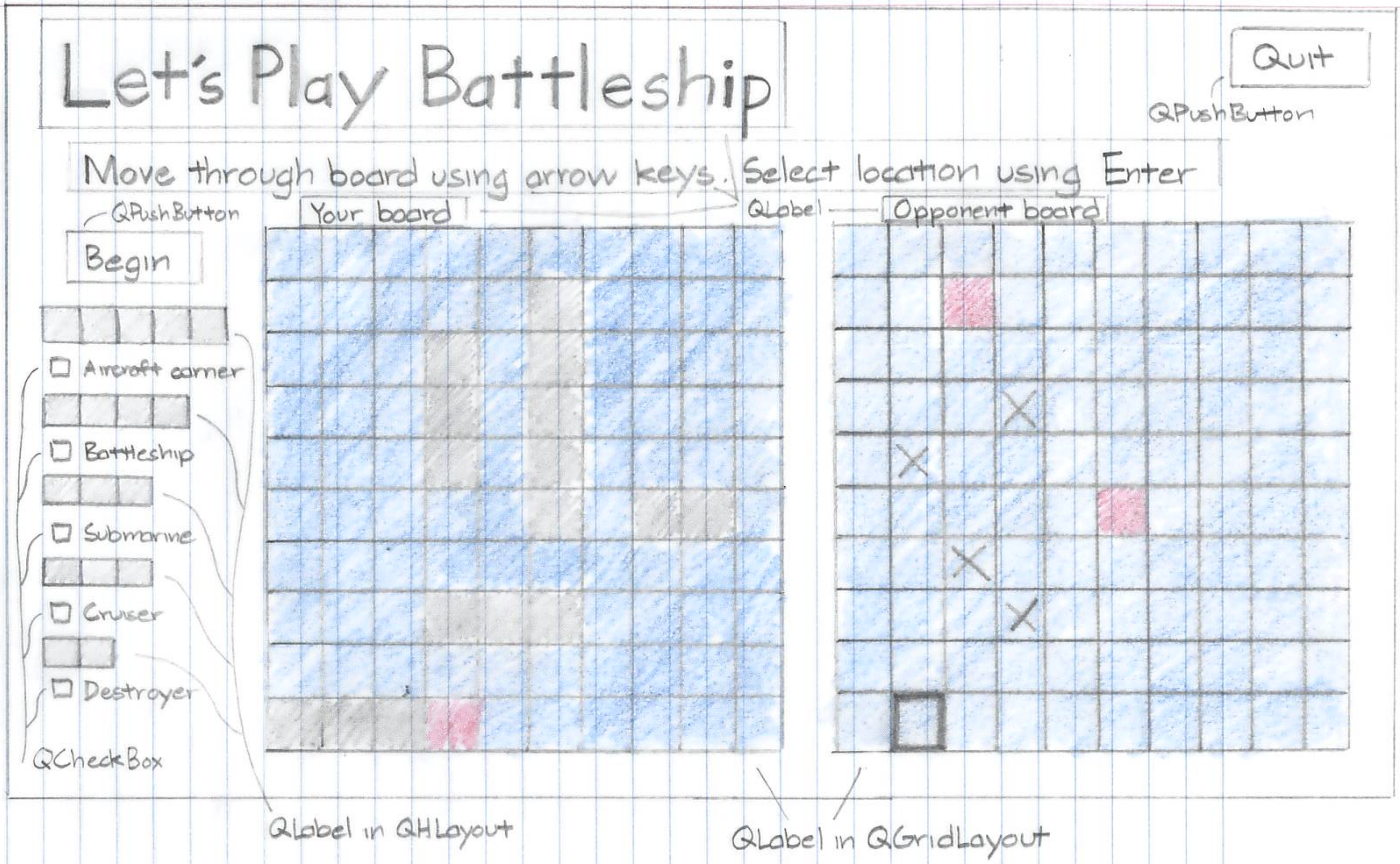
Created by Norris Khoo 004-420-262

Part 1



Let's Play Battleship

Quit

Move through board using arrow keys. Select location using Enter

Begin

Your board

Opponent board

□ Aircraft carrier

□ Battleship

□ Submarine

□ Cruiser

□ Destroyer

Created by Norris Khoo 004-420-262

Part 2

I will be creating a single player version of the board game Battleship. In this game, players set ships of varying lengths on a grid only they can see. Afterwards, each player takes turns guessing locations on their opponent's grid. If an enemy ship occupies said location, the opponent must denote that their ship has been hit. The player who destroys all of their opponent's ships first by hitting all tiles occupied by them wins the game.

For my Qt port of the game, the main window consists of two boards each represented by 100 QLabels arranged in a square grid. The board on the left is for the player and before they press QPushButton “Begin” to start the game, they must arrange their ships. Movement through the board is accomplished by arrow keys and will be tracked by a thick black border that iterates through the grid as the player dictates. To set their own ships on the board, the player will select locations on the grid and press Enter. After setting each ship, they must check the corresponding QCheckBox on the left to ensure that the location is valid (i.e. no overlap or diagonal placement). If the location is not valid, the ship placement denoted by gray QLabel will not be saved and the player will have to reset their ship then press QCheckBox again. Once all the checkboxes have been filled, the player can press “Begin” to start the AI.

Gameplay then moves to the right board and the player selects locations on the opponent's grid using the same arrow keys/Enter setup. One guess by the player corresponds with one guess by the opponent. The AI works by using a random number generator to guess a spot on the player's grid. If it happens to guess correctly, the next turn it will guess one spot from the correct tile (25% chance for each of the four directions). If it guesses wrong, it will repeat checking the remaining directions until another tile has been found. Once it is correct again, the next turn it will guess for either tile in the same linear direction (i.e. top down) for the correct set. By process of elimination, it will continue in this linear path where correct tiles are found until it reaches errors on both sides. Afterwards, the AI will resume random guessing until another ship has been found. For both boards, hits are marked by changing the tile to red and misses are marked by changing the tile to one with a X. Varying difficulty may be implemented later by changing the effectiveness of the AI (i.e. purposely programming it to not follow up on hits).