CS 213 – Software Methodology
Spring 2023

Sesh Venugopal

Apr 5 - Android Programming

ListView/Multiple Activities/Raw resource/Toolbar/Up Navigation

Example: Rutgers Bus Routes

Project

Make an Android application project called RU NB Bus Routes (or whatever).

Choose the Empty Activity option when asked to select a template.

(You should have a
`public class MainActivity extends AppCompatActivity`)

Refactor --> Rename MainActivity to Routes, and activity_main.xml to routes_list.xml

Part 1:
Showing a List of Route Names

routes_list.xml layout

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools=http://schemas.android.com/tools
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".Routes">

    <ListView
        android:id="@+id/routes_list"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

</LinearLayout>
```

placeholder ListView, the actual list will be populated in the Java code

Routes in `strings.xml` file

```
<resources>
       <string name="app name">RU NB Bus Routes</string>
       <string-array name="routes_array">
              <item>A: College Ave/Busch</item>
              <item>B: Livingston/Busch</item>
              <item>B-HE: Livingston/Busch</item>
              <item>C: Busch Commuter Shuttle</item>
              <item>EE: College Ave/Cook-Douglass</item>
              <item>F: College Ave/Cook-Douglass</item>
              <item>H: College Ave/Busch</item>
              <item>LX: College Ave/Livingston</item>
              <item>REXB: Cook-Douglass/Busch</item>
              <item>REXL: Cook-Douglass/Livingston</item>
       </string-array>
</resources>
```

We will read these names into our `ListView` when the app executes.

See https://developer.android.com/guide/topics/resources/string-resource.html

Layout for each ListView item

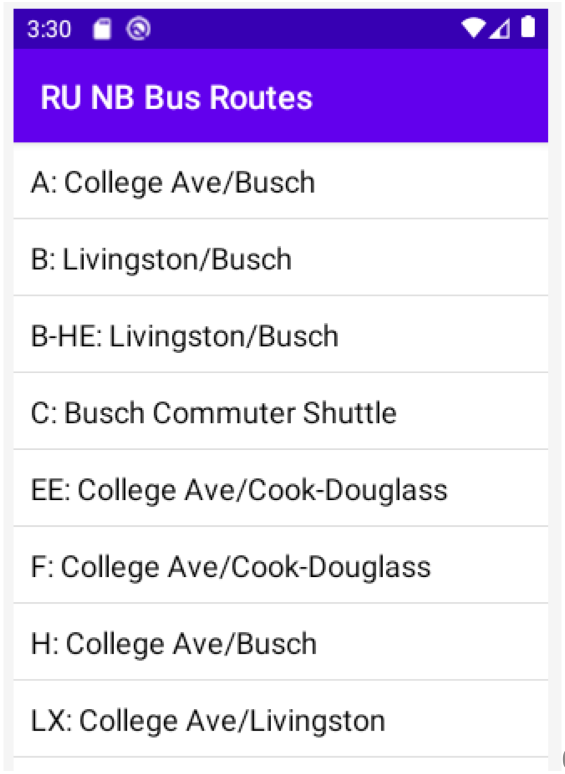
Make a file called route.xml in the layout folder, with a top level TextView tag, with values for text size, text color (e.g. black), background color (e.g. white), and padding.

route.xml

```
<?xml version="1.0" encoding="utf-8"?>
<TextView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
    android:textColor="#000000"
    android:textSize="18sp"
    android:padding="10dp" />
```

#ffffff (hex) = White (RGB = 255,255,255)

#000000 (hex) = Black (RGB = 0,0,0)


3:30

RU NB Bus Routes

A: College Ave/Busch

B: Livingston/Busch

B-HE: Livingston/Busch

C: Busch Commuter Shuttle

EE: College Ave/Cook-Douglass

F: College Ave/Cook-Douglass

H: College Ave/Busch

LX: College Ave/Livingston

Coding the main (launch) activity

When the app is launched, the `onCreate` method in the `Routes` activity will be called. See the Manifest (`app->manifests->AndroidManifest.xml`):

```
...
<activity android:name=".Routes"
          android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

This activity will need to get the list of routes from `strings.xml` and populate the `ListView`

Coding the main (launch) activity

Load the list of routes from `strings.xml` and populate the `ListView`

`Routes.java`

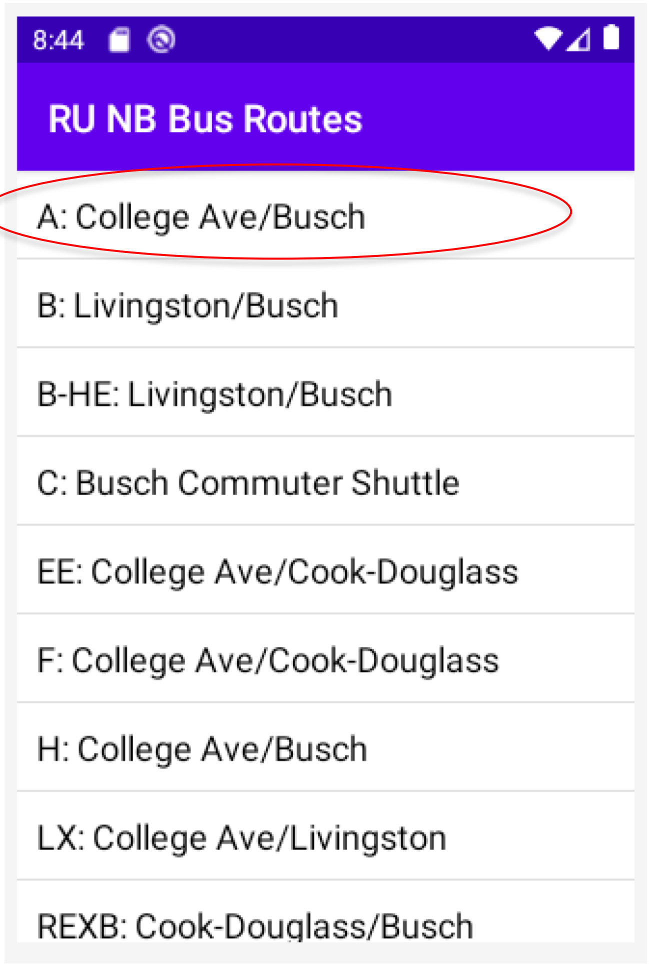
```
private ListView listView;
private String[] routeNames;

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

   listView = findViewById(R.id.routes_list);
   routeNames = getResources().getStringArray(R.array.routes_array);
   ArrayAdapter<String> adapter =
      new ArrayAdapter<>(this, R.layout.route, routeNames);
   listView.setAdapter(adapter);
}
```

See https://developer.android.com/guide/topics/ui/binding   TRY OUT THE APP!

Part 2:
Showing Details (Sequence of Stops) for Routes

List Item Selection – Showing Route Stops

When any of the list items (routes) is clicked, we want to show another screen with the details of all the stops on that route:



8:44

RU NB Bus Routes

A: College Ave/Busch

B: Livingston/Busch

B-HE: Livingston/Busch

C: Busch Commuter Shuttle

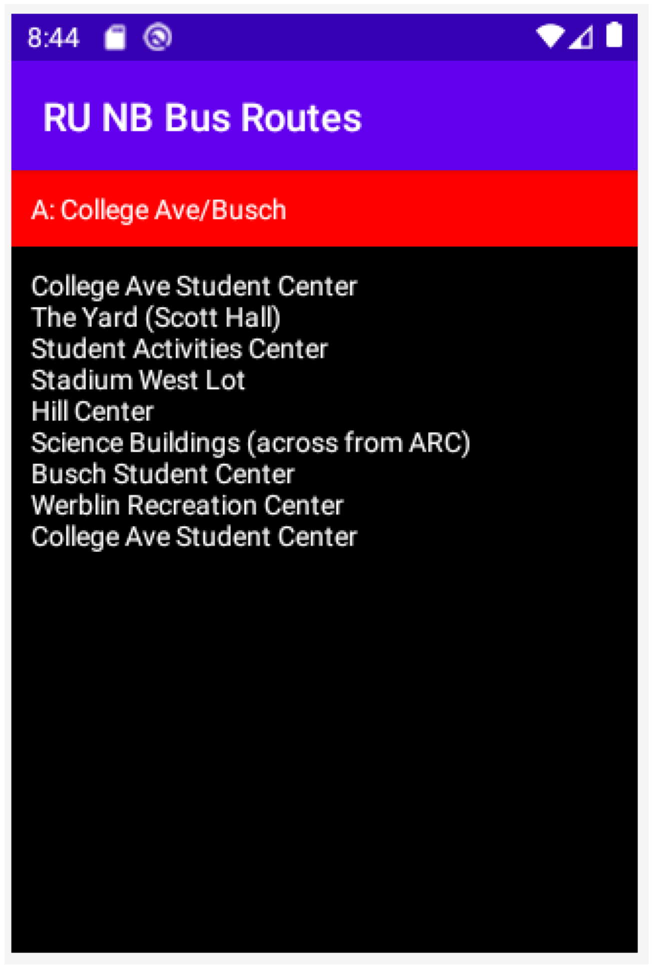
EE: College Ave/Cook-Douglass

F: College Ave/Cook-Douglass

H: College Ave/Busch

LX: College Ave/Livingston

REXB: Cook-Douglass/Busch



8:44

RU NB Bus Routes

A: College Ave/Busch

College Ave Student Center
The Yard (Scott Hall)
Student Activities Center
Stadium West Lot
Hill Center
Science Buildings (across from ARC)
Busch Student Center
Werblin Recreation Center
College Ave Student Center

List Item Selection – Showing Route Stops

This requires us to do the following:

- Get the route details from somewhere (an input file)
- Set up a click listener for the list items
- When a route is clicked, launch a new *detail activity* that has its own layout, and shows the details of the stops on the selected route
- Have the main activity pass to the new activity info on which item/route was clicked (it's *not* passed as a parameter to a method)

Setting up a read-only input file

If a file is read-only, it can be placed in the resource space of the app:

- Set up a folder (directory) called `raw` within the res folder
- Place the file (`routes.txt`) in the `raw` folder

See https://developer.android.com/guide/topics/resources/providing-resources.html

`res/raw/routes.txt`

A

```
College Ave Student Center
The Yard (Scott Hall)
Student Activities Center
Stadium West Lot
Science Buildings (across from ARC)
Busch Student Center
Werblin Recreation Center
College Ave Student Center
*****************
```

B

```
Livingston Student Center
Quads
Hill Center
Science Buildings (across from ARC)
Busch Student Center
Livingston Plaza
Livingston Student Center
*****************
...
```

Each block of lines up to
****
is the sequence of stops on a route. The blocks are in the order of the routes listed in the array resource itemized in `strings.xml`

Reading from read-only input file

The raw resource file can be loaded into the program via `R.raw.routes`

`Routes.java`

```
...
private String[] routeDetails;  ← field in Routes class
...

InputStream is = getResources().openRawResource(R.raw.routes);
Scanner sc = new Scanner(new InputStreamReader(is));
routeDetails = new String[routeNames.length];

for (int i = 0; i < routeNames.length; i++) {
    StringBuilder sb = new StringBuilder();
    String line = sc.nextLine();
    while (!line.startsWith("*")) {
       sb.append(line);
       sb.append("\n");
       line = sc.nextLine();
    }
    routeDetails[i] = sb.toString();
}
...
```

Setting up a click listener/event handling code

Routes.java

```
...
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

    public void onItemClick(AdapterView<?> parent, View view,
                            int position, long id) {
        showRoute(position);
    }
});
...
```

Or, you can use a lambda since `AdapterView.OnItemClickListener` is a functional interface:

```
...
listView.setOnItemClickListener(
    (p,v,pos,id) -> showRoute(pos));
...
```

Starting route detail activity and passing info to it

Routes.java

```
public static final String ROUTE_NAME = "route_name";
public static final String ROUTE_DETAIL = "route_detail";


private void showRoute(int pos) {
   Bundle bundle = new Bundle();
   bundle.putString(ROUTE_NAME,routeNames[pos]);
   bundle.putString(ROUTE_DETAIL,routeDetails[pos]);
   Intent intent = new Intent(this, ShowRoute.class);
   intent.putExtras(bundle);
   startActivity(intent);
}
```

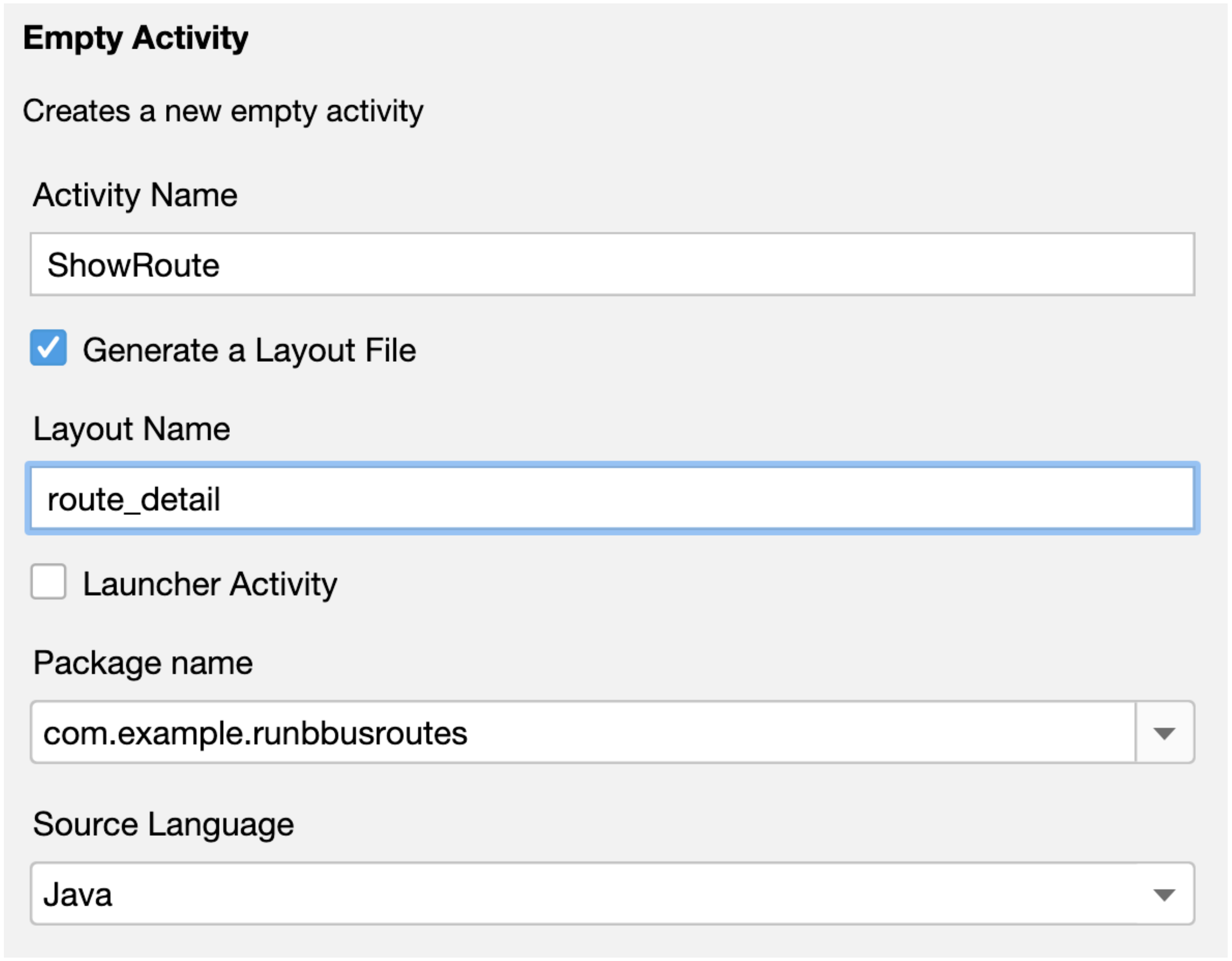
↑ Activity to launch for showing route details.

A `Bundle` can be populated with (key -> value) mappings, and can be passed to another activity as info

See https://developer.android.com/guide/components/activities/parcelables-and-bundles.html

Implementing activity ShowRoute

File -> New -> Activity -> Empty Activity



Empty Activity

Creates a new empty activity

Activity Name

ShowRoute

[✓] Generate a Layout File

Layout Name

route_detail

[ ] Launcher Activity

Package name

com.example.runbbusroutes

Source Language

Java

Designing a layout for ShowRoute

route_detail.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android=http://schemas.android.com/apk/res/android
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent”>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/route_name"
        android:background="#ff0000"
        android:textColor="#ffffff"
        android:padding="10dp"  />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/route_detail"
        android:padding="10dp"
        android:textColor="#ffffff"
        android:background="#000000"  />

</LinearLayout>
```

Red bg, white text

Black bg, white text

Need to ID route name and detail content so they can be assigned values in the Java code

Coding activity ShowRoute

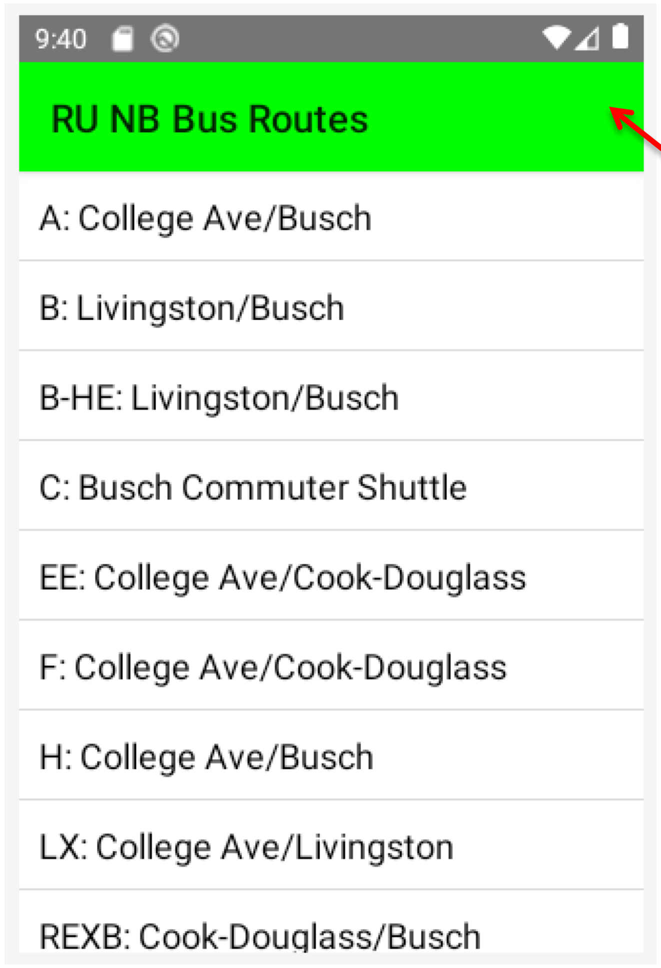
```
public class ShowRoute extends AppCompatActivity {

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

        // get route name and detail from bundle
        Bundle bundle = getIntent().getExtras();
        String routeName = bundle.getString(Routes.ROUTE_NAME);
        String routeDetail = bundle.getString(Routes.ROUTE_DETAIL);

        // get the route name and detail view objects
        TextView routeNameView = findViewById(R.id.route_name);
        TextView routeDetailView = findViewById(R.id.route_detail);

        // set name and detail on the views
        routeNameView.setText(routeName);
        routeDetailView.setText(routeDetail);
    }
}
```

TRY OUT THE APP!

Part 3:
Setting up a Toolbar with Up Navigation

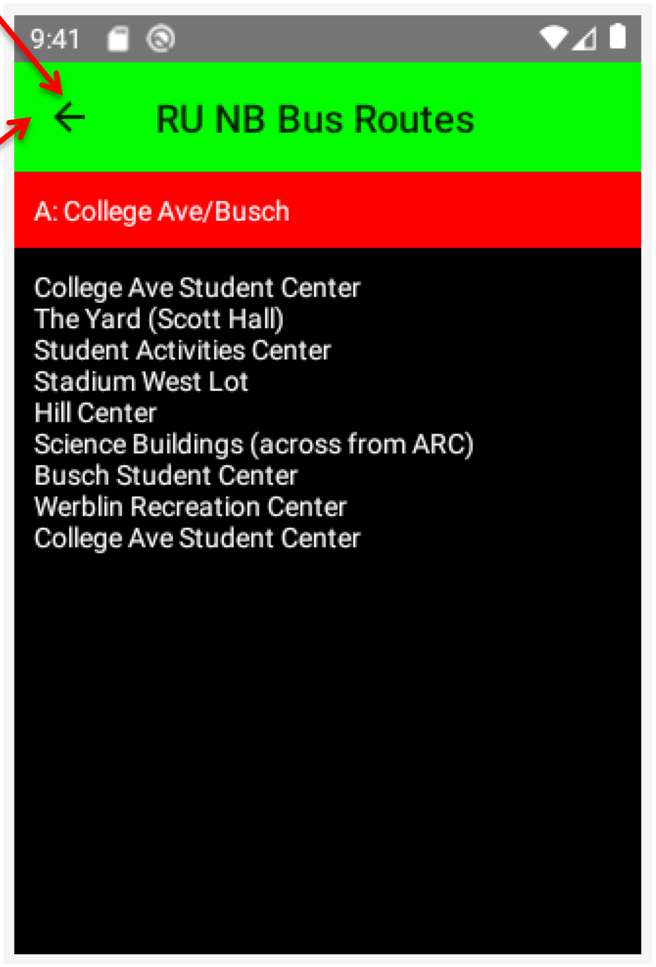
Toolbar + Up Navigation

"Up" navigation (standard for child activities), should be mapped to parent activity in manifest file



A: College Ave/Busch
B: Livingston/Busch
B-HE: Livingston/Busch
C: Busch Commuter Shuttle
EE: College Ave/Cook-Douglass
F: College Ave/Cook-Douglass
H: College Ave/Busch
LX: College Ave/Livingston
REXB: Cook-Douglass/Busch

Toolbar (used instead of native ActionBar)



A: College Ave/Busch

College Ave Student Center
The Yard (Scott Hall)
Student Activities Center
Stadium West Lot
Hill Center
Science Buildings (across from ARC)
Busch Student Center
Werblin Recreation Center
College Ave Student Center

https://developer.android.com/training/appbar

Toolbar

Add the following layout code to `routes_list.xml` and `route_detail.xml`

```
<androidx.appcompat.widget.Toolbar
    android:id="@+id/my_toolbar"
    android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
    android:background="#00ff00"
    android:elevation="4dp"
    android:theme="@style/ThemeOverlay.AppCompat.ActionBar"
    app:popupTheme="@style/ThemeOverlay.AppCompat.Light"/>
```

~~android.support.v7.widget.Toolbar~~
(don't use the support.v7 version shown in the documentation, use the appcompat version instead)

`routes_list.xml`

```
<LinearLayout … >


    <ListView … />

</LinearLayout>
```

`route_detail.xml`

```
<LinearLayout … >


    <TextView … />
    <TextView … />

</LinearLayout>
```

https://developer.android.com/training/appbar/setting-up

Add Toolbar – Java code

Add the following code to `Routes.java` and `ShowRoute.java`

```
Toolbar myToolbar = (Toolbar) findViewById(R.id.my_toolbar);
setSupportActionBar(myToolbar);
```

`Routes.java`

```
setContentView(R.layout.route_detail);


listView = findViewById(R.id.routes_list);
```

`ShowRoute.java`

```
setContentView(R.layout.route_detail);


Bundle bundle = getIntent().getExtras();
```

NOTE: The toolbar version to use is `androidx.appcompat.widget.Toolbar`

Enable Up Navigation – Java code

Add the following code to `ShowRoute.java`

```
// Enable the Up button
getSupportActionBar().setDisplayHomeAsUpEnabled(true);
```

`ShowRoute.java`

```
setSupportActionBar(myToolbar);


Bundle bundle = getIntent().getExtras();
```

We only add this to `ShowRoute.java` since it is the child activity

https://developer.android.com/training/appbar/up-action

Toolbar + Up Navigation – Manifest File

Replace existing theme with this

```
<application
    ...
    android:supportsRtl="true"
    android:theme="@style/Theme.AppCompat.Light.NoActionBar”
    ...
    >
    <activity
        android:name=".ShowRoute"
        android:exported="false"
        android:parentActivityName="com.example.runbbusroutes.Routes"/>
    ...
</application>
```

Add this so “Up” Nav knows which is the parent activity

https://developer.android.com/training/appbar/up-action

https://developer.android.com/guide/topics/manifest/manifest-intro.html

DONE!!