Mobile Application Assignment 3

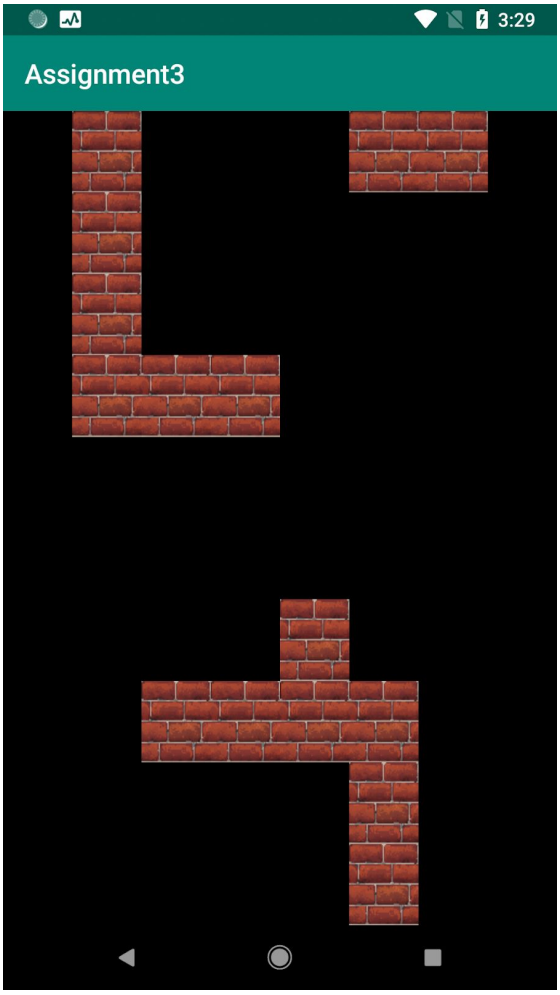
Instructors: Legand Burge, Asad Ullah Naweed
Due Date: 09/27/2018

Be sure to click on the links in this doc to view more information about certain topics. These may or may not help you in your lab, but can always provide some extra context and information.

Initial Setup and Sprite Artifacts

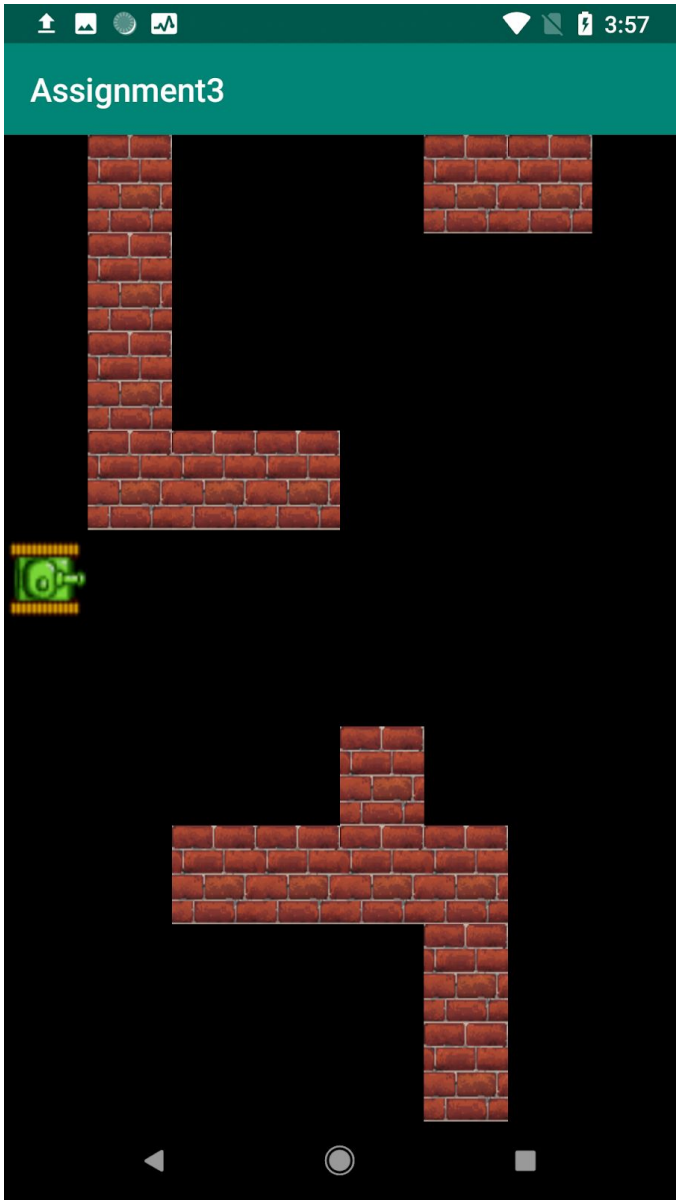
For this assignment, you'll be making a full-fledged two player game. This will be a two-player game in which you will need to somehow communicate with another player. The game will be a 2D tanks shooting game, in which you will move tanks around in a 2D grid, where the tanks can only move in the left, right, top and bottom directions.

You will need to set up a maze for your tanks containing brick walls:



This is just an arbitrary arrangement. You can use any initial arrangement you want. You can use this sprite for the bricks.

You then need to add in the tanks. Add in two tanks, but for the first part, assume that only the first tank is moving. You should use this sprite sheet to draw your tanks.



Moving the tank

The tank should move from one cell to another based on touch input. The following are rules that need to be implemented:

- The tank cannot move beyond the screen, or through walls.
- The tank should move smoothly from one cell to another.
- While moving, the tank cannot stop until it has reached its destination cell. It can, however, start moving in the opposite direction if prompted by the user.
- The tank cannot make a 90-degree turn while moving. It can only make a 90-degree turn if it is stationary, or when it was moving and has reached its destination cell.

There are many different ways you can use touch to control your tank, so feel free to implement whatever scheme suits you the best. See this website for more information on handling touch-based gestures in Android.

One alternative to raw touch gestures would be to have a row of buttons underneath the play area, with each button corresponding to a single command. This would be simpler to implement, if you want to get started quickly.

Shooting Fireballs

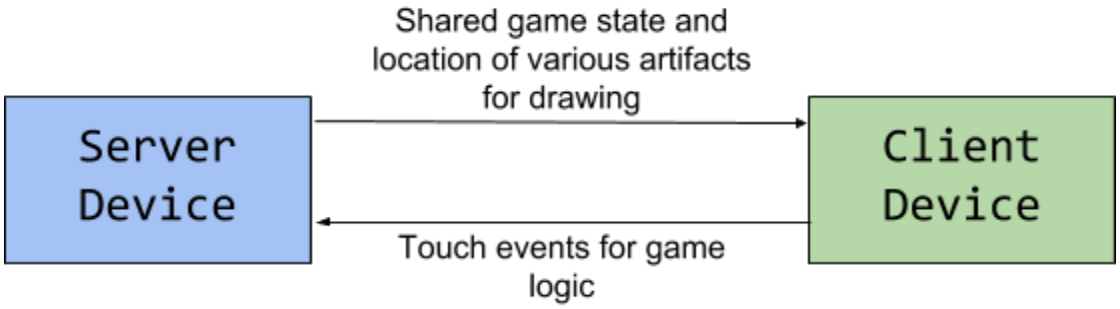
The tank should also shoot a fireball in the direction that it is currently facing. The fireball should move in a straight direction until it encounters a boundary, a brick wall, or a tank. The following rules should be implemented:

- If a fireball hits a brick wall, the fireball should disappear, and the brick wall should become a **damaged** brick wall. The damaged brick wall should be drawn as a slightly differently colored variant of the standard brick wall.
- If a fireball hits a damaged brick wall, the fireball and damaged brick wall should both disappear.
- If a fireball hits a tank, the player who shot the fireball gains a point.
- If a fireball hits a boundary wall, it should just disappear.
- A tank cannot have more than fireball on the screen at a given time.

Multiplayer

Once you have your single-player version of the game working, with the second tank doing nothing at all, you'll now need to set up peer-to-peer networking to make this work. An ideal way to do this is to have one device be the server, and have another device attempt to connect to the server device. The way to do this is to use Sockets.

Sockets are bi-directional channels of communication that enable the two endpoints to transfer bytes in any manner you want. For this game, in order to maintain a synchronized game state between the two devices, you will want the game logic to be executed on only the server device.



To set up these sockets, you should take a look at the ServerSocket and Socket classes. Other valuable resources include this tutorial, which might be a little dated, but is still relevant.

It is worth noting that you need to somehow query the IP address of the server device, and be on the same wifi network in order to set up these sockets. You can do away with all of these requirements by instead using the Android libraries for Wifi Direct connections. While the code for this is more advanced, it will result in a significantly smoother user experience.