

# Mobile Application Development

---

Produced  
by

David Drohan ([ddrohan@wit.ie](mailto:ddrohan@wit.ie))

Department of Computing & Mathematics  
Waterford Institute of Technology

<http://www.wit.ie>



Waterford Institute of Technology  
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE





# User Interface Design & Development – Part 3

---





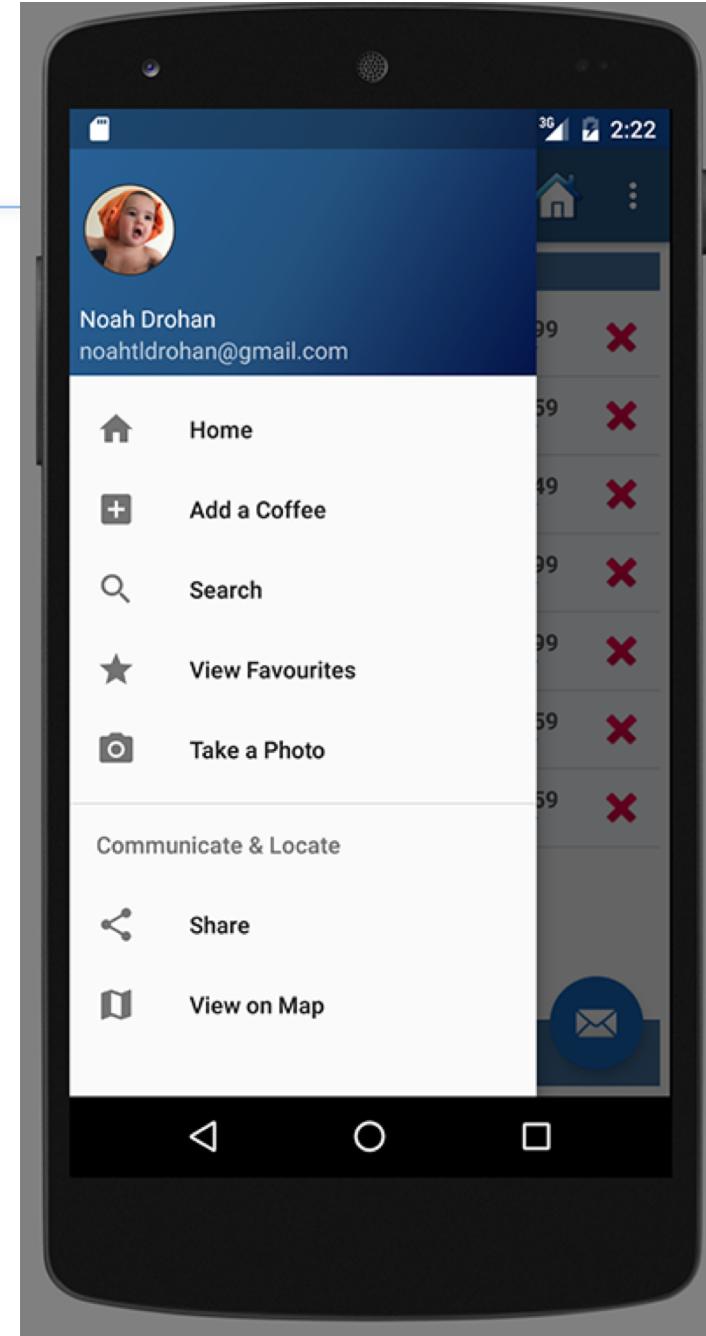
# Goals of this Section

---

- ❑ Be able to create and use a **NavigationDrawer** to implement more effective navigation within an app (CoffeeMate 4.0+)
- ❑ Investigate replacing our **ListView** and **ArrayAdapter** to use a **RecyclerView** & **RecyclerAdapter**

# Case Study

- ❑ **CoffeeMate** – an Android App to keep track of your Coffees, their details, and which ones you like the best (your favourites)
- ❑ App Features with Google+ Sign-In
  - ❑ List all your Coffees
  - ❑ View specific Coffee details
  - ❑ Filter Coffees by Name and Type
  - ❑ Delete a Coffee
  - ❑ List all your Favourite Coffees
  - ❑ View Coffees on a Map





---

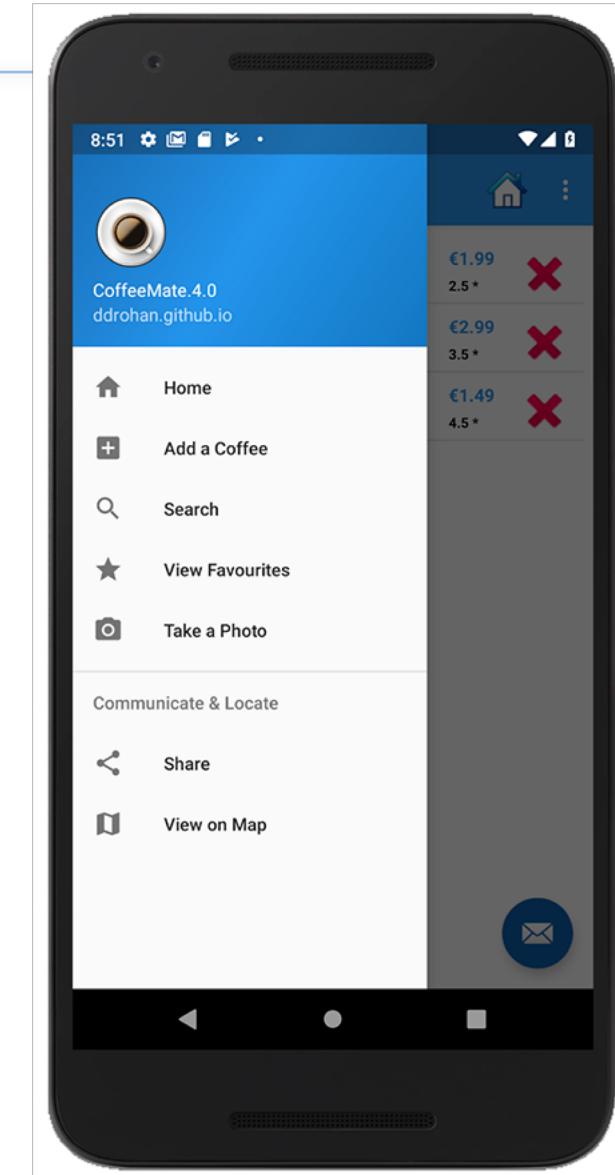
# CoffeeMate 4.0+

## Using The Navigation Drawer



# Navigation Drawer Overview

- ❑ <https://developer.android.com/training/implementing-navigation/nav-drawer.html>
- ❑ The navigation drawer is a panel that displays the app's main navigation options on the left edge of the screen. It is hidden most of the time, but is revealed when the user swipes a finger from the left edge of the screen or, while at the top level of the app, the user touches the app icon in the action bar.





# Navigation Drawer Overview

---

- ❑ Android Studio does a lot of the heavy lifting for you, but generally the following steps are necessary to add a Navigation Drawer to your app
  - ❑ *Create drawer layout*
  - ❑ *Bind to navigation drawer layout*
  - ❑ *Handle navigation drawer click and*
  - ❑ *Update content based on user selection*



# Overview - Create Drawer Layout

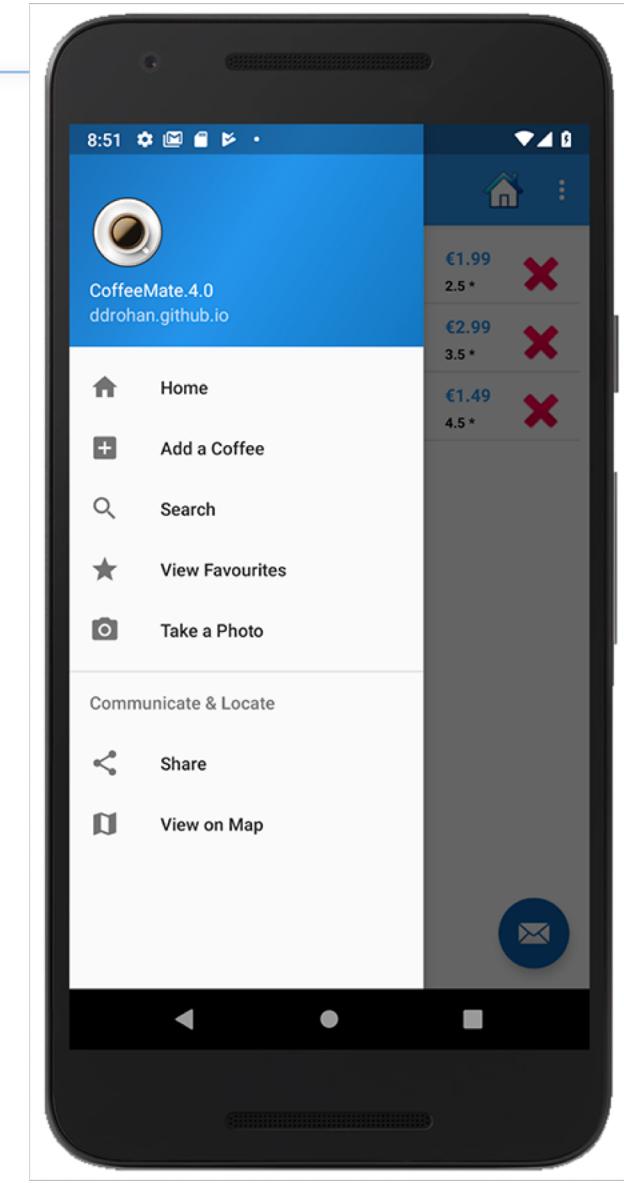
---

- ❑ For creating a navigation drawer, first we need to declare the drawer layout in your main activity where you want to show the navigation drawer.
- ❑ You add  
**`android.support.v4.widget.DrawerLayout`** as the root view of your activity layout.
- ❑ As already mentioned, Android Studio does a lot of this for you so it's more about understanding how it all pieces together to allow you to modify as necessary.
- ❑ We'll use **CoffeeMate** as the example to illustrate...



# Overview - Create Drawer Layout \*

```
home.xml x
1  <?xml version="1.0" encoding="utf-8"?>
2  <android.support.v4.widget.DrawerLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:id="@+id/drawer_layout"
7      android:layout_width="match_parent"
8      android:layout_height="match_parent"
9      android:windowSoftInputMode="adjustPan|adjustResize"
10     tools:openDrawer="start">
11
12     <include
13         layout="@layout/app_bar_home"
14         android:layout_width="match_parent"
15         android:layout_height="match_parent" />
16
17     <android.support.design.widget.NavigationView
18         android:id="@+id/nav_view"
19         android:layout_width="wrap_content"
20         android:layout_height="match_parent"
21         android:layout_gravity="start"
22         android:fitsSystemWindows="true"
23         app:headerLayout="@layout/nav_header_home"
24         app:menu="@menu/home_drawer" />
25 </android.support.v4.widget.DrawerLayout>
```





# Overview - Create Drawer Layout \*

```
home.xml x
1  <?xml version="1.0" encoding="utf-8"?>
2  <android.support.v4.widget.DrawerLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:id="@+id/drawer_layout"
7      android:layout_width="match_parent"
8      android:layout_height="match_parent"
9      android:windowSoftInputMode="adjustPan|adjustResize"
10     tools:openDrawer="start">
11
12     <include
13         layout="@layout/app_bar_home"
14         android:layout_width="match_parent"
15         android:layout_height="match_parent" />
16
17     <android.support.design.widget.NavigationView
18         android:id="@+id/nav_view"
19         android:layout_width="wrap_content"
20         android:layout_height="match_parent"
21         android:layout_gravity="start"
22         android:fitsSystemWindows="true"
23         app:headerLayout="@layout/nav_header_home"
24         app:menu="@menu/home_drawer" />
25
</android.support.v4.widget.DrawerLayout>
```

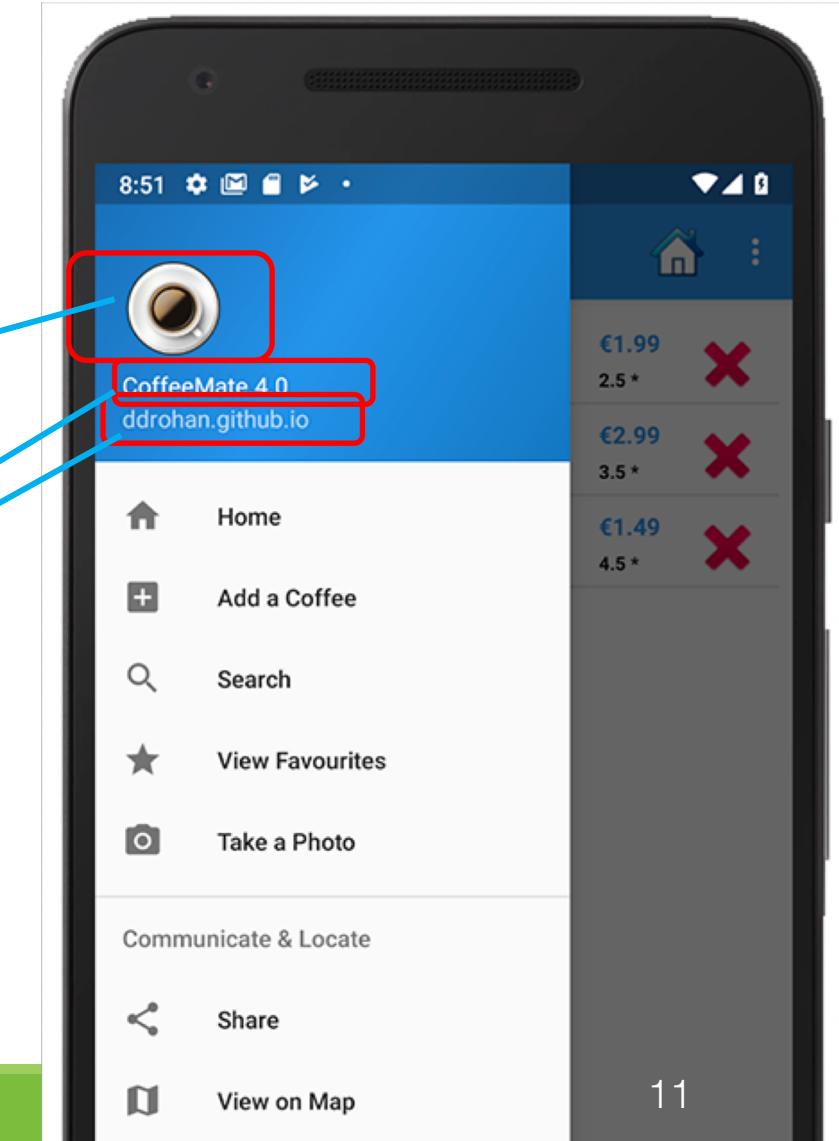
- *home.xml* contains the Navigation Header (*nav\_header\_home*) AND the Navigation Drawer Menu (*home\_drawer*) inside a **NavigationView**.
- *home* includes *app\_bar\_home* which will display our content
- Also, note the ‘ids’ of the widgets (for later on)



# Overview – nav\_header\_home \*

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="160dp"
    android:background="@drawable/side_nav_bar"
    android:gravity="bottom"
    android:orientation="vertical"
    android:paddingBottom="6dp"
    android:paddingLeft="6dp"
    android:paddingRight="6dp"
    android:paddingTop="6dp"
    android:theme="@style/ThemeOverlay.AppCompat.Dark">

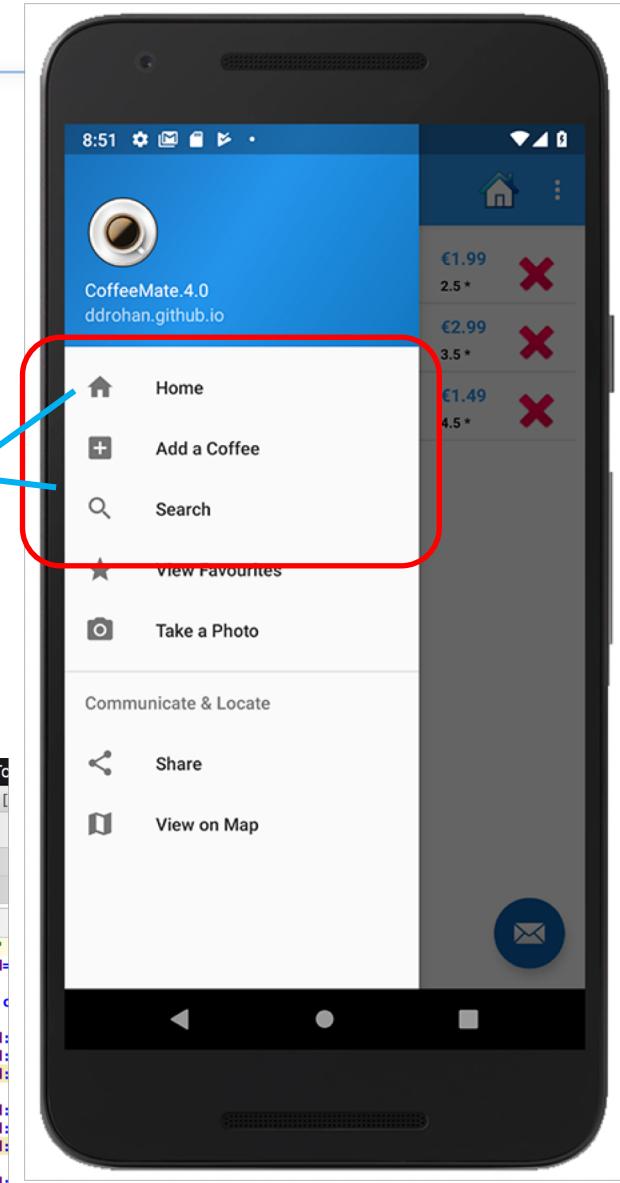
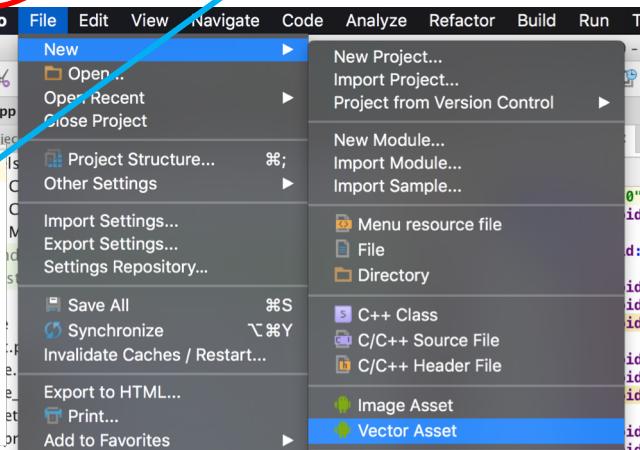
    <!-- https://github.com/vinc3m1/RoundedImageView -->
    <com.makeramen.roundedimageview.RoundedImageView...>
        <TextView
            android:id="@+id/googlename"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingTop="16dp"
            android:text="CoffeeMate"
            android:textAppearance="@style/TextAppearance.AppCompat.Body1" />
        <TextView...>
    </com.makeramen.roundedimageview.RoundedImageView...>
</LinearLayout>
```





# Overview – home\_drawer \*

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">  
  
    <group android:checkableBehavior="single">  
        <item  
            android:id="@+id/nav_home"  
            android:icon="@drawable/ic_menu_home"  
            android:title="Home" />  
        <item  
            android:id="@+id/nav_add"  
            android:icon="@drawable/ic_menu_addcoffee"  
            android:title="Add a Coffee" />  
        <item  
            android:id="@+id/nav_search"  
            android:icon="@drawable/ic_menu_search"  
            android:title="Search" />  
    </item>  
  
<vector xmlns:android="http://schemas.android.com/apk/res/android"  
        android:width="24dp"  
        android:height="24dp"  
        android:viewportWidth="24.0"  
        android:viewportHeight="24.0">  
    <path  
        android:fillColor="#FF000000"  
        android:pathData="M10,20v-6h4v6h5v-8h3L12,3 2,12h3v8z"/>  
</vector>
```





# Overview – app\_bar\_home \*

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:context=".activities.Home">

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

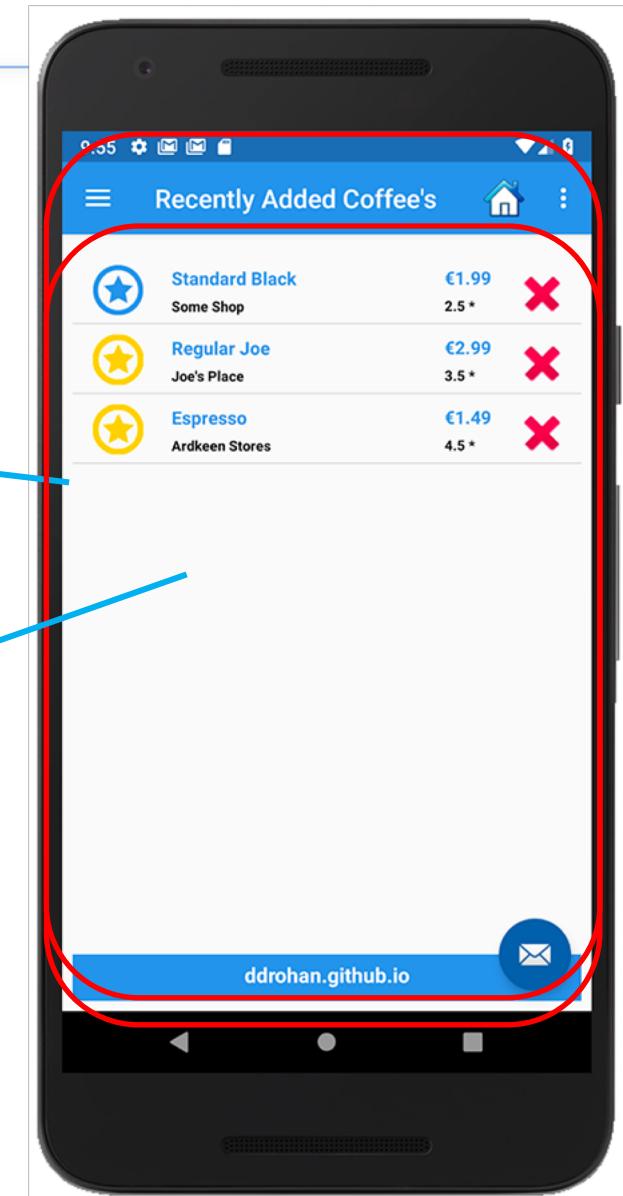
        <android.support.v7.widget.Toolbar ...>

    </android.support.design.widget.AppBarLayout>

    <include layout="@layout/content_home" />

    <android.support.design.widget.FloatingActionButton ...>

```



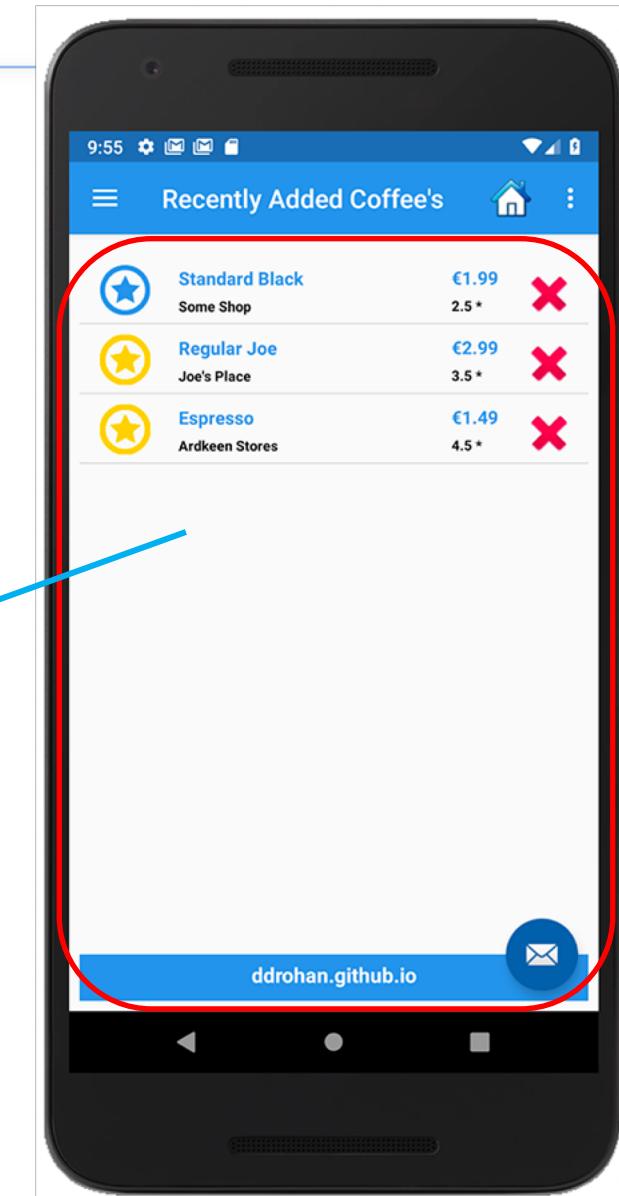


# Overview – content\_home \*

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    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/homeLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context=".activities.Home"
    tools:showIn="@layout/app_bar_home">

    <FrameLayout
        android:id="@+id/homeFrame"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```





# Overview – Bind to the Drawer Layout etc.

- ❑ Once you have the necessary layouts and menu in place, you then need to bind to the **Drawer** and **Navigation View** to allow you to handle the user navigation and switching content based on user selection.
- ❑ In your **onCreate()** you'll have something like the following

```
DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
    this, drawer, toolbar, "Open navigation drawer", "Close navigation drawer");
drawer.addDrawerListener(toggle);
toggle.syncState();

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

*We also setup GooglePhoto and Email for the Drawer here (later labs)*



# Overview – *Bind to the Drawer Layout etc.*

- You'll probably want to display some kind of initial landing page once the app starts so in our example, we load up the list of user coffees (maintained in our **CoffeeFragment**).
- Again, in your *onCreate()* you'll have something like the following

```
FragmentTransaction ft = getFragmentManager().beginTransaction();

CoffeeFragment fragment = CoffeeFragment.newInstance();
ft.replace(R.id.homeFrame, fragment);
ft.commit();
```

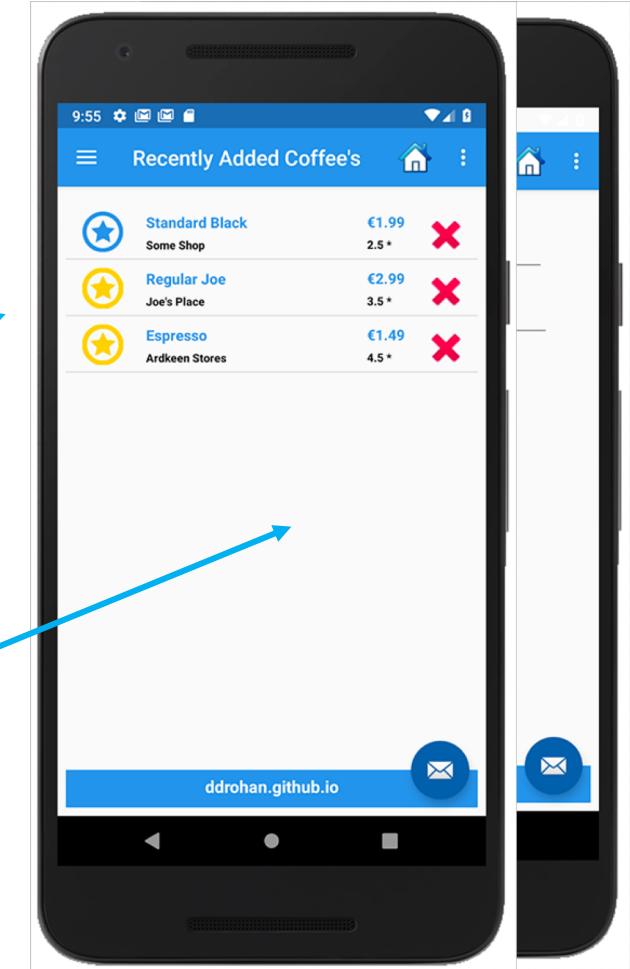
- This creates a new instance of a CoffeeFragment and replaces the fragment in our FrameLayout with this instance.



# Overview – Handle Drawer Click & Update Content \*

- To handle users menu selection we implement the following

```
public boolean onNavigationItemSelected(MenuItem item) {  
    // Handle navigation view item clicks here.  
  
    // http://stackoverflow.com/questions/32944798/switch  
  
    int id = item.getItemId();  
    Fragment fragment;  
    ft = getSupportFragmentManager().beginTransaction();  
  
    if (id == R.id.nav_home) {  
        this.setTitle(R.string.recentlyViewedLbl);  
        fragment = CoffeeFragment.newInstance();  
        ((CoffeeFragment)fragment).favourites = false;  
        ft.replace(R.id.homeFrame, fragment);  
        ft.addToBackStack(null);  
        ft.commit();  
    } else if (id == R.id.nav_add) {  
        this.setTitle(R.string.addACoffeeLbl);  
        fragment = AddFragment.newInstance();  
        ft.replace(R.id.homeFrame, fragment);  
        ft.addToBackStack(null);  
        ft.commit();  
    }  
}
```





---

# CoffeeMate 4.0+

**Using a  
RecyclerView  
and  
RecyclerAdapter**



# Introduction – Using RecyclerView & RecyclerAdapter

- ❑ The **RecyclerView** class supports the display of a collection of data.
- ❑ It is a modernized version of **ListView** and **GridView**
- ❑ Enforces a programming style that results in good performance.
- ❑ Comes with default animations for removing and adding elements.
- ❑ Can use different layout managers for positioning items.
- ❑ Uses a **ViewHolder** to store references to the views for one entry in the **RecyclerView**.



# Introduction – Using RecyclerView & RecyclerAdapter

- ❑ A **ViewHolder** class is a static inner class in your adapter which holds references to the relevant views. With these references your code can avoid the time-consuming **findViewById()** method to update the widgets with new data.
- ❑ As we know, An *adapter* manages the data model and adapts it to the individual entries in the widget, in our case, extending **RecyclerView.Adapter** and is assigned to the recycler view via the **RecyclerView.setAdapter**



# Introduction – Using RecyclerView & RecyclerAdapter

- ❑ **onCreateViewHolder** prepares the layout of the items by inflating the correct layout for the individual data elements and returns an object of type **ViewHolder** per visual entry in the recycler view. This instance is used to access the views in the inflated layout and is only called when a new view must be created.
- ❑ Every visible entry in a recycler view is filled with the correct data model item by the adapter. Once a data item becomes visible, the adapter assigns this data to the individual widgets via **onBindViewHolder**.



# CoffeeMate 4.0+ - Dependencies

```
dependencies {  
    implementation fileTree(dir: 'libs', include: ['*.jar'])  
    implementation 'com.android.support:appcompat-v7:28.0.0'  
    implementation 'com.android.support:support-v4:28.0.0'  
    implementation 'com.android.support:design:28.0.0'  
    implementation 'com.android.support.constraint:constraint-layout:1.1.3'  
    testImplementation 'junit:junit:4.12'  
    androidTestImplementation 'com.android.support.test:runner:1.0.2'  
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'  
  
    implementation 'com.makeramen:roundedimageview:2.2.1'  
  
    implementation "com.android.support:recyclerview-v7:28.0.0"  
    implementation "com.android.support:cardview-v7:28.0.0"  
}
```



# CoffeeMate 4.0+ - `coffeecard.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.v7.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/coffeecardview"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:clickable="true"
    android:elevation="30dp">

    <RelativeLayout
        android:id="@+id/cardviewlayout"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@drawable/background_activated">

        <ImageView...>
        <ImageView...>
        <TextView...>
        <TextView...>
        <TextView...>
        <TextView...>
        <TextView...>
    </RelativeLayout>

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

- Very similar to our existing `coffeerow`



# CoffeeMate 4.0+ - CoffeeViewHolder

```
public class CoffeeViewHolder extends RecyclerView.ViewHolder {  
    public View view;  
    public Coffee coffee;  
    public View.OnClickListener deleteListener;  
    public View.OnClickListener clickListener;  
  
    public CoffeeViewHolder(View v, View.OnClickListener delete,  
                          View.OnClickListener click) {  
        super(v);  
        view = v;  
        deleteListener = delete;  
        clickListener = click;  
    }  
  
    public void updateControls(Coffee coffee) {...}  
}
```

- Very similar to our existing **CoffeeItem**



# CoffeeMate 4.0+ - CoffeeRecyclerAdapter

```
public class CoffeeRecyclerAdapter extends  
        RecyclerView.Adapter<CoffeeViewHolder>  
{  
    public Context context;  
    public View.OnClickListener deleteListener, clickListener;  
    public List<Coffee> coffeeList;  
    public CoffeeViewHolder coffeeViewHolder;  
    public View coffeeCardView;  
  
    public CoffeeRecyclerAdapter(Context context,  
                                View.OnClickListener deleteListener,  
                                View.OnClickListener clickListener,  
                                List<Coffee> coffeeList)  
    {...}  
  
    @NotNull  
    @Override  
    public CoffeeViewHolder onCreateViewHolder(@NotNull ViewGroup viewGroup, int viewType) {  
        LayoutInflator inflater = LayoutInflator.from(viewGroup.getContext());  
        coffeeCardView = inflater.inflate(R.layout.coffeecard, viewGroup, false);  
  
        coffeeViewHolder = new CoffeeViewHolder(coffeeCardView, this.deleteListener,clickListener);  
        return coffeeViewHolder;  
    }  
  
    @Override  
    public void onBindViewHolder(@NotNull CoffeeViewHolder coffeeViewHolder, int i) {  
        Coffee coffee = coffeeList.get(i);  
        coffeeViewHolder.updateControls(coffee);  
    }  
  
    @Override  
    public int getItemCount() { return coffeeList.size(); }  
  
    public void add(int position, Coffee item) {...}  
    public void remove(int position) {...}
```



# CoffeeMate 4.0+ - CoffeeFragment

```
@Override  
public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
  
    clickListener = new View.OnClickListener() {  
        @Override  
        public void onClick(View view) {  
            Bundle activityInfo = new Bundle(); // Creates a new Bundle object  
            activityInfo.putString("coffeeId", (String) view.getTag());  
  
            Fragment fragment = EditFragment.newInstance(activityInfo);  
            getActivity().setTitle("Edit a Coffee");  
  
            getActivity().getSupportFragmentManager().beginTransaction()  
                .replace(R.id.homeFrame, fragment)  
                .addToBackStack(null)  
                .commit();  
        }  
    };  
    recyclerAdapter = new CoffeeRecyclerAdapter(activity, this, clickListener,  
                                                activity.app.coffeeList);  
    coffeeFilter = new CoffeeFilter(activity.app.coffeeList, "all", recyclerAdapter);  
}
```



# CoffeeMate 4.0+ - CoffeeFragment

```
@Override  
public View onCreateView(LayoutInflater inflater, ViewGroup parent,  
                           Bundle savedInstanceState) {  
  
    // Inflate the layout for this fragment  
    View v = inflater.inflate(R.layout.fragment_rhome, parent, false);  
  
    if (favourites) {  
        coffeeFilter.setFilter("favourites"); // Set the filter text field from 'all' to 'favourites'  
        coffeeFilter.filter(null); // Filter the data, but don't use any prefix  
        recyclerAdapter.notifyDataSetChanged(); // Update the adapter  
    }  
  
    recyclerView = v.findViewById(R.id.homeRecycler);  
    setRecyclerView(v);  
  
    return v;  
}
```



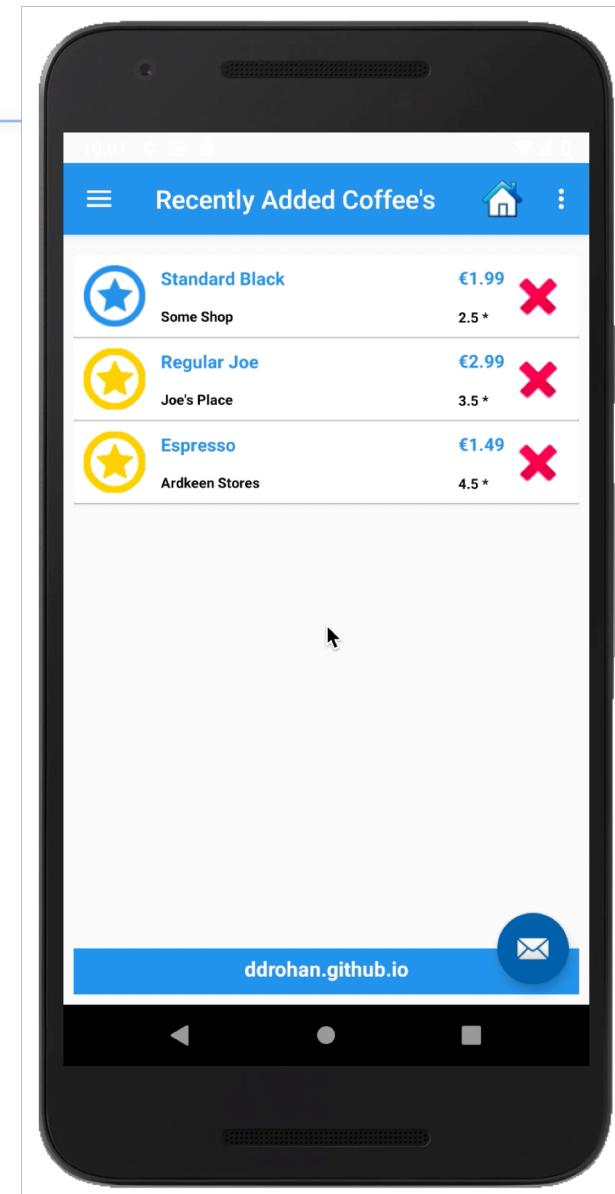
# CoffeeMate 4.0+ - CoffeeFragment

```
public void setRecyclerView(View view)
{
    recyclerView.setLayoutManager(new LinearLayoutManager(activity));
    recyclerView.setAdapter(recyclerAdapter);
    recyclerView.addItemDecoration(new DividerItemDecoration(getActivity(),
        LinearLayoutManager.VERTICAL));
    recyclerView.setHasFixedSize(true);

    ItemTouchHelper.SimpleCallback simpleItemTouchCallback =
        new ItemTouchHelper.SimpleCallback(0, ItemTouchHelper.LEFT |
                                         ItemTouchHelper.RIGHT) {
            @Override
            public boolean onMove(RecyclerView recyclerView,
                RecyclerView.ViewHolder viewHolder,
                RecyclerView.ViewHolder
                    target) {
                return false;
            }
            @Override
            public void onSwiped(RecyclerView.ViewHolder viewHolder, int swipeDir) {
                activity.app.coffeeList.remove(viewHolder.getAdapterPosition());
                recyclerAdapter.notifyItemRemoved(viewHolder.getAdapterPosition());
            }
        };
    ItemTouchHelper itemTouchHelper = new ItemTouchHelper(simpleItemTouchCallback);
    itemTouchHelper.attachToRecyclerView(recyclerView);
}
```



# Swipe-to-Delete feature





# Summary

---

- ❑ We made use of a **NavigationDrawer** to implement more effective navigation within our app (CoffeeMate 4.0+)
- ❑ We had a brief look at implementing a **RecyclerView** and **RecyclerAdapter** to replace our List.



# References

---

- ❑ <http://www.vogella.com/tutorials/AndroidRecyclerView/article.html>



---

# Questions?

