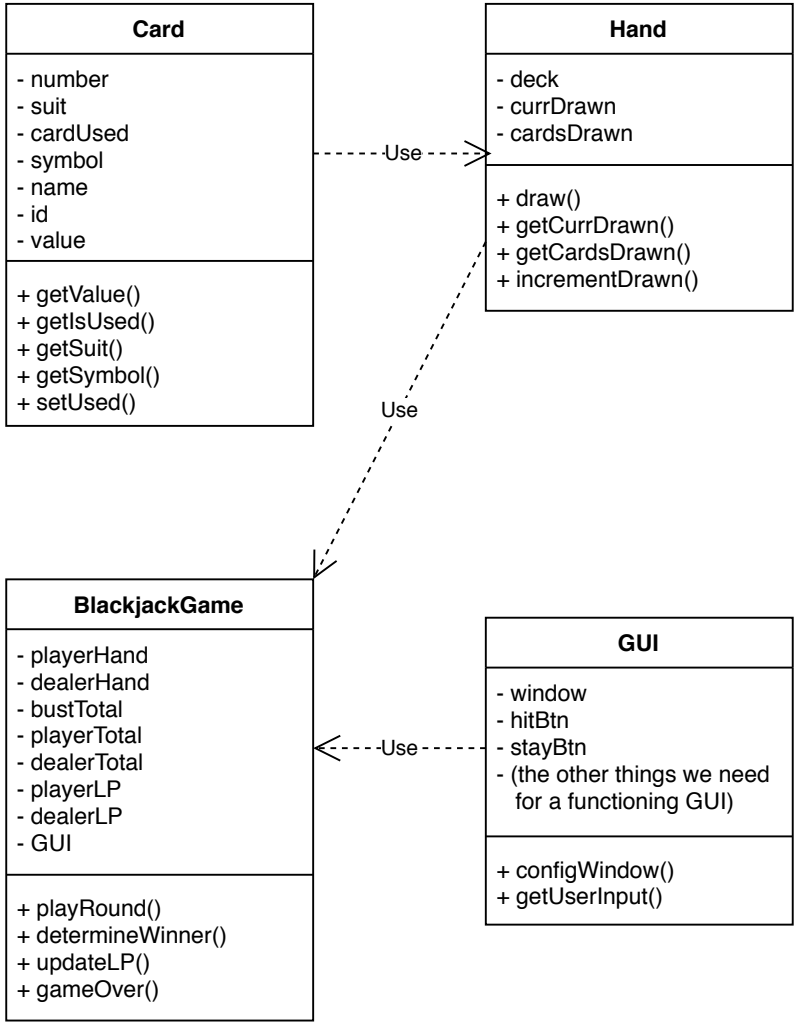

Note: Our team's current plan is to get the inner workings of our code (ie the non-GUI elements) working before we detail how we're going to get our GUI elements working. The current GUI class is a placeholder but we are hoping to implement the MVC design pattern if possible

Card

- Represents one card in a game of blackjack

Card Methods

- getValue(): returns point value of card
- getIsUsed(): returns true/false value of if the card was drawn
- getSuit(): returns suit of card (for printing in GUI purposes)
- getSymbol(): returns condensed symbol (also for printing in GUI purposes)
- setUsed(): change the drawn true/false value

Hand

- Represents one entity's (player or dealer) hand in the game of blackjack

Hand Methods

- draw(): draw a random card from the 52 card deck
- getCurrDrawn(): returns the currently drawn card stored in currDrawn
- getCardsDrawn(): returns the amount of cards drawn for the hand in question
- incrementDrawn(): adds one to the cardsDrawn amount.

BlackjackGame

- Runs the game of blackjack until either the player or dealer life points reach 0

BlackjackGame Methods

- playRound(): plays one round of blackjack (the drawing of the cards) and calls determine winner
- determineWinner(): determines winner based on bustValue, playerTotal, and dealerTotal
- updateLP(): updates player or dealer life points based on who wins the round
- gameOver(): detects when player or dealer life points reaches zero and ends the game

GUI

- (this will probably be split into more than one method) the base of the graphical-user-interface

GUI Methods (these are placeholders for now)

- configWindow(): generates the configuration window to start the game
- getUserInput(): gets whether user has selected hit or stay.