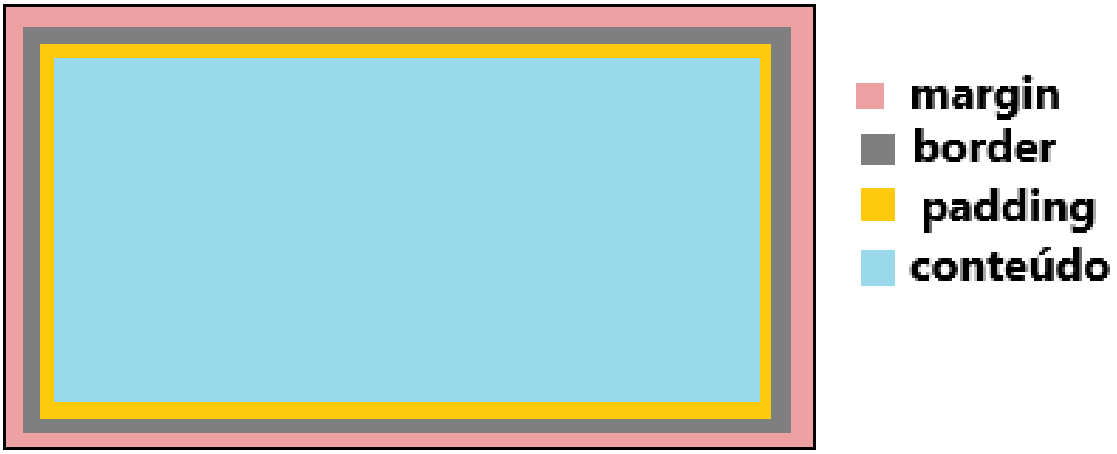

CSS Box-Model:
Space around an element = margin (spacing outside of the element);
Padding = spacing on the the inside of the element;

Display: define como o elemento será exibido

- block: Elemento ocupa toda a largura disponível e começa numa nova linha.
- inline: Elemento ocupa apenas o espaço necessário, sem quebrar para uma nova linha.
- inline-block: . Ocupa apenas o espaço necessário, mas permite definir `width`, `height`, `margin` e `padding`.
- flex: Configura o contêiner como um layout flexível, permitindo que os elementos internos se ajustem de acordo com o espaço disponível.
  -> Isto permite alinhar e distribuir os itens facilmente em filas ou colunas.

flex-direction: Define a direção na qual os elementos flexíveis (itens flex) são organizados dentro do contêiner flexível. Controla se os elementos são exibidos em linha horizontal ou coluna vertical.

- row: organiza horizontalmente, da esquerda para a direita.
- column: organiza verticalmente, de cima para baixo.
- (...-reverse): inverte a ordem

justify-content: Controla o alinhamento horizontal dos itens dentro de um contêiner flexível. Define como os itens são distribuídos ao

longo do eixo principal (dependendo do `flex-direction`, pode ser horizontal ou vertical).

- flex-start: Alinha os itens ao início do contêiner.
- center: Centraliza os itens dentro do contêiner.
- flex-end: Alinha os itens ao final do contêiner.
- space-between: Distribui os itens com espaço igual entre eles.
- space-around: Adiciona espaço ao redor de cada item, de forma equilibrada.

align-itens: Controla o alinhamento vertical dos itens dentro do contêiner flexível.

- flex-start: Alinha os itens no topo do contêiner (ou no início do eixo cruzado).
- center: Centraliza os itens verticalmente.
- flex-end: Alinha os itens ao fundo do contêiner (ou no final do eixo cruzado).
- stretch: Os itens esticam para preencher o contêiner (valor padrão).

flex-wrap: Define se os itens dentro do contêiner flexível devem ou não quebrar para uma nova linha quando o espaço for insuficiente.

- wrap: Itens podem quebrar para uma nova linha quando o espaço for limitado.
- nowrap: Itens ficam numa única linha, mesmo que não caibam (comportamento padrão);

gap: Define o espaçamento entre os itens num contêiner flex ou grid. Sem precisar usar `margin` ou `padding` individualmente em cada item.

Resumo:

- `margin`: Espaçamento externo do elemento.
- `padding`: Espaçamento interno do elemento.
- `display`: Controla como o elemento é exibido (`block`, `inline`, `flex`).
- `flex`: Estilo de layout flexível que permite organizar elementos de forma dinâmica.

- `flex-direction`: Define a direção dos itens dentro de um contêiner flexível.
- `justify-content`: Alinha os itens ao longo do eixo principal (horizontal ou vertical).
- `flex-wrap`: Permite ou não que os itens quebrem para novas linhas.
- `gap`: Define o espaçamento entre os itens flexíveis.
- `align-items`: Controla o alinhamento vertical dos itens dentro do contêiner flexível.

HTML e CSS Basics: Sintax:
<Opening Tag> Conteúdo dentro da tag </Closing Tag>
<TagName = qual tipo de elemento está criando>
<button>Botão</button> -> tag do tipo button

DIV:
div = division (just a box) / container
 can contain other elements inside
 assim como o <p> por padrão é um display: block.
Por ser block, quando é definido a width da
(com block element by default) it take up the entire line in their container (div). It groups all the elements together and keep it within a width area.

Flex:
similar to css grid but it's more flexible
Dentro de uma flex box as divs não se comportam como display-block (configuração padrão) e passam a se comportar mais como inline element (flex-direction: row; por exemplo), vão utilizar o espeço que precisam e não mais a tela inteira
Diferenças:
O layout é mais rigido para grids.
display flex elements take their width along with them if they move around. So it's going to be more flexible