# MCA: -LMC0422
# Mobile application design and development Practical

**CENTRE FOR ONLINE AND DISTANCE EDUCATION (CODE)**

**Mody University of Science and TechnologyLakshmangarh**

## Syllabus

| | |
|---|---|
| **Program** | MasterofComputerApplications |
| **Semester** | 4 |
| **CourseTitle** | Mobile application design and developmentPractical |
| **Course Code** | LMC0422 |
| **CourseCredits** | 02 |
| **Course Type** | CorePractical Course |

### 1. CourseSummary

- Learn Kotlin syntax, variable, function, decision making and loops.
- Introduction to Android SDK, UI development, activities, intents, and working with various UI widgets.
- Manage preferences, SQLite databases for data storage and retrieval.

### 2. CourseContents

| **Unit** | **Unit Description** | **Learning Outcome** |
|---|---|---|
| 1 | Introduction to Kotlin Programming: Basics of Kotlin, Decision Making Statements in Kotlin, Loop Control Statements, Creating and Calling Functions, Advantages and Disadvantages. | Student will Understand and Remember Kotlin Basics. (BTL 1,2) |
| 2 | Basic Android Views: TextView with properties method and listener, EditText with properties method and listener, Button with properties method and listener, ImageButton with properties method and listener. | To Understand, Apply and analyse user interfaces with various widgets. (BTL 2,3,4) |
| 3 | Advances Android Views: Radio Group, Radio Button with properties method and listener, CheckBox with properties method and listener. | To Apply and Analyse user interfaces and handle user interactions. (BTL 3,4) |
| 4 | Date & Time Views: DatePicker with properties method and listener, TimePicker with properties method and listener. | To Apply and Analyse user interfaces with widgets and handle user interactions. (BTL 3,4) |
| 5 | Shared Preferences: Creating Preference, Storing Data into Preference and Retrieving data from preferences. | Students will be able to analyse and implement data for storing etc. (BTL 4,6) |

| | | |
|---|---|---|
| 6 | Managing data using SQLite: SQLite Database Creation, Retrieve data from SQLiteDataBase, Update Data to SQLiteDataBase, Delete Data from SQLIteDataBase. | Students will be able to analyse and implement data management techniques. (BTL 4) |
| | **ListofExperiments** | |
| 1 | Write a Kotlin program that defines a class HelloWorld with a constructor that takes a String parameter name1 and a method word() which prints "name:[name1]" to the console. | • Student will Understand and Remember Kotlin Basics. |
| 2 | Write a Kotlin program that demonstrates the declaration and initialization of various variable types, including String, Byte, Short, Int, Long, Float, Double, Char, and Boolean. The program also prints the values of these variables to the console. | • Student will Understand and Remember Kotlin Basics. |
| 3 | Write a  Kotlin program that demonstrates the use of integer variables and their assignment: | • Student will Understand and Remember Kotlin Basics. |
| 4 | Write a simple Kotlin program that takes user input for their name and prints it to the console. | • Student will Understand and Remember Kotlin Basics. |
| 5 | Write a Kotlin program that demonstrates different ways to loop through ranges using for loops. The program utilizes various range operations such as.., step, and downTo. | • Student will Understand and Remember Kotlin Basics. |
| 6 | Develop an Android Application that will apply Constraint Layout with TextView and EditText. | • Understand, Apply and Analyze user interfaces with layouts and handle user interactions. |
| 7 | Develop an Android Application that will apply ReltiveLayout with Button and ImageButton. | • Understand, Apply and Analyze user interfaces with layouts and handle user interactions. |
| 8 | Develop an Android Application that will apply LinearLayout with DatePicker and TimePicker | • Understand, Apply and Analyze user interfaces with layouts and handle user interactions. |
| 9 | Develop an Android Application that will apply RelativeLayout with RadioButton and CheckBox | • Understand, Apply and Analyze user interfaces with layouts and handle user interactions. |
| 10 | Develop an Android Application that Store and retrieve data using Shared Preference | • Students will be able to analyses and implement data for storing etc. |

| | | |
|---|---|---|
| 11 | Develop an Android Application that Store and retrieve data using SQLiteDataBase | • Students will be able to analyse and implement data management techniques. |
| 12 | Develop an Android Application that Store and retrieve data using SQLiteDataBase | • Students will be able to analyse and implement data management techniques. |

## 3. CourseResources

1. Joseph Annuzzi& Lauren darcey& Shane Conder, Advanced Android Application Development, Wesley
2. Reto Meier. (2010), Professional Android 2 Application Development, Wiley
3. Ian F. Darwin, Android cookbook, O’Reilly
4. Jay A Kreibich. (2010) , Using SQLite , O′Reilly
5. Michael Burton. Android App Development for Dummies, Paperback.;3 edition
6. The Android Developer’s CookBook - Building Application with Android SDK.  Wesley.; 2 edition
7. Mobile Computing using Android, Bharat & Company

## Contents

# Unit1: Introduction

## Introduction of Kotlin

Kotlin is a modern programming language created by JetBrains, the company behind the popular IntelliJ IDEA, and was first released in 2011. It runs on the Java Virtual Machine (JVM) and can also be compiled to JavaScript or native code. Kotlin is an object-oriented language that is considered an enhanced version of Java, while still being fully interoperable with Java code. In 2017, Google officially recognized Kotlin as one of the primary languages for Android development.

Kotlin is a programming language that can run on JVM. Google has announced Kotlin as one of its officially supported programming languages in Android Studio; and the Android community is migrating at a pace from Java to Kotlin.

### Why Kotlin for Android Development?

• Google Support: Google officially supports Kotlin for Android development, ensuring seamless integration with Android Studio.

• Ease of Use: Kotlin's syntax is more concise and straightforward compared to Java, making the code easier to read and maintain.

• Java Compatibility: Kotlin can be used alongside Java in the same project, simplifying the transition from Java to Kotlin.

• Null Safety: Kotlin improves error prevention by providing clear handling of null values, reducing the risk of bugs commonly seen in Java.

• Enhanced Asynchronous Programming: Kotlin’s coroutines make it simpler to manage background tasks, such as network calls, without blocking the user interface.

• Vibrant Community: Kotlin benefits from a growing ecosystem of libraries, tools, and a supportive developer community.

## History of Android

- Thehistoryandversionsofandroidareinterestingtoknow.
- The code names of android ranges fromA toS currently, such as Aestro, Blender, Cupcake,Donut,Eclair,Froyo,Gingerbread,Honeycomb, Ice Cream Sandwitch, Jelly Bean,KitKat,Lollipop,MarshmallowuptoSnow
- Let'sunderstandtheandroidhistoryinasequence.

1) Initially,AndyRubinfoundedAndroidIncorporation in Palo Alto, California, UnitedStatesin October, 2003.
2) In17thAugust2005,Googleacquiredandroid Incorporation. Since then, it is in thesubsidiaryofGoogleIncorporation.
3) ThekeyemployeesofAndroidIncorporation are Andy Rubin, Rich Miner,ChrisWhite andNickSears.
4) Originally intended for camera but shifted tosmart phones later because of low marketforcameraonly.
5) AndroidisthenicknameofAndyRubingiven by coworkers because of his love torobots.
6) In 2007, Google announces the developmentofandroidOS.
7) In2008,HTClaunchedthefirstandroidmobile.

## AndroidVersions

| Android Version | Version Number | Release Date | API Level |
|---|---|---|---|
| **Cupcake** | 1.5 | April 27, 2009 | 3 |
| **Donut** | 1.6 | September 15, 2009 | 4 |
| **Eclair** | 2.0 - 2.1 | October 26, 2009 (2.0), January 12, 2010 (2.1) | 5 - 7 |
| **FroYo** | 2.2 | May 20, 2010 | 8 |
| **Gingerbread** | 2.3 | December 6, 2010 | 9 - 10 |
| **Honeycomb** | 3.0 - 3.2 | February 22, 2011 (3.0) | 11 - 13 |
| **Ice Cream Sandwich** | 4.0 - 4.0.4 | October 18, 2011 | 14 - 15 |
| **Jelly Bean** | 4.1 - 4.3.1 | July 9, 2012 (4.1) | 16 - 18 |
| **KitKat** | 4.4 - 4.4.4 | October 31, 2013 | 19 - 20 |
| **Lollipop** | 5.0 - 5.1.1 | November 12, 2014 | 21 - 22 |
| **Marshmallow** | 6.0 | October 5, 2015 | 23 |
| **Nougat** | 7.0 - 7.1.2 | August 22, 2016 | 24 - 25 |
| **Oreo** | 8.0 - 8.1 | August 21, 2017 | 26 - 27 |
| **Pie** | 9.0 | August 6, 2018 | 28 |
| **Android 10** | 10 | September 3, 2019 | 29 |
| **Android 11** | 11 | September 8, 2020 | 30 |
| **Android 12** | 12 | October 4, 2021 | 31 |
| **Android 13** | 13 | August 15, 2022 | 32 |
| **Android 14** | 14 | August 2023 | 33 |

## Advantages of Android

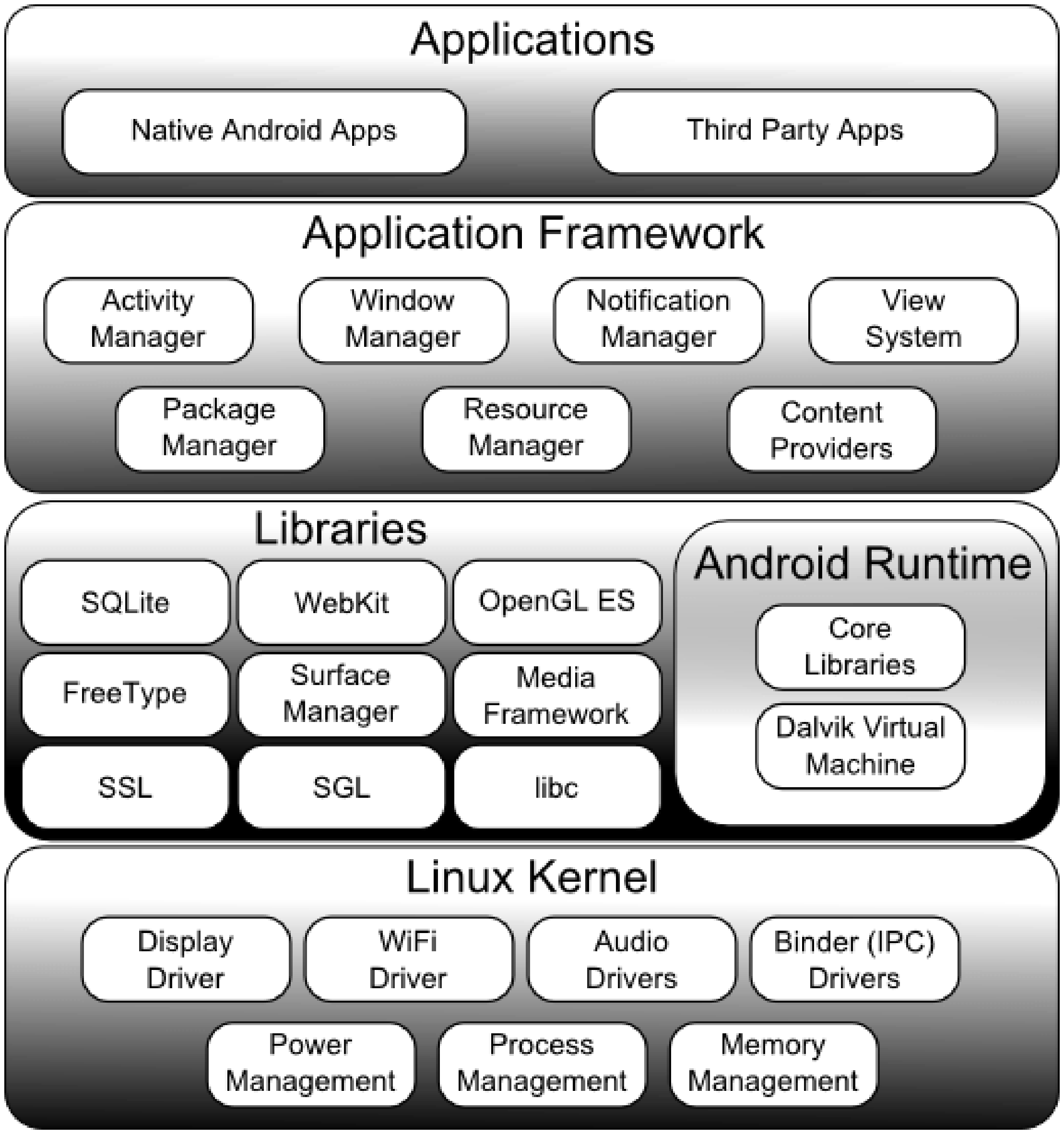
- **Customization**: Android offers a high degree of customization, allowing users to modify their home screens, widgets, and even the system itself through third-party apps and custom ROMs.
- **Wide Range of Devices**: Android is used by many manufacturers (Samsung, Google, OnePlus, Xiaomi, etc.), offering a wide variety of devices in different price ranges to suit different preferences and budgets.
- **Open Source**: Android is an open-source operating system, meaning that developers can access and modify the code, which fosters innovation and leads to a wide array of apps and custom features.
- **Google Integration**: Android seamlessly integrates with Google services, including Gmail, Google Maps, Google Drive, and Google Assistant, offering a cohesive and powerful experience.
- **Multitasking and Flexibility**: Android devices generally allow users to run multiple apps simultaneously, switch between them easily, and even split the screen for multitasking.
- **Variety of Apps**: The Google Play Store has millions of apps available for download, covering a broad range of categories, often including apps that offer more flexibility or functionality than their counterparts on iOS.
- **Expandable Storage**: Many Android devices offer the option to expand storage using a microSD card, providing users with more flexibility in managing their data.
- **Choice of Hardware**: Android users can choose from a variety of devices with different screen sizes, battery capacities, camera configurations, and other features.
- **File Management**: Android offers a more flexible file management system, allowing users to access and manage their files and folders directly, similar to how they would on a desktop.

## Disadvantages of Android

- **Software Updates**: One of the major drawbacks of Android is that updates are not rolled out uniformly across all devices. While Google releases updates for its Pixel phones, other manufacturers may delay updates or fail to provide them for older devices.
- **Fragmentation**: Android devices come in various versions, screen sizes, and configurations. This fragmentation can cause compatibility issues with apps or lead to inconsistency in the user experience.
- **Security Concerns**: While Android has made strides in security, it is still more susceptible to malware, viruses, and security breaches, largely due to the open nature of the ecosystem and the ability to install apps from third-party sources.
- **Bloatware**: Many Android devices come with pre-installed apps (bloatware) that cannot be uninstalled, taking up storage space and potentially affecting performance.
- **Performance Variability**: While premium Android devices are known for high performance, budget and mid-range devices may have issues with speed, lag, and overall performance due to lower-end hardware.
- **Inconsistent User Experience**: Due to the diversity of manufacturers and customizations, Android devices often have varying user experiences. This can be confusing or frustrating for users who switch between devices from different manufacturers.
- **Battery Drain**: Some Android devices experience battery drain due to aggressive background processes or poorly optimized apps, though this can vary by device.
- **App Quality Control**: While the Google Play Store offers a large number of apps, the quality of some apps may not be as high as those on iOS. This is often due to less stringent app approval processes.
- **Learning Curve**: For new users, the wide range of customization options and settings can be overwhelming, and it may take longer to learn how to use Android efficiently compared to other operating systems like iOS.

## Android Software Architecture

- Android is architected in the form of a software stack comprising applications, an operating system, run-time environment, middleware, services and libraries.
- This architecture can, perhaps, best be represented visually as outlined in Following Diagram.
- Each layer of the stack, and the corresponding elements within each layer, are tightly integrated and carefully tuned to provide the optimal application development and execution environment for mobile devices.



**The Linux Kernel**

- Positioned at the bottom of the Android software stack, the Linux Kernel provides a level of abstraction between the device hardware and the upper layers of the Android software stack.
- Based on Linux version 2.6, the kernel provides preemptive multitasking, low-level core system services such as memory, process and power management in addition to providing a network stack and device drivers for hardware such as the device display, Wi-Fi and audio.

**Android Runtime - Dalvik Virtual Machine (DVM)**

- As previously noted, the Linux kernel provides a multitasking execution environment allowing multiple processes to execute concurrently.
- It would be easy to assume, therefore, that each Android application simply runs as a process directly on the Linux kernel.
- In fact, each application running on an Android device does so within its own instance of the Dalvik virtual machine (VM).
- The Dalvik virtual machine was developed by Google and relies on the underlying Linux kernel for low-level functionality.
- It is more efficient than the standard Java VM in terms of memory usage, and specifically designed to allow multiple instances to run efficiently within the resource constraints of a mobile device.

**Android Libraries**

A summary of some key core Android libraries:

- **android.app** – Provides access to the application model and is the cornerstone of all Android applications.
- **android.content** – Facilitates content access, publishing and messaging between applications and application components.
- **android.database**– Used to access data published by content providers and includes SQLite database management classes.
- **android.graphics**– A low-level 2D graphics drawing API including colors, points, filters, rectangles and canvases.
- **android.text** – Used to render and manipulate text on a device display.
- **android.view** – The fundamental building blocks of application user interfaces.
- **android.widget -** A rich collection of pre-built user interface components such as buttons,labels, list views, layout managers, radio buttons etc.

**Application Framework**

- The Application Framework is a set of services that collectively form the environment in which Android applications run and are managed.
- This framework implements the concept that Android applications are constructed from reusable, interchangeable and replaceable components.

The Android framework includes the following key services:

- **Activity Manager**
    - Controls all aspects of the application lifecycle and activity stack.
- **Content Providers**
    - Allows applications to publish and share data with other applications.

- **Resource Manager**
    - Provides access to non-code embedded resources such as strings, color settings and user interface layouts.
- **Notifications Manager**
    - Allows applications to display alerts and notifications to the user.
- **View System**

- An extensible set of views used to create application user interfaces.

- **Package Manager**
  - The system by which applications are able to find out information about other applications currently installed on the device.
- **Telephony Manager**
  - Provides information to the application about the telephony services available on the device such as status and subscriber information.
- **Location Manager**
  - Provides access to the location services allowing an application to receive updates about location changes.

**Applications**

- Located at the top of the Android software stack are the applications.
- These comprise both the native applications provided with the particular Android implementation (for example web browser and email applications) and the third party applications installed by the user after purchasing the device.

# Unit2Android Studio Installation

## System Requirements

Before you start installing Android Studio requirements, it's crucial to make sure your system meets the minimum requirements for smooth performance. Here's what you'll need:

### Hardware

Operating System: Microsoft® Windows® 11/10 (64-bit)

- RAM: 8 GB or more is recommended.
- Storage: 4 GB of available disk space minimum (IDE + Android SDK and emulator system image). 8 GB of available disk space minimum if you also want to install Android Emulator and system image.
- CPU: x86_64 CPU architecture; 2nd generation Intel Core or newer, or an AMD CPU with support for Windows Hypervisor Framework.

### Software

Java Development Kit (JDK): Android Studio requires a JDK to function properly. You can download the latest version of OpenJDK from the official website or install the AdoptOpenJDK distribution.

For the most up-to-date and detailed system requirements, refer to the official Android Studio documentation on the Android Developers website.

While these are the minimum requirements, for optimal performance, especially if you plan to use the Android Emulator, it's recommended to have a more powerful system with the following:

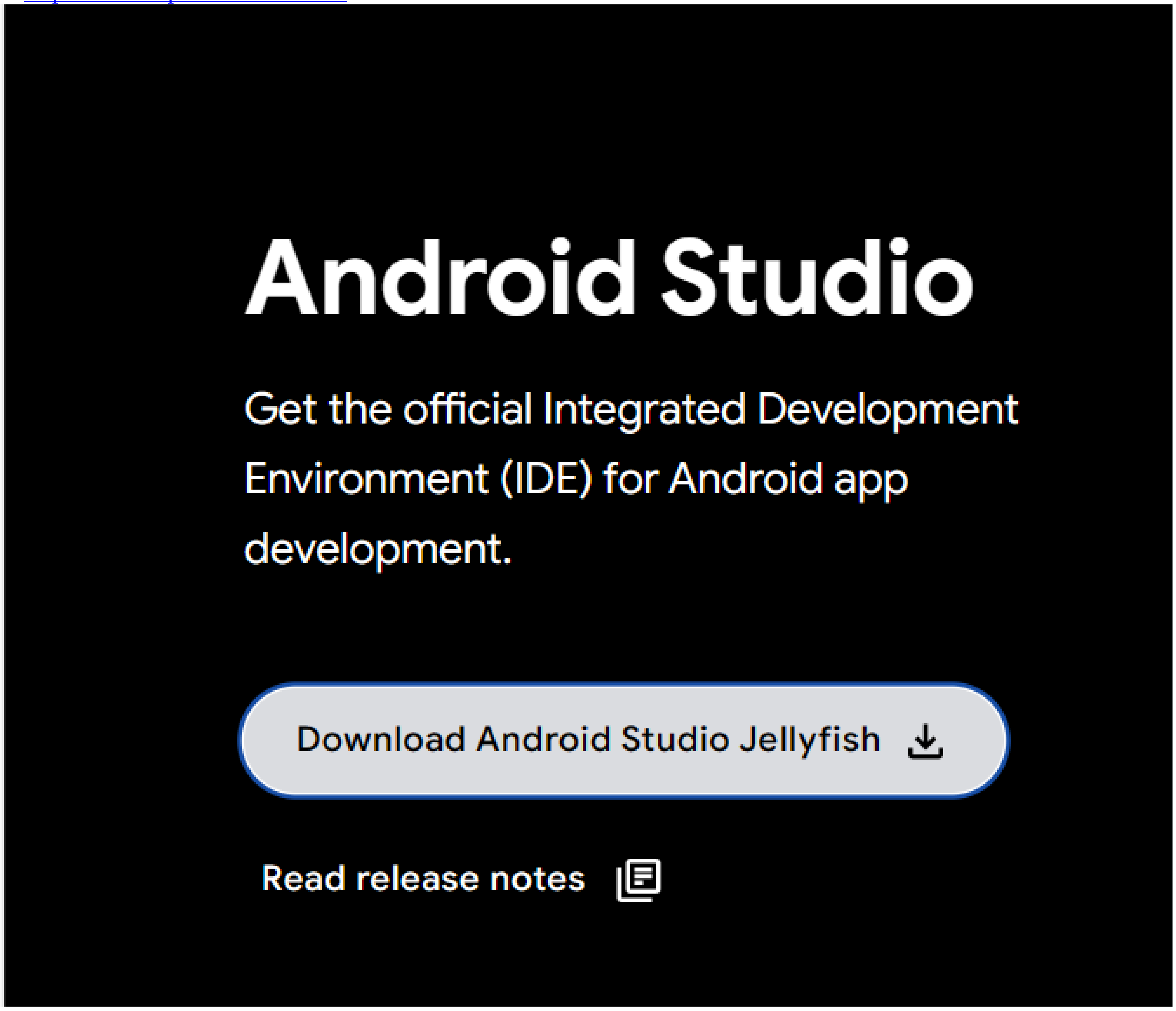
- RAM: 16 GB or more
- Storage: SSD (Solid State Drive) for faster loading and build times
- CPU:  A modern multi-core processor

By ensuring your system meets these requirements, you'll set yourself up for a smoother installation process and a more enjoyable Android development experience.

## Installing Android Studio on Windows

### Step 1: Downloading Android Studio for Windows

Now that you've confirmed your system meets the minimum requirements, it's time to download and install Android Studio on your Windows. This can be done by launching the Android Studio official website: https://developer.android.com/

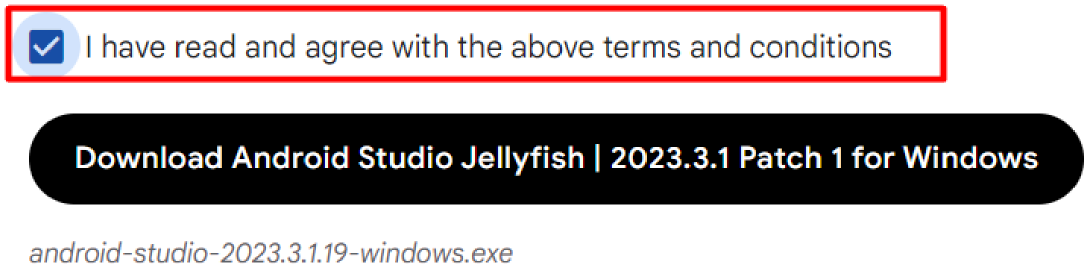

Android Studio
Get the official Integrated Development Environment (IDE) for Android app development.
Download Android Studio Jellyfish
Read release notes

Here, you'll find the latest stable release of Android Studio for Windows.

**Note:** For your security, it's crucial to download Android Studio only from the official Android Developers website. Downloading from unofficial sources may expose your system to potential security risks, such as malware or viruses.

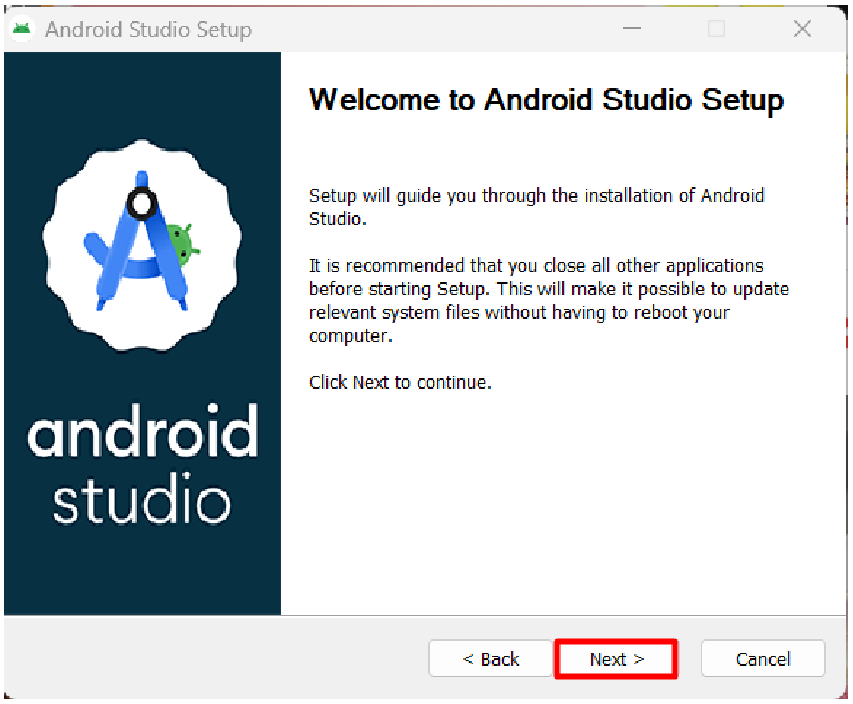
Before you can download Android Studio, you'll be presented with a Terms and Conditions page. This is a standard step in the download process, where you're required to acknowledge and agree to the licensing terms and conditions set by Google, the developer of Android Studio.

To proceed with the download, carefully read through the terms and conditions, then check the box to indicate your acceptance. This will enable the Download button, allowing you to download the Android Studio installation package.


I have read and agree with the above terms and conditions
Download Android Studio Jellyfish | 2023.3.1 Patch 1 for Windows
android-studio-2023.3.1.19-windows.exe

## Step 2: Start the Installer

Once the download is complete, locate the Android Studio installer file in your default downloads folder or wherever you choose to save it. Double-click the installer to begin the installation process.



Welcome to Android Studio Setup

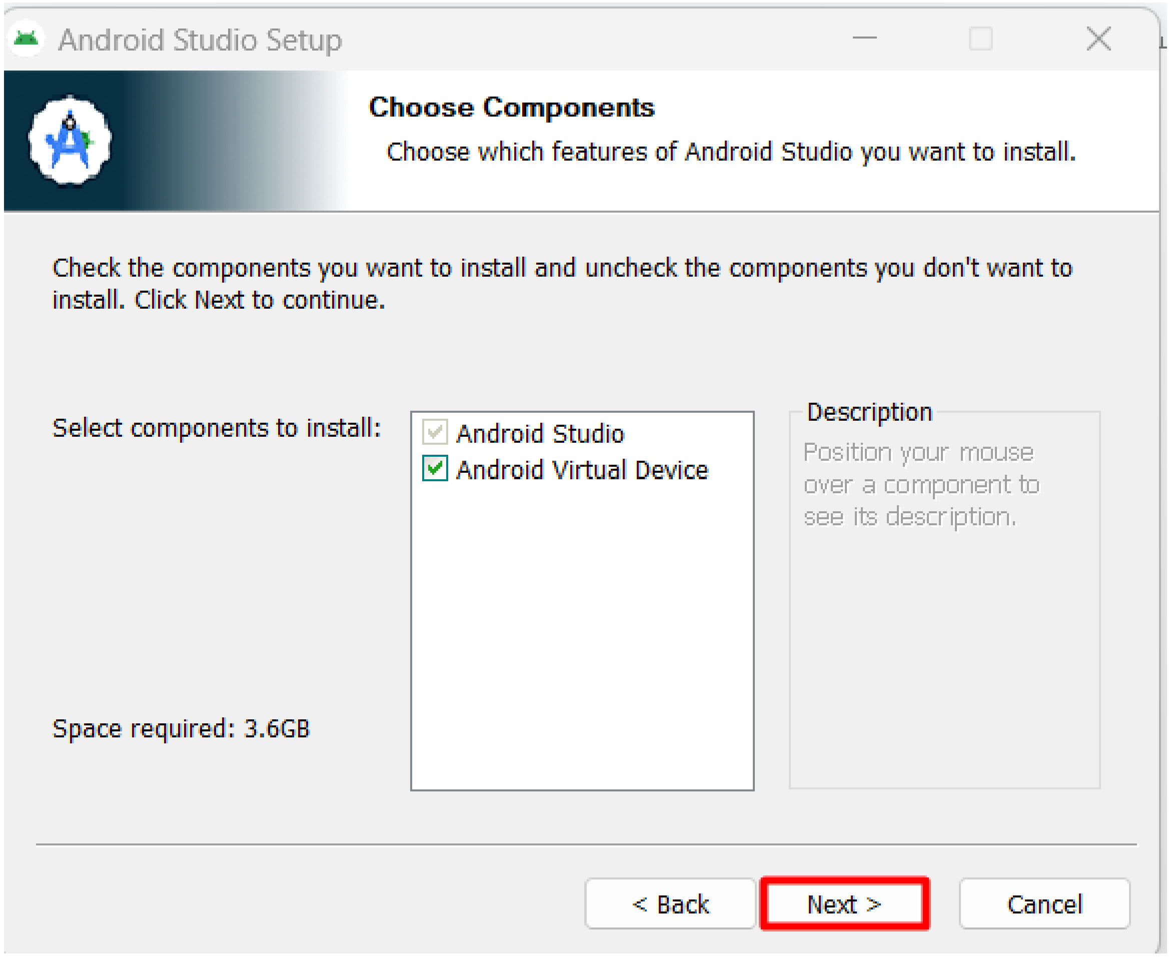
Setup will guide you through the installation of Android Studio.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

## Step 2: Choose Components

The first step in the installation wizard is to Choose Components. By default, the only component selected is the Android Virtual Device, which is required for running and testing your Android apps on an emulator:


Android Studio Setup

**Choose Components**

Choose which features of Android Studio you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

- [✓] Android Studio
- [✓] Android Virtual Device

Description

Position your mouse over a component to see its description.

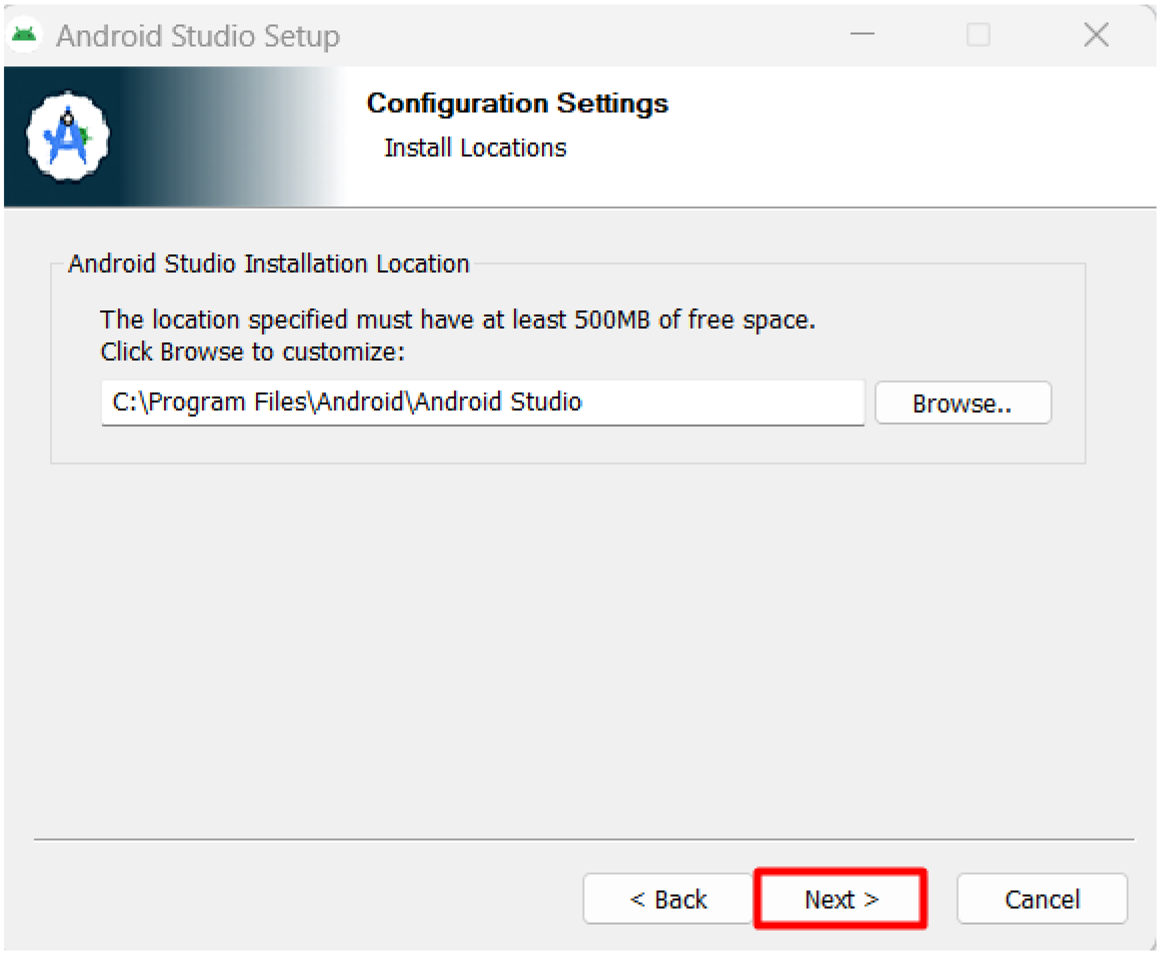
Space required: 3.6GB

In this step, you'll see a brief description of the Android Virtual Device component and the required disk space, which is approximately 3.6 GB.

For a standard Android development environment, it's recommended to install the Android Virtual Device component. This will allow you to create and manage virtual devices, which are essential for testing and debugging your Android apps.

## Step 3: Choose the Installation Location

In this step, you'll be asked to specify the installation location for Android Studio. This is where the Android Studio IDE will be installed on your system. The installation wizard will suggest a default installation location and you can either choose that or specify a different location on your system:



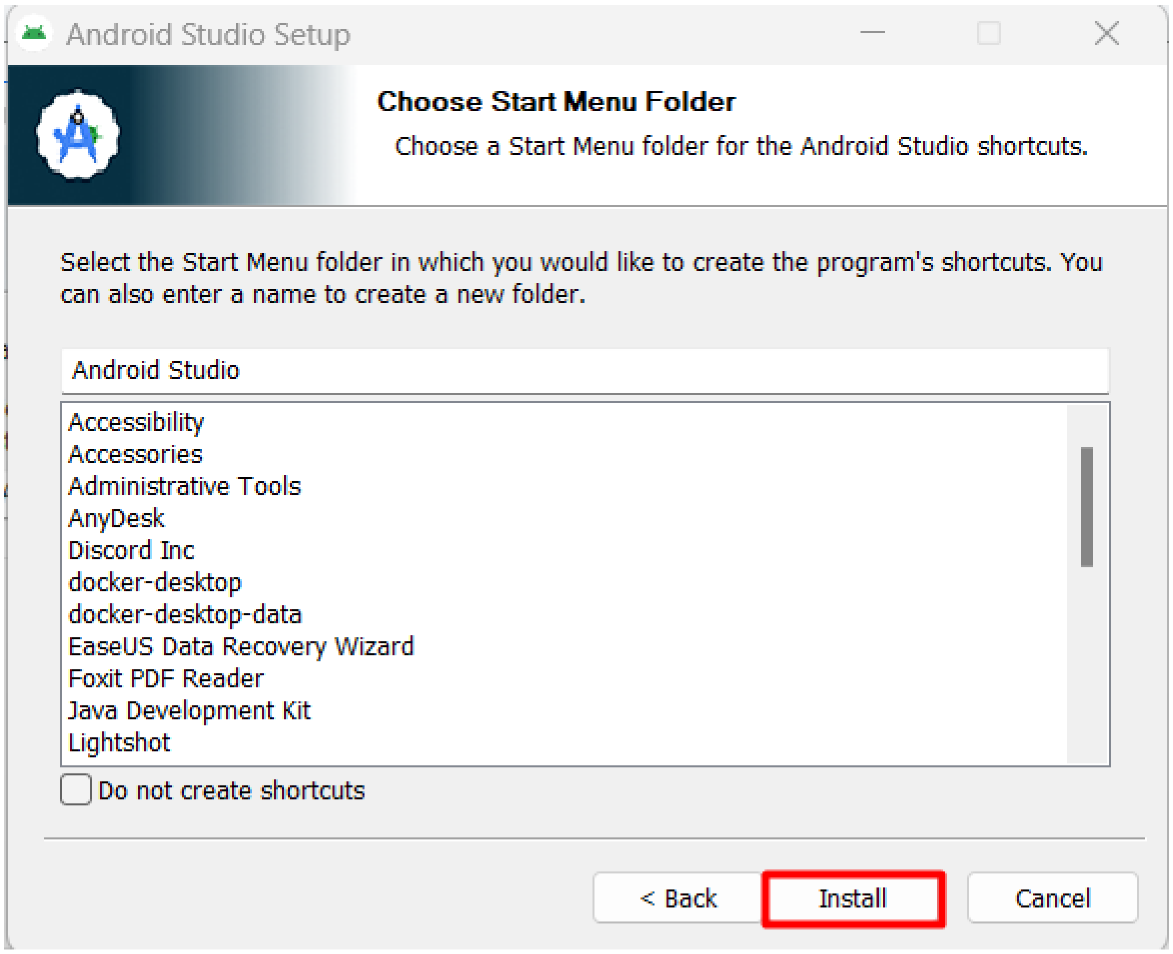
The location specified must have at least 500MB of free space.
Click Browse to customize:
C:\Program Files\Android\Android Studio

## Step 4: Choose the Start Menu Folder and Install

In this step, you'll be asked to specify the Start Menu Folder where you want to create shortcuts for Android Studio. The Start Menu Folder is where you'll find the Android Studio shortcuts in your Windows Start Menu.

Below the Start Menu Folder input field, you'll see a checkbox labeled Do not create shortcuts. If you check this box, the installer will not create any shortcuts for Android Studio in your Start Menu or on your desktop.

Once you've specified the Start Menu Folder and made your selection regarding shortcuts, click the Install button to begin the installation process. The installer will now copy the necessary files and configure Android Studio on your system:


Android Studio Setup

Choose Start Menu Folder

Choose a Start Menu folder for the Android Studio shortcuts.

Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

Android Studio

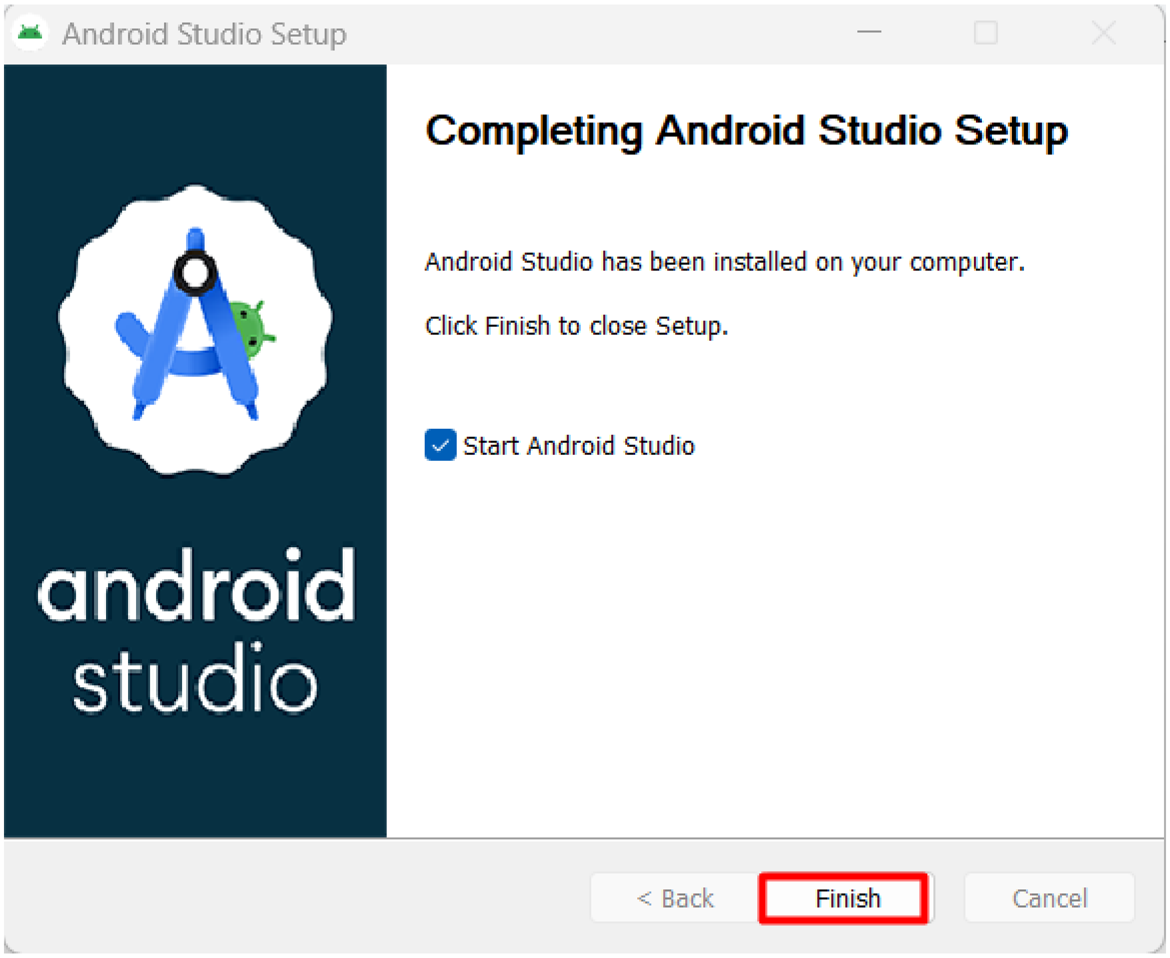
Accessibility
Accessories
Administrative Tools
AnyDesk
Discord Inc
docker-desktop
docker-desktop-data
EaseUS Data Recovery Wizard
Foxit PDF Reader
Java Development Kit
Lightshot

Do not create shortcuts

## Step 5: Complete the Installation

You've reached the final step of the installation process! In this step, you'll have the opportunity to launch Android Studio immediately after installation, and then complete the installation process.

At the bottom of the window, you'll see a checkbox labeled Start Android Studio. If you check this box, Android Studio will launch automatically once the installation is complete:



Completing Android Studio Setup

Android Studio has been installed on your computer.

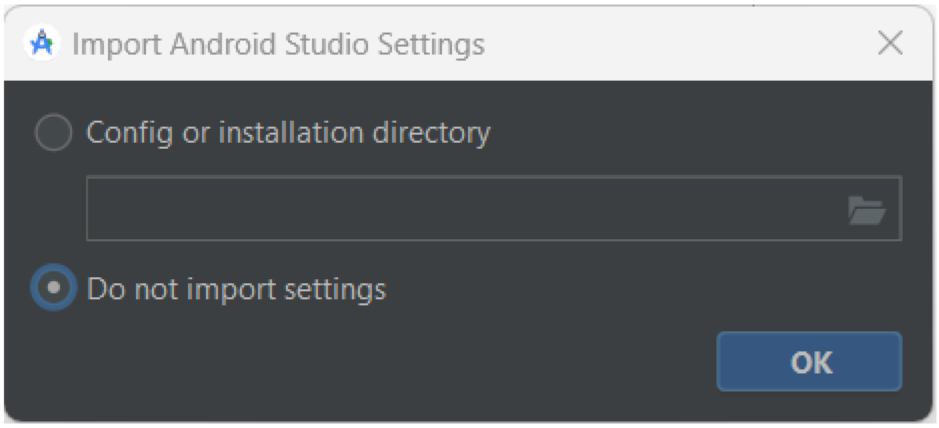
Click Finish to close Setup.

[✓] Start Android Studio

## Step 6: Import Android Studio Settings

After clicking the Finish button, a new window appears, asking you to Import Android Studio settings. This step allows you to customize your Android Studio experience by importing settings from a previous installation or skipping the import process altogether.

If you choose not to import settings, you'll need to set up Android Studio from scratch. While this may take some time, it ensures that you have a clean and customized installation tailored to your specific needs:

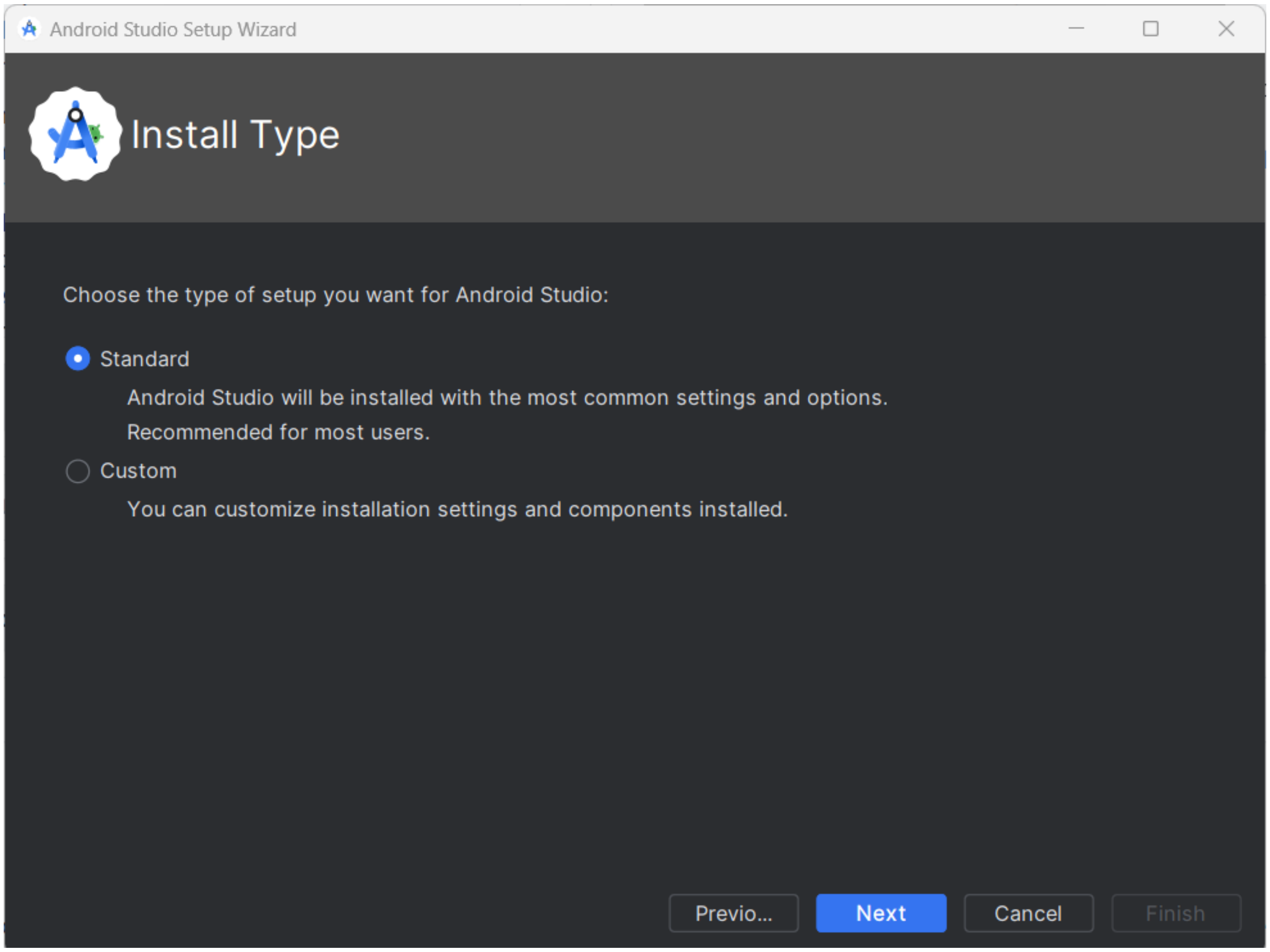


Config or installation directory [✓ Do not import settings]

## Step 7: Choose Settings

In this screen, you'll see two different settings options. The first option is Standard Settings which sets up Android Studio with a default configuration that includes the most commonly used settings and plugins. This is a good choice if you're new to Android Studio or want a straightforward setup process.

The second option is Custom Settings which allows you to customize your Android Studio installation by selecting specific settings and plugins that meet your specific needs. This is a good choice if you have specific requirements or prefer a more tailored setup.



Install Type

Choose the type of setup you want for Android Studio:

[✓ Standard]
Android Studio will be installed with the most common settings and options.
Recommended for most users.

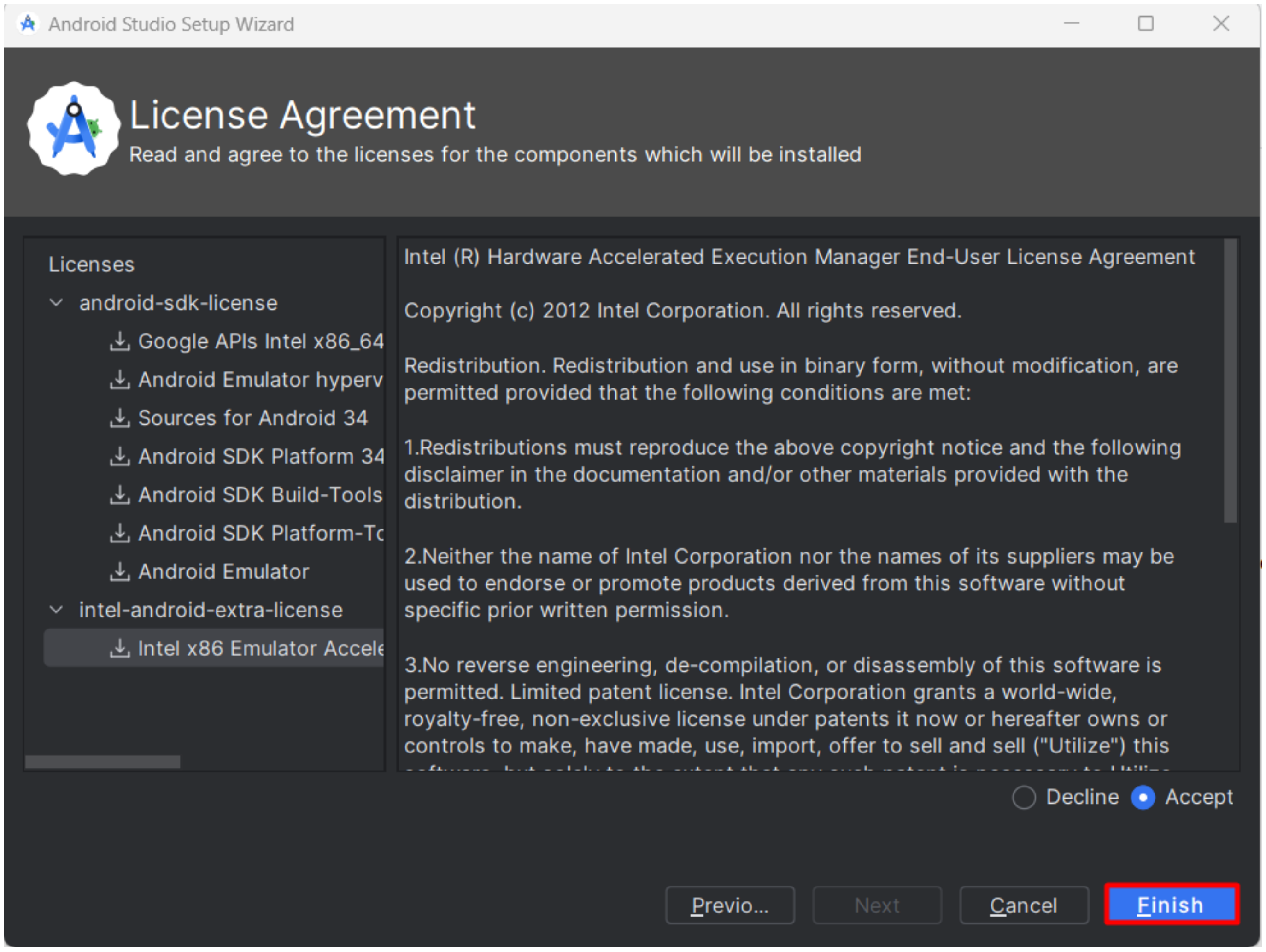
Custom
You can customize installation settings and components installed.

## Step 8: Accept License Agreement

Before you can complete the installation process, you'll need to agree to the Android Studio License Agreement. This agreement outlines the terms and conditions of using Android Studio, including intellectual property rights, warranties, and limitations of liability.

Take a moment to read through the license agreement carefully. It's essential to understand the terms and conditions of using Android Studio, especially if you plan to use it for commercial purposes.

At the bottom of the screen, you'll see a Finish button which will only be enabled if you accept the license agreement. Once you click Finish, the installation process will be complete, and Android Studio will be ready for use.



Intel (R) Hardware Accelerated Execution Manager End-User License Agreement

Copyright (c) 2012 Intel Corporation. All rights reserved.

Redistribution. Redistribution and use in binary form, without modification, are permitted provided that the following conditions are met:

1.Redistributions must reproduce the above copyright notice and the following disclaimer in the documentation and/or other materials provided with the distribution.

2.Neither the name of Intel Corporation nor the names of its suppliers may be used to endorse or promote products derived from this software without specific prior written permission.

3.No reverse engineering, de-compilation, or disassembly of this software is permitted. Limited patent license. Intel Corporation grants a world-wide, royalty-free, non-exclusive license under patents it now or hereafter owns or controls to make, have made, use, import, offer to sell and sell ("Utilize") this

Decline [✓ Accept]

## Step 9: SDK Download and Configure

1. When you first launch Android Studio, the "Welcome to Android Studio" window will appear.
2. Click on "Configure" > "SDK Manager."
3. Select the SDK platforms and tools you need.
4. Click "Apply" to download and install the selected SDK components.

## Unit3: List of Experiments

Anindicativelistof experiments isgiven inTable 1.2.

### ListofExperiments

| | |
|---|---|
| 1 | Write a Kotlin program that defines a class HelloWorld with a constructor that takes a String parameter name1 and a method word() which prints "name:[name1]" to the console. |
| 2 | Write a Kotlin program that demonstrates the declaration and initialization of various variable types, including String, Byte, Short, Int, Long, Float, Double, Char, and Boolean. The program also prints the values of these variables to the console. |
| 3 | Write a  Kotlin program that demonstrates the use of integer variables and their assignment: |
| 4 | Write a simple Kotlin program that takes user input for their name and prints it to the console. |
| 5 | Write a Kotlin program that demonstrates different ways to loop through ranges using for loops. The program utilizes various range operations such as.., step, and downTo. |
| 6 | Develop an Android Application that will apply Constraint Layout with TextView and EditText. |
| 7 | Develop an Android Application that will apply ReltiveLayout with Button and ImageButton. |
| 8 | Develop an Android Application that will apply LinearLayout with DatePicker and TimePicker. |
| 9 | Develop an Android Application that will apply RelativeLayout with RadioButton and CheckBox. |
| 10 | Develop an Android Application that Store and retrieve data using Shared Preference. |
| 11 | Develop an Android Application that Store and retrieve data using SQLiteDataBase. |
| 12 | Develop an Android Application that Store and retrieve data using SQLiteDataBase. |

Table1.2List of experiments

# Unit 4: SamplePrograms

## EXPERIMENT-1

**Problemdefinition:**Write a Kotlin program that defines a class HelloWorld with a constructor that takes a String parameter name1 and a method word() which prints "name:[name1]" to the console.

**SourceProgram:**

```
class HelloWorld(var name1:String) {
   fun word() {
      print("name:$name1")
   }
}

fun main() {
   HelloWorld("prints").word()
}
```

## EXPERIMENT-2

**Problemdefinition:**Write a Kotlin program that demonstrates the declaration and initialization of various variable types, including String, Byte, Short, Int, Long, Float, Double, Char, and Boolean. The program also prints the values of these variables to the console.

**Source Program:**

```
fun main() {
   var name : String
   name="kotlin"
println(name)

   var bytevar : Byte = 123
   var shortvar : Short = 321
   var intvar : Int = 45
   var longvar : Long = 1234567890123564555L
   var floatvar : Float = 3.145f   //3.12F
   var doublevar : Double = 1234.466589254
   var charvar : Char = 'k' //"k" --> Isn't allowed
   var IsAvailable : Boolean = true // or false


println(bytevar)
println(shortvar)
println(intvar)
println(longvar)
println(floatvar)
println(doublevar)
println(charvar)
println(IsAvailable)

}
```

## EXPERIMENT3

**ProblemDefinition:**Write a  Kotlin program that demonstrates the use of integer variables and their assignment:

**Source Program:**

```
fun main() {
var first : Int = 10
println("Before : $first ")
first = 20
println("After : $first ")

var point : Int = 12
}
```

## EXPERIMENT4

**ProblemDefinition:**Write a simple Kotlin program that takes user input for their name and prints it to the console.

**SourceProgram:**

```
fun main() {
print("Enter your name: ")
   var name : String? = readLine()  //readLine() used for User input
println("Name is $name")
}
```

## EXPERIMENT - 5

**Problem Definition:** Write a Kotlin program that demonstrates different ways to loop through ranges using for loops. The program utilizes various range operations such as.., step, and downTo.

**Source Program:**

```
fun main() {
for(i in 1..10) {
print("$i ")
   }
println()
for(i in 1 .. 10 step 2)        //Odd Numbers using 'Step'
   {
print("$i ")
   }
println()
for(i in 2 .. 10 step 2)       //Even Numbers using 'Step'
   {
print("$i ")
   }
println()
for(i in 10 downTo 1 )        //DownTo used for REVERSE Number
   {
print("$i ")
   }
println()
for(i in 10 downTo 1 step 2)   //DownTo with 'Step'
   {
print("$i ")
   }
}
```

## EXPERIMENT - 6

**Problem Definition:** Develop an Android Application that will apply Constraint Layout with

TextView and EditText.

**SourceProgram:**

**activity_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayoutxmlns:android="http://schemas.android.com/a
pk/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=".MainActivity">


<TextView
android:id="@+id/textView5"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="20dp"
android:layout_marginTop="10dp"
android:background="@color/black"
android:fontFamily="serif-monospace"
android:gravity="center"
android:padding="20dp"
android:shadowColor="#86B1FC"
android:shadowDx="10"
android:shadowDy="10"
android:shadowRadius="20"
android:text="Android"
android:textAllCaps="true"
android:textColor="#FFF"
android:textSize="50dp"
android:textStyle="bold|italic"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
tools:ignore="MissingConstraints" />

<TextView
android:id="@+id/textView6"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginStart="6dp"
android:layout_marginTop="50dp"
android:gravity="center"
android:text="TextView Demo \n using KOTLIN"
android:textSize="30dp"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/textView5" />

<TextView
android:id="@+id/textView7"
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginStart="0dp"
android:layout_marginBottom="200dp"
android:text="Click to Open www.google.com"
android:background="@color/purple_700"
android:textColor="#FFF"
android:textSize="40dp"
android:gravity="center"
android:padding="20dp"
android:autoLink="all"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintStart_toStartOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

**MainActivity.kt**

```
package com.example.firstapp

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.View
import android.widget.TextView
import android.widget.Toast

class MainActivity: AppCompatActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)

var textview: TextView = findViewById(R.id.textView5)
//var tv = findViewById<TextView>(R.id.textView)

textview.setOnClickListener{
Toast.makeText(applicationContext,textview.text,Toast.LENGTH_LONG).show()
}
/*      textview.setOnClickListener(View.OnClickListener {
          Toast.makeText(applicationContext,"Hello",Toast.LENGTH_LONG).show()
     })*/
}
}
```

## EXPERIMENT–7

**ProblemDefinition:**Develop an Android Application that will apply ReltiveLayout with Button and ImageButton.

### SourceProgram

**activity_main.xml**

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

```
<RelativeLayoutxmlns: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=".MainActivity">

<Button
android:id="@+id/button"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentEnd="true"
android:layout_alignParentBottom="true"
android:layout_marginEnd="301dp"
android:layout_marginBottom="639dp"
android:text="Button" />

<ImageButton
android:id="@+id/imageButton"
android:layout_width="150dp"
android:layout_height="150dp"
android:layout_alignParentEnd="true"
android:layout_alignParentBottom="true"
android:layout_marginEnd="23dp"
android:layout_marginBottom="566dp"
android:scaleType="centerCrop"
app:srcCompat="@drawable/au_logo" />


</RelativeLayout>
```

MainActivity.kt

```
package com.example.typesbuttondemo

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.*
import com.google.android.material.floatingactionbutton.FloatingActionButton

class MainActivity: AppCompatActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)

var btn: Button = findViewById(R.id.button)
var imgbtn: ImageButton= findViewById(R.id.imageButton)

btn.setOnClickListener{
Toast.makeText(applicationContext,"Button Clicked",Toast.LENGTH_LONG).show()
}

imgbtn.setOnClickListener{
Toast.makeText(applicationContext,"Image Button Clicked",Toast.LENGTH_LONG).show()
}
}
```

## EXPERIMENT-8

**ProblemDefinition:**Develop an Android Application that will apply LinearLayout with DatePicker and TimePicker.

**SourceProgram**

**activity_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayoutxmlns: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:orientation="vertical"
android:padding="10dp"
tools:context=".MainActivity">


<EditText
android:id="@+id/editTextDate"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:ems="10"
android:hint="Select Date"
android:inputType="date" />

<EditText
android:id="@+id/editTextTime"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:ems="10"
android:hint="Select Time"
android:inputType="time" />
</LinearLayout>
```

**MainActivity.kt**

```
package com.example.edirtextdemo

import android.app.DatePickerDialog
import android.app.TimePickerDialog
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.text.Editable
import android.text.TextWatcher
import android.widget.*
import java.util.*

class MainActivity: AppCompatActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)


var date = findViewById<EditText>(R.id.editTextDate)
var c = Calendar.getInstance()
/*date.setOnClickListener {
DatePickerDialog(this,DatePickerDialog.OnDateSetListener { datePicker, i, i2, i3 ->
date.setText("$i3/${i2+1}/$i")
            Toast.makeText(applicationContext,"$i3/${i2+1}/$i",Toast.LENGTH_LONG).show()
         },c.get(Calendar.YEAR),c.get(Calendar.MONTH),c.get(Calendar.DAY_OF_MONTH)).show()
      }*/

date.setOnClickListener{
DatePickerDialog(this,DatePickerDialog.OnDateSetListener{ datePicker, i, i2, i3 ->  },
2023,9,15).show()
}

var time = findViewById<EditText>(R.id.editTextTime)
time.setOnClickListener{
TimePickerDialog(this,TimePickerDialog.OnTimeSetListener{ timePicker, i, i2 ->
time.setText("$i : $i2")
},c.get(Calendar.HOUR),c.get(Calendar.MINUTE),false).show()
}
}
}
```

## EXPERIMENT – 9

**Problem definition:** Develop an Android Application that will apply RelativeLayout with RadioButton and CheckBox.

### Source Program

**activity_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayoutxmlns: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=".MainActivity">
```

```
<CheckBox
android:id="@+id/checkBox"
android:layout_width="85dp"
android:layout_height="wrap_content"
android:layout_alignParentStart="true"
android:layout_alignParentEnd="true"
android:layout_alignParentBottom="true"
android:layout_marginStart="30dp"
android:layout_marginEnd="293dp"
android:layout_marginBottom="288dp"
android:text="Java" />

<CheckBox
android:id="@+id/checkBox2"
android:layout_width="87dp"
android:layout_height="wrap_content"
android:layout_alignParentStart="true"
android:layout_alignParentEnd="true"
android:layout_alignParentBottom="true"
android:layout_marginStart="30dp"
android:layout_marginEnd="293dp"
android:layout_marginBottom="235dp"
android:text="Python" />

<CheckBox
android:id="@+id/checkBox3"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentStart="true"
android:layout_alignParentEnd="true"
android:layout_alignParentBottom="true"
android:layout_marginStart="30dp"
android:layout_marginEnd="294dp"
android:layout_marginBottom="182dp"
android:text="Android" />

<TextView
android:id="@+id/textView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentStart="true"
android:layout_alignParentEnd="true"
android:layout_alignParentBottom="true"
android:layout_marginStart="27dp"
android:layout_marginEnd="250dp"
android:layout_marginBottom="101dp"
android:textStyle="bold"
android:textSize="20dp"
android:text="Select Your \n Skills" />

<RadioGroup
android:id="@+id/radioGroup"
```

```
android:layout_width="150dp"
android:layout_height="700px"
android:layout_alignParentEnd="true"
android:layout_alignParentBottom="true"
android:layout_marginEnd="100dp"
android:layout_marginBottom="70dp" >

<RadioButton
android:id="@+id/radioButton1"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="BCA" />
<RadioButton
android:id="@+id/radioButton2"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="BSCIT" />
<RadioButton
android:id="@+id/radioButton3"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="MCA" />
<RadioButton
android:id="@+id/radioButton4"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="MSCIT" />
</RadioGroup>



</RelativeLayout>
```

**MainActivity.kt**

```
package com.example.typesbuttondemo

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.*
import com.google.android.material.floatingactionbutton.FloatingActionButton

class MainActivity: AppCompatActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)


var cb1 : CheckBox= findViewById(R.id.checkBox)
var cb2 : CheckBox= findViewById(R.id.checkBox2)
var cb3 : CheckBox= findViewById(R.id.checkBox3)
var skilltext: TextView = findViewById(R.id.textView)

cb1.setOnClickListener {
```

```
var str = "Java : ${cb1.isChecked}\nPython : ${cb2.isChecked}\nAndroid : ${cb3.isChecked}"
skilltext.setText(str)
}
cb2.setOnClickListener {
var str = "Java : ${cb1.isChecked}\nPython : ${cb2.isChecked}\nAndroid : ${cb3.isChecked}"
skilltext.setText(str)
}
cb3.setOnClickListener {
var str = "Java : ${cb1.isChecked}\nPython : ${cb2.isChecked}\nAndroid : ${cb3.isChecked}"
skilltext.setText(str)
}

var rg: RadioGroup= findViewById(R.id.radioGroup)
var tv2 : TextView = findViewById(R.id.textView2)

rg.setOnCheckedChangeListener{ radioGroup, i->
var rb= findViewById<RadioButton>(i)
if(rb!=null){
tv2.setText(rb.text)
        }
}

var reset : Button = findViewById(R.id.button2)
reset.setOnClickListener{
rg.clearCheck()
tv2.setText("Select Options")
}
}
}
```

## EXPERIMENT – 10

**Problem definition:** Develop an Android Application that Store and retrieve data using Shared Preference.

### Source Program

**activity_main.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayoutxmlns:android="http://schemas.android.com/a
pk/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=".MainActivity">

<EditText
android:id="@+id/editTextTextPersonName"
android:layout_width="wrap_content"
```

```
android:layout_height="wrap_content"
android:layout_marginTop="45dp"
android:ems="10"
android:inputType="textPersonName"
android:text=""
android:hint="Name"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent" />

<EditText
android:id="@+id/editTextTextPassword"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginTop="35dp"
android:ems="10"
android:hint="Password"
android:inputType="textPassword"
android:minHeight="48dp"
app:layout_constraintStart_toStartOf="@+id/editTextTextPersonName"
app:layout_constraintTop_toBottomOf="@+id/editTextTextPersonName" />

<Button
android:id="@+id/button"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginTop="31dp"
android:text="Login"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/editTextTextPassword" />

<Button
android:id="@+id/button2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginTop="24dp"
android:text="Save"
app:layout_constraintEnd_toEndOf="@+id/button"
app:layout_constraintTop_toBottomOf="@+id/button" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

**MainActivity.kt**

```
package com.example.sharedpreferance

import android.content.Context
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
```

```
class MainActivity: AppCompatActivity() {
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_main)

var ed1 = findViewById<EditText>(R.id.editTextTextPersonName)
var ed2 = findViewById<EditText>(R.id.editTextTextPassword)

var b1 : Button = findViewById(R.id.button)
var b2 : Button = findViewById(R.id.button2)

var sp= application.getSharedPreferences("userpass",Context.MODE_PRIVATE)
var editor = sp.edit()

ed1.setText(sp.getString("username",""))
ed2.setText(sp.getString("password",""))

b2.setOnClickListener {
editor.putString("username",ed1.text.toString())
editor.putString("password",ed2.text.toString())
editor.commit()
Toast.makeText(applicationContext,"Data Saved", Toast.LENGTH_LONG).show()
}
}
}
```

## EXPERIMENT- 11

**Problem definition:** Develop an Android Application that Store and retrieve data using SQLiteDataBase.

### Source Program

```
package com.example.dbdemo;

import androidx.appcompat.app.AppCompatActivity;

import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivityextends AppCompatActivity{

SQLiteDatabasemydatabase;

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

dbCreate();
tableCreate();
insertDate();
readData();
updateData();
deleteData();
   }

private void deleteData() {
mydatabase.execSQL("UPDATE mytable SET Password = 'abc' WHERE Password = 'admin'");
Toast.makeText(this, "Data Deleted", Toast.LENGTH_SHORT).show();
   }

private void updateData() {
mydatabase.execSQL("UPDATE mytable SET Password = 'abc' WHERE Password = 'admin'");
Toast.makeText(this, "Data Updated", Toast.LENGTH_SHORT).show();
   }

private void readData() {
Cursor resultSet= mydatabase.rawQuery("Select * from mytable",null);
resultSet.moveToFirst();
String username = resultSet.getString(0);
String password = resultSet.getString(1);
Toast.makeText(this, username + " and " +password, Toast.LENGTH_SHORT).show();
   }

private void insertDate() {
mydatabase.execSQL("INSERT INTO mytable VALUES('admin','admin');");
Toast.makeText(this, "Data Inserted", Toast.LENGTH_SHORT).show();
   }

private void tableCreate() {
mydatabase.execSQL("CREATE TABLE IF NOT EXISTS mytable(Password VARCHAR,Password
VARCHAR);");
Toast.makeText(this, "Table Created", Toast.LENGTH_SHORT).show();
   }

private void dbCreate() {
mydatabase= openOrCreateDatabase("myDB",MODE_PRIVATE,null);
Toast.makeText(this, "Database Created", Toast.LENGTH_SHORT).show();
   }
}
```

## EXPERIMENT - 12

**Problem definition:** Develop an Android Application that Store and retrieve data using SQLiteDataBase.

### Source Program

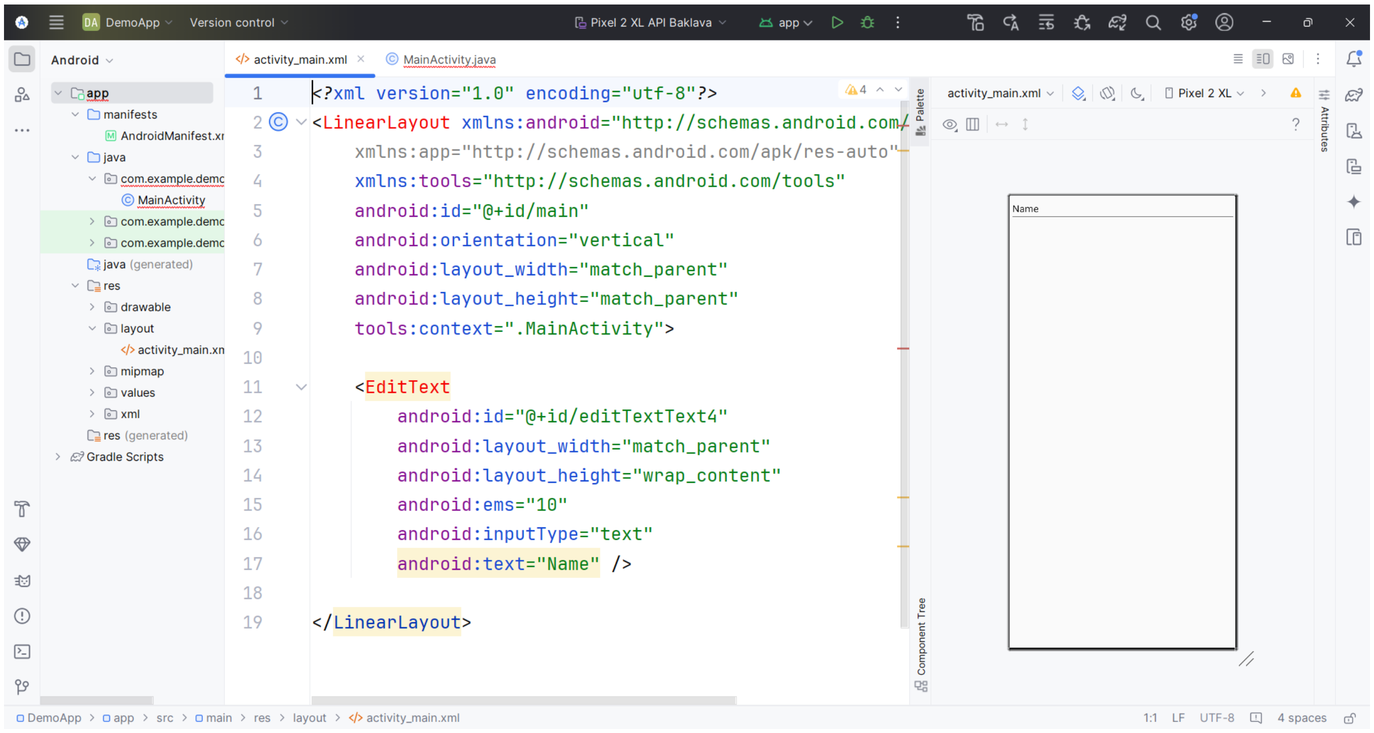
```
package com.example.dbdemo;

import androidx.appcompat.app.AppCompatActivity;

import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.widget.Toast;

public class MainActivityextends AppCompatActivity{

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

dbCreate();
tableCreate();
insertDate();
readData();
updateData();
deleteData();
   }

private void deleteData() {
mydatabase.execSQL("UPDATE mytable SET Password = 'abc' WHERE Password = 'admin'");
Toast.makeText(this, "Data Deleted", Toast.LENGTH_SHORT).show();
   }

private void updateData() {
mydatabase.execSQL("UPDATE mytable SET Password = 'abc' WHERE Password = 'admin'");
Toast.makeText(this, "Data Updated", Toast.LENGTH_SHORT).show();
   }

private void readData() {
Cursor resultSet= mydatabase.rawQuery("Select * from mytable",null);
resultSet.moveToFirst();
String username = resultSet.getString(0);
String password = resultSet.getString(1);
Toast.makeText(this, username + " and " +password, Toast.LENGTH_SHORT).show();
   }

private void insertDate() {
mydatabase.execSQL("INSERT INTO mytable VALUES('admin','admin');");
Toast.makeText(this, "Data Inserted", Toast.LENGTH_SHORT).show();
   }

private void tableCreate() {
mydatabase.execSQL("CREATE TABLE IF NOT EXISTS mytable(UserName VARCHAR,Password
VARCHAR);");
Toast.makeText(this, "Table Created", Toast.LENGTH_SHORT).show();
   }

private void dbCreate() {
mydatabase= openOrCreateDatabase("myDB",MODE_PRIVATE,null);
Toast.makeText(this, "Database Created", Toast.LENGTH_SHORT).show();
   }
}
```

## Unit 5: ExercisePrograms

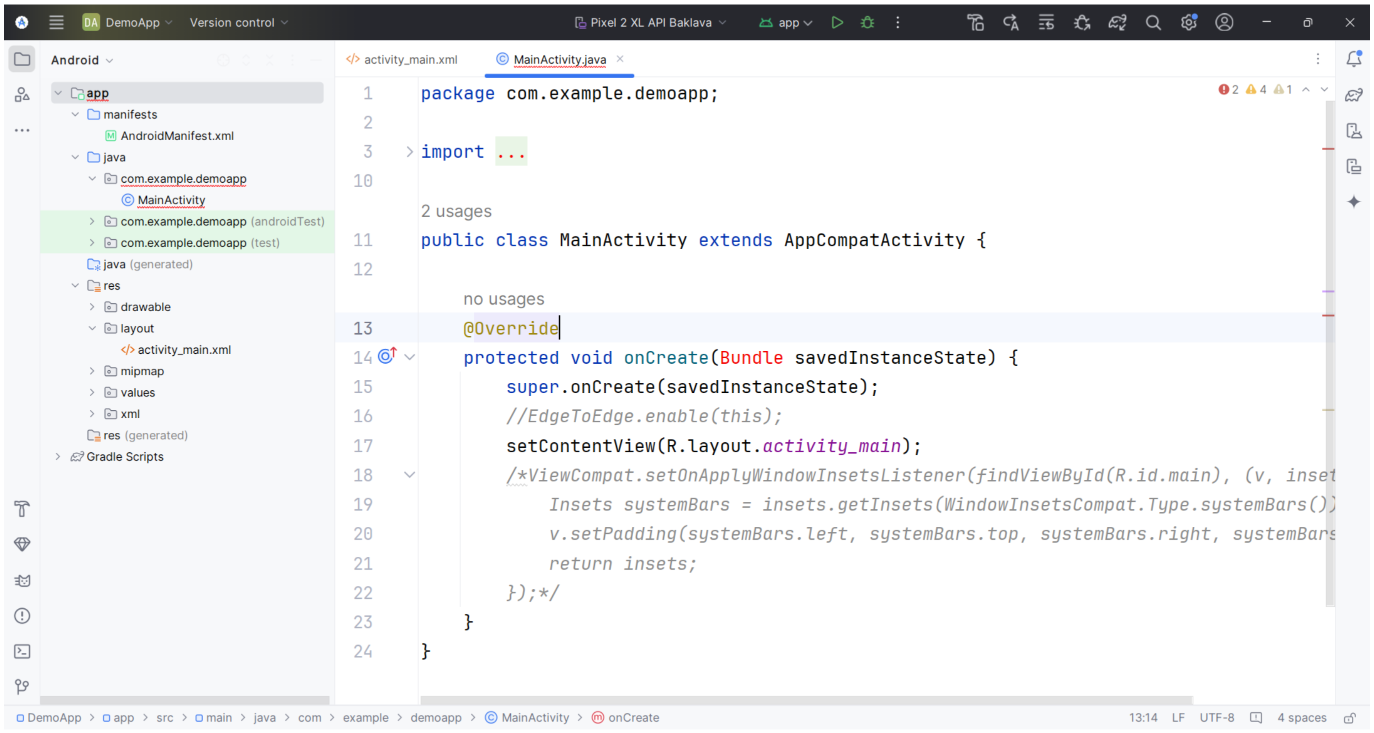
| Topic Covered | Problem Definition |
|---|---|
| **Class, Method, Object** | Write a Kotlin program that defines a class University with a constructor that takes a String parameter name and a method printnew() which prints "name:[name]" to the console. |
| **DataType** | Write a Kotlin program that demonstrates the declaration and initialization of various variable types, including String, Byte, Short, Int, Long, Float, Double, Char, and Boolean. The program also prints the values of these variables to the console. |
| **Variable** | Write a Kotlin program that demonstrates the use of integer and String variables and their assignment: |
| **User Input** | Write a simple Kotlin program that takes user input for their name& age prints it to the console. |
| **Loop** | Write a Kotlin program to take a input a number from user and find whether then entered number is prime or not. |
| **ConstraintLayout, TextView, EditText** | Develop an Android Application that will take input from user by EditText and print inputted text into TextView using ConstraintLayout. |
| **RelativeLayout, Button, ImageButton** | Develop an Android Application that will show Toast on the Button and ImageButton click using RelativeLayout. |
| **LinearLayout, DatePicker, TimePicker** | Develop an Android Application that will take users birthdate and birth timeusing LinearLayout, DatePicker and TimePicker. |
| **RelativeLayout, RadioButton, CheckBox** | Develop an Android Application that will take usres gender and hobby using RelativeLayout with RadioButton and CheckBox. |
| **SharedPreference** | Develop an Android Application that Store and retrieve user's data using Shared Preference. |
| **SQLiteDataBase** | Develop an Android Application that Store and retrieve user's data using SQLiteDataBase. |
| **SQLiteDataBase** | Develop an Android Application that Store and retrieve user's data using SQLiteDataBase. |

# Unit 6: Sample LabRecords

The screenshots for Android programs using Android Studiois given below.



```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/editTextText4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="text"
        android:text="Name" />

</LinearLayout>
```



```java
package com.example.demoapp;

import ...

2 usages
public class MainActivity extends AppCompatActivity {

    no usages
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        /*ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, inset
            Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars()
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars
            return insets;
        });*/
    }
}
```

# Unit 7: Recommended Reading

**Textbooks:**

1. Joseph Annuzzi& Lauren darcey& Shane Conder, Advanced Android Application Development, Wesley
2. Reto Meier. (2010), Professional Android 2 Application Development, Wiley
3. Ian F. Darwin, Android cookbook, O'Reilly
4. Jay A Kreibich. (2010) , Using SQLite , O′Reilly

**Reference Books:**

5. Michael Burton. Android App Development for Dummies, Paperback.;3 edition
6. The Android Developer's CookBook - Building Application with Android SDK. Wesley.; 2 edition
7. Mobile Computing using Android, Bharat & Company