CMSC203 Assignment 5

Fall 2015

Concepts tested by this program

Static variable and methods

Multiple classes (Enumeration/Data Element/Data Structure/Data Manager/JavaFX driver)

Class relationationships – Dependancy and Aggregation

Enumerated Types

ArrayLists

JUnit Tests

GUI:

Displaying Images

TextAreas

Panes and Component Layout

Borders (bordered radio group) use .setStyle

Using the .setStyle for font size and color

Deliverables:

Java files - The src folder with your driver (javafx application), enumerated class, data element, data manager and Junit Test (.java) files

Javadoc files - The doc folder with your javadoc for student generated files

UML Class Diagram (an image, not the proprietary format, must be a .jpg)

Deliverable format: The above deliverables will be packaged as follows. Two compressed files in the following formats:

LastNameFirstName\_AssignmentX\_Complete.zip [a compressed file containing the following]

UML.jpg

Assignment 5 Checklist (filled in with YES or NO or ?)

doc [a directory] *please include the entire doc folder with the javadoc for student*

*generated files*

file1.html (example)

file2.html (example)

src [a directory] *contains your driver (javafx application), enumerated class, data*

*element, data manager and Junit Test (.java) files*

File1.java (example)

File2.java (example)

File\_Test.java (example)

LastNameFirstName\_AssignmentX\_Moss.zip [a compressed file containing only the following]

*contains .java file which includes the driver (javafx application), enumerated*

*class, data element, data manager and Junit Test (.java) files – NO FOLDERS!!*

File1.java (example)

File2.java (example)

**BASKETBALL TEAM ROSTER GENERATOR**

At the start of a basketball game, each coach writes out their players’ names on the scoresheet and gives it to the score-keeper. When a player enters the game, the scorekeeper must keep track of who is in the game. Your task is to write a GUI-driven program to enter the starting roster and positions of a basketball team. Each team will include 2 forwards, 1 center and 2 guards.

Notes:

1. Your program ONLY records the starting team, not the entire team.
2. Your program will also note each player’s position (although actual scorekeepers do not do this).
3. Your program will not be required to keep track of the teams or players when the program goes on to the next team (the program would need to be extended to be useful to actual scorekeepers).
4. Your program will be GUI-driven, not command-line.
5. Your program will display an image of a basketball on the GUI.

## Operation

The user will start the application by selecting New Team. The user will then type in the Team’s name in the Team Name textbox.

The user will enter the first and last name of the team member and then indicate which position the member will play. The user then selects the Add Player button.

If the position selected is already filled (1 center, or 2 forwards or guards), an error message is displayed.

When the user selects the Print Team button, the names of the team will be displayed with their position in the Team Roster textArea.

When the user selects the New Team button, the team name field becomes available for user to type in team name.

When the user closes the frame or presses the Exit button, the application exits.

## Specifications

**Enumerated Class - Position**

Create an enumerated type called Position. The valid values will be “GUARD”, “CENTER”, “FORWARD”.

**Data Element – Player class**

Create a Player class. It will contain instance variables of type String for the first name and last name. It will contain an instance variable of type Position for the position. Provide a constructor and a toString method. Add any additional methods as required.

**Data Manager – Team class**

Create a Team class. It will contain an ArrayList of Player Objects. It will have a static variable that keeps track of the number of players that have been added to the team and a static method that returns the value of this variable. Provide a constructor and a method to addPlayer, a method to printTeam and a toString method. The addPlayer method will check that there is only 1 center, 2 forwards and 2 guards at a time and will check that there are no more than 5 members of the team. The addPlayer method will return a String indicating what the error is (i.e. “There is already a center on this team”) or a null if the player was added. **Make sure your class names and method names and parameters match the Javadoc that I provided for this class, or the JUnit test will not run correctly.**

**GUI driver – JavaFX**

Provide events/event handling for the following buttons:

* addPlayer – extract first and last name from textboxes, determine which position, call the addPlayer method from the Team class
* printTeam – call the printTeam method of the Team class and display results in the text area
* clear – clear the first and last name and any error messages, do not clear the team name, don’t change the position
* newTeam – make the teamName textbox editable. Clear the team name, first name and last name fields and the text area for the team roster. When the user types in the team name and hits enter, the teamName textbox becomes uneditable again and if the addPlayer, clear and printTeam buttons are disabled, enable them at this time (see event handling for teamName field).
* exit – exit the application

Provide event handling for the following textboxes:

* teamName – Extract the team name entered by user, create a new Team object with this team name, and make the teamName textbox uneditable. If the addPlayer, clear and printTeam buttons are disabled, enable them at this time. (Be aware that the user must press Enter after typing in the team name, for the event to be triggered)

Use Panes and Component Layout to organize your GUI (VBox and HBox, etc)

The teamName textbox is initially set to uneditable (.setEditable(false))

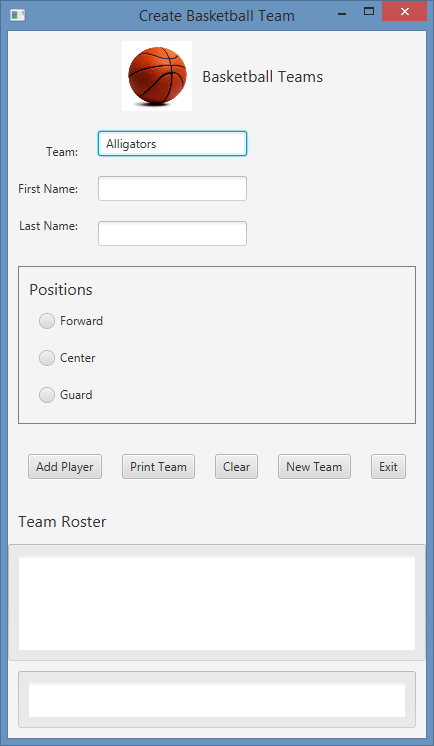
Use a border around the radio buttons box for Position (use .setStyle)

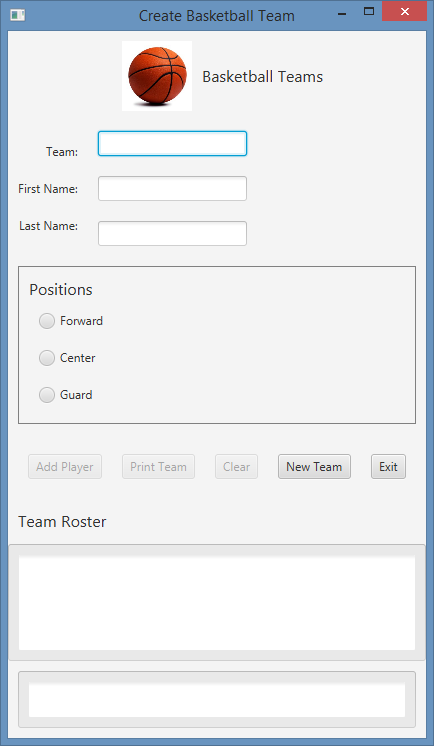
Display the basketball image at top of the panel.

Use an instance variable of type Team. Use the addPlayer and printTeam methods of the Team class.

Display an error message at the bottom of the screen when there is an error.

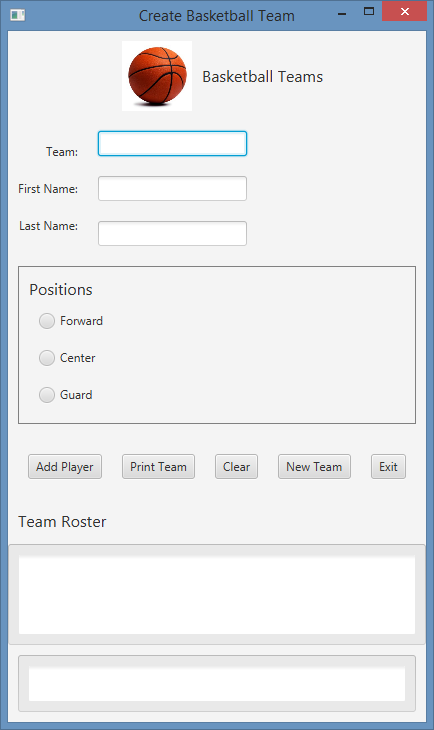
If the New Team button is selected, make the teamName textbox editable, after the user types in the team name, make the teamName textbox uneditable again.





Once the Team name has been typed in and user hits enter, the other buttons are enabled (.setDisabled(false))

Notice when the program begins, only the New Team and Exit buttons are available, all the other buttons have been disabled (.setDisabled(true))



Print Error messages here

Print Team Roster here

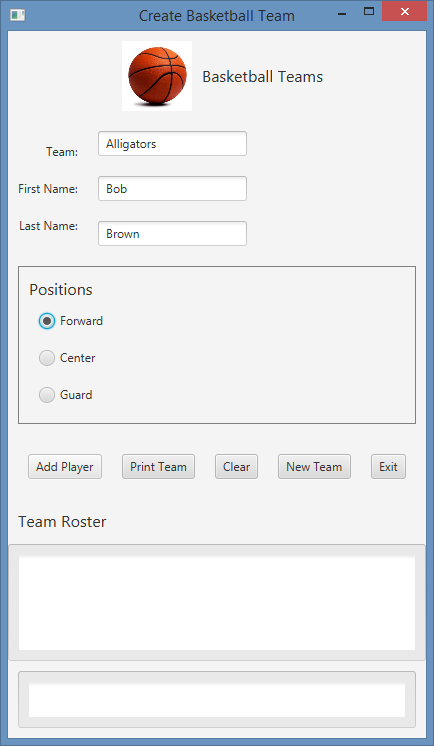
Select to add this player at this position

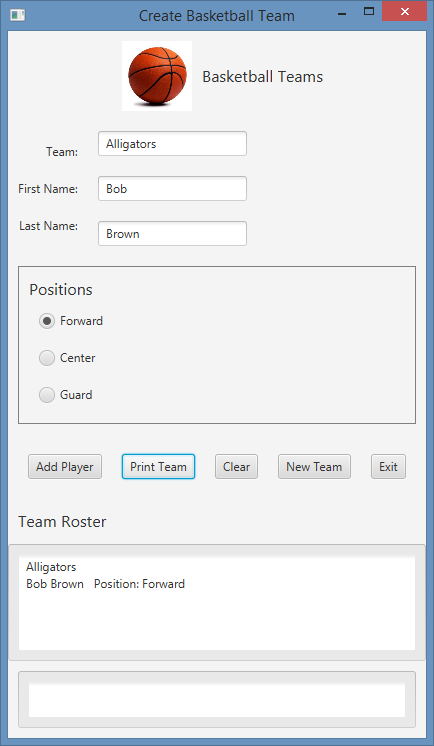
Team name

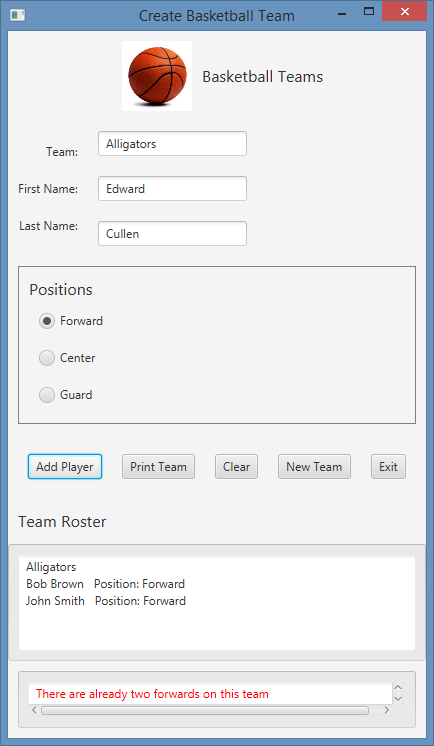
Player position

Player name

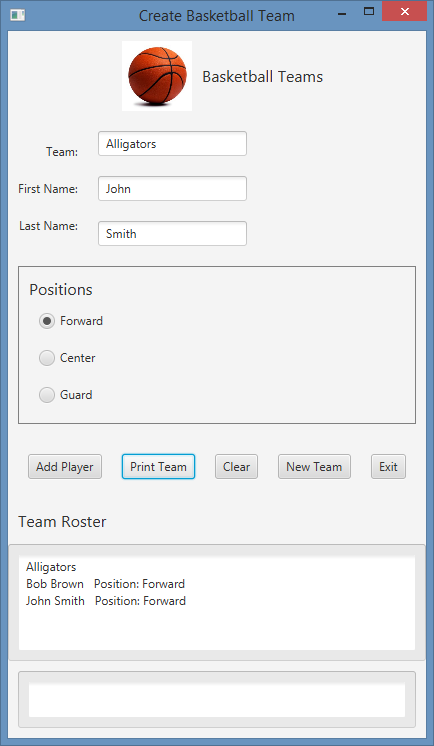
Example of adding a player







Add second player and Print Team



Error message

tried to add 3rd forward

## Documentation

Javadoc:

A set of html documents generated using Javadoc tags.

Comments in source code:

In addition to the documentation comments which are used by javadoc, include in-line comments as needed to further explain your code

## Testing

Your code should compile and run without errors.

You should test your GUI by manually selecting various sets of buttons and text entries.

JUnit Tests:

Check your program by running the JUnit tests found in BasketballTeamTest.java

Add STUDENT test(s) for the Team class in BasketballTeamTest.java.

Create a JUnit test for your Player class.

## Include your updated version of BasketballTeamTest.java in your submission.

**Program Grade Sheet Assignment #5**

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Submittal Date/Time ­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DOCUMENTATION Total 25 points

CheckList for Assignment 5 is included and completed 1 pt \_\_\_\_\_

Class Documentation – Javadoc for all student generated classes 10 pt \_\_\_\_

Class description, @author

Method description, @param, @return

JUnit Test Classes 5 pt \_\_\_\_\_

Implement the Student test methods of BasketballTeamTest

Create a test for the Player class

UML Diagram 4 pt \_\_\_\_\_

Lessons Learned 5 pt \_\_\_\_\_

In 3+ paragraphs, highlight your lessons learned and learning experience from working on this project. How did you do? What have you learned? What did you struggle with? How will you approach your next project differently?

PROGRAMMING Total 75 points

Internal class Javadoc documentation (within source code) 5 pts \_\_\_\_\_

Description of what each class does

Author’s Name, @author

Methods commented properly using Javadoc documents

Description

@param, @return

Compiles and Runs without runtime errors or warnings 10 pts \_\_\_\_\_

Accuracy

Public tests – JUnit test given to you 6 pts \_\_\_\_\_

Your JUnit Tests – added to BasketballTeamTest and PlayerTest 4 pts \_\_\_\_\_

Private tests – other tests run by the instructor 10 pts \_\_\_\_\_

Program Details

Data Element – Player class 7 pts \_\_\_\_\_

1. Contains instance variable of type Position
2. Contains constructor and toString methods
3. Contains getters and setters for instance variables

Data Manager – Team Class (use javadoc provided) 12 pts \_\_\_\_\_

1. Contains at least one static variable and static method
2. Contains methods to addPlayer, printTeam, etc
3. Contains an ArrayList of Player object references

Enumerated Type – Position 5 pts \_\_\_\_\_

1. Enumerated type Position is created with enumerated values

GUI – JavaFX application 10 pts \_\_\_\_\_

1. Provide events/event handling for the following buttons:

addPlayer, printTeam, clear, newTeam, exit

1. Provide event handling for the following textboxes: teamName
2. Use Panes and Component Layout to organize your GUI (VBox and HBox, etc)
3. The teamName textbox is initially set to uneditable (.setEditable(false))
4. Use a border around the radio buttons box for Position (use .setStyle)
5. Display the basketball image at top of the panel.
6. Use an instance variable of type Team. Use the addPlayer and printTeam

methods of the Team class.

1. Display an error message at the bottom of the screen when there is an error.
2. If the New Team button is selected, make the teamName textbox editable,

after the user types in the team name, make the teamName textbox uneditable again.

1. Program user interface 3 pt \_\_\_\_\_

Clear to user how data is to be entered

Output is easy to understand

1. Data Validations/Output formats 3 pt \_\_\_

Program should only accept valid inputs

invalid data must be detected, warn user and then ignored.

Program should not crash due to bad inputs.

Total 100 pts \_\_\_\_\_

|  |
| --- |
| Player |
| - firstName : String  - lastName : String  - playerPosition: Position |
| + Player (first, last : String, pos: Position)  + Player (info : Customer)  + getFirstName () : String  + getLastName () : String  + getPosition() : String  + toString () : String |

|  |
| --- |
| Team |
| - teamName : String  - numPlayers : int  - numCenters : int  - numForwards: int  - numGuards: int  - currentPlayer: ArrayList<Player> |
| + Team (tm : String)  + addPlayer(fName, lName, pos : String)  + printTeam () : String  + toString () : String |

Diamond pointing up

|  |
| --- |
| <enumeration>  Position |
| GUARD  FORWARD  CENTER |
| + Position (pos : String)  + getName() : String  + toString () : String |