Lab 18: Android Internal Storage

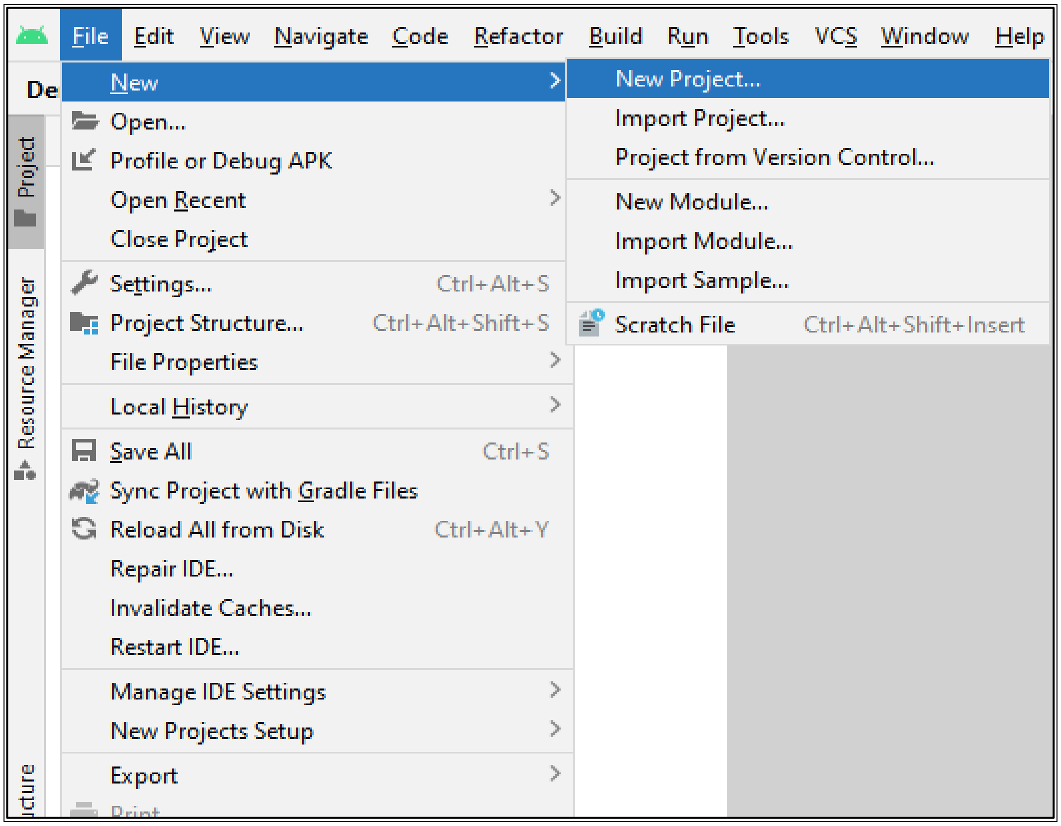
Introduction

Internal storage is the storage of the private data on the device memory. By default these files are private and are accessed by only your application and get deleted , when user delete your application.

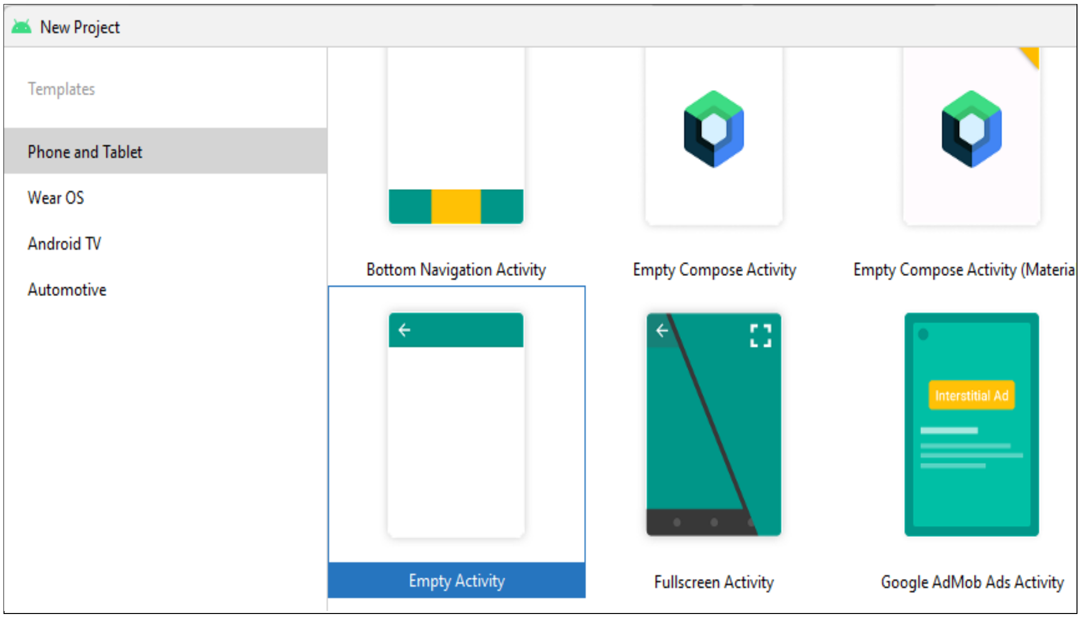
- **Write into File (FileOutputStream)**
- **Read from File (FileInputStream)**

Let’s get Started:

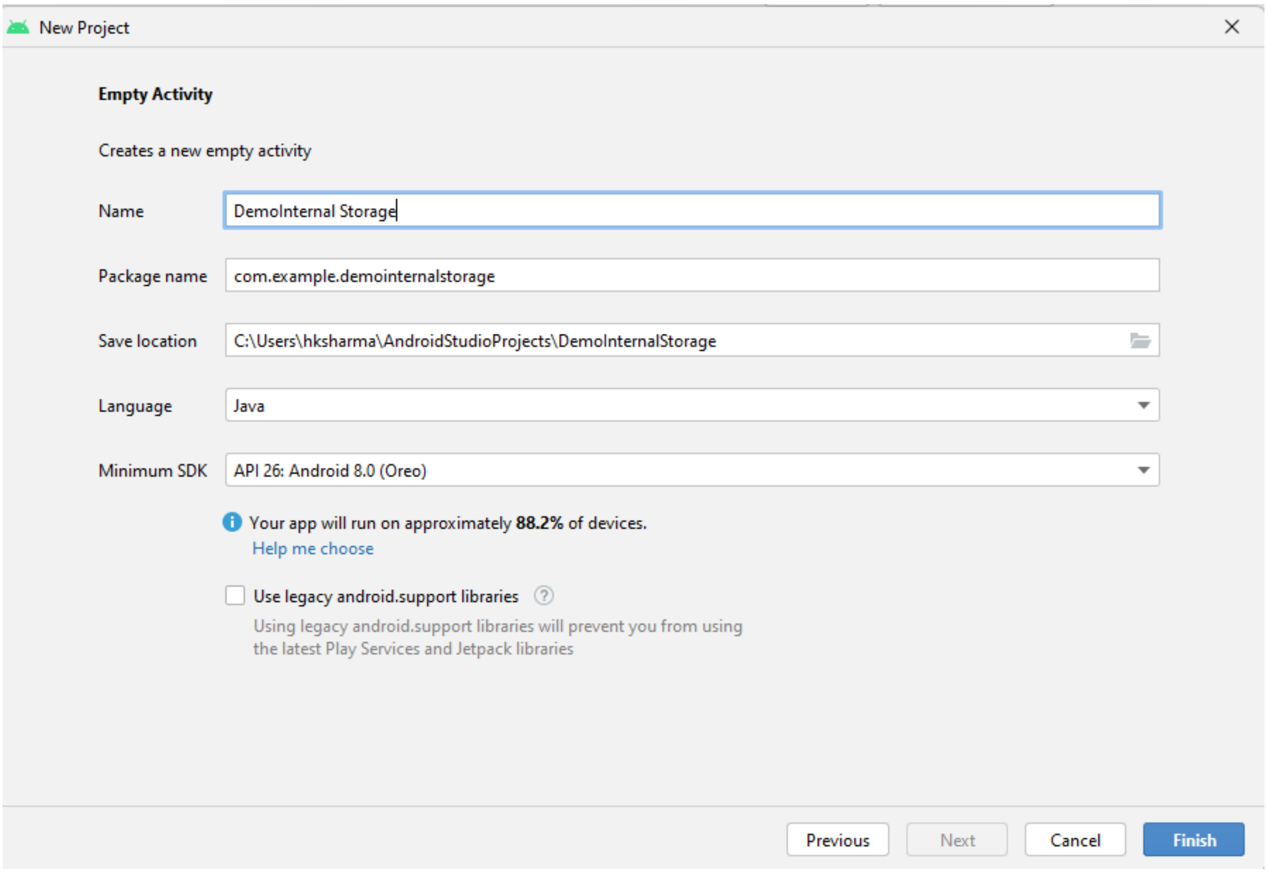
Step 1: Create a New Project in Android Studio as shown below



Step 2: Select Empty Activity as shown below



Step 3: Provide a Project Name as shown below



Empty Activity

Creates a new empty activity

Name: DemoInternal Storage

Package name: com.example.demointernalstorage

Save location: C:\Users\hksharma\AndroidStudioProjects\DemoInternalStorage

Language: Java

Minimum SDK: API 26: Android 8.0 (Oreo)

Your app will run on approximately **88.2%** of devices.
Help me choose

□ Use legacy android.support libraries

Using legacy android.support libraries will prevent you from using
the latest Play Services and Jetpack libraries

Step 4: Update MainActivity.java as per the code given below

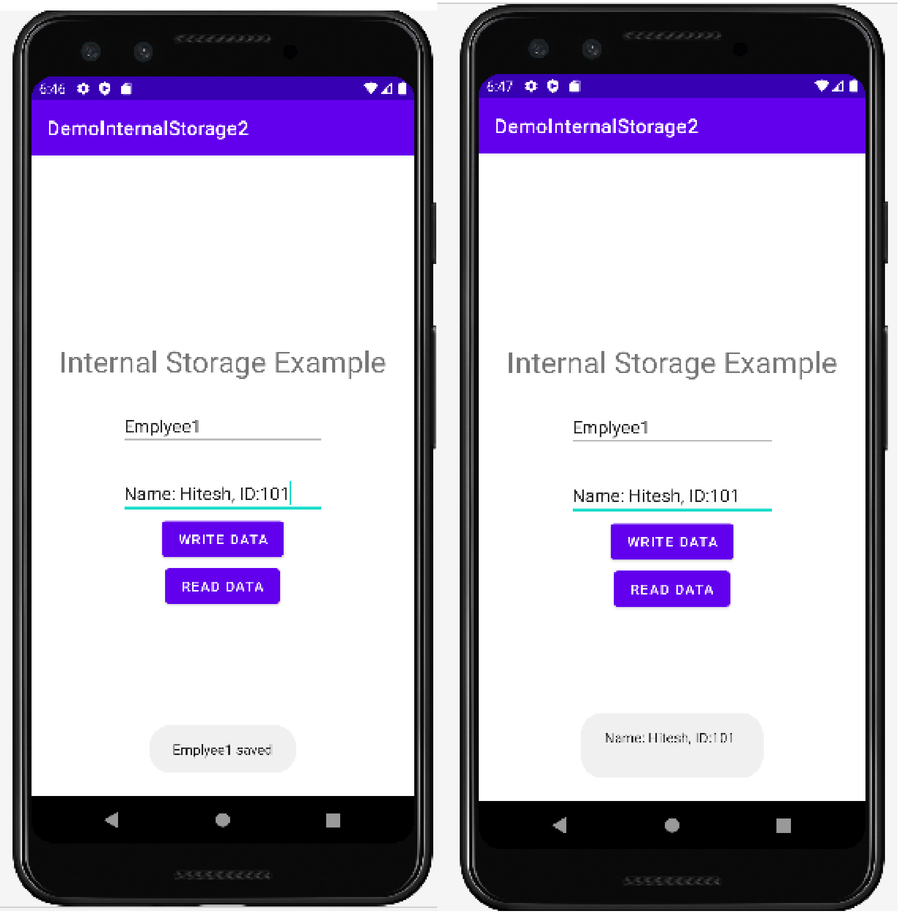
```
package com.example.demointernalstorage2;
import android.content.Context;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;

public class MainActivity extends AppCompatActivity {
    EditText editTextFileName,editTextData;
    Button saveButton,readButton;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        editTextFileName=findViewById(R.id.editText1);
        editTextData=findViewById(R.id.editText2);
        saveButton=findViewById(R.id.button1);
        readButton=findViewById(R.id.button2);

        //Performing Action on Read Button
        saveButton.setOnClickListener(new View.OnClickListener(){

            @Override
            public void onClick(View arg0) {
                String filename=editTextFileName.getText().toString();
                String data=editTextData.getText().toString();

                FileOutputStream fos;
                try {
                    fos = openFileOutput(filename, Context.MODE_PRIVATE);
                    //default mode is PRIVATE, can be APPEND etc.
                    fos.write(data.getBytes());
                    fos.close();

                    Toast.makeText(getApplicationContext(),filename + "
saved",
                            Toast.LENGTH_LONG).show();


                } catch (FileNotFoundException e) {e.printStackTrace();}
                catch (IOException e) {e.printStackTrace();}

            }
        });

        //Performing Action on Read Button
        readButton.setOnClickListener(new View.OnClickListener(){

            @Override
            public void onClick(View arg0) {
                String filename=editTextFileName.getText().toString();
                StringBuffer stringBuffer = new StringBuffer();
                try {
                    //Attaching BufferedReader to the FileInputStream by the
help of InputStreamReader
                    BufferedReader inputReader = new BufferedReader(new
InputStreamReader(
                            openFileInput(filename)));
                    String inputString;
                    //Reading data line by line and storing it into the
stringbuffer
                    while ((inputString = inputReader.readLine()) != null) {
                        stringBuffer.append(inputString + "\n");
                    }

                } catch (IOException e) {
                    e.printStackTrace();
                }
                //Displaying data on the toast

Toast.makeText(getApplicationContext(),stringBuffer.toString(),Toast.LENGTH_LO
NG).show();

            }

        });
    }
}
```

Step 5: Update activity_main.xml for Relative Layout as per the code given below

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Internal Storage Example"
        android:layout_centerHorizontal="true"
        android:textSize="30dp"
        />

    <EditText
```

```
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="Enter a File Name "
        android:layout_marginTop="24dp"
        android:ems="10" >
        <requestFocus />
    </EditText>

    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="24dp"
        android:hint="Enter Data to Store"
        android:ems="10" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Write Data" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"

        android:text="Read Data" />

</LinearLayout>
```

Step 6: Check Output on Android Emulator and it should look like as given below



DemoInternalStorage2

Internal Storage Example

Emplyee1

Name: Hitesh, ID:101

Emplyee1 saved

DemoInternalStorage2

Internal Storage Example

Emplyee1

Name: Hitesh, ID:101

Name: Hitesh, ID:101

Voila!! We have successfully completed this lab.