**Class Doc**

**Game:**

Properties:

* Rows: int- Gets the amount of rows the board currently has. This will be an Integer.
* CurrentRow: int- Gets the row the HumanPlayer is currently taking pieces from. This will be an Integer.
* Players: Player[]- An array that holds two instances of type Player.
* CurrentPlayerGoingIndex: int- This is an Integer that will indicate whose turn it is using 0 and 1. Will be mostly used to help get the index from the Players array. And to determine who has won.
* Board: Board- an instance of type Board

Methods:

* ComputerMove():void -This method will be called everytime the computer wishes to move.

**Board:**

Properties:

* TotalPieces: int - Gets the total pieces left on the board. This will be an Integer.
* BoardState: int[][] - A 2D array of type int that represents the board. 1s will be pieces and 0s will be blank spaces.

Methods:

* TakeAwayPiece(int Row, int Col):bool - This method takes in two parameters, the first one being the row and the second one being the column. This method will take this info, go to the specified index and change the 1 to a 0.

**Player:**

Properties:

* Name: string - gets the name of the player. This will be a string
* IsHuman: bool - get a bool that says if the player is human or not.

**MainWindow:**

Properties:

* Game: Game: This is an instance of type Game.

Methods:

* MakeGameBoard(): void - This method makes generates rectangles that based on the length of both dimensions in the game.board[][]. This will also give the rectangles a method for clicking the left mouse button.
* RulesButton\_Click(object sender, RoutedEventArgs e): void - Changes from the Nim Game Rules Screen to the Start Menu.
* PVCButton\_Click(object sender, RoutedEventArgs e): void - This method initializes the players that are appropriate to the Player vs CPU game mode and puts them into the Players array in the current Game object.
* PVPButton\_Click(object sender, RoutedEventArgs e): void - This method makes the players that are appropriate to the Player vs Player game mode and puts them into the Players array in the current game object.
* EasyButton\_Click(object sender, RoutedEventArgs e):void - will set the appropriate board size for the Board array inside of the current Game object.
* MediumButton\_Click(object sender, RoutedEventArgs e): void - will set the appropriate board size for the Board array inside of the current Game object.
* HardButton\_Click(object sender, RoutedEventArgs e): void - will set the appropriate board size for the Board array inside of the current Game object.
* HumanPlayerMove(object sender, MouseButtonEventArgs e): void - This method takes the sender object and will use to see if it is at the same row that the Game.CurrentRow lists. If so it will proceed with the move. If not, then it will do nothing until the player makes a valid move.