Labsheet 4 – Friday 17th September 2021

List Views and Array Adapters

Software Development for Portable Devices

Information about HD TA:

AASHITA DUTTA <h20201030130@hyderabad.bits-pilani.ac.in>

Breakout room link: https://meet.google.com/erq-uxqi-egv

Information about FD TA:

ARUNEEMA DESHMUKH <f20180568@hyderabad.bits-pilani.ac.in>

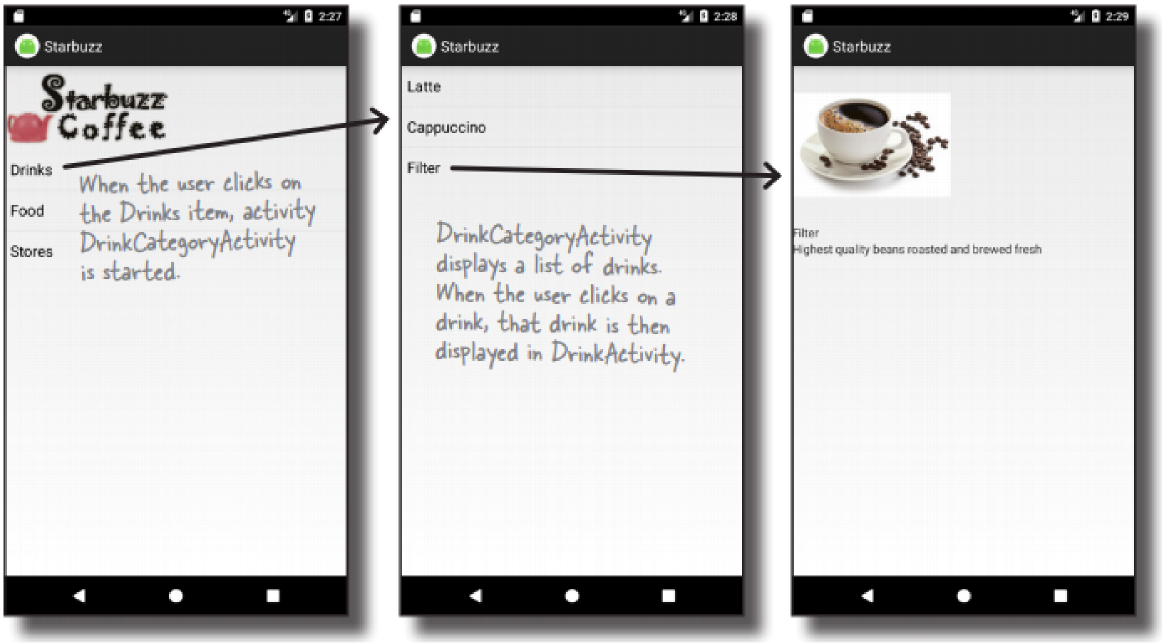
Breakout room link: https://meet.google.com/pvo-osiz-cuh

In this practical, you learn how to use list views to navigate to data. List views are commonly used to navigate between activities.

1. Create a project and add a Java class (Drink.java)
2. Add image resources
3. Use Image View and List View
   a. Make List Items Clickable using an event listener.
4. Use Array Adapter bind arrays to views.

App Description: Starbuzz - A Coffee Shop App

Top-level activity	The app will display a start screen with a list of options		
	Drinks	Food	Stores
Category activities	Display a list of the drinks that are sold	Display a menu showing all the food	Show a list of all stores
Detail activities	Show details of each drink	Show details of an item of food	Display the address and opening times of each store



Highest quality beans roasted and brewed fresh

When the user clicks on the Drinks item, activity DrinkCategoryActivity is started.

DrinkCategoryActivity displays a list of drinks. When the user clicks on a drink, that drink is then displayed in DrinkActivity.

App Structure (Only Drinks Part) **:**

TopLevelActivity *(activity_top_level.xml)* is the app's top-level activity and allows the user to navigate through the app.

DrinkCategoryActivity *(activity_drink_category.xml)* is a category activity; the activity uses a list view to display all the drinks.

The third activity, **DrinkActivity** *(activity_drink.xml)*, displays the name, an image, and a description of a given drink.

The details about the drink categories and drinks are present in the **Drink.java** class file.

Task 1: Create a new project: Starbuzz

1. Select **Empty Activity** as Project Template. After clicking next, you can name this activity (**TopLevelActivity**). [If you do not get this option, you can rename the activity later, follow Renaming an Activity]. Create two more Activities- **DrinkCategoryActivity** and **DrinkActivity**

Task 2: Add the Drink Class

The class defines an array of three drinks, where each drink is composed of a name, description, and image resource ID.

I. Right-Click on com.example.starbuzz[this will be your package name] package in the **app/src/main/java folder,** and go to File→New...→Java Class. Name it as Drink. A new class will be created in the package.

Drink.java

```
package com.example.starbuzz;

public class Drink {
   private String name;
   private String description;
   private int imageResourceId;

   //drinks is an array of Drinks
   public static final Drink[] drinks = {
           new Drink("Latte", "A couple of espresso shots with
steamed milk",
                   R.drawable.latte),
           new Drink("Cappuccino", "Espresso, hot milk, and a
steamed milk foam",
                   R.drawable.cappuccino),
           new Drink("Filter", "Highest quality beans roasted and
brewed fresh",
                   R.drawable.filter)
   };

   //Each Drink has a name, description, and an image resource
   private Drink(String name, String description, int
imageResourceId) {
       this.name = name;
       this.description = description;
       this.imageResourceId = imageResourceId;
   }
   public String getDescription() {
       return description;
   }
   public String getName() {
       return name;
   }
   public int getImageResourceId() {
       return imageResourceId;
   }
   public String toString() {
       return this.name;
   }
}
```

Questions:

1. Can you think about another app and organize your ideas as top-level activities, category activities, detail/edit activities. Can you list it down?
2. Is there any other we can hold drink data like name, description, image resources other than pure Java class?

Task 3: Add Image Resources

The Drink code includes three image resources for its drinks with IDs of R.drawable.latte, R.drawable.cappuccino, and R.drawable.filter. R.drawable.cappuccino refers to an image file called cappuccino.

1. To add image files to the project, create the **app/src/main/res/drawable** folder in your Starbuzz project (if it doesn't already exist).To do this, make sure you've switched to the Project view of Android Studio's explorer, select the app/src/main/res folder, go to the File menu, choose the New... option, then click on the option to create a new Android resource directory. When prompted, select a resource type of "drawable", name the folder "drawable", and click on OK. Get the images from https://drive.google.com/drive/folders/15mxcQAViKLOysUhid3Otoct34Uo53wk4?usp=sharing Add all the files to the drawable folder.

2. When you add images to your apps, you need to decide whether to display different images for different density screens. In our case, we're going to use the same resolution image irrespective of screen density, so we've put a single copy of the images in one folder. If you decide to cater to different screen densities in your apps, put images for the different screen densities in the appropriate drawable folders like drawable-ldpi, drawable-mdpi, drawable-hdpi, etc.

Task 4: TopLevelActivity

We need to change the layout to display an image (starbuzz logo) and a list (Drinks, Food, and Stores). We will be using the Linear Layout but feel free to use any other layout. We will implement an **event listener** to get the items in a list view to respond to clicks.

1. Use **ImageView** to display the logo image. Width = 200 dp, Height = 100 dp, image src = @drawable/starbuzz_logo. For content description use a string resource
   ```
   string name="starbuzz_logo">Starbuzz logo</string>
   ```
2. A **ListView** allows you to display a vertical list of data that people can use to navigate through the app. For now, we will add static values. Later we will make the list items clickable. To display items in the list of text views, add an array of entries to the list view by using the android:entries attribute and setting it to an array of Strings.
   ```
   <string-array name="options">
      <item>Drinks</item>
      <item>Food</item>
      <item>Stores</item>
   </string-array>
   ```

activity_top_level.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   tools:context=".TopLevelActivity" >
   <ImageView
       android:layout_width="200dp"
       android:layout_height="100dp"
       android:src="@drawable/starbuzz_logo"
       android:contentDescription="@string/starbuzz_logo" />
```

```
   <ListView
       android:id="@+id/list_options"
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:entries="@array/options" />
</LinearLayout>
```

Run the app once.

3. To get items in a list view to respond to clicks, we need to create an *OnItemClickListener* (AdapterView.OnItemClickListener) and implement its *onItemClick()* method. The OnItemClickListener listens for when items are clicked, and the onItemClick() method decides how the activity should respond to the click. The onItemClick() method includes several parameters that can be used to find out which item was clicked, such as a reference to the view item that was clicked, the item's position in the list view (starting at 0), and the row ID of the underlying data. OnItemClickListener is a nested class within the AdapterView class. A ListView is a subclass of AdapterView (AdapterView).

   We want to start **DrinkCategoryActivity** when the first item in the listview is clicked—the item at **position 0**. If the item at position 0 is clicked, we need to **create an intent** to start DrinkCategoryActivity.

   To create the listener, add this block of code to TopLevelActivity.java inside onCreate()

```
AdapterView.OnItemClickListener itemClickListener = new
AdapterView.OnItemClickListener(){
   public void onItemClick(AdapterView<?> listView,
                           View itemView,
                           int position,
                           long id) {
       if (position == 0) {
           Intent intent = new Intent(TopLevelActivity.this,
DrinkCategoryActivity.class);
           startActivity(intent);
       }
   }
};
```

4. To attach this listener to the ListView, use setOnItemClickListener() method, which takes in the listener itself as an argument.

```
ListView listView = (ListView) findViewById(R.id.list_options);
listView.setOnItemClickListener(itemClickListener);
```

TopLevelActivity.java

```
package com.example.starbuzz;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;

public class TopLevelActivity extends AppCompatActivity {

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

       //Create an OnItemClickListener
       AdapterView.OnItemClickListener itemClickListener =
               new AdapterView.OnItemClickListener(){
                   public void onItemClick(AdapterView<?> listView,
                                           View itemView,
                                           int position,
                                           long id) {
                       if (position == 0) {
                           Intent intent = new
Intent(TopLevelActivity.this,
                                   DrinkCategoryActivity.class);
                           startActivity(intent);
                       }
                   }
               };

       //Add the listener to the list view
       ListView listView = (ListView)
findViewById(R.id.list_options);
       listView.setOnItemClickListener(itemClickListener);
   }
}
```

Run the app and click on the list items.

Questions:

1) How do list views help in navigating through the app?
2) Why did we have to create an event listener? Can't we use something like android:onClick?
3) "OnItemClickListener is a nested class within the AdapterView class. A ListView is a subclass of AdapterView". Do you know the difference between subclass and nested class?

4) “OnItemClickListener is a nested class within the AdapterView class”. However, inside the AdapterView class, OnItemClickListener is declared as a public interface. So if it’s an interface, then why are we not implementing it? Can we define interface within class?

Task 5: DrinkCategoryActivity

We need to use an adapter if we need to display data in a listview that comes from a source other than strings.xml (such as a Java array or database). An **adapter** acts as a bridge between the data source and the list view.

An **ArrayAdapter** (android.widget.ArrayAdapter) is a type of adapter that specializes in working with arrays. It can be used with any subclass of the AdapterView class, such as list views and spinners.

1. **This Activity’s layout has only one component, ListView.** Add a ListView to *activity_drink_category.xml* but don’t add the android:entries attribute because if the data is held in an array programmatically created in Java code or stored in a database, then the android:entries attribute won’t work.
2. We will use an array adapter to display data from the *Drink.drinks* array in the list view. We need to initialize the array adapter and attach it to the list view.

   To initialize the array adapter, we specify what type of data is contained in the array that will bind to the list view; here it is *Drink*. Then we need to pass the array adapter three parameters: a Context (usually the current activity), a layout resource that specifies how to display each item in the array, and the array itself.

3. To attach the array adapter to the list view, use the ListView setAdapter() method.

DrinkCategoryActivity.java

```
package com.example.starbuzz;

import androidx.appcompat.app.AppCompatActivity;


import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;

public class DrinkCategoryActivity extends AppCompatActivity {

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_drink_category);
       ArrayAdapter<Drink> listAdapter = new ArrayAdapter<>(
               this,
               android.R.layout.simple_list_item_1,
               Drink.drinks);
```

```
        ListView listDrinks = (ListView)
findViewById(R.id.list_drinks);
        listDrinks.setAdapter(listAdapter);
    }
}
```

Run the app

Questions:

1. What is simple_list_item_1? Did we define it in our layout folder?
   Ref: android.R.layout
2. What is the difference between AdapterView and Adapter? What is common between ListView and Spinner?
3. How will the list display the drinks as they are Drink objects?

Task 6: Navigate to DrinkActivity

To make the list items in DrinkCategoryActivityCreate respond on clicking, create an *OnItemClickListener* and implement its *onItemClick()* method that will start the DrinkActvity.

Try it yourself: We have done it before in TopLevelActivity. However, we want to start DrinkActivity from all the list items. To display the details based on the drink, we will send extra information through the intent, which will be the row ID of the drink in the list.

DrinkCategoryActivity.java

```
package com.example.starbuzz;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.view.View;
import android.content.Intent;
import android.widget.AdapterView;
public class DrinkCategoryActivity extends AppCompatActivity {

   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_drink_category);
       ArrayAdapter<Drink> listAdapter = new ArrayAdapter<>(
               this,
               android.R.layout.simple_list_item_1,
               Drink.drinks);
       ListView listDrinks = (ListView)
findViewById(R.id.list_drinks);
       listDrinks.setAdapter(listAdapter);
```

```
        //Create the listener
        AdapterView.OnItemClickListener itemClickListener =
                new AdapterView.OnItemClickListener(){
                    public void onItemClick(AdapterView<?>
listDrinks,
                                            View itemView,
                                            int position,
                                            long id) {
            //Pass the drink the user clicks on to
DrinkActivity
            Intent intent = new
Intent(DrinkCategoryActivity.this,DrinkActivity.class);
            intent.putExtra(DrinkActivity.EXTRA_DRINKID, (int)
id);
            startActivity(intent);
                    }
                };

        //Assign the listener to the list view
        listDrinks.setOnItemClickListener(itemClickListener);
    }
}
```

Questions:

1. What is the difference between position and row id?
2. Earlier in TopLevelActivity.java we called setOnItemClickListener() using ListView object , but in DrinkCategoryActivity.java we are calling setAdapter() using ListView object. What's the difference?
3. Instead of using layout - android.R.layout.simple_list_item_1, can you try with other layouts w.r.t. Listviews? For example- what about if we want to change the background state to indicate a selected item in a list? Which layout will be used?

Task 7: DrinkActivity

Try it yourself: Display the details of the drink selected. Retrieve the data from the intent. You will need to add one ImageView for the Image and two TextViews for name and description in the layout.

Hints: int drinkId = (Integer)getIntent().getExtras().get(EXTRA_DRINKID); will give the array index of the selected drink.
Use the getter functions in drink class.
Use photo.setImageResource() and photo.setContentDescription() with ImageView photo

activity_drink.xml

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

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   tools:context=".DrinkActivity" >
   <ImageView
       android:id="@+id/photo"
       android:layout_width="190dp"
       android:layout_height="190dp" />

   <TextView
       android:id="@+id/name"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content" />

   <TextView
       android:id="@+id/description"
       android:layout_width="match_parent"
       android:layout_height="wrap_content" />
</LinearLayout>
```

DrinkActivity.java

```
package com.example.starbuzz;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.ImageView;
import android.widget.TextView;

public class DrinkActivity extends AppCompatActivity {

   public static final String EXTRA_DRINKID = "drinkId";

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

       //Get the drink from the intent
       int drinkId =
(Integer)getIntent().getExtras().get(EXTRA_DRINKID);
       Drink drink = Drink.drinks[drinkId];

       //Populate the drink name
       TextView name = (TextView)findViewById(R.id.name);
```

```
        name.setText(drink.getName());

        //Populate the drink description
        TextView description =
(TextView)findViewById(R.id.description);
        description.setText(drink.getDescription());

        //Populate the drink image
        ImageView photo = (ImageView)findViewById(R.id.photo);
        photo.setImageResource(drink.getImageResourceId());
        photo.setContentDescription(drink.getName());
    }
}
```

Questions:

1. Why are we using getExtras() instead of getExtra()?

Homework:

1. Complete the Starbuzz app to show the Food Menu and List of Stores.
   Customize it the way you want.
   Or
2. You need to design and implement an app for PVR cinemas. Make provisions to view movie lists and their details, make and view booking details.

Additional References

Using lists in Android wth ListView - Tutorial