**Podemos representar un color de 3 formas dentro de CSS:**

1. Palabra clave: red, blue, pink, etc.
2. Hexadecimales: 0123456789ABCDEF
3. hls() hlsa()

* #000000 = Negro
* #FFFFFF = Blanco

**Se representan por grupos de 2 ->** Red, Green, Blue

* #FF0000 = Rojo
* #00FF00 = Verde
* #0000FF = Azul

**Para obtener un color más exacto se utiliza rgb o rgba:**

* rgb(0, 200, 145)
* rgba(0, 50, 70, 0.5) para obtener transparencia

**¿Qué es un pixel?**  
La menor unidad homogenea en color que forma parte de una imagen digital.

Las etiquetas son la representación básica de la información en un documento html. Sirven para crear y organizar el contenido.

La sintaxis general de una etiqueta es:

<nombre> contenido </nombre>

Hay ciertas etiquetas que tienen una sintaxis diferente, ya que se cierran en sí mismas; es decir, no tienen etiqueta de cierre:

<nombre />

Algunas de las etiquetas más conocidas y usadas son:

Etiquetas de cabecera (head)

* doctype: indica al navegador el tipo de documento que se está mostrando.
* html: es la etiqueta que envuelve todo el documento
* head: es la cabecera del documento y contiene sub etiquetas que describen al documento o incluyen recursos adicionales.
* Etiquetas del cuerpo del documento (body):
* article: diferencia partes del contenido que pueden vivir por sí mismas.
* nav: para hacer menús de navegación.
* aside: contenido menos relevante, como publicidad, etc.
* section: sirve para diferenciar las secciones principales del contenido.
* header: cabecera del documento.
* footer: pie de página del documento.
* h1 - h6: títulos de nuestro sitio web.
* table: tablas de contenidos, similar a la estructura de las hojas de calculo.
* ul y ol: listas de items.
* div: cualquier división para organizar el contenido.

**Margin y Padding**

Diferencias de las propiedades padding y margin:

* Margin es el margen que hay desde un elemento hasta los que tenga al lado.
* Padding es el espacio que hay en un contenedor entre el contenido y los bordes del contenedor.

Así como el margen separa a los elementos html entre sí, la propiedad padding de relleno, permite definir una separación entre el contenido interno y el borde de un elemento. Al inspeccionar los elementos html en el navegador, se puede apreciar el margin con color naranja y el padding con color verde.

**Modelo de Caja**

El modelo de caja es un concepto teórico de css que representa a cada elemento html en base sus propiedades de: **margin**, **border**, **padding** y **dimensiones** (alto y ancho).  
Para visualizar un elemento html en su representación como modelo de caja debemos irnos a la parte baja de la sección styles del inspector de elementos, o en la sección llamada **Computed**.

En el modelo de caja, el **ancho total** de un elemento html equivale a la sumatoria de los valores de: **width**, **padding-left**, **padding-right**, **border-left-width**, **border-right-width**. De manera similar aplica para el **alto total** de cada elemento. Aunque **margin-left** y **margin-right**, forman parte del modelo de caja, no se incluyen para el calculo del ancho total.

Con la propiedad **box-sizing**, y en particular con el valor **border-box** de esta propiedad, podemos modificar el comportamiento del modelo de caja para que nuestro elemento nunca supere el tamaño máximo que le hayamos definido en **width** y **height**. Esta es la opción recomendad para trabajar.

**Tipos de Display**

Display es la propiedad de css que indica cómo debe ser mostrado un elemento html. Todos los elementos tienen algún tipo de display. Si un elemento no se ve en pantalla es porque segurmanete su display es none.

Los valores más comunes que puede recibir la propiedad **display** son:

* **block**: el elemento intenta abarcar todo el ancho posible.
* **inline**: reduce su tamaño exclusivamente hasta lo que abarca su contenido, descartando las propiedades width y height.
* **inline-block**: combina lo mejor de block e inline, ya que respeta las dimensiones indicadas en las propiedades width y height, pero coloca el elemento en línea (al costado) de elementos hermanos que también tengan display: inline o inline-block.
* **flex**: asume algunas propiedades por defecto que favorecen la alineación de los elementos internos.
* **grid**: similar a flex, asume algunas propiedades por defecto organizando los contenidos en filas y columnas.
* **none**: oculta el elemento.

Flexbox se refiere al tipo de display en css que permite un manejo flexible de la alineación, dimensionamiento y distribución de elementos html.

Esta propiedad se aplica a un elemento padre, pero va a afectar principalmente a sus elementos hijos directos. Por defecto, los elementos internos quedan alineados unos seguidos de los otros. El comportamiento del modelo de caja de estos elementos hijos también se ha modificado, ya que pierden el efecto de su propiedad margin.

Los elementos hijos de un padre con propiedad **display: flex** tienen a su disposición algunas nuevas propiedades que aportan mayor flexibilidad a su comportamiento. Una de estas propiedades es **flex-shrink** que, junto a la propiedad **flex-wrap** del padre, permite adaptar y distribuir los elementos de manera dinámica en el espacio horizontal disponible hasta ocupar todo el espacio, y luego pasar a ocupar dinamicamente las siguiente filas hacia abajo.

**Alineando elementos de forma horizontal**

Flexbox se refiere al tipo de display en css que permite un manejo flexible de la alineación, dimensionamiento y distribución de elementos html.

Esta propiedad se aplica a un elemento padre, pero va a afectar principalmente a sus elementos hijos directos. Por defecto, los elementos internos quedan alineados unos seguidos de los otros. El comportamiento del modelo de caja de estos elementos hijos también se ha modificado, ya que pierden el efecto de su propiedad margin.

Los elementos hijos de un padre con propiedad **display: flex** tienen a su disposición algunas nuevas propiedades que aportan mayor flexibilidad a su comportamiento. Una de estas propiedades es **flex-shrink** que, junto a la propiedad **flex-wrap** del padre, permite adaptar y distribuir los elementos de manera dinámica en el espacio horizontal disponible hasta ocupar todo el espacio, y luego pasar a ocupar dinamicamente las siguientes filas hacia abajo.

**Alineando elementos de forma vertical**

Similar a como sucede con justify-content, es posible alinear y distribuir los elementos internos en el espacio vertical disponible usando la propiedad **align-items**, que puede tomar también los valores de: **flex-start**, **flex-end** y **center**.

Algo que es muy importante y se debe tener en cuenta a la hora de usar **align-items** y **justify-content** es que dependiendo de la propiedad **flex-direction**que se haya definido, el efecto de ambos se invierte, no en cuanto a sus elementos internos, sino en cuanto a si se debe usar uno u otro de manera vertical u horizontal.

IMPORTANTE: Cuando la propiedad **flex-direction** se ha definido como **column**, la propiedad **justify-content** ya no va a aplicar sobre la alineación horizontal, sino sobre la vertical. Y **align-items** ya no aplicaría sobre la alineación vertical sino la horizontal. Se intercambian sus efectos.

Para centrar completamente los elementos internos de manera vertical y horizontal en su elemento padre, debemos usar el valor **center** en ambas propiedades.

