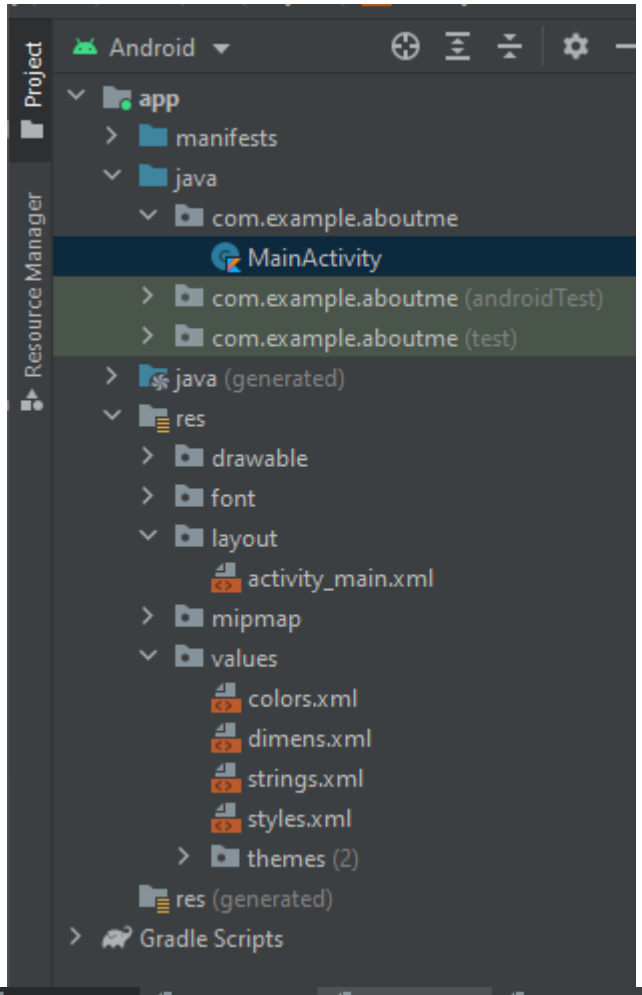
UNIVERSITI MALAYSIA TERENGGANU

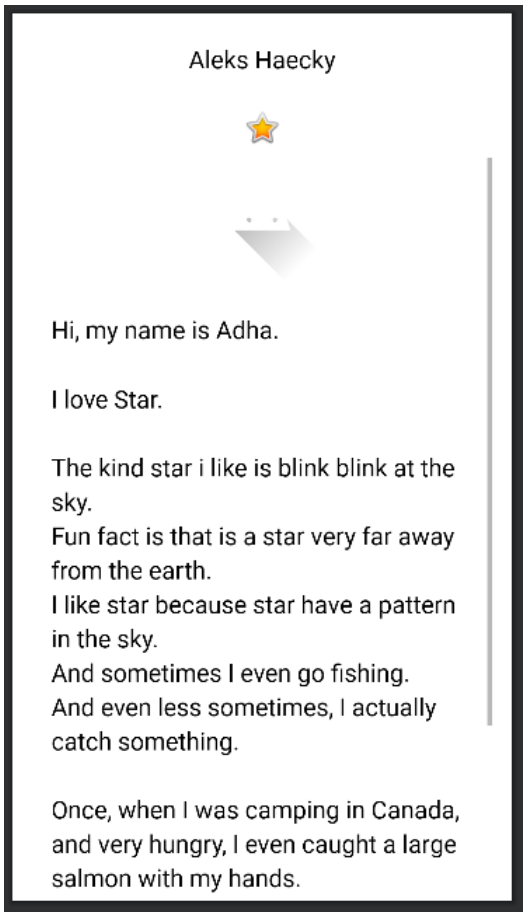
Lab 4 - https://github.com/khairuladha/Lab_4_CSM3505.git

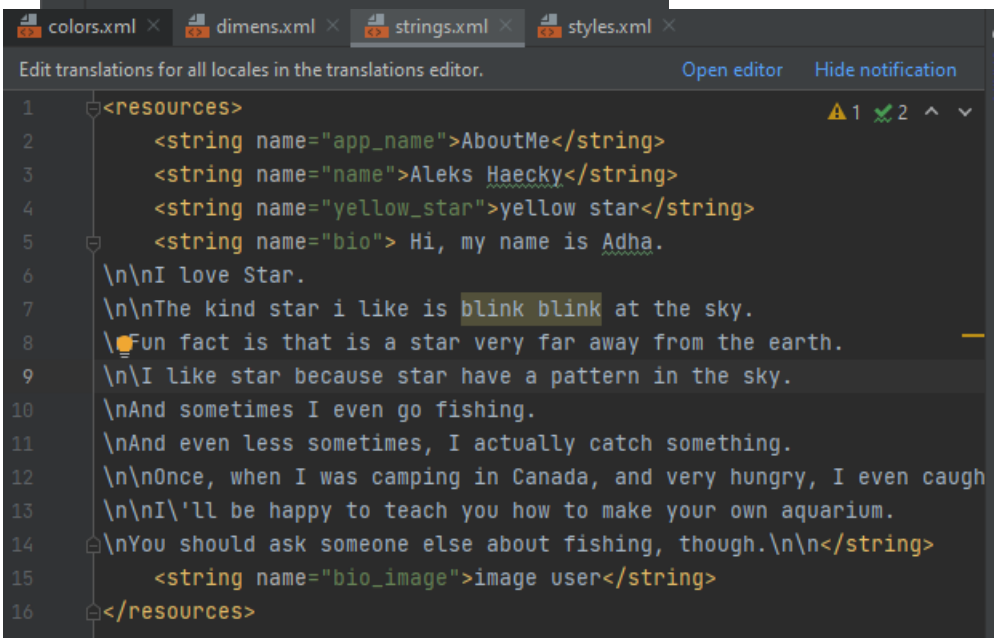
COURSE	:	**PENGATURCARAAN MUDAH ALIH NATIF**
COURSE CODE	:	**CSM3505**

NAME	:	**KHAIRULADHA BIN MISNAN**
MATRIC NO.	:	**S56878**
GROUP	:	**k1**
LECTURER	:	**DR. RABIEI B MAMAT**

Task 1: Android Kotlin Fundamentals: LinearLayout using the Layout Editor







colors.xml | dimens.xml | strings.xml | styles.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="text_size">20sp</dimen>
    <dimen name="small_padding">8dp</dimen>
    <dimen name="layout_margin">16dp</dimen>
    <dimen name="yellow_star">16dp</dimen>
    <dimen name="layout_padding">16dp</dimen>
</resources>
```

colors.xml | dimens.xml | strings.xml | styles.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

colors.xml | dimens.xml | strings.xml | styles.xml

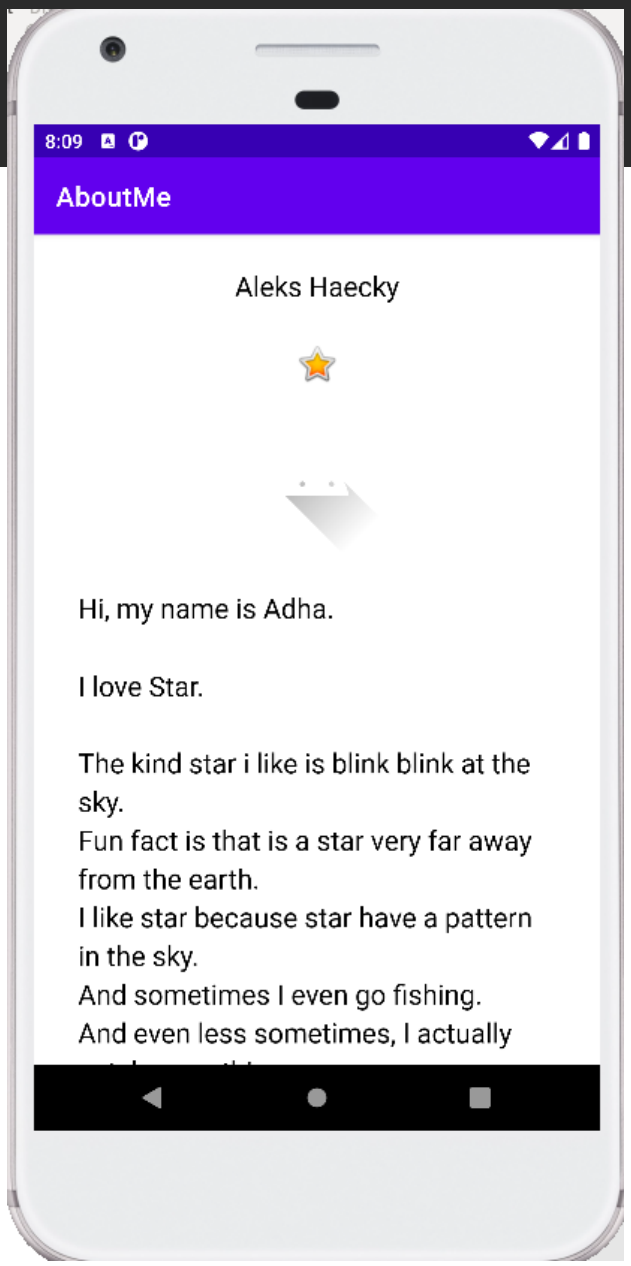
```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <style name="NameStyle">
        <item name="android:layout_marginTop">@dimen/layout_margin</i
        <item name="android:fontFamily">@font/roboto</item>
        <item name="android:paddingTop">@dimen/small_padding</item>
        <item name="android:textColor">@android:color/black</item>
        <item name="android:textSize">@dimen/text_size</item>
    </style>
</resources>
```

activity_main.xml

Code Split Design

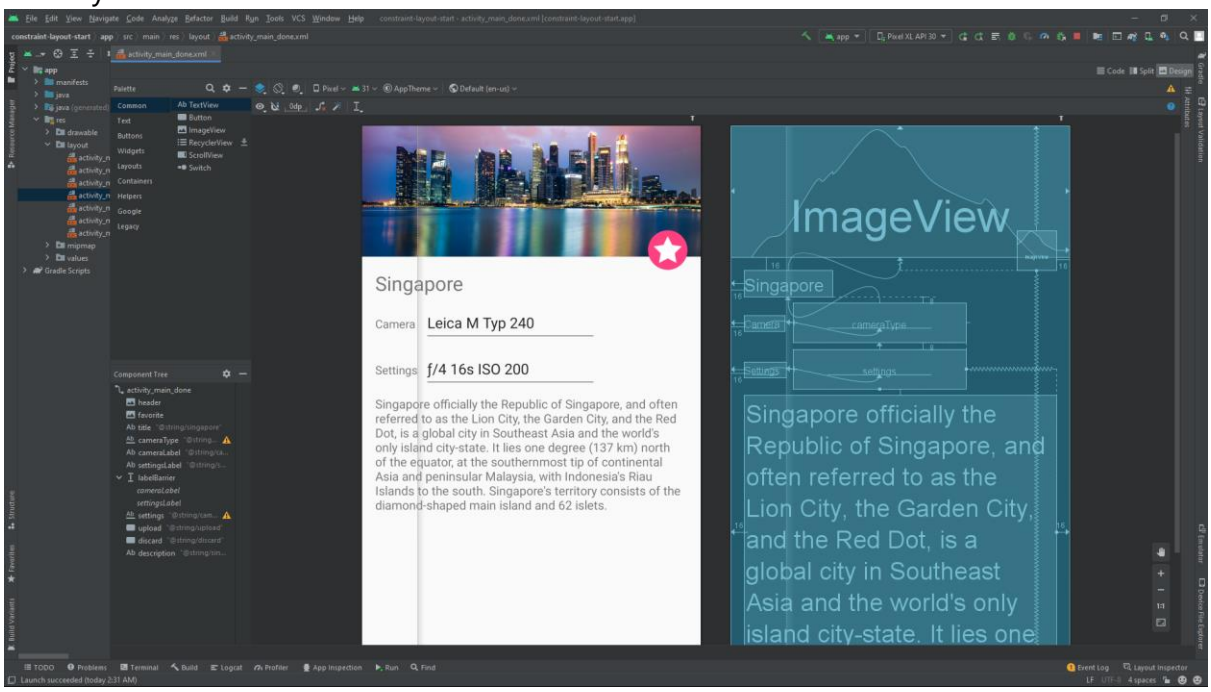
```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingStart="16dp"
    android:paddingEnd="16dp"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/name_text"
        style="@style/NameStyle"
        android:layout_width="match_parent"
        android:layout_height="46dp"
        android:text="Aleks Haecky"
        android:textAlignment="center" />

    <ImageView
        android:id="@+id/star_image"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:contentDescription="yellow star"
        app:srcCompat="@android:drawable/btn_star_big_on"
        tools:ignore="ImageContrastCheck" />

    <ScrollView
        android:id="@+id/bio_scroll"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">


        <LinearLayout
            android:layout_width="match_parent"
            android:orientation="vertical"
            android:paddingStart="16dp"
            android:paddingEnd="16dp"
            android:layout_height="wrap_content">

```

```
42          <ImageView
43              android:id="@+id/bio_image"
44              android:layout_width="match_parent"
45              android:layout_height="wrap_content"
46              android:layout_marginTop="16dp"
47              android:contentDescription="image user"
48              app:srcCompat="@drawable/ic_launcher_foreground"
49              tools:ignore="ImageContrastCheck" />
50
51          <TextView
52              android:id="@+id/bio_text"
53              style="@style/NameStyle"
54              android:layout_width="match_parent"
55              android:layout_height="wrap_content"
56              android:lineSpacingMultiplier="1.2"
57              android:text="Hi, my name is Adha.   I love Star.   The kin
58
59      </LinearLayout>
60  </ScrollView>
61
62 </LinearLayout>
```

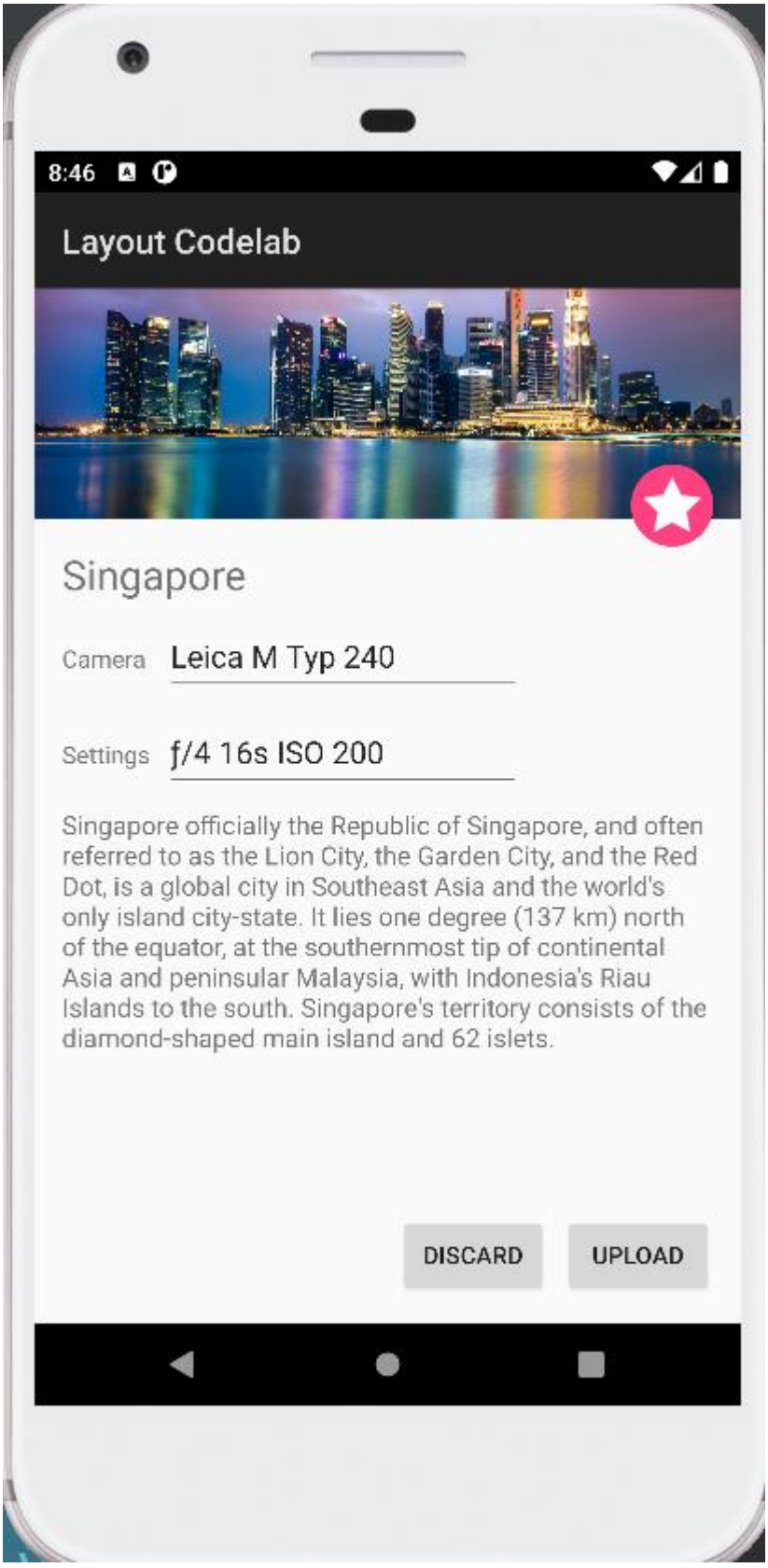


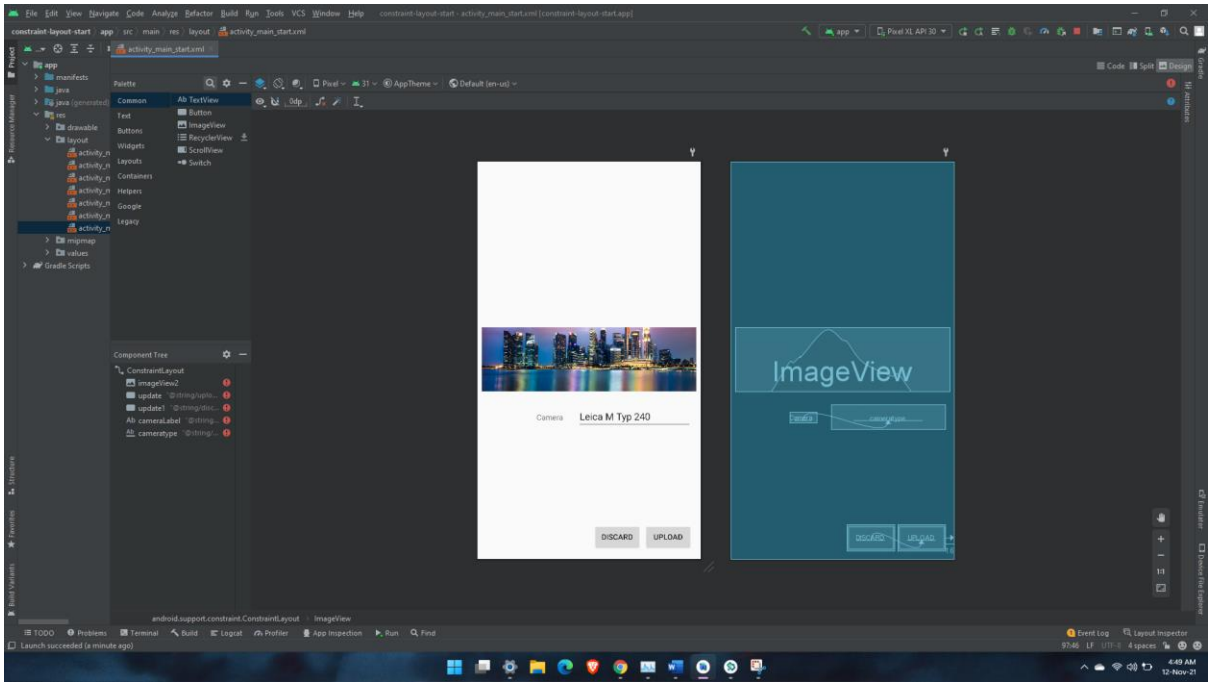
Task 2: Use ConstraintLayout to design your Android views

1. The Layout Editor

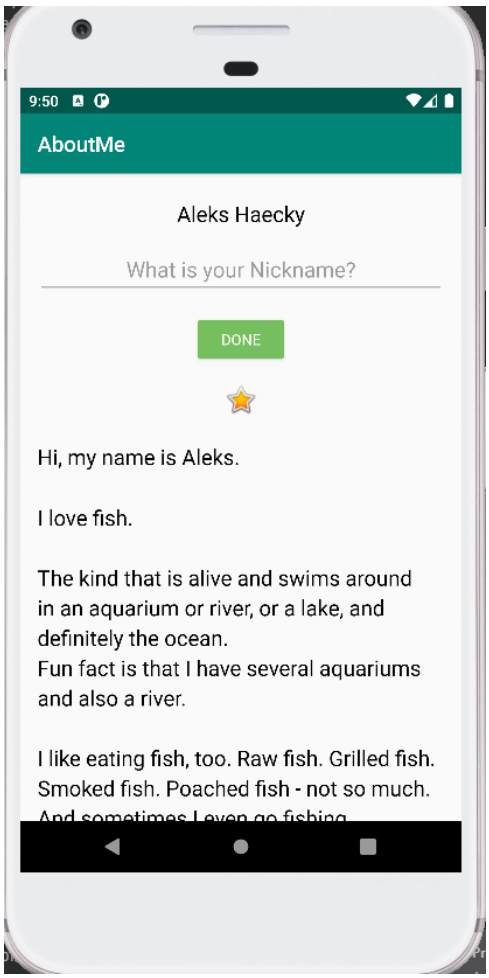


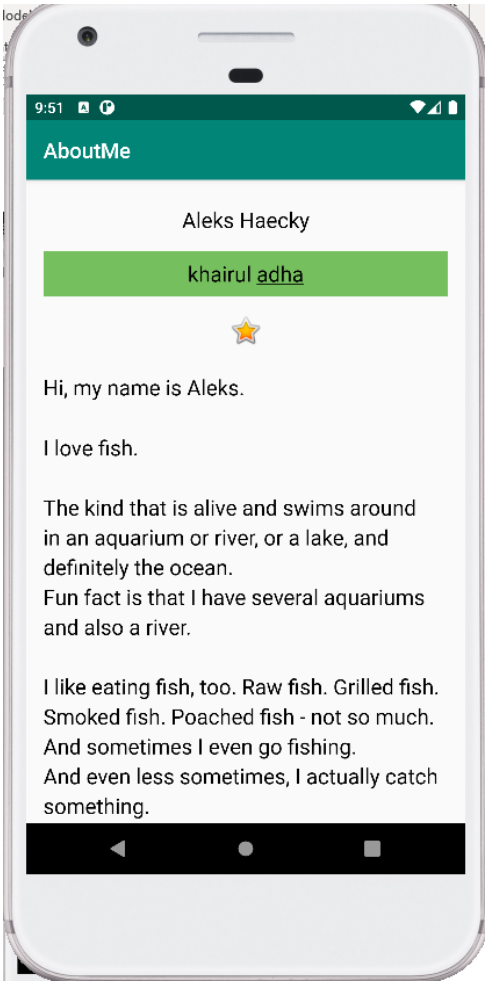
```
<EditText
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:minHeight="48dp"
    android:inputType="textPersonName"
    android:text="@string/camera_value"
    android:ems="10"
    app:layout_editor_absoluteX="73dp"
    app:layout_editor_absoluteY="176dp"
    android:id="@+id/cameraType"
    app:layout_constraintLeft_creator="1"
    app:layout_constraintTop_creator="1"
    app:layout_constraintRight_creator="1"
    app:layout_constraintLeft_toLeftOf="@+id/settings"
    tools:layout_constraintLeft_creator="1"
    app:layout_constraintTop_toBottomOf="@+id/title"
    android:layout_marginTop="8dp"
    tools:layout_constraintTop_creator="0"
    app:layout_constraintRight_toRightOf="@+id/settings"
    tools:layout_constraintRight_creator="1"
    app:layout_constraintStart_toEndOf="@+id/labelBarrier"
    android:layout_marginStart="8dp" />
```


8:46
Layout Codelab
Singapore
Camera
Leica M Typ 240
Settings
ƒ/4 16s ISO 200
Singapore officially the Republic of Singapore, and often referred to as the Lion City, the Garden City, and the Red Dot, is a global city in Southeast Asia and the world's only island city-state. It lies one degree (137 km) north of the equator, at the southernmost tip of continental Asia and peninsular Malaysia, with Indonesia's Riau Islands to the south. Singapore's territory consists of the diamond-shaped main island and 62 islets.
DISCARD
UPLOAD

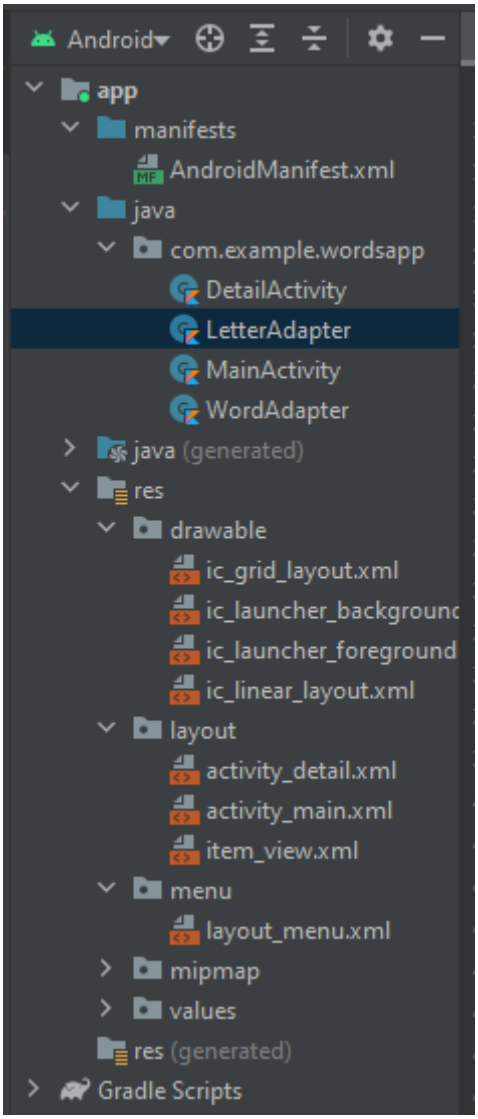


Task 3: Android Kotlin Fundamentals: Add user interactivityz



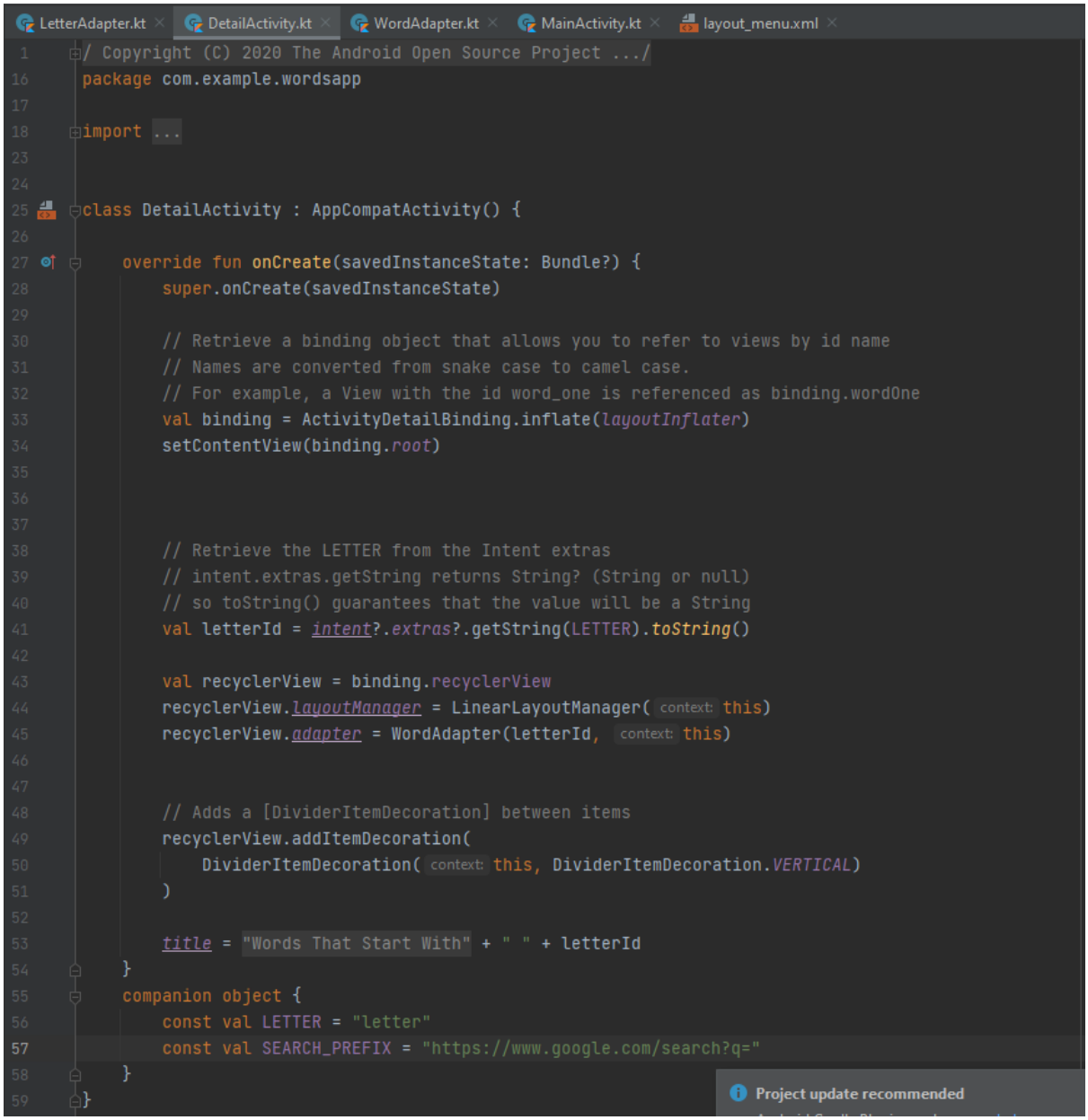


Task 4: Activities and Intents | Android Developers


Android
app
manifests
AndroidManifest.xml
java
com.example.wordsapp
DetailActivity
LetterAdapter
MainActivity
WordAdapter
java (generated)
res
drawable
ic_grid_layout.xml
ic_launcher_backgrounc
ic_launcher_foreground
ic_linear_layout.xml
layout
activity_detail.xml
activity_main.xml
item_view.xml
menu
layout_menu.xml
mipmap
values
res (generated)
Gradle Scripts

LetterAdapter.kt | DetailActivity.kt | WordAdapter.kt | MainActivity.kt | layout_menu.xml

```kotlin
/* Copyright (C) 2020 The Android Open Source Project ...*/
package com.example.wordsapp

import ...

/**
 * Adapter for the [RecyclerView] in [MainActivity].
 */
class LetterAdapter :
    RecyclerView.Adapter<LetterAdapter.LetterViewHolder>() {

    // Generates a [CharRange] from 'A' to 'Z' and converts it to a list
    private val list = ('A').rangeTo( other: 'Z').toList()

    /**
     * Provides a reference for the views needed to display items in your list.
     */
    class LetterViewHolder(val view: View) : RecyclerView.ViewHolder(view) {
        val button = view.findViewById<Button>(R.id.button_item)
    }

    override fun getItemCount(): Int {
        return list.size
    }

    /**
     * Creates new views with R.layout.item_view as its template
     */
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): LetterViewHolder {
        val layout = LayoutInflater
                .from(parent.context)
                .inflate(R.layout.item_view, parent, attachToRoot: false)
        // Setup custom accessibility delegate to set the text read
        layout.accessibilityDelegate = Accessibility
        return LetterViewHolder(layout)
    }

    /**
     * Replaces the content of an existing view with new data
     */
    override fun onBindViewHolder(holder: LetterViewHolder, position: Int) {
        val item = list.get(position)
        holder.button.text = item.toString()

        holder.button.setOnClickListener { it: View!
            val context = holder.view.context
            val intent = Intent(context, DetailActivity::class.java)
            intent.putExtra(DetailActivity.LETTER, holder.button.text.toString())
            context.startActivity(intent)
        }

    }

    // Setup custom accessibility delegate to set the text read with
    // an accessibility service
    companion object Accessibility : View.AccessibilityDelegate() {
        @RequiresApi(Build.VERSION_CODES.LOLLIPOP)
        override fun onInitializeAccessibilityNodeInfo(
            host: View?,
            info: AccessibilityNodeInfo?
        ) {
            super.onInitializeAccessibilityNodeInfo(host, info)
            // With `null` as the second argument to [AccessibilityAction], the
            // accessibility service announces "double tap to activate".
            // If a custom string is provided,
            // it announces "double tap to <custom string>".
            val customString = "Show Stored Words"
            val customClick =
                AccessibilityNodeInfo.AccessibilityAction(
                    AccessibilityNodeInfo.ACTION_CLICK,
                    customString
                )
            info?.addAction(customClick)
        }
    }
}
```


LetterAdapter.kt
DetailActivity.kt
WordAdapter.kt
MainActivity.kt
layout_menu.xml
/ Copyright (C) 2020 The Android Open Source Project .../
package com.example.wordsapp
import ...
class DetailActivity : AppCompatActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
// Retrieve a binding object that allows you to refer to views by id name
// Names are converted from snake case to camel case.
// For example, a View with the id word_one is referenced as binding.wordOne
val binding = ActivityDetailBinding.inflate(layoutInflater)
setContentView(binding.root)
// Retrieve the LETTER from the Intent extras
// intent.extras.getString returns String? (String or null)
// so toString() guarantees that the value will be a String
val letterId = intent?.extras?.getString(LETTER).toString()
val recyclerView = binding.recyclerView
recyclerView.layoutManager = LinearLayoutManager(context: this)
recyclerView.adapter = WordAdapter(letterId, context: this)
// Adds a [DividerItemDecoration] between items
recyclerView.addItemDecoration(
DividerItemDecoration(context: this, DividerItemDecoration.VERTICAL)
)
title = "Words That Start With" + " " + letterId
}
companion object {
const val LETTER = "letter"
const val SEARCH_PREFIX = "https://www.google.com/search?q="
}
}
Project update recommended

LetterAdapter.kt | DetailActivity.kt | WordAdapter.kt | MainActivity.kt | layout_menu.xml

```kotlin
/* Copyright (C) 2020 The Android Open Source Project .../
package com.example.wordsapp

import ...

/**
 * Adapter for the [RecyclerView] in [DetailActivity].
 */
class WordAdapter(private val letterId: String, context: Context) :
    RecyclerView.Adapter<WordAdapter.WordViewHolder>() {

    private val filteredWords: List<String>

    init {
        // Retrieve the list of words from res/values/arrays.xml
        val words = context.resources.getStringArray(R.array.words).toList()

        filteredWords = words
            // Returns items in a collection if the conditional clause is true,
            // in this case if an item starts with the given letter,
            // ignoring UPPERCASE or lowercase.
            .filter { it.startsWith(letterId, ignoreCase = true) }
            // Returns a collection that it has shuffled in place
            .shuffled()
            // Returns the first n items as a [List]
            .take( n: 5)
            // Returns a sorted version of that [List]
            .sorted()
    }

    class WordViewHolder(val view: View) : RecyclerView.ViewHolder(view) {
        val button = view.findViewById<Button>(R.id.button_item)
    }

    override fun getItemCount(): Int = filteredWords.size

    /**
     * Creates new views with R.layout.item_view as its template
     */
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): WordViewHolder {
        val layout = LayoutInflater
            .from(parent.context)
            .inflate(R.layout.item_view, parent, attachToRoot: false)

        // Setup custom accessibility delegate to set the text read
        layout.accessibilityDelegate = Accessibility

        return WordViewHolder(layout)
    }

    /**
     * Replaces the content of an existing view with new data
     */
    override fun onBindViewHolder(holder: WordViewHolder, position: Int) {

        val item = filteredWords[position]
        // Needed to call startActivity
        val context = holder.view.context

        holder.button.setOnClickListener { it: View!
            val queryUrl: Uri = Uri.parse( uriString: "${DetailActivity.SEARCH_PREFIX}${item}")
            val intent = Intent(Intent.ACTION_VIEW, queryUrl)
            context.startActivity(intent)
        }

        // Set the text of the WordViewHolder
        holder.button.text = item

    }
```

```kotlin
    // Setup custom accessibility delegate to set the text read with
    // an accessibility service
    companion object Accessibility : View.AccessibilityDelegate() {
        @RequiresApi(Build.VERSION_CODES.LOLLIPOP)
        override fun onInitializeAccessibilityNodeInfo(
            host: View?,
            info: AccessibilityNodeInfo?
        ) {
            super.onInitializeAccessibilityNodeInfo(host, info)
            // With `null` as the second argument to [AccessibilityAction], the
            // accessibility service announces "double tap to activate".
            // If a custom string is provided,
            // it announces "double tap to <custom string>".
            val customString = "Look up word in a Browser Search"
            val customClick =
                AccessibilityNodeInfo.AccessibilityAction(
                    AccessibilityNodeInfo.ACTION_CLICK,
                    customString
                )
            info?.addAction(customClick)
        }
    }
}
```

```kotlin
private fun setIcon(menuItem: MenuItem?) {
    if (menuItem == null)
        return

    // Set the drawable for the menu icon based on which LayoutManager is currently in use

    // An if-clause can be used on the right side of an assignment if all paths return a value.
    // The following code is equivalent to
    // if (isLinearLayoutManager)
    //     menu.icon = ContextCompat.getDrawable(this, R.drawable.ic_grid_layout)
    // else menu.icon = ContextCompat.getDrawable(this, R.drawable.ic_linear_layout)
    menuItem.icon =
        if (isLinearLayoutManager)
            ContextCompat.getDrawable( context: this, R.drawable.ic_grid_layout)
        else ContextCompat.getDrawable( context: this, R.drawable.ic_linear_layout)
}

override fun onCreateOptionsMenu(menu: Menu?): Boolean {
    menuInflater.inflate(R.menu.layout_menu, menu)

    val layoutButton = menu?.findItem(R.id.action_switch_layout)
    // Calls code to set the icon based on the LinearLayoutManager of the RecyclerView
    setIcon(layoutButton)

    return true
}

override fun onOptionsItemSelected(item: MenuItem): Boolean {
    return when (item.itemId) {
        R.id.action_switch_layout -> {
            // Sets isLinearLayoutManager (a Boolean) to the opposite value
            isLinearLayoutManager = !isLinearLayoutManager
            // Sets layout and icon
            chooseLayout()
            setIcon(item)

            return true
        }
```

LetterAdapter.kt | DetailActivity.kt | WordAdapter.kt | MainActivity.kt | layout_menu.xml

```kotlin
/ Copyright (C) 2020 The Android Open Source Project .../
package com.example.wordsapp

import ...

/**
 * Main Activity and entry point for the app. Displays a RecyclerView of letters.
 */
class MainActivity : AppCompatActivity() {
    private lateinit var recyclerView: RecyclerView
    private var isLinearLayoutManager = true

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)


        val binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        recyclerView = binding.recyclerView
        // Sets the LinearLayoutManager of the recyclerview
        chooseLayout()


        recyclerView = binding.recyclerView
        // Sets the LinearLayoutManager of the recyclerview

        recyclerView.layoutManager = LinearLayoutManager( context: this)
        recyclerView.adapter = LetterAdapter()
    }

    private fun chooseLayout() {

        if (isLinearLayoutManager) {
            recyclerView.layoutManager = LinearLayoutManager( context: this)
        } else {
            recyclerView.layoutManager = GridLayoutManager( context: this,  spanCount: 4)
        }
        recyclerView.adapter = LetterAdapter()
    }

            //  Otherwise, do nothing and use the core event handling

            // when clauses require that all possible paths be accounted for explicitly,
            //  for instance both the true and false cases if the value is a Boolean,
            //  or an else to catch all unhandled cases.
            else -> super.onOptionsItemSelected(item)
        }
    }


}
```

Project update recommended


11:02
Words
A
B
C
D
E
F
G
H
I
J
11:03
Words
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z