DeploymentController.vb

- Script that handles player’s actions and UI during deployment phase.

DiscoveryController.vb

-Script that handles player’s actions and UI during battle phase.

EndingGameController.vb

-Script that handles interactions at the end of a game

GameController.vb

-Script that handles user input, displaying the current state of the game, and control of the game

GameLogic.vb

-Script that handles the initialisation of the game, and the main game loop.

GameResources.vb

-Script that handles the loading of all game resources, including images, fonts, and sounds. Also compiles these into public dictionaries to be accessed as requested by other scripts.

GameState.vb

-Script containing a simple enum for all possible game states

HighScoreController.vb

-Script that controls the formatting, loading, saving, displaying and inputting of high scores.

MenuController.vb

-Script that handles the drawing and input of menus within the game, including the main menu, game menu and the settings menu.

UtilityFunctions.vb

-Script that contains a number of utility methods and constants for the game, including cell width and height constants, colours for different ships, the drawing of the grid and ships, and so on.

Model Folder Contents:

AIHardPlayer.vb

-Script that inherits from AIPlayer. Handles the AI of the enemy player. The AI can ‘learn’ ship locations from hits, and can determine if multiple ships are being hit based on hit locations.

AIMediumPlayer.vb

-Script that inherits from AIPlayer. Handles the AI of the enemy player. The AI will simple try to destroy a ship if it has found one, without any further ‘thinking’.

AIOption.vb

-Script that contains an enum for AIOption, defining ‘easy’, ‘medium’ and ‘hard’ as possible values.

AIPlayer.vb

-Script that inherits from Player. Handles AI of the enemy player. The AI will generate coordinates, and fire randomly at said co-ordinates.

AttachResult.vb

-Script that collects information about an attack, then displays the result in text.

BattleShipsGame.vb

-Script that checks battleship game logic, i.e. ensuring two players are in a game, all ships deployed before start, swapping turns, and checking if players are destroyed.

Direction.vb

-Script that contains a simple enum defining Direction for ship orientation, namely ‘LeftRight’ or ‘UpDown’.

ISeaGrid.vb

-Script that defintes the read only interface of a grid, allowing players to see and attack their opponent’s grid.

Player.vb

-Script that defines a Player and what objects they have access to. Namely, A Player has a grid, can see enemy grids, can check if all theirs ships are deployed or destroyed.

ResultOfAttack.vb

-Script that defines the enum ResultOfAttack. These values can be ‘Hit’, ‘Miss’, ‘Destroyed’, ‘ShotAlready’, and ‘GameOver’.

SeaGrid.vb

-Script that controls what a grid contains, in addition to interactions and controls item placement in the grid, such as ships or targeting markers. A Grid is made up of multiple tiles, has a predefined width and height, and can return ships killed, checks if all ships are deployed, and the contents of a tile within the grid.

SeaGridAdapter.vb

-Script that controls the updating of specific sea grid tiles. When a change occurs in a tile, such as a ship sinking, this class is responsible for updating the visuals.

Ship.vb

-Script that controls each individual ship, what it knows, and what it can do. Ships know their name, their size, how many times they’ve been hit, their position, and their direction/orientation. They can return all of these values to outside classes via public methods, and clear the tiles they’re placed.

ShipName.vb

-Script that contains an enum for each ship name. These values are ‘None’ (0), ‘Tug’ (1), ‘Submarine’ (2), ‘Destroyer’ (3), ‘Battleship’ (4), and ‘AircraftCarrier’ (5).

Tile.vb

-Script that defines a Tile, which is used throughout a grid. Each tile knows its placement in a grid both row and column –wise, knows if a ship is on it in addition to what kind of ship it is, and knows if it’s been shot. These can all be returned via public methods.

TileView.vb

-Script that defines a public enum intended for representing what a player can see in a tile. Possible values are ‘Sea’, ‘Miss’, ‘Ship’, ‘Hit’.