



48024

Programming 2

Assignment 2

Topics:

OO design, GUIs, MVC, tables, lists

Learning Outcomes:

This assignment supports objectives 3 - 5

Due date:

13 May 2024 - 11:59PM (Monday Week 12)

Weight:

25%

Individual Work

All work is individual. You must write every line of code yourself except for code copied from study module sample code, lecture sample code, tutor demos or lab code.

In most cases, you may discuss ideas, approaches and problems. However, if an assignment task is labeled as “Advanced”, you must not discuss ideas, approaches and problems. Advanced tasks are designed to test your ability to think on your own.

You **MUST NOT** let another student see your solution code, and you **MUST NOT** look at another student's solution code. Sharing your code on public forums such as the discussion board, or Internet forums such as stackoverflow.com is not permitted. More information about Academic Misconduct can be found at:

<http://www.gsu.uts.edu.au/rules/student/section-16.html>

Working Language

You can choose either Java or Python to complete assignment 2. The higher mark between your Java solution and Python solution will be counted into your final grade. However, you are only credited with one of your solutions, either Java or Python, not both of them or the mixture.

The specification is illustrated based on Java. You can simply translate the Java syntax to Python for your Python solution. Detailed explanations about Python criteria will be posted on the FAQ page.

Skeleton Code

As a starting point for this assignment, you must use the skeleton code provided in Assignment 2 Canvas page. There are one option for Python and one option for Java. **You are NOT allowed to modify the model, add or remove any of the starting classes or FXML files.**

Expected workload

The time to do the assignment to a credit level (i.e. a mark between 65% to 75%) has been estimated at 25 hours for a student of average ability who has completed all the tutorial and lab exercises.

Specification

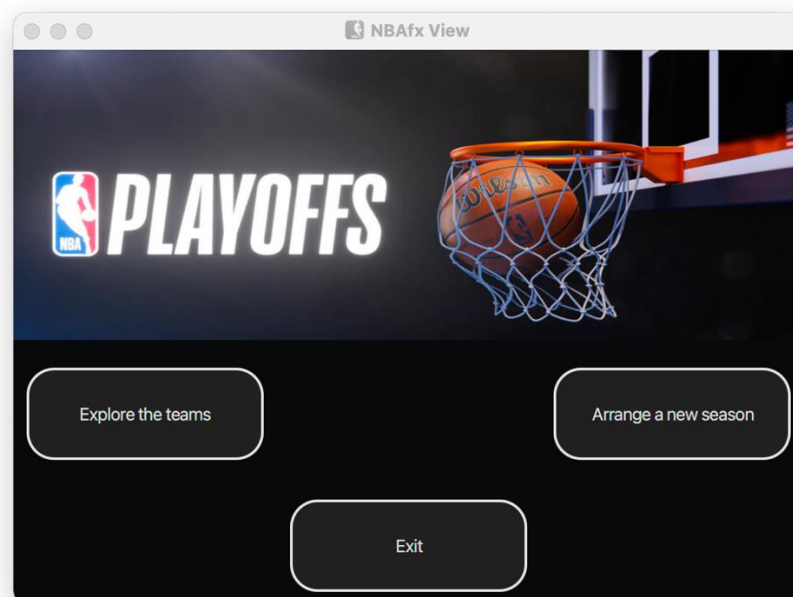
The specification is presented in several parts. In this document is given a series of screen shots and textual descriptions for visual reference.

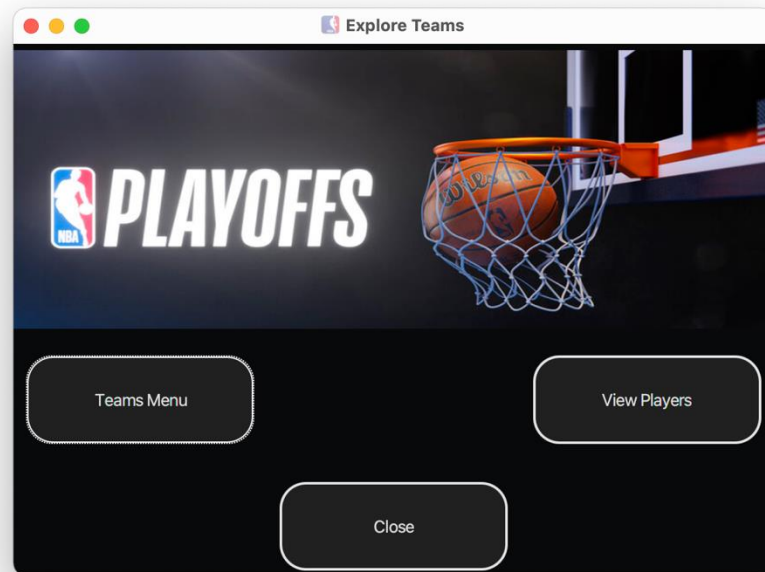
A demonstration video is also presented on Canvas. This demonstration video is considered part of the specification and contains important details about the dynamic function of the assignment.

Menus

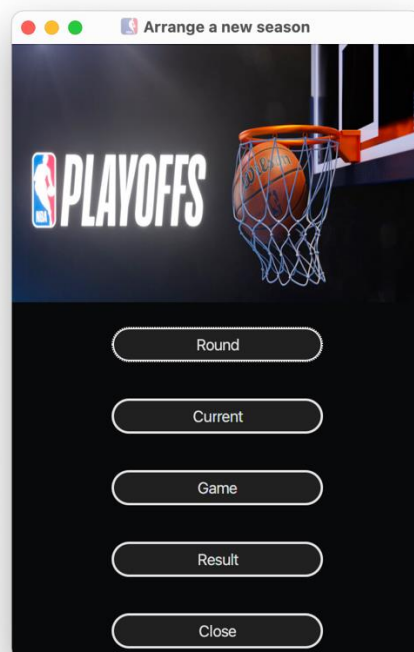
There are two menu layers in the application. The main menu ([AssociationView.fxml](#)) has two sub-menus, the team menu ([ExploreTeamsView.fxml](#)) and the season menu ([SeasonView.fxml](#)). The main menu and the team menu share the same layout but have different functions.

The main menu is opened when the application launches. It has “Explore the teams” button to access the team menu and “Arrange a new season” button to access the season menu. It also has an exit button which shuts the entire application down. The main menu has a header section with a logo “PLAYOFFS”. The title of the window includes an “NBA” icon and the tile “NBAfx View”.





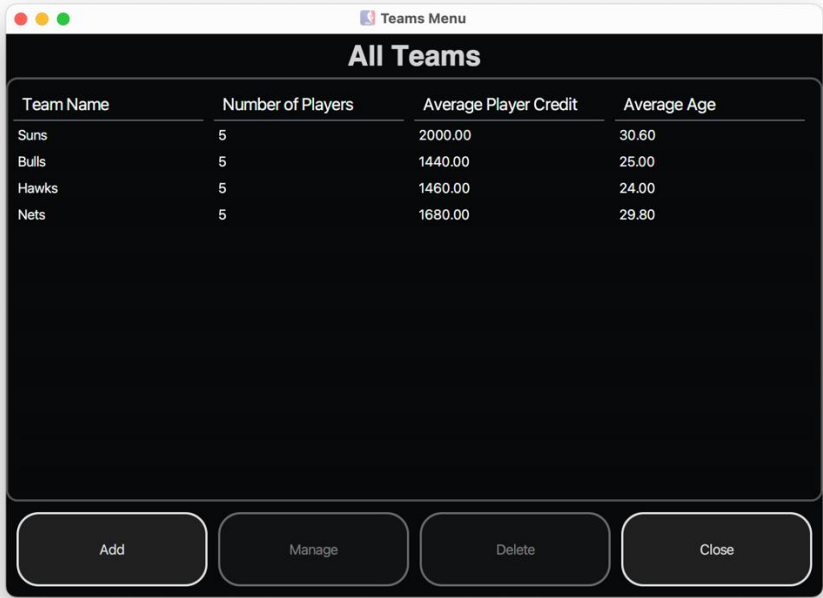
The team menu is launched from the main menu by opening a separate window. It has “Teams Menu” button to access the teams table window ([TeamsTable.fxml](#)) and “ViewPlayers” button to access the players window ([PlayersView.fxml](#)). It has an “Close” button to close the current stage. The team menu also has a header section with a logo “PLAYOFFS”. The title of the window includes an “NBA” icon and the tile “Explore Teams”.



The season menu is launched from the main menu by opening a separate window. It has “Round” button to access the season round window ([SeasonRoundView.fxml](#)) and “Current” button to access the current round teams window ([CurrentRoundTeam.fxml](#)). It has “Game” button to run the tournament and “Result” button to display the record window ([RecordView.fxml](#)). It has an “Close” button to close the current stage. The season menu also has a header section with a logo “PLAYOFFS”. The title of the window includes an “NBA” icon and the tile “Arrange a new season”.

Teams Table Window

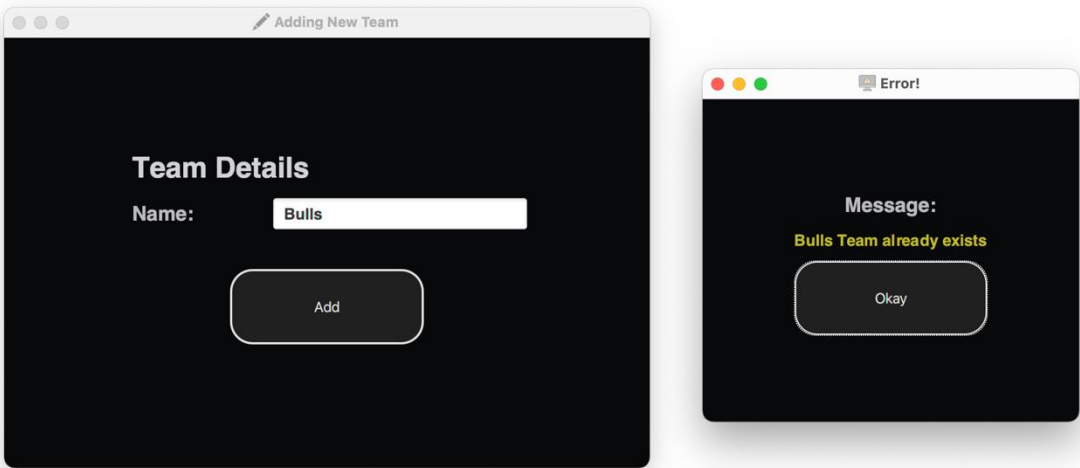
The team table window ([TeamsTable.fxml](#)) is launched from the team menu by opening a separate window. The team table window displays the teams' information by a table with 4 columns "Team Name", "Number of Players", "Average Player Credit" and "Average Age". The team table window has four buttons to add, manage or delete a team. Only "Add" and "Close" buttons are enabled initially.



Team Name	Number of Players	Average Player Credit	Average Age
Suns	5	2000.00	30.60
Bulls	5	1440.00	25.00
Hawks	5	1460.00	24.00
Nets	5	1680.00	29.80

Buttons: Add, Manage, Delete, Close

When click on "Add" button, a new window with title "Adding New Team" ([AddTeam.fxml](#)) is launched. A new team will be added to the team table window after input a new team's name. The new team's average credit and average age are 00.00 as the number of players is 0. If input an existing team's name, an error window will pop up with the message " (*Name*) Team already exists". Details of the Error message window will be explained in the later session.



Adding New Team

Team Details

Name:

Add

Error!

Message:

Bulls Team already exists

Okay

When click on one team record in the table, the selected record is highlighted and the "Manage" and "Delete" buttons are enabled. Meanwhile, "Add" button is disabled. Click the "Delete" button will remove the selected team from the table. The "Manage" and "Delete" buttons will become disabled after the removal.

Teams Menu

All Teams

Team Name	Number of Players	Average Player Credit	Average Age
Suns	5	2000.00	30.60
Bulls	5	1440.00	25.00
Hawks	5	1460.00	24.00
Nets	5	1680.00	29.80
Knicks	0	0.00	0.00

Add

Manage

Delete

Close

The Close button closes the teams table window. The teams table window includes an “NBA” icon. The title of the window is “Teams Menu”.

Manage a Team

The team manage window ([ManageTeamView.fxml](#)) is launched from the “Manage” button in the team table window by opening a separate window. The team manage window displays the teams’ players information by a table with 4 columns “Player Name”, “Player credit”, “Player Age” and “Player No”. The team manage window has four buttons to add, update or delete a player. Only "Add" and "Save and Close" buttons are enabled initially.

Managing Team: Nets

Team Name: Nets

Player Name	Player credit	Player Age	Player No
Ben Simmons	2000.0	26	10
Joe Harris	1200.0	31	12
Mikal Bridges	2400.0	26	1
Patty Mills	900.0	34	8
Seth Curry	1900.0	32	30

Add

Update

Delete

Save and Close



The team's name is displayed in a text field on the top of the window, where the team's name can be updated while clicking "Save and Close" button.

When click on one player record in the table, the selected record is highlighted and the "Update" and "Delete" buttons are enabled. Meanwhile, "Add" button is disabled. Click the "Delete" button will remove the selected player from the table. The "Update" and "Delete" buttons will become disabled after the removal. The "Add" button will become enabled after the removal.

The Save and Close button closes the team manage window. The team manage window includes an "edit" icon. The title of the window is "Managing Team:" plus the team's name.

Player Update Window

The player update window ([PlayerUpdateView.fxml](#)) is launched from the team manage window by clicking "Update" for a player's record. The player update window displays the information of "Player Details" for an existing player.

Updating Player: Joe Harris

Player Details

Player Name:

Player Cred...:

Player Age:

Player No:

Update Add Close

The player update window has buttons to update the player's information and has close button to close the player update window. Only "Update" and "Close" buttons are enabled.

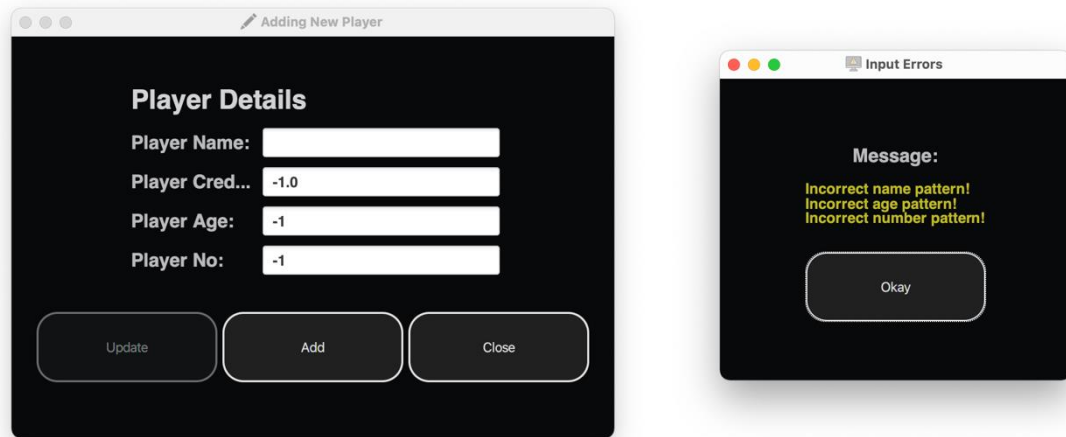
Update button opens the error window when the input is not valid. The button closed the window when the input is valid or there is not input. The input validation is described in validator.java and associated with the error window. Meanwhile, team manage window immediately updates the player's information.

The player update window includes an "edit" icon. The title of the window is "Updating Player: " plus the name of the player.

Add a Player

The player update window ([PlayerUpdateView.fxml](#)) is also launched from the team manage window by clicking "Add". When adding a new player, the window displays the empty name and default values, waiting for the input of "Player Details" for a new added player. The title of the window is "Adding New Player".

The player update window has buttons to add and update player's information and has close button to close the player update window. Only "Add" and "Close" buttons are enabled.



Add button opens the error window when the input is not valid. Add button closed the window when the input is valid. The input validation is described in validator.java and associated with the error window. Meanwhile, team manage window immediately updates the player's information.

Error Window

The error window is popped up when the input is not valid. The error window displays the input error messages based on the input validation described in validator.java.

- Name Pattern: First name and Last name are two strings with capital letters.
- Number Pattern: any positive integer
- Decimal Pattern: any decimal number (- or +)

The error window has "Okay" button to close the window. The title of the window includes an "error" icon and the name is subject to the launched window.

- Adding New Team/ Team Manage window: Error!
- Player Update Window: Input Errors!

Players Window

The players window ([PlayersView.fxml](#)) is launched from the team menu by clicking "View Players". The players window displays all players information in a table view. There are six columns, "Team", "Player Name", "Player credit", "Player Age", "Player No", "Player Level". The table can be sorted by each of the column and the selected players will be highlighted in blue. The default setting is to sort ascended by "Team".

The players window has "Close" button to close the window. The title of the window includes an "NBA" icon, and the name is "Players".

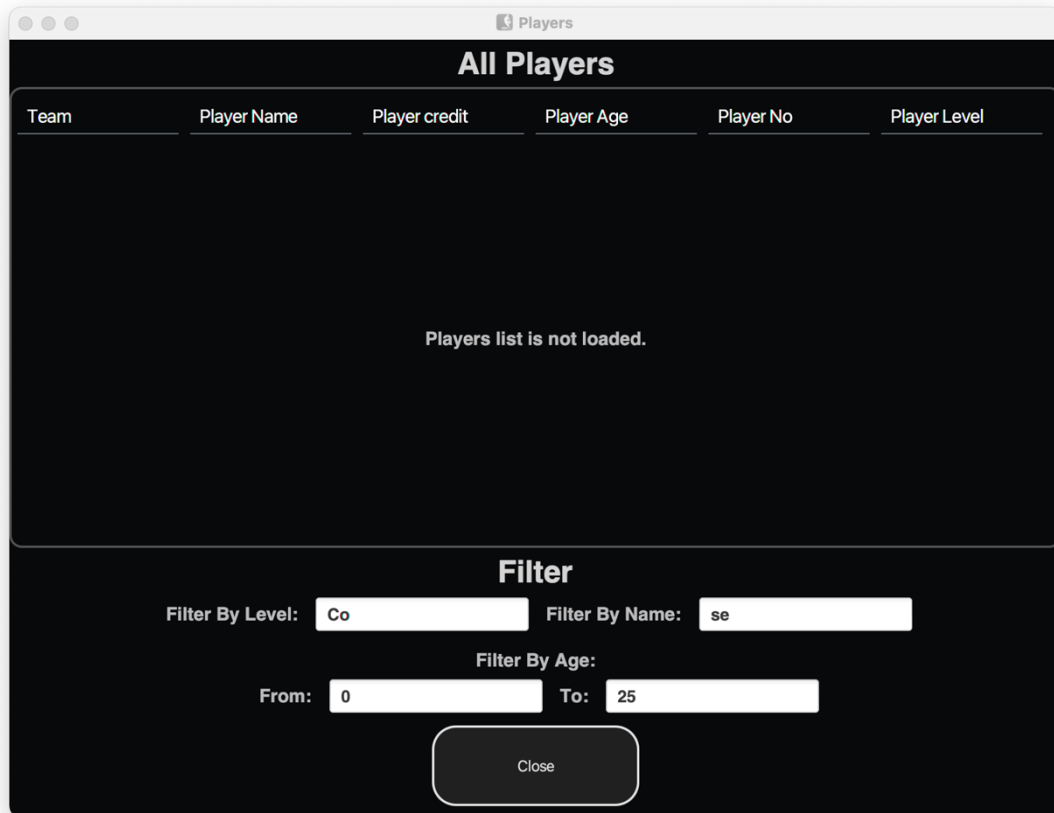
On the bottom of the players window, there are three filters, which display the players with matched name, or level or age. The filters work collaboratively with non-case sensitive partial matching function. For example:



When the Name filter is set with “se”, the players window displays the player list based on the matched records in the level filter result of “Co”.

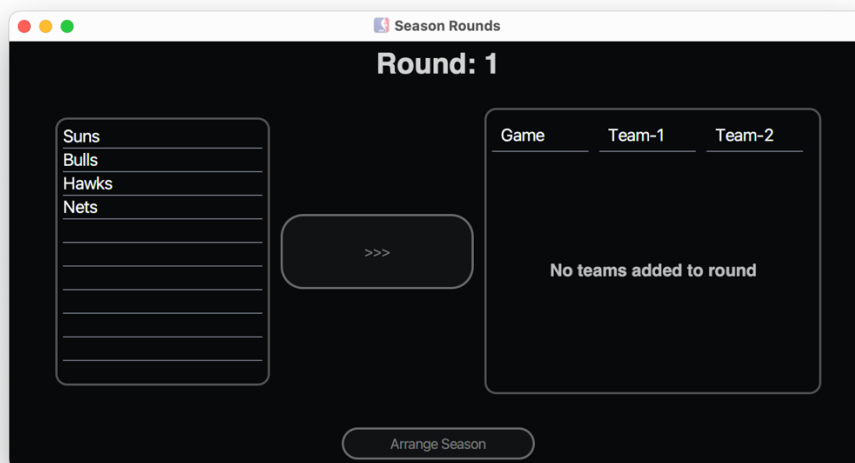


The players window prints "Players list is not loaded" when there is no matches result.



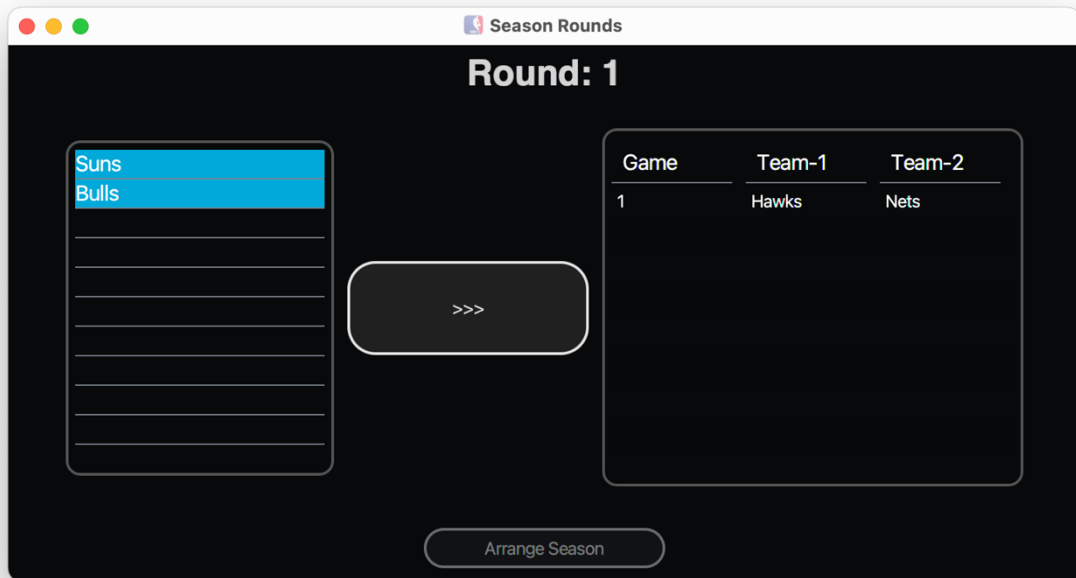
Season Round Window

The season round window ([SeasonRoundView.fxml](#)) is launched from the season menu by clicking "Round". The title of the window includes an "NBA" icon and the name is "Season Rounds". The season round window contains one team list, one game table and two buttons. The initial view is titled with "Round: 1" and has four teams in the left list (Suns, Bulls, Hawks, Nets). The buttons ">>>" and "Arrange Season" are disabled.

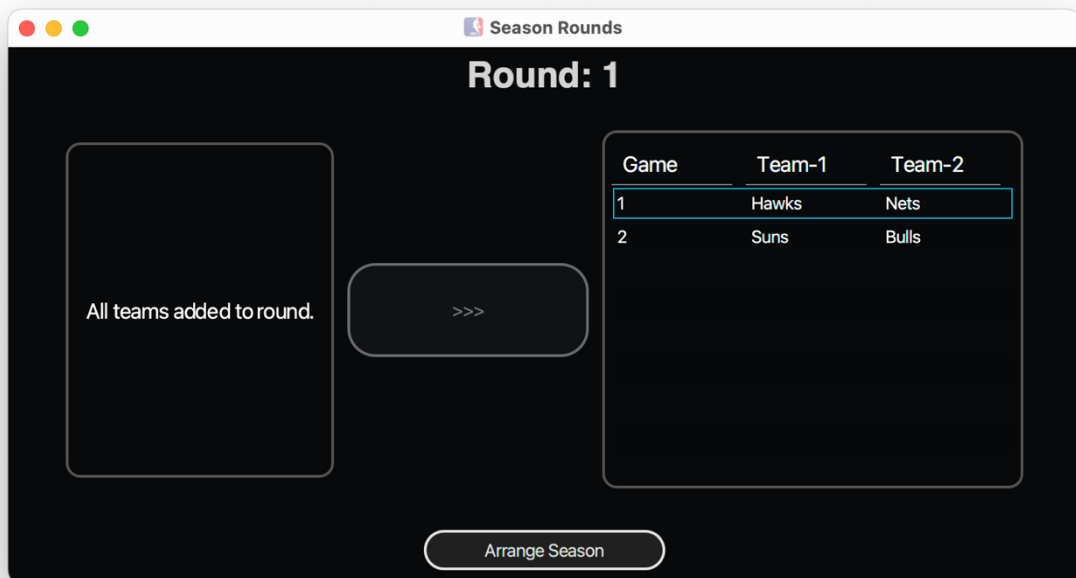




The “>>>” button will be enabled only when two teams are selected in the list. By clicking “>>>”, the two teams will be added to the game table.



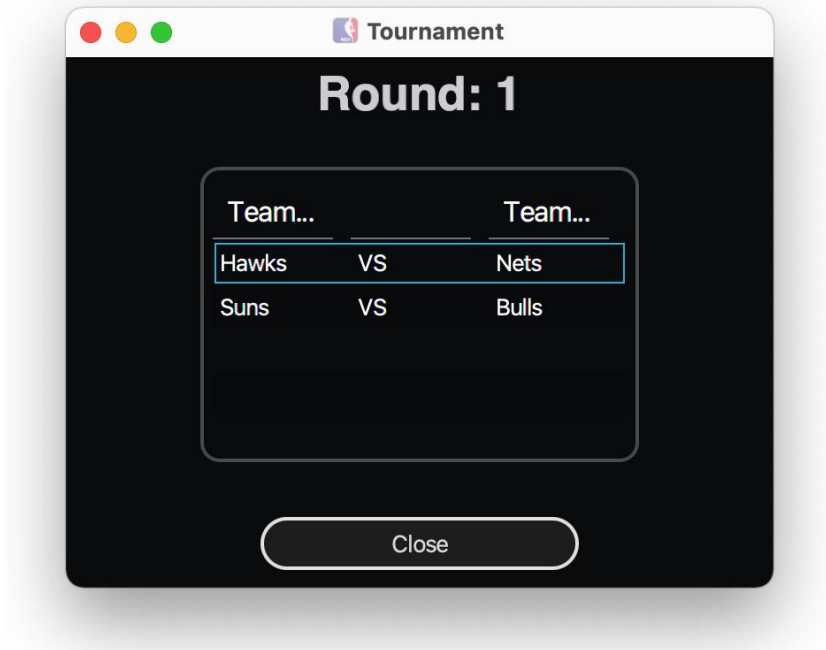
The “Arrange Season” button will be enabled only when all the teams in the list are added to the game table. The empty team list will display “All teams added to round” and the “>>>” button is disabled. Clicking “Arrange Season” confirms the season round teams and closes the season round window.





Current Round Teams Window

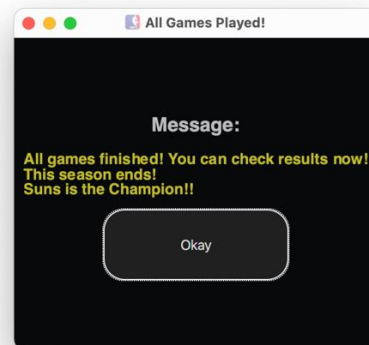
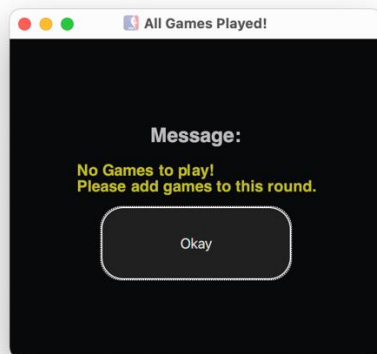
The current round teams window ([CurrentRoundTeams.fxml](#)) is launched from the season menu by clicking "Current". The title of the window includes an "NBA" icon and the name is "Tournament". The current round teams window contains one teams table and one "Close" button. The table view is titled with "Round: 1" and displays the teams' pairs with "VS" in the separation column. The Round number and the teams' pairs will be updated after each round. The "Close" button closes the current window.



Game Window

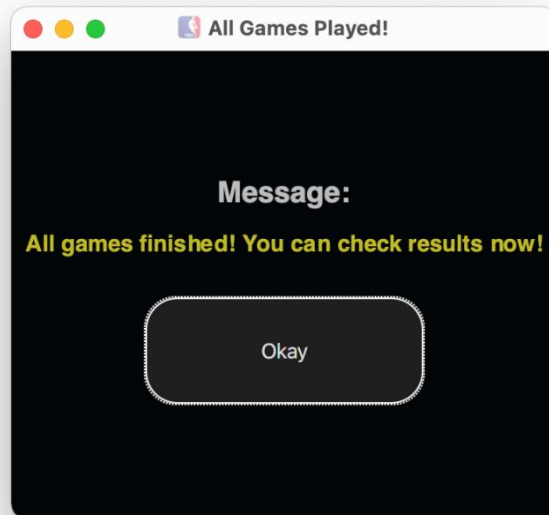
The game window is launched from the season menu by clicking "Game". The game window shares the view of error window ([error.fxml](#)). The title of the window includes an "NBA" icon and the name is "All Games Played!". The "Okay" button closes the game window.

When no teams are added to the round, the game window displays the message "No Games to play! Please add games to this round." When the season completes with all the rounds, the game window displays the message "All games finished! You can check results now! This season ends! Suns is the Champion!!"

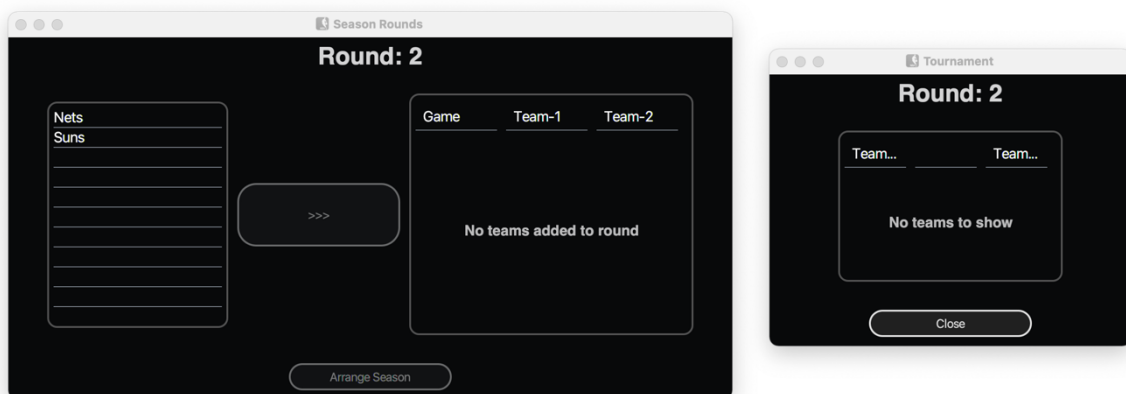




When the season has been arranged with the first round, the game window displays the message “All games finished! You can check results now!”



The round number in season round window and current round teams window will increase by 1. Meanwhile, the winning teams in the first round will be added to the team list in season round window.



Record Window

The record window ([RecordView.fxml](#)) is launched from the season menu by clicking “Result”. The title of the window includes an “NBA” icon and the name is “Season Record”.

The record window contains one result table and one “Close” button. The table view is titled with “Season Record” and displays the game result in each round for each pair. There are four columns in the table “Round”, “Game”, “Winning Team” and “Losing Team”. The result table will be updated each time when “Game” button is clicked. The “Close” button closes the record window.



Round	Game	Winning Team	Losing Team
1	1	Nets	Hawks
1	2	Suns	Bulls

Requirements

Layout

To get full marks, you should layout your windows to look as close as possible to the screenshots. This means that you should try to duplicate the spacing between and around nodes that is shown in the screenshots, and the width and height of the nodes, and the alignment of the nodes.

Style

A CSS file is included in the skeleton code which provides all the styles used in the assignment. You don't need to modify the CSS file.

Code

For JavaFX, your solution must satisfy the following code requirements:

- Your solution must follow the MVC architecture and utilize multiple window display.
- Your solution must keep the package structure and class names that were provided in the skeleton code.
- The models must notify the views of changes by correctly applying the JavaFX property patterns and observable lists. Model data that can change must be observable. Model data that never changes need not be observable.
- The views must be laid out in FXML.

Submission to ED

You will need to submit the source code to ED. The marker will check the source code to make sure all the requirements have been met. Missing the source code will result 10 marks deduction of the assignment 2.

Submission to Canvas

For JavaFX solution, your assignment should be submitted as a JAR file that includes:

- All FXML, CSS and image files required to run your assignment.
- The binary files(.class) of the runnable jar file.

You can check the JAR file on a lab PC to make sure it works. Kindly note that jdk1.8 is the baseline for the assignment. If your work is based on other jdk version, it's your own responsibility to generate a compatible jar that can run on jdk1.8. The incompatible jar file will be marked 0 directly.

For Tkinter solution, your assignment should be submitted as a zip file that includes all the source codes.

The corresponding coversheet stating the reference of the codes and the use of GenAIs should be submitted to Canvas for the assessment. Detailed specification about GenAI clarification refers to the document "GenAI-Intro-Slides-for-students-48024.pdf" in GenAI overview on Canvas.

Demonstration and Extension

You will be marked by the Canvas submission for the features that can be demonstrated to work on a lab PC. Your tutor will ask questions based on your demonstration in the last lab class. Aside from marks for the functionality, the marker will also check your code to ensure that all code requirements have been met. Your final mark will be a combination of marks for functionality, marks for code (See Rubric table on Canvas) and moderation marks for answering questions.

Marking the code and analyzing spoofing, cheating and plagiarism is done in the two weeks following the due date. If you are suspected of Academic Misconduct, I will forward your case to the Misconduct Committee, and you will receive a "high similarity" note on ED submission. Your mark will be finalized within 2 weeks of the due date.

The due date won't be extended as assignment 2 requires a 3-minute demonstration in the last lab class. Submission extension before the last lab class will be granted automatically with a late penalty. An extension CANNOT be given after the due date. For any extension beyond the last lab class, you will need to submit a Special Consideration following the [Special Consideration process](#) for special consideration for reasons including unexpected health, family, work problems, or other extenuating circumstances.

Online support

A FAQs (Frequently Asked Questions) page will be posted as a slide in the Assignment page on Ed. This is not a static page filled with text, it is a question thread where you can post questions for the teaching staff to answer. Your question should strictly relate to the specification, questions regarding code problems or testcase issues should go on the regular discussion board. Any questions that do not adhere to this rule will be deleted. If you have a question, check the FAQ first, it may already have been answered there. Anything posted to the FAQ is considered to be part of the assignment specification.

As for "normal" assignment questions and assignment help, the preferred way to ask is through participating in the lectures, lab activities, UPASS and consultation sessions.

The teacher may be contacted by email if you have matters of a personal nature to discuss, e.g., illness, personal issues or other matters of importance. All emails sent to the head teacher, tutors or lecturers must have a clear subject line that states the subject number followed by the subject of the email. e.g.: [Subject 48024, Appointment Request], and must be sent from your UTS email address.

Marking Scheme

The marking scheme for the assignment is posted as Rubric Table on Canvas. Note that individual tests may test several functionality components, and a functionality component may be tested by several tests.