**Winter 2021**

This offering has a final project assignment in which you can build something of your own choosing. You can build in Android, IOS, or if you want use a framework like REACT Native as well. What you build is up to you but we want you to describe it as an assignment first and then provide the solution.

Using one of your course assignments 1,2 or 3 as a model make up an assignment that you describe and list 10 itemized requirements. For this part write up the assignment description as though you want someone else to build it. Then provide the coded solution that implements your 10 stated requirements. Finally provide a screen capture video with sound that demonstrates your code and demonstrates that the itemized requirements were met.

If you use frameworks additional to the usual Android Studio or Xcode that we use for the course tutorials make sure to include installation instructions so we can launch and run your code as well.

**Assignment Grading:**

The project assignment is graded out of 35 as follows

5 marks for a written description of the assignment. That is, describe the scenario and background and introduce what you want someone to build.

10 marks for provide 10 itemized requirements numbered **R1.1**... **R1.10**. One mark each judged on how appropriate they are to for the scenario and whether they are at the level appropirate for the course.

20 marks for your code and implementation and demonstration of the 10 itemized requirements. We will award up to 2 marks each for the itemized requirements as follows.

0 marks -not implemented.  
1 mark -partial completion, or full iplementation but not demonstrated on video.  
2 full completed and demonstrated on in the screen capture video.

**Background**

The purpose of this assignment is to have you do some advance Swift programing using an app that retrieves some sort of data from an API and display it on the app. The assignment is based on class Tutorials 9A (touch events), 9B (multi-touch and gestures), 10A (Web Access Part A) , 10B (Web Access Part ) .

In this assignment we will build a stock game that a user can use to look at different stocks.

**Assignment Requirements**

We want you to build an IOS-based tic tac toe game that is played by drawing on the device screen with your finger, or mouse in the case of the simulator. We are going to use the pencil-on-paper metaphor meaning the user experience should be similar to them playing the game with pencil and paper. We are thus NOT going to employ buttons or menus but do everything by drawing on the screen or using simple on-screen gestures like tap, double tap, long tap. etc. We don't want any GUI gadgets on the user interface.

We want you to build an IOS-based stock tracker app that is interacted by using simple gestures like tap, double tap etc.

# Design Requirements

**R1.1** The game should be built with Xcode 12.x, and Swift 5 or later. You can restrict the game to use gestures supported by the simulator if you want. The game should provide a two-player tic tac toe game played by two people on the same device taking turns to draw on the screen (the application does not play as one of the opponents).

**R1.2** When the game launches the screen should be completely empty. A user should then be able to draw the game board using a two-finger vertical drag and a two finger horizontal drag. Once the game board has been drawn the game should be ready to receive player inputs. (On the simulator it can take a while for the "blank" screen to load. You can put something on the screen so the user knows the app is loaded and ready.) Also on the simulator you can do a two-finger drag by holding down the option key when you drag your mouse.

**R1.3** When the game board has been constructed the game should acknowledge this by printing a caricature of the empty game board on the console output. (or at the bottom of the game screen if you prefer). Use a caricature made of mono-space font characters like courier.

**R1.4** After each player plays a new caricature should be printed which represents the state of the board. For example (again this can be on the console or a corner of the game screen if you prefer):

**R1.5** The first player should be able to start with either an X or an O. After that the players play in turn. The game should determine whether the correct symbol (X, or O) has been drawn by the user with their finger.

**R1.6** The player should enter their symbol by drawing it with a finger, or mouse:

**R1.7** The game should retain the shape they draw for their X or O symbol and use that shape in the game board display. That is, don't replace it with a font character.

**R1.8** The game should determine whether they drew an X, an O, or some other (non-legal) symbol. The game should use the touch points and determine whether what was drawn was sufficiently like an X or an O. Also when a player plays the game board caricature should update.

**R1.9** If the player plays out of turn or draws a shape that is not sufficiently like an X or an O the shape should be erased and the current player's turn should still be in effect (allowing them to draw another shape).

**R1.10** If the game ends with someone winning the "winning line" should be drawn by the game (the application should recognize this situation):