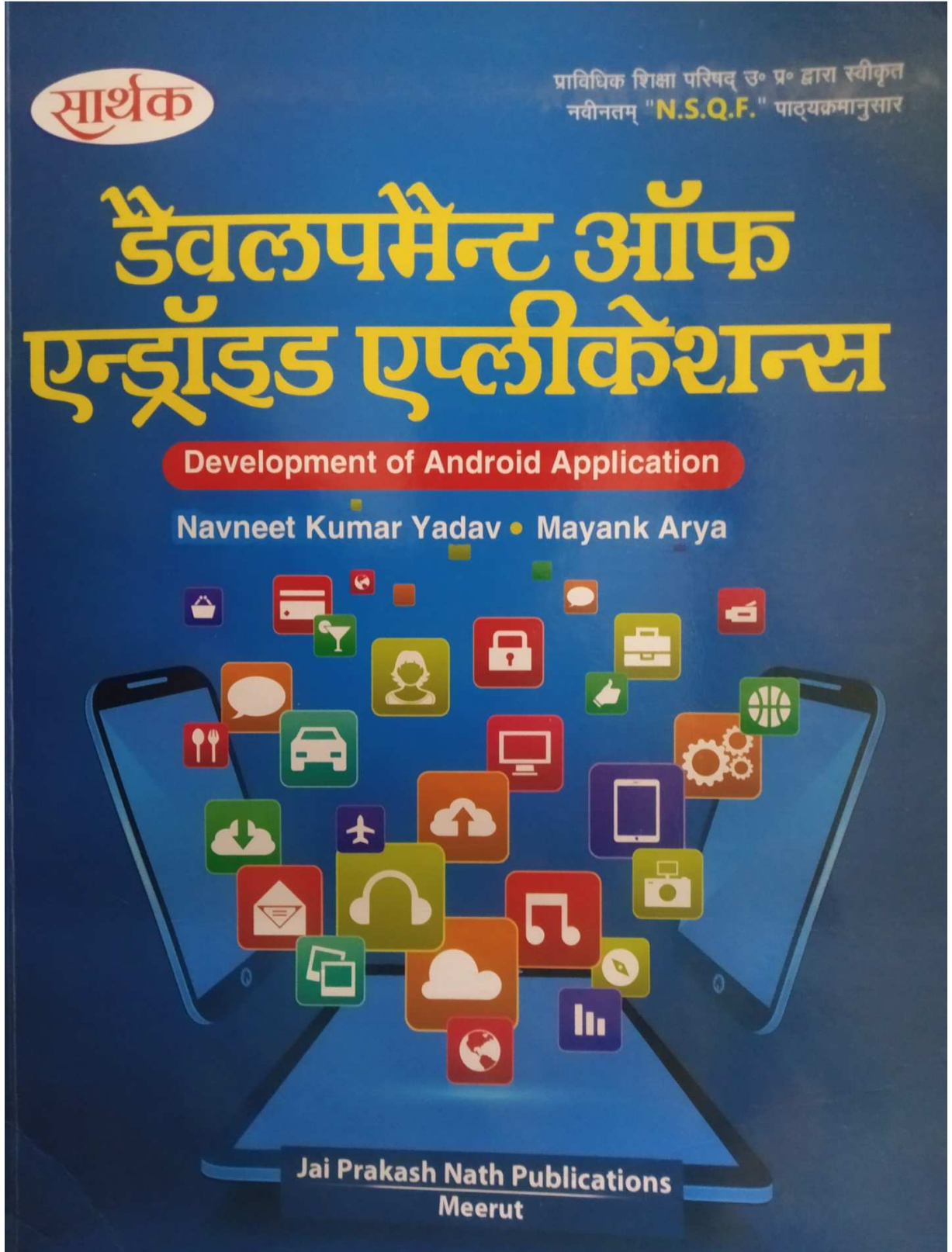




AVNISH MISHRA

SYLLABUS

DEVELOPMENT OF ANDROID APPLICATIONS

RATIONALE

Knowing the details of Mobile and their working principle are need of the every common man. Mobile Application development is the very hot business domain. Majority of the corporate have separate division for the development of mobile applications. It is essential that diploma student must know the way to apply advanced data communicating methods and networking protocols for wireless and mobile devices. Hence this subject.

LEARNING OUTCOMES

After undergoing this subject, the students will be able to :

- ❖ illustrate the usage of different components of Android OS in detail
- ❖ develop a mobile application using different components of Android
- ❖ choose appropriate controls to design the GUI to meet desired needs
- ❖ consume JSON data and call web services from Android mobile app.
- ❖ write a program in Android to store data in databases
- ❖ develop Mobile applications using Android

DETAILED CONTENTS

1. Introduction to ANDROID

(10 periods)

What is Android? Dalvik Virtual Machine & .apk file extension, Fundamentals: Basic Building blocks -Activities, Services, Broadcast Receivers & Content providers, UI Components - Views & notifications, Components for communication -Intents & Intent Filters, Android API levels (versions & version names)

2. Environment Setup and Basic Project Structure

(08 periods)

Setting up development environment Android, Manifest.xml, Gradle, Uses-permission & uses-sdk, Resources & R.java, Assets, Layouts & Drawable Resources, First sample Application, Launching emulator, Editing emulator settings, Emulator shortcuts, Logcat usage, Introduction to DDMS, Hello World App, Creating your first project The manifest file Layout resource, Running your app on Emulator, Debugging the Android App.

3. Android Fundamentals and User Interface Design*(12 periods)*

Activities and Activity lifecycle, Permission System

Basic UI Components : Text View, Button, Radio Button , Edit Text, Image View for image, Check Box , Progress Bar, Event Handling in Android

Layouts : Liner Layout, Relative Layout, Frame Layout, Coordinate Layout, [dip, dp, sip, sp] versus px

Intents : Intents introduction and importance, Types of Intents (Explicit Intents, Implicit intents)

4. Menus and Preferences*(10 periods)*

Introduction to Menus, Types of Menus (Option menu, Context menu), Uses of Shared Preferences

5. Advanced UI Components*(12 periods)*

Time and Date, List View, Grid View, Card View, recycler view Adaptors (Base Adaptor, Array Adaptor) & View Holder, Dialogs, Toast, Popup, Fragments, Material Design(Introduction , Navigation, Floating Button , Tool bar).

6. Threads in Android*(12 periods)*

Threads running on UI thread (run on UI Thread), Worker thread, Handlers & Runnable, AsyncTask , calling web services and consuming JSON data from Web Services.

7. Notifications & Services*(10 periods)*

Broadcast Receivers (Introduction, different ways to register a broadcast receiver), Introduction to Notification, Overview & Types of services, implementing a Service, Service lifecycle

8. Storage and Content Provider*(10 periods)*

Supported Storage in Android (Internal memory, External memory, Shared Preferences and network), SQLite introduction, CRUD Operations in SQLite database (cursor, content values etc) , Basics of Content Provider

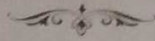
LIST OF PRACTICALS

1. Install the Android Studio and Setup the Development Environment
2. Write a program to demonstrate activity (Application Life Cycle)
3. Write a program to demonstrate different types of layouts

4. Write a program to implement simple calculator using text view, edit view, option button and button
5. Write a program to develop app having multiple activities and user should be able switch between the activities by using intents.
6. Write a program to demonstrate list view
7. Write a program to demonstrate photo gallery
8. Write a program to demonstrate Date picker and time picker
9. Develop an simple application with context menu and option menu.
10. Write a program to demonstrate the functionality of Shared Preferences.
11. Develop a sample Android application having navigation items similar to Gmail Application.
12. Write a program to demonstrate a service
13. Write a program to demonstrate the application of intent class
14. Write a program to create a text file in a external memory
15. Write a program to store and fetch data from SQL life database.

विषय-सूची

क्रमांक	अध्याय का नाम	पृष्ठ
1.	एन्ड्रॉइड का परिचय (Introduction to ANDROID)	1-30
2.	वातावरण सेटअप तथा बुनियादी परियोजना संरचना (Environment Setup and Basic Project Structure)	31-63
3.	एन्ड्रॉइड फंडामेंटस और यूजर इंटरफेस डिजाइन (Android Fundamentals and User Interface Design)	64-85
4.	मेन्यू और प्राथमिकताएँ (Menus and Preferences)	86-96
5.	अग्रिम UI घटक (Advanced UI Components)	97-130
6.	एन्ड्रॉइड में थ्रेड्स (Threads in Android)	131-152
7.	सूचनाएँ और सेवाएँ (Notifications & Services)	153-164
8.	भंडारण और सामग्री प्रदाता (Storage and Content Provider)	165-192
9.	प्रयोगात्मक कार्य (Practical Work)	193-254





एन्ड्रॉइड का परिचय (Introduction to ANDROID)

What is Android? Dalvik virtual machine & .apk file extension, fundamentals : basic building blocks activities, services, broadcast receivers & content providers, UI components - views & notifications, components for communication - intents & intent filters, android API levels (versions & version names).

1.1. परिचय

आज की दुनिया में एन्ड्रॉइड डिवाइस मोबाइल फोन उपयोगकर्ताओं (users) की जिंदगी का एक बहुत ही आवश्यक (essential) हिस्सा है। आज के दौर में स्मार्टफोन में android सबसे अधिक उपयोग किया जाने वाला OS है।

Android क्या है?

एन्ड्रॉइड एक ऑपरेटिंग सिस्टम और प्रोग्रामिंग प्लेटफॉर्म है जो Google द्वारा मोबाइल फोन और अन्य मोबाइल उपकरणों जैसे टैबलेट, कम्प्यूटर और स्मार्टफोन के लिए विकसित किया गया है।

यह google द्वारा और बाद में OHA (ओपन हैंडसेट एलायंस) विकसित किया गया है। एन्ड्रॉइड के कई नाम हैं जैसे लॉलीपॉप, किटकैट, जेली बीन, आइसक्रीम सैंडविच, फ्रियो, इकलियार, डोनट आदि।

एन्ड्रॉइड एक ओपन सोर्स और लिनक्स (linux) आधारित ऑपरेटिंग सिस्टम है। यह कई अलग-अलग manufacturers के कई अलग-अलग उपकरणों पर उपयोग किया जाता है। एन्ड्रॉइड में एक सॉफ्टवेयर डेवलपमेंट किट (SDK) शामिल है जो आपको मूल कोड लिखने में एन्ड्रॉइड, मोबाइल उपकरणों के लिए एप्लिकेशन बनाने के लिए सॉफ्टवेयर मॉड्यूल इकट्ठा करता है। एन्ड्रॉइड, मोबाइल उपकरणों के लिए एप्लिकेशन development के लिए एक unified approach प्रदान करता है, जिसका अर्थ है कि डेवलपर्स केवल एन्ड्रॉइड के लिए app विकसित करते हैं और उनके applications को एन्ड्रॉइड द्वारा संचालित विभिन्न उपकरणों पर चलाने में सक्षम भी होते हैं। एन्ड्रॉइड, एप्लिकेशन वितरित करने के लिए एक बाजार प्रदान करता है। एन्ड्रॉइड, मोबाइल एप्लिकेशन के लिए एक ecosystem का प्रतिनिधित्व करता है।



चित्र 1.1

2 डेवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

एन्ड्रॉइड के लिए सोर्स कोड free and open सोर्स सॉफ्टवेयर लाइसेंस के तहत उपलब्ध है। Google, Apache license version 2.0 और बाकी, लिनक्स कर्नेल परिवर्तन के तहत अधिकांश कोड को GNU General Public License version 2 के तहत प्रकाशित करता है।

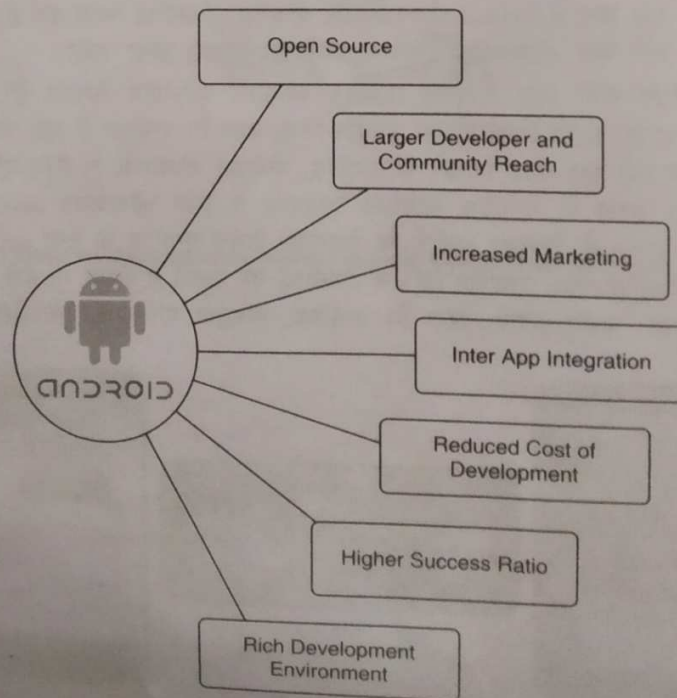
ओपन हैंडसेट एलायंस (OHA) क्या है?

यह 84 कंपनियों जैसे google, samsung, AKM, synaptics, KDDI, Garmin, Teleca, Ebay, Intel आदि का संघ है। यह 5 नवम्बर, 2007 को google के नेतृत्व में स्थापित किया गया था। यह एन्ड्रॉइड प्लेटफॉर्म का उपयोग करके open standards को आगे बढ़ाने, सेवाएँ प्रदान करने और हैंडसेट को तैनात करने के लिए प्रतिबद्ध है।

Android क्यों?

Android एक मोबाइल एप्लीकेशन डेवलपमेंट प्लेटफॉर्म है और इसके अन्य प्वाइंट दिए गए फिगर में प्रदर्शित किये गए हैं—

- Open source
- Larger developer and community reach
- Increased marketing
- Inter app integration
- Reduced cost of development
- Higher success ratio
- Rich development environment



चित्र 1.2

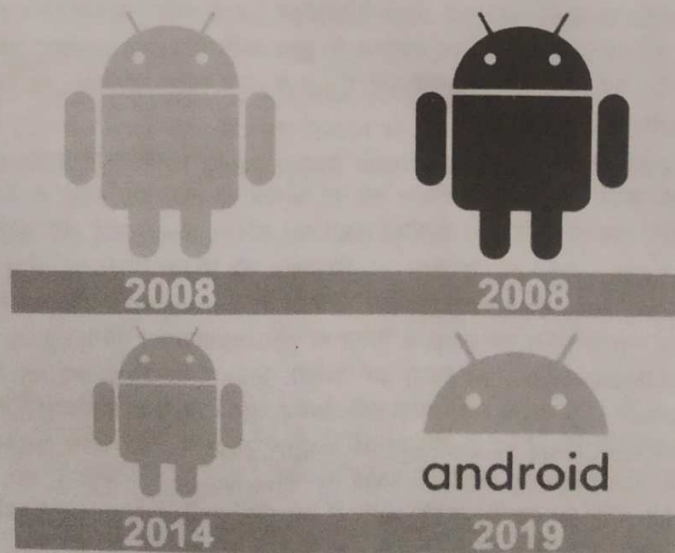
1.2. Android का इतिहास

एन्ड्रॉइड का इतिहास अक्टूबर 2003 में शुरू होता है, इससे पहले कि स्मार्टफोन शब्द का व्यापक रूप से उपयोग किया जाता था और कई साल पहले एप्पल (apple) ने अपने पहले आईफोन (iPhone) और आईओएस (ios) की घोषणा की थी। Android inc. की स्थापना कैलिफोर्निया के पालो अल्टो में हुई थी।

इसके चार संस्थापक रिच माइनर, निक सियर्स, क्रिस व्हाइट और एंडी रुबिन थे। उस समय, रुबिन को यह कहते हुए उद्धृत (quoted) किया गया था कि android inc. “ऐसे स्मार्ट मोबाइल उपकरणों को विकसित करने जा रहा है जो अपने स्वयं के स्थान और वरीयताओं के बारे में अधिक जानते हैं।” हालांकि, बाद में कंपनी को धन की अपर्याप्तता का सामना करना पड़ा जिसने google को picture में लाया। Google 2005 में android का अधिग्रहण करने के लिए \$50 मिलियन का सौदा किया। सभी चार co-founders जल्द ही googleplex में चले गए ताकि अपने नए मालिकों के तहत ओएस को और अधिक विकसित कर सकें।

The Android Logo

एन्ड्रॉइड ओएस के लिए अब परिचित लोगों, जो एक रोबोट और एक हरे रंग की बग के संयोजन की तरह दिखाता है, इसे इरिना ब्लोक ने बनाया था जबकि वह google द्वारा नियोजित किया गया था। एक चीज जो ब्लोक और google ने तय की थी, वह एन्ड्रॉइड रोबोट को एक ओपन सोर्स प्रोजेक्ट बनाना था। लगभग हर दूसरी huge कंपनी ऐसे logo or mascot को फिर से डिजाइन करने और दूसरों द्वारा उपयोग किए जाने से बचाएगी। हालांकि, एन्ड्रॉइड रोबोट को अब संशोधित किया गया है और उन लोगों द्वारा उपयोग किया जाता है, सभी क्योंकि google क्रिएटिव कॉमन्स 3.0 एट्रीब्यूशन लाइसेंस के तहत इस तरह के दबलाव की अनुमति देता है।



चित्र 1.3

पहले सार्वजनिक एन्ड्रॉइड बीटा वर्जन 1.0 को अंततः 5 नवम्बर 2007 को प्रकाशित किया गया था।

4 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

सभी प्रमुख एन्ड्रॉइड OS संस्करण (Major OS versions)

1. Android 1.0 (API 1) और 1.1 (API 2)—Android 1.0 (API 1) को 23 सितम्बर 2008 को लॉन्च किया गया था। इसे HTC ड्रीम स्मार्टफोन (अमेरिका में उर्फ टी मोबाइल G1) में शामिल किया गया था। इस प्रकार यह पहला एन्ड्रॉइड डिवाइस बन गया। इसके द्वारा दी जाने वाली सुविधाओं में Google मैप्स, YouTube, एक HTML ब्राउज़र, जीमेल, कैमरा, ब्लूटूथ, वाई-फाई और बहुत कुछ शामिल हैं।

उस समय की unique feature एक android market (अब play store) की उपस्थिति थी, जहाँ से users उन एन्ड्रॉइड एप्लिकेशन को डाउनलोड और अपडेट कर सकते थे जो पहले से इंस्टॉल किए गए थे। कुछ महीनों बाद, फरवरी 2009 में, google ने एचटीसी ड्रीम के लिए एन्ड्रॉइड 1.1 (एपीआई 2) अपडेट जारी किया। इस बार इसका उद्देश्य इसे ज्यादा अधिक मजबूत और उपयोगकर्ता के अनुकूल बनाना था।

2. Android 1.5 Cupcake—यह version अप्रैल 2009 के अन्त में आया और Google की dessert-themed वाले naming scheme के लिए सबसे पहले था और इसे सैमसंग गैलेक्सी फोन श्रृंखला में शामिल किया गया था। इसे बहुत-सी कार्यक्षमताओं के साथ पेश किया गया था, जिन्हें आज भी हम स्वीकार करते हैं।

Cupcake में पहले दो versions की तुलना में काफी कुछ नई सुविधाओं और सुधारों को जोड़ा गया। इसमें वे चीजे शामिल हैं जो अब भी use करते हैं, जैसे कि YouTube पर वीडियो अपलोड करने की क्षमता, फोन के डिस्प्ले को automatically घुमाने का तरीका और third-party के कीबोर्ड के लिए support।

3. Android 1.6 (API 4) aka Donut—जल्द ही Google से सितम्बर 2009 में एन्ड्रॉइड 1.6 डोनट को लॉन्च किया। नए ओएस ने अब सीडीएमए आधारित नेटवर्क offered support किया। इससे एन्ड्रॉइड फोन को दुनिया भर की सभी जगह बेचा जा सकता है। अन्य विशेषताओं में quick सर्च बॉक्स की शुरुआत और कैमरा, कैमकार्डर और गैलरी के बीच quick toggling करना शामिल था। डोनट में Wi-Fi, Bluetooth, GPS आदि के प्रबंधन के लिए power control widget भी पेश किया।

4. Android 2.0-2.1 Eclair—अक्टूबर 2009 में एन्ड्रॉइड 1.0 के लॉन्च के लगभग एक साल बाद Google ने ओएस के संस्करण 2.0 को आधिकारिक कोडनेम एक्लेयर के साथ जारी किया। यह संस्करण टेक्स्ट-टू-स्पीच स्पॉट को जोड़ने वाला पहला था और कई अन्य नई सुविधाओं और सुधारों के बीच लाइव वॉलपेपर, मल्टीपल अकाउंट स्पॉट और Google मैप्स नेविगेशन भी पेश किया।

5. Android 2.2 Froyo—एन्ड्रॉइड 2.2 Froyo ("frozen yogurt") आधिकारिक तौर पर मई 2010 में लॉन्च किया गया था। Froyo को स्पॉट करने वाले स्मार्टफोन कई नए फीचर्स का लाभ उठा सकते हैं, जिनमें वाई-फाई मोबाइल हॉटस्पॉट फंक्शंस, एन्ड्रॉइड क्लाउट टू डिवाइस मैसेजिंग (D2DM) सर्विस, flash स्पॉट और बहुत कुछ शामिल हैं।

6. Android 2.3 Gingerbread—एन्ड्रॉइड 2.3 जिंजरब्रेड को सितंबर 2010 में लॉन्च किया गया था। ओएस जिंजरब्रेड के तहत user interface refresh करता था। इस संस्करण में, user interface डिजाइन को अधिक सरल बनाया गया था। यह extra-large screen sizes को support करता था और resolutions को integrate किया गया था।

7. Android 3.0 Honeycomb—इसे पहली बार फरवरी, 2011 में पेश किया गया था। ओएस का यह संस्करण शायद oddball of the bunch है। हनीकॉम्ब बड़े डिस्प्ले वाले टैबलेट और अन्य मोबाइल उपकरणों के लिए बनाया गया था।

8. Android 4.0 Ice Cream Sandwich—इसे अक्टूबर 2011 में जारी किया गया, एंड्रॉइड के आइसक्रीम सैंडविच संस्करण ने कई नए फीचर लाए। इसमें होम स्क्रीन पर "favorites tray" शामिल है, साथ ही owner's face की तस्वीर लेने के लिए अपने कैमरे का उपयोग करके फोन को अनलॉक करने के feature भी शामिल हैं। आईसीएस के साथ अन्य बदलावों में सभी ऑन-स्क्रीन बटन, notifications और ब्राउज़र टैब को dismiss करने के लिए swipe gestures और मोबाइल और वाई-फाई पर आपके डेटा उपयोग की monitor करने की क्षमता शामिल है।

9. Android 4.1-4.3 Jelly Bean—एन्ड्रॉइड 4.1 के रिलीज के साथ जून 2012 में एन्ड्रॉइड का Jelly Bean Era शुरू हुआ। Google ने क्रमशः जेली बीन लेबल के तहत क्रमशः अक्टूबर 2012 और जुलाई 2013 में 4.2 और 4.3 दोनों संस्करण जारी किए।

10. Android 4.4 Kitkat—एन्ड्रॉइड 4.4 किटकैट का शुभारंभ 2013 में हुआ था और इसमें कई विशिष्ट विशेषताएँ थीं। 'Ok Google' command, का प्रयोग कर एक उपयोगकर्ता किसी भी समय Google तक पहुँच जाता है और 512 MB की न्यूनतम रैम मेमोरी वाले फोन पर काम कर सकता है।

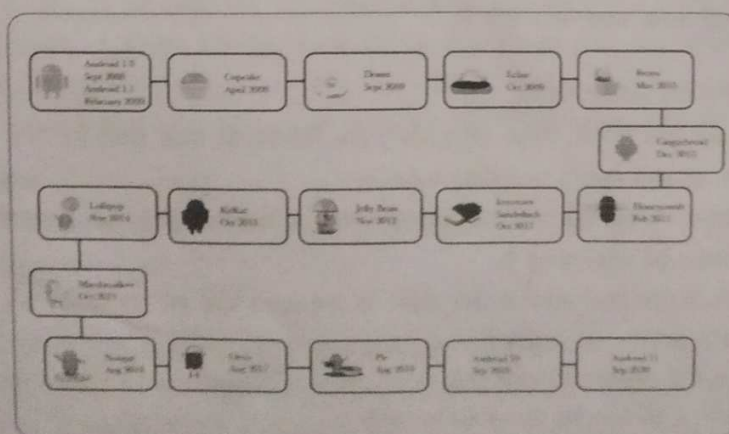
11. Android 5.0 Lollipop—पहली बार 2014 के last में लॉन्च किया गया, एन्ड्रॉइड 5.0 लॉलीपॉप ऑपरेटिंग सिस्टम के overall look में एक major shakeup था। एन्ड्रॉइड 5.1 अपडेट ने कुछ अंडर-हुड परिवर्तन किए। इसमें दुअल सिम, एचडी वॉयस कॉल और डिवाइस प्रोटेक्शन के लिए official support शामिल था, ताकि factory reset के बाद भी चोर आपके फोन से बाहर रहे।

12. Android 6.0 Marshmallow—Android मार्शमैलो अक्टूबर, 2015 में लॉन्च किया गया था और इसने बायोमेट्रिक फिंगरप्रिंट अनलॉकिंग और USB टाइप C पोर्ट जैसे उल्लेखनीय फीचर्स लाए थे, Doze मोड की शुरुआत जिसने CPU की गति को कम कर दिया, जबकि display remains turned off रहा जिससे बैटरी लाइफ बढ़ी। सर्च बार के द्वारा आसानी से applications को एक्सेस किया। एन्ड्रॉइड पे, मेमोरी मैनेजर की शुरुआत, ऐप्स के भीतर कोवर्ड से contextual search, डिवाइस, मीडिया के लिए वॉल्यूम सेट करना जैसे फीचर ऐड किये गए।

13. Android 7.0 Nougat—अगस्त 2016 में, Google ने एन्ड्रॉइड 7.0 Nougat जारी किया। इसने विशेष रूप से बड़ी स्क्रीन वाले उपकरणों के लिए बेहतर multitasking फीचर प्रस्तुत किए, उदाहरण के लिए, split-screen mode को अनुप्रयोगों के साथ fast switching करने के प्रावधान के साथ पेश किया गया था। इसने ऐप्स को गति देने के लिए एक नए JIT कंपाइलर पर स्विच किया, तेजी से 3D रेंडरिंग के लिए Vulkan API को पोर्ट किया और OEM को इसके अब डिफेक्ट विड्रॉल वीआर प्लेटफॉर्म को पोर्ट करने में सक्षम किया।

14. Android 8.0 Oreo—यह संस्करण अगस्त 2017 में दिखाई दिया। Google द्वारा दावे के अनुसार, pixel उपकरणों पर परीक्षण किए जाने पर, यह नौगट की तुलना में 2x बृद्ध गति वाला अधिक शक्तिशाली और तेज संस्करण है। यह संस्करण और भी स्मार्ट था जो ऑटोफिल, पिक्चर इन पिक्चर मोड और notification डॉट्स जैसे functionalities की शुरुआत के माध्यम से स्पष्ट था, जिसके माध्यम से उपयोगकर्ता जल्दी से नई जानकारी पकड़ सकता है।

15. Android 9.0 Pie—Android 9 को अगस्त 2018 में पेश किया गया था। इसने visual aspect में जबरदस्त सुधार लाया और कृत्रिम बुद्धिमत्ता की शक्ति का असाधारण उपयोग किया। एन्ड्रॉइड 9.0 पाई में आपके स्मार्टफोन की battery life को extend करने में मदद करने के लिए डिजाइन की गई कुछ नई सुविधाएँ भी शामिल थीं। पाई में shush भी है, जो automatically आपके फोन को do not disturb मोड में डालता है जब आप अपने फोन की स्क्रीन को flat surface पर नीचे flip करते हैं।



चित्र 1.4

6 डेवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

16. Android 10—सितम्बर 2019 में एन्ड्रॉइड 10 के साथ, Google ने ऑपरेटिंग सिस्टम को rebrand करने की घोषणा की, जो पहले के संस्करणों के लिए उपयोग की जा रही sweets-name आधारित नामकरण योजना को समाप्त कर रहा था। इस संस्करण के साथ एक new logo और different colour scheme की घोषणा की गई थी। यह upcoming फोल्डेबल फोन का support करता है।

17. Android Version 11 (API 30)—एन्ड्रॉइड 11 को हाल ही में 8 सितम्बर 2020 को जारी किया गया था। यह संस्करण एक टैगलाइन के साथ आया है “The OS that gets to what’s important and it’s pretty much justified”. एन्ड्रॉइड 11 बहुत सारे नए फीचर्स के साथ आया है। इसमें एक नया conversations notification श्रेणी शामिल है जहाँ विभिन्न ऐप्स के आपके सभी चैट एक ही स्थान पर एकत्र किए गए हैं।

1.3. Android की विशेषताएँ

एन्ड्रॉइड, ऐपल के साथ प्रतिस्पर्धा करने वाला एक शक्तिशाली ऑपरेटिंग सिस्टम है और शानदार सुविधाओं का समर्थन करता है। उनमें से कुछ नीचे सूचीबद्ध हैं—

- Beautiful UI—एन्ड्रॉइड ओएस बेसिक स्क्रीन एक सुंदर और सहज उपयोगकर्ता इंटरफेस प्रदान करता है।
- Connectivity—सभी GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, ब्लूटूथ, वाई-फाई, LTE, NFC और WiMAX सभी को सपोर्ट करता है।
- Storage—SQLite, एक हल्के रिलेशनल डेटाबेस का उपयोग डेटा भण्डारण उद्देश्यों के लिए किया जाता है।
- Multi-tasking—उपयोगकर्ता एक कार्य से दूसरे पर जा सकता है और उसी समय विभिन्न application एक साथ चला सकता है।
- Multi-touch—एन्ड्रॉइड में मल्टी-टच के लिए, को सपोर्ट करता है।
- Multi-language—single direction and bi-directional text सपोर्ट करता है।
- Resizable widgets—widget resizable हैं, इसलिए उपयोगकर्ता more content दिखाने के लिए उनका expand कर सकते हैं या उन्हें space save के लिए shrink कर सकते हैं।
- Wi-Fi डायरेक्ट—एक ऐसी तकनीक जो उच्च बैंडविड्थ वाले पीयर-टू-पीयर कनेक्शन पर ऐप्स को सीधे खोजती है और जोड़ी बनाती है।
- Android बीम—एक लोकप्रिय NFC आधारित तकनीक जो उपयोगकर्ताओं को तुरन्त दो NFC सक्षम फोन को एक साथ स्पर्श करके साझा करने देती है।

Android के लाभ—

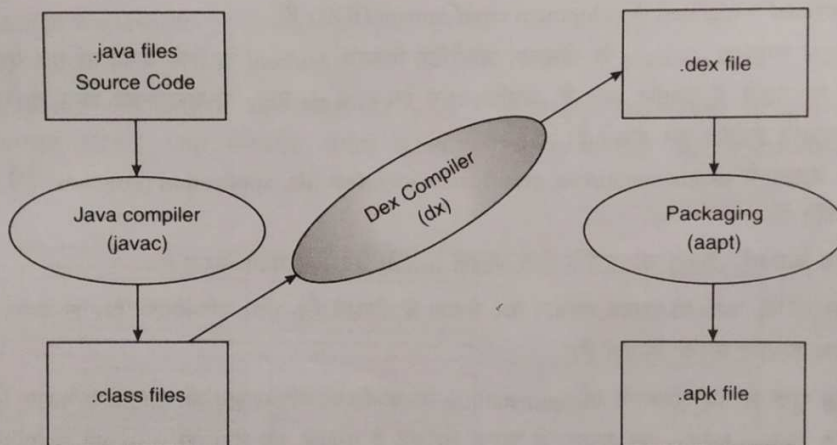
- Android ऐप्स तक आसान पहुँच।
- आप बैटरी और मास स्टोरेज, डिस्क ड्राइव और UDB विकल्प को बदल सकते हैं।
- यह Google की सभी सेवाओं का समर्थन करता है।
- ऑपरेटिंग सिस्टम आपको एक नए एसएमएस और ईमेल या नवीनतम अपडेट की सूचना देने में सक्षम है।
- यह मल्टीटास्किंग को सपोर्ट करता है।
- एन्ड्रॉइड फोन इंटरनेट साझा करने के लिए राउटर के रूप में भी कार्य कर सकता है।
- यह अनुकूलित करने के लिए स्वतंत्र है।
- एक संशोधित रॉम स्थापित कर सकते हैं।
- यह 2 डी और 3 डी ग्राफिक्स का समर्थन करता है।

1.4. Dalvik Virtual Machine (DVM)

Dalvik आइसलैंड में एक शहर का एक नाम है। Dalvik VM डैन बॉर्नस्टीन ने डिजाइन और लिखा था।

Dalvik वर्चुअल मशीन (DVM) मोबाइल उपकरणों के लिए अनुकूलित (optimized) एक android वर्चुअल मशीन है। यह मेमोरी, battery life और performance के लिए वर्चुअल मशीन का अनुकूलन (optimizes) करता है। सा कि हम जानते हैं कि आधुनिक JVM high performance और excellent memory management प्रदान करता है। किन इसे low-powered handheld devices के लिए भी optimize किया जाना चाहिए। जावा कंपाइलर (javac) जावा source code जावा बाइट-कोड (.class) में परिवर्तित करता है। तब DEX कंपाइलर इस (.class) फाइल को dalvik बाइट कोड या “dex” (Dalvik Executable File) फाइल में बदल देता है।

डेक्स कंपाइलर .class की फाइलों को .dex फाइल में बदल देता है, जो डाल्ट्रिक वीएस पर चलती है। मल्टीपल क्लास फाइलें एक dex फाइल में बदल जाती हैं। उपरोक्त फिगर में सोर्स फाइल से compiling और packaging प्रक्रिया को प्रदर्शित किया गया है।



चित्र 1.5

- Javac टूल java source फाइल को .class फाइल में कन्वर्ट करता है।
- Dx टूल आपके एप्लिकेशन की सभी क्लास फाइल्स लेता है और एक .dex फाइल बनाता है। यह एक प्लेटफॉर्म specific उपकरण है।
- Android Assessts Packaging Tool (aapt) packaging प्रक्रिया को handle करता है।

1.4.1. Advantages

- DVM केवल android ऑपरेटिंग सिस्टम का समर्थन करता है।
- DVM executable में apk है।
- Execution तेज है।
- एन्ड्रॉइड 2.2 से SDK dalvik में इसका अपना (JIT) जस्ट-इन-टाइम कंपाइलर है।
- DVM को इसलिए डिजाइन किया गया है ताकि कोई डिवाइस वर्चुअल मशीन के multiple instances को effectively से run कर सके।

डिवाइसमें ऑफ एन्ड्रॉइड एप्लिकेशन्स

4.2. Disadvantages

- DVM केवल Android ऑपरेटिंग सिस्टम का समर्थन करता है।
- DVM के लिए बहुत कम re-tools उपलब्ध हैं।
- समान high-level कोड को लागू करने के लिए रजिस्टर मशीनों से अधिक instructions की आवश्यकता होती है।
- dex के कारण ऐप इंस्टॉलेशन में अधिक समय लगता है।
- अधिक internal storage की आवश्यकता है।

1.5. APK File Extension

APK (Android Package Kit) एक एन्ड्रॉइड एप्लिकेशन पैकेज है। एपीके एन्ड्रॉइड के बीच उपयोग किया जाने वाला एप्लिकेशन फाइल प्रकार है। apk फाइलें Android studio में compiled की जाती हैं, जो एन्ड्रॉइड सॉफ्टवेयर के निर्माण के लिए official integrated development environment (IDE) है।

एपीके फाइल एन्ड्रॉइड, Google के मोबाइल ऑपरेटिंग सिस्टम Android के लिए बनाई गई एक ऐप है। कुछ ऐप एन्ड्रॉइड डिवाइस पर पहले से इंस्टॉल आते हैं, जबकि अन्य ऐप Google Play से डाउनलोड किए जा सकते हैं, आप एपीके फाइल्स मैनुअली इंस्टॉल कर सकते हैं।

- एपीके फाइल में assets, resources, certificates, manifest file, application programs जैसे dex फाइल्स आदि होते हैं।
- एन्ड्रॉइड फ्रेमवर्क client को एपीके डेटा फाइलें install करने में मदद करता है।
- कई कारणों से, कोई भी इसका उपयोग कर सकता है, जिसमें गैर-स्टोर एप्लिकेशन इंस्टॉल करना या पहले का संस्करण इंस्टॉल करना शामिल है।
- हालांकि शुरू में नए संस्करण को uninstalling करके किसी एप्लिकेशन को डाउनग्रेड करना संभव है, यह एन्ड्रॉइड Debug bridge के माध्यम से करना सुरक्षित है क्योंकि यह डेटा को save की अनुमति देता है।
- एपीके फाइलें एक compressed Zip प्रारूप में सहेजी जाती हैं और इन्हें किसी भी zip decompression टूल द्वारा खोला जा सकता है।

1.6. Android-Environment Setup

Android एनवायरनमेंट को स्थापित करने के लिए following सॉफ्टवेयर पैकेज स्थापित किया जाना चाहिए—

- Window/Linux new version
- Java JDK5 or later version
- Android studio

1.6.1. Java Development Kit (JDK)

आप Oracle के जावा साइट से जावा JDK का नवीनतम संस्करण डाउनलोड कर सकते हैं। आपको डाउनलोड की गई फाइलों में जेडीके स्थापित करने के लिए निर्देश मिलेगा, सेटअप को install और configure करने के लिए दिए गए निर्देशों का पालन करें। अंत में PATH और JAVA-HOME वातावरण variables को उस directory के संदर्भ में सेट करें जिसमें क्रमशः java और javac शामिल हैं।

1.6.2. Android IDEs

एन्ड्रॉइड एप्लिकेशन, familiar technologies को विकसित करने के लिए बहुत सारे sophisticated टेक्नोलॉजीज उपलब्ध हैं, जो मुख्य रूप से निम्नानुसार tools का उपयोग कर रहे हैं—

Android studio

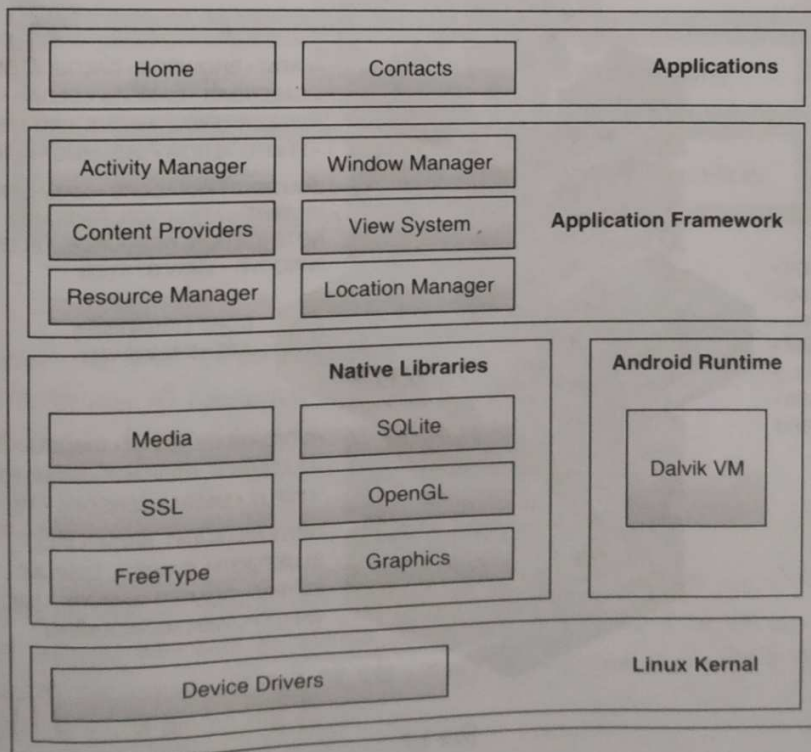
Eclipse IDE

1.7. Android Architecture

Android आर्किटेक्चर या android सॉफ्टवेयर स्टैक को पाँच भागों में वर्गीकृत किया गया है—

1. Linux kernel
2. Native libraries (middleware)
3. Android runtime
4. Application framework
5. Applications

1. Linux kernel—यह एन्ड्रॉइड आर्किटेक्चर का heart है जो एन्ड्रॉइड आर्किटेक्चर के root में मौजूद है। लिनक्स कर्नेल डिवाइस ड्राइवर्स, पावर मैनेजमेंट, मेमोरी मैनेजमेंट, डिवाइस मैनेजमेंट और रिसोर्स एक्सेस के लिए जिम्मेदार है।



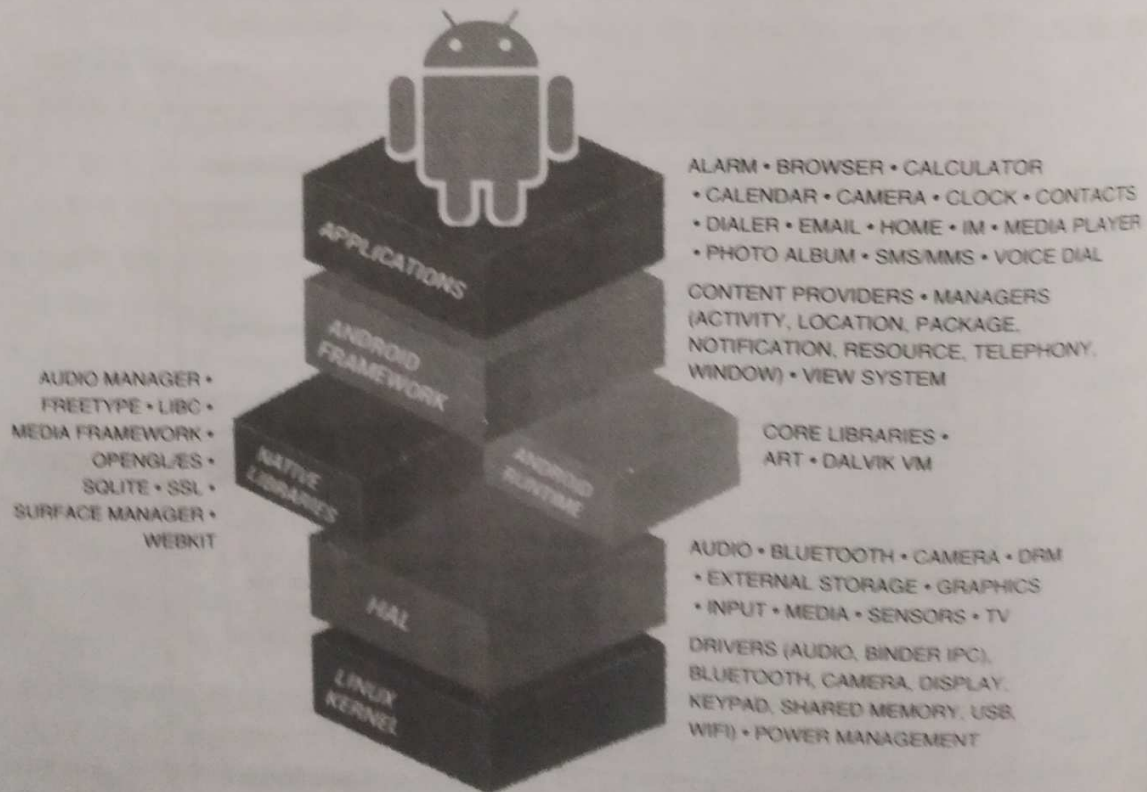
चित्र 1.6

2. **Native libraries (middleware)**—लिनक्स कर्नेल के top पर libraries का एक सेट है, जिसमें ओपन-सोर्स वेब ब्राउजर इंजन WebKit, libraries libc, SQLite डेटाबेस शामिल हैं, जो एप्लिकेशन डेटा के storage और sharing करने के लिए, ऑडियो और वीडियो चलाने और रिकॉर्ड करने के लिए एक उपयोगी storage है, SSL लाइब्रेरी इंटरनेट सुरक्षा आदि के लिए जिम्मेदार library है।

3. **Android runtime**—एन्ड्रॉइड रनटाइम में, कोर लाइब्रेरी और DVM (dDalvik वर्चुअल मशीन) है। एन्ड्रॉइड एप्लिकेशन को चलाने के लिए जिम्मेदार है। डीवीएम जेवीएम की तरह है लेकिन यह मोबाइल उपकरणों के लिए अनुकूलित है। यह less memory consumes करता है और fast performance प्रदान करता है।

4. **Application framework**—नेटिव लाइब्रेरी और एन्ड्रॉइड रनटाइम के शीर्ष पर एन्ड्रॉइड फ्रेमवर्क है। एन्ड्रॉइड फ्रेमवर्क में एन्ड्रॉइड API जैसे UI (user इंटरफेस), टेलीफोनी, संसाधन, स्थान, content provider (डेटा) और पैकेज manager शामिल हैं। यह android application development के लिए बहुत सारी classes और interface प्रदान करता है।

5. **Applications**—Android framework के शीर्ष पर application हैं। सभी एप्लिकेशन जैसे home, contact सेटिंग्स, गेम, ब्राउजर एन्ड्रॉइड फ्रेमवर्क एन्ड्रॉइड में रनटाइम और लाइब्रेरीज का उपयोग कर रहे हैं। एन्ड्रॉइड रनटाइम और native libraries में लिनक्स कर्नेल का उपयोग किया जाता है।

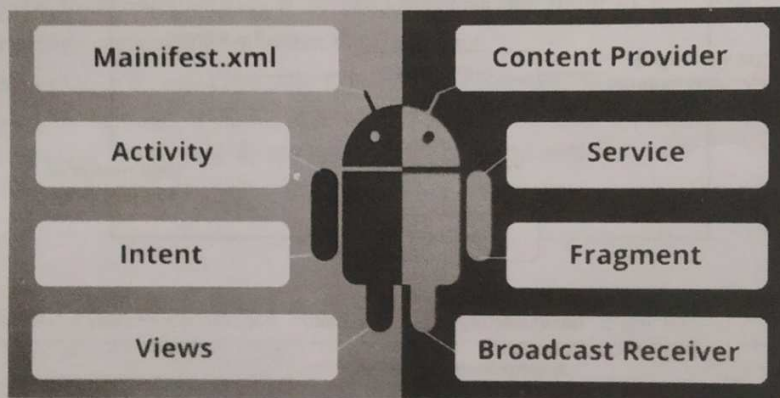


चित्र 1.7

1.8. Android Basic Building Blocks/Component

Component एक Android के आवश्यक building blocks हैं एक एन्ड्रॉइड component piece of code है, इसमें एक अच्छी तरह से परिभाषित life cycle है e.g., activity, receiver, service आदि। basic building block या एन्ड्रॉइड के basic component निम्नलिखित हैं—

1. Activities
2. Services
3. Broadcast receivers
4. Content providers
5. Views
6. Intents
7. Fragments
8. AndroidManifest.xml

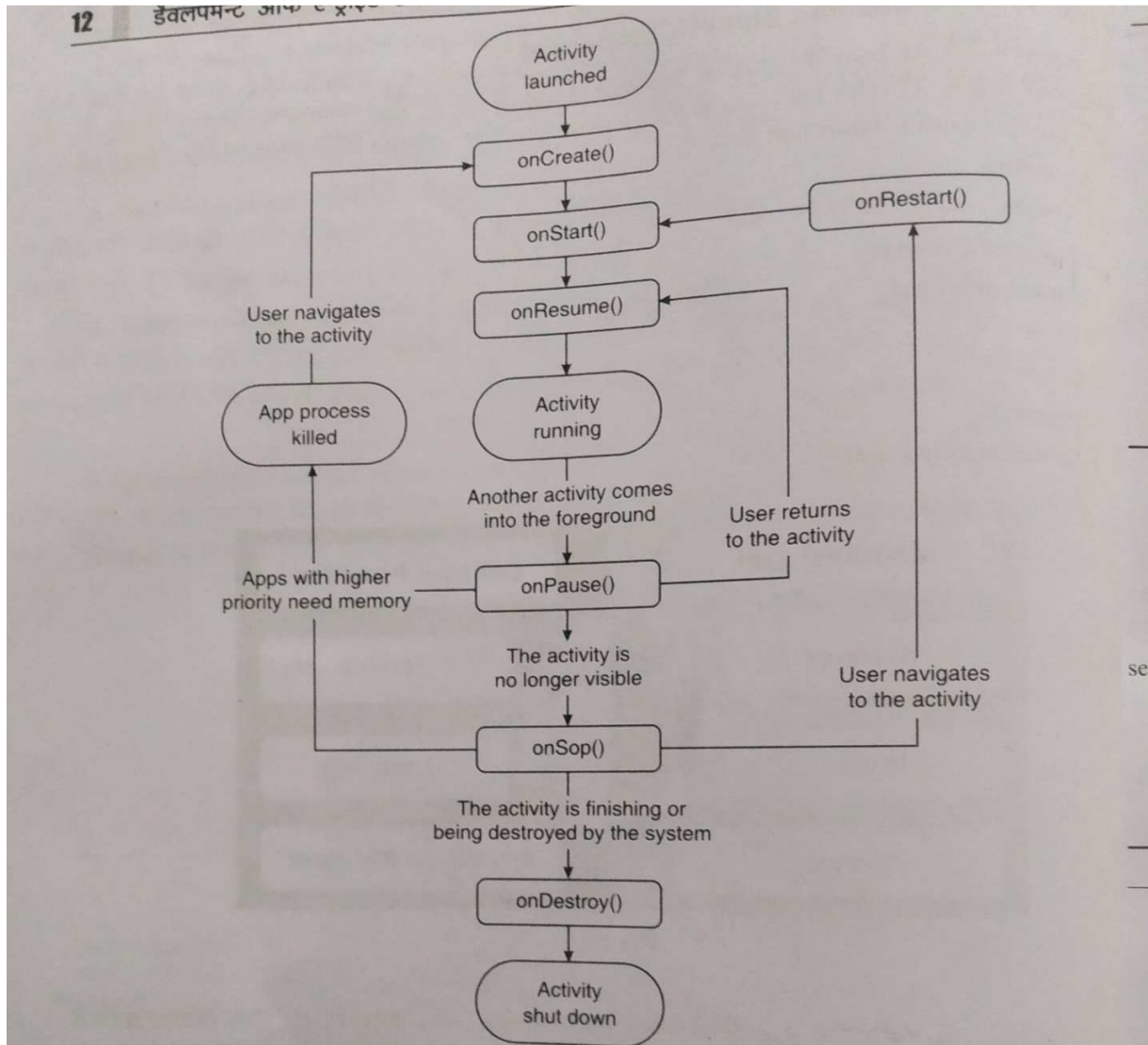


चित्र 1.8

1. Activities—यह केवल एक class है जो एकल स्क्रीन को represents करता है। वे UI को निर्देशित करते हैं और स्मार्ट फोनस्क्रीन पर user की interaction को संभालते हैं। यह AWT में एक फ्रेम की तरह है। एक activity को activity class के subclass के रूप में implemented किया जाता है—

```
public class Main Activity extends Activity
{
    .....
}
```

(a) Activity life cycle—Android सिस्टम onCreate()callback method पर call के साथ शुरू होने वाली activity से अपना program शुरू करता है। कॉलबैक विधियों का एक sequence है जो एक activity शुरू करता है callback methods का एक sequence होता है जो एक activity को फाड़ (tear down) देता है जैसा कि नीचे दी गई activity life cycle diagram में दिखाया गया है।



चित्र 1.9

Activity class निम्नलिखित call backs method को परिभाषित करता है—

S.No.	Callback & Description
1.	onCreate() यह पहला callback है और जब activity पहली बार बनाई जाती है तो उसे कॉल किया जाता है।
2.	onStart() user को activity दिखाई देने पर इस callback को call किया जाता है।
3.	onResume() यह तब call किया जाता है जब user एप्लिकेशन के साथ interact करना शुरू करता है।

- | | |
|----|---|
| 4. | onPause()
रुकी हुई activity user इनपुट प्राप्त नहीं करती है और किसी भी कोड को execute नहीं करती है और इसको तब call किया जाता है जब वर्तमान activity को रोका जा रहा है और पिछली activity को फिर से शुरू किया जा रहा है। |
| 5. | onStop()
यह callback तब किया जाता है जब activity दिखाई नहीं देती है। |
| 6. | onDestroy
सिस्टम द्वारा activity को destroy करने से पहले यह callback किया जाता है। |
| 7. | onRestart()
यह callback तब किया जाता है जब activity को रोकने के बाद फिर से शुरू होता है। |

6. Services

Services एक background process है जो लम्बे समय तक चल सकती है। वे 2 प्रकार के होते हैं—

- Local service : accessed from within the application.
- Remote service : accessed remotely from other applications running on the same device.

एक service, background में music play कर सकती है जबकि user एक different एप्लिकेशन में है। एक service को service class के subclass के रूप में implemented किया जाता है—

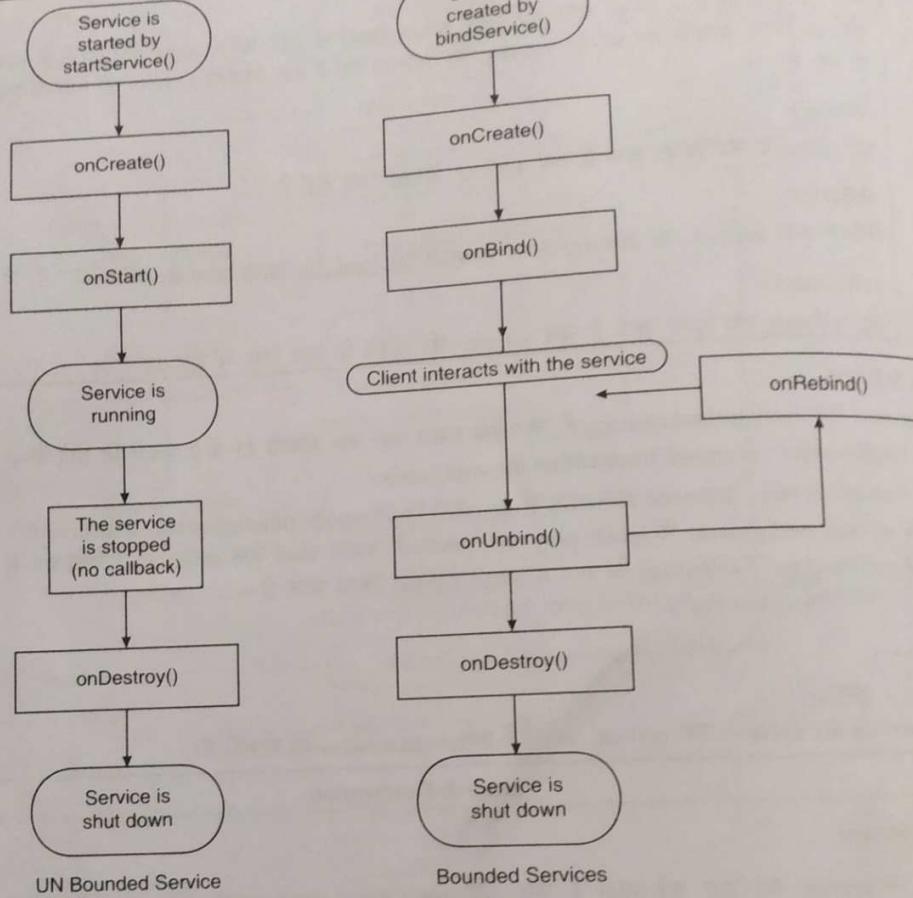
```
public class MyService extends Service
{
    ....
}
```

a. Service life cycle—एक service अनिवार्य रूप से two states हो सकती है।

S.No.	State & Description
1.	Started एक service तब शुरू की जाती है जब कोई application component, जैसे कि एक activity, इसे startService () call कर शुरू करता है। एक बार service शुरू होने के बाद, background अनिश्चित काल (indefinitely) चल सकती है, भले ही यह शुरू करने वाले component को destroy कर दिया गया हो।
2.	Bound एक service तब bound होती है जब कोई application component उसे bindService() कहकर बांधता है। एक bound service एक क्लाइंट-सर्वर इंटरफेस प्रदान करती है जो components को service के साथ interact करने, send requests, result प्राप्त करने और यहाँ तक कि inter-process communication (IPC) के साथ process भी करती है।

Service में life cycle कॉलबैक methods हैं जिन्हें आप service's state में परिवर्तन की monitor के लिए लागू कर सकते हैं और आप उचित स्तर पर काम कर सकते हैं। जब सेवा startService () के साथ बनाई गई है उसको नीचे दिए गए चित्र left side life cycle दर्शाया गया है। जब सेवा bindService () के साथ बनाई गई है उसको नीचे दिए गए चित्र right side life cycle दर्शाया गया है।

14 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स



चित्र 1.10

Service base विभिन्न callback method को परिभाषित करता है और सबसे महत्वपूर्ण नीचे दिये गये हैं—

S.No.	Callback & Description
1.	onStartCommand() सिस्टम इस method को कॉल करता है जब another component, जैसे कि एक activity, request करता कि service शुरू की जाए, तब startService() कहकर service शुरू की जाती है। यदि आप इस method को implement करते हैं, तो यह आपकी जिम्मेदारी है कि जब सेवा बंद हो जाए तो सेवा को रोक दें, स्टॉपसेवा या स्टॉप सर्विस () विधियों को कॉल करके।
2.	onBind() सिस्टम इस method को तब कॉल करता है जब कोई another component bindService() को कॉल करके service के साथ bind करना चाहता है।

3.	onUnbind() सिस्टम इस method को तब कॉल करता है जब सभी क्लाइंट particular interface डिस्कनेक्ट होना चाहते हैं।
4.	onRebind() जब नए क्लाइंट service से जुड़ जाते हैं, तो सिस्टम इस method को कॉल करता है।
5.	onCreate() जब सिस्टम पहली बार onStartCommand () या onBind () का उपयोग करके बनाया जाता है, तो सिस्टम इस method को कॉल करता है। यह कॉल one-time set-up करने के लिए आवश्यक है।
6.	onDestroy() सिस्टम इस method को तब कॉल करता है जब service अब उपयोग नहीं की जाती है और destroy हो रही है।

7. Broadcast Receivers—Broadcast Receivers केवल अन्य एप्लिकेशन या सिस्टम से Broadcast messages का जवाब देता है। वे एंड्रॉइड ओएस और एप्लिकेशन के बीच संचार को संभालते हैं। उदाहरण के लिए, एप्लिकेशन अन्य applications को यह बताने के लिए broadcasts कर सकते हैं कि कुछ डेटा डिवाइस में डाउनलोड कर लिए गए हैं और उनका उपयोग करने के लिए उपलब्ध है, इसलिए यह broadcast receiver है जो इस संचार को intercept करेगा और appropriate action. शुरू करेगा।

एक broadcast receiver को BroadcastReceiver class के subclass के रूप में implements किया जाता है और प्रत्येक message broadcaster के रूप में एक Intent object होता है।

```
public class MyReceiver extends BroadcastReceiver
{
    public void onReceive(context,intent){}
}
```

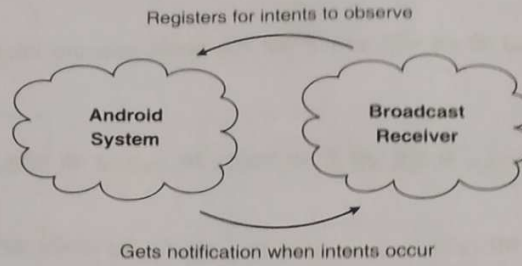
निम्नलिखित दो महत्वपूर्ण steps के द्वारा BroadcastReceiver बनाया जाता है।

- Creating the Broadcast Receiver.
- Registering Broadcast Receiver.

(a) Creating the Broadcast Receiver—ब्रॉडकास्ट रिसीवर को BroadcastReceiver class के subclass रूप में implement किया जाता है और onReceive() method को ओवरराइड किया जाता है। जहाँ प्रत्येक message को Intent object parameter के रूप में प्राप्त किया जाता है।

```
public class MyReceiver extends BroadcastReceiver {
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context, "Intent
        Detected.", Toast.LENGTH_LONG).show();
    }
}
```

(b) Registering Broadcast Receiver—एक application specific broadcast intents तभी listen कर सकता है जब broadcast receiver को AndroidManifest.xml फाइल में register करे।



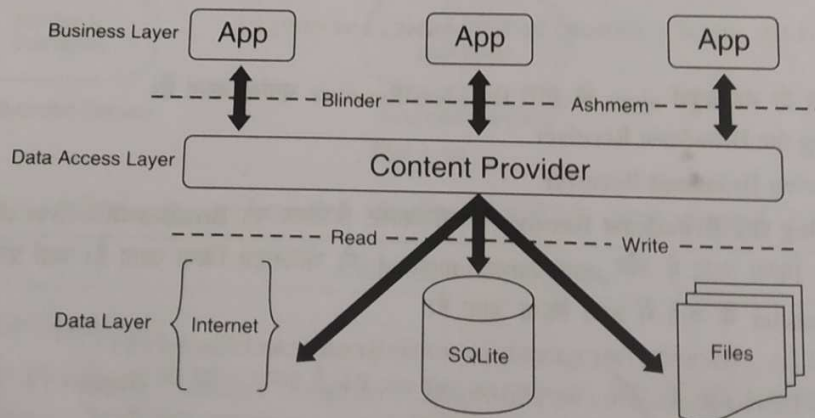
चित्र 1.11

8. Content Providers—Content Providers डेटा को एन्कैप्सुलेट करके application को content प्रदान करते हैं और इसे एक single Content Resolver इंटरफेस के माध्यम से application को प्रदान करते हैं। कंटेंट प्रोवाइडर आवश्यक होते हैं, यदि आपको multiple applications के बीच डेटा share करने की आवश्यकता है e.g., contacts data का उपयोग multiple applications द्वारा किया जाता है और list content provider में store किया जाता है।

एक Content Providers को ContentProvider class के subclass के रूप में implement किया जाता है।

```

public class MyContentProvider extends ContentProvider
{
    public void onCreate()
    {
        // ...
    }
}
  
```



चित्र 1.12

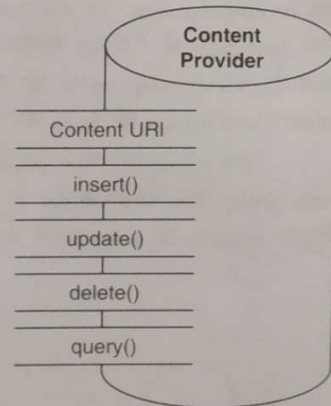
एक content provider एक डेटाबेस की तरह व्यवहार करता है जहाँ आप इसे query कर सकते हैं, इसका content को edit कर सकते हैं, साथ ही insert(), update(), delete() and query() methods का उपयोग करके सामान को add या delete कर सकते हैं। ज्यादातर मामलों में यह डेटा एक SQLite डेटाबेस में store होता है।

(a) Create Content Provider—

- सबसे पहले एक Content Provider class बनाने के लिये ContentProviderbaseclass को extends किया जाता है।
- दूसरा, आपको अपने content provider URI address को define करने की आवश्यकता है जिसका उपयोग content तक पहुँचने के लिए किया जाएगा।
- Next आपको content रखने के लिए अपना own का डेटाबेस बनाना होगा।
- Next आपको विभिन्न डेटाबेस specific operation करने के लिए Content Provider queries को लागू करना होगा।
- अंत में <provider>tag का उपयोग करके अपनी Content Provider को अपनी activity life में register करें।

यहाँ उन methods की सूची दी गई है, जिन्हें आपको अपने Content Provider के काम करने के लिए Content Provider class में override करने की आवश्यकता है—

- **onCreate()** जब provider शुरू किया जाता है तो इस method को call किया जाता है।
- **query()** यह method क्लाइंट से request प्राप्त करती है। result को Cursor object के रूप में लौटाया जाता है।
- **insert()** यह method content provider में एक new record को insert करती है।
- **delete()** यह method content provider में एक existing record को delete करती है।
- **update()** यह method content provider में एक existing record को update करती है।
- **getType()** यह method दिए गए URI में MIME प्रकार का डेटा return करती है।



चित्र 1.13

9. **Views**—Views एक UI element है जैसे बटन, लैबल, टेक्स्ट फील्ड, इमेज।

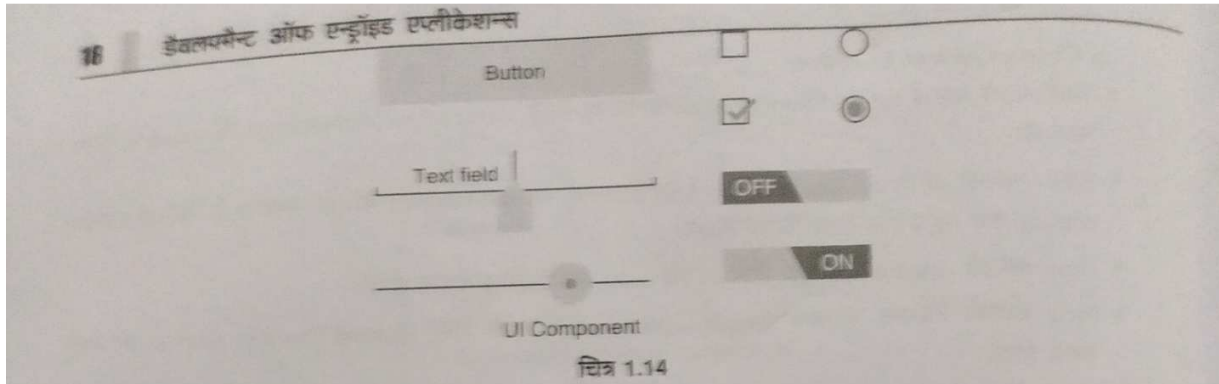
10. **Intents**—यह एक शक्तिशाली inter-application message passing फ्रेमवर्क है।

11. **Fragments**—ये एक Activity का हिस्सा हैं। एक Activity एक ही समय में स्क्रीन पर एक या अधिक fragments प्रदर्शित कर सकती है।

12. **AndroidManifest.xml**—इसमें activities, content providers, permissions आदि के बारे में सुझाव है। यह Java EE में वेब .Xml फाइल की तरह है।

1.9. UI Components

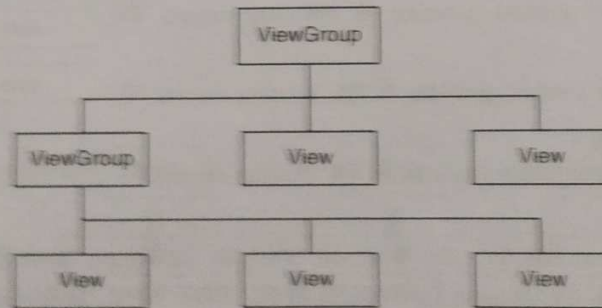
एंड्रॉइड ऐप के लिए यूजर इंटरफेस (UI) को ले-आउट और विजेट्स के hierarchy के रूप में बनाया गया है। एंड्रॉइड विभिन्न प्रकार के pre-built UI components जैसे layout objects और UI controls प्रदान करता है जो आपको अपने ऐप के लिए ग्राफिकल यूजर इंटरफेस में हेल्प करता है। एन्ड्रॉइड आपके यू-आई में प्रयोग किया जाने वाले विभिन्न प्रकार के controls प्रदान करता है, जैसे buttons, text fields, seek bars, check box, zoom buttons, toggle buttons, और कई अन्य।



1.9.1. Views

user interface के लिए basic विलडिंग एक View object है जो view class से बनाया गया है और स्क्रीन पर एक rectangular area पर कब्जा कर लेता है। और दृष्टि और इवेंट हैंडलिंग के लिए जिम्मेदार है। View, widgets के लिए base class है, जिसका उपयोग interactive UI components जैसे बटन, टेक्स्ट फ़ील्ड आदि बनाने के लिए किया जाता है। ViewGroup, view का एक subclass है और invisible container प्रदान करता है जो other Views या other ViewGroups को hold करता है और उनके layout properties को परिभाषित करते हैं।

“एक view एक ऐसी object है जो स्क्रीन पर कुछ draws करता है जिससे user interact कर सकता है और view group एक ऐसा object है जो user interface के layout को परिभाषित करने के लिए other View (और व्यूगट) objects को hold करता है।



चित्र 1.15

Android ViewGroup और View कक्षाओं के लिए एक XML शब्दावली प्रदान करता है, इसलिए आपका अधिकांश UI XML फाइलों में परिभाषित किया गया है। Third level पर हमारे पास अलग-अलग ले-आउट है जो ViewGroup class के subclasses हैं और एक typical ले-आउट एक एंड्रॉइड user interface के लिए visual structure को परिभाषित करता है और View/ViewGroup objects का उपयोग करके runtime पर बनाया जा सकता है या आप अपने ले-आउट को XML file main-layout.xml में declare कर सकते हैं जो आपके प्रोजेक्ट के res/layout folder में स्थित है।

आप दो तरीकों से एक ले-आउट को declare कर सकते हैं—

(i) **Declare UI elements in XML :** एंड्रॉइड एक सरल XML vocabulary प्रदान करता है जो View classes और subclasses से मेल खाता है, जैसे कि widgets और layouts ।

(ii) **Instantiate layout elements at runtime :** आप View और ViewGroup objects की प्रोग्रामिंग से बन सकते हो।

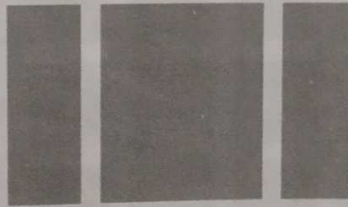

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is a TextView"/>
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is a Button"/>
    <!-- More GUI components go here -->
</LinearLayout>

```

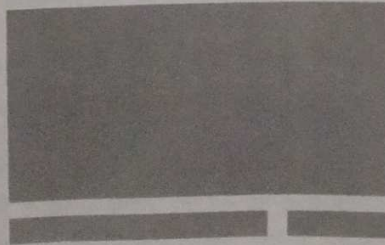
(a) **Android Layout Types**—एन्ड्रॉइड द्वारा प्रदान किए गए ले-आउट्स—

(i) **Linear Layout**—Linear Layout एक view group है जो सभी children को एक ही direction में vertically या horizontally रूप से align करता है।



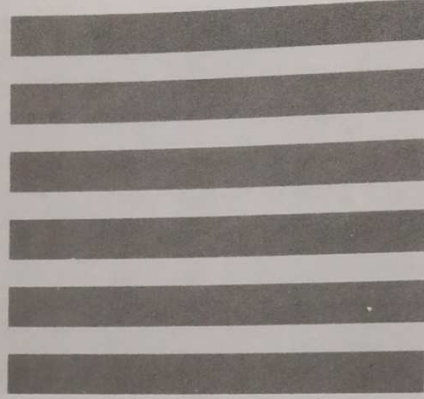
चित्र 1.16

(ii) **Relative Layout**—Relative Layout एक view group है जो relative positions में child views को displays करता है।



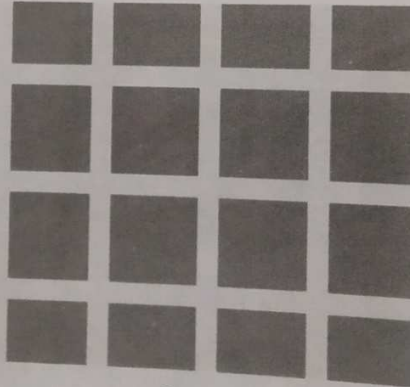
चित्र 1.17

- (iii) **Table Layout**—Table Layout एक view है जो groups rows और columns में देखता है।
- (iv) **Absolute Layout**—निरपेक्ष ले-आउट आपको children के exact location को specify करने में सक्षम बनाता है।
- (v) **Frame Layout**—Frame Layout स्क्रीन पर एक placeholder है जिसे आप single view प्रदर्शित करने के लिए उपयोग कर सकते हैं।
- (vi) **List View**—List View एक view group है जो scroll करने योग्य items की list प्रदर्शित करता है।



चित्र 1.18

- (vii) **Grid View**—Grid View एक view group है जो two-dimensional, scroll करने योग्य ग्रिड में items प्रदर्शित करता है।



चित्र 1.19

(b) **Android UI Controls**—एन्ड्रॉइड द्वारा प्रदान किए गए UI controls आपको app के लिए ग्राफिकल यूजर इंटरफेस बनाने की provided किए गए हैं।

- (i) **Text View**—इस control का प्रयोग टेक्स्ट डिस्प्ले करने के लिए किया जाता है।
- (ii) **Edit Text**—Edit Text, Text View का एक predefined subclass है जिसका प्रयोग editing के लिए किया जाता है।
- (iii) **Button**—push-button को कोई action perform करने के लिए press और click किया जाता है।
- (iv) **Check Box**—Check box का प्रयोग ऑप्शन को सलेक्ट करने के लिए किया जाता है।
- (v) **Radio Button**—Radio Button की two states होती हैं। या तो checked या unchecked होगी।
- (vi) **Progress Bar**—Progress Bar view कुछ ongoing tasks के बारे में visual feedback प्रदान करता है।

(vii) **Spinner**—एक drop-down list जो user को एक सेट से एक value का select करने की अनुमति देती है।

(viii) **Time Picker**—Time Picker view user को 24-घण्टे मोड या AM/PM मोड में, दिन के समय का चयन करने में सक्षम बनाता है।

(ix) **Data Picker**—Data Picker view user को दिन की एक तारीख का चयन करने में सक्षम बनाता है।

(c) **Create UI Controls**—इनपुट controls आपके ऐप के user इंटरफेस में interactive components हैं। एन्ड्रॉइड आपके यूआई में उपयोग किए जाने वाले विभिन्न प्रकार के controls प्रदान करता है, जैसे buttons, text fields, seek bars, check box, zoom buttons, toggle buttons और कई अन्य।

UI Control/View/Widget बनाने के लिए आपको layout फाइल में view/widget को परिभाषित करना होगा और इसे एक unique ID के assign करना होगा—

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">

    <TextView android:id="@+id/text_id"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="I am a TextView"/>

</LinearLayout>
```

फिर अंत में Control object का एक instance बनाएँ और इसे layout से capture करें, निम्नलिखित का उपयोग करें—

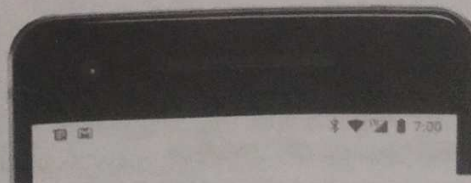
```
TextView myText=(TextView)findViewById(R.id.text_id);
```

1.9.2. Notifications

Notification एक message है जिसको एन्ड्रॉइड आपके app's UI के outside display करता है ताकि user को reminders, अन्य लोगों से communication, या आपके ऐप के बारे में other information timely प्रदान कर सके। user आपके एप्लिकेशन को open करने के लिए notification पर tap कर सकते हैं या notification से सीधे action कर सकते हैं।

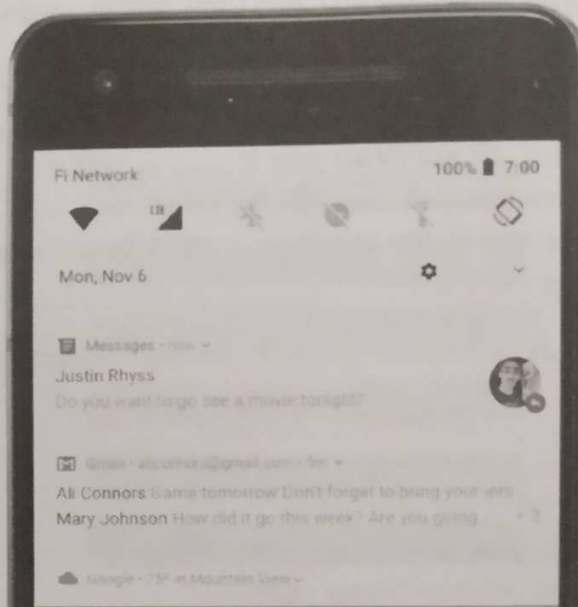
Notifications विभिन्न locations और formats में user को दिखाई देती हैं, जैसे status bar में एक icon etc. different types ऑफ नोटिफिकेशन को निम्नलिखित बताया गया है।

(a) **Status bar and notification drawer**—यह page उन सूचनाओं का overview provides करता है जहाँ सूचनाएँ और features दिखाई देती हैं। जब आप एक notification जारी करते हैं, तो यह पहली बार status bar में एक icon के रूप में दिखाई देता है।



चित्र 1.20—Notification icons appear on the left side of the status bar

Users notification drawer खोलने के लिए status bar को नीचे स्वाइप कर सकते हैं, जहाँ वे more details देख सकते हैं और notification के साथ actions कर सकते हैं।



चित्र 1.21—Notifications in the notification drawer

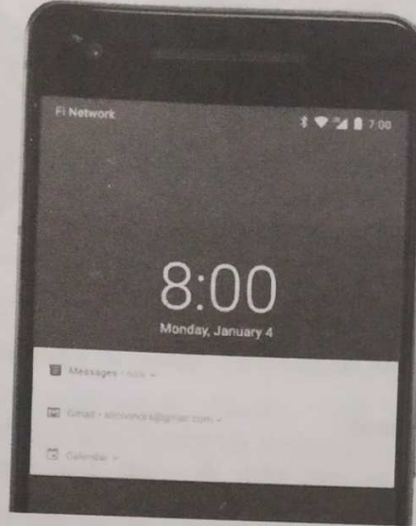
app या user द्वारा dismissed किए जाने तक एक सूचना notification drawer में दिखाई देती है।

(b) Heads-up notification—एन्ड्रॉइड 5.0 के साथ शुरुआत, notifications संक्षेप में एक फ्लोटिंग विंडो में दिखाई दे सकती है जिसे हेड-अप अधिसूचना कहा जाता है। यह behavior आमतौर पर important notifications के लिए होता है जो उपयोगकर्ता को तुरंत पता होनी चाहिए, और यह केवल तब दिखाई देता है जब डिवाइस अनलॉक हो।



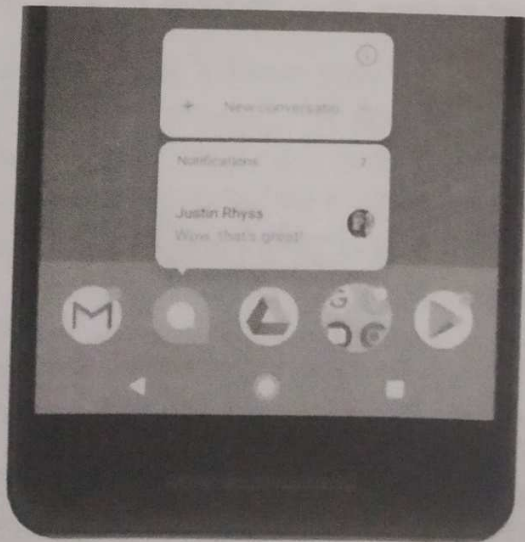
चित्र 1.22—A heads-up notification appears in front of the foreground app

(c) **Lock screen**—एन्ड्रॉइड 5.0 के साथ शुरुआत, लॉक स्क्रीन पर सूचनाएँ दिखाई दे सकती हैं। आप ऐप द्वारा सुरक्षित लॉक स्क्रीन पर पोस्ट किए गए नोटिफिकेशन में दिखाई दे रहे विवरण के स्तर को प्रोग्रामेटिक रूप से सेट कर सकते हैं। Users लॉक स्क्रीन सूचनाओं में दिखाई देने वाले विस्तार के स्तर को चुनने के लिए और सभी लॉक स्क्रीन सूचनाओं को निष्क्रिय करने के लिए सिस्टम सेटिंग्स का उपयोग कर सकते हैं।



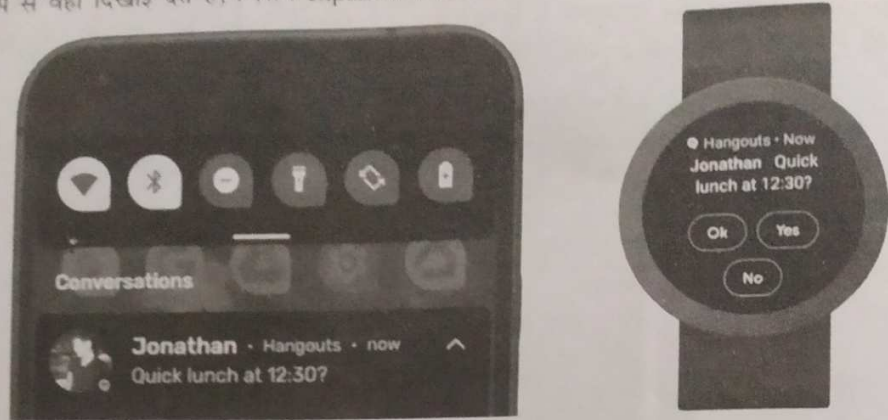
चित्र 1.23—Notifications on the lock screen with sensitive content hidden

(d) **App icon badge**—एन्ड्रॉइड 8.0 और उच्चतर पर चलने वाले उपकरणों पर, ऐप आइकन सम्बन्धित app launcher icon पर एक रंगीन “badge” के साथ new notifications दर्शाते हैं। Users उस ऐप के लिए सूचनाएँ देखने के लिए ऐप आइकन पर लम्बे समय तक दबा सकते हैं। उपयोगकर्ता तब उस मेनू से सूचनाओं को dismiss या act on कर सकते हैं, notification drawer के समान।



चित्र 1.24—Notification badges and the long-press menu

(e) Wear OS devices—यदि उपयोगकर्ता के पास Wear OS डिवाइस है, तो आपके सभी नोटिफिकेशन स्वचालित रूप से वहाँ दिखाई देते हैं, जिसमें expandable detail और action buttons शामिल हैं।



चित्र 1.25-Notifications automatically appear on a paired wear OS device

1.9.3. Create and Send Notifications

आपके पास notification बनाने का सरल तरीका है। notification बनाने के लिए अपने app में निम्नलिखित steps का पालन करें—

Step 1 : Create Notification Builder—first step के रूप में NotificationCompat.Builder.build () का उपयोग करके एक notification builder बनाना है। आप अपने small और large icons, title, priority आदि जैसे विभिन्न Notification properties को सेट करने के लिए Notification Builder का उपयोग करेंगे।

NotificationCompat.Builder mBuilder=new NotificationCompat.Builder(this)

यदि आपके पास Builder object है, तो आप अपनी आवश्यकता के अनुसार Builder object का उपयोग करके Notification properties सेट कर सकते हैं। लेकिन at least निम्नलिखित सेट करना अनिवार्य है—

- A small icon, set by setSmallIcon()
- A title, set by setContentType()
- Detail text, set by setContentText()

```
mBuilder.setSmallIcon(R.drawable.notification_icon);
mBuilder.setContetTitle("Notification Alert, Click Me!");
mBuilder.setContentText("Hi, This is Android Notification
Detail!");
```

Step 3 : Attach Actions—यह optional part है, यह आवश्यक है यदि आप notification के साथ एक action attach करना चाहते हैं। Action, user को notification से सीधे आपके application में एक Activity तक जाने की अनुमति देती है, जहाँ वे एक या अधिक event को देख सकते हैं या further work कर सकते हैं।

Action को एक PendingIntent जो एक ऐसे Intent को contain किये हुए है जो आपके application में activity शुरू करता है, के द्वारा परिभाषित किया गया है PendingIntent को एक gesture के साथ जोड़ने के लिए NotificationCompat.Builder के appropriate method को कॉल करें।

उदाहरण के लिए, यदि आप user को notification drawer में notification text पर क्लिक करते समय activity शुरू करना चाहते हैं, तो आप `setContentIntent()` को कॉल करके `PendingIntent` जोड़ें।

`PendingIntent` object आपके application की ओर से action करने में आपकी सहायता करता है, अक्सर बाद के समय में, चाहे आपकी एप्लिकेशन चालू हो या न हो।

हम `stack builder` object की मदद लेते हैं जिसमें शुरू की गई Activity के लिए artificial back stack होता है। यह ensure करता है कि Activity से backward navigating करने से आपके application Home screen से बाहर तो नहीं हो जाते हैं।

```
Intent resultIntent = new Intent(this, ResultActivity.class);
TaskStackBuilder stackBuilder = TaskStackBuilder.create(this);
stackBuilder.addParentStack(ResultActivity.class);
// Adds the Intent that starts the Activity to the top of the stack
stackBuilder.addNextIntent(resultIntent);
PendingIntent resultPendingIntent =
    stackBuilder.getPendingIntent(0, PendingIntent.FLAG_UPDATE_CURRENT);
mBuilder.setContentIntent(resultPendingIntent);
```

Step 4 : Issue the notification—Finally, आप अपनी Notification भेजने के लिए `NotificationManager.notify()` को call करके सिस्टम को Notification object pass करते हैं। यह सुनिश्चित करें कि आप `NotificationCompat.Builder.build()` method को notifying करने से पहले builder object पर call करें। यह method उन सभी options को combines करता है जो सेट किए गए हैं और एक नई **Notification** object को return करता है।

```
NotificationManager mNotificationManager = (NotificationManager)
    getSystemService(Context.NOTIFICATION_SERVICE);
// notificationID allows you to update the notification later on.
mNotificationManager.notify(notificationID, mBuilder.build());
```

1.10. संचार के स्रोत (Components for Communication)

1.10.1. Intents

एक Intent एक messaging object है जिसका उपयोग आप किसी another app component से action perform करने के लिए कर सकते हैं। हालांकि intents कई तरीकों से components के बीच communication की सुविधा प्रदान करते हैं, तीन fundamental उपयोग के मामले जो निम्नलिखित हैं—

(i) **Starting an activity**—एक activity एक ऐप में एक स्क्रीन को represent करती है। आप एक activity का एक new instance शुरू करने के `startActivity()` में Intent को pass करके शुरू कर सकते हैं। Intent किसी भी activity को start करने और आवश्यक डेटा को ले जाने के method का वर्णन करता है यदि आप किसी इसे activity के समाप्त होने पर एक परिणाम प्राप्त करना चाहते हैं, तो `startActivityForResult()` कॉल करें।

(ii) **Starting a service**—एक service एक component है जो user interface के बिना background में operations performs करता है। आप `JobScheduler` के साथ एक Service शुरू कर सकते हैं।

आप Service class के तरीकों का method करके service शुरू कर सकते हैं। आप one-time operation perform करने के लिए `startService()` में Intent को pass करके एक service शुरू कर सकते हैं। Intent किसी भी activity को start करने और आवश्यक डेटा को ले जाने के method का वर्णन करता है।

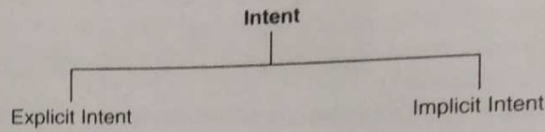
26 डेवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

यदि service client-server interface के साथ डिजाइन की गई है, तो आप किसी अन्य component bindService() पर एक Intent पास करके service को bind कर सकते हैं।

(iii) **Delivering a broadcast**—Broadcast एक message है जिसे कोई भी ऐप receive कर सकता है। सिस्टम, system events के लिए various broadcast को deliver करता है, जैसे कि जब सिस्टम boots up होता है। डिवाइस चार्ज करना शुरू होता है। आप SendBroadcast() या sendOrderedBroadcast() पर एक Intent पास कर other apps पर broadcast कर सकते हैं।

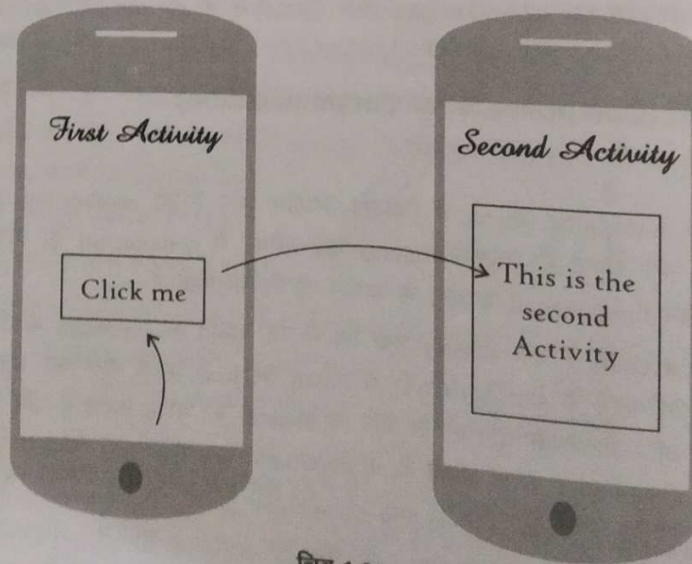
1.10.2. Intent types.

Intents *explicit* या *implicit* हो सकते हैं—



(i) **Explicit intent**—Explicit intents specify करते हैं कि कौन-सा target app's package name या fully-qualified component class name का प्रयोग करके, intent को satisfy करेगा।

Explicit intent एप्लीकेशन की internal world से जुड़ा होता है, मान लीजिए कि आप किसी activity किसी अन्य activity से connect करना चाहते हैं, तो हम इसको explicit intent से कर सकते हैं, नीचे diagram पहली गतिविधि को दूसरी गतिविधि को बटन पर क्लिक करके जोड़ रही है।



चित्र 1.26

ये intents अपने name से target component को नामित करते हैं और वे आमतौर पर application-internal messages के लिए उपयोग किए जाते हैं—जैसे कि एक activity एक subordinate service शुरू करने या launching sister activity शुरू करने के लिए। उदाहरण के लिए—

```
// Explicit Intent by specifying its class name
Intent i =new Intent(FirstActivity.this,SecondActivity.class);

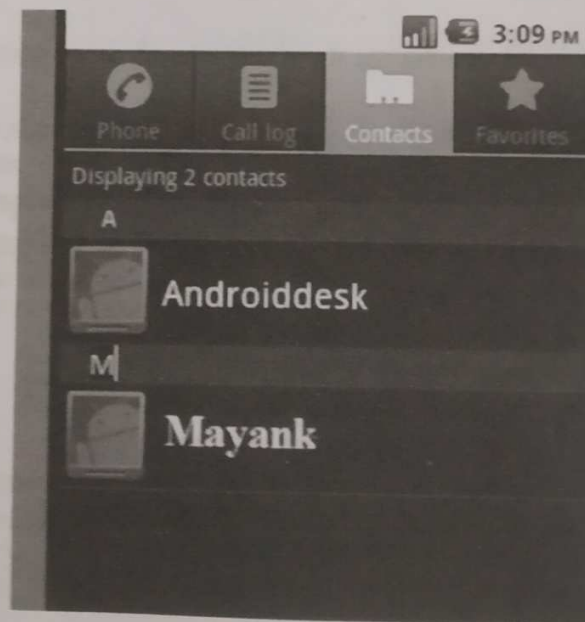
// Starts TargetActivity
startActivity(i);
```

(ii) **Implicit intent**—ये intents किसी specific activity या intent को प्राप्त करने के लिए other component को specify नहीं करते हैं। इसके बजाय, आप action perform करने के लिए एक general action को declare करते हैं, और एन्ड्रॉइड सिस्टम आपके request को किसी activity या other component से match करेगा जो requested action को handle कर सकता है।

अन्य application में component को activate करने के लिए अक्सर Implicit intents का उपयोग किया जाता है। उदाहरण के लिए—

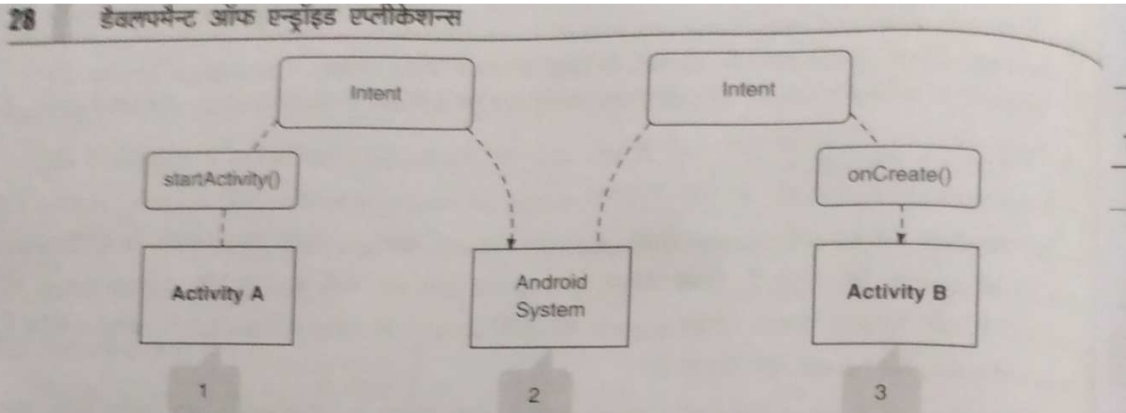
```
Intent read1=new Intent();
read1.setAction(android.content.Intent.ACTION_VIEW);
read1.setData(ContactsContract.Contacts.CONTENT_URI);
startActivity(read1);
```

उपरोक्त कोड नीचे दिखाएँ figure अनुसार result देगा—



चित्र 1.27

चित्र दिखाता है कि activity शुरू करते समय एक intent का उपयोग कैसे किया जाता है। जब intent object किसी specific activity component को explicitly रूप से name देता है, तो सिस्टम तुरन्त उस component को शुरू करता है।



चित्र 1.28

Implicit intent कैसे सिस्टम में अन्य activity शुरू करता है।

- (i) Activity A, action description के साथ intent को create करती है।
- (ii) Android सिस्टम सभी ऐप intent filter खोजता है जो intent से match करे। जब एक match पाया जाता है।
- (iii) सिस्टम अपने onCreate() method को invoke करके और Intent को pass करके matching activity (Activity B) शुरू करता है।

1.10.3. Intent Filters

एक intent filter एक ऐप की manifest file में एक expression है जो उस प्रकार के intents को specify करता है जो component प्राप्त करना चाहते हैं।

उदाहरण के लिए, किसी activity के लिए एक intent filter की घोषणा करके, आप अन्य ऐप्स के लिए एक निश्चित प्रकार के intent से सीधे अपनी activity शुरू करना सम्भव बनाते हैं। इसी तरह, यदि आप किसी activity के लिए कोई intent filter घोषित नहीं करते हैं, तो यह केवल एक explicit intent के साथ शुरू किया जा सकता है।

Intent Filters (<intent-filter>) जो कुछ भी हम manifest फाइल में परिभाषित करते हैं, उन्हें सम्बन्धित app components में nested किया जा सकता है और हम इन three elements का उपयोग करके type of intents को specify कर सकते हैं।

<action>—यह एक intended action को परिभाषित करता है।

<category>—यह एक intended category को परिभाषित करता है।

<data>—यह स्वीकार किए जाने वाले डेटा के प्रकार को परिभाषित करता है।

Intent Filter in Manifest File—नीचे दिखाए गए अनुसार एंड्रॉइड मेनिफेस्ट फाइल (AndroidManifest.xml) में इंटेंट फिल्टर (<intent-filter>) के साथ एक activity को define कराने का कोड दिया गया है।

```

<activity android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
        <data android:mimeType="text/plain"/>
    </intent-filter>
</activity>
  
```

1.11. Android API Levels (Versions & Version Names)

Code name	Version	API level/NDK release
Android11	11	API level 30
Android10	10	API level 29
Pie	9	API level 28
Oreo	8.1.0	API level 27
Oreo	8.0.0	API level 26
Nougat	7.1	API level 25
Nougat	7.0	API level 24
Marshmallow	6.0	API level 23
Lollipop	5.1	API level 22
Lollipop	5.0	API level 21
KitKat	4.4-4.4.4	API level 19
Jelly Bean	4.3x	API level 18
Jelly Bean	4.2x	API level 17
Jelly Bean	4.1x	API level 16
Ice Cream Sandwich	4.0.3-4.0.4	API level 15, NDK 8
Ice Cream Sandwich	4.0.1-1.0.2	API level 14, NDK 7
Honeycomb	3.2x	API level 13
Honeycomb	3.1	API level 12, NDK 6
Honeycomb	3.0	API level 11
Gingerbread	2.3.3-2.3.7	API level 10
Gingerbread	2.3-2.3.2	API level 9, NDK 5
Froyo	2.2.x	API level 8, NDK 4
Eclair	2.1	API level 7, NDK 3
Eclair	2.01	API level 6
Eclair	2.0	API level 5

30 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

Donut	1.6	API level 4, NDK 2
Cupcake	1.5	API level 3, NDK 1
(no code name)	1.1	API level 2
(no code name)	1.0	API level 1

प्रश्नावली

1. Android क्या है?
2. एन्ड्रॉइड आर्किटेक्चर क्या है?
3. Android के फायदे क्या हैं?
4. APK प्रारूप क्या है?
5. Android फ्रेमवर्क का वर्णन करें।
6. Dalvik Virtual Machine क्या है?
7. Android के कोर बिल्डिंग ब्लॉक क्या हैं?
8. Android के कोड नाम क्या हैं?
9. Android में गतिविधि क्या है?
10. एन्ड्रॉइड Activity Life Cycle के तरीके क्या हैं?
11. intent क्या है?
12. explicit Intent क्या है?
13. implicit Intent क्या है?
14. Android में व्यूग्रुप क्या है?
15. Android कार्यक्रम में View Element की पहचान कैसे की जाती है?
16. Android में impotent फोल्डरों की सूची दें।
17. Android में एक सामग्री प्रदाता क्या है?
18. Android में क्या notification उपलब्ध है?



वातावरण सेटअप तथा बुनियादी परियोजना संरचना (Environment Setup and Basic Project Structure)

Setting up development environment Android, Manifest.xml, Gradle, Uses-permission & uses-sdk, Resources & R.java, Assets, Layouts & Drawable Resources, First sample Application, Launching emulator, Editing emulator settings, Emulator shortcuts, Log cat usage, Introduction to DDMS, Hello World App, **Creating your first project, The manifest file, Layout resource**, Running your app on Emulator, Debugging the Android App.

2.1. Setting up Development Environment Android

आप निम्न में से किसी भी ऑपरेटिंग सिस्टम पर अपना Android एप्लिकेशन डेवलपमेंट शुरू कर सकते हैं—

- (i) Microsoft Windows
- (ii) Mac OS
- (iii) Linux

आपको अपने ऐप्स विकसित करने के लिए कुछ वातावरण की आवश्यकता है। इसलिए, आपको development विकास के लिए Android Environment को सेटअप करने के लिए आवश्यक tools, SDK और setup फाइलों को डाउनलोड करना होगा। ये आवश्यक उपकरण freely उपलब्ध हैं और इन्हें वेब से डाउनलोड किया जा सकता है। Android एप्लिकेशन विकसित करने के लिए आवश्यक Tool और वितरण के नाम निम्नलिखित हैं—

- (i) Java Development Kit
- (ii) Android SDK

2.1.1. Java Development Kit (JDK)

Android SDK, जावा डेवलपमेंट किट (JDK) पर काम करता है। इसलिए, पहले आपके सिस्टम में JDK इंस्टॉल होना चाहिए। आप Oracle's Java साइट से जावा JDK का नवीनतम संस्करण डाउनलोड कर सकते हैं। आपको डाउनलोड की गई फाइलों में JDK install करने के लिए निर्देश मिलेगा, सेटअप को install करने और कॉन्फिगर करने के लिए दिए गए निर्देशों का पालन करें। एक बार जब आप इसको कर लेते हैं, तो आप अगले चरण के लिए जा सकते हैं।

2.1.2. Android SDK/Android Studio/Eclipse

इसके बाद एंड्रॉयड SDK आता है। इसमें एंड्रॉयड के रिलीज किए गए संस्करणों के प्रत्येक API Level के लिए libraries, debugger, एक emulator, document, sample codes और tutorials शामिल हैं। आप <https://developer.android.com> से पूरा Android Development Environment बंडल (Android Studio + SDK) डाउनलोड कर सकते हैं।

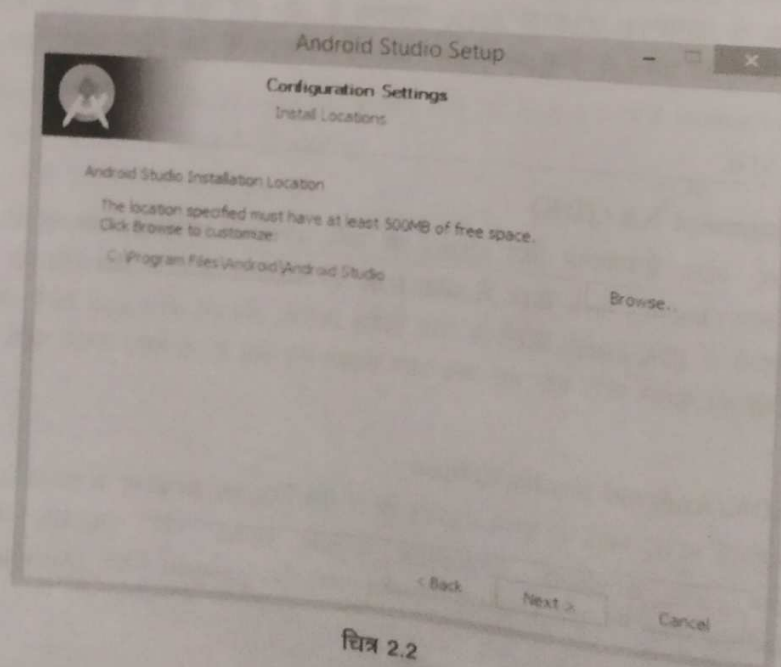
2.1.3. Setup and Installation of Android Studio

एक बार जब आप JDK install कर लेते हैं। Android Studio IDE सेटअप फाइल डाउनलोड करें। जब फाइल डाउनलोड हो जाये, तो Android स्टूडियो की .exe फाइल (Setup File) पर डबल क्लिक करें। Next बटन पर क्लिक करें।



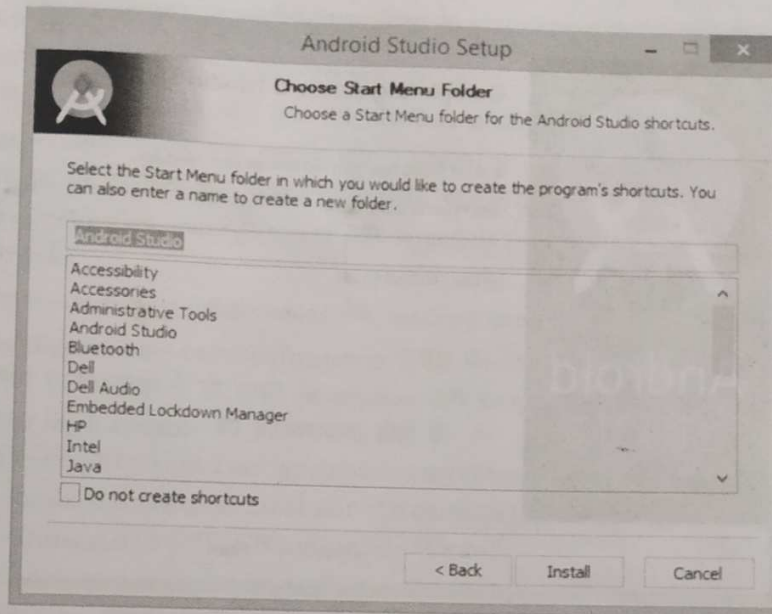
चित्र 2.1

यह फाइल को सेटअप के लिए prepare करेगा और आपसे उस location के लिए पूछेगा जहाँ आप इसे install करना चाहते हैं, जैसा कि नीचे दी गई image में दिखाया गया है—



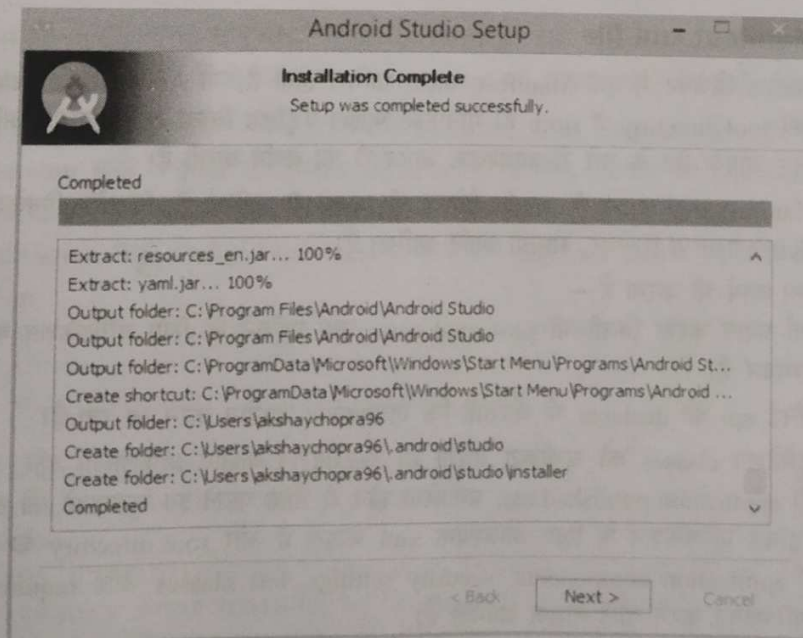
चित्र 2.2

एक उपयुक्त location का चयन करें और नेक्स्ट बटन पर क्लिक करें।



चित्र 2.3

Install बटन पर क्लिक करें। यह नीचे दिखाए image के अनुसार installing करना शुरू कर देगा।



Next बटन पर क्लिक करें।



चित्र 2.5

एंड्रॉइड स्टूडियो को सीधे इंस्टॉल किया गया है और यह उपयोग के लिए तैयार है। फिनिश बटन पर क्लिक करें और एंड्रॉइड स्टूडियो खुल जाएगा।

2.2. AndroidManifest.xml file

एंड्रॉइड के प्रत्येक प्रोजेक्ट में एक Manifest फाइल शामिल होती है, जो AndroidManifest.xml है, जो अपने प्रोजेक्ट hierarchy की root directory में store है। मैनिफेस्ट फाइल एंड्रॉइड बिल्ड टूल, एंड्रॉइड ऑपरेटिंग सिस्टम और Google Play के लिए आपके ऐप के बारे में आवश्यक जानकारी का वर्णन करती है।

“AndroidManifest.xml फाइल में आपके पैकेज की जानकारी शामिल है, जिसमें एप्लिकेशन के components जैसे activity, service”, प्रसारण रिसीवर, सामग्री आदि शामिल हैं।

यह कुछ अन्य कार्य भी करता है—

- (i) अनुमतियाँ प्रदान करके किसी भी protected parts तक पहुँचने के लिए एप्लिकेशन को protect करने के लिये जिम्मेदार है।
- (ii) यह एंड्रॉइड api को declares भी करता है कि एप्लिकेशन उपयोग करने जा रहा है।
- (iii) यह इंस्ट्रुमेंटेशन classes को सूचीबद्ध करता है। इंस्ट्रुमेंटेशन कक्षाएँ प्रोफाइलिंग और अन्य जानकारी प्रदान करती हैं। application published etc. प्रकाशित होने से ठीक पहले इन सूचनाओं को हटा दिया जाता है। यह सभी एंड्रॉइड एप्लिकेशन के लिए आवश्यक xml फाइल है और root directory के अन्दर स्थित है। एप्लिकेशन के requirements को परिभाषित करने वाले नोड्स शामिल हैं।

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.mayank.hello"
    android:versionCode="1"
```

```
android:versionName="1.0">
[...manifest nodes...]
```

</manifest>—manifest is the root element of the AndroidManifest.xml file. It has **package** attribute that describes the package name of the activity class.

मुख्य रूप से उपयोग किए जाने वाले कुछ उप-नोड (sub-node) tag इस प्रकार हैं—

(i) **uses-sdk**—इसका उपयोग एक न्यूनतम और अधिकतम SDK संस्करण को परिभाषित करने के लिए किया जाता है जो एक डिवाइस पर उपलब्ध होना चाहिए ताकि हमारा एप्लिकेशन ठीक से काम करे, और target SDK जिसके लिए इसे क्रमशः minSdkVersion, maxSdkVersion और targetSdkVersion attributes के combination का उपयोग करके तैयार किया गया है।

(ii) **<uses-sdk**—android:minSdkVersion="8" android:targetSdkVersion="15"/>

(iii) **uses-configuration**—uses-configuration नोड्स का उपयोग input mechanisms के combination को specify करने के लिए किया जाता है जो हमारे permission द्वारा supported हैं। यह उन games के लिए उपयोगी है जिनके लिए particular input controls की आवश्यकता होती है।

```
<uses-configuration android:reqTouchScreen="finger"
    android:reqNavigation="trackball"
    android:reqHardKeyboard="true"
    android:reqKeyboardType="qwerty"/>
```

1. **Uses-features**—इसका उपयोग यह specify करने के लिए किया जाता है कि कौन-सा हार्डवेयर आपकी एप्लिकेशन की आवश्यकता है। यह हमारे एप्लिकेशन को एक device पर installed होने से रोकेगा जिसमें NFC हार्डवेयर जैसे हार्डवेयर का एक आवश्यक parts शामिल नहीं है—

```
<uses-feature android:name="android.hardware.nfc"/>
```

2. **Application**—एप्लिकेशन मैनिफेस्ट का sub element है। इसमें namespace declaration शामिल है। इस element में कई sub elements होते हैं जो एप्लिकेशन component की घोषणा करते हैं, जैसे activity आदि।

आमतौर पर इस element पर इस्तेमाल किए जाने वाले attributes, icon, label, theme आदि हैं।

(a) **android:icon** सभी एन्ड्रॉइड एप्लिकेशन components के लिए आइकन को represent करता है।

(b) **android:label** सभी एप्लिकेशन components के लिए डिफॉल्ट लेबल के रूप में काम करता है।

(c) **android:theme** सभी Android activities के लिए एक common theme का प्रतिनिधित्व करता है।

<application

```
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme">
```

<activity

```
    android:name=".MainActivity"
    android:label="@string/title_activity_main">
```

<intent-filter>

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

</activity>

</application>

Activity

Activity application का subelement है और इस activity को AndroidManifest.xml फाइल में परिभाषित किया जाना चाहिए। इसमें कई attribute हैं जैसे label, name, theme, launchMode आदि।

- (i) **android:label** एक लेबल का प्रतिनिधित्व करता है यानी स्क्रीन पर प्रदर्शित होता है।
- (ii) **android:name** activity class के लिए एक नाम का प्रतिनिधित्व करता है। यह आवश्यक attribute है।

Intent-filter

Intent-filter, activity का sub-element है जो उस activity के प्रकार के intent को describes करता है जिस पर activity, service या broadcast receiver प्रतिक्रिया कर सकता है।

Action

यह intent-filter के लिए एक action जोड़ता है। intent-filter में कम-से-कम एक action element होना चाहिए।

Category

यह intent-filter के लिए एक category का नाम जोड़ता है।

Example : AndroidManifest.xml file

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.mayank.hello"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15"/>
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:label="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

2.3. Gradle

gradle multi-language सॉफ्टवेयर development के लिए एक build automation टूल है। यह testing deployment और publishing के लिए, compilation और पैकेजिंग के कार्यों में development process को control करता है।

Android Studio, Gradle उपयोग निर्माण प्रक्रिया को स्वचालित और प्रबंधित करने के लिए, जबकि Gradle आपको flexible custom build configurations परिभाषित करने की अनुमति देते हैं। आपके App के सभी versions के लिए भागों का पुनः उपयोग करते हुए प्रत्येक build configuration code और resources के अपने स्वयं सेट को परिभाषित कर सकता है।

Gradle की आवश्यकता क्यों है?

प्रत्येक एन्ड्रॉइड प्रोजेक्ट को .java और .xml फाइलों से apk फाइल generate करने के लिए एक Gradle की आवश्यकता होती है। सीधे शब्दों में कहें, एक Gradle सभी स्रोत फाइलों (जावा और XML) को लेता है और उपयुक्त उपकरण लागू करता है, जावा फाइलों को dex फाइलों में परिवर्तित करता है और सभी फाइल को compress करके एक apk फाइल बनाता है।

जैसा कि एन्ड्रॉइड स्टूडियो Gradle सिस्टम के साथ आता है, प्रोजेक्ट को बनाने के लिए अतिरिक्त रनटाइम सॉफ्टवेयर को install करने की आवश्यकता नहीं है। जब भी आप एन्ड्रॉइड स्टूडियो में Run बटन पर क्लिक करते हैं, तो एक Gradle कार्य automatically चालू हो जाता है और प्रोजेक्ट का निर्माण शुरू कर देता है और gradle अपना कार्य पूरा करने के बाद, App, AVD या कनेक्टेड डिवाइस में चलना शुरू कर देता है।

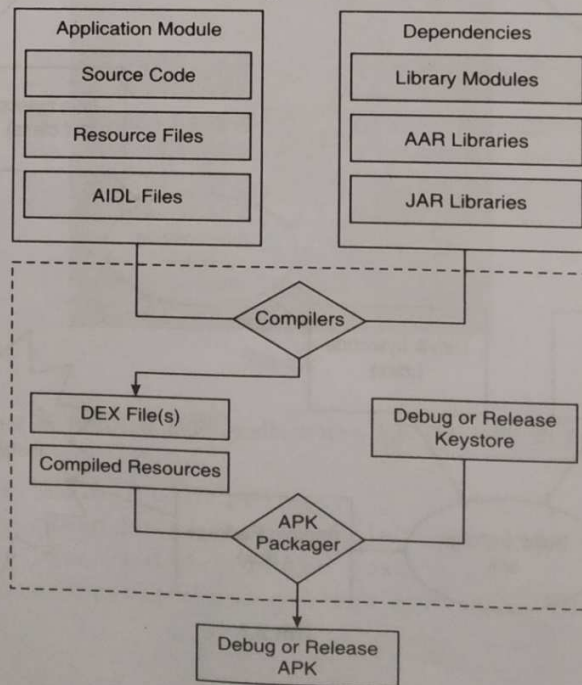
Gradle सिस्टम कंपाइलर, लिंकर आदि नहीं है, लेकिन यह compilation के operation को control और निगरानी करता है, फाइलों को जोड़ता है, परीक्षण मामलों को चलाता है, और अंत में आपके एन्ड्रॉइड एप्लिकेशन के लिए Apk फाइल बनाता है।

हर Android स्टूडियो प्रोजेक्ट के लिए दो बिल्ड ग्रेडल फाइलें हैं, जिनमें से—

- (i) एक एप्लीकेशन के लिए है।
- (ii) प्रोजेक्ट लेवल (मॉड्यूल लेवल) बिल्ड फाइल्स के लिए है।

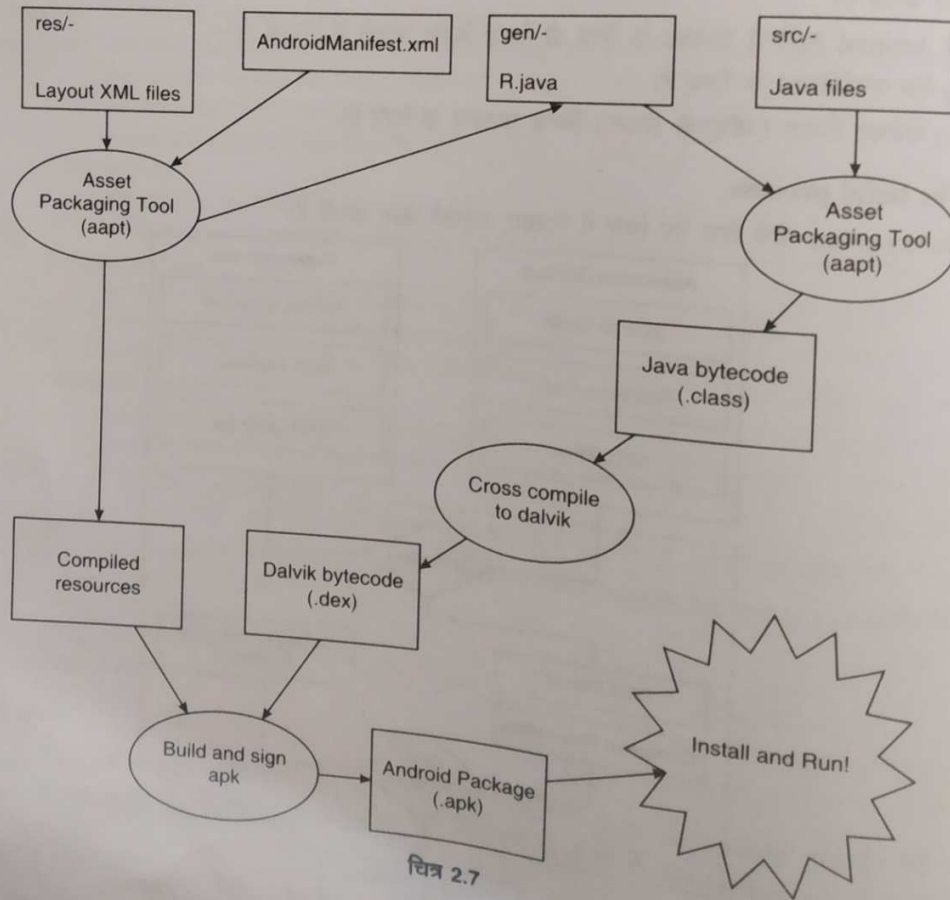
2.3.1. The build process.

Build process नीचे दिए गए चित्र में दिखाए अनुसार काम करती है।



चित्र 2.6

- (i) Build process में कंपाइलर source code, resource, external लाइब्रेरीज JAR फाइलें और AndroidManifest.xml (जिसमें एप्लीकेशन के बारे में मेटा-डेटा होता है) लेता है और उन्हें dex (Dalvik Executable files) फाइलों में परिवर्तित कर देता है, जिसमें bytecode शामिल होता है।
- (ii) APK Packager, DEX फाइलों और resources को combine करके एक ही APK में जोड़ता है। इससे पहले कि आपका App एन्ड्रॉइड डिवाइस में इंस्टॉल और deploy किया जा सके, एपीके पर हस्ताक्षर किए जाने चाहिए।
- (iii) APK Packager डिबग या रिलीज कीस्टोर का उपयोग करके आपके APK पर हस्ताक्षर करता है—
- यदि आप अपने ऐप के debug version का निर्माण कर रहे हैं, यानी आप केवल परीक्षण और प्रोफाइलिंग के लिए ही एक ऐप चाहते हैं, तो पैकर debug keystore के साथ आपके ऐप पर हस्ताक्षर करता है। एन्ड्रॉइड स्टूडियो automatically रूप से डिबग कीस्टोर के साथ new projects को कॉन्फिगर करता है।
 - यदि आप अपने ऐप के release version का निर्माण कर रहे हैं, जिसे आप externally रूप से रिलीज करने का इरादा रखते हैं, तो पैकर आपके ऐप को release keystore के साथ साइन करता है।
 - आपके अंतिम एपीके को जेनरेट करने से पहले, packager एक डिवाइस पर चलने के दौरान कम मेमोरी का उपयोग करने के लिए आपके ऐप को optimize करने के लिए zipalign tool का उपयोग करता है।



चित्र 2.7

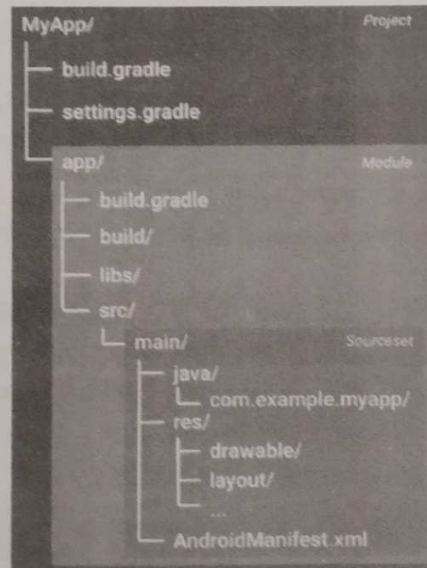
Build process के अंत में, आपके पास अपने ऐप का एक debug APK है जिसे आप external users को deploy, test या रिलीज करने के लिए उपयोग कर सकते हैं। जिसको Install और Run किया जाता है।

2.3.2. Gradle files का detail Description निम्नलिखित हैं—

(i) **setting.gradle**—setting.gradle (Gradle setting) फाइल का उपयोग आपके ऐप में उपयोग किए जाने वाले सभी मॉड्यूल को specify करने के लिए किया जाता है।

(ii) **build.gradle (project level)**—Top level (मॉड्यूल) build.gradle फाइल project level build फाइल है, जो प्रोजेक्ट Level पर build configurations को परिभाषित करता है। यह फाइल एन्ड्रॉइड एप्लिकेशन प्रोजेक्ट के सभी मॉड्यूल पर कॉन्फिगरेशन लागू करती है।

(iii) **build.gradle (application level)**—Application level build.gradle file Android प्रोजेक्ट के प्रत्येक मॉड्यूल में स्थित है। इस फाइल में एक specific एप्लिकेशन मॉड्यूल के लिए आपके पैकेज का नाम ApplicationID, version name apk version, version code, minimum और target SDK शामिल हैं। जब आप external libraries.jar फाइलों को शामिल नहीं कर रहे हैं, को शामिल करते हैं, तो आपको एप्लिकेशन में निर्भरता के रूप में उन्हें अपनी project में शामिल करने के लिए ऐप Level के gradle file में इसका mention करना होगा।



चित्र 2.8

Command line tool के माध्यम से अपने gradle system शुरू कर सकते हैं। Following commands इसके लिए उपयोग किए जाते हैं—

- `./gradlew build`—(build project)
- `./gradlew clean build`—(build project complete scratch)
- `./gradlew clean build`—(run the test)
- `./gradlew wrapper`—(to see all the available tasks)

2.4. ANDROID PERMISSIONS

एन्ड्रॉइड सिस्टम Permission का उपयोग करता है जो किसी भी एप्लिकेशन में क्या कर सकते हैं और क्या नहीं कर सकते, को manage करता है।

2.4.1. <uses-permission>

Permission है, जो आप <uses-permission> के माध्यम से अनुरोध करते हैं। एक सिस्टम permission को Specifies करता है जिसे user को App को सही ढंग से संचालित करने के लिए देना होगा। एप्लिकेशन इंस्टॉल होने पर उपयोगकर्ता द्वारा अनुमतियाँ दी जाती हैं, न कि जब वह चल रही हो।

Syntax :

```
<uses-permission android:name="string"/>
```

Contained in :

```
<manifest>
```

Attributes :

```
android:name
```

permission का नाम है। यह एप्लिकेशन द्वारा परिभाषित permission हो सकती है < permission > element, किसी अन्य एप्लिकेशन द्वारा परिभाषित एक अनुमति, या "android.permission.CAMERA" या "android.permission.READ_CONTACTS" जैसे मानक सिस्टम अनुमतियों में से एक। जैसा कि यह उदाहरण दिखाते हैं, एक अनुमति नाम में आमतौर पर prefix के रूप में पैकेज का नाम शामिल होता है।

2.4.2. <uses-permission-sdk-23>

एन्ड्रॉइड sdk-23 और इसके बाद के संस्करण के लिए, आपके पास runtime पर permission का अनुरोध करने का विकल्प है, लेकिन installation के समय permission उनके डिफॉल्ट स्थिति में होगी और उपयोगकर्ता को इंस्टॉलेशन के लिए प्रेरित नहीं किया जाएगा।

SDK 22 और below, user को permission के लिए installation पर संकेत दिया गया है। जैसा कि कुछ permission उपयोगकर्ता को suspicious या doubtful लग सकती है, आप SDK 22 के लिए इनका अनुरोध नहीं कर सकते हैं और below जैसा कि आप इस बात का explanation नहीं दे सकते हैं कि आपको उनकी पहले से आवश्यकता क्यों है, इसलिए <uses-permission-sdk-23>tag का उपयोग करता है।

syntax:

```
<uses-permission-sdk-23 android:name="string"/>
```

Contained in :

```
<manifest>
```

Attributes :

```
android:name
```

permission का नाम है। यह एप्लिकेशन द्वारा परिभाषित permission हो सकती है < permission > element, किसी अन्य एप्लिकेशन द्वारा परिभाषित एक अनुमति, या "android.permission.CAMERA" या "android.permission.READ_CONTACTS" जैसे मानक सिस्टम अनुमतियों में से एक। जैसा कि यह उदाहरण दिखाते हैं, एक अनुमति नाम में आमतौर पर prefix के रूप में पैकेज का नाम शामिल होता है।

2.4.3. <uses-sdk>

आपको API Level integer के माध्यम से Android प्लेटफॉर्म के एक या अधिक संस्करणों के साथ एक application's compatibility व्यक्त करने देता है। किसी एप्लिकेशन द्वारा व्यक्त किया गया API Level किसी दिए गए Android सिस्टम के API Level की तुलना में होगा, जो विभिन्न Android उपकरणों के बीच भिन्न हो सकता है।

अपने नाम के बावजूद, इस element का उपयोग API Level को specify करने के लिए किया जाता है, न कि SDK (सॉफ्टवेयर डेवलपमेंट किट) या एन्ड्रॉइड प्लेटफॉर्म के version number को specify करने के लिए। API Level हमेशा single integer होता है। आप इसके सम्बन्ध Android संस्करण संख्या से API Level प्राप्त नहीं कर सकते हैं।

syntax :

```
<uses-sdk android:minSdkVersion="integer"
  android:targetSdkVersion="integer"
  android:maxSdkVersion="integer"/>
```

contained in:

<manifest>

Attributes :

android:minSdkVersion

एप्लिकेशन को चलाने के लिए आवश्यक minimum API level को specify करने वाला पूर्णांक। एन्ड्रॉइड सिस्टम उपयोगकर्ता को एप्लिकेशन को इंस्टॉल करने से रोक देगा यदि सिस्टम का API level इस attribute में specified value से कम है। आपको हमेशा इस attribute को declare करना चाहिए।

android:targetSdkVersion

एक पूर्णांक API Level को नामित करता है जो application को target करता है।

android:maxSdkVersion

एक integer जो अधिकतम API Level नामित करता है जिस पर एप्लिकेशन को चलाने के लिए डिजाइन किया गया है।

2.4.4. अब यह question arise होता है ...API Level क्या है?

API Level एक integer value है जो विशिष्ट रूप से एन्ड्रॉइड प्लेटफॉर्म version द्वारा पेश की गई फ्रेमवर्क API संशोधन की पहचान करता है।

फ्रेमवर्क API में निम्न शामिल हैं—

- (i) Packages और classes का एक मुख्य group
- (ii) XML elements और attributes का एक सेट जो manifest file declare करने के लिए
- (iii) XML elements और attributes का एक सेट जो resources की घोषणा करने और उन तक पहुँचने के लिए
- (iv) Intents का एक सेट
- (v) Permissions का एक सेट जो कि आवेदन का अनुरोध कर सकता है, साथ ही सिस्टम में शामिल अनुमति लागू भी कर सकता है।

2.5. Resources & R.java

2.5.1. App Resources

कई और आइटम हैं जो आप एक अच्छा एन्ड्रॉइड एप्लिकेशन बनाने के लिए उपयोग करते हैं। “Resources” अतिरिक्त फाइलें और static content होते हैं जिसको आपका कोड उपयोग करता है, जैसे कि bitmaps, layout definitions, user interface strings, animation instructions और बहुत कुछ।

इन resources को हमेशा प्रोजेक्ट की res/directory के तहत विभिन्न sub-directories में अलग-अलग बनाया रखा जाता है।

1. Organize resource in Android Studio

आपको अपनी project के res/directory के एक विशिष्ट subdirectory में प्रत्येक प्रकार के resource रखने चाहिए। उदाहरण के लिए, यहाँ एक simple project के लिए file hierarchy है—

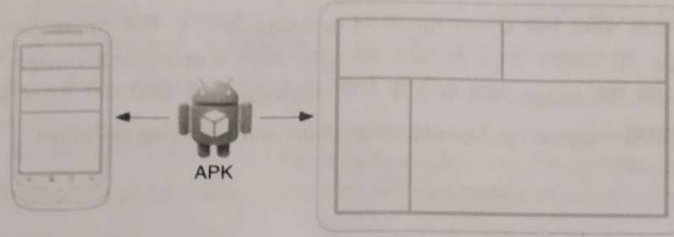
```
MyProject/
  app/
    manifest/
      AndroidManifest .xml
    java/
      MainActivity.java
    res/
      drawable/
        icon.png
      layout/
        activity_main.xml
        info.xml
      values/
        strings.xml
```

Directory	Resource Type
color/	XML फाइलें जो color की एक state list को परिभाषित करती हैं।
anim/	XML फाइलें जो tween एनिमेशन को परिभाषित करती हैं।
animator/	XML फाइलें जो property animations को परिभाषित करती हैं।
drawable/	Bitmap files
mipmap/	अलग-अलग launcher icon densities के लिए बनाने योग्य फाइलें।
layout/	XML फाइलें जो एक यूजर इंटरफेस लेआउट को परिभाषित करती हैं।
menu/	XML फाइलें जो ऐप मेन्यू को परिभाषित करती हैं।
raw/	अपने RAW form में save के लिए Arbitrary फाइलें
values/	XML फाइलें जिनमें simple values होते हैं, जैसे कि strings, integers और color

2. Alternative resources—हर ऐप को specific डिवाइस कॉन्फिगरेशन का समर्थन करने के लिए alternative resources प्रदान करना चाहिए। दो अलग-अलग डिवाइस, प्रत्येक अलग-अलग लेआउट resources का उपयोग करते हैं।

resources के सेट के लिए configuration-specific विकल्प specify करने के लिए, निम्न चरणों का पालन करें—

(i) फॉर्म में `<resource_name> - <config-qualifier>` के द्वारा res/ नाम की एक नई डायरेक्टरी बनाएँ। यह resource_name उपरोक्त तालिका में उल्लिखित resources में से कोई भी होगा, जैसे कि layout, drawable आदि। qualifier एक individual कॉन्फिगरेशन specify करेगा जिसके लिए इन resources का उपयोग किया जाना है।



चित्र 2.9

- इस नई directory में सम्बन्धित वैकल्पिक resources को save करें। resource फाइलों को बिल्कुल वही नाम दिया जाना चाहिए जैसा कि नीचे दिए गए उदाहरण में दिखाया गया है, लेकिन इन फाइलों में alternative के लिए विशिष्ट content होगी। उदाहरण के लिए हालांकि image फाइल का नाम समान होगा लेकिन high रिजॉल्यूशन स्क्रीन के लिए इसका रिजॉल्यूशन high होगा।

MyProject?

```

app/
  manifest/
    AndroidManifest.xml
  java/
    MainActivity.java
  res/
    drawable/
      icon.png
      background.png
    drawable-hdpi/
      icon.png
      background.png
    layout/
      activity_main.xml
      info.xml
    values/
      strings.xml
  
```

3. Accessing Resources.—आपके एप्लिकेशन डेवलपमेंट के दौरान आपको defined resources को अपने कोड में, या अपने लेआउट XML फाइलों में एक्सेस करना होगा। निम्नलिखित बताता है कि दोनों scenarios में अपने resources का उपयोग कैसे करें।

(i) **Accessing Resources in Code**—जब आपका एन्ड्रॉइड एप्लिकेशन compile किया जाता है, तो R class उत्पन्न होता है, जिसमें आपके res/directory में उपलब्ध सभी resource के लिए resource ID होती है। आप sub-directory और resource name या directly resource ID का उपयोग करके उस resource तक पहुँचने के लिए R class का उपयोग कर सकते हैं।

(a) Drawable Resource उदाहरण—

Res/drawable/myimage.png तक पहुँचने और एक ImageView सेट करने के लिए आप निम्नलिखित कोड का उपयोग करेंगे—

```

ImageView imageView = (ImageView)
    findViewById(R.id.myimageview);
imageView.setImageResource(R.drawable.myimage);
  
```

यहाँ कोड की पहली पंक्ति एक layout file में id myimageview के साथ ImageView परिभाषित करने के लिए R.id.myimageview का उपयोग करती है। कोड की दूसरी पंक्ति R.drawable.myimage का उपयोग करता है, जिसका नाम चित्रण के साथ एक image प्राप्त करने के लिए, drawable सब-डायरेक्टरी में/res के तहत उपलब्ध है।

(b) layout उदाहरण—layout res/layout/activity_main.xml following definition :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView"/>
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a Button"/>
</LinearLayout>
```

यह एप्लिकेशन कोड इस गतिविधि के लिए इस लेआउट को लोड करेगा, onCreate () विधि में निम्नानुसार है—

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

(ii) Accessing Resources in XML—निम्नलिखित resource पर विचार करें XML res/values/strings.xml फाइल जिसमें एक color resource और a string resource शामिल है—

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="opaque_red">#f00</color>
    <string name="hello">Hello!</string>
</resources>
```

अब आप निम्न लेआउट फाइल में इन संसाधनों का उपयोग टेक्स्ट कलर और टेक्स्ट स्ट्रिंग को निम्नानुसार सेट करने के लिए कर सकते हैं—

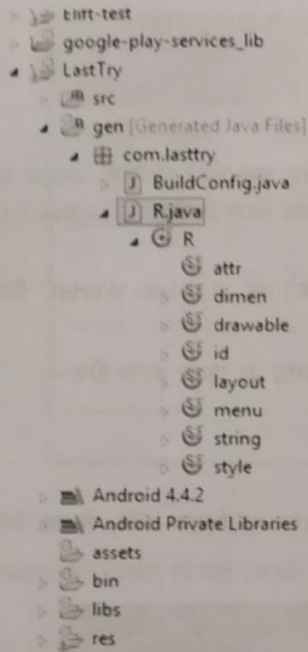
```
<?xml version="1.0" encoding="utf-8"?>
<EditText
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:textColor="@color/opaque_red"
    android:text="@string/hello" />
```

2.5.2. Android R.java file

Android R.java, aapt (Android Asset पैकेजिंग टूल) द्वारा एक ऑटो-जनरेट की गई फाइल है जिसमें res/directory के सभी resource के लिए resource ID होती है। यदि आप activity_main.xml फाइल में कोई भी

component बनाते हैं, तो सम्बन्धित component के लिए ID automatically रूप से इस फाइल में बनाई जाती है। component पर किसी भी क्रिया को करने के लिए इस ID का उपयोग activity source file में किया जा सकता है।

Android R.java Path: /gen/com/foo/R.java



```
package com.example.helloandroid;
public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int ic_launcher=0x7f020000;
    }
    public static final class id {
        public static final int menu_settings=0x7f070000;
    }
    public static final class layout {
        public static final int activity_main=0x7f030000;
    }
    public static final class menu {
        public static final int activity_main=0x7f060000;
    }
    public static final class string {
        public static final int app_name=0x7f040000;
        public static final int hello_world=0x7f040001;
    }
}
```



```
public static final int menu_settings=0x7f040002;
}
public static final class style {
public static final int AppBaseTheme=0x7f050000;
public static final int AppTheme=0x7f050001;
}
}
```

2.6. First Sample Application

एन्ड्रॉइड एसडीके का उपयोग करके अपना पहला उदाहरण लिखना शुरू करने से पहले, आपको यह सुनिश्चित करना होगा कि आपने अपने एन्ड्रॉइड डेवलपमेंट वातावरण को ठीक से सेट किया है जैसा कि section 2.1 में समझाया गया है।

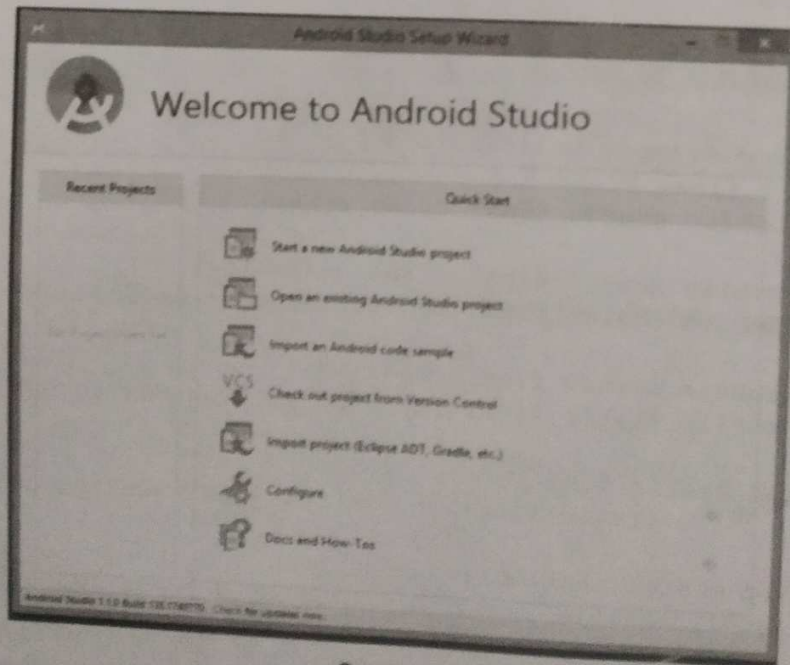
तो आए हम एक simple एन्ड्रॉइड एप्लिकेशन लिखने के लिए आगे बढ़ें जो "Hello World!" प्रिंट करेगा।

Hello Android Examples :

Hello android application बनाने के लिए नीचे बताए गए 3 चरणों का पालन करना होगा—

1. Create the new project
2. Write the message (optional)
3. Run the android application

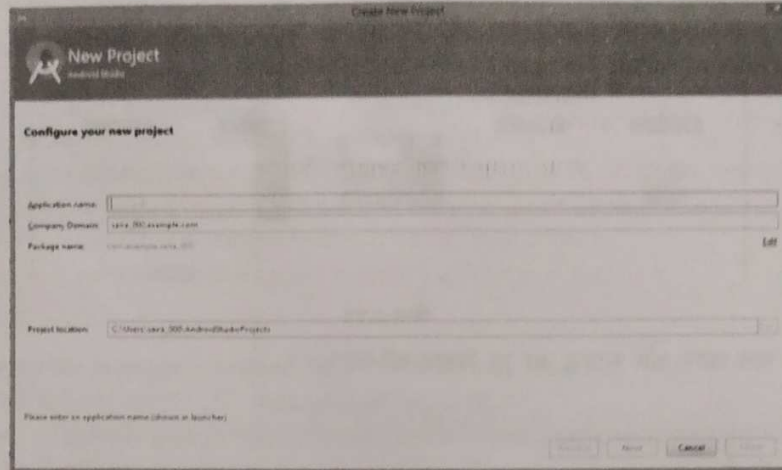
1. Create the new project—जब आप एन्ड्रॉइड स्टूडियो आइकन पर क्लिक करेंगे, तो यह नीचे दिखाए गए अनुसार स्क्रीन दिखेगा।



चित्र 2.10

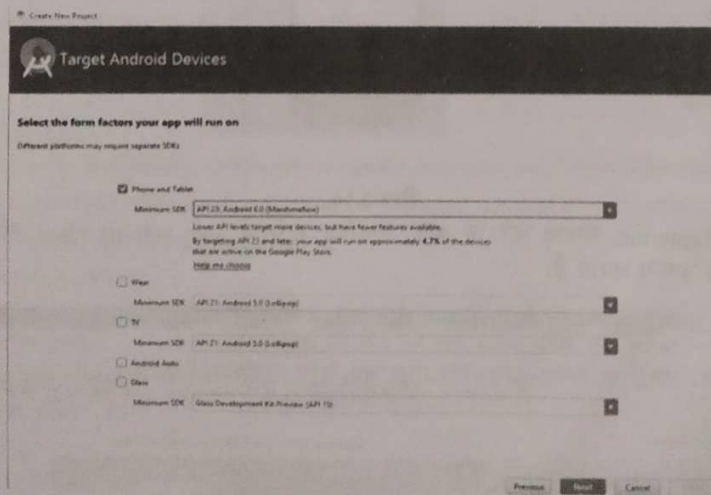
को

आप start a new android studio project पर क्लिक करके अपना एप्लिकेशन डेवलपमेंट शुरू कर सकते हैं। नए इंस्टॉलेशन फ्रेम में प्रोजेक्ट का नाम, पैकेज की जानकारी और प्रोजेक्ट का location पूछेगा फिर next पर क्लिक करें।



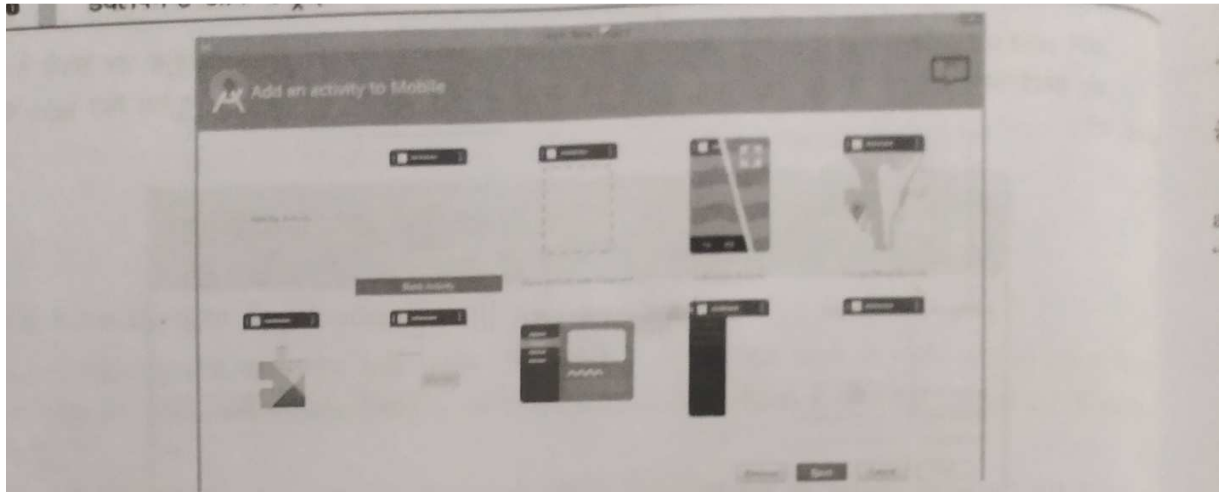
चित्र 2.11

एप्लिकेशन नाम लिखने के बाद आपको फॉर्म फैक्टर को सेलेक्ट करना होगा।



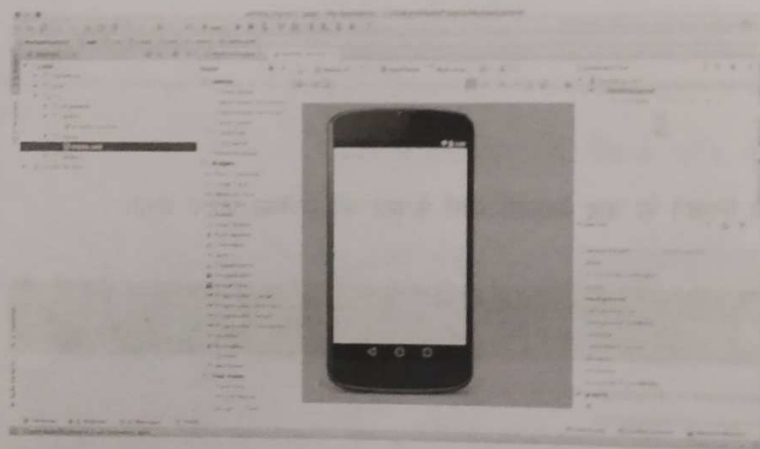
चित्र 2.12

Next level में मोबाइल के लिए activity का चयन करना चाहिए, यह Applications के लिए डिफॉल्ट लेआउट को specify करता है।



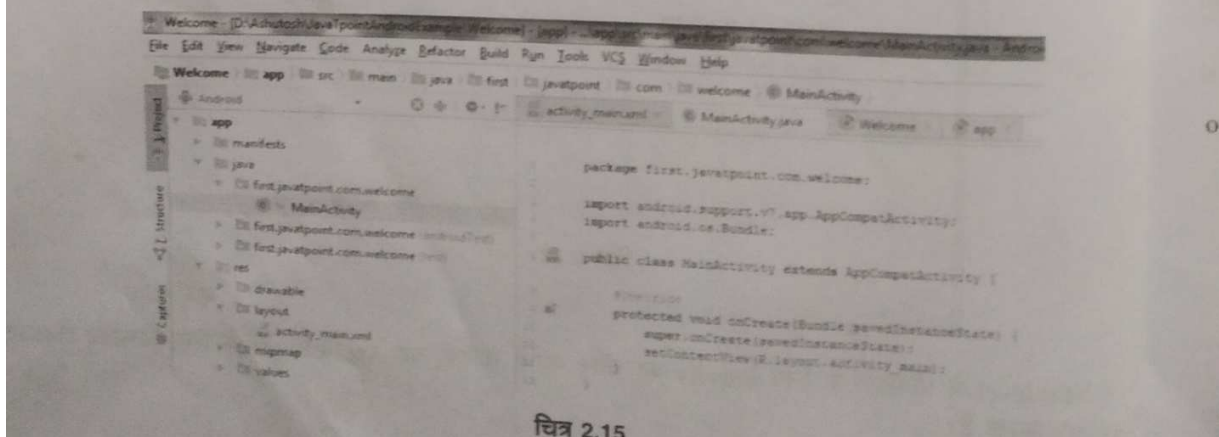
चित्र 2.13

Activity नाम प्रदान और समाप्त करें पर क्लिक करें—



चित्र 2.14

Activity Configuration समाप्त करे के बाद, एन्ड्रॉइड स्टूडियो auto activity class और अन्य आवश्यक कॉन्फिगरेशन फाइलों को उत्पन्न करता है।



चित्र 2.15

अब एक android प्रोजेक्ट create किया गया है। आप एन्ड्रॉइड प्रोजेक्ट को देख सकते हैं और simple प्रोग्राम देख सकते हैं, यह इस तरह दिखता है।

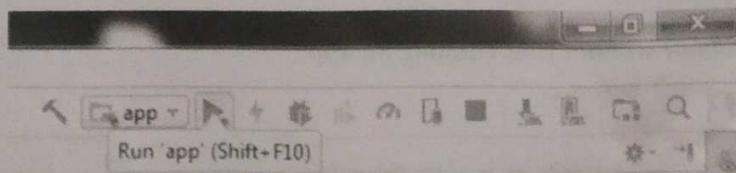
1. Main Activity File—Main activity code एक जावा फाइल MainActivity.java है। यह actual application फाइल है जो अंततः एक Dalvik executable में परिवर्तित हो जाती है और आपके आवेदन runs करती है “Hello Word” application के लिए application wizard द्वारा उत्पन्न डिफॉल्ट कोड निम्नलिखित है—

```
package com.example.helloworld;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

2. Write the message—Android स्टूडियो ऑटो activity_main.xml फाइल के लिए कोड बनाता है।

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.myapplication">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportRtl="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>
    </application>
</manifest>
```

3. Run the android application—To run the android application, click the run icon on the toolbar or simply press Shift + F10.



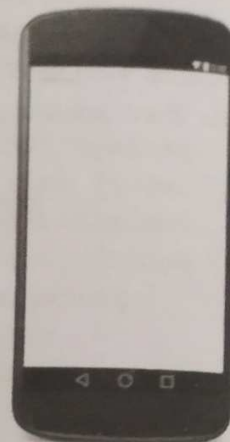
चित्र 2.16

2.7. Android Emulator

एन्ड्रॉइड Emulator आपके कम्प्यूटर पर एन्ड्रॉइड डिवाइस को simulates करता है ताकि आप प्रत्येक physical डिवाइस की आवश्यकता के बिना विभिन्न उपकरणों और एन्ड्रॉइड API levels पर अपने एप्लिकेशन का परीक्षण कर सकें।

एम्प्यूलेटर एक वास्तविक एन्ड्रॉइड डिवाइस की लगभग सभी क्षमताओं को प्रदान करता है। आप आने वाले फोन कॉल और टेक्स्ट संदेशों को simulate कर सकते हैं, डिवाइस का स्थान specify कर सकते हैं, विभिन्न नेटवर्क speeds को simulate कर सकते हैं, रोटेशन और अन्य हार्डवेयर sensors को simulate कर सकते हैं, Google Play Store तक पहुँच सकते हैं, और बहुत कुछ।

एम्प्यूलेटर पर अपने ऐप का Test करना physical device पर ऐसा करने की तुलना में कुछ तरीके से faster और easier है। एम्प्यूलेटर विभिन्न Android फोन, टैबलेट, Android TV के लिए पूर्वनिर्धारित कॉन्फिगरेशन के साथ आता है।



चित्र 2.17

2.7.1. Android Virtual Devices (AVD)

एन्ड्रॉइड एम्प्यूलेटर का प्रत्येक instance एन्ड्रॉइड वर्जन और AVD का उपयोग करता है, जो कि एन्ड्रॉइड वर्जन और simulated डिवाइस की हार्डवेयर विशेषताओं को specify करता है। अपने ऐप को प्रभावी ढंग से test के लिए, आपको एक AVD बनाना चाहिए जो प्रत्येक डिवाइस को models करे जिस पर आपका ऐप चलाने के लिए डिजाइन किया गया है। AVDs बनाने और manage करने के लिए AVD manage का उपयोग करें।

प्रत्येक AVD, user data, SD card और इसी तरह के लिए अपने private storage के साथ एक independent device के रूप में कार्य करता है। डिफॉल्ट रूप से, एम्प्यूलेटर उस AVD के लिए specific directory में user data SD card data और cache को store करता है। जब आप एम्प्यूलेटर लॉन्च करते हैं, तो यह AVD directory से user data और SD card data को लोड करता है।

2.7.2. Install the emulator

एन्ड्रॉइड स्टूडियो install करते समय एन्ड्रॉइड एम्प्यूलेटर को भी install किया जाता है। एन्ड्रॉइड एम्प्यूलेटर को install करने के लिए, SDK Manager के SDK Tools tab में एन्ड्रॉइड एम्प्यूलेटर component का चयन करें।

Run an Android app on the Emulator

(i) एन्ड्रॉइड स्टूडियो में, एक एन्ड्रॉइड वर्चुअल डिवाइस (AVD) एमुलेटर है जो आपके ऐप को इंस्टाल और चलता है।

(ii) Toolbar में, AVD का चयन करें जिसे आप अपने ऐप को target डिवाइस डॉप-डाउन मेनू से चलाना चाहते हैं।

(iii) Run पर क्लिक करें।

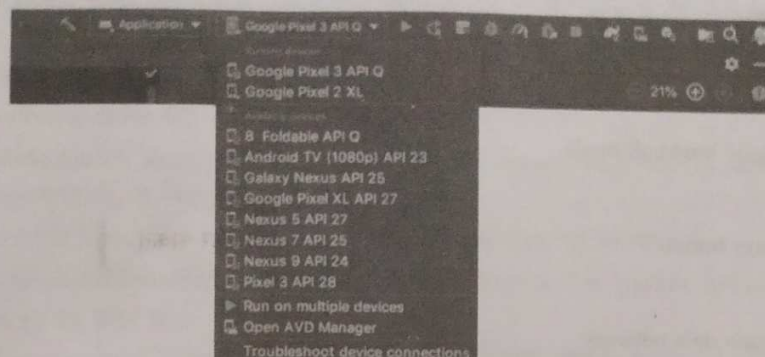
Launch the Emulator without first running an app

Emulator शुरू करने के लिए—

(i) AVD Manager खोलें।

(ii) AVD पर डबल-क्लिक करें, या Run पर क्लिक करें।

Android एम्प्यूलेटर लोड होता है।



चित्र 2.18

जबकि एम्यूलेटर चल रहा है, आप एन्ड्रॉइड स्टूडियो प्रोजेक्ट चला सकते हैं और एम्यूलेटर को target device के रूप में चुन सकते हैं। आप उन्हें install करने और run करने के लिए एम्यूलेटर पर एक या एक से अधिक APK भी खींच सकते हैं।

Run the Android Emulator directly in Android Studio

एन्ड्रॉइड स्टूडियो में एम्यूलेटर चलाने के लिए इन चरणों का पालन करें—

- File > Settings > Tools > Emulator (or Android studio > Preferences > Tools > Emulator on macOS) पर क्लिक करें, फिर टूल विंडो में लॉन्च का चयन करें और OK पर क्लिक करें।
- यदि एम्यूलेटर विंडो automatically दिखाई नहीं देती है, तो इसे View>Tool Windows > Emulator पर क्लिक करके खोलें।
- AVD Manager का उपयोग करके या अपने ऐप को चलाते समय इसे target करके अपने virtual डिवाइस को प्रारम्भ करें।

Emulator shortcuts/command

एम्यूलेटर के अन्य बहुत उपयोगी कमांड हैं वे नीचे सूचीबद्ध हैं—

Sr. No.	Command & Description
1.	Home Shifts to main screen
2.	F2 Toggles context sensitive menu
3.	F3 Bring out call log
4.	F4 End call

52 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

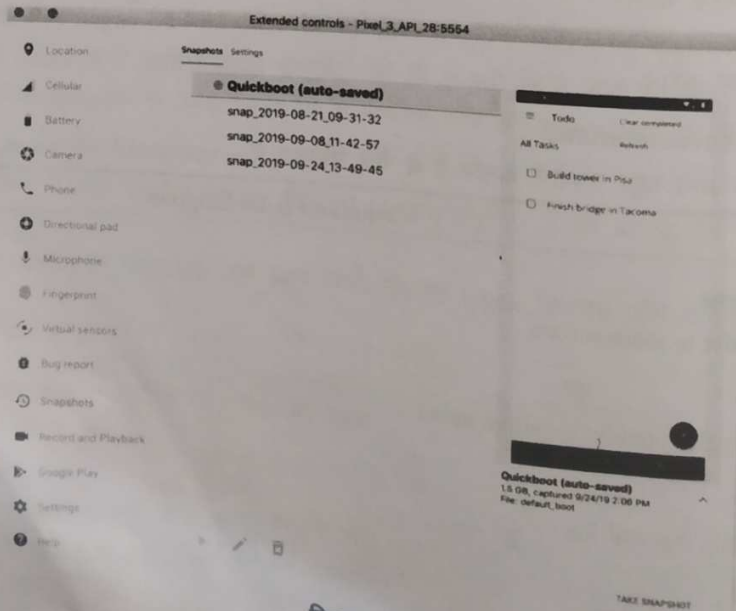
- | | |
|-----|------------------------------------|
| 5. | F5
Search |
| 6. | F6
Toggle trackball mode |
| 7. | F7
Power button |
| 8. | F8
Toggle data network |
| 9. | Ctrl+F5
Ring volume up |
| 10. | Ctrl+F6
Ring volume down |

Install and add files

Emulated डिवाइस पर APK फाइल इंस्टॉल करने के लिए emulator स्क्रीन पर APK फाइल Drag करें।

(i) **Snapshots**—एक snapshot एक AVD (Android वर्चुअल डिवाइस) की एक stored image है जो समय डिवाइस की सम्पूर्ण स्थिति को preserve करता है जो इसे save किया गया था।

snapshot को saving, loading करने और managing करने के लिए अधिकांश control snapshot Settings टैब में हैं।



चित्र 2.19

AVNISH MISHRA

(ii) **Save Quick Boot snapshots**—जब आप किसी AVD को बंद करते हैं, तो आप specify कर सकते हैं कि एम्यूलेटर बंद होने पर automatically स्नैपशॉट को save करता है या नहीं। एम्यूलेटर की Extended controls window खोलें। स्नैपशॉट की control की श्रेणी में, सेटिंग टैब पर नेविगेट करें। निम्न विकल्पों में से एक का चयन करने के लिए Quickboot drop-down मेनू में ऑटो-सेव करेंट स्टेट का उपयोग करें।

(iii) **Disable Quick Boot**—यदि आप Quick Boot को disable करना चाहते हैं तो आपका AVD हमेशा cold boot perform करता है, तो निम्न कार्य करें—

1. Tools>AVD Manager का चयन करें और इस AVD को Edit पर क्लिक करें।
2. Show Advanced Settings पर क्लिक करें और scroll down to Emulated Performance करें।
3. Cold boot का चयन करें।

(iv) **Screen recording**—आप एन्ड्रॉइड एम्यूलेटर से वीडियो और ऑडियो रिकॉर्ड कर सकते हैं और रिकॉर्डिंग को वेबएम या एनिमेटेड जीआईएफ फाइल में store कर सकते हैं। स्क्रीन रिकॉर्डिंग control Extended control विंडो के स्क्रीन रिकॉर्ड टैब में है।

(v) **Camera support**—एम्यूलेटर पहले के एन्ड्रॉइड वर्जन के लिए आपके वर्चुअल डिवाइस पर बेसिक कैमरा फंक्शनलिटी के इस्तेमाल का समर्थन करता है।

Android नवीनतम संस्करण निम्नलिखित अतिरिक्त Android एम्यूलेटर कैमरा क्षमताओं का समर्थन करता है—

- RAW capture
- YUV reprocessing
- Level 3 devices
- लॉजिकल कैमरा स्पोर्ट
- सेंसर manager से डेटा का उपयोग करके सेंसर orientation का simulate करना।
- handshake frequency को कम करके वीडियो stabilization लागू करना।
- YUV पाइपलाइन में आमतौर पर किए गए upscaling को हटाकर edge enhancement को लागू करना।
- Concurrent कैमरे

2.7.3. Emulator Settings and Extended Controls

डेटा भेजने, डिवाइस के गुण बदलने, ऐप्स को control करने और बहुत कुछ करने के लिए extended control का उपयोग करें। extended control विंडो खोलने के लिए, एम्यूलेटर पैनल में More एम्यूलेटर extended control आइकन पर क्लिक करें।

[1] Extended controls

- **Location** : एम्यूलेटर आपको “my location” जानकारी simulate करने देता है।
- **Displays** : एम्यूलेटर आपको अपने ऐप को multiple displays में deploy करने की अनुमति देता है।
- **Cellular** : एम्यूलेटर आपको विभिन्न नेटवर्क स्थितियों का simulate करने देता है।
- **Battery** : आप यह देखने के लिए किसी डिवाइस की बैटरी के गुणों का simulate कर सकते हैं कि आपका ऐप विभिन्न परिस्थितियों में कैसा प्रदर्शन करता है।
- **Phone** : एम्यूलेटर आपको आने वाले फोन कॉल और टेक्स्ट संदेशों को simulate करने देता है।
- **Many others**

[2] Settings—

(a) General

- I. Emulator window theme : लाइट या डार्क का चयन करें।
- II. Send keyboard shortcuts : आप वर्चुअल डिवाइस पर सभी कीबोर्ड इनपुट भेजने के लिए इस सेटिंग्स को बदल सकते हैं, जिसमें इनपुट भी शामिल है जो एम्यूलेटर में एक shortcut होगा।
- III. Screenshot save location : एम्यूलेटर स्क्रीन को स्क्रीनशॉट को सहेजने के लिए स्थान specify करने के लिए फोल्डर आइकन पर क्लिक करें।
- IV. Use detected ADB location : यदि आप एम्यूलेटर को एंड्रॉइड स्टूडियो के बाहर से चलाते हैं और चाहते हैं कि यह एक विशेष ADB executable योग्य का उपयोग करे, तो इस विकल्प को रद्द करें और ADB टूल्स स्थान specify करें।
- V. When to send crash reports : Always, Never, or Ask चयन करें।
- VI. Show window frame around device : डिवाइस स्क्रीन फाइलों के साथ एम्यूलेटर को आस पास के विंडो फ्रेम के बिना दिखाया गया है।

Proxy—डिफॉल्ट रूप से, एम्यूलेटर एंड्रॉइड स्टूडियो HTTP प्रॉक्सी सेटिंग्स का उपयोग करता है, लेकिन यह स्क्रीन आपको एम्यूलेटर के लिए HTTP प्रॉक्सी कॉन्फिगरेशन को मैनुअल रूप से परिभाषित करने की अनुमति देता है।

(b) Advanced—

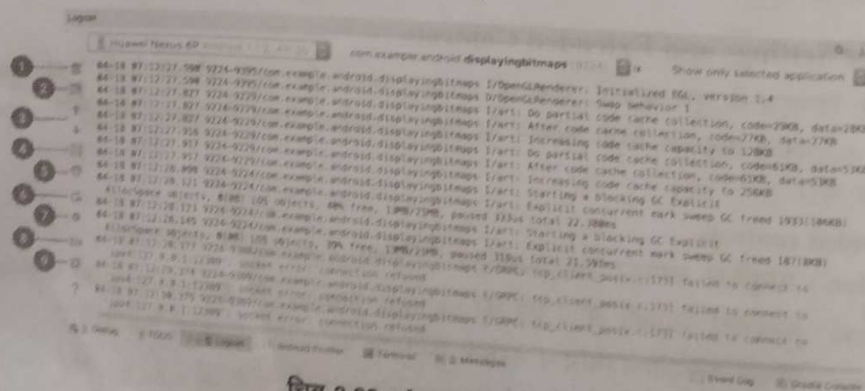
- I. OpenGL ES renderer : graphics acceleration प्रकार का चयन करें।
- II. OpenGL ES API level : एम्यूलेटर में उपयोग करने के लिए OpenGL ES के maximum संस्करण का चयन करें।

2.8. Logcat Usage

एंड्रॉइड स्टूडियो में Logcat window सिस्टम संदेशों को प्रदर्शित करती है।

I. View your app logs—एप्लिकेशन के लिए log messages प्रदर्शित करने के लिए—

- (i) किसी डिवाइस पर अपना ऐप बनाएँ और चलाएँ।
- (ii) **View > Tool Windows > Logcat** पर क्लिक करें।



चित्र 2.20 : Logcat usage

Logcat toolbar निम्नलिखित बटन प्रदान करता है—

I. Clear logcat



: visible log को clear करने के लिए क्लिक करें।

2. **Scroll to the end**  : लॉग के bottom पर जाने के लिए क्लिक करें और नवीनतम लॉग संदेश देखें।

3. **Up the stack trace**  **and Down the stack trace**  : लॉग में स्टैक के निशान को ऊपर-नीचे

नेविगेट करने के लिए क्लिक करें, इसके बाद के फाइलनामों का चयन करें (और संपादक में संगत लाइन संख्याओं को देखकर) जो printed exceptions में दिखाई देते हैं।

4. **Use soft wraps**  : line wrapping को enable करने और horizontal scrolling को रोकने के लिए क्लिक करें।

5. **Print**  : लॉगकैट संदेशों को print करने के लिए क्लिक करें।

6. **Restart**  : Click to clear the log and restart logcat. लॉग clear करने और logcat को पुनरारंभ के लिए क्लिक करें।

7. **Logcat header Configure Logcat Header** dialog खोलने के लिए क्लिक करें, जहाँ आप प्रत्येक लॉगकैट संदेश की उपस्थिति को customize कर सकते हैं।

8. **Screen capture**  : स्क्रीनशॉट कैप्चर करने के लिए क्लिक करें।

9. **Screen record**  : डिवाइस का वीडियो रिकॉर्ड करने के लिए क्लिक करें।

II. log messages—Log class आपको log messages बनाने की अनुमति देता है जो logcat में दिखाई देते हैं। निम्न लॉग method, उच्चतम से निम्नतम प्राथमिकता के क्रम में सूचीबद्ध—

- (i) Log.e(String, String) (error)
- (ii) Log.w(String, String) (warning)
- (iii) Log.i(String String) (information)
- (iv) Log.d(String String) (debug)
- (v) Log.v(String String) (verbose)

2.9. Introduction to DDMS

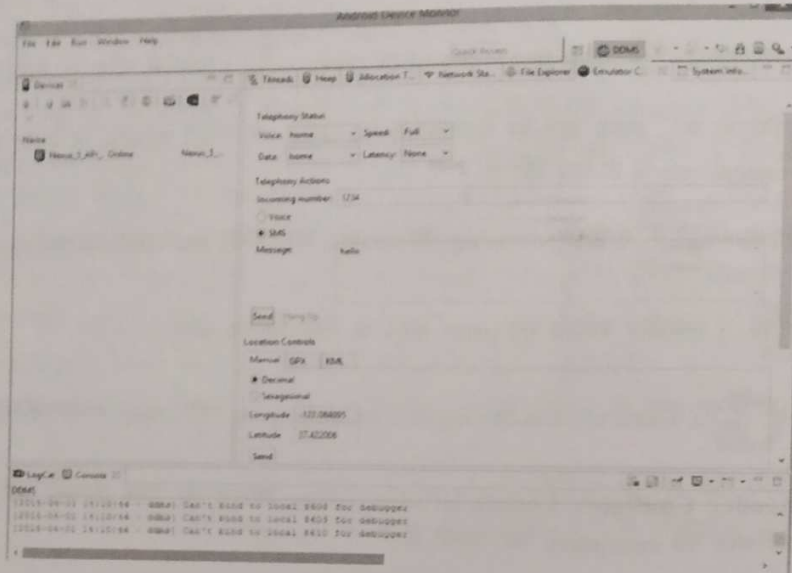
DDMS का मतलब Dalvik debug monitor server है, जो डिवाइस पर कई सेवाएँ प्रदान करता है। सेवा में message formation, call spoofing, capturing screenshot, exploring internal threads और फाइल सिस्टम e.t.c. शामिल हो सकते हैं।

(i) **Working**—Android स्टूडियो से **Tools>Android>Android device Monitor** पर क्लिक करें।

प्रत्येक एप्लिकेशन अपनी process में run करता है और प्रत्येक process वर्चुअल मशीन में run करती है। प्रत्येक virtual machine (VM) एक unique port को exposes करता है, जिससे debugger जुड़ सकता है।

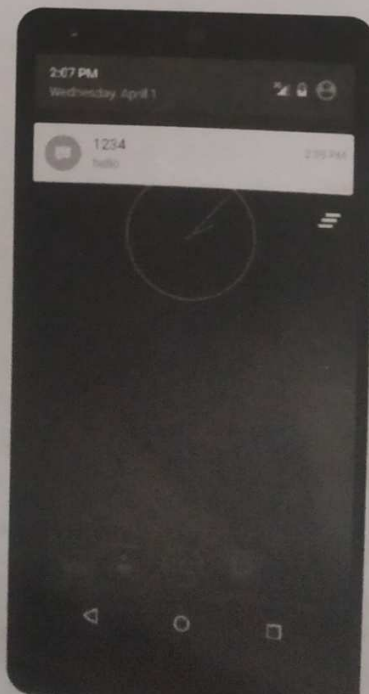
जब DDMS शुरू होता है, तो यह ADB से जुड़ता है। जब कोई डिवाइस कनेक्ट होता है, तो एक VM monitoring service ADB और DDMS के बीच बनाई जाती है, जो डिवाइस पर एक VM शुरू होने या समाप्त होने पर DDMS को सूचित करता है।

(ii) SMS करना—एमुलेटर से SMS बनाना। हमें telnet client और server को नीचे दिखाए गए अनुसार कॉल करना होगा।



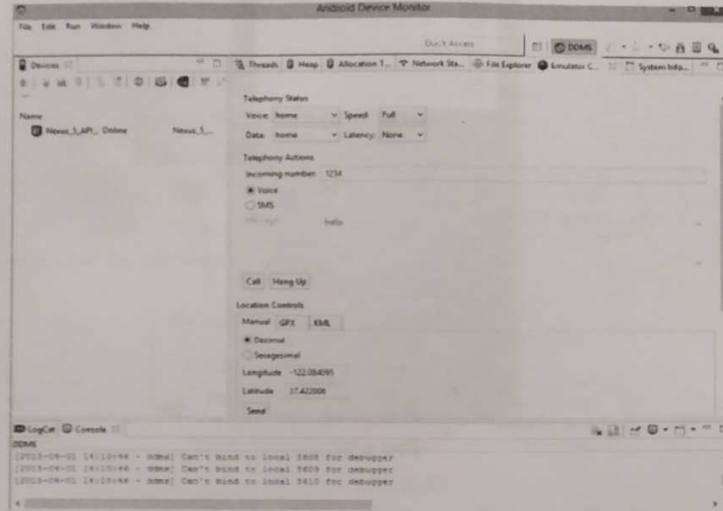
चित्र 2.21

अब Send बटन पर क्लिक करें, और आपको Emulator window में एक SMS notification दिखाई देगा। इसे नीचे दिखाया गया है—



और

(iii) Call करना—DDMS में, Emulator Control tab चुनें। Emulator Control tab में, आवाज पर क्लिक करें और फिर आने वाले नंबर टाइप करना शुरू करें। यह नीचे दी गई तस्वीर में दिखाया गया है—



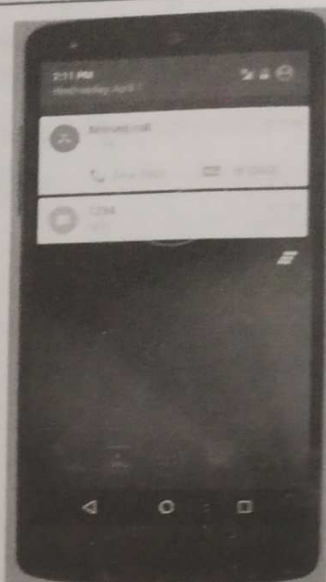
चित्र 2.23

अब अपने Emulator पर कॉल करने के लिए Call बटन पर क्लिक करें। इसे नीचे दिखाया गया है—



चित्र 2.24

अब कॉल को समाप्त करने के लिए Android studio window में hang-up पर क्लिक करें। फर्जी एसएमएस और कॉल को अधिसूचना विंडो से केवल माउस का उपयोग करके केंद्र में ले जाया जा सकता है। इसे नीचे दिखाया गया है—



चित्र 2.25

Platform tools—प्लेटफॉर्म टूल नवीनतम एन्ड्रॉइड प्लेटफॉर्म की सुविधाओं का समर्थन करने के लिए अनुकूलित हैं।

प्लेटफॉर्म टूल आमतौर पर हर बार जब आप एक नया एसडीके प्लेटफॉर्म स्थापित करते हैं तो अपडेट होते हैं। प्लेटफॉर्म टूल का प्रत्येक अपडेट पुराने प्लेटफॉर्म के साथ पिछड़ा हुआ है।

Platform के कुछ tools नीचे दिए गए हैं—

- (i) Android Debug Bridge (ADB)
- (ii) Android Interface Definition Language (AIDL)
- (iii) aapt, dexdump and dex e.t.c.

2.10. Creating Your First Project

Section 2.6 में प्रोजेक्ट बनाने के step बताये गए हैं, इसलिए Section 2.6 refer करें।

2.11. The Manifest file

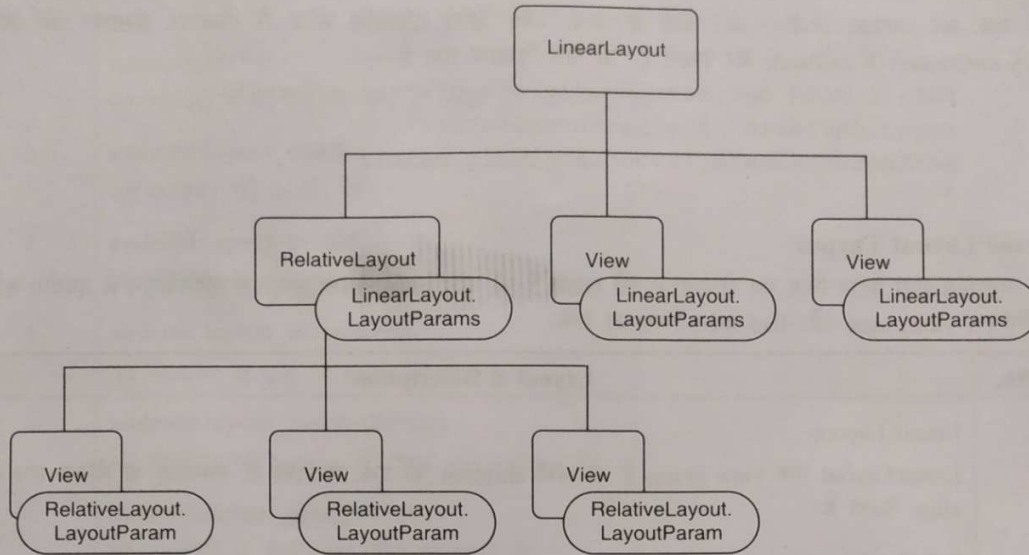
manifest file : refer Section 2.2

2.12. Layout Resource

एक लेआउट resource एक Activity में UI के लिए architecture या UI के एक component को परिभाषित करता है।

File location : res/layout/filename.xml.

फाइलनाम का उपयोग resource ID के रूप में किया जाएगा। उपयोगकर्ता इंटरफेस के लिए बुनियादी ब्लॉक एक View ऑब्जेक्ट है जो View क्लास से बनाया गया है और स्क्रीन पर एक आयताकार क्षेत्र पर कब्जा करता है और ड्राइंग और इवेंट हैंडलिंग के लिए जिम्मेदार है। View, विजेट्स के लिए बेस क्लास है, जिसका उपयोग इंटरैक्टिव यूआई घटकों जैसे बटन, टेक्स्ट फील्ड आदि बनाने के लिए किया जाता है।



चित्र 2.26

तीसरे स्तर पर हमारे पास अलग-अलग लेआउट हैं जो ViewGroup class के subclass हैं और एक विशिष्ट लेआउट एक एन्ड्रॉइड उपयोगकर्ता इंटरफेस के लिए visual structure को परिभाषित करता है और View/ViewGroup objects का उपयोग करके रन टाइम पर बनाया जा सकता है या आप अपने लेआउट को XML फाइल के माध्यम से घोषित कर सकते हैं main_layout.xml जो आपके प्रोजेक्ट के रिस/लेआउट फोल्डर में स्थित है।

एक लेआउट में किसी भी प्रकार के विजेट्स जैसे बटन, लेबल, टेक्स्ट बाक्स इत्यादि हो सकते हैं। निम्नलिखित LinearLayout XML फाइल का उदाहरण है—

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res
/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is a TextView"/>
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="This is a Button"/>
    <!-- More GUI components go here -->
</LinearLayout>
  
```

60 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

एक बार आपका लेआउट बन जाने के बाद, आप अपने गतिविधि कोड से लेआउट संसाधन को अपने `Activity.onCreate()` में callback कर सकते हैं, जो नीचे दिखाया गया है—

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}
```

Android Layout Types

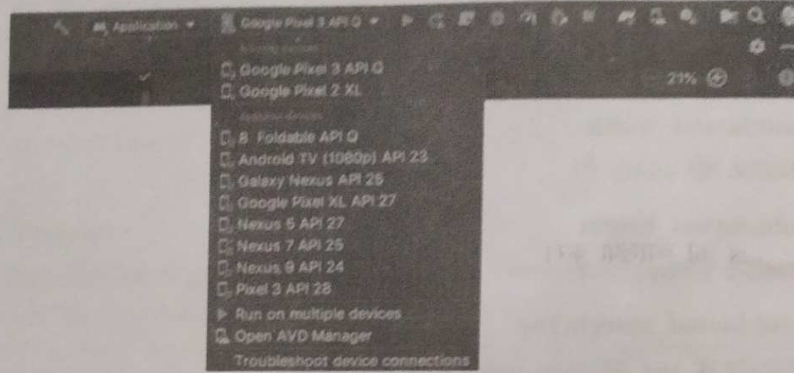
एन्ड्रॉइड द्वारा प्रदान किए गए लेआउट्स की संख्या है जो आप लगभग सभी एन्ड्रॉइड एप्लिकेशन में उपयोग करें ताकि विभिन्न view, look और feel प्रदान किए जा सकें।

Sr.No.	Layout & Description
1.	Linear Layout LinearLayout एक view group है जो सभी children को एक ही दिशा में, लम्बवत् या क्षैतिज रूप से align करता है।
2.	Relative Layout Relative Layout एक view group है जो relative स्थितियों में child views को प्रदर्शित करता है।
3.	Table Layout Table Layout एक view है जो group rows और column में देखता है।
4.	Absolute Layout AbsoluteLayout अपने children के exact location को specify करने में enable बनाती है।
5.	Frame Layout FrameLayout स्क्रीन पर एक placeholder है जिसे आप एकल view प्रदर्शित करने के लिए उपयोग कर सकते हैं।
6.	List View ListView एक view group है जो स्कॉल करने योग्य items की सूची प्रदर्शित करता है।
7.	Grid View GridView एक ViewGroup है जो two-dimensional, स्कॉल करने योग्य ग्रिड में आइटम प्रदर्शित करता है।

Layout Attributes

प्रत्येक लेआउट में attributes का एक सेट होता है जो उस लेआउट के visual properties को परिभाषित करता है। सभी लेआउट के बीच कुछ common attributes हैं और उनकी अन्य attributes हैं जो उस लेआउट के लिए specific हैं। निम्नलिखित common attributes हैं और सभी लेआउट पर लागू की जाएंगी—

Sr.No.	Attribute & Description
1.	android:id यह वह ID है जो विशिष्ट रूप से view को पहचानती है।
2.	android:layout_width यह लेआउट की width है।
3.	android:layout_height यह लेआउट की height है।
4.	android:layout_marginTop यह लेआउट के top पर extra space है।
5.	android:layout_marginBottom यह लेआउट की bottom की तरफ extra space है।
6.	android:layout_marginLeft यह लेआउट के बाईं ओर अतिरिक्त स्थान है।
7.	android:layout_marginRight यह लेआउट के दाईं ओर अतिरिक्त स्थान है।
8.	android:layout_gravity यह specifies करता है कि child Views कैसे positioned किए गए हैं।
9.	android:layout_weight यह specifies करता है कि लेआउट में कितना अतिरिक्त स्थान View को आवंटित किया जाना चाहिए।
10.	android:layout_x यह लेआउट के x-coordinate को specify करता है।
11.	android:layout_y यह लेआउट के y-coordinate को specify करता है।
12.	android:layout_width यह लेआउट की width है।
13.	android:paddingLeft यह लेआउट के लिए भरा हुआ left padding है।
14.	android:paddingRight यह लेआउट के लिए भरा हुआ right padding है।
15.	android:paddingTop यह लेआउट के लिए भरा top पैडिंग है।
16.	android:paddingBottom यह लेआउट के लिए भरा bottom पैडिंग है।



चित्र 2.27

2.13. Running Your App on Emulator

Refer section 2.7

2.14. Debugging the Android App

एंड्रॉइड स्टूडियो एक डिबगर प्रदान करता है जो आपको निम्नलिखित कार्य करने की अनुमति देता है—

- (i) अपने एप्लिकेशन को डिबग करने के लिए एक उपकरण चुनें।
- (ii) अपने Java, Kotlin और C/C++ कोड में ब्रेकपॉइंट सेट करें।
- (iii) variables की जाँच करें रनटाइम पर expressions को evaluate करें।

2.14.1. Start debugging

आप निम्नानुसार डिबगिंग शुरू कर सकते हैं—

1. ऐप कोड में कुछ breakpoints सेट करें।
2. toolbar में, target डिवाइस ड्रॉप-डाउन मेनू से अपने ऐप को डिबग करने के लिए एक device का चयन करें।
3. Debug  पर क्लिक करें।

2.14.2. Attach the debugger to a running app

यदि आपका ऐप पहले से ही आपके डिवाइस पर चल रहा है, तो आप अपने ऐप को इस प्रकार शुरू किए बिना डिबगिंग शुरू कर सकते हैं—

- (i) Android प्रक्रिया में Attach पर क्लिक कर debugger को अटैच करें।
- (ii) Process dialog में, उस process का चयन करें जिसे आप डिबगर को संलग्न करना चाहते हैं।
- (iii) OK पर क्लिक करें।

प्रश्नावली

1. Android environment को setup करने के लिए अलग-अलग स्टेप का उपयोग किया जाता है। कैसे?
2. Android मैनिफेस्ट फाइल क्या है?
3. Android Gradle क्या है?
4. Uses-permission को समझाइए।
5. Uses-sdk, Resources की व्याख्या करें।
6. R.Java file क्या है?
7. Android में प्रयुक्त होने वाले Layouts को समझाइए।
8. Emulator क्या होता है?
9. Logcat usage फाइल का उपयोग बताइए।
10. DDMS क्या है?
11. Hello World App का प्रोग्राम बनाए।





एन्ड्रॉइड फंडामेंटस और यूजर इंटरफेस डिजाइन (Android Fundamentals and User Interface Design)

Activities and activity lifecycle, Permission system

Basic UI Components : Text View, Button, Radio button, Edit text, Image view for image, Check box, Progress bar, Event handling in Android

Layouts : Liner Layout, Relative layout, Frame layout, Coordinate layout, [dip, dip, sip, sp] versus px

Intents : Intents introduction and importance, Types of intents (Explicit intents, implicit intents)

इस अध्याय में, आप basic एन्ड्रॉइड प्रोग्रामिंग concept को सीखते हैं और विभिन्न ऐप्स का निर्माण करते हैं, हैलो वर्ल्ड के साथ शुरू करते हैं और उन ऐप्स पर अपना काम करते हैं जो schedule jobs, update settings और आर्किटेक्चर कंपोनेंट्स का उपयोग करते हैं।

3.1. Activities and Activity Lifecycle

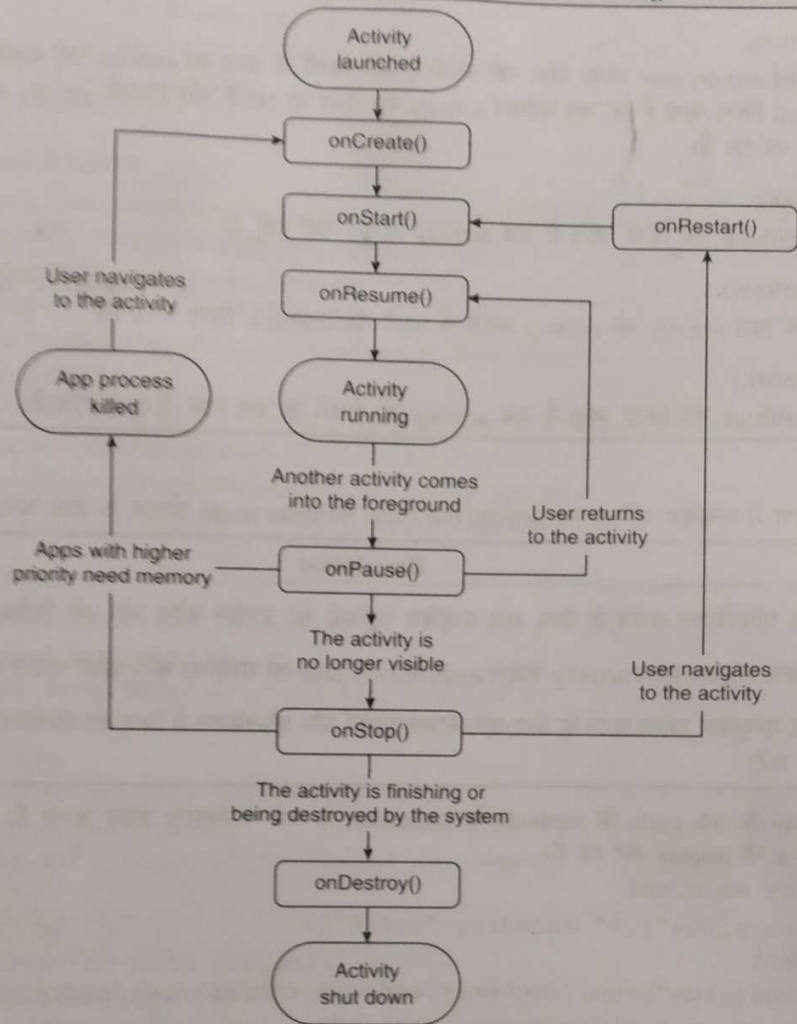
यह केवल एक class है जो एकल स्क्रीन का represents करता है। वे UI को निर्देशित करते हैं और स्मार्ट फोन स्क्रीन पर user की interaction को संभालते हैं। यह AWT में एक फ्रेम की तरह है। एक activity को activity class के subclass के रूप में implemented किया जाता है।

```
public class MainActivity extends Activity
{
    .....
}
```

Activity life cycle

यदि आपने C, C++ या Java प्रोग्रामिंग भाषा के साथ काम किया है, तो आपने देखा होगा कि आपका प्रोग्राम main () फंक्शन से शुरू होता है। इसी तरह से, Android system में onCreate () callback method पर कॉल के साथ शुरू होने वाली activity से अपना program शुरू करते हैं।

Android सिस्टम onCreate () callback method पर call के साथ शुरू होने वाली activity से अपना program शुरू करता है। कॉलबैक method का एक sequence है जो एक activity शुरू करता है। callback method का एक sequence होता है जो एक activity को फाड़ (tear down) देता है जैसा कि नीचे दी गई activity life cycle diagram में दिखाया गया है।



चित्र 3.1

Activity class निम्नलिखित call backs method को परिभाषित करता है—

S. No.	Callback & Description
1.	onCreate() यह पहला callback है और जब activity पहली बार बनाई जाती है तो उसे कॉल किया जाता है।
2.	onStart() user को activity दिखाई देने पर इस callback को call किया जाता है।
3.	onResume() यह तब call किया जाता है जब user एप्लिकेशन के साथ interact करना शुरू करता है।

डवलपमेंट ऑफ एन्ड्रॉइड एप्लिकेशन

4.	onPause() रुकी हुई activity user इनपुट प्राप्त नहीं करती है और किसी भी कोड को executes नहीं करती है और इसको तब call किया जाता है कि जब वर्तमान activity को रोका जा रहा है और पिछली activity को फिर से शुरू किया जा रहा है।
5.	onStop() यह callback तब किया जाता है जब activity दिखाई नहीं देती है।
6.	onDestroy() सिस्टम द्वारा activity को destroy करने से पहले यह callback किया जाता है।
7.	onRestart() यह callback तब किया जाता है जब activity को रोकने के बाद फिर से शुरू होता है।

उदाहरण

इस उदाहरण में एन्ड्रॉइड एप्लिकेशन activity life cycle को hello world प्रोग्राम के द्वारा समझाया गया है।

Step	Description
1.	एन्ड्रॉइड एप्लिकेशन बनाने के लिए आप एन्ड्रॉइड स्टूडियो का उपयोग करेंगे और इसे हैलोवर्ल्ड नाम दें।
2.	नीचे बताए अनुसार मुख्य activity फाइल main activity.java को संशोधित करें। बाकी फाइलें अपरिवर्तित रखें।
3.	एन्ड्रॉइड एम्यूलटर लॉन्च करने के लिए एप्लिकेशन चलाएँ और एप्लिकेशन में किए गए परिवर्तनों के परिणाम को verify करें।

यह activity के life cycle के methods के invocation के बारे में वितरण प्रदान करता है। इस उदाहरण में, हम logcat content भी display कर रहे हैं।

```
activity main.xml
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.myapplication">
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    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>
</application>
</manifest>
```



```

        </intent-filter>
    </activity>
</application>
</manifest>

```

File: MainActivity.java

```

package com.example.helloworld.activitylifecycle;
import android.app.Activity;
import android.os.Bundle;
import android.util.Log;
public class MainActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Log.d("lifecycle", "onCreate invoked");
    }
    @Override
    protected void onStart() {
        super.onStart();
        Log.d("lifecycle", "onStart invoked");
    }
    @Override
    protected void onResume() {
        super.onResume();
        Log.d("lifecycle", "onResume invoked");
    }
    @Override
    protected void onPause() {
        super.onPause();
        Log.d("lifecycle", "onPause invoked");
    }
    @Override
    protected void onStop() {
        super.onStop();
        Log.d("lifecycle", "onStop invoked");
    }
    @Override
    protected void onRestart() {
        super.onRestart();
        Log.d("lifecycle", "onRestart invoked");
    }
    @Override
    protected void onDestroy() {

```

```
super.onDestroy();
Log.d("lifecycle", "onDestroy invoked");
}
```

Android स्टूडियो में LogCat विंडो में लॉग संदेश

अब logcat पर देखें—onCreate, onStart और onResume method लागू हैं।

```
02-15 11:21:06.567 4590-4590/
com.example.helloworld.activitylifecycle F/Android :: The
onCreate() event
```

```
02-15 11:21:06.589 4590-4590/
com.example.helloworld.activitylifecycle F/Android :: The
onStart() event
```

```
02-15 11:21:06.691 4590-4590/
com.example.helloworld.activitylifecycle F/Android :: The
onResume() event
```

अब HOME Button पर क्लिक करें। आप देखेंगे कि onPause method लागू है।

```
02-15 11:22:12.567 4590-4590/
com.example.helloworld.activitylifecycle F/Android :: The
onPause() event
```

थोड़ी देर के बाद, आप देखेंगे कि onStop method लागू है।

```
02-15 11:23:20.690 4590-4590/
com.example.helloworld.activitylifecycle F/Android :: The
onStop() event
```

3.2. Permissions on Android

Permission शब्द अधिकारिक अर्थ है किसी व्यक्ति को किसी special कार्य को करने की permission देना यह किसी भी प्रकार की कार्रवाई करने के लिए दी गई consent या authorization है।

किसी भी कम्प्यूटर की फाइल प्रणाली को सीधे एक्सेस करना एक महत्वपूर्ण security risk हो सकता है, इसलिए यह सुनिश्चित करने के लिए पर्याप्त सुरक्षा है कि कोई भी malicious app यूजर के डेटा पर चले इसके लिए परमिशन प्रोवाइड की गयी है। जिसका use करके यूजर किसी भी app को रोक सकता है।

App permission access की सुरक्षा करके यूजर की गोपनीयता का समर्थन करने में मदद करती हैं—

(i) Restricted data जैसे कि system state और a user's contact information.

(ii) Restricted actions जैसे कि connecting to a paired device और recording audio.

3.2.1. Importance of setting up permission in android

यदि कोड किसी के लिए और प्रतिबंधों के बिना accessible है, तो एक scenario हो सकता है जहाँ कोड ने छेड़छाड़ की जाती है, जिसके परिणामस्वरूप defect leakage होता है। अनुमतियाँ सेट होने के बाद, कोड केवल authorized users के लिए उपलब्ध हो जाता है।

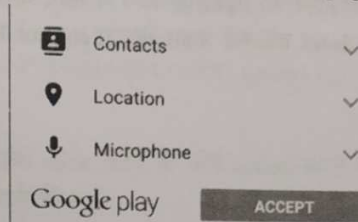
3.2.2. Permissions के प्रकार

एन्ड्रॉइड विभिन्न प्रकारों में permission को वर्गीकृत करता है, जिसमें

1. Install-time permissions
2. Runtime permissions
3. Special permissions

प्रत्येक permission का प्रकार restricted डेटा के दायरे को और restricted action को इंगित करता है जिसे आपका App एक्सेस और perform कर सकता है, जब सिस्टम आपके ऐप को permission देता है।

1. Install-time permissions—इंस्टॉल-टाइम permission आपके ऐप को restricted डेटा तक सीमित पहुँच देती है और वे आपके ऐप को restricted कार्यों को करने की permission देते हैं जो सिस्टम या अन्य ऐप्स को न्यूनतम प्रभावित करते हैं। जब आप अपने ऐप में इंस्टॉल-टाइम permission declare करते हैं, तो यूजर द्वारा आपके ऐप को इंस्टॉल करने पर सिस्टम automatically आपके ऐप को permission देता है। एक app store यूजर को एक इंस्टॉल-टाइम permission notice प्रस्तुत करता है जब वे ऐप के details page को देखते हैं।



चित्र 3.2

एन्ड्रॉइड install-time permissions में

(i) Normal permissions

(ii) Signature permissions

सहित कई प्रकार के permission शामिल हैं।

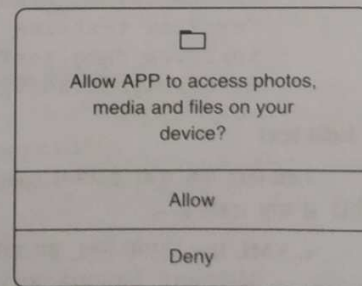
(i) Normal permissions—ये permission उन डेटा और कार्यों तक पहुँच की permission देती हैं जो आपके ऐप के सैंडबॉक्स से beyond हैं। हालांकि data और actions यूजर की privacy और अन्य एप्लिकेशन के संचालन के लिए बहुत कम risk रखती हैं।

(ii) Signature permissions—यदि ऐप एक हस्ताक्षर की permission की घोषणा करता है जिसे किसी अन्य ऐप ने परिभाषित किया है और यदि दो ऐप एक ही प्रमाण पत्र द्वारा हस्ताक्षरित हैं, तो सिस्टम पहले ऐप को इंस्टॉल समय पर permission देता है। अन्यथा, उस पहले ऐप को permission नहीं दी जा सकती।

2. Runtime permissions—Runtime permission, जिन्हें dangerous permission भी कहा जाता है, ऐप को restricted डेटा तक अतिरिक्त पहुँच प्रदान करती हैं, और वे आपके ऐप को restricted कार्यों को करने की permission देते हैं जो सिस्टम और अन्य एप्लिकेशन को अधिक प्रभावित करते हैं। इसलिए, आपको restricted डेटा तक पहुँचने या restricted कार्यों को करने से पहले अपने ऐप में Runtime permission का अनुरोध करना होगा। जब आपका ऐप runtime permission का अनुरोध करता है, तो सिस्टम runtime permission prompt प्रस्तुत करता है, जैसा कि चित्र में दिखाया गया है।

कई runtime permission private user डेटा तक पहुँचती हैं, एक special प्रकार का restricted डेटा जिसमें संभावित sensitive information शामिल होती है। Private user डेटा के उदाहरणों में location और contact information शामिल हैं।

3. Special permissions—Special permission special एप्लिकेशन संचालन के अनुरूप हैं। केवल प्लेटफॉर्म और OEM special permission को



चित्र 3.3

परिभाषित कर सकते हैं। इसके अतिरिक्त, प्लेटफॉर्म और OEM आमतौर पर special permission को परिभाषित करते हैं, जब वे special रूप से शक्तिशाली कार्यों तक पहुँच की रक्षा करना चाहते हैं।

सिस्टम सेटिंग्स में special app access पेज में user-toggleable operations का एक सेट होता है। इनमें से कई ऑपरेशनों को special permission के रूप में लागू किया गया है।

3.3. Basic UI Components

एन्ड्रॉइड UI कंट्रोल एन्ड्रॉइड के वे component हैं जो UI को अधिक इंटरैक्टिव तरीके से डिजाइन करने के लिए उपयोग किए जाते हैं। यह हमें एक ऐसे एप्लिकेशन को विकसित करने में मदद करता है जो view components के साथ user interaction को बेहतर बनाता है। Android हमें कई प्रकार के UI control प्रदान करता है जैसे कि

1. Text view
2. Edit text
3. Button
4. Radio button
5. Image view for image
6. Check bar
7. Progress bar

Text view

TextView एक UI कंपोनेंट है जो टेक्स्ट को यूजर को अपनी डिस्प्ले स्क्रीन पर दिखाता है। हम इसे दो तरीकों से बना सकते हैं—

I. XML file—इसके लिए, हम इसे layout टैग में निम्नानुसार declare करते हैं—

```
<LinearLayout xmlns:android =
    "http://schemas.android.com/apk/res/android"
    <TextView
        //attributes to describe it
    />
</LinearLayout>
```

II. Activity file—इसमें, हम setText() method का उपयोग करते हुए इसे declare करते हैं।

```
setContentview(R.layout.activity_main);
LinearLayout linearlayout_name =
    (LinearLayout)findViewById(R.id.LinearLayout);
TextView textview_name = new TextView(this);
textview_name.setText("Hello I am Text View");
linearlayout.addView(textview);
```

Edit text

Edit text एक यूजर इंटरफेस control है जो यूजर को कुछ text enter करने की अनुमति देता है। हम इसे दो तरीकों से बना सकते हैं—

I. XML file—इसके लिए, हम इसे layout टैग में निम्नानुसार declare करते हैं—

```
<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android">
```



```

<EditText
//attributes
/>
</LinearLayout>

```

II. Activity file—इसमें, हम `getText()` method का उपयोग करते हुए इसे declare करते हैं—

```

setContentView(R.layout.activity_main);
LinearLayout linearlayout_name = (LinearLayout) findViewById
(R.id.LinearLayout) ;
EditText edittext_name = new EditText(this);
edittext_name.setHint("Hello I am EditText");
linearlayout.addView(edittext_name);

```

3. Button

यह एक UI है जो यूजर के द्वारा click करते ही कोई action करने के लिए उपयोग किया जाता है। हम इसे दो तरीकों से बना सकते हैं—

I. XML file—इसके लिए, हम इसे layout टैग में निम्नानुसार declare करते हैं—

```

<Linear Layout xmlns:android=
"http://schemas.android.com/apk/res/android">
<Button
//attributes
/>
</LinearLayout>

```

II. Activity file—Activity में, हम इसे नीचे की तरह प्रोग्रामेटिक रूप से declare करते हैं—

```

setContentView(R.layout.activity_main);
LinearLayout linearlayout_name =
(LinearLayout) findViewById(R.id.LinearLayout);
Button btn_name = new Button(this);
btn_name.setText("Hello I am Button");
linearlayout.addView(btn_name);

```

4. Radio Button

एन्ड्रॉइड में रेडियो बटन वह है जिसमें केवल दो possible states हैं, जो या तो check या uncheck किए किये जाते हैं। प्रारंभ में, यह unchecked state में हैं, एक बार checked के बाद यह unchecked नहीं हो सकता है। हम इसे दो तरीकों से बना सकते हैं—

I. XML file—इसके लिए, हम इसे layout टैग में निम्नानुसार declare करते हैं—

```

<Linear Layout xmlns:android=
"http://schemas.android.com/apk/res/android">
<RadioButton
android:text="Radio button"
android:checked="true"
/>
</LinearLayout>

```

II. Activity file—Activity में, हम इसे नीचे की तरह प्रोग्रामेटिक रूप से declare करते हैं—

```

setContentView(R.layout.activity_main);
LinearLayout linearlayout_name =
    (LinearLayout) findViewById(R.id.LinearLayout);
RadioButton btn_name = new RadioButton(this);
btn_name.setText("Hello");
btn_name.setChecked(true);
linearLayout.addView(btn_name);

```

5. Image Button

यह एक बटन के समान है, लेकिन यह एक कार्य करने के लिए बटन पर एक image प्रदर्शित करता है। इसमें हमें image का source देने की आवश्यकता है ताकि सिस्टम इसे लोड कर सके।

हम इसे दो तरीकों से बना सकते हैं—

I. XML file—इसके लिए, हम इसे layout टैग में निम्नानुसार declare करते हैं—

```

<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android">
    <ImageButton
        //other attributes ....
        android:src="@drawable/add_icon"/>
</LinearLayout>

```

II. Activity file—Activity में, हम इसे नीचे की तरह प्रोग्रामेटिक रूप से declare करते हैं—

```

setContentView(R.layout.activity_main);
LinearLayout linearlayout_name =
    (LinearLayout) findViewById(R.id.LinearLayout);
ImageButton btn_name = new Button(this);
btn_name.setImageResource(R.drawable.add_icon);
linearLayout.addView(btn_name);

```

6. CheckBox

एक check box UI control है जिसमें two states हैं जो या तो check या uncheck किए किये जाते हैं। अगर हमारे पास check box का एक group है, तो हम जितने चाहें उतने चुन सकते हैं।

हम इसे दो तरीकों से बना सकते हैं—

I. XML file —इसके लिए, हम इसे layout टैग में निम्नानुसार declare करते हैं—

```

<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android">
    <CheckBox
        android:checked="true"
        android:text="CheckBox"
        // other attributes
    />
</LinearLayout>

```

II. Activity file—Activity में, हम इसे नीचे की तरह प्रोग्रामेटिक रूप से declare करते हैं—

```

setContentView(R.layout.activity_main);
LinearLayout linearlayout_name =
    (LinearLayout) findViewById(R.id.LinearLayout);
CheckBox cb_name = new CheckBox(this);

```

```
cb_name.setText("Mayank");
cb_name.setChecked(true);
layout.addView(cb_name);
```

6. Progress bar

एन्ड्रॉइड में, हमारे पास एक progress bar होती है जो some action की progress को दिखाती है। एक progress bar दो मोड में हो सकती है—

(i) **Determinate mode**—इसमें progress को पूर्ण कार्रवाई के प्रतिशत के साथ दिखाया जाता है। साथ ही, लिया जाने वाला समय पहले से निर्धारित होता है।

(ii) **Indeterminate mode**—इसमें कार्य कब पूरा होगा, इसका कोई idea नहीं है, इसलिए, यह लगातार कार्य करता है।

हम इसे दो तरीकों से बना सकते हैं—

I. **XML file**—इसके लिए, हम इसे layout टैग में निम्नानुसार declare करते हैं—

```
<Linear Layout
    xmlns:android="http://schemas.android.com/apk/res/android">
    <ProgressBar
        // attributes, here we define the speed, layout, id, etc.
    />
</LinearLayout>
```

II. **Activity file**—Activity में, हम इसे नीचे की तरह प्रोग्रामेटिक रूप से declare करते हैं—

```
setContentView(R.layout.activity_main);
pgsBar = (ProgressBar) findViewById(R.id.pBar);
txtView = (TextView) findViewById(R.id.tView);
Button btn = (Button) findViewById(R.id.btnShow);
```

3.4. Event Handling in Android

Event handling इवेंट और हैंडलिंग के साथ आता है। यहाँ event एक action के अलावा और कुछ नहीं है। यहाँ handling का मतलब है manage करना। सम्पूर्ण Android event हैंडलिंग में हम सभी कार्य करते हैं। जो क्रियाएँ उपयोगकर्ता के इंटरैक्शन से संबंधित हैं। ये Event इंटरैक्टिव घटकों का उपयोग करके उपयोगकर्ता की बातचीत के बारे में डेटा एकत्र करते हैं। उदाहरण के लिए, जब आप कुछ बटन पर क्लिक करते हैं, तो onClick () का उपयोग किया जाता है। Android में सभी event फर्स्ट इन फर्स्ट आउट (FIFO) शेड्यूलिंग का उपयोग करके एक कतार में store किए जाते हैं। Android event प्रबंधन में संबंधित तीन अवधारणाएँ निम्नलिखित हैं—

1. Event listeners
2. Event handlers
3. Event listener registration

1. **Event listeners**—Event listener android के view class में एक इंटरफेस है। यह एक single कॉलबैक method है। जब user's action के द्वारा listener के साथ registration किया गया view activate हो जाता है, तो यह method call किये जाते हैं।

74 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लिकेशन

निम्नलिखित कुछ महत्वपूर्ण event listener हैं—

Event Listeners	Events Listener Description
onClick()	यह event तब होती है जब उपयोगकर्ता टच स्क्रीन पर एक आइटम पर क्लिक करता है।
onLongClick()	यह event तब होती है जब उपयोगकर्ता 1 सकेण्ड से अधिक समय तक किसी आइटम या स्क्रीन पर क्लिक करता है।
onFocusChange()	यह event तब होती है जब कोई उपयोगकर्ता किसी आइटम से दूर जाता है जो फोकस पर था।
onKey()	यह event तब होती है जब कोई उपयोगकर्ता किसी वस्तु पर focus करता है और क्लिक करता है।
onTouch()	यह event तब होती है जब कोई उपयोगकर्ता gestures या साधारण touch या टैप से किसी particular range को touch करता है।
onCreateContextMenu()	यह event तब होती है जब एक context menu बनाया जाता है।
onMenuItemClick()	यह तब होता है जब उपयोगकर्ता मेनू से किसी आइटम पर क्लिक करता है या उसका चयन करता है।

2. Event handlers—Event handlers वास्तविक तरीके हैं जिनके द्वारा एक्शन परफॉर्म होती है। एक event होने के बाद और event listener registration हैं, Event listener event handles को call करते हैं। ये कुछ कॉलबैक विधियों को परिभाषित करने के लिए उपयोगी हैं।

निम्नलिखित कुछ महत्वपूर्ण इवेंट handles हैं—

Event handler	Event Handler Description
onKeyUp()	जब कोई नई key event होती है तो सिस्टम इस method को लागू करता है।
onKeyDown()	जब key down event होती है तो सिस्टम इस method को लागू करता है।
onTouchEvent()	जब some touch event होती है तो सिस्टम इस method को लागू करता है।
onFocusChange()	सिस्टम इस method को तब लागू करता है जब कोई वस्तु फोकस में आ जाती है या फोकस खो देती है।

3. Event listener registration—Event listener registration वह प्रक्रिया है जिसमें इवेंट listeners इवेंट handler के साथ registered होते हैं। यह महत्वपूर्ण है क्योंकि handle केवल एक listener event को fire करेगा।

Android event listener registration निम्नलिखित तीन तरीकों से किया जा सकता है—

1. event listeners को registration करने का पहला तरीका सीधे activity_main.xml में उनका उल्लेख करना है।
2. हम listener इंटरफेस को लागू करने वाले activity class का उपयोग करके event listeners को भी register कर सकते हैं।
3. Anonymous class का उपयोग करके।

Touch mode—जबकि एक उपयोगकर्ता डिवाइस के साथ इंटरैक्ट करता है तब वे स्क्रीन को रोल करते हैं। इसलिए उन items पर focus देना आवश्यक है जो actionable हैं। touch functionality वाले उपकरणों में और

उपयोगकर्ता इसके साथ interact करना शुरू कर देता है तब यह महत्वपूर्ण नहीं है कि touchable items पर focus दें। Interaction का यह तरीका "टच मोड" होता है। एक बार जब कोई उपयोगकर्ता स्क्रीन को छूता है, तो डिवाइस टच मोड में प्रवेश करती है और फिर एक्शनेबल आइकन और बटन जैसे फोकस पर रखे जाते हैं।

Handling focus—फ्रेमवर्क यूजर इनपुट के अनुसार फोकस मूवमेंट को हैंडल करता है। फोकस को हैंडल करने में उपयोगकर्ता इनपुट movement के अनुसार फोकस को जोड़ना और हटाना शामिल है। व्यूज की मदद से फोकस में बदलाव होता है। आप यह देख सकते हैं कि view, focusable mode मोड में है या isFocusable() का उपयोग नहीं कर रहा है और हम setFocusable() का उपयोग करके फोकस मोड भी बदल सकते हैं।

Implementing android event handling

Step 1 : नया प्रोजेक्ट खोलें और सभी required details भरें।

Step 2 : ऐसा करने के बाद अपना प्रोजेक्ट खोलें और layout और element की आवश्यक ID सेट करें ताकि उन्हें आसानी से कॉल कर सकें।

Activity_main.xml फाइल खोलें। इस फाइल में निम्न कोड लिखा जाना है—

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/
res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginLeft="100dp"
        android:fontFamily="@font/arya_bold"
        android:text="Mayank"
        android:textColor="@color/colorPrimaryDark"
        android:textSize="50dp" />
    <Button
        android:id="@+id/btnClick"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="150dp"
        android:layout_marginTop="100dp"
        android:fontFamily="@font/arya_bold"
        android:text="Click Event" />
    <TextView
        android:id="@+id/txtResult"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#009999"
        android:layout_marginLeft="150dp"
        android:textSize="20dp"
        android:textStyle="bold"
        android:layout_marginTop="12dp"/>
```

76 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

```

<TextView
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_below="@+id/button2"
    android:layout_centerHorizontal="true"
    android:fontFamily="@font/arya_bold"
    android:gravity="center"
    android:text="See me!"
    android:textColor="#009999"
    android:textSize="25dp" />
</LinearLayout>

```

Step 3 : activity.xml का कोड लिखने के बाद। अब main_activity.java के लिए कोड इस प्रकार लिखेंगे—

```

package com.Mayank.mayankeventhandling;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.Button;
import androidx.appcompat.app.AppCompatActivity;
import android.widget.TextView;
public class MainActivity extends AppCompatActivity {
    Button btn;
    TextView tView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btn = (Button)findViewById(R.id.btnClick);
        tView = (TextView)findViewById(R.id.txtResult);
        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                TextView txtView = (TextView) findViewById(R.id.textView);
                txtView.setText("You've Clicked\n The Event has taken place");
                txtView.setTextSize(25);
                txtView.setGravity(Gravity.CENTER);
                txtView.setTextColor(R.color.colorAccent);
            }
        });
    }
}

```

Step 4 : अब इस क्रम में color.xml खोलें- app>src>main>res>values>colours.xml

```

<?xml version="1.0" encoding="utf-8"?>
<resources>
<color name="colorPrimary">#008577</color>
<color name="colorPrimaryDark">#00574B</color>

```

```
<color name="colorAccent">#D81B60</color>
</resources>
```

Step 5: उपरोक्त कोड लिखने के बाद, हम एप्लिकेशन को run करें।

3.5. Layouts

layout का उपयोग किसी app या activity के लिए user interface को परिभाषित करने के लिए किया जाता है और यह UI elements को रखेगा जो user को दिखाई देगा। एन्ड्रॉइड ऐप में यूजर इंटरफेस व्यू और व्यूग्रुप ऑब्जेक्ट्स के संग्रह के साथ बनाया गया है। जिसको chapter 1 के section 1.9 में discuss किया गया है।

3.5.1. layout के प्रकार

कई तरह के layout हैं। जिनमें से कुछ नीचे सूचीबद्ध हैं—

1. Linear layout
2. Absolute layout
3. Table layout
4. Frame layout
5. Relative layout

1. Linear layout—Linear layout को horizontal और vertical layout में विभाजित किया गया है। इसका मतलब है कि यह एक single column या single row में view को arrange कर सकता है। यहाँ linear layout का code है जिसमें एक text view शामिल है।

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.
com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/hello" />
</LinearLayout>
```

2. Absolute layout—Absolute लेआउट आपको अपने children के exact location को specify करने में enables बनाता है। इसे इस तरह declare किया जा सकता है।

```
<AbsoluteLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:android="http://schemas.android.com/apk/res/android" >

    <Button
        android:layout_width="188dp"
        android:layout_height="wrap_content"
        android:text="Button"
        android:layout_x="126px"
        android:layout_y="361px" />
</AbsoluteLayout>
```


78 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

3. Table layout—Table layout समूह rows और columns में देखता है। इसे इस तरह declare किया जा सकता है।

```
<TableLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_height="fill_parent"
    android:layout_width="fill_parent" >

    <TableRow>
        <TextView
            android:text="User Name:"
            android:width="120dp"
            />

        <EditText
            android:id="@+id/txtUserName"
            android:width="200dp" />
    </TableRow>

</TableLayout>
```

4. Frame layout—Relative layout आपको यह specify करने में enables करता है कि child views एक दूसरे के सापेक्ष कैसे positioned हैं। इसे इस तरह declare किया जा सकता है।

```
<RelativeLayout
    android:id="@+id/RLayout"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    xmlns:android="http://schemas.android.com/apk/res/android" >
</RelativeLayout>
```

5. Relative layout—Frame layout स्क्रीन पर एक placeholder है जिसे आप single view प्रदर्शित करने के लिए उपयोग कर सकते हैं। इसे इस तरह घोषित किया जा सकता है।

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/lblComments"
    android:layout_below="@+id/lblComments"
    android:layout_centerHorizontal="true" >

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

6. Coordinate layout—इस layout की power child views के बीच coordinate dependencies करने की क्षमता है। हम सभी को coordinator layout के अंदर views को wrap की आवश्यकता है। यह नया लेआउट behavior में परिभाषित नियमों के एक सेट पर आधारित है। Coordinator layout, frame layout की features को विस्तार करता है। ये दो primary use cases हैं—

- (i) एक top-level applications decor के रूप में।
- (ii) एक या अधिक child views के साथ एक specific interaction के लिए container के रूप में।


```
<CoordinatorLayout>
<AppBarLayout/>
<NestedScrollView/>
</CoordinatorLayout>
```

3.5.2. Units of measurement

जब आप Android UI पर किसी element का size specify कर रहे हैं, units of measurement निम्नलिखित हैं—

S. No.	Unit	Description
1.	dp/dip	Density-independent pixel —एक abstract unit जो स्क्रीन के physical density पर आधारित है। एक dp एक 160 dpi स्क्रीन पर एक पिक्सेल के बराबर है।
2.	sp	Scale-independent pixel —यह dp के समान है और font sizes specify करने के लिए recommended है।
3.	pt	Points —स्क्रीन के physical size के आधार पर एक इंच का 1/72
4.	px	Pixels —corresponds to actual pixels on the screen. स्क्रीन पर actual pixels के corresponds

3.6. Intents

एक intent एक messaging object है जिसका उपयोग आप किसी another app component से action perform करने के लिए कर सकते हैं।

एन्ड्रॉइड इंटेंट का उपयोग मुख्य रूप से किया जाता है—

- (i) Start the service
- (ii) Launch an activity
- (iii) Display a web page
- (iv) Display list of contacts
- (v) Broadcast a message
- (vi) Dial a phone call etc.

3.6.1. Importance

Intents कई तरीकों से components के बीच communication की सुविधा प्रदान करते हैं, तीन fundamental उपयोग के मामले जो निम्नलिखित हैं—

(i) **Starting in activity**—एक activity एक ऐप में एक स्क्रीन को represent करती है। आप एक activity का एक new instance शुरू करने के start A activity() में intent को pass करके शुरू कर सकते हैं। Intent किसी भी activity को start करने और आवश्यक डेटा को ले जाने के method का वर्णन करता है। यदि आप किसी activity के समाप्त होने पर एक परिणाम प्राप्त करना चाहते हैं, तो start activity for result () कॉल करें।

(ii) **Starting a service**—एक service एक component है जो user interface के बिना background में operations performs करता है। आप jobScheduler के साथ एक service शुरू कर सकते हैं।

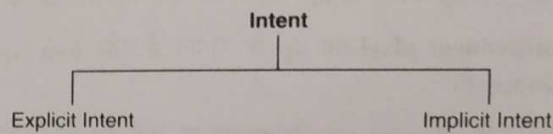
आप service class के तरीको का method करके service शुरू कर सकते हैं। आप one-time operation perform करने के लिए start service() में intent को pass करके एक service शुरू कर सकते हैं। Intent किसी भी activity को start करने और आवश्यक डेटा को ले जाने के method का वर्णन करता है।

यदि service client-server interface के साथ डिजाइन की गई है, तो आप किसी अन्य component में BindService() पर एक intent पास करके service को bind कर सकते हैं।

(iii) **Delivering a broadcast**—Broadcast एक message है जिसे कोई भी ऐप receive कर सकता है। सिस्टम, system events के लिए various broadcast को deliver करता है, जैसे कि जब सिस्टम boots up होता है या डिवाइस चार्ज करना शुरू होता है। आप SendBroadcast() या sendOrderedBroadcast() पर एक intent पास करके other apps पर broadcast कर सकते हैं।

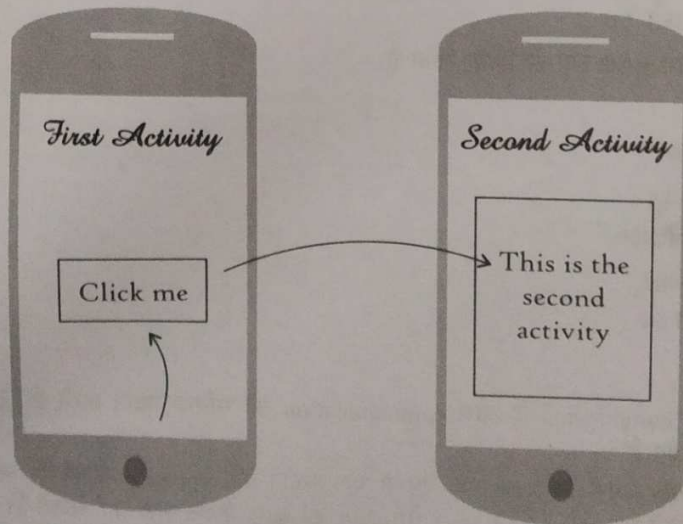
3.6.2. Intent types

Intents explicit या implicit हो सकते हैं—



Explicit intent—Explicit intents specify करते हैं कि कौन-सा target app's package name या a fully-qualified component class name का प्रयोग करके, intent को satisfy करेगा।

Explicit intent एप्लिकेशन की internal world से जुड़ा होता है, मान लीजिए कि आप किसी activity को किसी अन्य activity से connect करना चाहते हैं, तो हम इसको explicit intent से कर सकते हैं, नीचे diagram में पहली गतिविधि/दूसरी गतिविधि को बटन पर क्लिक करके जोड़ रही है।

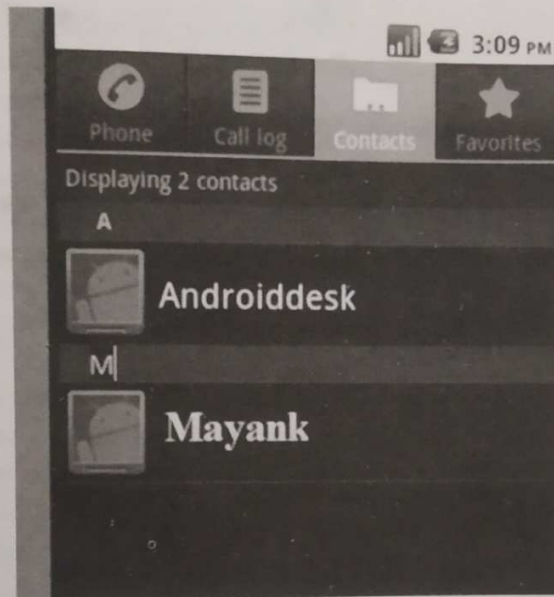


चित्र 3.4

ये intents अपने name में target component को नामित करते हैं और वे आमतौर पर application-internal messages के लिए उपयोग किए जाते हैं—जैसे कि एक activity एक subordinate service शुरू करने या launching a sister activity शुरू करने के लिए। उदाहरण के लिए—

```
// Explicit Intent by specifying its class name
Intent i = new Intent(FirstActivity.this, SecondActivity.class);
// Starts TargetActivity
startActivity(i);
```

ये intents किसी specific activity या intent को प्राप्त करने के लिए other component को specify नहीं करते हैं। इसके बजाय, आप action perform करने के लिए एक general action को declare करते हैं और एन्ड्रॉइड सिस्टम आपके request को किसी activity या other component से match करता है जो requested action को handle कर सकता है।



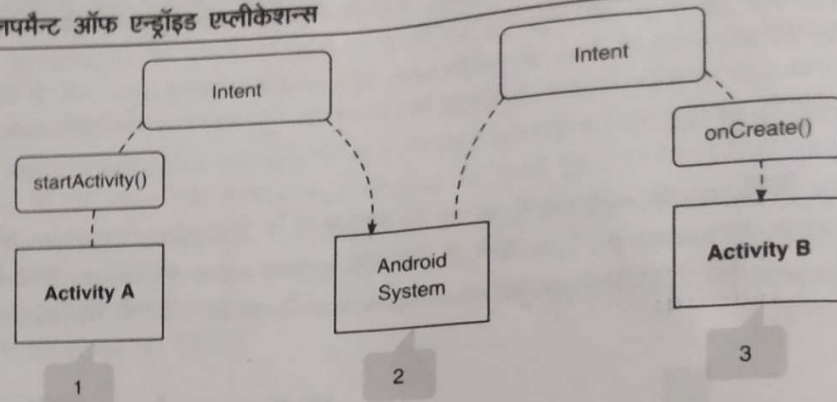
चित्र 3.5

अन्य application में component को activate करने के लिए अक्सर implicit intents का उपयोग किया जाता है। उदाहरण के लिए—

```
Intent read1=new Intent();
read1.setAction(android.content.Intent.ACTION_VIEW);
read1.setData(ContactsContract.Contacts.CONTENT_URI);
startActivity(read1);
```

उपरोक्त कोड नीचे दिखाए figure अनुसार result देगा—

चित्र दिखाता है कि activity शुरू करते समय एक intent का उपयोग कैसे किया जाता है। जब intent object किसी specific activity component को explicitly रूप से name देता है, तो सिस्टम तुरंत उस component को शुरू करता है।



चित्र 3.6

Implicit intent कैसे सिस्टम में अन्य activity शुरू करता है।

- (i) Activity A, action description के साथ intent को create करती है।
- (ii) Android सिस्टम सभी ऐप intent filter खोजता है जो intent से match करे, जब एक match पाया जाता है।
- (iii) सिस्टम अपने onCreate() method को invoke करके और intent को pass करके matching activity (Activity B) शुरू करता है।

3.6.3 Implementation of intent in android

इस एप्लिकेशन में, हम उपयोगकर्ता को गतिविधि से संबंधित URL पर ले जाएंगे।

Step 1 : सबसे पहले, एक नया प्रोजेक्ट बनाएँ और आवश्यक विवरण भरें।

Step 2 : अब, main_activity.xml फाइल खोलें और दिए गए कोड को लिखें।

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.mayank_intent.MainActivity">
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginLeft="100dp"
        android:text="DataFlair "
        android:textColor="@color/colorPrimaryDark"
        android:textSize="50dp" />
  
```



```

<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/urlText"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="100dp"
    android:layout_marginLeft="50dp"
    android:ems="10" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/browse"
    android:layout_below="@+id/urlText"
    android:text="Let's Go"
    android:layout_marginLeft="50dp"
    android:layout_centerHorizontal="true" />
</RelativeLayout>

```

Step 3 : उपरोक्त चरण के बाद main activity.java फाइल में निम्न कोड लिखें—

```

package com.Mayank.mayank_intent;
import android.content.Intent;
import android.net.Uri;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
public class MainActivity extends AppCompatActivity
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final EditText ditText = (EditText) findViewById(R.id.urlText);
        Button btn = (Button) findViewById(R.id.browse);
        btn.setOnClickListener(new View.OnClickListener()
        {
            @Override
            public void onClick(View v)
            {

```

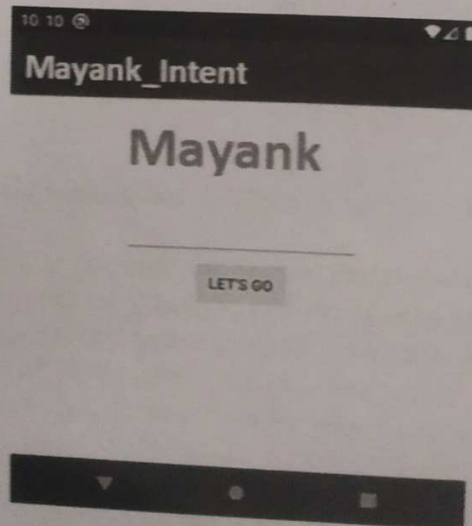
```
String url = editText.getText().toString();
Intent intent = new Intent(Intent.ACTION_VIEW,
Uri.parse(url));
startActivity(intent);
}
```

Step 4 : इसके बाद हमने ऐसा किया है। अगला कदम रन बटन को दबाना है।



चित्र 3.9

आपको वह URL दर्ज करना होगा जिसे आप एप्लिकेशन के माध्यम से खोलना चाहते हैं।



चित्र 3.8

प्रश्नावली

1. एक activity क्या है?
2. एन्ड्रॉइड activity lifecycle के methods क्या हैं?
3. ऐप डेवलपमेंट में सेटिंग परमिशन का क्या महत्व है?
4. Android में प्रयुक्त होने वाले UI Components को समझाइए।
5. Liner layout क्या है?
6. Relative layout को समझाइए।
7. Frame layout का वर्णन करें।
8. Intent क्या है?
9. Explicit intent क्या है?
10. Implicit intent क्या है?





मेन्यू और प्राथमिकताएँ (Menus and Preferences)

Introduction to menus, Types of menus (option menu, context menu), Uses of shared preferences.

4.1. Introduction

एन्ड्रॉइड में, Menu यूजर इंटरफ़ेस (UI) component का एक हिस्सा है जिसका उपयोग एप्लिकेशन के आस-पास कुछ सामान्य functionality को संभालने के लिए किया जाता है। हमारे अनुप्रयोगों में menu का उपयोग करके, हम applications में better और consistent यूजर experience प्रदान कर सकते हैं। हम अपने एन्ड्रॉइड एप्लिकेशन activities में यूजर actions और अन्य options को represent करने के लिए menu APIs का उपयोग कर सकते हैं।

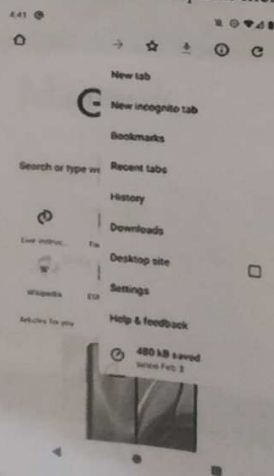
4.1.1. Android menu के प्रकार

एन्ड्रॉइड में 3 प्रकार के menu हैं—

1. Option menu
2. Context menu
3. Pop-up menu

1. Option menu

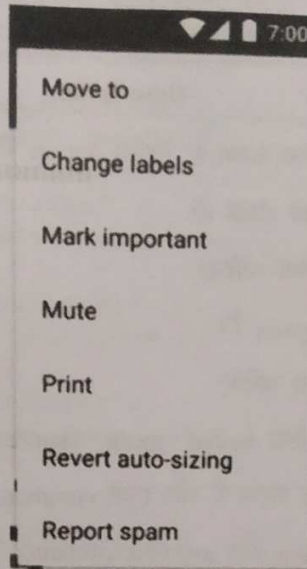
Option menu एक गतिविधि के लिए menu आइटम का प्राथमिक संग्रह है। Option menu एक गतिविधि के लिए विकल्पों का एक संग्रह है। इसमें उन वस्तुओं का एक सेट है जो क्रिया करने के लिए उपयोगी हैं। यह हमें एक साथ कई क्रियाओं को संयोजित करने में मदद करता है। निम्नलिखित option menu का एक उदाहरण है—



चित्र 4.1

2. Pop Up menu

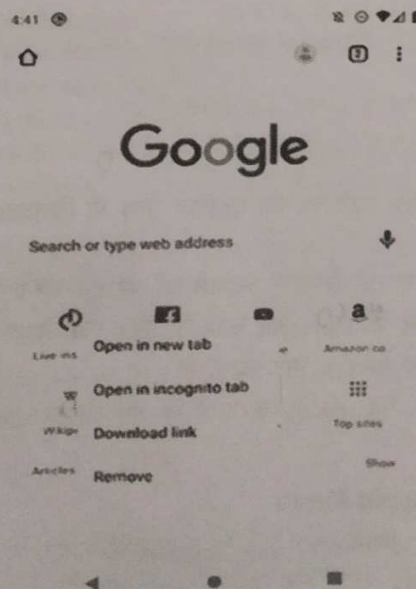
Pop-Up menu एक menu है जो Pop-Up विंडो में आइटमों की सूची प्रदर्शित करता है। Pop-Up menu डिफॉल्ट रूप से view के नीचे दिखाई देता है, यदि कोई स्थान नहीं है, तो यह इसके ऊपर दिखाई देता है।



चित्र 4.2

3. Contextual menu

एक contextual menu एक floating menu है। यह तभी appear होता है जब यूजर किसी element को long-press तक दबाते हैं या उस पर राइट क्लिक करते हैं। यह आमतौर पर selected element को प्रभावित करता है।



चित्र 4.3

4.2. How to Define Android Menu?

इसे define करने के लिए, हमें एक layout फाइल में इसके कोड को लिखना होगा और वह फाइल resource folder में res/menu/file.xml के रूप में बनाई जाती है। यहाँ, फाइल आपके द्वारा प्रदान किया जाने वाला कोई भी नाम हो सकता है। Menu के महत्वपूर्ण element निम्नलिखित हैं—

<element>	Description
<menu>	1. यह एक menu को परिभाषित करता है, जिसमें option होते हैं। 2. एक या अधिक element को रखता है। 3. यह एक फाइल का root होना चाहिए।
<items>	1. आइटम menu में मौजूद option हैं। 2. यह menu element में होना चाहिए। 3. इसमें submenu बनाने के लिए nested <menu> element हो सकता है।
<group>	1. Menu आइटम को वर्गीकृत करता है और इसमें common properties होते हैं। 2. यह element के लिए option और invisible container है।

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item
    android:id="@+id/search"
    android:title="Search Here" />
  <item
    android:id="@+id/filter"
    android:title="Apply Filter" />
  <item
    android:id="@+id/My_account"
    android:title="My Account" />
</menu>
```

किसी भी अन्य UI घटक की तरह यहाँ तक कि एन्ड्रॉइड मेन्यू भी विशेषताओं की मदद से अनुकूलित किया जा सकता है।

1. **android: id**—यह विशिष्ट रूप से मेन्यू के आइटम की पहचान करता है।
2. **android: icon**—यह आइटम को represent करने के लिए एक आइकन सेट करता है।
3. **android: title**—यह आइटम का title सेट करता है।
4. **android: showAsAction**—यह specify करता है कि यह आइटम app bar में कब और कैसे प्रदर्शित होना चाहिए।

4.3. Implementation of Android Menu

1. **Option menu**—Android एप्लिकेशन में option menu को बनाने का उदाहरण निम्नलिखित है—
 1. सबसे पहले, एक नया प्रोजेक्ट बनाएँ और OptionMenu नाम दें।

2. project resource directory (Res/menu l) के अंदर एक नया new folder menu बनाएँ और मेन्यू बनाने के लिए एक नई XML (Options_menu.xml) फाइल जोड़ें।

3. newly created xml (options_menu.xml) फाइल खोलें और नीचे दिखाए अनुसार कोड को लिखें—

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <item
    android:id="@+id/search"
    android:title="Search Here" />
  <item
    android:id="@+id/filter"
    android:title="Apply Filter" />
  <item
    android:id="@+id/wishlist"
    android:title="My Wishlist" />
  <item
    android:id="@+id/MyCart"
    android:title="My Cart" />
  <item
    android:id="@+id/My_account"
    android:title="My Account" />
</menu>
```

4. एक बार जब मेन्यू creation हो जाते हैं, तो onCreateOptionsMenu () कॉलबैक method का उपयोग करके अपनी activity से इस menu XML resource को load करने की आवश्यकता होती है, इसके लिए main activity फाइल MainActivity.java से java\com.mayank.options menu path को open करे और नीचे दिखाया गया कोड लिखें हैं—

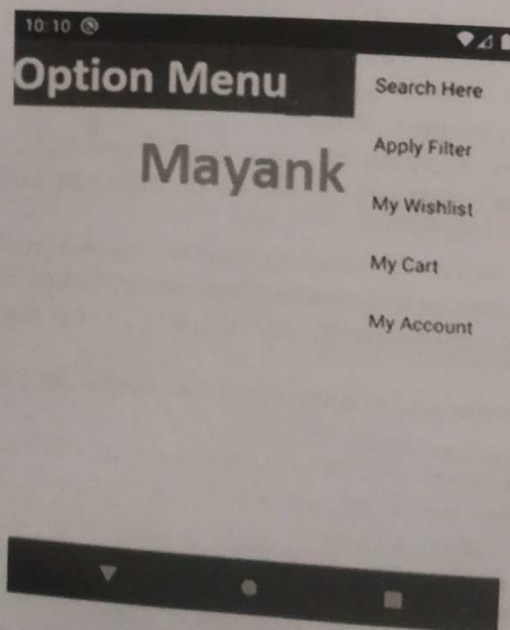
```
package com.mayank. optionsmenu;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    // Overriding onCreateOptionMenu() to make Option menu
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        //Inflating menu by overriding inflate() method of
        MenuInflater class.
        //Inflating here means parsing layout XML to views.
        getMenuInflater().inflate(R.menu.main_menu, menu);
        return true;
    }
}
```

```

//Overriding onOptionsItemSelected to perform event on menu items
@Override
public boolean onOptionsItemSelected(MenuItem menuItem) {
    Toast.makeText(this, "You chose : " + menuItem.getTitle(),
        Toast.LENGTH_SHORT).show();
    switch (menuItem.getItemId()) {
        case R.id.search:
            //Your code here
            return true;
        case R.id.filter:
            //Your code here
            return true;
        case R.id.wishlist:
            //Your code here
            return true;
        case R.id.MyCart:
            //Your code here
            return true;
        case R.id.My_account:
            //Your code here
            return true;
        default:
            return super.onOptionsItemSelected(menuItem);
    }
}

```

This is the menu :



2. **Context menu**—Android एप्लिकेशन में context menu को बनाने का उदाहरण निम्नलिखित है—
1. सबसे पहले, एक नया प्रोजेक्ट बनाएँ और context menu नाम दें।
 2. project resource directory (Res/menu) के अंदर एक नया new folder menu बनाएँ और मेन्यू बनाने के लिए एक नई XML (contextMenu.xml) फाइल जोड़ें।

3. newly created xml (ContextMenu.xml) फाइल खोलें और नीचे दिखाए अनुसार कोड को लिखें—

```
<?xml version="1.0" encoding="utf-8"?> <LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" > <Button
    android:id="@+id/btnShow"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Mayank"
    android:layout_marginTop="200dp"
    android:layout_marginLeft="100dp"/>
</LinearLayout>
```

4. एक बार जब मेन्यू creation हो जाते हैं उसके बाद MainActivity.java फाइल में निम्न कोड लिखें—

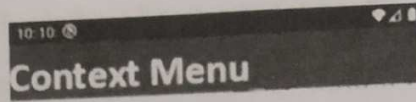
```
package com.Mayank.contextmenu;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button btn = (Button) findViewById(R.id.btnShow);
        registerForContextMenu(btn);
    } @Override
    public void onCreateContextMenu(ContextMenu menu, View v,
    ContextMenu.ContextMenuInfo menuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);
        menu.setHeaderTitle("Context Menu");
        menu.add(0, v.getId(), 0, "Upload");
        menu.add(0, v.getId(), 0, "Search");
        menu.add(0, v.getId(), 0, "copy");
        menu.add(0, v.getId(), 0, "Paste");
    }
}
```

```

}
@Override
public boolean onContextItemSelected(MenuItem item) {
    Toast.makeText(this, "Selected Item: " + item.getTitle(),
        Toast.LENGTH_SHORT).show(); return true;
}
}

```



Mayank Button

Upload
Search
Copy
Paste



चित्र 4.5

4.4. Shared Preferences

Android किसी एप्लिकेशन के डेटा को store करने के कई तरीके प्रदान करता है। इनमें से एक shared preferences है। Shared preferences आपको of key, value pair के रूप में डेटा को store और retrieve करने की अनुमति देता है। Preference file को handle करने के लिए और preference data को पढ़ने, लिखने और manage करने के लिए, sharedPreferences class का उपयोग करें।

Preference file—एक preference file वास्तव में डिवाइस की internal मेमोरी में save की गई एक xml फाइल है। हर एप्लिकेशन के पास कुछ डेटा होता है जो किसी डायरेक्टरी डेटा/डेटा/एप्लिकेशन पैकेज नाम में मेमोरी में संग्रहीत होता है, इसलिए जब भी getSaredPreferences () function को call किया जाता है, तो यह फाइल को validates करता है कि यदि यह मौजूद है या यह नहीं है तो एक नया xml, passed name के साथ बनाया जाता है। Shared preferences का उपयोग करने के लिए आपको getSharedPreferences () method को call करना होगा, जो कि shared preference instance रिटर्न फाइल को इंगित करता है जिसमें values of preferences शामिल हैं।

```

SharedPreferences sharedPreferences =
    getSharedPreferences(MyPREFERENCES, Context.MODE_PRIVATE);

```

पहला पैरामीटर key है और दूसरा पैरामीटर MODE है। Private mode के अलावा अन्य मोड उपलब्ध हैं जो नीचे listed हैं।

- (i) **MODE_APPEND**- यह पहले से मौजूद preferences के साथ नई preferences को जोड़ **MODE_ENABLE_WRITE_AHEAD_LOGGING**- डेटाबेस open flag है। जब यह set हो जाता है, तो यह डिफॉल्ट रूप से आगे लॉगिंग करने में सक्षम होगा।
- (ii) **MODE_MULTI_PROCESS**-यह method preferences के संशोधन के लिए जाँच करेगी भले ही the shared preference instance पहले से लोड किया गया हो।
- (iii) **MODE_PRIVATE** फाइल को केवल calling एप्लिकेशन का उपयोग करने एक्सेज किया जा सकता है।
- (iv) **MODE_WORLD_READABLE**-अन्य एप्लिकेशन को preferences पढ़ने की अनुमति दें।
- (v) **MODE_WORLD_WRITEABLE**-अन्य एप्लिकेशन को preferences लिखने की अनुमति दें।

You will call the edit method of shared preference instance and will receive it in an editor object. Its syntax is –

आप SharedPreferences.Editor class का उपयोग करके sharedPreferences में कुछ save कर सकते हैं। आप SharedPreferences instance के edit method को call कर सकते हैं और इसे एक editor object में प्राप्त कर सकते हैं। इसका syntax -

```
Editor editor = sharedPreferences.edit();
editor.putString("key", "value");
editor.commit();
```

putString method के अलावा, editor class में उपलब्ध methods हैं जो shared preferences के अंदर डेटा के manipulation की अनुमति देती हैं। वे इस प्रकार हैं—

(i) **apply()** - यह एक abstract method है। यह editor से आपके द्वारा साझा किए जा रहे SharedPreferences object पर आपके परिवर्तनों को वापस करेगा।

(ii) **clear()** - यह editor से सभी values को remove कर देगा।

(iii) **remove(String key)**-यह उस value को हटा देगा जिसकी key एक parameter के रूप में pass की गई है।

(iv) **putLong(String key, long value)** - यह एक preference editor में long value को बचाएगा।

(v) **putLong(String key, int value)** - यह preference editor में integer value में बचाएगा।

(vi) **putFloat(String key, float value)**-यह एक preference editor में float value बचाएगा।

उदाहरण

दो edit texts हैं, जो पहले दर्ज किए गए डेटा को save और retain रखते हैं। Applications forms इस प्रकार की सुविधा देखी जा सकती है। Shared preferences का उपयोग करते हुए, User को बार-बार विवरण नहीं भरना होगा। UI को लागू करने के लिए activity_main.xml फाइल के अंदर निम्नलिखित कोड शामिल करें—

activity_main.xml file

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    tools:ignore="HardcodedText">
```



```

<TextView
    android:id="@+id/textview"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="32dp"
    android:text="Shared Preferences Demo"
    android:textColor="@android:color/black"
    android:textSize="24sp" />

<!--EditText to take the data from the user
    and save the data in SharedPreferences-->
<EditText
    android:id="@+id/edit1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/textview"
    android:layout_marginStart="16dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="16dp"
    android:hint="Enter your Name"
    android:padding="10dp" />

<!--EditText to take the data from the user and
    save the data in SharedPreferences-->
<EditText
    android:id="@+id/edit2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/edit1"
    android:layout_marginStart="16dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="16dp"
    android:hint="Enter your Age"
    android:padding="10dp" />
</RelativeLayout>

```

MainActivity.java file

```

import androidx.appcompat.app.AppCompatActivity;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.EditText;
public class MainActivity extends AppCompatActivity {
    private EditText name, age;
    @Override

```



```

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    name = findViewById(R.id.edit1);
    age = findViewById(R.id.edit2);
}

// Fetch the stored data in onResume()
// Because this is what will be called
// when the app opens again
@Override
protected void onResume() {
    super.onResume();

    // Fetching the stored data
    // from the SharedPreferences
    SharedPreferences sh = getSharedPreferences("MySharedPref",
        MODE_PRIVATE);

    String s1 = sh.getString("name", "");
    int a = sh.getInt("age", 0);

    // Setting the fetched data
    // in the EditTexts
    name.setText(s1);
    age.setText(String.valueOf(a));
}

// Store the data in the SharedPreferences
// in the onPause() method
// When the user closes the application
// onPause() will be called
// and data will be stored
@Override
protected void onPause() {
    super.onPause();

    // Creating a shared pref object
    // with a file name "MySharedPref"
    // in private mode
    SharedPreferences sharedPreferences = getSharedPreferences(
        "MySharedPref", MODE_PRIVATE);
    SharedPreferences.Editor myEdit = sharedPreferences.edit();
    // write all the data entered by the user in
    // SharedPreferences and apply myEdit.putString("name",

```

6

डैवलपमेंट ऑफ एंड्रॉइड एप्लीकेशन्स

```
name.getText().toString());  
myEdit.putInt("age",  
Integer.parseInt(age.getText().toString()));  
myEdit.apply();  
}
```

प्रश्नावली

1. Menu को define करें।
2. Android में प्रयुक्त होने वाले menu के प्रकार समझाइए।
3. Option menu क्या होता है?
4. Context menu को program को help से समझाइए।
5. Shared preferences क्या होता है?
6. Android में shared preferences का उपयोग बताइए।



अग्रिम UI घटक (Advanced UI Components)

Time and Date, List View, Grid View, Card View, recycler view Adaptors (Base Adaptor, Array Adaptor) & View Holder, Dialogs, Toast, Popup, Fragments, Material Design(Introduction, Navigation, Floating Button , Tool bar).

5.1. Current Time and Date on Android

ओरेकल दस्तावेज के अनुसार, SimpleDateFormat एक स्थान-संवेदनशील तरीके से तिथियों को formatting और parsing करने के लिए एक class है। इस उदाहरण में, हमने नीचे दिखाए गए जावा से सरल तिथि प्रारूप CLASS का import किया है—

```
import java.text.SimpleDateFormat;
import java.util.Date;
```

Step 1 : ANDROID स्टूडियो में एक नया प्रोजेक्ट बनाएं, File ⇒ New Project पर जाएँ और एक नया प्रोजेक्ट बनाने के लिए सभी आवश्यक विवरण भरें।

Step 2 : निम्नलिखित कोड जोड़ें res/layout/activity_main.xml.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:gravity="center"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/date"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:textSize="30sp"
        android:layout_marginBottom="36dp" />
</RelativeLayout>
```

उपरोक्त कोड में, हमने Text View दिया है, यह विंडो मैनेजर पर वर्तमान तिथि print करने जा रहा है।

Step 3 : निम्नलिखित कोड जोड़ें src/MainActivity.java

```
package com.example.myapplication;
import android.annotation.TargetApi;
```

```

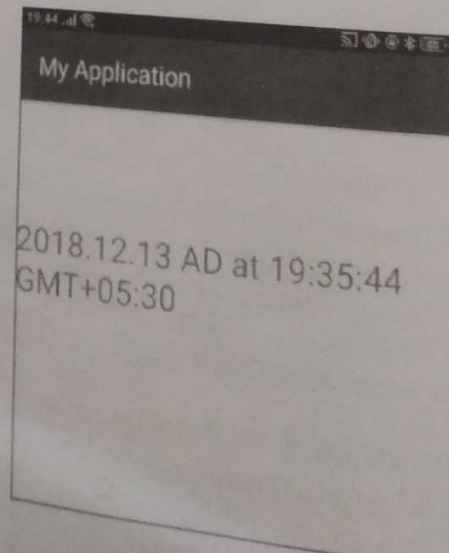
import android.os.Build;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.widget.TextView;
import java.text.SimpleDateFormat;
import java.util.Date;

public class MainActivity extends AppCompatActivity {
    @TargetApi(Build.VERSION_CODES.O)
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        TextView textView=findViewById(R.id.date);
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy.MM.dd G 'at'
        HH:mm:ss z");
        String currentDateandTime = sdf.format(new Date());
        textView.setText(currentDateandTime);
    }
}

```

उपरोक्त कोड में हम SimpleDateFormat को कॉल कर रहे हैं और SimpleDateFormat से, हम स्ट्रिंग में वर्तमान तिथि तक पहुँच रहे हैं।

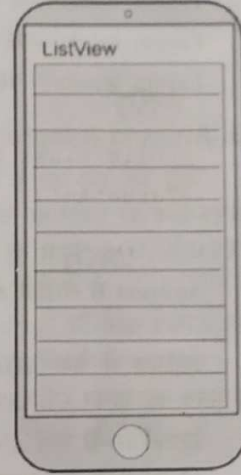
Step 4 : manifest.xml को बदलने की कोई जरूरत नहीं। चलो अपना आवेदन Run करते हैं—
उपरोक्त उदाहरण में, इसमें एक वर्तमान तिथि और समय शामिल है।



ListView

- ANDROID **ListView** एक View है जो कई items को समूह बनाता है और उन्हें vertical स्क्रॉल करने योग्य सूची में प्रदर्शित करता है। सूची आइटम स्वचालित रूप से एक **Adapter** का उपयोग करके सूची में डाला जाता है जो किसी स्रोत या डेटाबेस जैसे **MATERIAL** को खींचता है।
- ANDROID में **LISTVIEW** आपके डेटा को सूची रूप में बदलने का समय।
- ANDROID में **LISTVIEW** एक प्रकार का व्यूग्रूप है जो पंक्तियों के रूप में वस्तुओं की सूची दिखाता है। **LISTVIEW** में लिस्ट आउटकआउट **LISTVIEW** का उपयोग करके बनाया गया है। यह ANDROID में सबसे महत्वपूर्ण यूआई घटकों में से एक है। यह वर्टिकल रूप में सूची दिखाता है और यह स्वचालित रूप से स्क्रॉल हो जाता है।

ADAPTER का उपयोग करके आइटम स्वचालित रूप से जोड़े जाते हैं। एक **ADAPTER** मूल रूप में यूआई घटक और वस्तुओं के लिए डेटा स्रोत के बीच Bridge का कार्य करता है। एक **ADAPTER** डेटा रखता है और उसे **ADAPTER** व्यू में भेजता है उसके बाद व्यू **ADAPTER** व्यू से डेटा लेता है और इसे **ListsViews** में दिखाता है।



चित्र 2

ANDROID में **LISTVIEW** के पैरामीटर

इस विधि में 4 पैरामीटर पारित किए जाते हैं—

LISTVIEW : इसमें **VIEW** का आइटम होता है।

VIEW : यह उस विशिष्ट **VIEW** को बताता है जिसे चुना गया था।

State : यह सारणी में चयनित आइटम की स्थिति बताता है।

Id : आईडी का उपयोग किसी **VIEW** को विशिष्ट रूप से पहचानने के लिए किया जाता है।

हमारे आवेदन में एक **LISTVIEW** जोड़ने के लिए हम इसे निम्नलिखित का उपयोग करके परिभाषित करते हैं—

```
<ListView xmlns:android=
"http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/MyLV"
>
```

ANDROID में **LISTVIEW** के गुण

ANDROID में **LISTVIEW** को अनुकूलित करने के लिए, हम इसमें कुछ विशेषताएँ जोड़ सकते हैं, जैसे—

Id—आईडी विशिष्ट रूप से एक **LISTVIEW** की पहचान करता है।

डिवाइडर—यह विभिन्न सूची वस्तुओं के बीच रंग या आकर्षित को परिभाषित करता है।

डिवाइडरहाइट—यह सूची के बीच डिवाइडर की ऊँचाई को परिभाषित करता है।

आइटम लिस्टसेल्टर—यह संपत्ति **LISTVIEW** के लिए चयनकर्ता सेट करती है।

प्रविष्टियाँ—यह सारणी संसाधन के संदर्भ को निर्दिष्ट करता है।

पृष्ठभूमि—यह **LISTVIEW** की पृष्ठभूमि सेट करता है। पृष्ठभूमि या तो रंग या यहाँ तक कि एक छवि हो सकती है।

ChoiceMode—यह उन वस्तुओं की संख्या निर्धारित करता है जिन्हें एक बार में चुना जा सकता है।

टेक्स्टसाइज—यह प्रदर्शित किए जाने वाले पाठ का आकार सेट करता है।

Layout_width—यह लेआउट की चौड़ाई निर्धारित करता है।

Layout_height—यह लेआउट की ऊँचाई सेट करता है।

Layout_gravity—यह लेआउट की गंभीरता को सेट करता है।

5.3. Android Adapter

जैसा कि हम जानते हैं, LISTVIEW में ADAPTER का प्रयोग व्यू को स्रोत से डेटा लाने और उपयोगकर्ताओं को स्क्रीन पर दिखाने के लिए किया जाता है। अब यहाँ हम LISTVIEW के बारे में बात कर रहे हैं, और LISTVIEW ADAPTER व्यू क्लास का एक उप CLASS है और हमारे पास ANDROID में दो ADAPTER हैं—

Android से सारणी Adapter—जब हम एक सारणी द्वारा समर्थित एकल वस्तुओं की सूची चाहते हैं तो हम इसका उपयोग करते हैं।

एंड्रॉयड में बेस ADAPTER—यह LISTVIEW में उपयोग किए जाने वाले ADAPTER के सामान्य कार्यान्वयन का आधार CLASS है।

डेटाफ्लैयर के साथ ANDROID ADAPTER की गहरी जानकारी प्राप्त कर सकते हैं।

ANDROID में LISTVIEW कैसे बनाएँ—

अब, ANDROID LISTVIEW में, हम Array ADAPTER और बेस ADAPTER का उपयोग करके ANDROID में LISTVIEW को लागू करने का तरीका सीखने जा रहे हैं। आइए सबसे पहले ANDROID में सारणी ADAPTER के साथ शुरू करें।

A. ANDROID में सारणी ADAPTER का उपयोग करना।

अब हम निम्नलिखित चरणों के साथ Array अडेटर का उपयोग करके अपने आवेदन में इस LISTVIEW को लागू करेंगे—

1. इसे लागू करने के लिए, हम पहले एक नई परियोजना बनाएँगे और इसका नाम देंगे। फिर हम activity_main.xml फाइल में LISTVIEW इस प्रकार बनाएँगे—

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <ListView
        android:id="@+id/listview1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:divider="#07755C"
        android:dividerHeight="2dp"></ListView>
</LinearLayout>
```

2. इसके बाद हम निम्नलिखित कोड लिखेंगे MainActivity.java इस प्रकार फाइल करेंगे—

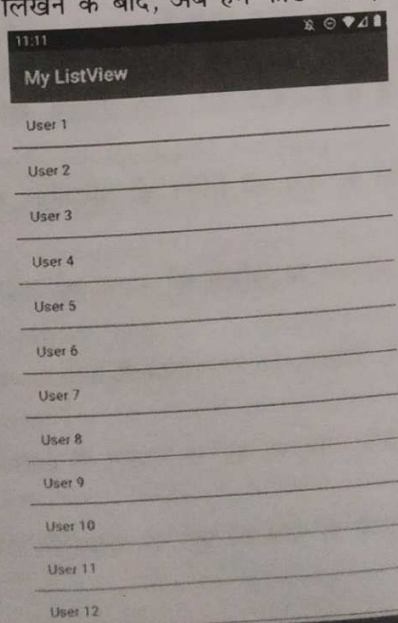
```
package com.DataFlair.mylistview;

import android.os.Bundle;
import android.widget.AdapterView;
import android.widget.ListView;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    private ListView mylist;
    private ArrayAdapter myAdapter;
    private String[] names = {"User 1", "User 2", "User 3", "User 4", "User 5", "User 6", "User 7", "User 8", "User 9", "User 10", "User 11", "User 12", "User 13", "User 14", "User 15", "User 16", "User 17", "User 18", "User 19", "User 20"};

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mylist = findViewById(R.id.listview1);
        myAdapter = new ArrayAdapter(this,
            android.R.layout.simple_list_item_1, names);
        mylist.setAdapter(myAdapter);
    }
}
```

3. सभी कोड लिखने के बाद, अब हम कोड चलाएँगे।



5.4. Android Grid View

ANDROID GRIDVIEW लेआउट हमारे पास बहुत महत्वपूर्ण लेआउट में से एक है। यह हमें ग्रिड के रूप में डेटा दिखाने में मदद करता है ताकि हम एक समय में कई छवियों या माउस को दिखा सकें। हम गैलरी से ऑडियो, वीडियो या छवियों को दिखाने के लिए आवेदन में इसका उपयोग करते हैं।

एक ग्रिड मूल रूप से 2-D दृश्य में वस्तुओं की सूची दिखाता है। माउस या छवियों की सूची का 2-D दृश्य निम्नलिखित आरेख android GRIDVIEW की तरह पंक्तियों और स्तम्भों के रूप में दिखाया गया है।

यदि आइटम की सूची स्क्रीन में फिटिंग नहीं है, तो यह स्वचालित रूप से स्क्रॉल करने योग्य हो जाती है, हमें स्क्रॉल दृश्य या इसके साथ कुछ करने की आवश्यकता नहीं है। ग्रिड ANDROID GRIDVIEW की मदद से बनाया गया है जो कंटेनरों में मौजूद है। एक ग्रिड कंटेनरों की सूची से घसीटा जा सकता है और स्क्रीन पर गिरा दिया जा सकता है। लेआउट को माउस या छवियों से भरने के लिए हमें ADAPTER का उपयोग करने की आवश्यकता है। एक ADAPTER बस स्रोत से MATERIAL ले जाकर GRIDVIEW में डेटा सम्मिलित करता है।

ANDROID ADAPTER का उपयोग करके ADAPTER में डेटा आइटम तक पहुँचने का तरीका देखें।

Android Gridview की विशेषताएँ

जब हम ANDROID GRIDVIEW को परिभाषित करते हैं, तो हम इसे अपनी आवश्यकताओं और इच्छा के अनुसार अनुकूलित भी कर सकते हैं। ग्रिड लेआउट को अनुकूलित करने के लिए हम निम्नलिखित विशेषताओं का उपयोग करते हैं—

एन्ड्रॉइड : आईडी—आवश्यकता के मामले में इसे चुनने के लिए एक ग्रिड को विशिष्ट रूप से पहचाना जाना चाहिए। इस उद्देश्य के लिए, इस विशेषता का उपयोग करके प्रत्येक ग्रिड के लिए आईडी को परिभाषित किया गया है।

ANDROID : Gravity—यह विशेषता GRIDVIEW के संरेखण को सेट करता है। हम ग्रिड का उपयोग करके, बाएँ, दाएँ, नीचे आदि का उपयोग करके संरेखित कर सकते हैं।

ANDROID : कॉलमविड्थ—इस विशेषता का उपयोग कॉलम की चौड़ाई को डीपी, एसपी, पीएक्स, इन या मिमी का उपयोग करके एक निश्चित आकार में सेट करने के लिए किया जाता है।

ANDROID : स्ट्रेचमोड—स्ट्रेचमोड विशेषता का उपयोग किसी भी सेल में खाली जगह को फैलाने और भरने के लिए किया जाता है। जो मान इसे धारण कर सकते हैं वे हैं—कोई नहीं, स्पेसिंगविड्थ, कॉलमविड्थ, स्पेसिंगविड

ANDROID : क्षैतिज अंतर—यह विशेषता स्तंभों के बीच की रिक्ति को परिभाषित करती है। रिक्ति डीपी, एसपी, पीएक्स, इन या एमएम में हो सकती है।

ANDROID : न्यूकोल्यूमन—यह विशेषता उन आइकन या छवियों की संख्या को परिभाषित करती है जिन्हें कॉलम में व्यवस्थित किया जा सकता है।

ANDROID : ऊँचाई—यह विशेषता डीपी, एसपी, पीएक्स, एन या एमएम में ANDROID GRIDVIEW की ऊँचाई को सेट करता है।

ANDROID : चौड़ाई—यह विशेषता डीपी, एसपी, पीएक्स, इन या एमएम में ANDROID GRIDVIEW की चौड़ाई निर्धारित करता है।

ANDROID : वर्टिकल अंतर—यह विशेषता पंक्तियों के बीच की दूरी को परिभाषित करती है। रिक्ति डीपी, एसपी, पीएक्स, इन या एमएम में हो सकती है।

Android में Gridview को कैसे लागू करें—

अब यह देखने के लिए ANDROID GRIDVIEW उदाहरण को लागू करने का समय है कि हम ग्रिड फॉर्म : छवियों को कैसे व्यवस्थित कर सकते हैं। हमें कदम-से-कदम की प्रक्रिया के बाद के साथ शुरू करते हैं।

चरण 1—सबसे पहले हम एक नई परियोजना बनाएँगे और इसका नाम लेंगे। फिर हम ANDROID GRIDVIEW को activity_main.xml फाइल में फॉलो भी परिभाषित करेंगे—

```
<?xml version="1.0" encoding="utf-8"?>
<GridView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/mygrid"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:columnWidth="110dp"
    android:numColumns="auto_fit"
    android:stretchMode="columnWidth"
    android:verticalSpacing="15dp"
    android:horizontalSpacing="15dp"
    android:gravity="center" />
```

चरण 2—इसके बाद, अब हम एक फाइल बनाएँगे, जिसका नाम SetImageAdapter.java फाइल है। हम इसे उसी फोल्डर में बनाएँगे, जहाँ हमारे पास MainActivity.java फाइल है। अब, हम SetImageAdapter.java में निम्नलिखित कोड लिखेंगे—

```
package com.DataFlair.mycats;

import android.content.Context;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.GridView;
import android.widget.ImageView;

public class SetImageAdapter extends BaseAdapter {
    public Integer[] thumbImages = {
        R.drawable.img1, R.drawable.img2,
        R.drawable.img11, R.drawable.img12,
        R.drawable.img3, R.drawable.img13,
        R.drawable.img4, R.drawable.img9,
        R.drawable.img5, R.drawable.img8,
        R.drawable.img1, R.drawable.img1,
        R.drawable.img6, R.drawable.img7,
        R.drawable.img11, R.drawable.img10,
        R.drawable.img1, R.drawable.img2,
```

```

        R.drawable.img5, R.drawable.img8,
        R.drawable.img1, R.drawable.img1,
        R.drawable.img6, R.drawable.img7,
        R.drawable.img11, R.drawable.img10,
        R.drawable.img1, R.drawable.img2,
        R.drawable.img11, R.drawable.img12,
        R.drawable.img3, R.drawable.img13,
        R.drawable.img4, R.drawable.img9,
        R.drawable.img5, R.drawable.img8,
        R.drawable.img1, R.drawable.img1,
        R.drawable.img6, R.drawable.img7
    };
    private Context Cont;
    public SetImageAdapter(Context c) {
        Cont = c;
    }
    public int getCount() {
        return thumbImages.length;
    }
    public Object getItem(int position) {
        return null;
    }
    public long getItemId(int position) {
        return 0;
    }
    public View getView(int position, View convertView, ViewGroup
    parent){
        ImageView imgview = new ImageView(Cont);
        imgview.setLayoutParams(new GridView.LayoutParams(370, 250));
        imgview.setScaleType(ImageView.ScaleType.CENTER_CROP);
        imgview.setPadding(10,10,10, 10);
        imgview.setImageResource(thumbImages[position]);
        return imgview;
    }
}

```

चरण 3—अब हम MainActivity.java फाइल निम्नलिखित कोड लिखेंगे—
 package com.DataFlair.mycats;

```
import android.os.Bundle;
import android.widget.GridView;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        GridView gv = findViewById(R.id.mygrid);
        gv.setAdapter(new SetImageAdapter(this));
    }
}
```

चरण 4—उसके बाद, हम स्क्रीन पर निम्नलिखित ऐप चलाएंगे—



चित्र 5.3

CardView in Android with Example

CARDVIEW ANDROID में एक नया विजेट (an small gadget or application) है जिसका उपयोग एक विशिष्ट ऊँचाई के साथ गोल कॉर्नर लेआउट प्रदान करके किसी भी प्रकार के डेटा को प्रदर्शित करने के लिए किया जा सकता है। CARDVIEW वह दृश्य है जो एक-दूसरे के ऊपर दृश्य प्रदर्शित कर सकता है। CARDVIEW का मुख्य

उपयोग यह है कि यह यूआई डिजाइन को समझ महसूस करने और देखने में मदद करता है।
अलग-अलग ANDROID ऐप्स में आसानी से देखा जा सकता है। CARDVIEW का उपयोग LISTVIEW में आइटम बनाने या रीसाइक्लर व्यू के अंदर आइटम बनाने के लिए किया जा सकता है। CARDVIEW के बारे में सबसे अच्छी बात यह है कि यह फ्रेमलेआउट का विस्तार करता है और इसे ANDROID के सभी प्लेटफार्मों पर प्रदर्शित किया जा सकता है। अब हम CARDVIEW कार्यान्वयन का सरल उदाहरण देखेंगे।

चरण 1—नया ANDROID स्टूडियो प्रोजेक्ट बनाएँ।
नए एंड्रॉइड स्टूडियो परियोजना बनाने के लिए File > न्यू > न्यू प्रोजेक्ट पर क्लिक करें।
अपनी भाषा को जावा के रूप में रखना सुनिश्चित करें और खाली गतिविधि का चयन करें।

चरण 2—build.gradle फाइल में MATERIAL निर्भरता जोड़ें।
बिल्डल स्क्रिप्ट पर नेविगेट करें ताकि build.gradle (ऐप) हो और फिर इसमें निर्भरता से नीचे जोड़ें।
इस निर्भरता को जोड़ने के बाद आपको शीर्ष दाएँ कोने में एक पॉपअप विकल्प दिखाई देगा जो अब सिंक प्रदर्शित करता है। अपनी परियोजना को सिंक करने के लिए उस विकल्प पर क्लिक करें।

चरण 3—अब हम एक सरल CARDVIEW बनाएँगे।

ऐप > रॉइटआउट पर नेविगेट करें > activity_main.xml फिर इसके लिए नया CARDVIEW विजेट बनाएँ।

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<RelativeLayout
```

```
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
xmlns:tools="http://schemas.android.com/tools"
```

```
android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
```

```
xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
tools:context=".MainActivity">
```

```
<androidx.cardview.widget.CardView
```

```
android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
```

```
app:cardElevation="10dp"
```

```
app:cardCornerRadius="20dp"
```

```
android:layout_margin="10dp"
```

```
app:cardBackgroundColor="@color/white"
```

```
app:cardMaxElevation="12dp"
```

```
app:cardPreventCornerOverlap="true"
```

```
app:cardUseCompatPadding="true"
```

```
>
```

```
<ImageView
```

```
android:layout_width="200dp"
```



```

        android:layout_height="200dp"
        android:layout_gravity="center"
        android:src="@drawable/gfgimage"
        android:layout_margin="10dp"
        android:id="@+id/img"
        android:contentDescription="@string/app_name" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/app_name"
        android:layout_gravity="bottom|center_horizontal"
        android:layout_marginTop="20dp"
        android:layout_marginBottom="20dp"
        android:textSize="20sp"
        android:textStyle="bold"
    />
</androidx.cardview.widget.CardView>
</RelativeLayout>

```

अब अपने एम्युलेटर या डिवाइस पर अपना ऐप चलाएँ आपको आउटपुट देखने को मिलेगा।

आउटपुट—



चित्र 5.5

RecyclerView in Android with Example

RECYCLEVIEW GRIDVIEW और LISTVIEW के उत्तराधिकारी के रूप में ANDROID स्टूडियो में जोड़ा गया व्यूग्रुप है। यह एक आइटम के रूप में XML लेआउट के साथ किसी भी सूची का सम्भव निर्माण करने के लिए बनाया गया है जिसे LISTVIEW और GRIDVIEW की दक्षता में सुधार करते हुए काफी अनुकूलित किया जा सकता है। यह सुधार उन विचारों को रीसाइक्लिंग करके हासिल किया जाता है जो उपयोगकर्ता की दृश्यता से बाहर हैं। उदाहरण के लिए, यदि कोई उपयोगकर्ता उस स्थिति में स्क्रॉल करता है जहाँ आइटम 4 और 5 दिखाई देते हैं; आइटम 1, 2 और 3 स्मृति की खपत को कम करने के लिए स्मृति से मंजूरी दे दी जाएगी।

Working : एक बुनियादी रीसाइक्लरव्यू को लागू करने के लिए तीन उप-भागों का निर्माण करने की आवश्यकता होती है जो उपयोगकर्ताओं को उनकी पसंद के अलग-अलग डिजाइन बनाने में आवश्यक नियंत्रण की डिग्री प्रदान करते हैं।

कार्ड लेआउट—कार्ड लेआउट एक एक्सएमएल लेआउट है जिसे रीसाइक्लरव्यू द्वारा बनाई गई सूची के लिए एक आइटम के रूप में माना जाएगा।

व्यूहोल्डर—व्यूहोल्डर एक जावा CLASS है जो कार्ड लेआउट दृश्यों के संदर्भ को संग्रहित करता है जिसे ऑनलाइन डेटाबेस द्वारा प्राप्त डेटा की सूची द्वारा कार्यक्रम के निष्पादन के दौरान गतिशील रूप से संशोधित किया जाना है या किसी अन्य तरीके से जोड़ा जाता है।

डेटा क्लास—डेटा क्लास एक कस्टम जावा CLASS है जो रीसाइक्लरव्यू के हर आइटम के लिए जानकारी रखने के लिए एक संरचना के रूप में कार्य करता है।

```
// ViewHolder code for RecyclerView
package com.example.admin.example;
import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.TextView;
public class examViewHolder
    extends RecyclerView.ViewHolder {
    TextView examName;
    TextView examMessage;
    TextView examDate;
    View view;
    examViewHolder(View itemView)
    {
        super(itemView);
        examName
        = (TextView)itemView
        examViewHolder.java
        .findViewById(R.id.examName);
        examDate
        = (TextView)itemView
        .findViewById(R.id.examDate);
        examMessage
        = (TextView)itemView
        .findViewById(R.id.examMessage);
    }
}
```

```

        view = itemView
    }
}

package com.prodigieux.admin.prodigieux;

public class examData {
    String name;
    String date;
    String message;
    examData(String name,
              String date,
              String message)
    {
        this.name = name;
        this.date = date;
        this.message = message;
    }
}

public class ClickListiner{
    // here index is index
    // of item clicked W
    public click(int index);
}

To click on item of recycler view pass the instance of click
interface in constructor of adapter

```

RecyclerView Implementation in an activity:

```

package com.example.admin.example;
import android.content.Context;
import android.content.Intent;
import android.support.v7.app.ActionBarDrawerToggle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.support.v7.widget.Toolbar;
import android.view.LayoutInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import com.prolificinteractive
    .materialcalendarview

```

```

        .MaterialCalendarView;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
public class exam extends AppCompatActivity
    implements NavigationView
        .OnNavigationItemSelectedListener {
    ImageGalleryAdapter2 adapter;
    RecyclerView recyclerView;
    ClickListiner listiner;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_exam);
        Toolbar toolbar
= (Toolbar)findViewById(R.id.toolbar);
        toolbar.setTitle("");
        setSupportActionBar(toolbar);
        List<examData> list = new ArrayList();
        list = getData();
        recyclerView
= (RecyclerView)findViewById(
        R.id.recyclerView);
        listiner = new ClickListiner() {
    @Override
        public void click(int index){
            Toast.makeText(this,"clicked item index is
            "+index,Toast.LENGTH_LONG).show();
        }
    };
        adapter
= new ImageGalleryAdapter2(
        list, getApplication(),listiner);
        recyclerView.setAdapter(adapter);
        recyclerView.setLayoutManager(
        new LinearLayoutManager(exam.this));
    }
    @Override
    public void onBackPressed()
    {
        super.onBackPressed();
    }
}

```



```

    }
    // Sample data for RecyclerView
    private List<examData> getData()
    {
        List<examData> list = new ArrayList();
        list.add(new examData("First Exam",
                               "May 23, 2015",
                               "Best Of Luck"));
        list.add(new examData("Second Exam",
                               "June 09, 2015",
                               "b of l"));
        list.add(new examData("My Test Exam",
                               "April 27, 2017",
                               "This is testing exam ..
                               "});
    }
};

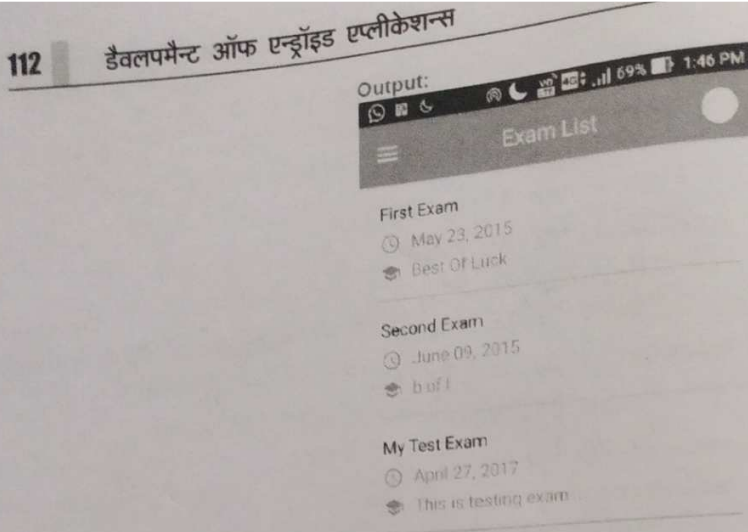
```

activity_exam.xml

```

<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="56dp"
    android:background="#FFFFFF"
    tools:context=".exam" >
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical" >
        <android.support.v7.widget.RecyclerView
            android:nestedScrollingEnabled="false"
            android:id="@+id/recyclerView"
            android:layout_width="match_parent"
            android:overScrollMode="never"
            android:layout_height="wrap_content"/>
        </LinearLayout>
    </ScrollView>

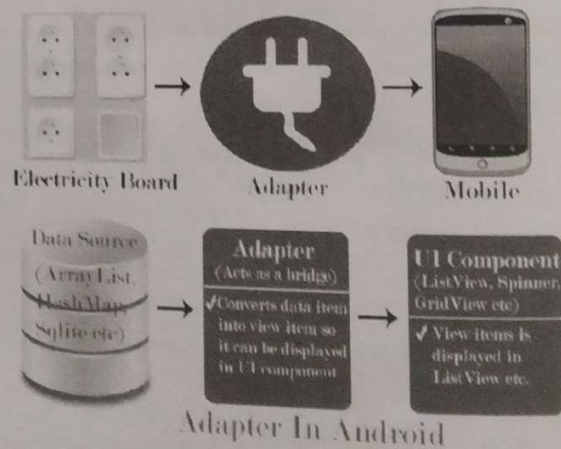
```



चित्र 5.6

Adapter

ANDROID में, ADAPTER यूआई घटक और डेटा स्रोत के बीच एक पुल है जो हमें यूआई घटक में डेटा भरने में मदद करता है। यह डेटा रखता है और डेटा को ADAPTER व्यू में भेजता है, फिर व्यू ADAPTER व्यू से डेटा ले सकता है और LISTVIEW, GRIDVIEW, स्पinner आदि जैसे विभिन्न दृश्यों पर डेटा दिखाता है। विचारों में ऑफ अनुकूलन के लिए हम आधार ADAPTER या कस्टम ADAPTER का उपयोग करते हैं।



चित्र 5.7

किसी सूची या ग्रिड में डेटा भरने के लिए हमें ADAPTER को लागू करने की आवश्यकता है। ADAPTER यूआई घटक और डेटा स्रोत के बीच एक पुल की तरह कार्य करता है। यहाँ डेटा स्रोत वह स्रोत है जहाँ हमें डेटा मिलता है और यूआई घटक सूची या ग्रिड आइटम हैं जिनमें हम उस डेटा को प्रदर्शित करना चाहते हैं।

Android में Adapter

यूआई घटकों में डेटा भरने के लिए उपयोग किए जाने वाले ADAPTER में कुछ आमतौर पर उपयोग किए जाने वाले ADAPTER हैं।

बेस ADAPTER—यह अन्य सभी ADAPTER के लिए Parent ADAPTER है।

एरे ADAPTER—जब भी हमारे पास एकल वस्तुओं की सूची होती है तो इसका उपयोग किया जाता है जो एक सारणी द्वारा समर्थित है।

कस्टम एरे ADAPTER—जब भी हमें कस्टम सूची प्रदर्शित करने की आवश्यकता होती है तो इसका उपयोग किया जाता है।

सिंपल ADAPTER—आपकी एक्सएमएल फाइल में परिभाषित दृश्यों के लिए स्थिर डेटा को मैप करना एक आसान ADAPTER है।

कस्टम सिंपल ADAPTER—जब भी हमें एक अनुकूलित सूची प्रदर्शित करने की आवश्यकता होती है और सूची या ग्रिड के बच्चे आइटम तक पहुँचने की आवश्यकता होती है तो इसका उपयोग किया जाता है।

अब हम एक-एक करके विस्तार से प्रत्येक ADAPTER का वर्णन करते हैं—

1. बेसएडेप्टर—बेस ADAPTER एक ADAPTER के सामान्य कार्यान्वयन का एक सामान्य आधार CLASS है जिसका उपयोग LISTVIEW, GRIDVIEW, स्पिनर आदि में किया जा सकता है। जब भी हमें GRIDVIEW में LISTVIEW या कस्टमाइज्ड ग्रिड में कस्टमाइज्ड लिस्ट की जरूरत होती है तो हम अपना ADAPTER बनाते हैं और उसमें बेस ADAPTER का विस्तार करते हैं। कस्टम सूची आइटम प्रदर्शित करने के लिए कस्टम ADAPTER बनाने के लिए बेस ADAPTER बढ़ाया जा सकता है। एरे Adapter भी बेसएडेप्टर का कार्यान्वयन है।

कस्टम ADAPTER कोड जो उसमें बेस ADAPTER का विस्तार करता है—

```
public class CustomAdapter extends BaseAdapter {
    @Override
    public int getCount() {
        return 0;
    }
    @Override
    public Object getItem(int i) {
        return null;
    }
    @Override
    public long getItemId(int i) {
        return 0;
    }
    @Override
    public View getView(int i, View view, ViewGroup viewGroup) {
        return null;
    }
}
```

उपरोक्त कोड स्निपेट में हम बेसएडेप्टर के ओवरराइड फंक्शन देखते हैं जिनका उपयोग डेटा को सूची, ग्रिड या स्पिनर में सेट करने के लिए किया जाता है।

2. एरेएडेप्टर—जब भी हमारे पास एकल वस्तुओं की सूची होती है जो किसी सारणी द्वारा समर्थित होती है, तो हम एरेएडेप्टर का उपयोग कर सकते हैं। उदाहरण के लिए, फोन संपर्कों, देशों या नामों की सूची।

यहाँ बताया गया है कि ADAPTER एरेअडेप्टर कैसा दिखता है—

```
ArrayAdapter(Context context, int resource, int textViewResourceId,
T[] objects)
```

3. **कस्टम एरे Adapter**—एरे ADAPTER भी बेसएडेप्टर का कार्यान्वयन है, इसलिए यदि हम अधिक अनुकूलन चाहते हैं तो हम एक कस्टम ADAPTER बना सकते हैं और उसमें एरेअडेप्टर का विस्तार कर सकते हैं। चूँकि सारणी ADAPTER बेस ADAPTER का कार्यान्वयन है, इसलिए हम अपने कस्टम ADAPTER में बेसएडेप्टर के सभी फंक्शन को ओवरराइड कर सकते हैं।

कस्टम ADAPTER क्लास MyAdapter के नीचे उस में ArrayAdapter फैली हुई है—

```
public class MyAdapter extends ArrayAdapter {
    public MyAdapter(Context context, int resource, int
    textViewResourceId, List objects) {
        super(context, resource, textViewResourceId, objects);
    }
    @Override
    public int getCount() {
        return super.getCount();
    }
    @Override
    public View getView(int position, View convertView, ViewGroup
    parent) {
        return super.getView(position, convertView, parent);
    }
}
```

4. **सिंपल ADAPTER**—ADAPTER सिंपल एडेप्टर में एक्सएमएल फाइल (लेआउट) में परिभाषित दृश्यों के लिए स्थिर डेटा को मैप करने के लिए एक आसान ADAPTER है। ADAPTER में हम डेटा बैकिंग को मैप्स की एक सारणी सूची (यानी हशमैप या अन्य) के रूप में एक सूची में शामिल कर सकते हैं। एक सारणी सूची में प्रत्येक प्रविष्टि एक सूची की एक पंक्ति के अनुरूप है।

मानचित्र में प्रत्येक पंक्ति के लिए डेटा होता है। यहाँ हम एक XML फाइल (कस्टम सूची आइटम फाइल) भी निर्दिष्ट करते हैं जो पंक्ति को प्रदर्शित करने के लिए उपयोग किए जाने वाले दृश्यों को परिभाषित करता है और मानचित्र में चाबियों से विशिष्ट दृश्यों तक मैपिंग करता है।

जब भी हमें कस्टम सूची बनानी होती है तो हमें कस्टम ADAPTER को लागू करने की आवश्यकता होती है। जैसा कि हम पहले चर्चा करते हैं, जब हमारे पास एक सारणी द्वारा समर्थित एकल आइटम की सूची होती है, तो पहले एरेअडेप्टर का उपयोग किया जाता है। इसलिए यदि हमें LISTVIEW या GRIDVIEW में अधिक अनुकूलन की आवश्यकता है तो हमें सरल ADAPTER को लागू करने की आवश्यकता है।

सरल ADAPTER कोड—एक सरल ADAPTER के उपरोक्त मापदंडों को उदाहरण के साथ सिंपल ADAPTER ट्यूटोरियल में वर्णित किया गया है।

```
SimpleAdapter (Context context, List<? extends Map<String, ?>> data,
int resource, String[] from, int[] to)
```

5. **ADAPTER में कस्टम सिंपल ADAPTER**—जब भी हमें कस्टम सूची बनानी होती है तो हमें कस्टम ADAPTER को लागू करने की आवश्यकता होती है। जैसा कि हम पहले चर्चा करते हैं, जब हमारे पास एक सारणी द्वारा समर्थित एकल आइटम की सूची होती है, तो पहले एरेअडेप्टर का उपयोग किया जाता है। इसलिए यदि हमें

LISTVIEW या GRIDVIEW में अनुकूलन की आवश्यकता है तो हमें सरल ADAPTER को लागू करने की आवश्यकता है, लेकिन जब हमें सूची या ग्रिड आइटम में अधिक अनुकूलन की आवश्यकता होती है जहाँ हमारे पास एक सूची आइटम में कई दृश्य होते हैं और फिर हमें किसी विशेष दृश्य के लिए क्लिक या किसी अन्य घटना जैसी किसी भी घटना को अंजाम देना होता है तो हमें एक कस्टम ADAPTER को लागू करने की आवश्यकता होती है जो हमारी आवश्यकता को पूरा करता है और लागू करने में काफी आसान है।

बेस ADAPTER अन्य सभी ADAPTER के लिए पैरेंट ADAPTER है, इसलिए यदि हम एक सिंपल ADAPTER का विस्तार करते हैं तो हम उस CLASS में बेस ADAPTER के कार्य को भी ओवरराइड कर सकते हैं।

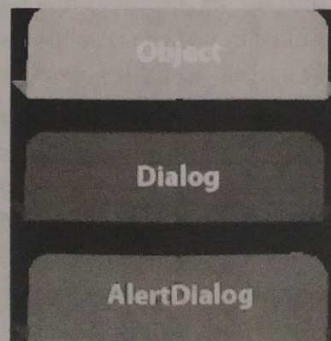
महत्वपूर्ण नोट—हम किसी सूची या ग्रिड के बाल आइटम पर क्लिक और अन्य ईवेंट जैसी घटनाओं को अंजाम नहीं दे सकते हैं, लेकिन यदि हमारे पास ऐसा करने के लिए कुछ आवश्यकताएँ हैं तो हम अपना कस्टम ADAPTER बना सकते हैं और उसमें सरल ADAPTER का विस्तार कर सकते हैं।

कस्टम ADAPTER उसमें सिंपलएडेप्टर का विस्तार करता है—

```
public class CustomAdapter extends SimpleAdapter {
    public CustomAdapter(Context context, List<? extends Map<String, ?>>
        data, int resource, String[] from, int[] to)
    {
        super(context, data, resource, from, to);
    }
    @Override
    public View getView(int position, View convertView, ViewGroup
        parent) {
        return super.getView(position, convertView, parent);
    }
    @Override
    public int getCount() {
        return super.getCount();
    }
}
```

5.6. Android AlertDialog

ANDROID अलर्टडायलॉग का उपयोग ओके और कैंसिल बटन के साथ संवाद संदेश प्रदर्शित करने के लिए किया जा सकता है। इसका उपयोग उपयोगकर्ता को अपनी पसंद के बारे में बाधित करने और उसकी पसंद के बारे में पूछने के लिए किया जा सकता है।



चित्र 5.8

ANDROID अलर्टडायलॉग तीन क्षेत्रों से बना है—शीर्षक, MATERIAL क्षेत्र और एक्शन बटन।
एंड्रॉइड AlertDialog संवाद CLASS के उप CLASS हैं।

Methods of AlertDialog class

Method	Description
public AlertDialog.Builder setTitle (CharSequence)	AlertDialog का शीर्षक निर्धारित करने के लिए किया जाता है।
public AlertDialog.Builder setMessage(CharSequence)	AlertDialog के लिए संदेश सेट करने के लिए किया जाता है।
public AlertDialog.Builder setIcon (int)	AlertDialog पर आइकन सेट करने के लिए किया जाता है।

Android अलर्टडायलॉग

उदाहरण—आइए ANDROID अलर्ट संवाद का एक सरल उदाहरण देखें।

activity_main.xml

आपके पास कई घटक हो सकते हैं, यहाँ हमारे पास केवल एक टेक्स्टव्यू है।

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context="example.javatpoint.com.alertdialog.MainActivity">
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/button"
    android:text="Close app"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

strings.xml

Optionally, you can store the dialog message and title in the strings.xml file.

File: strings.xml

```
<resources>
```

```

    <string name="app_name">AlertDialog</string>
    <string name="dialog_message">Welcome to Alert Dialog</string>
    <string name="dialog_title">Javatpoint Alert Dialog</string>
</resources>

```

Activity class

Let's write the code to create and show the AlertDialog.

File: MainActivity.java

```

package example.javatpoint.com.alertdialog;
import android.content.DialogInterface;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.app.AlertDialog;
import android.widget.Toast;
public class MainActivity extends AppCompatActivity {
    Button closeButton;
    AlertDialog.Builder builder;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        closeButton = (Button) findViewById(R.id.button);
        builder = new AlertDialog.Builder(this);
        closeButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Uncomment the below code to Set the message and title from
                the strings.xml file
                builder.setMessage(R.string.dialog_message)
                .setTitle(R.string.dialog_title);
                //Setting message manually and performing action on button
                click
                builder.setMessage("Do you want to close this application ?")
                setCancelable(false)
                setPositiveButton("Yes", new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        finish();
                        Toast.makeText(getApplicationContext(), "you choose yes action
                        for alertbox",

```

```

        Toast.LENGTH_SHORT).show();
    }

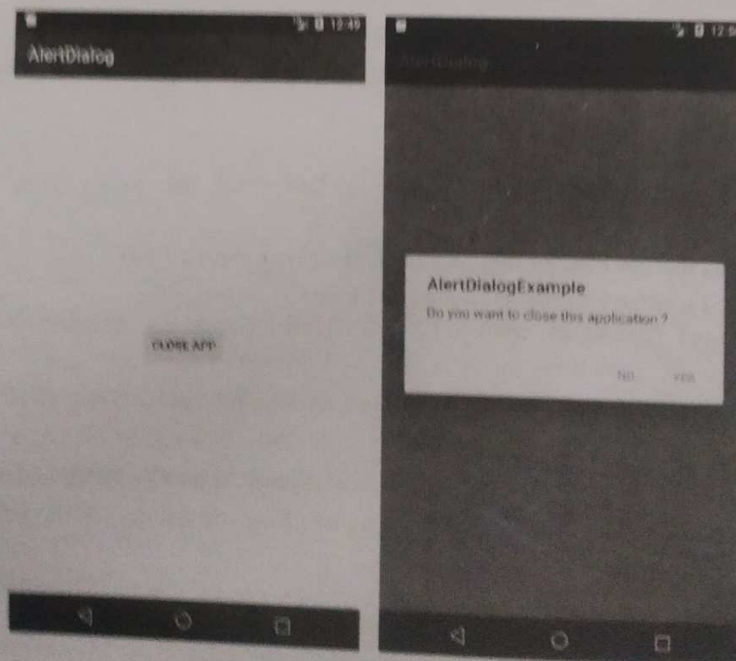
    })
    .setNegativeButton("No", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            // Action for 'NO' Button
            dialog.cancel();
            Toast.makeText(getApplicationContext(), "you choose no action for",
                alertbox",
                Toast.LENGTH_SHORT).show();
        }
    });

    //Creating dialog box
    AlertDialog alert = builder.create();
    //Setting the title manually
    alert.setTitle("AlertDialogExample");
    alert.show();
}
}):
}
}

```

Output

android alert dialog example output 1 android alert dialog example output



चित्र 5.9

Android Toast

ANDROID TOAST का उपयोग कम समय के लिए जानकारी प्रदर्शित करने के लिए किया जा सकता है। एक TOAST संदेश होता है जो जल्दी से प्रदर्शित किया जाना है और कुछ समय के बाद गायब हो जाता है।

`android.widget.Toast` CLASS `java.lang.Object` class का subCLASS है।

आप कस्टम TOAST भी बना सकते हैं और उदाहरण के लिए TOAST छवि प्रदर्शित कर सकते हैं। कस्टम TOAST के लिए कोड देखने के लिए आप अगले पृष्ठ पर जा सकते हैं।

Toast Class

TOAST CLASS का उपयोग एक विशेष अंतराल के लिए अधिसूचना दिखाने के लिए किया जाता है। कुछ समय बाद यह गायब हो जाता है। यह उपयोगकर्ता इंटरैक्शन को ब्लॉक नहीं करता है।

Toast Class के Constants

Constant	Description
<code>public static final int LENGTH_LONG</code>	की लम्बी अवधि के लिए दृश्य प्रदर्शित करता है।
<code>public static final int LENGTH_SHORT</code>	समय की छोटी अवधि के लिए दृश्य प्रदर्शित करता है।

Methods of Toast Class

Method	Description
<code>public static Toast makeText (Context context, CharSequence text, int duration)</code>	TOAST को पाठ और अवधि युक्त बनाता है।
<code>public void show()</code>	TOAST प्रदर्शित करता है।
<code>public void setMargin (float horizontalMargin, float verticalMargin)</code>	क्षैतिज और ऊर्ध्वाधर मार्जिन अंतर को बदलता है।

5.7. Android Toast Example

```
Toast.makeText(getApplicationContext(), "Hello
Javatpoint", Toast.LENGTH_SHORT).show();
```

Another code:

```
Toast toast=Toast.makeText(getApplicationContext(), "Hello
Javatpoint", Toast.LENGTH_SHORT);
toast.setMargin(50, 50);
toast.show();
```

Here, `getApplicationContext()` method returns the instance of Context.

Full code of activity class displaying Toast

Let's see the code to display the toast.

File: MainActivity.java

```

package example.javatpoint.com.toast;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;

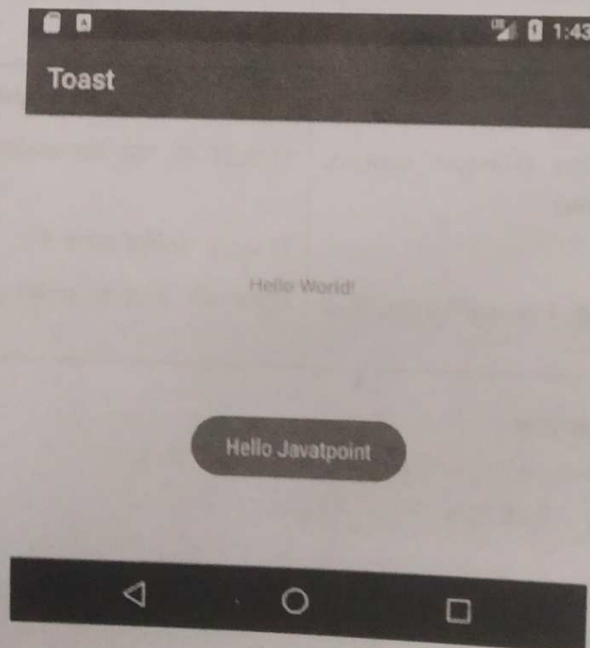
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //Displaying Toast with Hello Javatpoint message
        Toast.makeText(getApplicationContext(), "Hello
        Javatpoint", Toast.LENGTH_SHORT).show();
    }
}

```

Output:



चित्र 5.10

5.6. Android Popup Menu

एंड्रॉयड पॉपअप मेनू एंकर पाठ के नीचे मेनू प्रदर्शित करता है अगर अंतरिक्ष अन्यथा लंगर पाठ के ऊपर उपलब्ध है। अगर आप पॉपअप मेन्यू के बाहर क्लिक करते हैं तो यह गायब हो जाता है।

android.widget.PopupMenu java.lang.Object class का प्रत्यक्ष SUBCLASS है।

Android पॉपअप मेनू उदाहरण

आइए देखें कि ANDROID में पॉपअप मेनू कैसे बनाया जाए—

activity_main.xml

It contains only one button.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.javatpoint.com.popupmenu.MainActivity">

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/button"
    android:text="Click"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

popup_menu.xml

It contains three items as show below. It is created inside the res/menu directory.

File: poupup_menu.xml

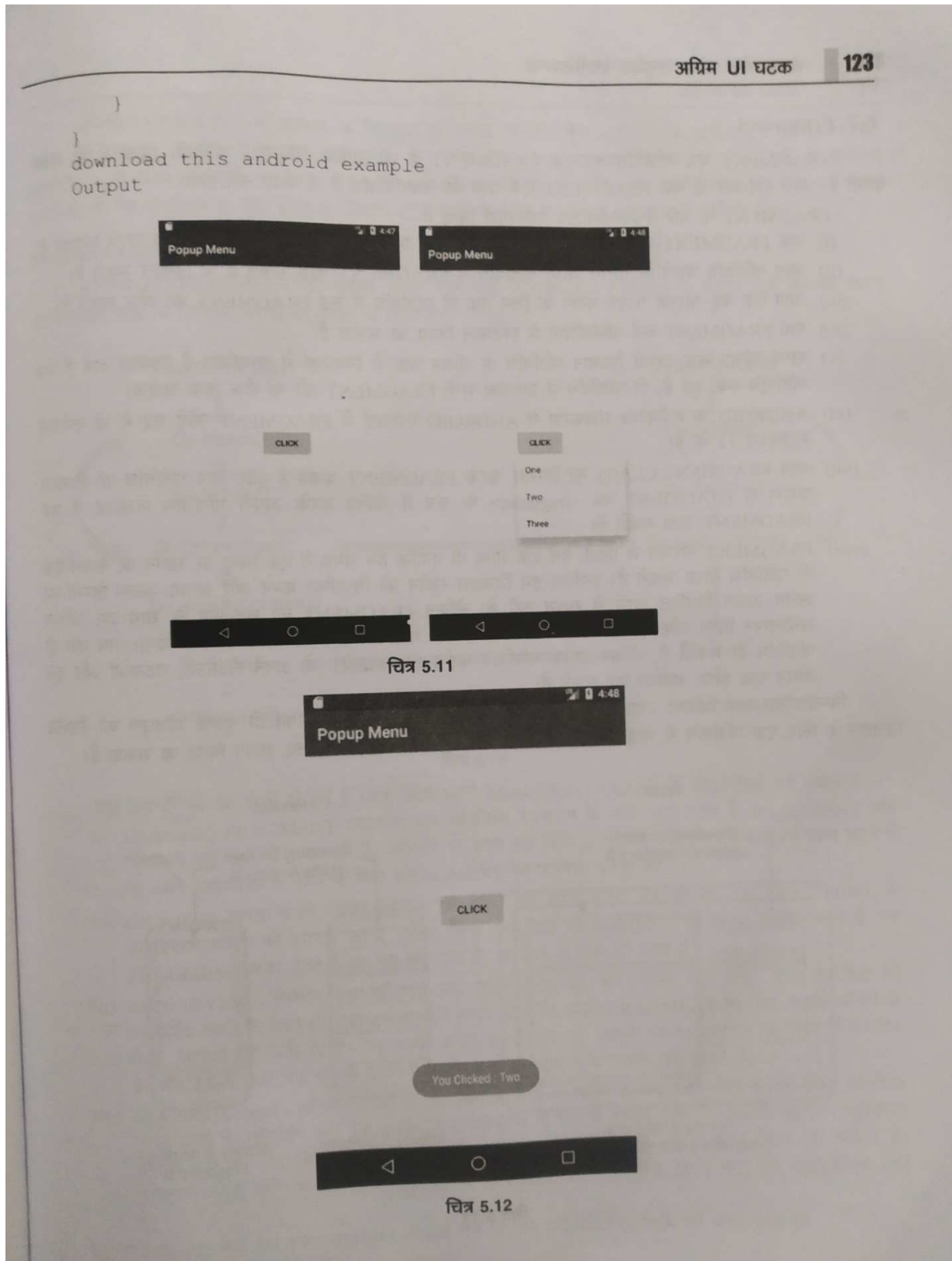
```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
<item
    android:id="@+id/one"
    android:title="One" />
<item
    android:id="@+id/two"
    android:title="Two"/>
<item
    android:id="@+id/three"
    android:title="Three"/>
```

```

</menu>

Activity class
It displays the popup menu on button click.
File: MainActivity.java
package example.javatpoint.com.popupmenu;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.PopupMenu;
import android.widget.Toast;
public class MainActivity extends AppCompatActivity {
    Button button;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        button = (Button) findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Creating the instance of PopupMenu
                PopupMenu popup = new PopupMenu(MainActivity.this, button);
                //Inflating the Popup using xml file
                popup.getMenuInflater().inflate(R.menu.popup_menu,
                popup.getMenu());
                //registering popup with OnMenuItemClickListener
                popup.setOnMenuItemClickListener(new
                PopupMenu.OnMenuItemClickListener() {
                    public boolean onMenuItemClick(MenuItem item) {
                        Toast.makeText(MainActivity.this, "You Clicked : " +
                        item.getTitle(), Toast.LENGTH_SHORT).show();
                        return true;
                    }
                });
                popup.show();//showing popup menu
            }
        }); //closing the setOnClickListener method
    }
}

```

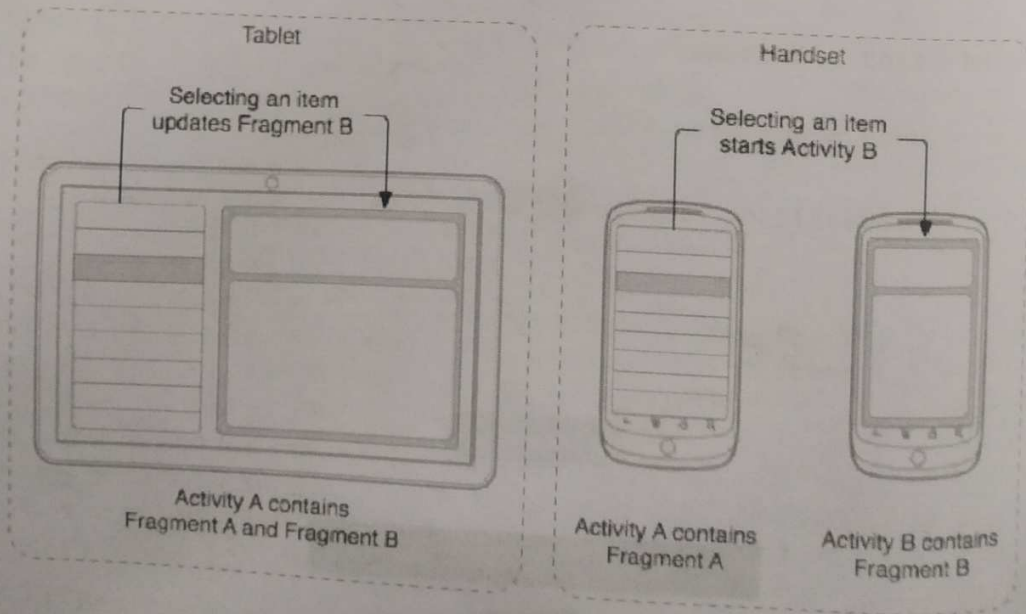
5.7. Fragment

FRAGMENT एक गतिविधि का एक FRAGMENT है जो अधिक मॉड्यूलर गतिविधि डिजाइन को करता है। अगर हम कहें तो एक FRAGMENT एक तरह की उपगतिविधि है तो गलत नहीं होगा।

FRAGMENT के बारे में निम्नलिखित महत्वपूर्ण बिन्दु हैं—

- (i) एक FRAGMENT अपने स्वयं के लेआउट और अपने स्वयं के जीवनचक्र कॉलबैक के साथ अपने व्यवहार को नियंत्रित करता है।
- (ii) आप गतिविधि चलने के दौरान किसी गतिविधि में FRAGMENT जोड़ सकते हैं या निकाल सकते हैं।
- (iii) आप एक बहु-फलक यूआई बनाने के लिए एक ही गतिविधि में कई FRAGMENT को जोड़ सकते हैं।
- (iv) एक FRAGMENT कई गतिविधियों में इस्तेमाल किया जा सकता है।
- (v) खण्ड जीवन चक्र अपनी मेजबान गतिविधि के जीवन चक्र से निकटता से सम्बन्धित है जिसका अर्थ है कि गतिविधि रुकी हुई है, तो गतिविधि में उपलब्ध सभी FRAGMENT को भी रोक दिया जाएगा।
- (vi) ANDROID के हनीकॉम्ब संस्करण में ANDROID एपीआई में FRAGMENT जोड़े गए थे जो एंड्रॉइड संस्करण 11 के हैं।
- (vii) आप FRAGMENT CLASS का विस्तार करके FRAGMENT बनाते हैं और आप गतिविधि की लैफ़ाईल में FRAGMENT को <fragment> के रूप में घोषित करके अपनी गतिविधि लेआउट में FRAGMENT डाल सकते हैं।
- (viii) FRAGMENT परिचय से पहले, हम एक सीमा थी क्योंकि हम समय में एक बिन्दु पर स्क्रीन पर केवल एक ही गतिविधि दिखा सकते हैं। इसलिए हम डिवाइस स्क्रीन को विभाजित करने और अलग-अलग हिस्सों को अलग-अलग नियंत्रित करने में सक्षम नहीं थे। लेकिन FRAGMENT की शुरुआत के साथ हमें अलग-अलग लचीलापन मिला और एक समय में स्क्रीन पर एक ही गतिविधि होने की सीमा को हटा दिया। अब एक गतिविधि हो सकती है, लेकिन प्रत्येक गतिविधि कई FRAGMENT जो अपने लेआउट, घटनाओं और जीवन चक्र होगा, शामिल कर सकते हैं।

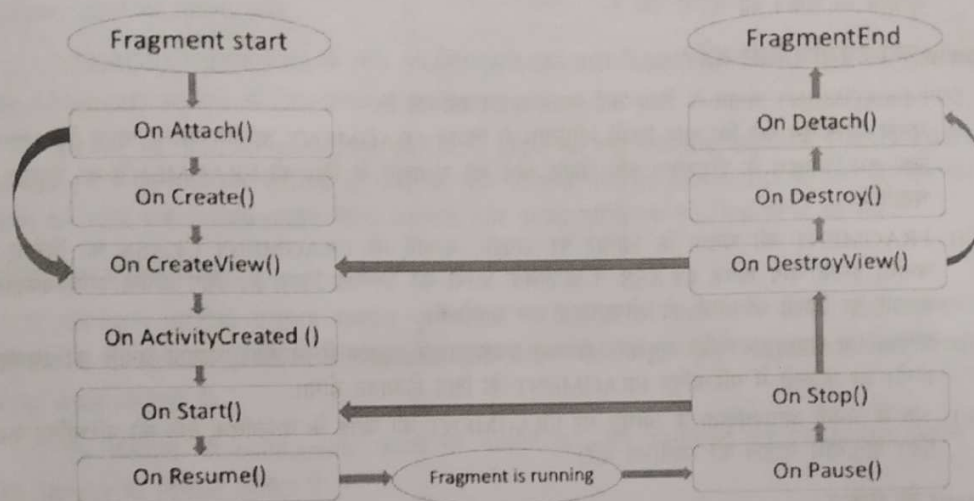
निम्नलिखित एक विशिष्ट उदाहरण है कि कैसे FRAGMENT द्वारा परिभाषित दो यूआई मॉड्यूल को एक डिजाइन के लिए एक गतिविधि में जोड़ा जा सकता है, लेकिन हैंडसेट डिजाइन के लिए अलग किया जा सकता है।



एप्लिकेशन एक टैबलेट के आकार के डिवाइस पर चलने पर गतिविधि A में दो FRAGMENT को एम्बेड कर सकता है। हालांकि, हैंडसेट के आकार की स्क्रीन पर, दोनों FRAGMENT के लिए पर्याप्त जगह नहीं है, इसलिए गतिविधि A में लेखों की सूची के लिए केवल FRAGMENT शामिल है, और जब उपयोगकर्ता एक लेख का चयन करता है, तो यह गतिविधि B शुरू होता है, जिसमें लेख पढ़ने के लिए दूसरा FRAGMENT शामिल है।

Fragment Life Cycle

ANDROID FRAGMENT का अपना जीवन चक्र एक ANDROID गतिविधि के समान होता है। यह खण्ड अपने जीवन चक्र के विभिन्न चरणों के संक्षेप में जाता है।



चित्र 5.14

यहाँ उन तरीकों की सूची दी गई है जिन्हें आप अपने FRAGMENT CLASS में ओवरराइड कर सकते हैं—

- OnAttach() FRAGMENT उदाहरण एक गतिविधि उदाहरण के साथ जुड़ा हुआ है। FRAGMENT और गतिविधि पूरी तरह से शुरू नहीं है। आमतौर पर आप इस विधि में गतिविधि के लिए एक संदर्भ प्राप्त करते हैं जो आगे आरम्भकरण कार्य के लिए FRAGMENT का उपयोग करता है।
- onCreate() खण्ड बनाते समय सिस्टम इस विधि को कॉल करता है। आपको उस FRAGMENT के आवश्यक घटकों को प्रारम्भ करना चाहिए जिन्हें आप उस FRAGMENT को बनाए रखना चाहते हैं जब FRAGMENT रुका हुआ है या बंद हो जाता है, या फिर से शुरू हो जाता है।
- onCreateView() सिस्टम इस कॉलबैक को कॉल करता है। जब पहली बार अपने यूजर इंटरफेस को आकर्षित करने के लिए FRAGMENT का समय होता है। अपने FRAGMENT के लिए यूआई खींचने के लिए, आपको इस विधि से एक व्यू घटक वापस करना होगा जो आपके FRAGMENT के लेआउट की जड़ है। यदि FRAGMENT यूआई प्रदान नहीं करता है तो आप शून्य वापस कर सकते हैं।
- onActivityCreated() ऑनएक्टिविटी () को ऑनक्रिएटव्यू () विधि के बाद कहा जाता है जब होस्ट गतिविधि बनाई जाती है। गतिविधि और FRAGMENT उदाहरण के रूप में अच्छी तरह से गतिविधि के दृश्य पदानुक्रम बनाया गया है। इस बिन्दु पर, व्यू को findViewById () विधि के साथ एक्सेस किया जा सकता है। उदाहरण—इस विधि में आप उन वस्तुओं को तत्काल कर सकते हैं जिन्हें संदर्भ वस्तु की आवश्यकता होती है।
- onStart() एक बार FRAGMENT दिखाई देने के बाद ऑनस्टार्ट () विधि को कहा जाता है।

- (vi) onResume() FRAGMENT सक्रिय हो जाता है।
- (vii) onPause() सिस्टम इस विधि को पहले संकेत के रूप में कहता है कि उपयोगकर्ता FRAGMENT छोड़ रहा है। यह आमतौर पर वह जगह है जहाँ आपको कोई भी परिवर्तन करना चाहिए जो वर्तमान उपयोगकर्ता सत्र से परे कायम होना चाहिए।
- (viii) onStop() FRAGMENT onStop() फोन करके बंद किया जा रहा है।
- (ix) onDestroyView() खण्ड दृश्य इस विधि को कॉल करने के बाद नष्ट हो जाएगा।
- (x) onDestroy() FRAGMENT राज्य के अंतिम साफ करने के लिए कहा जाता है, लेकिन एंड्रॉइड मंच द्वारा बुलाया जा सकने की गारंटी नहीं है।

Fragment का उपयोग कैसे करें?

इसमें FRAGMENT बनाने के लिए कई सरल कदम शामिल हैं—

- (i) सबसे पहले तय करें कि आप किसी गतिविधि में कितने FRAGMENT उपयोग करना चाहते हैं। उदाहरण के लिए हम डिवाइस के लैंडस्केप और पोर्ट्रेट मोड को संभालने के लिए दो FRAGMENT का उपयोग करना चाहते हैं।
- (ii) FRAGMENT की संख्या के आधार पर अगली, कक्षाएँ जो FRAGMENT CLASS का विस्तार होंगी, बनाएँ। इसके बाद खण्ड CLASS ने कॉलबैक कार्यों का उल्लेख किया है। आप अपनी आवश्यकताओं के आधार पर किसी भी कार्य को ओवरराइड कर सकते हैं।
- (iii) प्रत्येक FRAGMENT के अनुसार, आपको एक्समपल फाइल में लेआउट फाइल बनाने की आवश्यकता होगी। इन फाइलों में परिभाषित FRAGMENT के लिए लेआउट होगा।
- (iv) अंत में अपनी आवश्यकता के आधार पर FRAGMENT की जगह के वास्तविक तर्क को परिभाषित करने के लिए गतिविधि फाइल को संशोधित करें।

Fragment के प्रकार

मूल रूप से FRAGMENT को नीचे दिखाए गए तीन चरणों के रूप में विभाजित किया गया है—

- (i) सिंगल फ्रेम के FRAGMENT—सिंगल फ्रेम के FRAGMENT मोबाइल जैसे हैंड होल्ड डिवाइसेज के लिए इस्तेमाल कर रहे हैं, वहाँ हम एक व्यू के तौर पर सिर्फ एक FRAGMENT दिखा सकते हैं।
- (ii) सूची के FRAGMENT—विशेष सूची दृश्य वाले FRAGMENT को सूची FRAGMENT कहा जाता है।
- (iii) FRAGMENT लेनदेन—FRAGMENT लेनदेन के साथ उपयोग करना। हम एक FRAGMENT को दूसरे FRAGMENT में ले जा सकते हैं।

5.8. Android Material Components (MDC)

1. MATERIAL डिजाइन—बोल्ड और सुंदर डिजिटल उत्पादों के निर्माण के लिए एक प्रणाली है। सिद्धांतों और घटकों के एक सुसंगत सेट के तहत शैली, ब्रांडिंग, बातचीत और गति को एकजुट करके, उत्पाद टीम अपनी सबसे बड़ी डिजाइन क्षमता का एहसास कर सकती है।

ANDROID अनुप्रयोगों के लिए, ANDROID के लिए MATERIAL घटक (एमडीसी ANDROID) आपके Application में स्थिरता बनाने के लिए घटकों की लाइब्रेरी के साथ डिजाइन और इंजीनियरिंग को एकजुट करता है। जैसे-जैसे MATERIAL डिजाइन प्रणाली विकसित होती है, इन घटकों को Google के फ्रंट-एंड विकास मानकों के लगातार पिक्सेल-परफेक्ट कार्यान्वयन और पालन सुनिश्चित करने के लिए अपडेट किया जाता है। एमडीसी वेब, आईओएस और स्वयं के लिए भी उपलब्ध है।

2. अपने विकास के माहौल की स्थापना एंड्रॉयड स्टूडियो शुरू—जब आप ANDROID स्टूडियो खोलते हैं, तो इसे “वेल्कम टू ANDROID स्टूडियो” नामक एक विंडो प्रदर्शित करनी चाहिए। हालांकि, अगर यह पहली बार ANDROID स्टूडियो लॉन्च कर रहा है, तो डिफॉल्ट मूल्यों के साथ ANDROID स्टूडियो सेटअप जादूगर चरणों के माध्यम से जाएँ। यह कदम आवश्यक फाइलों को डाउनलोड करने और स्थापित करने में कई मिनट लग सकते हैं, इसलिए अगले अनुभाग करते समय पृष्ठभूमि में चलने वाले तथा इसे छोड़ने के लिए स्वतंत्र महसूस करें।

एंड्रॉयड स्टूडियो में स्टार्टर कोड

एक बार सेटअप खत्म हो जाता है और एंड्रॉयड स्टूडियो विंडो में आपका स्वागत है, एक मौजूदा एंड्रॉयड स्टूडियो परियोजना खोलो पर क्लिक करें।

ANDROID स्टूडियो विंडो के नीचे गतिविधि संकेतकों द्वारा दिखाए गए प्रोजेक्ट को बनाने और सिंक करने के लिए ANDROID स्टूडियो के लिए एक पल का इंतजार करें।

इस बिंदु पर, ANDROID स्टूडियो कुछ बिल्ड त्रुटियों को बढ़ा सकता है क्योंकि आप ANDROID एसडीके को याद कर रहे हैं या उपकरण बना रहे हैं, जैसे कि नीचे दिखाया गया है। ANDROID स्टूडियो में दिए गए निर्देशों का पालन करें ताकि इन्हें इंस्टॉल/अपडेट किया जा सके और अपनी परियोजना को सिंक किया जा सके।

परियोजना निर्भरता जोड़ें

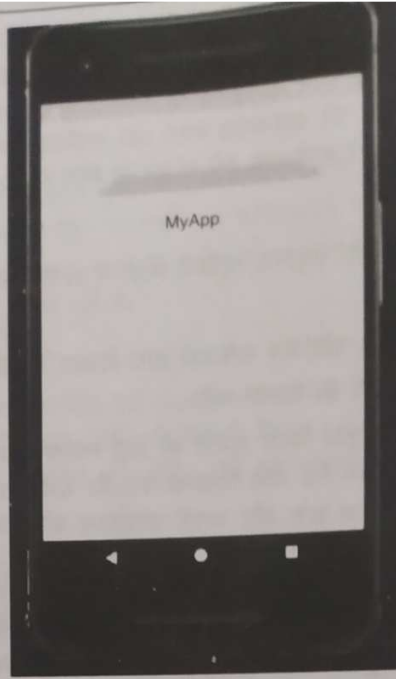
परियोजना एमडीसी एंड्रॉयड समर्थन पुस्तकालय पर निर्भरता की जरूरत है। आपके द्वारा डाउनलोड किए गए नमूना कोड में पहले से ही यह निर्भरता सूचीबद्ध होनी चाहिए, लेकिन यह सुनिश्चित करने के लिए निम्नलिखित चरण करना अच्छा अभ्यास है।

ऐप मॉड्यूल की build.gradle फाइल पर नेविगेट करें और सुनिश्चित करें कि निर्भरता ब्लॉक में एमडीसी ANDROID पर निर्भरता शामिल है—

```
api 'com.google.android.material:material:1.1.0-alpha06'
```

(वैकल्पिक) यदि आवश्यक हो, तो निम्नलिखित निर्भरताओं को जोड़ने और परियोजना को सिंक करने के लिए build.gradle फाइल को संपादित करें।

```
dependencies {
    api 'com.google.android.material:material:1.1.0-alpha06'
    implementation 'androidx.legacy:legacy-support-v4:1.0.0'
    implementation 'com.android.volley:volley:1.1.1'
    implementation 'com.google.code.gson:gson:2.8.5'
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:1.3.21"
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test:core:1.1.0'
    androidTestImplementation 'androidx.test.ext:junit:1.1.0'
    androidTestImplementation 'androidx.test:runner:1.2.0-alpha05'
    androidTestImplementation
    'androidx.test.espresso:espresso-core:3.2.0-alpha05'
```

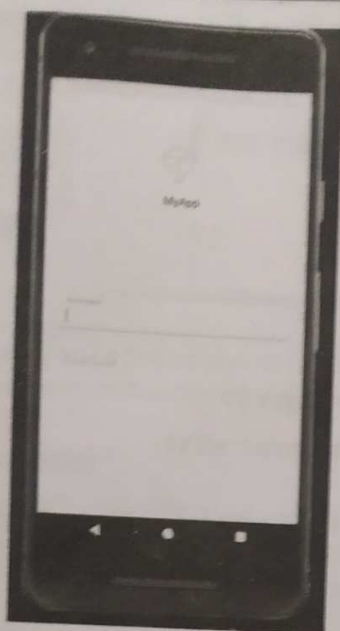


चित्र 5.15

Run the starter app

- (i) सुनिश्चित करें कि रन/प्ले बटन के बाईं ओर बिल्ड कॉन्फिगरेशन ऐप है।
- (ii) ऐप बनाने और चलाने के लिए ग्रीन रन/प्ले बटन दबाएँ।
- (iii) सेलेक्ट तैनाती टारगेट विंडो में, यदि आपके पास पहले से ही कोई ANDROID डिवाइस है जो आपके उपलब्ध उपकरणों में सूचीबद्ध है, तो चरण 8 पर छोड़ दें। अन्यथा, नए वर्चुअल डिवाइस बनाएँ पर क्लिक करें।
- (iv) चुनिंदा हार्डवेयर स्क्रीन में पिक्सल 2 जैसे फोन डिवाइस का चयन करें और फिर नेक्स्ट पर क्लिक करें।
- (v) अधिमानतः उच्चतम एपीआई स्तर सिस्टम इमेज स्क्रीन में, हाल ही में ANDROID संस्करण का चयन करें। यदि यह इंस्टॉल नहीं है, तो दिखाए गए डाउनलोड लिंक पर क्लिक करें और डाउनलोड को पूरा करें।
- (vi) आगे क्लिक करें।
- (vii) ANDROID वर्चुअल डिवाइस (AVD) स्क्रीन पर, सेटिंग्स को छोड़ दें क्योंकि वे हैं और फिनिश पर क्लिक करें।
- (viii) तैनाती लक्ष्य संवाद से एक एंड्रॉयड डिवाइस का चयन करें।
- (ix) ठीक क्लिक करें।
- (x) ANDROID स्टूडियो ऐप बनाता है इसे तैनात करता है, और स्वचालित रूप से इसे लक्ष्य डिवाइस पर खोलता है।

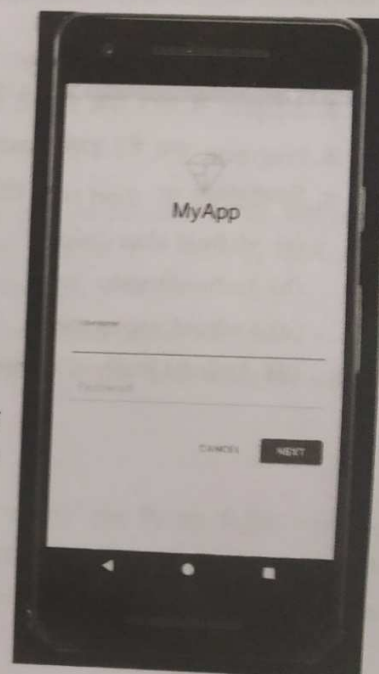
3. Add text fields—यह शुरू करने के लिए, हम लोगो को अपना उपयोगकर्ता नाम और पासवर्ड दर्ज करने के लिए अपने लॉगिन पेज पर दो टेक्स्ट फील्ड जोड़ेंगे हम एमडीसी टेक्स्ट फील्ड घटक का उपयोग करेंगे, जिसमें अंतर्निहित कार्यक्षमता शामिल है जो फ्लोटिंग लेबल और त्रुटि संदेश प्रदर्शित करती है।



चित्र 5.16

4. Add buttons—इसके बाद, हम अपने लॉगिन पेज पर दो बटन जोड़ेंगे cancel करें। हम एमडीसी बटन घटक का उपयोग करेंगे, जो प्रतिष्ठित सामग्री डिजाइन स्याही तरंग प्रभाव निर्मित के साथ आता है।

5. Navigate to the next Fragment—अंत में, हम अपने Next बटन को दूसरे टुकड़े पर हुक करने के लिए लॉगिन फ्रागमेंट java में कुछ जावा कोड जोड़ देंगे, आप देखेंगे कि हमारे लेआउट में जोड़े गए प्रत्येक घटक में एक आईडी थी। हम अपने कोड में घटकों को संदर्भित करने और कुछ त्रुटि जाँच और नेविगेशन जोड़ने के लिए इन आईडी का उपयोग करेंगे।



चित्र 5.17

आइए लॉगिन फ्रागमेंट में एक निजी बूलियन इस पासवर्ड वैलिड विधि जोड़ें `javaonCreateView()` के नीचे, यह निर्धारित करने के लिए तर्क के साथ कि पासवर्ड मान्य है या नहीं। इस डेमो के उद्देश्यों के लिए, हम बस यह सुनिश्चित करेंगे कि पासवर्ड कम-से-कम 8 वर्ण लम्बा हो—

```
private boolean is PasswordValid (@Nullable Editable text) {
    return text != null && text.length () >= 8;
}
```

6. All done—जावा की बुनियादी एक्सएमएल मार्कअप और ~ 30 लाइनों का उपयोग करके, ANDROID लाइब्रेरी के लिए सामग्री घटकों ने आपको एक सुंदर लॉगिन पेज बनाने में मदद की है जो सामग्री डिजाइन दिशा-निर्देशों के अनुरूप है, और सभी उपकरणों में लगातार दिखता है और व्यवहार करता है।

प्रश्नावली

1. Android में date and time को किस प्रकार प्रदर्शित किया जाता है?
2. निम्नलिखित पर short note दीजिए—
 - (a) List view
 - (b) Grid view
 - (c) Card view
 - (d) Recycler view
3. Adapter से आप क्या समझते हैं? यह कितने प्रकार के होते हैं?
4. Fragment क्या है? इसके प्रकार सहित lifecycle की व्याख्या कीजिए।
5. निम्नलिखित पर short note दीजिए—
 - (a) Android alert dialog
 - (b) Android toast
 - (c) Android popup menu
 - (d) Android material components



एन्ड्रॉइड में थ्रेड्स (Threads in Android)

Threads, running on UI thread (run on UI Thread), worker thread, handlers & runnable, AsyncTask, calling web services and consuming JSON data from web services.

6.1. थ्रेड्स (Threads)

एक thread अपने stack और local variables के साथ एक lightweight process है। यह एक individual process की तरह व्यवहार करता है लेकिन ऐसा नहीं है। जब भी आप कोई process शुरू करते हैं, तो Android OS concurrently रूप से काम करने के लिए Main के रूप में एक new thread शुरू करता है।

Thread execution की एक concurrent unit है (A thread is a concurrent unit of execution.)

एन्ड्रॉइड स्मार्टफोन मल्टीटास्किंग में अच्छा है। उदाहरण के लिए call, मैसेज और बैटरी मॉनिटरिंग आदि के लिए वह पहले से ही कई threads handle कर रहा है।

Thread क्यों महत्वपूर्ण है?

एक thread हमें एक ही समय में कई कार्य (things) करने की अनुमति देता है। यह विशेष रूप से game development, बैकग्राउंड services और multithreading processing में महत्वपूर्ण है क्योंकि अधिकांश स्मार्टफोन में multiple कोर प्रोसेसर होता है इसलिए इसका लाभ प्राप्त करें। यदि कोई thread नहीं है, तो आप शायद multiple कोर प्रौद्योगिकी को बर्बाद कर रहे हैं।

हम दो method से threads बना सकते हैं।

1. By extending the Thread class.

```
class Test extends Thread {
    @Override
    public void run() {
        Log.d("Threading", "Test class thread is . .
        >" + Thread.currentThread().getName());
    }
}
```

2. By implementing a Runnable interface.

```
class Test implements Runnable {
    @Override
    public void run() {
        Log.d("Test", "Test class thread is
        >" + Thread.currentThread().getName());
    }
}
```

8.2 थ्रैडिंग इन आंदोलन

थ्रैडिंग इन आंदोलन को किसी भी तरह की गतिशीलता कि या अन्य कारणों, जिसके कारणों को यह थ्रैडिंग इन आंदोलन कि या अन्य ही एंडर, या गतिशीलता के एक कारणों को परिभाषित करता है कि यह एक बात है।

8.2.1 थ्रैडिंग इन आंदोलन

थ्रैडिंग इन आंदोलन को दो तरह से वर्गीकृत कर सकते हैं—

1. थ्रैडिंग इन आंदोलन को दो प्रकारों में वर्गीकृत किया जा सकता है—

2. थ्रैडिंग इन आंदोलन को दो प्रकारों में वर्गीकृत किया जा सकता है—

8.2.1.1 थ्रैडिंग इन आंदोलन को दो प्रकारों में वर्गीकृत किया जा सकता है—

(i) थ्रैडिंग इन आंदोलन—थ्रैडिंग इन आंदोलन को दो तरह से वर्गीकृत किया जा सकता है—

(ii) थ्रैडिंग इन आंदोलन—थ्रैडिंग इन आंदोलन को दो तरह से वर्गीकृत किया जा सकता है—

1. थ्रैडिंग इन आंदोलन

2. थ्रैडिंग इन आंदोलन

8.2.1.2 थ्रैडिंग इन आंदोलन को दो प्रकारों में वर्गीकृत किया जा सकता है—

(i) थ्रैडिंग इन आंदोलन—थ्रैडिंग इन आंदोलन को दो तरह से वर्गीकृत किया जा सकता है—

(ii) थ्रैडिंग इन आंदोलन—थ्रैडिंग इन आंदोलन को दो तरह से वर्गीकृत किया जा सकता है—

8.2.2 थ्रैडिंग इन आंदोलन

थ्रैडिंग इन आंदोलन को दो तरह से वर्गीकृत किया जा सकता है—

1. थ्रैडिंग इन आंदोलन

2. थ्रैडिंग इन आंदोलन

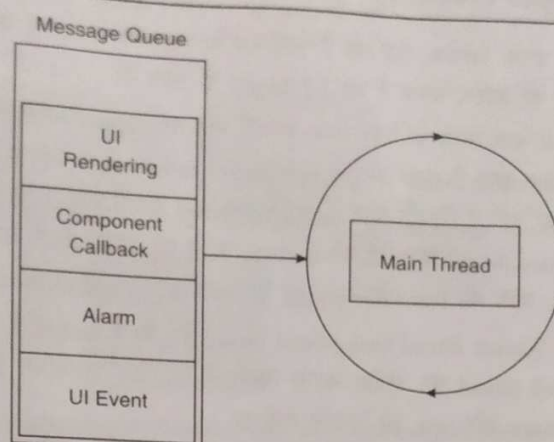
3. थ्रैडिंग इन आंदोलन

4. थ्रैडिंग इन आंदोलन

5. थ्रैडिंग इन आंदोलन

6. थ्रैडिंग इन आंदोलन

थ्रैडिंग इन आंदोलन को दो तरह से वर्गीकृत किया जा सकता है—

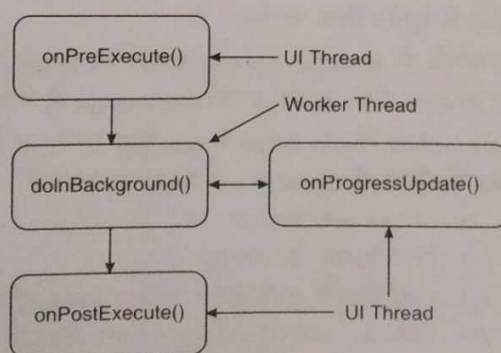


चित्र 1

अपने एप्लिकेशन को responsive बनाए रखने के लिए, किसी भी ऑपरेशन को करने के लिए main thread का उपयोग करने से बचना आवश्यक है जो इसे blocked रखते हुए समाप्त हो सकता है।

नेटवर्क संचालन और डेटाबेस call, साथ-ही-साथ कुछ components को लोड करना, ऑपरेशन के सामान्य उदाहरण हैं जिन्हें main thread से बचना चाहिए। जब उन्हें main thread में call किया जाता है, तो उन्हें synchronously कहा जाता है, जिसका अर्थ है कि ऑपरेशन पूरा होने तक UI पूरी तरह से unresponsive रहेगा। इस कारण से, वे आमतौर पर अलग-अलग thread में किए जाते हैं, जिससे UI को block करने से बचा जाता है जबकि वे perform कर रहे हैं (अर्थात्, उन्होंने UI से asynchronously perform किया जाता है)।

2. UI thread—जब एक एन्ड्रॉइड ऐप शुरू होता है, तो यह main thread बनाता है, जिसे अक्सर UI thread कहा जाता है।



चित्र 2

UI Thread आपके application के execution का main thread है। यह वह जगह है जहाँ आपके अधिकांश एप्लिकेशन कोड run किये जाते हैं। आपके सभी एप्लिकेशन कंपोनेंट्स activities, services, content providers, broadcast receivers इस thread में बनाए गए हैं, और उन कंपोनेंट्स को कोई भी सिस्टम call इस thread से की जाती है।

उदाहरण के लिए, मान ले कि आपका application, single activity class है। फिर सभी lifecycle methods और आपके अधिकांश event handling कोड इस UI Thread में चलते हैं। ये method जैसे onCreate, onPause,

onDestroy, onClick आदि हैं। इसके अलावा, यह वह है जहाँ UI के सभी अपडेट किए जाते हैं। कुछ भी जो UI को update करने या change करने का कारण बनता है वह UI thread पर होता है।

जब आप background में काम करने के लिए स्पष्ट रूप से एक नया thread बनाते हैं, तो यह कोड UI Thread पर run नहीं किया जाता है। तो क्या होता है अगर इस background thread को कुछ ऐसा करने की आवश्यकता होती है जो UI को change करता है। यह वही है जिसके लिए runOnUiThread है। वास्तव में आप एक handler का उपयोग करने वाले हैं। यह इन background thread के कोड को execute करने की क्षमता प्रदान करता है जो UI को modify कर सकते हैं। वे UI-modifying कोड को Runnable object में runOnUiThread method में pass करते हैं।

3. Worker thread : Worker thread background thread है। ये वे thread हैं जो UI thread के अलावा अलग से बनाए गए हैं। चूंकि UI thread को ब्लॉक करना नियम के अनुसार प्रतिबंधित है, इसलिए उपयोगकर्ता को worker thread में child processes और task को चलाना चाहिए।

Worker thread के निर्माण और काम करने का एक उदाहरण नीचे दिया गया है—

```
Public void onClick(View v) { new Thread(new Runnable() {
    public void run() {
        Bitmap b =loadImageFromNetwork("http://example.com/image.png");
        mImageView.setImageBitmap(b);
    }).start(); }
```

उपरोक्त उदाहरण कोड में डाउनलोड ऑपरेशन UI thread के अलावा दूसरे thread द्वारा नियंत्रित किया जाता है। लेकिन कार्यक्रम दूसरे नियम का उल्लंघन करता है। UI thread से imageView इस worker thread से manipulate कर रहा है।

दूसरे नियम के अनुसार, UI को UI thread के बाहर से एक्सेस नहीं किया जा सकता है। इस तरह के प्रतिबंध का समाधान runOnUiThread (Runnable) method है। RunOnUiThread (Runnable) method का उपयोग करके main या UI thread को अन्य thread से एक्सेस किया जा सकता है।

नतीजतन, इस method के माध्यम से specified runnable एक्शन UI thread पर run करेगी। यदि वर्तमान thread UI में ही है, तो एक्शन तुरंत execute होगी। अन्यथा एक्शन event queue में भेजी जाएगी।

4. Any thread : Annotated method को any thread से call किया जा सकता है। यदि एनोटेड element एक class है, तो class में सभी methods को किसी भी thread से call किया जा सकता है।

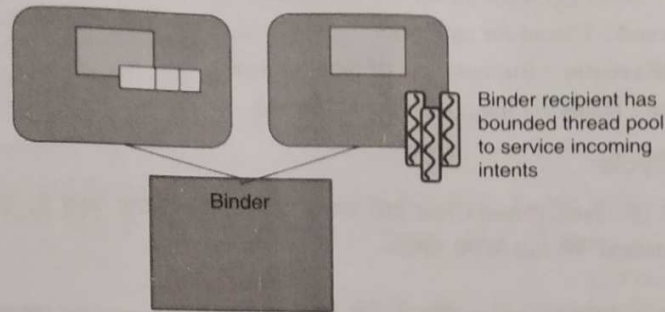
```
@Target([AnnotationTarget.FUNCTION,
    AnnotationTarget.PROPERTY_GETTER,
    AnnotationTarget.PROPERTY_SETTER, AnnotationTarget.CONSTRUCTOR,
    AnnotationTarget.CLASS, AnnotationTarget.FILE,
    AnnotationTarget.VALUE_PARAMETER]) class AnyThread
```

इस method का main उद्देश्य यह इंगित करना है कि आप मानते हैं कि method को any thread से call किया जा सकता है।

Example:

```
<code>
    @AnyThread
    Public void deliverResult(D data) { ... }
</code>
```

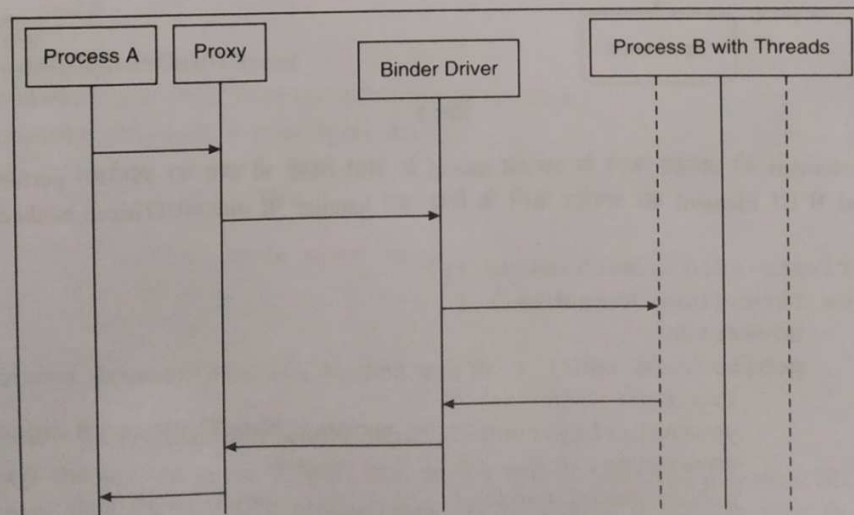

5. Binder thread : Binder thread आपकी service के एक separate thread का प्रतिनिधित्व करता है। Binder एक mechanism है जो इंटरप्रोसेस कम्युनिकेशन प्रदान करता है।



- The initiator thread effectively transfers to the recipient activity
- Max of 15 binder threads per process
- Max 1 MB of transaction buffer

चित्र 3

आइए एक उदाहरण पर विचार करें। कल्पना करें कि आपके पास service प्रक्रिया B (चित्र देखें) है और आपके पास कई एप्लिकेशन हैं जो इस service B के साथ communicate करते हैं (इस एप्लिकेशन में से एक, उदाहरण के लिए, प्रक्रिया A है।) इस प्रकार, एक service B को अलग-अलग एप्लिकेशन के लिए अलग-अलग परिणाम प्रदान करना चाहिए। इस प्रकार, आपको विभिन्न एप्लिकेशन के लिए service B की कई प्रतिकृतियाँ चलाने की आवश्यकता है। एन्ड्रॉइड इन प्रतिकृतियों को प्रोसेस B के विभिन्न thread में चलाता है और इन thread को “Binder thread # N” कहा जाता है।



चित्र 4-Binder communication

6.4. विभिन्न thread classes

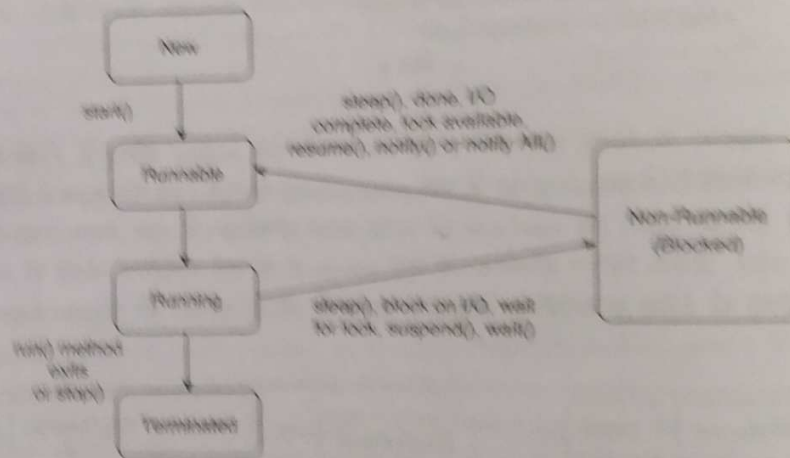
1. Async Task : Helps get work on other UI thread
2. Handler Thread : Thread for callbacks
3. ThreadPoolExecutor : Running lots of parallel work
4. IntentService : Helps get intents off the UI thread

6.5. Thread Life Cycle

प्रत्येक thread को java.lang.Thread Class द्वारा बनाया और नियंत्रित किया जाता है। Thread execution शुरू करने के लिए हमें start method को कॉल करना होगा—

```
new Test().start();
```

Thread lifecycle में विभिन्न states होता है जैसे कि new, runnable, running, non-runnable (blocked, terminated)।



चित्र 6

हम UI element को अपडेट करने के अलग thread के अंदर किसी भी तरह का ऑपरेशन perform कर सकते हैं। किसी thread से UI Element को अपडेट करने के लिए, हमें Handler या runnable thread method को इस्तेमाल करना होगा।

```

private void ...MethodName...() {
    new Thread(new Runnable() {
        @Override
        public void run() {
            try {
                yourActivity.runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        textView.setText("some value");
                        editText.setText("some new value");
                    }
                });
            } catch (Exception e) {
                // Handle exception
            }
        }
    }).start();
}
  
```

```

    });
    } catch (Exception e) {
        //print the error here
    }
}
}).start();
}

```

`runOnUiThread()` UI thread पर specified action Run करता है। यदि वर्तमान thread UI thread है, तो action तुरंत execute किया जाता है। यदि वर्तमान thread UI thread नहीं है, तो action, UI thread की event queue में post की जाती है।

6.6. Running on UI Thread

Android applications में एक main thread होता है, जिसे UI thread भी कहा जाता है। Application components जो same process में चलते हैं, जो डिफॉल्ट रूप से होता है, सभी same UI thread का उपयोग करते हैं। उन सभी life cycle methods, जिनके बारे में हम बात कर रहे हैं, `OnCreate`, `OnStart`, आदि, UI thread में संभाले जाते हैं।

एन्ड्रॉइड, UI thread पर चलने की guaranteed वाले कई method प्रदान करता है। उन method में से दो निम्नलिखित हैं—

- The `View.post` method

```

boolean View.post (Runnable action)
{
    handler.post(new Runnable() {
        public void run() {
            // UI code goes here
        }
    });
}

```

- The `Activity.runOnUiThread`

```

o Activity.runOnUiThread (Runnable action)
runOnUiThread ( new Runnable()
{
    public void run()
    {
        // UI code goes here
    }
});

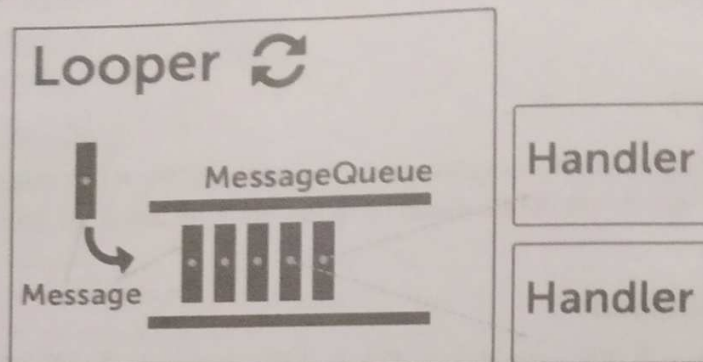
```

दोनों method `Runnable` parameter का प्रयोग करते हैं।

6.7. Message, MessageQueue, Looper

Message Queue : एक queue है जिसमें कार्यों की एक सूची है। Messages, runnables जिसे एक निश्चित thread में execute किया जाएगा। एन्ड्रॉइड MessageQueue को main thread में maintain करता है। Looper और handler के साथ, messageQueues एन्ड्रॉइड में threading के building blocks का हिस्सा है और सिस्टम में हर जगह उनका प्रयोग किया जाता है। यह class looper द्वारा भेजे जाने वाले messages की सूची रखता है। आप केवल

messages की सूची प्राप्त करने के लिए `Looper.myQueue ()` को call कर सकते हैं। Messages सीधे एक `MessageQueue` में नहीं, बल्कि `Looper` से जुड़े `Handler` ऑब्जेक्ट्स के माध्यम से जोड़े जाते हैं। हम `Looper.myQueue ()` के साथ `current thread` के लिए `MessageQueue` पुनर्प्राप्त कर सकते हैं।



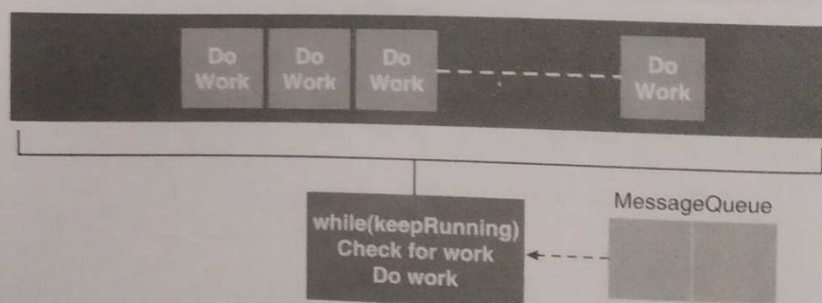
चित्र 7

Looper : Thread को alive रखने के लिए जिम्मेदार है। यह एक प्रकार का worker है जो वर्तमान thread के लिए एक `MessageQueue` की service देता है। `Looper` एक messages queue के माध्यम से loops करता है और प्रोसेस करने के लिए संबंधित threads को messages भेजता है।

एक thread में केवल एक unique `Looper` होगा। इसका मतलब है कि केवल एक `MessageQueue` प्रति thread। किसी एक thread में `Handler` की संख्या कितनी भी हो सकती है। तो, `Looper` एक loop में run करने के लिए सुविधा के साथ अपने स्वयं के `MessageQueue` से thread प्रदान करता है।

Looper

“Event Loop”



चित्र 8

Messages : एक messages को परिभाषित करता है जिसमें एक description और arbitrary डेटा objects होता है जिसे `Handler` को भेजा जा सकता है। यह object दो अतिरिक्त `int` फील्ड और objects फील्ड contain करता है जो आपको कई cases में allocation नहीं करने की अनुमति देता है। हम बस यह कह सकते हैं कि messages एक bundle की तरह है जो डेटा के transfer के लिए उपयोग किया जाता है। जबकि messages का constructor public है।

इन्में से किसी एक को प्राप्त करने का सबसे अच्छा तरीका `Message.obtain()` या `handler.obtainMessage()` methods में से एक को Call करना है, जो उन्हें recycled objects के पूल से खींच लेगा।

6.8. Handlers & Runnable

दो thread के बीच communication करने के लिए एक **Handler** का प्रयोग किया जाता है।

जैसा कि केवल एक thread है जो UI को अपडेट करता है, जो कि main thread है, हम background में multiple tasks को करने के लिए विभिन्न अन्य threads का उपयोग करते हैं लेकिन अंत में, UI को अपडेट करने के लिए हमें result को main या UI thread में post करना होता है। यदि आप threads के एक बड़े समूह का manage कर रहे हैं, तो इन सभी threads के साथ communication को manage करना कठिन होगा। इसलिए, एन्ड्रॉइड ने handlers को inter-process communication को आसान बनाने के लिए प्रदान किया है।

Handlers आपको एक thread के MessageQueue से जुड़े process Message और Runnable objects को भेजने की अनुमति देता है। प्रत्येक handlers instance एक thread और उस thread की message queue से जुड़ा होता है।

Handlers एक component है जिसे एक thread से जोड़ा जा सकता है और फिर simple messages या Runnable tasks के माध्यम से उस thread पर action करने के लिए बनाया जाता है। यह एक अन्य component या loopers जो एक विशेष thread में Message processing का प्रभारी है, के साथ मिलकर काम करता है।

जब एक handlers बनाया जाता है, तो यह constructor में एक Looper objects प्राप्त कर सकता है, जो इंगित करता है कि handlers किस thread से जुड़ा हुआ है। यदि आप main thread से जुड़े handlers का उपयोग करना चाहते हैं, तो आपको `Looper.getMainLooper()` function को कॉल करके main thread से जुड़े Looper का उपयोग करना होगा।

एक handlers के दो main उपयोग हैं—

- भविष्य में किसी बिन्दु पर execute किए जाने वाले messages और runnables को schedule करने के लिए।
- अपने से एक अलग thread पर perform किए जाने वाले action को enqueue करना।

Scheduling messages को निम्नलिखित function के द्वारा पूरा किया जाता है—

- **POST** : `post(Runnable)`, `postAtTime(Runnable, long)`, `postDelayed(Runnable, long)`,
- **SendMessage** : `sendMessage(int)`, `sendMessage(Message)`, `sendMessageAtTime(Message, long)` and `sendMessageDelayed(Message, long)`.

Post versions आपको Runnable objects को message queue द्वारा call करने की अनुमति देते हैं जब वे receive होते हैं।

SendMessage versions आपको एक Message object को डेटा के बंडल के साथ संलग्न करने की अनुमति देता है जो handler के `handleMessage(Message)` method द्वारा process किया जाएगा।

Post() और sendMessage() के बीच अंतर

जब हम अपने handler object के बारे में कुछ भी जाने बिना UI thread पर कुछ कोड execute करना चाहते हैं तो हम post का उपयोग करते हैं। यह कई मामलों में समझ में आता है जहाँ UI thread पर arbitrary कोड को execute करने की आवश्यकता होती है।

उदाहरण के लिए, कुछ thread के साथ background thread से UI को अपडेट करने के लिए, यहाँ आप UI thread से जुड़ा hanlder बना सकते हैं, और फिर Runnable के रूप में एक्शन post कर सकते हैं।

```
Handler handler = new Handler(Looper.getMainLooper());
handler.postDelayed(new Runnable() {
    @Override
```

140 डेवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन

```

public void run() {
    // update the UI from here
}
}, 1000);

```

लेकिन कुछ मामलों में, यदि हम यह व्यवस्थित करना चाहते हैं कि UI thread को क्या sent किया जा रहा है और हमारे पास specific functions हैं जिन्हें हम execute करना चाहते हैं, तो उस स्थिति में आप sendMessage() का उपयोग कर सकते हैं।

```

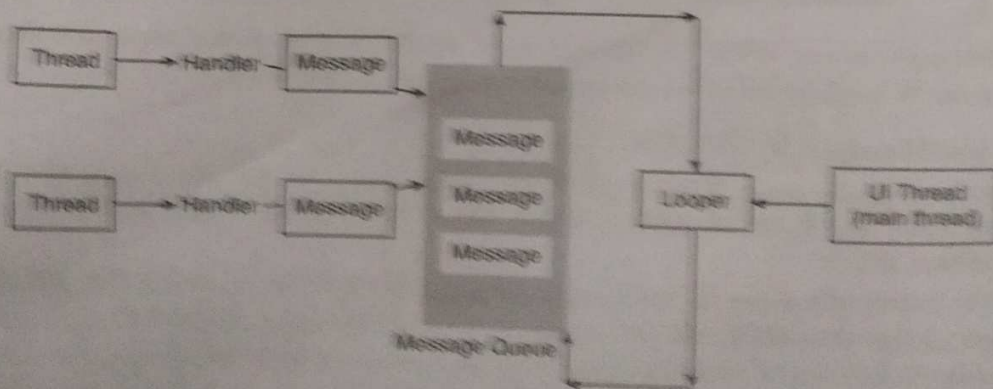
Handler handler = new Handler() {
    @Override
    public void handleMessage(Message msg) {
        if (msg.what == 0) {
            updateUI();
        } else {
            showError();
        }
    }
};

```

```

Thread thread = new Thread() {
    @Override
    public void run() {
        doSomeWork();
        if (succeed) {
            //we can't update the UI from here so we'll signal our handler
            and it will do it for us.
            handler.sendMessage(0);
        } else {
            handler.sendMessage(1);
        }
    }
};

```



चित्र 9

6.9. Runnable

जावा में interface runnable मूल रूप से Abstraction for an Executable Command है। (In java the interface Runnable is basically in Abstraction for an Executable Command.)

यह एक इंटरफेस है और इसे java.lang पैकेज में परिभाषित किया गया है और इसमें abstract method है जिसे run() कहा जाता है।

```
package java.lang;

public interface Runnable
{
    public abstract void run();
}
```

Runnable इंटरफेस को किसी भी class द्वारा implement किया जाना चाहिए, जिनके instances को thread द्वारा execute किया जाना है। Class को no arguments RUN () method के द्वारा परिभाषित करना चाहिए।

यह इंटरफेस उन object के लिए एक सामान्य प्रोटोकॉल प्रदान करने के लिए डिज़ाइन किया गया है जो active होते समय code को execute करना चाहते हैं। उदाहरण के लिए, Runnable को class Thread द्वारा implement किया जाता है। active होने का मतलब है कि एक thread शुरू किया गया है और अभी तक stop नहीं गया है।

ज्यादातर मामलों में, यदि आप केवल run() method और कोई अन्य Thread methods को ओवरराइड करने की योजना बना रहे हैं तो Runnable इंटरफेस का उपयोग किया जाना चाहिए।

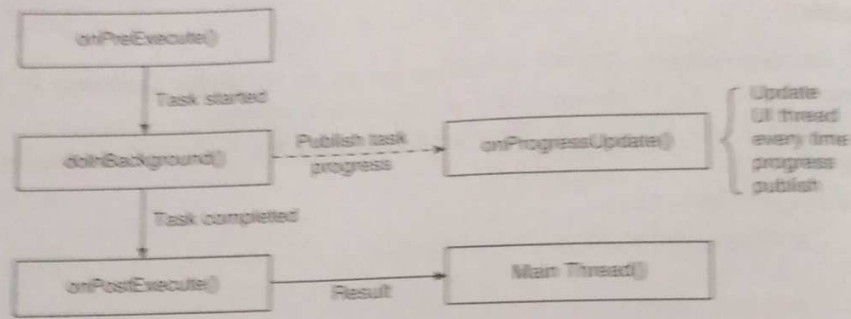
6.10. AsyncTask

AsyncTask एन्ड्रॉइड द्वारा प्रदान किया गया एक abstract class है जो हमें background में heavy tasks करने के लिए स्वतंत्रता देता है और UI Thread को light में रखता है और इस प्रकार एप्लिकेशन को अधिक responsive बनाता है।

Android एप्लिकेशन लॉन्च होने पर एक ही Thread पर चलता है। इस single thread मॉडल कार्यों के कारण प्रतिक्रिया प्राप्त करने में अधिक समय लगता है जो एप्लिकेशन को non-responsive बना सकता है। इससे बचने के लिए हम dedicated thread पर background में हवी टास्क करने के लिए एंड UI thread पर Result pass करने के लिए एन्ड्रॉइड AsyncTask का इस्तेमाल करते हैं। इसलिए Android एप्लिकेशन में AsyncTask का उपयोग UI Thread को हर समय responsive रखता है।

Android AsyncTask क्लास में उपयोग किए जाने वाले basic method को नीचे परिभाषित किया गया है—

- **doInBackground()** : इस method में कोड शामिल है जिसे background में execute करने की आवश्यकता है। इस method में हम UI Thread को कई बार publishProgress() method से results भेज सकते हैं। यह सूचित करने के लिए कि background प्रोसेसिंग पूरी हो गई है, हमें सिर्फ return statements का उपयोग करने की आवश्यकता है।
- **onPreExecute()** : इस method में वह कोड होता है जिसे background processing शुरू होने से पहले execute किया जाता है।
- **onPostExecute()** : इस method को doInBackground method के पूर्ण होने के बाद call किया जाता है। DoInBackground के Result इस method में pass किया जाता है।
- **onProgressUpdate()** : यह method doInBackground method से progress अपडेट प्राप्त करती है, जो कि publishProgress method के माध्यम से publish होती है, और यह method UI Thread को update करने के लिए इस progress update का उपयोग कर सकती है।



चित्र 9

Android AsyncTask क्लास में उपयोग किए जाने वाले तीन generic type's नीचे दिए गए हैं—

- **Params** : execution पर भेजे गए task के parameter का प्रकार।
- **Progress** : background computation के दौरान published progress इकाइयों का प्रकार।
- **Result** : background computation के result का प्रकार।

6.11. Calling Web Services and Consuming JSON Data from Web Services

(a) **Android Web Services**—एक web service मूल रूप से interoperability प्रदान करने के लिए आवश्यक है, यह विभिन्न एप्लिकेशन को एक-दूसरे के साथ communicate करने और आपस में डेटा और services साझा करने को अनुमति देता है। web service सभी प्रकार के client application के लिए इन प्रकार के app server कार्यों करने के लिए standard प्रदान करता है।

(b) **Android Web Services Components**—Web सर्वर आर्किटेक्चर में आमतौर पर निर्धारित roles होते हैं—

1. **Publisher**—Publisher को Service provider के रूप में समझा जा सकता है। Publisher Web Services बनाने और Client के लिए उपलब्ध करने के लिए जिम्मेदार है।

2. **Subscriber**—Subscriber और कुछ नहीं service requester है। Service requester वह है जिसे Web Service से संपर्क करने की आवश्यकता है। Client क्लाइंट एप्लिकेशन के माध्यम से संपर्क करता है। यह एप्लिकेशन किसी भी Language पर आधारित हो सकता है।

3. **Broker**—यहाँ broker वह एप्लिकेशन है जो UDDI तक पहुंच प्रदान करता है। UDDI उपयोगकर्ता descriptive, discovery और integration के लिए है। यह client एप्लिकेशन को web service का पता लगाने में मदद करता है। जो Service इस प्रकार है—

(i) **Publish**—Publishers publishing the web services means informing the broker about its existence. It is done using the Broker's interface to make it easily accessible to the subscribers.

Web services को प्रकाशित करने वाले प्रकाशकों का मतलब है कि ब्रोकर को उसके अभिलेख को बना देना। यह ब्रोकर के इंटरफेस का उपयोग करके किया जाता है ताकि प्रकाशकों तक अभिलेख में पहुंच आ सके।

(ii) **Subscribe**—The subscriber will consult the broker to locate the published web services easily.

संभावित रूप से प्रकाशित Web service का पता लगाने के लिए ब्रोकर से संपर्क किया जाता है।

(iii) Bind—Once the information regarding the web services is gained from the broker, the subscriber can bind the web service.

एक बार ब्रोकर से Web services के बारे में जानकारी प्राप्त हो जाने के बाद, सब्सक्राइबर Web service से जुड़ सकता है।

(c) **WebService Architecture**—एक ideal Webservice provider को चार logical layers के around डिजाइन किया गया है, जो उन तरीकों को परिभाषित करता है जिनमें डेटा को transport, encode, expose और users द्वारा खोजा जाता है।

Layers	Responsibility
Transport	नेटवर्क के माध्यम से message ले जाना (using HTTP, SMTP, FTP,...)
Messaging	एक्सचेंज किए जाने वाले डेटा की एन्कोडिंग करना (XML)
Description	WSDL (Web Service Description Language) end point से उपलब्ध public methods का वर्णन करने के लिए उपयोग किया जाता है।
Discovery	UDDI (Universal Description & Discovery Integration) common registry के माध्यम से services के location और publishing की सुविधा प्रदान करता है।

(d) **Characteristics of Web Services in Android**—Web Services की विशेषतायें निम्नलिखित हैं—

1. Web services XML—आधारित है। वे इसे अपने data representational layer और इसकी transportational layer पर उपयोग करते हैं क्योंकि यह नेटवर्किंग, ऑपरेटिंग सिस्टम या यहाँ तक कि प्लेटफॉर्म बाइंडिंग को हटा देता है। ये services अपने core level पर अत्यधिक interoperable हैं।

2. Web services loosely coupled हैं। इसका मतलब है कि consumer Web Services और Web Services के provider's सीधे एक साथ बंधे नहीं हैं।

3. Web services में सिंक्रोनस या एसिंक्रोनस होने की क्षमता होती है। यहाँ Synchronous को क्लाइंट को service के execution की binding के रूप में समझा जा सकता है। दूसरी ओर Asynchronous का अर्थ है client को पहले किसी service को लागू करने की अनुमति देना और बाद में अन्य कार्यों को execute करना।

4. Web services, Remote Procedure Calls का समर्थन करती हैं। Remote Procedure Calls को अक्सर RPC के रूप में संदर्भित किया जा सकता है। ये RPC क्लाइंट को XML का उपयोग करके Remote object पर विभिन्न function, methods और services को लागू करने देती हैं।

5. Web services में Document exchange का समर्थन है। वास्तव में, XML के पास डेटा के साथ-साथ complex documents का प्रतिनिधित्व करने के लिए एक बहुत ही सामान्य तरीका है। इसके साथ ही, इन documents का प्रतिनिधित्व करने के लिए इसके विभिन्न तरीके हैं।

(e) **Types of Web Services in Android**—विभिन्न वेब सेवाएँ निम्नलिखित हैं—

1. **XML-RPC**—In XML-RPC, RPC remote procedure calls के लिए है। यह इंटरनेट पर उपकरणों की एक विशाल श्रृंखला के बीच डेटा के आदान-प्रदान के लिए XML आधारित प्रोटोकॉल है।

2. **UDDI**—UDDI का अर्थ है सार्वभौमिक वर्णनात्मक, खोज और एकीकरण। यह एक XML आधारित standard है, जिसका उपयोग नई Web services की detailing, publishing और discovering के लिए किया जाता है।

3. **SOAP**—यहाँ SOAP सिंपल ऑब्जेक्ट एक्सेस प्रोटोकॉल के लिए है। यह एक XML आधारित Web service प्रोटोकॉल है जिसका उपयोग HTTP (हाइपरटेक्स्ट ट्रांसफर प्रोटोकॉल) या एसएमटीपी (सिंपल मैसेज ट्रांसफर

144 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

प्रोटोकॉल) पर डेटा या डॉक्यूमेंट के आदान-प्रदान के लिए किया जाता है। यह स्वतंत्र प्रक्रियाओं के communication के अनुमति देता है जो disparate systems पर काम करते हैं।

4. REST—REST Representational State Transfer है। यह devices और इंटरनेट के बीच संचार और कनेक्टिविटी प्रदान करता है।

(f) Advantages of Web Services—Web services के लाभ निम्नलिखित हैं—

1. Web services विभिन्न Application के बीच interoperability को enable करती है।
2. Web services का उपयोग करने के बहुत महत्वपूर्ण लाभों में से एक पुनः प्रयोज्य है।
3. Web services application और organization के भीतर और बाहर तेजी से संचार प्रदान करती है।
4. वे विभिन्न Application के बीच संचार को सक्षम करने के लिए एक quality industry-standard प्रोटोकॉल का उपयोग करते हैं।
5. वे Web services को लागू करने के लिए कम लागत वाले इंटरनेट के उपयोग को सक्षम करने के लिए HTTP पर SOAP का उपयोग करते हैं।
6. Web services को स्टैंडर्ड इंटरनेट टेक्नोलॉजी पर deploy किया जाता है।
7. वे हमें इंटरनेट पर मौजूदा कोड के कार्यों को उजागर करने की अनुमति देते हैं।

(g) Android Web Services Limitations—नीचे दी गई sub services के कुछ हानियाँ भी हैं—

1. Web services ब्राउजर से access नहीं हो पाती है।
2. वे emerging Web developments का लाभ नहीं उठाते हैं।
3. Web services द्वारा उपयोग किया जाने वाला HTTP प्रोटोकॉल विश्वसनीय नहीं है और असुरक्षित है।

(h) The Client Side - Consuming WebServices—WebServices को invoke करने और उपभोग करने के दो व्यापक रूप से उपयोग किए जाते हैं—

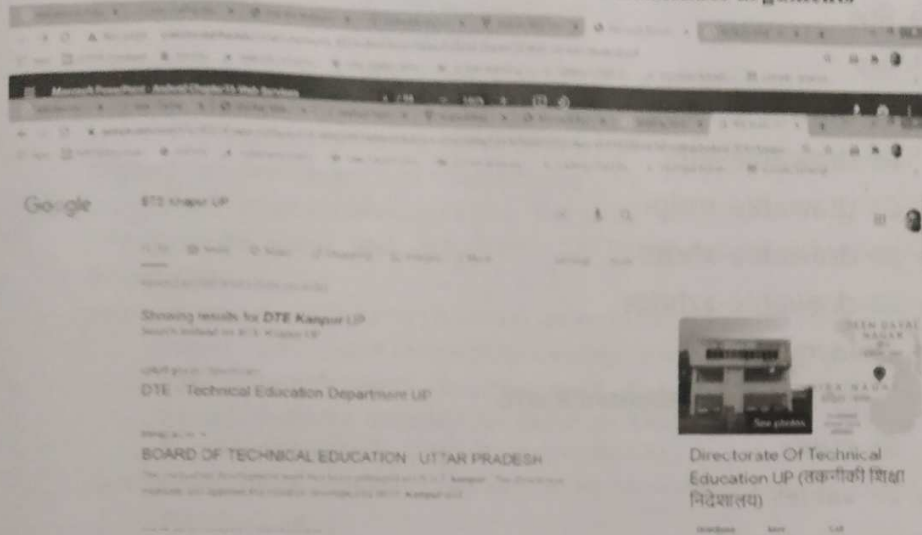
Representational State Transfer (REST) यह HTTP/HTTPS के लिए सामान्य ऑपरेशन GET, POST, PUT, DELETE में इसके ऑपरेशन को सामान्य तरीके से जोड़कर HTTP प्रोटोकॉल के साथ टाई करता है। इस मॉडल में एक साधारण invocation mode है। service कॉल एक URL पर rely करते हैं जो arguments को भी carry कर सकता है। सेन्डर और रिसीवर को इस बात की समझ होनी चाहिए कि वे एक-दूसरे से डेटा आइटम कैसे pass करते हैं। एक उदाहरण के रूप में Google Maps API REST मॉडल का उपयोग करता है।

Remote Procedure Call (RPC) - Remote services को End Point Provider द्वारा संग्रहीत और expose किए जाने योग्य discoverable functions (या method calls) के coherent संग्रह के रूप में देखा जाता है। इस श्रेणी के कुछ कार्यान्वयन में शामिल हैं—सिंपल ऑब्जेक्ट एक्सेस प्रोटोकॉल (SOAP), कॉमन ऑब्जेक्ट रिक्वेस्ट ब्रोक आर्किटेक्चर (CORBA), माइक्रोसॉफ्ट के डिस्ट्रिब्यूटेड कॉम्पोनेंट ऑब्जेक्ट मॉडल (DCOM) और सन माइक्रोसिस्टम्स के जावा/रिमोट मेथड इनवोकेशन (RMI)।

उदाहरण—REST का उपयोग करना। निम्न URL का उपयोग Google खोज service पर कॉल करने के लिए किया जाता है, जो “BTE Kanpur UP” विषय पर लिंक प्रदान करने के लिए कहता है।

Transport	Provider	Action	Arguments
↓	↓	↓	↓
https://www.google.com/search?q=BTE+kanpur+UP			

Example of a REST web-service called with a URL that includes arguments



चित्र 10

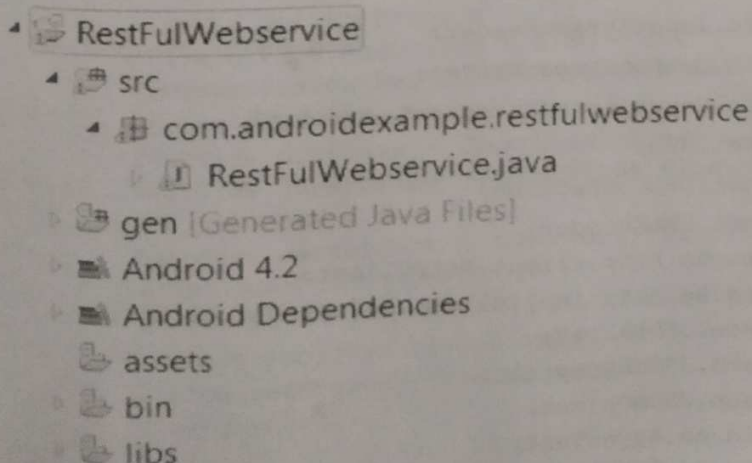
(i) Restful Webservice Call & Get and Consume & Parse JSON Data

इस उदाहरण में JSON डेटा प्राप्त करने के लिए और उस JSON डेटा को parse करने के लिए restful webservice को कॉल करना समझाया गया है।

Steps :

1. सर्वर से php फाइल (php) को कॉल करें और php में JSON डेटा बनाएँ।
2. Consume JSON data by android और activity पर दिखाएँ।
3. Parse JSON डेटा और स्क्रीन (activity) पर parsed डेटा दिखाएँ।

Project Structure :



- ↳ libs
- ↳ res
 - ↳ drawable-hdpi
 - ↳ drawable-ldpi
 - ↳ drawable-mdpi
 - ↳ drawable-xhdpi
 - ↳ drawable-xxhdpi
 - ↳ layout
 - rest_ful_webservice.xml
 - ↳ menu
 - ↳ values
 - ↳ values-v11
 - ↳ values-v14
- AndroidManifest.xml
- ic_launcher-web.png
- proguard-project.txt
- project.properties

File : src/RestfulWebservice.java

This file contain Asynchronous server request, got JSON response and parse JSON and show on screen.

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.UnsupportedEncodingException;
import java.net.URL;
import java.net.URLConnection;
import java.net.URLEncoder;
import org.apache.http.client.HttpClient;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import android.os.AsyncTask;
import android.os.Bundle;
```



```

import android.app.Activity;
import android.app.ProgressDialog;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class RestFulWebservice extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.rest_ful_webservice);
        final Button GetServerData = (Button)
            findViewById(R.id.GetServerData);
        GetServerData.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View arg0) {
                // WebServer Request URL
                String serverURL = "/media/webservice/JSONReturn.php";
                // Use AsyncTask execute Method To Prevent ANR Problem
                new LongOperation().execute(serverURL);
            }
        });
    }

    // Class with extends AsyncTask class
    private class LongOperation extends AsyncTask<String, Void, Void> {
        // Required initialization
        private final HttpClient Client = new DefaultHttpClient();
        private String Content;
        private String Error = null;
        private ProgressDialog Dialog = new
            ProgressDialog(RestFulWebservice.this);
        String data = "";
        TextView UIUpdate = (TextView) findViewById(R.id.output);
        TextView JSONParsed = (TextView) findViewById(R.id.JSONParsed);
        int sizeData = 0;
        EditText serverText = (EditText) findViewById(R.id.serverText);
        protected void onPreExecute() {
            // NOTE: You can call UI Element here.
            //Start Progress Dialog (Message)
            Dialog.setMessage("Please wait..");
            Dialog.show();
            try{

```

```

        // Set Request parameter
        data += "&" + URLEncoder.encode("data", "UTF-8") +
"="+serverText.getText();
    } catch (UnsupportedEncodingException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

// Call after onPreExecute method
protected void doInBackground(String... urls) {
    /******* Make Post Call To Web Server *****/
    BufferedReader reader=null;
    // Send data
    try
    {
        // Defined URL where to send data
        URL url = new URL(urls[0]);
        // Send POST data request
        URLConnection conn = url.openConnection();
        conn.setDoOutput(true);
        OutputStreamWriter wr = new
        OutputStreamWriter(conn.getOutputStream());
        wr.write( data );
        wr.flush();
        // Get the server response
        reader = new BufferedReader(new
        InputStreamReader(conn.getInputStream()));
        StringBuilder sb = new StringBuilder();
        String line = null;
        // Read Server Response
        while((line = reader.readLine()) != null)
        {
            // Append server response in string
            sb.append(line + " ");
        }
        // Append Server Response To Content String
        Content = sb.toString();
    }
    catch(Exception ex)
    {
        Error = ex.getMessage();
    }
    finally
    {

```

AVNISH MISHRA

```

/***** End Parse Response JSON Data *****/
//Show Parsed Output on screen (activity)
jsonParsed.setText( OutputData );
} catch (JSONException e) {
e.printStackTrace();
}
}
}
}
}

```

File : rest_ful_webservice.xml

```

<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillViewport="true"
    android:background="#FFFFFF"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical" >
        <EditText
            android:paddingTop="20px"
            android:id="@+id/serverText"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="" />
        <Button
            android:paddingTop="10px"
            android:id="@+id/GetServerData"
            android:text="Restful Webservice Call"
            android:cursorVisible="true"
            android:clickable="true"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
        />
        <TextView
            android:paddingTop="20px"
            android:textStyle="bold"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Server Response (JSON): " />
    </LinearLayout>
</ScrollView>

```



```

<TextView
    android:paddingTop="16px"
    android:id="@+id/output"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Output : Click on button to get server data." />
<TextView
    android:paddingTop="20px"
    android:textStyle="bold"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Parsed JSON : " />
<TextView
    android:paddingTop="16px"
    android:id="@+id/JSONParsed"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
/>
</LinearLayout>
</ScrollView>

```

File : AndroidManifest.xml

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/
android"
    package="com.ggpmbd.restfulwebservice"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="8" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.androidexample.restfulwebservice.RestFulWebse
            rvice"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

```

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

Server file : /media/webservice/JSONReturn.php

Create JSON data on server

```
<?php
// Get Post Data
$data = urldecode($_POST['data']);
$JSONData = array();
$JSONTempData = array();
for($i=1;$i<4; $i++)
{
    $JSONTempData = array();
    $JSONTempData['name'] = $data.$i;
    $JSONTempData['number'] = $data.$i;
    $JSONTempData['date_added'] = $data.$i;
    $JSONData[] = $JSONTempData;
}
$outputArr = array();
$outputArr['Android'] = $JSONData;
// Encode Array To JSON Data
print_r(JSON_encode($outputArr));
?>
```

प्रश्नावली

1. Thread क्या है? Android में इनका उपयोग समझाइये।
2. Android में thread को किन method के द्वारा बनाया जाता है?
3. Run() function क्या है?
4. Thread के प्रकार को समझाइये।
5. Main thread को बताये।
6. UI thread को समझाइये?
7. Handles क्या है? इसका प्रयोग कैसे किया जाता है?
8. Asyn task को समझाइये।
9. Thread life cycle को चित्र सहित समझाइये।
10. Looper क्या है?
11. Web service क्या होती है? इसकी विशेषतायें बताइये।
12. विभिन्न प्रकार की web service के बारे में समझाइये।
13. Restfull webservice call तथा parse JSON Data के अनुप्रयोग को समझाइये।

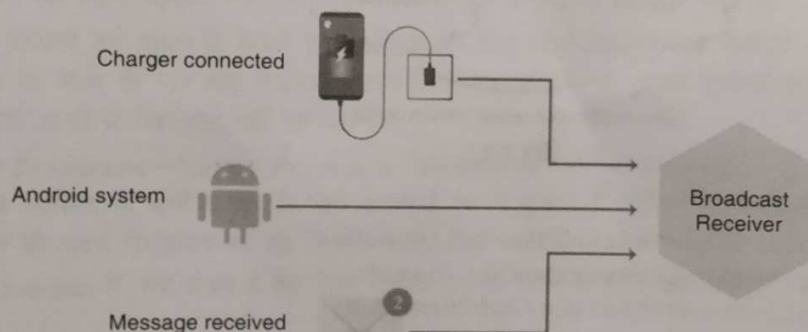


सूचनाएँ और सेवाएँ (Notifications & Services)

Broadcast Receivers (Introduction, different ways to register a broadcast receiver), Introduction to Notification, Overview & Types of services, implementing a service, Service lifecycle

7.1. Broadcast Receivers

एन्ड्रॉइड ब्रॉडकास्ट रिसीवर एक एन्ड्रॉइड घटक है जिसका उपयोग संदेशों को सिस्टम या अन्य अनुप्रयोगों पर प्रसारित करने के लिए किया जाता है। प्रसारण संदेश को घटना या आशय के रूप में संदर्भित किया जाता है। प्रसारण रिसीवर, गतिविधियों के विपरीत, उपयोगकर्ता इंटरफेस है। एक प्रकाशित सदस्यता डिजाइन पैटर्न के समान है। इसका उपयोग Asynchronous इंटर-प्रोसेस संचार के लिए किया जाता है।



चित्र 1

Broadcast Receivers अन्य अनुप्रयोगों से या सिस्टम से ही प्रसारण संदेशों का जवाब देते हैं। इन संदेशों को कुछ समय इवेंट या इरादे कहा जाता है। उदाहरण के लिए, अन्य application को यह बताने के लिए प्रसारण भी शुरू कर सकते हैं कि कुछ डेटा डिवाइस पर डाउनलोड किया गया है और उनके उपयोग के लिए उपलब्ध है, इसलिए यह प्रसारण रिसीवर है जो इस संचार को रोक देगा और उचित कार्रवाई शुरू करेगा।

सिस्टम ब्रॉडकास्ट किए गए इरादों के लिए ब्रॉडकास्ट रिसीवर वर्क्स बनाने के लिए दो महत्वपूर्ण कदमों का पालन कर रहे हैं—

- प्रसारण रिसीवर बनाना।
- पंजीकरण प्रसारण रिसीवर।

यदि आप अपने कस्टम इरादों को लागू करने जा रहे हैं तो आपको उन इरादों को बनाना और प्रसारित करना होगा।

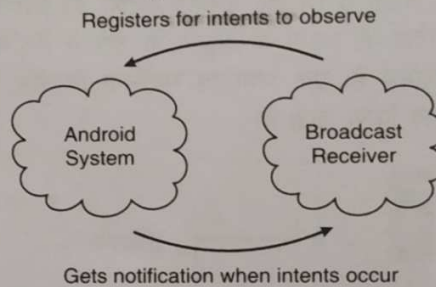
Broadcast Receiver बनाना

एक Broadcast Receiver को ब्रॉडकास्ट रिसीवर क्लास के उपवर्ग के रूप में लागू किया जाता है। onReceive() विधि को implement किया जाता है जहाँ प्रत्येक संदेश को एक आशय ऑब्जेक्ट पैरामीटर के रूप में होता है।

```
public class MyReceiver extends BroadcastReceiver
{
    @Override
    public void onReceive(Context context, Intent intent) {
        Toast.makeText(context, "Intent Detected.",
            Toast.LENGTH_LONG).show();
    }
}
```

Broadcast Receiver का पंजीकरण

एक आवेदन AndroidManifest.xml फाइल में एक प्रसारण रिसीवर दर्ज करके विशिष्ट प्रसारण इरादों के सुनता है। विचार करें कि हम सिस्टम जनित इवेंट ACTION_BOOT_COMPLETED के लिए MyReceiver पंजी करने जा रहे हैं जो एन्ड्रॉइड सिस्टम द्वारा बूट प्रक्रिया पूरी करने के बाद सिस्टम द्वारा निकाल दिया जाता है।



चित्र 7.2

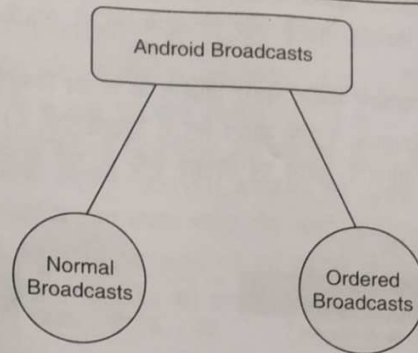
```
<application
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme" >
    <receiver android:name="MyReceiver">
        <intent-filter>
            <action
                android:name="android.intent.action.BOOT_COMPLETED">
            </action>
        </intent-filter>
    </receiver>
</application>
```

अब जब भी आपका एन्ड्रॉइड डिवाइस हटा दिया जाता है, तो इसे ब्रॉडकास्टर मॉनिसर द्वारा रोक दिया जाएगा और onReceive() के अंदर लागू तर्क निष्पादित किया जाएगा।

एन्ड्रॉइड में Broadcasts के प्रकार—

एन्ड्रॉइड में मोटे तौर पर दो प्रकार के प्रसारण रिसीवर हैं—

1. Ordered Broadcasts
2. Normal Broadcasts



चित्र 7.3

1. Ordered Broadcasts—Ordered Broadcasts समकालिक प्रसारण होते हैं, और उचित क्रम में किए जाते हैं। यह आदेश एन्ड्रॉइड प्राथमिकता विशेषता द्वारा तय किया जाता है। उच्चतम प्राथमिकता के साथ प्रसारण पहले निष्पादित (execute) होगा और एक ही प्राथमिकता के साथ प्रसारण किसी भी आदेश का पालन नहीं करेगा।

Ordered Broadcasts उचित क्रम में किया जाता है क्योंकि उन्हें कुछ प्राथमिकता के साथ सौंपा जाता है। प्राथमिकता एन्ड्रॉइड-प्राथमिकता का उपयोग करके प्रसारण को सौंपा जाता है। प्राथमिकता की मदद से, उच्चतम प्राथमिकता के साथ प्रसारण पहले निष्पादित (execute) होगा। यदि दो प्रसारणों को एक ही प्राथमिकता दी जाती है तो कोई आदेश का पालन नहीं किया जाएगा। आदेशित प्रसारण को सिंक्रोनस प्रसारण भी कहा जाता है।

ऐसे में एक प्रसारण एक समय में केवल एक रिसीवर को दिया जाता है। जब एक रिसीवर एक प्रसारण प्राप्त करता है यह रिसीवर पर निर्भर है। यदि कोई रिसीवर चाहता है, तो यह प्रसारण को अगले रिसीवर तक पहुँचाता है और नहीं चाहता तो प्रसारण अगले रिसीवर तक नहीं पहुँचाता है।

2. Normal Broadcasts—Normal Broadcasts अतुल्यकालिक और अव्यवस्थित होते हैं। ये रिसीवर एक समय में, कभी-कभी अक्रमित या सभी चलाते हैं। यही कारण है कि वे कुशल हैं, लेकिन परिणामों के पूर्ण उपयोग की कमी है। इन प्रसारणों को संदर्भ सेंडब्रॉडकास्ट का उपयोग करके भेजा जाता है।

Normal Broadcasts में, यह संभव है कि सिस्टम ओवरहेड से बचने के लिए एक समय में केवल एक प्रसारण भेजा है।

आवेदनों से हमें जो NOTIFICATION प्राप्त होती है, उसे भी मौन (mute) किया जा सकता है।

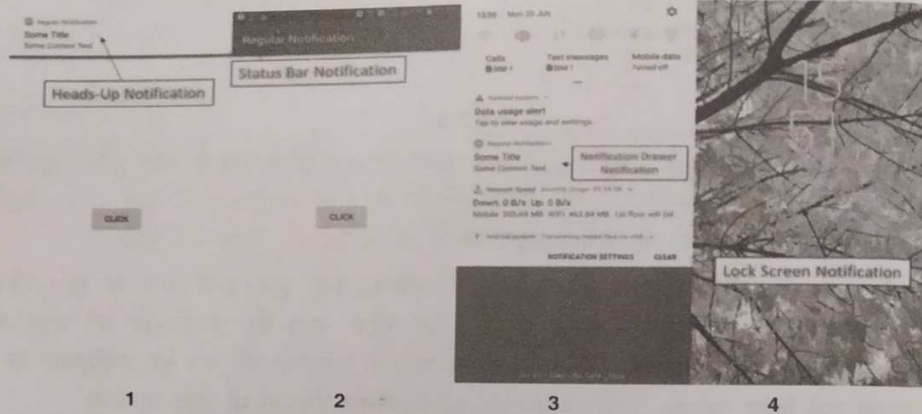
7.2. Notifications in Android

एक Notification एक संदेश है जिसे आप अपने एप्लिकेशन के सामान्य यूआई के बाहर उपयोगकर्ता को प्रदर्शित कर सकते हैं। जब आप सिस्टम को notification जारी करने के लिए कहते हैं, तो यह पहली बार notification क्षेत्र में एक आइकन के रूप में दिखाई देता है। Notification का विवरण देखने के लिए, उपयोगकर्ता notification window खोलता है। notification क्षेत्र और notification window दोनों सिस्टम-नियंत्रित क्षेत्र हैं जिन्हें उपयोगकर्ता किसी भी समय देख सकता है।

NOTIFICATION एक प्रकार का संदेश, चेतावनी या किसी एप्लिकेशन की स्थिति (शायद पृष्ठभूमि में चल रहा है) है जो एन्ड्रॉइड के यूआई तत्वों में दिखाई दे रहा है या उपलब्ध है। यह एप्लिकेशन पृष्ठभूमि में चल रहा हो सकता है लेकिन उपयोगकर्ता द्वारा उपयोग में नहीं। एक NOTIFICATION का उद्देश्य उपयोगकर्ता द्वारा या तो उपयोगकर्ता या सिस्टम द्वारा आवेदन में शुरू की गई प्रक्रिया के बारे में उपयोगकर्ता को सूचित करना है। यह लेख किसी ऐसे व्यक्ति की मदद कर सकता है जो विकास के उद्देश्यों के लिए NOTIFICATION बनाने के लिए कड़ी मेहनत कर रहा है।

डेवलपर के आधार पर सूचनाएँ विभिन्न प्रारूपों और डिजाइनों की जा सकती हैं। सामान्य तौर पर, किसी ने चार प्रकार की सूचनाओं को देखा होगा—

1. Status Bar Notification (वर्तमान समय, बैटरी प्रतिशत के समान लेआउट में दिखाई देती है)
2. Notification drawer Notification (ड्रॉप-डाउन मेनू में दिखाई देती है)
3. Heads-Up Notification (ओवरले स्क्रीन पर दिखाई देती है, पूर्व व्हाट्सएप NOTIFICATION ओर संदेश)
4. लॉक-स्क्रीन NOTIFICATION



चित्र 7.4

NOTIFICATION का विवरण देखने के लिए, आपको आइकन का चयन करना होगा जो NOTIFICATION के बारे में विस्तार से NOTIFICATION दराज प्रदर्शित करेगा। वर्चुअल डिवाइस के साथ एमुलेटर के साथ काम करते समय, आपको इसे विस्तारित करने के लिए स्टेटस बार को क्लिक करना और नीचे खींचना होगा जो आपको इस प्रकार विस्तार देगा। यह सिर्फ 64 डीपी लम्बा होगा और और इसे सामान्य दृश्य कहा जाएगा।



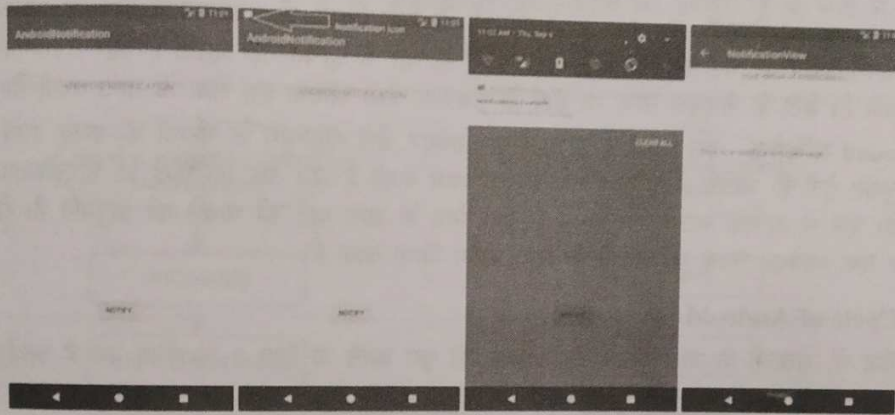
चित्र 7.5

Set Android Notification Properties

एन्ड्रॉइड NOTIFICATION की संपत्तियाँ NotificationCompat.Builder object का उपयोग करके सेट की जाती हैं। NOTIFICATION गुणों में से कुछ का नीचे उल्लेख कर रहे हैं—

- setSmallIcon () : यह NOTIFICATION का आइकन सेट करता है।
- setContentType () : इसका उपयोग NOTIFICATION का शीर्षक निर्धारित करने के लिए किया जाता है।
- setContentText () : इसका उपयोग टेक्स्ट संदेश सेट करने के लिए किया जाता है।
- setAutoCancel () : यह NOTIFICATION की रद्द की गई संपत्ति निर्धारित करता है।
- setPriority () : यह NOTIFICATION की प्राथमिकता निर्धारित करता है।

Android Notification Example



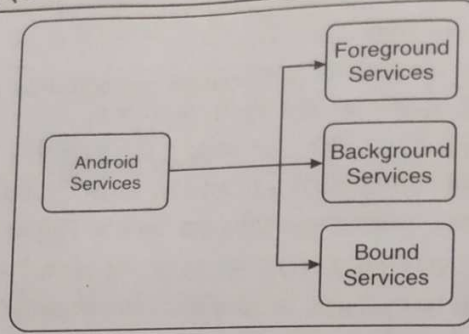
चित्र 7.6

7.3. Services in Android

एन्ड्रॉइड में सेवाएँ एक विशेष घटक हैं जो लम्बे समय से चल रहे ऑपरेशन कार्यों को करने के लिए पृष्ठभूमि में चलाने के लिए एक आवेदन की सुविधा प्रदान करता है। एक सेवा का मुख्य उद्देश्य यह सुनिश्चित करना है कि आवेदन पृष्ठभूमि में सक्रिय रहे ताकि उपयोगकर्ता एक ही समय में कई अनुप्रयोगों को संचालित कर सके। एन्ड्रॉइड सेवाओं के लिए एक उपयोगकर्ता इंटरफेस वांछनीय नहीं है क्योंकि इसे बिना किसी उपयोगकर्ता के हस्तक्षेप के लम्बे समय से चलने वाली प्रक्रियाओं को संचालित करने के लिए डिज़ाइन किया गया है। एक सेवा पृष्ठभूमि में लगातार चल सकती है, भले ही आवेदन बंद हो या उपयोगकर्ता किसी अन्य एप्लिकेशन पर स्विच कर ले। इसके अलावा, आवेदन घटक अंतर-प्रक्रिया संचार (आईपीसी) को पूरा करने के लिए खुद को सेवा के लिए बाध्य कर सकते हैं। एन्ड्रॉइड सेवाओं और thread के बीच एक बड़ा अंतर है, किसी को दोनों के बीच भ्रमित नहीं किया जाना चाहिए। थ्रेड ऑपरेटिंग सिस्टम द्वारा प्रदान की जाने वाली एक सुविधा है जो उपयोगकर्ता को पृष्ठभूमि में संचालन करने की अनुमति देगी है। जबकि सेवा एक एन्ड्रॉइड घटक है जो लम्बे समय से चल रहे ऑपरेशन को करता है जिसके बारे में उपयोगकर्ता को पता नहीं हो सकता है क्योंकि उसके पास यूआई नहीं है।

Types of Android Services

1. **Foreground सेवाएँ**—उपयोगकर्ता को इसके चल रहे कार्यों के बारे में सूचित करने वाली सेवाओं को Foreground सेवाएँ कहा जाता है। उपयोगकर्ता चल रहे कार्य के बारे में प्रदान की गई सूचनाओं द्वारा सेवा के साथ बातचीत कर सकते हैं। जैसे कि फाइल डाउनलोड करने में, उपयोगकर्ता डाउनलोड करने में प्रगति को ट्रैक रख सकता है और प्रक्रिया को रोक और फिर से शुरू भी कर सकता है।



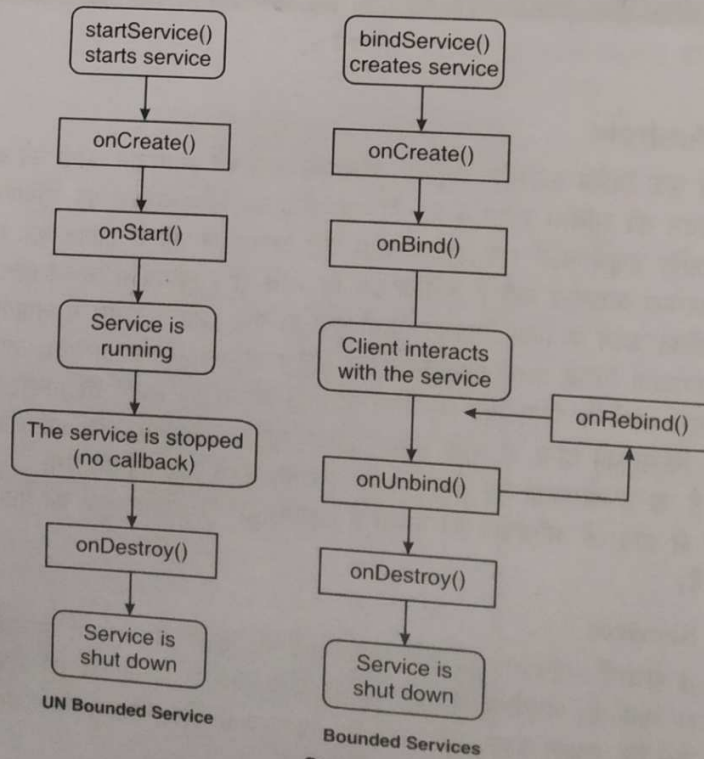
चित्र 7.7

2. Background सेवाएँ—Background सेवाओं के लिए किसी भी उपयोगकर्ता हस्तक्षेप की आवश्यकता नहीं है। ये सेवाएँ उपयोगकर्ता को चल रहे Background कार्यों के बारे में सूचित नहीं करती हैं और उपयोगकर्ता भी उन तक नहीं पहुँच सकते हैं। डेटा के शेड्यूल सिंक या डेटा के भंडारण जैसी प्रक्रिया इस सेवा के तहत आती हैं।

3. Bound सर्विसेज—इस प्रकार एन्ड्रॉइड सेवा आवेदन जैसे गतिविधि के घटकों को इसके साथ खुद को बांधने की अनुमति देती है। बाउंड सेवाएँ तब तक अपना काम करते हैं जब तक कि कोई भी एप्लिकेशन कंपोनेंट इसके लिए बाध्य हो। एक से अधिक घटक एक समय में एक सेवा के साथ खुद को बांधने की अनुमति है। सेवा onBind() विधि के साथ एक आवेदन घटक को बांधने के लिए प्रयोग किया जाता है।

The Life Cycle of Android Services

एन्ड्रॉइड में, सेवाओं के पास अपने जीवन चक्र को पूरा करने के लिए 2 संभावित पथ हैं अर्थात् Started and Bounded.

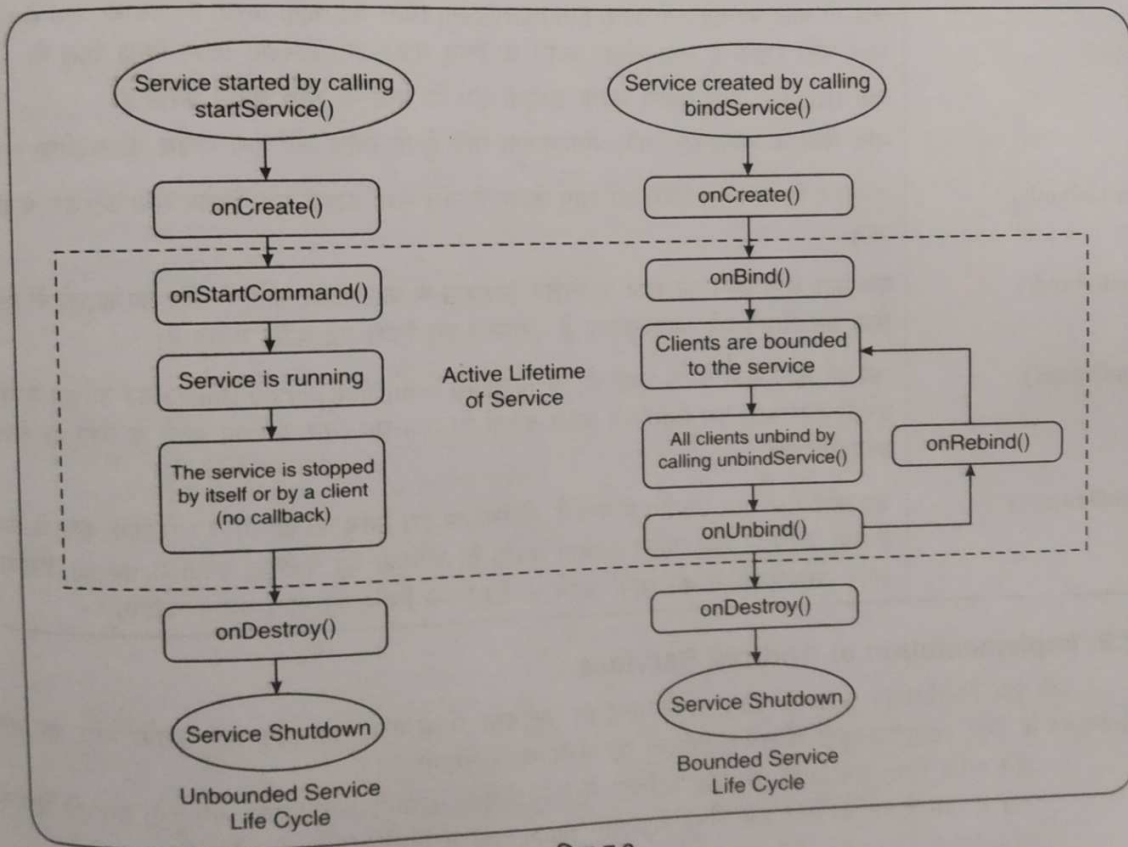


चित्र 7.8

1. Started Service (Unbounded Service)—इस मार्ग का पालन करके, एक सेवा शुरू होगी जब एक आवेदन घटक `startService()` विधि कहता है। एक बार शुरू होने के बाद, सेवा पृष्ठभूमि में लगातार चल सकती है, भले ही घटक नष्ट हो जाए जो सेवा की शुरुआत के लिए जिम्मेदार था। सेवा के निष्पादन को रोकने के लिए दो विकल्प उपलब्ध हैं—

- `stopService()` विधि को कॉल करके,
- सेवा `stopService()` विधि का उपयोग करके खुद को रोक सकती है।

2. Bounded Service—इसे क्लाइंट-सर्वर इंटरफेस में सर्वर के रूप में माना जा सकता है। इस रास्ते का पालन करके, एन्ड्रॉइड एप्लिकेशन घटक सेवा के लिए अनुरोध भेज सकते हैं और परिणाम प्राप्त कर सकते हैं। एक सेवा को तब बाध्य कहा जाता है जब कोई आवेदन घटक स्वयं को `bindService()` विधि कहकर सेवा से बांधता है। इस सेवा के निष्पादन को रोकने के लिए, सभी घटकों को `unbindService()` विधि का उपयोग करके सेवा से खुद को अनबाइंड करना चाहिए।



चित्र 7.9

पृष्ठभूमि में डाउनलोडिंग कार्य करने के लिए, `startService()` विधि को बुलाया जाएगा। जबकि डाउनलोड प्रगति के बारे में जानकारी प्राप्त करने के लिए और प्रक्रिया को थामने या फिर से शुरू करने के लिए, जबकि आवेदन अभी भी पृष्ठभूमि में है, सेवा एक घटक है जो इन कार्यों को कर सकते हैं के साथ घिरा होना चाहिए।

Fundamentals of Android Services

एक सामान्य वर्ग के माध्यम से उपयोगकर्ता-परिभाषित सेवा बनाई जा सकती है जो class Service का विस्तार कर रही है। इसके अलावा, अनुप्रयोगों पर सेवा के संचालन को पूरा करने के लिए, कुछ कॉलबैक तरीके हैं जिन्हें अधिष्ठा करने की आवश्यकता है। एन्ड्रॉइड सेवाओं के कुछ महत्वपूर्ण तरीके निम्नलिखित हैं—

Methods	Description
onStartCommand()	एन्ड्रॉइड सेवा इस विधि को कॉल करती है जब एक घटक (उदाहरण के लिए activity) startService() का उपयोग करके सेवा शुरू करने का अनुरोध करती है एक बार सेवा शुरू हो जाने के बाद, इसे स्पष्ट रूप से stopService() या stopSelf() तरीकों का उपयोग करके रोक जा सकता है।
onBind()	यह विधि एन्ड्रॉइड सेवा में लागू करने के लिए अनिवार्य है और इसे लागू किया गया है। जब भी कोई एप्लिकेशन घटक bindService() विधि को कॉल करता है। तब यह एक सेवा के साथ बांधे रखता है तथा संवाद करने के लिए यूजर-इंटरफेस भी प्रदान किया जाता है। एक IBinder वस्तु वापस करके प्रभावी ढंग से सेवा के साथ प्रदान करता है। यदि सेवा के बाध्यकारी की आवश्यकता नहीं है तो विधि को शून्य वापस करना होगा।
onUnbind()	एन्ड्रॉइड सिस्टम इस विधि को लागू करता है जब सभी ग्राहक एक विशेष सेवा इंटरफेस से कट जाए।
onRebind()	एक बार सभी ग्राहकों को सेवा के विशेष इंटरफेस से काट दिया जाता है और नए ग्राहकों के साथ सेवा को जोड़ने की आवश्यकता है, सिस्टम इस विधि को कॉल करता है।
onCreate()	जब भी कोई सेवा बनाई जाती है, तो onStartCommand() or onBind() का उपयोग करके, एन्ड्रॉइड सिस्टम इस विधि को कॉल करता है। एक बार-बार सेटअप करने के लिए यह विधि आवश्यक है।
onDestroy()	जब कोई सेवा अब उपयोग में नहीं है, तो सिस्टम इस विधि को अंतिम रूप से साफ कॉल के रूप में नष्ट करने से ठीक पहले आह्वान करता है। सेवाओं को पंजीकृत श्रोताओं, thread, रिसोर्स आदि जैसे संसाधनों को साफ करने के लिए इस विधि को लागू करना चाहिए।

7.3. Implementation of Android Services

अब हम निम्नलिखित कोड के साथ अपने एन्ड्रॉइड स्टूडियो में सेवाओं को लागू करें। इसके साथ, हम सेवा कार्यान्वयन के लिए आवश्यक कुछ कॉलबैक विधियों का कार्यान्वयन देखेंगे।

1. अब इसके लिए, हम अपने एन्ड्रॉइड स्टूडियो में एक प्रोजेक्ट बनाएंगे और इसका नाम देंगे। हम इसे सेवा के रूप में नाम दे रहे हैं। फिर हम अपने activity_main.xml में निम्नलिखित कोड लिखेंगे—

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/textView2"
```

```

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginLeft="100dp"
        android:text="Myapp "
        android:textColor="@color/colorPrimaryDark"
        android:textSize="50dp" />
    <Button
        android:id="@+id/btnStart"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="50dp"
        android:layout_marginTop="50dp"
        android:onClick="startService"
        android:text="Start Service" />
    <Button
        android:id="@+id/btnstop"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="50dp"
        android:layout_marginTop="50dp"
        android:onClick="stopService"
        android:text="Stop Service" />
</LinearLayout>

```

2. इसके बाद, हम MainActivity.java में निम्नलिखित कोड लिखेंगे—

```

package com.Myapp.Myapp_Services;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void startService(View view) {
        startService(new Intent(this, MyappService.class));
    }
    public void stopService(View view) {
        stopService(new Intent(this, MyappService.class));
    }
}

```


3. अब Manifest.xml फाइल खोलें। मैनिफेस्ट फाइल में निम्नलिखित कोड लिखा जाना है—

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
package="com.Myapp.Myapp_Services">
<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="com.Myapp.Myapp_Services.MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<service android:name="com.Myapp.Myapp_Services.MyappService" />
</application>
</manifest>
```

4. और अंत में, हम एक फाइल, एक जावा वर्ग बनाएंगे और इसका नाम देगे। हम इसे MyappService नाम दे रहे हैं। इस फाइल के बनने के बाद हम इसमें निम्नलिखित कोड लिखेंगे—

```
package com.Myapp.Myapp_Services;
import android.app.Service;
import android.content.Intent;
import android.media.MediaPlayer;
import android.os.IBinder;
import android.provider.Settings;
import android.widget.Toast;
public class MyappService extends Service {
    private MediaPlayer player;
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }
    @Override
    public void onCreate() {
        Toast.makeText(this, "Service was Created",
        Toast.LENGTH_LONG).show();
    }
    @Override
```



```

public int onStartCommand(Intent intent, int flags, int startId) {
    player = MediaPlayer.create(this,
        Settings.System.DEFAULT_RINGTONE_URI);
    player.setLooping(true);
    player.start();
    Toast.makeText(this, "Service Started",
        Toast.LENGTH_LONG).show();
    return START_STICKY;
}

@Override
public void onDestroy() {
    super.onDestroy();
    player.stop();
    Toast.makeText(this, "Service Stopped",
        Toast.LENGTH_LONG).show();
}
}

```

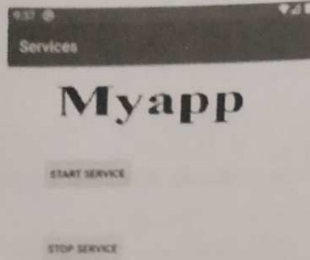
5. क्रमशः सभी कोड लिखने के बाद, हम अब अपना ऐप चलाएंगे। निम्नलिखित हमारे उत्पादन के रूप में हमें दिखाया जाएगा।



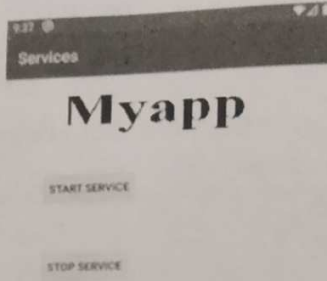
चित्र 7.10

यह हमारा application है जिसमें हम अपनी सेवा शुरू और रोक सकते हैं। (चित्र 7.10)
 अब हमने स्टार्ट सर्विस पर क्लिक किया है, इसलिए सर्विस पहले बनाई गई। (चित्र 7.11)
 सेवा बनने के बाद इसका execution शुरू हो जाता है। (चित्र 7.12)

उसके बाद जब भी हम उन्हें रोकना चाहें तो अपनी सेवा रोक सकते हैं। जैसा कि हम नीचे देख सकते हैं, सेवा बंद कर दिया है (चित्र 7.13)।



चित्र 7.11



चित्र 7.12



चित्र 7.13

प्रश्नावली

1. Broadcast receivers से आप क्या समझते हैं? इसके प्रकारों सहित व्याख्या करें।
2. Broadcast receivers का पंजीकरण किस प्रकार किया जाता है?
3. Android में Notifications से आप क्या समझते हैं?
4. Android में services तथा इसके प्रकारों को विस्तार से समझाइये।
5. Android services के lifecycle को Explain कीजिये।



भंडारण और सामग्री प्रदाता (Storage and Content Provider)

Supported storage in android (internal memory, external memory, shared preferences and network), SQLite introduction, CRUD operations in SQLite database (cursor, content values etc), basics of content provider.

8.1. Supported Storage in Android

एंड्रॉइड आपके लिए एप्लिकेशन डेटा को save करने के लिए कई विकल्प प्रदान करता है। आपके द्वारा चुना गया solution आपकी specific needs पर निर्भर करता है, जैसे कि डेटा आपके application के लिए private होना चाहिए या अन्य application (और user) के द्वारा accessible होना चाहिए और आपके डेटा को कितनी space की आवश्यकता होती है।

आपके डेटा स्टोरेज विकल्प निम्नलिखित हैं—

1. Shared preferences
Store private primitive data in key-value pairs.
2. Internal storage
Store private data on the device memory.
3. External storage
Store public data on the shared external storage
4. SQLite databases
Store structured data in a private database.
5. Network connection
Store data on the web with your own network server.

एंड्रॉइड आपके लिए एक content provider के साथ other applications के लिए अपने private data को भी expose करने का एक तरीका प्रदान करता है। एक content provider एक optional component है जो आपके एप्लिकेशन डेटा तक पढ़ने/लिखने को expose करता है।

8.1.1. Shared Preferences

SharedPreferencesClass एक framework प्रदान करता है जो आपको primitive data types के persistent key-value pairs को save और retrieve करने की अनुमति देता है। आप किसी भी primitive data को save के लिए shared preferences का उपयोग कर सकते हैं—boolean, float, int, long और string। यह डेटा पूरे user sessions में बना रहेगा (भले ही आपका application kill हो गया हो)।

User Preferences

Shared preferences user preferences को save कराने के लिए नहीं है, जैसे कि user ने किस रिंगटोन को चुना है। यदि आप अपने एप्लिकेशन के लिए user preferences बनाने में रुचि रखते हैं, तो PreferenceActivity देखें, जो आपके लिए user preferences बनाने के लिए एक activity framework प्रदान करती है, जो shared प्राथमिकताओं का उपयोग करके स्वचालित रूप से बनी रहेगी।

अपने application के लिए shared preferences object प्राप्त करने के लिए दो methods हैं—

- **get shared preference ()**—इसका उपयोग करें यदि आपको name से पहचानी जाने वाली multiple preferences files की आवश्यकता है, जिसे आप parameter के साथ specify करते हैं।
- **get Preferences ()**—यदि आपको अपनी activity के लिए केवल एक preferences file की आवश्यकता है, तो इसका उपयोग करें। क्योंकि यह आपकी activity के लिए only preferences फाइल होगी।

Value write करने के लिए—

1. Call **edit()** to get a **SharedPreferences Editor**.
2. Add values with methods such as **putBoolean()** and **putString()**.
3. Commit the new values with **commit()**.

Values Read करने के लिए—

1. use **SharedPreferences** methods such as **getBoolean()** and **getString()**. Here is an example that saves a preference for silent keypress mode in a calculator :

```
public class Calc extends Activity {
    public static final String PREFS_NAME = "MyPrefsFile";
    @Override
    protected void onCreate(Bundle state) {
        super.onCreate(state);
        // Restore preferences
        SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
        boolean silent = settings.getBoolean("silentMode", false);
        setSilent(silent);
    }
    @Override
    protected void onStop() {
        super.onStop();
        // We need an Editor object to make preference changes.
        // All objects are from android.context.Context
        SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
        SharedPreferences.Editor editor = settings.edit();
        editor.putBoolean("silentMode", mSilentMode);
        // Commit the edits!
        editor.commit();
    }
}
```

8.1.2. Internal Storage

आप डिवाइस के internal storage पर सीधे फाइलें save कर सकते हैं। डिफॉल्ट रूप से, internal storage में save की गई फाइलें आपके एप्लिकेशन के लिए private हैं और अन्य एप्लिकेशन उन्हें एक्सेस नहीं कर सकते हैं न ही user। जब user आपके एप्लिकेशन को अनइंस्टॉल करता है, तो ये फाइलें हट जाती हैं।

Internal storage में private फाइल create और write करने के लिए—

1. फाइल के नाम और ऑपरेटिंग मोड के साथ OpenFileOutput () फंक्शन को कॉल करें। यह एक FileOutputStream देता है।
2. फाइल को write करने के लिए write() फंक्शन को कॉल करें।
3. Close करने के लिए close() फंक्शन को कॉल करें।

For example :

```
String FILENAME = "hello_file";
String string = "hello world!";
FileOutputStream fos = openFileOutput(FILENAME,
Context.MODE_PRIVATE);
fos.write(string.getBytes());
fos.close();
```

MODE_PRIVATE फाइल create करता है या उसी नाम की फाइल को replace कर देगा और इस file को आपके application के लिए private कर देगा। अन्य मोड MODE_APPEND, MODE_WORLD_READABLE, और MODE_WORLD_WRITEABLE हैं।

Internal storage से फाइल read करने के लिए—

1. जो भी फाइल read करनी है उसका नाम openFileInput() pass करते हुए कॉल करें।
2. फाइल को read करने के लिए read() फंक्शन को कॉल करें।
3. Close करने के लिए close() फंक्शन को कॉल करें।

Other useful methods

(i) getFilesDir()

फाइल सिस्टम directory के लिए absolute path प्राप्त करता है जहाँ आपकी internal फाइलें save की जाती हैं।

(ii) getDir()

आपके internal storage स्थान के भीतर (या एक मौजूदा) directory बनाता है।

(iii) deleteFile()

Internal storage पर save की गई फाइल delete करता है।

(iv) FileList()

वर्तमान में आपके एप्लिकेशन द्वारा save की गई फाइलों की array of file देता है।

8.1.3. External Storage

हर Android-compatible डिवाइस एक shared external storage का समर्थन करता है जिसका उपयोग आप फाइलों को save के लिए कर सकते हैं। यह एक removable storage media जैसे SD कार्ड या internal non-removable storage हो सकता है। external storage में save की गई फाइलें world-readable हैं और user द्वारा परोक्षित की जा सकती हैं जब भी वे कंप्यूटर पर फाइलों को transfer करने के लिए USB मास स्टोरेज को enable करते हैं।

Saving files on external storage

यदि आप उन फाइलों को save करना चाहते हैं जो आपके एप्लिकेशन के लिए विशिष्ट नहीं हैं और जिन्हें आपके एप्लिकेशन को uninstall होने पर हटाया नहीं जाना चाहिए, तो उन्हें external storage पर public directories में से एक में सेवें। ये directory, external storage, जैसे संगीत/चित्र/रिंगटोन और अन्य के root में होती हैं।

External storage का उपयोग उन फाइलों के लिए बहुत अच्छा है जिन्हें आप अन्य एप्लिकेशन के साथ साझा करना चाहते हैं या उपयोगकर्ता को कम्प्यूटर के साथ उपयोग करने की अनुमति देते हैं। आप दो भिन्न प्रकार की फाइलों को सहेज सकते हैं—

(i) **Public files**—ऐसी फाइलें जो अन्य ऐप्स और उपयोगकर्ता के लिए स्वतंत्र रूप से उपलब्ध होनी चाहिए। जब उपयोगकर्ता आपके ऐप को अनइंस्टॉल करता है, तो भी ये फाइलें उपयोगकर्ता के लिए उपलब्ध होनी चाहिए। उदाहरण के लिए, आपके ऐप या अन्य डाउनलोड की गई फाइलों द्वारा फैचर की गई तस्वीरों को सार्वजनिक फाइलों के रूप में सहेजा जाना चाहिए।

(ii) **Private files**—फाइलें जो आपके ऐप से सम्बन्धित हैं और जब उपयोगकर्ता आपके ऐप को अनइंस्टॉल करता है तो उसे हटा दिया जाएगा। यद्यपि ये फाइलें उपयोगकर्ता और अन्य ऐप्स द्वारा तकनीकी रूप से accessible हैं, क्योंकि वे external storage पर हैं, वे आपके ऐप के बाहर उपयोगकर्ता को मूल्य प्रदान नहीं करते हैं।

Save to a public directory

```
public File getPublicAlbumStorageDir(String albumName) {
    // Get the directory for the user's public pictures directory.
    File file = new File(Environment.getExternalStoragePublicDirectory(
        Environment.DIRECTORY_PICTURES), albumName);
    if (!file.mkdirs()) {
        Log.e(LOG_TAG, "Directory not created");
    }
    return file;
}
```

यदि आप external storage पर public फाइलों को save करना चाहते हैं, तो external storage पर उपयोगकर्ता का प्रतिनिधित्व करने वाली फाइल प्राप्त करने के लिए `getExternalStoragePublicDirectory()` विधि का उपयोग करें। method एक argument लेती है जो उस फाइल के प्रकार को निर्दिष्ट करता है जिसे आप सहेजना चाहते हैं ताकि वे logically रूप से अन्य public फाइलों के साथ व्यवस्थित हो सकें, जैसे कि `DIRECTORY_MUSIC` या `DIRECTORY_PICTURES`।

Request external storage permission

Public external storage में लिखने के लिए, आपको अपनी manifest file में `WRITE_EXTERNAL_STORAGE` अनुमति का अनुरोध करना होगा—

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

यदि आपके ऐप को केवल external storage (लेकिन इसे लिखना नहीं) में पढ़ना है, तो आपको `READ_EXTERNAL_STORAGE` permission declare करने की आवश्यकता है—

```
<manifest ...>
    <uses-permission
        android:name="android.permission.READ_EXTERNAL_STORAGE" />
    ...
</manifest>
```

Checking media availability

इससे पहले कि आप external storage के साथ कोई काम करते हैं, आपको हमेशा यह जांचने के लिए की मीडिया उपलब्ध है या नहीं `getExternalStorageState()` को कॉल करना चाहिए। मीडिया को कंप्यूटर पर mounted किया जा सकता है, missing, केवल पढ़ने के लिए, या किसी other state में। उदाहरण के लिए, यहाँ आप उपलब्धता की जाँच कर सकते हैं—

```
boolean mExternalStorageAvailable = false;
boolean mExternalStorageWriteable = false;
String state = Environment.getExternalStorageState();
if (Environment.MEDIA_MOUNTED.equals(state)) {
    // We can read and write the media
    mExternalStorageAvailable = mExternalStorageWriteable = true;
} else if (Environment.MEDIA_MOUNTED_READ_ONLY.equals(state)) {
    // We can only read the media
    mExternalStorageAvailable = true;
    mExternalStorageWriteable = false;
} else {
    // Something else is wrong. It may be one of many other states,
    // but all we need
    // to know is we can neither read nor write
    mExternalStorageAvailable = mExternalStorageWriteable = false;
}
```

यह उदाहरण जाँचता है कि क्या external storage पढ़ने और लिखने के लिए उपलब्ध है। `getExternalStorageState()` method other state को लौटाती है जिन्हें आप जाँचना चाहते हैं, जैसे कि मीडिया shared किया जा रहा है (कंप्यूटर से जुड़ा हुआ है), पूरी तरह से missing है, removed badly, आदि। आप इनका उपयोग user को सूचित करने के लिए कर सकते हैं जब आपके एप्लिकेशन को मीडिया तक पहुँचने की आवश्यकता होती है।

Accessing files on external storage

`getExternalFilesDir()` का उपयोग करके फाइल खोलें जो external storage directory को represent करती है जहाँ आपको अपनी फाइलों को save करना चाहिए। यह method एक type parameter लेती है, जो आपके द्वारा वांछित subdirectory को निर्दिष्ट करता है, जैसे कि DIRECTORY MUSIC और DIRECTORY RINGTONES। यदि आवश्यक हो तो यह method उपयुक्त directory बनाएगी। directory के प्रकार को निर्दिष्ट करके आप यह सुनिश्चित करते हैं कि एन्ड्रॉइड का मीडिया स्कैनर आपकी फाइलों को सिस्टम में ठीक से वर्गीकृत करेगा उदाहरण के लिए, रिंगटोन को रिंगटोन के रूप में पहचाना जाता है, संगीत के रूप नहीं। यदि उपयोगकर्ता आपके एप्लिकेशन को अनइंस्टॉल करता है तो यह directory और उसकी सभी सामग्री हटा दी जाएगी।

Using Databases (SQLite)

Android SQLite डेटाबेस के लिए पूर्ण समर्थन प्रदान करता है। आपके द्वारा बनाया गया कोई भी डेटाबेस application में किसी भी class के नाम से accessible होगा, लेकिन application के बाहर accessible नहीं होगा।

एक नया SQLite डेटाबेस बनाने के लिए `SQLiteOpenHelper` का subclass और `onCreate()` विधि को ओवरराइड करना है, जिसमें आप डेटाबेस में table बनाने के लिए SQLite कमांड execute कर सकते हैं।

उदाहरण के लिए—

```
public class DictionaryOpenHelper extends SQLiteOpenHelper {
    private static final int DATABASE_VERSION = 2;
    private static final String DICTIONARY_TABLE_NAME = "dictionary";
    private static final String DICTIONARY_TABLE_CREATE =
        "CREATE TABLE " + DICTIONARY_TABLE_NAME + " (" +
        KEY_WORD + " TEXT, " +
        KEY_DEFINITION + " TEXT);";
    DictionaryOpenHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(DICTIONARY_TABLE_CREATE);
    }
}
```

डेटाबेस से लिखने और पढ़ने के लिए, क्रमशः `getWritableDatabase()` और `getReadableDatabase()` कॉल करें। ये दोनों `SQLiteDatabase` ऑब्जेक्ट लौटाते हैं जो डेटाबेस का प्रतिनिधित्व करता है और `SQLite` ऑपरेशन के लिए `method` प्रदान करता है।

आप `SQLiteDatabaseQuery()` विधियों का उपयोग करके `SQLite queries` को `execute` कर सकते हैं, जो विभिन्न `query parameters` को स्वीकार करते हैं, जैसे कि तालिका से `query`, `projection`, `selection`, `columns`, `grouping` और `others!` `complex queries` के लिए, जैसे कि `column aliases` की आवश्यकता होती है, आपको `SQLiteQueryBuilder` का उपयोग करना चाहिए, जो प्रश्नों के निर्माण के लिए कई `convenient method` प्रदान करता है।

प्रत्येक `SQLite query` एक `Cursor` लौटाएगा जो `query` द्वारा मिली सभी `rows` को इंगित करता है। `Cursor` वह `mechanism` होता है जिसके साथ आप डेटाबेस `query` से `result` नेविगेट कर सकते हैं और `rows` और `column` पढ़ सकते हैं।

Database debugging—`Android SDK` में एक `sqlite3 database` टूल शामिल है जो आपको `table contents` `browse` करने, `SQL कमांड` चलाने और `SQLite डेटाबेस` पर अन्य उपयोगी कार्य करने की अनुमति देता है।

8.1.4. Using a Network Connection

आप अपनी स्वयं की `web-based services` पर डेटा `store` करने और `retrieve` करने के लिए `नेटवर्क` का उपयोग कर सकते हैं। `नेटवर्क संचालन` करने के लिए निम्न पैकेजों में `class` का उपयोग करें—

(i) `java.net.*`

(ii) `android.net.*`

8.2. SQLite Introduction

`SQLite` एक ओपनसोर्स `SQL डेटाबेस` है जो डेटा को एक डिवाइस पर एक `text file` में `store` करता है। `एन्ड्रॉइड built in SQLite डेटाबेस implementation` के साथ आता है।

SQLite के Features

1. `Transactions ACID` का पालन करता है। *i.e.*, `atomicity`, `consistency`, `isolation` and `durability`.
2. कॉन्फिगरेशन प्रक्रिया बहुत आसान है, कोई सेटअप या `administration` की आवश्यकता नहीं है।

3. SQL के सभी फीचर्स इसमें कुछ अतिरिक्त फीचर्स जैसे कि partial इंडेक्स, इंडेक्स ऑन एक्सप्रेशन, JSON और कॉमन टेबल एक्सप्रेशन के साथ लागू किए गए हैं।
4. कभी-कभी यह डायरेक्ट फाइल सिस्टम I/O से तेज होता है।
5. यह टेराबाइट आकार के डेटाबेस और गीगाबाइट आकार के तार और ब्लूब्स का समर्थन करता है।
6. लगभग सभी OS Android, BSD, iOS, Linux, Mac, Solaris, VxWorks और Windows (Win32, WinCE, आदि) जैसे SQLite को सपोर्ट करते हैं।
7. पूरा डेटाबेस एक सिंगल-प्लेटफॉर्म डिस्क फाइल में संग्रहीत किया जा सकता है।

SQLite के अनुप्रयोग

1. अपने small code और मेमोरी के कुशल उपयोग के कारण, यह सेलफोन, पीडीए, एमपी प्लेयर 3, सेटटॉप बॉक्स और अन्य इलेक्ट्रॉनिक गैजेट्स में डेटाबेस इंजन के लिए लोकप्रिय विकल्प है।
2. यह XML, JSON, CSV या कुछ proprietary format लिखने के लिए एक विकल्प के रूप में प्रयोग किया जाता है।
3. चूंकि इसमें कॉन्फिगरेशन simple है और आसानी से फाइल को एक साधारण डिस्क फाइल में संग्रहीत किया जाता है, इसलिए इसे medium sized की वेबसाइटों के लिए डेटाबेस के रूप में उपयोग किया जा सकता है।
4. यह third-party tools के माध्यम से faster और accessible है, इसलिए विभिन्न सॉफ्टवेयर प्लेटफॉर्म में इसका शानदार अनुप्रयोग है।

Disadvantages of SQLite

1. इसका उपयोग केवल वहाँ किया जाता है जहाँ मध्यम से कम traffic requests होता है।
2. डेटाबेस का size restricted है।

SQLite Commands

(i) Creating Table :

```
CREATE TABLE STUDENT (
  Enroll_NO INT PRIMARY KEY NOT NULL,
  NAME TEXT NOT NULL,
  AGE INT NOT NULL,
  ADDRESS CHAR(50),
);
```

(ii) Insert Command:

```
INSERT INTO STUDENT (ID, NAME, AGE, ADDRESS)
VALUES (2021001, 'Mayank', 32, 'Moradabad');
```

(iii) Drop Table

```
Drop Table Student;
```

	SQL	SQLite
(i)	जिसका उपयोग रिलेशनल डेटाबेस सिस्टम को query करने के लिए किया जाता है। यह C में लिखा गया है।	SQLite एक Relational database management system है जो ANSI-C में लिखा गया है।
(ii)	SQL मानक है जो specifies करता है कि relational schema कैसे बनाया जाता है, डेटा insert या update किया जाये।	SQLite फाइल आधारित है। यह अन्य-SQL डेटाबेस से अलग है क्योंकि अधिकांश अन्य SQL डेटाबेस के विपरीत, SQLite में अलग सर्वर प्रक्रिया नहीं है।

(iii)	SQL के मुख्य घटक डेटा डेफिनिशन लैंग्वेज (DML), डेटा कंट्रोल लैंग्वेज (DCL) हैं।	SQLite SQL की कई विशेषताओं का समर्थन करता है और इसमें उच्च प्रदर्शन होता है लेकिन store procedure का समर्थन नहीं करता है।
(iv)	SQL structured query language है जो MySQL, Oracle, Microsoft SQL server, IBM DB2 आदि डेटाबेस के साथ प्रयोग की जाती है।	SQLite पोर्टेबल डेटाबेस resource है। यह उस डेटाबेस को एक्सेस करने के लिए उपयोग की जाने वाली प्रोग्रामिंग भाषा में एक extension प्राप्त कर सकता है।
(v)	एक conventional SQL डेटाबेस को कनेक्ट करने और बहुत सारी कार्यक्षमता प्रदान करने के लिए OracleDB जैसी सेवा के रूप में चलने की आवश्यकता है।	SQLite डेटाबेस सिस्टम ऐसी कार्यक्षमता प्रदान नहीं करता है।
(vi)	SQL query भाषा है जो अन्य SQL डेटाबेस द्वारा उपयोग की जाती है। यह स्वयं डेटाबेस नहीं है।	SQLite रिलेशनल डेटाबेस मैनेजमेंट सिस्टम है जो SQL का उपयोग करता है।

SQLite in Android

SQLite एक open source RDBMS है जिसका प्रयोग column और row के रूप में store डेटाबेस पर संचालन करने के लिए किया जाता है।

1. SQLite निम्नलिखित तीन प्रकार के डेटा का समर्थन करता है—

(i) **Text type** : to store strings or character type data.

(ii) **Integer type** : to store the integer data type.

(iii) **Real type** : to store long values.

2. एन्ड्रॉइड एप्लीकेशन में SQLite का उपयोग करने के लिए, हम पैकेज android.database.SQLite का उपयोग करते हैं। इस पैकेज में SQLite का उपयोग करने के लिए सभी API हैं।

3. **SQLiteOpenHelper** : SQLiteOpenHelper एक class है जो डेटाबेस बनाने और इसे manage करने के लिए उपयोगी है। SQLiteOpenHelper class के दो constructors हैं—

(i) **SQLiteOpenHelper (Context context, String name, SQLiteDatabase.CursorFactory factory, int version_no)**—यह डेटाबेस create, open और manage के लिए ऑब्जेक्ट बनाता है।

(ii) **SQLiteOpenHelper (Context context, String name, SQLiteDatabase.CursorFactory factory, int version_no, DatabaseErrorHandler errorHandler)**—इसका उद्देश्य डेटाबेस को create, open और manage करने के साथ error handler को specifies करता है।

4. **SQLiteDatabase** एक ऐसा class है जिसमें ऑपरेशन करने के तरीके हैं जैसे create, update, delete आदि।

8.3. CRUD Operations in SQLite Database

CRUD उन basic operations के लिए एक abbreviation है जो हम किसी भी डेटाबेस में perform करते हैं। ये बेसिक ऑपरेशन—

(i) Create

(ii) Read

- (iii) Update
- (iv) Delete

8.3.1. Creating New Project

सबसे पहले, हम android एप्लीकेशन बनाएंगे। फिर, हम देखेंगे कि एप्लीकेशन में SQLiteDatabase का उपयोग करके CRUD ऑपरेशन कैसे करें।

- (i) Android स्टूडियो खोलें।
- (ii) Start a new Android Studio Project पर क्लिक करें फिर अपना प्रोजेक्ट चुनें और empty activity चुनें।
- (iii) एप्लीकेशन का नाम लिखें।
- (iv) Java का चयन करें।
- (v) एक बार जब आपका प्रोजेक्ट लोड हो जाता है, तो हम इस पर काम करना शुरू कर सकते हैं।

8.3.2. CRUD and other Database Queries

(a) The Database Structure

Employee		
Id	Int	PK
Name	Varchar (200)	
Department	Varchar (200)	
Joining Data	Data Time	
Salary	Double	

(b) Creating the table

```
CREATE TABLE employees (
    id INTEGER NOT NULL CONSTRAINT employees_pk PRIMARY KEY
    AUTOINCREMENT,
    name varchar(200) NOT NULL,
    department varchar(200) NOT NULL,
    joiningdate datetime NOT NULL,
    salary double NOT NULL
);
```

(c) Insert a new record

```
INSERT INTO employees
(name, department, joiningdate, salary)
VALUES
('Mayank', 'Computer', '2021-3-01 11:05:17', '50000');
```

(d) Reading all existing records

```
SELECT * FROM employees;
```

(e) Reading specific record

```
SELECT * FROM employees WHERE id = 1;
```


(f) Updating a record

```
UPDATE employees
SET
name = 'Navneet',
department = 'Computer',
salary = '45000'
WHERE id = 1;
```

(g) Deleting a record

```
DELETE FROM employees WHERE id = 1;
```

8.3.3. User Interface Design

हमें एक ऐप बनाने की आवश्यकता है जो above queries को user के साथ interact करेगी।

(a) Activity_Main.Xml

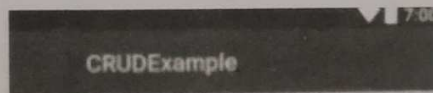
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="CRUDEExample.MainActivity">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerVertical="true"
        android:orientation="vertical"
        android:padding="16dp">
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="12dp"
            android:text="Add a new Employee"
            android:textAlignment="center"
            android:textAppearance="@style/Base.TextAppearance.AppCompat.Large"/>
        <EditText
            android:id="@+id/editTextName"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Enter Employee Name" />
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:paddingLeft="6dp"
            android:text="Select Department" />
```



```

<Spinner
    android:id="@+id/spinnerDepartment"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:entries="@array/departments"/>
<EditText
    android:id="@+id/editTextSalary"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:entries="@array/departments"/>
<EditText
    android:id="@+id/editTextSalary"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:digits="0123456789"
    android:hint="Enter Employee Salary"
    android:inputType="number"/>
<Button
    android:id="@+id/buttonAddEmployee"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Add Employee" />

```



Add a new Employee

Enter Employee Name

Select Department

Computer

Enter Employee Salary

ADD EMPLOYEE

View Employees



चित्र 8.1

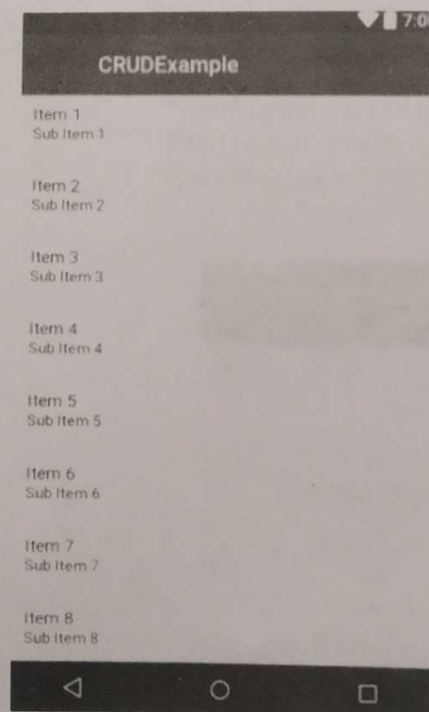
```

<TextView
    android:id="@+id/textViewEmployees"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="16dp"
    android:text="View Employees"
    android:textAlignment="center"
    android:textAppearance="@style/Base.TextAppearance.AppCompat.Me
dium"
    android:textStyle="bold" />
</LinearLayout>
</RelativeLayout>

```

8.3.4. Fetching All the Employees

इस स्क्रीन के लिए हम निम्नलिखित डिजाइन का उपयोग कर सकते हैं—



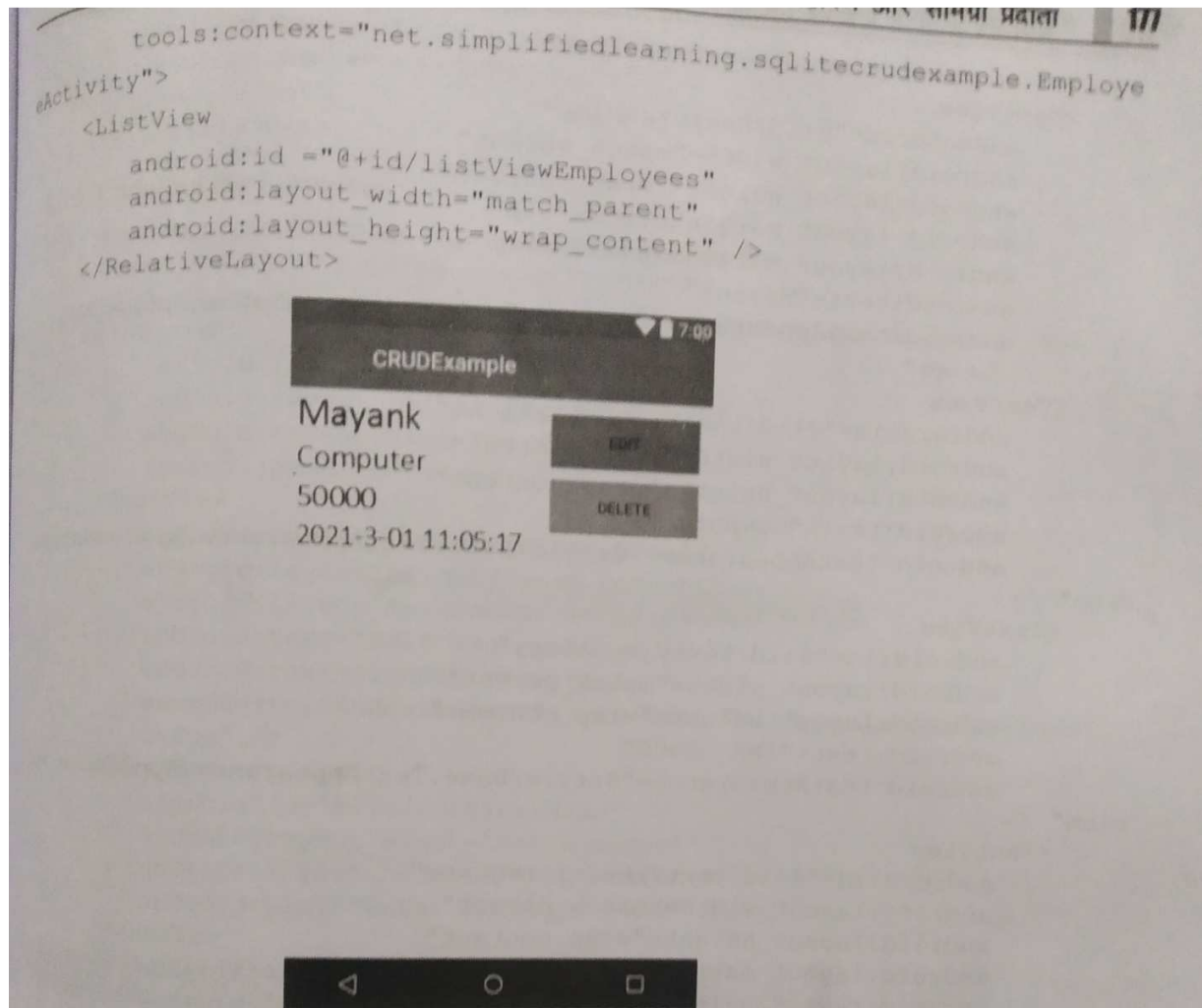
चित्र 8.2

इस स्क्रीन में केवल एक सूची दृश्य शामिल है। उपरोक्त स्क्रीन के लिए xml नीचे है—

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

```



चित्र 8.3

ListView आइटम के लिए हमें एक और लेआउट की भी आवश्यकता है, इसलिए list_layout_employee.xml नाम की एक और लेआउट फाइल बनाएँ और निम्नलिखित xml कोड लिखें। नीचे दिया गया xml कोड निम्नलिखित लेआउट उत्पन्न करेगा। फाइल का नाम list_layout_employee.xml है और निम्नलिखित xml कोड लिखें।

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res
/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:padding="8dp">
<LinearLayout
    android:layout_width="230dp"
    android:layout_height="wrap_content"

```

```

        android:orientation="vertical">
<TextView
    android:id="@+id/textViewName"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="5dp"
    android:layout_marginTop="10dp"
    android:text="Mayank"
    android:textAppearance="@style/Base.TextAppearance.AppCompat
    .Large" />
<TextView
    android:id="@+id/textViewDepartment"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Computer"
    android:textAppearance="@style/Base.TextAppearance.AppCompat.Me
dium" />
<TextView
    android:id="@+id/textViewSalary"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="INR 50000"
    android:textAppearance="@style/Base.TextAppearance.AppCompat.Me
dium" />
<TextView
    android:id="@+id/textViewJoiningDate"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:text="2021-3-01 11:05:17" />
</LinearLayout>
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
<Button
    android:id="@+id/buttonEditEmployee"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="5dp"
    android:background="@color/colorPrimary"
    android:text="Edit" />
<Button
    android:id="@+id/buttonDeleteEmployee"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="5dp"

```



```

        android:background="@color/colorAccent"
        android:text="Delete" />
    </LinearLayout>
</LinearLayout>

```

8.3.5. Updating the Employee

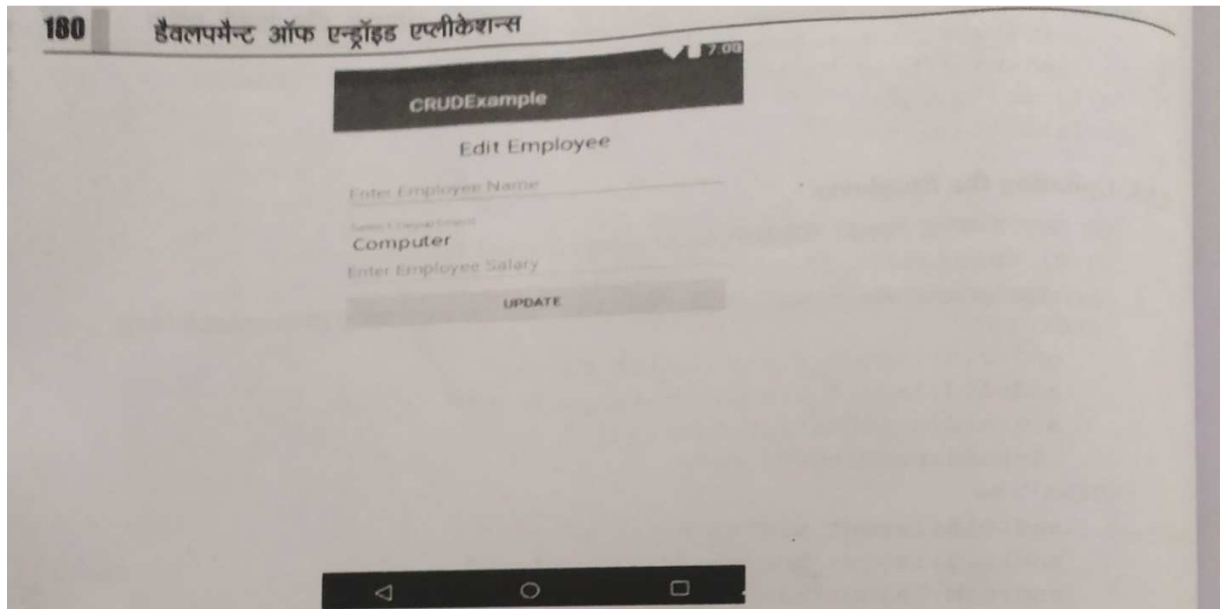
नया लेआउट फाइल जिसका नाम Dial_update_employee.xml है।

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res
/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="12dp"
        android:text="Edit Employee"
        android:textAlignment="center"
        android:textAppearance="@style/Base.TextAppearance.AppCompat.
        Large" />
    <EditText
        android:id="@+id/editTextName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Employee Name" />
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:paddingLeft="6dp"
        android:text="Select Department" />
    <EditText
        android:id="@+id/editTextSalary"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:digits="0123456789"
        android:hint="Enter Employee Salary"
        android:inputType="number" />
    <Button
        android:id="@+id/buttonUpdateEmployee"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Update" />
</LinearLayout>

```

यह कोड अग्रलिखित डिजाइन उत्पन्न करेगा।



चित्र 8.5

8.3.6. Coding the Application

Adding an Employee

```
package net.simplifiedlearning.sqlitecrudexample;
import android.content.Intent;
import android.database.sqlite.SQLiteDatabase;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
    public static final String DATABASE_NAME = "myemployeedatabase";
    TextView textViewViewEmployees;
    EditText editTextName, editTextSalary;
    Spinner spinnerDepartment;
    SQLiteDatabase mDatabase;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        textViewViewEmployees = (TextView)
        findViewById(R.id.textViewViewEmployees);
        editTextName = (EditText) findViewById(R.id.editTextName);
        editTextSalary = (EditText) findViewById(R.id.editTextSalary);
    }
}
```

```

        spinnerDepartment = (Spinner)
        findViewById(R.id.spinnerDepartment);
        findViewById(R.id.buttonAddEmployee).setOnClickListener(this);
        textViewViewEmployees.setOnClickListener(this);
        //creating a database
        mDatabase = openOrCreateDatabase(DATABASE_NAME, MODE_PRIVATE,
        null);
    }
    //this method will validate the name and salary
    //dept does not need validation as it is a spinner and it cannot be
    empty
    private boolean inputsAreCorrect(String name, String salary) {
        if (name.isEmpty()) {
            editTextName.setError("Please enter a name");
            editTextName.requestFocus();
            return false;
        }
        if (salary.isEmpty() || Integer.parseInt(salary) <= 0) {
            editTextSalary.setError("Please enter salary");
            editTextSalary.requestFocus();
            return false;
        }
        return true;
    }
    //In this method we will do the create operation
    private void addEmployee() {
    }
    @Override
    public void onClick(View view) {
        switch (view.getId()) {
            case R.id.buttonAddEmployee:
                addEmployee();
                break;
            case R.id.textViewViewEmployees:
                startActivity(new Intent(this, EmployeeActivity.class));
                break;
        }
    }
}

```

उपरोक्त कोड में आप देखते हैं कि हमने एक विधि का उपयोग किया है। अब हम employee table बनाने के लिए एक विधि बनाएंगे। इसलिए createEmployeeTable () नाम का एक तरीका बनाएँ।

```

openOrCreateDatabase ().

```

```

    private void createEmployeeTable() {
        mDatabase.execSQL(
            "CREATE TABLE IF NOT EXISTS employees (\n" +

```

```

" id int NOT NULL CONSTRAINT employees_pk PRIMARY KEY,\n" +
" name varchar(200) NOT NULL,\n" +
" department varchar(200) NOT NULL,\n" +
" joiningdate datetime NOT NULL,\n" +
" salary double NOT NULL\n" +
");"
);
}

```

- अब इस विधि को केवल openOrCreateDatabase() विधि कहकर पुकारें। उपरोक्त विधि में हम अपने डेटाबेस टेबल को बनाने के लिए केवल मेथड execSQL() को call रहे हैं।
- execSQL() Method केवल table create, Record insert या update करने के लिए।
- अब तक हमारे पास डेटाबेस है, अब हमें employee को table में insert करने की आवश्यकता है और हमें इसे addEmployee() method के अंदर करने की आवश्यकता है।

```

private void addEmployee() {
    String name = editTextName.getText().toString().trim();
    String salary = editTextSalary.getText().toString().trim();
    String dept = spinnerDepartment.getSelectedItem().toString();
    //getting the current time for joining date
    Calendar cal = Calendar.getInstance();
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-mm-dd
hh:mm:ss");
    String joiningDate = sdf.format(cal.getTime());
}

```

Android Emulator - Nexus_5_API_26:5554




```
//validating the input
if (inputsAreCorrect(name, salary)) {
    String insertSQL = "INSERT INTO employees \n" +
        "(name, department, joiningdate, salary)\n" +
        "VALUES \n" +
        "(?, ?, ?, ?)";
    //using the same method execsql for inserting values
    //this time it has two parameters
    //first is the sql string and second is the parameters that is
    //to be binded with the query
    mDatabase.execSQL(insertSQL, new String[]{name, dept,
        joiningDate, salary});
    Toast.makeText(this, "Employee Added Successfully",
        Toast.LENGTH_SHORT).show();
}
}
```

अब आप क्रिएट ऑपरेशन के लिए अपने आवेदन का परीक्षण कर सकते हैं।

8.4. Cursors

Cursor वे होते हैं जिनमें एन्ड्रॉइड में डेटाबेस में किए गए क्वेरी का परिणाम सेट होता है। Cursor class में एक API होता है जो किसी एप्लिकेशन को (एक प्रकार से सुरक्षित तरीके से) columns पढ़ने की अनुमति देता है जो क्वेरी से लौटाए जाने के साथ-साथ परिणाम सेट की rows पर पुनरावृत्ति करता है।

Reading Cursor Data

एक बार जब एक cursor डेटाबेस क्वेरी से लौटाया जाता है, तो एक ऐप को परिणाम सेट पर पुनरावृत्ति करने और cursor से कॉलम डेटा पढ़ने की आवश्यकता होती है। आंतरिक रूप से, cursor क्वेरी द्वारा लौटाए गए डेटा की Row को एक स्थिति के साथ संग्रहीत करता है जो परिणाम सेट में डेटा की वर्तमान Row को इंगित करता है। जब कोई cursor query () method से लौटाया जाता है, तो उसकी स्थिति डेटा की पहली Row से पहले की ओर इशारा करती है। इसका मतलब यह है कि डेटा की किसी भी Row को cursor से पढ़ा जा सकता है, स्थिति को डेटा की एक valid Row को इंगित करने के लिए स्थानांतरित किया जाना चाहिए।

अपनी internal position में manipulate करने के लिए निम्नलिखित method प्रदान करता है—

- (i) boolean Cursor.move (int offset) : दिए गए ऑफसेट position पर moves करता है।
- (ii) boolean Cursor.moveToFirst () : Moves the position to the first row
- (iii) boolean Cursor.moveToLast () : position को पहली row में ले जाता है।
- (iv) boolean Cursor.moveToNext () : वर्तमान स्थिति के सापेक्ष अगली row में कर्सर ले जाता है।
- (v) boolean Cursor.moveToPosition (int position) : कर्सर को Specific position पर ले जाता है।
- (vi) Cursor.moveToPrevious () : वर्तमान स्थिति के सापेक्ष कर्सर को previous row में ले जाता है।

Reading Cursor Data

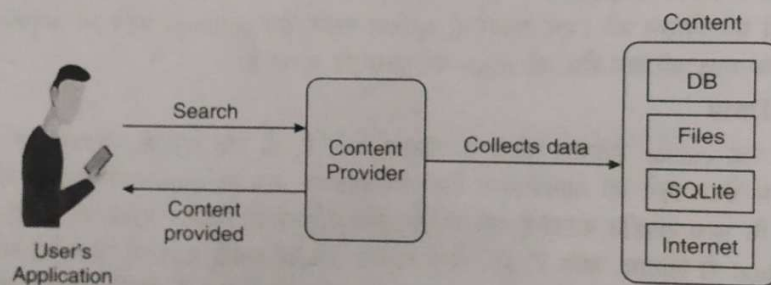
```
Reading Cursor Data
SQLiteDatabase db = getDatabase();
String[] columns = {"first_name",
    "last_name",
    "id"};
```

184 डैवलपमेंट ऑफ एन्ड्रॉइड एप्लीकेशन्स

```
Cursor cursor = db.query("people",
    columns,
    null,
    null,
    null,
    null,
    null);
while(cursor.moveToNext()) {
    int index;
    index = cursor.getColumnIndexOrThrow("first_name");
    String firstName = cursor.getString(index);
    index = cursor.getColumnIndexOrThrow("last_name");
    String lastName = cursor.getString(index);
    index = cursor.getColumnIndexOrThrow("id");
    long id = cursor.getLong(index);
    //... do something with data
}
```

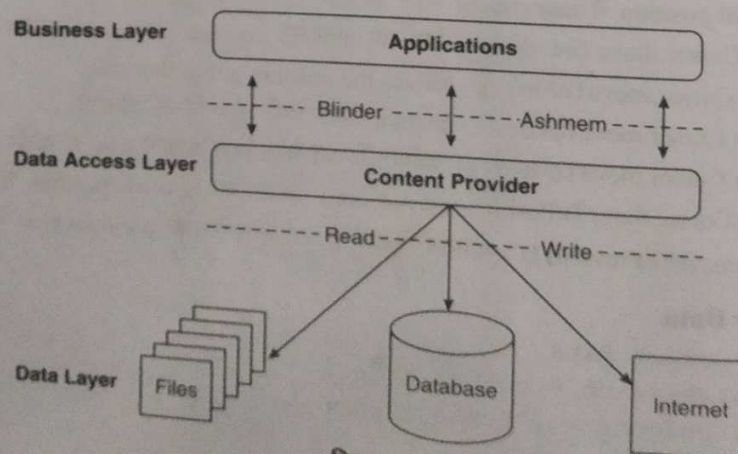
8.5. Content Provider

Content providers एन्ड्रॉइड सिस्टम या अन्य एन्ड्रॉइड एप्लीकेशन से एन्ड्रॉइड एप्लीकेशन को content/data प्रदान करते हैं।



चित्र 8.7

8.5.1. Content Providers Working



चित्र 8.8

8.5.2. Methods of Content Provider in Android

यहाँ उन methods की सूची दी गई है, जिन्हें आपको अपने content provider के काम करने के लिए content provider class में override करने की आवश्यकता है—

- (i) **onCreate()**—जब provider शुरू किया जाता है तो इस method को call जाता है।
- (ii) **query()**—यह method क्लाइंट से request प्राप्त करती है। result को Cursor object के रूप में लौटाया जाता है।
- (iii) **insert()**—यह method content provider में एक new record को insert करती है।
- (iv) **delete()**—यह method content provider में एक existing record को delete करती है।
- (v) **update()**—यह method content provider में एक existing record को update करती है।
- (vi) **getType()**—यह method दिए गए URI में MIME प्रकार का डेटा return करती है।

8.5.3. Content URIs

Content URIs एक uniform resource identifiers हैं जो content providers में डेटा की पहचान करते हैं। एक content URI में दो चीजें शामिल हैं—Authority जो Provider का प्रतीकात्मक नाम है और एक path जो एक ऐसा नाम है जो डेटा की ओर इंगित करता है। प्रत्येक content provider methods में एक argument है जो URI है।

Content Provider URI :

content://<authority>/<path>/<optional_id>

- (i) **content**—यह हमेशा मौजूद है, और URI का scheme portion है।
- (ii) **authority**—यह content provider का unique name है जैसे फोटो, संपर्क। यह एक स्ट्रिंग है जो सम्पूर्ण content provider को identify करता है।
- (iii) **path**—इसका उपयोग अक्सर कुछ या अन्य डेटा के provider की पहचान करने के लिए किया जाता है। path का उपयोग ज्यादातर individual tables की identify करने के लिए किया जाता है।
- (iv) **optional_id** : ID का उपयोग फाइल के particular record तक पहुँचने के लिए किया जाता है। हम इसका उपयोग केवल उन मामलों में करते हैं, जहाँ हमें केवल एक particular record तक पहुँचने की आवश्यकता होती है, न कि सम्पूर्ण फाइल की। data table की किसी particular row तक पहुँचने के लिए यह numeric identifier है।

8.5.4. Implementation of Content Provider in Android

1. ContentProvider.java

```
package com.ggp.contentprovider;
import android.content.ContentProvider;
import android.content.ContentUris;
import android.content.ContentValues;
import android.content.Context;
import android.content.UriMatcher;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteException;
import android.database.sqlite.SQLiteOpenHelper;
import android.database.sqlite.SQLiteQueryBuilder;
import android.net.Uri;
import java.util.HashMap;
```



```

public class GgpProvider extends ContentProvider {
    static final String PROVIDER_NAME =
        "com.ggp.contentprovider.UserProvider";
    static final String URL = "content://" + PROVIDER_NAME +
        "/users";
    static final Uri CONTENT_URI = Uri.parse(URL);
    static final String id = "id";
    static final String name = "name";
    static final int uriCode = 1;
    static final UriMatcher uriMatcher;
    private static HashMap<String, String> values;
    static {
        uriMatcher = new UriMatcher(UriMatcher.NO_MATCH);
        uriMatcher.addURI(PROVIDER_NAME, "users", uriCode);
        uriMatcher.addURI(PROVIDER_NAME, "users/*", uriCode);
    }
    @Override
    public String getType(Uri uri) {
        switch (uriMatcher.match(uri)) {
            case uriCode:
                return "vnd.android.cursor.dir/users";
            default:
                throw new IllegalArgumentException("Unsupported URI: " + uri);
        }
    }
    @Override
    public boolean onCreate() {
        Context context = getContext();
        DatabaseHelper dbHelper = new DatabaseHelper(context);
        db = dbHelper.getWritableDatabase();
        if (db != null) {
            return true;
        }
        return false;
    }
    @Override
    public Cursor query(Uri uri, String[] projection, String selection,
        String[] selectionArgs, String sortOrder) {
        SQLiteQueryBuilder qb = new SQLiteQueryBuilder();
        qb.setTables(TABLE_NAME);
        switch (uriMatcher.match(uri)) {
            case uriCode:
                qb.setProjectionMap(values);
                break;
            default:
                throw new IllegalArgumentException("Unknown URI " + uri);
        }
    }
}

```



```

if (sortOrder == null || sortOrder == "") {
    sortOrder = id;
}

Cursor c = qb.query(db, projection, selection, selectionArgs,
    null, null, sortOrder);
c.setNotificationUri(getContext().getContentResolver(), uri);
return c;
}

@Override
public Uri insert(Uri uri, ContentValues values) {
    long rowID = db.insert(TABLE_NAME, "", values);
    if (rowID > 0) {
        Uri _uri = ContentUris.withAppendedId(CONTENT_URI, rowID);
        getContext().getContentResolver().notifyChange(_uri, null);
        return _uri;
    }
    throw new SQLiteException("Failed to add a record into" + uri);
}

@Override
public int update(Uri uri, ContentValues values, String selection,
    String[] selectionArgs) {
    int count = 0;
    switch (uriMatcher.match(uri)) {
        case uriCode:
            count = db.update(TABLE_NAME, values, selection,
                selectionArgs); break;
        default:
            throw new IllegalArgumentException("Unknown URI " + uri);
    }
    getContext().getContentResolver().notifyChange(uri, null);
    return count;
}

@Override
public int delete(Uri uri, String selection, String[]
    selectionArgs) {
    int count = 0;
    switch (uriMatcher.match(uri)) {
        case uriCode:
            count = db.delete(TABLE_NAME, selection, selectionArgs);
            break;
        default:
            throw new IllegalArgumentException("Unknown URI " + uri);
    }
    getContext().getContentResolver().notifyChange(uri, null);
    return count;
}

private SQLiteDatabase db;

```

```

        static final String DATABASE_NAME = "EmpDB";
        static final String TABLE_NAME = "Employees";
        static final int DATABASE_VERSION = 1;
        static final String CREATE_DB_TABLE = " CREATE TABLE " +
TABLE_NAME
        + " (id INTEGER PRIMARY KEY AUTOINCREMENT, "
        + " name TEXT NOT NULL);";
        private static class DatabaseHelper extends SQLiteOpenHelper {
            DatabaseHelper(Context context) {
                super(context, DATABASE_NAME, null, DATABASE_VERSION);
            }
            @Override
            public void onCreate(SQLiteDatabase db) {
                db.execSQL(CREATE_DB_TABLE);
            }
            @Override
            public void onUpgrade(SQLiteDatabase db, int oldVersion, int
            newVersion) {
                db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
                onCreate(db);
            }
        }
    }
}

```

2. MainActivity.java

```

package com.ggp.contentprovider;
import androidx.appcompat.app.AppCompatActivity;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.view.MotionEvent;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    @Override
    public boolean onTouchEvent(MotionEvent event) {
        InputMethodManager imm =
        (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVI
        CE);
    }
}

```

```

imm.hideSoftInputFromWindow(getCurrentFocus().getWindowToken(),
0);
return true;
}

public void onClickAddDetails(View view) {
    ContentValues values = new ContentValues();
    values.put(GgpProvider.name, ((EditText)
    findViewById(R.id.txtName)).getText().toString());
    getContentResolver().insert(GgpProvider.CONTENT_URI, values);
    Toast.makeText(getBaseContext(), "New Record Inserted",
    Toast.LENGTH_LONG).show();
}

public void onClickShowDetails(View view) {
    TextView resultView= (TextView) findViewById(R.id.res);
    Cursor cursor =
    getContentResolver().query(Uri.parse("content://com.ggp.content
    provider.UserProvider/users"), null, null, null, null);
    if(cursor.moveToFirst()) {
        StringBuilder strBuild=new StringBuilder();
        while (!cursor.isAfterLast()) {
            strBuild.append("\n"+cursor.getString(cursor.getColumnIndex("id
            "))+ "-" + cursor.getString(cursor.getColumnIndex("name")));
            cursor.moveToNext();
        }
        resultView.setText(strBuild);
    }
    else {
        resultView.setText("No Records Found");
    }
}
}
}

```

3. उसक बाद, activity_main.xml खोलें, निम्नलिखित कोड लिखें, यहाँ हमने एप्लिकेशन का लेआउट डिजाइन किया है—

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Ggp "
        android:textColor="#a4c639"
        android:layout_marginLeft="100dp"
        android:textSize="50dp"
        android:layout_centerHorizontal="true" />

```



```

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="50dp"
    android:layout_marginTop="50dp"
    android:text="Data Flair User" />
<EditText
    android:id="@+id/txtName"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="50dp"
    android:ems="10" />
<Button
    android:id="@+id/btnAdd"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="50dp"
    android:onClick="onClickAddDetails"
    android:text="Add User" />
<Button
    android:id="@+id/btnRetrieve"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="50dp"
    android:onClick="onClickShowDetails"
    android:text="Show Users" />
<TextView
    android:id="@+id/res"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="50dp"
    android:clickable="false"
    android:ems="10" />
</LinearLayout>

```

4. manifest.xml file, निम्नलिखित कोड लिखें—

आप content provider की मदद से डेटा दर्ज कर सकते हैं और उसे पुनः प्राप्त कर सकते हैं—

```

<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.ggp.contentprovider">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/AppTheme">

```



```

<activity android:name="com.ggp.contentprovider.MainActivity">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
<provider
    android:authorities="com.ggp.contentprovider.UserProvider"
    android:name="com.ggp.contentprovider.GgpProvider">
</provider>
</application>
</manifest>

```

ContentProvider

ADD USER

SHOW USERS

चित्र 8.9

आप Content Provider की मदद से डेटा दर्ज कर सकते हैं। और उसे पुनः प्राप्त कर सकते हैं।

ContentProvider

ADD USER

SHOW USERS

1. Ggp
2. Abc
3. Ffc
4. Gmn
5. Dfg
6. k jh

चित्र 8.10

प्रश्नावली

1. Android में Data store करने के विभिन्न तरीके समझाइये।
2. Shared preference क्या है? समझाइये।
3. Internal storage में file को save करने के लिये कौन से function प्रयोग किये जाते हैं?
4. External storage में फाइल को save करने के syn tax को लिखें।
5. SQLite क्या है? समझाइये।
6. Web-based service में data को save और retrieve करने के लिये कौन-सी class का प्रयोग किया जाता है?
7. SQLite के features बताइये।
8. SQL और SQLite में difference समझाइये।
9. CRUD operation कौन से हैं?
10. Cursor क्या है? इसका उपयोग को उदाहरण सहित वर्णन करें।
11. Content provider की प्रक्रिया का विस्तृत वर्णन करें।
12. Content URI का उपयोग समझाइये।