Kotlin

MOBILE PROGRAMMING

OOP WITH KOTLIN & MORE COMPOSABLE UI

ROBBY TAN

OOP Review

OOP Pillar

- Encapsulation
- Abstraction
- Inheritance
- Polymorphism

Class

- Blueprint sebuah objek.
- Memiliki **atribut** (*property*) dan ***method***.
- Dalam Kotlin, sebuah *class* minimal memiliki **sebuah *constructor***.
- Atribut dalam Kotlin secara **otomatis** sudah diberikan **set** dan **get** *method*.
- Terdapat 2 jenis *constructor*:
  - ***Default*** *constructor*
  - ***Secondary*** *constructor*

```
1.  class Person {
2.      var firstName: String? = null
3.      var lastName: String? = null
4.  }
```

```
1.  fun main() {
2.      val person = Person()
3.      person.firstName = "Robby"
4.      person.lastName = "Tan"
5.
6.      println("First Name: ${person.firstName}")
7.      println("Last Name: ${person.lastName}")
8.  }
```

Constructor in Class Declaration

- Kotlin menyederhanakan struktur pembuatan sebuah *class* di mana *constructor* langsung dituliskan pada deklarasi *class*.
- Penulisan seperti ini menghilangkan *default constructor*.

```
1.  class Person(var firstName: String, var lastName: String) {
2.  }
```

```
1.  fun main() {
2.      val person = Person("Robby", "Tan")
3.      println("First Name: ${person.firstName}")
4.      println("Last Name: ${person.lastName}")
5.  }
```

Secondary Constructor

- Didefinisikan dengan keyword *constructor* (baris ke-4).
- Pembuatan objek dapat menggunakan constructor di baris ke-1 atau baris ke-4.

```
1.  class Person(var firstName: String, var lastName: String) {
2.      var age: Int? = null
3.
4.      constructor(firstName: String, lastName: String, age: Int) : this(firstName, lastName) {
5.          this.age = age
6.      }
7.  }
```

```
1.  fun main() {
2.      val person = Person("Robby", "Tan")
3.      println("First Name: ${person.firstName}")
4.      println("Last Name: ${person.lastName}")
5.
6.      val personWithAge = Person("John", "Doe", 20)
7.      println("First Name: ${personWithAge.firstName}")
8.      println("Last Name: ${personWithAge.lastName}")
9.      println("Age: ${personWithAge.age}")
10. }
```

Inheritance (1/2)

- Secara default, seluruh *class* pada Kotlin adalah **final** (tidak dapat dibuat *subclass*).
- Diberikan *keyword* **open** pada bagian awal deklarasi *class* sehingga dapat memiliki *subclass*.

```
1.  open class Person(var firstName: String, var lastName: String) {
2.      var age: Int? = null
3.
4.      constructor(firstName: String, lastName: String, age: Int) : this(firstName, lastName) {
5.          this.age = age
6.      }
7.
8.      open fun fullDetail(): String {
9.          return "Name: $firstName $lastName, Age: ${age ?: "Not specified"}"
10.     }
11. }
```

Inheritance (2/2)

```
class Student: Person {
    var studentId: String? = null

    constructor(studentId: String, firstName: String, lastName: String) : super(firstName, lastName) {
        this.studentId = studentId
    }

    constructor(studentId: String, firstName: String, lastName: String, age: Int) : super(firstName, lastName, age) {
        this.studentId = studentId
    }

    override fun fullDetail(): String {
        return "Student ID: $studentId, Name: $firstName $lastName, Age: ${age ?: "Not specified"}"
    }
}
```

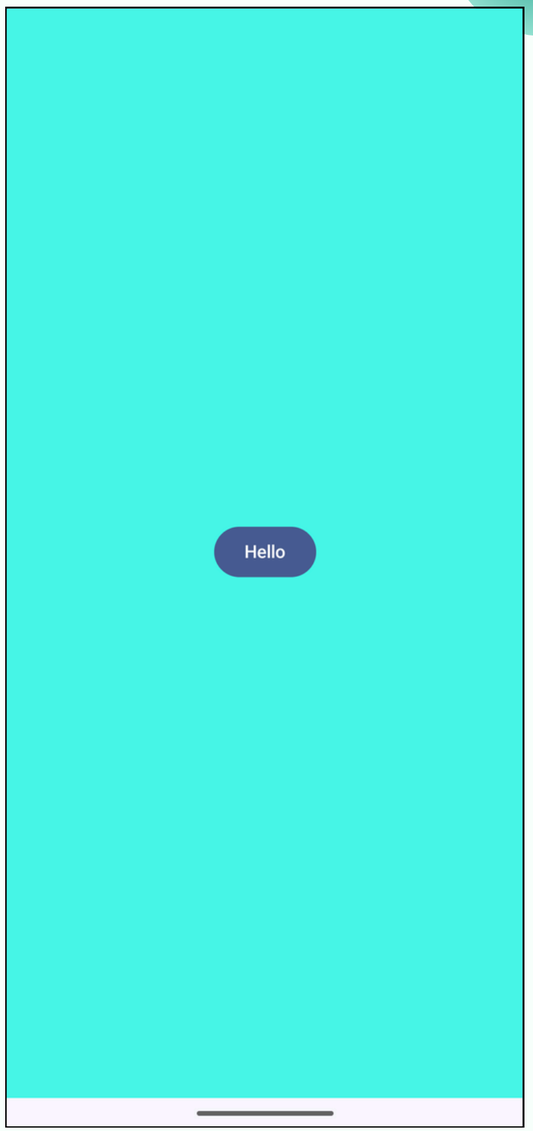
```
fun main() {
    val student = Student("001", "Robby", "Tan")
    println(student.fullDetail())

    val studentWithAge = Student("002","John", "Doe", 20)
    println(studentWithAge.fullDetail())
}
```

Button, AppBar, Toast

Button on Android

- Use Button Composable (androidx.compose.material3.Button).
- Untuk menampilkan tulisan dalam tombol, diperlukan Text Composable.

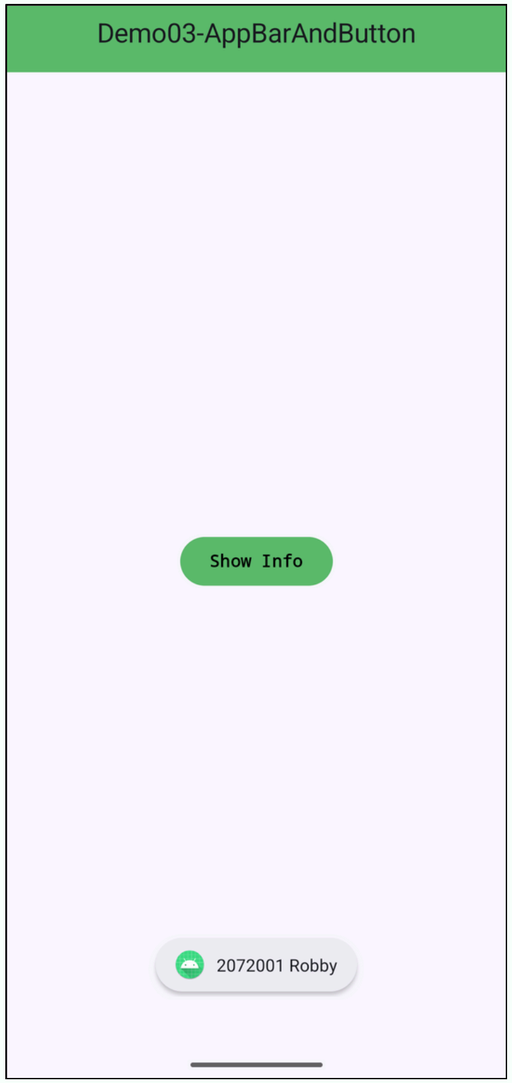
```
1.  @Composable
2.  private fun ButtonExample(modifier: Modifier = Modifier) {
3.      val context = LocalContext.current
4.      Box(
5.          modifier = modifier
6.              .fillMaxSize()
7.              .background(Color(0xFF46F8E9))
8.      ) {
9.          Column(
10.             modifier = modifier.align(Alignment.Center),
11.             horizontalAlignment = Alignment.CenterHorizontally
12.         ) {
13.             Button(
14.                 onClick = { Toast.makeText(context, "Hello", Toast.LENGTH_SHORT).show() }
15.             ) {
16.                 Text(text = "Hello")
17.             }
18.         }
19.     }
20. }
```



AppBar and OOP

- Terdapat beberapa Composable Class untuk TopBar
  - TopAppBar
  - CenterAlignedTopAppBar
- Penggunaan TopBar diletakkan pada Scaffold untuk parameter topbar

```
1.  Scaffold(
2.    topBar = { TODO("Create Your TopBar") },
3.  ) { innerPadding ->
4.    AppBarDemo(modifier = Modifier.padding(innerPadding))
5.  }
```



Person and Student Class

- *Class* Person dibuat **open** sehingga dapat dibuat *subclass*.
- *Method* showInfo() pada *class* Person juga diberikan *keyword* **open**.
- *Class* Student adalah *subclass* dari Person.
- Pembuatan (*instantiate*) *class* Student pada MainActivity.kt pada sebuah *method*.

```kotlin
open class Person(var firstName: String, var lastName: String? = null) {
  open fun showInfo(): String {
    return lastName?.let { lName -> "$firstName $lName" } ?: firstName
  }
}
```

```kotlin
class Student(val studentId: String, firstName: String, lastName: String? = null) : Person(firstName, lastName) {
  override fun showInfo(): String {
    return studentId + (lastName?.let { it -> " $firstName $it" } ?: " $firstName")
  }
}
```

```kotlin
private fun showStudentInfo(context: Context) {
  val student = Student("2072001", "Robby")
  Toast.makeText(context, student.showInfo(), Toast.LENGTH_LONG).show()
}
```

Update MainActivity.kt

- Pembuatan Composable Top Bar
  - dapat ditulis langsung pada Scaffold
  - dapat ditulis terpisah pada sebuah Composable function

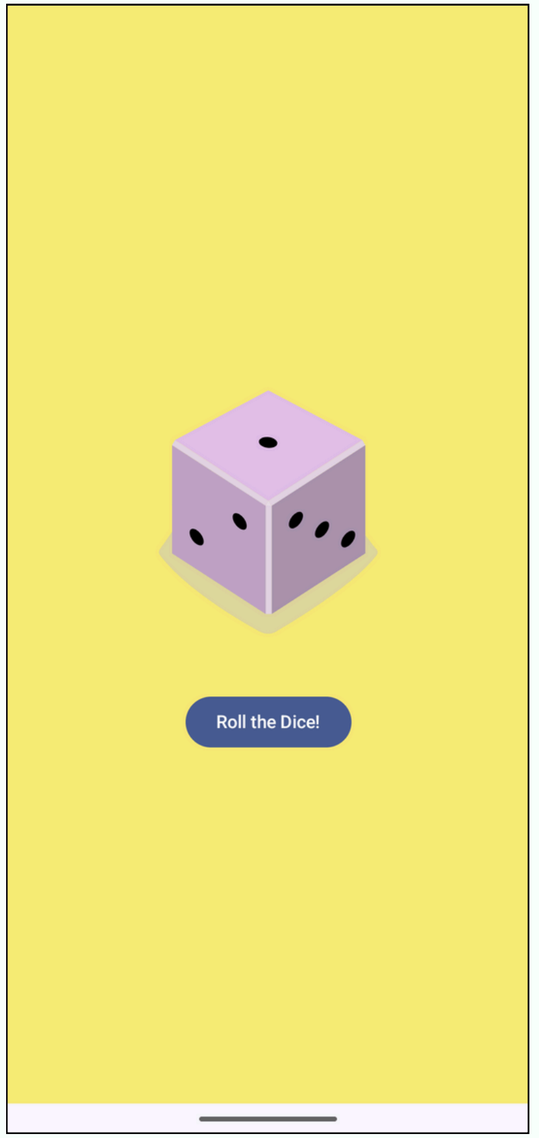
```
@OptIn(ExperimentalMaterial3Api::class)
@Composable
private fun TopBar() {
  CenterAlignedTopAppBar(
    title = {
      Text(
        text = stringResource(R.string.app_name)
      )
    },
    colors = TopAppBarDefaults.topAppBarColors(
      containerColor = Color(0xFF5DBB6D)
    )
  )
}
```

```
@Composable
private fun AppBarDemo(modifier: Modifier = Modifier) {
  val context = LocalContext.current
  Column (
    modifier = modifier.fillMaxSize(),
    horizontalAlignment = Alignment.CenterHorizontally,
    verticalArrangement = Arrangement.Center
  ) {
    Button(
      onClick = { showStudentInfo(context) },
      colors = ButtonDefaults.buttonColors(
        containerColor = Color(0xFF5DBB6D)
      ),
    ) {
      Text(
        text = "Show Info",
        color = Color.Black,
        fontWeight = FontWeight.Bold,
        fontFamily = FontFamily.Monospace
      )
    }
  }
}
```

Exercise

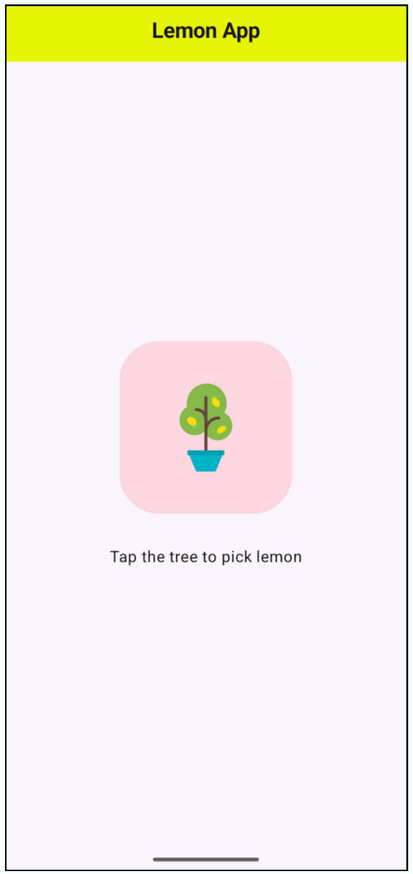
Exercise: Dice Roll

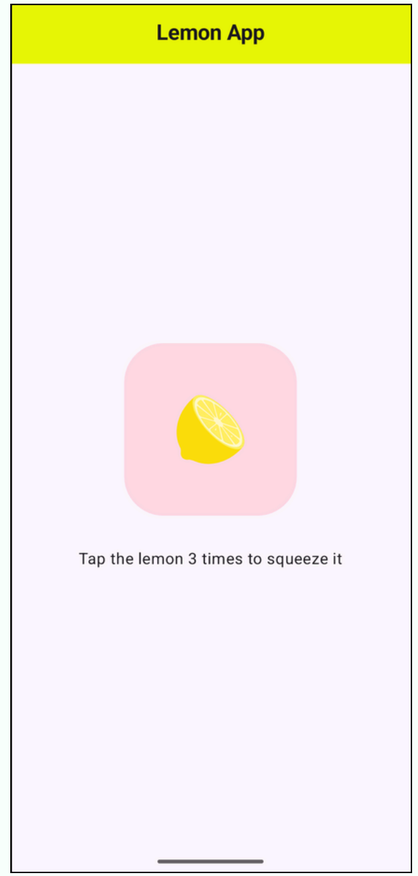
- Buatlah sebuah aplikasi Android berbasis Kotlin di mana yang memiliki beberapa Composable Material:
  - Box
  - Column
  - Image
  - Text
- Tampilan aplikasi seperti gambar di samping.



Exercise: Lemon Click

- Buatlah sebuah aplikasi Android berbasis Kotlin seperti gambar di samping.
- Keterangan:
  - Terdapat 4 tahap dari aplikasi
  - Tahapan pertama muncul gambar pohon lemon yang jika ditekan akan berganti ke gambar di sampingnya
  - Tahap kedua, Anda harus melakukan klik/ tap gambar lemon sehingga berganti ke gambar berikutnya (Gambar gelas)
  - Tahap ketiga, tap gambar (1x) gelas berisi lemon sehingga kosong
  - Tahap keempat, tap gambar (1x) sehingga kembali ke awal.





Exercise: Lemon Click

- Ketentuan:
  - Pisahkan kode program untuk TopAppBar dalam sebuah fungsi Composable tersendiri
  - Seluruh teks yang dimunculkan pada gambar wajib disimpan dalam string resource (strings.xml).

Thank You

ROBBY.TAN@IT.MARANATHA.EDU