Rationale

By Chao Li (chaol1)

For the milestone A, I finished object-oriented analysis and object-oriented design, through which I obtained the domain model, interaction model and object model. In the following paragraph, I will talk about how I designed this project and why I did that.

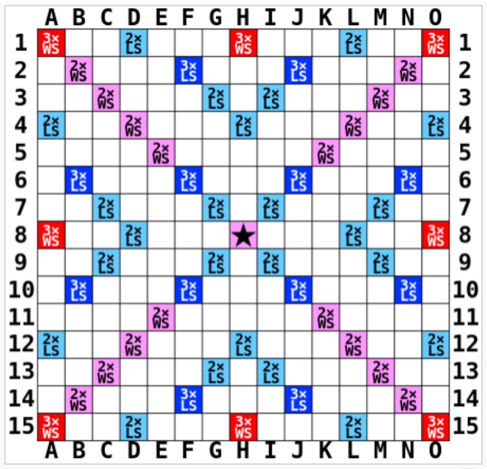
Domain Model

Since I have never heard about this game before, I first referred to the Wikipedia page and videos. Then I tried to abstract the key concepts according to the rules of the game. Game is the framework that we need to hold all other objects and control the overall process of the game. When we add the GUI in the future, the GUI will mainly talk to every object through game. So this is the fundamental object in our project.

Then we need Player object, which is the participant for this game. The player object will provide all variable and methods that a real player need during the game. In addition, since the tiles are stored in player's rack and the rack also some executes some command the player, I made the Rack as an independent object. Every player has a rack, and every rack holds 0 to 7 seven tiles.

What's more, the game also needs a tile bag, which contains 0 to 100 regular tiles. So I abstracted these two as two objects, the tile bag stores the number of tiles and return a tile for the player. And the regular tile has a letter and corresponding points.

Next, I abstracted the board that holds 225 board content like the following figure. This board is like a container so that we can put features, special tiles and regular tiles on it.



The official *Scrabble* board design
Key:
2×LS = Double letter score 3×LS= Triple letter score
2×WS / ★ = Double word score 3×WS = Triple word score

In addition, to check if the player move right, I set a board referee class, which plays a role to record new words, new tiles and new locations and make a djugement to the behavior of the player.

Interaction Model

For the first scenario, I started from the time when player selects the location they want to fill in and tiles they want to use. So the game know the new Location array and new Tiles array. Then our GUI calls a method called isValid() in the game object, this method will pass the new Location array and new Tiles array to the game. Then the game will call isValidLocation() in board to check if the location the player selects is valid rather than out of bound. Then I check if the location is empty, if true, the game will get new words that player formulates by his moving. This method will return all the words in an array. Next, the game will call board-referee to check every word in the array, if all true, then game will check if the player actually has the tiles he claimed, and the player will ask his own rack. If returning true, the player will use the tile and the move is valid.

For the second scenario, the GUI will call the move() method in Game by passing both the new tile array and the new location array. For every new location, the game will call checkBoardContent() in board-referee and get the board content object back. By accessing the board content, everything in that location, for example, feature, tile and special tiles in that space will all be returned. If there is special tile, game will first get who this tile belongs to. And then make the special view by passing the game object to the special view method. After checking the board content, the player will call use tile to place it on the board and remove it from its rack. When this for-loop finishes, the game will call countScore() in board-referee to calculate the total score the player get in this move. And the player will add that score to himself. After that, player will ask himself how many tiles he has, and call drawTiles() method in Rack to refill his rack. When the rack is refilled to 7 tiles, the game will execute the turn() method to change player.

Object Model

By considering the two scenarios above and informal questions after, I added methods needed for every object in the Domain Model. Finally, the object model is finished. Maybe they cannot completely solve all the problems in this project, but basically it helps greatly. I will continue to add necessary methods when I write code in the future.