

Laboratory – Graphics

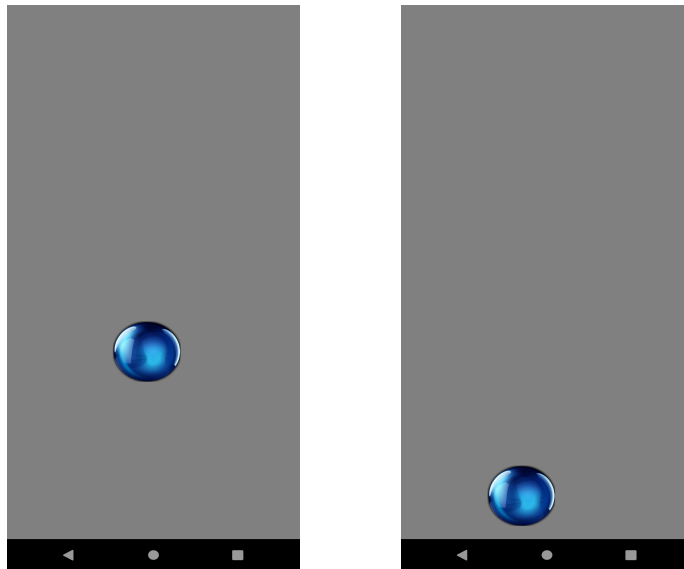
Learn about creating and animating custom Views

Objectives

Familiarize yourself with creating a custom View and then animating the movement of that View on the display. Create an application that displays the image of a bubble (or ball) that moves in a given direction on the display. When the bubble hits the edge of the display it bounces back approximately as one would expect from a real-life object. Once you’ve completed this Lab you should understand how to create custom Views, how to implement the View’s `onDraw()` method, and how to manage this app’s state across reconfigurations.

LabGraphics

This lab involves an app called LabGraphics. When it runs, the app displays a user interface like that shown below. Over time, the bubble image drifts across the display, bouncing in a natural way whenever it hits an edge of the display.



This application includes several components. It includes a ViewModel class called “BubblePositionViewModel,” which simulates the movement of the bubble and publishes the current position of the bubble as LiveData. The app also includes an Activity called “BubbleActivity.” BubbleActivity is responsible for displaying the bubble at its current position. It also provides the BubblePositionViewModel with information about the size of the display and tells the BubblePositionViewModel when to start its simulation. BubblePositionViewModel observes the lifecycle of the BubbleActivity, so that it can pause the simulation when the BubbleActivity is not in the foreground.

Note: There are Todos in both BubbleActivity and BubblePositionViewModel. See the screencast, LabGraphics.mp4, that's included in the Lab directory.

Testing

There are three test cases with several evaluation points. Each evaluation occurs at a step labelled “evaluation point”

The first test case operates as follows:

1. Start the app in portrait mode.
2. Check that the UI shows the bubble (evaluation point 1).
3. Check that the UI shows the bubble hitting an edge of the display and redirecting the bubble’s trajectory (evaluation point 2).

The second test case operates as follows:

1. Start the app in portrait mode.
2. Check that the UI shows the bubble (evaluation point 1)
3. Rotate the display.
4. Check that the bubble continues moving (evaluation point 2)

The third test case operates as follows:

1. Start the app in portrait mode.
2. Check that the UI shows the bubble (evaluation point 1)
3. Put the app into the background
4. Bring the app back into the foreground.
5. Check that the bubble continues moving from roughly the same point it was when the app when into the background (evaluation point 2)

After completing your solution, you will record a screencast while performing the manual test. Afterward, you will submit your code and the screencast via git. You can record a screencast using services available in the Logcat console. See: <https://developer.android.com/studio/debug/am-video>

Submission

When you are ready just commit your solution to your repo on GitLab by running the following commands:

```
% git add path/to/changed/files
% git commit -m “completed Lab9_GraphicsLab”
% git push origin main
```

Note: if you have not already pushed this branch to your repo on GitLab you will need to make a slight modification for this first time and run this instead:

```
git push -u origin main
```

This sets up tracking between your local branch and a branch with the same name on your repo in GitLab.

Some Implementation Notes

We are providing template code and layout resources for this application. Only modify the areas marked with the word `TODO`.

We have done our testing on an emulator using a Pixel 5 AVD with API level 31. To limit configuration problems, you should test your app against a similar AVD.