U. V. Patel College of Engineering

Ganpat Vidhyanagar, Mehsana-Gandhinagar Highway, Mehsana - 384012

Assignment-1

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.

→ As per my knowledge, one notable trend in the mobile app industry that has been influencing the android platform is the rise of progressive web apps (PWAs). PWAs are web-applications that often app-like experiences directly through web browsers.

- Impact on android app developers:

(1) Cross-platform Compatibility: PWAs are designed to work seamlessly across various platforms and devices, including android. Developers have to consider creating PWAs alongside traditional android apps to ensure broad accessibility.

(2) Enhanced user experience: PWAs aimed to provide a smoother and more engaging user experience, which set higher expectations for android app developers. This encouraged them to focus on improving the quality and performance of their apps to compete effectively.

(3) progressive ~~En~~ Enhancement: Developers needed to adopt progressive enhancement strategies to ensure that android apps remained competitive by offering progressive and responsive user experiences, similar to PWAs.

- Impact on business in the mobile app industry:

(1) Cost Savings: Business could potentially save on development costs by investing in a single PWA that works across multiple platforms, including Android, rather than building separate native apps.

(2) Increased reach: PWAs enabled business to reach a wider audience, including users with android devices, without relying solely on app store distribution. This broader reach could lead to increased user acquisition,

GHANSHYAM

(3) Improved engagement: The focus on delivering app like experiences through PWAs encouraged business to prioritize user engagement and retention, ultimately benefiting their mobile strategy.

(4) Competition and Innovation: The rise of PWAs introduced competition, driving business to innovate their android apps to keep up with evolving user-expectations and technology trends.

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

→ In Android development, an 'Inflater' is a crucial component that is used to instantiate an 'Inflate' the layout XML files into their corresponding View objects. The structure and attributes of UI elements, into actual objects that can be manipulated programmatically.

– here's how it fits into the architecture of Android Layouts:

(1) Layout XML files: In Android, UI elements are defined in xml files within the 'res/layout' directory of an Android project. These XML files contain the structure of the user interface, specifying things like buttons, text fields, images, etc. as well as their attributes.

(2) Inflating Layouts: When an Android app runs the layout XML files need to be displayed on the screen. There are where the 'inflater' comes in, it's used to read the XML files and create the corresponding View objects.

(3) Inflating with Context: The 'Inflater' requires a 'context' object to perform the inflation about the current state of the application. This Context is typically provided by an Activity or fragment.

(4) Attaching to parent: The 'Inflater' also has the option to attach the inflated layout to a parent view, if specified. This allows the newly created View, to be automatically added to the specified Container within the parent View.

(5) Returning a View: Once the inflation process is completed, the 'inflater' returns the root view of the inflated layout, which can be used within the application.

6. (6) Manipulating Views Programmatically: After initiation, developers can programmatically interact with the views. Such as setting text, applying styles, adding event listeners and more.
(7) Displaying in the UI: Finally, the manipulated views can be added to the user interface using methods like (setContentView (for activities) or by adding them to the view hierarchy through their parent ~~the~~ ViewGroup.

3. Explain the Concept of a Custom Dialog Box in Android applications. provide examples to illustrate its use.

→ In Android Applications, a Custom Dialog Box is a pop-up window that overlays the current activity and is often used to interact with the user, gather input or display information. Overview of a Custom 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.
- Components: A Custom dialog typically consists of various UI elements like buttons, textviews, images or input fields, tailored to the specific interaction you want to facilitate.
- Customization: Developers can design the dialog's appearance, layout and behaviour according to their app's branding or specific requirements. This Customization allows for creativity in design and functionality.
- Simple example of creating and using a Custom dialog in Android.

GHANSHYAM

```
fun showCustomDialog(){
    val customDialog = Dialog(this)
    customDialog.setContentView(R.layout.custom_dialog_layout)
    val messageTextView = customDialog.findViewById<TextView>(R.id.msgTextView)
    val okButton = customDialog.findViewById<Button>(R.id.OKBtn)
    messageTextView.text = "This is a customdialog"
    okButton.setOnClickListener{
        customDialog.dismiss()
    }
    customDialog.show()
}
```

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. They can run even if the app's UI not visible.

3) Android manifest file:

- Role: The Android manifest 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:

```
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 create Notification(){
  val channelID = "my_channel"
  if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.O){
    val name = "my channel"
    val notificationManager.createNotificationChannel(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 is a crucial Component in 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 application. It serves several important purposes and its Content significantly impacts how the android System interacts and manage your app.

Significance of the Android Manifest file:

- App Configuration • Components Declaration • Permission • Intent Filters • App Lifecycle

-> example

```
<?xml version="1.0" encoding="utf-8" ?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.INTERNET" />
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="Assignment"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Assignment"
        tools:targetApi="31" />
```

```
<activity
  android:name=".MainActivity"
  android:exported="true">
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
  </intent-filter>
</activity>
<activity android:name="SecondActivity">
</activity>
</application>
</manifest>
```

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

→ Resources in Android development play a vital role in separating the presentation and content of an app from its code. They enhance maintainability, support various device configuration and improve overall development efficiency.

1. layout Resources

XML layouts These define the structure and appearance of your app UI elements such as button, textView and images etc.

example: creating a activity_main.xml layout file to design main screen of an app.

2. Drawable Resources

Drawable resources store image and icon used in your app. Different version provided for different Screen.

example: storing app icons, images for buttons or background graphics in drawable folders.

3. String Resources

Text and localization: String resources store text used in your apps UI making it easy to support multiple languages.

example: using string.xml to the store app lable like welcome.

4. Colour Resources

Color resources define the colors used thoughout your app, promoting Consistency and ensuring dynamic.

example: Defining primary and accent colors in color.xml for a Consistent color Scheme.

5. mipmap resources

mipmap resources store app icons, they are use to generate launcher icons at different resolutions.

example: placing app icons in mipmap folders for different Screen densities.

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

-> Background processing: services enable the execution of long running operation in the background such as downloading files monitoring sensors, or handling network request. This ensure that critical functionality can continue even if the user switches to another app or locks their devices.

Communication: Services can facilitate communication between different parts of an app or between different part of an app or between multiple apps. They provide a way for components like activities and broadcast receivers to send and receive data or instruction even when they are not actively visible to the user.

Multitasking: Services help in multitasking by allowing apps to perform tasks concurrently. For example music apps use services to play the app in their devices.

Notification: Services can create notification to keep user information about ongoing background activities, ensuring a seamless and informative user experience.

- Developing and android Service involves the following step.

Create a Service class: start by creating a class that extends the Service class or its subclass.

Practical No. 14

- Define life cycle method : override key lifecycle method such as onCreate(), onStartCommand() and onDestroy().
These method control when the service starts what it does and when it stops.

- If a Component starts the Service by calling startService() the service continues to run until it stops itself with stopSelf() or another Component stop it by calling stopService()

- Declaring a Service in the manifest
  - you must declare all service in your application's manifest file, just as you do for activities and other Components.

```
<manifest ...>
    <application ...>
        <service android:name=".ExampleService" />
    </application>
</manifest>
```

6-10-23