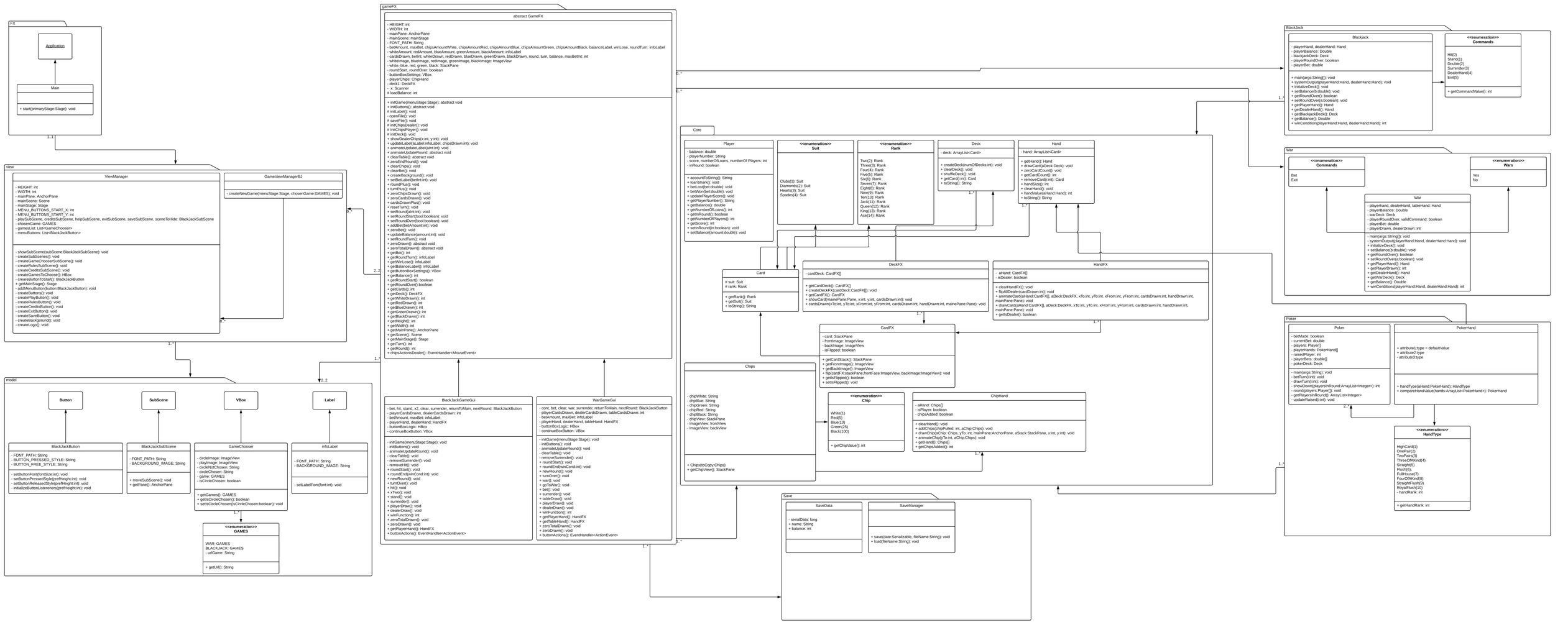
FX
Application
Main
+ start(primaryStage:Stage): void
1..1
view
ViewManager
- HEIGHT: int
- WIDTH: int
- mainPane: AnchorPane
- mainScene: Scene
- mainStage: Stage
- MENU_BUTTONS_START_X: int
- MENU_BUTTONS_START_Y: int
- playSubScene, creditsSubScene, helpSubScene, exitSubScene, saveSubScene, sceneToHide: BlackJackSubScene
- chosenGame: GAMES
- gamesList: List<GameChooser>
- menuButtons: List<BlackJackButton>
- showSubScene(subScene:BlackJackSubScene): void
- createSubScenes(): void
- createGameChooserSubScene(): void
- createRulesSubScene(): void
- createCreditsSubScene(): void
- createGamesToChoose(): HBox
- creareButtonToStart(): BlackJackButton
+ getMainStage(): Stage
- addMenuButton(button:BlackJackButton): void
- createButtons(): void
- createPlayButton(): void
- createRulesButton(): void
- createCreditsButton(): void
- createExitButton(): void
- createSaveButton(): void
- createBackground(): void
- createLogo(): void
GameViewManagerBJ
- createNewGame(menuStage:Stage, chosenGame:GAMES): void
0..*
2..2
0..*
1..*
2..2
model
Button
SubScene
VBox
Label
BlackJackButton
- FONT_PATH: String
- BUTTON_PRESSED_STYLE: String
- BUTTON_FREE_STYLE: String
- setButtonFont(fontSize:int): void
- setButtonPressedStyle(prefHeight:int): void
- setButtonReleasedStyle(prefHeight:int): void
- initializeButtonListeners(prefHeight:int): void
BlackJackSubScene
- FONT_PATH: String
- BACKGROUND_IMAGE: String
+ moveSubScene(): void
+ getPane(): AnchorPane
GameChooser
- circleImage: ImageView
- playImage: ImageView
- circleNotChosen: String
- circleChosen: String
- game: GAMES
- isCircleChosen: boolean
+ getGames(): GAMES
+ getIsCircleChosen(): boolean
+ setIsCircleChosen(isCircleChosen:boolean): void
1..1
infoLabel
- FONT_PATH: String
- BACKGROUND_IMAGE: String
- setLabelFont(fontInt): void
<<enumeration>>
GAMES
WAR: GAMES
BLACKJACK: GAMES
- urlGame: String
+ getUrl(): String
gameFX
abstract GameFX
- HEIGHT: int
- WIDTH: int
- mainPane: AnchorPane
- mainScene: mainStage
- FONT_PATH: String
- betAmount, maxBet, chipsAmountWhite, chipsAmountRed, chipsAmountBlue, chipsAmountGreen, chipsAmountBlack, balanceLabel, winLose, roundTurn: infoLabel
- whiteAmount, redAmount, blueAmount, greenAmount, blackAmount: infoLabel
- cardsDrawn, betInt, whiteDrawn, redDrawn, blueDrawn, greenDrawn, blackDrawn, round, turn, balance, maxBetInt: int
- whiteImage, blueImage, redimage, greenImage, blackImage: ImageView
- white, blue, red, green, black: StackPane
- roundStart, roundOver: boolean
- buttonBoxSettings: VBox
- playerChips: ChipHand
- deck1: DeckFX
- x: Scanner
loadBalance: int
+ initGame(menuStage:Stage): abstract void
+ initButtons(): abstract void
# initLabel(): void
- openFile(): void
saveFile(): void
initChipsDealer(): void
initChipsPlayer(): void
initDeck(): void
+ showDealerChips(x:int, y:int): void
+ updateLabel(aLabel:infoLabel, chipsDrawn:int): void
+ animateUpdateLabel(aInt:int): void
+ animateUpdateRound: abstract void
+ clearTable(): abstract void
+ zeroEndRound(): void
+ clearChips(): void
+ clearBet(): void
+ createBackground(): void
+ setBetLabel(betInt:int): void
+ roundPlus(): void
+ turnPlus(): void
+ zeroChipsDrawn(): void
+ zeroCardsDrawn(): void
+ cardsDrawnPlus(): void
+ resetTurn(): void
+ setRound(aInt:int): void
+ setRoundStart(bool:boolean): void
+ setRoundOver(bool:boolean): void
+ addBet(betAmount:int): void
+ zeroBet(): void
+ updateBalance(amount:int): void
+ setRoundTurn(): void
+ zeroDrawn(): abstract void
+ zeroTotalDrawn(): abstract void
+ getBet(): int
+ getRoundTurn(): infoLabel
+ getWinLose(): infoLabel
+ getBalanceLabel(): infoLabel
+ getButtonBoxSettings(): VBox
+ getBalance(): int
+ getRoundStart(): boolean
+ getRoundOver(): boolean
+ getCards(): int
+ getDeck(): DeckFX
+ getWhiteDrawn(): int
+ getRedDrawn(): int
+ getBlueDrawn(): int
+ getGreenDrawn(): int
+ getBlackDrawn(): int
+ getHeight(): int
+ getWidth(): int
+ getMainPane(): AnchorPane
+ getScene(): Scene
+ getMainStage(): Stage
+ getTurn(): int
+ getRound(): int
+ chipsActionsDealer(): EventHandler<MouseEvent>
1..*
BlackJackGameGui
- bet, hit, stand, x2, clear, surrender, returnToMain, nextRound: BlackJackButton
- playerCardsDrawn, dealerCardsDrawn: int
- betAmount, maxBet: infoLabel
- playerHand, dealerHand, tableHand: HandFX
- buttonBoxLogic: HBox
- continueBoxButton: VBox
- initGame(menuStage:Stage): void
- initButtons(): void
- animateUpdateRound(): void
- clearTable(): void
- removeSurrender(): void
- removeHit(): void
+ roundStart(): void
+ roundEndWinCond int): void
+ newRound(): void
+ turnOver(): void
+ hit(): void
+ xTwo(): void
+ stand(): void
+ surrender(): void
+ playerDraw(): void
+ dealerDraw(): void
+ winFunction(): int
+ zeroTotalDrawn(): void
+ zeroDrawn(): void
+ getPlayerHand(): HandFX
+ buttonActions(): EventHandler<ActionEvent>
WarGameGui
- cont, bet, clear, war, surrender, returnToMain, nextRound: BlackJackButton
- playerCardsDrawn, dealerCardsDrawn, tableCardsDrawn: int
- betAmount, maxBet: infoLabel
- playerHand, dealerHand, tableHand: HandFX
- buttonBoxLogic: HBox
- continueBoxButton: VBox
- initGame(menuStage:Stage): void
- initButtons(): void
- animateUpdateRound(): void
- clearTable(): void
- removeSurrender(): void
+ roundStart(): void
+ roundEndWinCond int): void
+ newRound(): void
+ turnOver(): void
+ war(): void
+ goToWar(): void
+ bet(): void
+ surrender(): void
+ tableDraw(): void
+ playerDraw(): void
+ dealerDraw(): void
+ winFunction(): int
+ getPlayerHand(): HandFX
+ getTableHand(): HandFX
+ zeroTotalDrawn(): void
+ zeroDrawn(): void
+ buttonActions(): EventHandler<ActionEvent>
0..*
0..*
1..*
0..*
Core
Player
- balance: double
- playerNumber: String
- score, numberOfLoans, numberOf Players: int
- inRound: boolean
+ accountToString(): String
+ loanShark(): void
+ betLost(bet:double): void
+ betWon(bet:double): void
+ updatePlayerScore(): void
+ getPlayerNumber(): String
+ getBalance(): double
+ getNumberOfLoans(): int
+ getInRound(): boolean
+ getNumberOfPlayers(): int
+ getScore(): int
+ setInRound(in:boolean): void
+ setBalance(amount:double): void
<<enumeration>>
Suit
Clubs(1): Suit
Diamonds(2): Suit
Hearts(3): Suit
Spades(4): Suit
<<enumeration>>
Rank
Two(2): Rank
Three(3): Rank
Four(4): Rank
Five(5): Rank
Six(6): Rank
Seven(7): Rank
Eight(8): Rank
Nine(9): Rank
Ten(10): Rank
Jack(11): Rank
Queen(12): Rank
King(13): Rank
Ace(14): Rank
Deck
- deck: ArrayList<Card>
+ createDeck(numOfDecks:int): void
+ clearDeck(): void
+ shuffleDeck(): void
+ getCard(i:int): Card
+ toString(): String
1..*
Hand
- hand: ArrayList<Card>
+ getHand(): Hand
+ drawCard(aDeck:Deck): void
+ zeroCardCount(): void
+ getCardCount(): int
+ removeCard(i:int): Card
+ handSize(): int
+ clearHand(): void
+ handValue(aHand:Hand): int
+ toString(): String
1..*
Card
suit: Suit
rank: Rank
+ getRank(): Rank
+ getSuit(): Suit
+ toString(): String
DeckFX
- cardDeck: CardFX[]
+ getCardDeck(): CardFX[]
+ createDeckFX(cardDeck:CardFX[]): void
+ getCardFX(): CardFX
+ showCard(mainePane:Pane, x:int, y:int, cardsDrawn:int): void
+ cardsDrawn(xTo:int, yTo:int, xFrom:int, yFrom:int, cardsDrawn:int, handDrawn:int, mainePane:Pane): void
1..*
HandFX
- aHand: CardFX[]
- isDealer: boolean
+ clearHandFX(): void
+ flipAllDealer(cardDrawn:int): void
+ animateCard(aHand:CardFX[], aDeck:DeckFX, xTo:int, yTo:int. xFrom:int, yFrom:int, cardsDrawn:int, handDrawn:int, mainePane:Pane): void
+ drawCard(aHand:CardFX[], aDeck:DeckFX, xTo:int, yTo:int. xFrom:int, yFrom:int, cardsDrawn:int, handDrawn:int, mainePane:Pane): void
+ getisDealer(): boolean
1..*
CardFX
- card: StackPane
- frontImage: ImageView
- backImage: ImageView
- isFlipped: boolean
+ getCardStack(): StackPane
+ getFrontImage(): ImageView
+ getBackImage(): ImageView
+ flip(cardFX:stackPane,frontFace:ImageView, backImage:ImageView): void
+ getisFlipped(): boolean
+ setisFlipped(): void
Chips
- chipWhite: String
- chipBlue: String
- chipGreen: String
- chipRed: String
- chipBlack: String
- chipView: StackPane
- chipImage: frontView
- ImageView: backView
+ Chips(noCopy:Chips):
+ getChipView(): StackPane
<<enumeration>>
Chip
White(1)
Red(5)
Blue(10)
Green(25)
Black(100)
+ getChipValue(): int
ChipHand
- aHand: Chips[]
- isPlayer: boolean
- chipsAdded: boolean
+ clearHand(): void
+ addChips(chipPulled: int, aChip:Chips): void
+ drawChips(aChip: Chips, yTo: int, mainPane:AnchorPane, aStack:StackPane, x:int, y:int): void
+ animateChip(yTo:int, aChip:Chips): void
+ getHand(): Chips[]
+ getChipsAdded(): int
1..*
Save
SaveData
- serialData: long
+ name: String
+ balance: int
SaveManager
+ save(date:Serializable, fileName:String): void
+ load(fileName:String): void
BlackJack
Blackjack
- playerHand, dealerHand: Hand
- playerBalance: Double
- blackjackDeck: Deck
- playerRoundOver: boolean
- playerBet: double
+ main(args:String[]): void
+ systemOutput(playerHand:Hand, dealerHand:Hand): void
+ initializeDeck(): void
+ setBalance(b:double): void
+ getRoundOver(): boolean
+ setRoundOver(a:boolean): void
+ getPlayerHand(): Hand
+ getDealerHand(): Hand
+ getBlackjackDeck(): Deck
+ getBalance(): Double
+ winCondition(playerHand:Hand, dealerHand:Hand): int
1..*
<<enumeration>>
Commands
Hit(0)
Stand(1)
Double(2)
Surrender(3)
DealerHand(4)
Exit(5)
+ getCommandValue(): int
War
<<enumeration>>
Commands
Bet
Exit
<<enumeration>>
Wars
Yes
No
War
- playerhand, dealerHand, tableHand: Hand
- playerBalance: Double
- warDeck: Deck
- playerRoundOver, validCommand: boolean
- playerBet: double
- playerDrawn, dealerDrawn: int
- main(args:String[]): void
- systemOutput(playerHand:Hand, dealerHand:Hand): void
+ initializeDeck(): void
+ setBalance(b:double): void
+ getRoundOver(): boolean
+ setRoundOver(a:boolean): void
+ getPlayerHand(): Hand
+ getPlayerDrawn(): int
+ getDealerHand(): Hand
+ getWarDeck(): Deck
+ getBalance(): Double
+ winConditions(playerHand:Hand, dealerHand:Hand): int
Poker
Poker
- betMade: boolean
- currentBet: double
- players: Player[]
- playerHands: PokerHand[]
- raisedPlayer: int
- playerBets: double[]
- pokerDeck: Deck
- main(args:String): void
- betTurn():void
- drawTurn():void
- showDown(playersInRound:ArrayList<Integer>): int
- roundplayers:Player[]): void
- getPlayersInRound(): ArrayList<Integer>
- updateRaised(int): void
2..*
PokerHand
+ attribute1:type = defaultValue
+ attribute2:type
- attribute3:type
+ handType(aHand:PokerHand): HandType
+ compareHandValue(hands:ArrayList<PokerHand>): PokerHand
1..*
<<enumeration>>
HandType
HighCard(1)
OnePair(2)
TwoPairs(3)
ThreeOfAKind(4)
Straight(5)
Flush(6)
FullHouse(7)
FourOfAKind(8)
StraightFlush(9)
RoyalFlush(10)
- handRank: int
+ getHandRank: int