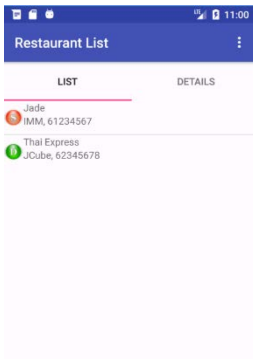
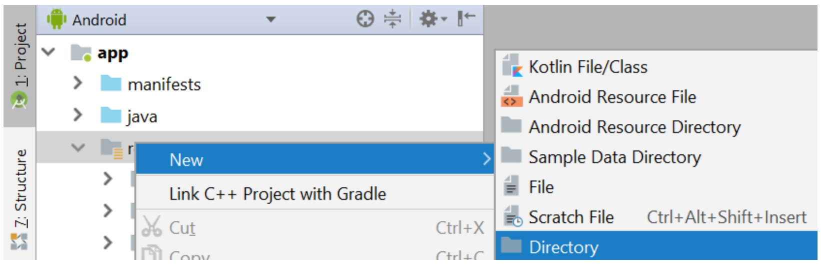
Practical 3: Menu and SQLite Database

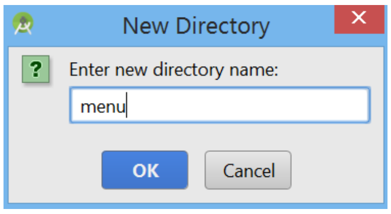
At the end of the session, you will learn how to create a Menu to activate a Toast to show information entered in the restaurant detail form. In order to keep the restaurant list data persistent, you will learn how to create a database for the restaurant list using SQLite.



Part I – Create Menu & Detect Menu Item Select

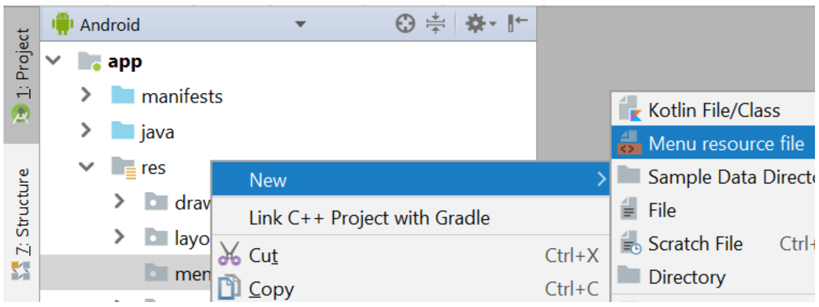
1. In this lab, we will learn how to include an image icon to a **Menu** item and detect the menu item selection.
2. Create a new project with the following information:
   - **Application Name** : *Restaurant List*
   - **Company Domain** : *sp.com*
   - **Project Location** : *C:\MAD\AndroidStudioProjects\Lab3a or D:\MAD\AndroidStudioProjects\Lab3a*
   - **Minimum SDK** : *Android 5.0 (Lollipop)*
   - **Empty Activity name** : *RestaurantList*
   - **Layout name** : *main*
3. Close *RestaurantList.java* and *main.xml* files in the **Editor** pane
4. Open your **Windows File Explorer** and navigate to your **Android Studio** workspace where all your projects are created
5. 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 file and folder
6. 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
7. At Android Studio, right click on the **res** folder and select **New > Directory** to create folder named menu

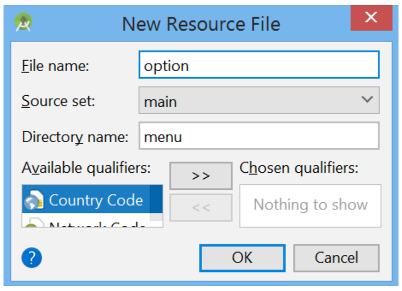



New Directory

Enter new directory name: menu

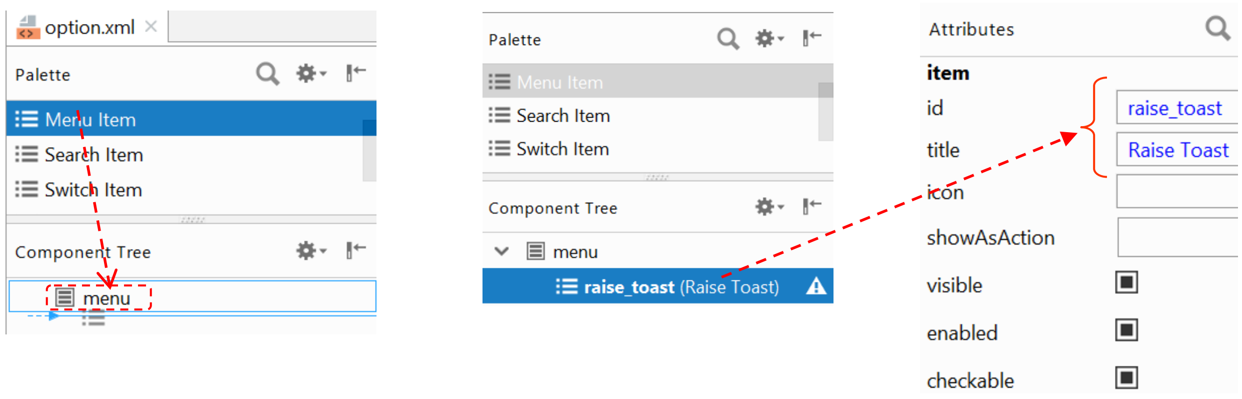
8. Right click on the newly created folder **res/menu** and select **New > Menu resource file** to create a menu file named '*option*'




New Resource File

File name: option
Source set: main
Directory name: menu
Available qualifiers: Country Code, Network Code
Chosen qualifiers: Nothing to show

9. At Design Editor, drag a **Menu Item** from **Palette** and drop into **menu** under Component Tree. Update the item **id** and **title** to *raise_toast* and *Raise Toast* respectively.

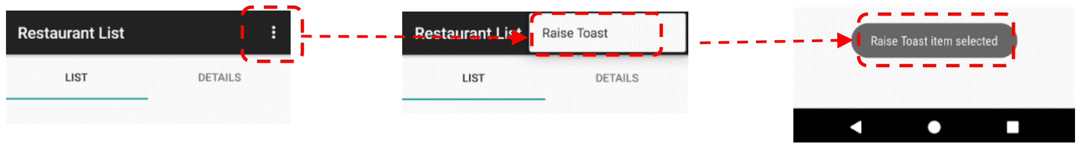


id: raise_toast
title: Raise Toast
icon:
showAsAction:
visible: ▣
enabled: ▣
checkable: ▣

10. Open the *RestaurantList.java* file. Update the program with the following code to create the ***option* menu** and **detect menu item selection**.

```
package com.sp.restaurantlist;

import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v7.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<Restaurant>();
    private RestaurantAdapter adapter = null;
    private ListView list;
    private TabHost host;

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

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

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

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

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

        host = (TabHost)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) {
        switch (item.getItemId()) {
            case (R.id.raise_toast):
                Toast.makeText(this, "Raise Toast item selected", Toast.LENGTH_LONG).show();
                break;
        }
        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
            switch (restaurantTypes.getCheckedRadioButtonId()) {
                case R.id.chinese:
                    restType = "Chinese";
                    break;
                case R.id.western:
                    restType = "Western";
                    break;
                case R.id.indian:
                    restType = "Indian";
                    break;
                case R.id.indonesian:
                    restType = "Indonesian";
                    break;
                case R.id.korean:
                    restType = "Korean";
                    break;
                case R.id.japanese:
                    restType = "Japanese";
                    break;
                case R.id.thai:
                    restType = "Thai";
                    break;
            }
            //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);
        }
    };

    AdapterView.OnItemClickListener onListClick = new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            Restaurant r = model.get(position);

            restaurantName.setText(r.getName());
            restaurantAddress.setText(r.getAddress());
            restaurantTel.setText(r.getTelephone());

            if (r.getRestaurantType().equals("Chinese")) {
                restaurantTypes.check(R.id.chinese);
            } else if (r.getRestaurantType().equals("Western")) {
                restaurantTypes.check(R.id.western);
            } else if (r.getRestaurantType().equals("Indian")) {
                restaurantTypes.check(R.id.indian);
            } else if (r.getRestaurantType().equals("Indonesia")) {
                restaurantTypes.check(R.id.indonesian);
            } else if (r.getRestaurantType().equals("Korean")) {
                restaurantTypes.check(R.id.korean);
            } else if (r.getRestaurantType().equals("Japanese")) {
                restaurantTypes.check(R.id.japanese);
            } else {
                restaurantTypes.check(R.id.thai);
            }
            host.setCurrentTab(1);
        }
    };
    static class RestaurantHolder {
        private TextView restName = null;
        private TextView addr = null;
        private ImageView icon = null;
        RestaurantHolder(View row) {
            restName = (TextView)row.findViewById(R.id.restName);
            addr = (TextView)row.findViewById(R.id.restAddr);
            icon = (ImageView)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);
        }

        @NonNull
        @Override
        public View getView(int position, @Nullable View convertView, @NonNull 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);
        }
    }
}
```

11. Run the *Lab3a* project. Click on the MENU button, the **Raise Toast** option menu item will pop-up. When click on the item from the option menu,



Extra Credit

12. At the moment, the **Raise Toast** menu can be activated at the '*List*' tab and '*Details*' tab. We will now update the `onCreateOptionsMenu(Menu menu)` method to limit the **Raise Toast** menu to pop-up only when user is at '*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' view, showMenu is set to ***false***. Otherwise, showMenu is set to ***true***.

PART I – add within *onCreate()* method

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

PART II –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 ***invalidatOptionMenu()*** method is called.

Update *onCreateOptionsMenu()* callback method

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

iv. Add a ***onStart()* callback method which call the *invalidateOptionsMenu()*** method when the activity is start up.

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

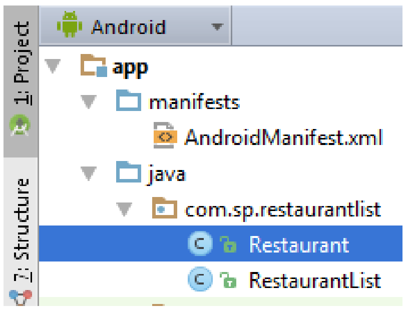
13. Run *Lab3a* project. Test the **MENU** display constraint and show to your lecturer if successful.

Lecturer Signature : ____

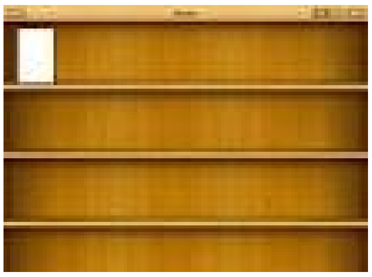
Part II – Using Android SQLite Database

14. At the moment, all data saved in the Restaurant List will 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
15. Create a new project with the following information:
    - **Application Name** : *Restaurant List*
    - **Company Domain** : *sp.com*
    - **Project Location** : *C:\MAD\AndroidStudioProjects\Lab3b or D:\MAD\AndroidStudioProjects\Lab3b*
    - **Minimum SDK** : *Android 5.0 (Lollipop)*
    - **Empty Activity name** : *RestaurantList*
    - **Layout name** : *main*

16. Close all the files in Android Studio
17. 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.
18. Double click to open the *Lab3a* project folder and navigate down to "**app\src\main**" folder. Copy **AndroidManifest.xml** file, **java** and **res** folders
19. Go to newly created project "*Lab3b\app\src\main*" folder and paste into it to overwrite existing folders and files
20. Go back to Android Studio. Open the *RestaurantList.java* and *main.xml* files and they should show the content from *Lab3b*
21. Expand the **java/com.sp.restaurantlist** folder. Right click on the *Restaurant.java* file and select **Delete** to remove the file from the project



22. 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**.
23. 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

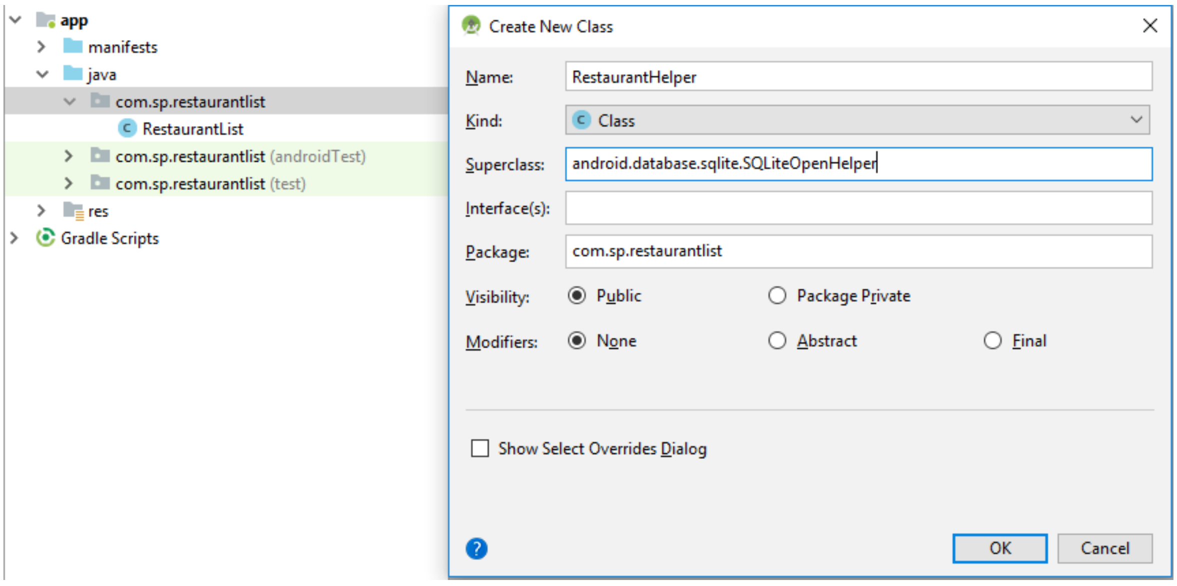

(i) Bookshelf → Database

(ii) File folder → Database Table

(iii)Forms → Records in Table

24. 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

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



Name: RestaurantHelper
Kind: Class
Superclass: android.database.sqlite.SQLiteOpenHelper
Interface(s):
Package: com.sp.restaurantlist
Visibility: [✓ Public] Package Private
Modifiers: [✓ None] Abstract Final
Show Select Overrides Dialog

```
class RestaurantHelper extends SQLiteOpenHelper {
```

26. 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 the database is not 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
}
```

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

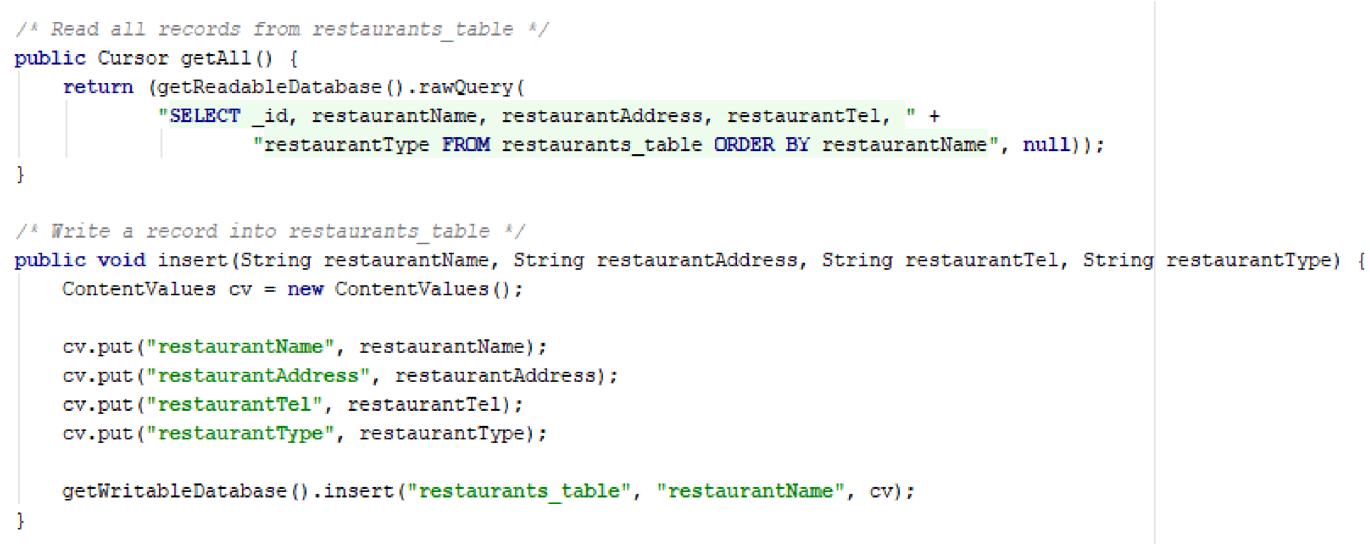
Database's Name	`restaurentlist.db`
Table's Name	`restaurants_table`

`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".

28. In **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 an SQLiteDatabase object; either in read or write mode.



```
/* 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);
}
```

29. Right click on the **java/com.sp.restaurantlist** folder, select **New > Class** and enter file named as *RestaurantHelper* to create a **SQLiteOpenHelper** subclass. Update the file with the following content and save

```
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));
    }

}
```

30. In 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
31. Open the *RestaurantList.java* file, remove extraneous code and simplify the code as follow. Save the file when completed.

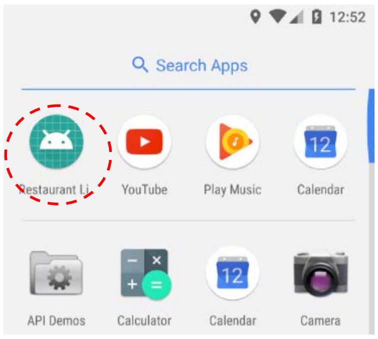
```
package com.sp.restaurantlist;

import android.content.Context;
import android.database.Cursor;
import android.support.v7.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.Button;
import android.widget.CursorAdapter;
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;


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 = (EditText) findViewById(R.id.restaurant_name);
        restaurantTypes = (RadioGroup) findViewById(R.id.restaurant_types);

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

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

        helper = new RestaurantHelper(this);
        list = (ListView) findViewById(R.id.restaurants);
        model = helper.getAll();
        adapter = new RestaurantAdapter(this, model, 0);
        list.setAdapter(adapter);

        host = (TabHost) 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);

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

    @Override
    protected void onDestroy() {
        helper.close();
        super.onDestroy();
    }

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

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

```
                return true;
            else
                return false;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case (R.id.raise_toast):
                Toast.makeText(this, "Raise Toast item selected", Toast.LENGTH_LONG).show();
                break;
        }
        return super.onOptionsItemSelected(item);
    }

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

    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
            switch (restaurantTypes.getCheckedRadioButtonId()) {
                case R.id.chinese:
                    restType = "Chinese";
                    break;
                case R.id.western:
                    restType = "Western";
                    break;
                case R.id.indian:
                    restType = "Indian";
                    break;
                case R.id.indonesian:
                    restType = "Indonesian";
                    break;
                case R.id.korean:
                    restType = "Korean";
                    break;
                case R.id.japanese:
                    restType = "Japanese";
                    break;
                case R.id.thai:
                    restType = "Thai";
                    break;
            }
            //Insert record into SQLite table
            helper.insert(nameStr, addressStr, telStr, restType);

            model = helper.getAll();  //Update Cursor after new record is added
            adapter.swapCursor(model);
            host.setCurrentTab(0);
        }
    };
```

```

        AdapterView.OnItemClickListener onListClick = new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                model.moveToPosition(position);
                restaurantName.setText(helper.getRestaurantName(model));
                restaurantAddress.setText(helper.getRestaurantAddress(model));
                restaurantTel.setText(helper.getRestaurantTel(model));

                if (helper.getRestaurantType(model).equals("Chinese")) {
                    restaurantTypes.check(R.id.chinese);
                } else if (helper.getRestaurantType(model).equals("Western")) {
                    restaurantTypes.check(R.id.western);
                } else if (helper.getRestaurantType(model).equals("Indian")) {
                    restaurantTypes.check(R.id.indian);
                } else if (helper.getRestaurantType(model).equals("Indonesia")) {
                    restaurantTypes.check(R.id.indonesian);
                } else if (helper.getRestaurantType(model).equals("Korean")) {
                    restaurantTypes.check(R.id.korean);
                } else if (helper.getRestaurantType(model).equals("Japanese")) {
                    restaurantTypes.check(R.id.japanese);
                } else {
                    restaurantTypes.check(R.id.thai);
                }
                host.setCurrentTab(1);
            }
        };

        static class RestaurantHolder {
            private TextView restName = null;
            private TextView addr = null;
            private ImageView icon = null;

            RestaurantHolder(View row) {
                restName = (TextView) row.findViewById(R.id.restName);
                addr = (TextView) row.findViewById(R.id.restAddr);
                icon = (ImageView) row.findViewById(R.id.icon);
            }

            void populateFrom(Cursor c, RestaurantHelper helper) {
                restName.setText(helper.getRestaurantName(c));
                String temp = helper.getRestaurantAddress(c) + ", " + helper.getRestaurantTel(c);
                addr.setText(temp);

                if (helper.getRestaurantType(c).equals("Chinese")) {
                    icon.setImageResource(R.drawable.ball_red);
                } else if (helper.getRestaurantType(c).equals("Western")) {
                    icon.setImageResource(R.drawable.ball_yellow);
                } else {
                    icon.setImageResource(R.drawable.ball_green);
                }
            }

        }

        class RestaurantAdapter extends CursorAdapter {
            RestaurantAdapter(Context context, Cursor cursor, int flags) {
                super(context, cursor, flags);
            }

            @Override
            public void bindView(View view, Context context, Cursor cursor) {
                RestaurantHolder holder = (RestaurantHolder) view.getTag();
                holder.populateFrom(cursor, helper);
            }

            @Override
            public View newView(Context context, Cursor cursor, ViewGroup parent) {
```

```
                        LayoutInflater inflater = getLayoutInflater();
                        View row = inflater.inflate(R.layout.row, parent, false);
                        RestaurantHolder holder = new RestaurantHolder(row);
                        row.setTag(holder);
                        return (row);
                    }
                }
            }
```

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



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

33. Click on the **Restaurant List** App to run the app again. The previously saved data will stay on the list
34. If you have completed, show to your lecturer

Lecturer Signature : ______________________________

-END-