Seng440 Project 1 Postmortem

Name: Luke Walsh
UserID: lwa383
ID: 33930340

Application Purpose

Drum tutor is designed by drummers for drummers. We offer a simple and refreshing way to practice fundamental rudiments with a modernised gamified approach. Drum tutor aims to go above and beyond helping you not only improve rudiment accuracy but act as a single go-to place for tightening up your grooves using our built-in metronome. At drum tutor we don't claim to be the experts on how you should play, however, we do know who is. We have done the heavy lifting for you and provided links to articles for more information on the rudiments provided from reputable resources including Vic firth and 40drumrudiments.com

Development Process

Drum Tutor development process followed a loose kanban agile development framework. All grade-bearing requirements were treated as PBIs (Product Backlog Items). At the start of each week, I would try to commit to a limited number of PBI's with the goal of completing them in the working week whilst limiting work in progress. PBI's were often but not strictly picked up in a top-down order as some PBI's required more background knowledge so were moved down the priority order. After the requirements were outlined wire-frame mockups were designed before development offering a reference point to aid with devlopment. The original rudiment trainer wire-frames[1.1] were re-worked halfway through development into new mockups [1.2] Towards the end of development, feature branching strategy was incorporated. Merging all features into dev and finally to master before the weekly update was posted.

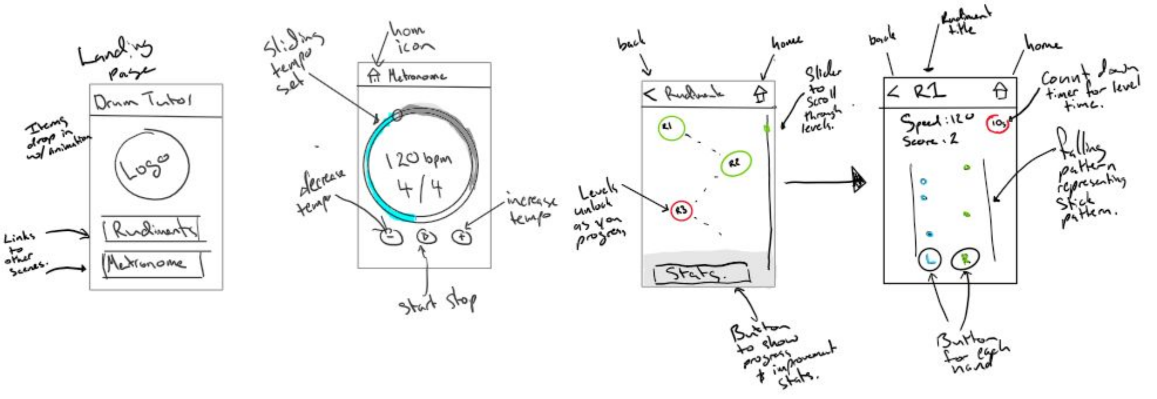


Figure [1.1] Initial wire-frame mockups

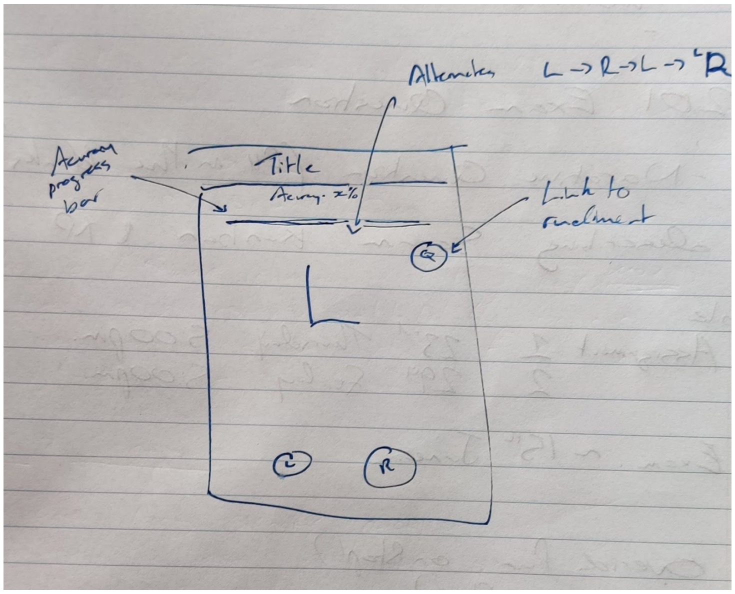


Figure [1.2] Revised rudiment trainer activity

Requirements Met

1.Compose your app out of at least three screens

The three core screens are made up of activity_main, activity_rudiment_trainer and activity_metronome shown respectively in the screenshots below [2] where each activity is spawned via an explicit intent.

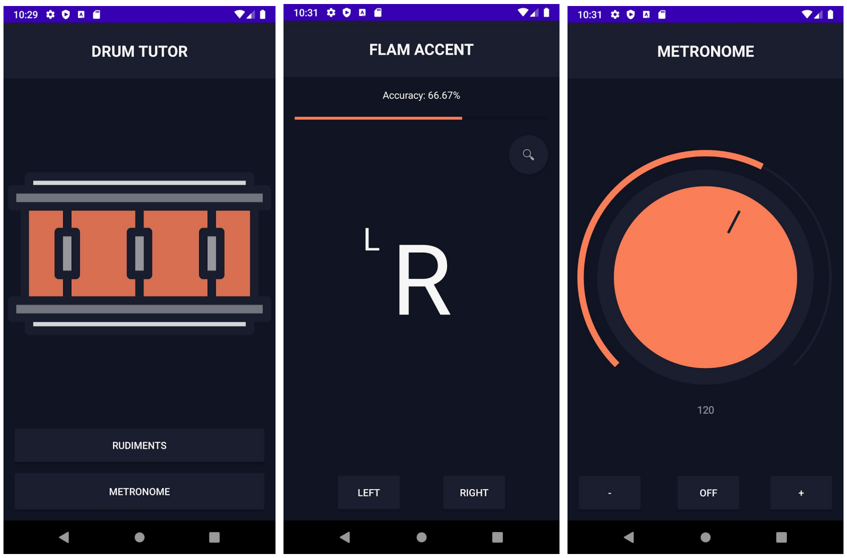


Figure [2] Core activity screen

2. Invoke at least one other app on the system via an explicit intent.

Each rudiment has a web URL attached providing more information on the rudiment provided. Webpage opened in the system web browser [3]

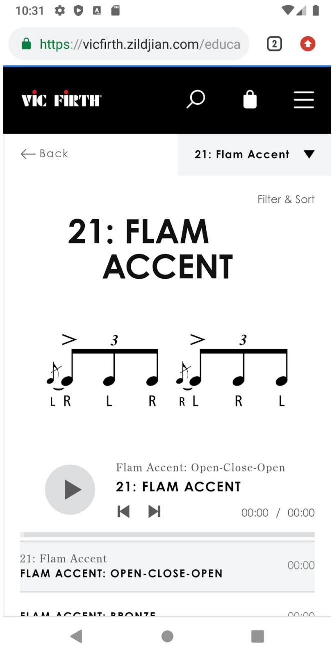


Figure [3] Flam Accent URL opened in the system web browser

3. Include a list view

Included RecyclerView found in RudimentsHome class. Scene shown in screenshots below [4]

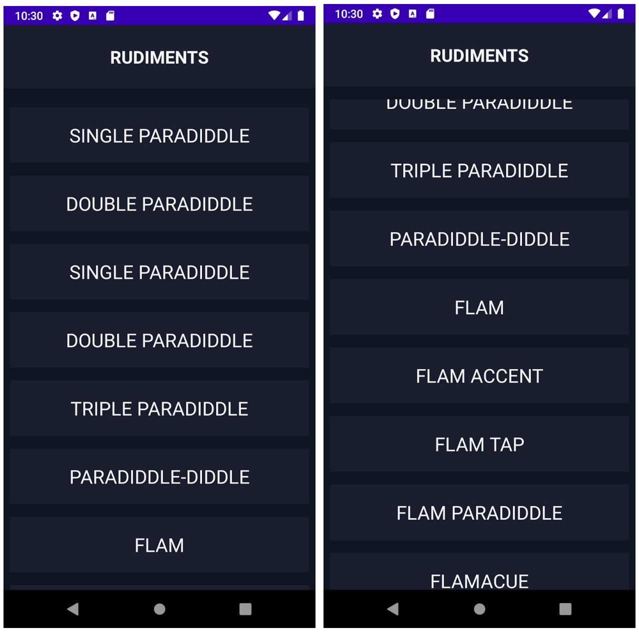


Figure [4] RecyclerView screenshots before and after scroll

4. Compose your list view using a custom adapter whose view creation method uses a custom layout.

RecyclerView custom adapter found in RudimentAdapter class and uses custom item layout where activity_rudiment_list.xml is the list ConstraintLayout and rudiment_item.xml is the Item LinearLayout.

5. Include at least five different kinds of widgets in the user interface, and handle their interactions with event listeners.

Widgets include:

a. Image View
b. Text View
c. Button
d. Toggle Button
e. Progress Bar

6. Use at least two different layout groups.

LinearLayout used for rediment_item.xml. ConstraintLayout used for everything else.

7. Support both landscape and portrait orientations in all views.

All views have automatic orientation support with each widget containing the nessacary constraints.

8. Provide separate landscape and portrait layout resources for at least one of the views.

Separate Landscape and portrait layout resources provided for activity_metronome.xml shown in the screenshots below [5].

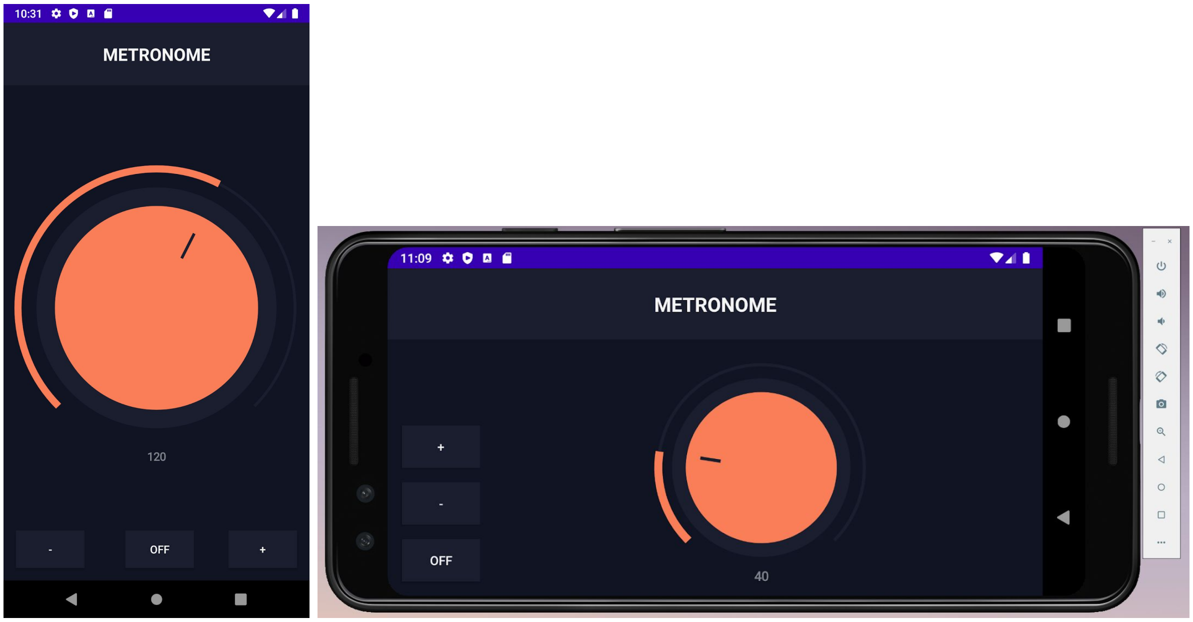
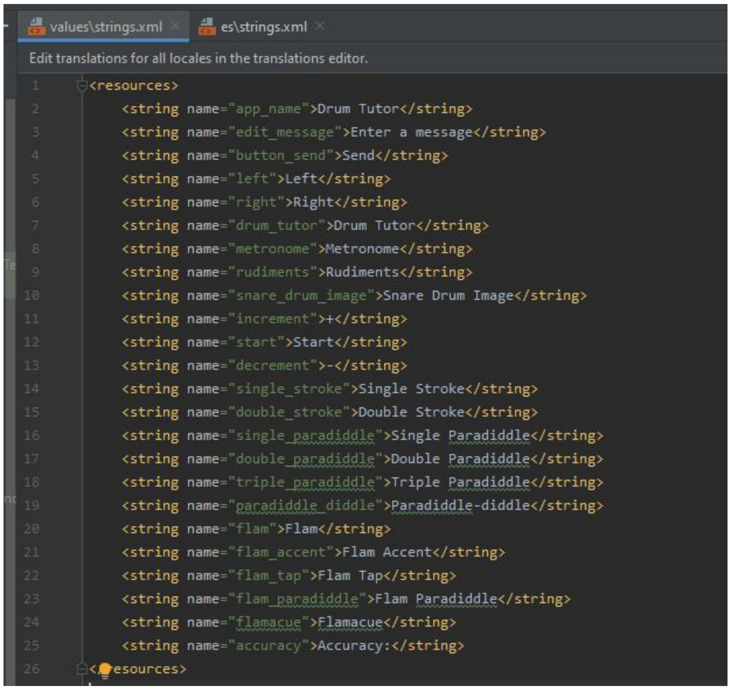


Figure [5] metronome landscape and portrait resources

9. Use string resources for all static text on the user interface.

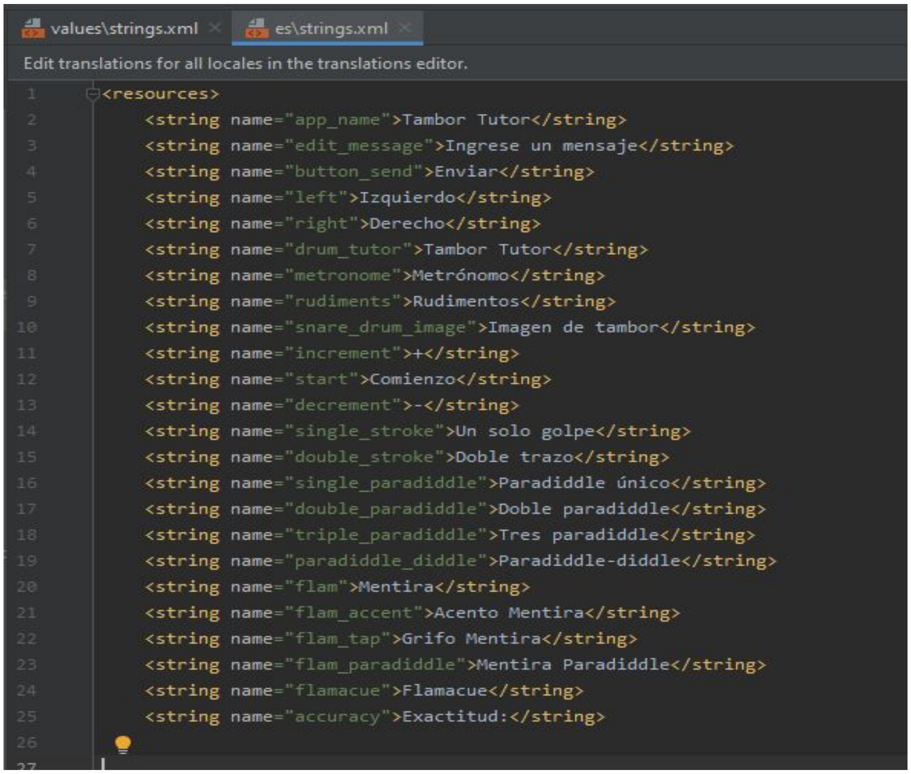


Edit translations for all locales in the translations editor.

```
<resources>
    <string name="app_name">Drum Tutor</string>
    <string name="edit_message">Enter a message</string>
    <string name="button_send">Send</string>
    <string name="left">Left</string>
    <string name="right">Right</string>
    <string name="drum_tutor">Drum Tutor</string>
    <string name="metronome">Metronome</string>
    <string name="rudiments">Rudiments</string>
    <string name="snare_drum_image">Snare Drum Image</string>
    <string name="increment">+</string>
    <string name="start">Start</string>
    <string name="decrement">-</string>
    <string name="single_stroke">Single Stroke</string>
    <string name="double_stroke">Double Stroke</string>
    <string name="single_paradiddle">Single Paradiddle</string>
    <string name="double_paradiddle">Double Paradiddle</string>
    <string name="triple_paradiddle">Triple Paradiddle</string>
    <string name="paradiddle_diddle">Paradiddle-diddle</string>
    <string name="flam">Flam</string>
    <string name="flam_accent">Flam Accent</string>
    <string name="flam_tap">Flam Tap</string>
    <string name="flam_paradiddle">Flam Paradiddle</string>
    <string name="flamacue">Flamacue</string>
    <string name="accuracy">Accuracy:</string>
</resources>
```

Figure [6] String resources for all static text on UI

10. Provide default definitions for your string resources in English and one other language.

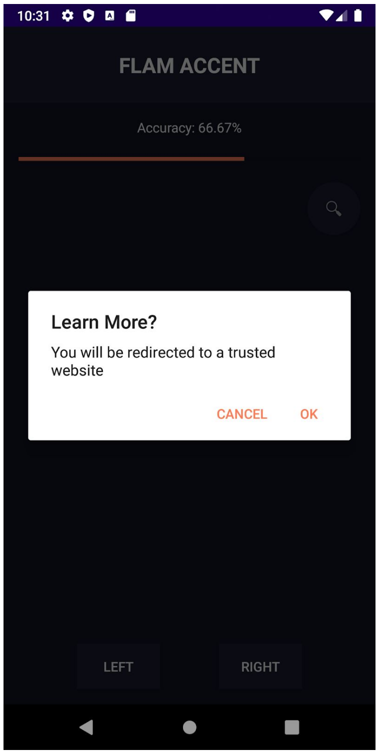


Edit translations for all locales in the translations editor.

```
<resources>
    <string name="app_name">Tambor Tutor</string>
    <string name="edit_message">Ingrese un mensaje</string>
    <string name="button_send">Enviar</string>
    <string name="left">Izquierdo</string>
    <string name="right">Derecho</string>
    <string name="drum_tutor">Tambor Tutor</string>
    <string name="metronome">Metrónomo</string>
    <string name="rudiments">Rudimentos</string>
    <string name="snare_drum_image">Imagen de tambor</string>
    <string name="increment">+</string>
    <string name="start">Comienzo</string>
    <string name="decrement">-</string>
    <string name="single_stroke">Un solo golpe</string>
    <string name="double_stroke">Doble trazo</string>
    <string name="single_paradiddle">Paradiddle único</string>
    <string name="double_paradiddle">Doble paradiddle</string>
    <string name="triple_paradiddle">Tres paradiddle</string>
    <string name="paradiddle_diddle">Paradiddle-diddle</string>
    <string name="flam">Mentira</string>
    <string name="flam_accent">Acento Mentira</string>
    <string name="flam_tap">Grifo Mentira</string>
    <string name="flam_paradiddle">Mentira Paradiddle</string>
    <string name="flamacue">Flamacue</string>
    <string name="accuracy">Exactitud:</string>
```

Figure [7] Spanish string resource file

11. Use a Toast message or dialog to alert or interact with the usera.Toasts are used in multiple places throughout the application.
Toast message on Rudiment trainer telling user they will leave the app and the web browser will be opened. Dialogue has custom colours on buttons [8]



Learn More?

You will be redirected to a trusted website

Figure [8] Toast/Dialog alert to interact with the user.

12. Use an AsyncTask to trigger some computation without blocking the user interface.
In the RudimentTrainer class, the accuracy calculation is done asynchronously with co-routines. The updateAccuracy() function is run on the Default thread running the accuracy calculation then the result is applied on the main thread to both the status bar and accuracy text display.

13. Share a plan for your app before Saturday week 2.
Posted in slack.
14. Share an update of your work before Saturday of week 3.
Posted in slack.

15. Share an update of your work before Saturday of week 4.
Posted in slack.

16. Share an update of your work before Saturday of week 5.
Posted in slack.

17. Share an update of your work in the holidays (week 6 update).
Posted in slack.

18. Incorporate an animation into your UI.
Main activity loads with multiple animations. The main logo scales up to full size in the centre of the scene after which both the rudiments and metronome buttons fall in place one after the other. The animation is also called on onStart and onResume so every time the scene is run the animation is started

19. Incorporate some other android feature.
android.media.AudioManager and android.media.ToneGenerator is used to create the sound output for the metronome. The sound is regulated using a concurrent timer task calculating the pause based on the set metronome bpm. All functionality can be found in the Metronome class.

20. Incorporate some other Android feature.
Vibration is used in the RudimentTrainer class whenever the user gets a note in the rudiment incorrect this is done with android.os.VibrationEffect and android.os.Vibrator

Extras

The Drum Tutor snare drum logo has also been added as the launch icon and circle icon displayed in the screenshot below [9]

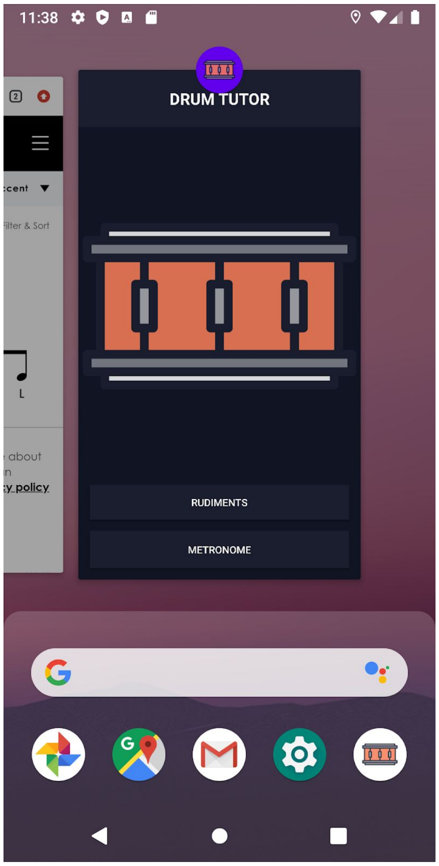


Figure [9] Screenshot showing custom icon logo.