Android Jetpack Compose

Less code. More expressive UIs

Kevin Perdana

Struktur Dasar

Blok-blok kode ini terdiri dari beberapa bagian penting yang membentuk aplikasi Android. Berikut adalah penjelasan setiap bagian. Silakan buat project baru.

1. **MainActivity**
   MainActivity, kelas utama yang mewarisi dari ComponentActivity. Ini adalah titik masuk aplikasi Anda.
   onCreate, metode yang dipanggil saat aktivitas dibuat. Di sini, Anda menetapkan konten UI menggunakan setContent.
   RowColumnBoxTheme, tema kustom yang didefinisikan di aplikasi Anda. Ini memungkinkan Anda untuk mengatur gaya dan warna untuk aplikasi.
   Surface, komponen yang menyediakan latar belakang untuk konten di dalamnya. Di sini, Modifier.fillMaxSize() digunakan untuk membuat Surface mengisi seluruh layar, dan color = MaterialTheme.colors.background menetapkan warna latar belakang sesuai tema.
   Greeting, memanggil fungsi Greeting dengan parameter string "Android Jetpack Compose".

```
15 </> class MainActivity : ComponentActivity() {
16 ⓞ↑     override fun onCreate(savedInstanceState: Bundle?) {
17             super.onCreate(savedInstanceState)
18             setContent {
19                 RowColumnBoxTheme {
20                     // A surface container using the 'background' color from the theme
21                     Surface(modifier = Modifier.fillMaxSize(), color = MaterialTheme.colors.background) {
22                         Greeting( name: "Android Jetpack Compose")
23                     }
24                 }
25             }
26         }
27 }
```

2. **Greeting Composable**
   Greeting, fungsi composable yang menerima parameter name dan menampilkan teks "Hello" diikuti dengan nama yang diberikan. Text adalah komponen yang digunakan untuk menampilkan teks di UI.

```
29 @Composable
30 fun Greeting(name: String) {
31     Text(text = "Hello $name!")
32 }
```


Kevin Perdana
https://medium.com/@kevinperdana

3. Preview

@Preview, anotasi yang memungkinkan Anda melihat pratinjau dari composable di IDE tanpa menjalankan aplikasi.

DefaultPreview, fungsi composable yang menampilkan pratinjau dari Greeting dengan nama "Android". Ini berguna untuk melihat bagaimana UI Anda akan terlihat tanpa harus menjalankan aplikasi di emulator atau perangkat.

```
34  @Preview(showBackground = true)
35  @Composable
36  fun DefaultPreview() {
37      RowColumnBoxTheme {
38          Greeting( name: "Android")
39      }
40  }
```

Row, Column, Color, Typography, TextField, Modifier, Spacer

Terdapat beberapa komponen layout dasar yang digunakan untuk mengatur tata letak elemen UI. Mari kita bahas masing-masing komponen ini dan bagaimana cara mengaturnya.

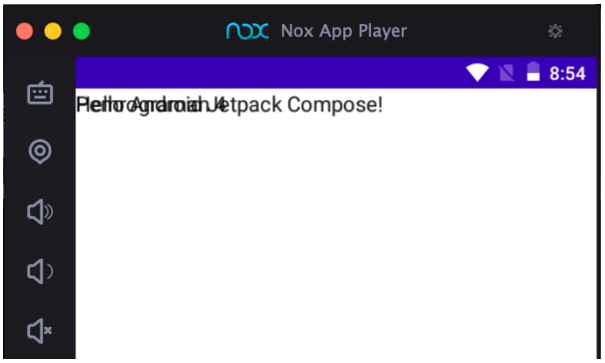
Buat 2 kalimat text seperti berikut.

```
class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            RowColumnBoxTheme {
                // A surface container using the 'background' color from the theme
                Surface(modifier = Modifier.fillMaxSize(), color = MaterialTheme.colors.background) {
                    Greeting( name: "Android Jetpack Compose",  mk: "Pemrograman 4")
                }
            }
        }
    }
}

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

@Preview(showBackground = true)
@Composable
fun DefaultPreview() {
    RowColumnBoxTheme {
        Greeting( name: "x",  mk: "xx")
    }
}
```

Output :

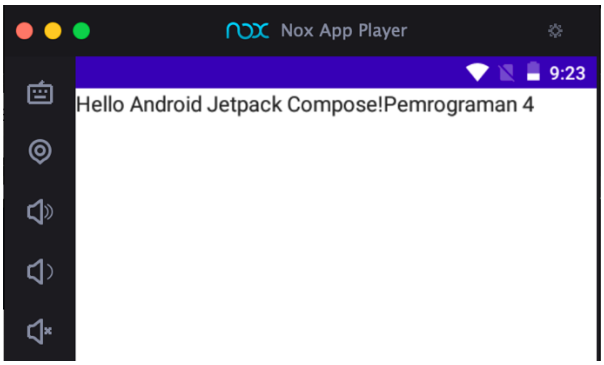


Kevin Perdana
https://medium.com/@kevinperdana

1. Row

Anda dapat menggunakan Row untuk menampilkan elemen UI secara horizontal. Row adalah komponen yang memungkinkan Anda untuk menyusun elemen secara berurutan dari kiri ke kanan.

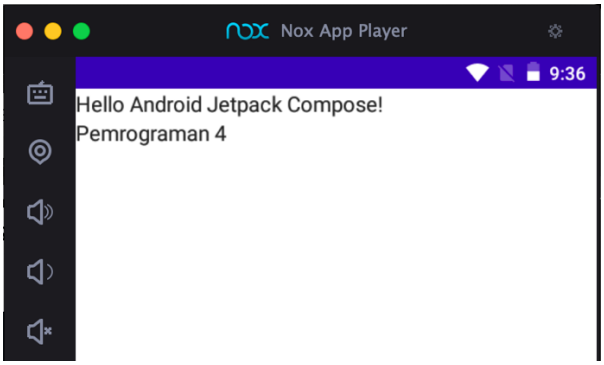
```
30  @Composable
31  fun Greeting(name: String, mk: String) {
32      Row { this: RowScope
33          Text(text = "Hello $name!")
34          Text(text = mk)
35      }
36  }
```



2. Column

Column adalah komponen yang digunakan untuk menampilkan elemen UI secara vertikal. Dengan menggunakan Column, Anda dapat menyusun elemen secara berurutan dari atas ke bawah.

```
31  @Composable
32  fun Greeting(name: String, mk: String) {
33      Column { this: ColumnScope
34          Text(text = "Hello $name!")
35          Text(text = mk)
36      }
37  }
```



3. Color

Menentukan warna bisa menggunakan MaterialTheme.

MainActivity.kt | Theme.kt | Color.kt

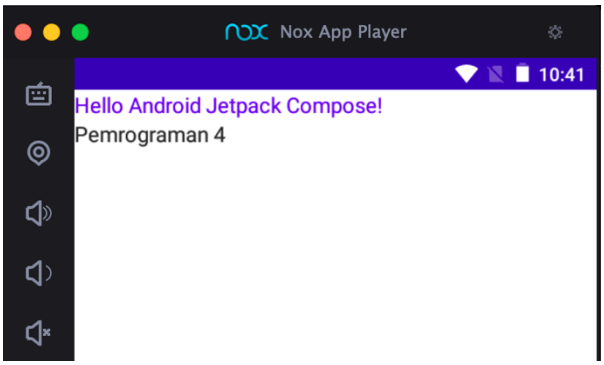
```
31
32  @Composable
33  fun Greeting(name: String, mk: String) {
34      Column { this: ColumnScope
35          Text(
36              text = "Hello $name!",
37              color = MaterialTheme.colors.primary
38          )
39          Text(text = mk)
40      }
41  }
```

MainActivity.kt | Theme.kt | Color.kt

```
3  import ...
8
9  private val DarkColorPalette = darkColors(
10     primary = Purple200,
11     primaryVariant = Purple700,
12     secondary = Teal200
13 )
14
15 private val LightColorPalette = lightColors(
16     primary = Purple500,
17     primaryVariant = Purple700,
18     secondary = Teal200
```

MainActivity.kt | Theme.kt | Color.kt

```
1  package com.example.rowcolumnbox.ui.theme
2
3  import androidx.compose.ui.graphics.Color
4
5  val Purple200 = Color( color: 0xFFBB86FC)
6  val Purple500 = Color( color: 0xFF6200EE)
7  val Purple700 = Color( color: 0xFF3700B3)
8  val Teal200 = Color( color: 0xFF03DAC5)
```



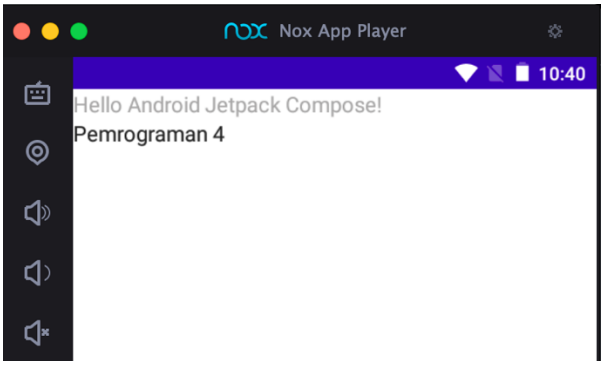
Atau bisa menambahkan warna sendiri pada **Color.kt**

Color.kt

```
1  package com.example.rowcolumnbox.ui.theme
2
3  import androidx.compose.ui.graphics.Color
4
5  val Purple200 = Color( color: 0xFFBB86FC)
6  val Purple500 = Color( color: 0xFF6200EE)
7  val Purple700 = Color( color: 0xFF3700B3)
8  val Teal200 = Color( color: 0xFF03DAC5)
9  val Silver01 = Color( color: 0xFFA7A6A6)
```

MainActivity.kt

```
31
32 @Composable
33 fun Greeting(name: String, mk: String) {
34     Column { this: ColumnScope
35         Text(
36             text = "Hello $name!",
37             color = Silver01
38         )
39         Text(text = mk)
40     }
41 }
```

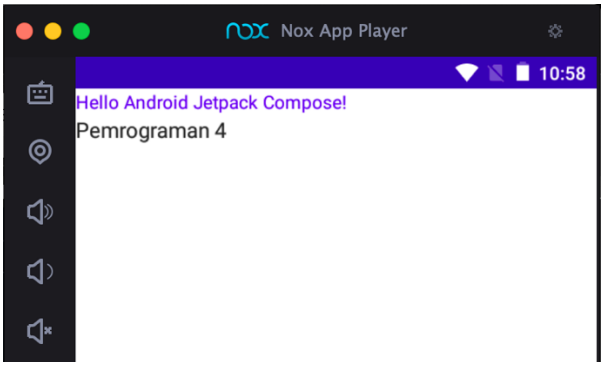


4. Typography

Gaya tipografi tersedia di MaterialTheme, cukup tambahkan ke komposisi Teks.

```
MainActivity.kt

32  @Composable
33  fun Greeting(name: String, mk: String) {
34      Column { this: ColumnScope
35          Text(
36              text = "Hello $name!",
37              color = MaterialTheme.colors.primary,
38              style = MaterialTheme.typography.subtitle2
39          )
40          Text(text = mk)
41      }
42  }
```

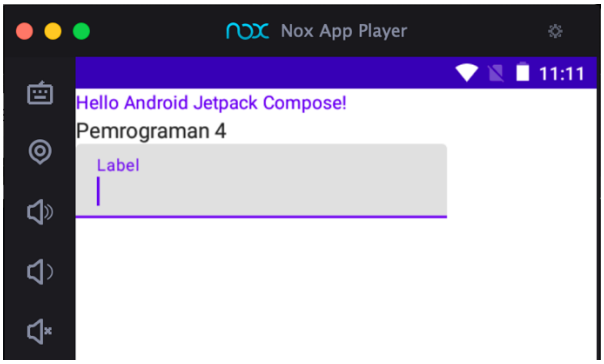


5. TextField

TextField memungkinkan pengguna memasukkan teks.

```
MainActivity.kt

33  @Composable
34  fun Greeting(name: String, mk: String) {
35      var text1 by remember { mutableStateOf( value: "") }
36      Column { this: ColumnScope
37          Text(
38              text = "Hello $name!",
39              color = MaterialTheme.colors.primary,
40              style = MaterialTheme.typography.subtitle2
41          )
42          Text(text = mk)
43          TextField(
44              value = text1,
45              onValueChange = { text1 = it },
46              label = { Text( text: "Label") }
47          )
48      }
49  }
```



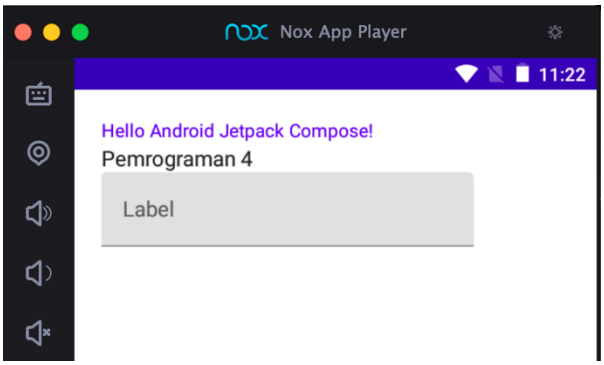
Kevin Perdana
https://medium.com/@kevinperdana

6. Modifier

Modifier memungkinkan Anda untuk menghias atau menambah komposisi. Modifier memungkinkan Anda melakukan hal-hal berikut, mengubah ukuran, tata letak, perilaku, dan tampilan komposisi.

MainActivity.kt

```
35  @Composable
36  fun Greeting(name: String, mk: String) {
37      var text1 by remember { mutableStateOf( value: "") }
38      Column (
39          modifier = Modifier.padding(all = 20.dp)
40      ) { this: ColumnScope
41          Text(
42              text = "Hello $name!",
43              color = MaterialTheme.colors.primary,
44              style = MaterialTheme.typography.subtitle2
45          )
46          Text(text = mk)
47          TextField(
48              value = text1,
49              onValueChange = { text1 = it },
50              label = { Text( text: "Label") }
51          )
52      }
53  }
```



7. Spacer

Gunakan Spacer saat Anda perlu menambahkan spasi di antara komponen tata letak seperti Row atau Column. Spacer berfungsi sebagai composable tak terlihat yang hanya mengisi ruang.

```
MainActivity.kt
47      Text(text = mk)
48      Spacer(
49          modifier = Modifier.padding(vertical = 5.dp)
50      )
51      TextField(
52          value = text1,
53          onValueChange = { text1 = it },
54          label = { Text( text: "Label") }
55      )
```