para todos.

```
div {
    padding: 20px; /* 20px de espacio interno en todos los lados */
}
```

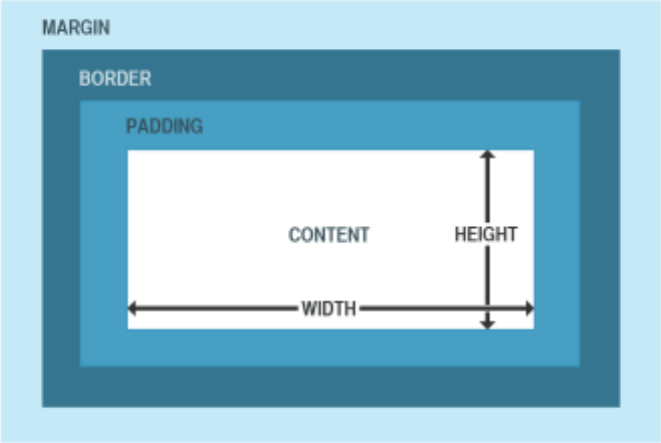
- **Borde (border)**: Es el contorno del elemento. Se define con `border-width`, `border-style` y `border-color`.

```
div {
    border: 2px solid black; /* Borde de 2px, sólido y negro */
}
```

- **Margen (margin)**: Es el espacio externo que separa el elemento de otros elementos adyacentes.

```
div {
    margin: 15px; /* 15px de separación en todos los lados */
}
```

1.2. Ilustración del Modelo de Caja



2. Uso de `box-sizing: border-box`

Por defecto, el tamaño de un elemento (`width` y `height`) **no incluye** el `padding` ni el `border`, lo que puede causar problemas en el diseño. Para evitarlo, se usa `box-sizing: border-box`, que ajusta el tamaño total del elemento incluyendo `border` y `padding`.

Ejemplo sin `box-sizing: border-box` (comportamiento por defecto)

```
div {
    width: 200px;
    padding: 20px;
    border: 5px solid black;
}
```

🔷 **Tamaño real** = `width` (200px) + `padding` (20px * 2) + `border` (5px * 2) = **250px**

Ejemplo con `box-sizing: border-box` (corrige el tamaño)

```
div {
    width: 200px;
    padding: 20px;
```

```
    border:                 5px                 solid                 black;
    box-sizing:                                                   border-box;
}
```

🔷 **Tamaño real** = **200px** (el `padding` y el `border` están incluidos dentro del ancho).

✔ **Recomendación**: Para evitar inconsistencias, es una buena práctica aplicar `box-sizing: border-box` globalmente:

```
*                                                                          {
    box-sizing:                                                   border-box;
}
```

3. Tipos de Posicionamiento en CSS

CSS proporciona varios métodos para controlar la ubicación de los elementos en la página mediante la propiedad `position`.

3.1. `static` (posición por defecto)

Es el valor predeterminado de todos los elementos. No permite desplazar el elemento con `top`, `right`, `bottom`, o `left`.

```
div                                                                        {
    position:                                                     static;
}
```

📌 **Ejemplo:**

```
<div style="position: static; background: lightblue;">Elemento con position: static</div>
```

3.2. `relative` (posición relativa a su posición original)

Permite mover un elemento desde su posición original sin afectar otros elementos.

```
div {
    position: relative;
    top: 20px; /* Baja el elemento 20px desde su posición original */
    left: 10px; /* Mueve el elemento 10px a la derecha */
}
```

📌 **Ejemplo:**

```
<div style="position: relative; top: 20px; left: 10px; background: lightcoral;">
    Elemento con position: relative
</div>
```

3.3. absolute (posición en relación con el elemento contenedor más cercano)

El elemento se mueve en función del primer ancestro con `position: relative, absolute` o `fixed`. Si no hay un contenedor posicionado, se mueve respecto al body.

```
div {
    position: absolute;
    top: 50px;
    left: 50px;
}
```

📌 **Ejemplo:**

```
<div style="position: relative; width: 200px; height: 200px; background: lightgray;">
    <div style="position: absolute; top: 50px; left: 50px; background: lightgreen;">
        Elemento con position: absolute
    </div>
</div>
```

3.4. `fixed` (fijo en la pantalla, sin importar el desplazamiento)

El elemento permanece fijo en la pantalla aunque la página se desplace.

```
div {
    position: fixed;
    top: 10px;
    right: 10px;
    background: yellow;
}
```

📌 **Ejemplo:**

```
<div style="position: fixed; top: 10px; right: 10px; background: yellow;">
    Barra fija en la esquina
</div>
```

3.5. `sticky` (se mantiene visible mientras se desplaza la página)

El elemento se comporta como `relative` hasta que el usuario lo desplaza más allá de un umbral definido, momento en el cual se convierte en `fixed`.

```
div {
    position: sticky;
    top: 0;
    background: lightblue;
}
```

📌 **Ejemplo:**

```
<div style="position: sticky; top: 0; background: lightblue; padding: 10px;">
    Encabezado Sticky
</div>
```

Conclusión

🔷 **El modelo de caja en CSS** define cómo se distribuyen los elementos en la página, y el uso de `box-sizing: border-box` ayuda a gestionar el tamaño sin complicaciones.

🔷 **Los diferentes tipos de `position`** permiten posicionar elementos de manera flexible en la interfaz:

- `static`: Por defecto.
- `relative`: Se mueve respecto a su posición original.
- `absolute`: Se posiciona en relación con su contenedor más cercano con `position` definido.
- `fixed`: Se mantiene fijo en la pantalla.
- `sticky`: Se mantiene visible hasta cierto punto del desplazamiento.

Recuerda: **Dominar estos conceptos es clave para estructurar diseños web efectivos y responsivos.**

INSTITUTO

KHIPU