Blackjack simulator / framework for client optimisation

Henrik Eideberg

2017-06-04

Table of Contents

Blackjack simulator / framework for client optimisation **1**
- Table of Contents 2
- Introduction 3
  - Purpose 3
  - Class Diagram 3
  - Project documentation 4
    - XML 4
    - Visual studio repo 4
  - What subject from the course is covered in the project 4
- Testing 5
  - Unit testing 5
  - Human Player 5
- Limitations 5
  - Stackoverflow exception 5
  - Split not implemented 5
  - ‘Money’ not implemented 5
- Backlog and further work 5
- Time spent during project 6

Introduction

Black jack game application to evaluate the ‘best game rules’ for a blackjack player/client.

Purpose

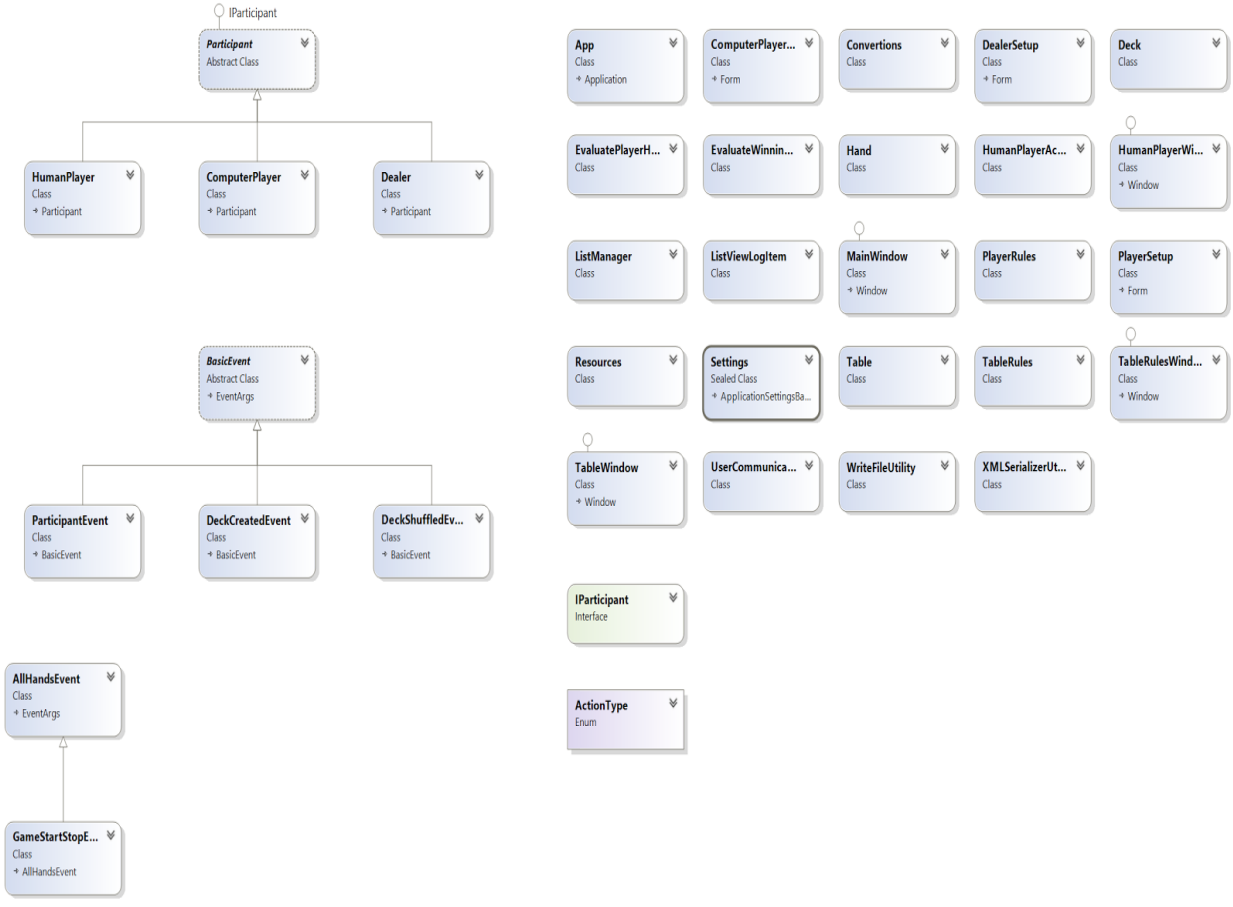
Some co-workers are developing a blackjack server in Erlang. They have invited players to create blackjack clients/players to compete with other clients/players using their server. To be able to win, I would like to create my own application in which I could simulate and come up with the best client, to work out ‘the best set of rules’ for my client.

Hence, the purpose of this application is to be able to

- log played hands and output them in somewhat human readable format
- easily set up a rules which can affect the game outcome.
  This version of the application includes very limited number of rules (nr of games to be played and when to stand). Adding more rules and corresponding logic is not part of this project.

A human player is also included in this application, mostly used for testing the user interface of the application.

Class Diagram



Class diagram is also available in the project.

Project documentation

XML

XML documentation is autogenerated for the two included projects

BlackJack: bin\Debug\BlackJack.XML
BlackJackUnitTestProject: bin\Debug\BlackJackUnitTestProject.XML

Visual studio repo

The project is available in GIT repository at visualstudio.com and can be cloned with:

GIT: ssh://henrikeideberg@henrikeideberg.visualstudio.com:22/_git/BlackJack
HTTPS: https://henrikeideberg.visualstudio.com/_git/BlackJack

What subject from the course is covered in the project

Below are some examples of where in the code material from the course is used.

- **Inheritance**
  The abstract class Participants implements the interface IParticipant. The classes Dealer, HumanPlayer and ComputerPlayer inherit the abstract class Participant.
- **Polymorphism**
  The class Table maintains a list/dictionary of Participants in which it will at run time add either HumanPlayer, ComputerPlayer or Dealer.
- **Abstraction**
  The class Participant is an abstract class.
- **Exception**
  Exception handling is e.g. used at XML serialisation/deserialisation of rules (player-rules and table-rules).
- **Serialisation/Data perseverance**
  The player and table rules can be imported and exported from and to XML. Demofiles are included in folder DemoFiles.
- **Generics**
  The table maintains two dictionaries in which ParticiPants and list of cards (List<Hand>) are stored during the game.
- **Events**
  The class Table receives and sends events to e.g. user interface of the Table (class TableWindow) and HumanPlayer. Example of event DeckShuffledEvent, which is an event from Table to TableWindow to inform that the deck has been shuffled.
- **Delegates**
  Besides the previously mentioned Events, there is a simple delegate implementation in class EvaluatePlayerHand.

Testing

Unit testing

Following unit tests have been created;

- ComputerPlayerUnitTest
- DealerUnitTest
- DeckUnitTest
- EvaluatePlayerHandUnitTest
- EvaluateWinningHandUnitTest
- HandUnitTest
- HumanPlayerUnitTest
- PlayerRulesUnitTest
- TableUnitTest

Human Player

The human player has been used to manually test and verify the user interface and interactions.

Limitations

Stackoverflow exception

While running the ComputerPlayer a bug in the design was found. When asking the ComputerPlayer to run more than 200 games then application crashes with a StackOverflowException because of a mutual recursion problem (two or more methods call each other in repeated succession) with methods BlackJack.Table.ContinueGame and BlackJack.Table.DealInitialCards. A call stack is attached in folder ‘StackOverflow crashes’. The problem remains in this version of the application. To prevent the application from crashing the ComputerPlayers are limited to 50 games per play.

Split not implemented

‘Money’ not implemented

Since ‘money’ is not implemented, also betting, side-betting and the action Double are missing from the game.

Backlog and further work

Table 1 shows the scope of this project in backlog-‘items’. The work completed is represented by green boxes and the work not completed is represented by red boxes.

Time spent during project

Below table shows time (~in nr of days) spent on each backlog item.

Backlog		May																June			
Item	Description	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	1	2	3	4
	Project proposal																				
1	Set up Project/Solution																				
2	Deck																				
3	Table. 1st version																				
4	GUI. Lobby and Table																				
5	Hand																				
6	IParticipant																				
7	Participant																				
8	Dealer																				
9	DealerSetupWindow																				
10	PlayerSetup																				
11	HumanPlayer <class>																				
12	HumanPlayer <Window>																				
13	TableWindow <Window>																				
14	Table <class>																				
15	HumanPlayer <class>																				
16	HumanPlayer <Window>																				
17	TableWindow <Window>																				
18	Table <class>																				
19	HumanPlayer <class>																				
20	HumanPlayer <Window>																				
21	EvaluateWinningHand																				
22	TableRules <class>																				
23	EvaluatePlayerHand																				
24	Table <class>																				
25	ComputerPlayer <class>																				
26	PlayerRules <class>																				
27	ComputerPlayerSetup-window																				
28	ComputerPlayer v2 <class>																				

29	Table export/import rules <class>																					
30	Table export/import rules <window>																					
31	XML documentation																					
32	Project submission																					
33	Game statistics for Player																					
34	Hand analysis for Player																					
35	Table <class> handle split																					
36	Solve stack overflow crash																					
37	Export listbox to textfile																					

Table 1: Backlog and time spent on each item