(get and set methods have been excluded. In addition attributes such as buttons and labels have also been excluded.)

class BattleshipGame

- attributes:
  - player1sg - ShipGrid: object for the first player
  - player2sg - ShipGrid: object for the second player
  - current_player - ShipGrid: the player whose turn it is
  - other_player - ShipGrid: the player whose turn it isn't
  - setup_over - Boolean: says if the setup is over
  - message - str: message detailing what happened on the last turn
  - det_message - str: message detailing what happened on the last turn with more detail than message
  - winner - None or ShipGrid: the player who has been declared the winner
- methods:
  - create_message - in: location(tuple of ints), grid_str(str), name(str); out: None; creates message and det_message for the instance using the missile's location, what it hit, and the player's name.
  - check_sink – in: ship(str); out: Boolean; given a ship, checks to see if that ship was sunk in the shipgrid other_player
  - change_player - in: None; out: None; swaps current_player and other_player
  - check_game_over – in: None; out: None or ShipGrid; updates the winner attribute and returns it
  - surrender – in: None; out: None; sets winner to other_player

class ShipGrid

- attributes:
  - grid – List of List of str: a grid that holds the ships for each player and missiles fired at them
  - player_name – str: the name of the player
- methods:
  - place_ship – in: ship(str), location(tuple of ints), orientation(str); out: Boolean; given a ship, location, and orientation, the ship is removed from the ship if already there and att empts to place a new one (based on the constraints in the SRS document). If it fails it

returns False, else True.

Completed_setup – in: None; out: Boolean; checks to see if all ships have been sunk and returns True if they have else False.

class App

attributes:

game – BattleshipGame: main game object

container – Frame: main frame object of the program

methods:

show_frame – in: cont(Class); out: None; given a frame class, it shows that frame.

make_frame – in: cont(Class); out: None; given a frame class, it makes that frame.

remove_make_show – in: cur_frame(Class), new_frame(Class); out: None; given a frame class, cur_frame, it destroys that frame then makes the frame, new_frame, then finally shows it.

class MainMenu

attributes:

None

methods:

None

class PlayerNames

attributes:

None

methods:

None

class ShipSetup

attributes:

methods:

try_place_ship – in: player: ShipGrid, ship: str, location: tuple of ints, orientation: str; out: None; tries to place a ship in a ship grid given the grid, ship, location, and orientation. If it fails, then a pop up is displayed

update_buttons – in: None; out: None; updates the buttons on the screen next_button_select – in: None; out: None; selects the next ship radio button

next_button_state_check – in: None; out: None; enables the next button if the setup has been completed

class Intermediate

attributes:

None

methods:

None

class Turn

attributes:

location – tuple of ints: the location the user has selected to fire

player – ShipGrid: the current active player

enemy – ShipGrid: the opposing player

methods:

pick_next_screen - in: None, out: None; checks if the game is over and calls remove_make_show for GameOver if true or Intermediate if not.

update_missile_grid – in: None; out: None; updates the missile grid to accurately reflect what has happened after the user fires

try_firing – in: None, out: None; tries to fire at the location location, and if it fails shows a pop up

class GameOver

attributes:

None

methods:

new_game – in: None; out: None; begins a new game by creating a new BattleShipGame object and going to the main menu

func create_grid – in: rows(int), columns(int), filler(any); out: list of lists of filler; creates a list of lists of inputted data

func pop_up – in: text(str); out: None; creates a popup that contains the text inputted

func rules – in: None; out: None; creates a popup containing the rules of the game

func main – in: None; out: None; initiates the game