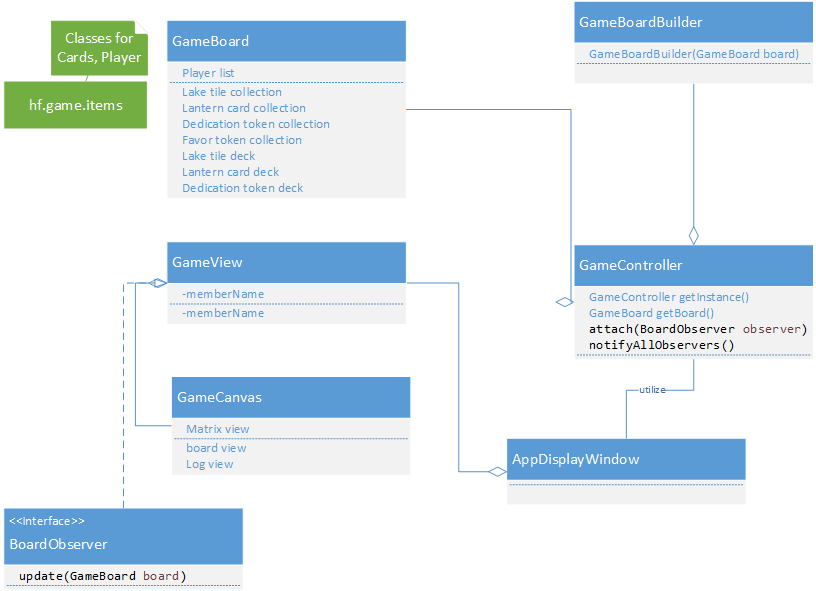
1. Architecture

We use the Model-View-Controller (MVC) which is a commonly used and a powerful high level architecture system for GUIs. The MVC paradigm is a way of breaking an application, or even just a piece of an application's interface, into three parts: the model, the view, and the controller.

* 1. Model

The model is used to manage information and notify observers when that information changes. It only contains data and functionality that are related by a common purpose. A model is meant to serve as a computational approximation or abstraction of some real world process or system. It captures not only the state of a process or system, but also how the system works. This makes it very easy to use real-world modeling techniques in defining our models.

Game board is virtualized as the overall model, and all the UI views are built based on data structures kept in the model. GameBoardBuilder class is the constructor class for game board, it will contain functions to load cards collections, create decks of cards, distribute player hand cards and etc…



GameController class is defined as a singleton class to create and keep only one instance of the game board object, and it features the observer pattern to notify any interested component about a game board update, for instance a saved game has been loaded or a new game has been started.

Two 3rd party libraries have been chosen to better support the project:

* Slick 2D library to provide an easy way of creating UI components, and control UI behaviors
* XStream library to provide an easy way of loading and saving game board object from to XML documents.