INSTITUTO FEDERAL
Paraná
Campus Paranaguá

Ministério da Educação

ENSINO MÉDIO INTEGRADO: INFORMÁTICA
Disciplina de Desenvolvimento Web
Aula 05: CSS / Flexbox e SASS

Gil Eduardo de Andrade

Conceitos Preliminares

CSS - Cascading Style Sheets (https://www.w3schools.com/html/)

CSS é a sigla para o termo em inglês Cascading Style Sheets (Folha de Estilo em Cascatas), sendo utilizado para estilizar elementos escritos na linguagem HTML. O CSS foi desenvolvido pelo W3C (World Wide Web Consortium - https://www.w3.org/Style/CSS/) em 1996, com intuito de dar suporte a definição de estilos dentro do HTML, que não foi projetado com tags voltadas a formatar e estilizar uma página.

Flexbox

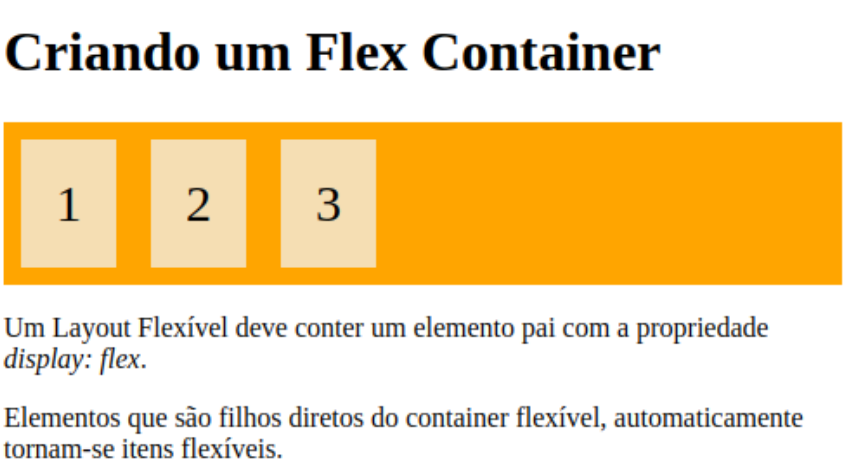
Antes do surgimento do Flexbox como ferramenta para criação de Layout, eram utilizados outras quatro abordagens:

- ***Block***, para seções em uma página da web;
- ***Inline***, para texto;
- ***Table***, para dados de tabela bidimensionais;
- ***Positioned***, para posição explícita de um elemento;

Em contrapartida às abordagens anteriores, o Flexbox facilita a construção de uma estrutura de layout responsiva flexível sem necessidade de uso da flutuação ou posicionamento.
Para que seja possível utilizar o Flexbox é necessário, primeiramente, definir um flex container, como mostrado no exemplo a seguir:

Arquivo: ./css/exemplos/flexbox/ex_flexbox01.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           .flex-container {
               display: flex;
               background-color: orange;
           }
           .flex-container > div {
               background-color: wheat;
               margin: 10px;
               padding: 20px;
               font-size: 30px;
           }
       </style>
   </head>
   <body>
       <h1>Criando um Flex Container</h1>
       <div class="flex-container">
           <div>1</div>
           <div>2</div>
           <div>3</div>
       </div>
          <p>Um Layout Flexível deve conter um elemento pai com a propriedade
<em>display: flex</em>.</p>
              <p>Elementos que são filhos diretos do container flexível,
automaticamente tornam-se itens flexíveis.</p>
   </body>
</html>
```



Propriedades do Elemento Pai (*Container*)

Como visto, o contêiner flexível (*Flexbox*) se torna flexível ao definir a propriedade "***display: flex***". Neste contexto, temos as seguintes propriedades para um contêiner flexível:

- *flex-direction*
- *flex-wrap*
- *flex-flow*
- *justify-content*
- *align-items*
- *align-content*

PROPRIEDADE	**DESCRIÇÃO**
display	Especifica o tipo de caixa usada para um elemento HTML.
flex-wrap	Especifica se os itens flexíveis devem ser quebrados ou não, caso não haja espaço suficiente para eles em uma linha flexível.
flex-direction	Especifica a direção de exibição dos itens flexíveis dentro de um contêiner flexível.
flex-flow	Forma de escrita abreviada para as propriedades flex-wrap e flex-direction.
justify-content	Alinha horizontalmente os itens flexíveis quando estes não

	ocupam todo o espaço disponível no eixo principal.
align-items	Alinha verticalmente os itens flexíveis quando estes não ocupam todo o espaço disponível no eixo secundário.
align-content	Modifica o comportamento da propriedade flex-wrap. É semelhante a align-items, mas em vez de alinhar itens, alinha linhas.

A propriedade *flex-direction*

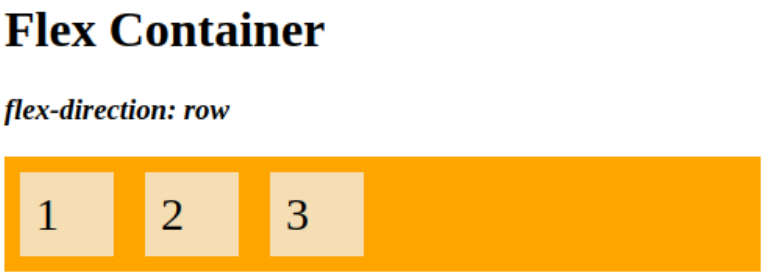
A propriedade "*flex-direction*" define em qual direção o contêiner deve empilhar os itens flexíveis. A propriedade "***flex-direction***" pode assumir os seguintes valores: "***row***", "***column***", "***row-reverse***" e "***column-reverse***", como podemos ver nos exemplos a seguir:

"***flex-direction: row***"

Arquivo: ./css/exemplos/flexbox/ex_flexbox02.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           .flex-container {
               display: flex;
               flex-direction: row;
               background-color: orange;
           }
           .flex-container > div {
               background-color: wheat;
               margin: 10px;
               padding: 10px;
               font-size: 30px;
               width: 40px;
           }
       </style>
   </head>
   <body>
```

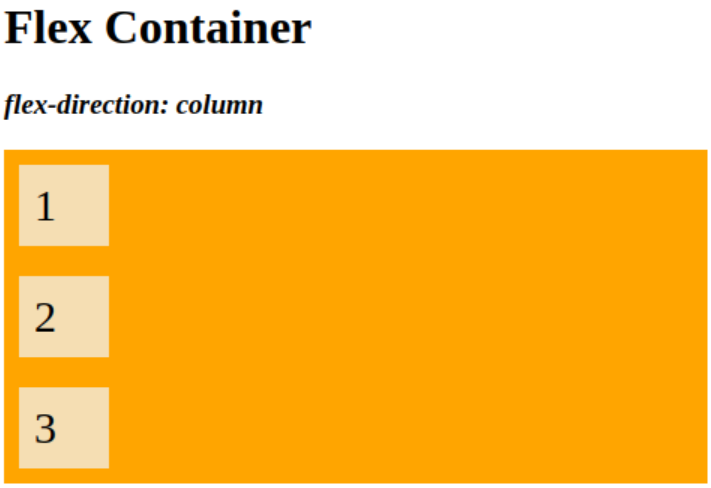
```
        <h1>Flex Container</h1>
        <h3><i>flex-direction: row</i></h3>
        <div class="flex-container">
            <div>1</div>
            <div>2</div>
            <div>3</div>
        </div>
    </body>
</html>
```



“*flex-direction: column*”

Arquivo: ./css/exemplos/flexbox/ex_flexbox03.html

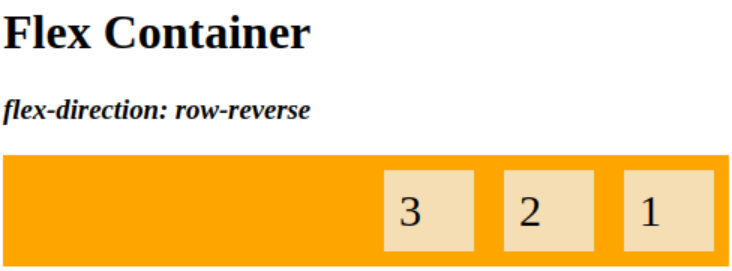
```
.flex-container {
    display: flex;
    flex-direction: column;
    background-color: orange;
}
```



"*flex-direction: row-reverse*"

Arquivo: ./css/exemplos/flexbox/ex_flexbox04.html

```
.flex-container {
    display: flex;
    flex-direction: row-reverse;
    background-color: orange;
}
```



"*flex-direction: column-reverse*"

Arquivo: ./css/exemplos/flexbox/ex_flexbox05.html

```
.flex-container {
    display: flex;
```

```
    flex-direction: column-reverse;
    background-color: orange;
}
```

Flex Container

flex-direction: column-reverse

3

2

1

A propriedade *flex-wrap*

A propriedade "*flex-wrap*" especifica se os itens flexíveis devem ser encapsulados ou não. A propriedade "***flex-wrap***" pode assumir os seguintes valores: "***wrap***", "***nowrap***", e "***wrap-reverse***", como podemos ver nos exemplos a seguir:

"***flex-wrap: wrap***"

Arquivo: ./css/exemplos/flexbox/ex_flexbox06.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           .flex-container {
               display: flex;
               flex-wrap: wrap;
               background-color: orange;
           }
           .flex-container > div {
               background-color: wheat;
               margin: 10px;
```

```
            padding: 10px;
            font-size: 30px;
            width: 150px;
            text-align: center;
        }
    </style>
</head>
<body>
    <h1>Flex Container</h1>
    <h3><i>flex-wrap: wrap</i></h3>
    <p>Redimensione a janela para visualizar o efeito.</p>
    <div class="flex-container">
        <div>1</div>
        <div>2</div>
        <div>3</div>
        <div>4</div>
        <div>5</div>
        <div>6</div>
        <div>7</div>
        <div>8</div>
        <div>9</div>
        <div>10</div>
        <div>11</div>
        <div>12</div>
    </div>
</body>
</html>
```

Flex Container

flex-wrap: wrap

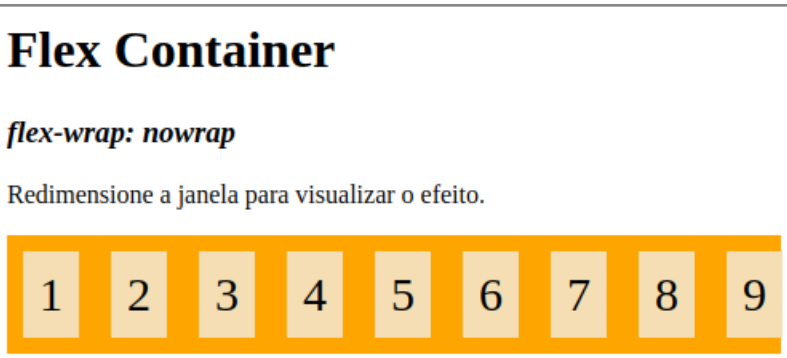
Redimensione a janela para visualizar o efeito.

1	2
3	4

"flex-wrap: nowrap"

Arquivo: ./css/exemplos/flexbox/ex_flexbox07.html

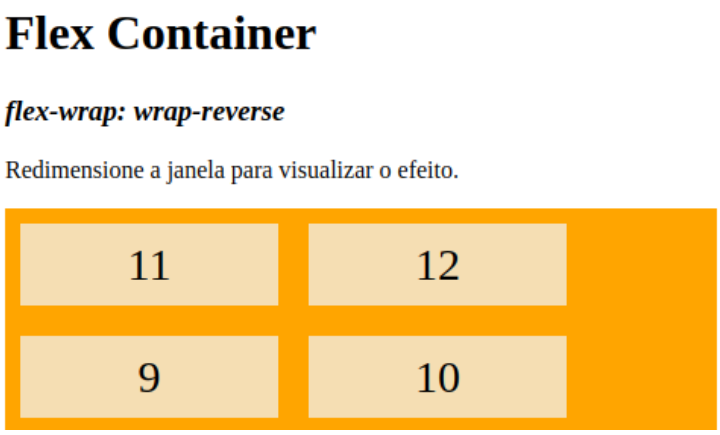
```
.flex-container {
    display: flex;
    flex-wrap: nowrap;
    background-color: orange;
}
```



"flex-wrap: wrap-reverse"

Arquivo: ./css/exemplos/flexbox/ex_flexbox08.html

```
.flex-container {
    display: flex;
    flex-wrap: wrap-reverse;
    background-color: orange;
}
```

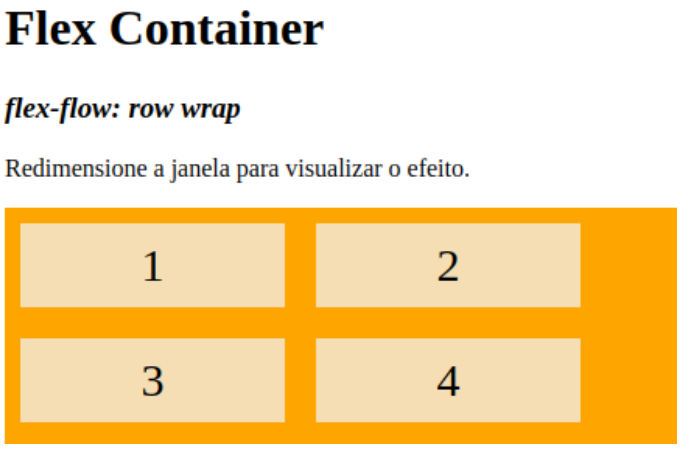


A propriedade *flex-flow*

A "*flex-flow*" é uma propriedade abreviada que possibilita definir, em uma única linha, as propriedades "*flex-direction*" e "*flex-wrap*". Veja o exemplo a seguir:

Arquivo: ./css/exemplos/flexbox/ex_flexbox09.html

```
.flex-container {
    display: flex;
    flex-flow: row wrap;
    background-color: orange;
}
```



A propriedade *justify-content*

A propriedade "*justify-content*" permite que os itens flexíveis sejam alinhados. A propriedade "***justify-content***" pode assumir os seguintes valores: "***center***", "***flex-start***", "***flex-end***", "***space-around***", e "***space-between***", como podemos ver nos exemplos a seguir:

"***justify-content: center***"

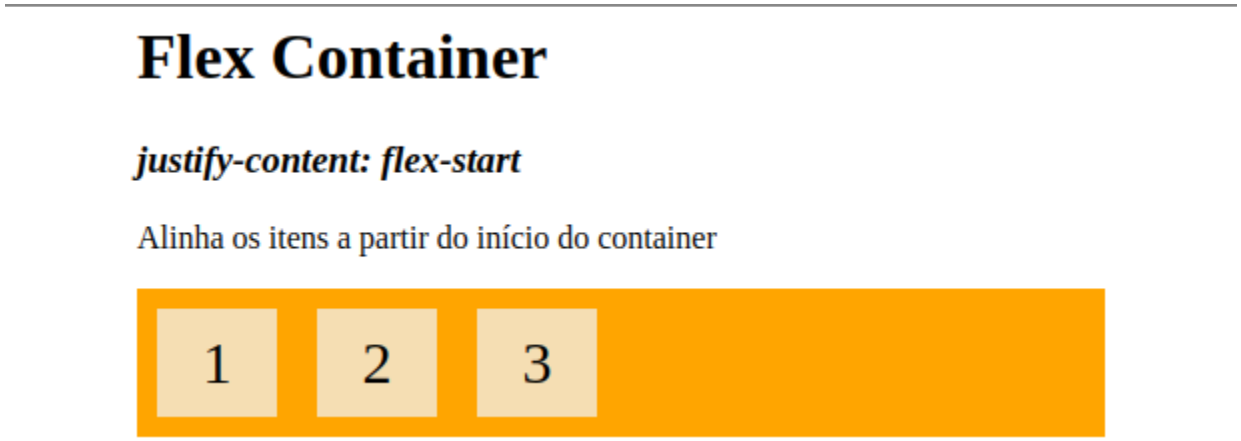
Arquivo: ./css/exemplos/flexbox/ex_flexbox10.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           .flex-container {
               display: flex;
               justify-content: center;
               background-color: orange;
           }
           .flex-container > div {
               background-color: wheat;
               margin: 10px;
               padding: 10px;
               font-size: 30px;
               width: 40px;
               text-align: center;
           }
       </style>
   </head>
   <body>
       <h1>Flex Container</h1>
       <h3><i>justify-content: center</i></h3>
       <p>Alinha os itens no centro</p>
       <div class="flex-container">
           <div>1</div>
           <div>2</div>
           <div>3</div>
       </div>
   </body>
</html>
```

"***justify-content: flex-start***"

Arquivo: ./css/exemplos/flexbox/ex_flexbox11.html

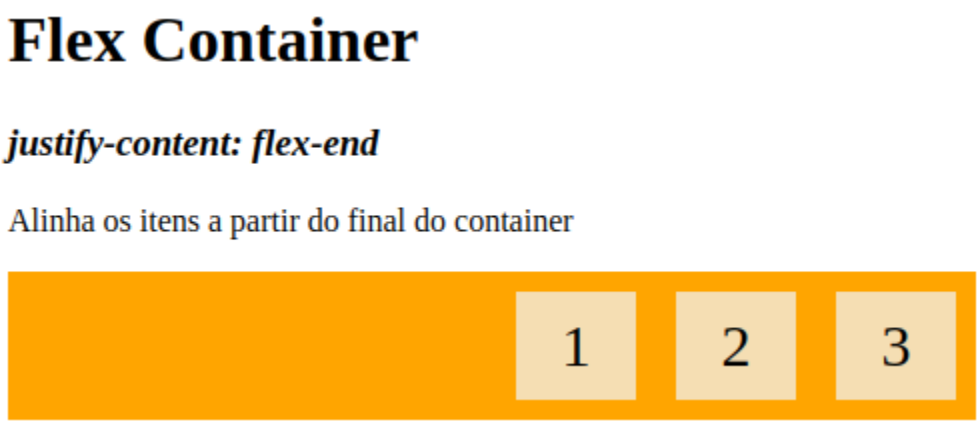
```
.flex-container {
    display: flex;
    justify-content: flex-start;
    background-color: orange;
}
```



"***justify-content: flex-end***"

Arquivo: ./css/exemplos/flexbox/ex_flexbox12.html

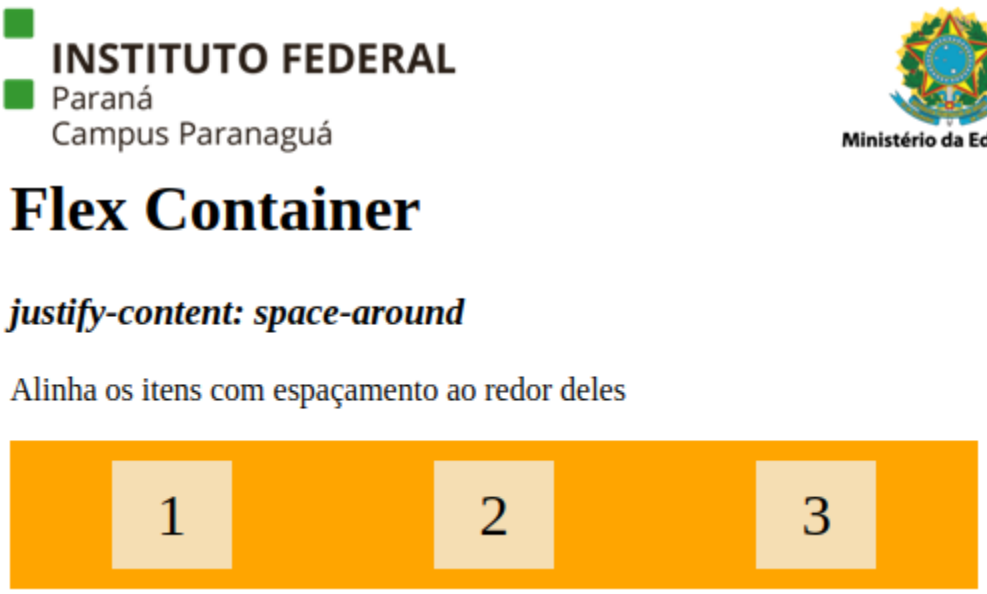
```
.flex-container {
    display: flex;
    justify-content: flex-end;
    background-color: orange;
}
```



*"**justify-content: space-around**"*

Arquivo: ./css/exemplos/flexbox/ex_flexbox13.html

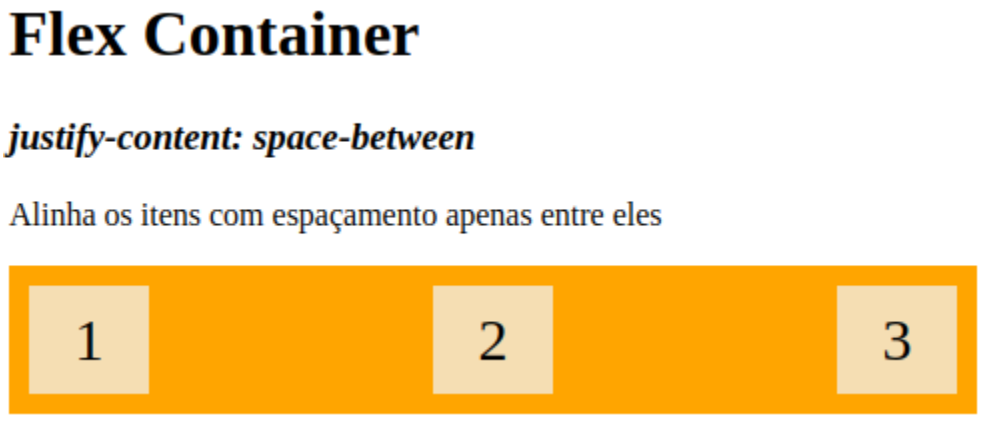
```
.flex-container {
    display: flex;
    justify-content: space-around;
    background-color: orange;
}
```



"***justify-content: space-between***"

Arquivo: ./css/exemplos/flexbox/ex_flexbox14.html

```
.flex-container {
    display: flex;
    justify-content: space-between;
    background-color: orange;
}
```



A propriedade *align-items*

A propriedade "*align-items*" permite que os itens flexíveis sejam alinhados. A propriedade "***align-items***" pode assumir os seguintes valores: "***center***",

"***flex-start***", "***flex-end***", "***stretch***", e "***baseline***", como podemos ver nos exemplos a seguir:

"***align-items: center***"

Arquivo: ./css/exemplos/flexbox/ex_flexbox15.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           .flex-container {
               display: flex;
               height: 200px;
               align-items: center;
               background-color: orange;
           }
           .flex-container > div {
               background-color: wheat;
               margin: 10px;
               padding: 10px;
               font-size: 30px;
               width: 40px;
               text-align: center;
           }
       </style>
   </head>
   <body>
       <h1>Flex Container</h1>
       <h3><i>align-items: center</i></h3>
       <p>Alinha os itens no centro</p>
       <div class="flex-container">
           <div>1</div>
           <div>2</div>
           <div>3</div>
       </div>
   </body>
</html>
```

"*align-items: flex-start*"

Arquivo: ./css/exemplos/flexbox/ex_flexbox16.html

```
.flex-container {
    display: flex;
    height: 200px;
    align-items: flex-start;
    background-color: orange;
}
```

"*align-items: flex-end*"

Arquivo: ./css/exemplos/flexbox/ex_flexbox17.html

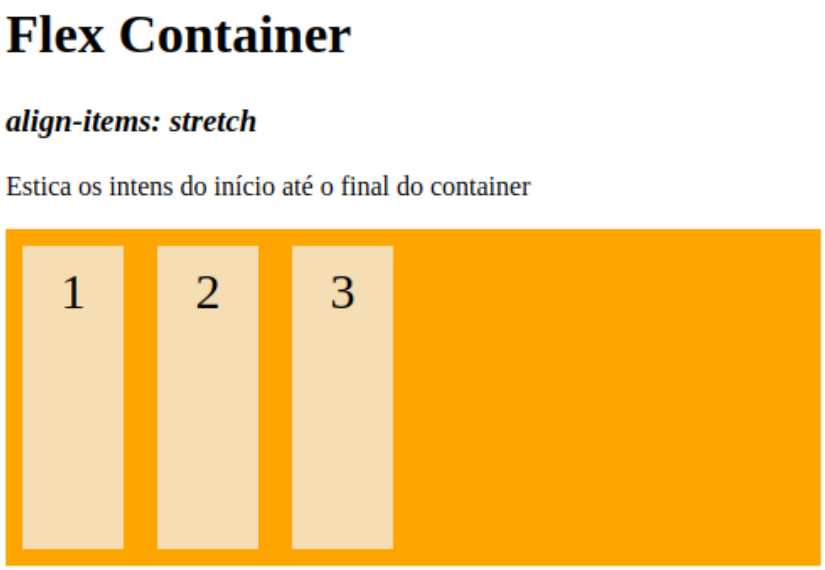
```
.flex-container {
    display: flex;
    height: 200px;
    align-items: flex-end;
    background-color: orange;
}
```

---

*"**align-items: stretch**"*

Arquivo: ./css/exemplos/flexbox/ex_flexbox18.html

```
.flex-container {
    display: flex;
    height: 200px;
    align-items: stretch;
    background-color: orange;
}
```



"*align-items: baseline*"

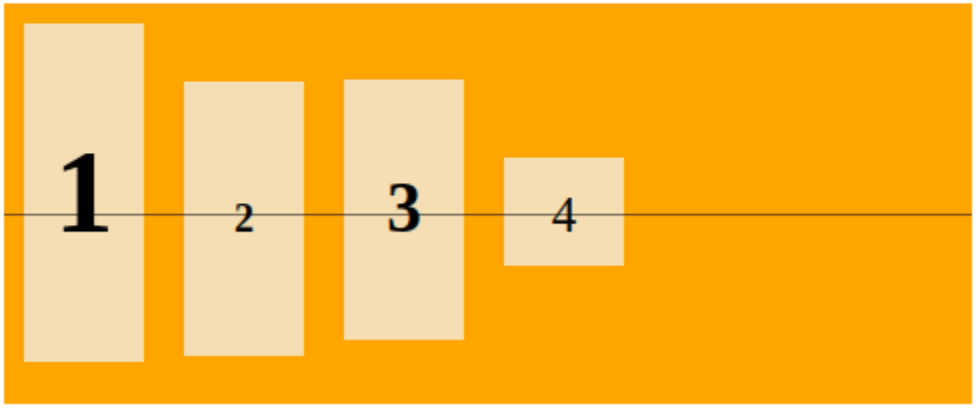
Arquivo: ./css/exemplos/flexbox/ex_flexbox19.html

```
.flex-container {
    display: flex;
    height: 200px;
    align-items: baseline;
    background-color: orange;
}
```

Flex Container

align-items: baseline

Alinha os intens considerando uma linha de base



A propriedade *align-content*

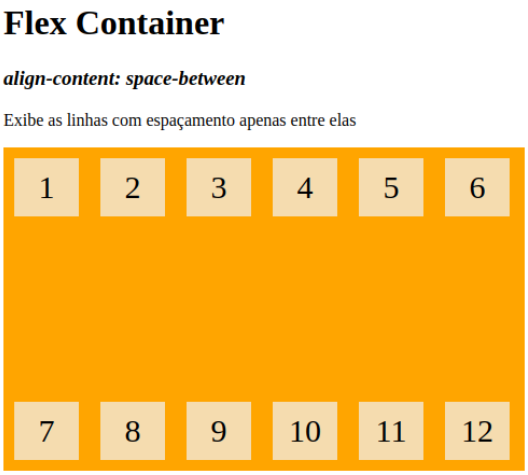
A propriedade "*align-content*" permite que as linhas flexíveis sejam alinhadas. A propriedade "***align-content***" pode assumir os seguintes valores: "***center***", "***flex-start***", "***flex-end***", "***stretch***" "***space-around***", e "***space-between***". Nos exemplos a seguir, usamos um container de 300px de altura, com a propriedade "***flex-wrap***" definida como "***wrap***", o que permite uma melhor demonstração sobre o funcionamento da propriedade "***align-content***".

"***align-content: space-between***"

Arquivo: ./css/exemplos/flexbox/ex_flexbox20.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           .flex-container {
               display: flex;
               flex-wrap: wrap;
               height: 300px;
```

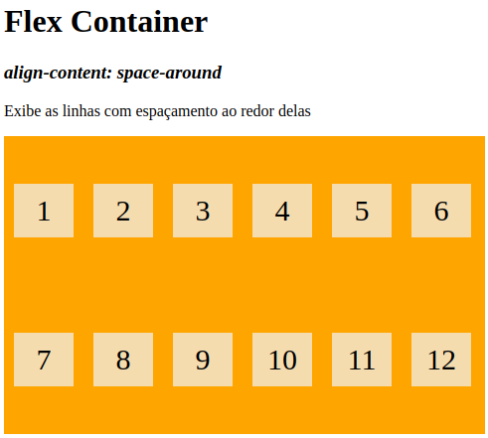
```
            align-content: space-between;
            background-color: orange;
        }
        .flex-container > div {
            background-color: wheat;
            margin: 10px;
            padding: 10px;
            font-size: 30px;
            width: 40px;
            text-align: center;
        }
    </style>
</head>
<body>
    <h1>Flex Container</h1>
    <h3><i>align-content: space-between</i></h3>
    <p>Exibe as linhas com espaçamento apenas entre elas</p>
    <div class="flex-container">
        <div>1</div>
        <div>2</div>
        <div>3</div>
        <div>4</div>
        <div>5</div>
        <div>6</div>
        <div>7</div>
        <div>8</div>
        <div>9</div>
        <div>10</div>
        <div>11</div>
        <div>12</div>
    </div>
</body>
</html>
```



"align-content: space-around"

Arquivo: ./css/exemplos/flexbox/ex_flexbox21.html

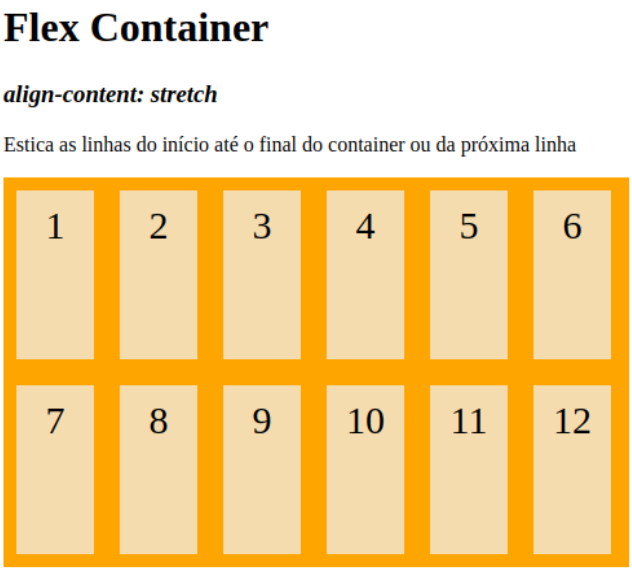
```
.flex-container {
    display: flex;
    flex-wrap: wrap;
    height: 300px;
    align-content: space-around;
    background-color: orange;
 }
```



“*align-content: stretch*”

Arquivo: ./css/exemplos/flexbox/ex_flexbox22.html

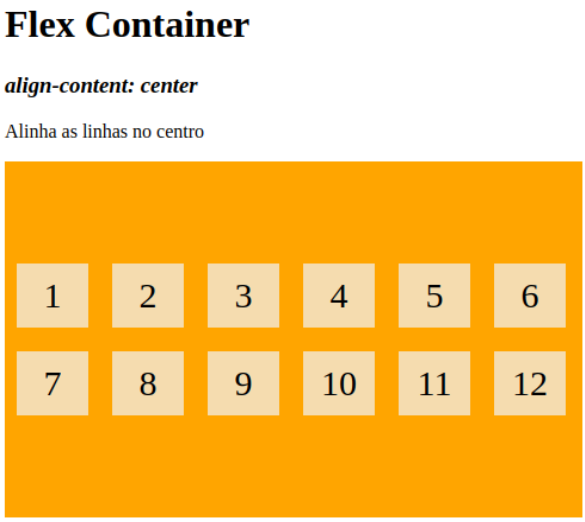
```
.flex-container {
    display: flex;
    flex-wrap: wrap;
    height: 300px;
    align-content: stretch;
    background-color: orange;
 }
```



“*align-content: center*”

Arquivo: ./css/exemplos/flexbox/ex_flexbox23.html

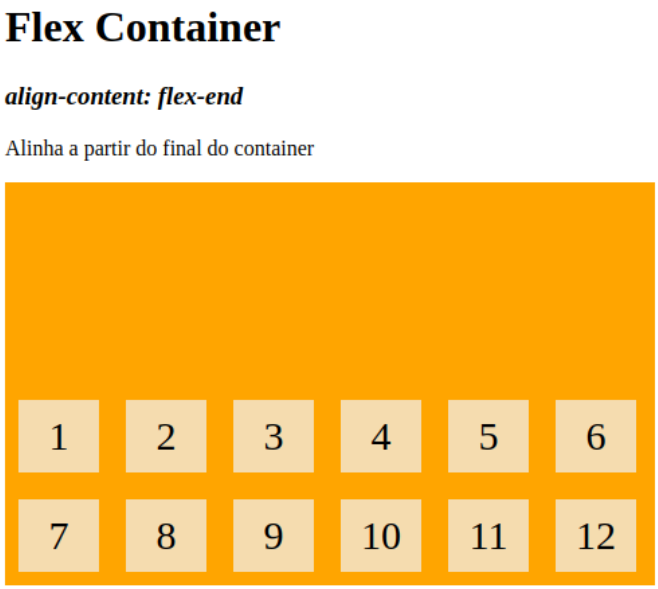
```
.flex-container {
    display: flex;
    flex-wrap: wrap;
    height: 300px;
    align-content: center;
    background-color: orange;
 }
```



"*align-content: flex-end*"

Arquivo: ./css/exemplos/flexbox/ex_flexbox24.html

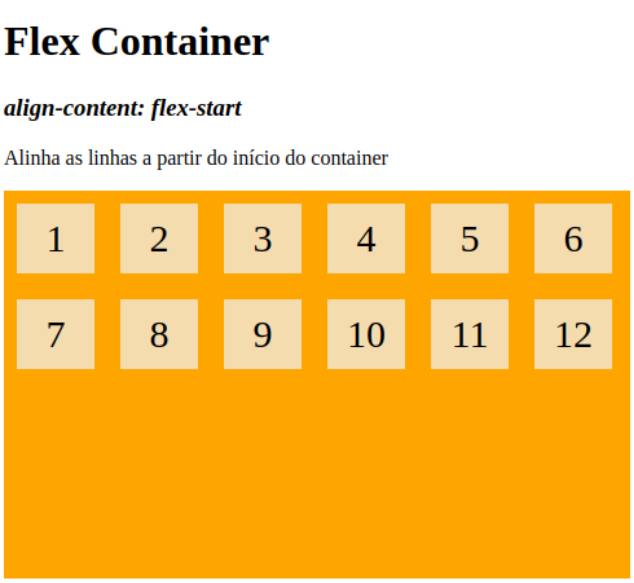
```
.flex-container {
    display: flex;
    flex-wrap: wrap;
    height: 300px;
    align-content: flex-end;
    background-color: orange;
 }
```



“*align-content: flex-start*”

Arquivo: ./css/exemplos/flexbox/ex_flexbox25.html

```
.flex-container {
    display: flex;
    flex-wrap: wrap;
    height: 300px;
    align-content: flex-start;
    background-color: orange;
 }
```



Alinhamento Central Perfeito

A seguir é mostrada uma forma simples, utilizando flexbox, de resolver um problema muito comum durante a construção de layouts de páginas web, que é o alinhamento central de um elemento, tanto na vertical, quanto na horizontal.

Para tal basta utilizar ambas as propriedades "***justify-content***" e "***align-items***", configuradas com o valor "***center***". Veja no exemplo a seguir:

Arquivo: ./css/exemplos/flexbox/ex_flexbox26.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           .flex-container {
               display: flex;
               height: 300px;
               justify-content: center;
               align-items: center;
               background-color: orange;
           }
           .flex-container > div {
               background-color: wheat;
               width: 100px;
```

```
            height: 100px;
        }
    </style>
</head>
<body>
    <div class="flex-container">
        <div></div>
    </div>
</body>
</html>
```

Itens Flexíveis - Propriedades (*Flex Items*)

Os elementos que são filhos diretos de um contêiner flexível tornam-se, automaticamente, itens flexíveis (*Flex Items*). Os itens flexíveis possuem as seguintes propriedades:

PROPRIEDADE	DESCRIÇÃO
align-self	Especifica o alinhamento de um item flexível (substitui a propriedade align-items do contêiner flexível).
flex	Forma de escrita abreviada para as propriedades flex-grow,

	flex-shrink e flex-basis.
flex-basis	Especifica o comprimento inicial de um item flexível.
flex-grow	Especifica a proporção em que um item flexível deve crescer em relação ao restante dos itens flexíveis que estão no mesmo contêiner.
flex-shrink	Especifica a proporção em que um item flexível deve encolher em relação ao restante dos itens flexíveis que estão no mesmo contêiner.
order	Especifica a ordem de distribuição dos itens flexíveis que estão no mesmo contêiner.

A propriedade *order*

A propriedade "*order*" permite especificar a ordem de distribuição dos itens que estão no mesmo contêiner. A propriedade "***order***" deve assumir, obrigatoriamente, valores inteiros. Por padrão ela possui o valor 0.

Arquivo: ./css/exemplos/flexbox/ex_flexbox27.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           .flex-container {
               display: flex;
               align-items: stretch;
               background-color: orange;
           }
           .flex-container > div {
               background-color: wheat;
               width: 100px;
               margin: 10px;
               text-align: center;
               line-height: 75px;
               font-size: 30px;
           }
       </style>
```

```
    </head>
    <body>
        <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>
    </body>
</html>
```

A propriedade *flex-grow*

A propriedade "*flex-grow*" permite especificar a proporção que um item deve crescer em relação ao restante dos outros itens flexíveis. A propriedade "***flex-grow***" deve assumir, obrigatoriamente, valores inteiros. Por padrão ela possui o valor 0.

Arquivo: ./css/exemplos/flexbox/ex_flexbox28.html

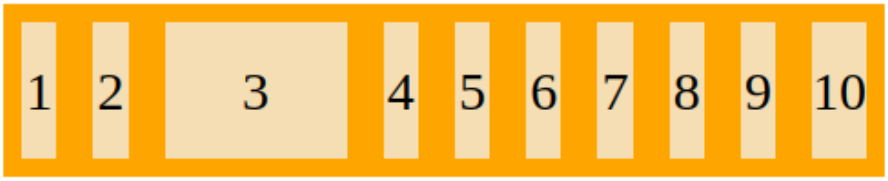
```
.flex-container > div {
    background-color: wheat;
    margin: 10px;
    text-align: center;
    line-height: 75px;
    font-size: 30px;
}

<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>
```

A propriedade *flex-shrink*

A propriedade "*flex-shrink*" permite especificar a proporção que um item deve encolher em relação ao restante dos itens flexíveis. A propriedade "***flex-shrink***" deve assumir, obrigatoriamente, valores inteiros. Por padrão ela possui o valor 1.

Arquivo: ./css/exemplos/flexbox/ex_flexbox29.html

```
.flex-container > div {
    background-color: wheat;
    width: 100px;
    margin: 10px;
    text-align: center;
    line-height: 75px;
    font-size: 30px;
}
<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>
```



A propriedade *flex-basis*

A propriedade "*flex-basis*" permite especificar o comprimento inicial de um item flexível.

Arquivo: ./css/exemplos/flexbox/ex_flexbox30.html

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

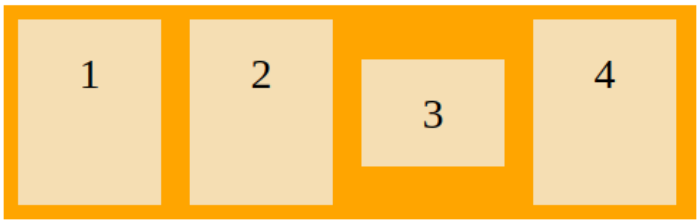
A propriedade *align-self*

A propriedade "*align-self*" permite especificar o alinhamento de um item flexível (substitui a propriedade align-items do contêiner flexível).

Arquivo: ./css/exemplos/flexbox/ex_flexbox31.html

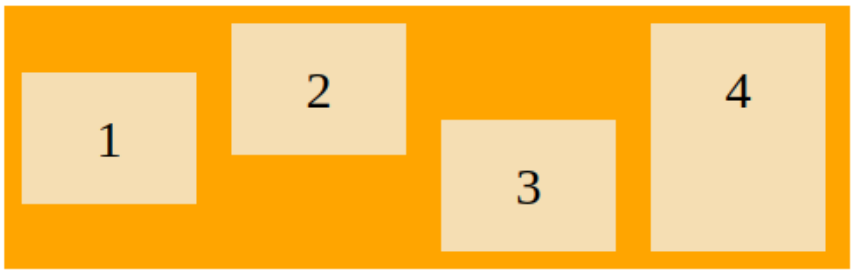
```
.flex-container {
    display: flex;
    align-items: stretch;
    height: 150px;
    background-color: orange;
```

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



Arquivo: ./css/exemplos/flexbox/ex_flexbox32.html

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



Responsive Flexbox

Como abordado na seção de Media Queries, é possível usar este conceito para criar layouts distintos, considerando os diferentes tamanhos de tela dos dispositivos (smartphones, tablets, computadores, etc).

Arquivo: ./css/exemplos/flexbox/ex_flexbox33.html

```
<!DOCTYPE html>
<html>
   <head>
       <style>
           * {
               box-sizing: border-box;
           }
           .flex-container {
               display: flex;
               flex-wrap: wrap;
               font-size: 30px;
               text-align: center;
           }
           .flex-item {
               background-color: #f1f1f1;
               padding: 10px;
               flex: 50%;
               border: 1px solid black;
           }
           /* Responsivo - Uma coluna ao invés de Duas colunas */
           @media (max-width: 800px) {
               .flex-item {
                   flex: 100%;
               }
           }
       </style>
   </head>
   <body>
       <h1>Flexbox - Responsivo</h1>
       <p>Altera o percentual do flex para diferentes tamanhos de tela.</p>
       <p><b>Redimensione o navegador para uma largura menor que 800px.</b></p>
       <div class="flex-container">
           <div class="flex-item">Menu-Esquerda</div>
```

```
            <div class="flex-item">Menu-Direita</div>
        </div>
    </body>
</html>
```

Flexbox - Responsivo

Altera o percentual do flex para diferentes tamanhos de tela.

Redimensione o navegador para uma largura menor que 800px.

Menu-Esquerda	Menu-Direita

⇩

Flexbox - Responsivo

Altera o percentual do flex para diferentes tamanhos de tela.

Redimensione o navegador para uma largura menor que 800px.

Menu-Esquerda
Menu-Direita

SASS - *Syntactically Awesome Style Sheets*

Folhas de Estilo Sintaticamente Incríveis

(https://www.w3schools.com/sass/)

(https://sass-lang.com/)

(https://marketplace.visualstudio.com/items?itemName=ritwickdey.live-sass)

O que é o SASS e suas Características?

- Uma extensão do CSS;
- Um pré-processador de CSS;
- Totalmente compatível com todas as versões de CSS;

- Reduz a repetição de CSS, economizando tempo;

Por que usar SASS?

A evolução da tecnologia, a maior complexidade na construção das folhas de estilo, normalmente extensas, tornam mais difícil sua manutenção, e a utilização de um pré-processador de CSS pode ajudar.
O Sass permite usar recursos que não existem no CSS, como variáveis, regras aninhadas, importações, herança, funções integradas e outros recursos.

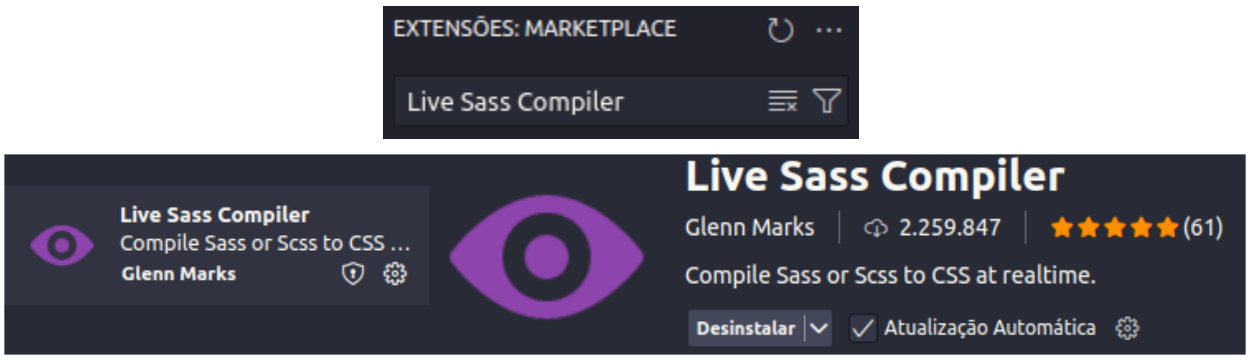
Instalação

```
Unset
npm install -g sass
```

Execução

```
Unset
sass style.scss style.css
```

Plugin VSCode - Live SASS Compiler



Variáveis SASS

As variáveis, no SASS, permitem armazenar valores que serão utilizados posteriormente, como, por exemplo, na construção de classes de estilos CSS. As variáveis podem armazenar valores como:

- strings
- numbers
- colors
- booleans
- lists
- nulls

Sintaxe - Variáveis SASS

```
Unset
$nome_variavel: valor;
```

O exemplo a seguir declara 4 variáveis que são utilizadas, posteriormente, para definir os estilos de fonte, tamanho de fonte, cor e largura de alguns componentes HTML.

Diretório: ./css/exemplos/sass/variavel/

Arquivo SCSS

```
/* Declara as Variáveis SASS*/
$myFont: Helvetica, sans-serif;
$myColor: lightblue;
$myFontSize: 26px;
$myWidth: 680px;

/* Utiliza as Variáveis SASS*/
body {
 font-family: $myFont;
 font-size: $myFontSize;
 color: $myColor;
}

#container {
 width: $myWidth;
```

```
}
```

Arquivo CSS - Gerado

```
@charset "UTF-8";
/* Declara as Variáveis SASS*/
/* Utiliza as Variáveis SASS*/
body {
 font-family: Helvetica, sans-serif;
 font-size: 26px;
 color: lightblue;
}

#container {
 width: 680px;
}/*# sourceMappingURL=style.css.map */
```

Arquivo HTML

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Variáveis SASS</title>
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <h1>Exemplo: Utilizando Variáveis SASS</h1>
   <p>Declaração e Utilização de Variáveis SASS</p>
   <div id="container">Container</div>
</body>
</html>
```

Exemplo: Utilizando Variáveis SASS

Declaração e Utilização de Variáveis SASS

Container

Regras Aninhadas

O SASS permite que você aninhe seletores CSS da mesma forma como é feito no HTML. Veja o exemplo a seguir para uma tag <nav>:

Diretório: ./css/exemplos/sass/regras_aninhadas/

Arquivo SCSS

```
$backColor: lightcoral;
$itemColor: white;
$fontSize: 21px;

/* Regras Aninhadas - SASS */
nav {
   ul {
     margin: 0;
     padding: 0;
     list-style: none;
   }
   li {
     display: inline-block;
   }
   a {
     display: block;
     padding: 6px 12px;
     text-decoration: none;
```

```
    background-color: $backColor;
    color: $itemColor;
    font-size: $fontSize;
  }

  a:hover {
      opacity: 0.7;
  }
}
```

Arquivo CSS - Gerado

```
/* Regras Aninhadas - SASS */
nav ul {
 margin: 0;
 padding: 0;
 list-style: none;
}
nav li {
 display: inline-block;
}
nav a {
 display: block;
 padding: 6px 12px;
 text-decoration: none;
 background-color: lightcoral;
 color: white;
 font-size: 21px;
}
nav a:hover {
 opacity: 0.7;
}/*# sourceMappingURL=style.css.map */
```

Arquivo HTML

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

```
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Regras Aninhadas SASS</title>
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <h1>Regras Aninhadas SASS: Navigation</h1>
   <nav>
       <ul>
         <li><a href="#">HTML</a></li>
         <li><a href="#">CSS</a></li>
         <li><a href="#">SASS</a></li>
       </ul>
     </nav>
</body>
</html>
```

Regras Aninhadas SASS: Navigation

HTML CSS SASS

Importação (@import)

O SASS possui recursos que permitem a reutilização de estilos pré-definidos, com objetivo de evitar a reescrita de código e aumentar a produtividade. Em outras palavras, é possível escrever pequenos trechos de CSS em diferentes arquivos e incluí-los em outros. Podemos ter, como exemplos: arquivo de redefinição, variáveis, cores, fontes, tamanhos de fonte, entre outros.

Assim como o CSS, o SASS também suporta a diretiva ***@import***, permitindo que o conteúdo de um arquivo seja incluído em outro arquivo. A diretiva ***@impor do CSS*** possui uma grande desvantagem, devido a problemas de desempenho, pois cria uma requisição HTTP extra cada vez que você a chama.

Por outro lado, a diretiva ***@import do SASS*** inclui importado o arquivo no CSS final, fazendo com que nenhuma chamada HTTP extra seja necessária.

Sintaxe - *@import*

```
Unset
@import nome_arquivo;
```

Por padrão, o SASS transpila todos os arquivos "***.scss***" diretamente. No entanto, quando você deseja importar um arquivo, não é necessário que ele seja transpilado. Diante deste contexto, o SASS possui um mecanismo para evitar que arquivos de importação sejam transpilados desnecessariamente. Para tal, basta iniciar o nome do arquivo com um "***sublinhado*** _" (***underline***). Arquivos nomeados dessa forma são chamados de "***partials***" no SASS.
Observe o exemplo a seguir:

Diretório: ./css/exemplos/sass/import/

Arquivo SCSS - Reset

```
html, body, ul, ol {
 margin: 0;
 padding: 0;
}
```

Arquivo SCSS - Importação

```
@import "reset";

body {
 font-family: Helvetica, sans-serif;
 font-size: 18px;
 color: lightskyblue;
 margin-left: 10px;
}
```

Arquivo CSS

```
html, body, ul, ol {
 margin: 0;
 padding: 0;
}

body {
 font-family: Helvetica, sans-serif;
 font-size: 18px;
 color: lightskyblue;
 margin-left: 10px;
}/*# sourceMappingURL=style.css.map */
```

Arquivo HTML

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Regras Aninhadas SASS</title>
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <h1>Importação SASS: @import</h1>
   <p>Utilizando o recurso @import</p>
   <ul>
       <li><a href="#">HTML</a></li>
       <li><a href="#">CSS</a></li>
       <li><a href="#">SASS</a></li>
   </ul>
</body>
</html>
```

Importação SASS: @import

Utilizando o recurso @import

HTML
CSS
SASS

SASS: @Mixin e @include

A diretiva ***@mixin*** permite a criação de código CSS para utilização em todo a página web que está sendo criado. Já a diretiva ***@include***, permite que o "***mixin***", criado, seja utilizado (incluído).

Sintaxe - *@mixin*

```
Unset
@mixin nome {
  propriedade: valor;
  propriedade: valor;
}
```

Sintaxe - *@include*

```
Unset
selector {
  @include mixin-name;
}
```

O exemplo a seguir cria um "mixin" para um texto importante:

Diretório: ./css/exemplos/sass/mixin_include/

Arquivo SCSS

```
@mixin texto-importante {
```

```
   color: white;
   background-color: darkred;
   font-size: 22px;
   font-weight: bold;
   border: 2px solid lightcoral;
   box-sizing: border-box;
}

@mixin dimensao {
   width: 100%;
   height: 50px;
   text-align: center;
   padding: 8px;
}

.aviso {
   @include texto-importante;
   @include dimensao;
}
```

Arquivo CSS

```
.aviso {
 color: white;
 background-color: darkred;
 font-size: 22px;
 font-weight: bold;
 border: 2px solid lightcoral;
 box-sizing: border-box;
 width: 100%;
 height: 50px;
 text-align: center;
 padding: 8px;
}/*# sourceMappingURL=style.css.map */
```

Arquivo HTML

```
<!DOCTYPE html>
```

```
<html lang="pt-br">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Mixin e Include SASS</title>
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <h1>Mixin e Include SASS: @mixin / @include</h1>
   <p class="aviso">Utilizando o recurso @mixin / @include</p>
</body>
</html>
```

Mixin e Include SASS: @mixin / @include

Utilizando o recurso @mixin / @include

SASS: @extends

A diretiva ***@extend*** permite compartilhar um conjunto de propriedades CSS de um determinado seletor para outro. A diretiva ***@extend*** é útil em situações onde elementos possuem estilos quase idênticos, que se diferem em pequenos detalhes.

O exemplo a seguir cria, primeiramente, um estilo básico (padrão) para botões. Na sequência são criados novos estilos para um botão de "Relatório" e um botão de "Enviar". Os dois botões herdam todas as propriedades CSS da classe "***.botao-basic***", através da utilização da diretiva ***@extend***.

Diretório: ./css/exemplos/sass/extend/

Arquivo SCSS

```
.botao-basico  {
   border: none;
   padding: 15px 30px;
```

```
    text-align: center;
    font-size: 21px;
    font-weight: bold;
    border-radius: 10px;
    cursor: pointer;
}

.botao-basico:hover {
    opacity: 0.6;
}
 .botao-relatorio  {
    @extend .botao-basico;
    background-color: sandybrown;
}

.botao-envio  {
    @extend .botao-basico;
    background-color: lightseagreen;
    color: white;
}
```

Arquivo CSS

```
.botao-basico, .botao-envio, .botao-relatorio {
 border: none;
 padding: 15px 30px;
 text-align: center;
 font-size: 21px;
 font-weight: bold;
 border-radius: 10px;
 cursor: pointer;
}

.botao-basico:hover, .botao-envio:hover, .botao-relatorio:hover {
 opacity: 0.6;
}

.botao-relatorio {
 background-color: sandybrown;
```

```
}

.botao-envio {
 background-color: lightseagreen;
 color: white;
}/*# sourceMappingURL=style.css.map */
```

Arquivo HTML

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Extend SASS</title>
   <link rel="stylesheet" href="style.css">
</head>
<body>
   <h1>Herança SASS: @extend</h1>
   <a nohref class="botao-relatorio">Relatório</a>
   <a nohref class="botao-envio">Enviar</a>
</body>
</html>
```

Funções SASS:

- String: https://www.w3schools.com/sass/sass_functions_string.php
- Número: https://www.w3schools.com/sass/sass_functions_numeric.php
- Lista: https://www.w3schools.com/sass/sass_functions_list.php
- Mapa: https://www.w3schools.com/sass/sass_functions_map.php

- Seletor: https://www.w3schools.com/sass/sass_functions_selector.php
- Cor: https://www.w3schools.com/sass/sass_functions_color.php