Meda Payal
En0:-22172012017
Batch: 5/D-3

Ganpat University | U.V. Patel College of Engineering

U. V. Patel College of Engineering

Ganpat Vidhyanagar, Mehsana-Gandhinagar Highway, Mehsana - 384012

Assignment-1

Based on your understanding identify a recent business trend has influenced the android platform. explain how this trend impacts Android app developers and business in the mobile app industry.

One of the significant business trends influencing the android platform was the rise of progressive web apps and the increasing focus on cross-platform development. while don't have access to real-time information, I can provide insights into how this trend was likely impacting android app developers and business in the mobile app industry up to that point.

Progressive web apps are web applications that offer a mobile app like experience but are built using web technologies HTML, CSS and JS, These PWAs can be accessed through web browsers but can also be added to user's home screen, marketing.

1. cost-efficiency
2. Broader Reach
3. Improved user-experience

4. Reduce App store dependencies
5. Easier maintence

Q.2 What is the purpose of the Inflator la
in Android development and how does it f
into the architecture of Android layout?

→ In Android development an Inflater is a
component used to create view object fro
xml layout resource file. It plays a signi
cant role in the process of converting xml
definition into actual view objects that c
displayed on the screen. Herels breakdown
its purpose and how it fits into the arch
re of Android layouts.

* Purpose of a layout Inflate.

1. Dynamic UI creation: The primary purpose
of a layout Inflater is to dynamically
create view objects at runtime based on th
layout xml files you define in your app
resources. This allows you to create and
fy the user interface of your android a
tion programitically.

2. Resource separation: It helps separate the
entation from the logic (Java / Kotlin co
you can define the structure and appe
of your UI in xml files, making it e
to maintain and change the UI without
modifying the code.

3. Reuse and modularity: Layout Inflation facilities the reuse of UI in xml file and reuse it in multiple activities, fragments, or custom views.

* How LayoutInflater fits into the Architecture.

1: xml. layout files: In Android, you define the structure and appearance of your UI components using xml layout files (e.g. activity_main.xml) This files contain xml tags repre-senting various UI elements like buttons, text, views, and layouts.

2.: layout resources: These xml layout files are placed in the 'res/layout' directory of your Android project as resources.

3: Layout Inflation: When your app runs, and you want to display a UI on the screen you use the LayoutInflater's to inflate, (create) the view objects described in your xml layout files, you typically obtain a reference to the 'LayoutInflater' from the 'context'.

LayoutInflater inflater = getLayoutInflate

();
View rootView = inflater.inflate(R.Layout.ca
_main, null);

5. Display: Finally, you can add the inflate
view to your app's UI hierarchy, usually
by setting it as the content view of the
'Activity', adding it to a 'Fragment' or a
ng into a custome 'ViewGroup'.

Q.3 Explain the concept of a custome Dialog
in Android applications. Provide example
to illustrate its use.

→ In android application, a custome dialog is
a user Interface component that is used
display information, prompt the user for in
or perform specific actions within a po
window. unlike standard dialogs provided
android which have predefined layouts
functionality. Customized user Interface t
to their app's requirements.

*concepts of custom dialogbox.

1. Custom UI: custom dialogbox enable developer
design a dialog's appearance and content
ng to their specific needs. This includes
ng the layout, adding widgets, and sty
the dialog to match the app's design.

2. Programmatic control: Developers have pr
programatic control over customes dialogs a

tion to define the behaviour and interaction of elements within the dialog.

- Example of DialogBox :

```
<LinearLayout
  xmlns:android:id="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_Height="wrap_content"
  android:orientation="vertical" >

<TextView
  android:id="@+id/dialog_message"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:text="This is a custome" />

<Button
  android:id="@+id/dialog"
  android:layout_width="wrap_content"
  android:layout_Height="wrap_content"
  android:text="close" />

kotlin.kt

import android.app.Dialog
import android.content.Context
import android.os.Bundle
```

```
import android.view.View
class customDialog(context: context): DialogCo
{
override fun onCreate(savedInstanceState: Bundle
{
 super.onCreate(savedInstanceState: Bundle?
 setContentView(R.layout.customne_dialog)
 setTitle("customeDialog")
}
fun onclose Button clicked(view: View){
 dismiss()
 }
}
```

Q.4 How do activities, services and the android manifest file work together to provide a bu example of how they cooperate to design a app.

→ In an Android app, Activities, services, and the Android manifest file together to Provide core structure of and functionality of the a Each component plays a specific role in defin the user interface managing background t and configuring the apps behaviour.

1. Activities:
Role: Activities represents the UI compone an Android app. Each screen or user In is typically implemented as an activity

2. Services: Role; services are components run in the background and perform

- running tasks or handle background operations independently of the UI.

Example: Suppose your app needs to download data from a server periodically. You could create a service to handle this background task.

3: Android manifest file:

Role: The Android manifest file is a configuration file that provides essential information about the app to the Android system. It defines the app's components, permissions and other settings.

example: In the Androidmanifest.xml file you specify the activities and service, define permission, and configure various app-related settings.

How does the Android manifest file impact the development of an android app? Provide an example to demonstrate its significance.

→ The Android manifest file is a crucial component of android app development as it serval important purposes that impact the development and behaviour of an Android application.

Here's an explanation of how the Android file impacts app development and can be illustrate its significance.

2. permission: you specify the permissions your app requires in the manifest file. Andro enforces these permission to ensure that app operates securely and that sensitive rces are uploaded.

3. Intent filters: you can intent filters in manifest in the to specify how your app nds to implicit intents. This is how o apps or system components can interac your app. Intent filters specify which ac can handle certain types of actions, su opening a particular file type or res to a specific URL.

4. App configuration: you can specify vari settings and configurations for your a in the manifest file, such as the app's i , lable, theme and supported screen orie

5. App entry point: you specify the main activi the manifest file This is activity that android system launches when the user op your app. It serves as the entry point app.

* manifest declarations.

In your Android manifest.xml, you decl the two activities and any required pe Here's simplified example:

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.mynotesapp" />

<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

<application
android:icon="@mipmap/ic_launcher"
android:label="@string/app_name" />

<activity
android:name="mainActivity"
android:label="my notes" />

<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>

<activity android:name=".EditNote" />
</application>
</manifest>

Permission: The manifest also declares the 'WRITE_EXTERNAL_STORAGE' permission, indicating that the app needs permission

to write to external storage to save notes.

* Intent Filters:

If your app needs to responds specific actions like opening a node file, you can declare in Filters the relevant activities in the manifest file.

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 play a crucial role in creating well-structured applications. They are essential for separating various aspects of an android app. such as layout, strings, images, and more, from the source code. This separation enhances code maintainability, localization, and overall development efficiency. Here's an overview of the role of resources in Android development, along with various types and examples.

* Roles of Resources:

1. Separation of concerns: Resources separate different aspects of an app. making it easier to manage and maintain each aspect independently. This separation follows the principle of separation of concerns.

2. Localization and internalization: Resources facilitate localization by allowing developers to provide alternate versions developers to provide alternate different languages and regions. This is crucial for creating a globally accessible app.

3. Adaptations to various device configurations: Android devices come in various sizes, resolutions, and screen densities. Resources enable the adaptation of layouts, image and other assets to different device configurations, ensuring a consistent user experience.

4. Code Reusability: By using resources, you can reuse layouts, styles and assets across multiple parts of your app, reducing redundancy in your codebase.

1. Layout Resources:

significance: Layout resources define the structure and appearance of UI elements in xml format, separating the UI design from the code.

Example: consider a layout xml for a login screen.

Linear Layout.

xmlns: android="http://schemas.android.com
apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical">

<EditText
android:id="@+id/e1"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:hint="@string/username"/>

<EditText
android:id="@+id/e2"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Click"/>

</LinearLayout>

2. String Resources:

- significance: string resources store text that is displayed in the UI or used in the app. This enable easy translation and localization of the app.

Example: In "res/values/string.xml"

<string name="username_hint">username</string>

<string name = "login_button_table">Login
</string>

3) Drawable resources:
significance, string resources store text that is displayed in the UI or used in the app code. This enables easy translation and localization of the app.

Example: storing an image in /res/drawable/res/drawable/ic_app_icon.png

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

An Android service plays a crucial role in enhancing the functionality and capabilities of a mobile application. It allows you to perform background tasks and long-running operations, ensuring that your app remains responsive and efficient. Here's how an android service contributes to the functionality of a mobile applications.

Developing an Android service.
1) Create a service class:

start by creating a Java or Kotlin class
extends the 'Services' class. This class will con
the logic for your service you can override m
like oncreate(), onstart(), and ondestroy()
manage the services lifecycle

```
class myservice : service(){
    override fun oncreate(){}
override fun onstartcommand(intent.Inte
    flags: Int, startId: Int){
return service.START_STICKY
}
override fun ondestroy(){
    //cleanup and release resources
}
}
```

2. Declare the service in the manifest:
In your android manifest.xml, declare
service to make it known to the Android s
you specify the service's name and any re
permissions.

```
<service android:name=".myservice" />
```

3. Foreground service = If your service should
the foreground, create a notification
and use 'startForeground()' to place t
in the foreground state.