CSCI 5708
Assignment 2

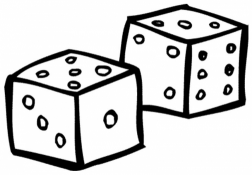
Instructor: Michael Hackett

Due: 11:30pm, Monday, June 10, 2019

The purpose of this assignment is to give you some experience developing a simple Android application.

1 Background

Dice are an important part of many games. However, on a crowded game table, real dice can be a problem. Hence, electronic dice are the wave of the future! An application that rolls virtual dice would simplify all our lives. The problem is that different games require different numbers of dice, and in some cases different types of dice. In most cases a die is a six sided cube, numbered 1 through 6, typically represented by dots.



Your task is to write a dice-rolling application for Android devices.

2 Your Task

Implement an Android application that allows the user to roll one or more dice. The application should be a **native application**, implemented in Java or Kotlin. The application should meet the following **minimum** requirements:

- The interface should allow the user to specify the number of dice to roll.
- The interface should have a button that allows the user to start a dice roll.
- Whenever the button is pressed, the application should roll the specified number of dice.
- The user should also be able to shake the device to start a roll.
- The dice rolls should be randomly generated.
- The application should display the results of the roll.
- The application should allow the user to roll again.

Naturally, you may choose to improve on this interface. For example, colour, sound effects and animations add polish to the application. *Marks will be awarded partly on how polished the application is.*

Tip: You can test the "shake to roll" functionality in the Android Emulator, by going to the Settings, choosing "Virtual sensors", then "Rotate" or "Move" (depending on how you implement your detection scheme—"Move" only affects the accelerometer; "Rotate" also triggers gyroscope changes), and then quickly dragging one of the sliders back and forth. See Figure 1.

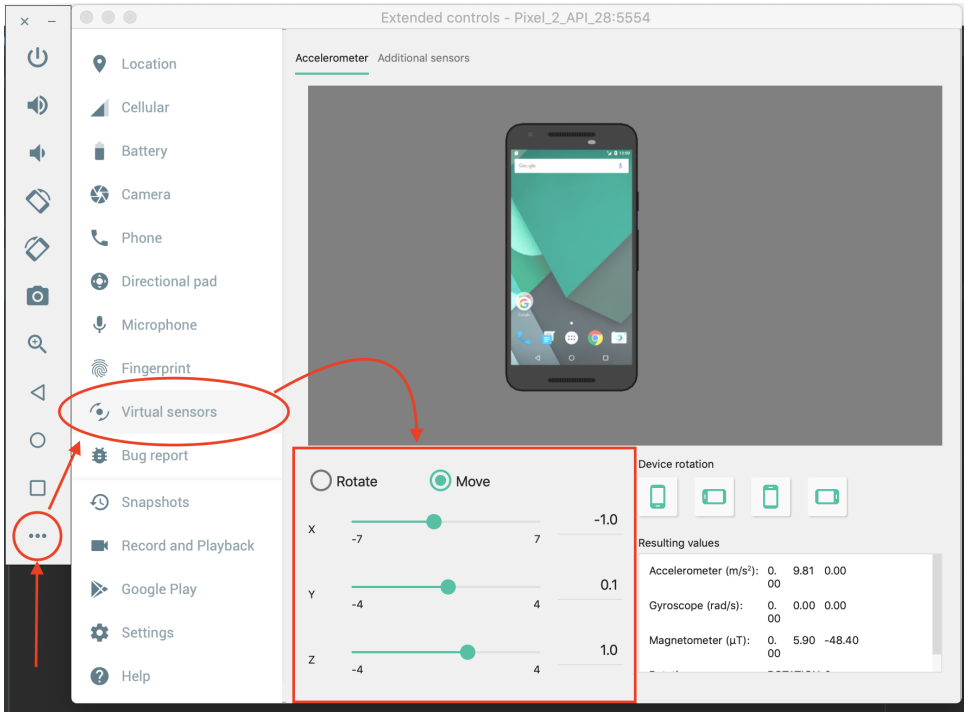


Figure 1: Using virtual sensors in Android emulator.

For exceptional functionality allow the user to create their own dice by i) specifying the number of sides on the dice, and ii) specifying the value (or figure) on each side. For example, many table-top role-playing games also use 4-, 8-, 10-, 12-, and 20-sided dice. And Cranium[1] uses a 10-sided dice where each side is either red, yellow, green, blue, or purple.

3 What to Hand In

Submit a zip file (via `https://dal.brightspace.com`) containing the Android Studio project directory (including all source code files) for your assignment, including an executable APK (in the top-level folder). Please indicate in a `README` file (in the root of the directory) what functionality is present (including being able to use the accelerometer and/or gyroscope to initiate a roll, if you were able to complete this) as well as any known bugs.

[1] Published by Hasbro

4 Grading

The assignment will be graded using the rubric given in Table 1.

	Exceptional: A	Acceptable: B	Substandard: C-D	Unacceptable: F
Functionality (40%)	Application meets all functional requirements. Application does not crash. Application has significant and useful additional functionality.	Application meets all functional requirements. Application crashes rarely or not at all.	Application meets some but not all functional requirements. Application crash often.	Application does not work.
Usability (20%) screen layout, intuitive, consistent interface, etc.	The application is intuitive and easy to use. The screen structure and layout are well organized and most use cases have been anticipated. The interface is consistent.	The interface is consistent. Most common use cases have been anticipated.	Some common use cases have been anticipated and the screen layout includes all controls necessary for the application. The application is usable.	The interface is disorganized and does not correspond to the application's function. The application is not usable.
Polish (20%) background, icons, button images, transitions, themes, special effects, etc.	The application is very polished and engaging, with icons, images and special effects to engage the user. The application looks professional.	The application is somewhat polished. Examples where polish has been applied are evident, e.g., images and icons. Additional polish is necessary before the application can be released.	A minimal amount of polish has been applied. The application uses standard buttons and text fields. Basic polish such as application icon are present.	No application polish is evident.
Code (20%) organization, commenting, readability, etc.	Code is very well organized, commented, readable, and maintainable. Code looks professional.	Code is competently organized, commented and understandable. Requires minor tuning to bring it up to professional looking code.	Code is poorly organized, uncommented, or confusing. Requires significant effort to bring it up to professional looking code.	Code is unintelligible.

Table 1: Rubric for Assignment 2.