Skills Demonstration 1 – Blackjack



Dylan Convery

SD6

Component Title and Code – GUI Programming 6N0736

**Blackjack Development Log**

|  |  |
| --- | --- |
| **0.1:** | Initially built out a basic structure for a card class which generated a random number from 0 to 12 that represented from Ace to Jack, and a random number for the suit of the card. |
| **0.2:** | I knew I needed somewhere to store a players cards and so I set out in creating a class that could do just that called Hand. I tried writing my classes in such a way as to make them portable as to be able to create other card games.  I then implemented basic game functionality by creating a Game class and using message boxes and buttons to display my hand value. |
| **0.3:** | I then moved onto implementing some GUI and visual elements. I did this by creating a HandUtility class. This class would be to handle the drawing the corresponding cards in the players hand to the screen through the use of some functions called genCardName() which creates a list of all the names of the cards in the players hand, and then passes this list of names to Draw() which then loops through all the images of cards and find the corresponding image for each name in the list returned by genCardName(). |
| **0.4** | I revisited the Game class, rewriting some of the functions and re-implementing the core game mechanics. I also rewrote how the sum of each players hand is calculated.  I also added a loop for a Dealer which plays the role of the opponent in Blackjack. It would loop until the total value of the dealers hand was greater than or equal to the value 16 like in typical Blackjack. |
| **0.5** | By this point, the game of semi functional. I then set out in creating a Login form which would be shown before the game begins.  I knew I wanted watermarking where various textboxes and other elements would display information before they had been clicked and reshow that information after they had been exited and left blank so I set out in doing so. |
| **0.6** | After finishing the implementation of the Login Screen, I revisited the core game. I wanted to have card flipping so I came up with a way of storing a list of flipped cards and their images for a later time and a way of flipping them back to their original position and image. |
| **0.7** | I then began reworking various functions throughout the program to offer better interactions to the various classes that they belonged to. I also took this time to add some error checking. |
| **0.8** | I then decided to revisit the interaction behind the core game and the GUI. I rewrote and better expressed some functions. |
| **0.9** | I decided to add some offsetting to where and how the various cards in both the player and house hands are drawn as to have the game reflect how cards are drawn on a real blackjack table. |
| **1.0** | I polished up a lot of the code, rewrote the various comments and fine-tuned the GUI elements of game and Login Screen. |

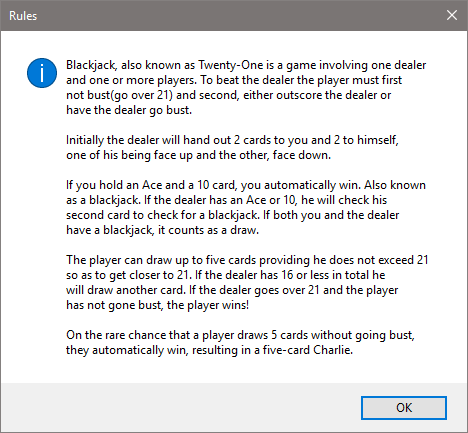
**Data Dictionary**

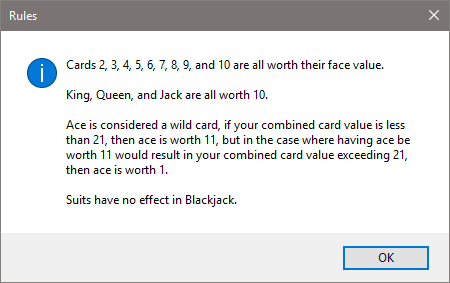
|  |  |  |  |
| --- | --- | --- | --- |
| **Entity Name** | **Class** | **Data Type** | **Description** |
| image\_list | HandUtility | ImageList | Holds the card images. |
| form | HandUtility | Form2 | Used for operating and drawing to form2. |
| hand | HandUtility | Hand | Stores a copy of the hand to be drawn to the screen. |
| card\_point | HandUtility | Point | Point on the screen used for reference on where to start drawing from. |
| x\_offset | HandUtility | int | Used for offsetting the X position of a PictureBox object. |
| y\_offset | HandUtility | int | Used for offsetting the Y position of a PictureBox object, |
| CARD\_SIZE | HandUtility | Size | Represents the constant size of a card image. |
| flipped\_cards | HandUtility | List<FlippedCard> | Holds a list of flipped cards. |
| picture\_boxes | HandUtility | List<PictureBox> | Holds a list of PictureBoxes. |
| HandUtility | HandUtility | Function | Constructor of HandUtility used for creating a HandUtility object. |
| Draw | HandUtility | Function | Used for drawing the cards stored in hand to the screen. |
| Flip | HandUtility | Function | Used for ‘flipping’ a specific card in a hand and setting its back colour. |
| Unflip | HandUtility | Function | Used for ‘unflipping’ a specific card in a hand and restoring its corresponding image. |
| genCardName | HandUtility | Function | Used for generating the correct image names for the corresponding cards in hand. |
| clearImages | HandUtility | Function | Used for removing all generated PictureBoxes on Form2. |
| my\_hand | Hand | List<Card> | Holds a list of all cards currently in your hand. |
| Hand | Hand | Function | Constructor of Hand used for creating a Hand object. |
| getHand | Hand | Function | Returns a read-only list of all cards currently in a hand. |
| currentCardCount | Hand | Function | Returns a count of all the cards currently in a hand. |
| drawCard | Hand | Function | Adds a new card to a hand. |
| addCard | Hand | Function | Add a specific card to a hand. |
| removeCard | Hand | Function | Can remove a specific card from a hand. |
| TWENTY\_ONE | Game | int | Represents the game constant of 21. |
| HOUSE\_THRESHOLD | Game | int | Represents the maximum hand value which the dealer can draw a new card up until. |
| ACE\_\_MULTIPLIER | Game | int | Represents the value which is added to a hand value if the hand holds an ace. |
| TOTAL\_CARDS\_ALLOWED | Game | int | Represents the total number of cards that each user is allowed to draw. |
| user\_hand | Game | Hand | Stores the players hand. |
| house\_hand | Game | Hand | Stores the dealers hand. |
| user\_image\_util | Game | HandUtility | Used for drawing the users hand to the screen. |
| house\_image\_util | Game | HandUtility | Used for drawing the dealers hand to the screen. |
| form | Game | Form2 | Used for operating and drawing to form2. |
| Game | Game | Function | Constructor of Game used for creating a Game object. |
| Hit | Game | Function | Provides a new card to a player and checks whether they have blackjack or have exceeded 21. |
| Hold | Game | Function | Triggers the end game scenario and operates the dealers’ role in the game. |
| calculateScore | Game | Function | Checks the score of the player and dealer to determine a winner. |
| Sum | Game | Function | Calculates the sum of all the cards in a hand. |
| playAgain | Game | Function | Calls various clean-up functions and sets up a new game. |
| Play | Game | Function | Hands out the 2 initial cards to both the player and dealer. |
| front\_of\_card | FlippedCard | Image | Stores the front face image of a card that has been flipped. |
| position | FlippedCard | int | Holds the position of the card in a hand that has been ‘flipped’. |
| get\_position | FlippedCard | Function | Returns the position in a hand of a card that has been flipped. |
| FlippedCard | FlippedCard | Function | Constructor of FlippedCard used for creating a FlippedCard object. |
| CardNumber | Card | enum | Holds the various values that will represent cards. |
| CardSuit | Card | enum | Holds the various values that will represent card suits. |
| CardBack | Card | enum | Holds the various values that will represent card back colours. |
| rand | Card | Random | Used for generating random numbers. |
| card | Card | CardNumber | Used for storing a cards number. |
| suit | Card | CardSuit | Used for storing a cards suit. |
| Card | Card | Function | Constructor of Card used for creating a Card Object. |
| CardValidity | Card | Function | Used for ensuring a card is valid. |
| getCardNumber | Card | Function | Returns a cards number. |
| getCardSuit | Card | Function | Returns a cards suit. |
| light\_blue | Form1 | Color | Represents the light blue RGB value used on various GUI elements of Form1. |
| dark\_blue | Form1 | Color | Represents the dark blue RGB value used on various GUI elements of Form1. |
| fn\_watermark | Form1 | string | Represents the watermarking text to be shown within the first name textbox. |
| ln\_watermark | Form1 | string | Represents the watermarking text to be shown within the last name textbox. |
| d\_watermark | Form1 | string | Represents the watermarking text to be shown within the day combobox. |
| m\_watermark | Form1 | string | Represents the watermarking text to be shown within the month combobox. |
| y\_watermark | Form1 | string | Represents the watermarking text to be shown within the year combobox. |
| gen\_watermark | Form1 | string | Represents the watermarking text to be shown within the gender combobox. |
| fn\_watermarked | Form1 | bool | Tells whether the watermarking text has been removed or not from the first name textbox. |
| ln\_watermarked | Form1 | bool | Tells whether the watermarking text has been removed or not from the last name textbox. |
| submitButton\_Click | Form1 | Function | Ensures the login screen has been properly filled out. |
| wordFormatter | Form1 | Function | Capitalizes the first letter for the users first and last name. |
| cancelButton\_Click | Form1 | Function | Invalidates and removes all data entered in the login screen. |
| game | Form2 | Game | Game object used for creating the game. |
| Form2\_Load | Form2 | Function | Initializes the game object. |
| hitButton\_Click | Form2 | Function | Tells the Game object to draw a card for the user. |
| holdButton\_Click | Form2 | Function | Tells the Game object to trigger the end game scenario. |
| playAgainButton\_Click | Form2 | Function | Tells the Game object to trigger the play again scenario. |
| getCardsImageList | Form2 | Function | Returns Form2’s ImageList of cards. |
| toggleButtons | Form2 | Function | Toggles the various buttons on Form2 on and off. |
| fileReader | Form2 | Function | Reads a specified file and then prints its information to the screen in the form of a MessageBox. |

**Controls Dictionary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Control Name** | **Form** | **Control Type** | **Description** |
| buttonHold | Form2 | Button | Tells the Game object to trigger the end game scenario. |
| buttonHit | Form2 | Button | Tells the Game object to draw a card for the user. |
| playAgainButton | Form2 | Button | Tells the Game object to trigger the play again scenario. |
| House | Form2 | Label | Expresses to the user that cards drawn next to this label are those of the dealers. |
| Player | Form2 | Label | Expresses to the user that cards drawn next to this label are his own. |
| menuStrip1 | Form2 | MenuStrip | Used for holding various tool drop down menus. |
| helpToolStripMenuItem | Form2 | ToolStripMenuItem | Holds the various help options available to a user. |
| fileToolStripMenuItem | Form2 | ToolStripMenuItem | Holds the quit option available to a user. |
| Name | Form1 | Label | Expresses to the user that the various controls below are for entering a name. |
| firstNameTextBox | Form1 | TextBox | Used for entering a first name. |
| lastNameTextBox | Form1 | TextBox | Used for entering a last name. |
| Birthday | Form1 | Label | Expressed to the user that the various controls below are for entering a birth date. |
| dayComboBox | Form1 | ComboBox | Used for entering the day part of a birth date. |
| monthComboBox | Form1 | ComboBox | Used for entering the month part of a birth date. |
| yearComboBox | Form1 | ComboBox | Used for entering the year part of a birth date. |
| Gender | Form1 | Label | Expressed to the user that the various controls below are for entering a gender. |
| genderComboBox | Form1 | ComboBox | Used for entering a gender. |
| pictureBox1 | Form1 | PictureBox | Draws a picture to the login screen for aesthetic purposes. |
| termsCheckBox | Form1 | CheckBox | Used for agreeing to the terms and conditions of the game. |
| okayButton | Form1 | Button | Used for confirming the information entered in the login screen. |
| cancelButton | Form1 | Button | Invalidates and removes all data entered in the login screen. |

**Screenshots**

****

****

**Blackjack Software Testing**

**Test Case -** Starting a new game with a hand consisting of an Ace and a King before a fix was applied.

**Expected Results:**

The Form should load to the screen and then show that the user has either drawn with the dealer or that he has won the game.

**Actual Results:**

A MessageBox is shown before the Form is drawn to the screen either stating that the user has drawn with the dealer or that he has won the game. After dismissing the MessageBox, the Form is then drawn to the screen showing what the MessageBox had previously said.

**Test Case -** Using the rules MenuStripItem before a fix was applied.

**Expected Results:**

A MessageBox should display with all the information within the *rules.txt* file located in the root of the project folder.

**Actual Results:**

The program crashes because an exception is thrown. The exception being thrown? The *System.IO.FileNotFoundException.* This is due to me misspelling the title of the file *rules.txt*.

**Test Case -** Using the Stand and Play Again buttons before a fix was applied.

**Expected Results:**

After pressing Stand, the game should trigger the end game scenario where the Hit and Stand button are temporarily disabled and hidden, the dealers’ cards are flipped and drawn, and a winner is determined. After which, the Play Again button should be visible. After pressing the Play Again button, the game should restart and the Hit and Stand buttons should be made available once again along with the Play Again button being hidden.

**Actual Results:**

After pressing Stand, the game triggers the end game scenario where the Hit and Stand button are temporarily disabled and hidden, the dealers’ cards are flipped and drawn, and a winner is determined. After which, the Play Again button is made visible. After pressing the Play Again button, the game restarts and the Hit and Stand buttons are made available once again along with the Play Again button being hidden. The problem is that the border size of the Stand button is for some unknown reason, made to be of BorderSize ‘1’ instead of reflecting what’s set in the Properties tab stating a size of ‘2’.

**Test Case -** Using the Draw function in HandUtility before a fix was applied.

**Expected Results:**

The images of the cards that are held within a hand object would be drawn to the screen by placing a PictureBox onto Form2 and placing the corresponding card image inside.

**Actual Results:**

The function was drawing a new set of PictureBoxes on top of the ones drawn previously each time a user added a card to his hand and so, was wasting unnecessary resources.

**Test Case –** Clicking in and out of either of the Name Textboxes on Form1 before a fix was applied.

**Expected Results:**

The watermarking text would clear and reappear if nothing was entered into the TextBox.

**Actual Results:**

The watermarking text was staying inside the textbox meaning a user would have to first clear the text manually, and then enter his or her name inside. This also meant that the watermarking text was never cleared because text was being left inside the textbox.

**Test Case –** Clicking in and out of any of the various ComboBoxes before a fix was applied.

**Expected Results:**

The watermarking text would clear and reappear if nothing was entered into the TextBox.

**Actual Results:**

The watermarking text was being added as an item inside the ComboBox meaning a user could select “Day”, “Month”, “Year” or “I am…” as an entry into these ComboBoxes. This was not ideal and also broken the watermarking meaning it would never clear.

**Test Case –** Confirming my entry in Form1 but leaving a required field empty before a fix had been applied.

**Expected Results:**

I would be notified of the missing entry in that field and would not be allowed continue until I had fixed this.

**Actual Results:**

The program notified me about the wrong field meaning there was a problem in the steps it ran through to check everything was valid.

**Test Case –** Adding a timer to add some time between each of the dealers’ cards being shown to the screen as to add more immersion.

**Expected Results:**

The end game scenario would run, the dealers’ cards would then be drawn one at a time with a wait time of two seconds between each draw.

**Actual Results:**

The program hung for two seconds and then all at once, the dealers’ cards were drawn to the screen like always.