Tata Letak Adaptif

Tata letak adaptif adalah kemampuan aplikasi untuk menyesuaikan tampilannya secara otomatis sesuai dengan ukuran layar perangkat yang berbeda-beda, seperti ponsel, tablet, atau perangkat lipat. Dengan tata letak adaptif, aplikasi Anda akan terlihat dan berfungsi dengan baik di berbagai jenis perangkat. Dengan Compose, tata letak adaptif bisa dilakukan secara mudah dengan bantuan **`BoxWithConstraints`**, **`Modifier`**, serta **`WindowMetrics`**.

Langkah-langkah Membuat Aplikasi Tata Letak Adaptif dengan Jetpack Compose

1. Persiapkan Project

1. **Buka Android Studio**.
2. Klik **"Start a new Android Studio project"**.
3. Pilih template **"Empty Compose Activity"**, lalu klik **"Next"**.
4. Isi detail proyek:
    - **Name**: `Tata Letak Adaptif`
    - **Package name**: `com.example.tataletakadaptif`
    - **Save location**: Pilih lokasi yang diinginkan.
    - **Language**: **Kotlin**
    - **Minimum SDK**: Pilih sesuai kebutuhan.
5. Klik **"Finish"** dan tunggu hingga proyek selesai dibuat.

2. Buat Layout untuk Layar Kecil dan Layar Besar

Untuk implementasi, buat dua layout yang berbeda untuk menyesuaikan layar kecil dan besar.

1. **Buat File Kotlin untuk Layar Kecil**:
    - Klik kanan pada package `com.example.tataletakadaptif`.
    - Pilih **New** > **Kotlin File/Class**.
    - Beri nama `ColumnLayout`.
    - Beri pilihan **File**

```
package com.example.tataletakapadtif

import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.material3.Button
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.sp
```

```
@Composable
fun ColumnLayout() {
    Column(
        modifier = Modifier.fillMaxSize(),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {
        Text(text = "Layar Kecil", fontSize = 24.sp)
        Button(onClick = { /* Aksi */ }) {
            Text(text = "Tombol 1")
        }
        Button(onClick = { /* Aksi */ }) {
            Text(text = "Tombol 2")
        }
    }
}
```

2. **Buat File Kotlin untuk Layar Besar:**
    - Klik kanan pada package `com.example.tataletakadaptif`.
    - Pilih **New** > **Kotlin File/Class**.
    - Beri nama `RowLayout`.
    - Beri pilihan **File**

```
package com.example.tataletakapadtif

import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.material3.Button
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.sp

@Composable
fun RowLayout() {
    Row(
        modifier = Modifier.fillMaxSize(),
        horizontalArrangement = Arrangement.SpaceEvenly,
        verticalAlignment = Alignment.CenterVertically
    ) {
        Text(text = "Layar Besar", fontSize = 24.sp)
        Button(onClick = { /* Aksi */ }) {
            Text(text = "Tombol 1")
        }
        Button(onClick = { /* Aksi */ }) {
            Text(text = "Tombol 2")
        }
```

```
    }
}
```

Pada `ColumnLayout`, komponen diatur secara vertikal untuk layar kecil, sedangkan di `RowLayout`, komponen diatur secara horizontal untuk layar besar.

3. Gunakan BoxWithConstraints untuk Tata Letak Adaptif

Salah satu cara paling umum untuk membuat tata letak adaptif adalah dengan menggunakan `BoxWithConstraints`, yang memungkinkan kita menyesuaikan tampilan berdasarkan ukuran layar.

Dalam MainActivity ubah kode menjadi seperti berikut.

```
package com.example.tataletakapadtif

import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.activity.enableEdgeToEdge
import androidx.compose.foundation.layout.BoxWithConstraints
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.Scaffold
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import com.example.tataletakapadtif.ui.theme.TataLetakApadtifTheme

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

@Composable
fun AdaptiveLayoutApp(modifier: Modifier = Modifier) {
    BoxWithConstraints {
        val screenWidth = maxWidth
        val screenHeight = maxHeight
```

```
        if (screenWidth < 600.dp) {
            // Layout untuk layar kecil (misal: ponsel)
            ColumnLayout()
        } else {
            // Layout untuk layar besar (misal: tablet)
            RowLayout()
        }
    }
}
```

Pada contoh di atas, kita mengecek lebar layar dengan `maxWidth`. Jika lebarnya kurang dari 600dp (misalnya, untuk ponsel), kita menggunakan layout berbentuk **kolom**. Sedangkan jika lebih besar, kita menggunakan layout berbentuk **baris** (misal, untuk tablet).

4. Menggunakan `Configuration` untuk Adaptasi Orientasi

Selain mengandalkan ukuran layar, Anda juga bisa menggunakan `orientation` (orientasi layar) untuk mengatur tata letak.

```
import android.content.res.Configuration
import androidx.compose.ui.platform.LocalConfiguration

@Composable
fun AdaptiveLayoutApp2(modifier: Modifier = Modifier) {
    val configuration = LocalConfiguration.current

    if (configuration.orientation == Configuration.ORIENTATION_LANDSCAPE)
{
        // Layout untuk orientasi landscape
        RowLayout()
    } else {
        // Layout untuk orientasi portrait
        ColumnLayout()
    }
}
```

Di sini kita menggunakan `LocalConfiguration` untuk mendapatkan informasi tentang orientasi layar dan menyesuaikan layout secara dinamis berdasarkan apakah perangkat dalam mode **portrait** atau **landscape**.

5. Testing dengan Berbagai Ukuran Layar

Untuk memastikan tata letak adaptif bekerja dengan baik, Anda bisa menggunakan **Android Emulator** dengan berbagai ukuran layar atau orientasi:

- Uji di emulator ponsel dengan orientasi portrait dan landscape.
- Uji di emulator tablet untuk melihat tata letak pada layar besar.

Penjelasan

- **`BoxWithConstraints`**: Digunakan untuk mendapatkan informasi mengenai ukuran layar dan membuat keputusan layout berdasarkan ukuran tersebut.
- **`LocalConfiguration`**: Digunakan untuk mendapatkan informasi konfigurasi perangkat, termasuk orientasi.
- **`Column`** dan **`Row`**: Ini adalah layout dasar di Compose yang digunakan untuk mengatur elemen secara vertikal (Column) dan horizontal (Row).

Dengan pendekatan ini, aplikasi Anda dapat menyesuaikan tata letaknya secara adaptif berdasarkan ukuran layar dan orientasi, sehingga tampilannya tetap optimal di berbagai perangkat.

Perubahan Tata Letak Berdasarkan Aspect Ratio

Berikut adalah contoh lain untuk membuat tata letak adaptif di aplikasi Android menggunakan **Compose**. Kali ini kita akan menggunakan pendekatan yang lebih dinamis dengan beberapa tata letak berbeda yang beradaptasi berdasarkan ukuran layar, termasuk perubahan tata letak berdasarkan **perbandingan lebar dan tinggi layar** (***aspect ratio***), yang sering berguna ketika menangani berbagai perangkat seperti ponsel, tablet, atau bahkan foldable devices.

1. Persiapkan Project

1. **Buka Android Studio**.
2. Klik **"Start a new Android Studio project"**.
3. Pilih template **"Empty Compose Activity"**, lalu klik **"Next"**.
4. Isi detail proyek:
    - **Name**: `Adaptive Aspect Layout`
    - **Package name**: `com.example.adaptiveaspectlayout`
    - **Save location**: Pilih lokasi yang diinginkan.
    - **Language**: **Kotlin**
    - **Minimum SDK**: Pilih sesuai kebutuhan.
5. Klik **"Finish"** dan tunggu hingga proyek selesai dibuat.

2. Buat Layout Berbeda Berdasarkan Rasio Layar

Buat tiga tata letak yang berbeda untuk ditampilkan sesuai dengan rasio layar.

1. **Buat File Kotlin untuk Layar Portrait**:
    - Klik kanan pada package `com.example.adaptiveaspectlayout`.
    - Pilih **New** > **Kotlin File/Class**.
    - Beri nama `PortraitLayout`.
    - Beri pilihan **File**

```
package com.example.adaptiveaspectlayout

import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.Button
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp

@Composable
fun PortraitLayout() {
    Column(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {
        Text(text = "Portrait Layout", fontSize = 24.sp)
        Button(onClick = { /* Aksi */ }) {
            Text(text = "Tombol 1")
        }
        Button(onClick = { /* Aksi */ }) {
            Text(text = "Tombol 2")
        }
    }
}
```

2. **Buat File Kotlin untuk Layar Persegi:**
    - Klik kanan pada package `com.example.adaptiveaspectlayout`.
    - Pilih **New** > **Kotlin File/Class**.
    - Beri nama `SquareLikeLayout`.
    - Beri pilihan **File**

```
package com.example.adaptiveaspectlayout

import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Box
```

```
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.Button
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp

@Composable
fun SquareLikeLayout() {
    Box(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp),
        contentAlignment = Alignment.Center
    ) {
        Column {
            Text(text = "Square-like Layout", fontSize = 24.sp)
            Row(
                horizontalArrangement = Arrangement.SpaceAround
            ) {
                Button(onClick = { /* Aksi */ }) {
                    Text(text = "Tombol A")
                }
                Button(onClick = { /* Aksi */ }) {
                    Text(text = "Tombol B")
                }
            }
        }
    }
}
```

3. **Buat File Kotlin untuk Landscape:**
    - Klik kanan pada package `com.example.adaptiveaspectlayout`.
    - Pilih **New** > **Kotlin File/Class**.
    - Beri nama `LandscapeLayout`.
    - Beri pilihan **File**

```
package com.example.adaptiveaspectlayout

import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.Button
```

```
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp

@Composable
fun LandscapeLayout() {
    Row(
        modifier = Modifier
            .fillMaxSize()
            .padding(16.dp),
        verticalAlignment = Alignment.CenterVertically,
        horizontalArrangement = Arrangement.SpaceEvenly
    ) {
        Text(text = "Landscape Layout", fontSize = 24.sp)
        Column {
            Button(onClick = { /* Aksi */ }) {
                Text(text = "Tombol X")
            }
            Button(onClick = { /* Aksi */ }) {
                Text(text = "Tombol Y")
            }
        }
    }
}
```

- **PortraitLayout**: Tata letak vertikal dengan teks dan dua tombol.
- **SquareLikeLayout**: Tata letak untuk layar yang hampir persegi, dengan teks di tengah dan dua tombol diatur secara horizontal.
- **LandscapeLayout**: Tata letak horizontal yang cocok untuk layar lebar, dengan teks dan dua tombol diatur secara vertikal.

3. Gunakan `BoxWithConstraints` untuk Mendeteksi Ukuran Layar

Sekali lagi kita akan menggunakan **BoxWithConstraints** untuk mendeteksi ukuran layar, tetapi kali ini kita akan membuat tata letak berdasarkan **aspect ratio** (rasio layar).

Dalam MainActivity tambahkan kode berikut.

```
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.activity.enableEdgeToEdge
import androidx.compose.foundation.layout.BoxWithConstraints
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.padding
import androidx.compose.material3.Scaffold
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import com.example.adaptiveaspectlayout.ui.theme.AdaptiveAspectLayoutTheme
```

```
@Composable
fun AdaptiveAspectLayout(modifier: Modifier = Modifier) {
    BoxWithConstraints {
        val screenWidth = maxWidth
        val screenHeight = maxHeight
        val aspectRatio = screenWidth / screenHeight

        when {
            aspectRatio < 1f -> {
                // Layout untuk layar portrait (tinggi lebih besar dari
lebar)
                PortraitLayout()
            }
            aspectRatio > 1f && aspectRatio <= 1.5f -> {
                // Layout untuk layar dengan aspect ratio hampir persegi
(misal: foldable devices)
                SquareLikeLayout()
            }
            else -> {
                // Layout untuk layar landscape (lebar lebih besar dari
tinggi)
                LandscapeLayout()
            }
        }
    }
}
```

Dalam fungsi setContent, panggil fungsi AdaptiveAspectLayout.

```
setContent {
    AdaptiveAspectLayoutTheme {
        Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
            AdaptiveAspectLayout(
                modifier = Modifier.padding(innerPadding)
            )
        }
    }
}
```

Pada contoh ini, tata letak akan berubah sesuai dengan **rasio layar**:

- **`aspectRatio < 1f`**: Digunakan untuk layar dengan orientasi **portrait**.
- **`aspectRatio > 1f && aspectRatio <= 1.5f`**: Untuk layar dengan rasio hampir **persegi** (misalnya, perangkat yang bisa dilipat).
- **`aspectRatio > 1.5f`**: Untuk layar **landscape** dengan lebar lebih dominan.

6. Menjalankan dan Menguji di Emulator

Anda bisa menguji aplikasi ini di beberapa perangkat emulator yang berbeda untuk melihat hasilnya:

- Coba di ponsel dengan **orientasi portrait**.
- Coba di **foldable devices** atau emulator dengan layar persegi.
- Coba di tablet atau perangkat dengan **orientasi landscape**.

Penjelasan

- **Aspect Ratio**: Kita membandingkan lebar dengan tinggi layar untuk mendapatkan rasio layar (*aspectRatio*). Ini berguna saat menangani berbagai jenis perangkat dengan ukuran layar berbeda (termasuk foldable devices).
- **`BoxWithConstraints`**: Digunakan untuk mendapatkan dimensi layar yang dapat diakses melalui `maxWidth` dan `maxHeight`.
- **Layout yang berbeda**: Dengan memanfaatkan rasionya, kita bisa membuat beberapa tata letak yang lebih sesuai untuk berbagai jenis perangkat.

Keunggulan Pendekatan Ini:

1. **Adaptasi berdasarkan rasio layar**: Pendekatan ini cocok untuk perangkat yang punya berbagai aspek rasio layar.
2. **Dukungan perangkat foldable**: Dengan membedakan tata letak untuk layar persegi dan hampir persegi, aplikasi Anda bisa beradaptasi pada perangkat lipat (foldable).
3. **Kode yang modular**: Setiap tata letak terpisah dalam Composable sendiri, membuat kode lebih terorganisir dan mudah dikelola.

Dengan pendekatan ini, aplikasi Anda akan responsif dan adaptif terhadap berbagai perangkat, memastikan pengalaman pengguna yang optimal pada semua ukuran layar.
