**Lab Exercise 6- Shapes in Jetpack Compose**

**Objective:**

Create a simple Jetpack Compose app to demonstrate how to customize and use shapes (rounded corners, cut corners, etc.) in your UI components like buttons and cards.

**Steps:**

**Step 1: Set up a New Jetpack Compose Project**

1. Open **Android Studio**.
2. Create a new project:
   * Select **New Project** -> **Empty Compose Activity**.
   * Set the project name to ShapeDemoCompose and finish the setup.
3. Make sure your project includes the necessary Jetpack Compose dependencies:

dependencies {

implementation "androidx.activity:activity-compose:1.7.0"

implementation "androidx.compose.ui:ui:1.4.0"

implementation "androidx.compose.material3:material3:1.1.0"

implementation "androidx.compose.ui:ui-tooling-preview:1.4.0"

debugImplementation "androidx.compose.ui:ui-tooling:1.4.0"

}

**Step 2: Define Custom Shapes**

1. Navigate to the ui/theme folder in your project and open the **Theme.kt** file.
2. Inside Theme.kt, define your custom shapes:

import androidx.compose.foundation.shape.RoundedCornerShape

import androidx.compose.material3.Shapes

import androidx.compose.runtime.Composable

import androidx.compose.ui.unit.dp

// Define custom shapes

val CustomShapes = Shapes(

small = RoundedCornerShape(8.dp),

medium = RoundedCornerShape(16.dp),

large = RoundedCornerShape(24.dp)

)

@Composable

fun MyAppTheme(content: @Composable () -> Unit) {

MaterialTheme(

colorScheme = lightColorScheme(),

typography = Typography,

shapes = CustomShapes, // Apply custom shapes here

content = content

)

}

**Step 3: Use Custom Shapes in Your UI**

1. Open the MainActivity.kt file and modify the content to use the custom shapes:

import android.os.Bundle

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.compose.foundation.background

import androidx.compose.foundation.layout.\*

import androidx.compose.material3.\*

import androidx.compose.runtime.Composable

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.tooling.preview.Preview

import androidx.compose.ui.unit.dp

import com.example.shapedemocomepose.ui.theme.MyAppTheme

class MainActivity : ComponentActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContent {

MyAppTheme {

ShapeDemoScreen()

}

}

}

}

@Composable

fun ShapeDemoScreen() {

Column(

modifier = Modifier

.fillMaxSize()

.padding(16.dp),

verticalArrangement = Arrangement.spacedBy(16.dp),

horizontalAlignment = Alignment.CenterHorizontally

) {

// Button with custom shape

Button(

onClick = { /\* Do something \*/ },

shape = MaterialTheme.shapes.medium, // Use medium shape

modifier = Modifier.width(200.dp)

) {

Text(text = "Rounded Button")

}

// Card with custom shape

Card(

modifier = Modifier

.fillMaxWidth()

.height(150.dp)

.background(MaterialTheme.colorScheme.primary),

shape = MaterialTheme.shapes.large // Use large shape

) {

Box(contentAlignment = Alignment.Center) {

Text(text = "Rounded Card", color = Color.White)

}

}

}

}

@Preview(showBackground = true)

@Composable

fun ShapeDemoPreview() {

MyAppTheme {

ShapeDemoScreen()

}

}

**Explanation:**

1. **Custom Shapes**: You defined custom shapes in Theme.kt using the Shapes class and assigned rounded corners of different sizes (8dp, 16dp, 24dp) for small, medium, and large components.
2. **Button with Custom Shape**: The Button composable uses MaterialTheme.shapes.medium, which applies a 16dp rounded corner.
3. **Card with Custom Shape**: The Card composable uses MaterialTheme.shapes.large, which applies a 24dp rounded corner, creating a more prominently rounded shape for larger components.
4. **MaterialTheme.shapes**: You access the shapes from the theme (defined in Theme.kt) using MaterialTheme.shapes for a consistent design throughout the app.

**Step 4: Run the Application**

1. Select an emulator or a physical device.
2. Run the project.
3. The app should display a screen with:
   * A button with 16dp rounded corners.
   * A card with 24dp rounded corners.

**Task: Customize the Shapes Further**

* **Task 1**: Modify the shapes to use CutCornerShape instead of RoundedCornerShape.

Example:

val CustomShapes = Shapes(

small = CutCornerShape(8.dp),

medium = CutCornerShape(16.dp),

large = CutCornerShape(24.dp)

)

* **Task 2**: Add more UI elements, such as TextField or Surface, and apply different shapes to them.

**Conclusion:**

This lab exercise demonstrates how to define and apply custom shapes in Jetpack Compose using MaterialTheme.shapes. You learned how to use the shape system to create visually appealing UIs by rounding corners or using cut corners.