**PRACTICAL # 08**

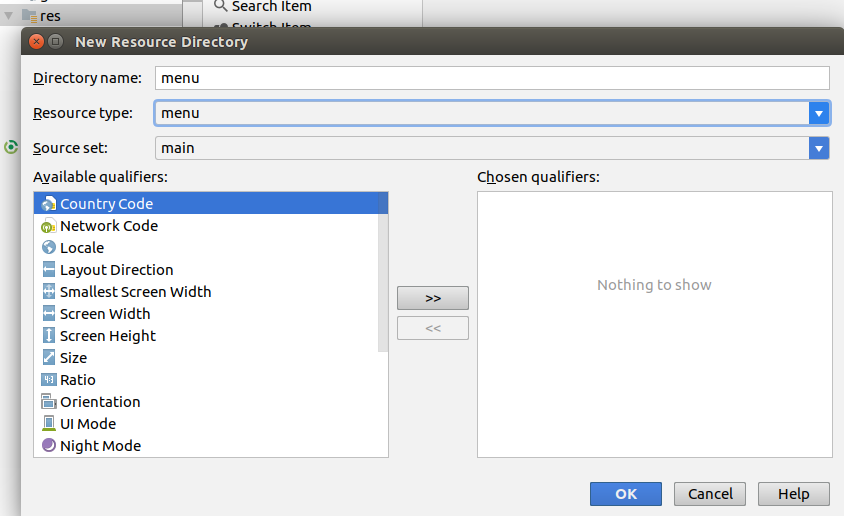
**OBJECT:**

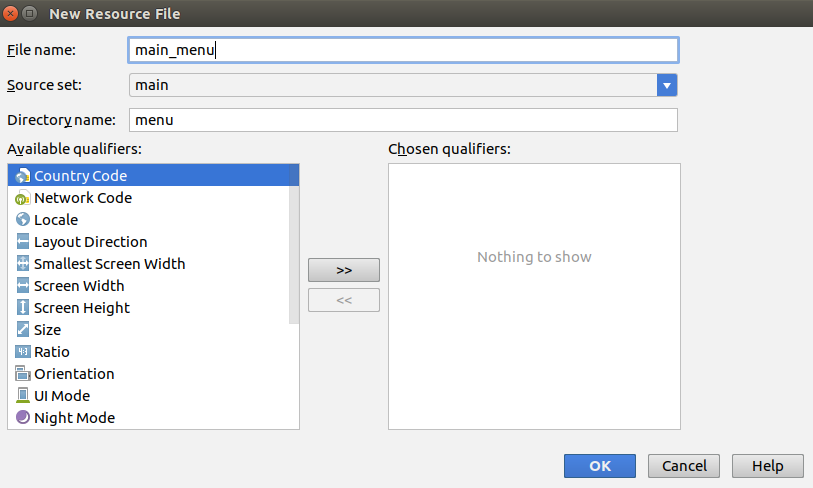
Working with Android Menus

**THEORY:**

Android supports menus.

To add menu resource directory, right click res folder, then New and Android Resource Directory. The following menu will appear. Select menu as Resource type and press OK button.

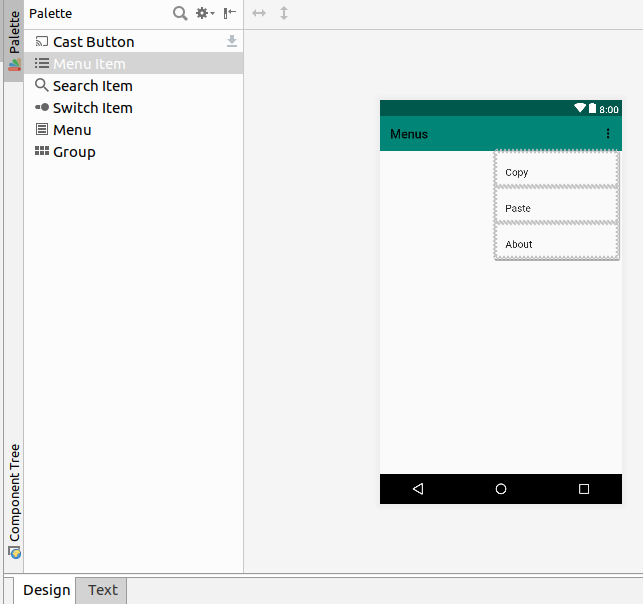
****

**After adding the menu resource folder, right click the folder → New → Menu Resource File**

Name the file and press OK. The menu resource XML file will be added.

**Adding Menu Items:**

Drag and drop menu items on the UI.



The xml behind the design is:

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

<**menu xmlns:app="http://schemas.android.com/apk/res-auto"**

**xmlns:android="http://schemas.android.com/apk/res/android"**>

<**item android:title="Copy" android:id="@+id/copyItem"**/>

<**item android:title="Paste" android:id="@+id/pasteItem"**/>

<**item android:title="About" android:id="@+id/about"**/>

</**menu**>

**Inflating Menu:**

After designing the menu items, open the activity code to inflate the menu with the code below.

@Override

**public boolean** onCreateOptionsMenu(Menu menu){

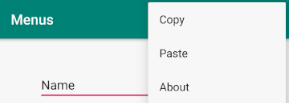
MenuInflater inflater = getMenuInflater();

inflater.inflate(R.menu.***main\_menu***, menu);

**return true**;

}

The inflate method takes first argument of the menu file to be inflated.





**Binding Item Click Events:**

After inflating the menu, bind item click events by overriding the onOptionsItemSelected method.

@Override

**public boolean** onOptionsItemSelected(MenuItem item) {

*// Handle item selection*

**switch** (item.getItemId()) {

**case** R.id.***copyItem***:

copyToClipBoard();

**return true**;

**case** R.id.***pasteItem***:

pastClipBoard();

**return true**;

**default**:

**return super**.onOptionsItemSelected(item);

}

}

**Using Clipboard to copy and paste text:**

*// Copy EditCopy text to the ClipBoard*

**private void** copyToClipBoard()

{

ClipboardManager clipboard = (ClipboardManager) getSystemService(***CLIPBOARD\_SERVICE***);

ClipData clip = ClipData.*newPlainText*(**"usersname"**, **mEditText**.getText().toString());

clipboard.setPrimaryClip(clip);

}

**private void** pastClipBoard(){

android.content.ClipboardManager clipboard = (android.content.ClipboardManager) getSystemService(Context.***CLIPBOARD\_SERVICE***);

android.content.ClipData.Item item = clipboard.getPrimaryClip().getItemAt(0);

**if** (item.getText() != **null**) {

**mTextView**.setText(item.getText());

}

}

**Creating Nested Sub-Menu:**

Sub-menus can also be created simply by nested another menu element within an item:

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

<**menu xmlns:app="http://schemas.android.com/apk/res-auto"**

**xmlns:android="http://schemas.android.com/apk/res/android"**>

<**item android:id="@+id/file"**

**android:icon="@drawable/ic\_launcher\_background"**

**android:title="File"** >

*<!-- "file" submenu -->*

<**menu**>

<**item android:id="@+id/create\_new"**

**android:title="Create New"** />

<**item android:id="@+id/open\_file"**

**android:title="Open File"** />

</**menu**>

</**item**>

<**item android:title="Copy" android:id="@+id/copyItem"**/>

<**item android:title="Paste" android:id="@+id/pasteItem"**/>

<**item android:title="About" android:id="@+id/about"**/>

</**menu**>

**Handling submenu item click events:**

Handling submenu events is similarly done by handling appropriate case.

@Override

**public boolean** onOptionsItemSelected(MenuItem item) {

*// Handle item selection*

**switch** (item.getItemId()) {

**case** R.id.***create\_new***:

Toast.*makeText*(MainActivity.**this**, **"create file"**, Toast.***LENGTH\_LONG***).show();

**return true**;

**case** R.id.***open\_file***:

Toast.*makeText*(MainActivity.**this**, **"open file"**, Toast.***LENGTH\_LONG***).show();

**return true**;

**case** R.id.***copyItem***:

copyToClipBoard();

**return true**;

**case** R.id.***pasteItem***:

pastClipBoard();

*//Toast.makeText(MainActivity.this, "new", Toast.LENGTH\_LONG).show();*

**return true**;

**case** R.id.***about***:

Toast.*makeText*(MainActivity.**this**, **"about"**, Toast.***LENGTH\_LONG***).show();

**return true**;

**default**:

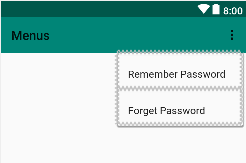
**return super**.onOptionsItemSelected(item);

}

}

**Creating Context Menu:**

Context menu is created the same way as the standard menu.



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

<**menu xmlns:app="http://schemas.android.com/apk/res-auto"**

**xmlns:android="http://schemas.android.com/apk/res/android"**>

<**item android:title="Remember Password" android:id="@+id/remember"**/>

<**item android:title="Forget Password" android:id="@+id/forget"**/>

</**menu**>

**Registering Context Menu:**

Context menu needs to be registered for every view that needs it. The method to register context menu is registerForContextMenu(**view**)

The code below registers two textviews for context menus.

@Override

**protected void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_main***);

**mEditText** = findViewById(R.id.***editText2***);

*//get a reference to the view for pressing*

**mTextView** = findViewById(R.id.***textViewUserName***);

**mTextViewPass** = findViewById(R.id.***textViewPassword***);

*//register views for context menus*

registerForContextMenu(**mTextView**);

registerForContextMenu(**mTextViewPass**);

}

**Inflating Context Menu:**

Inflating context menu is similar to inflating a standard menu. Here we inflate different context menu for the two textViews.

**@Override**

**public void** onCreateContextMenu(ContextMenu menu, View v,

ContextMenu.ContextMenuInfo menuInfo) {

**super**.onCreateContextMenu(menu, v, menuInfo);

MenuInflater inflater = getMenuInflater();

**if**(v.getId() == R.id.***textViewUserName***) {

inflater.inflate(R.menu.***context\_menu***, menu);

}

**else if** (v.getId() == R.id.***textViewPassword***) {

inflater.inflate(R.menu.***context\_menu\_password***, menu);

}

}

**Handling Context Menu Item Click Events:**

The code below handles each item click for context menu.

@Override

**public boolean** onContextItemSelected(MenuItem item) {

*//handle menu item pressed*

**switch** (item.getItemId()) {

**case** R.id.***remember***:

Toast.*makeText*(MainActivity.**this**, **"Remember Password"**, Toast.***LENGTH\_LONG***).show();

**return true**;

**case** R.id.***forget***:

Toast.*makeText*(MainActivity.**this**, **"Forget Password"**, Toast.***LENGTH\_LONG***).show();

**return true**;

**default**:

**return false**;

}

}

**ACTIVITIES**

**Activity 1**

Create an application with menu and context menu of your own choice.

**REVIEW QUESTIONS**

1. What is the difference between menu and context menu?
2. How do you differentiate between views in context menu?
3. How do you create nested submenu?
4. What are the benefits of using menus?