CompositionLocal

- A tool to pass data through Compositions implicitly
- Used to scope data to a sub-tree
- Different sub-trees can have different implementations

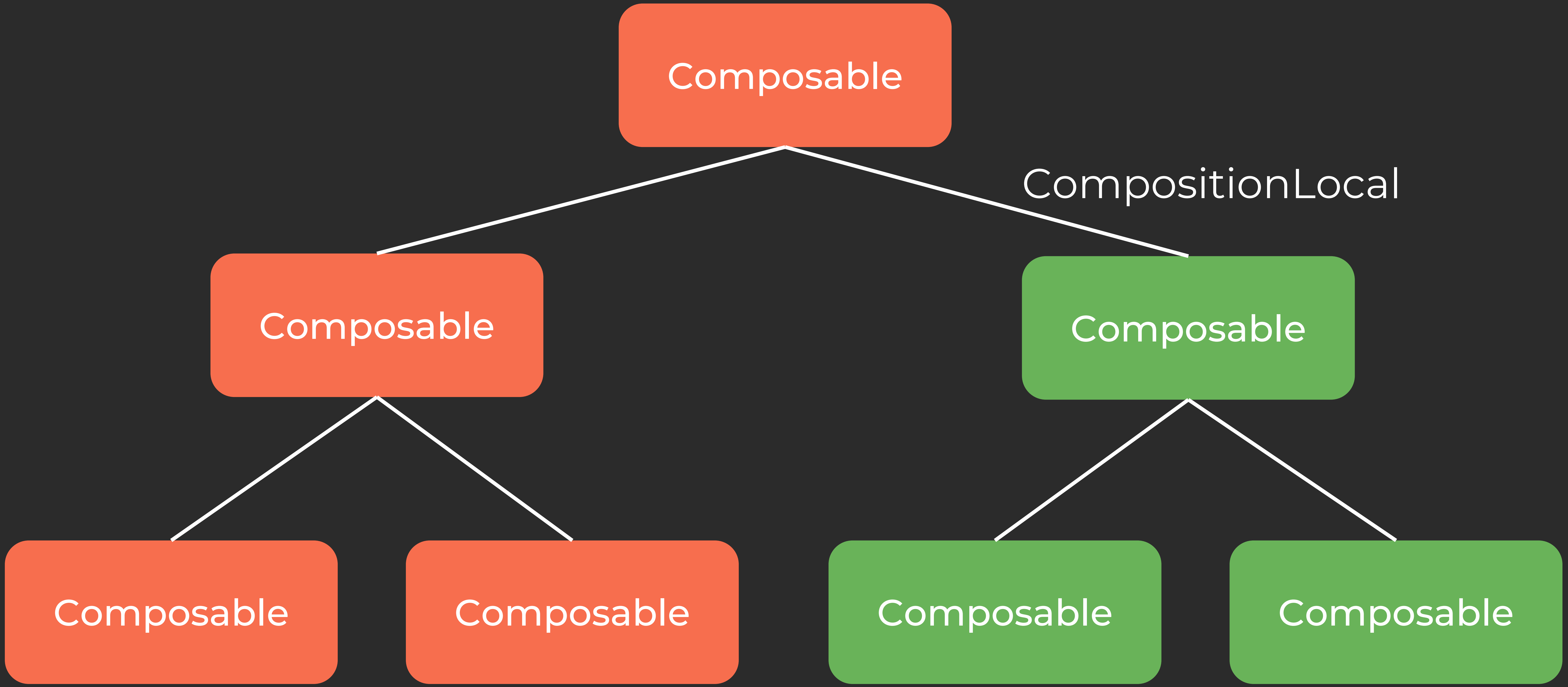
Explicit Composable Parameters

```
@Composable
fun App(name: String) {
   MaterialTheme {
      TextBar("Hello", Color.Cyan)
    }

}

@Composable
fun TextBar(name: String, color: Color) {
   Text(text = "Hello $name!", color = color )
}
```

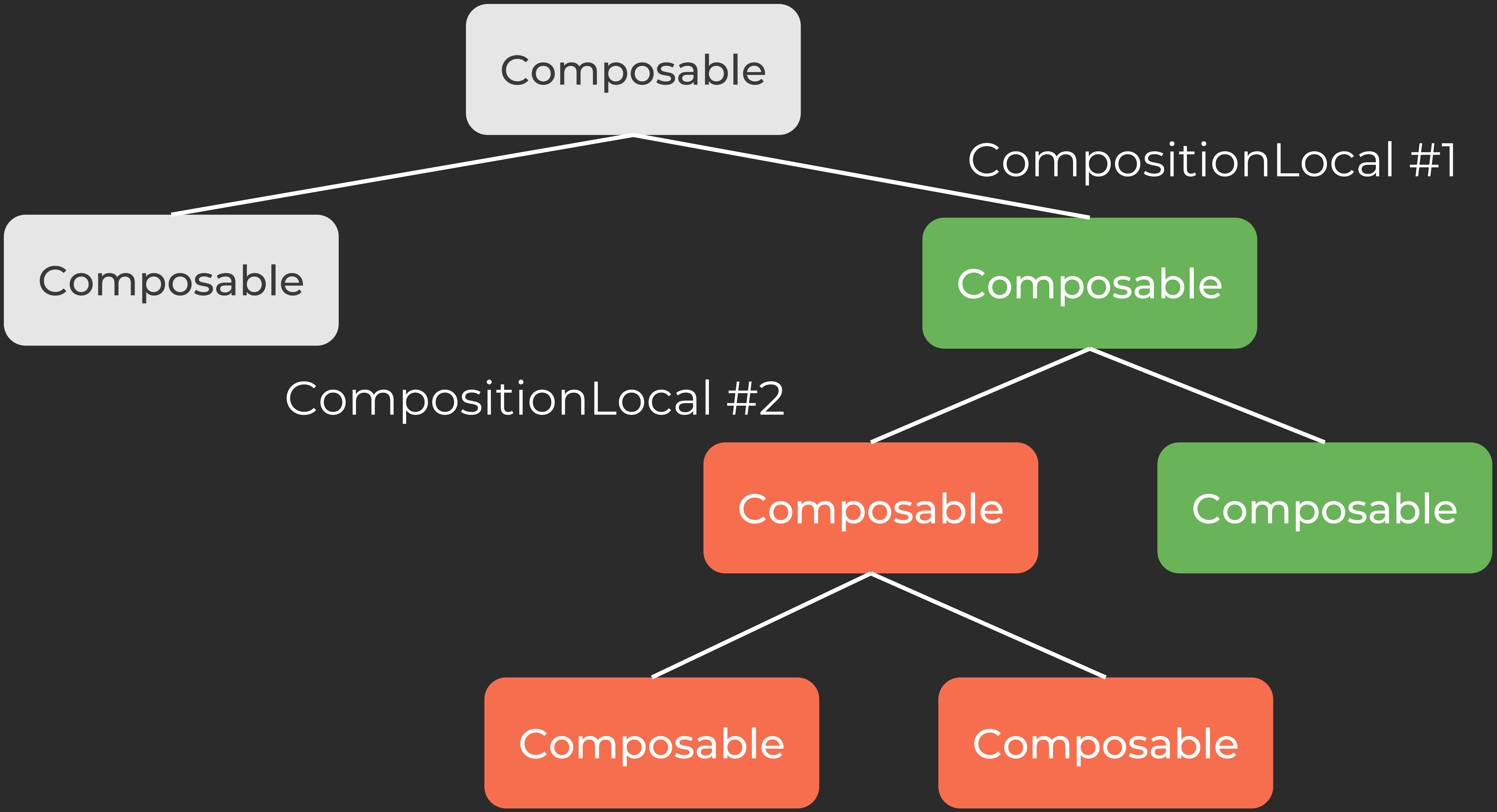
Implicit Composable Parameters

```
@Composable
fun App(name: String) {
   MaterialTheme {
      TextBar("Hello”)
    }

}

@Composable
fun TextBar(name: String) {
   Text(text = "Hello $name!”, color = MaterialTheme.Colors.onSurface )
}
```

CompositionLocal Tree



CompositionLocal Tree



Ways to Create `CompositionLocal`

- `compositionLocalOf`
  - For values that change often or are animated
  - Eg: colors, dimensions

- `staticCompositionLocalOf`
  - For values that are less likely to change in the sub-tree
  - Eg: font loader

Summary

In this section, you learned about:

- Best practices for adding a dark theme
- Creating a theme switcher dialog
- CompositionLocal tree
- Use compositionLocalOf for values that change often
- Use staticCompositionLocalOf for values less likely to change often

In the next section, you will learn about adding customizable previews.