Battleships Technical Specification

Group Project

Git Repo: <https://github.com/DrGrimshaw/BattleshipsJavaGame>

**Ship**

State

* Length
* Health
* Starting Position
* Orientation

Behaviours

* Construct with length, starting position and orientation
* Get length
* Get starting co-ordinates
* Get ending co-ordinates
* Get an array of all co-ordinates
* Get orientation
* Check whether ship exists at co-ordinates
* Take a hit
* Check whether the ship is sunk

**Cell State**

* Nothing
* Miss
* Ship Hit
* Ship not hit
* Ship Sunk

**Player Board**

State

* A board has a grid of cell states
* A board has a list of ships
* A board has a width and height

Behaviours

* Create a board with a width and height
* Get width
* Get height
* Place a ship on to board
* Get cell state at co-ordinates
* Check whether co-ordinates have been guessed before
* Take a hit at co-ordinates, returning the new cell state
* Get ship at co-ordinates if any
* Get number of ships remaining
* Get an unmodifiable list of ships
* Check if all ships are sunk

**Enemy Board**

State

* Board has a grid of cell states
* Board has width
* Board has height
* Number of ships remaining

Behaviour

* Construct an enemy board with… width, height, number of ships
* Update state of cell at co-ordinates – if battleship sunk, reduce number of ships remaining by one
* Get number of ships remaining
* Check whether co-ordinates have been guessed before
* Get cell state at co-ordinates, returning “nothing” in place of “ship not hit” and “ship hit” in place of “ship sunk”
* Get Width
* Get Height

**Game Over Message**

- You won

- You lost

- You resigned

- Your enemy resigned

**View**

Behaviours

* View the states of the player and enemy boards to the user
* View welcome message/game setup message to the user
* View game over message
* View who has won
* Show instructions to user
* View the CellState after the result of the last move made on the given coordinates
* View lengths of ships left to place (during setup)
* View the CellState after the result of the last enemy move made on the given coordinates

**Player**

State

* Player board
* Enemy board

Behaviours

* Construct a player with a new board height, new width and a number of ships remaining
  + (if player has resigned throw player quit exception)
* Ask the player to place a ship of given length with a chosen position and orientation
* Choose coordinates to fire on
  + (if player has resigned throw player quit exception)
* Take a hit at coordinates returning new cell state
  + (if player has resigned throw player quit exception)
* Check whether the player has lost
* Check whether co-ordinates have been guessed before by this player
* View this player’s state
* Report the new CellState and coordinates as a result of the enemy’s move, to update the enemy board.
* Get the player’s view.

**TextView** *implements View*

**Human Player** *implements Player, using console I/O*

**Random Player** *implements Player, choosing randomly*

**Clever AI** *implements Player, using a explore and scan strategy*

**Game**

State

* Two players
* Next player
* List of ship lengths

Behaviour

* Construct a game with two players and a list of ship lengths
* Get the next player
* Get the enemy player
* Play
  + Set up the game by welcoming both players, showing instructions, and ask players to place their ships.
  + *While there is no winner:* get the next player, ask the player to make a move, check with other player whether hit/miss/sunk, report result back to player.
  + If enemy has lost, than announce the result of the game to both players.
  + If either player has resigned announce the result of the game
  + If a player quits without resigning then we print an error message.

**Program**

Constructs the players

Constructs the game

Plays the game

**Network Program Protocol**

Place ship on player board