2016

GVSU Blackjack Game

CIS350 TERM PROJECT – RELEASE ONE

NATHAN LINDENBAUM – EMERSON VEENSTRA

Project Description:

GVSU Blackjack is a graphics based blackjack game. In the game the player will play the card game blackjack against a dealer. The dealer will be played automatically by the program. The player will be able to bet credits one an infinite number of hands.

Feature List for Release 1:

Game:

1. Deal
2. Hit
3. Stand
4. Double Down
5. Bet
6. Shuffle

GUI:

1. Display Cards
2. Update Playing Cards
3. Display Hint Card
4. Turn Buttons on and off

Instructions for Playing:

Once you run the program an Option box will appear. It will ask the player how much they want to bet on their first hand. The player is initialized with 50 credits.

1. Enter the amount of credits you'd like to bet.
2. GUI will appear with option to change bet or deal cards. Select deal cards.
3. Play blackjack until you either run out of money or want to stop.
4. To display a hint card:
    a. Select "File"
    b. Then click "Hint Card"

Use Case 1:

Name	Starting a game
ID	UC1
Brief Description	The steps to start a new game
Actors (primary and supporting/secondary)	Dealer Player
Triggers	Running the application
Preconditions	None
Primary Flow	1. Player enters the amount to bet 2. Player clicks OK 3. Player clicks Deal 4. Dealer deals initial 4 cards
Alternate Flows	1. Player quits game before hand is dealt a. Game ends 2. Player clicks on bet to change the amount of money. . Return to step 1 of primary flow and start over
Minimal Guarantees	None
Success Guarantees	The initial 4 cards are dealt

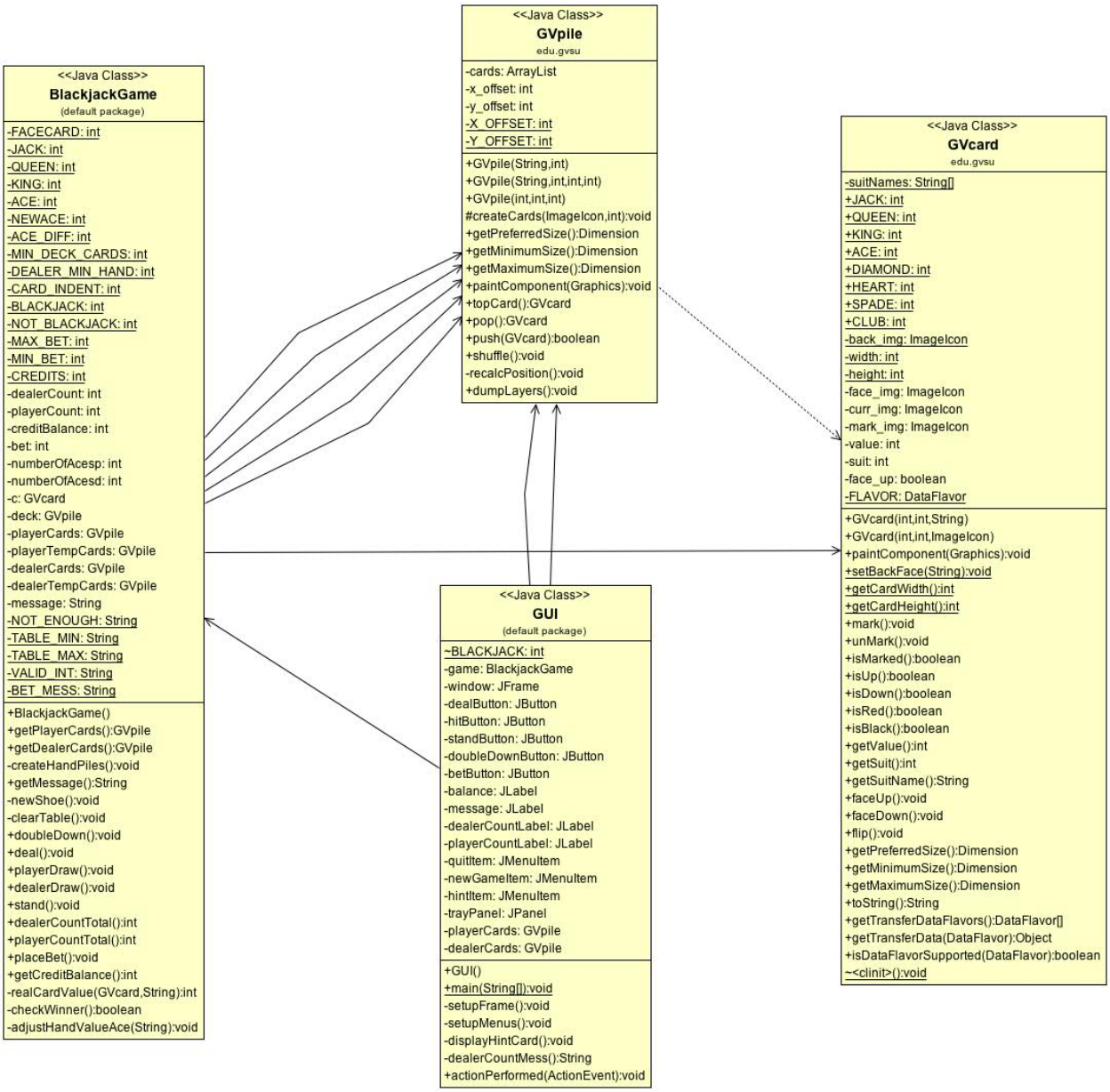
Use Case 2:

Name	Player draws card
ID	UC2
Brief Description	Dealing another card to the player
Actors (primary and supporting/secondary)	Player, Dealer
Triggers	Clicking the Draw button
Preconditions	The player must be dealt the initial two cards already
Primary Flow	1. The player clicks the deal button 2. The dealer gives the player a card 3. The score is updated and the player can choose the next move
Alternate Flows	1. The player draws a card that puts them over 21 a. The dealer wins the hand and the next round starts 2. The player draws a card that gives a score of exactly 21 . The player wins the hand and the next round starts
Minimal Guarantees	The player will have all the cards for the round dealt to them
Success Guarantees	The player will get a card and have the option to choose another

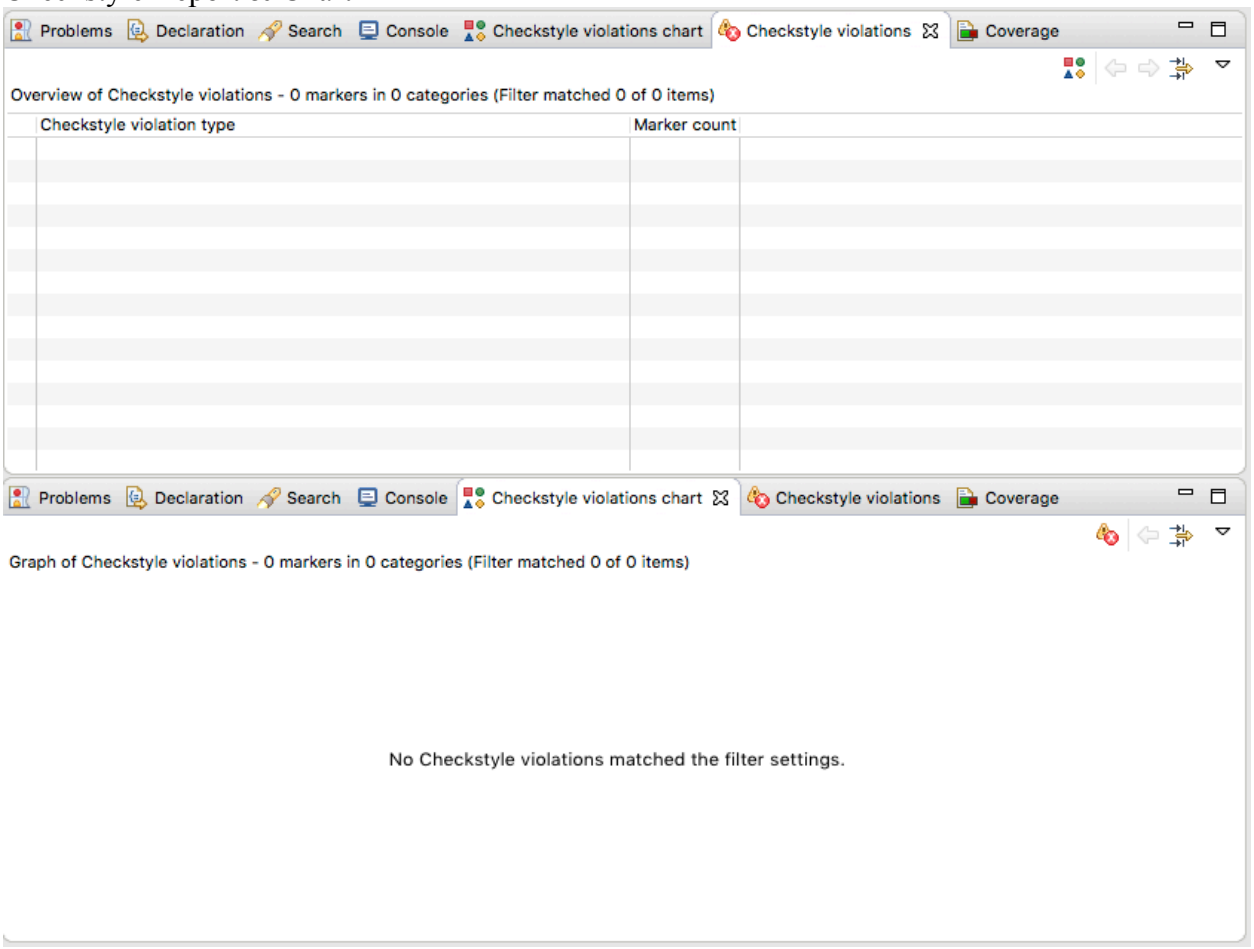
Use Case 3:

Name	Dealer plays
ID	UC3
Brief Description	When the dealer plays the round and the round ends
Actors (primary and supporting/secondary)	Player, dealer
Triggers	Player stands
Preconditions	The player must have less than 21 points in the round
Primary Flow	1. Player hits the Stand button 2. Dealer draws card a. repeat until dealer has likely chance of winning 3. Player starts a new round by clicking Deal button
Alternate Flows	1. Player decides to exit game instead of playing another round
Minimal Guarantees	The round was finished and a winner was decided
Success Guarantees	The player finished the round and started a new round

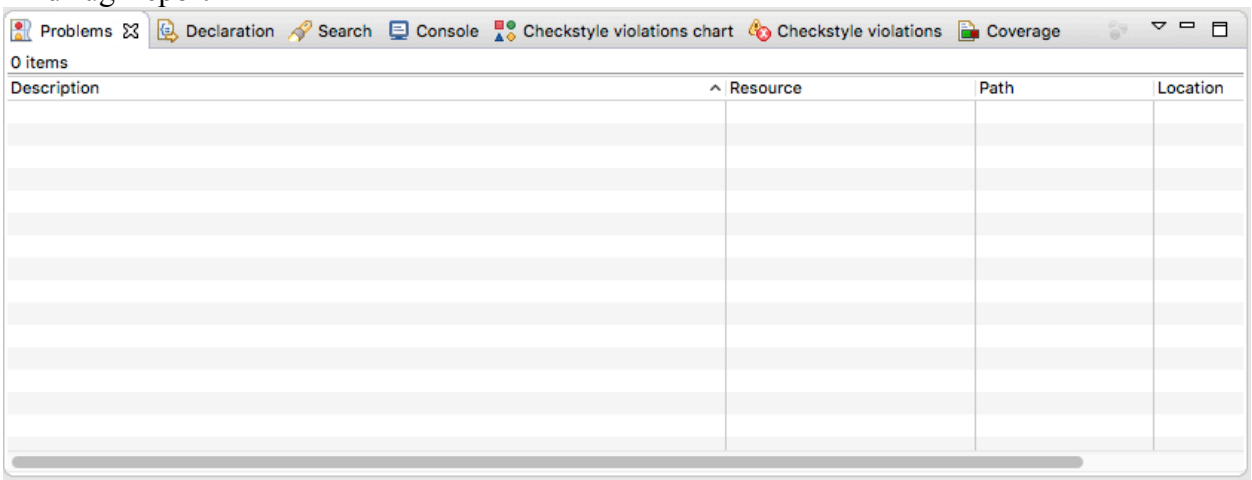
UML Class Diagrams



Checkstyle Report & Chart



FindBug Report



GIT Log and URL

Commits on Oct 19, 2016

Created UseCase Diagram LindyMan93 committed just now	0a13e65
Javadoc and EclEmma LindyMan93 committed 3 hours ago	c1018e4
Fixing JavaDocs LindyMan93 committed 8 hours ago	d7d7409
Create README.md LindyMan93 committed on GitHub 8 hours ago	aa2b554
Package Javadoc LindyMan93 committed 8 hours ago	fb423a5
Fixed JavaDoc Checkstyle errors. LindyMan93 committed 9 hours ago	1aa8ed6
Created UML Diagram LindyMan93 committed 9 hours ago	a20e9b8

Commits on Oct 18, 2016

Fix long lines introduced by previous commit emersonveenstra committed a day ago	Verified	9697695
Added findbugs report, fixed any errors it found emersonveenstra committed a day ago	1 Verified	90e9496
Fixed all checkstyle bugs except javadoc ones emersonveenstra committed a day ago	Verified	117eb0a
Fixed hardocded gvsu.zip path emersonveenstra committed a day ago	Verified	9f825e7
First Commit LindyMan93 committed 2 days ago		791b3a5

https://github.com/LindyMan93/CIS350_Blackjack

Javadoc API of classes

This will be located in two separate links attached to Email.

Eclipse Emma Code Coverage Reports:

Problems | Declaration | Search | Console | Checkstyle violations chart | Checkstyle violations | Coverage

GUI (1) (Oct 19, 2016 6:40:33 PM)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
CIS350_Blackjack	90.3 %	1,258	135	1,393
src	90.3 %	1,258	135	1,393
(default package)	90.3 %	1,258	135	1,393
BlackjackGame.java	89.2 %	670	81	751
BlackjackGame	89.2 %	670	81	751
adjustHandValueAce(String)	56.8 %	63	48	111
deal()	89.6 %	164	19	183
placeBet()	65.9 %	27	14	41
BlackjackGame()	100.0 %	29	0	29
checkWinner()	100.0 %	53	0	53
clearTable()	100.0 %	17	0	17
createHandPiles()	100.0 %	33	0	33
dealerCountTotal()	100.0 %	3	0	3
dealerDraw()	100.0 %	29	0	29
doubleDown()	100.0 %	14	0	14
getCreditBalance()	100.0 %	3	0	3
getDealerCards()	100.0 %	3	0	3
getMessage()	100.0 %	3	0	3
getPlayerCards()	100.0 %	3	0	3
newShoe()	100.0 %	16	0	16
playerCountTotal()	100.0 %	3	0	3
playerDraw()	100.0 %	53	0	53
realCardValue(GVcard, String)	100.0 %	75	0	75
stand()	100.0 %	79	0	79
GUI.java	91.6 %	588	54	642
GUI	91.6 %	588	54	642
actionPerformed(ActionEvent)	81.8 %	243	54	297
main(String[])	100.0 %	3	0	3
GUI()	100.0 %	72	0	72
dealerCountMess()	100.0 %	10	0	10
displayHintCard()	100.0 %	25	0	25
setupFrame()	100.0 %	172	0	172
setupMenus()	100.0 %	63	0	63

Responsibilities/Role of Each Team Member

Nathan Lindenbaum:

1. Research Blackjack rules and strategies.
2. Create Blackjack Game and GUI Classes.
    a. All blackjack options (Hit, Stand, Deal, Cards)
    b. Use GVpile and GVcard packages.
3. Do initial debugging and fix use errors in GUI.
4. Fix initial Checkstyle violations.
5. Create all Javadoc for:
    a. Classes
    b. Methods
    c. Variables
    d. Package
6. Create UML Diagrams for package and exterior packages.
7. Create Cover Page and Project Descriptions.
8. Dictate feature list for release 1.

Emerson Veenstra:

1. Play Game and do Use Cases
2. Create Use Case Diagram
3. Do coverage reports from unit and system using EclEmma