Android Jetpack Compose Notes

# Compose Essentials

Jetpack Compose is Declarative UI Framework. You declare what you UI should contain and compose creates the elements UI using Kotlin.

You construct the UI by describing what and not how.

You don’t need to use XML Views, the UI will be described in code using Kotlin.

In Compose, UI elements are functions, known as composables, and not objects. That means you can’t find references to them and mutate them. Instead, UI elements are controlled by the state or arguments you pass.

We don’t tell Compose how it should render states. To do that, we use Event Handlers, which decides if the UI element’s state should be changed. If the UI state changes, the functions or UI elements that depend on that state is re-executed, this is called recomposition.

## Composable Functions

A composable function is a function that has a @Composable annotation. This annotation indicates the compiler that this function is intended to convert data into UI.

The composable functions allow you have reusable components.

Composables are immutable, that means you can’t hold a reference to them and later update its value.

Recomposition happens when a composable is re-invoked with different functions parameters but it can also happen when internal state in the function changes.

MutableState, remember and rememberSaveable can be used to store a component state and have Compose automatically track and recompose changes.

Composable functions can execute in any order.

Composable functions can also run in parallel.

Recomposition skips as much as possible.

Composable functions might run frequently.