

Linguagens de Marcação e Scripts

Prof. Aníbal Cavalcante de Oliveira

UFC - QXD0164 - 2019.2

Agenda - Aula 13

- CSS Box-sizing
- Introdução ao design responsivo.
- Flexbox

CSS Box-Sizing

A propriedade `box-sizing` nos permite incluir o preenchimento e a borda na largura e altura totais de um elemento.

Por padrão, a largura e a altura de um elemento é calculado assim:

1 - largura + padding + borda = Largura efetiva de um elemento

2 - altura + padding + borda = Altura efetiva de um elemento

CSS Box-Sizing

Exemplo: As duas `<div>`'s tem a mesma altura e largura, porém a segunda possui padding de 50px.

This div is smaller (width is 300px and height is 100px).

This div is bigger (width is also 300px and height is 100px).

A propriedade `box-sizing` resolve este problema.

CSS Box-Sizing

A propriedade **box-sizing** permite incluir o **padding** e a **borda** na largura e altura total de um elemento.

Se você definir a propriedade CSS **box-sizing: border-box;** o elemento terá sua **borda** e seu **padding** incluídos dentro da caixa, na largura e altura:

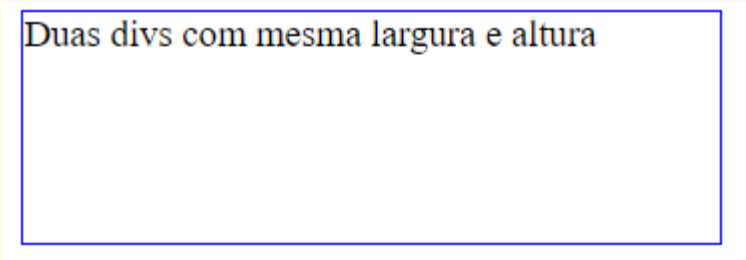
```
.div1 {  
  width: 300px;  
  height: 100px;  
  border: 1px solid blue;  
  box-sizing: border-box;  
}  
  
.div2 {  
  width: 300px;  
  height: 100px;  
  padding: 50px;  
  border: 1px solid red;  
  box-sizing: border-box;  
}
```

CSS Box-Sizing

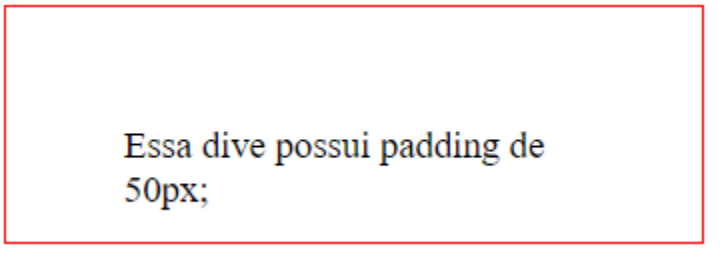
Vantagem:

Segundo o Box Model padrão das CSS, o elemento terá três propriedades somadas para determinar o tamanho final do elemento.

Com o box-sizing não precisamos mais perder tempo fazendo cálculos para determinar o tamanho final dos boxes dos nossos elementos quando declararmos as propriedades padding e border. Seus valores são incorporados ao box.



Duas divs com mesma largura e altura

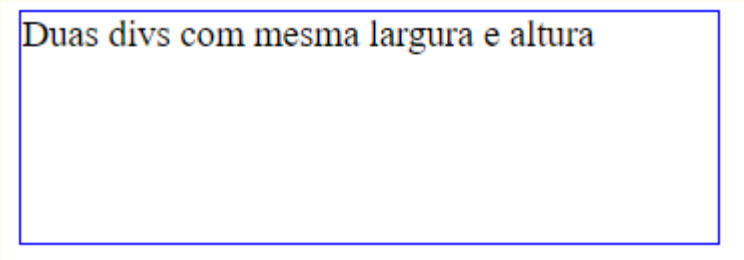


Essa dive possui padding de
50px;

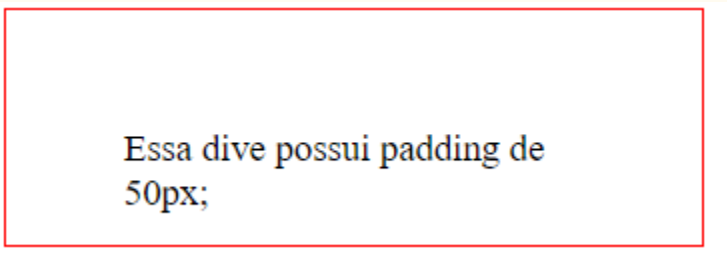
CSS Box-Sizing

A propriedade box-sizing possui 2 valores:

1. **content-box** - é o valor padrão, as propriedades de largura e altura incluem apenas o conteúdo. Borda e preenchimento não serão incluídos.
2. **border-box** - as propriedades de largura e altura incluem: conteúdo, preenchimento e borda.



Duas divs com mesma largura e altura



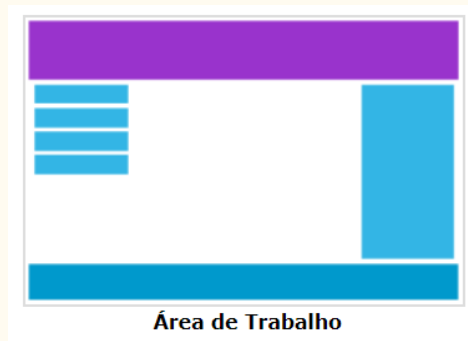
Essa dive possui padding de 50px;

O que é Web Design Responsivo? (Abrir arquivo designResponsivo.html)

O design responsivo permite que sua páginas web seja vista usando dispositivos diferentes: computadores, tablets e telefones.

Projetar de forma responsiva permite que sua interface seja fácil de usar, e forneça uma boa experiencia independentemente do dispositivo.

Sua página não deve deixar de fora informações, mas sim adaptar seu conteúdo para caber em dispositivos com telas menores.



Web Design Responsivo

Existem diversas maneiras de posicionar elementos em CSS, porém para layouts responsivos hoje os trabalhamos com 2 ferramentas:

- **FlexBox** – a primeira solução a surgir, boa para layouts em uma única dimensão.
- **CSS Grid Layouts** – surgiu logo depois, boa para layouts em duas dimensões.

Um erro comum é pensar que o **CSS Grid** veio para ocupar o lugar do **FlexBox**. Porém:

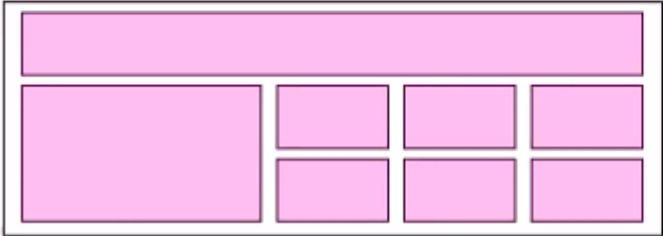
- 1 - **Flexbox** faz coisas que o **CSS Grid** não pode fazer;
- 2 - **CSS Grid** faz coisas que o **Flexbox** não pode;
- 3 - Ambos podem trabalhar em conjunto;
- 4 - Há coisas que podemos fazer com ambos, mas que um faz de maneira melhor e mais simples;



CSS Grid vs Flexbox

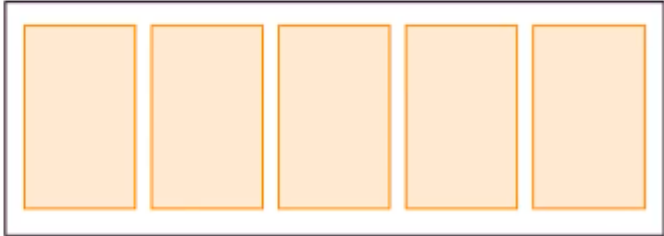
CSS Grid

Two-dimensional Positioning



CSS Flexbox

One-dimensional Positioning



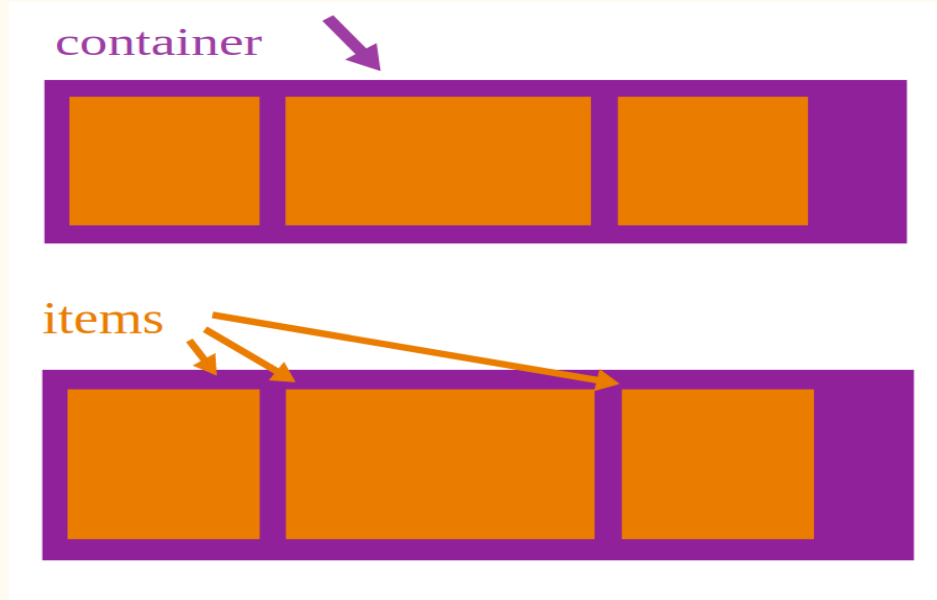
CSS3 Flexbox - CSS Flexible Box Layout

O **Flexbox** é um modelo unidimensional de layout oferece distribuição de espaço e capacidades de alinhamento entre itens num container. O layout flexível permite que os elementos responsivos dentro de um contêiner sejam organizados automaticamente, dependendo do tamanho da tela (ou dispositivo).



CSS Flexbox - CSS Flexible Box Layout

Basicamente existem dois elementos no modelo FlexBox, o **container** e os **itens**.



CSS Flexbox

Para habilitarmos o flexbox basta utilizarmos a propriedade **display: flex;** no container, ou seja na div pai. Nesse momento ele se torna flexível para vários tamanhos de tela. Exemplo: **flexbox01.html**

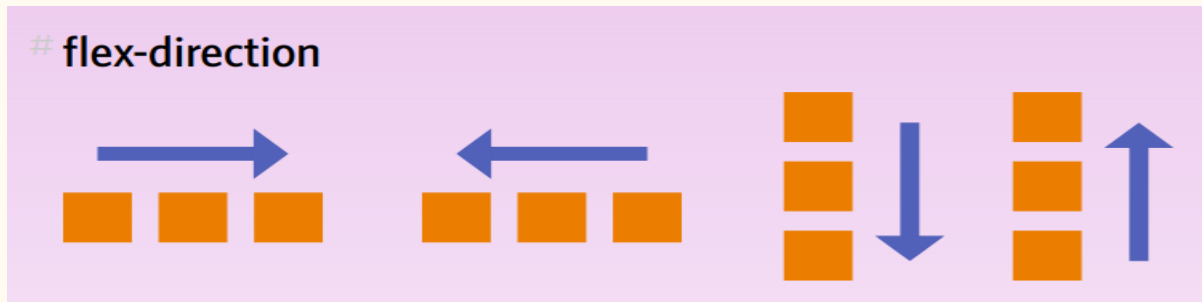
```
<style>
.flex-container {
  display: flex;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  margin: 10px;
  padding: 20px;
  font-size: 30px;
}
</style>
```

CSS Flexbox - A propriedade flex-Direction (flexbox02.html)

A propriedade `flex-direction` em que eixo os itens devem estar dispostos.

Seus valores são: **flex-direction**: `row` | `row-reverse` | `column` | `column-reverse`;



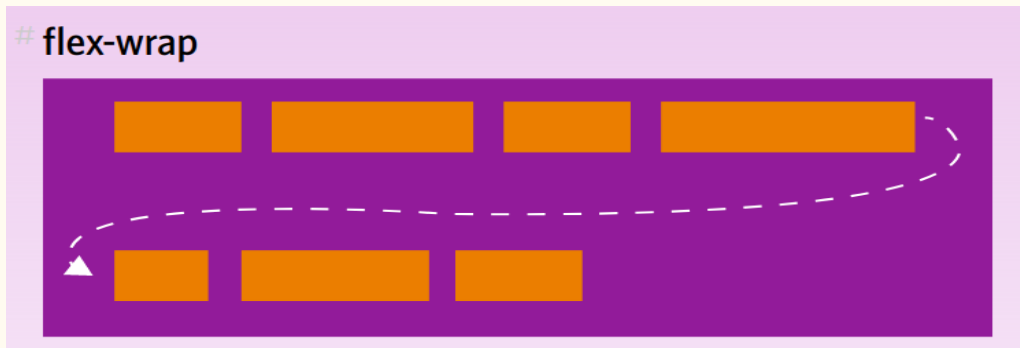
```
<style>
.flex-container {
  display: flex;
  flex-direction: column; /* row | row-reverse | column | column-reverse; */
  background-color: DodgerBlue;
}
</style>
```

CSS Flexbox – A propriedade flex-wrap (flexbox03.html)

Por padrão, o flexbox vai tentar colocar todos os elementos na mesma linha.

Mas é possível mudar esse comportamento com a propriedade:

* **flex-wrap**: nowrap | wrap | wrap-reverse;



```
.flex-container {  
  display: flex;  
  flex-wrap: wrap; /*nowrap | wrap | wrap-reverse*/  
  background-color: DodgerBlue;  
}
```

CSS Flexbox – A propriedade flex-flow (flexbox04.html)

A propriedade flex-flow permite usar as propriedades flex-direction e flex-wrap juntas.

* **flex-flow**: row wrap | column wrap |;

```
.flex-container {  
  display: flex;  
  flex-wrap: wrap; /*nowrap | wrap | wrap-reverse*/  
  background-color: DodgerBlue;  
}
```


CSS Flexbox - A propriedade justify-content (flexbox05.html)

A propriedade alinha os elementos em relação ao eixo horizontal:

* **justify-content:** flex-start | flex-end | center | space-between | space-around | space-evenly;



```
.flex-container {  
  display: flex;  
  justify-content: center;  
}
```

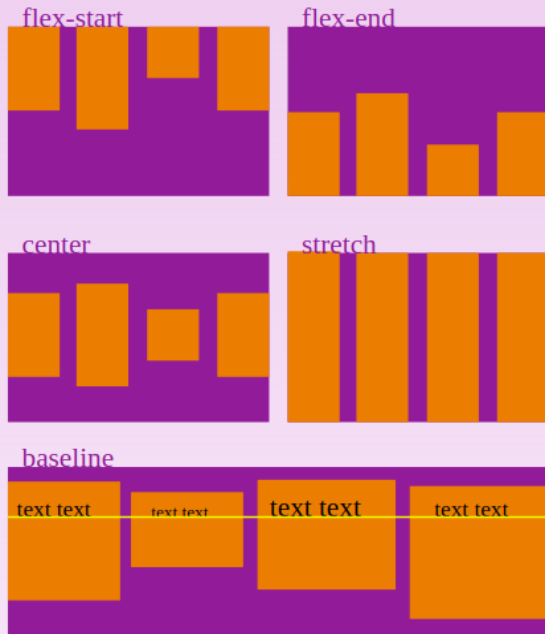
CSS Flexbox - A propriedade align-items (flexbox06.html e flexbox07.html)

A propriedade alinha os elementos em relação ao eixo **vertical**:

* **align-items**: flex-start | flex-end | center | baseline | stretch;

```
.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: center;  
}
```

#align-items



CSS Flexbox - A propriedade align-content (flexbox08.html)

A propriedade alinha os elementos quando temos mais de uma linha de itens:

* **align-content:** flex-start | flex-end | center | space-between | space-around | stretch;

```
.flex-container {  
  display: flex;  
  height: 600px;  
  flex-wrap: wrap; /* wrap-reverse; */  
  align-content: space-between;  
}
```

align-content



CSS Flexbox

Trabalhando com os itens do container, ou os elementos filhos.

Os elementos filho diretos de um contêiner flexível automaticamente se tornam itens flexíveis (flex).

Como são elementos flexíveis, também possuem propriedades que podemos utilizar para alterar sua disposição dentro do container (div pai). Eles são:

- 1 - **order**
- 2 - **flex-grow**
- 3 - **flex-shrink**
- 4 - **flex-basis**
- 5 - **flex**
- 6 - **align-self**

CSS Flexbox – A propriedade order (flexbox09.html)

A propriedade **order** especifica a ordem dos elementos filhos (itens):

```
.flex-container {  
  display: flex;  
  align-items: stretch;  
  background-color: #f1f1f1;  
}  
  
<div class="flex-container">  
  <div style="order: 3">1</div>  
  <div style="order: 2">2</div>  
  <div style="order: 4">3</div>  
  <div style="order: 1">4</div>  
</div>
```



CSS Flexbox - A propriedade flex-grow (flexbox10.html)

A propriedade **flex-grow** especifica a proporção de crescimento de cada elemento, o valor padrão é 0.

```
.flex-container {  
  display: flex;  
  align-items: stretch;  
  background-color: #f1f1f1;  
}  
  
<div class="flex-container">  
  <div style="flex-grow: 1">1</div>  
  <div style="flex-grow: 1">2</div>  
  <div style="flex-grow: 8">3</div>  
</div>
```

1

2

3

CSS Flexbox - A propriedade flex-shrink (flexbox11.html)

A propriedade **flex-shrink** especifica a proporção de encolhimento de um elemento em relação aos outros, o valor padrão é 1.

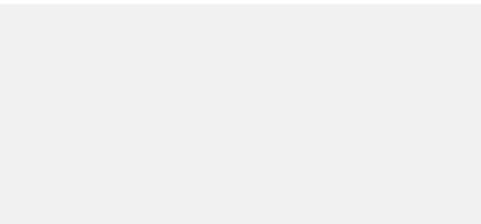
```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div style="flex-shrink: 0">3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
</div>
```



CSS Flexbox - A propriedade flex-basis(flexbox12.html)

A propriedade **flex-basis** especifica o comprimento inicial de um item flexível, o valor padrão é auto.

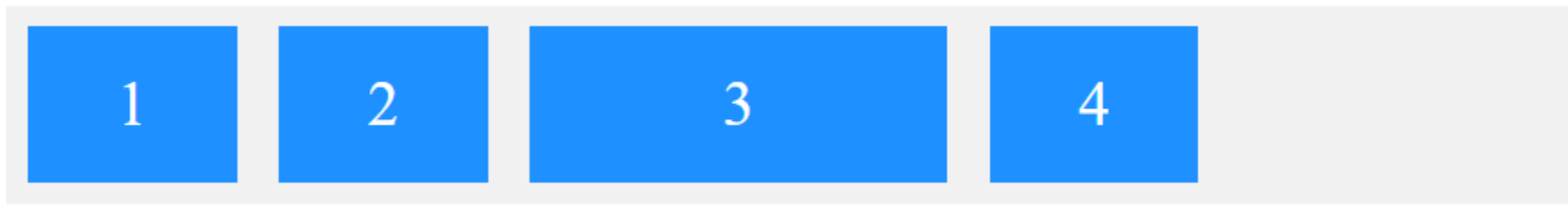
```
<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex-basis:200px">3</div>  
  <div>4</div>  
</div>
```



CSS Flexbox - A propriedade flex

A propriedade **flex** é uma propriedade abreviada para as propriedades **flex-grow**, **flex-shrink** e **flex-basis**.

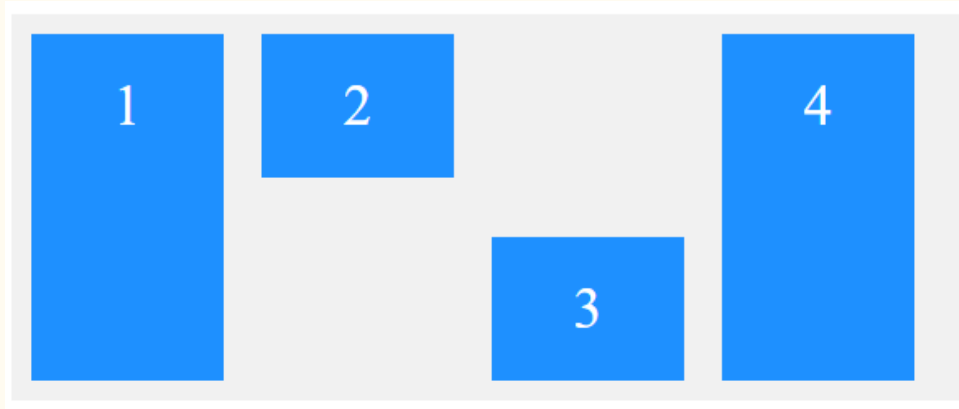
```
<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex: 0 0 200px">3</div>  
  <div>4</div>  
</div>
```



CSS Flexbox - A propriedade align-self (flexbox13.html)

A propriedade **align-self** especifica o alinhamento para o item selecionado dentro do contêiner flexível. Ela substitui o alinhamento padrão definido pela propriedade **align-items** do contêiner.

```
<div class="flex-container">  
  <div>1</div>  
  <div style="align-self: flex-start">2</div>  
  <div style="align-self: flex-end">3</div>  
  <div>4</div>  
</div>
```



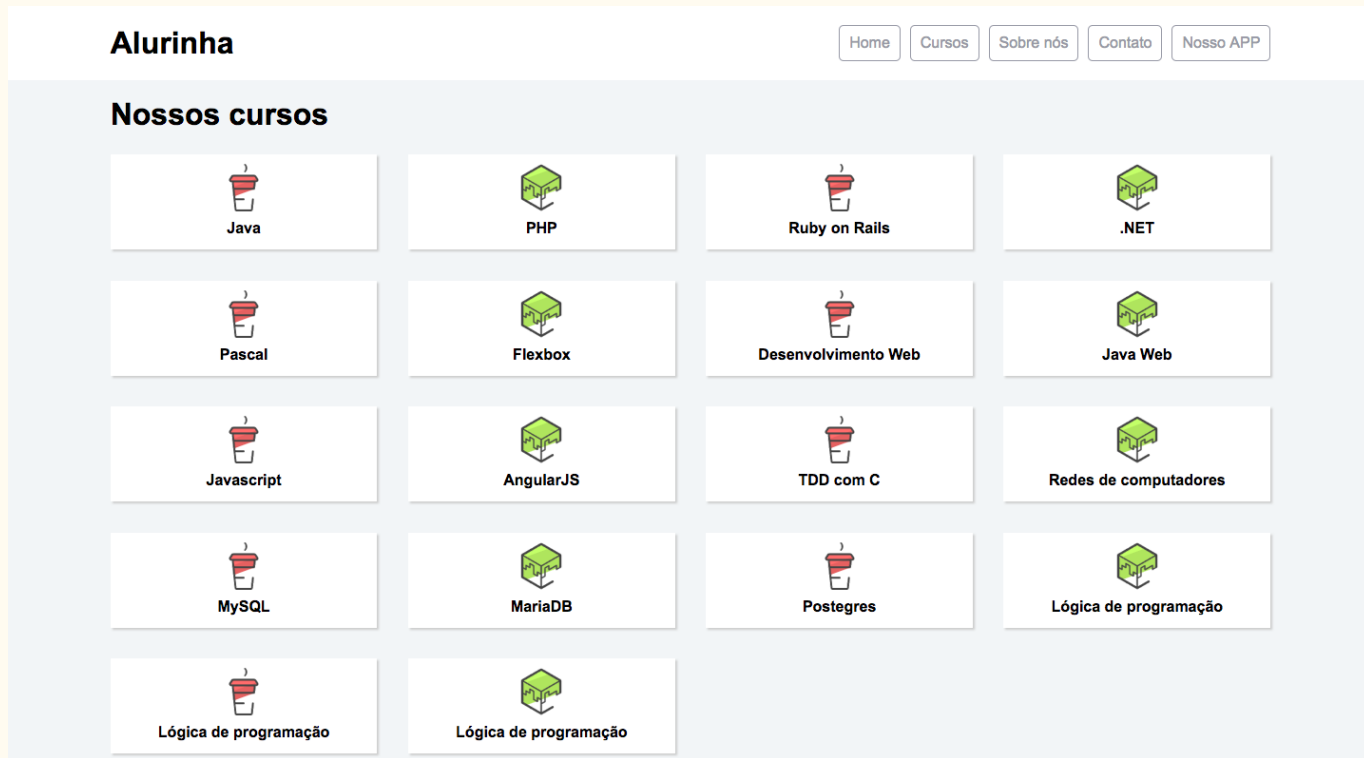
Exercício Prático FlexBox

- Exercício baseado no material do curso **Alura** e dos livros da **Casa do Código** chamado:

Flexbox: Posicione elementos na tela

Entrega dia 15/10/2019 até 23:59

Objetivo 1 (Tela de Notebook)



Objetivo 2 (Tela Mobile)



Vamos fazer a barra de navegação juntos!!

1. Dentro da pasta `css`, crie um novo arquivo chamado **`flexbox.css`**.
2. Importe esse arquivo para o `index.html`

```
<link rel="stylesheet" href="css/flexbox.css">
```

3. No arquivo **`flexbox.css`** vamos acrescentar o **`.cabecalhoPrincipal .container`** e dentro disso adicionamos o `display: flex` e para alinhar os itens no centro o **`align-items: center`**

```
.cabecalhoPrincipal .container {  
    display: flex;  
    align-items: center;  
}
```

Vamos fazer a barra de navegação juntos!!

4. Agora, vamos inserir um espaço vazio entre o título e a lista de itens. Portanto, vamos utilizar o **justify-content:space-between;**

```
.cabecalhoPrincipal .container {  
  display: flex;  
  align-items: center;  
  justify-content: space-between;  
}
```

Vamos fazer a barra de navegação juntos!!

4. Por último vamos adicionar a propriedade **display: flex** nos elementos do menu que estão agrupados dentro do nav que é, o elemento "pai".

```
.cabecalhoPrincipal-nav {  
    display: flex;  
  
}
```