Ganpat University | U.V. Patel College of Engineering

U. V. Patel College of Engineering

Ganpat Vidhyanagar, Mehsana-Gandhinagar Highway, Mehsana - 384012

Patel Krupa D.
21012021070
CEIT-B (5B-3)
MAD (Mobile Application Development)
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 significant bussiness trend that influenced the Android platform was the growing focus on privacy and data security. Google, the company behind Andro has been implementing stricter privacy measure and encouraging developers to follow suit.

- Here's how this trend impacts Android app developers and businesses in the mobile app industry.

1 Privacy Regulation and compliance :-

→ With the introduction of privacy regulations like GDPR and CCPA, Android app developers have had to ensure that their apps are compliant with these laws. This often involves obtaining explicit user consent for data collection and implementing mechanisms for users to control their data.

GHANSHYAM

2. User trust :-

- Building and maintaining user trust has become cru
Apps that handle user data must be transparent about
data practices. Developers need to provide clear privacy
policies and minimize data collection to gain user
confidence.

3. Advertising & Monetization :-

- Changes in Android's advertising and tracking policies
affect how businesses monetize their apps. Developers may
need to find revenue streams, such as subscription model
to target users with ads or rely on alternative, to
sustain their businesses.

4. App store Guidelines :-

- Google play store has been tightening its guidelines related t
privacy and data security. Developers must adhere to these
guildines to get their apps published, which can impact
app development timelines and strategies.

5. Data Minimization :-

- Business have to be more selective about the data they
collect and how they use it. This can lead to more efficien
app development and potentially reduce legal and security
risks.

6. Security Investments :-

As data breaches become more costly and damaging to
businesses, there's a growing need for increased
security measures within apps. Developers need to invest
in robust encryption, authentication, and data protection
mechanisms.

7. User Experience :- Stricter privacy measures can sometimes lead to a less personalized user experience. Developers need to find a balance b/w respecting user privacy and providing valuable and services.

Q=2] What is the purpose of an Intluter of Layout in Android development, and how does it fit into the architecture of Android layouts?

→ In Android development, an intluter is a mechanism used to convert an XML layout file into corresponding View objects in your app.

- Purpose of layout Intluter :-

1. Dynamic UI Generation :- Intluters enable dynamic UI genration by allowing developers to create views programmatically at runtime based on predefined XML layouts.

2. Reuse of layout components :- Intluters facilities the reuse of layout components. Instead of duplicating the same layout definition in multiple places within your app, you can define it once in XML and intlate it whenever needed.

3. Separation of concerns :- In Android, UI components are typically defined in XML layout files, promoting a separation of concern between the UI and business logic.

- How it fits into Android Layout Architecture :-

1. XML Layout definition :-

- Developers define the structure and appearance of UI components using XML layout files in the "res/ layout" directory.

2. Activity / Fragment Initialization :-

- In the "on create" method of an Activity or Fragment, the layout inflater is typically used to set the content view.

3. Inflating Layouts :- The "layout Inflater" class is employed to instantiate XML layouts, converting them into view objects.

Ex Val inflater = Layout Inflater.from (context)
Val root view = inflater.inflate (R.layout.my_layout, parent view, false)

4. Accessing Views :-

- Views within the inflated layout can be accessed Programmation

Ex Val my Text view = root view.findViewById<TextViews> (R.id.textviews)

5. Setting Content View :-

- The inflated view hierarchy is set as the content view of the Activity.

Ex SetContentView (root view)

37 Explain the concept of a customer DialogBox in Android applications. Provide examples to illustrate its use.

→ A "customDialogBox" in Android is a pop-up window that developers can design and customize to suit the specific needs and branding of their application. It's way to present information, prompt user input, or confirm actions in a visually customized manner. The Android SDK provides the "Dialog" class, and developers often extend it or use the "AlertDialog" class to create custom dialogs.

EX MainActivity.kt :=

```
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import android.widget.Button
import com.example.myapp.CustomDialog
import com.example.myapp.R

class MainActivity : AppCompatActivity()
{
  override fun onCreate(savedInstanceState: Bundle?)
  {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    val showDialogButton: Button = findViewById(
                                   R.id.showDialogButton)
    showDialogButton.setOnClickListener
    { val customDialog = CustomDialog(this)
    }
  }
}
```

Q=4] How do activities, services, and the Android ma... work together to make an Android app? Can yo... their main roles and provide a basic example of ho... they cooperate to design a mobile app?

→ 1. Activities :-

- Role :- Activities are the user interface components of an Android app. They represent individual screens with which users can interact. Each activity is a self-contained unit with its own UI layout.

2. Services :- Services perform background tasks without a user interface. They are used for tasks that need to run independently of the UI, such as playing music, fetching data from the internet, or performing other long-running operations.

3 Android Maintest file :-

- The Android Manifest.xml file is a configuration file that provides essential information about the app to the Android system. It declares the app's components their properties, and the permissions they require.

Ex
```
class MainActivity : AppCompatActivity()
{
override fun onCreate(savedInstanceState: Bundle?)
  {
  super.onCreate(savedInstanceState)
  setContentView(R.layout.activity_main)
  val playButton: Button = findViewById(R.id.playButton)
  playButton.setOnClickListener
  {
   val intent = Intent(this, MusicService::class.java)
   startService(intent)
  }
}
```

Example Of Services :-

```
class music Service : Service()
{
Private lateinit var media player : Media player
Override fun on Bind (intent : Intent?) : I Binder?
{
return null
}
Override fun on start command ( Intent : Intent?,
        flags : Int, start Id : Int) : Int
{
media player = media player. create (this, R.raw.
media player. start ()                    song)
return START - STICKY
}
Override fun on Destroy()
{
Media player. release()
super. on Destroy ()
}
}
```

→ Example Android Mainfest file :-

```
<manifest xmlns: android = "http://schemas.android.
        com/apk/res/android"
package = "com. example. music player" >
<application
    android: allow Backup = "true"
```

```
android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
android:roundIcon="@mipmap/ic_launcher_rou
android:supportsRtl="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>
<service android:name=".MusicService" />
  </application>
</manifest>
```

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

→ The Android manifest file impact the development:-

1. Declaration of App components :-

→ An manifest file declares all the components of an Android app, including activities, services, broadcast receivers and content providers.

2. Setting Main Activity :-

→ The manifest file specifies the main activity, which is the entry point of the app. This is the activity that is launched when the app is started.

3. Permission and security :-

→ The manifest file is used to declare the permissions that the app requires to access certain device features or data

4. Intent filters :- Intent filters in manifest file define how the app responds to implicit intents. They specify the types of actions, categories and data types the app can handle.

E.x
```
<manifest xmlns:android="https://schemas.android.com/apk/res/android"
    package="com.example.cameraapp">
    <uses-permission android:name="android.permission.CAMERA" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="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=".CameraActivity">
        </activity>
        <service android:name=".CameraService" />
    </application>
</manifest>
```

Q-6] What is the role of resources in Android dev... Discuss the various types of resources and the... in creating well-structured applications. Prov... examples to clarify your points.

→ In Android development, resources play a crucial role in creating well-structured and adaptable applications. Resources in Android are external elements such as images, strings, layouts, color and other assets that are separate from the application code. They are used to provide flexibility, maintainability and support for various device configurations.

1. String Resources :- String resources are used to store text strings that are displayed in user interface. Storing strings in a separate resource file makes it easier to manage translations and adapt to different screen sizes.

Ex res/values/strings.xml:-

```
<resources>
    <string name="app_name">MyApp</string>
    <string name="welcome_message">Welcome to my App!
    </string>
</resources>
```

2. Drawable resources :-

Role:- Drawable resources include images and graphics used in the UI. Different versions of images can be provide for different screen densities.

Ex res/drawable/icon.png :-

```
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/icon" />
```

3. Color Resources :-
Role :- It store color values that can be easily reuse across the app. This allows for consistent theming and makes it simple to update the color scheme.

Ex res/values/colors.xml :-

```
<resources>
<color name = "primary_color"> #3498db </color>
<color name = "accent_color"> #ff4081 </color>
</resources>
```

Q=7] How does an Android Services contribute to the functionality of a mobile applications? Describe the process of developing an Android services.

→ Contribution to functionality :-

1. Background processing :-

Services are ideal for tasks that should continue running even when the app is not actively interacting with the user. Examples include playing music, fetching data from the internet, or processing updates in the background.

2. Inter-component Communication :-

HANSHYAM Services can communicate with other app components

Such as activities or other services, using Andro
Process communication mechanisms like Intent
Binder.

3. Long-Running Operations :-

- Services are suitable for executing long-running operation such as file downloads, synchronization with servers, or continuous sensor monitoring.

4 Foreground service:-

- Foreground services are a special type of service that provides a Presistent notification, ensuring that the user is aware of ongoing tasks.

- Process of Developing an Android service :-

1. Create a service class :-

- Create a new class that extends the 'service' class. This class will contain the logic for your service.

```
- class my service : service()
{
override fun onBind (intent : Intent?) IBinder?
{
return null
}
override fun on start command (intent : Intent?, flags:
                    Int, startId) : Int) : Int
{
return START_STICKY
}
override fun onDestroy()
{
super.onDestroy()
}
```

:- Declare the service in the manifest :-

Declare your service in the Android Maintest.xml file to let the Android system know about it.

<service android: name=".My service" />

3. Start the service :-

- You can start the service by creating an 'Intent' and using 'start service()'.

- Val intent = Intent (context, My service :: class.java)

Context. start Service (intent)

4 Handle Service life cycle :-

The service lifecycle methods ('onCreate()', 'on start Command()', 'On Bind()' and 'on Destroy()' allow you to manage the behavior of your service at different points in its life.