AIM: Programming UI elements ,AppBar, UI Components

Pract5-A Calculator Application

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
        <EditText android:id="@+id/et1"
            android:layout_width="match_parent"
            android:layout_height="70dp"
            android:ems="10"
            android:inputType="textPersonName"
            android:text="Input1" />
        <EditText android:id="@+id/et2"
            android:layout_width="match_parent"
            android:layout_height="64dp"
            android:ems="10"
            android:inputType="textPersonName"
            android:text="Input2" />
        <Button android:id="@+id/btnAdd"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Addition" />
        <Button android:id="@+id/btnSub"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Subtraction" />
        <Button android:id="@+id/btnMult"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Multiplication" />
        <Button android:id="@+id/btnDiv"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Division" />
        <Button android:id="@+id/btnClear"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Clear" />
        <TextView android:id="@+id/tv1"
            android:layout_width="match_parent"
            android:layout_height="63dp"
            android:text="Output"
            android:textColor="@android:color/background_dark"
            android:textSize="18sp"
            android:textStyle="bold"
            app:fontFamily="casual" />
</LinearLayout>
```

Main_Activity.java

```
package com.example.pract5_a;
import androidx.appcompat.app.AppCompatActivity;
```

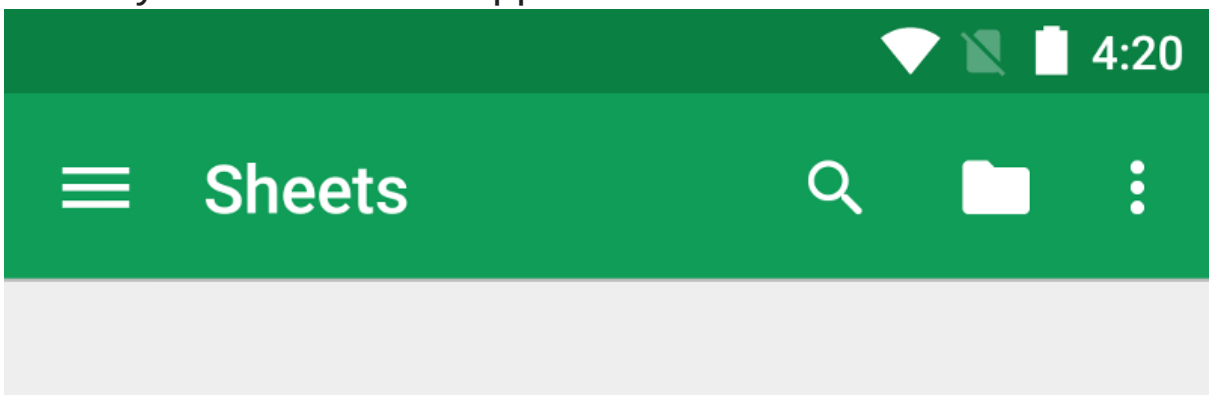
```
import android.view.View;
import android.widget.*;
import static android.view.View.*;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
    EditText t1,t2;
    Button b1,b2,b3,b4,b5;
    TextView tv1;
    int n1=0,n2=0;
    String s1,s2;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        t1 = (EditText) findViewById(R.id.et1);
        t2 = (EditText) findViewById(R.id.et2);
        b1 = (Button) findViewById(R.id.btnAdd);
        b2 = (Button) findViewById(R.id.btnSub);
        b3 = (Button) findViewById(R.id.btnMult);
        b4 = (Button) findViewById(R.id.btnDiv);
        b5 = (Button) findViewById(R.id.btnClear);
        tv1 = (TextView) findViewById(R.id.tv1);

        b1.setOnClickListener(new OnClickListener()
        { @Override
        public void onClick(View v)
        {
            try
            {
                String s1 = t1.getText().toString();
                String s2 = t2.getText().toString();
                n1 = Integer.parseInt(s1);
                n2 = Integer.parseInt(s2);
                int sum = n1 + n2;
                tv1.setText("Addition ="+sum);
            }
            catch (NumberFormatException e) { }
        }
        });

        b2.setOnClickListener(new OnClickListener()
        { @Override
        public void onClick(View v)
        {
            try
            {
                String s1 = t1.getText().toString();
                String s2 = t2.getText().toString();
                n1 = Integer.parseInt(s1);
                n2 = Integer.parseInt(s2);
                int sub = n1 - n2;
                tv1.setText("Subtraction ="+sub);
            }
            catch (NumberFormatException e) { }
        }
        });

        b3.setOnClickListener(new OnClickListener()
        { @Override
        public void onClick(View v)
```

```
        {
            try
            {
                String s1 = t1.getText().toString();
                String s2 = t2.getText().toString();
                n1 = Integer.parseInt(s1);
                n2 = Integer.parseInt(s2);
                int mul = n1 * n2;
                tv1.setText("Multiplication ="+mul);
            }
            catch (NumberFormatException e) { }
        }
        });

        b4.setOnClickListener(new OnClickListener()
        { @Override
        public void onClick(View v)
        {
            try
            {
                String s1 = t1.getText().toString();
                String s2 = t2.getText().toString();
                n1 = Integer.parseInt(s1);
                n2 = Integer.parseInt(s2);
                int div = n1 / n2;
                tv1.setText("Division ="+div);
            }
            catch (NumberFormatException e) { }
        }
        });

        b5.setOnClickListener(new OnClickListener()
        { @Override
        public void onClick(View v)
        {
            t1.setText(" ");
            t2.setText(" ");
            tv1.setText(" ");
        }
        });
    }
}
```

Pract 5-B AppBar :

The *app bar*, also known as the *action bar*, is one of the most important design elements in your app's activities, because it provides a visual structure and interactive elements that are familiar to users. Using the app bar makes your app consistent with other Android apps, allowing users to quickly understand how to operate your app and have a great experience.

The key functions of the app bar are as follows:



- A dedicated space for giving your app an identity and indicating the user's location in the app.
- Access to important actions in a predictable way, such as search.
- Support for navigation and view switching (with tabs or drop-down lists).

This class describes how to use the v7 appcompat support library's `Toolbar` widget as an app bar. There are other ways to implement an app bar—for example, some themes set up an `ActionBar` as an app bar by default—but using the appcompat `Toolbar` makes it easy to set up an app bar that works on the widest range of devices, and also gives you room to customize your app bar later on as your app develops.

Create New Project AppBar

1. Create AppBar
2. Adding Menu to AppBar
3. Handling event

Menus

- Menus are a common user interface component in many types of applications. To provide a familiar and consistent user experience, you should use the Menu APIs to present user actions and other options in your activities.

Options menu and app bar

- The options menu is the primary collection of menu items for an activity. It's where you should place actions that have a global impact on the app, such as "Search," "Compose email," and "Settings."

Context menu and contextual action mode

- A context menu is a floating menu that appears when the user performs a long-click on an element. It provides actions that affect the selected content or context frame.
- The contextual action mode displays action items that affect the selected content in a bar at the top of the screen and allows the user to select multiple items.

1. Create AppBar

Remove the default Action Bar.

Goto values→ themes.xml

Change

```
<style name="Theme.Pract5A"
parent="Theme.MaterialComponents.DayNight.DarkActionBar">
```

To

```
<style name="Theme.AppBar"
parent="Theme.MaterialComponents.DayNight.NoActionBar">
```

2. Add a `Toolbar` to the activity's layout.

Goto resources →layout

Add new layout(toobar_layout, set root element to-androidx.appcompat.widget.Toolbar)

toolbar_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.appcompat.widget.Toolbar
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
    android:background="@color/black"
    android:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"
    app:popupTheme="@style/ThemeOverlay.AppCompat.Light"
    android:id="@+id/t1">
</androidx.appcompat.widget.Toolbar>
```

3. Include this layout to **activity_main.xml** File.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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=".MainActivity"
    android:orientation="vertical">
    <include layout="@layout/toolbar_layout"/>
</LinearLayout>
```

4. **Setting the toolbar as default bar**

In the activity's `onCreate()` method, call the activity's `setSupportActionBar()` method, and pass the activity's toolbar. This method sets the toolbar as the app bar for the activity.

```
public class MainActivity extends AppCompatActivity {
    private Toolbar t2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        t2=findViewById(R.id.t1);
         setSupportActionBar(t2);
    }
```

Your app now has a basic action bar. By default, the action bar contains just the name of the app and an overflow menu. The options menu initially contains just the **Settings** item. You can add more actions to the action bar and the overflow menu as follows.

5. Add and handle actions

The app bar allows you to add buttons for user actions. This feature lets you put the most important *actions* for the current context right at the top of the app

All action buttons and other items available in the action overflow are defined in an XML menu resource. To add actions to the action bar, create a new XML file in your project's res/menu/ directory.

Add an <item> element for each item you want to include in the action bar, as shown in this code example of a menu XML file:

Add menu to appbar

➔ Right clic on res folder add new android resources file

➔ Name it app_bar_menu

➔ Set resource type-menu

app_bar_menu.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto" >
    <item
        android:id="@+id/action_search"
        android:title="Search"
        android:icon="@drawable/ic_search"
        app:showAsAction="ifRoom"/>
    <item
        android:id="@+id/action_setting"
        android:title="Setting"
        android:icon="@drawable/ic_setting"
        app:showAsAction="ifRoom"/>
    <item
        android:id="@+id/action_about"
        android:title="About"
        app:showAsAction="never"/>
    <item
        android:id="@+id/action_logout"
        android:title="Logout"
        app:showAsAction="never"/>
</menu>
```

The `app:showAsAction` attribute specifies whether the action should be shown as a button on the app bar. If you set `app:showAsAction="ifRoom"` , the action is displayed as a button if there is room in the app bar for it; if there is not enough room, excess actions are sent to the overflow menu. If you set `app:showAsAction="never"`, the action is always listed in the overflow menu, not displayed in the app bar.

6. To add this Menu to AppBar(Action Bar)

When the user selects one of the app bar items, the system calls your activity's `onOptionsItemSelected()` callback method, and passes a `MenuItem` object to indicate which item was clicked. In your implementation of `onOptionsItemSelected()`, call the `MenuItem.getItemId()` method to determine which item was pressed. The ID returned matches the value you declared in the corresponding `<item>` element's `android:id` attribute.

```
package com.example.appbar;

import androidx.annotation.NonNull;
```

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.widget.Toast;

import androidx.appcompat.widget.Toolbar;

public class MainActivity extends AppCompatActivity {
    private Toolbar t2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        t2=findViewById(R.id.t1);
        setSupportActionBar(t2);
    }

    @Override
     public boolean onCreateOptionsMenu(Menu menu) {
         MenuInflater mi=getMenuInflater();
         mi.inflate(R.menu.app_bar_menu,menu);
         return super.onCreateOptionsMenu(menu);
     }

     @Override
     public boolean onOptionsItemSelected(@NonNull MenuItem
item) {
         switch (item.getItemId()) {

             case R.id.action_search:
                 Toast.makeText(this, "search menu click",
Toast.LENGTH_SHORT).show();
                 return true;
             case R.id.action_setting:
                 Toast.makeText(this, "setting menu click",
Toast.LENGTH_SHORT).show();
                 return true;
             case R.id.action_about:
                 Toast.makeText(this, "About menu click",
Toast.LENGTH_SHORT).show();
                 return true;
             case R.id.action_logout:
                 Toast.makeText(this, "Logout click",
Toast.LENGTH_SHORT).show();
                 return true;
             default:
                 return super.onOptionsItemSelected(item);
         }
    }
}
```

add ic_searcch.xml and ic_setting.xml to drawable folder