Black Jack

OOAD Project

East China Normal University

Computer Science & Software Engineering College

Grade 2015

Weiwen Chen 10152510217

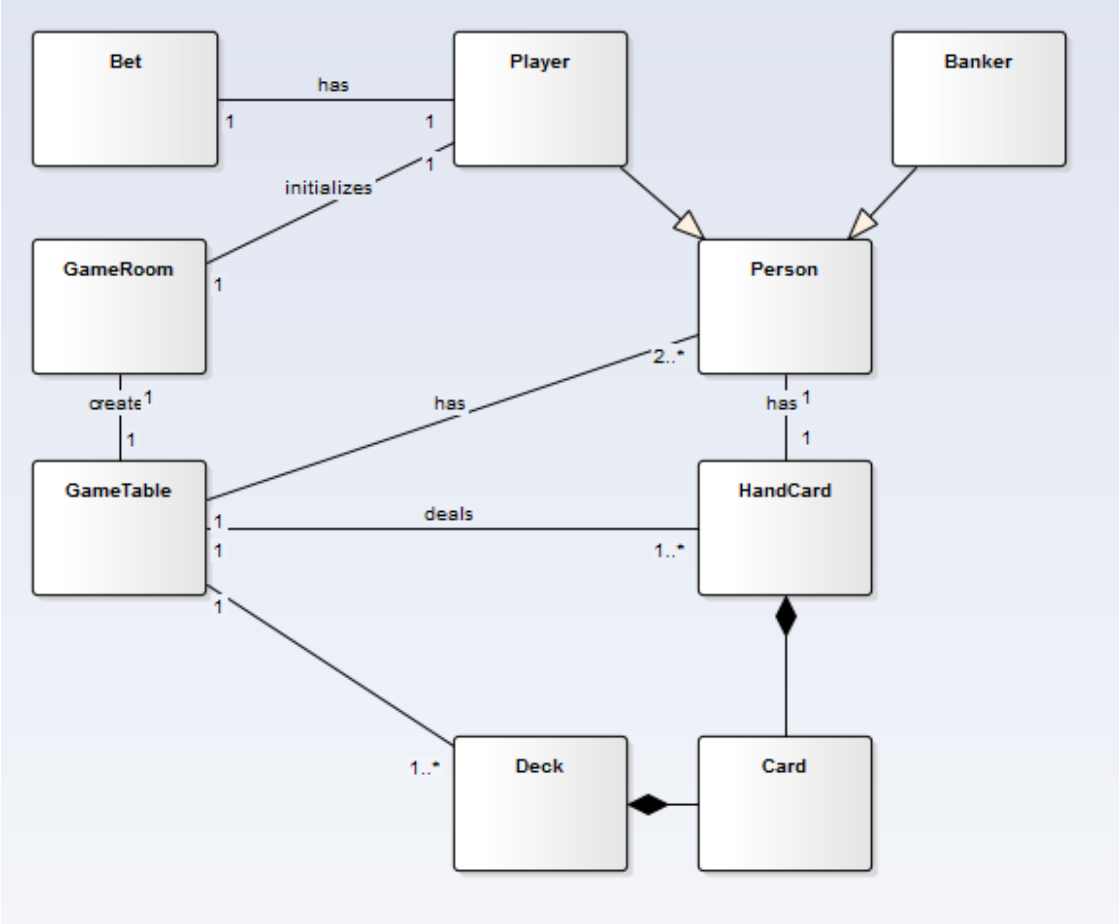
Haoqian You 10152510210

2017.12.07

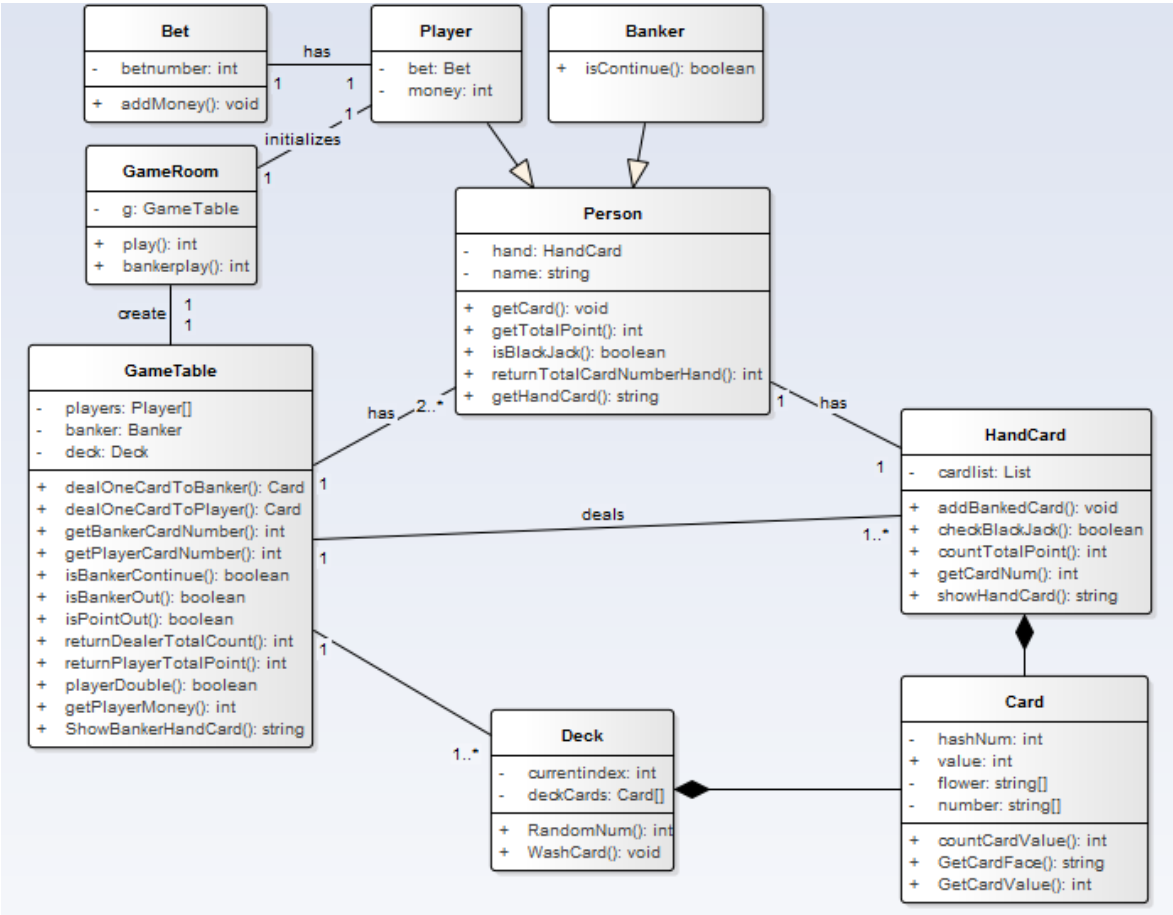
BackGround

Firstly, to be aware of how BlackJack works, we did some research on 4399. We saw too many brilliant GUI. Our instructor, however, told us UI is not so significant as our design because this course is called 'OOAD', which emphasizes Analysis & Design. As a result, in our first edition, we made our users to play the game with a console. What's more, many players play one game in one PC is stupid, so, we have only one player in our first edition.

Domain Model



Class Model



The following is our introduction for every class:

Class Card: This class has four attributes, hashNum, value, flower, number. However only two of them have meaning. The flower and number are two arraies of const. Because the attribute flower doesn't have any usage in comparing handcard's value between player and banker, we abandon this attribute. As we all know, a pack of cards has 52 cards(drop the Jokers). We number them from 0 to 51, which is called hashNumber(attribute hashNum). hashNum / 4 + 1 is this card's value, and hashNum % 4 is No of its suits(flower[hashNum % 4] is the suit string of this card). So at first our deck are arranged like this: '♥A, ♦A, ♠A, ♣A, ♥2, ♦2, ♠2, ♣2, ♥3, ♦3, ♠3, ♣3 ....', for card 'A', in different conditions it has different values, and to display it we need A instead of 1, we have a special method to handle this. Another const array number, this is only to dis three cards J Q K, their value(calc by hashNum algorithm) is 11, 12, 12, respectively. However, in BlackJack Game, we make their value to 10, 10, 10, which is managed in function countTotalPoint().

Class HandCard: It is inheritancing from Card. It can calculate cards's value which are in person's hand. We matain a List of Card here, which is called CardList, to express the cards in person's hand.

Class Deck: It is describe the card's characteristics. It is mainly to generate a pokercard for players to use. what's more, it can wash card, deal card and so on. What we need to say more here is, the functionWashCard and how we deal cards. To wash our deck, we randomly choose two cards, the swap them(I have to say, C# does not have its own swap() method, which is really stupid = =). We do this operation for many times, which simulates the process of washing cards. Then we always select card from the top of the deck.

Class Person: It mainly contains players's and bankers's shared attributes and operates for them to inheritance to reduce unnecessary code. Both player and banker have HandCard, so it is here. They both have names, even if this does not really meaningful.

Class Player: This class is inherited from Class Person. Attribute bet is to record bet money. Attribute Money is to record remained money.

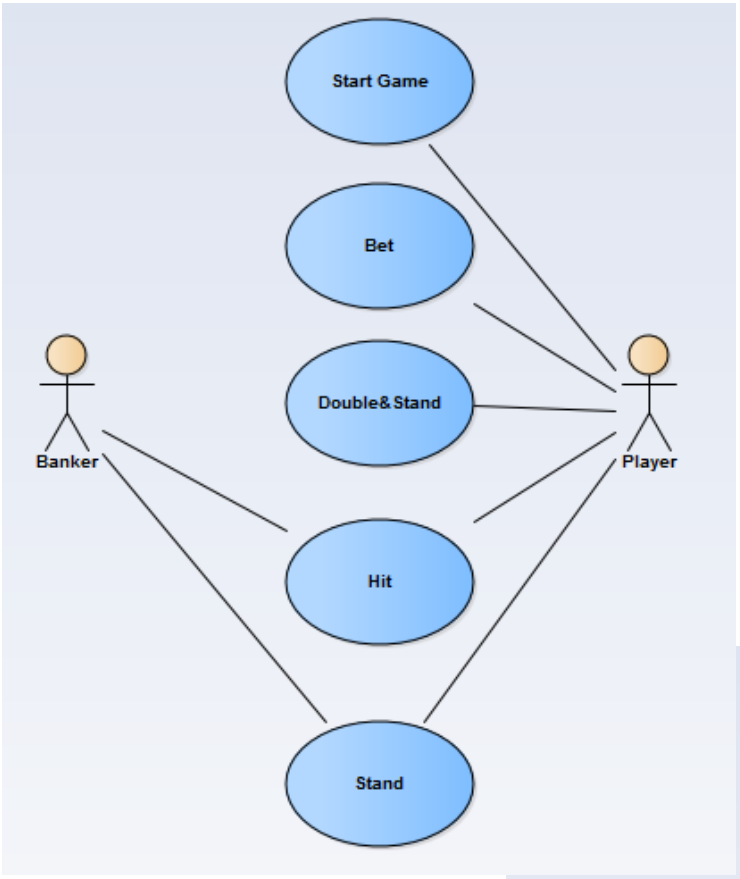
Class Banker: This class is also inherited from Class Person. It has a method to decide whether banker has to hit or not. The difference between player and banker is, the player has money limits, where the banker does not.

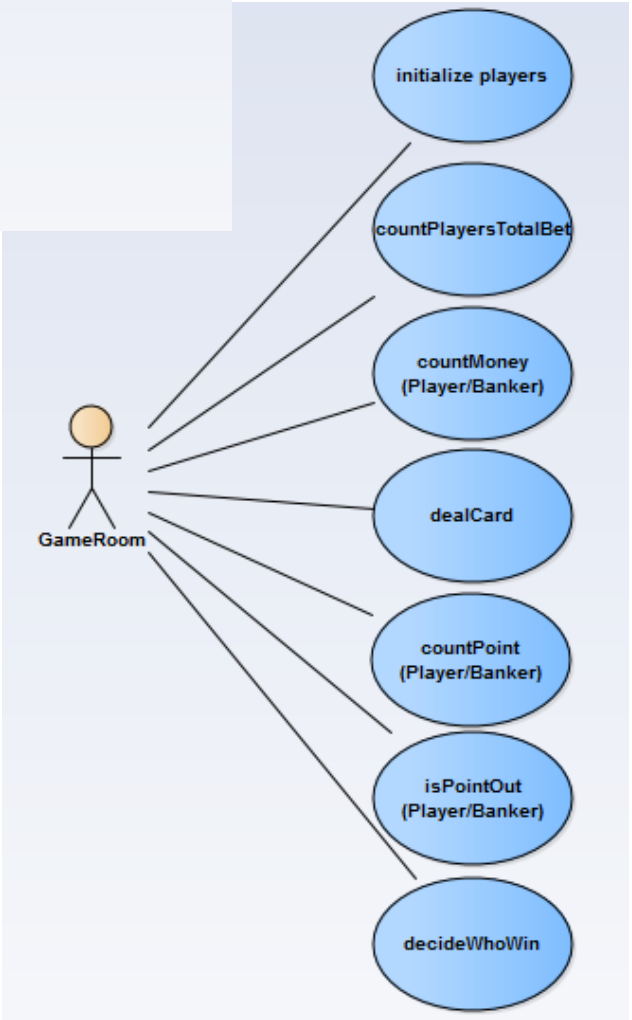
Class GameTable: This class contains almost all class above. Oh actually we have three: Player, Banker, Deck, which make up a BlackJack Game. A great many methods are here to simulate the operations in a game, such as deal card, get money, etc. Aditionally, Class GameRoom mostly communicates with this class.

Class GameRoom: Image that you enter a room of BlackJack Game. You need to talk to the System. It realizes functions that initialize GameTable and Players and begins the game. If we want to a Gamelobby in the future, turn gameTable into gameTables[].

Like money and betmoney the two attributes, there is no need for them to become a class, for the reason that they does not have their own attributes and operates and a simple int has ability to express them.

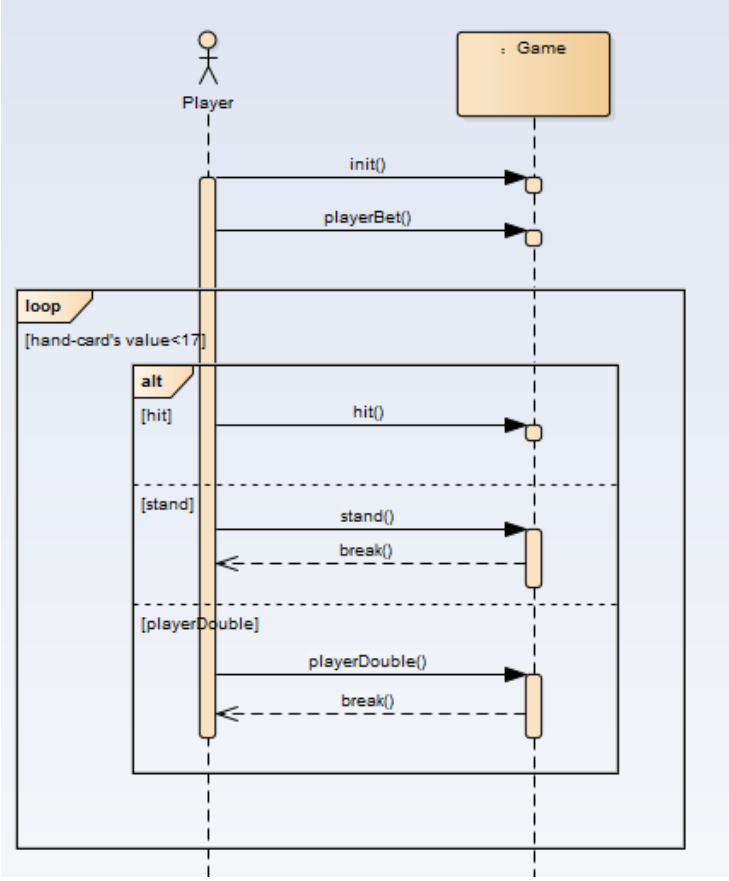
Use Case Model



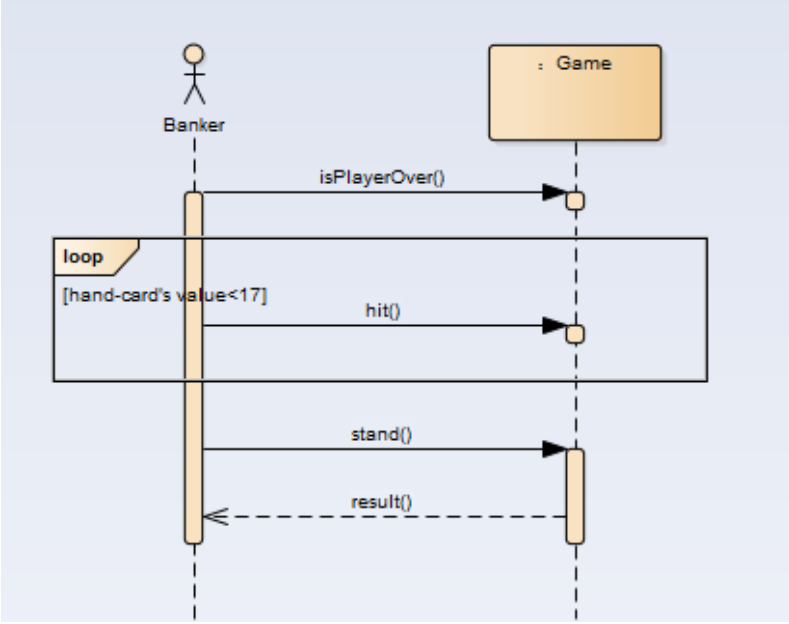


System Sequence Diagram

Player



Banker



Operation Contracts

Player:

Contract CO1:	init
operation:	init()
cross reference:	begin game
precondition:	system is going normally
postcondition:	player is already to play this game

Contract CO2:	playerBet
operation:	playerBet()
cross reference:	do bet
precondition:	game begins and player is already
postcondition:	system records player's bet and his remained money updates

Contract CO3:	dealOneCardToPlayer
operation:	dealOneCardToPlayer()
cross reference:	hit
precondition:	player's hand-card's total value is under 21 and player wants to hit
postcondition:	a new card adds to player's hand-card and hand-card's value updates

Contract C04:	Stand()
operation:	Stand()
cross reference:	directly end
precondition:	player's hand-card's total value is under 21 and player wants to stop getting another card
postcondition:	player's turn is over

Contract CO5:	playerDouble
operation:	playerDouble()
cross reference:	double and end
precondition:	player's hand-card's total value is under 21 and player wants to double the bet and stop getting another card
postcondition:	system records player's bet to double and his remained money updates and player's turn is over

Banker:

Contract CO1:	decide whether player's turn is over
operation:	no opetation
cross reference:	no
precondition:	game is going normally
postcondition:	banker's turn is beginning

Contract CO2:	dealOneCareToBanker
operation:	dealOneCareToBanker()
cross reference:	hit
precondition:	banker's hand-card's value is under 17
postcondition:	a new card adds to banker's hand-card and hand-card's value updates

Contract CO3:	directlystop
operation:	no operation and break directly
cross reference:	end
precondition:	banker's hand-card's value is not under 17
postcondition:	banker's turn is over and system begins to decide who is the winner

Display

We do not have GUI here, because our instructor said that is not important (The real reason is we are lazy).

Example1

```
来啊，快活啊，反正有，大把时光= =
Please enter your name:
sweet
Welcome, sweet
现在您已坐在 BlackJack 的桌前， 来一盘吗（y/n）
y
您的余额为： 1000, 请下注：
1000
得到一张牌：黑桃 10
得到一张牌：红桃 6
您有三种选择：stand(直接结束), hit(继续要牌), double(加倍结束).
输出(s/h/d)进行您的选择：
d
钱不够加倍了，请做其他选择
您的点数为 16
--------您的表演结束了---------
庄家爆牌了，输了输了
庄家手牌为：红桃 7 梅花 9 梅花 8
发生这种事，我很抱歉。
现在余额为 0
再来一轮？
n
Goodbye.
```

Example2

```
您的余额为： 1000, 请下注：
1000
得到一张牌：方片 6
得到一张牌：梅花 4
您有三种选择：stand(直接结束), hit(继续要牌), double(加倍结束).
输出(s/h/d)进行您的选择：
h
得到一张牌：梅花 A
您有三种选择：stand(直接结束), hit(继续要牌), double(加倍结束).
输出(s/h/d)进行您的选择：
s
您的点数为 21
--------您的表演结束了---------
庄家手牌为：梅花 2 红桃 5 黑桃 2 红桃 10
赌神你好。。
现在余额为 2000
```

Summary

This lab is not only exercises our logical thinking ability but also strengthen our abilities of OOAD. We tasted sour and bitter, sweet and happiness in this lab. We found the joy of analysis&design and inspired our passion to do the best. Thanks to this lab, we learn more new things and reinforce our knowledge. We love BlackJack! We love OOAD, GPA++.

Now the only thing I want is, my hair.

 --> 

Reference

UML 和模式应用

BlackJack(Baidubake)

BlackJack in 4399.com

Another BlackJack

Read More

https://github.com/LittleSweetHeart/BlackJack

2017.12.07 Wednesday

in Fifth Dormitory