OFX Coding Exercise

Battleship State Tracker

Practicalities

Your take-home coding exercise is designed to three to four hours to complete. You are under no obligation to take any specific amount of time to complete the task.

The rules

The code you write should be your own and should be written without direct assistance. However, feel free to use as many reference resources (Stack Overflow, MSDN, Google, textbooks) as you like.

The application should be completed in C#.

Please submit your solution as a link to GitHub / GitLab project. Ideally it will contain project definitions, e.g. if using Visual Studio, provide a project that can easily be loaded and run. If any specific instructions are required, please include a readme.

The application should be deployed and available publicly

Please deploy your application to a web server and provide us a link to the application so that we can see it working. You can choose any hosting provider you deem suitable.

Background

This exercise is based on the classic game “Battleship”.

- Two players
- Each have a 10x10 board
- During setup, players can place an arbitrary number of “battleships” on their board. The ships are 1-by-n sized, must fit entirely on the board, and must be aligned either vertically or horizontally.
- During play, players take turn “attacking” a single position on the opponent’s board, and the opponent must respond by either reporting a “hit” on one of their battleships (if one occupies that position) or a “miss”
- A battleship is sunk if it has been hit on all the squares it occupies
- A player wins if all of their opponent’s battleships have been sunk.

The Task

The task is to implement a Battleship state tracking API for a single player that must support the following logic:

- Create a board
- Add a battleship to the board
- Take an “attack” at a given position, and report back whether the attack resulted in a hit or a miss.

The API should not support the entire game, just the state tracker. No graphical interface or persistence layer is required.

What we are looking for

There is no one “correct” solution, and there is no trick. The focus is not to use fancy algorithms. Instead, we are looking for thoughtfully written, high quality software.