Week 2 First Android App

This lab exercise is fully based on the official Android tutorial, but we extend it a bit with providing more materials about some functionality.

We highly recommend that you can finish all the following tasks, which are important to your coursework.

Task 1: “Hello Android”

We learn how to create the first android app in the Android Studio and run it on the virtual emulator and physical devices. From this example, you should understand the basic program structure about Kotlin and Jetpack compose and know some basic operations like changing background colour or adding paddings.

Please visit this link <https://developer.android.com/courses/pathways/android-basics-compose-unit-1-pathway-2> to create the first app. You can get help from the lab tutors or the lecture videos if some descriptions are not clear.

Based on the “Hello Android” example, you need to finish the following small tasks.

1. Change the background colour of the text to cyan.

***Surface:***

@[**Composable**](https://developer.android.com/reference/kotlin/androidx/compose/runtime/Composable)  
fun [**Surface**](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0))(  
    modifier: [**Modifier**](https://developer.android.com/reference/kotlin/androidx/compose/ui/Modifier) = Modifier,  
    shape: [**Shape**](https://developer.android.com/reference/kotlin/androidx/compose/ui/graphics/Shape) = RectangleShape,  
    color: **[Color](https://developer.android.com/reference/kotlin/androidx/compose/ui/graphics/Color)** = MaterialTheme.colors.surface,  
    contentColor: **[Color](https://developer.android.com/reference/kotlin/androidx/compose/ui/graphics/Color)** = contentColorFor(color),  
    border: **[BorderStroke](https://developer.android.com/reference/kotlin/androidx/compose/foundation/BorderStroke)**? = null,  
    elevation: **[Dp](https://developer.android.com/reference/kotlin/androidx/compose/ui/unit/Dp)** = 0.dp,  
    content: @[**Composable**](https://developer.android.com/reference/kotlin/androidx/compose/runtime/Composable) () -> [**Unit**](https://kotlinlang.org/api/latest/jvm/stdlib/kotlin/-unit/index.html)  
): [**Unit**](https://kotlinlang.org/api/latest/jvm/stdlib/kotlin/-unit/index.html)

The Surface is responsible for:

1. Clipping: Surface clips its children to the shape specified by [shape](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0))
2. Elevation: Surface draws a shadow to represent depth, where [elevation](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0)) represents the depth of this surface. If the passed [shape](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0)) is concave the shadow will not be drawn on Android versions less than 10.
3. Borders: If [shape](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0)) has a border, then it will also be drawn.
4. Background: Surface fills the shape specified by [shape](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0)) with the [color](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary" \l "Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0)). If [color](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary" \l "Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0)) is [Colors.surface](https://developer.android.com/reference/kotlin/androidx/compose/material/Colors" \l "surface()), the [ElevationOverlay](https://developer.android.com/reference/kotlin/androidx/compose/material/ElevationOverlay) from [LocalElevationOverlay](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary" \l "LocalElevationOverlay()) will be used to apply an overlay - by default this will only occur in dark theme. The color of the overlay depends on the [elevation](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0)) of this Surface, and the [LocalAbsoluteElevation](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary" \l "LocalAbsoluteElevation()) set by any parent surfaces. This ensures that a Surface never appears to have a lower elevation overlay than its ancestors, by summing the elevation of all previous Surfaces.
5. Content color: Surface uses [contentColor](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary" \l "Surface(androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Shape,androidx.compose.ui.graphics.Color,androidx.compose.ui.graphics.Color,androidx.compose.foundation.BorderStroke,androidx.compose.ui.unit.Dp,kotlin.Function0)) to specify a preferred color for the content of this surface - this is used by the [Text](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Text(kotlin.String,androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Color,androidx.compose.ui.unit.TextUnit,androidx.compose.ui.text.font.FontStyle,androidx.compose.ui.text.font.FontWeight,androidx.compose.ui.text.font.FontFamily,androidx.compose.ui.unit.TextUnit,androidx.compose.ui.text.style.TextDecoration,androidx.compose.ui.text.style.TextAlign,androidx.compose.ui.unit.TextUnit,androidx.compose.ui.text.style.TextOverflow,kotlin.Boolean,kotlin.Int,kotlin.Int,kotlin.Function1,androidx.compose.ui.text.TextStyle)) and [Icon](https://developer.android.com/reference/kotlin/androidx/compose/material/package-summary#Icon(androidx.compose.ui.graphics.vector.ImageVector,kotlin.String,androidx.compose.ui.Modifier,androidx.compose.ui.graphics.Color)) components as a default color.
6. Blocking touch propagation behind the surface.

Note that this function may change with the different versions of Android systems.

***Modifier:***

Modifiers allow you to decorate or augment a composable. Modifiers let you do these sorts of things:

1. Change the composable's size, layout, behaviour, and appearance
2. Add information, like accessibility labels
3. Process user input
4. Add high-level interactions, like making an element clickable, scrollable, draggable, or zoomable

You can visit this link <https://developer.android.com/jetpack/compose/modifiers> to understand more about the usage of modifier.

Task 2: Build a basic layout

We learn how to build a simple app with text composables with basic layout UI and images. Please visit this link <https://developer.android.com/courses/pathways/android-basics-compose-unit-1-pathway-3> and finish the tutorial.

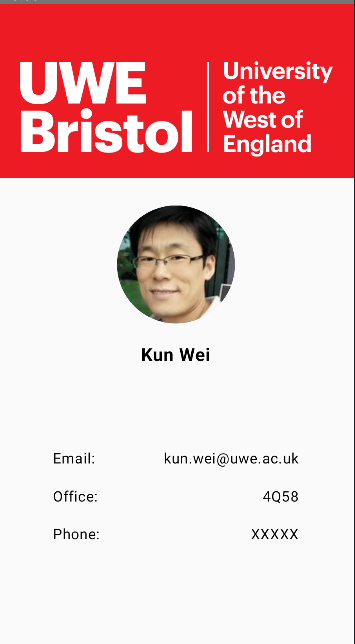
This is a long tutorial, and you need to finish a couple of apps. The birthday card app is the main app where you can learn a lot about the basic layout. Then, as the practices, from Activity 5, you can exercise THREE apps, compose article, task manager and compose quadrant. The solution is also provided within the tutorial. Note that the solution is variable, you can work out your own version.

It is unlikely that you can remember or be familiar with the layout immediately after the tutorial. Therefore, Google or ChatGPT might be the handy tools to use whenever you are looking for a function to change the layout.

Task 3: UWE contacts app

To test your study, we give a challenge that you build an UWE contacts app. This is a simple app, but it will be augmented with more views or activities later.

You need to build one view that includes an UWE logo, a contact person photo, name, email, an office location, and a phone number. The layout can be referred to the following screenshot. It is unnecessary to be exactly same, but the basic layout should be similar.



The UWE logo and the module leader’s photo can be downloaded from BB, and of course, you can use other persons’ photos.