



Jetpack Compose

PRAKTIKUM PEMROGRAMAN MOBILE #3

Wanda Gusdya Purnama
TEKNIK INFORMATIKA UNIVERSITAS PASUNDAN
2025

Pendahuluan

Semua penyebab masalah yang muncul pada pendekatan pembuatan UI menggunakan XML terdapat dalam cara Android View membangun state-nya dan menampilkan dirinya serta subkelasnya. Untuk menghindari masalah tersebut, kita perlu menggunakan blok program yang berbeda. Di Jetpack Compose, blok program tersebut disebut fungsi composable.

Berikut adalah contoh fungsi composable:

```
@Composable
fun MyComposableFunction() {
    // TODO
}
```

Kita perlu memberi anotasi `@Composable` pada fungsi tersebut. Fungsi apa pun yang dianotasi dengan cara ini disebut fungsi composable, karena kita dapat menyusunnya (compose) di dalam fungsi composable lainnya. Anotasi menyederhanakan kode dengan melampirkan metadata ke dalamnya. Javac, kompiler java, menggunakan kakas pemroses anotasi untuk memindai dan memproses anotasi pada saat kompilasi. Proses tersebut membuat file sumber baru dengan metadata yang ditambahkan. Singkatnya, dengan menggunakan anotasi, kita dapat menambahkan perilaku ke kelas dan menghasilkan kode yang bermanfaat, tanpa menulis banyak boilerplate. Anotasi khusus ini mengubah jenis fungsi tersebut menjadi composable, artinya hanya fungsi composable lainnya yang dapat memanggilnya.

Latihan 1

Pada latihan ini, kita akan mempelajari fungsi composable dan prinsip event-driven programming dengan sebuah form login. Bukalah kembali project HelloCompose kemudian hapus fungsi Greeting, kemudian buatlah fungsi composable FormLogin di MainActivity dengan kode seperti berikut.

```

@Composable
fun FormLogin(modifier: Modifier = Modifier) {
    val username = remember { mutableStateOf(TextFieldValue("")) }
    val password = remember { mutableStateOf(TextFieldValue("")) }

    Column(modifier = modifier
        .fillMaxWidth()) {
        Text(text = "Username", modifier = Modifier.padding(4.dp).fillMaxWidth())
        TextField(value = username.value, onValueChange = {
            username.value = it
        }, modifier = Modifier.padding(4.dp).fillMaxWidth())

        Text(text = "Password", modifier = Modifier.padding(4.dp).fillMaxWidth())
        TextField(value = password.value,
            visualTransformation = PasswordVisualTransformation(),
            onValueChange = {
                password.value = it
            }, keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Password),
            modifier = Modifier.padding(4.dp).fillMaxWidth())

        val loginButtonColors = ButtonDefaults.buttonColors(
            containerColor = Purple80,
            contentColor = Pink40
        )

        val resetButtonColors = ButtonDefaults.buttonColors(
            containerColor = Pink40,
            contentColor = Purple80
        )

        Spacer(modifier = Modifier.weight(1f).width(0.dp))

        Row(modifier = Modifier.padding(4.dp).fillMaxWidth()) {
            Button(modifier = Modifier.weight(5f), onClick = {

            }, colors = loginButtonColors) {
                Text(
                    text = "Login",
                    style = TextStyle(
                        color = Color.White,
                        fontSize = 18.sp
                    ), modifier = Modifier.padding(8.dp)
                )
            }

            Button(modifier = Modifier.weight(5f), onClick = {

            }, colors = resetButtonColors) {
                Text(
                    text = "Reset",
                    style = TextStyle(
                        color = Color.White,
                        fontSize = 18.sp
                    ), modifier = Modifier.padding(8.dp)
                )
            }
        }
    }
}

```

Warna Purple80 dan Pink40 berasal dari file Color di package ui.theme berikut.

```
private val DarkColorScheme = darkColorScheme(
    primary = Purple80,
    secondary = PurpleGrey80,
    tertiary = Pink80
)

private val LightColorScheme = lightColorScheme(
    primary = Purple40,
    secondary = PurpleGrey40,
    tertiary = Pink40

    /* Other default colors to override
    background = Color(0xFFFFFBFE),
    surface = Color(0xFFFFFBFE),
    onPrimary = Color.White,
    onSecondary = Color.White,
    onTertiary = Color.White,
    onBackground = Color(0xFF1C1B1F),
    onSurface = Color(0xFF1C1B1F),
    */
)
```

Warna-warna tersebut dapat diganti dengan warna apapun sesuai keinginan.

File MainActivity akan terlihat seperti ini.

```
package id.ac.unpas.hellocompose

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.activity.enableEdgeToEdge
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.width
import androidx.compose.foundation.text.KeyboardOptions
import androidx.compose.material3.Button
import androidx.compose.material3.ButtonDefaults
import androidx.compose.material3.Scaffold
import androidx.compose.material3.Text
import androidx.compose.material3.TextField
import androidx.compose.runtime.Composable
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.input.KeyboardType
import androidx.compose.ui.text.input.PasswordVisualTransformation
import androidx.compose.ui.text.input.TextFieldValue
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import id.ac.unpas.hellocompose.ui.theme.HelloComposeTheme
import id.ac.unpas.hellocompose.ui.theme.Pink40
import id.ac.unpas.hellocompose.ui.theme.Purple80

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContent {
            HelloComposeTheme {
                Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
                    FormLogin(modifier = Modifier.padding(innerPadding))
                }
            }
        }
    }
}
```

```

@Composable
fun FormLogin(modifier: Modifier = Modifier) {
    val username = remember { mutableStateOf(TextFieldValue("")) }
    val password = remember { mutableStateOf(TextFieldValue("")) }

    Column(modifier = modifier
        .fillMaxWidth()) {
        Text(text = "Username", modifier = Modifier.padding(4.dp).fillMaxWidth())
        TextField(value = username.value, onValueChange = {
            username.value = it
        }, modifier = Modifier.padding(4.dp).fillMaxWidth())

        Text(text = "Password", modifier = Modifier.padding(4.dp).fillMaxWidth())
        TextField(value = password.value,
            visualTransformation = PasswordVisualTransformation(),
            onValueChange = {
                password.value = it
            }, keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Password),
            modifier = Modifier.padding(4.dp).fillMaxWidth())

        val loginButtonColors = ButtonDefaults.buttonColors(
            containerColor = Purple80,
            contentColor = Pink40
        )

        val resetButtonColors = ButtonDefaults.buttonColors(
            containerColor = Pink40,
            contentColor = Purple80
        )

        Spacer(modifier = Modifier.weight(1f).width(8.dp))

        Row(modifier = Modifier.padding(4.dp).fillMaxWidth()) {
            Button(modifier = Modifier.weight(5f), onClick = {

            }, colors = loginButtonColors) {
                Text(
                    text = "Login",
                    style = TextStyle(
                        color = Color.White,
                        fontSize = 18.sp
                    ), modifier = Modifier.padding(8.dp)
                )
            }

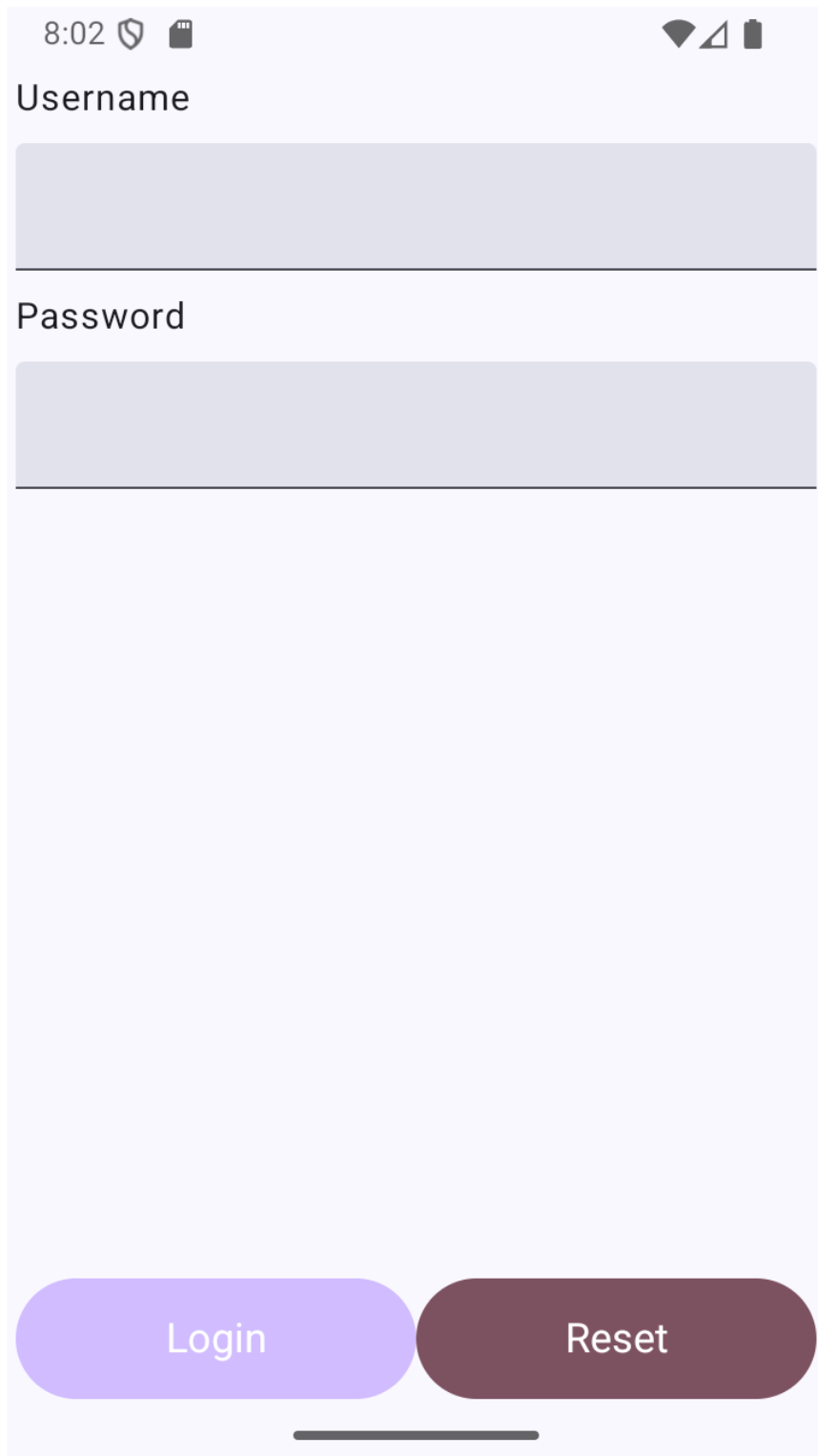
            Button(modifier = Modifier.weight(5f), onClick = {

            }, colors = resetButtonColors) {
                Text(
                    text = "Reset",
                    style = TextStyle(
                        color = Color.White,
                        fontSize = 18.sp
                    ), modifier = Modifier.padding(8.dp)
                )
            }
        }
    }
}

@Preview(showBackground = true)
@Composable
fun GreetingPreview() {
    HelloComposeTheme {
        FormLogin()
    }
}

```

Kode di atas baru membuat sebuah form login yang diatur dalam kolom dan baris seperti pada tangkapan layar berikut.



A mobile application login screen mockup. At the top, a status bar shows the time 8:02, a shield icon, a SIM card icon, a Wi-Fi signal icon, and a battery level icon. Below the status bar, the word "Username" is displayed in a dark font. Underneath is a light gray rectangular input field. Below the input field, the word "Password" is displayed in a dark font. Underneath is another light gray rectangular input field. At the bottom of the screen, there are two rounded rectangular buttons: a light purple button labeled "Login" and a dark purple button labeled "Reset". A thin horizontal line is visible at the very bottom of the screen.

Column (kolom) dapat menampung komponen GUI dalam baris-baris secara vertikal, sedangkan Row (baris) dapat menampung komponen GUI dalam kolom-kolom secara horizontal. Kontainer Column dan Row tersebut dapat mengatur komponen berdasarkan tinggi (height) dan lebar (width), atau dengan bobot (weight). Pada contoh di atas, komponen Spacer digunakan sebagai pemisah yang memiliki weight 1f (f untuk float seperti di Java), sedangkan komponen lain dalam Column tidak memiliki weight sehingga komponen Divider akan memenuhi sisa ruang dalam Column. Jika setiap komponen dalam Column atau Row memiliki weight, maka tinggi (untuk Column)/lebar (untuk Row) komponen-komponen tersebut akan diatur sesuai dengan persentase weight-nya. Contohnya pada contoh di atas, kedua Button memiliki weight 5f sehingga lebar masing-masing Button adalah 50% (5/10) dari lebar Row.

Selanjutnya, tambahkan fungsionalitas untuk form login yang kita buat dengan mengisi event handler onClick pada masing-masing button. Pertama, buat variabel context di fungsi FormLogin untuk mengambil context aplikasi (pemegang lifecycle).

```
@Composable
fun FormLogin(modifier: Modifier = Modifier) {
    val context = LocalContext.current
```

Kemudian, untuk Button dengan text Login, isi fungsi onClick dengan kode berikut.


```

Button(modifier = Modifier.weight(5f), onClick = {
    if (username.value.text == "admin" && password.value.text == "admin") {
        Toast.makeText(context, "Login Sukses", Toast.LENGTH_LONG).show()
    } else {
        Toast.makeText(context, "Login Gagal", Toast.LENGTH_LONG).show()
    }
}, colors = loginButtonColors) {
    Text(
        text = "Login",
        style = TextStyle(
            color = Color.White,
            fontSize = 18.sp
        ), modifier = Modifier.padding(8.dp)
    )
}

```

Sedangkan untuk Button dengan text Reset, isi fungsi onClick dengan kode berikut.

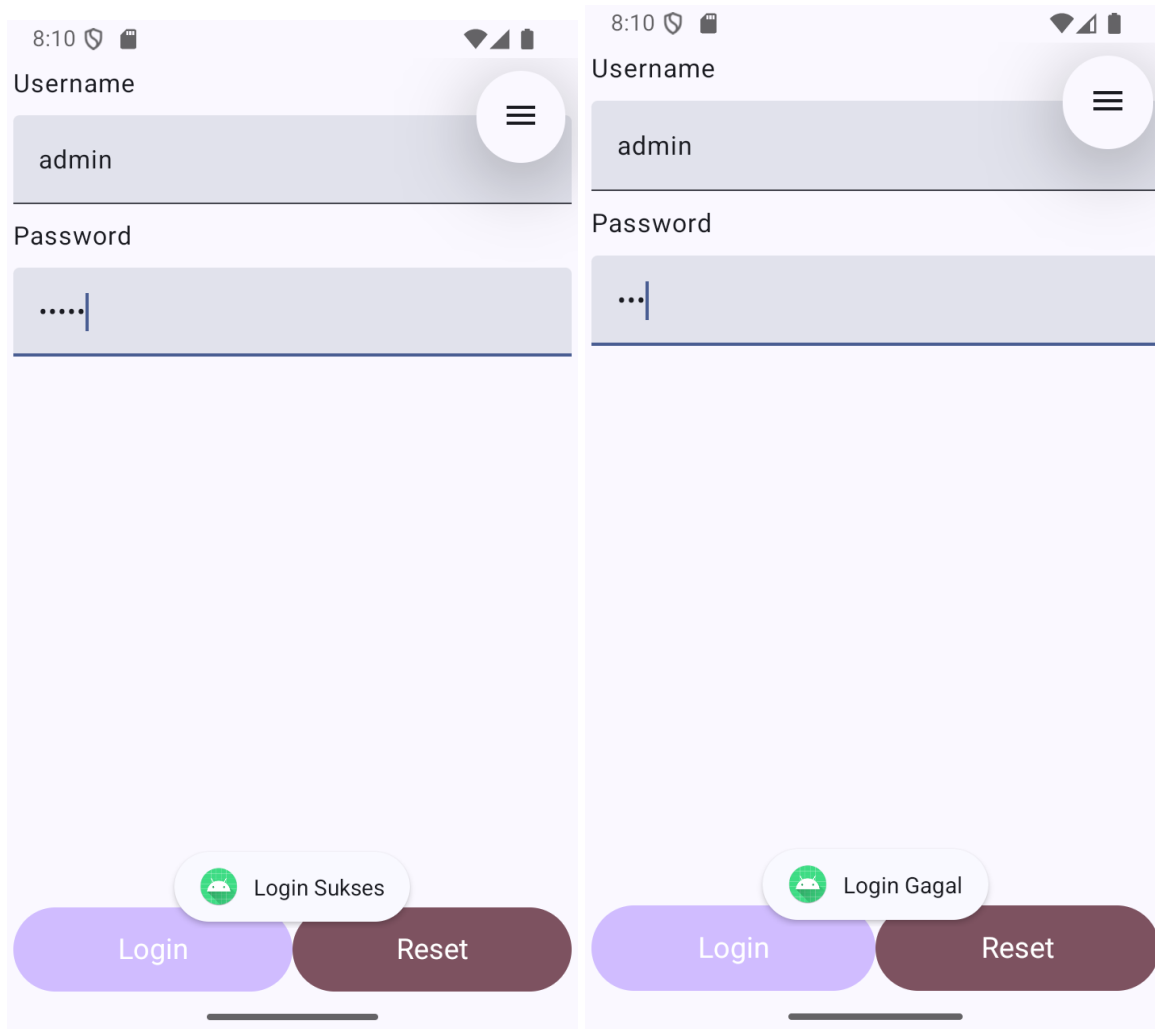
```
Button(modifier = Modifier.weight(5f), onClick = {  
    username.value = TextFieldValue("")  
    password.value = TextFieldValue("")  
}, colors = resetButtonColors) {  
    Text(  
        text = "Reset",  
        style = TextStyle(  
            color = Color.White,  
            fontSize = 18.sp  
        ), modifier = Modifier.padding(8.dp)  
    )  
}
```

Berikut ini adalah perubahan deklarasi import di awal kelas MainActivity.

```
package id.ac.unpas.hellocompose

import android.os.Bundle
import android.widget.Toast
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.activity.enableEdgeToEdge
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.layout.width
import androidx.compose.foundation.text.KeyboardOptions
import androidx.compose.material3.Button
import androidx.compose.material3.ButtonDefaults
import androidx.compose.material3.Scaffold
import androidx.compose.material3.Text
import androidx.compose.material3.TextField
import androidx.compose.runtime.Composable
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.input.KeyboardType
import androidx.compose.ui.text.input.PasswordVisualTransformation
import androidx.compose.ui.text.input.TextFieldValue
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import id.ac.unpas.hellocompose.ui.theme.HelloComposeTheme
import id.ac.unpas.hellocompose.ui.theme.Pink40
import id.ac.unpas.hellocompose.ui.theme.Purple80
```

Jalankan aplikasi, kemudian coba masukkan input username dan password. Jika aplikasi menerima input username = admin dan password = admin, maka akan muncul pesan Login Sukses, jika selain itu akan muncul pesan Login Gagal. Tombol Reset dapat digunakan untuk menghapus inputan.



Latihan 2

Buatlah aplikasi baru untuk menerima form registrasi dengan isian nama, username, nomor telepon, email, dan alamat rumah. Cobalah eksplorasi KeyboardOptions yang cocok untuk nomor telepon dan email. Tambahkan dua buah tombol yaitu Simpan, yang jika semua inputan diisi akan memunculkan pesan Halo, \$nama. Jika ada inputan yang tidak diisi, aplikasi harus menampilkan pesan bahwa semua inputan harus diisi. Tombol kedua adalah tombol Reset yang dapat digunakan untuk menghapus semua inputan.