

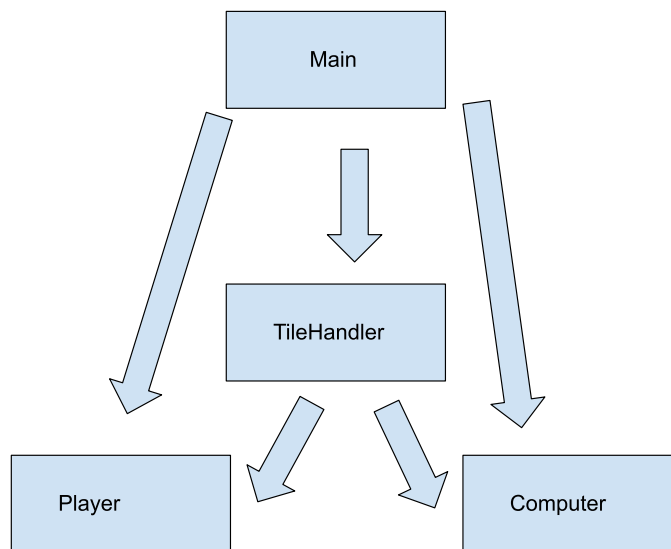
# GUI DESIGN

**Main:** This class contains the game loop and collects all user input. Runs the main parts of the game. Also handles most of the GUI.

**TileHandler:** This class handles everything done with the tiles. It plays tiles, handles the game board, and handles the boneyard.

**Player:** This class handles the player's tiles. It adds and removes tiles, as well as rotates them.

**Computer:** This class handles the computer's tiles. It adds and removes tiles, as well as rotates them. It also plays the computer's round.



# DESCRIPTION OF OBJECTS IN Main.java

main(): This is the main function that runs the entire program. It calls start().

start(): This function sets up all parts of the GUI, and calls all of the other classes that run the game.

# DESCRIPTION OF OBJECTS IN TileHandler.java

## Variables:

int[][] allTiles: all possible game tiles

Int[][] boneyard: all tiles in the boneyard

Button[] boneyardVis: the buttons corresponding to the tiles in the boneyard

Button[][] boardTilesVis: the buttons corresponding to the tiles on the game board

Int boneLength: number of tiles in the boneyard

Int[][][]boardTiles: all tiles on the game board

Int topBoardLen: length of the top row of the game board

Int botBoardLen: length of the bottom row of the game board

Int lastTop: index of the last tile on the top row of the game board

Int lastBot: index of the last tile on the bottom row of the game board

## Functions:

TileHandler(Player player, Computer computer): Constructor deals tiles to the players and initializes all variables.

Param player is the player object

Param computer is the computer object

getRandTile(): gets a random tile from the boneyard. Returns the tile.

getNumTiles(): gets the number of tiles in the boneyard. Returns this number.

addBoardTile (int[] tile, int loc): Adds a tile to the board.

Param tile: The tile to be added.

Param loc: the location of the tile to be added. Loc 1 is top left, loc 2 is bottom left, loc 3 is top right, loc 4 is bottom right.

getBoard(): creates the visuals for the board. Return the string that contains the board.

isPlayable(int[] tile, int loc): determines if a tile is playable. Returns the location where is playable.

Param tile is the tile to be checked

Param loc is the location where the tile will be checked. Loc 1 is left and loc 2 is right.

getBoardNum(): Returns the number of tiles on the board

gameOver (Player player, Computer computer, int lastPlayed): handles the end of the game. Determines the winner.

Param player is the player object.

Param computer is the computer object

Param lastPlayed is the last player to place a tile

getBoneyard(int pos): Returns a tile from the boneyard at a given position.

Param pos is the position of the tile in the boneyard

getBoneButt(int pos): Returns the button of the tile at the given position

Param pos is the position of the tile in the boneyard

drawBoneyard(): draws a tile from the boneyard. Returns the location of the tile in the boneyard.

# DESCRIPTION OF OBJECTS IN Player.java

## Variables:

int[][] playerTiles: tiles in the player's tray

Button[] playerTilesVis: The Buttons that correspond to each tile in the player's hand

int numTiles: number of tiles in the player's tray

## Functions:

giveTile(int[] tile): gives the player a tile.

Param tile is the tile to be given

removeTile(int pos): removes a tile at a given position

Param pos is the location of the tile to be removed

getTilesInt(): returns the player's tiles in int[][] format

getNumTiles(): returns the number of tiles in the player's hand

rotateDomino(int pos): rotates a tile at a given position

Param pos is the location of the tile to be rotated

getTileInt(int pos): returns a tile at a given position

Param pos is the location of the tile to be given

getTileButt(int pos): returns the button of a tile at a given position

Param pos is the location of the tile to be given

# DESCRIPTION OF OBJECTS IN Computer.java

## Variables:

`int[][] compTiles`: tiles in the computer's tray

`Button[] compTilesVis`: The Buttons that correspond to each tile in the computer's hand

`int numTiles`: number of tiles in the computer's tray

## Functions:

`giveTile(int[] tile)`: gives the computer a tile.

Param tile is the tile to be given

`removeTile(int pos)`: removes a tile at a given position

Param pos is the location of the tile to be removed

`getTilesInt()`: returns the computer's tiles in `int[][]` format

`getNumTiles()`: returns the number of tiles in the computer's hand

`rotateDomino(int pos)`: rotates a tile at a given position

Param pos is the location of the tile to be rotated

`getTileInt(int pos)`: returns a tile at a given position

Param pos is the location of the tile to be given

`getTileButt(int pos)`: returns the button of a tile at a given position

Param pos is the location of the tile to be given

`canPlay(TileHandler tileHandler)`: plays one of the computer's tiles if possible. Returns false if the computer must draw.

Param tileHandler is the TileHandler object.