



# Pemrograman Sisi Cleint **SCSS & SASS**

Muhammad Syaifur Rohman, S.Kom, M.CS

# Definisi



## FYI

SCSS (Sassy CSS) adalah preprocessor CSS yang memungkinkan Anda menulis CSS yang lebih bersih, lebih modular, dan lebih mudah untuk dipelihara. SCSS menawarkan fitur-fitur seperti variabel, nesting, mixins, inheritance, dan lain-lain, yang tidak tersedia dalam CSS biasa. Ini membantu dalam membuat stylesheet yang lebih terorganisir dan mengurangi pengulangan kode.

# SCSS vs SASS

## SCSS

```
// Variabel
$primary-color: #007bff;
$hover-color: darken($primary-color, 10%);

// Mixin
@mixin button-style {
  padding: 0.5rem 1rem;
  border: none;
  border-radius: 0.25rem;
  cursor: pointer;
  &:hover {
    background-color: $hover-color;
  }
}

// Button
.btn-primary {
  background-color: $primary-color;
  color: white;
  @include button-style;
}

// Inheritance
.btn-secondary {
  @extend .btn-primary;
  background-color: lighten($primary-color, 10%);
}
```



```
// Variabel
$primary-color: #007bff
$hover-color: darken($primary-color, 10%)

// Mixin
=button-style
  padding: 0.5rem 1rem
  border: none
  border-radius: 0.25rem
  cursor: pointer
  &:hover
    background-color: $hover-color

// Button
.btn-primary
  background-color: $primary-color
  color: white
  +button-style

// Inheritance
.btn-secondary
  @extend .btn-primary
  background-color: lighten($primary-color, 10%)
```

# SCSS

## Apa itu ?

SCSS adalah sintaks baru dari Sass (Syntactically Awesome Stylesheets), yang menggunakan format file .scss. SCSS sepenuhnya kompatibel dengan CSS, namun menambahkan kekuatan Sass ke CSS biasa.

## Mengapa ?

Dengan SCSS, Anda dapat membuat kode CSS yang lebih kering (DRY - Don't Repeat Yourself), lebih mudah dibaca, dan lebih mudah untuk dipelihara.



## Variabel

Menyimpan warna, font, atau nilai lainnya untuk digunakan kembali di seluruh stylesheet.



```
$primary-color: #333;  
body {  
    color: $primary-color;  
}
```

# Fitur Utama SCSS



## Nesting

Memungkinkan Anda untuk menulis CSS dalam cara yang mengikuti struktur HTML.

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



## Partials dan Import

Membagi CSS menjadi beberapa file untuk memudahkan pengelolaan kode.

- Partials: File SCSS dimulai dengan `_`, tidak akan dikompilasi menjadi file CSS sendiri.
- Import: Menggabungkan file `_partials` ke dalam file SCSS utama.



```
// _reset.scss  
* { margin: 0; padding: 0; }  
  
// styles.scss  
@import 'reset';  
body { font: 100% Helvetica, sans-serif; }
```

# Fitur Utama SCSS

## Mixins

Grup deklarasi yang dapat digunakan kembali di seluruh stylesheet.

```
● ● ●  
  
@Mixin border-radius($radius) {  
  -webkit-border-radius: $radius;  
  -moz-border-radius: $radius;  
  -ms-border-radius: $radius;  
  border-radius: $radius;  
}  
  
.box { @include border-radius(10px); }
```



## Extend/Inheritance

Berbagi serangkaian properti CSS dari satu pemilih ke pemilih lain.



```
.btn {  
  padding: 10px 15px;  
  border: none;  
  font-size: 16px;  
}  
  
.btn-primary {  
  @extend .btn;  
  background-color: blue;  
  color: white;  
}
```

# Fitur Utama SCSS

## Operator

Melakukan operasi matematika dengan CSS.



```
.container {  
    width: 100%;  
    margin: 0 auto;  
    padding: (20px / 2);  
}
```

## Cara Kerja dengan SCSS

- **Setup Environment:** Pastikan Anda memiliki preprocessor SCSS yang terinstal, seperti Sass CLI, Node-sass, atau melalui task runner seperti Gulp atau Webpack.
- **Kompilasi SCSS ke CSS:** Kode SCSS perlu dikompilasi menjadi CSS agar bisa digunakan oleh browser. Contoh kompilasi menggunakan Sass CLI:  
`sass input.scss output.css`

# Best Practice



**Gunakan variabel untuk warna, font-size, dan properti yang sering digunakan.**



**Manfaatkan nesting dengan bijak untuk menghindari spesifisitas yang berlebihan.**



**Pisahkan kode Anda menjadi partials untuk memudahkan pengelolaan.**



**Gunakan mixins untuk styling yang perlu digunakan ulang tanpa mengulang kode.**



**Hindari penggunaan @extend secara berlebihan karena dapat membuat CSS akhir lebih besar dan lebih sulit untuk dipahami.**

S A S  
S V S



# Definisi



## FYI

SASS adalah preprocessor CSS yang memungkinkan Anda menggunakan fitur-fitur seperti variabel, nested rules, mixins, inheritance, dan lain-lain, yang tidak tersedia dalam CSS murni. Dengan SASS, pengembangan CSS menjadi lebih dinamis, terorganisir, dan efisien. SASS menawarkan dua sintaks: SCSS (Sassy CSS), yang menggunakan format file .scss dan lebih mirip dengan CSS, serta sintaks indentasi yang lebih lama, menggunakan format file .sass.

# CSS vs SCSS

## CSS

```
/* Menggunakan CSS modern */
:root {
    --bg-color: #f8f9fa;
    --text-color: #333;
    --padding: 20px;
}

.card {
    background: var(--bg-color);
    color: var(--text-color);
    padding: var(--padding);
    border-radius: 5px;
}

.card .card-header {
    margin-bottom: calc(var(--padding) / 2);
    font-weight: bold;
}

@media (max-width: 600px) {
    .card {
        padding: calc(var(--padding) / 2);
    }
}
```

// Menggunakan SCSS

```
$bg-color: #f8f9fa;
$text-color: #333;
$padding: 20px;
```

```
.card {
    background: $bg-color;
    color: $text-color;
    padding: $padding;
    border-radius: 5px;
```

// Nesting untuk elemen di dalam .card

```
.card-header {
```

```
    margin-bottom: $padding / 2;
    font-weight: bold;
}
```

// Media query di dalam .card

```
@media (max-width: 600px) {
```

```
    padding: $padding / 2;
}
}
```

# SASS

## Apa itu ?

SASS adalah sebuah ekstensi dari CSS yang memungkinkan Anda untuk menggunakan kekuatan pemrograman dalam stylesheet Anda, membuatnya lebih modular dan mudah untuk dipelihara.

## Mengapa ?

SASS memperkenalkan fitur-fitur baru yang mempermudah pengelolaan stylesheet yang kompleks, mengurangi pengulangan, dan mempercepat proses pengembangan.



## Variabel

Menyimpan nilai yang sering digunakan seperti warna, ukuran font, dan lainnya untuk memudahkan penggunaan ulang. sass



```
$primary-color: #3498db  
body  
color: $primary-color
```

# Fitur Utama SASS



## Nesting

Menulis CSS dengan cara yang mengikuti struktur HTML, membuat kode lebih terorganisir.

```
nav
  ul
    margin: 0
    padding: 0
    list-style: none
  li
    display: inline-block
  a
    display: block
    padding: 6px 12px
    text-decoration: none
```



## Partials dan Import

Memungkinkan Anda memisahkan stylesheet menjadi beberapa file, membuat kode lebih modular.

```
// _reset.sass
*
  margin: 0
  padding: 0

// styles.sass
@import reset
body
  font: 100% Helvetica, sans-serif
  background-color: #efefef
```

# Fitur Utama SASS

## Mixins

Mendefinisikan grup deklarasi yang dapat digunakan kembali di seluruh stylesheet.

```
● ● ●  
=border-radius($radius)  
  -webkit-border-radius: $radius  
  -moz-border-radius: $radius  
  -ms-border-radius: $radius  
  border-radius: $radius  
  
.box  
+border-radius(10px)
```



## Extend/Inheritance

Membagikan properti dari satu selector ke selector lain.

```
● ● ●  
.btn  
  padding: 10px 15px  
  border: none  
  font-size: 16px  
.btn-primary  
  @extend .btn  
  background-color: blue  
  color: white
```

# Fitur Utama SASS

## Operator

Melakukan operasi matematika langsung dalam CSS.



```
● ● ●  
.container  
width: 100%  
margin: 0 auto  
padding: 20px / 2
```

## Cara Kerja dengan SASS

- **Instalasi:** Pastikan Anda memiliki Ruby dan SASS terinstal. Instal SASS melalui RubyGems: `gem install sass`.
- **Kompilasi SCSS ke CSS:** Gunakan perintah SASS untuk mengompilasi file `.sass` atau `.scss` menjadi `.css`.
  - **sass input.scss output.css**
- atau untuk mode watching, yang otomatis mengompilasi saat ada perubahan:
  - **sass --watch input.scss:output.css**

## Best Practice



- Gunakan variabel untuk memudahkan pengelolaan warna, font, dan nilai yang sering digunakan.
- Manfaatkan fitur nesting dengan bijak untuk menghindari spesifisitas yang berlebihan.
- Pisahkan stylesheet Anda menjadi beberapa partials untuk memudahkan pengelolaan dan pemeliharaan.
- Gunakan mixins untuk mengurangi pengulangan kode.
- Pilihlah @extend dengan bijak untuk menghindari pembengkakan kode CSS yang tidak perlu.

Menggunakan SASS dalam proyek Anda memungkinkan pengelolaan stylesheet yang lebih baik dan mempercepat proses pengembangan. Dengan mengadopsi SASS, Anda dapat mengambil keuntungan dari fitur-fitur canggih



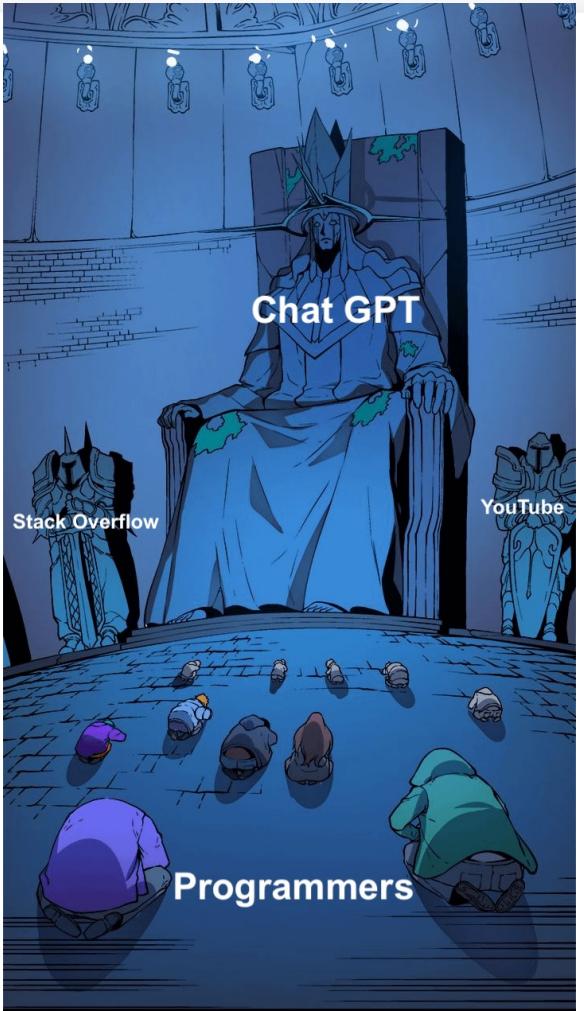
# Kesimpulan

## Saran

### Penggunaan

Memilih antara SCSS, SASS, CSS murni, Bootstrap, dan Tailwind CSS bergantung pada kebutuhan proyek, tim, dan preferensi pribadi.

- **SCSS/SASS:** Cocok untuk proyek yang membutuhkan logika styling yang kompleks dan organisasi kode yang baik.
- **CSS Murni:** Terbaik untuk proyek kecil hingga menengah yang ingin menghindari overhead dari preprocessor dan memaksimalkan kompatibilitas browser tanpa kompilasi.
- **Bootstrap:** Ideal untuk tim yang membutuhkan prototyping cepat atau proyek yang mengutamakan kecepatan pengembangan dengan UI yang konsisten dan responsif.
- **Tailwind CSS:** Pilihan terbaik untuk proyek yang menginginkan kontrol penuh atas desain dengan pendekatan utility-first, memungkinkan desain yang sangat kustom sambil mempertahankan stylesheet yang ramping.



# Thanks

Muhammad Syaifur Rohman, S.Kom, M.CS