Short description of game:

Yu-Gi-Oh! is a card game in which two players attempt to defeat each other by decreasing their opponent's Life Points (down to 0) using a collection of monster, spell, and trap cards.

UML diagrams:

Use cases diagram:

Use cases:

use case : Play game

Quick description: Playing game with opponent

Actors: Two players

Basic stream:

1. The player selected play game button

2. Choose which player will play first move

3. First player draw card

   3.1 Put new card into hand

4. Decide to activate spell/trap card if possible

5. Summon new card from hand, put it on the table

6. Attack opponent if player decided to

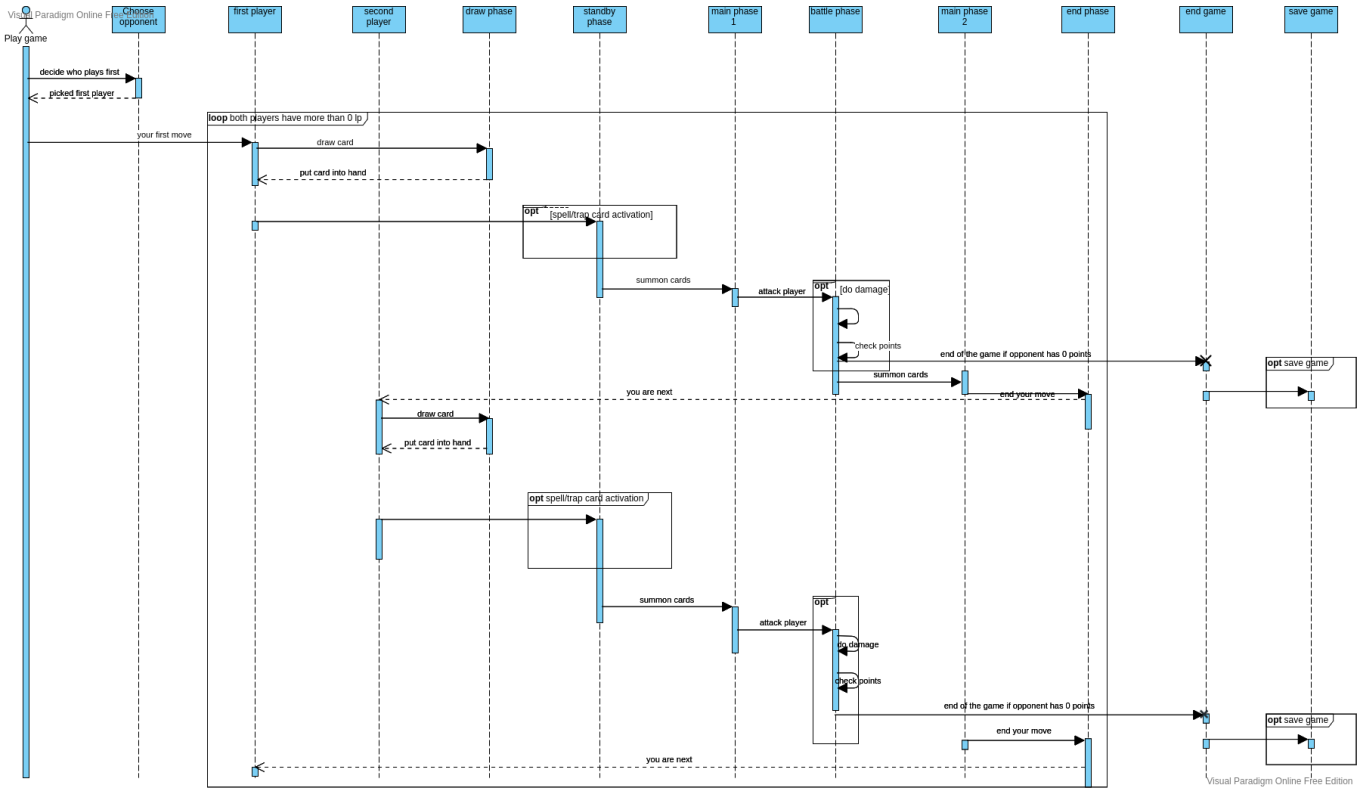
   6.1 If opponet has once or more card on the table, player has to attack that cards first
   6.2 If opponent doesn't have any card on the table, player attack him and do damage as same as card has
   6.2.1 After attack, if opponent doesn't have any lp, game immediately ends, player wins. Next step is 10

7. Summon new card if played want to

8. Finish his move, opponent is next player

9. New current player do same steps from 1-8.

10. Save game if player wants that



use case : Game settings

Quick description: The player changes the default number of initial: life points, the duration of the move and the number of cards in the hand.

Actors: One player

Main course of events:

1. The player chooses the "Game settings" button from the main menu

2. The application opens game settings options

3. If the player chooses the "Set life points" button

3.1. The application displays combobox of offered points, as well as an additional field where the player can enter an arbitrary number of life points (in the appropriate domain).

3.2. The player enters or chooses the points (if he wants to change the default ones)

3.2.1. The number of life points must be a positive integer from the domain [1000,15000] or the changes will not be accepted with a value out of range error sent

3.3. The application temporarily remembers changes to initial life point (if any)

4.If the player chooses the "Set time per move" button

4.1. The application displays combobox with the offered times in minutes, as well as an additional field where the player can enter an arbitrary time in minutes (in the appropriate domain).

4.2. The player enters or chooses a new move duration (if he wants to change the default ones)

4.2.1. The duration of the move must be a positive integer from the domain [3,100], or the changes will not be accepted with a value out of range error sent

4.3. The application temporarily remembers changes to time per move (if any)

5.If the player chooses the "Set initial number of cards" button

5.1. The application displays a list with the offered numbers, as well as an additional field where the player can enter an arbitrary number of cards (in the appropriate domain).

5.2. The player enters or chooses a new initial number of cards (if he wants to change the default ones)

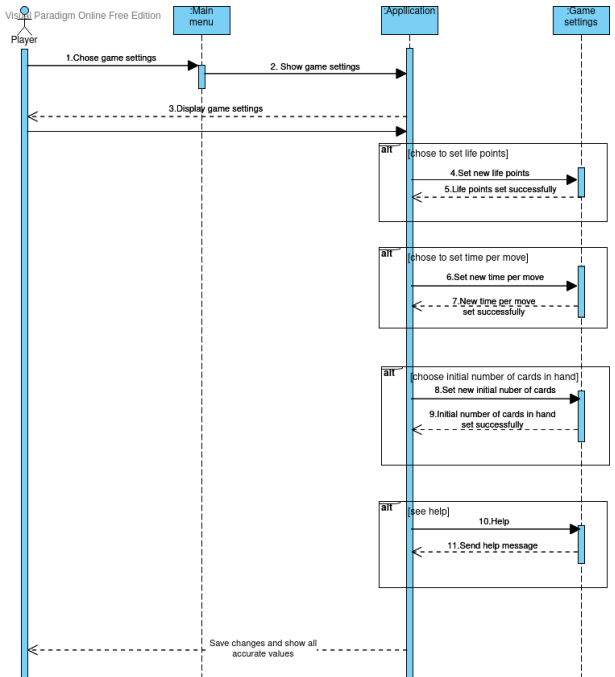
5.2.1. The number of the cards must be a positive integer from the domain [3,100], or the changes will not be accepted with a value out of range error sent

5.3. The application temporarily remembers changes to initial number of cards (if any)

6.If the player selects the "Get help" button

6.1 The application displays a brief description of the game so that the player knows how best to set life points, time per move and initial number of cards in hand.

7. Permanently save all changes for the given player, which will be ready for the game.



use case : Profile settings

Quick description: the player changes their avatar and card sleeve

Actors: One player

Basic stream:

1. The player with a click of the "Profile settings" button opens the profile settings

2. The application opens the profile settings

3.1 The player chooses between the offered avatars

3.2 The player chooses between the offered card sleeves

4. Application goes back to the main menu

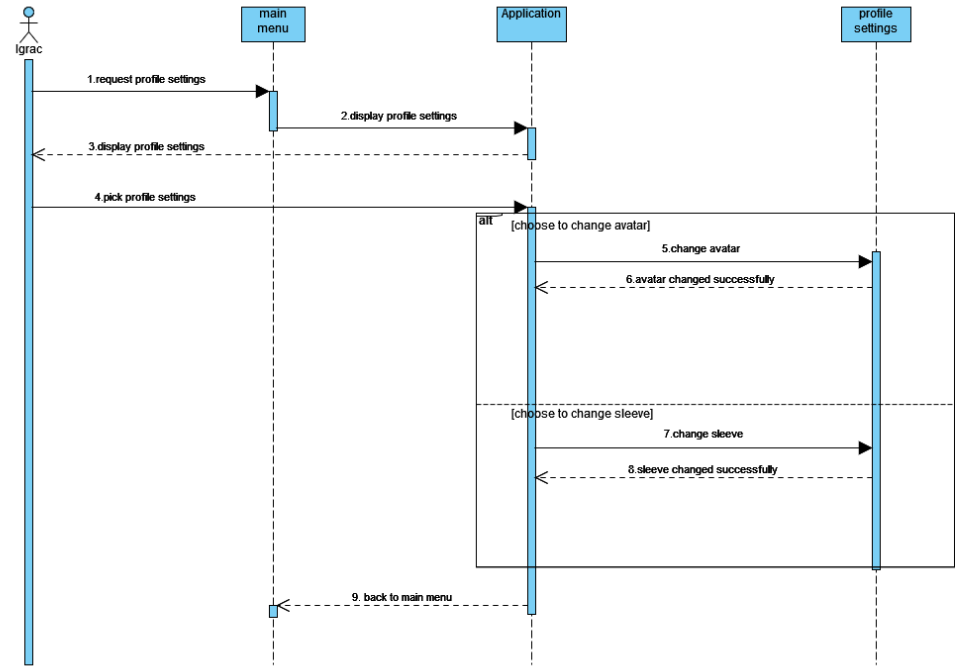
Alternate streams:

1. Unexpected shut down of the application. If the application at any part of the procedure turns off, the settings will not be saved

Substreams:
/

Special demands:
/

Additional information:
/



use case : Deck editor

Quick description:The player adds new decks, deletes the existing ones or renames decks.

Actors: One player

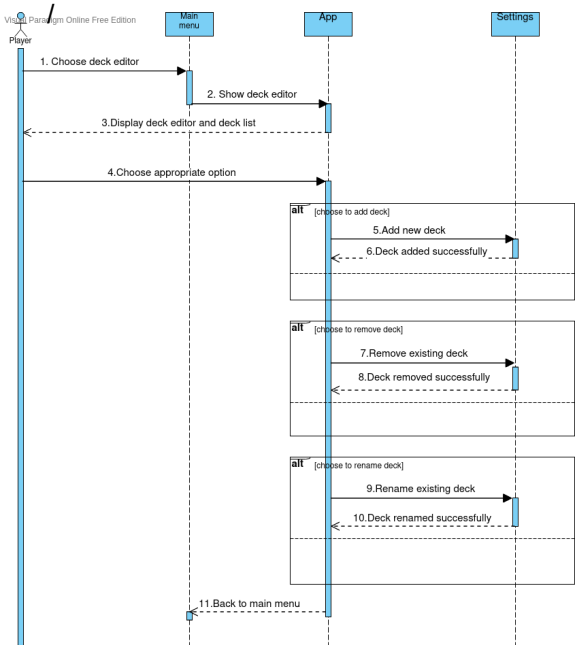
Main course of events:

1. The player chooses the "Deck editor" button from main menu
2. The application opens the deck editor menu
3. The player chooses the "Add deck" button
   3.1. The application shows a list of available cards
   3.2. The player chooses cards to add to a new deck.
      3.2.1. Deck must have minimum 40 cards
   3.3. The player enters the name for a deck
   3.4. The application saves the deck with chosen cards and name
4. The player chooses the "Remove deck" button
   4.1. The application shows a list of existing decks
      4.1.1. If there is no decks, application alerts the player that there is no existing decks
   4.2. The player chooses what deck to remove
   4.3. The application removes chosen deck
5. The player chooses the "Rename deck" button
   5.1. The application shows a list of existing decks
      5.1.1. If there is no decks, application alerts the player that there is no existing decks
   5.2. The player chooses deck to be renamed
   5.3. The player enters the name
   5.4. The application renames the chosen deck with the entered name
6. Application goes back to the main menu

Alternate streams:

1. Unexpected shut down of the application. If the application at any part of the procedure turns off, the settings will not be saved

Substreams:
/
Special demands:
/
Additional information:
/



use case : Load game
Quick description: The player loads the existing savefile from the main menu.

Actors: One player

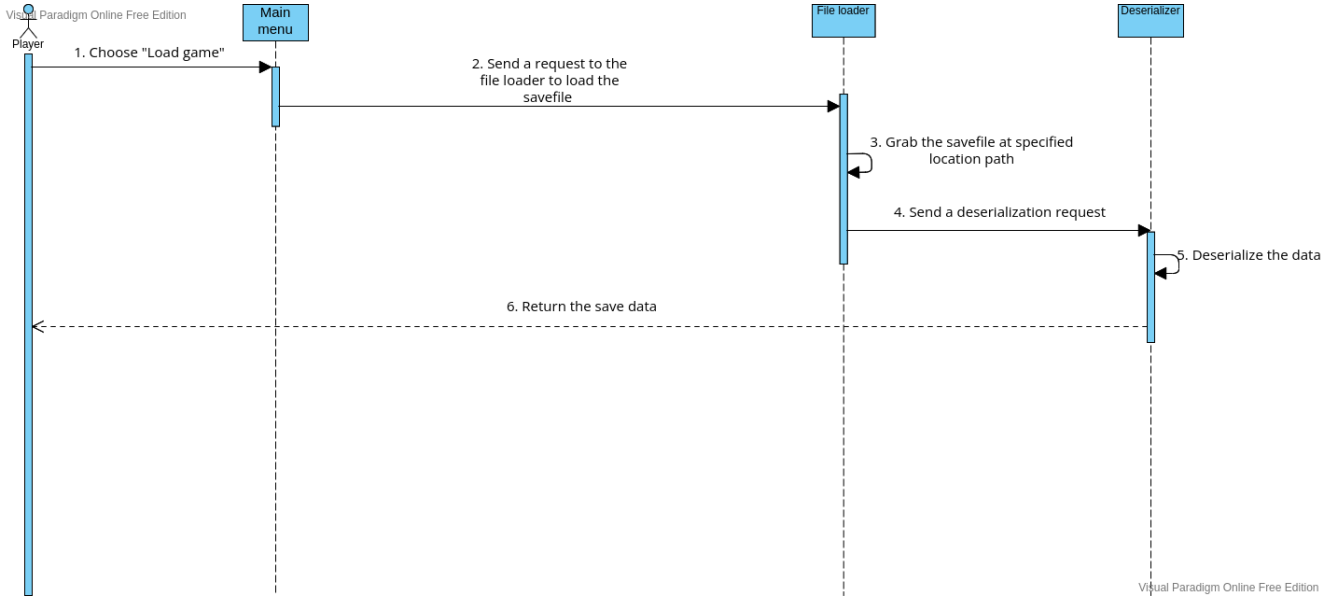
Main course of events:

1. The player chooses "Load game" in the main menu
2. The application sends a request to the file loader for savefile to be loaded
3. The file loader searches for the file
   3.1 When the file is located, it is returned to the file loader
   3.2 The deserialization request is sent by the file loader to the deserializer
4. The deserializer accepts deserialization request
   4.1 The data goes through the deserialization process
5. The data is sent back and the game can continue from that state.

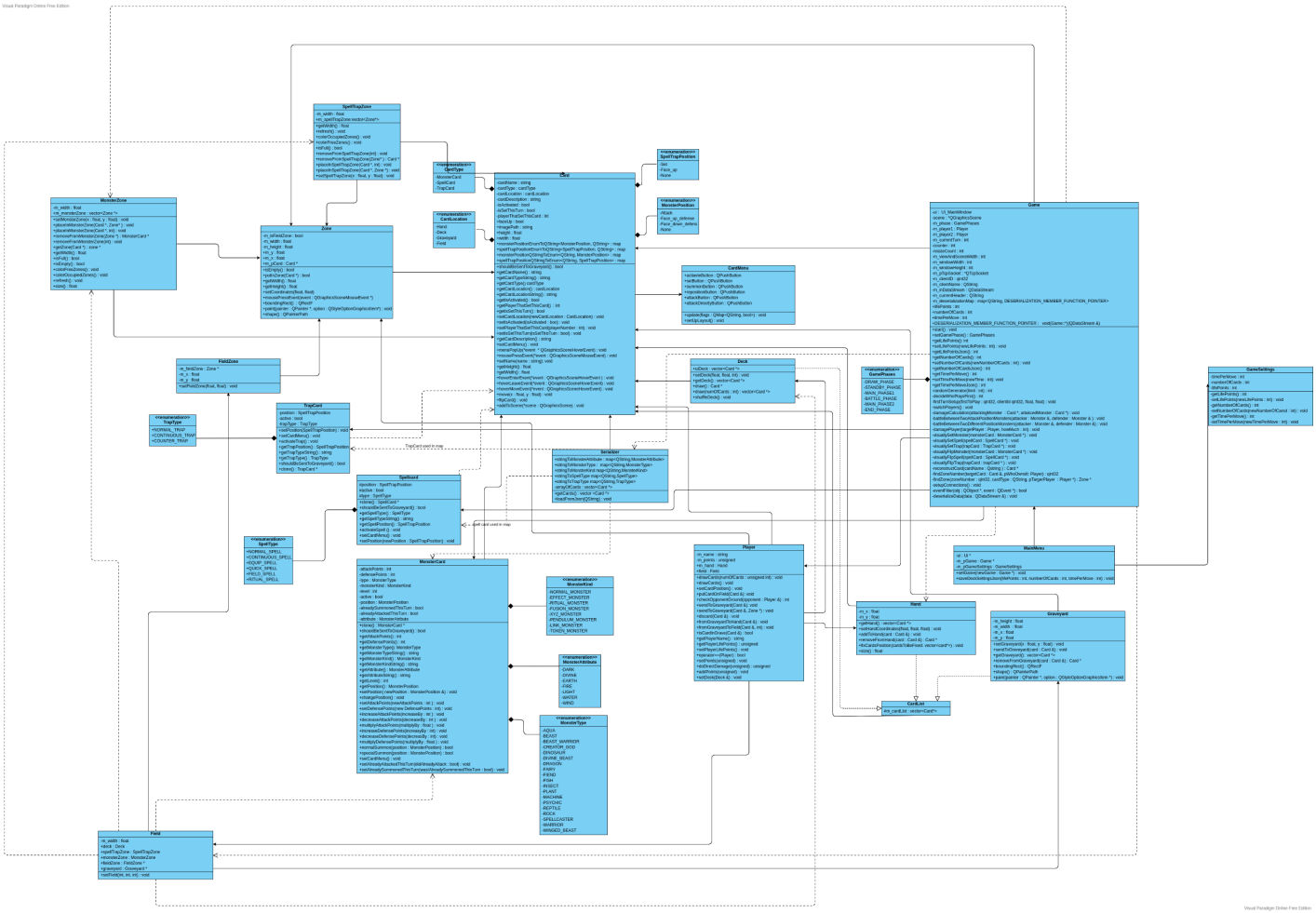
Alternate course of events:
/
Subcourses of events:
/
Special demands:

Savefile has to exist on the disk.

Additional information:
/



Class diagram:



Coponents diagram: