Media Computing Project

Python 3 blackjack game

Jose Flores, Andrew Long
Computer Science
University of Massachusetts Lowell
Lowell, MA United States of America

***Abstract*—A Python 3 blackjack terminal simulator with a pyGame generated user interface which was done in lieu of completing any other assignment or exams with the intent of fulfilling the requirements for COMP 1000 Media Computing under professor Byung Kim for the Fall 2016 semester.**

***Keywords*—Python 3, pyGame, GUI, blackjack, Media Computing**

I. Project Expectations

A. *Changes to original guidelines*

- Documentation shall not be found on the *gh-pages* branch of the *blackjack* repository of the *uml-cm-f16* organization or the projects Github webpage. **Instead it shall be found within the component classes** Ref. [2] [3]

II. Installation

A. *Dependency Installation*

1. Download and install Python 3 for your system Ref. [5].

2. Download and install Ref. [6].

B. *Source Code Installation*

1. With a terminal client clone the project repository
   git clone https://github.com/uml-cm-f16/blackjack.git

2. Navigate into the project directory:
   cd blackjack

3. With terminal run the command:
   python ./setup.py install

III. Operation

A. *Running the terminal program*

1. [Optional] To test the code base with terminal run:
   python ./terminal.py

B. *Running the GUI program*

1. [Optional] To test the code base with terminal run:
   python ./gui.py

C. *Code management*

1. To unit test the code base with terminal run:
   python ./setup.py test

2. To update the Git repository with terminal run:
   python ./setup.py save {message}
   where {message} is your Git commit message

IV. Code Base

At the root of the project directory are three python files and one directory. They python files are the project landing points.

A. *setup.py*

This is the project setup file and can install dependencies, unit test the code base, and update the Git repository. The following are the commands it can take.

python ./setup.py install

- Installs the application dependencies:
  - pip – Package installer
  - pygame – Graphical User Interface (GUI) framework
  - pytest – Unit test framework
  - pylint – Code format linter

python ./setup.py test

- Runs and auto discovers the unit tests.

python ./setup.py save {message}

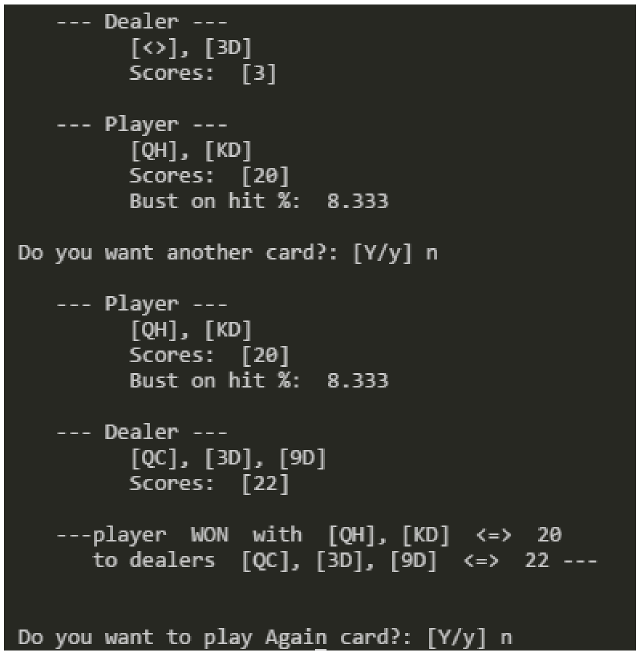
- Updates the Git master branch with changes

B. *simulator.py*

The simulator runs a terminal blackjack game for play between one human player and a computerized dealer. The terminal game shows the cards in play, the player score, and the probability to bust on next hit. This game follows the rules as described in the bicycle cards webpage Ref [4].

python ./terminal.py

- Runs the terminal simulator



```
   --- Dealer ---
        [<>], [3D]
        Scores:  [3]

   --- Player ---
        [QH], [KD]
        Scores:  [20]
        Bust on hit %:  8.333

Do you want another card?: [Y/y] n

   --- Player ---
        [QH], [KD]
        Scores:  [20]
        Bust on hit %:  8.333

   --- Dealer ---
        [QC], [3D], [9D]
        Scores:  [22]

   ---player  WON  with  [QH], [KD]  <=>  20
      to dealers  [QC], [3D], [9D]  <=>  22 ---


Do you want to play Again card?: [Y/y] n
```

Fig1 Terminal simulator screenshot.

C. *gui.py*

The GUI application runs in a windowed environment of 1000px wide and 800px high. It updates the screen on a looping interval. The GUI like the simulator, whom share the same logic, create a blackjack game environment for play between one human player and a computerized dealer. The terminal game shows the cards in play, the player score, and the probability to bust on next hit. This game follows the rules as described in the bicycle cards webpage Ref [4].

python ./gui.py

- Runs the windowed GUI simulator

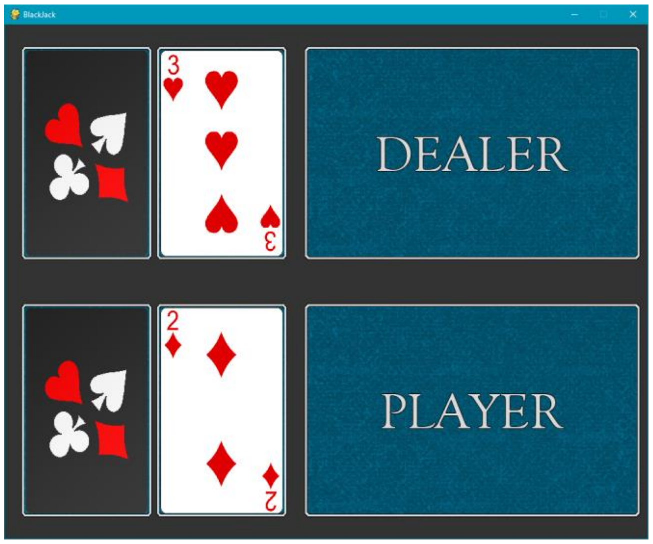


Fig 2. Gui application screen shot, after initial deal.

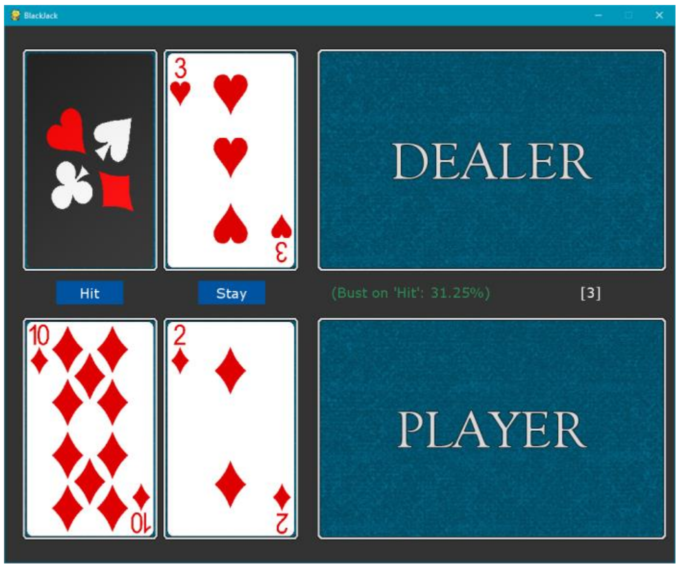


Fig 3. Gui application screen shot, at the players turn.

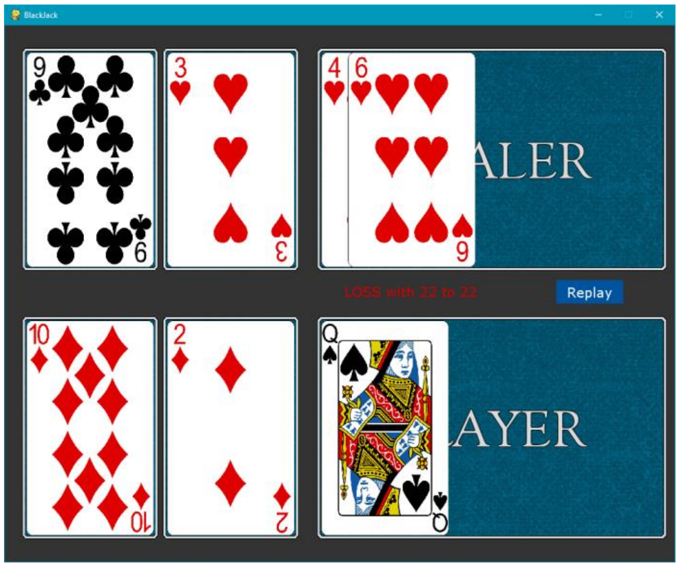


Fig 4. Gui application screen shot, after the dealers turn.

D. *App/*

The application directory contains four sub directories. With the remainder of our files. Thy are organized as follows.

1. doc – The application documents, such as the proposal and reports.
2. img – Holds all the images, and psd files that were used to generate them.
3. src - The application classes
4. test – The unit test files

V. Code Authors

TABLE I. Ownership

Task	**Description**	**Owner**
setup.py	Python script to install dependencies, test source, and deploy to Git repository.	J
gui.py	The GUI program launcher.	A
terminal.py	The terminal application launcher.	J
app/src/logic/card.py	A card class.	J
app/test/test_card.py	Unit tests for the card class.	J
app/src/logic/deck.py	A deck class of 52 cards.	J
app/test/test_deck.py	Unit tests for the deck class.	J
app/src/logic/hand.py	A hand of cards class.	J
app/test/test_hand.py	Unit tests for the hand class.	J
app/src/logic/player.py	Extends Hand by giving player actions.	J
app/test/test_player.py	Unit tests for the player class.	J
app/src/logic/dealer.py	Extends Player by giving deck handling actions.	J
app/test/test_dealer.py	Unit tests for the dealer class.	J
app/src/logic/blackjack.py	Extends Dealer by adding blackjack rules	J
app/test/test_blackjack.py	Unit tests for the blackjack class.	J
app/src/pygame/engine.py	The UI game interface	B
img/*	Card, buttons, background, etc.	A
UI design	Determine layout, calculate positions for buttons graphics and probability display.	A
UI proof of concept	Create a window, add UI elements, and input elements. Test inputs and output functionality.	A
UI game flow	Make sure game actions can be implemented. Such as adding a card, and folding for both the dealer and player	A
UI/ Sim integration	Integrate UI to call simulator	B
Game testing	Test completed application for functionality and usability.	B

References

[1] *https://github.com/uml-cm-f16/blackjack.*

[2] *https://github.com/uml-cm-f16/blackjack/tree/gh-pages*

[3] *https://uml-cm-f16.github.io/blackjack/*

[4] http:// *www.bicyclecards.com/how-to-play/blackjack/*

[5] https://www.python.org/downloads/

[6] https://git-scm.com/downloads