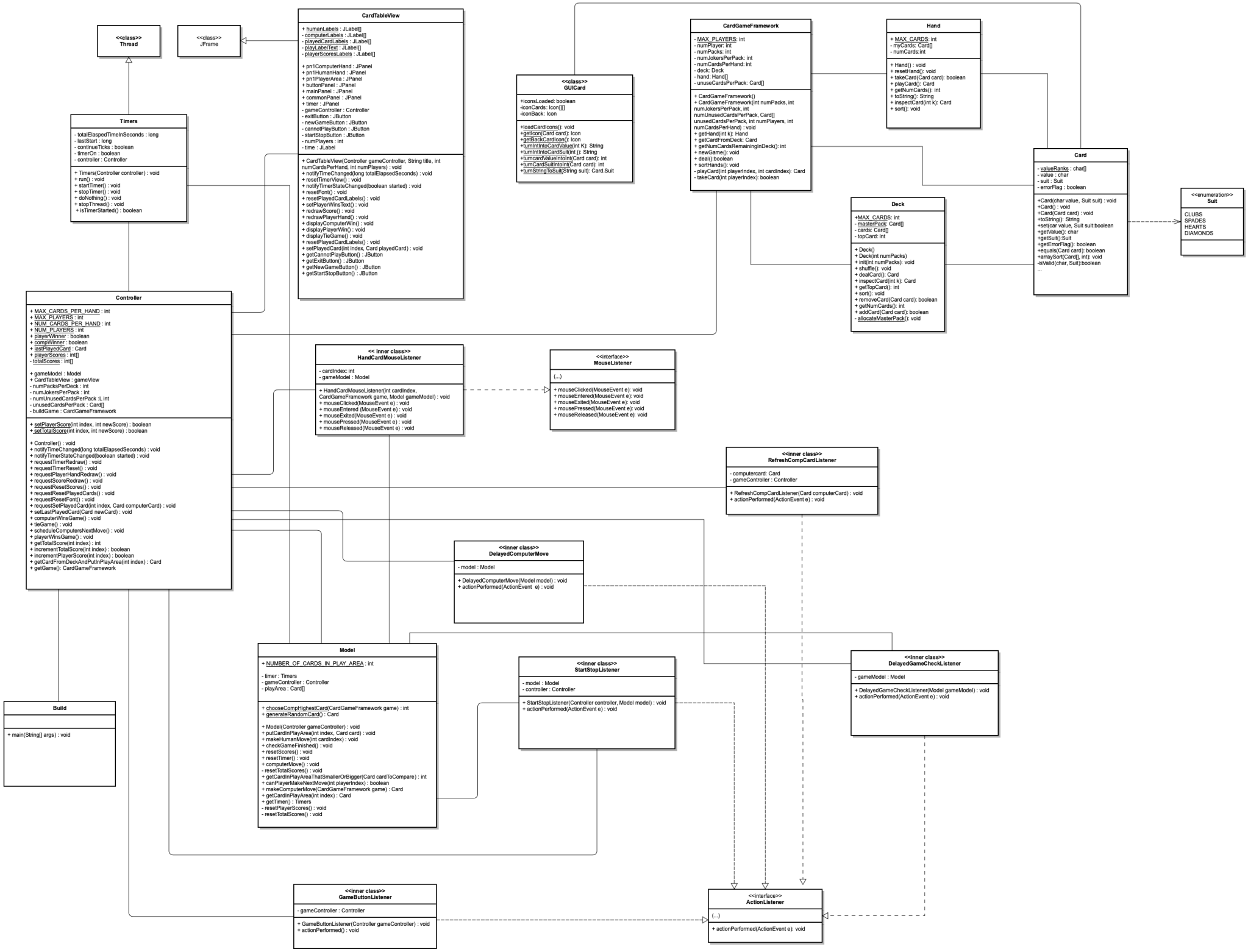
<<class>>
Thread
<<class>>
JFrame
CardTableView
+ humanLabels : JLabel[]
- computerLabels : JLabel[]
- playedCardLabels : JLabel[]
- playLabelText : JLabel[]
- playerScoresLabels : JLabel[]
+ pn1ComputerHand : JPanel
+ pn1HumanHand : JPanel
+ pn1PlayerArea : JPanel
+ buttonPanel : JPanel
+ mainPanel : JPanel
+ commonPanel : JPanel
+ timer : JPanel
- gameController : Controller
- exitButton : JButton
- newGameButton : JButton
- cannotPlayButton : JButton
- startStopButton : JButton
- numPlayers : int
- time : JLabel
+ CardTableView(Controller gameController, String title, int numCardsPerHand, int numPlayers) : void
+ notifyTimeChanged(long totalElapsedSeconds) : void
+ resetTimerView() : void
+ notifyTimerStateChanged(boolean started) : void
+ resetFont() : void
+ resetPlayedCardLabels() : void
+ setPlayerWinsText() : void
+ redrawScore() : void
+ redrawPlayerHand() : void
+ displayComputerWin() : void
+ displayPlayerWin() : void
+ displayTieGame() : void
+ resetPlayedCardLabels() : void
+ setPlayedCard(int index, Card playedCard) : void
+ getCannotPlayButton() : JButton
+ getExitButton() : JButton
+ getNewGameButton() : JButton
+ getStartStopButton() : JButton
<<class>>
GUICard
+iconsLoaded: boolean
+iconCards: Icon[][]
+iconBack: Icon
+loadCardIcons(): void
+getIcon(Card card): Icon
+getBackCardIcon(): Icon
+turnIntIntoCardValue(int K): String
+turnIntIntoCardSuit(int j): String
+turncardValueIntoInt(Card card): int
+turnCardSuitIntoInt(Card card): int
+turnStringToSuit(String suit): Card.Suit
CardGameFramework
- MAX_PLAYERS: int
- numPlayer: int
- numPacks: int
- numJokersPerPack: int
- numCardsPerHand: int
- deck: Deck
- hand: Hand[]
- unuseCardsPerPack: Card[]
+ CardGameFramework()
+ CardGameFramework(int numPacks, int numJokersPerPack, int numUnusedCardsPerPack, Card[] unusedCardsPerPack, int numPlayers, int numCardsPerHand) : void
+ getHand(int k): Hand
+ getCardFromDeck: Card
+ getNumCardsRemainingInDeck(): int
+ newGame(): void
+ deal():boolean
+ sortHands(): void
- playCard(int playerIndex, int cardIndex): Card
- takeCard(int playerIndex): boolean
Hand
+ MAX_CARDS: int
- myCards: Card[]
- numCards:int
+ Hand() : void
+ resetHand(): void
+ takeCard(Card card): boolean
+ playCard(): Card
+ getNumCards(): int
+ toString(): String
+ inspectCard(int k): Card
+ sort(): void
Timers
- totalElaspedTimeInSeconds : long
- lastStart : long
- continueTicks : boolean
- timerOn : boolean
- controller : Controller
+ Timers(Controller controller) : void
+ run() : void
+ startTimer() : void
+ stopTimer() : void
+ doNothing() : void
+ stopThread() : void
+ isTimerStarted() : boolean
Card
- valueRanks : char[]
- value : char
- suit : Suit
- errorFlag : boolean
+Card(char value, Suit suit) : void
+Card() : void
+Card(Card card) : void
+toString(): String
+set(car value, Suit suit:boolean
+getValue(): char
+getSuit():Suit
+getErrorFlag(): boolean
+equals(Card card): boolean
+arraySort(Card[], int): void
-isValid(char, Suit):boolean
...
<<enumeration>>
Suit
CLUBS
SPADES
HEARTS
DIAMONDS
Deck
+MAX_CARDS: int
- masterPack: Card[]
- cards: Card[]
- topCard: int
+ Deck()
+ Deck(int numPacks)
+ init(int numPacks): void
+ shuffle(): void
+ dealCard(): Card
+ inspectCard(int k): Card
+ getTopCard(): int
+ sort(): void
+ removeCard(Card card): boolean
+ getNumCards(): int
+ addCard(Card card): boolean
- allocateMasterPack(): void
Controller
+ MAX_CARDS_PER_HAND : int
+ MAX_PLAYERS : int
+ NUM_CARDS_PER_HAND : int
+ NUM_PLAYERS : int
+ playerWinner : boolean
+ compWinner : boolean
+ lastPlayedCard : Card
+ playerScores : int[]
- totalScores : int[]
+ gameModel : Model
+ CardTableView : gameView
- numPacksPerDeck : int
- numJokersPerPack : int
- numUnusedCardsPerPack :L int
- unusedCardsPerPack : Card[]
- buildGame : CardGameFramework
+ setPlayerScore(int index, int newScore) : boolean
+ setTotalScore(int index, int newScore) : boolean
+ Controller() : void
+ notifyTimeChanged(long totalElapsedSeconds) : void
+ notifyTimerStateChanged(boolean started) : void
+ requestTimerRedraw() : void
+ requestTimerReset() : void
+ requestPlayerHandRedraw() : void
+ requestScoreRedraw() : void
+ requestResetScores() : void
+ requestResetPlayedCards() : void
+ requestResetFont() : void
+ requestSetPlayedCard(int index, Card computerCard) : void
+ setLastPlayedCard(Card newCard) : void
+ computerWinsGame() : void
+ tieGame() : void
+ scheduleComputersNextMove() : void
+ playerWinsGame() : void
+ getTotalScore(int index) : int
+ incrementTotalScore(int index) : boolean
+ incrementPlayerScore(int index) : boolean
+ getCardFromDeckAndPutinPlayArea(int index) : Card
+ getGame(): CardGameFramework
<< inner class>>
HandCardMouseListener
- cardIndex: int
- gameModel : Model
+ HandCardMouseListener(int cardIndex, CardGameFramework game, Model gameModel) : void
+ mouseClicked(MouseEvent e) : void
+ mouseEntered (MouseEvent e) : void
+ mouseExited(MouseEvent e) : void
+ mousePressed(MouseEvent e) : void
+ mouseReleased(MouseEvent e) : void
<<interface>>
MouseListener
(...)
+ mouseClicked(MouseEvent e): void
+ mouseEntered(MouseEvent e): void
+ mouseExited(MouseEvent e): void
+ mousePressed(MouseEvent e): void
+ mouseReleased(MouseEvent e): void
<<inner class>>
RefreshCompCardListener
- computercard: Card
- gameController : Controller
+ RefreshCompCardListener(Card computerCard) : void
+ actionPerformed(ActionEvent e) : void
<<inner class>>
DelayedComputerMove
- model : Model
+ DelayedComputerMove(Model model) : void
+ actionPerformed(ActionEvent e) : void
Model
+ NUMBER_OF_CARDS_IN_PLAY_AREA : int
- timer : Timers
- gameController : Controller
- playArea : Card[]
+ chooseCompHighestCard(CardGameFramework game) : int
+ generateRandomCard() : Card
+ Model(Controller gameController) : void
+ putCardInPlayArea(int index, Card card) : void
+ makeHumanMove(int cardIndex) : void
+ checkGameFinished() : void
+ resetScores() : void
+ resetTimer() : void
+ computerMove() : void
- resetTotalScores() : void
+ getCardInPlayAreaThatSmallerOrBigger(Card cardToCompare) : int
+ canPlayerMakeNextMove(int playerIndex) : boolean
+ makeComputerMove(CardGameFramework game) : Card
+ getCardInPlayArea(int index) : Card
+ getTimer() : Timers
- resetPlayerScores() : void
- resetTotalScores() : void
<<inner class>>
StartStopListener
- model : Model
- controller : Controller
+ StartStopListener(Controller controller, Model model) : void
+ actionPerformed(ActionEvent e) : void
<<inner class>>
DelayedGameCheckListener
- gameModel : Model
+ DelayedGameCheckListener(Model gameModel) : void
+ actionPerformed(ActionEvent e) : void
Build
+ main(String[] args) : void
<<inner class>>
GameButtonListener
- gameController : Controller
+ GameButtonListener(Controller gameController) : void
+ actionPerformed() : void
<<interface>>
ActionListener
(...)
+ actionPerformed(ActionEvent e): void