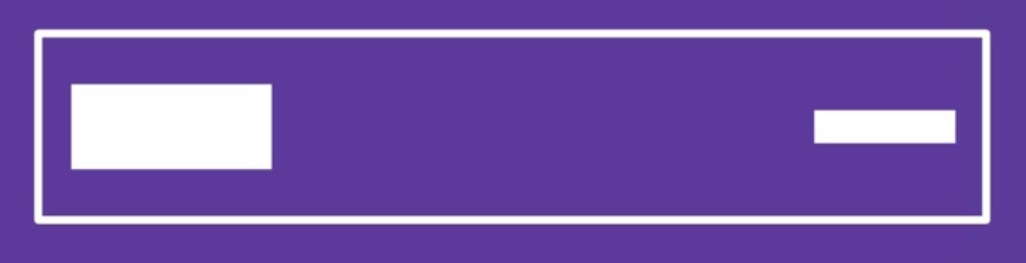
**Flexbox**

Ao longo desse curso aprofundamos nossos conhecimentos em CSS e HTML. Vimos como o CSS é poderoso para trabalharmos com *layouts* dinâmicos e interativos e como o HTML pode ser muito mais semântico. Mesmo assim, ainda existe alguns problemas difíceis de serem solucionados.

O maior desses problemas ainda é o de *layout*. Nós utilizamos de três propriedades para criá-lo:

* *position*
* *display*
* *float*

Mesmo com elas ainda sofremos bastante. Queremos, por exemplo, criar algo desse tipo:



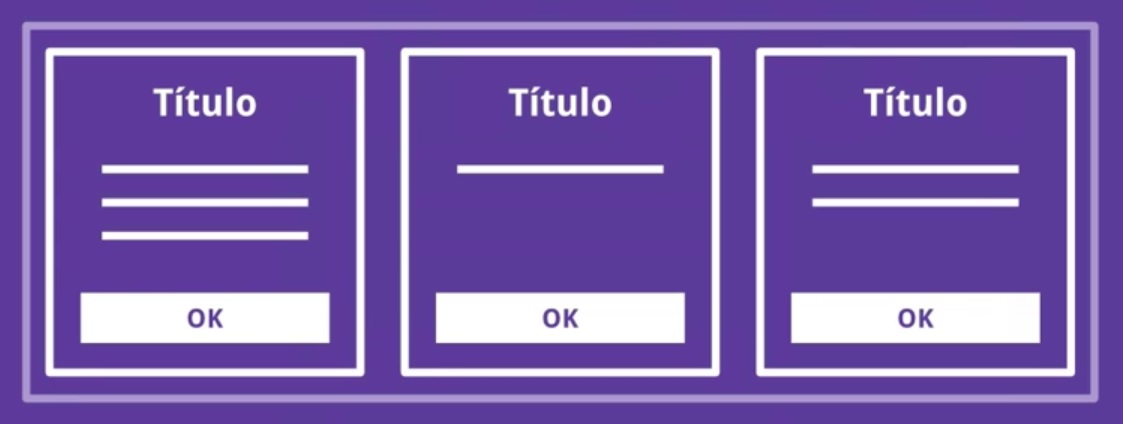
Uma caixa com um elemento ocupando toda sua altura à esquerda e à direita um outro ocupando metade da altura. Ambos devem estar centralizados verticalmente.

Outro exemplo:



Uma caixa com um botão sempre na parte de baixo.

Ou ainda:



Várias caixas de mesma altura lado a lado dentro de um mesmo *container*, independente do conteúdo que elas possuem.

Esses elementos são muito complicados de serem feitos no CSS, pois as regras que vimos até agora servem para controlar os elementos filhos em relação ao pai. Nenhuma delas nos permite controlá-los a partir do pai.

Porém, recentemente, surgiu uma nova especificação para CSS chamada ***flexbox***, a qual permite termos esse controle a partir de um elemento pai.

Pegando nosso primeiro exemplo, podemos resolvê-lo usando três linhas no pai:

.container {

display: flex;

align-items: center;

justify-content: space-between;

}

* "display: flex": faz com que o *container* se comporte como um *flexbox*, ou seja, ele controlará o *layout* de seus filhos. Por padrão, os filhos terão a mesma altura e alinhados à esquerda;
* "align-items": center": centraliza verticalmente os elementos filhos dentro do pai;
* "justify-content: space-between": distribui entre os elementos o espaço que está sobrando.

Perceba que em nenhum momento passamos a informação no código que se tratavam de dois elementos, ou que o espaço entre eles valia alguma porcentagem do total. Isso acontece porque o *flexbox* automaticamente sabe quantos filhos ele terá. É tão dinâmico o *layout* que se colocássemos um terceiro elemento, este já fica alinhado e justificado de acordo:



No outro caso, conseguimos resolver o problema das três caixas fazendo

.container {

display: flex;

}

Isto serve para que as caixas tenham a mesma altura, já que é padrão do *flexbox* fazê-lo. Agora, para manter o conteúdo das caixas justificado verticalmente basta repetir a ideia para elas só que utilizando a propriedade "*flex-direction*":

.box {

display: flex;

flex-direction:column;

}

Porém, as caixas com mais texto ficarão com largura maior, então devemos defini-la usando a noção de proporção. Como as queremos do mesmo tamanho, essa proporção é igual a 1:

.box {

display: flex;

flex-direction:column;

flex: 1;

}

Se quisermos que uma das caixas tenha o dobro de tamanho das duas, por exemplo, a segunda, basta fazermos:

.box:nth-child(2) {

flex: 2;

}



Perceba que se adicionarmos uma quarta caixa, o *flexbox* automaticamente redimensiona todas. O nosso *layout* é muito flexível.

Um outro recurso dessa propriedade é pegar esse elemento que deixamos maior e deixá-lo na frente, sem mexer no HTML:

.box:nth-child(2) {

flex: 2;

order: 1;

}

Isso é muito importante do ponto de vista de acessibilidade e de semântica.

Falta agora deixarmos os botões dentro das caixas sempre em baixo:

.box p {

flex: 1;

}

Porque deixar os botões em baixo nada mais é do que deixar os parágrafos do mesmo tamanho.

Vamos rever alguns recursos do *flexbox*:

* *display: flex*: ativa o *flexbox* no elemento;
* *align-items*: distribui verticalmente os elementos dentro de um *container flex*;
* *justify-content*: distribuir os elementos espaçadamente dentro do *container*;
* *flex-direction*: permite inverter o *align-items*;
* *order*: muda a ordem dos elementos;
* *flex*: indica a proporção do tamanho dos elementos. Ele é um atalho para mais três propriedades:
  + *flex-grow*: determina quanto o elemento deve crescer;
  + *flex-shrink*: determina quanto o elemento deve diminuir;
  + *flex-basis*: determina o tamanho mínimo do elemento.

O *flexbox* é uma dentre várias outras especificações que estão surgindo no CSS para facilitar nosso trabalho com *layout* e deixar o código mais simples.

Primeiro, transformamos a lista de trabalhos num container flex:

.trabalhos ul {

/\* podemos tirar o overflow também! \*/

display: flex;

}

Com isso, agora podemos usar nos elementos da lista a propriedade flex, para garantir que eles cresçam e diminuam automaticamente de acordo com o tamanho da lista e com o número de itens:

.trabalhos li {

/\* não precisamos mais de float nem width! \*/

flex: 1;

}