**COSC220 – ASSIGNMENT 3 – T2 2023**

**SydneyNepal CONTRIBUTION SUMMARY**

**SydneyNepal’s names and emails:**

* Bikash Neupane, [bneupan2@myune.edu.au](mailto:bneupan2@myune.edu.au)
* Sabin Dhital, [sdhital@myune.edu.au](mailto:sdhital@myune.edu.au)
* Anil Khattri, [akhattri@myune.edu.au](mailto:akhattri@myune.edu.au)

**Description and Demo Video:**

The tic tac toe game is played on a grid of 3x3 squares between two players. One player takes the “X” symbol and other takes the “O” symbol. Players take turns putting their symbols in empty squares. The first player to get 3 of the marks in a row (up, down, across, or diagonally) is the winner.

[Sprint Video](https://kaf.une.edu.au/media/Kaltura+Capture+recording+-+September+25th+2023%2C+11A05A30+am/1_csfo0f6d)

[Final Demo Video](https://kaf.une.edu.au/media/Kaltura%20Capture%20recording%20-%20September%2027th%202023%2C%206%3A55%3A30%20pm/1_65rw33qa)

**Some difficulties and their resolutions:**

**Integrating the Game UI inside the mnclient**

We first created the game only from the perspective of client side and planned to integrate it to server later. It was very challenging for us to integrate it as we had used JFrame and menubar. But later on, we decided to stick only on JPanel and could only success thenafter.

**Moving the client-side logic to server side**

Again, after completing the integration part in mnclient we had another challenge to shift our client-side logic to server side as well. But after a lot of research, findings, and discussion we finally succeed in that part as well in some extent.

**Links to Wiki Pages:**

[Feature page](https://gitlab.une.edu.au/cosc220-2023/classproject/-/wikis/Tic-Tac-Toe)

[Group page](https://gitlab.une.edu.au/cosc220-2023/classproject/-/wikis/SydneyNepal)

**Group Branch Name:**

[SydneyNepal](https://gitlab.une.edu.au/cosc220-2023/classproject/-/tree/SydneyNepal?ref_type=heads)

**Git issue numbers:**

[#262 - SydneyNepal](https://gitlab.une.edu.au/cosc220-2023/classproject/-/issues/262)

We made single issue and single branch and then did almost all the commits in that branch. Sometimes, we mistakenly made the commit directly to the main branch as well. But then again later on we tried our best to stick on our branch.

**List of main classes:**

**Client:**

* Images,
* MazeDisplay,
* SpaceBot,
* SpaceMaze,
* SpaceMazeSound,
* StatusBar

**Common:**

* SpaceEntity

**Server:**

* GameTimer,
* InteractiveResponses,
* MazeControl,
* MazeInit,
* SpaceMazeGame,
* SpaceMazeServer,
* SpacePlayer

**List of tests created:**

* **SpaceMazeClientTests:**
* testBotMoveUp()
* testBotMoveRight()
* testBotMoveDown()
* testBotMoveLeft()
* testBotMoveCommand()
* testIllegalBotMoveCommand()
* testDistanceCloser()
* testChooseSeeking()
* testMoveCloser()
* testMoveRandom()
* testInvalidMoves()
* testValidMovesAreValid()
* testFindingAChar()
* testLoadCustomFontPublicPixel()
* testLoadCustomFontAquire()
* **SpaceMazeGameTests:**
* testTimerConstructor()
* testTimerStarts()
* testTimeTakenWhenGameRunning()
* testPlayerConstructor()
* testPlayerGetLocation()
* testPlayerUpdateLocation()
* testPlayerCalculateScore()
* testPlayerCheckNumberOfKeys()
* testPlayerAddKey()
* testPlayerResetKey()
* testValidMoveInvalid()
* testValidMoveValid()
* testUpdateKeyStatus()
* testAllKeysStatus()
* testUnlockExit()
* testUpdatePlayerLocationMaze()
* testWormHole()
* testNewLevel()
* testCheckGameOver()
* testMazeControlAchievementVariables()

**Team member’s personal contributions:**

Nikolas Olins:

* Setting up most but not all of the organising documents on O365.
* Designing and implementing the following classes:
* SpaceEntity
* SpacePlayer
* SpaceBot.
* Integrating bot classes where required throughout other classes.
* Creating the sequence diagram.
* Contributing to the class UML diagram and changing the relations to better reflect what they are.
* Attempting to rationalise our server / client class split and responsibilities
* Implemented the Achievements API with Natasha

Andrew McKenzie:

* Designing and creating the following classes:
* Images
* MazeDisplay (with Niraj)
* SpaceMazeSound
* GameTimer
* SpaceMazeServer
* Team Leader
* Planning the meeting agenda
* Running the meetings
* Keeping track of assigned tasks
* Handling most client-server commands with Niraj
* Created and oversaw most of SpaceMazeGame class
* Sprint video editing and base PowerPoint creation
* Recorded the demo video
* Created all sound effects
* Designing and implementing tests for:
* GameTimer
* MazeDisplay

Natasha Hay:

* Designing and implementing the following classes:
* MazeControl
* MazeInit
* Designing and implementing tests for:
* MazeControl class within SpaceMazeGameTests.java
* Implemented the Achievements API with Nik
* Creating all images for the game
* User Story (#6 - SpaceMaze Game (mini-game))
* Wiki feature page (creation, content, editing)
* UML Class diagrams
* Created most of the teams gitlab issues/tasks
* Design committee rep
* Contributed to the Sprint video (end design mockup image and 'Hurdles so far’ section)

Niraj Rana Bhat:

* Designing and implementing the following classes:
* MazeDisplay (with Andrew)
* SpaceMaze
* StatusBar
* InteractiveResponses
* Designing and implementing tests for
* Fonts within clients/SpaceMaze.java
* Designing (with team) and implementing overall UI and UX for:
* Main Menu section
* How to play section
* Status bar section
* Game Over section
* High Score section (Although discarded)
* Implementing simple Header Animation on main menu with Natasha
* Overall Navigation features of the game.
* Created and worked on issues.
* Handling most client-server commands with Andrew
* Contributed to the Sprint video ('Feedback’ section)
* Fixed bugs and optimized code wherever seemed fit.