U V PATEL COLLEGE OF ENGINEERING
GANPAT UNIVERSITY

U. V. Patel College of Engineering

GANPAT UNIVERSITY, KHERVA-384012, DIST - MEHSANA. (N.G.)

Assignment :- 1

Q:-1 Based on your understanding, identify a recent business trend that has influenced the Android platform. Explain how this trend impacts Android app developers and businesses in the mobile app industry.

⇒ One recent business trend that has influenced the Android platform is the growing demand for mobile gaming.

- Increasing popularity of Smartphone and advancement in technology, more and more people are turning to their Android device for gaming.
- It has also opened up new opportunities for businesses in the mobile app industry, as they can tap into the locrative gaming market by creating and monetizing enjoying and impersive gaming experience for android user.
- This trend impacts Android app developer and businesses in the mobile app industry by creating a huge market for gaming apps.
- Developers have the opportunity to create innovative and addictive games that cater to the interest of android users.
- This can lead to increased downloads, user engagement and monetization through in-app purchess or ads.
- Overall, the demand for mobile gaming on android presents a valuable opportunity for developers and businesses to thrive in the mobile app industry.

Q:-2 What is the purpose of an Inflator of layout in Android development, and how does it fit into the architecture of Android layouts?

→ The purpose of an Inflator in android development is to take an xml layout file and convert it into corresponding View objects in memory.

- It is responsible for parsing the xml file, creating the necessary view objects, and setting their attributes based on the information provided in the xml.
- In terms of the architecture of Android layouts, the Inflator plays a crucial role. when an activity needs to display a layout, it uses the Inflator to convert the xml layout.
File into a View hierarchy.
- The Inflator participates in the process of creating the UI and populating its contents. It traverse the xml file, create the necessary View objects, and attaches them to their parent containers. It also handle the attributes value specified in the xml, such as setting text, colors, size and other properties on the corresponding view objects.
- Overall, the Inflator is a Fundamental component in Android layouts as it bridges the gap between the xml Based layout file and the actual View Object that compose the User interface.

DEEP XEROX

Q:-3 Explain the concept of a customDialogBox in Android applications. provide examples to illustrate its use.

⇒ In Android application, a customeDialogBox is a user interface element that allows developers to create customized dialog windows to promt the user information or display important message.
- It provides flexibility in terms of appearance and behavior compared to the standard Alert Dialog provided By the Android SDK.

Examples:-

MainActivity.kt :-

```
import android.os.Bundle
import android.widget.Button
import android.appcompact.app.AppCompatActivity()
class MainActivity : AppCompatActivity() {
    Override Fun OnCreate(SavedInstanceState : Bundle)
    { super.OnCreate(SavedInstanceState)
      SetContentView(R.layout.activity_main)
      Val Dialog = findViewById Button(R.id.btn)
      Dialog.setonClickSter {
          Val customDialog = Dialog(this)
          customDialog.show()
      }
    }
}
```

→ activity-main.xml :-

```
<androidx.constraintlayout.widget.ConstraintLayout
 xmlns:android="http://schemas.android.com/apk/res/
                                android"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 xmlns:app="http://schemas.android.com/apk/res-auto">
<Button
    android:id="@+id/btn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:text="Custom Dialog Box" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

→ Dialog.kt :-

```
import android.app.Dialog
import android.content.context
import android.os.Bundle
import android.widget.Button
class Dialog(context: context) : Dialog(context) {
   override fun onCreate(savedInstanceState: Bundle) {
   super.onCreate(savedInstanceState)
   setContentView(R.layout.activity_dialog)
   val btn = findViewById<Button>(R.id.btn)
   btn.setOnClickListener {
       dismiss()
   }
}
}
```

→ activity_dialog.xml:-

```
<linearlayout
  xmlns:android = "http://schemas.android.com/apk/res/android"
  android:layout_width = "match_parent"
  android:layout_height = "wrap_content"
  xmlns:android = "http://schemas.android.com/apk/tools"
  android:orientation = "vertical"
  tools:context = ".Dialog" >

<TextView
    android:layout_width="wrap_content"
    android:layout_hight = "wrap_content"
    android:text = "This is custom Dialog Box"
    android:id = "@+id/tv" >
</TextView>

<Button
    android:id = "@+id/btn"
    android:layout_width = "match_parent"
    android:layout_height = "~~match_parent~~ wrap_content"
    android:text = "ok" />

</linearlayout>
```

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?

⇒ Every app project must have an Android manifest file at the root of the project source set.

- The manifest file describes essential information about your app to the Android build tools, the android operating system and google play.
- The component of the app which include all activities, Services, broadcast receiver and content providers.

- Activities:-

Activities are the building blocks of the user interface in Android apps. they represent individual screen or user interface.

Example:-

Activities handle the UI design and manage the UI state. Display the content to user.

- Services:-

Services perform background tasks independtly of the UI, ensuring task continue running even when the app is not in the foreground.

example:-

Service are responsible for long operation. such as file download, music playback etc.

→ Android Manifest.xml :-

The Androidmanifest.xml file is the app's configuration file, providing essential information to the Android system about the app's components and behavior.

Example :-

The mainfest file is used to specify how the app interact with the android system.

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

⇒ Androidmanifest.xml file significantly impacts the development of an android application in several ways.
- It serves as a crucial configuration file that defines the app's structure, behavior and interaction with the Android System.

1. Component Declaration:-
The manifest file declares all the components of the app, including activities, services, broadcast Reciever and content providers. All component tell the Android system how to manage and interct with this components.

2. Permissions:-
Permissions ~~your app~~ require to access System resource and user data. properly declaring permissions in the manifest informs user about the app's capabilities and requests their consent.

3. App configuration:-
The manifest file include metadata and setting that configure various aspects of the app, such as its theme, icon, label, orientation and launch mode.

4. intent filters :-

Intent filter are defined in the manifest to specify how your app responds to external event and requests. For example, you can define an intent filter to handle opening specific file type.

Example:-

```
<uses-permission android:name="android.permission.
                                camera" />
<activity android:name=".CameraActivity">
  <intent-filter>
    <action android:name="android.intent.action.
                                MAIN" />
    <category android:name="android.intent.category.
                                LAUNCHER" />
  </intent-filter>
</activity>
<application
    android:label="My CameraApp"
    android:icon="@drawable/camera_icon>
</application>
```

Q:-6 What is the role of resources in Android development? Discuss the various types of resources and their significance in creating well-structured applications. Provide examples to clarify your points.

→ Resources in Android development are essential for providing various type of content to your app. Such as images, strings, layouts and more. They help in organizing and managing the app's content separately from the code, making it easier to update and localize your app.

Types of Resources:-

1. layout Resources ('res/layout/'):-
- Define the structure and arrangement of user interface elements.
- Separating UI layout from code promotes maintainbity and adaptability across device and orientations.
- Example:-

```
<linearlayout
  xmlns:android="http://schemas.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical">
<TextView
    android:id="@+id/tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello" />
</linearlayout>
```

2. String Resource ('res/value/string.xml):-
- Store text strings used in the app facilitiing localization and interctionalization.
- Allow for translation of app content into different language without changing code.
- Example:-

```
<resources>
    <string name="F-Name">Jensi</string>
    <string name="L-Name">Patel</string>
</resources>
```

3. Drawable Resource ('res/drawable'):-
- Store images, icons and graphics used in the app with variations for different screen densities.
- Ensure proper image resolution and quality on various devices.
- Example:- 'res/drawable/ic-launcher.png'

4. Color Resources (res/values/colors.xml):-
- Define color values used throughout the app. For consistent theming.
- Promotes visual consistency and make it easier to ~~uptoid~~ update the app's color scheme
- Example:-

```
<resource>
    <color name="primary_color">#007ACC</color>
    <color name="accent_color">#FF4081</color>
</resources>
```

5. Style and theme Resource ('res/values/style.xml):
- Define styles and themes for UI elements.
- allowing for consistent styling.
- make it easier to apply and maintain a consistent look and feel across the app.
- Example:-

```
<resource>
    <style name="AppTheme"
        Parent="theme.AppCompat.Light.DarkActionBar">

    </style>
</resource>
```

6. Raw Resource (res/raw/):-
- Store raw data files, such as audio or video to be used in the app.
- Allow for the inclusion of binary data or media without compression.
- Example:- 'res/raw/audio.mp3'

Q:-7 How does an Android Service contribute to the functionality of a mobile application? Describe the process of developing an Android Services.

- Android service is a components that is used to perform operations on the background such as playing music, handle network transection interacting content providor etc. It doesn't hus any UI.
- The service runs in the Background idefinitely even if application is destroyed.
- Service can be bounded by a component to perform interactivity and inter process communication.
- Developing an Android Services:-

(1) Create Service class:-
- The create a Java or Kotlin class that extends the Service class or its Subclass.
- Implement the service core functionality within this class, typically within the onCreate and onStartCommand methods. Ensure that the service operation can run independently in the background.

2. Declare the service in the Androidmanfest.xml:-
- Register the service in the Androidmanifest file. Specify its name and any required permission.

3. Start and Stop Service:-
- You can start a Service using an 'Intent' from any component of your appon even from other apps. Use 'startServices(Intent)' to initiate the Services execution.

- Use 'StopService(intent)' or stopSelf()' from within the service to stop it when its no longer needed.
- Example:-

```
Val intent = Intent(this, MyService::class.Java)
StartService(intent)
```

4. Destroy Services:-
- Service is no longer used and is being destory.
- Your service implement this to clean up any resource such as threads, registred listeners receivers, etc.