

# VEM SER

**HTML e CSS**

Aula 06 - SASS

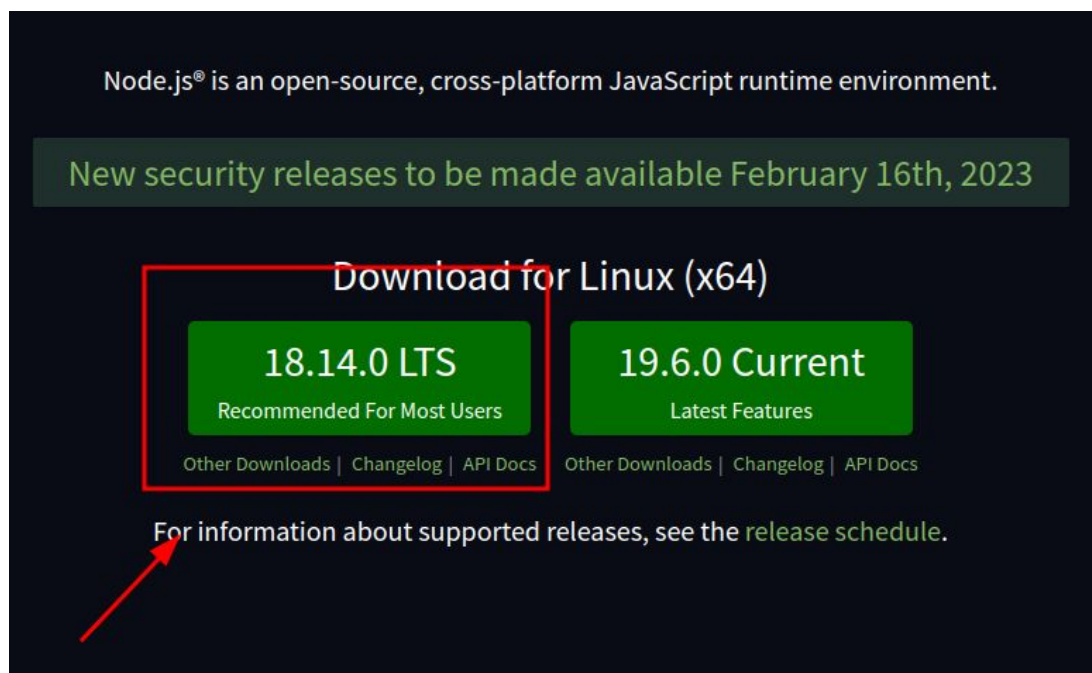


# Antes do SASS

Vamos entender o que é Node e npm.

# Instalando Node

Baixem a versão LTS no link: <https://nodejs.org/en/>



Node.js® is an open-source, cross-platform JavaScript runtime environment.

New security releases to be made available February 16th, 2023

## Download for Linux (x64)

18.14.0 LTS	19.6.0 Current
Recommended For Most Users	Latest Features
<a href="#">Other Downloads</a>   <a href="#">Changelog</a>   <a href="#">API Docs</a>	<a href="#">Other Downloads</a>   <a href="#">Changelog</a>   <a href="#">API Docs</a>

For information about supported releases, see the [release schedule](#).

Após isso, rodem os seguintes comandos no terminal:

```
PROBLEMAS  SAÍDA  CONSOLE DE DEPURAÇÃO  TERMINAL  COMENTÁRIOS

● PS D:\web_projects\teste> node -v
● v18.16.0
  PS D:\web_projects\teste> npm -v
  9.5.1
○ PS D:\web_projects\teste> 
```



# Instalando sass

1. Certifique-se de que possui o node e npm instalados;
2. No seu terminal rode o comando: `npm install -g sass`

Documentação de instalação: <https://sass-lang.com/install>

# Convertendo o arquivo SCSS para CSS

Rode o comando: `sass --watch input.scss output.css`

**input** = nome do arquivo que contém SCSS/SASS

**output** = nome do arquivo que será gerado a partir do input

```
rafael@rafael MINGW64 /d/web_projects/teste  
$ sass --watch styles.scss style.css
```

Esta **conversão será em tempo real**. O sass irá monitorar alterações no arquivo de input para gerar um novo css .



**SASS**



# O que é?

**SASS** é um acrônimo para (*Syntactically Awesome Style Sheets*), em tradução livre: **folha de estilo sintaticamente incrível**.

Ele é um dos principais pré-processadores de CSS atualmente.



# SASS vs SCSS

## .SCSS

```
.button {  
  background: cornflowerblue;  
  border-radius: 5px;  
  padding: 10px 20px;  
  
  &:hover {  
    cursor: pointer;  
  }  
  
  &:disabled {  
    cursor: default;  
    background: grey;  
    pointer-events: none;  
  }  
}
```

## .sass

```
.button  
  background: cornflowerblue  
  border-radius: 5px  
  padding: 10px 20px  
  
  &:hover  
    cursor: pointer  
  
  &:disabled  
    cursor: default  
    background: grey  
    pointer-events: none
```

# Aninhamento

Vamos imaginar que temos o seguinte código HTML:

```
1 <div class="container">
2   <div class="card"></div>
3 </div>
```

Com o CSS puro:

```
1 .container .card {
2
3 }
```



```
1  .container {  
2    width: 500px;  
3    height: 500px;  
4    background-color: red;  
5  }  
6  
7  .container .card {  
8    background-color: blue;  
9    width: 100px;  
10   height: 100px;  
11 }
```

# Aninhamento com SASS

```
1  .container {  
2    width: 500px;  
3    height: 500px;  
4    background-color: red;  
5  
6    .card {  
7      background-color: blue;  
8      width: 100px;  
9      height: 100px;  
10   }  
11 }
```

# Variáveis

No CSS puro:

```

1  :root {
2    --rosa-vem-ser: #b53795;
3    --azul-vem-ser: #3d66b7;
4  }
5
6  .container {
7    width: 500px;
8    height: 500px;
9    background-color: var(--azul-vem-ser);
10 }
11
12 .container .card {
13   background-color: var(--rosa-vem-ser);
14   width: 100px;
15   height: 100px;
16 }

```

# Variáveis no SASS

```
1 $rosa-vem-ser: #b53795;  
2 $azul-vem-ser: #3d66b7;  
3  
4 .container {  
5     width: 500px;  
6     height: 500px;  
7     background-color: $azul-vem-ser;  
8  
9     .card {  
10         background-color: $rosa-vem-ser;  
11         width: 100px;  
12         height: 100px;  
13     }  
14 }
```

# Reutilizando código com CSS puro

Código mal aproveitado:

```

1  .container {
2    width: 500px;
3    height: 500px;
4    background-color: #3d66b7;
5    display: flex;
6  }
7  .container .card {
8    background-color: #b53795;
9    width: 100px;
10   height: 100px;
11 }
12 .container .outro-card {
13   background-color: #6FEDD6;
14   width: 100px;
15   height: 100px;
16 }

```

Código melhor aproveitado:

```

1  .container {
2    width: 500px;
3    height: 500px;
4    background-color: #3d66b7;
5    display: flex;
6  }
7
8  .container .card, .container .outro-card {
9    width: 100px;
10   height: 100px;
11 }
12
13 .container .card {
14   background-color: #b53795;
15 }
16
17 .container .outro-card {
18   background-color: #6FEDD6;
19 }
20

```

# Reutilizando código com SASS: mixin

```

1  @mixin tamanho-card {
2    width: 100px;
3    height: 100px;
4  }

```

```

1  .card {
2    background-color: $rosa-vem-ser;
3    @include tamanho-card();
4  }
5
6  .outro-card {
7    background-color: $algum-verde;
8    @include tamanho-card();
9  }

```



# Mixin com parâmetros

```
1 @mixin tamanho-card($cor-do-card) {  
2   width: 100px;  
3   height: 100px;  
4   background-color: $cor-do-card;  
5 }  
6
```

```
1 .card {  
2   @include tamanho-card($rosa-vem-ser);  
3 }  
4  
5 .outro-card {  
6   @include tamanho-card($algum-verde);  
7 }
```

# Seletor '&' no SASS

```
1  .card {  
2    @include tamanho-card($rosa-vem-ser);  
3  
4    @keyframes girar {  
5      from { rotate: 0;}  
6      to { rotate: 45deg;}  
7    }  
8  
9    &:hover {  
10     animation: girar 3s forwards;  
11   }  
12 }
```

# Cálculos no SASS



```
1 $tamanho-do-texto: 2rem;
```



```
1 .card {
2   @include tamanho-card($rosa-vem-ser);
3
4   @keyframes girar {
5     from { rotate: 0; }
6     to { rotate: 45deg; }
7   }
8
9   &:hover {
10    animation: girar 3s forwards;
11  }
12
13  p {
14    font-size: $tamanho-do-texto;
15  }
16 }
```



```
1 .outro-card {
2   @include tamanho-card($algum-verde);
3
4   p {
5     font-size: $tamanho-do-texto / 2;
6   }
7 }
```



Let's *Tech Up Together*