**SOEN 6441**

**Advanced Programming Practices**

**Summer 2015**

**Software Architecture**

“[Lanterns: The Harvest Festival](http://foxtrotgames.com/lanterns/)”

**{**Build-1, Build-2, **Build-3}**

(Team-D)

Khan, Muhammad Raza

Momo Vofo, Patrick

Muco, Brice

Patil, Sushil

Puranik, Tejas

Ragunathan, Vinod Kumar

Contents

[I) Introduction 3](#_Toc427247949)

[II) Description and Functionalities of the modules in the class diagram 4](#_Toc427247950)

[III) Relationship between classes 7](#_Toc427247951)

[IV) Reasoning of choosing the corresponding data structures 8](#_Toc427247952)

# Introduction

This document provides a brief architectural overview of the system. The module view of “[Lanterns: The Harvest Festival](http://foxtrotgames.com/lanterns/)” is explained in detail using class diagram. The description covers the features implemented in the build 3. The build 1 and build 2 implementation detail documents are attached in Fig 2 and Fig 3 respectively.



Fig 1: Class Diagram of “[Lanterns: The Harvest Festival](http://foxtrotgames.com/lanterns/)” application



Fig 2: Implementation details of Build 1



Fig 3: Implementation details of Build 2

# Description and Functionalities of the modules in the class diagram

**Greedy Player**

Description:

The player is assigned a strategy at the start of the game. The player can be a Greedy Player i.e. he or she always makes the move which will bring the best immediate return. The player will choose the lake tile to be placed on the board in such a way that it will helps him to make a dedication as earlier as possible.

Functionality:

It performs the exchange of lantern cards and determines whether the exchange is successful or not.

It performs greedy dedication for computer player.

The logic of deciding which lake tile to be placed on the board that helps makes dedication.

It performs a check on lantern cards that should be exchanged to perform dedication of Seven Unique.

It performs a check on lantern cards that should be exchanged to perform dedication of Four of Kind.

It performs a check on lantern cards that should be exchanged to perform dedication of three pair.

It performs a check on Four of Kind Dedication whether it is possible for a greedy Player to dedicate and does it if possible.

It performs a check on Three Pair Dedication whether it is possible for a greedy Player to dedicate and does it if possible.

It performs a check on Seven Unique Dedication whether it is possible for a greedy Player to dedicate and does it if possible.

The functionality to simulate the dedication event.

**Human Player**

**Description**

Human player is one of the five player types included in the strategy pattern.

**Functionality**

Player is provided an option to choose either one of the following

1. Exchange lantern cards
2. Make a dedication
3. Place a lake tile

Perform the exchange of lantern cards based on availability of sufficient favor tokens, lantern card that would be returned and availability of specific lantern card that player choose to pick in the process of exchange.

Perform the dedication based on the type i.e. “Three pair” or “four of kind” or “seven unique” of dedication that player chooses.

Based on the choice of dedication the stack of global and player stacks is incremented and decremented respectively.

Perform Lake Tile placement on the board.

**Random Player**

**Description**

The random player performs the operation such as exchange of lantern cards, make a dedication, or placement of Lake Tile at random.

**Functionality**

Player is provided an option to choose either one of the following

1. Exchange lantern cards
2. Make a dedication
3. Place a lake tile

The option is determined using random number.

Perform the exchange of lantern cards based on availability of sufficient favor tokens, lantern card that would be returned and availability of specific lantern card that random player choose to pick in the process of exchange. The return and pick of specific color lantern card is determined randomly using random number generator.

Perform the dedication based on the type i.e. “Three pair” or “four of kind” or “seven unique” of dedication that random player chooses. The choice of the type is determined using random number generator.

Based on the choice of dedication the stack of global and player stacks is incremented and decremented respectively.

The lake tile placement i.e. the degree (0, 90, 180, and 270) of rotation of Lake Tile is determined in random.

The possibility of dedication is determined based on the availability of specific types in player’s lantern cards stack.

**UnfriendlyPlayer**

**Description**

Unfriendly player causes harm to one or more opponents. He or she prevent the opponents from at least minimizing the bonus that they would get. Also, prevent the opponent from obtaining the required lantern card that may help them to make a dedication.

**Functionality**

The player prevents other player to receive dedication tokens.

He or she prevent other players to get a better score on the next round by obtaining the specific lantern card that other user expect to obtain.

Find the score that player may get if he/she gets lantern card with particular color instead of lantern card with another color.

Decide the placement of Lake Tile to prevent other players from benefiting out of it.

Find the score that player may get if he/she place the lake tile at the specific position on the board.

**Clever Player**

**Description**

Clever player uses the strategy of random, greedy and unfriendly players.

**Functionality**

It implements multiple strategies in a round robin method.

Proper assignment of strategies to each player during run time based on the number of players plays.

# Relationship between classes

|  |  |  |
| --- | --- | --- |
| **Class (C1)** | **Class (C2)** | **Relationship Description** |
| UnfriendlyPlayer | Player | C1 is the subclass of C2. It accesses the player lantern card, lake tile stack to do appropriate modification as the player makes a move. |
| PlayerStrategy | C1 implements C2 |
| GameEngine | C1 has an instance of C2 to access the current state of the game and do appropriate modification while player make dedication, place lake tile, or exchange lantern cards. |
| RandomPlayer | Player | C1 is the subclass of C2. It accesses the player lantern card, lake tile stack to do appropriate modification as the player makes a move. |
| PlayerStrategy | C1 implements C2 |
| GameEngine | C1 has an instance of C2 to access the current state of the game and do appropriate modification while player make dedication, place lake tile, or exchange lantern cards. |
| CleverPlayer | Player | C1 is the subclass of C2. It access the player lantern card, lake tile stack to do appropriate modification as the player make a move. |
| PlayerStrategy | C1 implements C2 |
| GameEngine | C1 has an instance of C2 to access the current state of the game and do appropriate modification while player make dedication, place lake tile, or exchange lantern cards. |
| HumanPlayer | Player | C1 is the subclass of C2. It accesses the player lantern card, lake tile stack to do appropriate modification as the player makes a move. |
| PlayerStrategy | C1 implements C2 |
| GameEngine | C1 has an instance of C2 to access the current state of the game and do appropriate modification while player make dedication, place lake tile, or exchange lantern cards. |
| GreedyPlayer | Player | C1 is the subclass of C2. It accesses the player lantern card, lake tile stack to do appropriate modification as the player makes a move. |
| PlayerStrategy | C1 implements C2 |
| GameEngine | C1 has an instance of C2 to access the current state of the game and do appropriate modification while player make dedication, place lake tile, or exchange lantern cards. |

# Reasoning of choosing the corresponding data structures

Vectors is used to store and return lantern Cards and favor Tokens used for simulation back to game.

Vector is used over array due to below reasons

* Vector is synchronized.
* Vector contains many legacy methods that are not part of the collections framework.
* Implements dynamic array

The below methods of vector are used

removeElementAt(int)

lastElement()

size()

elements()