Andorid Programming

Week 5

Mina Jung

EECS, Syracuse University

Spring 2017

Part I

ToolBar/AppBar/ActionBar(Deprecated)

Outline I

Outline II

1. Toolbar(Toolbar widget)
   - generalization of the Action Bar
   - more control and flexiblity
     - easier to position, animate and control
   - Toolbar is a view included in a layout like other views
   - multiple distinct Toolbar elements can be defined in the same activity
2. Use a Toolbar as an Action Bar
   - ensure the AppCompat-v7 support library is added to your gradle
3. Use a standalone Toolbar
   - showing multiple toolbars on the screen

- spanning only part of the width, and so on.

4. Click! Toolbars and Appbars with Material Design

- One of the most important design elements in your app's activities
    - 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)

1. Make sure the activity extends AppCompatActivity

```
public class MyActivity extends AppCompatActivity {
  // ...
}
```

2. Set the <application> element to use one of appcompat's NoActionBar themes (in Manifest file)
   - Prevent the app from using the native ActionBar class

```
<application
    android:theme="@style/Theme.AppCompat.Light.NoActionBar"
    />
```

3. Add a Toolbar to the activity's layout
   - Position the toolbar at the top of the activity's layout

```
<android.support.v7.widget.Toolbar
   android:id="@+id/my_toolbar"
   android:layout_width="match_parent"
   android:layout_height="?attr/actionBarSize"
   android:background="?attr/colorPrimary"
   android:elevation="4dp"
   android:theme="@style/ThemeOverlay.AppCompat.ActionBar"
   app:popupTheme="@style/ThemeOverlay.AppCompat.Light"/>
```

4. Set the toolbar as the app bar for the activity in the activity's onCreate() method

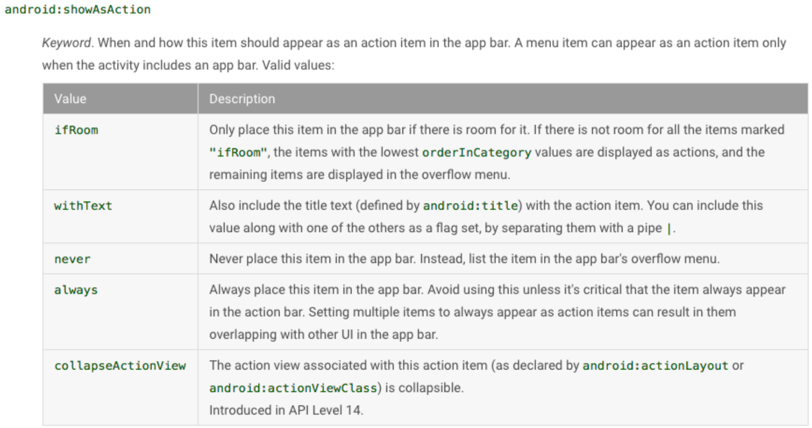
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_my);
    Toolbar myToolbar = (Toolbar) findViewById(R.id.my_toolbar);
    setSupportActionBar(myToolbar);
}
```

5. By default, the action bar contains just the name of the app and an overflow menu
6. To use the ActionBar utility methods, call the activity's getSupportActionBar() method

1. Add Action Buttons and Others in the Overflow memnu
    - defined in an XML menu resource

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <item
        android:id="@+id/action_favorite"
        android:icon="@drawable/ic_favorite_black_48dp"
        android:title="@string/action_favorite"
        app:showAsAction="ifRoom"/>

    <item android:id="@+id/action_settings"
          android:title="@string/action_settings"
          app:showAsAction="never"/>
</menu>
```


`android:showAsAction`

Keyword. When and how this item should appear as an action item in the app bar. A menu item can appear as an action item only when the activity includes an app bar. Valid values:

Value	Description
`ifRoom`	Only place this item in the app bar if there is room for it. If there is not room for all the items marked `"ifRoom"`, the items with the lowest `orderInCategory` values are displayed as actions, and the remaining items are displayed in the overflow menu.
`withText`	Also include the title text (defined by `android:title`) with the action item. You can include this value along with one of the others as a flag set, by separating them with a pipe `|`.
`never`	Never place this item in the app bar. Instead, list the item in the app bar's overflow menu.
`always`	Always place this item in the app bar. Avoid using this unless it's critical that the item always appear in the action bar. Setting multiple items to always appear as action items can result in them overlapping with other UI in the app bar.
`collapseActionView`	The action view associated with this action item (as declared by `android:actionLayout` or `android:actionViewClass`) is collapsible. Introduced in API Level 14.

2. Inflate the menu in the Activity

```java
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.my_activity_actions, menu);

    return true;
}
```

3. Respond to Actions
   - When one of the app bar items is selected, the activity's onOptionsItemSelected() callback method is called with a MenuItem object

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
  switch (item.getItemId()) {
    case R.id.action_settings:
      // User chose the "Settings" item, show the app settings UI...
      return true;

    case R.id.action_favorite:
      // User chose the "Favorite" action, mark the current item
      // as a favorite...
      return true;

    default:
      // If we got here, the user's action was not recognized.
      // Invoke the superclass to handle it.
      return super.onOptionsItemSelected(item);
    }
}
```

- Create Custom Style

```
<style name="ToolbarTheme"
    parent="@style/ThemeOverlay.AppCompat.Dark.ActionBar">
  <!-- android:textColorPrimary is the color of the title text in
      the Toolbar  -->
  <item
      name="android:textColorPrimary">@android:color/holo_blue_light</item>
  <!-- actionMenuTextColor is the color of the text of action
      (menu) items  -->
  <item
      name="actionMenuTextColor">@android:color/holo_green_light</item>
  <!-- Tints the input fields like checkboxes and text fields -->
  <item name="colorAccent">@color/cursorAccent</item>
  <!-- Applies to views in their normal state. -->
  <item name="colorControlNormal">@color/controlNormal</item>
  <!-- Applies to views in their activated state (i.e checked or
      switches) -->
  <item name="colorControlActivated">@color/controlActivated</item>
  <!-- Applied to framework control highlights (i.e ripples or list
      selectors) -->
  <item name="colorControlHighlight">@color/controlActivated</item>

  <!-- Enable these below if you want clicking icons to trigger a
      ripple effect -->
  <!--
```

```
    <item
         name="selectableItemBackground">?android:selectableItemBackground</it
    <item
         name="selectableItemBackgroundBorderless">?android:selectableItemBack
    -->
  </style>

  <!-- This configures the styles for the title within the Toolbar
      -->
  <style name="Toolbar.TitleText"
      parent="TextAppearance.Widget.AppCompat.Toolbar.Title">
      <item name="android:textSize">21sp</item>
      <item name="android:textStyle">italic</item>
  </style>
```

- Apply the custom style to Toolbar

```
<android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:minHeight="?attr/actionBarSize"
    android:background="?attr/colorPrimary"
    android:theme="@style/ToolbarTheme"
    app:titleTextAppearance="@style/Toolbar.TitleText"
    app:popupTheme="@style/ThemeOverlay.AppCompat.Light"
/>
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  // ...
  // Find the toolbar view and set as ActionBar
  Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
  setSupportActionBar(toolbar);
  // ...
  // Display icon in the toolbar
  getSupportActionBar().setDisplayShowHomeEnabled(true);
  getSupportActionBar().setLogo(R.mipmap.ic_launcher);
  getSupportActionBar().setDisplayUseLogoEnabled(true);
  // ...
}
```

- You may need to adjust some margins to properly display icon

```
<android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
    android:minHeight="?attr/actionBarSize"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:titleTextColor="@android:color/white"
    android:background="?attr/colorPrimary">

    <TextView
        android:id="@+id/toolbar_title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Toolbar Title"
        android:textColor="@android:color/white"
        style="@style/TextAppearance.AppCompat.Widget.ActionBar.Title"
        android:layout_gravity="center"
     />

     <ImageView
        android:id="@+id/toolbar_image"/>

</android.support.v7.widget.Toolbar>
```

- If you want to set a new Title with the TextView, need to disable default title

```
// Remove default title text
getSupportActionBar().setDisplayShowTitleEnabled(false);
// Get access to the custom title view
TextView mTitle = (TextView)
     toolbar.findViewById(R.id.toolbar_title);
```

- use CoordinatorLayout

```
<android.support.design.widget.CoordinatorLayout
    android:id="@+id/main_content"
    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">

    <!-- AppBarLayout is a wrapper for a Toolbar in order to apply
         scrolling effects. -->
    <!-- Note that AppBarLayout expects to be the first child
         nested within a CoordinatorLayout -->
    <android.support.design.widget.AppBarLayout
        android:id="@+id/appBar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/ThemeOverlay.AppCompat.ActionBar">

        <!-- Toolbar is the actual app bar with text and the action
             items -->
        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
```

```
                android:background="?attr/colorPrimary"
                app:layout_scrollFlags="scroll|enterAlways" />
        </android.support.design.widget.AppBarLayout>

        <!-- This could also be included from another file using the
             include tag -->
        <!-- i.e `res/layout/content_main.xml` -->
        <!-- `app:layout_behavior` is set to a pre-defined standard
             scrolling behavior -->
        <android.support.v7.widget.RecyclerView
            android:id="@+id/my_recycler_view"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:clipToPadding="false"
            app:layout_behavior="@string/appbar_scrolling_view_behavior"
                />

    </android.support.design.widget.CoordinatorLayout>
```

1. Create Menu for Fragment

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/trash_can"
        android:icon="@drawable/trash_can"
        app:showAsAction="ifRoom"
        android:title="Compose">
    </item>
</menu>
```

2. Enable OptionMenu (inside Fragment)

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup
     container, Bundle savedInstanceState) {
    setHasOptionsMenu(true);
    return inflater.inflate(R.layout.fragment_interest, container,
         false);
}
```

3. Add Action Items by inflating Fragment's menu(inside Fragment)

   ```
   @Override
   public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {
       super.onCreateOptionsMenu(menu, inflater);
       inflater.inflate(R.menu.menu_frag, menu);
   }
   ```

   ◊ **Question:** What happens when device rotate?

4. Handle Actions (inside Fragment)

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
  switch (item.getItemId()) {
    case R.id.trash_can:
      Toast.makeText(getActivity(), "Clip clicked!",
           Toast.LENGTH_SHORT).show();
      return true;
  }
  return super.onOptionsItemSelected(item);
}
```

◊ **Question:** If the activity's onOptionItemSelected method has a hanling routine for the same action item, which one will perform?

- An **action view** is an action that provides rich functionality within the app bar
  - example: search action view allows the user to type their search text in the app bar, without having to change activities or fragments

- An **action provider** is an action with its own customized layout. The action initially appears as a button or menu item, but when the user clicks the action, the action provider controls the action's behavior in any way you want to define. For example, the action provider might respond to a click by displaying a menu.

1. Add an Action View
   - create an <item> element in the toolbar's menu resource

```
<item android:id="@+id/action_search"
     android:title="@string/action_search"
     android:icon="@drawable/ic_search"
     app:showAsAction="ifRoom|collapseActionView"
     app:actionViewClass="android.support.v7.widget.SearchView" />
```

2. Configure the action in your activity's onCreateOptionsMenu() callback

```
@Override
public boolean onCreateOptionsMenu(Menu menu, MenuInflater
     inflater) {

  if(menu.findItem(R.id.action_search) == null)
    inflater.inflate(R.menu.menu_action_view, menu);

  SearchView search = (SearchView)
       menu.findItem(R.id.action_search).getActionView();

  // Configure the search info and add any event listeners...
  if(search != null) {
    search.setOnQueryTextListener(new
         SearchView.OnQueryTextListener() {
      @Override
      public boolean onQueryTextSubmit(String query) {
        int pos = movieData.findFirst(query);
        if(pos >= 0)
          mRecyclerView.scrollToPosition(pos);
        return true;
      }

      @Override
      public boolean onQueryTextChange(String query) {
        return true;
      }
    });
  }

  return super.onCreateOptionsMenu(menu, inflater);
}
```

3. Responding to action view expansion

```
@Override
public boolean onCreateOptionsMenu(Menu menu, MenuInflater
     inflater) {
    inflater.inflate(R.menu.menu_action_view, menu);
    // ...

    // Define the listener
    OnActionExpandListener expandListener = new
         OnActionExpandListener() {
        @Override
        public boolean onMenuItemActionCollapse(MenuItem item) {
            // Do something when action item collapses
            return true;  // Return true to collapse action view
        }

        @Override
```

```
            public boolean onMenuItemActionExpand(MenuItem item) {
                // Do something when expanded
                return true;  // Return true to expand action view
            }
        };

        // Get the MenuItem for the action item
        MenuItem actionMenuItem = menu.findItem(R.id.action_search);

        // Assign the listener to that action item
        MenuItemCompat.setOnActionExpandListener(actionMenuItem,
            expandListener);

        // Any other things you have to do when creating the options
            menu

        return true;
    }
```

- create an <item> element in the toolbar's menu resource

```
<item android:id="@+id/action_share"
    android:title="@string/share"
    app:showAsAction="ifRoom"
    app:actionProviderClass="android.support.v7.widget.ShareActionProvider"
```

- Set up ShareActionProvider

```
@Override
public void onCreateOptionsMenu(Menu menu, MenuInflater inflater){
  inflater.inflate(R.menu.menu_detail, menu);

  MenuItem shareItem = menu.findItem(R.id.action_share);
  mShareActionProvider = (ShareActionProvider)
      MenuItemCompat.getActionProvider(shareItem);


  Intent intentShare = new Intent(Intent.ACTION_SEND);
  intentShare.setType("text/plain");
  intentShare.putExtra(Intent.EXTRA_TEXT, (String)
      movie.get("name"));

  if(mShareActionProvider != null && intentShare != null)
    mShareActionProvider.setShareIntent(intentShare);

  super.onCreateOptionsMenu(menu, inflater);
}
```

1. Menu resource

```
<menu 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">
    <item
        android:id="@+id/action_delete"
        android:icon="@drawable/ic_menu_delete"
        app:showAsAction="ifRoom|withText"
        app:itemIconTint="@color/white"
        android:title="Delete movie" />
    <item
        android:id="@+id/action_duplicate"
        android:icon="@drawable/ic_menu_duplicate"
        app:showAsAction="ifRoom|withText"
        app:itemIconTint="@color/colorAccent"
        android:title="Duplicate movie" />
</menu>
```

2. Register a click(long-press) event to a view (inside Fragment)

```
myAdapter.setOnItemClickListener(new
     MyRecyclerAdapter.OnItemClickListener(){
  // ...
  @Override
  public void onItemLongClick(View view, int position) {
    getActivity().startActionMode(new ActionBarCallBack(position));
  }

  // ...
});
```

3. Implement ActionBarCallback class (inner class of Fragment)

```
class ActionBarCallBack implements ActionMode.Callback{
  int position;

  public ActionBarCallBack(int position){
   this.position = position;
  }

  @Override
  public boolean onCreateActionMode(ActionMode mode, Menu menu) {
    mode.getMenuInflater().inflate(R.menu.menu_popup, menu);

    return true;
  }

  @Override
  public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
    HashMap movie = (HashMap)movieData.getItem(position);
    mode.setTitle((String)movie.get("name"));
    return false;
  }

  @Override
  public boolean onActionItemClicked(ActionMode mode, MenuItem
       item) {
  int id = item.getItemId();
```

```
      switch (id){
        case R.id.action_delete:
          movieData.deleteItem(position);
          myAdapter.notifyItemRemoved(position);
          mode.finish();
          break;
        case R.id.action_duplicate:
          movieData.addItem(position+1,(HashMap)
               ((HashMap)movieData.getItem(position)).clone());
          myAdapter.notifyItemInserted(position+1);
          mode.finish();
          break;
        default:
          break;
      }

      return false;
    }

    @Override
    public void onDestroyActionMode(ActionMode mode) {
    }
  }
```

- Make a TextView's content selectable

```
<TextView
    android:layout_weight="4"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:textIsSelectable="true"
    android:id="@+id/mDes" />
```

1. Make Menu in XML (use same memu)

2. Register Event Listener and Set up Popup menu (inside Fragment)

```
myAdapter.setOnItemClickListener(new
     MyRecyclerAdapter.OnItemClickListener(){
  // ...
@Override
public void onOverflowMenuClick(View view, final int position) {
  PopupMenu popup = new PopupMenu(getActivity(), view);
  popup.setOnMenuItemClickListener(new
       PopupMenu.OnMenuItemClickListener() {
    @Override
    public boolean onMenuItemClick(MenuItem item) {
      int id = item.getItemId();
      switch (id){
        case R.id.action_delete:
          movieData.deleteItem(position);
          myAdapter.notifyItemRemoved(position);
          return true;
        case R.id.action_duplicate:
          movieData.addItem(position+1,(HashMap)
              ((HashMap)movieData.getItem(position)).clone());
          myAdapter.notifyItemInserted(position+1);
          return true;
      }
      return false;
    }
});

MenuInflater menuInflater = popup.getMenuInflater();
menuInflater.inflate(R.menu.menu_popup, popup.getMenu());
popup.show();

  // ...
});
```

3. Register Event Handler (inside Adapter)

```
public interface OnItemClickListener{
    public void onItemClick(View view, int position);
    public void onItemLongClick(View view, int position);
    public void onOverflowMenuClick(View view, int position);
}

public void setOnItemClickListener(final OnItemClickListener
     mItemClickListener){
    this.mItemClickListener = mItemClickListener;
}
```

4. Trigger Event Handler in a View (inside ViewHolder)

```
public ViewHolder(View view) {
  super(view);
  // ...
  overFlow = (ImageView) view.findViewById(R.id.overFlow);

  // ...

  if(overFlow != null){
    overFlow.setOnClickListener( new View.OnClickListener(){
      @Override
      public void onClick(View v) {
        if(mItemClickListener != null){
          mItemClickListener.onOverflowMenuClick(v,
               getAdapterPosition());
        }
      }
    });
  }
}
```

- Not App Bar (ActionBar)
- need to inflate toolbar's menu
- need to register item cick listener

1. Make Toolbar's layout

2. Inflate the menu

```
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_bottom_toolbar);
  topT = (Toolbar) findViewById(R.id.toolbar);
  setSupportActionBar(topT);


  bottomT = (Toolbar) findViewById(R.id.bot_toolbar);
  bottomT.inflateMenu(R.menu.bottom_toolbar);
  setupBottomToolbarItemSelected();

  // ...
}
```

3. Register and Handler Item Click Event

```
private void setupBottomToolbarItemSelected(){
    bottomT.setOnMenuItemClickListener(new
         Toolbar.OnMenuItemClickListener(){

        @Override
        public boolean onMenuItemClick(MenuItem item) {
            int id = item.getItemId();

            switch (id){
                case R.id.bottom_action1:
                    Toast.makeText(getApplicationContext(),
                         "Clicked Bottom Action
                         1",Toast.LENGTH_SHORT).show();
                    return true;
                case R.id.bottom_action2:
                    Toast.makeText(getApplicationContext(),
                         "Clicked Bottom Action
                         2",Toast.LENGTH_SHORT).show();
                    return true;
                default:
                    break;
            }
            return false;
        }
    });
```

```
        bottomT.setNavigationIcon(R.drawable.ic_close_bottom);
        bottomT.setNavigationOnClickListener(new
             View.OnClickListener(){

            @Override
            public void onClick(View v) {
                bottomT.setVisibility(View.GONE);
            }
        });

    }
```

Part II

Navigation Drawer

Outline I

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v4.widget.DrawerLayout
     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:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:openDrawer="start">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
    <include
        layout="@layout/app_bar_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />


    <FrameLayout
        android:layout_marginTop="100dp"
        android:layout_below="@+id/toolbar"
        android:id="@+id/me_container"
```

```
            android:layout_width="match_parent"
            android:layout_height="match_parent">
        </FrameLayout>

        </RelativeLayout>

        <android.support.design.widget.NavigationView
            android:id="@+id/nav_view"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_gravity="start"
            android:fitsSystemWindows="true"
            app:headerLayout="@layout/nav_header_main"
            app:menu="@menu/activity_main_drawer" />


    </android.support.v4.widget.DrawerLayout>
```

```
<?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="wrap_content"
    android:background="@drawable/nav_back"
    android:gravity="bottom"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:theme="@style/ThemeOverlay.AppCompat.Dark">

    <com.mikhaellopez.circularimageview.CircularImageView
        android:paddingTop="20dp"
        android:layout_gravity="center_horizontal"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:src="@drawable/olaf"
        app:civ_border_color="#EEEEEE"
        app:civ_border_width="4dp"
        app:civ_shadow="true"
        app:civ_shadow_radius="10"
        app:civ_shadow_color="#8BC34A"/>
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="@dimen/nav_header_vertical_spacing"
        android:text="Mina Jung"
        android:textAppearance="@style/TextAppearance.AppCompat.Body1"
            />

    <TextView
        android:id="@+id/myEmail"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="mijung@syr.edu" />

</LinearLayout>
```

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

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_aboutme"
            android:icon="@drawable/ic_person_outline_black_24dp"
            android:title="About Me" />
        <item
            android:id="@+id/nav_layout"
            android:icon="@drawable/ic_menu_gallery"
            android:title="Check Layout Manager" />
        <item
            android:id="@+id/nav_recycler"
            android:icon="@drawable/ic_listview"
            android:title="Movie RecyclerView " />
        <item
            android:id="@+id/nav_toolbar"
            android:icon="@drawable/ic_b_toolbar"
            android:title="Toolbars with Movie" />
    </group>

    <item android:title="Communicate">
        <menu>
            <item
                android:id="@+id/nav_share"
```

```
                    android:icon="@drawable/ic_menu_share"
                    android:title="Share" />
                <item
                    android:id="@+id/nav_send"
                    android:icon="@drawable/ic_menu_send"
                    android:title="Send" />
            </menu>
        </item>

    </menu>
```

```
public class MainActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // Toolbar
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);


        // Navigation Drawer
        DrawerLayout drawer = (DrawerLayout)
            findViewById(R.id.drawer_layout);
        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
                this, drawer, toolbar,
                    R.string.navigation_drawer_open,
                    R.string.navigation_drawer_close);
        drawer.setDrawerListener(toggle);
        toggle.syncState();

        NavigationView navigationView = (NavigationView)
            findViewById(R.id.nav_view);
        navigationView.setNavigationItemSelectedListener(this);
    }

    @Override
    public void onBackPressed() {
        DrawerLayout drawer = (DrawerLayout)
             findViewById(R.id.drawer_layout);
        if (drawer.isDrawerOpen(GravityCompat.START)) {
            drawer.closeDrawer(GravityCompat.START);
        } else {
            super.onBackPressed();
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if
             it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Handle action bar item clicks here. The action bar will
        // automatically handle clicks on the Home/Up button, so
             long
        // as you specify a parent activity in AndroidManifest.xml.
        int id = item.getItemId();

        //noinspection SimplifiableIfStatement
        switch(id){
            case R.id.action_settings:
                Toast.makeText(getApplicationContext(),"Action
                    Setting",Toast.LENGTH_SHORT).show();
                return true;
            case R.id.action_compose:
                Toast.makeText(getApplicationContext(),"Action
                    Email Compose",Toast.LENGTH_SHORT).show();
                return true;
            case R.id.action_help:
                Toast.makeText(getApplicationContext(),"Action
                    Help",Toast.LENGTH_SHORT).show();
                return true;
            case R.id.action_activity:
                Toast.makeText(getApplicationContext(),"Action
                    Activity",Toast.LENGTH_SHORT).show();
                return true;
        }


        return super.onOptionsItemSelected(item);
    }
```

```
@SuppressWarnings("StatementWithEmptyBody")
@Override
public boolean onNavigationItemSelected(MenuItem item) {
    // Handle navigation view item clicks here.
    int id = item.getItemId();

    if (id == R.id.nav_aboutme) {
        // AboutMe Fragment
        MeFragment fr = MeFragment.newInstance();

        if (findViewById(R.id.me_container) != null && fr !=
             null) {
            getSupportFragmentManager().beginTransaction().add(R.id.me_
                 fr).commit();
        }
    }
    else if (id == R.id.nav_layout) {
        // New Activity
        Intent intent1 = new Intent(MainActivity.this,
             LayoutManagerActivity.class);
        startActivity(intent1);

    }
    else if (id == R.id.nav_recycler) {
        // New Activity
```

```
                Intent intent2 = new Intent(MainActivity.this,
                    RecyclerViewActivity.class);
                startActivity(intent2);
            }
            else if (id == R.id.nav_toolbar) {
                // New Activity
                Intent intent3 = new Intent(MainActivity.this,
                    BottomToolbarActivity.class);
                startActivity(intent3);
            }
            else if (id == R.id.nav_share) {

            }
            else if (id == R.id.nav_send) {

            }

            DrawerLayout drawer = (DrawerLayout)
                findViewById(R.id.drawer_layout);
            drawer.closeDrawer(GravityCompat.START);
            return true;
        }
    }
```