

PROGRAMMING FOR ANDROID (01CE0513) Lab Manual

Name: Asif Alam

ER No.: 92201703058

Class: EC3

Batch: A

INDEX

Lab	Program	Date	Marks	Signature
1.	Install Android Studio with Latest Configuration in your System.			
2.	Study of layouts and UI components used for UI Design.			
3.	Designing UI for Sign in and Sign up screen.			
4.	Building “Hello World” application with customized theme and material design.			
5.	Demonstration of Android Activity and Fragment life cycle for “Hello World” application.			
6.	Create Sign in and Sign up screen using Toast and Recycler view.			
7.	Develop a Torch Light android application using Broadcast Receiver.			
8.	Create an android application using File Handling for text file.			
9.	Create Student Profile application to perform CRUD operations using SQLite.			
10.	Create CGPA Calculator application to perform CRUD operations using SQLite.			
11.	Develop a Horoscope application using API.			
12.	Create weather application using API.			
13.	Develop Mini Project using multimedia.			
14.	Develop Mini Project for Quiz.			



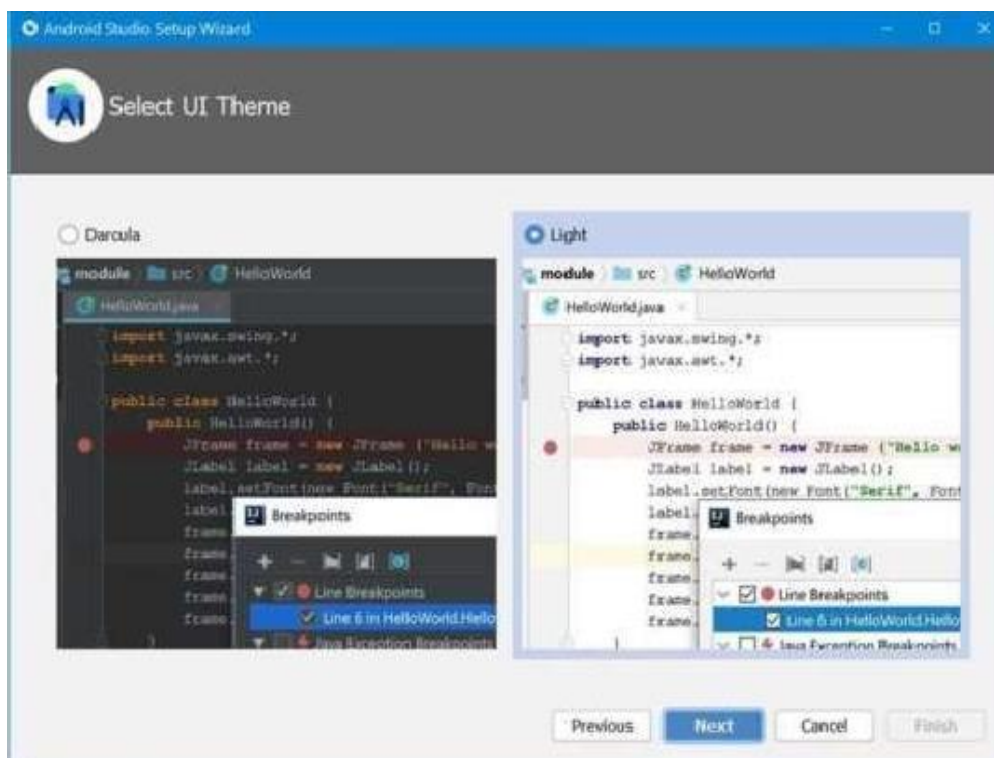
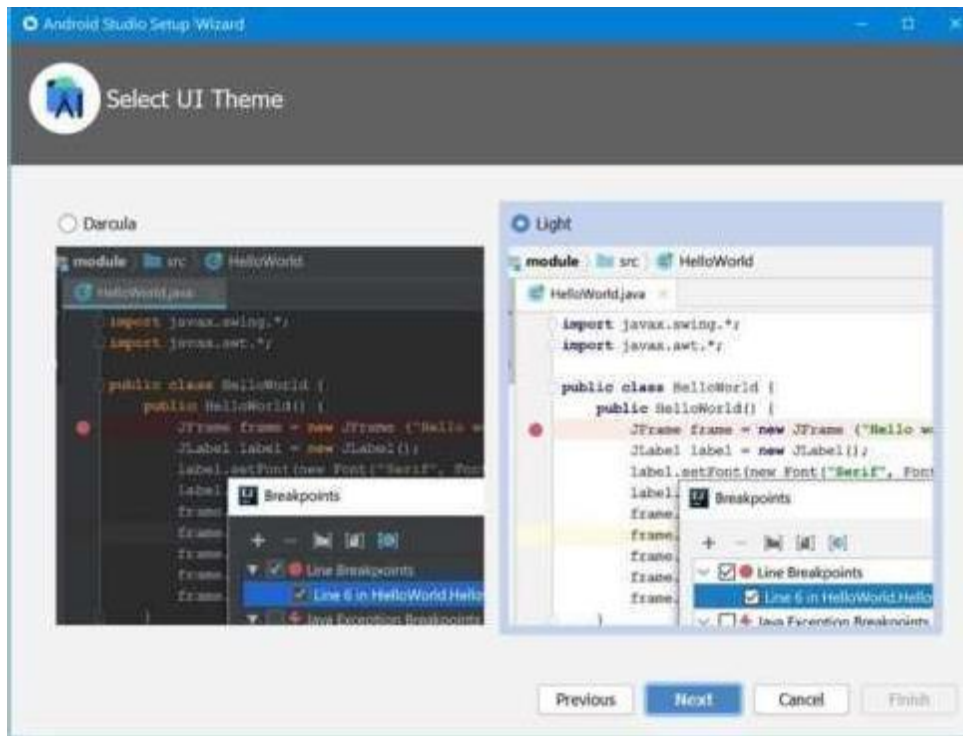
Experiment 1

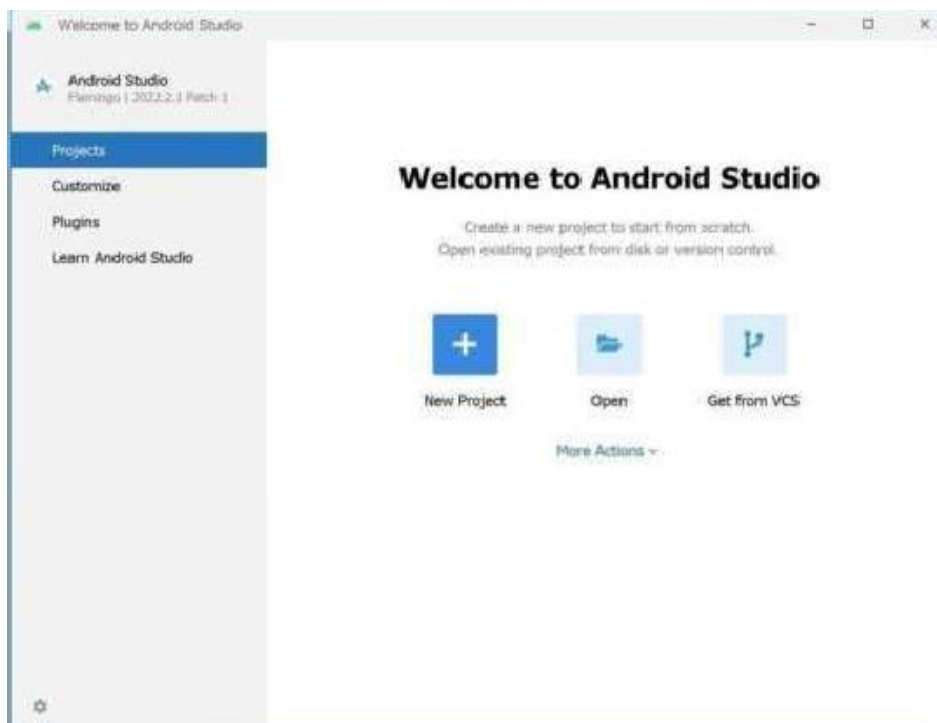
Title: Install Android Studio with Latest Configuration in your System.

Screenshots:

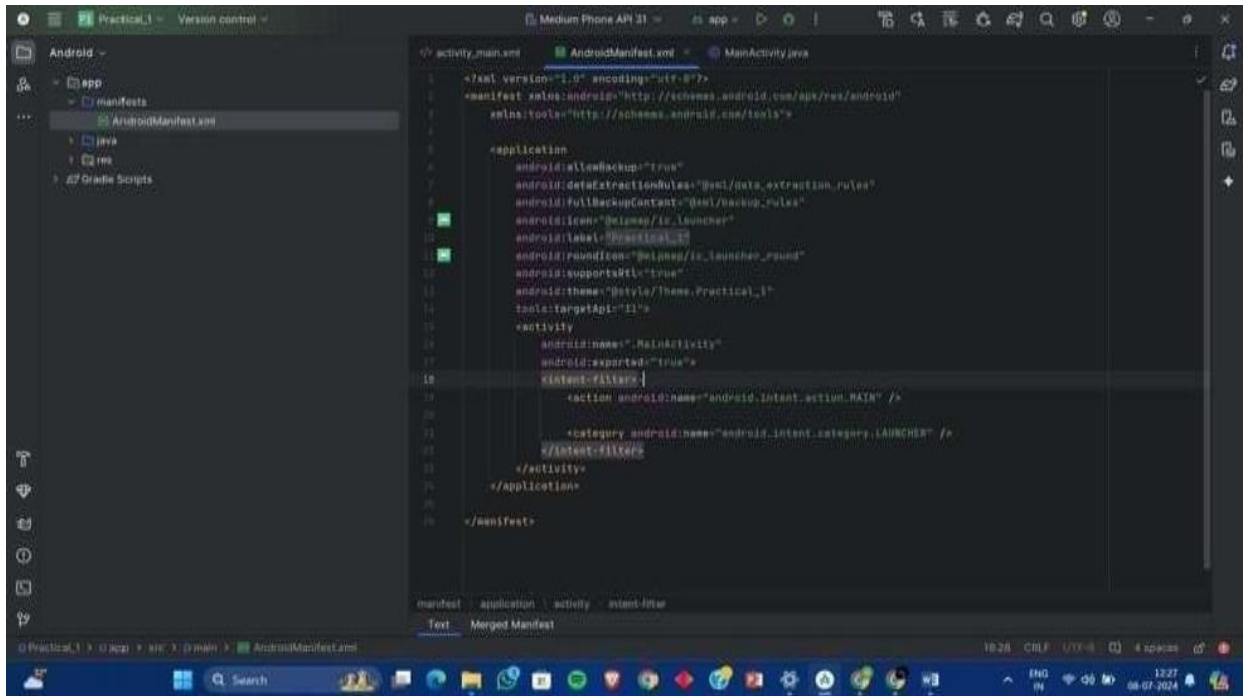








Layout/Screen XML code:



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/Theme.Practical_1"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```



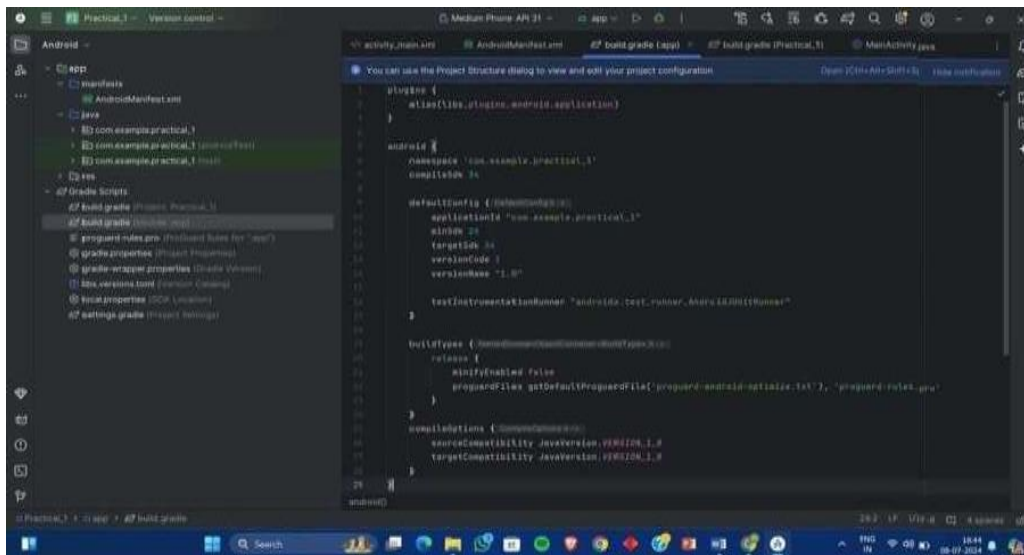

Layout/Screen Manifest XML code Description:

- **XML Declaration:** `<?xml version="1.0" encoding="utf-8"?>` specifies that this is an XML file using UTF-8 encoding.
- **Manifest Root Element:** `<manifest>` is the root element required for an Android manifest file.

It defines the structure of the application.

- **xmlns:android="http://schemas.android.com/apk/res/android":** Declares the XML namespace for Android attributes.
- **package="com.example.myapp":** specifies the unique identifier for your application.
- **Application Element:** `<application>` contains the main components of your application, such as activities, services, broadcast receivers, and content providers.
- **android:label and android:icon** define the application's label (name) and icon.
- **Activity Element:** `<activity>` describes an activity component within your application.
- **android:name** specifies the Java class that implements the activity.
- **android:label** provides the label (title) for the activity.
- **Intent Filter:** `<intent-filter>` specifies the types of intents an activity can respond to.
- **<action>** defines the action name (e.g., MAIN for the main entry point).
- **<category>** specifies the category of intent (e.g., LAUNCHER for the app launcher).
- **Permissions and Features:** You can also declare permissions (`<uses-permission>`) and hardware features (`<uses-feature>`) required by your application.

Layout/Screen XML code:



```
1 plugins {
2     id 'com.android.application'
3 }
4
5 android {
6     namespace 'com.example.practical_1'
7     compileSdk 34
8
9     defaultConfig {
10         applicationId "com.example.practical_1"
11         minSdk 25
12         targetSdk 34
13         versionCode 1
14         versionName "1.0"
15
16         testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
17     }
18
19     buildTypes {
20         debug {
21             minifyEnabled false
22             proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
23         }
24     }
25
26     compileOptions {
27         sourceCompatibility JavaVersion.VERSION_11
28         targetCompatibility JavaVersion.VERSION_11
29     }
30 }
```

Layout/Screen XML code Description:

1. **Buildscript Block:**

1. This block configures the build script itself.
2. **repositories:** Specifies repositories from where Gradle should resolve dependencies.
 - a. **google():** Repository for Android-specific dependencies.
 - b. **jcenter():** General repository for Maven Central and Bintray JCenter.
3. **dependencies:** Specifies the dependencies needed for the build script.
 - a. **classpath 'com.android.tools.build:gradle:4.2.0':** The Android Gradle plugin version used by the project. You can adjust the version number based on your project's requirements.

allprojects Block :

- ─ Configures repositories for all projects/modules in the build.
- ─ Includes the same repositories (google() and jcenter()) to resolve dependencies for all modules.

2. **task clean(type: Delete):**

- Defines a task named clean that deletes the build directory of the root project.
- This task is automatically added by the Android Gradle plugin and is used to clean up build outputs.



Layout/Screen XML code:

Java File :

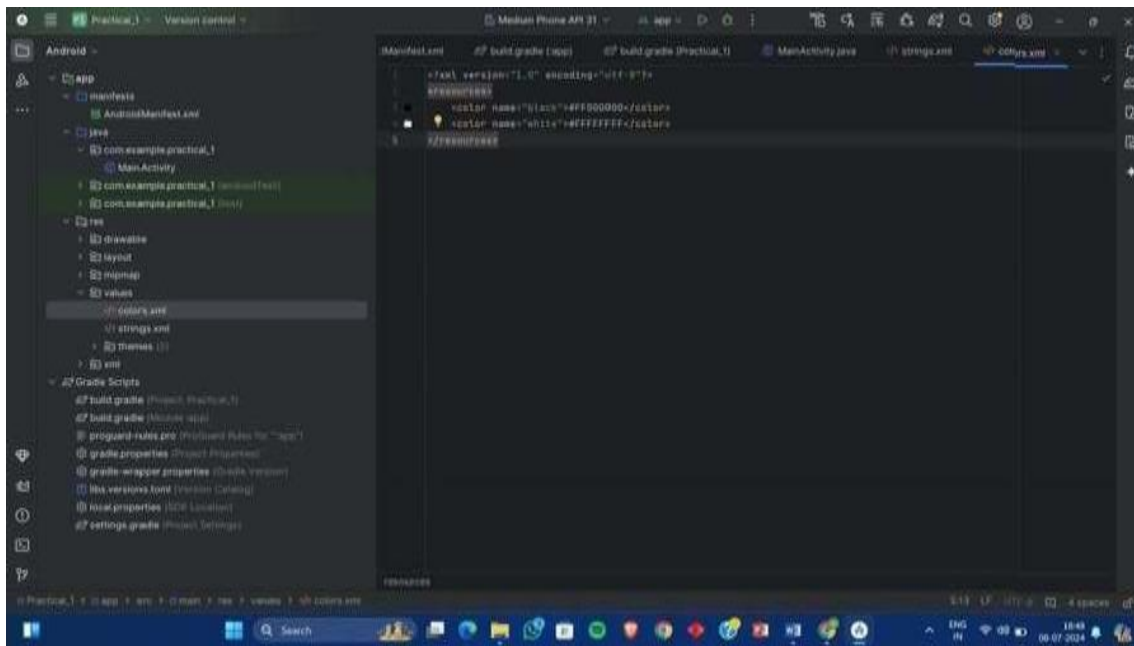
- Written in the Java programming language.
- Contain the core logic of the app, including:
- Defining app functionalities and features.
- Handling user interactions and events.
- Accessing device resources like sensors or cameras.
- Interacting with databases or web services.
- Organized into classes that represent different aspects of the app.
- For instance, you might have a class for handling user authentication or another class for managing data retrieval from a web API.

XML File :

- Written in the Extensible Markup Language (XML).
- Define the layout and visual elements of the app's user interface (UI).
- Common XML files in Android development include:
- Layout files (res/layout/) - Describe the structure and arrangement of UI elements like buttons, text views, and images on the screen.
- String resources (res/values/strings.xml) - Store text content used throughout the app, like button labels or error messages. This allows for centralized management and easy language translation.

Color resources (res/values/colors.xml) - Define color values used in the app's layout, backgrounds, or text elements. This promotes easier color scheme changes without altering the main code.

Layout/Screen XML code:



Layout/Screen XML code Description:

String Resources :

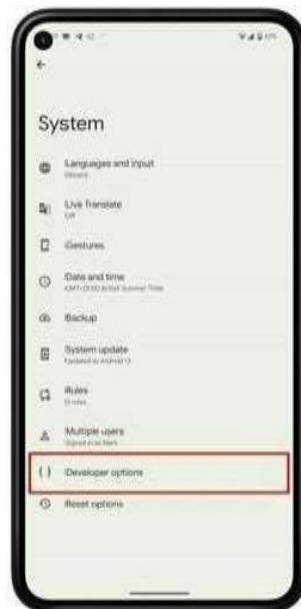
- Provide text content for your app's UI elements like buttons, labels, and error messages.
- Defined in XML files with the <string> tag.
- Can include formatting options like bold or italics.
- Can reference placeholder values using @string/name syntax within the code.

Color Resources :

- Define colors used in the app's layout, backgrounds, or text.
- Defined in XML files with the <color> tag.
- Colors can be specified in various formats like #RRGGBB (hexadecimal) or @color/name for referencing within code.
- Overall, string and color resources promote cleaner, more manageable, and efficient Android app development.

Steps to enable developer mode:







Steps to enable Developer mode Description:

- Open the Settings app on your Android device.
- Scroll down and tap on About phone or About device.
- Look for Build number. In some cases, you might need to tap on Software information first.
- Tap on Build number seven times consecutively. You'll see a message on the screen indicating the number of taps remaining until developer mode is enabled (e.g., "You are now x taps away from being a developer").
- Enter your PIN, pattern, or password if prompted for verification.
- Once enabled, you'll see a message stating "You are now a developer."
- Go back to the main Settings menu. You should now see a new option called Developer options.
- Note: The steps might slightly differ depending on the device manufacturer and Android version. If you have trouble finding the specific options, consult your device's user manual or search online for your specific model.