Version 1.0

syst 17796

Deliverable 3

Contents

[JUNIT Tests RESULTS 1](#_Toc38058525)

[CLASS DIAGRAM UPDATED 2](#_Toc38058530)

[GIT REPOSITORY 3](#_Toc38058531)

[GIT REPOSITORY OF JUNIT Tests 3](#_Toc38058532)

[MANUAL Tests RESULTS 3](#_Toc38058533)

[rubric 4](#_Toc38058538)

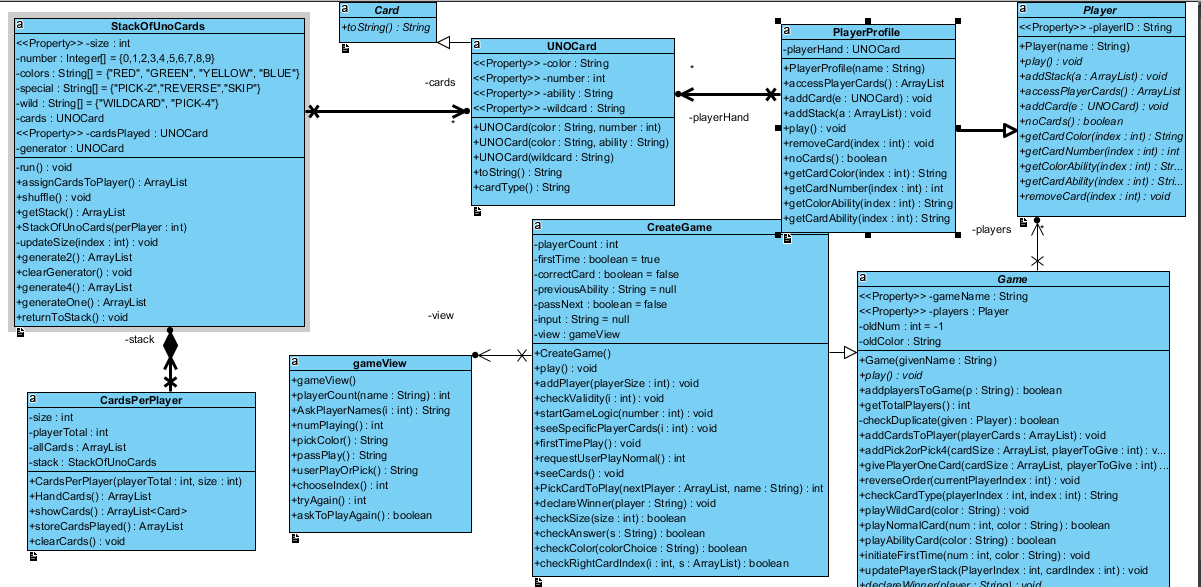
[Final Document Layout 6](#_Toc38058539)

## JUNIT Tests RESULTS

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement | Use Case | Test Method (ClassName.methodName) | Status (Date) |
| Test if user receives right cards when given pick 4 wildcard/pick 2 | Regular play | CardsPerPlayerTest.checkUserGets4Cards()  CardsPerPlayerTest.checkUserGets2Cards() | Pass (April 9, 2020) |
| User picks one card when they don’t have any | Pick card | CardsPerPlayerTest.checkUserGets1Card() | Pass(April 9,2020) |
| Verify each player gets their own hand | Regular play – Each player receives same hand | CardsPerPlayerTest.checkPlayerHasCards() | Pass(April 11, 2020) |
| Check number of players satisfies 2-4 players | Add players to game | CreateGame.checkSizeTest()  CheckSizeTestNegative() | Pass(April 11  2020) |
| Check the right UNO color card was played by user | Play card – verify correct card to continue | CreateGame.checkColorTest()  CreateGame.checkColorTestRandom()  CreateGame.checkColorTestUpperCase() | Pass(April 11,2020) |
| Go to next player | Pass or play | CreateGame.checkPassPlayTest()  CreateGame.NotcheckPassPlayTest()  CreateGame.checkPlayTestRandom()  CreateGame.checkPlayPassNumber() | Pass(April 11,2020) |
| Check right index picked by user to choose card | Play card | CreateGame.checkRightCardIndexTest()  CreateGame.checkRightCardNegativeIndexTest() | Pass(April 11,2020) |
|  |  |  |  |

## CLASS DIAGRAM UPDATED

The following class diagram has been updated with all the attributes and methods completed. One class was added, that is the view class, this class is used by the user to see what will be displayed on their screen and it is this class where they’ll, always be prompted to answer questions in order for the gameController class to determine the next move for each player and plan the game.



## GIT REPOSITORY

<https://github.com/mohd009/simpleUNO>

## GIT REPOSITORY OF JUNIT Tests

<https://github.com/mohd009/simpleUNO/tree/branchFile/test>

## MANUAL Tests RESULTS

Test report results showing manual test scripts.

**Manual test scripts tested while using the application**

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement | Use Case | (ClassName.methodName) | Status (Date) |
| Player picks the right card by selecting the right index | “First time Play” | CreateGame.requestUserPlayNormal() | Pass (April 9, 2020) |
| Give each player same size of cards | Hand each player a hand | CardsPerPlayer.HandCards | Pass(April 9,2020) |
| Go to next player after picking or playing card | Pass to next player | CreateGame.startLogic | Pass(April 11, 2020) |
| Enter player names and check duplicates | Enter player name when start game | CreateGame.addPlayer | Pass(April 11  2020) |
| Check if user used a special card | Regular play- to determine who’s next | CreateGame.startLogic | Pass(April 11,2020) |
| Remove a card from player each time they play a card | Regular play- to update card hand of each player | Game.updatePlayerStack | Pass(April 11,2020) |
| Determine winner of game when they have no cards | Declare winner | PlayerProfile.noCards , CreateGame.declareWinner | Pass(April 11,2020) |
| Determine if player wants to play again | Game done, ask to play again | View.askToPlayAgain | Pass(April 12,2020) |

**You should also include a zip export of your project including the source code for your game and for your tests, labelled with your group name.**

### rubric

| Item | Criteria | Points | Weight |
| --- | --- | --- | --- |
| Class Diagram | Updated to reflect feedback from Deliverable 2, methods and attributes included. Notationally correct and solid design as described in Deliverable 2 Design Document. | 10 | 1 |
| Source Code | Code is completed and conforms to the design specified by the class diagram. Code is functionally correct. Code follows standard coding conventions for comments, naming and indentation | 10 | 3 |
| Source Code Design | The Design produced follows the principles of OOD studied in the course and provides flexibility, reusability and efficiency | 10 | 2 |
| Test Code + Scripts | JUnit tests are completed and pass. JUnit tests validate the one thing they are written to validate (each). JUnit tests conform to test standards taught in class. Manual test instructions are included for any requirements that are unable to be validated using JUnit | 10 | 2 |
| Quality | JUnit tests cover the scope of the project as defined by the students in Deliverable 2 (use cases). Test results report is accurate and shows good, bad and boundary cases for all requirements | 10 | 2 |

## Final Document Layout

One PDF with the following sections denoted using page numbers, headers and a table of contents:

* Updated Class Diagram with methods and attributes
* Reference to the Git repository containing your source code
* Reference to the Git repository location (directory) of your JUnit tests
* Any manual test instructions
* Test Results Report

**AND** One zip archive containing the source code for your game and your JUnit tests, labelled with your group name.