O modelo de caixas (box model)

Tudo no CSS possuí uma caixa ao seu redor, e cada caixa possui quatro áreas: **conteúdo**, **padding**, **border** e **margin**.

Temos vários tipos de caixas, sendo que essas geralmente sem encaixam na categoria de **caixas em bloco** (block boxes) e as **caixas em linha** (inline blocks). O tipo vai definir como a caixa se comporta na página e em relação as outras caixas.

pedro.mateus@unicesumar.edu.br

Caixas Block

As caixas em bloco são caixas que **ocupam toda a largura disponível** e **são empilhadas uma em cima da outra**.

As propriedades **width** e **height** são aplicadas a essas caixas, e elas **podem** ter **margin**, **padding** e **border**.

A menos que sejam alterados manualmente, alguns elementos HTML são caixas em bloco por padrão, como `<div>`, `<p>`, `<h*>`, `<main>`, `<nav>`, `<header>`, `<article>`, `<footer>`, etc.

Caixas Inline

As caixas em linha são caixas que não ocupam toda a largura disponível, são exibidas uma ao lado da outra.

As propriedades **width** e **height** não são aplicadas a essas caixas, **margin** e padding são aplicadas apenas horizontalmente.

Elementos HTML como `<span>`, `<a>`, `<strong>`, `<em>`, `<small>`, `<button>`, etc, são caixas **inline** por padrão.

Tipos de exibição outer (externos) e inner (internos)

As caixas **inline** e **block** são exemplos de caixas do tipo **outer**. Pois ditam como os outros elementos se comportam em relação a elas.

As caixas também possuem um tipo de exibição **inner**, indicando como os elementos na caixa se comportam. Por padrão os elementos dentro de uma caixa são posicionados em um fluxo normal (normal flow), significando que se comportam como caixas em bloco ou em linha.

É possível alterar o tipo de exibição inner utilizando a propriedade **display**.

Como por exemplo, a propriedade **display: flex** transforma a caixa em um **flex container**, o "outer display" continuará se comportando como os tipos "block", mas todos os filhos diretos da caixa se comportarão como **flex items**, seguindo as regras do **flexbox**.

Flexbox e Grid Layout

> Flexbox

Flexbox é um modelo de layout que permite o design de layout responsivo e previsível, sem depender de tamanhos específicos de tela, trabalha de uma maneira unidimensional, ou seja, ele lida com um eixo por vez, seja ele o eixo horizontal ou vertical.

Para adicionar o flex container a um elemento, basta adicionar a propriedade `display: flex`.

Algumas de suas principais propriedades são:

1. **flex-direction**: Define a direção principal do flex container, podendo ser `row` (padrão), `row-reverse`, `column` e `column-reverse`. Essa propriedade também **altera** qual será **o eixo principal**.

2. **flex-wrap**: Define se os itens devem ser quebrados em várias linhas, podendo ser `nowrap` (padrão), `wrap` e `wrap-reverse`.

3. **justify-content**: Define o alinhamento dos itens ao longo do **eixo principal**, podendo ser `flex-start` (padrão), `flex-end`, `center`, `space-between`, `space-around` e `space-evenly`.

4. **align-items**: Define o alinhamento dos itens ao longo do eixo transversal, podendo ser `stretch` (padrão), `flex-start`, `flex-end`, `center` e `baseline`.

5. **align-content**: Define o alinhamento dos itens ao longo do eixo transversal quando há mais de uma linha, podendo ser `stretch` (padrão), `flex-start`, `flex-end`, `center`, `space-between` e `space-around`.

Documentação MDN - Flexbox

CSS Tricks - Complete Guide to Flexbox

Todo container flex terá elementos filhos que se comportarão como **flex items**, e esses elementos também possuem propriedades específicas:

1. **flex-grow**: Define a capacidade de um item de crescer em relação aos outros itens do container. Ex: **Se existem 3 itens, um possui `flex-grow: 1`, e os outros 2 possuem `flex-grow: 2`, o primeiro item ocupará 1/3 do espaço disponível, e os outros 2 ocuparão 2/3.**

2. **flex-shrink**: Define a capacidade de um item de encolher em relação aos outros itens do container. Ex: **Se existem 3 itens, um possui `flex-shrink: 1`, e os outros 2 possuem `flex-shrink: 2`, o primeiro item encolherá 1/3 do espaço disponível, e os outros 2 encolherão 2/3.**

3. **flex-basis**: Define o tamanho inicial de um item.

4. **order**: Define a ordem de um item em relação aos outros itens do container. Ex: **Se existem 3 itens, e o primeiro possui order: 1, o primeiro item será exibido por último.**

5. **align-self**: Define o alinhamento de um item em relação aos outros itens do container. É igual o align-items, mas aplicado (sobrescreve) apenas a um item.

Treina WEB - Guia interativo Flex Items