

## Lab session 1: Flutter Installation and UI Design using Flutter

### Task-6 Visualizing dynamic color in your app – Codelab

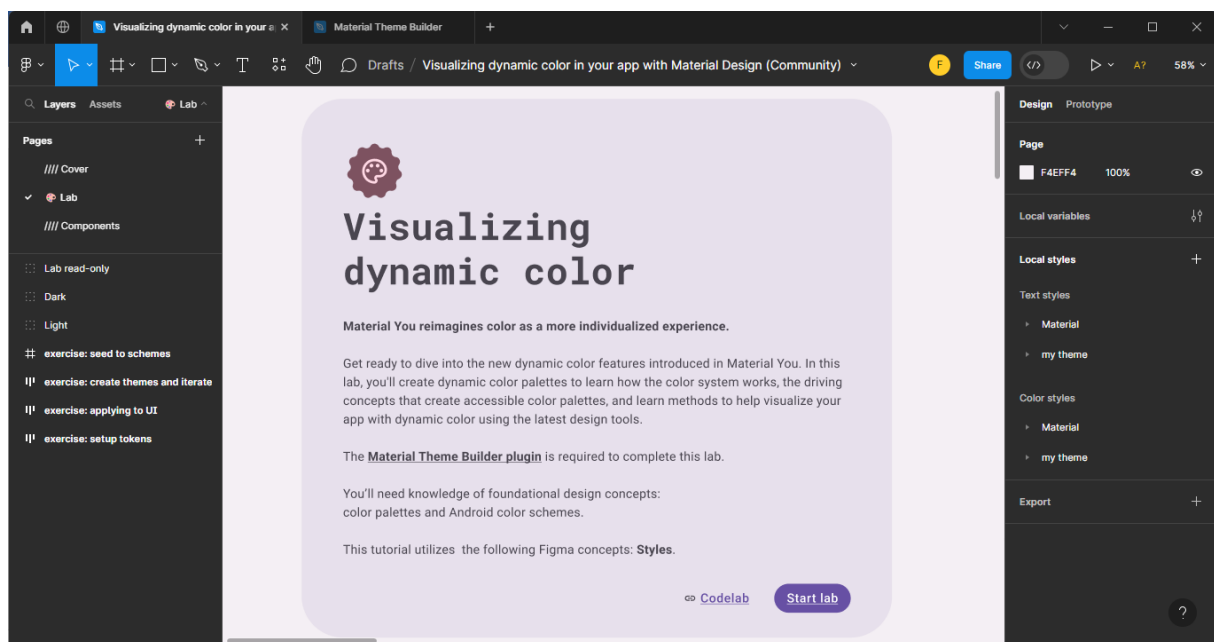
During the codlab session, we explored the new dynamic color features introduced with Material You. The focus was on creating dynamic color palettes and understanding the underlying concepts that contribute to accessible color schemes. Additionally, we learned how to apply user-generated colors to our app and utilized the latest design tools to visualize our app with dynamic colors.

In this session, we covered the following topics:

- Understanding the updates in Material Design color
- Applying user-generated colors to our app
- Utilizing tools such as Figma and the Material Theme Builder plugin
- Building upon foundational design concepts, including color palettes and Android color schemes

To participate in the codlab, we needed the following:

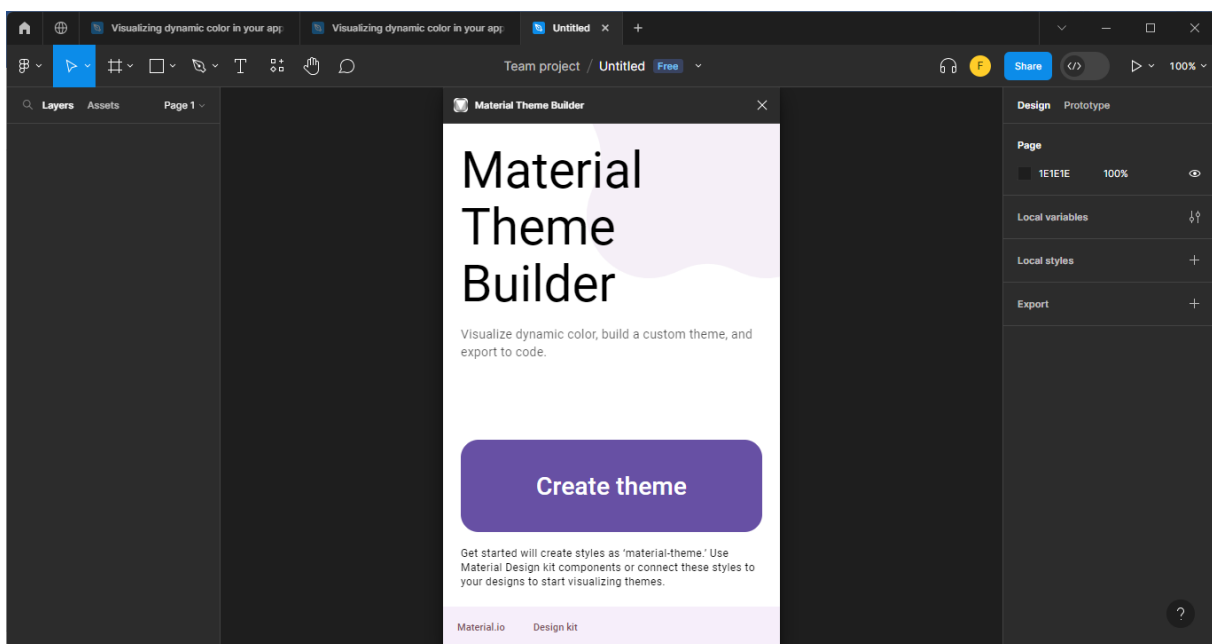
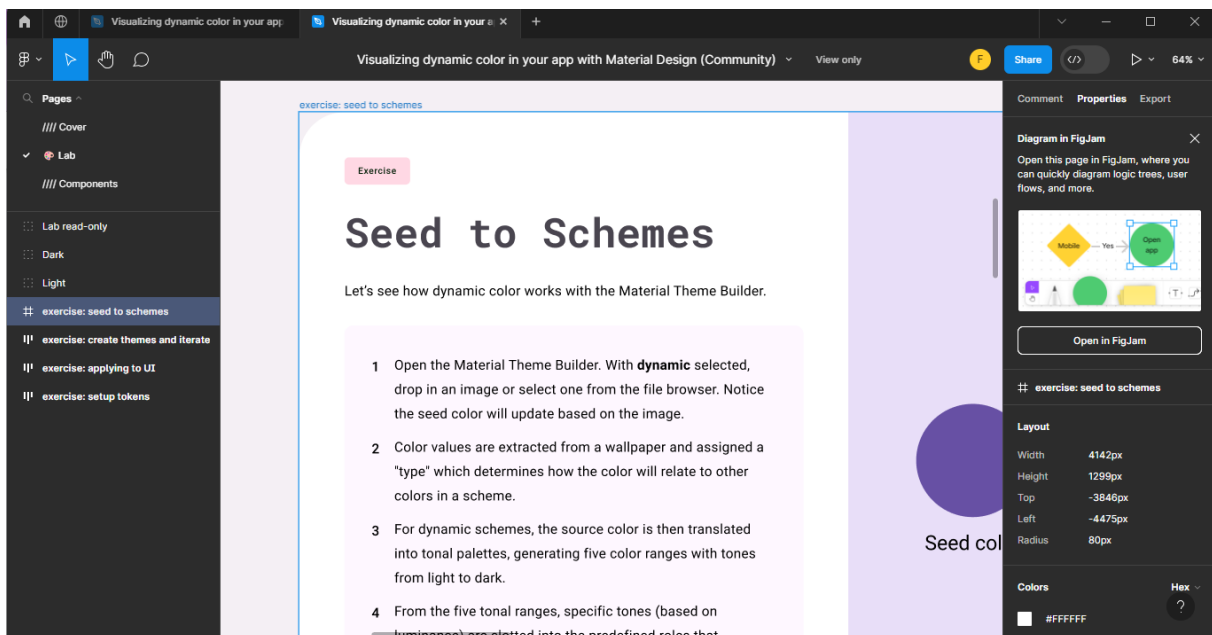
- A Figma Account
- The Figma Dynamic color Designlab file
- The Figma plugin Material Theme Builder

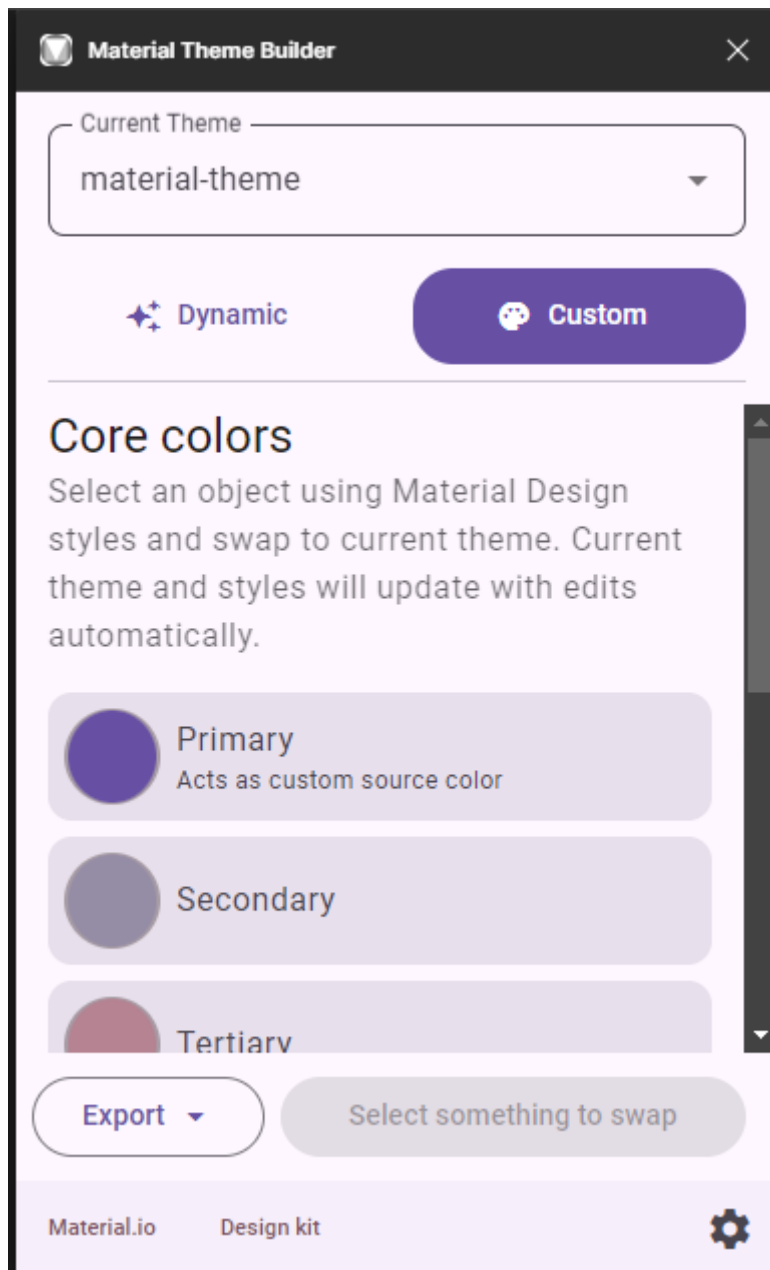


After the introduction, we accessed the 'Visualizing dynamic color in your app with Material Design' Figma file. This file served as a visual resource for understanding and implementing dynamic color concepts in our app. We explored the concept of dynamic color, which allows colors to adapt and change based on various factors. Additionally, we delved into color concepts such as luminance, which refers to the brightness of a color, and tonal palettes, which provide a range of colors within a specific hue. These discussions helped us gain a deeper understanding of color principles and how they can be applied effectively in our app design.

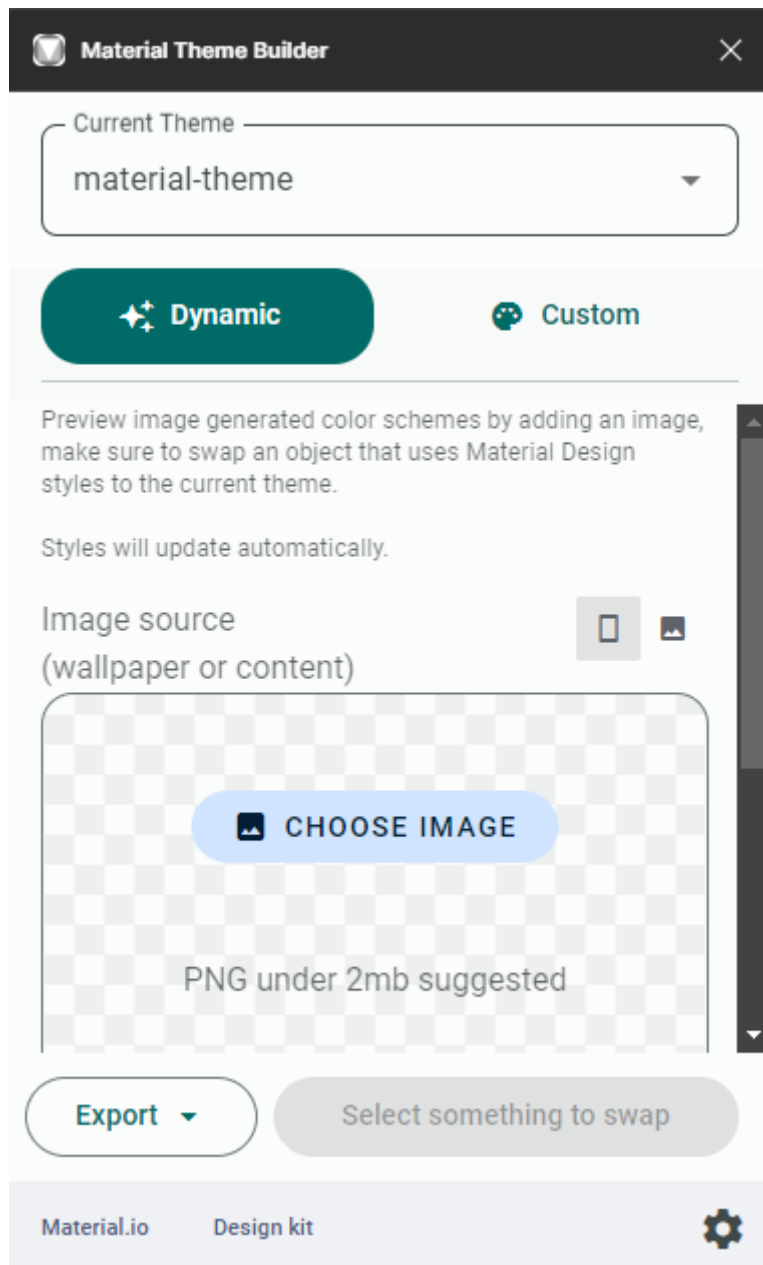
## Extracting colours

Next, the codlab introduced us to material theme design. We learned about the process of extracting colors and creating color schemes using the Material Theme Builder.

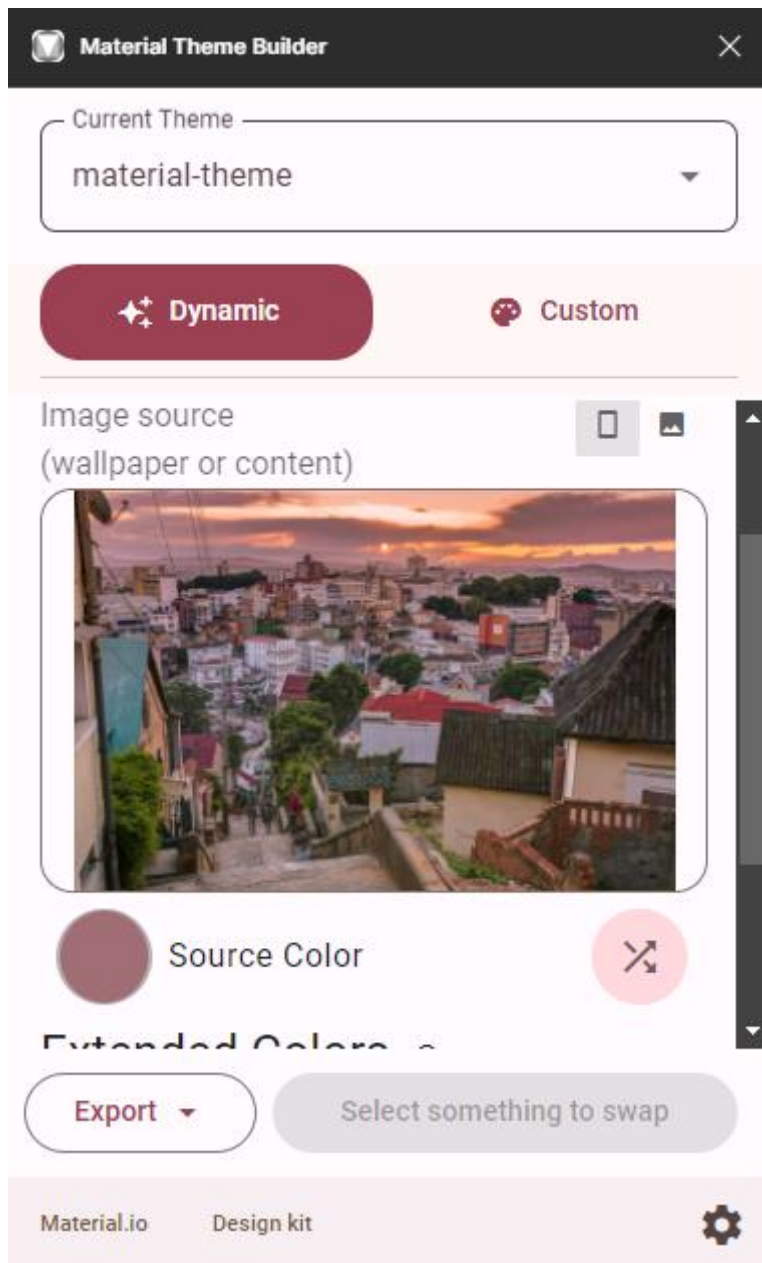




In this step, we utilized the Material Theme Builder tool. With the dynamic color option selected, we either dropped in an image or selected one from the file browser. We observed that the seed color automatically updated based on the image chosen.

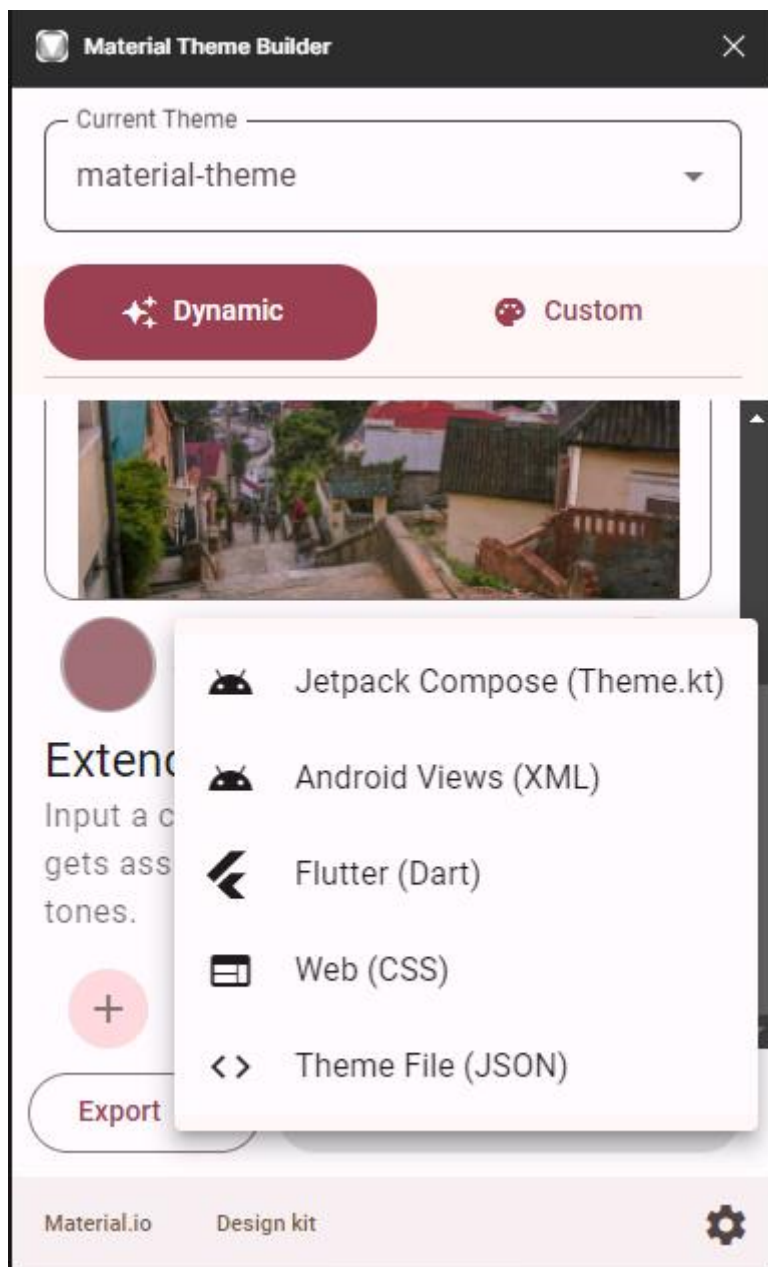


During this process, color values were extracted from the wallpaper or image, and each color was assigned a "type" that determined its relationship with other colors in the scheme. The "key colors" on the right-hand side of the interface were updated to reflect these extracted color values.



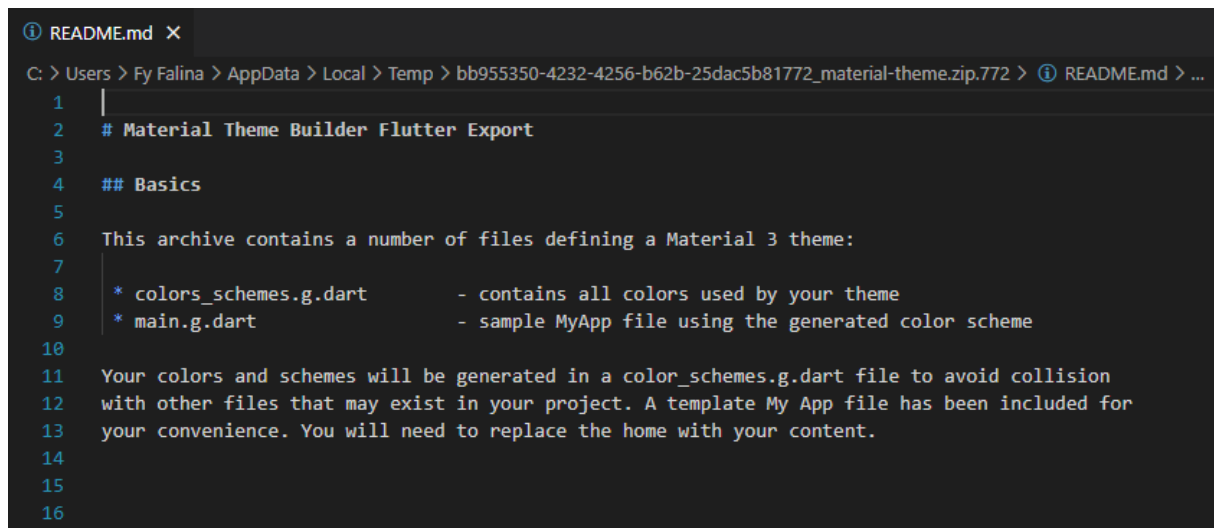
By exploring this step, we gained insights into how dynamic colors can be derived from images and how they are categorized to create harmonious color schemes within the Material Theme Builder.

Furthermore, we discovered that the Material Theme Builder offers the functionality to directly download the generated color scheme as a Dart file. This feature enables us to obtain a ready-to-use theme for our apps seamlessly. By downloading the color scheme as a Dart file, we can easily integrate it into our Flutter projects, saving time and effort in manually configuring the theme colors. This capability enhances the efficiency and convenience of implementing consistent and visually appealing themes across our applications.



» Téléchargements » material-theme.zip

Nom	Type	Taille compressée	Protégé pa...	Taille	Ratio	Modifié le
color_schemes.g.dart	Fichier source Dart	1 Ko	Non	3 Ko	75 %	28/10/2023 09:20
main.g.dart	Fichier source Dart	1 Ko	Non	2 Ko	59 %	28/10/2023 09:20
README.md	Fichier source Markdown	1 Ko	Non	1 Ko	45 %	28/10/2023 09:20



```

1 |
2 # Material Theme Builder Flutter Export
3
4 ## Basics
5
6 This archive contains a number of files defining a Material 3 theme:
7
8 * colors_schemes.g.dart      - contains all colors used by your theme
9 * main.g.dart                - sample MyApp file using the generated color scheme
10
11 Your colors and schemes will be generated in a color_schemes.g.dart file to avoid collision
12 with other files that may exist in your project. A template My App file has been included for
13 your convenience. You will need to replace the home with your content.
14
15
16

```

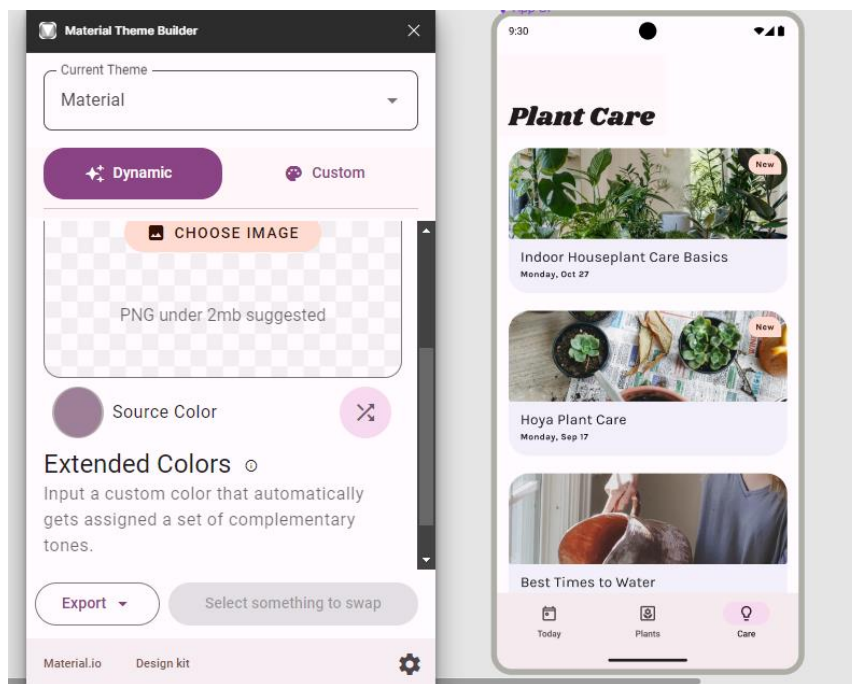
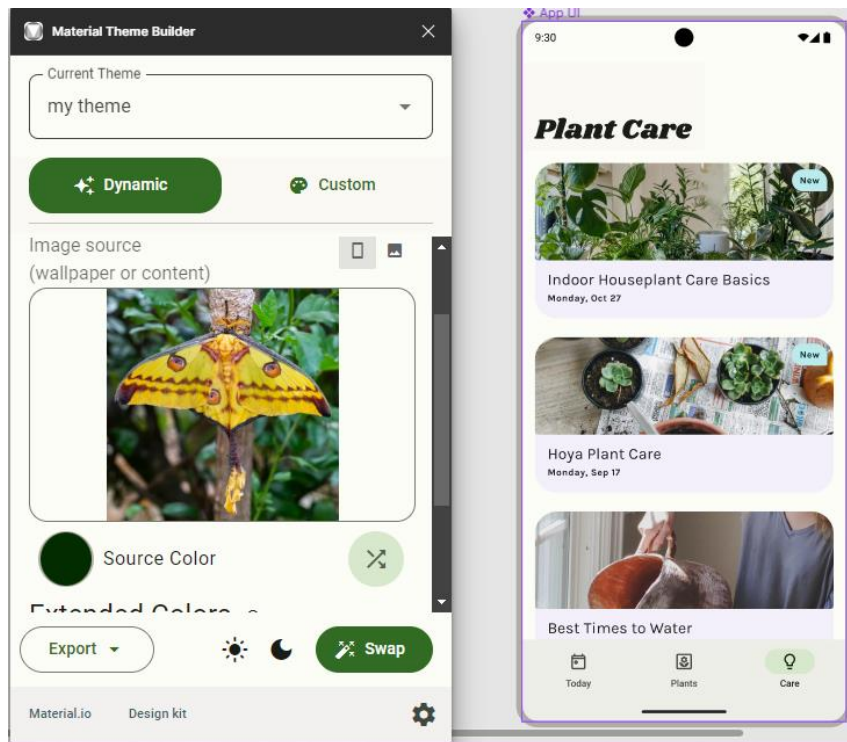
## Themes and tokens

Afterward, we proceeded to set up and utilize tokens in our designs. Design tokens play a crucial role in achieving flexibility and consistency across a product. They allow designers to assign an element's color role in a user interface (UI) rather than a specific color value. Tokens act as a connection between an element's assigned role and the chosen color value for that role. By designing based on color roles rather than specific colors, we embrace a more fundamental approach, especially with the introduction of dynamic color.

Themes encompass Material Design tokens for both color and typography, providing a single source of truth to represent the baseline design, including user-generated palettes and custom values.

In Figma, these tokens are generated as styles by the plugin. This means that if we utilize the generated styles, we will be employing the Material Design tokens.

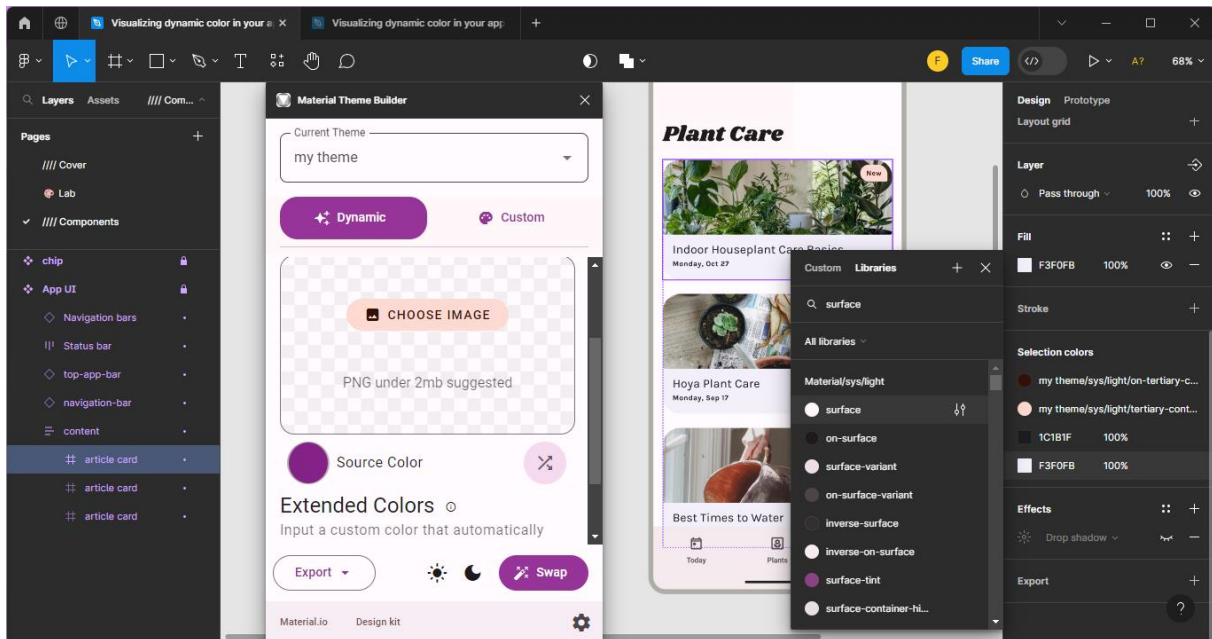
Colors in a tonal palette are mapped to either a light or dark scheme through design tokens. The mapping system assigns a specific tone to each element within a component.



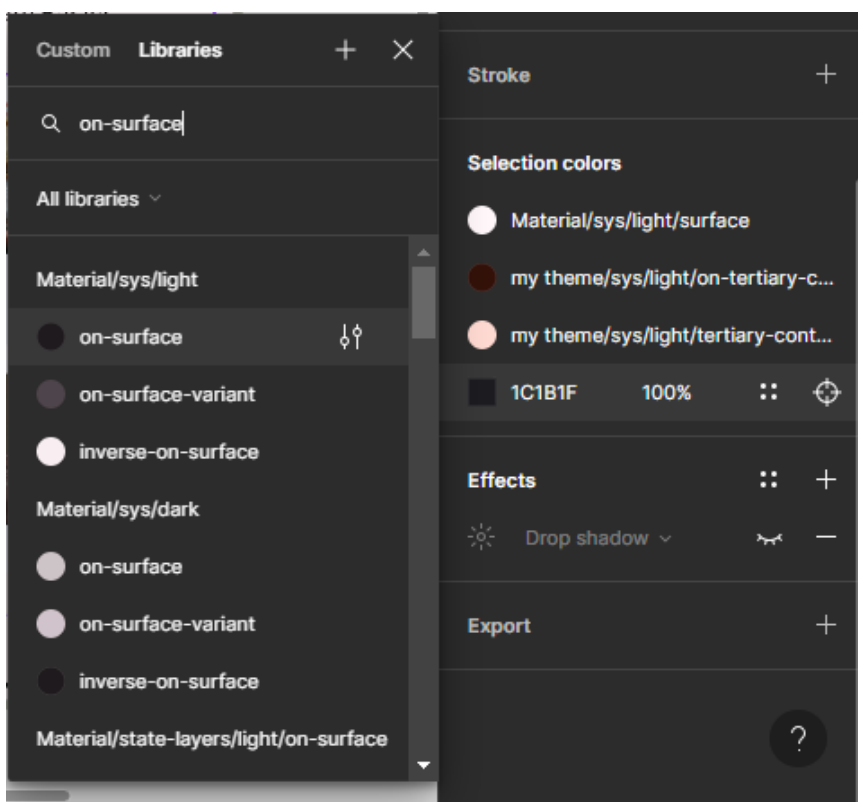
To set up the tokens, we needed to apply them to our designs. We accomplished this by selecting the frame of the layout and using the "swap" function to apply all the tokens (Figma styles) on the right side. As a result, we observed the style prefix updating in the selection colors. This step ensured that our layout utilized the designated theme with dynamic color applied through the tokens.



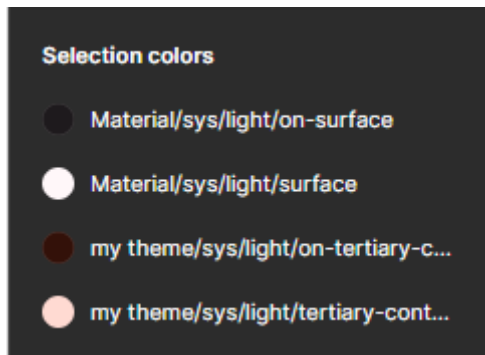
Moving on to applying the tokens to the user interface (UI), we encountered layouts that were initially created using the Material Design Kit, which already utilized Material Design tokens. However, there were a few custom elements that were not mapped to tokens.



To address this, we selected the article cards and adjusted their fill. By clicking on the style icon (depicted as four dots), we set the fill style to "material-theme/surface." Alternatively, we could search for "surface" to locate the appropriate style.



Next, we proceeded to set the type in the cards. Similar to the previous step, we selected the type elements within the cards and set them to "on-surface." Additionally, we adjusted the checkboxes and set them to "primary" to align with the desired token mappings.



By following these steps, we ensured that the UI elements, including the article cards' fill, type, and checkboxes, were appropriately mapped to the corresponding Material Design tokens. This helped maintain consistency and coherence throughout the design.

### Applying to UI

Utilizing Material Design, we can create multiple themes simultaneously and apply them to different duplicates of our design. This allows us to view and compare each theme side by side, facilitating the process of theme selection and evaluation.

