LobbyView:

**Variables:**

* -scene: Scene
  + The scene for the lobby view
* -title: String
  + The title displayed across the top of the screen
* -playerList: List
  + A list of all users that are going to be players once the game starts
* -observerList: List
  + A list of all users that are going to be observers once the game starts
* -computerSpinner: Spinner
  + A spinner that tracks how many computer players there will be once hte game starts

**Methods:**

* +init()
  + Method that creates all visual elements for the users to see
* -startGame()
  + Method that tells the controller to start the game
* -switchPlayer()
  + Method that tells the controller to swap the user from player to observer and vice-versa
* -kickPlayer()
  + Method that is only available to the host, will remove the selected player from the lobby
* -back()
  + Method that indicates to the controller to return you to the starting screen

StartView:

**Variables:**

* -scene: Scene
  + The scene for the starting view
* -title: String
  + The title displayed across the top of the screen

**Methods:**

* +init()
  + Method that creates all visual elements for the users to see
* -joinGame()
  + Method that opens an entry to type the address of the game you want to join. Calls connectToGame
* -connectToGame(host: String)
  + Method that takes in a String which is the address of the game to be connected to. Passes along to controller
* -hostGame()
  + Method that moves you to the lobby screen
* -exit()
  + Method that closes the program

GameView:

**Variables:**

* -scene: Scene
  + The scene for the starting view
* -title: String
  + The title displayed across the top of the screen
* +playerList: List
  + A list of the players still in the game
* +observerList: List
  + A list of the observers watching the game
* +robotList: List
  + A list of all the robots still alive in the game
* +tileArray: Array
  + An array containing every tile on the board
* -moving: boolean
  + A variable that determines if the player moves when they click on a tile
* -attacking: boolean
  + A variable that determines if the player attacks when they click on a tile
* -inspecting: boolean
  + A variable that determines if the player inspects tiles they click on

**Methods:**

* +init(playerList: List, observerList: List)
  + Method that creates all visual elements for the users to see. playerList and observerList are used to populate GameView’s Lists and determine map size.
* +updateView()
  + Method that updates the map after every action
* +move()
  + Method that sets the moving variable
* +inspect()
  + Method that sets the inspecting variable
* +attack()
  + Method that sets the attacking variable
* +selectTile()
  + Method that informs the controller what tile was clicked, and which of the three actions (moving, attack or inspecting), was currently true.
* +endTurn()
  + Method that informs the controller that the “End Turn” button was clicked
* +forfeit()
  + Method that informs the controller that the “Forfeit” button was clicked.
* -exit()
  + Method that informs the controller that the player has closed the game

InspectView

**Variable:**

* -scene: Scene
  + The scene for the starting view
* -title: String
  + The title displayed across the top of the screen

**Methods:**

* +init(robotList: List)
  + Method that creates all visual elements for the users to see. robotList is a list of the robots on the tile being inspected
* -close()
  + Method that closes the InspectView window

PostGameView:

**Variables:**

* -scene: Scene
  + The scene for the starting view
* -title: String
  + The title displayed across the top of the screen

**Methods:**

* +init(winner: String, robotStats: List)
  + Method that create all visual elements for users to see. winner is displayed as a label, while robotStats is used to populate a table.
* -finish()
  + Method that returns you to the starting screen.