Patel Anshkumar Hitendrabhai
21072011066 CE-IT-B ~~A05~~ B-4

MAD - Assignment I

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 business in the mobile app industry.

→ Recent business trend that has influenced the android platform is mobile Commerce or M-Commerce. Mobile Commerce is the trend of conducting online transactions, including shopping, payments, and other financial activities, through mobile devices. It had significant impact on Android app developers and business in the mobile app industry.

→ Following points describe how this trend impacts android app developers and business in mobile app industry:

1. Increased demand for E-Commerce Apps:

Trend drives the need for Android developers to create and maintain feature-rich e-commerce apps. Ex:- Amazon or Flipkart.

2. Payment Integration:

Developers need to stay updated with the latest payment technologies and integrate them into their apps. This might include support for mobile wallets, payment gateways, and contactless payment options. Ex:- Google Pay or Phone Pay.

Patel Anshkumar Hitendrabhai
21012077066
5B-4

3. Security:-

As mobile commerce involves sensitive financial information, security is a top priority. Android app developers must implement robust security measures to protect user data, which may include encryption, biometric authentication.

4. Cross-Platform Development:-

Developers need to write code that works on both Android and iOS devices, to reach a wider audience and reducing development time and costs.

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

→ Inflater or layout Inflater is a component used to create instances of Android view objects from XML layout resource files.

→ The purpose of an Inflater is to take a layout resource and convert it into a corresponding view hierarchy that can be used within an Android Application.

→ In other words, it helps you instantiate the UI components described in XML and incorporate them into your app's user interface.

Patel Anshkumar Hitendrabhai
210120107066 5B-4

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.

3. layout Inflater:-
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.

4. Binding Data:- once the view objects are created, they can be further customized and data can be bound to them. This is typically done using data binding techniques or programmatically setting properties and values.

5. Displaying UI:- After inflation and customization the view objects can be added to the app's layout hierarchy and displayed on the screen.

Patel Anshkumar Hitendrabhai
21012011066 5B-4

3. Explain the concept of a custom dialog box in Android applications. Provide Examples to illustrate its use.

→ Custom Dialog Box is a customized user interface component that displays info. or interactions in a pop-up window.

→ Custom dialog allows developers to create a unique user experience by designing the dialog's appearances and behavior according to their specific needs.

→ It can handle various user interactions and actions, such as input validation, button clicks, and data processing, depending on the use case.

```
Ex:- fun showCustomDialog() {
        val customDialog = Dialog(this)
customDialog.setContentView(R.layout.custom_dialog)
val msgTextView = customDialog.findViewById
                    (R.id.msgeTextView)
val OKButton = customDialog.findViewById(R.id.
                                    OKButton)
msgTextView.text = "This is a customDialog!"
OKButton.setOnClickListener {
                customDialog.dismiss()}
    customDialog.show()
    }
```

Patel Anshkumar Hitendrabhai
2707207066 5B-4

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.

→ 1. Activities:

Role: Activities represent individual screens or UI components in an Android app. They manage the user interface and user interactions.

2. Services:-

Role: Services are background components that perform long running operations or handle tasks that don't require a user interface.

3. Android manifest file:

Role: The Androidmanifest.xml file is like the apps blueprint. It declares the apps components and defines how they interact with the Android system and other components.

```
Ex:- class MainActivity : AppCompactActivity() {
    override fun onCreate(savedInstanceState:
                          Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        System.SetOnClickListener {
            val intent = Intent(this, NotificationService::
                                class.java)
            StartService(intent)
        }
    }
```

Patel Anshkumar Hitendrabhai
210120107066 5B-4

```
class NotificationService: IntentService("Notificatio
                                -ervice"){

    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.create NotificationChannel(channel)
                                                    }

        val Builder = NotificationCompat.Builder(this, channelID)
        .setSmallIcon(R.Drawable.ic_launcher_foreground)
        .setContentText("This is notification from service".)
    } }
```

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 is a crucial component in the development of an Android app application. It serves several important purposes, and its content significantly impacts how the android system interacts with and manages Your App.

- App Configuration
- Component declaration
- Intent filters
- App lifecycle
- permissions.

U.V. Patel College of Engineering
GANPAT UNIVERSITY, KHERVA-384012, DIST - MEHSANA. (N.G.)

Ex1 - <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: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=".SecondActivity">
... Declare additional activities here ....
</activity>
</application>
</manifest>

Patel Anshkumar Hitendrabhai
21072011066 5B-4

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

→ Resources play a fundamental role in Android development by providing a structured way to manage assets, values, layouts and other elements used in your app. They help create flexible, maintainable and device independent application.

1. layout Resources:
- type: XML files in the 'res/layout' directory.
- significance: Define the structure and appearance of the app's user interface.
- Ex:- activity_main.xml

```
<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'
- significance: store graphics, icons, and images used in your app.
- Ex:- 'ic_launcher.png' is the app's launcher icon

Patel Anshkumar Hitendrabhai
21012077066 5B-4

3. String Resources:
- type: Strings defined in xml files under 'res/values'.
- significance: store text strings, making it easier to provide translations and maintain consistency.
- Ex:- res/values/strings.xml
<string name="app_name">My App</string>
<string name="welcome_msg">welcome</string>

4. color resources:
- type: colors defined in xml files under 'res/values'.
- significance: store color values, ensuring consistency in the app's design.

Ex:- <color name="primary_color">#007Ace</color>
<color name="accept_color">#FPA500</color>

5. Style Resources:
- type: styles defined in xml files under 'res/values'.
- significance: Define reusable styles for UI components.
- Ex:- <style name="myButtonStyle">
<item name="android:background">@drawable/my_Button</item>
<item name="android:textcolor">@drawable/primary_color</item>
</style>

Patel Anshkumar Hitendrabhai
210120107066 5B4

6. Dimension Resources:
-type: Dimensions defined in XML files under res/value
-significance: store dimension values, ensuring a consistent layout.
-Ex:- <dimen name="margin_large">16dp</dimen>
<dimen name="Padding_medium">8dp</dimen>

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

→ Service is a component of the Android that performs long-running operations in the background, independently of the UI. They are used to execute tasks or provide functionality that should continue to run even when an app's user interface is not in the foreground and are an essential part of Android.

→ Background Processing: Services are used for bg processing tasks that don't require user interaction.
Ex:- they can perform download data, check for updates.

→ multithreading:- Services can run tasks on separate threads, which helps prevent blocking the main UI thread and the app responsive.

→ media playback: Services are commonly used for media playback because media playback should continue even when the app is not in the foreground.

→ location updates: Services can monitor location changes in the bg, enabling location-based services and features like tracking.

→ Inter-Component Comm:- Services can act as a comm. bridge btw different components of an app, such as Activities, Fragments, and Broadcast Receivers.

Process of creating service:

1. Create a Service class: create a Java or Kotlin class that extends the Service class or its subclasses.

2. Declare the service in the menifest:
Declare the service in menifest file by simply adding <service> element to in the menifest file specifying its name and other attributes as needed.

3. Start the service:
Start service using the startService() method, passing an Intent with the service class. And we can also bind to a service using the bindService() method.

4. Perform background Tasks: Inside your service, implement the logic for the bg tasks you want to perform. This may involve network operations, file I/O, database access, etc.

Patel Anshkumar Hitendrabhai
21012011066 5B-11

5. Stop the service:

when the service has completed its tasks is no longer needed, we can stop it using the stopself() method if it's a started service or by unbinding it if it is a bound service