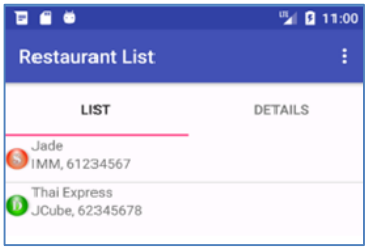
Practical 3: Menu and SQLite Database

In this session, you will learn how to create a Menu to activate a Toast to show information entered in the restaurant detail form. In order to keep the restaurant list data persistent, you will learn how to create a database for the restaurant list using SQLite.

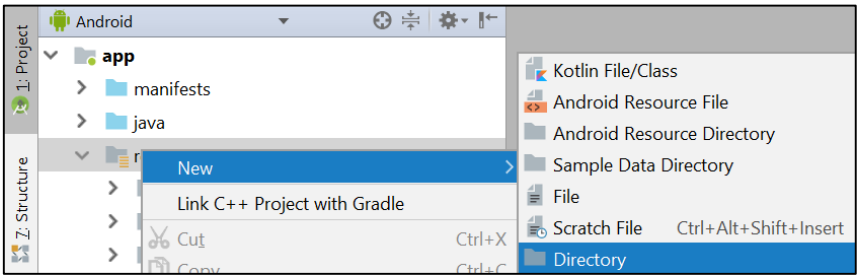


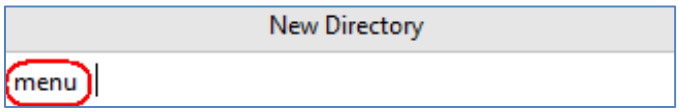
Part I – Create Menu & Detect Menu Item Select

1. In this lab, we will learn how to include an image icon to a **Menu** item and detect the menu item selection.
2. Create a new project with **No Activity** and the following information:

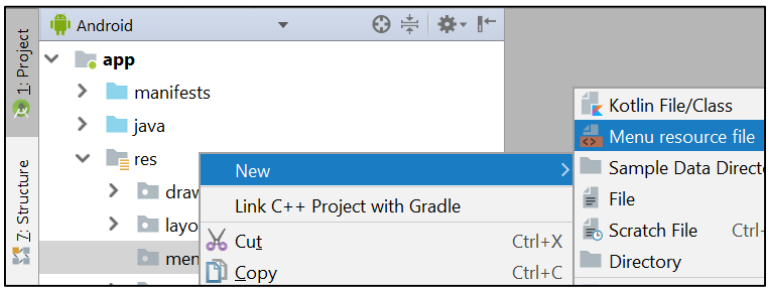
   - **Name**: *Restaurant List*
   - **Package name**: *com.sp.restaurantlist*
   - **Save location**: *C:\MAD\AndroidStudioProjects\Lab3a or D:\MAD\AndroidStudioProjects\Lab3a*
   - **Language:** *Java*
   - **Minimum SDK**: *API 27: Android 8.1 (Oreo)*
   - **Source Language:** *Java*
   - **Build configuration language:** Groovy DSL (build. grade)

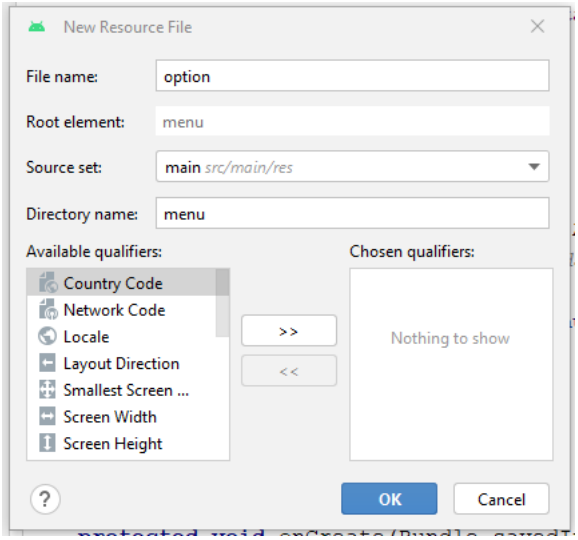
3. Open your **Windows File Explorer** and navigate to your **Android Studio** workspace where all your projects are created
4. Copy *AndroidManifest.xml* file, **java** and **res** folders from **Lab2b\app\src\main** project folder and paste into **Lab3a\app\src\main** folder to overwrite the existing files and folders
5. At this part of the exercise, we will learn how to include an **OPTION MENU** item and limit **MENU** to be shown on '*Details*' tab only
6. At Android Studio, right click on the **res** folder and select **New > Directory** to create folder named menu. Hit "Enter" key.



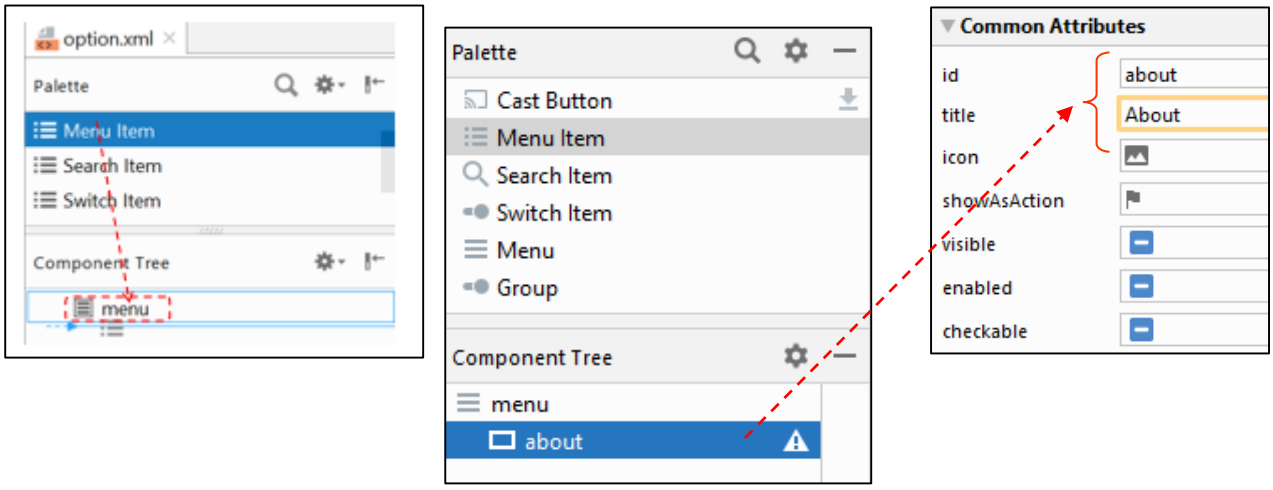


7. Right click on the newly created folder **res/menu** and select **New > Menu resource file** to create a menu file named '*option*'





8. At Design Editor (*option.xml*), drag a **Menu Item** from **Palette** and drop into **menu** under Component Tree. Update the item **id** and **title** to *about* and *About* respectively.



9. Open the *RestaurantList.java* file. Update the program with the following **highlighted** codes to create the ***option* menu** and detect **menu item selection**.

Commonly seen mistakes at this step are the spelling of these 2 methods names. Please take note.

```
public boolean onCreateOptionsMenu(Menu menu)
public boolean onOptionsItemSelected(MenuItem item)
```

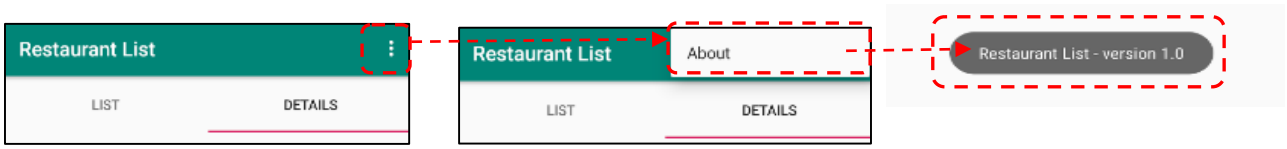
```java
package com.sp.restaurantlist;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.RadioGroup;
import android.widget.TabHost;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.List;

public class RestaurantList extends AppCompatActivity {
    private EditText restaurantName;
    private RadioGroup restaurantTypes;
    private Button buttonSave;
    private EditText restaurantAddress;
    private EditText restaurantTel;

    private List<Restaurant> model = new ArrayList<>();
    private RestaurantAdapter adapter = null;
    private ListView list;
    private TabHost host;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        restaurantName = findViewById(R.id.restaurant_name);
        restaurantTypes = findViewById(R.id.restaurant_types);

        buttonSave = findViewById(R.id.button_save);
        buttonSave.setOnClickListener(onSave);

        restaurantAddress = findViewById(R.id.restaurant_address);
        restaurantTel = findViewById(R.id.restaurant_tel);

        list = findViewById(R.id.restaurants);
        adapter = new RestaurantAdapter();
        list.setAdapter(adapter);

        host = findViewById(R.id.tabHost);
        host.setup();

```

```
        //Tab 1
        TabHost.TabSpec spec = host.newTabSpec("List");
        spec.setContent(R.id.restaurants_tab);
        spec.setIndicator("List");
        host.addTab(spec);

        //Tab 2
        spec = host.newTabSpec("Details");
        spec.setContent(R.id.details_tab);
        spec.setIndicator("Details");
        host.addTab(spec);
        host.setCurrentTab(1);
        list.setOnItemClickListener(onListClick);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {


            getMenuInflater().inflate(R.menu.option, menu);
            return super.onCreateOptionsMenu(menu);


    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if (item.getItemId() == R.id.about) {
            Toast.makeText(this, "Restaurant List - version 1.0", Toast.LENGTH_LONG).show();
        }
        return super.onOptionsItemSelected(item);
    }

    private View.OnClickListener onSave = new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // To read data from restaurantName EditText
            String nameStr = restaurantName.getText().toString();
            // To read data from restaurantAddress EditText
            String addressStr = restaurantAddress.getText().toString();
            // To read data from restaurantTel EditText
            String telStr = restaurantTel.getText().toString();

            String restType = "";
            //To read selection of restaurantTypes RadioGroup
            int radioID = restaurantTypes.getCheckedRadioButtonId();
            if (radioID == R.id.chinese ) {
                restType = "Chinese";
            } else
            if (radioID == R.id.western ) {
                restType = "Western";
            } else
            if (radioID == R.id.indian ) {
                restType = "Indian";
            } else
            if (radioID == R.id.indonesian ) {
                restType = "Indonesian";
            } else
            if (radioID == R.id.korean) {
                restType = "Korean";
            } else
```

```
            if (radioID == R.id.japanese) {
                restType = "Japanese";
            } else
            if (radioID == R.id.thai) {
                restType = "Thai";
            }
            //String combineStr = nameStr + "\n" + addressStr + "\n" + telStr + "\n" +restType;
            //Toast.makeText(v.getContext(), combineStr, Toast.LENGTH_LONG).show();
            Restaurant restaurant = new Restaurant();
            restaurant.setName(nameStr);
            restaurant.setAddress(addressStr);
            restaurant.setTelephone(telStr);
            restaurant.setRestaurantType(restType);
            adapter.add(restaurant);
            host.setCurrentTab(0);
        }
    };

    static class RestaurantHolder {
        private TextView restName = null;
        private TextView addr = null;
        private ImageView icon = null;
        RestaurantHolder(View row) {
            restName = row.findViewById(R.id.restName);
            addr = row.findViewById(R.id.restAddr);
            icon = row.findViewById(R.id.icon);
        }
        void populateFrom(Restaurant r) {
            restName.setText(r.getName());
            addr.setText(r.getAddress() + ", " + r.getTelephone());
            if (r.getRestaurantType().equals("Chinese")) {
                icon.setImageResource(R.drawable.ball_red);
            } else if (r.getRestaurantType().equals("Western")) {
                icon.setImageResource(R.drawable.ball_yellow);
            } else {
                icon.setImageResource(R.drawable.ball_green);
            }
        }
    }

    class RestaurantAdapter extends ArrayAdapter<Restaurant> {
        RestaurantAdapter() { super(RestaurantList.this,R.layout.row, model); }

        @Override
        public View getView(int position, View convertView, ViewGroup parent) {
            View row = convertView;
            RestaurantHolder holder;
            if (row == null) {
                LayoutInflater inflater = getLayoutInflater();
                row = inflater.inflate(R.layout.row, parent, false);
                holder = new RestaurantHolder(row);
                row.setTag(holder);
            } else {
                holder = (RestaurantHolder)row.getTag();
            }
            holder.populateFrom(model.get(position));
            return (row);
        }
    }
```

```
178
179         AdapterView.OnItemClickListener onListClick = new AdapterView.OnItemClickListener() {
180             @Override
181             public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
182                 Restaurant r = model.get(position);
183
184                 restaurantName.setText(r.getName());
185                 restaurantAddress.setText(r.getAddress());
186                 restaurantTel.setText(r.getTelephone());
187
188                 if (r.getRestaurantType().equals("Chinese")) {
189                     restaurantTypes.check(R.id.chinese);
190                 } else if (r.getRestaurantType().equals("Western")) {
191                     restaurantTypes.check(R.id.western);
192                 } else if (r.getRestaurantType().equals("Indian")) {
193                     restaurantTypes.check(R.id.indian);
194                 } else if (r.getRestaurantType().equals("Indonesian")) {
195                     restaurantTypes.check(R.id.indonesian);
196                 } else if (r.getRestaurantType().equals("Korean")) {
197                     restaurantTypes.check(R.id.korean);
198                 } else if (r.getRestaurantType().equals("Japanese")) {
199                     restaurantTypes.check(R.id.japanese);
200                 } else {
201                     restaurantTypes.check(R.id.thai);
202                 }
203                 host.setCurrentTab(1);
204             }
205         };
206     }
```

10. Run the *Lab3a* project. Click on the MENU button, the **About** option menu item will pop-up. When click on the item from the option menu, a toast will be displayed.



Further Enhancements

11. At the moment, the option menu and its **"About"** menu item can be activated at the *"LIST"* tab and "*DETAILS*" tab. We will now update the ***onCreateOptionsMenu(Menu menu)*** method to limit the **"About"** menu to pop-up only when user is at the "*DETAILS*" tab. The ***onCreateOptionsMenu(Menu menu)*** method is called every time when ***invalidateOptionsMenu()*** is called. The ***onCreateOptionsMenu(Menu menu)*** method must **return true** for the **menu to be displayed**; and **return false** to hide the menu.
    Hint: Open the **RestaurantList** activity and update the program with the following codes:
    i. Declare an extra **boolean** variable named **showMenu**

    Declaration

```
private boolean showMenu = false;
```

    ii. Set **TabHost** a ***setOnTabChangedListener*** which is a listener to detect change of tab view. By calling the method ***invalidateOptionsMenu()*** we update ***showMenu*** according to current tab view selected. If the tab is at "LIST" tab, showMenu is set to ***false***. Otherwise, showMenu is set to ***true***.

    PART I – add within *onCreate()* method
    The ***invalidateOptionsMenu()*** method is called when there is a change in the tab selection.

```
host.setOnTabChangedListener(new TabHost.OnTabChangeListener() {
    @Override
    public void onTabChanged(String tabId) {
        invalidateOptionsMenu();
    }
});
```

    PART II – after the *onCreate()* method add a *invalidateOptionsMenu()* callback method

```
@Override
public void invalidateOptionsMenu() {
    if (host.getCurrentTab() == 0) {
        showMenu = false;
    } else if (host.getCurrentTab() == 1) {
        showMenu = true;
    }
    super.invalidateOptionsMenu();
}
```

    iii. The ***onCreateOptionsMenu(Menu menu)*** method will be called automatically each time the ***invalidateOptionMenu()*** method is called.

    PART III – update *onCreateOptionsMenu()* callback method

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    if (showMenu == true) {
        getMenuInflater().inflate(R.menu.option, menu);
        return true;
    }
    else
        return false;
}
```

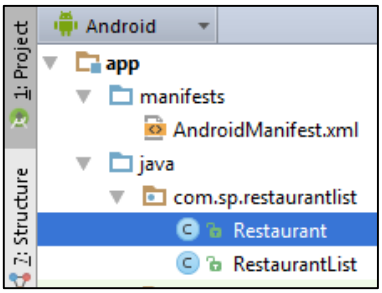
iv. **After** the ***onCreate()*** **method** add a ***onStart()*** **callback method which will call the** ***invalidateOptionsMenu()*** method when the activity is start up.

```
@Override
protected void onStart() {
    invalidateOptionsMenu();
    super.onStart();
}
```

12. Run *Lab3a* project. Test the **MENU** display constraint. **Show to your lecturer after you have completed.**

Part II – Using Android SQLite Database

13. At the moment, all data saved in the Restaurant List will be lost whenever the app is no more active. In this exercise, Android SQLite database will be used to hold the restaurant data. The data saved will stay persist with the app from run to run
14. Create a new project with **No Activity** and the following information:
    - **Name**: *Restaurant List*
    - **Package name**: *com.sp.restaurantlist*
    - **Save location**: *C:\MAD\AndroidStudioProjects\Lab3b or D:\MAD\AndroidStudioProjects\Lab3b*
    - **Language:** *Java*
    - **Minimum SDK**: *API 27: Android 8.1 (Oreo)*
    - **Source Language:** *Java*
    - **Build configuration language:** Groovy DSL (build. grade)
15. Open **Windows File Explorer** and navigate to your Android Studio workspace (*C:\MAD\AndroidStudioProjects* or *D:\MAD\AndroidStudioProjects*) where all you projects are created and saved.
16. Double click to open the *Lab3a* project folder and navigate down to "**app\src\main**" folder. Copy **AndroidManifest.xml** file, **java** and **res** folders
17. Go to newly created project "*Lab3b\app\src\main*" folder and paste into it to overwrite existing folders and files
18. Go back to Android Studio. Open the *RestaurantList.java* and *main.xml* files and they should show the content from *Lab3b*
19. Expand the **java/com.sp.restaurantlist** folder. Right click on the *Restaurant.java* file and select **Delete** to remove the file from the project. ***Restaurant.java*** will be replaced with ***RestaurantHelper.java*** that deals with the SQLite database operations.



20. SQLite is an Open Source Database which is embedded into Android. SQLite supports standard relational database features like SQL syntax, transactions and prepared statements. More information about SQLite can be found on the SQLite website: **http://www.sqlite.org**.
21. A database is like a bookshelf, a database table is like a file folder and records (database model) saved in database table is like form kept in file folder

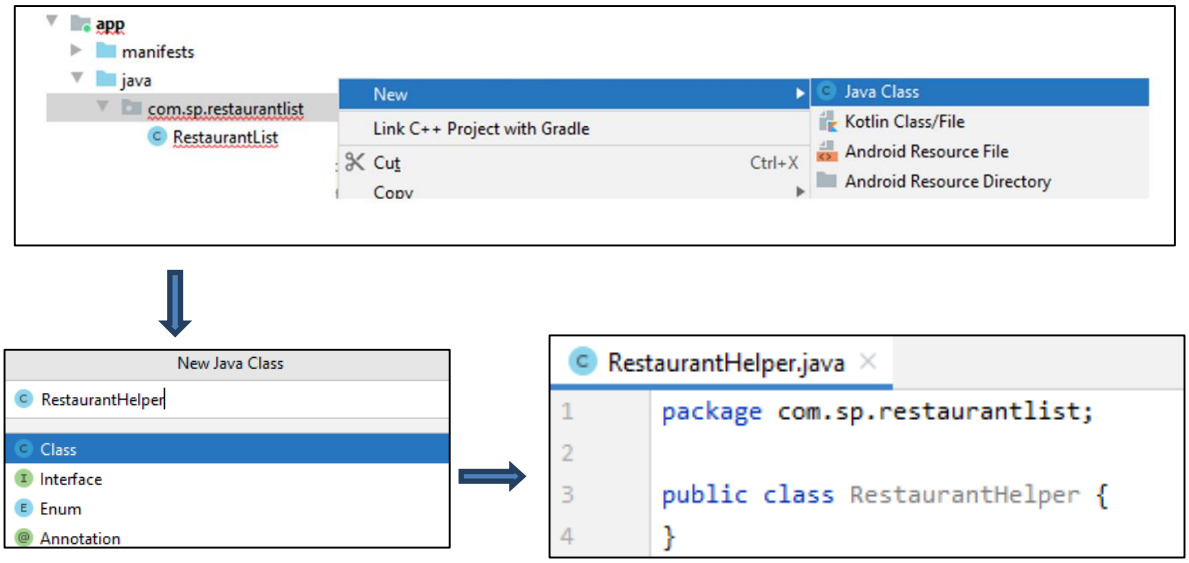
(i) Bookshelf → Database

(ii) File folder → Database Table

(iii) Forms → Records in Table

22. With the basic understanding of database functionality, we will first create a class that handles all the operations required to deal with the database such as creating the database, creating tables, inserting and deleting records and so on

23. The first step is to create a class **RestaurantHelper.java** that inherits from **SQLiteOpenHelper** class.



Add "extends SQLiteOpenHelper" to the class header:

```
class RestaurantHelper extends SQLiteOpenHelper {
```

24. This class provides two methods to override to deal with the database:

- onCreate(SQLiteDatabase db): invoked when the database is created, this is where we can create tables and columns to them, create views or triggers.

```
@Override
public void onCreate(SQLiteDatabase db) {
    // will be called once when database has not been created
    db.execSQL("CREATE TABLE restaurants_table ( " +
            "_id INTEGER PRIMARY KEY AUTOINCREMENT," +
            "restaurantName TEXT, " +
            "restaurantAddress TEXT, "+
            "restaurantTel TEXT," +
            "restaurantType TEXT);");
}
```

- onUpgrade(SQLiteDatabse db, int oldVersion, int newVersion): invoked when we make a modification to the database such as altering, dropping , creating new tables.

```
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    // Will not be called until SCHEMA_VERSION increases
    // Here we can upgrade the database e.g. add more tables
}
```

25. The database and table created for the Restaurant List application will have the following name and fields:

Database's Name	**restaurentlist.db**
Table's Name	**restaurants_table**

restaurants_table Format:

Field's Name	Type	Key	Description
_id	INTEGER AUTOINCREMENT	PRIMARY	Create a unique integer number for each record
restaurantName	TEXT		
restaurantAddress	TEXT		
restaurantTel	TEXT		
restaurantType	TEXT		

Take note of the `_id`, there should be no space between the underscore (_) symbol and "id".

26. In the **RestaurantHelper** class, **onCreate()** is called by the framework, if the database does not exists. `SQLiteOpenHelper` provides the methods **getReadableDatabase()** and **getWriteableDatabase()** to get access to a SQLiteDatabase object; either in read or write mode.

```
@Override
public void onCreate(SQLiteDatabase db) {
    // will be called once when database has not been created
    db.execSQL("CREATE TABLE restaurants_table ( " +
            "_id INTEGER PRIMARY KEY AUTOINCREMENT," +
            "restaurantName TEXT, " +
            "restaurantAddress TEXT, "+
            "restaurantTel TEXT," +
            "restaurantType TEXT);");
}

10 usages
@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    // Will not be called until SCHEMA_VERSION increases
    // Here we can upgrade the database e.g. add more tables
}
```

27. Update the *RestaurantHelper.java* file with the following content and save.

- **onCreate()** – creates a database table within the database called ***restaurantlist.db***
  This has to be done before any insert/select/delete SQL operations
- **getAll**() – triggers an SQL query. The return holds the result of the query
- **insert**() – inserts a new record (with restaurant info) into the database table
- c.**getString**(i) – retrieves data from index i of cursor c

All these methods are housed within the class *RestaurantHelper*.
Hence to use them, first of all a *RestaurantHelper* object must be created.
E.g.

```
RestaurantHelper helper = new RestaurantHelper(this);;
helper.getAll();
```

```java
package com.sp.restaurantlist;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class RestaurantHelper extends SQLiteOpenHelper {
    private static final String DATABASE_NAME = "restaurantlist.db";
    private static final int SCHEMA_VERSION = 1;

    public RestaurantHelper(Context context) {
        super(context, DATABASE_NAME, null, SCHEMA_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        // Will be called once when the database is not created
        db.execSQL("CREATE TABLE restaurants_table ( _id INTEGER PRIMARY KEY AUTOINCREMENT," +
                " restaurantName TEXT, restaurantAddress TEXT, restaurantTel TEXT, restaurantType TEXT);");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // Will not be called until SCHEMA_VERSION increases
        // Here we can upgrade the database e.g. add more tables
    }

    /* Read all records from restaurants_table */
    public Cursor getAll() {
        return (getReadableDatabase().rawQuery(
                "SELECT _id, restaurantName, restaurantAddress, restaurantTel," +
                        " restaurantType FROM restaurants_table ORDER BY restaurantName", null));
    }

    /* Write a record into restaurants_table */
    public void insert(String restaurantName, String restaurantAddress,
                       String restaurantTel, String restaurantType) {
        ContentValues cv = new ContentValues();

        cv.put("restaurantName", restaurantName);
        cv.put("restaurantAddress", restaurantAddress);
        cv.put("restaurantTel", restaurantTel);
        cv.put("restaurantType", restaurantType);

        getWritableDatabase().insert("restaurants_table", "restaurantName", cv);
    }

    public String getRestaurantName(Cursor c) {
        return (c.getString(1));
    }

    public String getRestaurantAddress(Cursor c) {
        return (c.getString(2));
    }

    public String getRestaurantTel(Cursor c) {
        return (c.getString(3));
    }

    public String getRestaurantType(Cursor c) {
        return (c.getString(4));
    }
}
```

28. In the previous Labs, we use **ArrayAdapter** to bind the **ArrayList** and **ListView**. Any new restaurant data model added to ArrayAdapter, it will update both ArrayList and ListView automatically. Whereas we cannot use ArrayList and ArrayAdapter any more for using SQLite database. Instead of **ArrayList**, it is **replaced by Cursor** and **ArrayAdapter** is **replaced by CursorAdapter**. Therefore, **CursorAdapter** is now being used to bind the Cursor and ListView for any new restaurant record added
29. Open the *RestaurantList.java* file, remove extraneous code and simplify the code as follows. Save the file when completed.

```
package com.sp.restaurantlist;

import androidx.appcompat.app.AppCompatActivity;
import android.database.Cursor;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.RadioGroup;
import android.widget.TabHost;
import android.widget.TextView;
import android.widget.Toast;

import android.content.Context;
import androidx.cursoradapter.widget.CursorAdapter;

public class RestaurantList extends AppCompatActivity {
    private EditText restaurantName;
    private RadioGroup restaurantTypes;
    private Button buttonSave;
    private EditText restaurantAddress;
    private EditText restaurantTel;

    private Cursor model = null;
    private RestaurantAdapter adapter = null;
    private ListView list;
    private RestaurantHelper helper = null;
    private TabHost host;
    private boolean showMenu = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        restaurantName = findViewById(R.id.restaurant_name);
        restaurantTypes = findViewById(R.id.restaurant_types);

        buttonSave = findViewById(R.id.button_save);
        buttonSave.setOnClickListener(onSave);

        restaurantAddress = findViewById(R.id.restaurant_address);
        restaurantTel = findViewById(R.id.restaurant_tel);
```

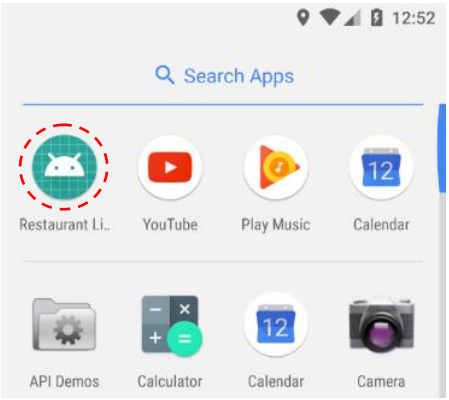
```
51
52          helper = new RestaurantHelper(this);
53          list = findViewById(R.id.restaurants);
54          model = helper.getAll();
55          adapter = new RestaurantAdapter(this, model, 0);
56          list.setAdapter(adapter);
57
58          host = findViewById(R.id.tabHost);
59          host.setup();
60
61          //Tab 1
62          TabHost.TabSpec spec = host.newTabSpec("List");
63          spec.setContent(R.id.restaurants_tab);
64          spec.setIndicator("List");
65          host.addTab(spec);
66
67          //Tab 2
68          spec = host.newTabSpec("Details");
69          spec.setContent(R.id.details_tab);
70          spec.setIndicator("Details");
71          host.addTab(spec);
72          host.setCurrentTab(1);
73          list.setOnItemClickListener(onListClick);
74
75          host.setOnTabChangedListener(new TabHost.OnTabChangeListener() {
76              @Override
77              public void onTabChanged(String tabId) {
78                  invalidateOptionsMenu();
79              }
80          });
81      }
```

```
82
83      @Override
84      protected void onDestroy() {
85          helper.close();
86          super.onDestroy();
87      }
88
89      @Override
90      protected void onStart() {
91          invalidateOptionsMenu();
92          super.onStart();
93      }
94
95      @Override
96      public void invalidateOptionsMenu() {
97          if (host.getCurrentTab() == 0) {
98              showMenu = false;
99          } else if (host.getCurrentTab() == 1) {
100             showMenu = true;
101         }
102         super.invalidateOptionsMenu();
103     }
104     @Override
105     public boolean onCreateOptionsMenu(Menu menu) {
106         if (showMenu == true) {
107             getMenuInflater().inflate(R.menu.option, menu);
108             return true;
109         }
110         else
111             return false;
112     }
113
114     @Override
115     public boolean onOptionsItemSelected(MenuItem item) {
116         if (item.getItemId() == R.id.about)
117         {
118             Toast.makeText(this, "Restaurant List - version 1.0", Toast.LENGTH_LONG).show();
119             break;
120         }
121         return super.onOptionsItemSelected(item);
122     }
123
124     private View.OnClickListener onSave = new View.OnClickListener() {
125         @Override
126         public void onClick(View v) {
127             // To read data from restaurantName EditText
128             String nameStr = restaurantName.getText().toString();
129             // To read data from restaurantAddress EditText
130             String addressStr = restaurantAddress.getText().toString();
131             // To read data from restaurantTel EditText
132             String telStr = restaurantTel.getText().toString();
133             String restType = "";
134             //To read selection of restaurantTypes RadioGroup
135             int radioID = restaurantTypes.getCheckedRadioButtonId();
136             if (radioID == R.id.chinese ) {
137                 restType = "Chinese";
138             } else
139             if (radioID == R.id.western ) {
140                 restType = "Western";
141             } else
```

```
142             if (radioID == R.id.indian ) {
143                 restType = "Indian";
144             } else
145             if (radioID == R.id.indonesian ) {
146                 restType = "Indonesian";
147             } else
148             if (radioID == R.id.korean) {
149                 restType = "Korean";
150             } else
151             if (radioID == R.id.japanese) {
152                 restType = "Japanese";
153             } else
154             if (radioID == R.id.thai) {
155                 restType = "Thai";
156             }
157             //Insert record into SQLite table
158             helper.insert(nameStr, addressStr, telStr, restType);
159
160             model = helper.getAll();    //Update Cursor after new record is added
161             adapter.swapCursor(model);
162             host.setCurrentTab(0);
163         }
164     };
165
166     static class RestaurantHolder {
167         private TextView restName = null;
168         private TextView addr = null;
169         private ImageView icon = null;
170
171         RestaurantHolder(View row) {
172             restName = row.findViewById(R.id.restName);
173             addr = row.findViewById(R.id.restAddr);
174             icon = row.findViewById(R.id.icon);
175         }
176
177         void populateFrom(Cursor c, RestaurantHelper helper) {
178             restName.setText(helper.getRestaurantName(c));
179             String temp = helper.getRestaurantAddress(c) + ", " + helper.getRestaurantTel(c);
180             addr.setText(temp);
181
182             if (helper.getRestaurantType(c).equals("Chinese")) {
183                 icon.setImageResource(R.drawable.ball_red);
184             } else if (helper.getRestaurantType(c).equals("Western")) {
185                 icon.setImageResource(R.drawable.ball_yellow);
186             } else {
187                 icon.setImageResource(R.drawable.ball_green);
188             }
189         }
190     }
191
192
```

```
193     class RestaurantAdapter extends CursorAdapter {
194         RestaurantAdapter(Context context, Cursor cursor, int flags) {
195             super(context, cursor, flags);
196         }
197
198         @Override
199 @       public void bindView(View view, Context context, Cursor cursor) {
200             RestaurantHolder holder = (RestaurantHolder) view.getTag();
201             holder.populateFrom(cursor, helper);
202         }
203
204         @Override
205         public View newView(Context context, Cursor cursor, ViewGroup parent) {
206             LayoutInflater inflater = getLayoutInflater();
207             View row = inflater.inflate(R.layout.row, parent, false);
208             RestaurantHolder holder = new RestaurantHolder(row);
209             row.setTag(holder);
210             return (row);
211         }
212     }
213
214     AdapterView.OnItemClickListener onListClick = new AdapterView.OnItemClickListener() {
215         @Override
216         public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
217             model.moveToPosition(position);
218             restaurantName.setText(helper.getRestaurantName(model));
219             restaurantAddress.setText(helper.getRestaurantAddress(model));
220             restaurantTel.setText(helper.getRestaurantTel(model));
221
222             if (helper.getRestaurantType(model).equals("Chinese")) {
223                 restaurantTypes.check(R.id.chinese);
224             } else if (helper.getRestaurantType(model).equals("Western")) {
225                 restaurantTypes.check(R.id.western);
226             } else if (helper.getRestaurantType(model).equals("Indian")) {
227                 restaurantTypes.check(R.id.indian);
228             } else if (helper.getRestaurantType(model).equals("Indonesian")) {
229                 restaurantTypes.check(R.id.indonesian);
230             } else if (helper.getRestaurantType(model).equals("Korean")) {
231                 restaurantTypes.check(R.id.korean);
232             } else if (helper.getRestaurantType(model).equals("Japanese")) {
233                 restaurantTypes.check(R.id.japanese);
234             } else {
235                 restaurantTypes.check(R.id.thai);
236             }
237             host.setCurrentTab(1);
238         }
239     };
240 }
```

Note: If you have any errors with the SQLite database, you need to uninstall the Restaurant List App before testing again.



30. Run the *Lab3b* project. Enter a restaurant data and save. Click on the back button to exit from the app

31. Click on the **Restaurant List** App to run the app again. The previously saved data will stay on the list
32. **If you have completed the project, demo it to your lecturer**

-END-