Level/Subject : **D3/Programming 6**
Topic : XML Drawables
Week : 9th
Activity : Creating Android apps with XML Drawables in Android Studio
Alocated time : 120 mins labs
Deliverables : Project folder
Due date : end of week

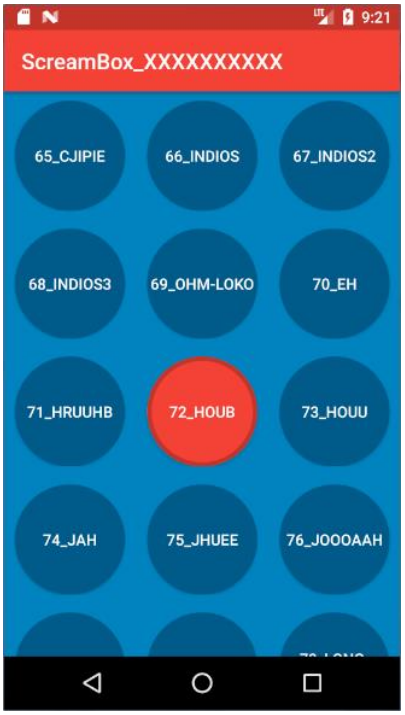
Competency :

Student expected to be able to create Android apps with XML Drawables using Android Studio IDE.

Example Practice Task:

Create an Android apps with XML Drawables using Android Studio.

1. Now that ScreamBox has been themed, it is time to do something about those buttons.
2. Currently, the buttons do not show any kind of response when you press on them, and they are just blue boxes. In this practice, you will use *XML drawables* to change the buttons.
3. Android calls anything that is intended to be drawn to the screen a drawable, whether it is an abstract shape, a clever bit of code that subclasses the **Drawable** class, or a bitmap image. In this practice, you will see *state list drawables*, *shape drawables*, and *layer list drawables*. All three are defined in XML files, so we group them in the category of XML drawables.



Making Uniform Buttons

4. Before creating any XML drawables, modify list_item_sound.xml.

list_item_sound.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<layout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <data>
        <variable
            name="viewModel"
            type="id.ac.astra.polytechnic.nimxxxxx.screambox_xxxxx.SoundViewModel" />
    </data>

    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="8dp">
        <Button
            style="@style/ScreamBoxButton"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_gravity="center"
            android:onClick="@{() -> viewModel.onButtonClicked()}"
            android:text="@{viewModel.title}"
            tools:text="Sound name"/>
    </FrameLayout>
</layout>
```

5. You gave each button a width and height of 100dp so that when the buttons are circles later on they will not be skewed.
6. Your recycler view will always show three columns, no matter what the screen size is. If there is extra room, the recycler view will stretch those columns to fit the device. You do not want the recycler view to stretch your buttons, so you wrapped your buttons in a frame layout. The frame layout will be stretched and the buttons will not.
7. Run ScreamBox and you will see that your buttons are all the same size and have some space between them.

Shape Drawables

8. Now, make your buttons round with a **ShapeDrawable**. Since XML drawables are not density specific, they are placed in the default drawable folder instead of a density-specific one.
9. In the project tool window, create a new file in `res/drawable` called `button_scream_box_normal.xml`.

button_scream_box_normal.xml

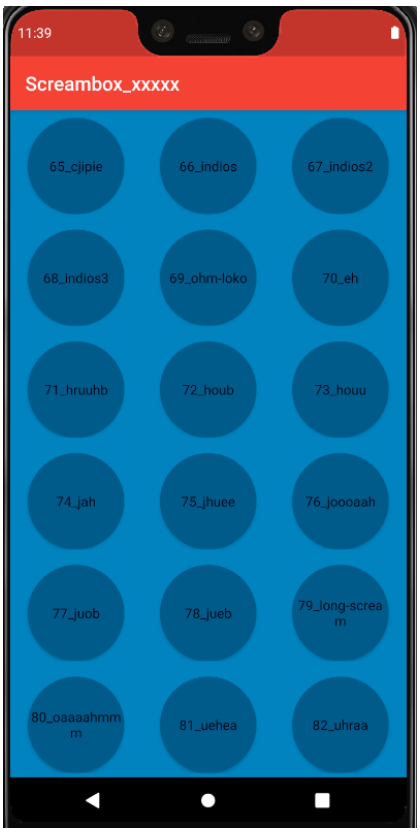
```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
       android:shape="oval">
    <solid
        android:color="@color/dark_blue"/>
</shape>
```

10. This file creates an oval shape drawable that is filled in with a dark blue color. There are additional customization options with shape drawables, including rectangles, lines, and gradients.
11. Check out the documentation at https://developer.android.com/guide/topics/resources/drawable-resource for details.
12. Apply button_scream_box_normal as the background for your buttons.

styles.xml

```
13
14      <style name="ScreamBoxButton" parent="Widget.AppCompat.Button">
15          <item name="android:background">@drawable/button_scream_box_normal</item>
16      </style>
17
18  </resources>
```

13. Run ScreamBox. Your buttons are now nice circles



14. Press a button. You will hear the sound, but the button will not change its appearance. It would be better if the button looked different once it was pressed.

State List Drawables

15. To fix this, first define a new shape drawable that will be used for the pressed state of the button.
16. Create `button_scream_box_pressed.xml` in `res/drawable`. Make this pressed drawable the same as the normal version but with a red background color.

button_scream_box_pressed.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
       android:shape="oval">
    <solid
        android:color="@color/red"/>
</shape>
```

17. Next, you are going to use this pressed version when the user presses the button. To do this, you will make use of a *state list drawable*.
18. A state list drawable is a drawable that points to other drawables based on the state of something. A button has a pressed and an unpressed state. You will use a state list drawable to specify one drawable as the background when pressed and a different drawable when not pressed.
19. Define a state list drawable named `button_scream_box.xml` in your drawable folder.

button_scream_box.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:drawable="@drawable/button_scream_box_pressed"
          android:state_pressed="true"/>
    <item android:drawable="@drawable/button_scream_box_normal" />
</selector>
```

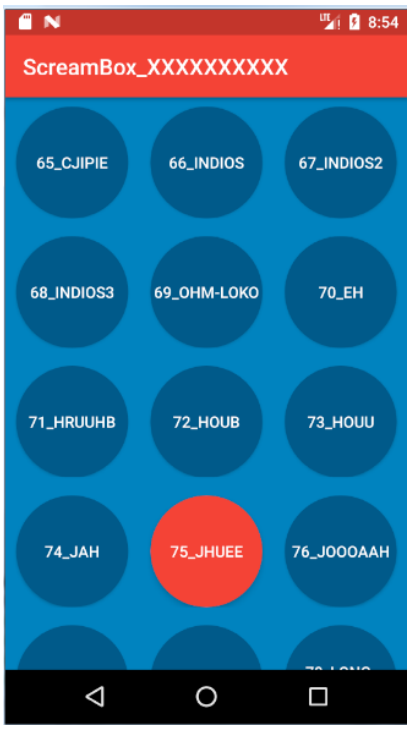
20. Now, modify your button style to use this new state list drawable as the button background.

styles.xml

```xml
    <style name="ScreamBoxButton" parent="Widget.AppCompat.Button">
        <item name="android:background">@drawable/button_scream_box</item>
    </style>

</resources>
```

21. When the button is in the pressed state, button_scream_box_pressed will be used as the background. Otherwise, button_scream_box_normal will be the background of the button.
22. Run ScreamBox and press a button. The button's background changes.



23. State list drawables are a handy customization tool. Many other states are also supported, including disabled, focused, and activated. Check out the documentation at https://developer.android.com/guide/topics/graphics/drawables#top_of_page for details.

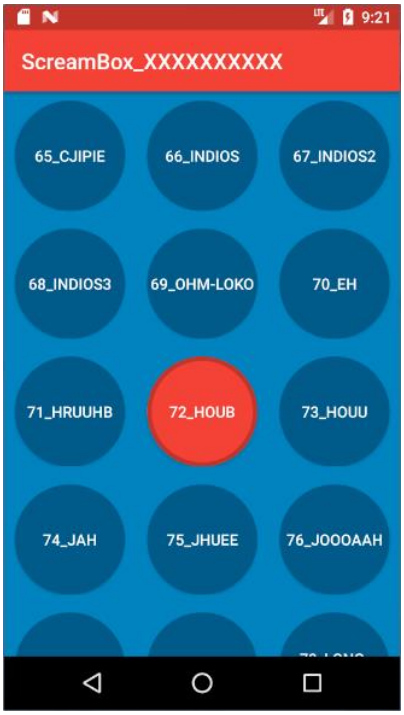
Layer List Drawables

24. ScreamBox is looking good. You now have round buttons and they visually respond to presses. Time for something a little more advanced.
25. *Layer list drawables* allow you to combine two XML drawables into one. Armed with this tool, add a dark ring around your button when in the pressed state.

button_scream_box_pressed.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<layer-list xmlns:android="http://schemas.android.com/apk/res/android">
    <item>
        <shape xmlns:android="http://schemas.android.com/apk/res/android"
               android:shape="oval">
            <solid
                android:color="@color/red"/>
        </shape>
    </item>
    <item>
        <shape
            android:shape="oval">
            <stroke
                android:width="4dp"
                android:color="@color/dark_red"/>
        </shape>
    </item>
</layer-list>
```

26. You specified two drawables in this layer list drawable. The first drawable is a red circle, as it was before this change. The second drawable will be drawn on top of the first. In the second drawable, you specified another oval with a stroke of 4dp. This will create a ring of dark red.
27. These two drawables combine to form the layer list drawable. You can combine more than two drawables in a layer list to make something even more complex.
28. Run ScreamBox and press on a button or two. You will see a nice ring around the pressed interface.



29. With the layer list drawable addition, ScreamBox is now complete. Remember how plain ScreamBox used to look?
30. Making your app a pleasure to look at makes it fun to use, and that will pay off in popularity.

Notes:

1. Create folder PRG6_M9_P4_XXXXXXXXXX.
2. Zip the folder and submit it to the server.

Bibliography:

- Marsicano, et. al., "Android Programming – The Big Nerd Ranch", 5th Ed, 2022, Pearson Technology.