



Modern Theming

In Angular and Material



Kobi Hari

- Freelancer
- Developer, Instructor and Consultant
- Angular, Async development, .NET core



My email: hari@applicolors.com



Courses on Udemy: <https://www.udemy.com/user/kobi-hari/>



My Angular Channel: <https://www.youtube.com/@kobihari>



GitHub Repository

<https://github.com/kobi-hari-courses/2506-angularTlv-meetup>



This lecture is **not** about

1. AI
2. Angular Material
3. **Angular...** actually



This lecture is **yes** about

1. Demo app with **AI** chat bots
2. The new approach of **Angular Material** to **theming**
3. **V20** of **Angular**... actually



“Angular Material” is known for

- Problematic code
- Hard to customize
- Breaks every few versions
- But why?



Theming

- Consistent **Graphical Concept**
- **Reuse** colors, fonts, icons



CSS Challenges

- No “variables”
- No “constants”
- No “functions”

* at least not in 2015

Copy – Paste



Sass to the rescue?

- It has Variables, Constants, Functions
- But... It's only in Compile time
- It generates the same "bad" CSS

styles.scss

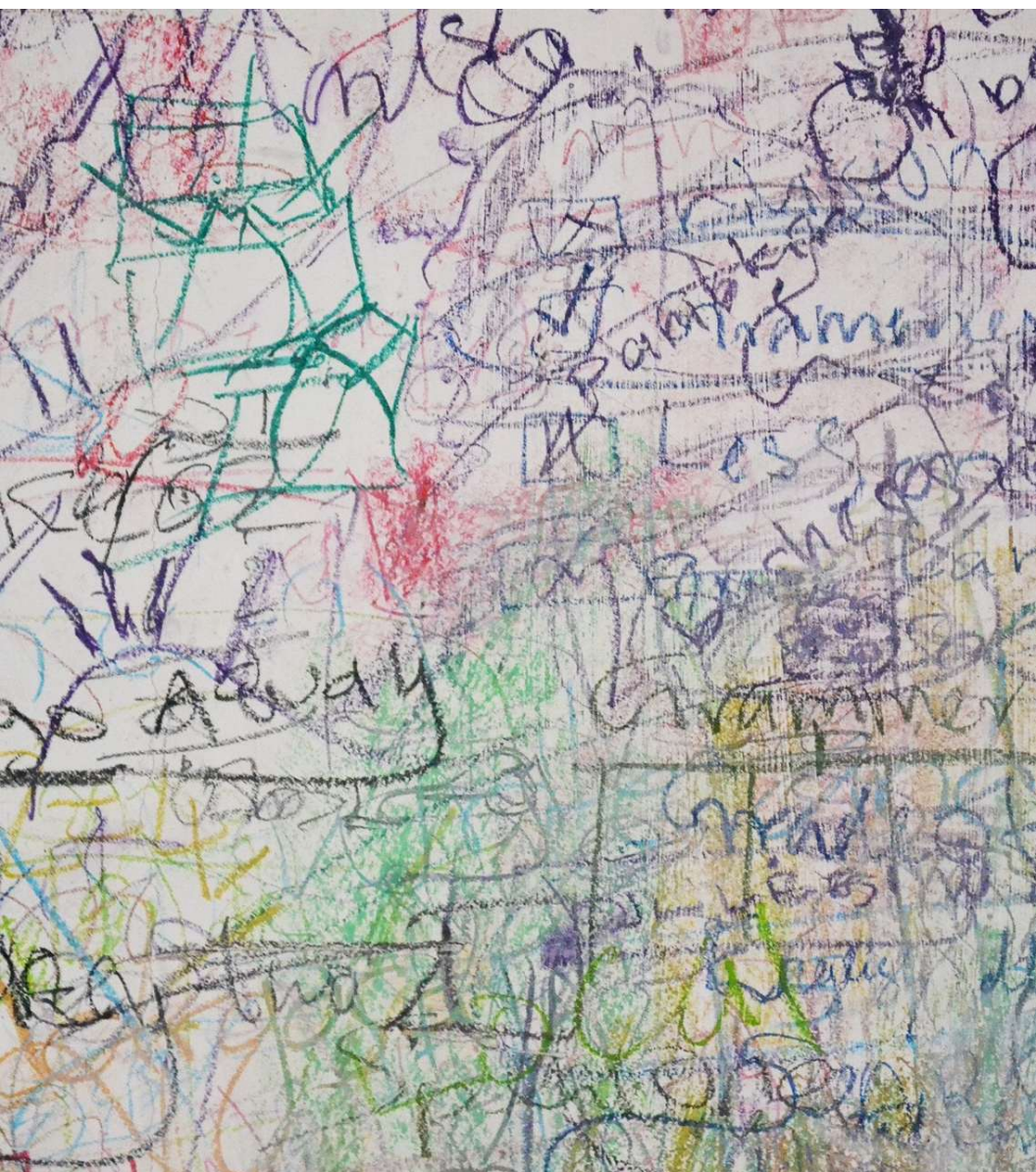
```
@use '@angular/material' as mat;

@include mat.core();

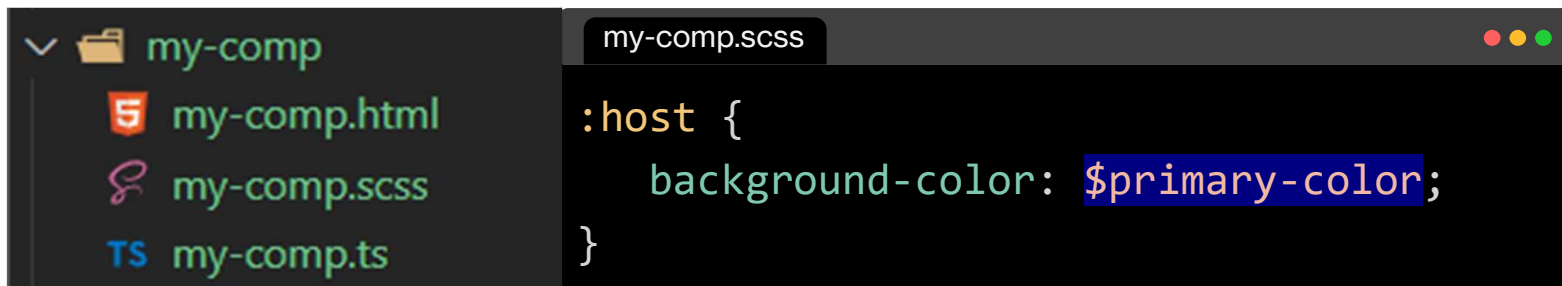
$my-primary: mat.define-palette(mat.$indigo-palette);
$my-accent: mat.define-palette(mat.$pink-palette);
$my-warn: mat.define-palette(mat.$red-palette);

$my-theme: mat.define-light-theme((
  color: (
    primary: $my-primary,
    accent: $my-accent,
    warn: $my-warn,
  ),
  typography: mat.define-typography-config(),
  density: 0,
));

@include mat.all-component-themes($my-theme);
```



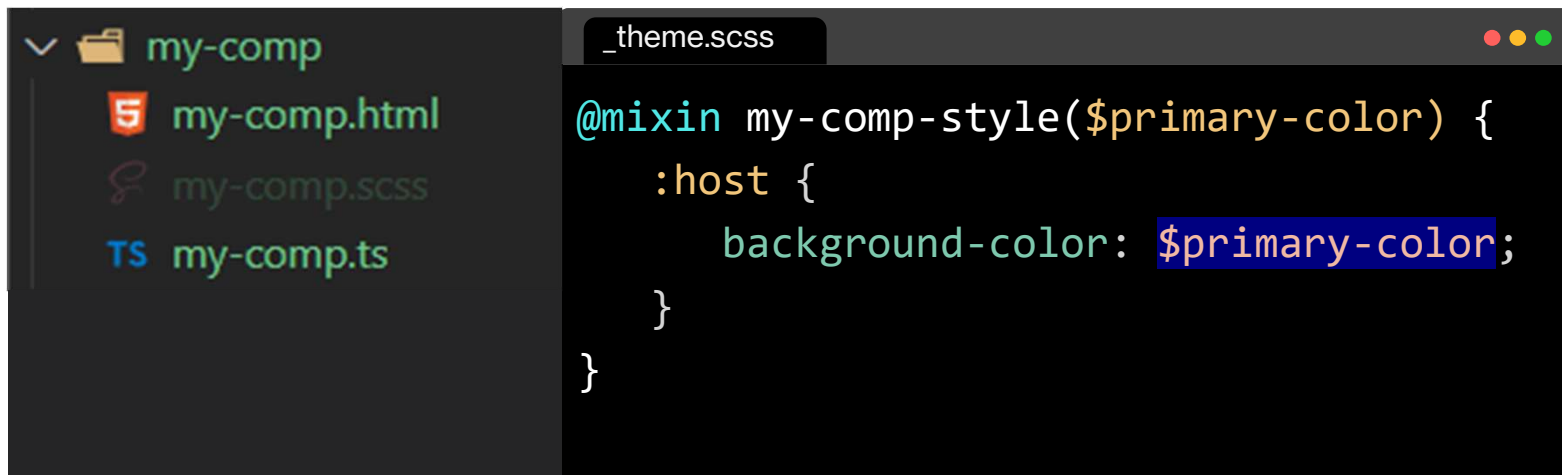
Why like this?



```
:host {  
  background-color: $primary-color;  
}
```

You are an Angular Component Library Developer
You create component SCSS files
And need to use a **variable** the user defines
In **his** `styles.scss`

How do you get the `$primary-color` variable?



```
@mixin my-comp-style($primary-color) {  
  :host {  
    background-color: $primary-color;  
  }  
}
```

Provide the styling as mixin
Accept the theme colors as parameter
Yack!

styles.scss

```
@use '@angular/material' as mat;

@include mat.core();

$my-primary: mat.define-palette(mat.$indigo-palette);
$my-accent: mat.define-palette(mat.$pink-palette);
$my-warn: mat.define-palette(mat.$red-palette);

$my-theme: mat.define-light-theme((
  color: (
    primary: $my-primary,
    accent: $my-accent,
    warn: $my-warn,
  ),
  typography: mat.define-typography-config(),
  density: 0,
));

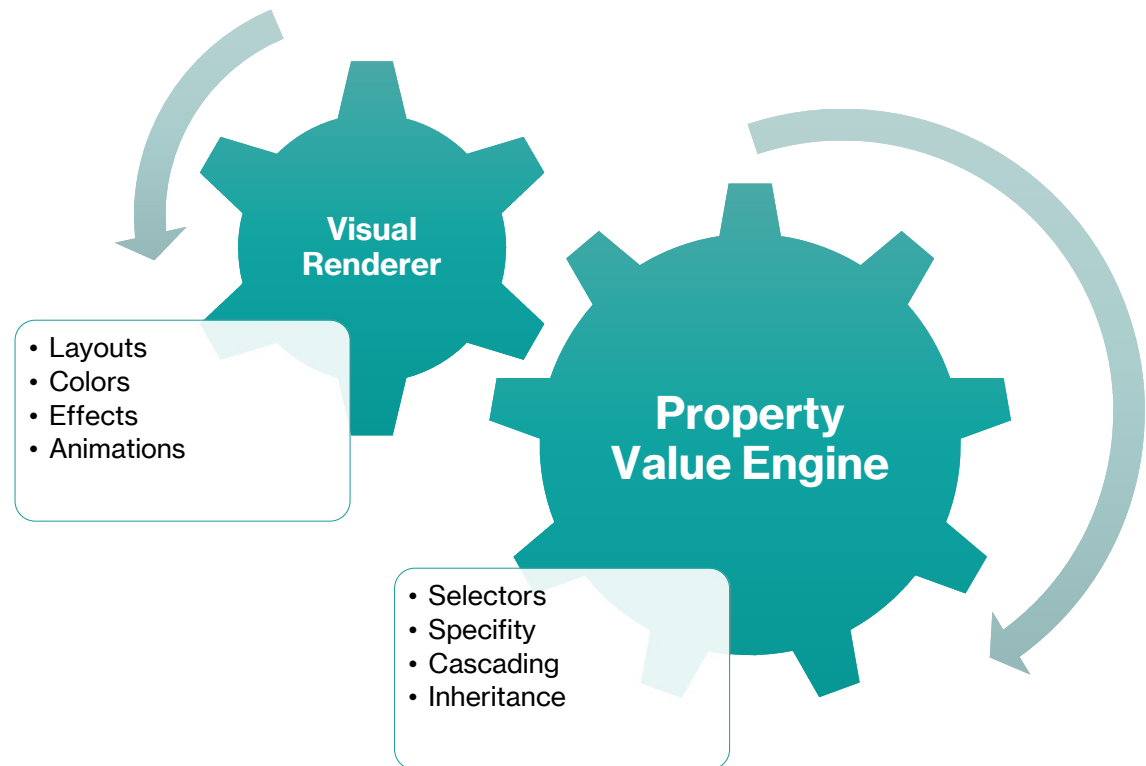
@include mat.all-component-themes($my-theme);
```




Custom Properties

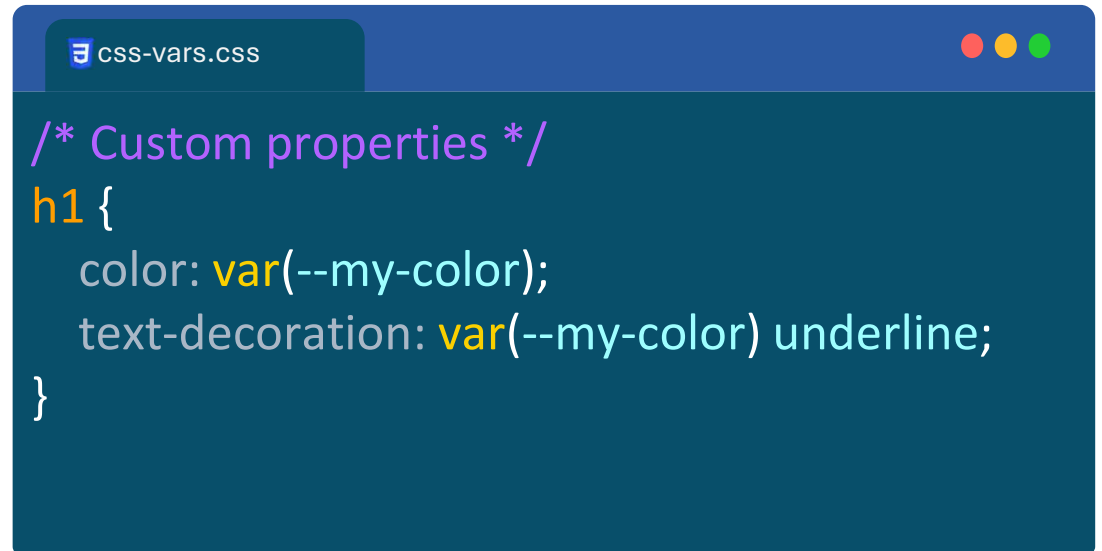
```
--my-color: magenta;  
background: var(--my-color);
```

**CSS
consists of
two parts.**





```
/* Custom properties */
:root {
  --my-prop: 50;
  --my-color: magenta;
}
```



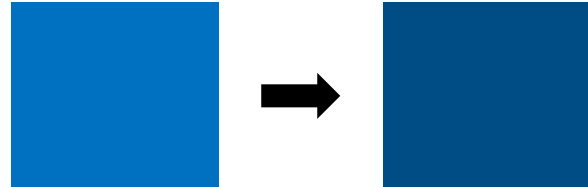
```
/* Custom properties */
h1 {
  color: var(--my-color);
  text-decoration: var(--my-color) underline;
}
```

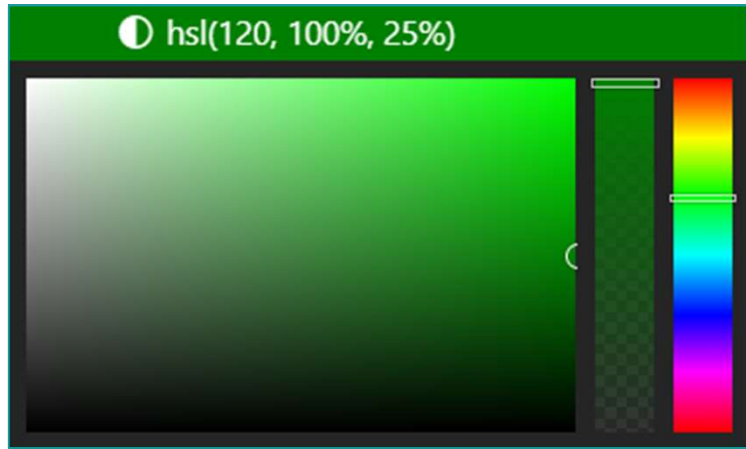


Relative Color Syntax

background:

```
hsl(from blue h s calc(l * 0.8));
```





H_{ue}

- Choose the base color on the color wheel
- From 0 degrees to 360

S_{aturation}

- Choose how intense you want it
- From 0% (grayscale) to 100% (fully saturated)

L_{ightness}

- Choose how bright you want it
- From 0% (light's off – black) to 100% (light's fully on, bright)

HSL

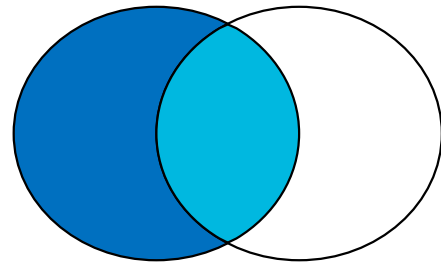
Hue, Saturation, Lightness



Mixing Colors

background:

`color-mix(in srgb, blue, white 50%);`





Color Schemes

color-scheme: light

Hello World

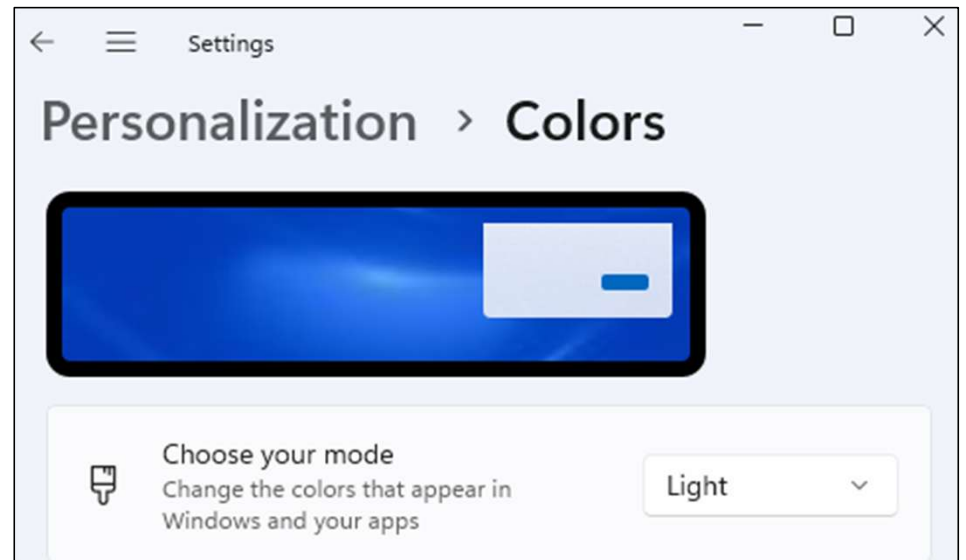
color-scheme: dark

Hello World



Color Schemes

`color-scheme: light dark; // by system`





Mixing Colors

color-scheme: light;

background:

■ light-dark(lightblue, darkblue);

color-scheme: dark;

background:

■ light-dark(lightblue, darkblue);



Angular Material 19+

- Design Tokens
- No more component Style Generations
- Each Theme supports **both** light and dark modes



Theme Setup

styles.scss

```
@use '@angular/material' as mat;  
html {  
  color-scheme: light-dark;  
  @include mat.theme(  
    color: (  
      primary: mat.$violet-palette,  
      tertiary: mat.$orange-palette  
    ),  
    typography: Roboto,  
    density: 0 ));  
}
```



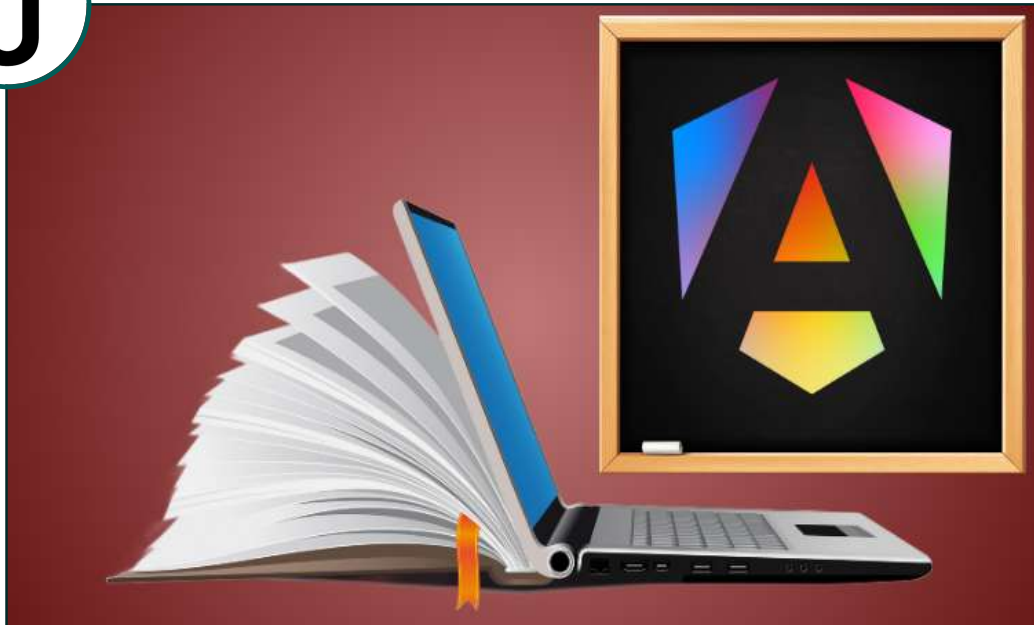
Design Tokens

System Tokens

- `--mat-sys-primary` →  #ABDBFF
- `--mat-sys-on-primary` →  #004376

Component Tokens

- `--mat-button-container-color` →  `var(--mat-sys-on-primary)`



Theming Angular & Material MD3

The missing guide

by Kobi Hari