

Name : Kushwaha Ashvini G.

Class : SCEIT- B

Batch : 5B3

En. no : 21012021044

Sub : Mobile Application Development



XEITS PES : Mobile Application Development

Assignment - 1

Q.1 Based on your understanding, identify a recent business trend that has influenced the Android App developers and business in the mobile app industry.

Ans. Based on your/my understanding, one recent business trend that has influenced the Android platform is the "rise of the metaverse". The metaverse is a shared virtual world where people can interact with each other and with digital objects.

→ It is still in its early stages of development, but it has the potential to revolutionize the way we use our smartphones and other devices.

→ Google is one of the companies that is heavily investing in the metaverse. In 2021, Google announced that it would be creating a new team to focus on developing augmented reality (AR) and virtual reality (VR) products.

→ The rise of the Metaverse is impacting Android app developers and business in the mobile app industry in a number of ways.

• It is creating new opportunities for developers to create innovative new apps and experiences.

For example, developers could create apps that allow users to explore the metaverse, play games in the metaverse, or even create their own virtual world.

- The rise of the metaverse is changing the way that business interact with their customer.

For example, business should use the metaverse to create virtual showrooms or customer support center. also use the metaverse to host events or to launch new products.

→ Some examples how the metaverse is impacting Android app developers and business in the mobile ~~app~~ app industry :-
~~mobile industry~~ :-

~~new opportunity~~

- New way to categories :- Opportunities to ~~cost~~ ~~cost~~ create innovative new apps and experiences.
- New ways to monetize app: To monetize their apps in new ways.
- New ~~ways~~ ways to engage ~~with~~ ~~with~~ customers: To ~~cost~~ engage with their customers in few ways, ex. to provides ~~cost~~ customer support.

→ Developers and business who are able to adapt to this trend will be well-positioned to succeed in the future.

Q.2 What is the purpose of an Inflater of layout in Android development and how does it fit into the architecture of Android layouts?

Ans. → In Android Development, an "Inflates" refers to the Layout Inflate which plays a crucial role in creating a user Interface (UI) from XML layout files. Its primary purpose is to take an XML layout resource and convert it into a corresponding view object in memory. Here's how the Layout Inflater fits into the architecture of Android layouts:

1) XML layout files :- In Android, UI components are often defined using XML layout files. These files describe the structure and appearance of the UI, specifying things like widgets, their properties and their placement within the UI.

2) Layout Inflation :- When your Android app runs, it needs to turn these XML layout files into actual view objects that can be displayed on the screen. This process is known as "layout inflation".

3) Layout Inflates :- It is responsible for reading the XML layout files and instantiating the corresponding view objects in memory. It takes the XML files as input, parses it, and creates the view objects, such as

Textviews, Buttons, etc. as specified in the XML.

- 4). Dynamic UI creation :- Layout inflation is particularly valuable when you need to create UI elements dynamically, for example in response to user interactions or when working with RecyclerView to display lists of items.
- 5). Binding Data :- Once the view objects are created, they can be bound to them. This is typically done using data binding technique or programmatically setting properties and values.
- 6). Displaying UI :- After inflation and customization, the view objects can be added to the app's layout hierarchy and displayed on screen.

Q.3 Explain the concept of a Custom Dialog Box in Android applications. Provide examples to illustrate its use.

Ans. → A custom dialog box in Android is a dialog box that is created by the developers, rather than using one of the built-in dialogs. This allows the developer to create a dialog box with a custom layout and functionality.

→ To create a custom dialog box, the developer first needs to create an XML layout file for the dialog box. The layout file can contain any view objects that the developer wants to display in the dialog box.

- Purpose : Custom-dialogs are used when you want to present information , receive user input or perform actions within a self contained , isolated UI element that temporarily interrupts.
- Customization : Developers can design the dialog's appearance , layout , and behaviors according to their app's branding or specific requirements . Their customization allows for creativity in design and functionality.

→ sample example of creating and using a custom dialog in Android :

```

fun showCustomDialog() {
    val customDialog = Dialog(this)
    customDialog.setContentView(R.layout.custom_dialog)
    val messageTextView = customDialog.findViewById<TextView>(R.id.message_TextView)
    val okButton = customDialog.findViewById<Button>(R.id.okButton)
    messageTextView.text = "This is a custom dialog!"
    okButton.setOnClickListener {
        customDialog.dismiss()
    }
    customDialog.show()
}

```

⇒ Use case of Custom dialog box : Login Confirmation Dialog, settings, Informational Pop-up, Media Playback controls.

Q.4 How do activities , services and the Android Manifest file work together to make an android app? Can you describe their main roles and provide a basic example of how they cooperate to design a mobile app?

Ans. • Activities :

Activities represent individual screens or UI components in an android app. They manage the user's interface and user interactions.

• Android Manifest file :

The `AndroidManifest.xml` file is like the app's blueprint. It declares the app's components and defines how they interact with the Android system and other components.

Example : In `AndroidManifest.xml`, you specify which activity are part of your app, their launch modes, permissions and service understand your App's structure and behaviour.

• Service :

Services are background components that perform long-running operations or handle tasks that don't require a user interface. They can run even if the app's UI is not visible.

```

→ class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        startServiceButton.setOnClickListener {
            val serviceIntent = Intent(this, NotificationService::class.java).startService(serviceIntent)
        }
    }
}
  
```

```

→ class NotificationService : IntentService("NotificationService") {
    override fun onHandleIntent(intent: Intent?) {
        if (intent != null) {
            createNotification()
        }
    }

    private fun createNotification() {
        val channelId = "my_channel"
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            val name = "my channel"
            val notificationManager = getSystemService(NotificationManager::class.java)
            notificationManager.createNotificationChannel(channel)
            val builder = NotificationCompat.Builder(this, channelId)
                .setSmallIcon(R.drawable.ic_launcher_foreground)
                .setContentText("This is notification from service")
        }
    }
}
  
```

Q.5 How does the Android Manifest file impact the development of an Android application? Provide an example to demonstrate its significance.

Ans. → The Android Manifest file is a crucial component in the development of an android application. It serves several important purposes and its content significantly impacts how the android system interacts with and manages your app. Significance of the Android Manifest File :

- App Configuration
- Intent Filters
- App LifeCycle
- Component Declaration
- Permissions

• Example :

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
    package="com.example.myapp">
```

```
    <application>
```

```
        android:allowBackup="true"  
        android:icon="@mipmap/ic_launcher"  
        android:label="@string/app-name"  
        android:soundIcon="@mipmap/ic_launcher-sound"  
        android:supportRtl="true"  
        android:theme="@style/AppTheme">
```

```
    <activity android:name=".MainActivity">
```

```
        <intent-filter>
```

```
            <action android:name="android.intent.action.MAIN"/>
```

```

<category android:name="android.intent.category.LAUNCHER">
</intent-filter>
</activity>
<activity android:name=".SecondActivity">
    ... Declare additional activities here ...
</activity>
<uses-permission android:name="android.permission.
    Intent/>
    ... Declare required permissions here ...
</application>
</manifest>

```

Q.6 What is the role of resources in Android Development ? Discuss the various types of resources and their significance in creating well-structured . provide examples to clarify your point .

Ans → Resources play a fundamental role in Android Development by providing a structured way to manage assets values , layouts and others elements used in your app . They help create flexible , maintainable and device independant app . The various types of resources and their significance with examples :

1) Layout Resources :

- Type : xml files in the 'res/Layout' directory
- Significance : Defines the structure and appearance of the app's user interface
- Example : 'activity_main.xml' defines the layout of your main activity , specification UI components like

buttons, text views and their arrangement.

< Button

```
    android:id="@+id/myButton"  
    android:layout_width="wrap-content"  
    android:layout_height="wrap-content"  
    android:text="click Me" />
```

< 2> Drawable Resources :

- type : Images and drawable assets in the 'res/drawable' directory.
- Significance : Store graphics, icons, and images used in your app.
- Example : 'Ic_launcher.png' is the app's launcher icon.

< 3> String Resources :

- Type : String defined in XML files under 'res/values'.
- Significance : store text strings, making it easier to provide translations and maintain consistency.
- Example : 'res/values/String.xml' contains string resources
 - <string name="app-name">my app</string>
 - <string name="welcome-message">Welcome to my app</string>

< 4> Color Resources :

- Type : colors defined in XML files under 'res/values'.
- Significance : store color values, ensuring consistency in the app's design.

- Example : "res/values/colors.xml" defines color resources.

```

<color name="Primary-color">#007ACC</color>
<color name="primary-color">#FFA500</color>
```

5) style Resources :

- type : styles defined in xml files under 'res/values'.
- Significance : Define reusable styles for UI components.
- Example : 'res/values/style.xml' defines style

```

<style name="myButtonStyle">
    <item name="android:background">@drawable/my-button
    </item>
    <item name="android:textColor">@color/primary-color
    </item>
</style>
```

6) Dimension Resources:

- type : Dimensions defined in xml files under 'res/values'
- Significance : store dimensions values, ensuring a consistent layout
- Example : 'res/values/dimens.xml' defines dimensions resources

```

<dimen name="margin_large">16dp</dimen>
<dimen name="padding_medium">8dp</dimen>
```

7) Raw Resources :

- type : files stored in the 'res/raw' directory
- Significance : store non-dml files, such as Json data, audio.
- Example : store a Json file for app configuration.

Q.7. How does an android service contribute to the functionality of a mobile application? Describe the process of developing an android service.

Ans. → Contribution of Android Service:

1] Background Processing : Services allow apps to perform tasks in the background without blocking the user interface.

2] Long-running Operations : Services are ideal for healing operations that require more time to complete , such as playing music.

3]. Inter-component communication : Service enable complements like activities , broad-cast receivers and services to communicate with each other efficiently.

4] Foreground services : Can run in the background, even when the app isn't in the foreground. This is useful for features that requires ongoing user interaction , like music playback.

Process of Developing an Android Service:

- Define the Service class: Create a new Java or kotlin class that extends the 'service' class. Override method like onCreate() , onDestroy() , onStartCommand() to define the behaviour of your service.

- Configure Service in Manifest : Declare your service in the Android Manifest.xml file to inform the Android System about its existence and configuration.

Ex.

```
<service android:name=".MyService"/>
```

- Start On Bind the Service : Decide whether you want to start your service or bind it to other components. Use startService() or bindService()
- Implement Services Logic : In services class, implement the specification logic. Use ~~the~~ your services needed to perform its ~~the~~ task.
- Handle Life cycle : Release resources when they are not longer needed and consider using 'stopSelf()' or 'stopService'.
- Interact with others components : Use appropriate mechanisms like intents, broadcast to facilitate ~~component~~ communications.
- Foreground service (optional) : If your service needs to run in the foreground, ~~startforeground~~.
- Testing : Thoroughly test your services to ensure if functionals as expected, including handling various scenarios like network failures.

- Optimization : Optimize your service for performance and resource efficiency to minimize battery usage.
- Error Handling and Logging : Implement proper error handling and logging mechanisms to diagnose and address issues that may occur while service is running.

6/10/23