COMS 362 Project

Iteration 1: Controllers

Overview

The client wants documentation of the controllers (GameController, MatchController and PlayController). This will help in evaluating the impact of change when adding new features to the game selection and setup processes. For example, what will need to change to support team based games? What about games that have distinct roles (e.g., dealer)? How much code needs to be written to support multiple rule variations of games that players can select from? How would we support a game like poker where individuals can leave and join the table in-between rounds?

Diagram Deliverable Artifacts

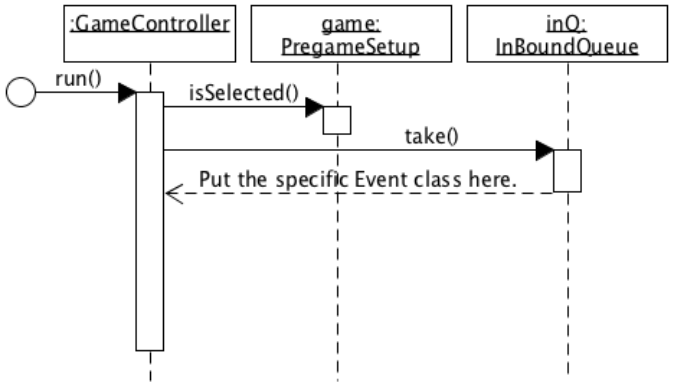
The first set of deliverables are:

1. GameController class diagram
2. MatchController class diagram
3. PlayController class diagram
4. GameController sequence diagram
5. MatchController sequence diagram
6. PlayController sequence diagram

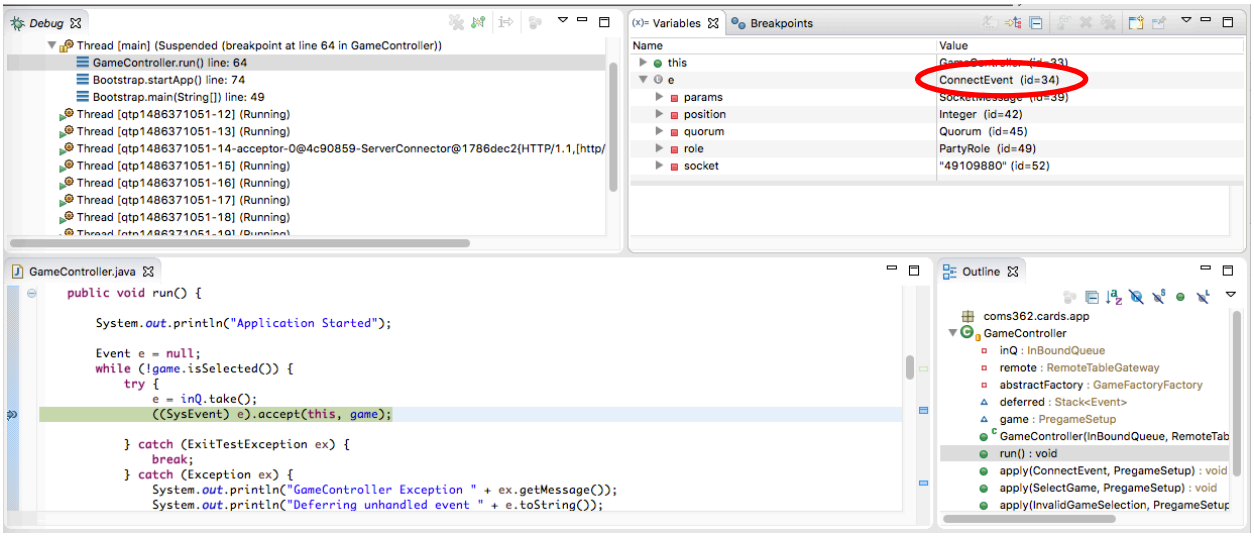
Each of these is produced by one team member, teams of 5 only need to produce 5 of the above deliverables. The details for the diagrams follow.

Create three class diagrams, one for each of the controllers. The diagrams should be **fully dressed** (e.g., attributes, methods, parameters, return types, visibility, dependencies, associations, compositions, multiplicity, etc.). Include the controller class and every class/interface on which it is directly dependent (e.g., InBoundQueue, Event, Rules, etc.) Do not include any Java library classes (e.g., Stack).

Create three sequence diagrams, one for each of the controllers - GameController, MatchController and PlayController - starting with the run( ), start( ) or play( ) as the founding method call respectively. Each diagram should include an instance of the controller and every object (except for Java library objects, e.g., System.out) the controller **directly** interacts with in its run( ), start( ) or play( ) method. Do not use a loop interaction frame. The purpose of the diagrams are to show **a specific scenario**. In the case of GameController or MatchController show the game or match being started. For PlayContoller, show a condensed version of the events during the play of a game. The following diagram show the start for GameController, replace the message “Put the specific Event class here” with the type of event that is returned and then continue the diagram from there.



It may not be easy to know the sequence of events in a typical game just from looking at the code. It is highly recommended to set a break point in the run( ), start( ) or play( ) method and then execute the application in debug mode. The red oval in the diagram below is highlighting that the first event is a ConnectEvent, now we know what inQ has returned.



Report Deliverable

As a team perform an impact analysis on one of the following changes:

- Support team based games.
- Support games that cannot begin until distinct roles have been assigned (e.g., dealer, banker, facilitator, etc.).
- Support the host being able to select from multiple rule variations of a game.
- Support a game like poker where individuals can leave and join the table in between rounds.
- Devise your own requirement that involves the setup of the game (with approval from you coach).

Generate a report of your impact analysis. The report should contain all relevant class and sequence diagrams. Any extension, change or noteworthy impact should be clearly marked in red. Write a couple of short paragraphs explaining the impact.

Dates

First commit of diagrams due: Sunday, March 28

Final commit of diagrams and report due: Sunday, April 4

Individual Responsibilities (50 points)

- Create one of the diagrams and push it to your team's project fork.
- (5 points) The first draft must be pushed by the end of March 28. The purpose of the first draft is to engage with your team and coach. You will be expected to use your diagram, to explain to your coach, how the controller works.
- (45 points) The final draft is due April 4 and must also be pushed at that time.

Team Responsibilities (50 points)

- Generate the report and push it to your repository by April 4.
- Work together with your team, one incomplete diagram is not an excuse for the report being unfinished.