Patel Agrshi R. 21012021060 5B-5

MEHSANA DISTRICT EDUCATION FOUNDATION SANCHALIT

GANPAT UNIVERSITY U V PATEL COLLEGE OF ENGINEERING

U. V. Patel College of Engineering

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

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 & businesses in the mobile app industry.

→ As per my knowledge, one notable trend in the mobile app industry that was influencing the android platform was the rise of progressive Web Apps (PWAs). PWAs are web applications that offer 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 & devices, including Android. Developers had to consider creating PWAs alongside traditional Android apps to ensure broad accessibility.

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

3 Progressive Enhancement : Developers needed to adopt progressive enhancement strategies to ensure that android apps remained competitive by offering progressive & responsive user experiences, similar to PWAs.

→ Impact on Businesses in the Mobile App Industry :-

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

2) Increased Reach: PWAs enabled businesses 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.

3) Improved Engagement: The focus on delivering app-like experiences through PWAs encouraged businesses to prioritize user engagement & retention, ultimately benefiting their mobile strategy.

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

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

→ In Android development, an "Inflator" refers to the Layout Inflater, 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 & convert it into a corresponding view object in memory. Here's how the layout Inflater fits into the ~~architecture~~ of android layouts:

i) XML Layout files: In android, UI components are often ~~desig~~ defined using XML layout files. These files describe the structure & appearance of the UI, specifying things like widgets, their properties & their placement within the UI.

ii) 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 called as "layout inflation".

iii) Layout Inflater: It is responsible for reading the XML layout files & instantiating the corresponding view objects in memory. It takes the XML file as input, passes it, & creates the view objects, such as TextViews, Buttons etc. as specified in XML.

iv) 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 Recycler views to display list of items.

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

v) Binding Data: Once the view objects are created, they can be further customized & data can be bound to them. This is typically done using data binding techniques or programmatically setting properties & values

vi) Displaying UI: After inflation & customization, the view objects can be added to app's layout hierarchy & displayed on the screen.

3 Explain the concept of a custom DialogBox in Android applications. provide examples to illustrate its use.

⇒ In Android applications, a custom DialogBox is a pop-up window that overlays the current activity & 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) activities within a self contained, isolated UI element that temporarily interrupts.
- components: It 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, & behaviour according to their app's branding or specific requirements. This customization allows for creativity in design & functionality
- Example:

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

→ use cases of custom dialog Box: • Login • confirmation dialog • settings • Informational pop-up • Media playback controls

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

* Activities:

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

* 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 apps UI not visible.

* Android Manifest File:

• Role: The Android Manifest.xml file is like the app's blueprint. It declares the app's components & defines how they interact with the Android system & other components

Example: In Android Manifest.xml, you specify which activities are part of your app, their launch modes, permissions & service declarations. This file acts as a blue print for Android system to understand your app's structure & behaviour.

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

```
* 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")
    }
}
```

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

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

Significance of Android Manifest File:

- App configuration
- Component declaration
- Permissions
- Intent filters
- App life cycle.

Eg <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>
<uses-permission android: name = "android.permission.INTERNET" />
</application>
</manifest>

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

Patel Ayushi R 21012021060 5B-5

→ Resources play a fundamental role in android development by providing a structured way to manage assets, values, layouts & other elements used in your app. They help to create flexible, maintainable & device independent application. The various types of resources & their significance with example

1 Layout Resources:
- type: XML files in res/layout directory
- significance: define the structure & appearance of app's UI
- Example: 'activity_main.xml' defines the layout of your main activity, specifying UI components like buttons, textviews & their arrangement.

2. Drawable Resources:
- type: Images & drawable assets in res/drawable directory
- significance: Store graphics, icons, & images used in your app.
- Example: 'ic_launcher.png' is app's launcher icon.

3 String Resources:
- type: strings defined in XML files render res/values
- Significance: Store text strings, making it easier to provide translations & maintain consistency.
- Example: res/values/strings.xml contains string resource <string name = "app_name"> Quiz App </string>

4 colors 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.

5> Style Resources:
- type: styles defined in XML file under res/values
- Significance: defines reusable styles for UI components
- Example: res/values/style.xml defines style.

<style name="MyButtonStyle">
<item name="android:textColor">@color/primary_color</item>
</style>

6> Dimension Resources:
- type: Dimensions defined in XML files under res/values
- Significance: store dimension values, ensuring a consistent layout
- Example: res/values/dimens.xml defines dimension resources

<dimen name="margin_large">16dp</dimen>

7> Raw Resources:
- type: files stored in res/raw directory.
- significance: store non-XML files, such as JSON data, audio
- Example: store a JSON file for app configuration.

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

* Contributions of Android Services:

i) Background processing: Services allow apps to perform tasks in the background without blocking the user interface.

ii) Long-running operations: Services are ideal for handling operations that require more time to complete, such as playing music.

iii) Inter-component communication: Services enable components like activities, broadcast receivers & other services to communicate with eachother efficiently.

iv) Foreground services: Android services can run in foreground, even when the app isn't in the foreground. This is useful for features that require ongoing user interaction, like music playback process of developing an Android services :-

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

a Define the Service class : Create a new Java or Kotlin class that extends Service class - override methods like onCreate(), onDestroy, onStartCommand() to define the behaviour of your service.

b Configure Service in Manifest : Declare your service in Android Manifest.xml files to inform the android system about its existence & configuration.
<service android:name=".MyService" />

c. Start or Bind Service : Decide whether you want to start your service or bind it to the other components. use startService() or bindService().

d Implement Service Logic : In Service class, implement the specific logic your service needs to perform its task.

e Handle Lifecycle : Release resources when they're no longer needed & consider using stopService() or stopSelf()

f Interact with other components : Use appropriate mechanisms like intents, broadcasts or callbacks to facilitate communication.

g Foreground Service (optional) : If your service needs to run in the foreground, startForeground()

h Testing : Thoroughly test your service to ensure it functions as expected, including handling various scenarios like network failures.

i) optimization: optimize your service for performance & resource efficiency to minimize battery usage.

j) Error Handling & logging: Implement proper error handling & logging mechanisms to diagonize & address issue that may occur while service is running.