BUILD YOUR FIRST APP

Tim Dosen Mobpro 1

D3 Rekayasa Perangkat Lunak Aplikasi
Fakultas Ilmu Terapan

SUSUNAN KODE ANDROID

```
package …

import …

class MainActivity : ComponentActivity() { … }

@Composable

@Preview
```

Ingat kembali susunan kode saat running modul praktikum

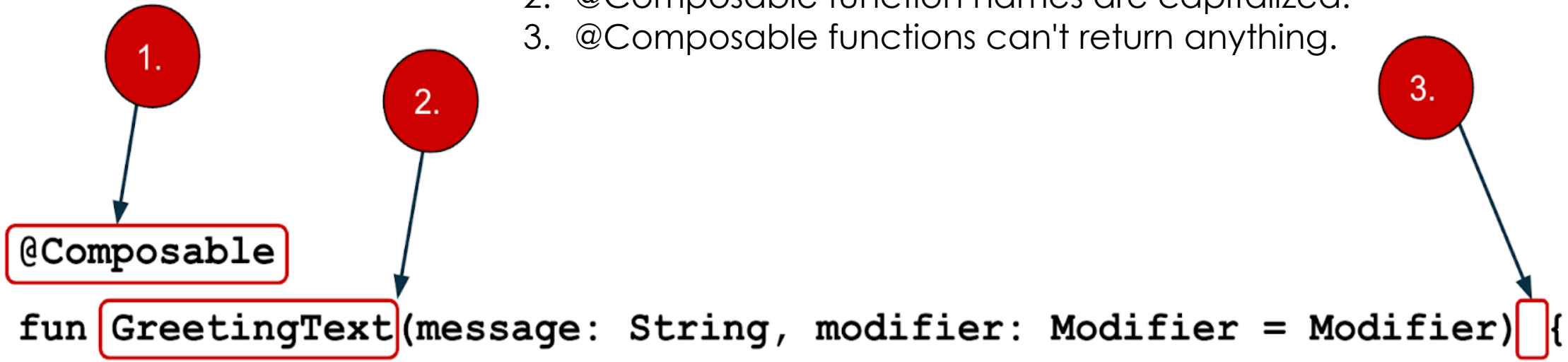
USER INTERFACE

- The user interface (UI) of an app is what you see on the screen:
  - text,
  - images,
  - buttons,
  - and many other types of elements.

- How the app shows things to the user?
- How it's laid out on the screen?
- How the user interacts with the app?

MEMBUAT UI DENGAN COMPOSE

1. Add the @Composable annotation before the function.
2. @Composable function names are capitalized.
3. @Composable functions can't return anything.



TEXT EXAMPLE

- `Text("Hello World", color = Color.Blue)`
- `Text("Hello World", fontSize = 30.sp)`
- `Text("Hello World", fontStyle = FontStyle.Italic)`
- `Text("Hello World", fontWeight = FontWeight.Bold)`
- `Text("Hello World", textAlign = TextAlign.Center)`

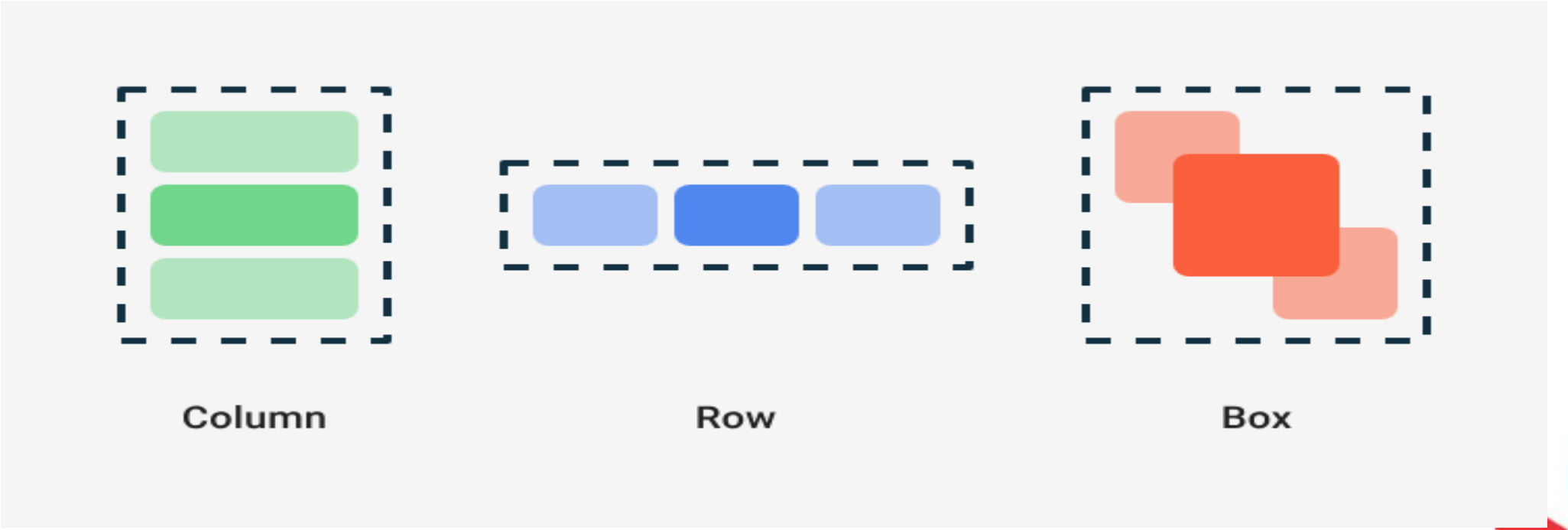
BUTTON EXAMPLE

Buttons are fundamental components that allow the user to trigger a defined action. There are five types of buttons.


```
Button(onClick = { onClick() }) {
    Text("Filled")
}
```

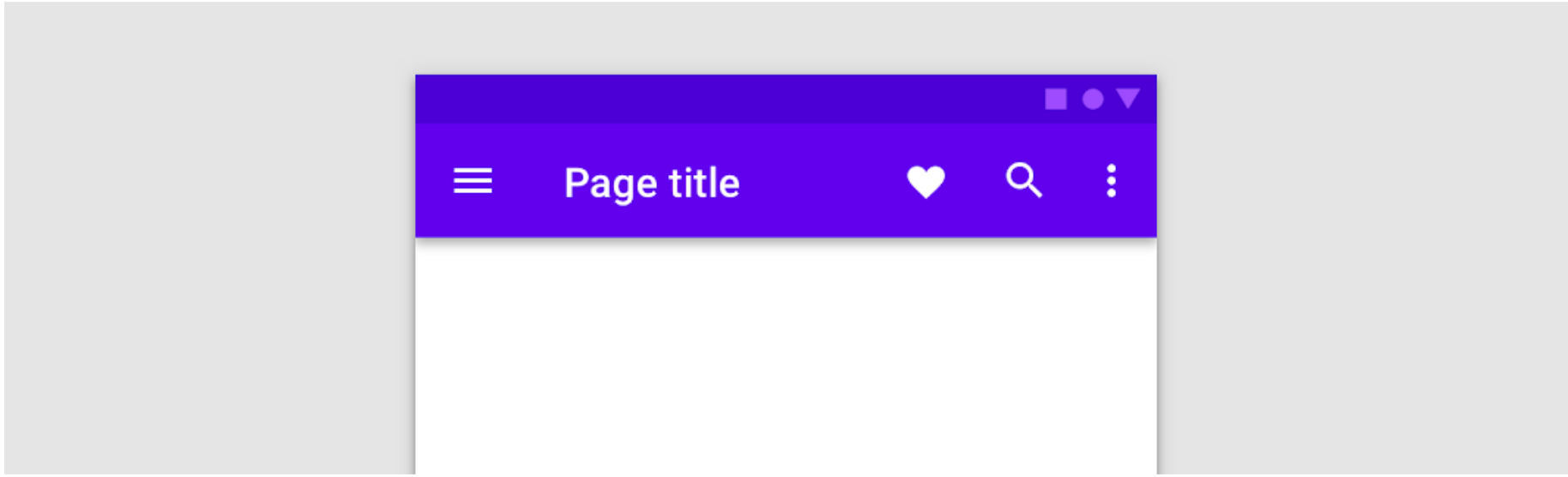
CONTAINER

Digunakan ketika user interface kita memiliki lebih dari satu komponen.



SCAFFOLD

Fundamental structure that provides a standardized platform for complex user interfaces. It holds together different parts of the UI, such as app bars and floating action buttons, giving apps a coherent look and feel.



MODIFIER

Modifiers allow you to decorate or augment a composable:

- Change the composable's size, layout, behavior, and appearance
- Add information, like accessibility labels
- Process user input
- Making an element clickable, scrollable, draggable, or zoomable

The order of modifier functions is significant. Since each function makes changes to the Modifierreturned by the previous function, the sequence affects the final result.

STATE

State in an app is any value that can change over time.

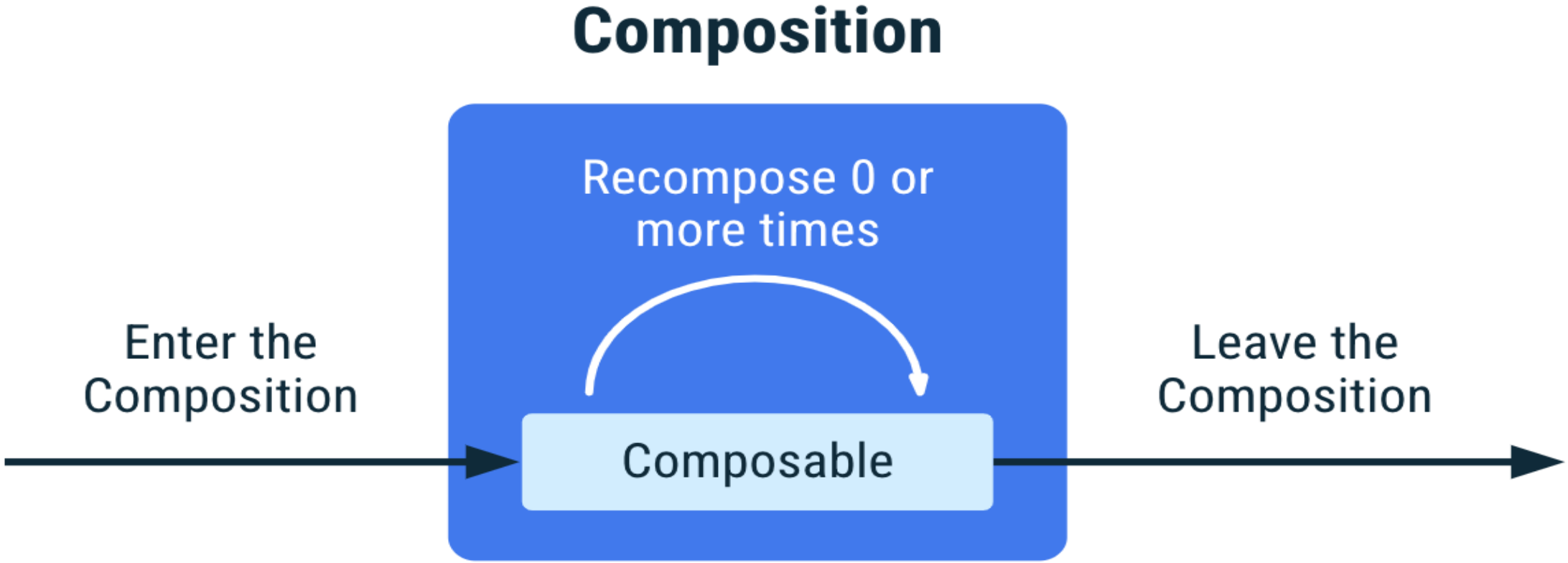
```
var number by remember { mutableIntStateOf(0) }

Text(text = number.toString())

Button(onClick = { number++ }) { … }
```

Any time a state is updated a recomposition takes place.

HOW IT WORKS?


Composition
Recompose 0 or more times
Enter the Composition
Composable
Leave the Composition

HOW IT WORKS?

- When Jetpack Compose runs your composables for the first time, during initial composition, it will keep track of the composables that you call to describe your UI in a Composition.

- Then, when the state of your app changes, Jetpack Compose schedules a recomposition.

- Recomposition is when Jetpack Compose re-executes the composables that may have changed in response to state changes, and then updates the Composition to reflect any changes.

REFERENCES

- Android Basics with Compose
  https://developer.android.com/courses/android-basics-compose/course

- Kotlin Bootcamp for Programmers
  https://developer.android.com/courses/kotlin-bootcamp/overview

- Jetpack Compose for Android Developers
  https://developer.android.com/courses/jetpack-compose/course