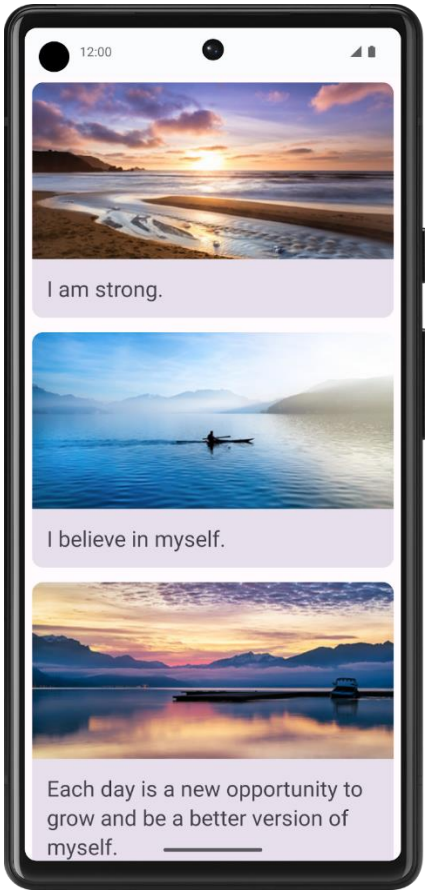
Menambahkan List yang dapat di-scroll

Proyek yang akan dibangun: Aplikasi **Affirmations**



URL kode awal:

https://github.com/google-developer-training/basic-android-kotlin-compose-training-affirmations

Aplikasi diharapkan menampilkan layar kosong saat dibangun dari kode cabang `starter`.

Membuat class data item daftar

Di aplikasi Android, daftar terdiri dari item daftar. Untuk data tunggal, ini bisa berupa hal sederhana seperti string atau bilangan bulat. Untuk item daftar yang memiliki beberapa data, seperti gambar dan teks, Anda memerlukan class yang berisi semua properti ini. Class data adalah jenis class yang hanya berisi properti. Class tersebut dapat menyediakan beberapa metode utilitas agar berfungsi dengan properti tersebut.

1. Buat paket baru di bagian **com.example.affirmations**. beri nama `model`.
2. Buat class baru di paket **com.example.affirmations.model**. beri nama `Affirmation` dan jadikan **Data class**.
3. Buat dua properti `val` di `class data Affirmation`.

```
import androidx.annotation.DrawableRes
import androidx.annotation.StringRes

data class Affirmation(
    @StringRes val stringResourceId: Int,
    @DrawableRes val imageResourceId: Int
)
```

4. Dalam paket **com.example.affirmations.data**, buka file **Datasource.kt** dan hapus tanda komentar.

```
import com.example.affirmations.R
import com.example.affirmations.model.Affirmation

/**
 * [Datasource] generates a list of [Affirmation]
 */
class Datasource() {
    fun loadAffirmations(): List<Affirmation> {
        return listOf<Affirmation>(
            Affirmation(R.string.affirmation1, R.drawable.image1),
            Affirmation(R.string.affirmation2, R.drawable.image2),
            Affirmation(R.string.affirmation3, R.drawable.image3),
            Affirmation(R.string.affirmation4, R.drawable.image4),
            Affirmation(R.string.affirmation5, R.drawable.image5),
            Affirmation(R.string.affirmation6, R.drawable.image6),
            Affirmation(R.string.affirmation7, R.drawable.image7),
            Affirmation(R.string.affirmation8, R.drawable.image8),
            Affirmation(R.string.affirmation9, R.drawable.image9),
            Affirmation(R.string.affirmation10, R.drawable.image10))
    }
}
```

Membuat kartu item daftar

Aplikasi ini dimaksudkan untuk menampilkan daftar afirmasi. Item akan terdiri dari composable `Card`, yang berisi `Image` dan composable `Text`. Di Compose, `Card` adalah platform yang menampilkan konten dan tindakan dalam satu penampung.

1. Buka file **MainActivity.kt**.
2. Buat metode baru di bawah metode `AffirmationsApp()`, yang disebut `AffirmationCard()`, dan anotasikan dengan anotasi `@Composable`.

```
import com.example.affirmations.model.Affirmation

@Composable
fun AffirmationsApp() {
}

@Composable
fun AffirmationCard() {

}
```

3. Di dalam metode `AffirmationCard`, panggil composable `Card`.

```
@Composable
fun AffirmationCard(affirmation: Affirmation, modifier: Modifier = Modifier) {
```

```
    Card(modifier = modifier) {
        Column {
            Image(
                painter = painterResource(affirmation.imageResourceId),
                contentDescription = stringResource(affirmation.stringResourceId),
                modifier = Modifier
                    .fillMaxWidth()
                    .height(194.dp),
                contentScale = ContentScale.Crop
            )
        }
    }
}
```

4. Di dalam `Column`, buat composable `Text` setelah composable `Image`.

```
import androidx.compose.material3.Text
import androidx.compose.foundation.layout.padding
import androidx.compose.ui.platform.LocalContext


Text(
    text = LocalContext.current.getString(affirmation.stringResourceId),
    modifier = Modifier.padding(16.dp),
    style = MaterialTheme.typography.headlineSmall
)
```

Pratinjau composable AffirmationCard

1. Buat metode pribadi bernama `AffirmationCardPreview()`. Anotasikan metode dengan `@Preview` dan `@Composable`.

```
@Preview
@Composable
private fun AffirmationCardPreview() {
    AffirmationCard(Affirmation(R.string.affirmation1, R.drawable.image1))
}
```

2. Buka tab **Split** dan Anda akan melihat pratinjau AffirmationCard. Jika perlu, klik **Build & Refresh** di panel **Design** untuk menampilkan pratinjau.

AffirmationCardPreview

Membuat daftar

Komponen item daftar adalah elemen penyusun daftar. Setelah item daftar dibuat, Anda dapat memanfaatkannya untuk membuat komponen daftar itu sendiri.

1. Buat fungsi yang disebut `AffirmationList()`, anotasikan dengan anotasi `@Composable`, dan deklarasikan `List` objek `Affirmation` sebagai parameter di deklarasi metode.

```
@Composable
fun AffirmationList(
    affirmationList: List<Affirmation>,
    modifier: Modifier = Modifier
) {

}
```

Di Jetpack Compose, daftar yang dapat di-scroll dapat dibuat menggunakan composable `LazyColumn`. Perbedaan antara `LazyColumn` dan `Column` adalah bahwa `Column` harus digunakan saat Anda memiliki sedikit item untuk ditampilkan, karena Compose memuat semuanya sekaligus. `Column` hanya dapat menyimpan composable dengan jumlah yang tetap atau telah ditentukan. `LazyColumn` dapat menambahkan konten sesuai keperluan, yang menjadikannya cocok untuk daftar panjang, terutama jika panjang daftar tidak diketahui. `LazyColumn` juga menyediakan scroll secara default, tanpa kode tambahan.

2. Deklarasikan composable `LazyColumn` di dalam fungsi `AffirmationList()`. Teruskan objek `modifier` sebagai argumen ke `LazyColumn`.

```
import androidx.compose.foundation.lazy.LazyColumn
```

```
@Composable
fun AffirmationList(
    affirmationList: List<Affirmation>,
    modifier: Modifier = Modifier
) {
    LazyColumn(modifier = modifier) {

    }
}
```

3. Dalam isi lambda `LazyColumn`, panggil metode `items()` dan teruskan `affirmationList`. Metode `items()` adalah cara Anda menambahkan item ke `LazyColumn`. Metode ini agak unik untuk composable ini, dan bukan praktik umum untuk sebagian besar composable.

```
import androidx.compose.foundation.lazy.items

@Composable
fun AffirmationList(
    affirmationList: List<Affirmation>,
    modifier: Modifier = Modifier
) {
    LazyColumn(modifier = modifier) {
        items(affirmationList) {

        }
    }
}
```

4. Panggilan ke metode `items()` memerlukan fungsi lambda. Dalam fungsi tersebut, tetapkan parameter `affirmation` yang mewakili satu item afirmasi dari `affirmationList`.

```
LazyColumn(modifier = modifier) {
    items(affirmationList) { affirmation ->

    }
}
```

5. Untuk setiap afirmasi dalam daftar, panggil composable `AffirmationCard()`. Teruskan `affirmation` dan objek `Modifier` dengan atribut `padding` yang disetel ke `8.dp`.

```
LazyColumn(modifier = modifier) {
    items(affirmationList) { affirmation ->
        AffirmationCard(
            affirmation = affirmation,
            modifier = Modifier.padding(8.dp)
        )
    }
}
```

Menampilkan daftar

1. Pada composable `AffirmationsApp`, ambil rute tata letak saat ini lalu simpan dalam variabel. Rute ini akan digunakan untuk mengonfigurasi padding nanti.

```
import androidx.compose.ui.platform.LocalLayoutDirection

@Composable
fun AffirmationsApp() {
    val layoutDirection = LocalLayoutDirection.current
}
```

2. Sekarang, buat composable `Surface`. Composable ini akan menetapkan padding untuk composable `AffirmationsList`.

```
@Composable
fun AffirmationsApp() {
    val layoutDirection = LocalLayoutDirection.current
    Surface(
        modifier = Modifier
            .fillMaxSize()
            .statusBarsPadding()
            .padding(
                start = WindowInsets.safeDrawing.asPaddingValues()
                    .calculateStartPadding(layoutDirection),
                end = WindowInsets.safeDrawing.asPaddingValues()
                    .calculateEndPadding(layoutDirection),
            ),
    ) {
    }
}
```

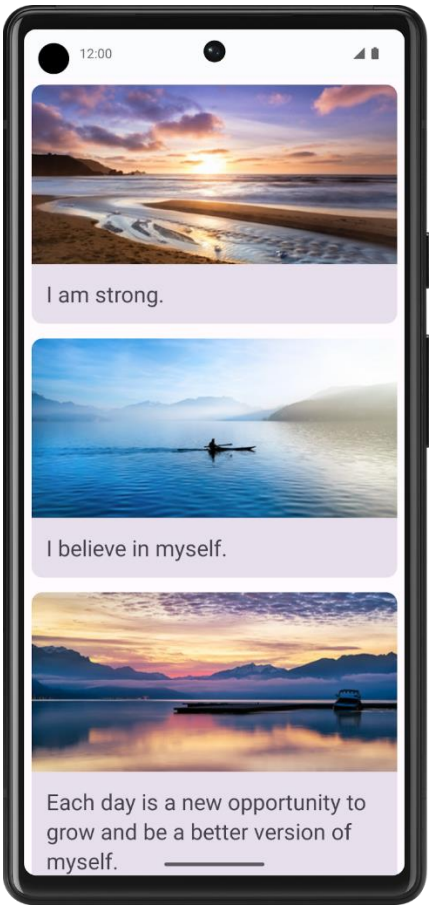
3. Di lambda untuk composable `Surface`, panggil composable `AffirmationList`, lalu teruskan `DataSource().loadAffirmations()` ke parameter `affirmationList`.

```
import com.example.affirmations.data.Datasource


@Composable
fun AffirmationsApp() {
    val layoutDirection = LocalLayoutDirection.current
    Surface(
        modifier = Modifier
            .fillMaxSize()
            .statusBarsPadding()
            .padding(
                start = WindowInsets.safeDrawing.asPaddingValues()
                    .calculateStartPadding(layoutDirection),
                end = WindowInsets.safeDrawing.asPaddingValues()
                    .calculateEndPadding(layoutDirection),
```

```
            ),
    ) {
        AffirmationList(
            affirmationList = Datasource().loadAffirmations(),
        )
    }
}
```

Jalankan aplikasi **Affirmations** di perangkat atau emulator dan lihat produk yang sudah selesai.



Sumber:

https://developer.android.com/codelabs/basic-android-kotlin-compose-training-add-scrollable-list?hl=id&continue=https%3A%2F%2Fdeveloper.android.com%2Fcourses%2Fpathways%2Fandroid-basics-compose-unit-3-pathway-2%3Fhl%3Did%23codelab-https%3A%2F%2Fdeveloper.android.com%2Fcodelabs%2Fbasic-android-kotlin-compose-training-add-scrollable-list#0