JetPack Compose

- Pengenalan Jetpack Compose
- Konsep Dasar Composable
- Layouting
- State compose
- Lazy Layout
- Navigation
- Testing
- Interoperability

hello compose!

less code

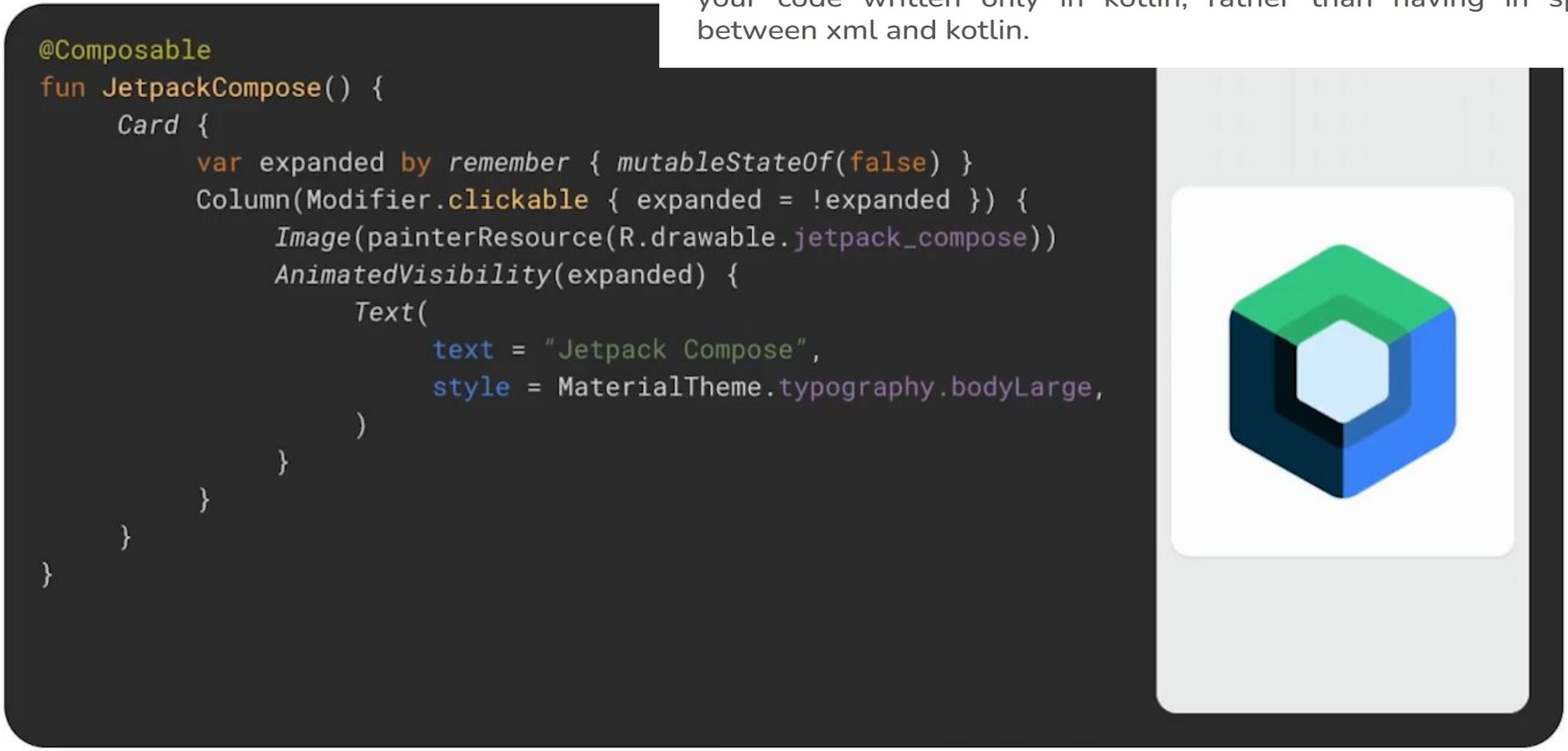
intuitive

accelerate development

powerful

less code

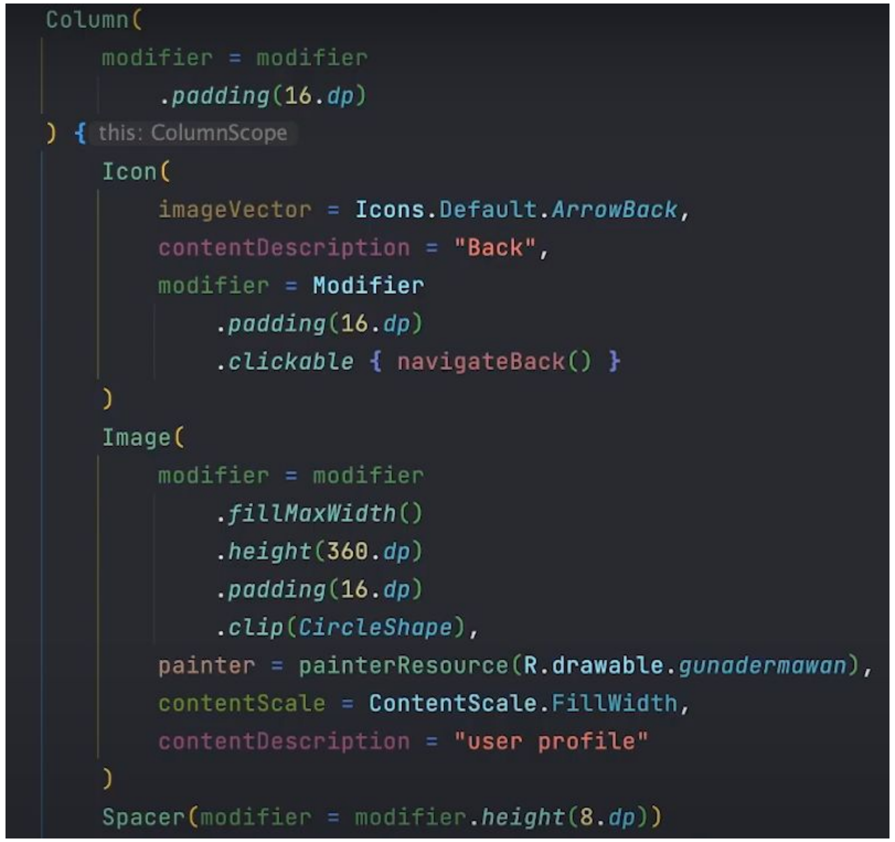
your code written only in kotlin, rather than having in split between xml and kotlin.



```
@Composable
fun JetpackCompose() {
    Card {
        var expanded by remember { mutableStateOf(false) }
        Column(Modifier.clickable { expanded = !expanded }) {
            Image(painterResource(R.drawable.jetpack_compose))
            AnimatedVisibility(expanded) {
                Text(
                    text = "Jetpack Compose",
                    style = MaterialTheme.typography.bodyLarge,
                )
            }
        }
    }
}
```

intuitive

compose use declarative paradigm to create UI, all you need to do is describe the UI

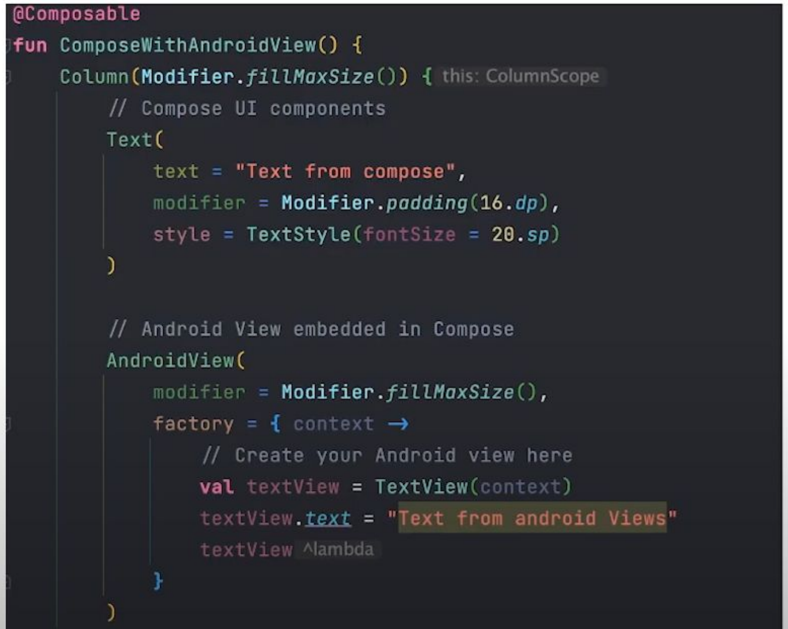


```
Column(
    modifier = modifier
        .padding(16.dp)
) { this: ColumnScope
    Icon(
        imageVector = Icons.Default.ArrowBack,
        contentDescription = "Back",
        modifier = Modifier
            .padding(16.dp)
            .clickable { navigateBack() }
    )
    Image(
        modifier = modifier
            .fillMaxWidth()
            .height(360.dp)
            .padding(16.dp)
            .clip(CircleShape),
        painter = painterResource(R.drawable.gundermawan),
        contentScale = ContentScale.FillWidth,
        contentDescription = "user profile"
    )
    Spacer(modifier = modifier.height(8.dp))
```

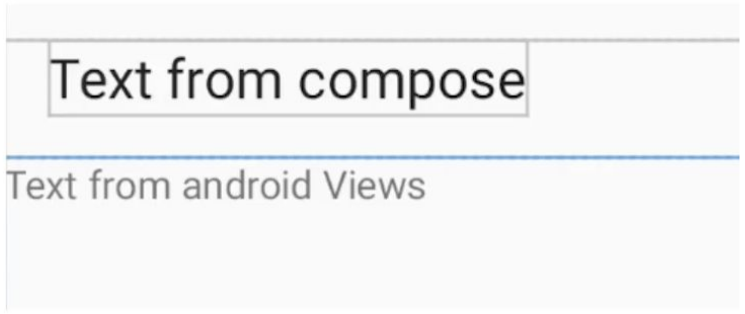
accelerate development

compose is compatible with all your existing code, you can call compose code from views and Views from compose

accelerate development



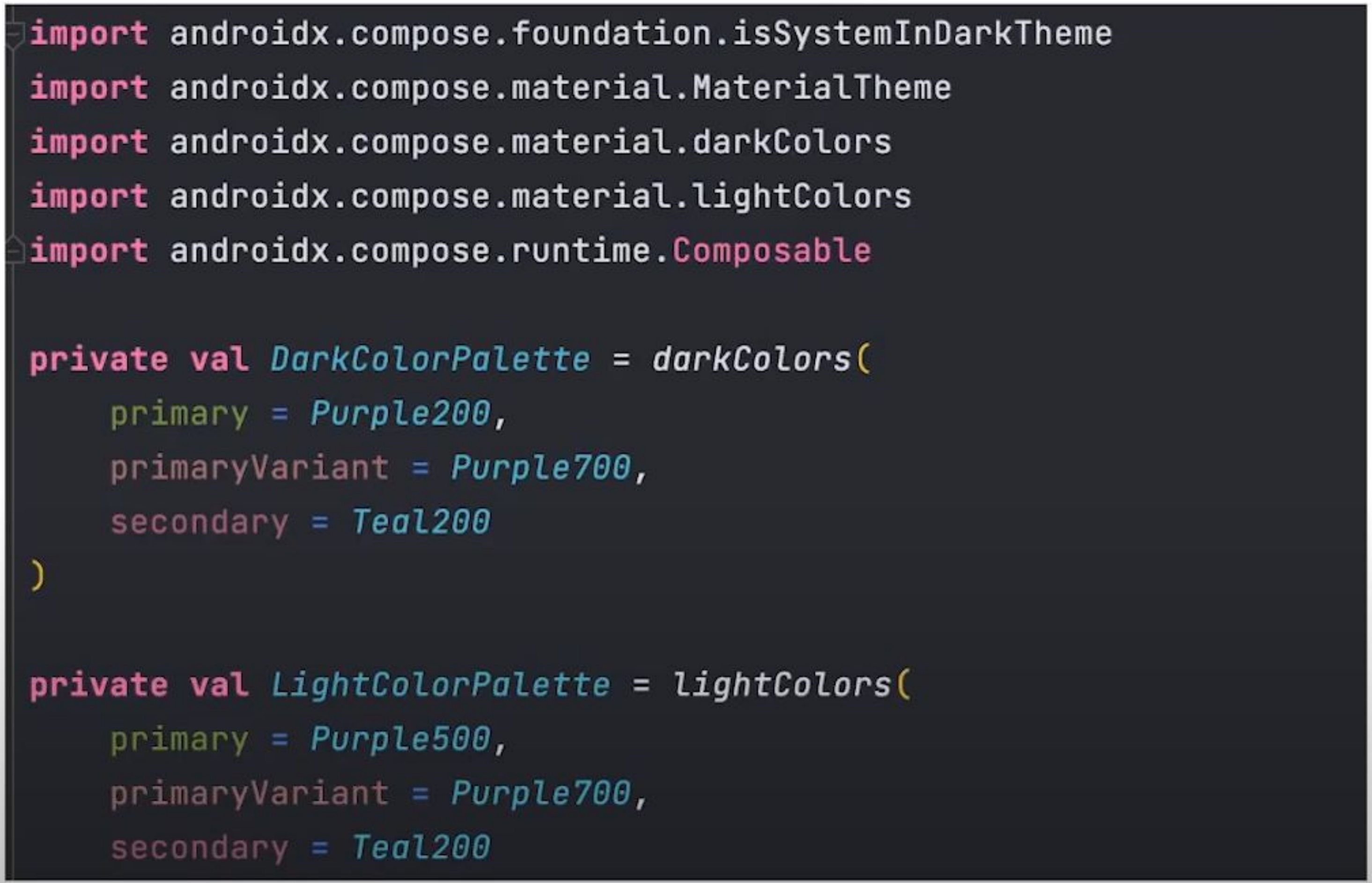
```
@Composable
fun ComposeWithAndroidView() {
    Column(Modifier.fillMaxSize()) { this: ColumnScope
        // Compose UI components
        Text(
            text = "Text from compose",
            modifier = Modifier.padding(16.dp),
            style = TextStyle(fontSize = 20.sp)
        )

        // Android View embedded in Compose
        AndroidView(
            modifier = Modifier.fillMaxSize(),
            factory = { context ->
                // Create your Android view here
                val textView = TextView(context)
                textView.text = "Text from android Views"
                textView ^lambda
            }
        )
```



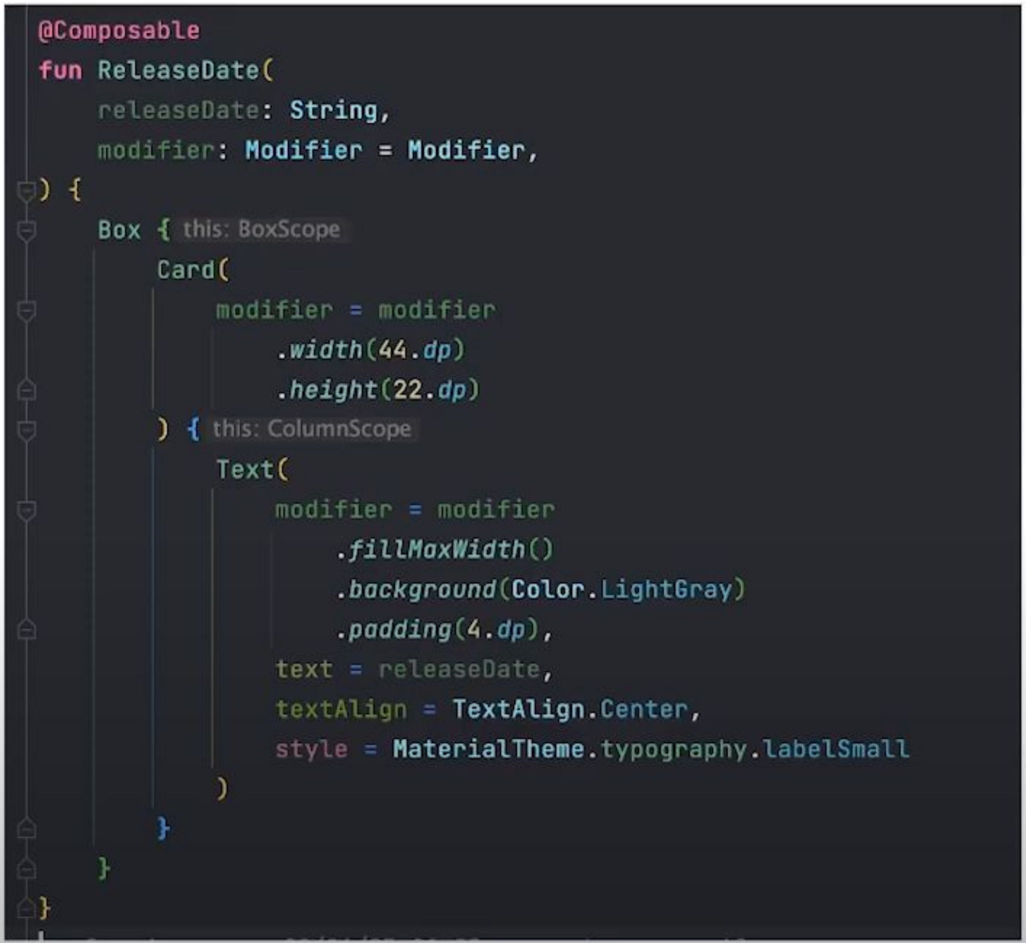
powerfull

compose enable to you use Android API Platform to create beautiful apps with support material design, dark theme, animation and more.

powerfull



```kotlin
import androidx.compose.foundation.isSystemInDarkTheme
import androidx.compose.material.MaterialTheme
import androidx.compose.material.darkColors
import androidx.compose.material.lightColors
import androidx.compose.runtime.Composable

private val DarkColorPalette = darkColors(
    primary = Purple200,
    primaryVariant = Purple700,
    secondary = Teal200
)

private val LightColorPalette = lightColors(
    primary = Purple500,
    primaryVariant = Purple700,
    secondary = Teal200
```



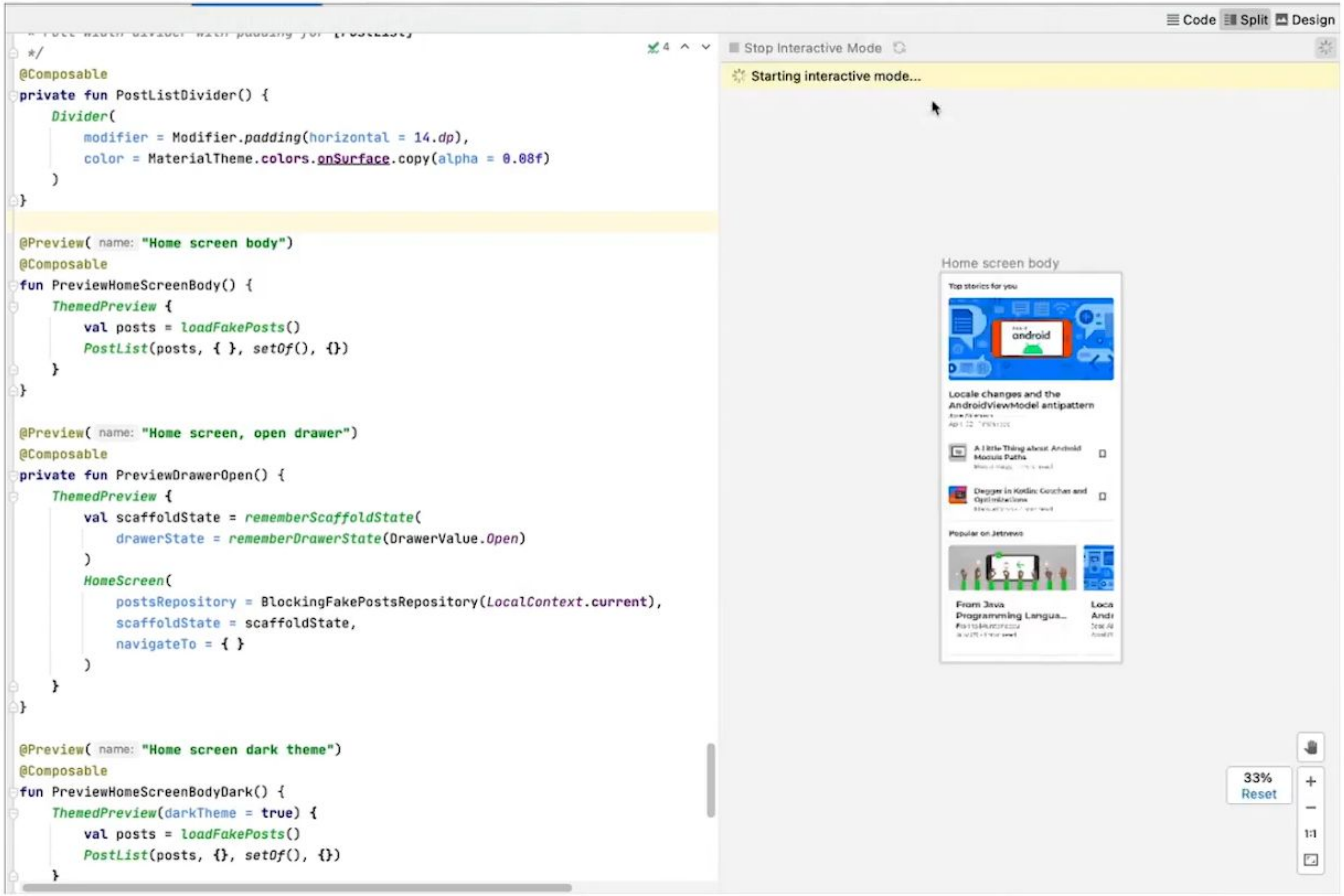
```kotlin
@Composable
fun ReleaseDate(
    releaseDate: String,
    modifier: Modifier = Modifier,
) {
    Box { this: BoxScope
        Card(
            modifier = modifier
                .width(44.dp)
                .height(22.dp)
        ) { this: ColumnScope
            Text(
                modifier = modifier
                    .fillMaxWidth()
                    .background(Color.LightGray)
                    .padding(4.dp),
                text = releaseDate,
                textAlign = TextAlign.Center,
                style = MaterialTheme.typography.labelSmall
            )
        }
    }
}
```

built with compose

tools compose

1. Interactive mode
2. Live Edit
3. Animation preview
4. Live Template
5. Preview Parameter

Interactive mode

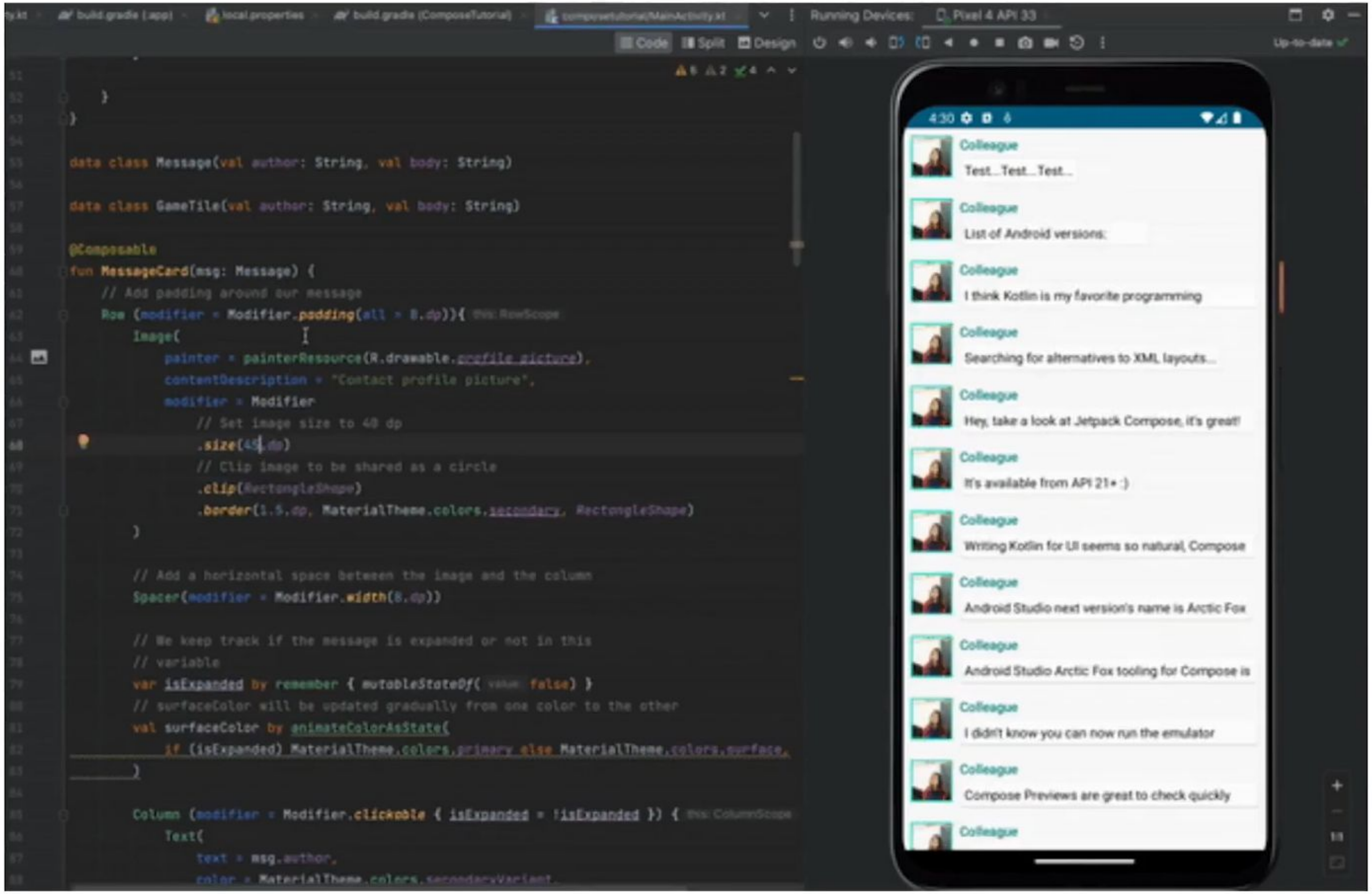


Code
Split
Design
Stop Interactive Mode
Home screen body
Top stories for you
Locale changes and the AndroidViewModel antipattern
A Little Thing about Android Module Paths
Dagger in Kotlin: Gotchas and Optimizations
Popular on Jetnews
From Java Programming Langua...
33%
Reset
1:1

Starting interactive mode...

```kotlin
 * Full width divider with padding for [PostList]
 */
@Composable
private fun PostListDivider() {
    Divider(
        modifier = Modifier.padding(horizontal = 14.dp),
        color = MaterialTheme.colors.onSurface.copy(alpha = 0.08f)
    )
}

@Preview( name: "Home screen body")
@Composable
fun PreviewHomeScreenBody() {
    ThemedPreview {
        val posts = loadFakePosts()
        PostList(posts, { }, setOf(), {})
    }
}

@Preview( name: "Home screen, open drawer")
@Composable
private fun PreviewDrawerOpen() {
    ThemedPreview {
        val scaffoldState = rememberScaffoldState(
            drawerState = rememberDrawerState(DrawerValue.Open)
        )
        HomeScreen(
            postsRepository = BlockingFakePostsRepository(LocalContext.current),
            scaffoldState = scaffoldState,
            navigateTo = { }
        )
    }
}

@Preview( name: "Home screen dark theme")
@Composable
fun PreviewHomeScreenBodyDark() {
    ThemedPreview(darkTheme = true) {
        val posts = loadFakePosts()
        PostList(posts, {}, setOf(), {})
    }
```

Live Edit



```kotlin
    }
}

data class Message(val author: String, val body: String)

data class GameTile(val author: String, val body: String)

@Composable
fun MessageCard(msg: Message) {
    // Add padding around our message
    Row (modifier = Modifier.padding(all = 8.dp)){
        Image(
            painter = painterResource(R.drawable.profile_picture),
            contentDescription = "Contact profile picture",
            modifier = Modifier
                // Set image size to 40 dp
                .size(45.dp)
                // Clip image to be shared as a circle
                .clip(RectangleShape)
                .border(1.5.dp, MaterialTheme.colors.secondary, RectangleShape)
        )

        // Add a horizontal space between the image and the column
        Spacer(modifier = Modifier.width(8.dp))

        // We keep track if the message is expanded or not in this
        // variable
        var isExpanded by remember { mutableStateOf(false) }
        // surfaceColor will be updated gradually from one color to the other
        val surfaceColor by animateColorAsState(
            if (isExpanded) MaterialTheme.colors.primary else MaterialTheme.colors.surface,
        )

        Column (modifier = Modifier.clickable { isExpanded = !isExpanded }) {
            Text(
                text = msg.author,
                color = MaterialTheme.colors.secondaryVariant,
```

Colleague
Test...Test...Test...

Colleague
List of Android versions:

Colleague
I think Kotlin is my favorite programming

Colleague
Searching for alternatives to XML layouts...

Colleague
Hey, take a look at Jetpack Compose, it's great!

Colleague
It's available from API 21+ :)

Colleague
Writing Kotlin for UI seems so natural, Compose

Colleague
Android Studio next version's name is Arctic Fox

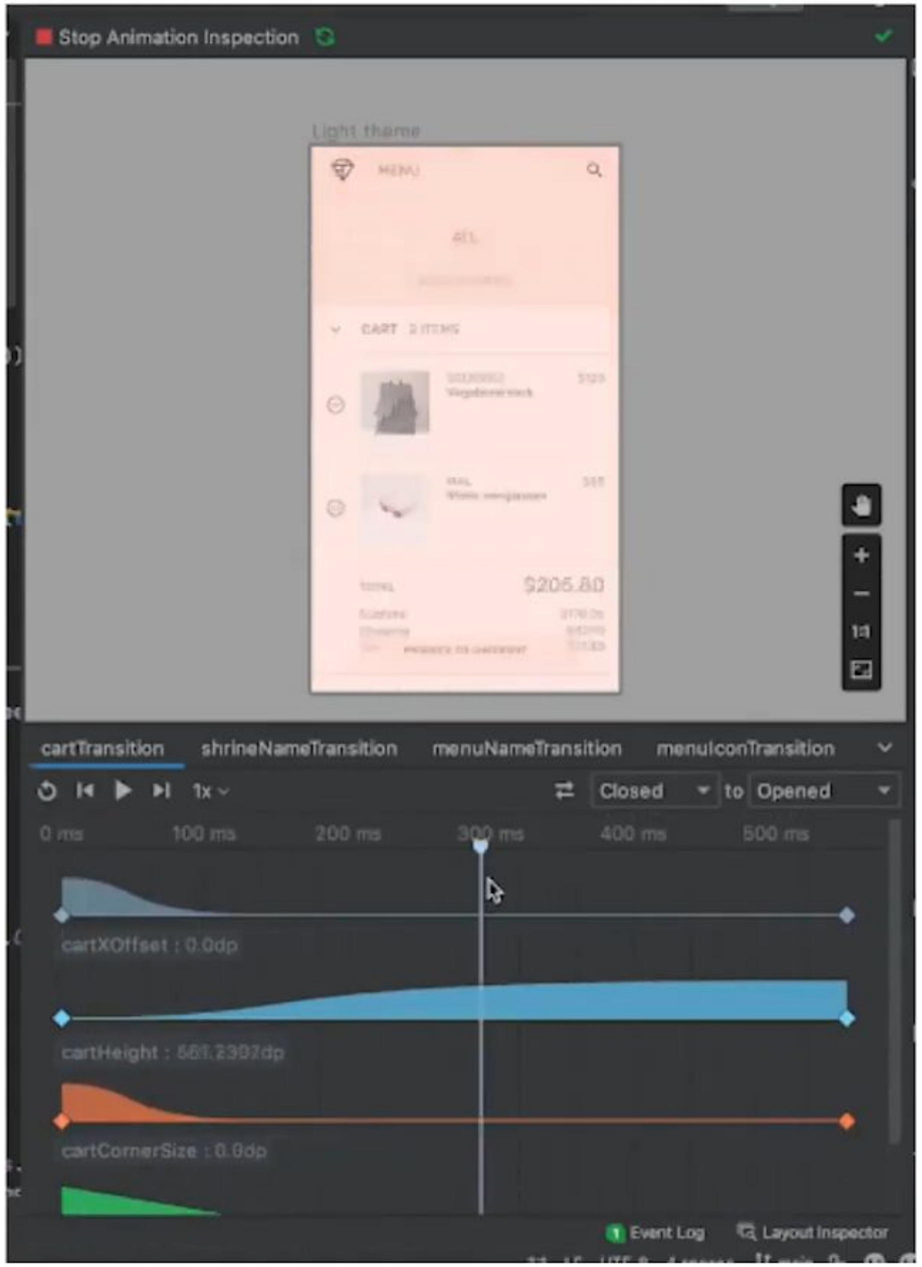
Colleague
Android Studio Arctic Fox tooling for Compose is

Colleague
I didn't know you can now run the emulator

Colleague
Compose Previews are great to check quickly

Colleague

Animation Preview



cartXOffset : 0.0dp

cartHeight : 561.2307dp

cartCornerSize : 0.0dp

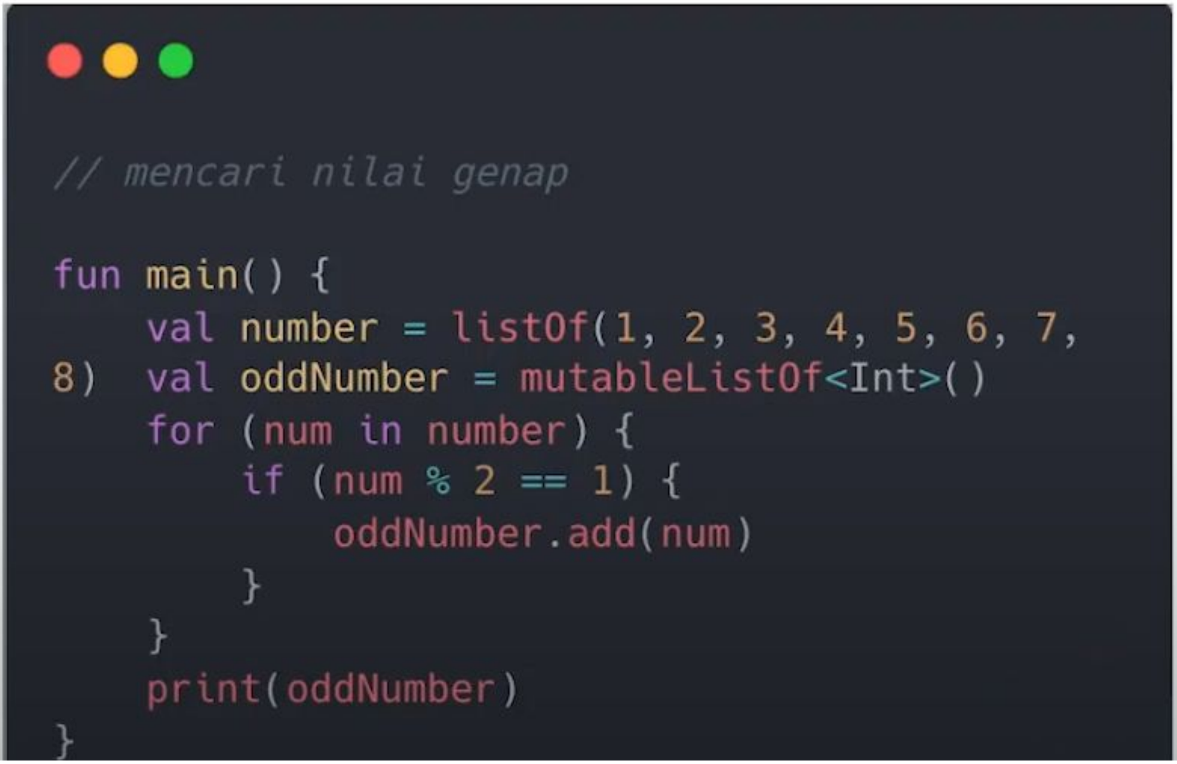
Deklaratif UI

- ➔ **Deklaratif programming**
  paradigma yang mendeskripsikan **'apa'** yang akan dilakukan tanpa memperdulikan urutanya
- ➔ **Imperatif programming**
  paradigma yang mendeskripsikan **'bagaimana'** suatu proses yang akan dilakukan dengan menjelaskan tiap langkahnya

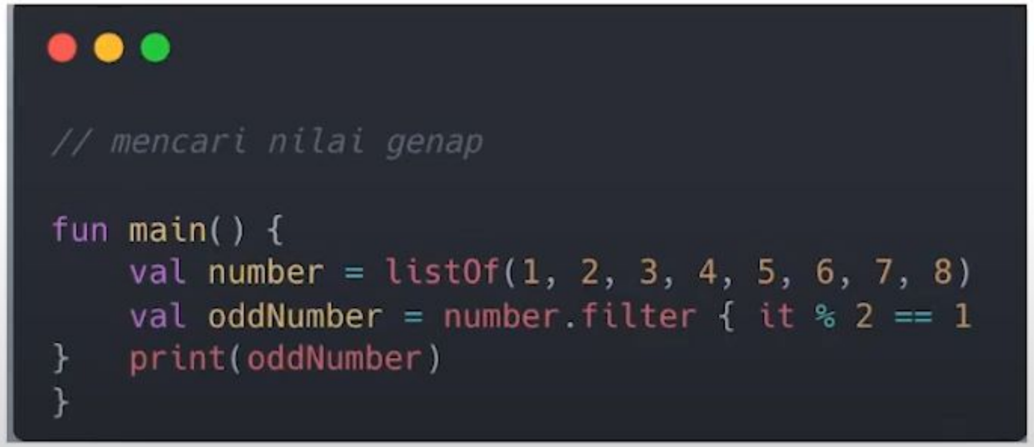
Code sample

Imperatif code

Deklaratif code



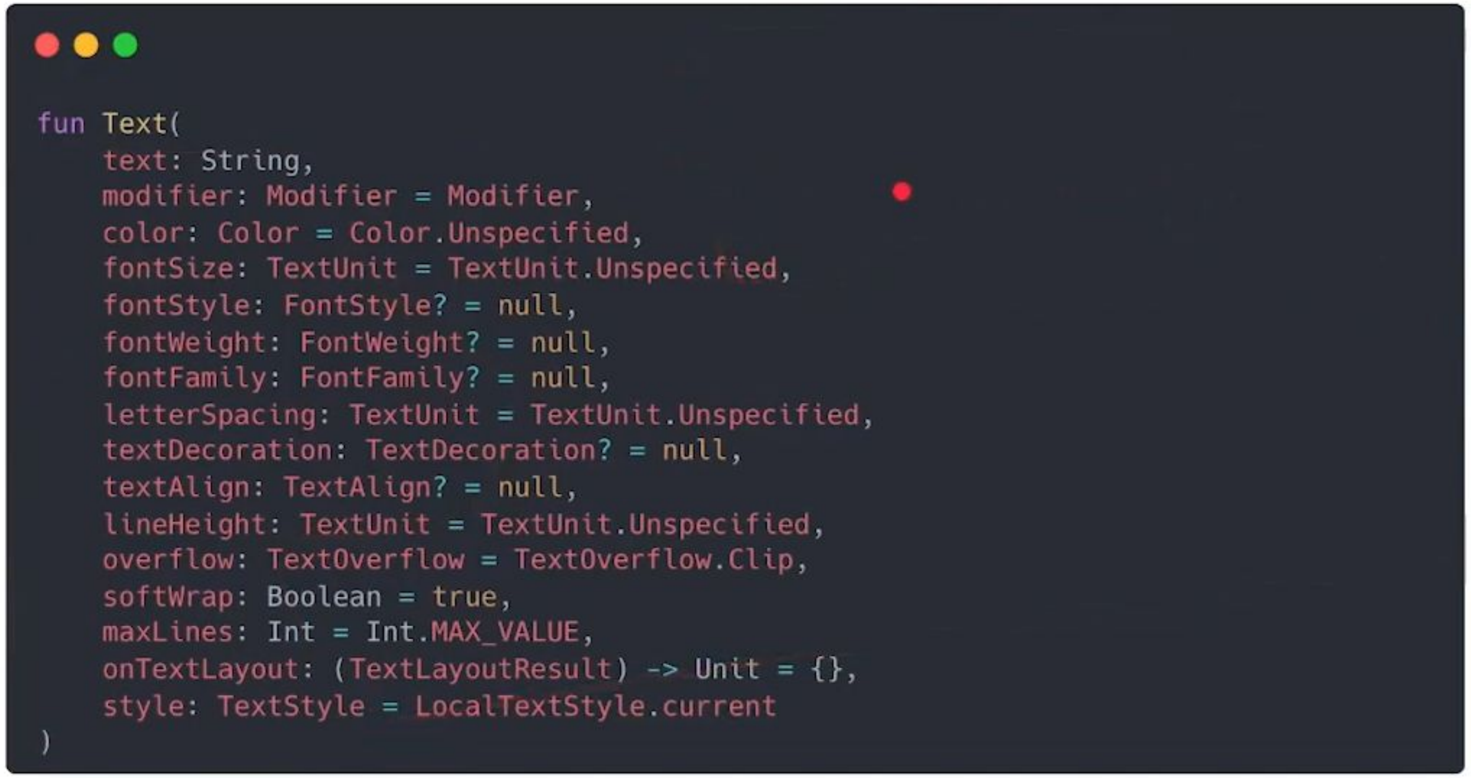
```
// mencari nilai genap

fun main() {
    val number = listOf(1, 2, 3, 4, 5, 6, 7,
8)  val oddNumber = mutableListOf<Int>()
    for (num in number) {
        if (num % 2 == 1) {
            oddNumber.add(num)
        }
    }
    print(oddNumber)
}
```



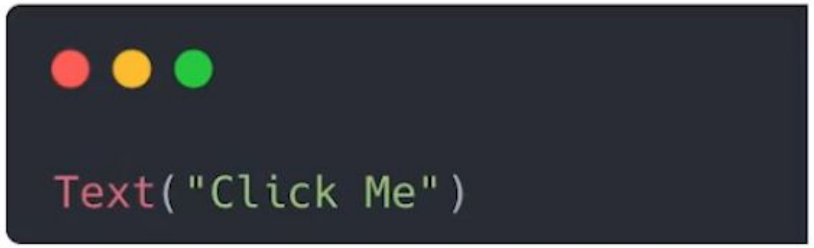
```
// mencari nilai genap

fun main() {
    val number = listOf(1, 2, 3, 4, 5, 6, 7, 8)
    val oddNumber = number.filter { it % 2 == 1
}   print(oddNumber)
}
```

1. Default Argument

Nilai yang secara bawaan telah diisi, jadi jika tidak ada nilai yang di inputkan, akan menggunakan nilai yang telah ada.



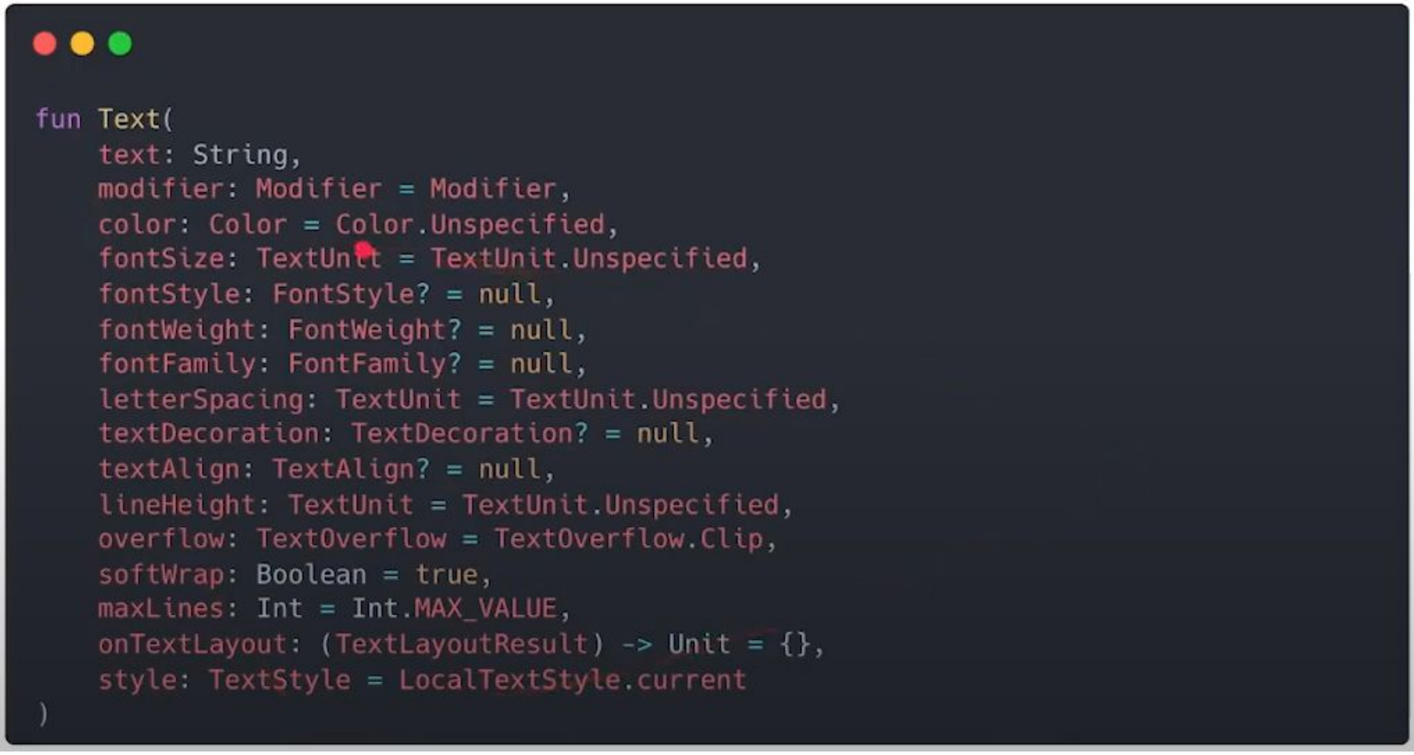
```kotlin
fun Text(
    text: String,
    modifier: Modifier = Modifier,
    color: Color = Color.Unspecified,
    fontSize: TextUnit = TextUnit.Unspecified,
    fontStyle: FontStyle? = null,
    fontWeight: FontWeight? = null,
    fontFamily: FontFamily? = null,
    letterSpacing: TextUnit = TextUnit.Unspecified,
    textDecoration: TextDecoration? = null,
    textAlign: TextAlign? = null,
    lineHeight: TextUnit = TextUnit.Unspecified,
    overflow: TextOverflow = TextOverflow.Clip,
    softWrap: Boolean = true,
    maxLines: Int = Int.MAX_VALUE,
    onTextLayout: (TextLayoutResult) -> Unit = {},
    style: TextStyle = LocalTextStyle.current
)
```



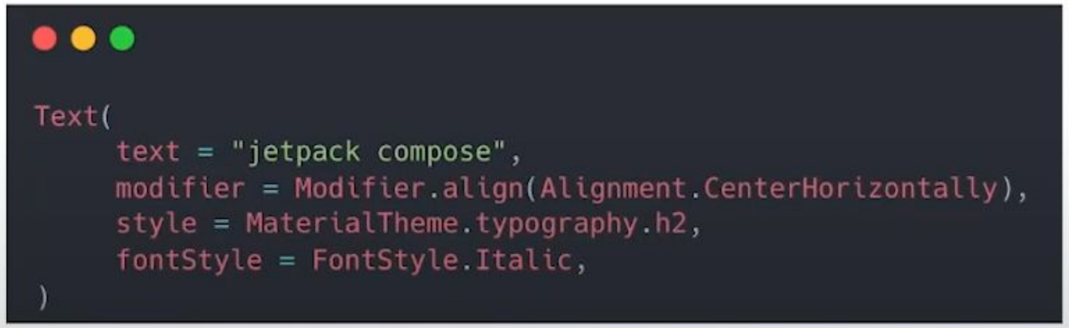
```kotlin
Text("Click Me")
```

2. Named Argument

Menuliskan parameter tanpa harus sesuai urutanya



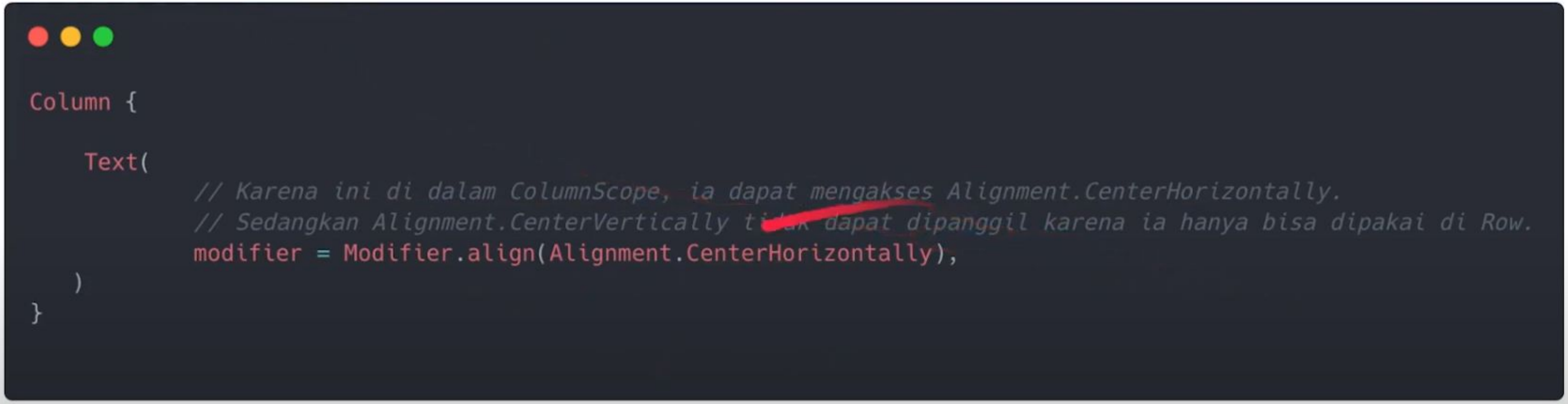
```kotlin
fun Text(
    text: String,
    modifier: Modifier = Modifier,
    color: Color = Color.Unspecified,
    fontSize: TextUnit = TextUnit.Unspecified,
    fontStyle: FontStyle? = null,
    fontWeight: FontWeight? = null,
    fontFamily: FontFamily? = null,
    letterSpacing: TextUnit = TextUnit.Unspecified,
    textDecoration: TextDecoration? = null,
    textAlign: TextAlign? = null,
    lineHeight: TextUnit = TextUnit.Unspecified,
    overflow: TextOverflow = TextOverflow.Clip,
    softWrap: Boolean = true,
    maxLines: Int = Int.MAX_VALUE,
    onTextLayout: (TextLayoutResult) -> Unit = {},
    style: TextStyle = LocalTextStyle.current
)
```



```kotlin
Text(
     text = "jetpack compose",
     modifier = Modifier.align(Alignment.CenterHorizontally),
     style = MaterialTheme.typography.h2,
     fontStyle = FontStyle.Italic,
)
```

3. Scope

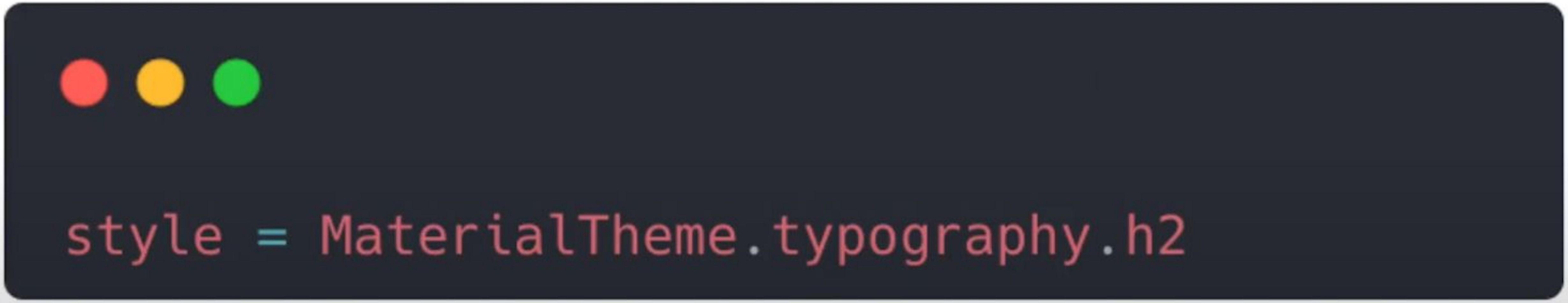
menuliskan kode sesuai dengan cakupan yang sesuai saka



```
Column {

    Text(
        // Karena ini di dalam ColumnScope, ia dapat mengakses Alignment.CenterHorizontally.
        // Sedangkan Alignment.CenterVertically tidak dapat dipanggil karena ia hanya bisa dipakai di Row.
        modifier = Modifier.align(Alignment.CenterHorizontally),
    )
}
```

4. Singleton Object

Pembuatan singleton akan lebih mudah dengan penggunaan object di kotlin

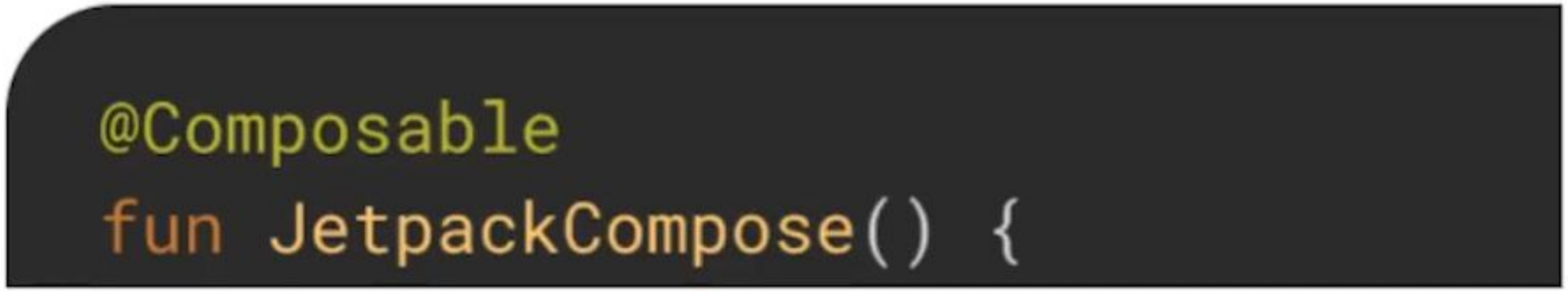


```
style = MaterialTheme.typography.h2
```

Composable Function

arti @composable

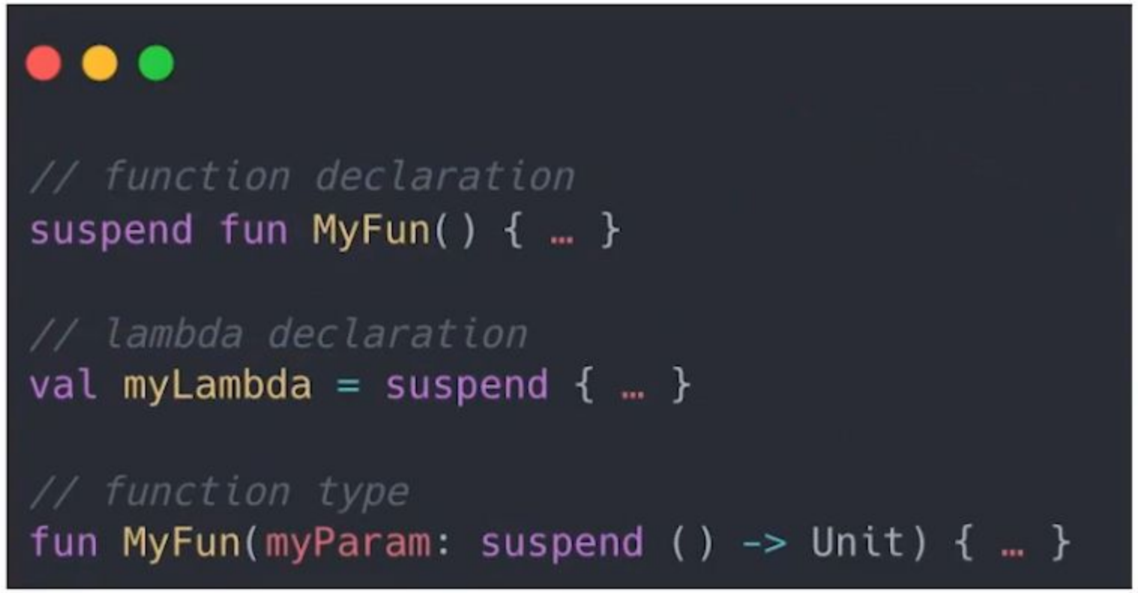
penggunaan **annotation** di compose dibantu dengan kotlin compiler plugin untuk mempercepat proses compile daripada menggunakan **annotation processor**



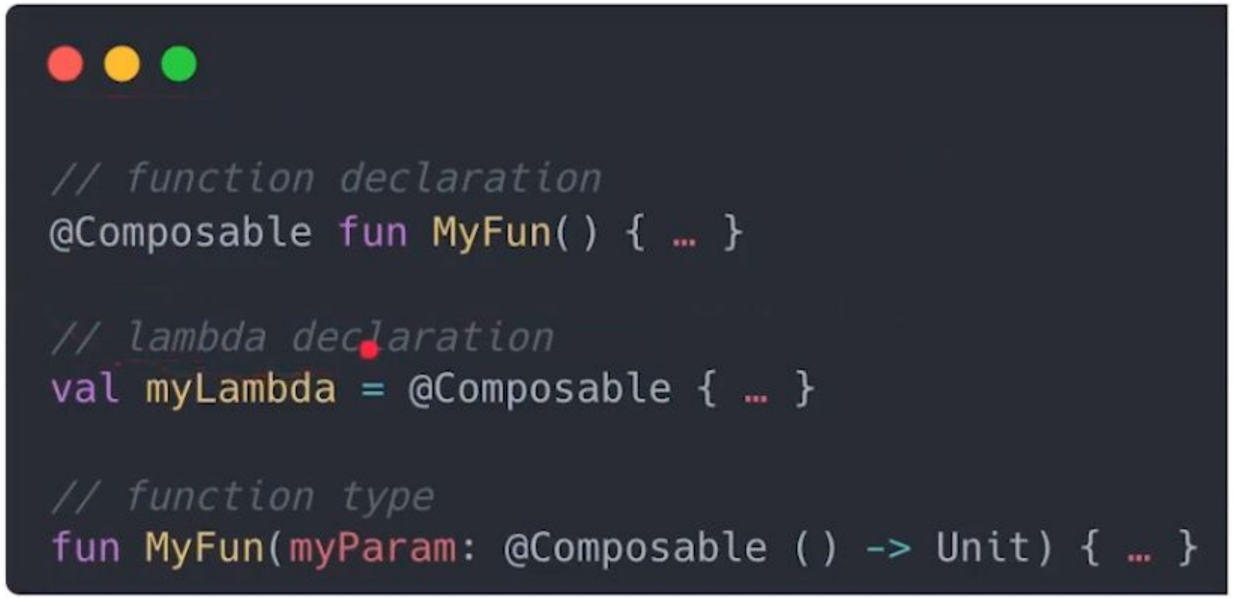
```
@Composable
fun JetpackCompose() {
```

arti @composable

annotation di compose, mirip dengan keyword **suspend,** dimana suspend bisa menjadi type function, lambda, maupun type kembalian

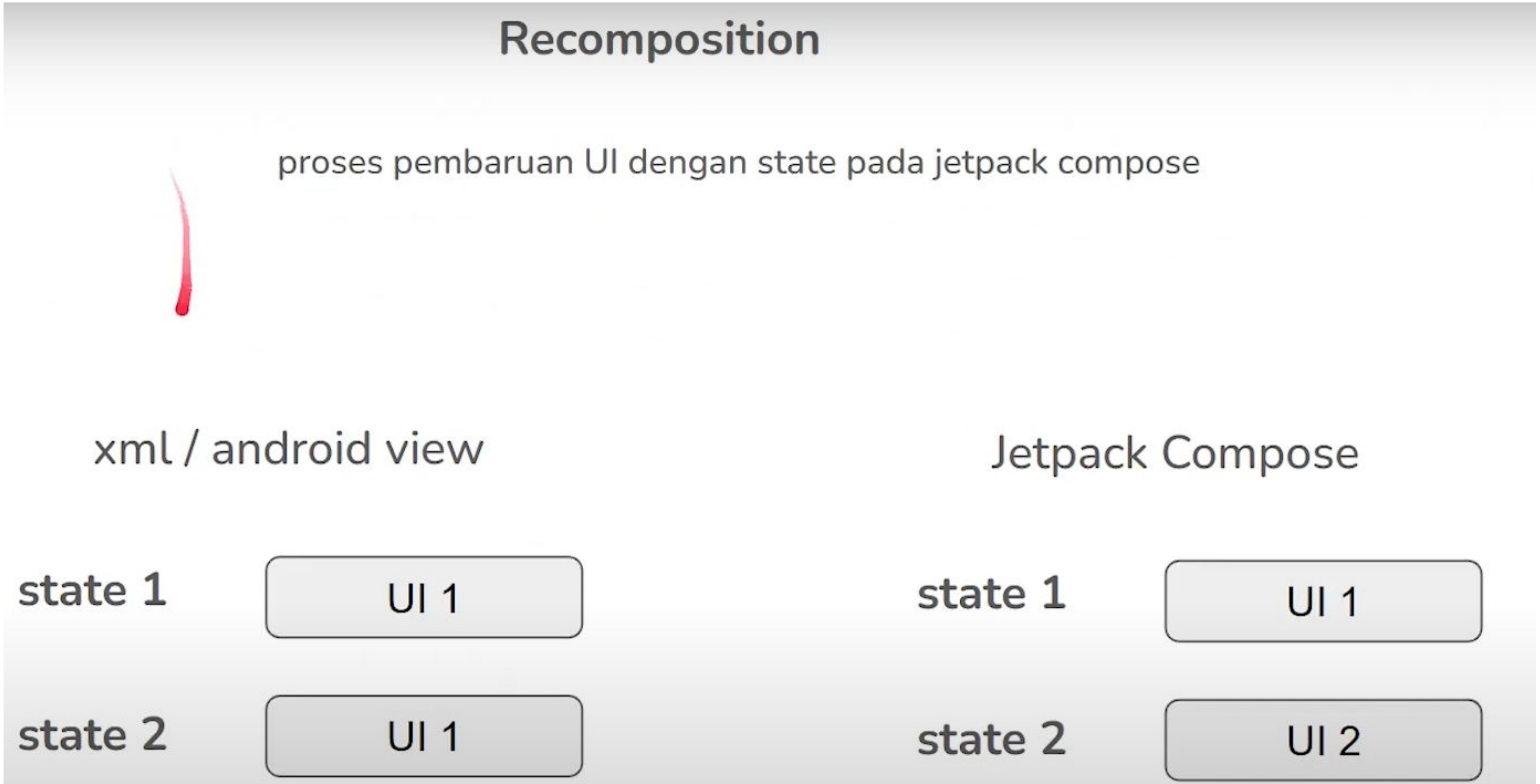


```kotlin
// function declaration
suspend fun MyFun() { … }

// lambda declaration
val myLambda = suspend { … }

// function type
fun MyFun(myParam: suspend () -> Unit) { … }
```



```kotlin
// function declaration
@Composable fun MyFun() { … }

// lambda declaration
val myLambda = @Composable { … }

// function type
fun MyFun(myParam: @Composable () -> Unit) { … }
```

Recomposition



Recomposition

best practice pada saat menggunakan rekomposisi fungsi composable

Fast

hindari proses yang berat seperti konek API pada func compose, karena bisa menyebabkan lag

Indempotent

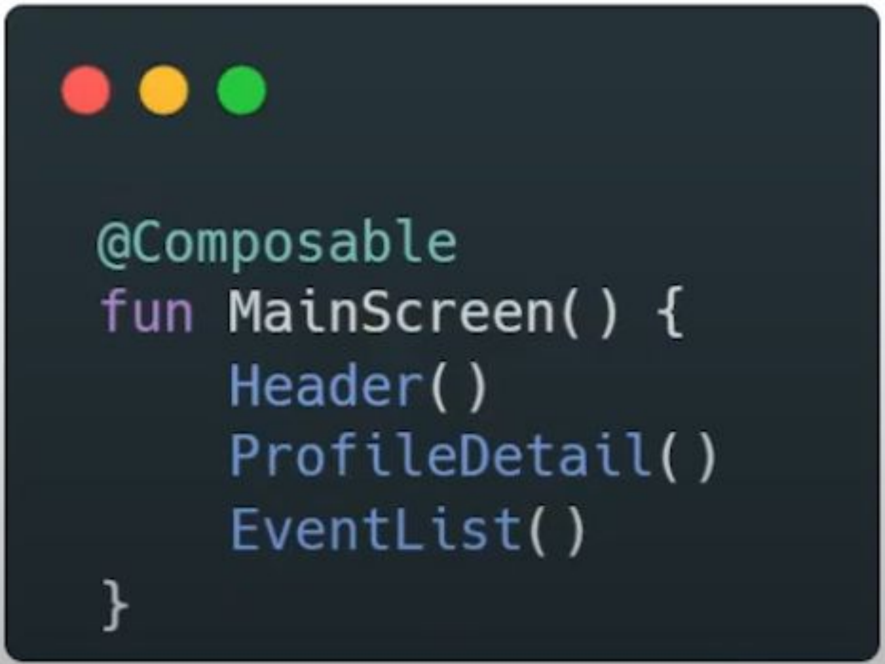
menghasilkan output yang sama selama input yang diberikan sama (konsistensi)

Side-effectfree

hindari state dari luar func compose, karena akan menggangu jalanya proses recomposition

Composable dapat dijalankan pada urutan yang berbeda

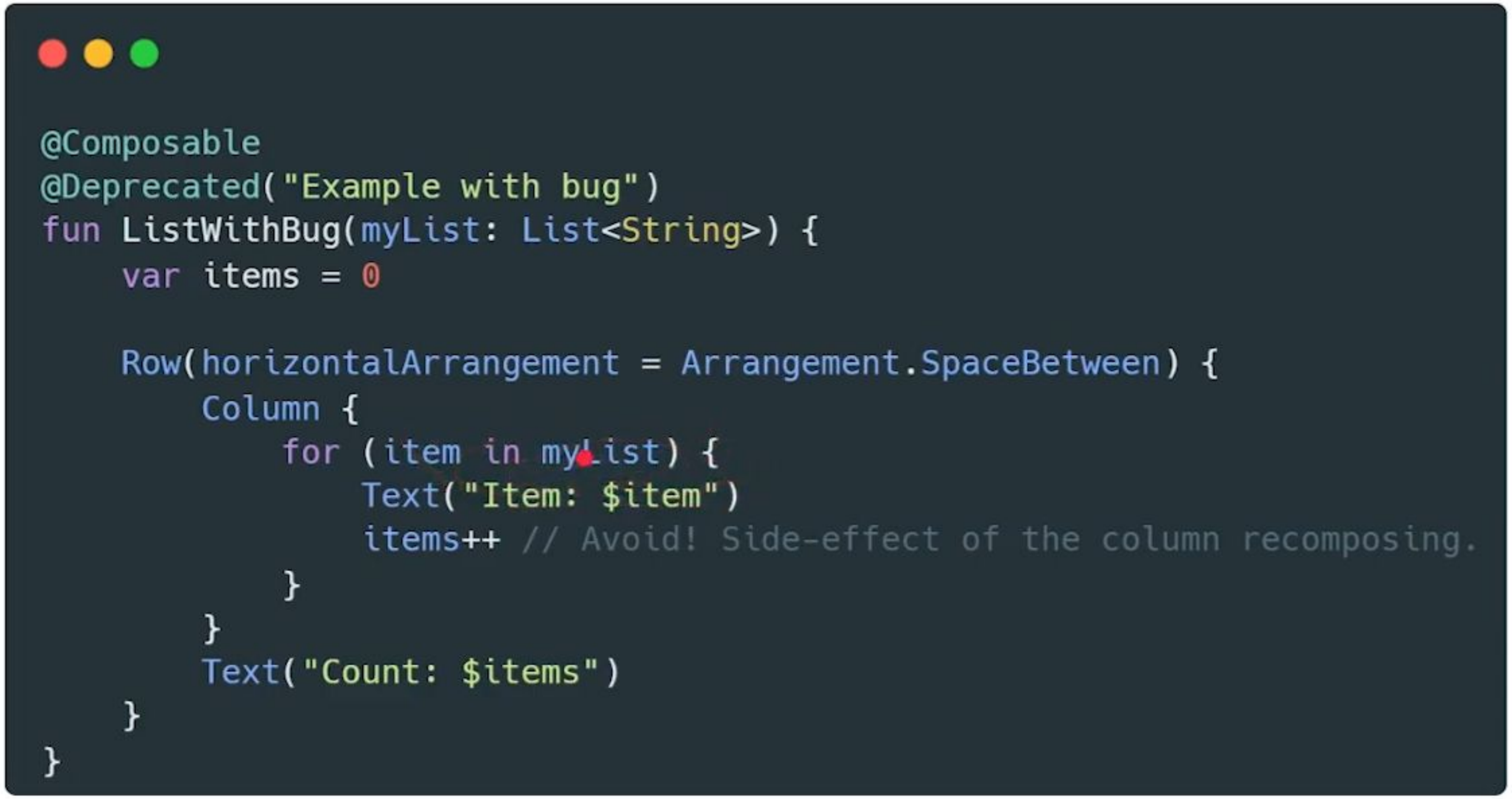
sistem akan memilih prioritas tertinggi pada setipa func compose, sehingga pastikan setiap function bersifat independen



```
@Composable
fun MainScreen() {
    Header()
    ProfileDetail()
    EventList()
}
```

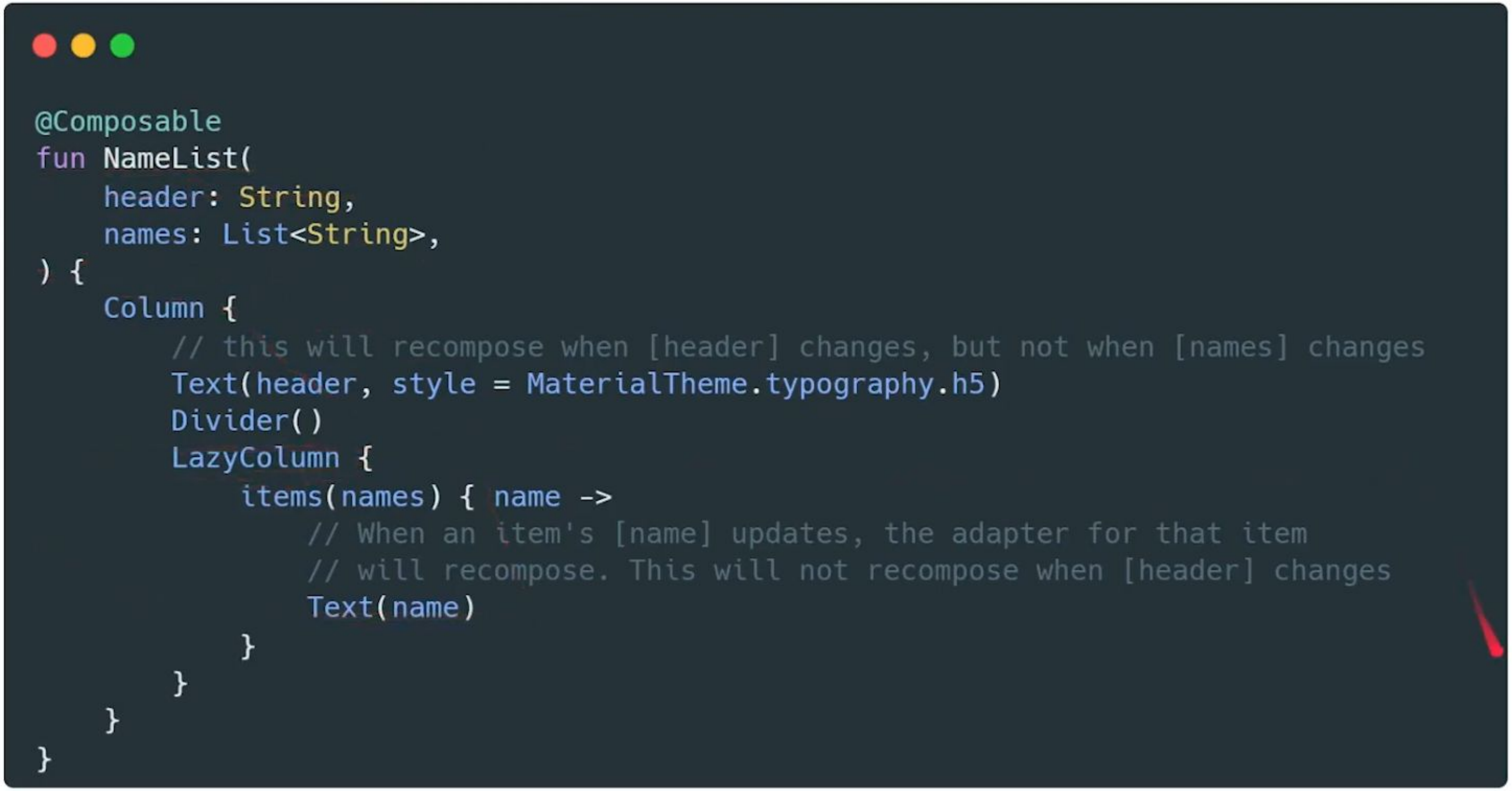
Composable dapat dapat berjalan secara paralel

karena bisa berjalan secara paralel, sangat disarankan untuk tidak menggunakan state diluar compose, karena dapat menimbulkan side effect

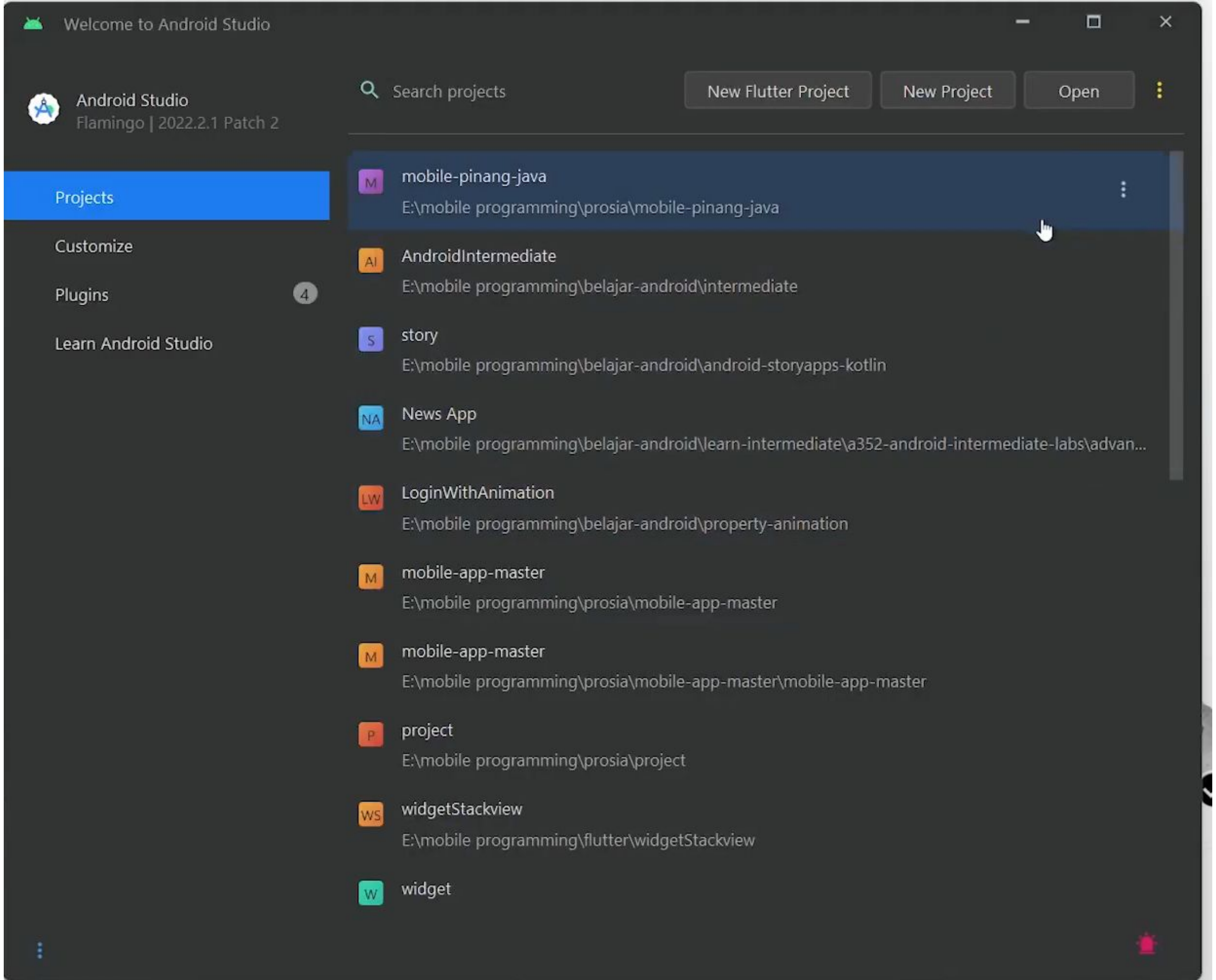


```kotlin
@Composable
@Deprecated("Example with bug")
fun ListWithBug(myList: List<String>) {
    var items = 0

    Row(horizontalArrangement = Arrangement.SpaceBetween) {
        Column {
            for (item in myList) {
                Text("Item: $item")
                items++ // Avoid! Side-effect of the column recomposing.
            }
        }
        Text("Count: $items")
    }
}
```

Composable dapat memilih secara pintar kode mana yang akan dilakukan recomposition



```kotlin
@Composable
fun NameList(
    header: String,
    names: List<String>,
) {
    Column {
        // this will recompose when [header] changes, but not when [names] changes
        Text(header, style = MaterialTheme.typography.h5)
        Divider()
        LazyColumn {
            items(names) { name ->
                // When an item's [name] updates, the adapter for that item
                // will recompose. This will not recompose when [header] changes
                Text(name)
            }
        }
    }
}
```



mobile-pinang-java
E:\mobile programming\prosia\mobile-pinang-java

AndroidIntermediate
E:\mobile programming\belajar-android\intermediate

story
E:\mobile programming\belajar-android\android-storyapps-kotlin

News App
E:\mobile programming\belajar-android\learn-intermediate\a352-android-intermediate-labs\advan...

LoginWithAnimation
E:\mobile programming\belajar-android\property-animation

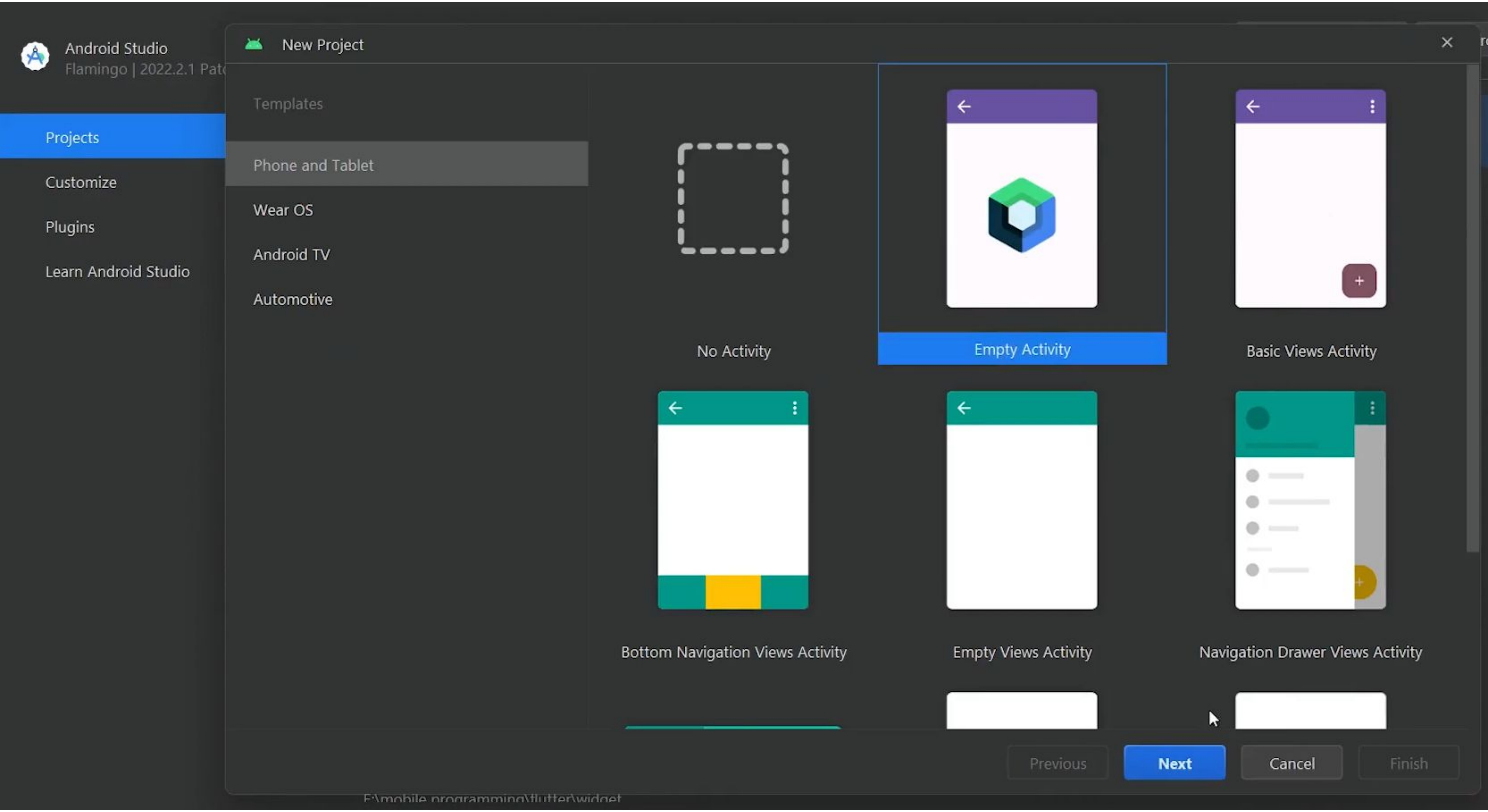
mobile-app-master
E:\mobile programming\prosia\mobile-app-master

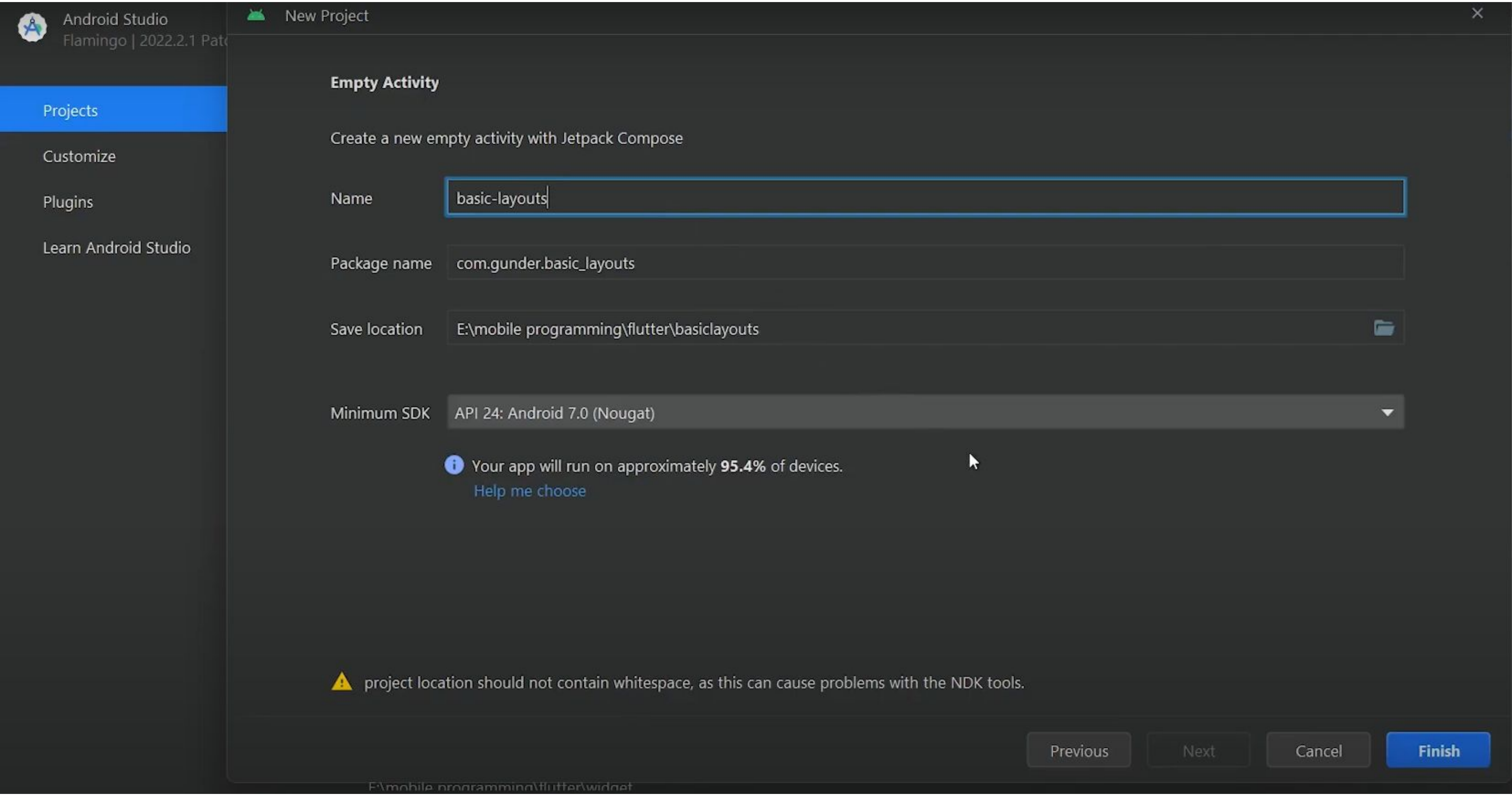
mobile-app-master
E:\mobile programming\prosia\mobile-app-master\mobile-app-master

project
E:\mobile programming\prosia\project

widgetStackview
E:\mobile programming\flutter\widgetStackview

widget


Android Studio
Flamingo | 2022.2.1 Pat
Projects
Customize
Plugins
Learn Android Studio
New Project
Templates
Phone and Tablet
Wear OS
Android TV
Automotive
No Activity
Empty Activity
Basic Views Activity
Bottom Navigation Views Activity
Empty Views Activity
Navigation Drawer Views Activity
Previous
Next
Cancel
Finish
F:\mobile programming\flutter\widget



Empty Activity

Create a new empty activity with Jetpack Compose

Name: basic-layouts

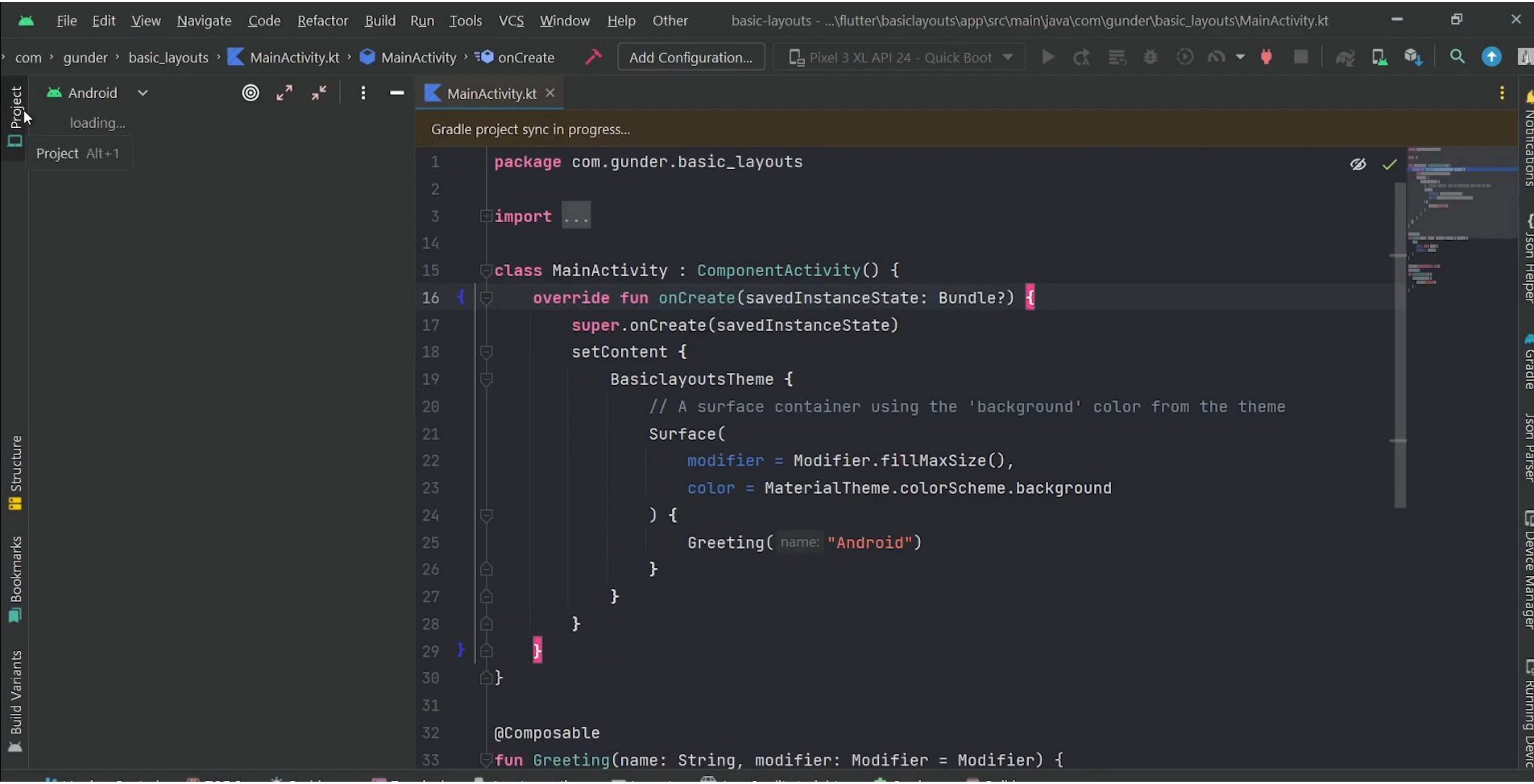
Package name: com.gunder.basic_layouts

Save location: E:\mobile programming\flutter\basiclayouts

Minimum SDK: API 24: Android 7.0 (Nougat)

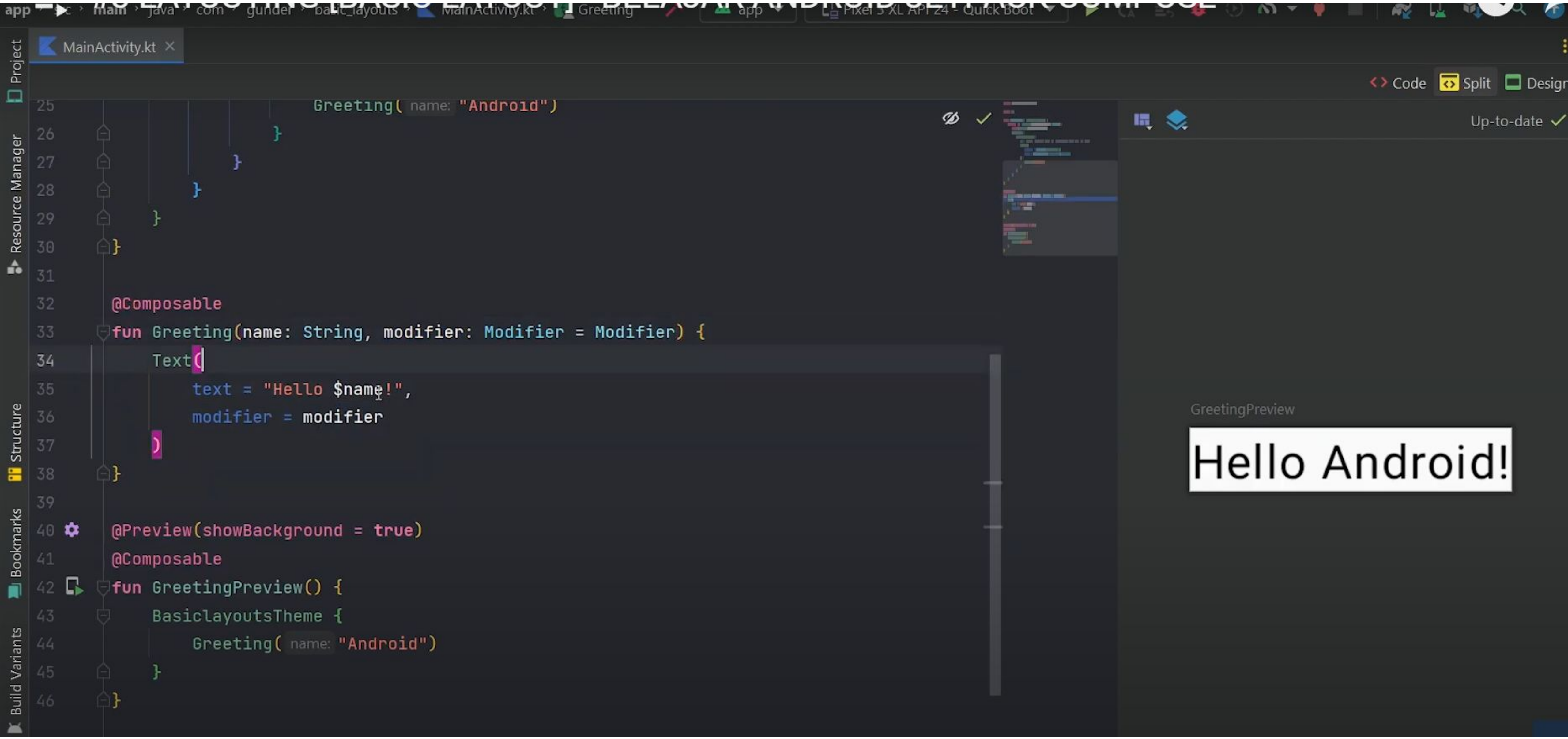
Your app will run on approximately **95.4%** of devices.
Help me choose

project location should not contain whitespace, as this can cause problems with the NDK tools.


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help Other
basic-layouts - ...\flutter\basiclayouts\app\src\main\java\com\gunder\basic_layouts\MainActivity.kt
com › gunder › basic_layouts › MainActivity.kt › MainActivity › onCreate
Add Configuration...
Pixel 3 XL API 24 - Quick Boot
Android
loading...
Project Alt+1
MainActivity.kt
Project
Structure
Bookmarks
Build Variants
Notifications
Json Helper
Gradle
Json Parser
Device Manager
Running Devices

Gradle project sync in progress...

```kotlin
package com.gunder.basic_layouts

import ...

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            BasiclayoutsTheme {
                // A surface container using the 'background' color from the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colorScheme.background
                ) {
                    Greeting( name: "Android")
                }
            }
        }
    }
}

@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
```



```
                    Greeting( name: "Android")
                }
            }
        }
    }
}

@Composable
fun Greeting(name: String, modifier: Modifier = Modifier) {
    Text(
        text = "Hello $name!",
        modifier = modifier
    )
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    BasiclayoutsTheme {
        Greeting( name: "Android")
    }
}
```