

CSS



Artist Image : Descourtilz, Jean-Théodore | Dates:179?-1855

# Guia iniciante: CSS

# ABOUT ME



*Brunna Croches*

***Developer Full Stack***

Brunna Croches é Dev FullStack, advogada e empreendedora.

Apaixonada por tech, vem adquirido vasto conhecimento na área.

Desenvolveu projetos ricos em diversidade, buscando captar as próximas tendências e necessidades do mercado.

Neste e-book você aprenderá ou recapitulará de forma simplificada e otimizados conceitos de programação feito por ela.

*let's share*



# SUMMARY

---

---

## FLOAT

**13**

*Criando caixas flutuantes*

---

## POSICIONAMENTO Z-INDEX

**14**

*Alinhando os elementos um do lado do outro e redimensionando*

---

## MODO RESPONSIVO

**15**

*Na prática com o modelo responsivo.*

---

## FLEX BOX

**16**

*Organizando ítems dentro de um elemento pai.*

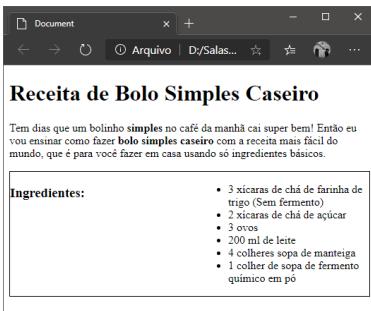
## 13- Float

### Float

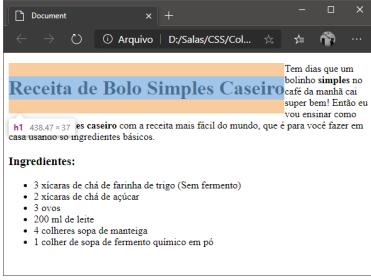
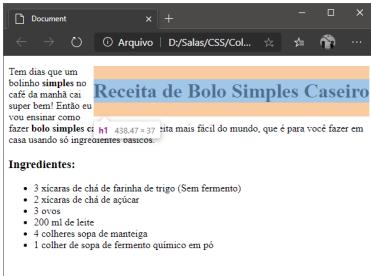
### Flutuação (float)

### HTML

Essa propriedade é muito utilizada para criar colunas e, por isso, é fundamental na construção de uma página.



Nesse modelo de posicionamento o **elemento sai do fluxo normal (Slide 1) e é posicionado totalmente a esquerda ou à direita do contêiner que o contém**. A caixa do elemento que flutua se torna de nível de bloco, caso ainda não seja, e cada outro elemento se ajusta ao redor dele. No Slide 1 vemos o h1 flutuar na página. Perceba como os demais elementos se posicionam ao redor dele.



```
<!DOCTYPE html>
<html lang="pt-br">
```

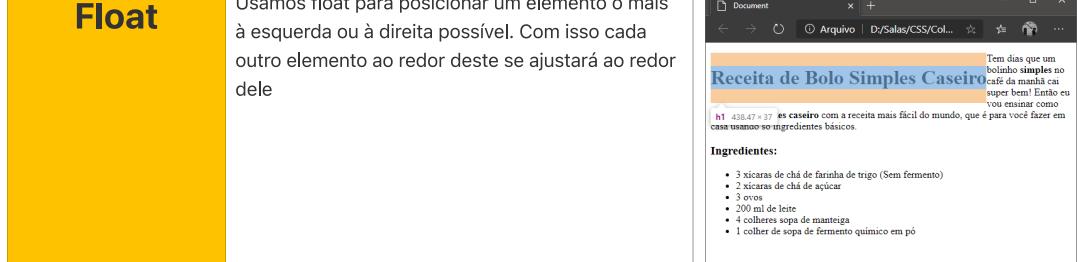
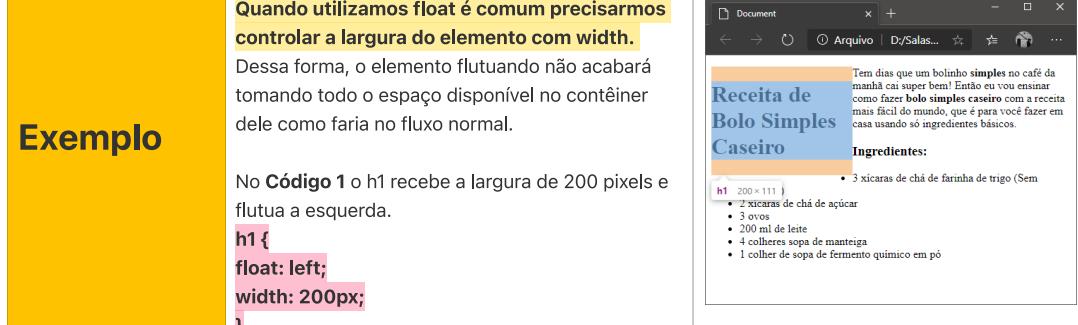
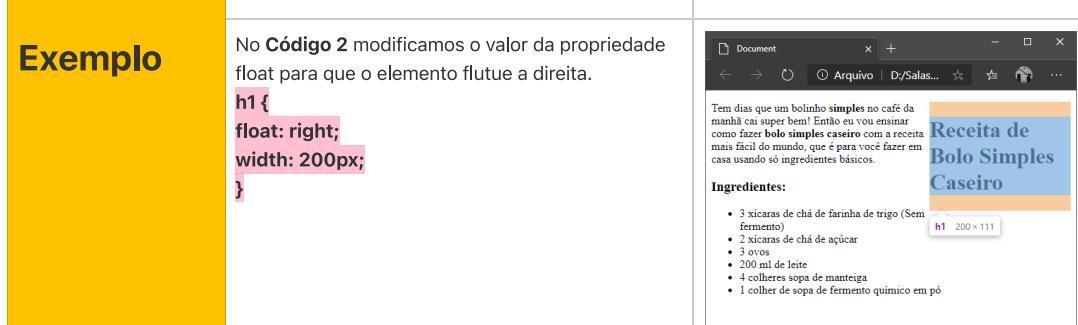
```
<head>
    <meta charset="UTF-8">
    <title>Document</title>
</head>

<body>
    <h1>Receita de Bolo Simples Caseiro</h1>

    <p>Tem dias que um bolinho
    <strong>simples</strong> no café da
    manhã cai super bem! Então eu vou ensinar
    como fazer
    <strong>bolo simples caseiro</strong> com a
    receita mais
    fácil do mundo, que é para você fazer em casa
    usando só
    ingredientes básicos.</p>
```

```
<h3>Ingredientes:</h3>

<ul>
    <li>3 xícaras de chá de farinha de trigo (Sem
    fermento)</li>
    <li>2 xícaras de chá de açúcar</li>
    <li>3 ovos</li>
    <li>200 ml de leite</li>
    <li>4 colheres sopa de manteiga</li>
```

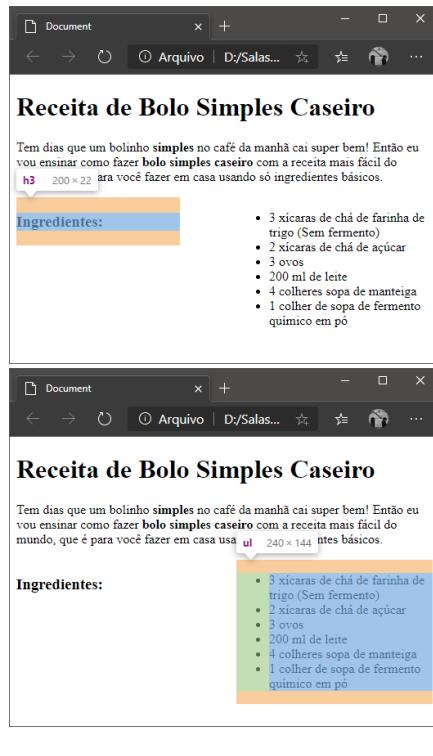
	<pre>&lt;li&gt;1 colher de sopa de fermento químico em pó&lt;/li&gt; &lt;/ul&gt; &lt;/body&gt;</pre>	
	<pre>&lt;/html&gt;</pre>	
<b>Float</b>	<p>Usamos float para posicionar um elemento o mais à esquerda ou à direita possível. Com isso cada outro elemento ao redor deste se ajustará ao redor dele</p>	 
<b>Sintaxe</b>	<p>Para utilizar esse modelo de posicionamento fazemos o seguinte:</p> <pre>float: left   right;</pre>	<p>Float possui dois valores comumente utilizados: left ou right. Left move o elemento o máximo possível para a esquerda e right faz o mesmo para a direita.</p>
<b>Exemplo</b>	<p><b>Quando utilizamos float é comum precisarmos controlar a largura do elemento com width.</b> Dessa forma, o elemento flutuando não acabará tomando todo o espaço disponível no contêiner dele como faria no fluxo normal.</p> <p>No <b>Código 1</b> o h1 recebe a largura de 200 pixels e flutua a esquerda.</p> <pre>h1 {     float: left;     width: 200px; }</pre>	
<b>Exemplo</b>	<p>No <b>Código 2</b> modificamos o valor da propriedade float para que o elemento flutue a direita.</p> <pre>h1 {     float: right;     width: 200px; }</pre>	

## Colocando Elementos lado a lado

	<p>No Código 1 dividimos uma área da página em duas colunas movendo o h3 para a esquerda e a lista para a direita.</p> <pre>h3 {     float: left; }</pre>	
--	---	--

# Criando colunas com Float

```
width: 200px;  
}  
  
ul {  
float: right;  
width: 200px;  
}
```



## Resolvendo Problemas com Float

### Resolvendo Problemas com Float

#### HTML

Quando usamos float em todos elementos dentro de um contêiner, alguns navegadores podem tratá-lo como se ele tivesse zero pixel de altura.

```
...  
<div>  
    <h3>Ingredientes:</h3>  
  
    <ul>  
        <li>3 xícaras de chá de farinha de trigo (Sem fermento)</li>  
        <li>2 xícaras de chá de açúcar</li>  
        <li>3 ovos</li>  
        <li>200 ml de leite</li>  
        <li>4 colheres sopa de manteiga</li>  
        <li>1 colher de sopa de fermento químico em pó</li>  
    </ul>  
</div>  
...
```

#### CSS

O problema do contêiner com altura de zero pixel **ocorre quando todos os elementos em um contêiner estão flutuando.**

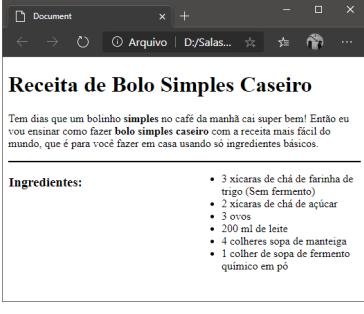
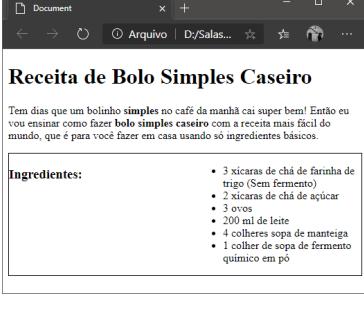
Para ver um exemplo disso usaremos as seguintes regras sobre o HTML apresentado anteriormente:

```
div {
```

**Código 2.** Regras para simular o problema  
A linha observada na **Figura 1** na verdade são as bordas da div. **Uma vez que ela tem altura de zero pixel, as bordas dela se encostam, causando a impressão de que vemos uma linha espessa.**

## A forma de resolver esse problema

## A forma de resolver esse problema

<pre>border: 1px solid #000; }  h3 {     float: left;     width: 200px; }  ul {     float: right;     width: 200px; }</pre>	 <p><b>Receita de Bolo Simples Caseiro</b> Tem dias que um bolinho simples no café da manhã cai super bem! Então eu vou ensinar como fazer <b>bolo simples caseiro</b> com a receita mais fácil do mundo, que é para você fazer em casa usando só ingredientes básicos.</p> <p><b>Ingredientes:</b></p> <ul style="list-style-type: none"><li>• 3 xícaras de chá de farinha de trigo (Sem fermento)</li><li>• 2 xícaras de chá de açúcar</li><li>• 3 ovos</li><li>• 200 ml de leite</li><li>• 4 colheres sopa de manteiga</li><li>• 1 colher de sopa de fermento químico em pó</li></ul>
Atualmente, a forma de resolver esse problema requer a utilização de duas propriedades para formar um tipo de Hack, um código CSS usada para corrigir um problema do navegador: <b>overflow</b> <b>width</b>	<b>overflow</b> <b>width</b>
O Código 2 demonstra como podemos usar essas propriedades para corrigir o problema da altura de zero pixel. É necessário modificar apenas a regra aplicada a div.  <pre>div {     border: 1px solid #000;     overflow: auto;     width: 100%; }</pre>	 <p><b>Receita de Bolo Simples Caseiro</b> Tem dias que um bolinho simples no café da manhã cai super bem! Então eu vou ensinar como fazer <b>bolo simples caseiro</b> com a receita mais fácil do mundo, que é para você fazer em casa usando só ingredientes básicos.</p> <p><b>Ingredientes:</b></p> <ul style="list-style-type: none"><li>• 3 xícaras de chá de farinha de trigo (Sem fermento)</li><li>• 2 xícaras de chá de açúcar</li><li>• 3 ovos</li><li>• 200 ml de leite</li><li>• 4 colheres sopa de manteiga</li><li>• 1 colher de sopa de fermento químico em pó</li></ul>

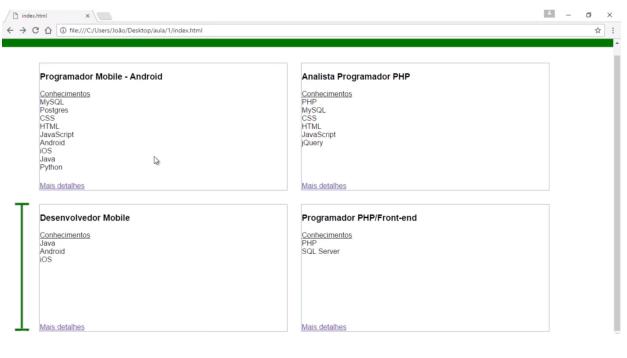
## 14- Posicionamento Z-Index

# Flex

## Como alinhar os elementos?

como alinhar os elementos um ao lado do outro, de forma que o layout não fique feio?

**o problema: como alinhar os elementos um ao lado do outro, de forma que o layout não fique feio**

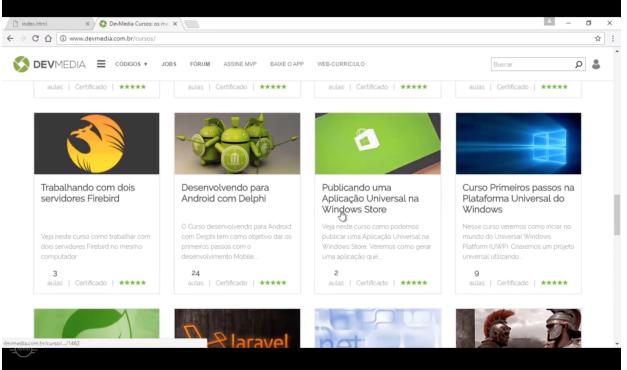


Uma solução de CSS em um contexto de cards

Uma página com cards sempre alinhado corretamente, que não importa a descrição do título o texto fica sempre alinhado no mesmo lugar.

A altura é sempre a mesma de acordo com o CARD

O CARD que tem mais conteúdo e o que manda na linha inteira.



A primeira versão da implementação

**index.html**

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <link rel="stylesheet" type="text/css" href="style.css">
5   </head>
6   <body>
7     <header>header</header>
8     <main>
9       <section class="container">
10        <div class="card">
11          <h3>Programador Mobile -<br>Android</h3>
12          <ul>
13            <li>MySQL<br>Postgres<br>CSS<br>HTML<br>JavaScript<br>Android<br>iOS<br>Java<br>Python<br></li>
14          <a href="http://www.devmedia.com.br/">Mais detalhes</a>
15        </div>
16      </section>
17    </main>
18  </body>
19 </html>

```

**style.css**

```

1 /*font-family: arial;*/
2 html, body{
3   margin: 0;
4   height:100%;
5 }
6 header{
7   display: table;
8   width: 100%;
9   height: 40px;
10  text-align: center;
11  color: #fff;
12  background-color: green;
13  margin-bottom: 20px;
14 }
15 /*****
16  container{
17   width: 90%;
18   margin: 0 auto;
19 }
20 .card{
21   border: 1px solid #86a5ba;
22   margin: 15px;
23   float: left;
24 }
25 .card{
26   border: 1px solid #86a5ba;
27   margin: 15px;
28   float: left;
29 }
30 .card{
31   border: 1px solid #86a5ba;
32   margin: 15px;
33   float: left;
34 }
35 .card{
36   border: 1px solid #86a5ba;
37   margin: 15px;
38   float: left;
39 }

```

**index.html**

```

4 <link rel="stylesheet" type="text/css" href="style.css">
5 </head>
6 <body>
7   <header>header</header>
8   <main>
9     <section class="container">
10      <div class="card">
11        <h3>Programador Mobile -<br>Android</h3>
12        <ul>
13          <li>MySQL<br>Postgres<br>CSS<br>HTML<br>JavaScript<br>Android<br>iOS<br>Java<br>Python<br></li>
14        <a href="http://www.devmedia.com.br/">Mais detalhes</a>
15      </div>
16    </section>
17  </main>
18 </body>
19 </html>

```

**style.css**

```

16 /*****
17  container{
18   width: 90%;
19   margin: 0 auto;
20 }
21 .card{
22   border: 1px solid #86a5ba;
23   margin: 15px;
24   float: left;
25   que e o que esta
26   fazendo um ficar
27   do lado um do outro
28   45% em relacao ao container
29   display: table;
30   position: relative;
31   width: 45%;
32   height: 280px;
33 }
34 .card{
35   position: absolute;
36   bottom: 0;
37   ele vai ficar no fundo
38 }
39 }

```

**index.html**

```

A proposta aqui no caso e nao se preocupar
com a altura do card, no caso a fazer com que o
CSS o fazer com que o LAYOUT organize
automaticamente sem a gente ficar se preocupando
com a imagem maior, texto maior, independente
do que seja

```

## Criando o HTML e o CSS base

Início da solução da organização  
automática  
dos CARDS

The screenshot shows a web page with three cards. The first card is titled 'Programador Mobile - Android' and contains a list of technologies: MySQL, Postgres, CSS, HTML, JavaScript, Android, iOS, Java, Python, and a 'Mais detalhes' link. The second card is titled 'Analista Programador PHP' and contains a list of technologies: PHP, MySQL, CSS, HTML, JavaScript, and a 'Mais detalhes' link. The third card is titled 'Desenvolvedor Mobile' and contains a list of technologies: MySQL, Postgres, CSS, HTML, JavaScript, and a 'Mais detalhes' link.

A parte do HTML  
indentica a outra vesao

o que muda e só o CSS

**index.html**

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <link rel="stylesheet" type="text/css" href="style.css">
5   </head>
6   <body>
7     <header>header</header>
8     <main>
9       <section class="container">
10        <div class="card">
11          <h3>Programador Mobile -<br>Android</h3>
12          <ul>
13            <li>MySQL<br>Postgres<br>CSS<br>HTML<br>JavaScript<br>Android<br>iOS<br>Java<br>Python<br></li>
14          <a href="http://www.devmedia.com.br/">Mais detalhes</a>
15        </div>
16      </section>
17    </main>
18  </body>
19 </html>

```

**style.css**

```

4   height:100%;
5 }
6 header{
7   display: table;
8   width: 100%;
9   height: 40px;
10  text-align: center;
11  color: #fff;
12  background-color: green;
13  margin-bottom: 20px;
14 }
15 /*****
16  container{
17   width: 90%;
18   margin: 0 auto;
19 }
20 .card{
21   border: 1px solid #86a5ba;
22   margin: 15px;
23 }
24 .card{
25   border: 1px solid #86a5ba;
26   margin: 15px;
27 }
28 .card{
29   border: 1px solid #86a5ba;
30   margin: 15px;
31 }
32 .card{
33   border: 1px solid #86a5ba;
34   margin: 15px;
35 }
36 .card{
37   border: 1px solid #86a5ba;
38   margin: 15px;
39 }

```

## Organizando os Elementos com Flex Box

Os cards já aparecem um do lado do outro e com comportamento de se auto ajustar dos cards

O HTML continua o mesmo

O que mudou foi o CSS na classe container e cards

```

1. <!DOCTYPE html>
2. <html>
3. <head>
4.   <link rel="stylesheet" type="text/css" href="style.css">
5. </head>
6. <body>
7.   <header></header>
8.   <main>
9.     <section><div>
10.    <h3>Programador Mobile -<br/>Android</h3>
11.    <div>
12.      <ul>
13.        <li>Conhecimentos</li>
14.        <br>
15.        MySQL<br>
16.        PostgreSQL<br>
17.        CSS<br>
18.        JavaScript<br>
19.        HTML<br>
20.        105<br>
21.        Java<br>
22.        Python<br>
23.      </div>
24.      <a href="http://www.devmedia.com.br/">Mais detalhes</a>
25.    </div>
26.  </section>
27. </main>
28. </body>
29. </html>

```

```

1. *{font-family: arial;}
2. html, body{
3.   margin: 0;
4.   height:100%;
5. }
6.
7. header{
8.   display: flex;
9.   width: 100%;
10.  height: 40px;
11.  text-align: center;
12.  color: #fff;
13.  background-color: green;
14.  margin-bottom: 20px;
15. }
16. /*****
17. .container{
18.   width: 90%;
19.   margin: 0 auto;
20.   display: flex;
21.   justify-content: space-around;
22.   align-items: center;
23.   border: 1px solid #86a5ba;
24.   padding: 10px;
25. }
26. .card{
27.   border: 1px solid #86a5ba;
28.   margin: 10px;
}

```

eu posso pegar um pedaco meu site e tornar flexivel eu nao preciso pegar ele inteiro

foi adicionado o display flex que toma os filhos do box container flexivel apenas os filhos diretos ou filhos dos filhos nao teria que definir outro flex para ele

fez as divs alinhadas em linhas fazer essas divs quebrarem para linha de baixo caso nao cabam dentro do container

```

container{
  display: flex;
  flex-flow: row wrap;
}

```

**flex-flow: row wrap;**

**flex-flow: row nowrap;**

ele não quebra linha e fica todos na mesma linha

**flex-flow= colum nowrap**

em forma de coluna

**CONTAINER ALINHADO POR LINHA**

index.html

```

<!DOCTYPE html>
<html>
<head>
</head>
<body>
    <header></header>
    <main>
        <section class="container">
            <div class="card">
                <h3>Programador Mobile - Android</h3>
                <div>
                    <u>Conhecimentos</u>
                    MySQL<br>
                    Postgres<br>
                    CSS<br>
                    HTML<br>
                    JavaScript<br>
                    Android<br>
                    iOS<br>
                    Java<br>
                    Python<br>
                    Mais detalhes
                </div>
            </div>
            <div class="card">
                <h3>Analista Programador PHP</h3>
                <div>
                    <u>Conhecimentos</u>
                    PHP<br>
                    MySQL<br>
                    CSS<br>
                    HTML<br>
                    JavaScript<br>
                    jQuery
                </div>
            </div>
            <div class="card">
                <h3>Desenvolvedor Mobile</h3>
                <div>
                    <u>Conhecimentos</u>
                    Java<br>
                    Android<br>
                    iOS
                </div>
            </div>
            <div class="card">
                <h3>Programador PHP/Front-end</h3>
                <div>
                    <u>Conhecimentos</u>
                    PHP<br>
                    SQL Server
                </div>
            </div>
        </section>
    </main>
</body>

```

style.css

```

height: 40px;
text-align: center;
color: #fff;
background-color: green;
margin-bottom: 20px;

/************/
.container{
width: 90%;
margin: 0 auto;
display: flex;
flex-wrap: row wrap;
}
.card{
border: 1px solid #86a5ba;
margin: 15px;
display: flex;
flex-flow: column nowrap;
flex-basis: 45%;
}
.card div{flex-grow: 1;}

```

45% da largura do container

flex-flow = row wrap

The screenshot shows a web page with a green header bar. Below it is a section with three cards. The first card is for 'Programador Mobile - Android' and the second for 'Analista Programador PHP'. Both cards have a 'Mais detalhes' button at the bottom right. The third card is for 'Desenvolvedor Mobile' and also has a 'Mais detalhes' button. A yellow box highlights the entire container area. Another yellow box highlights the 'Mais detalhes' button on the first card. A red box highlights the 'flex-flow: row wrap;' style in the developer tools' CSS panel.

## DEIXAR O MAIS DETALHES NO FINAL DA PAGINA

Efeito que faz sempre o MAIS DETALHES ficar alinhado no final

This screenshot shows the same page structure as the previous one, but the 'Mais detalhes' button for each card is now positioned at the very bottom of the page, after all other content. A red box highlights the 'Mais detalhes' button on the first card. A pink box highlights the CSS rule '.card div {flex-grow:1;}' in the developer tools' CSS panel.

This screenshot shows the code editor with the Sublime Text interface. It displays the HTML and CSS files for the page. A red box highlights the 'Mais detalhes' button on the first card. A pink box highlights the CSS rule '.card div {flex-grow:1;}' in the styles.css file.

ela faz com que tudo que venha a seguir, ganhe um espaço, seja empurrado para baixo ate o final do container isso garante que o MAIS DETALHES nao fique em cima de um elemento, conteudo

SEM O FLEX GROW - o mais detalhes fica em cima

This screenshot shows the page again, but the 'Mais detalhes' button is now placed directly above the content of the first card, instead of at the bottom of the page. A green box highlights the 'Mais detalhes' button on the first card. A blue box highlights the CSS rule '.card div {flex-grow:1;}' in the developer tools' CSS panel.

ele esta fora da DIV onde tem conteudo

```
INDEX.html
4 <link rel="stylesheet" type="text/css" href="style.css"/>
5 </head>
6 <body>
7   <header>header</header>
8   <main>
9     <section class="container">
10       <div class="card">
11         <h3>Programador Mobile -<br/>
12           Android</h3>
13         <div>
14           <ul>
15             <li>Conhecimentos</li>
16             <li>MySQL</li>
17             <li>PostgreSQL</li>
18             <li>CSS</li>
19             <li>HTML</li>
20             <li>JavaScript</li>
21             <li>Android</li>
22             <li>iOS</li>
23             <li>Python</li>
24           </ul>
25         </div>
26         <a href="http://www.devmedia.com.br/">www.devmedia.com.br/
27       </div>
28     </div>
29     <div class="card">
30       <h3>Analista Programador PHP</h3>
```

# DAR UMA RESPIRAÇÃO ENTRE AS LETRAS

The screenshot shows a developer's environment with several windows open:

- Sublime Text (Unregistered)**: An HTML file named "index.html" is being edited. The code includes sections for MySQL, PostgreSQL, CSS, HTML, JavaScript, and PHP.
- Google Chrome**: A tab titled "Programador Mobile - Android" is open, displaying the content from the index.html file.
- Firefox Developer Edition**: A tab titled "Analista Programador PHP" is open, also displaying the content from the index.html file.
- Developer Tools**: The bottom section shows the DOM, Properties, and Audits panels for the "Analista Programador PHP" page.

```
.container{  
    display: flex;  
    flex-flow: row wrap;
```

Definindo que os controles no interior do container serão organizados com display flex na forma de linhas (row) e com quebra de linha (wrap).

```
.card{  
    display: flex;  
    flex-flow: column wrap;  
    flex-basis: 45%;  
}
```

Definindo que os cards têm display flexível (flex), com seu conteúdo sendo organizado na forma de colunas (column), com quebra de coluna (wrap), e partindo de uma largura base de 45% do container.

```
.card div{  
    flex-grow: 1;  
}
```

Definindo que a div no interior do card deve ocupar todo o espaço disponível.

# Consulta rápida:

## 15- Modo Responsivo

### Programador Mobile - Android

#### Conhecimentos

MySQL  
Postgres  
CSS  
HTML  
JavaScript  
Android  
iOS  
Java  
Python

[Mais detalhes](#)

### Analista Programador PHP

#### Conhecimentos

PHP  
MySQL  
CSS  
HTML  
JavaScript  
jQuery

[Mais detalhes](#)

### Desenvolvedor Mobile

#### Conhecimentos

Java  
Android  
iOS

[Mais detalhes](#)

### Programador PHP/Front-end

#### Conhecimentos

PHP  
SQL Server

[Mais detalhes](#)

## Linhas adicionadas ao display flex para ser responsivo

```
index.html
```

```
1 <!doctype html>
2 <html>
3 <head>
4 <link rel="stylesheet" type="text/css" href="style.css">
5 </head>
6 <body>
7   <header>header</header>
8   <main>
9     <section class="container">
10       <div class="card">
11         <h3>Programador Mobile -<br/>Android</h3>
12         <div>
13           <u>Conhecimentos</u><br/>
14           MySQL<br/>
15           Postgres<br/>
16           CSS<br/>
17           HTML<br/>
18           JavaScript<br/>
19           Android<br/>
20           iOS<br/>
21           Java<br/>
22           Python<br/>
23         </div>
24         <a href="http://www.devmedia.com.br/">Mais<br/>detalhes</a>
25       </div>
26     </section>
27   </main>
28 </body>
29 </html>
```

```
style.css
```

```
4 height:100%;>
5 }
6
7 header{
8   display: table;
9   width: 100%;
10  height: 40px;
11  text-align: center;
12  color: #fff;
13  background-color: green;
14  margin-bottom: 20px;
15 }
16 ****
17 .container{
18   width: 90%;
19   margin: 0 auto;
20
21   display: -webkit-box;
22   display: -moz-box;
23   display: -ms-flexbox;
24   display: -webkit-flex;
25   display: flex;
26
27   -webkit-flex-flow: row wrap;
28   flex-flow: row wrap;
29 }
30
31 }
```

Código Para ficar Responsivo

The screenshot shows a code editor with two tabs: 'index.html' and 'style.css'. The 'index.html' tab contains the following HTML code:

```
1 <!doctype html>
2 <html>
3 <head>
4 <link rel="stylesheet" type="text/css" href="style.css">
5 </head>
6 <body>
7   <header>header</header>
8   <main>
9     <section class="container">
10       <div class="card">
11         <h3>Programador Mobile -<br/>Android</h3>
12         <div>
13           <u>Conhecimentos</u><br/>
14           MySQL<br/>
15           Postgres<br/>
16           CSS<br/>
17           HTML<br/>
18           JavaScript<br/>
19           Android<br/>
20           iOS<br/>
21           Java<br/>
22           Python<br/>
23         </div>
24         <a href="http://www.devmedia.com.br/">Mais<br/>detalhes</a>
25       </div>
```

The 'style.css' tab contains the following CSS code:

```
34 border: 1px solid #86a5ba;
35 margin: 15px;
36
37 display: -webkit-box;
38 display: -moz-box;
39 display: -ms-flexbox;
40 display: -webkit-flex;
41 display: flex;
42
43 -webkit-flex-flow: column wrap;
44 flex-flow: column wrap;
45
46 flex-basis: 45%;
47 }
48
49 .card div{
50   flex-grow: 1;
51   margin-bottom: 10px;
52 }
53
54 @media (max-width: 727px){
55   .card{
56     flex-basis:100%;
57     margin: 15px 0;
58   }
59 }
```

## Consulta rápida

### Consulta rápida:

```
.container{
  display: -webkit-box;
  display: -moz-box;
  display: -ms-flexbox;
  display: -webkit-flex;
  display: flex;
```

-webkit-flex-flow: row wrap;  
flex-flow: row wrap;

}

Código para que o display flex do container funcione em diversos navegadores, incluindo os mais antigos que requerem o uso de prefixos.

```
.card{
  display: -webkit-box;
  display: -moz-box;
  display: -ms-flexbox;
  display: -webkit-flex;
  display: flex;
```

-webkit-flex-flow: column wrap;  
flex-flow: column wrap;

flex-basis: 45%;

}

Código para que o display flex do card funcione em diversos navegadores, incluindo os mais antigos que requerem o uso de prefixos.

```
@media (max-width: 727px){
  .card{
    flex-basis:100%;
    margin: 15px 0;
  }
}
```

```
}
```

Media query para que a página seja responsiva, adequando o tamanho dos cards para 100% da largura do container em telas menores.

```
<!doctype html>
<html>
<head>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
    <header>header</header>
    <main>
        <section class="container">
            <div class="card">
                <h3>Programador Mobile - Android</h3>
                <div>
                    <u>Conhecimentos</u>

                    MySQL
                    Postgres
                    CSS
                    HTML
                    JavaScript
                    Android
                    iOS
                    Java
                    Python

                </div>
                <a href="http://www.devmedia.com.br/">Mais detalhes</a>
            </div>
            <div class="card">
                <h3>Analista Programador PHP</h3>
                <div>
                    <u>Conhecimentos</u>

                    PHP
                    MySQL
                    CSS
                    HTML
                    JavaScript
                    jQuery

                </div>
                <a href="http://www.devmedia.com.br/">Mais detalhes</a>
            </div>
            <div class="card">
                <h3>Desenvolvedor Mobile</h3>
                <div>
                    <u>Conhecimentos</u>

                    Java
                    Android
                    iOS

                </div>
                <a href="http://www.devmedia.com.br/">Mais detalhes</a>
            </div>
            <div class="card">
                <h3>Programador PHP/Front-end</h3>
                <div>
                    <u>Conhecimentos</u>

                    PHP
                    SQL Server

                </div>
                <a href="http://www.devmedia.com.br/">Mais detalhes</a>
            </div>
        </section>
    </main>
</body>
</html>
```

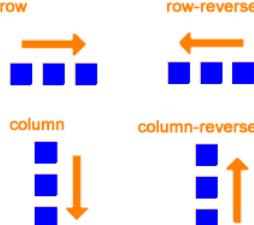
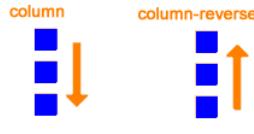
```
*{font-family: arial;}  
html, body{  
    margin: 0;  
    height:100%;  
}  
  
header{  
    display: table;  
    width: 100%;  
    height: 40px;  
    text-align: center;  
    color: #fff;  
    background-color: green;  
    margin-bottom: 20px;  
}  
  
*****  
  
.container{  
    width: 90%;  
    margin: 0 auto;  
  
    display: -webkit-box;  
    display: -moz-box;  
    display: -ms-flexbox;  
    display: -webkit-flex;  
    display: flex;  
  
    -webkit-flex-flow: row wrap;  
    flex-flow: row wrap;  
}  
  
.card{  
    border: 1px solid #86a5ba;  
    margin: 15px;  
  
    display: -webkit-box;  
    display: -moz-box;  
    display: -ms-flexbox;  
    display: -webkit-flex;  
    display: flex;  
  
    -webkit-flex-flow: column wrap;  
    flex-flow: column wrap;  
  
    flex-basis: 45%;  
}  
  
.card div{  
    flex-grow: 1;  
    margin-bottom: 10px;  
}  
  
 @media (max-width: 727px){  
    .card{  
        flex-basis:100%;  
        margin: 15px 0;  
    }  
}
```

## 16- Flex Box

<h3>Flex Box</h3>	<p>O Flexbox é um conjunto de propriedades que tem por objetivo organizar itens dentro de um elemento pai, normalmente chamado de <b>container</b></p>	
<h3>HTML "container"</h3>	<p>Estrutura dos elementos para aplicação do Flexbox Para utilizar esse recurso é necessário ter no HTML ao menos um elemento (<b>container</b>) contendo outros (<b>itens</b>)</p>	<pre>01 &lt;div class="container"&gt; 02   &lt;div class="item item1"&gt;&lt;/div&gt; 03   &lt;div class="item item2"&gt;&lt;/div&gt; 04   &lt;div class="item item3"&gt;&lt;/div&gt; 05 &lt;/div&gt;</pre>
<h3>Alinhamento dos eixos</h3>	<p>Para entender o funcionamento do Flexbox é necessário ter em mente alguns conceitos fundamentais que dizem respeito a como os itens são distribuídos no container.</p>	<p>O principal deles é o conceito de eixo principal e eixo transversal, que depende do valor atribuído à propriedade <b>flex-direction</b>.</p>
<h3>Alinhamento dos eixos HORIZONTAL E VERTICAL</h3>	<p>Se essa propriedade receber o valor : <b>row</b> ou <b>row-reverse</b> (organização em linhas), o eixo principal do container será o horizontal.</p>	<p>Já se essa propriedade receber o valor <b>column</b> ou <b>column-reverse</b> (organização em coluna), o eixo principal será o vertical.</p>
<h3>Alinhamento dos eixos</h3>	<ul style="list-style-type: none"> <li><b>Tamanho principal:</b> É a dimensão do elemento na direção do eixo principal (largura, caso horizontal e altura caso vertical);</li> <li><b>Tamanho transversal:</b> É a dimensão do elemento na direção do eixo transversal (largura, caso horizontal e altura caso vertical);</li> <li><b>Início principal e final principal:</b> Representam o início e o fim do eixo principal;</li> <li><b>Início transversal e final transversal:</b> Representam o início e o fim do eixo transversal.</li> </ul> <p>Essas informações serão importantes para compreendermos o funcionamento das diversas <b>propriedades do Flexbox</b> que veremos a seguir.</p>	

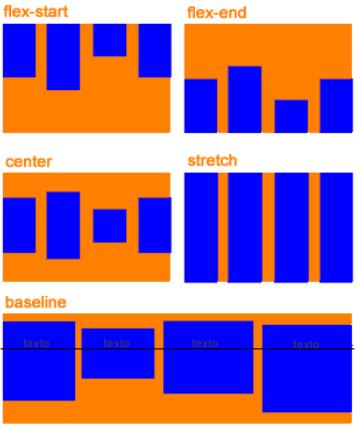
### O primeiro passo

<h3>Display</h3>	<p>O primeiro passo para utilizar o Flexbox é definir a propriedade <b>display</b> do container com o valor <b>flex</b>. Isso é necessário para que as demais propriedades</p>	<pre>01 .container { 02   display: flex; 03 }</pre>
------------------	--	---

	apresentem o resultado esperado.	
<b>Flex-Direction</b>	A propriedade <b>flex-direction</b> deve ser aplicada ao container e define o eixo/fluxo de exibição em que os elementos serão organizados.	01 .container { 02 display: flex; 03 flex-direction: [row / row-reverse / column / column-reverse]; 04 }
<b>Flex-Direction</b>	<ul style="list-style-type: none"> <li>• <b>row</b> (padrão): Os itens são organizados em forma de linha da esquerda para a direita;</li> <li>• <b>row-reverse</b>: Os itens são organizados em forma exibição em linha da direita para a esquerda;</li> <li>• <b>column</b>: Os itens são organizados em forma de colunas iniciando de cima para baixo;</li> <li>• <b>column-reverse</b>: Os itens são organizados em forma de colunas iniciando de baixo para cima.</li> </ul>	 
<b>Flex-flow</b>	<p>Esta propriedade é uma forma abreviada para a escrita das propriedades <b>flex-direction</b> e <b>flex-wrap</b>, nesta ordem. Portanto, ela se aplica ao container.</p>	<pre>01 .container { 02 display: flex; 03 flex-direction: column; 04 flex-wrap: wrap; 05 }</pre> <p>Já com o flex-flow podemos escrever as duas de forma simplificada:</p> <pre>flex-flow: column wrap;</pre>

<b>Justify-content</b>	A propriedade <b>justify-content</b> define o alinhamento dos itens ao longo do eixo principal do container.	01 .container { 02 display: flex; 03 justify-content: [flex-start/flex-end/center/space-between/space-around]; 04 }
<b>Justify-content</b>	<ul style="list-style-type: none"> <li>• <b>flex-start (padrão)</b>: Os itens são alinhados a partir do início do eixo principal;</li> <li>• <b>flex-end</b>: Os itens são alinhados a partir do fim do eixo principal;</li> <li>• <b>center</b>: Os itens são alinhados ao centro do eixo principal;</li> <li>• <b>space-between</b>: O primeiro item é deslocado para o início do eixo principal, o último é deslocado para o final do eixo principal e os demais são distribuídos uniformemente entre eles;</li> <li>• <b>space-around</b>:</li> </ul>	

	<p>Os itens são uniformemente distribuídos ao longo do eixo principal. Aqui, porém, são atribuídas margens iguais à esquerda e à direita (ou acima e abaixo, dependendo da direção do eixo principal). Por isso o primeiro e o último item não ficam "colados" nas bordas do container.</p>	<p>flex-start flex-end center space-between space-around</p>
<h3>Aling-content</h3>	<p>Essa propriedade define como as linhas são distribuídas ao longo do eixo transversal do container.</p>	<pre>01 .container { 02   display: flex; 03   align-content: [stretch/flex-start/flex-end/center/space-between/space-around]; 04 }</pre>
<h3>Aling-content</h3>	<ul style="list-style-type: none"> <li><b>stretch</b> (padrão): As linhas são distribuídas uniformemente ao longo do eixo transversal, ocupando todo o espaço disponível;</li> <li><b>flex-start</b>: As linhas são distribuídas a partir do início do eixo transversal;</li> <li><b>flex-end</b>: As linhas são distribuídas a partir do fim do eixo transversal;</li> <li><b>center</b>: As linhas são mantidas no centro do eixo transversal;</li> <li><b>space-between</b>: A primeira linha é deslocada para o início do eixo transversal, a última é deslocada para o final do eixo transversal e as demais são distribuídas uniformemente entre eles;</li> <li><b>space-around</b>: As linhas são uniformemente distribuídas ao longo do eixo transversal. Aqui, porém, são atribuídas margens iguais à esquerda e à direita (ou acima e abaixo, dependendo da direção do eixo transversal). Por isso a primeira e a última linha não ficam "coladas" nas bordas do container.</li> </ul>	<p>flex-start      flex-end center      stretch space-between      space-around</p>
<h3>Aling-items</h3>	<p>Essa propriedade define como os itens são distribuídos ao longo do eixo transversal do container.</p>	<pre>01 .container { 02   display: flex; 03   align-items: [stretch/flex-start/flex-end/center/space-between/space-around]; 04 }</pre>
<h3>Aling-items</h3>	<ul style="list-style-type: none"> <li><b>stretch</b> (padrão): Os itens serão esticados para preencher toda a dimensão do eixo transversal (altura ou largura);</li> <li><b>flex-start</b>: Os itens são deslocadas para o início do eixo transversal;</li> <li><b>flex-end</b>: Os itens são deslocadas para o final do eixo transversal;</li> <li>•</li> </ul>	

	<p><b>center</b>: Os itens são centralizados no eixo transversal;</p> <ul style="list-style-type: none"> <li>• <b>baseline</b>: Os itens são alinhados a partir da base da primeira linha de texto de cada um.</li> </ul>  <p>The diagram shows a horizontal container with four items. Top row: 'flex-start' (items aligned to the top), 'flex-end' (items aligned to the bottom). Middle row: 'center' (items centered vertically), 'stretch' (items stretched to fit the container height). Bottom row: 'baseline' (items aligned to their baseline), 'space-around' (items with equal gaps between them).</p>
--	--

Order	<p>Por padrão, os itens são distribuídos no container na ordem em que são inseridos no HTML.</p>	<pre>01 .item2 { 02   order: [número] 03 }</pre> <p>O valor numérico atribuído a essa propriedade define a ordem do item. Por exemplo, o valor 2 faz com que o item seja o segundo item ao longo do eixo principal, enquanto o valor -1 faz com que ele apareça antes do primeiro.</p>
Flex Grow	<p>Esta propriedade define a proporção com que um item deve crescer caso seja necessário. Por padrão seu valor é 0, o que indica que o item não deve crescer, e são aceitos apenas valores numéricos positivos.</p>	<pre>01 .item2 { 02   flex-grow: [número] 03 }</pre>
Flex Shrink	<p>Esta propriedade define a proporção com que um item deve encolher caso seja necessário. Por padrão seu valor é 0, o que indica que o item não deve encolher, e são aceitos apenas valores numéricos positivos.</p>	<pre>01 .item2 { 02   flex-shrink: [número] 03 }</pre>
Flex-basis	<p>O <b>flex-basis</b> define o tamanho inicial que um item deve ter antes que o espaço ao seu redor seja distribuído. Ou seja, dependendo da direção do eixo principal (horizontal ou vertical), essa propriedade define a largura ou altura mínima do elemento antes que ele seja redimensionado por outras propriedades.</p>	<pre>01 .item2 { 02   flex-basis: [número] 03 }</pre> <p>O valor atribuído a essa propriedade pode ser em percentual, em pixels, ou a palavra <b>auto</b>, que é o valor padrão (considera as dimensões do item - width e height).</p>
Flex	<p>Esta propriedade é uma forma abreviada para a escrita das propriedades <b>flex-grow</b>, <b>flex-shrink</b> e <b>flex-basis</b>, nesta ordem.</p>	<pre>01 .item2 { 02   flex: [flex-grow] [flex-shrink]       [flex-basis] 03 }</pre>
Align-Self	<p>Esta propriedade permite sobrescrever no item o comportamento que foi</p>	<pre>01 .item2 { 02   align-self: [stretch/flex-start/flex-end/center/space-between/space-around];</pre>

Align-Self	<p>definido pela propriedade <b>align-items</b>.</p> <ul style="list-style-type: none"> <li><b>auto</b> (padrão): Respeita o comportamento definido no container por meio do align-items;</li> <li><b>stretch</b>: O item será esticado para preencher toda a dimensão do eixo transversal (altura ou largura);</li> <li><b>flex-start</b>: O item é deslocado para o início do eixo transversal;</li> <li><b>flex-end</b>: O item é deslocado para o final do eixo transversal;</li> <li><b>center</b>: O item é centralizado no eixo transversal;</li> <li><b>baseline</b>: O item é alinhado a partir da base da primeira linha de texto dos demais.</li> </ul>	03
------------	--	----

## Exemplo Pratico

HTML	CSS	Linha
<pre> 01 &lt;div class="container"&gt; 02   &lt;div class="item item1"&gt;1&lt;/div&gt; 03   &lt;div class="item item2"&gt;2&lt;/div&gt; 04   &lt;div class="item item3"&gt;3&lt;/div&gt; 05 &lt;/div&gt; </pre>	<pre> 01 .container { 02   background-color: #FF5722; 03   height: 100vh; 04   width: 100%; 05   display: flex; 06   flex-direction: column; 07   justify-content: center; 08   align-items: center; 09 } 10 11 .item { 12   background-color: #0000ff; 13   color: #fff; 14   padding: 10px; 15 } 16 17 .item1{ 18   height: 100px; 19   width: 100px; 20 } 21 22 .item2{ 23   height: 50px; 24   width: 150px; 25 } 26 27 .item3{ 28   height: 100px; 29   width: 200px; 30 } </pre>	<p>Linha 2: Definimos a cor do plano de fundo do container para facilitar a visualização;</p> <p>Linhas 3 e 4: Definimos a altura e largura do container para ocupar 100% da página;</p> <p>Linha 5: Definimos o display flex no container para que as demais propriedades surtam o efeito esperado;</p> <p>Linha 6: Com a propriedade flex-direction definida como column definimos que os itens devem ser organizados em coluna (um abaixo do outro);</p> <p>Linha 7: Nesta linha definimos que os itens serão centralizados no eixo principal (vertical);</p> <p>Linha 8: A propriedade align-items define aqui que os itens serão centralizados também ao longo do eixo transversal (horizontal);</p> <p>Linhas 11 a 30: Neste trecho estamos definindo algumas características dos itens. Eles possuem diferentes alturas e larguras, o que facilita a visualização do resultado final (mesmo com diferentes dimensões, o layout é mantido organizado).</p>

# CONTACT



*Brunna Croches*

*Developer Full Stack*



*brunnacroches.dev*



*linkedin.com/brunnacroches*



*github.com/brunnacroches*



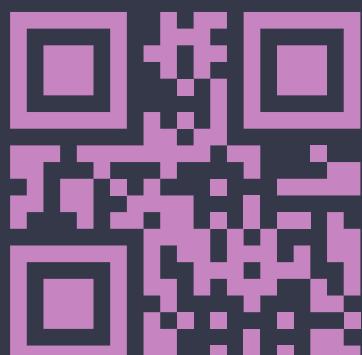
*@brunnacroches.dev*



*discord.com/brunnacroches*



*brunnacroches@gmail.com*



*let's share*