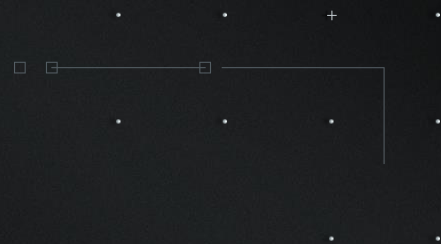




# SHIFT

 FIAP





# FRONT-END JOURNEY

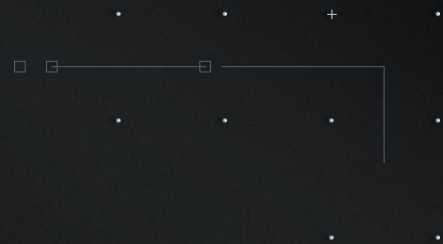
DESENVOLVIMENTO WEB COM ANGULAR & REACT



# FRONT: **PROGRAMAÇÃO**

HTML - CSS - BOOTSTRAP - SASS

---





## ISRAEL MARQUES JÚNIOR

PROFESSOR

---

- Israel é pós-graduado em Engenharia Web e trabalha com educação há 27 anos. Trabalhou no desenvolvimento de sistemas para desktop e, em seguida, migrou para a criação de aplicações para a Internet.
- Na FIAP, é professor nos cursos de: Sistemas de Informação, Análise e Desenvolvimento de Sistemas, Sistemas para Internet e Jogos Digitais para as disciplinas focadas em front end.

✉ [profisrael.copi@fiap.com.br](mailto:profisrael.copi@fiap.com.br)



# AGENDA

1

**AULA 1**

CONTEÚDO – Baixando o Editor + Introdução HTML + CSS

2

**AULA 2**

CONTEÚDO – Listas + Links + Imagens + Divs + Posicionamento

3

**AULA 3**

CONTEÚDO – Semântica – Background – Flexbox

4

**AULA 4**

CONTEÚDO – Tabelas – Formulários

5

**AULA 5**

CONTEÚDO – Design Responsivo + Media Queries + Mobile First

# AGENDA

6

**AULA 6**

CONTEÚDO – Bootstrap: Introdução + Containers + Utilitários

7

**AULA 7**

CONTEÚDO – Bootstrap: Grid + NavBar + Cards + Carousel

8

**AULA 8**

CONTEÚDO – Bootstrap: Accordion + Modal + Formulários

9

**AULA 9**

CONTEÚDO – Sass: Introdução + Instalação + Conceitos Iniciais

10

**AULA 10**

CONTEÚDO – Sass: Estilização + Variáveis + Funcionalidades

AULA 9

# Sass - CSS COM SUPERPODERES







# INTRODUÇÃO



## Introdução – o que é?

Sass é um pré-processador CSS com muitas funcionalidades que permitirão aumentar a produtividade no desenvolvimento de código CSS.

Entenda um pré-processador como um software que irá receber um código que será processado e automaticamente gerado um novo tipo. No nosso caso, escreveremos utilizando a sintaxe **Sass** ou **SCSS** e será gerado um CSS mais enxuto e de alta qualidade.

## Introdução – o que é?

Usaremos a sintaxe do **Sass** ou do **SCSS** para escrever formatações avançadas, que não são aceitas em CSS puro.

Conforme codificamos, o pré-processador gera um código CSS, com essas formatações mais avançadas, de forma que o browser possa entender e renderizar de forma correta a página

## Introdução - vantagens

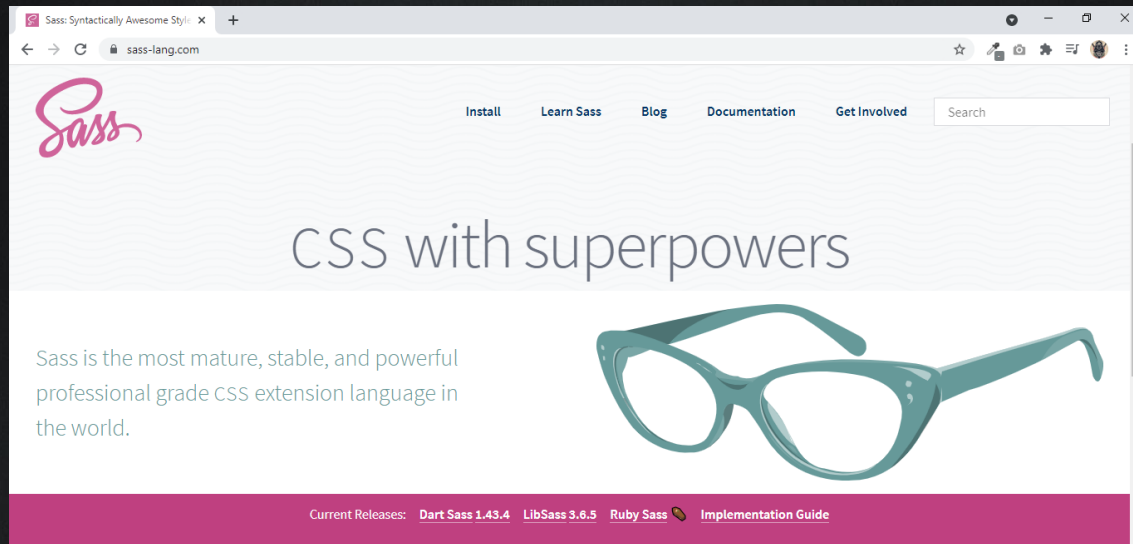
Com o uso de **Sass** temos algumas vantagens:

- Menor número de linhas;
- Código mais objetivo;
- Ganho na velocidade do desenvolvimento;
- Uso de várias funcionalidades que não existem no CSS puro;
- Amplamente utilizado no desenvolvimento de projetos web.

## Introdução – site oficial

Para acessar a  
documentação oficial  
do Sass, visite:

<https://sass-lang.com/>





## Introdução – tipos de sintaxe

Quando usamos o pré-processador, podemos utilizar duas sintaxes diferentes:

**Sass - Syntactically Awesome Style Sheets:** utiliza recuos para definir os elementos pais e filhos. Não usa chaves nem ponto e vírgula. Os arquivos que utilizam essa sintaxe devem ter a extensão .sass.

**SCSS - Sassy Cascadig Style Sheets:** utiliza sintaxe bem semelhante a uma regra CSS, inclusive com uso de chaves e ponto e vírgula. Os arquivos que utilizam essa sintaxe devem ter a extensão .scss.

## Introdução – sintaxe Sass

### HTML

```
<div>
  <h1>Olá, usando Sass</h1>
  <p>Lorem ipsum dolor...</p>
</div>
```

### Sass

```
div
  width: 90%
  h1
    font-size: 40px
  p
    font-size: 20px
```

# Introdução – sintaxe SCSS

## HTML

```
<div>
  <h1>Olá, usando Sass</h1>
  <p>Lorem ipsum dolor...</p>
</div>
```

## SCSS

```
div {
  width: 90%;
  h1 { font-size: 40px; }
  p { font-size: 20px; }
}
```

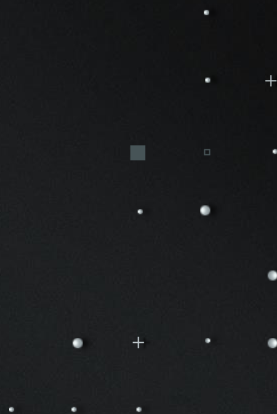
## Introdução – qual sintaxe usar?

Não importa a sintaxe utilizada, as duas sintaxes, assim que compiladas, irão retornar aquilo que mais desejamos: código CSS **enxuto** e totalmente **válido**.

Se você prefere escrever um código menor, então pode usar a sintaxe **Sass**, agora se deseja manter o mesmo padrão de uma regra CSS, com chaves e ponto e vírgula, use a sintaxe **SCSS**.



# INSTALAÇÃO

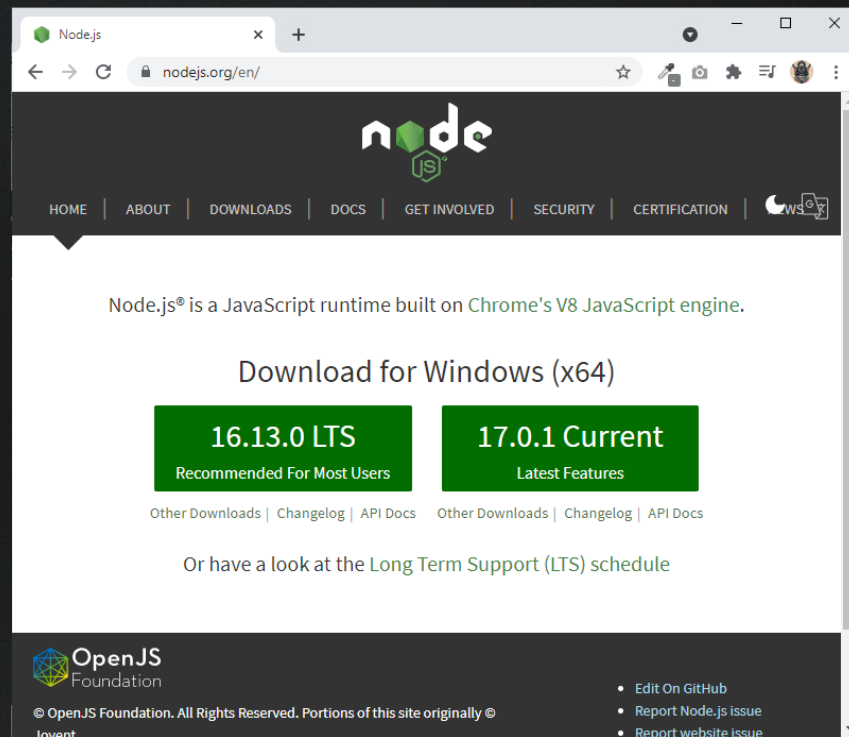




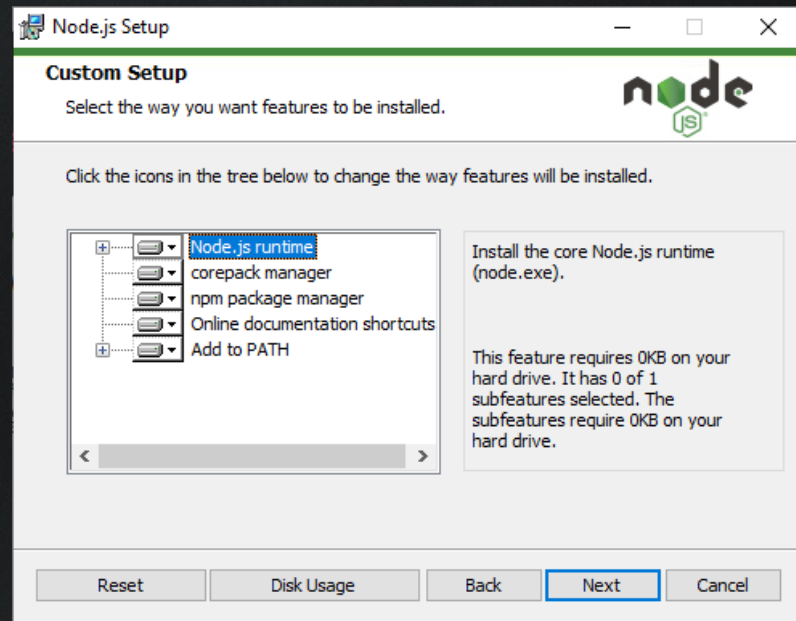
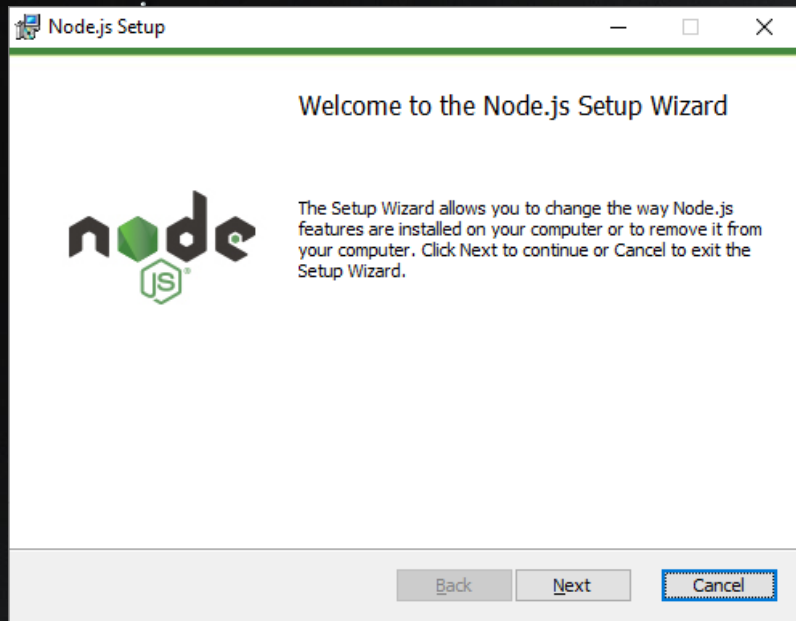
## Instalação - node.js

Para que você possa utilizar Sass,  
devemos fazer a instalação do  
node.js. Acesse:

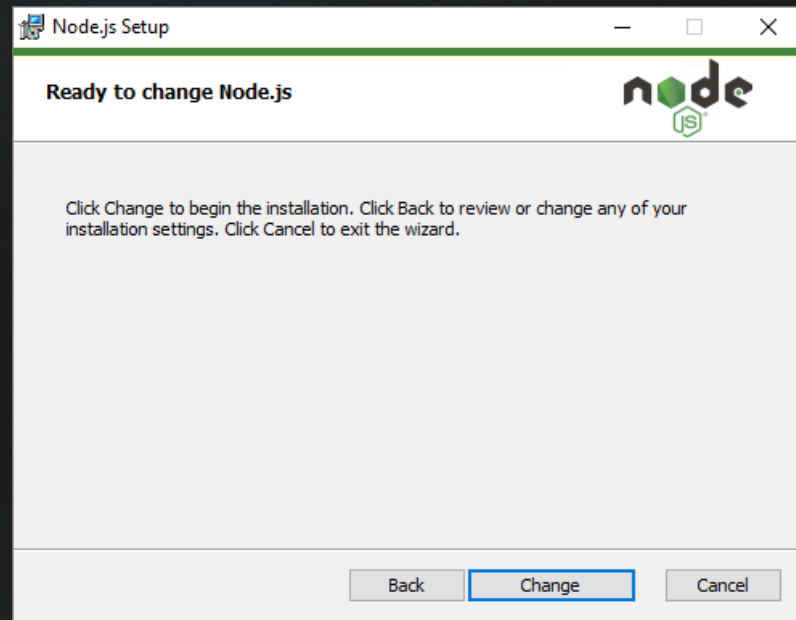
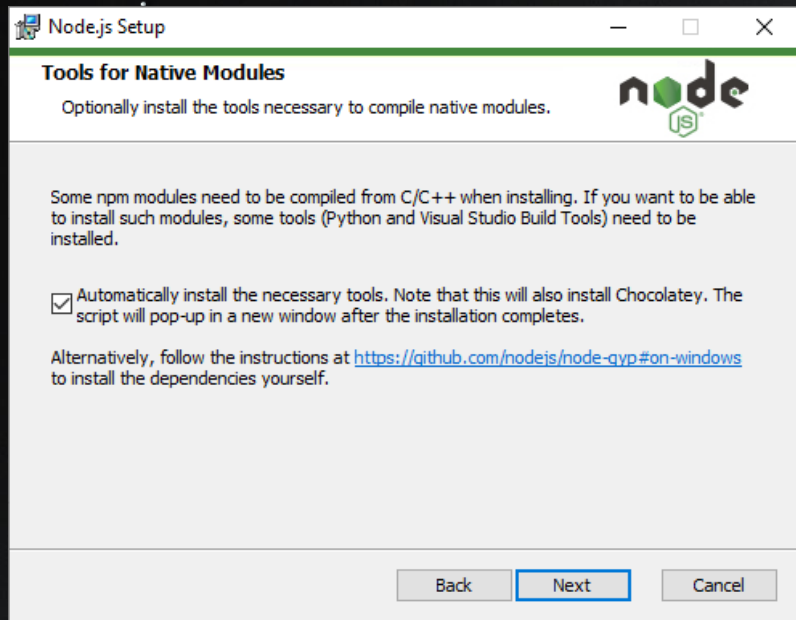
<https://nodejs.org/en/>



# Instalação - node.js



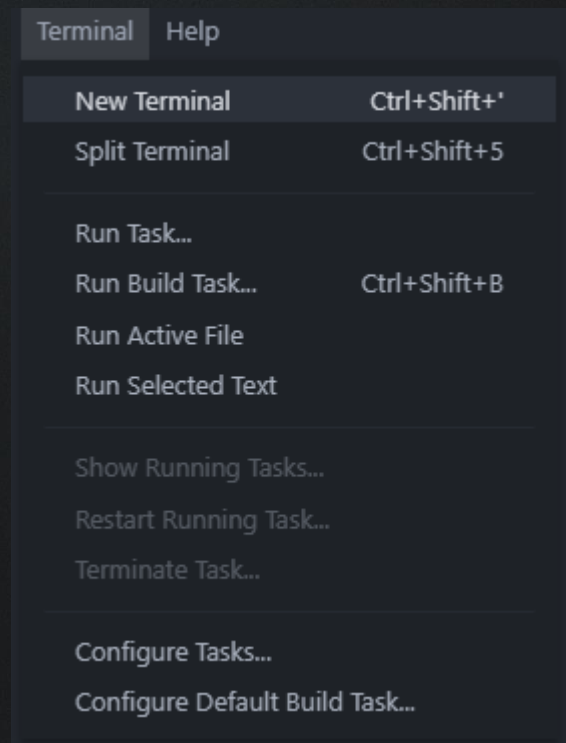
## Instalação - node.js



## Instalação – verificação node.js

Agora vamos no **VS Code** para verificar se o **node.js** foi corretamente instalado. Vamos dar uma olhada também no **NPM** – Node Package Manager, ele será usado para instalar o Sass.

- Acesse a opção Terminal no menu;
- Clique em New Terminal;



## Instalação - verificação node.js

No terminal digite: **node -v** - será exibida a versão do node instalado.

Agora digite: **npm -v** - será exibida a versão do NPM instalado.

Se aparecerem as versões, tudo está **OK**.

TERMINAL

DEBUG CONSOLE

OUTPUT

PROBLEMS

```
PS C:\Users\Israel\Desktop> node -v
v16.13.0
PS C:\Users\Israel\Desktop> npm -v
8.1.0
PS C:\Users\Israel\Desktop> 
```



## Instalação - Sass

Vamos instalar o Sass de forma global, isso quer dizer que qualquer projeto poderá utilizar os seus poderosos recursos.

No terminal digite: `npm install -g sass`

```
TERMINAL  DEBUG CONSOLE  OUTPUT  PROBLEMS
PS C:\Users\Israel\Desktop> npm install -g sass

changed 15 packages, and audited 16 packages in 958ms

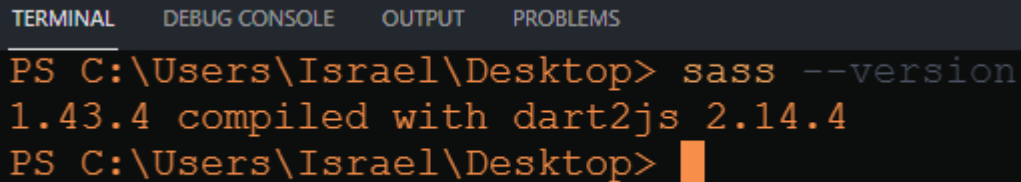
1 package is looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\Israel\Desktop> 
```

## Instalação - verificação Sass

Agora vamos ver se está tudo OK.

No terminal digite: `npm install -g sass`




TERMINAL    DEBUG CONSOLE    OUTPUT    PROBLEMS


```
PS C:\Users\Israel\Desktop> sass --version
1.43.4 compiled with dart2js 2.14.4
PS C:\Users\Israel\Desktop> █
```

## Instalação – extensões VS Code


O VS Code possui várias extensões que melhoram o desenvolvimento de aplicações em Sass. Basta fazer a procura na área de extensões do editor e fazer a instalação.



**Sass** v1.8.18  
Syler | 904.925 | ★★★★★ (6)  
Indented Sass syntax Highlighting, Autocomplete & Formatter  
Disable Uninstall ⚙️  
This extension is enabled globally.



**Live Sass Compiler** v3.0.0 Preview  
Ritwick Dey | 1.702.135 | ★★★★★ (184)  
Compile Sass or Scss to CSS at realtime with live browser reload.  
Disable Uninstall ⚙️  
This extension is enabled globally.



**Beautify css/sass/scss/less** v2.3.3  
michelemelluso | 1.015.916 | ★★★★★ (49)  
Beautify css, sass and less code (extension for Visual Studio Code)  
Disable Uninstall ⚙️  
This extension is enabled globally.

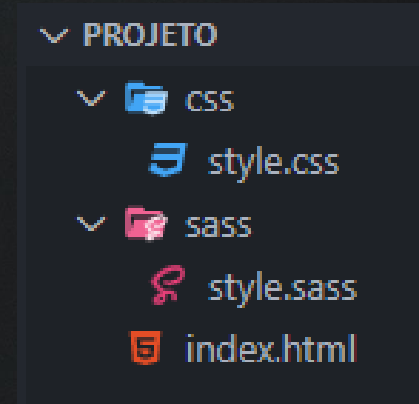


# CONCEITOS INICIAIS



## Conceitos - Pastas

Vamos criar um projeto de exemplo, para isso criamos uma pasta nova para os nossos arquivos em **Sass** que serão desenvolvidos, e uma pasta para os arquivos em **CSS**, que serão gerados automaticamente.







## Conceitos - arquivo index.html

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Projeto Exemplo</title>
  <link rel="stylesheet" href="/css/style.css">
</head>
<body>
  <h1>Usando Sass</h1>
  <p>
    Lorem ipsum dolor sit amet consectetur, adipisicing elit. Repudiandae, culpa?
  </p>
  <a href="">Veja mais...</a>
</body>
</html>
```

## Conceitos – arquivos style.sass / style.css

Com o código HTML desenvolvido, vamos escrever o código em Sass. Após isso devemos fazer a compilação do Sass para CSS, então digite a seguinte linha de comando:

```
sass sass/style.sass css/style.css
```

Pronto, será gerado o código compilado em CSS. Observe que precisamos definir as pastas de origem e destino dos arquivos.

## Conceitos – arquivos style.sass / style.css

```
body
  background-color: #000

h1
  font-size: 30px
  color: #f60

p
  font-size: 18px

a
  font-size: 18px
  text-decoration: none
```

```
body {
  background-color: #000;
}
body h1 {
  font-size: 30px;
  color: #f60;
}
body p {
  font-size: 18px;
}
body a {
  font-size: 18px;
  text-decoration: none;
}
```





## Conceitos - watch

Enquanto a verificação estiver rodando, você não conseguirá digitar nada no console. Para parar a verificação constante de alterações no código, basta pressionar **CTRL + C**.

```
PS C:\Users\Israel\Desktop\projeto> sass --watch sass/style.sass css/style.css
Sass is watching for changes. Press Ctrl-C to stop.
```

## Conceitos - comentários

Como qualquer outra linguagem, podemos comentar os códigos que estamos criando, para isso o Sass possui as seguintes opções:

- `//` - Os comentários inseridos serão exibidos apenas no arquivo `.sass`, o arquivo `.css` não exibirá esse comentário.
- `/*` - Os comentários inseridos serão exibidos nos arquivos `.sass` e `.css`.
- `/*!` - Os comentários inseridos serão exibidos nos arquivos `.sass` e `.css`, inclusive em arquivos minificados.
- `#` - Interpolação de valores em comentários. Ex: `#{5 + 5}`



## Conceitos - código minificado

Para minificar seu código Sass, na identificação do arquivo de destino do watch, mude a extensão de `.css` para `.min.css`, ao final da linha digite o comando `--style compressed`.

TERMINAL    DEBUG CONSOLE    OUTPUT    PROBLEMS

```
PS C:\Users\Israel\Desktop\projeto> sass --watch sass/style.sass:css/
style.min.css --style compressed
Sass is watching for changes. Press Ctrl-C to stop.
```

## Conceitos - nesting

Quando você faz o nesting você está aninhando os elementos da página no código Sass. A ideia é seguir a ordem de posicionamento das tags HTML, elemento pai e seus respectivos descendentes.

Cuidado para não se perder na organização do código, o conceito de containers e elementos descendentes é muito importante, por isso leia com atenção o que foi escrito no código HTML.



## Conceitos - nesting

```
<section>
  <h2>Lorem Ipsum</h2>
  <div>
    <p>Lorem ipsum dolor sit amet.</p>
    <a href="">Leia mais</a>
  </div>
</section>
```

section

background-color: #000

color: #fff

padding: 10px

h2

color: #900

div

p

font-size: 40px

a

font-size: 30px



## Conceitos - nesting

```
<section>
  <h2>Lorem Ipsum</h2>
</section>
<aside>
  <h3>Lorem Ipsum</h3>
</aside>
```

```
section, aside
  background-color: #900
  width: 50%
  height: 100px
```

```
section h2, aside h3
  background-color: #369
  width: 50%
  height: 50px
```

## Conceitos – parent selector

Podemos utilizar o símbolo **&** para fazer referência a um elemento pai, dessa forma podemos facilitar a criação de pseudo-classes, como por exemplo a definição de uma regra para o **:hover**.

Para ter mais informações e conhecer a lista completa de pseudo-classes, acesse a MDN: <https://developer.mozilla.org/pt-BR/docs/Web/CSS/Pseudo-classes>





## Conceitos – parent selector com sufixo

Podemos adicionar sufixos na criação de nossas regras, isso facilitará a criação de formatações mais complexas. Para isso, defina uma classe para o elemento e utilize esse nome nas variações de estilização para os outros elementos.

No exemplo criamos uma div com a classe `.card`. Outras duas divs serão criadas: `.card-blue` e `.card-red`. Utilizando o sufixo teremos o reaproveitamento da classe `.card`.

## Conceitos - parent selector com sufixo

```
<div class="card">
```

```
  <h2>Lorem Ipsum</h2>
```

```
  <p>Lorem ipsum dolor.</p>
```

```
  <a href="">Leia mais...</a>
```

```
</div>
```

```
<div class="card card-blue">
```

```
  <h2>Lorem Ipsum</h2>
```

```
  <p>Lorem ipsum dolor.</p>
```

```
  <a href="">Leia mais...</a>
```

```
</div>
```

```
<div class="card card-red">
```

```
  <h2>Lorem Ipsum</h2>
```

```
  <p>Lorem ipsum dolor.</p>
```

```
  <a href="">Leia mais...</a>
```

```
</div>
```

## Conceitos – parent selector com sufixo

.card

width: 20%

padding: 10px

border: 1px solid #333

&-blue

background-color: #009

&-red

background-color: #900

## Conceitos – seletor de placeholder

Imagine que em nossa aplicação existe algum elemento HTML que é repetido várias vezes, e queremos criar um padrão de estilo para todos eles. Para isso podemos usar os placeholder, criamos o padrão desejado e atribuímos aos elementos que devem ser estilizados. O placeholder só será compilado se existir a sua atribuição a algum seletor.

Para montar um placeholder, basta usar o sinal de **%** e definir um nome qualquer. A atribuição das suas regras será feita pelo comando **@extend** seguido do nome definido.

## Conceitos - seletor de placeholder

```
<button class="btn-blue">Botão 1</button>  
<button class="btn-red">Botão 2</button>
```

.btn-blue

background-color: #009

@extend %btn-base

.btn-red

background-color: #900

@extend %btn-base

%btn-base

padding: 10px 20px

border: 1px solid #000

border-radius: 7px

margin: 10px

color: #fff





# OBRIGADO



## FIAP

Copyright © 2021 | Professor Israel Marques Cajai Junior

Todos os direitos reservados. Reprodução ou divulgação total ou parcial deste documento, é expressamente proibido sem consentimento formal, por escrito, do professor/autor.

SHIFT

FIAP