Prova finale di ingegneria del software 2024 – gruppo LB08

View Documentation

Introduction

In this paper we will briefly discuss the implementation of the client User Interface in our version of "Codex Naturalis" and how it adapts to the changes during a game. It uses the MVC design pattern.

Structure

The Model-View-Controller design pattern in this case is implemented through a series of ObservableModel classes, each one with their relative ModelObserver. To make the interface reactive to the messages received from the server, the models that need updating are modified by the ClientMessageHandler (see the Network Implementation paper for more details), that is responsible for parsing messages. Then, after each update on the model, the relative observer is notified and calls the update method on the current ViewController. The message handler (except in cases of an echo message) does not directly update the UI directly, it all passes through the model first.

The reference to the currently active ViewController is stored in the StageManager class. ViewController is the abstract class that holds all the methods that are needed to update the interface (loadGameBoard, loadDecks, updatePlayerInfo etc.). This class makes the design flexible so that it can be used for either a GUI or a TUI view controller without modifying the model-observer structure.

Most of the models are used to update in game UI components, except for the ClientStateModel, that through an updatable variable called ClientState keeps track of the current actions that the client can perform (IN_LOBBY_STATE, NOT_PLAYING_STATE etc.). It's also used to independently load other screens in particular cases, for example when the connection is lost with the server.

GUI

The GUI is entirely made using JavaFX. The user can navigate through a series of scenes, each of them managed by their respective ViewController. Each scene loads a .FXML file that contains all the controls of each scene (Buttons, Labels, Text boxes etc.). All the information to load a specific scene and the current View Controller are stored in the StageManager class.

When the GUI application is launched, before loading the title screen scene, all the observers with their respective models are initialized.

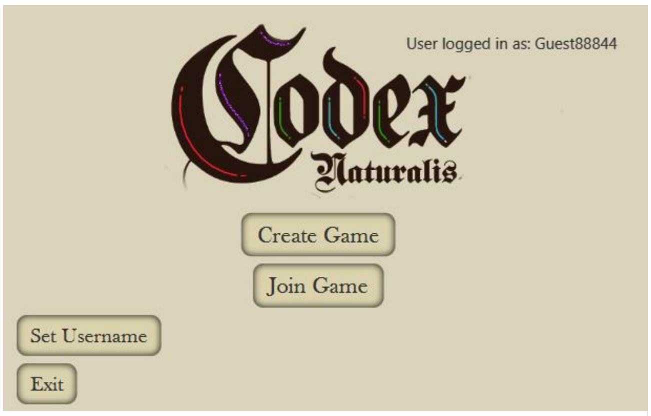
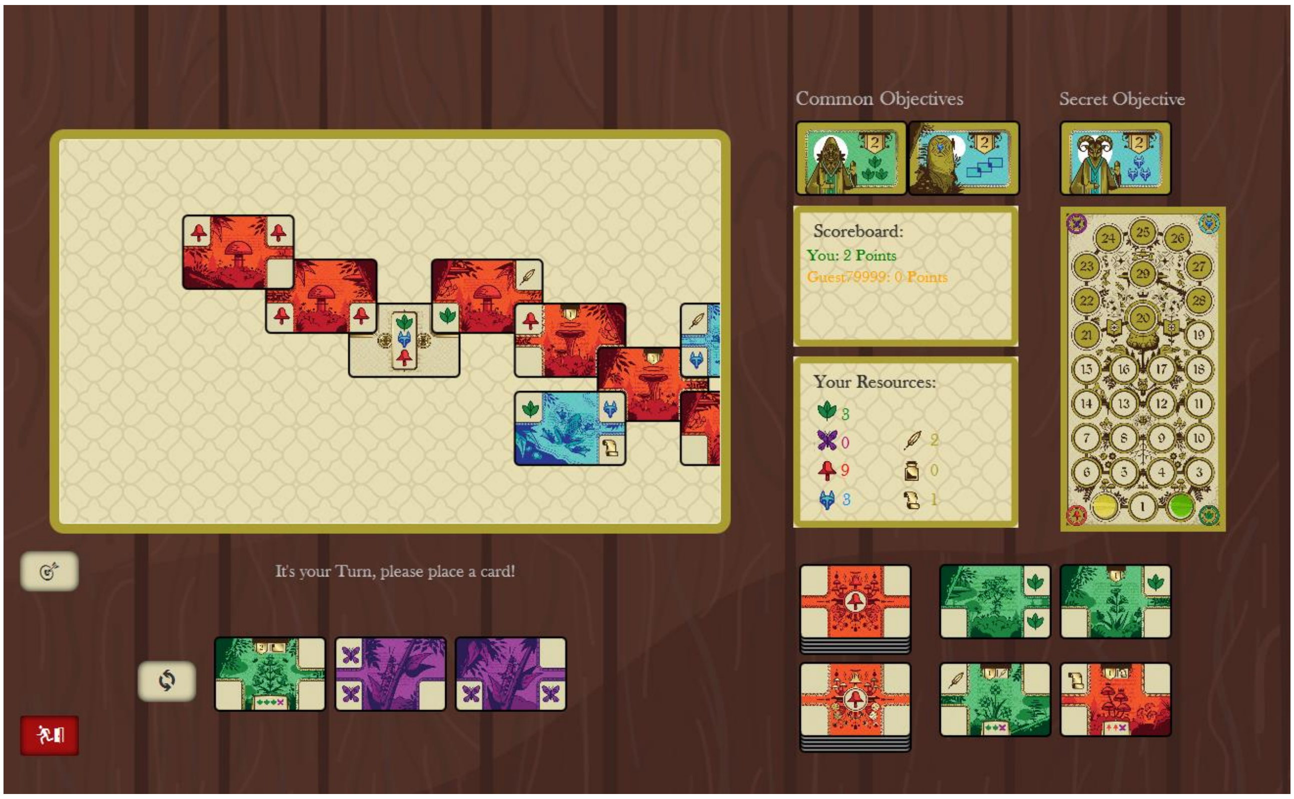


Figure 1 The Main Menu.


Common Objectives

Secret Objective

Scoreboard:
You: 2 Points
Guest79999: 0 Points

Your Resources:
3
0
9
3
2
0
1

It's your Turn, please place a card!

Figure 2 The gameboard. The user can also with the two buttons flip all the cards in his hand (to place them face down) and center the starter card inside of the pane.

TUI/CLI

The Terminal User Interface implements a TerminalInputReader that runs on a separate thread to get the input from the user. Since the CLI does not need to manage multiple scenes (and their relative ViewControllers), the StageManager holds a reference to only one ViewController (CLIViewController), that depending on the current state (stored in the ClientStateModel), prints the needed resources.

Differently from the GUI implementation, the cards on the game board are not directly printed on the board, but they are represented by a univocal ID, colored as the cards' kingdom to make the patterns more readable. If the user wishes to visualize the actual card to see the corner disposition or to check placement conditions, he can do that trough the "info <CardId>" command.

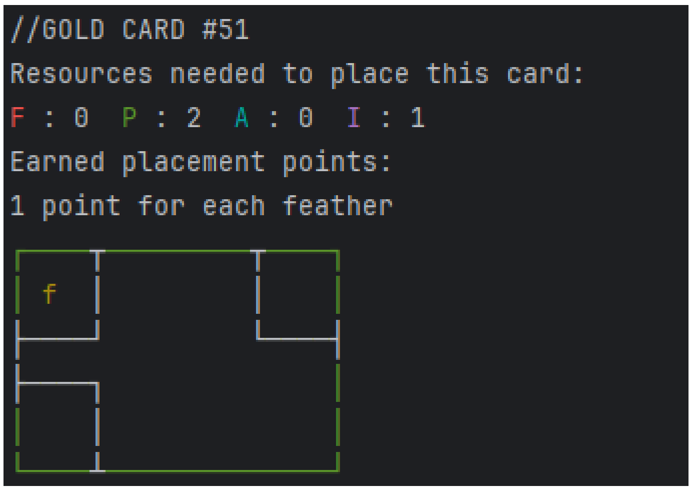
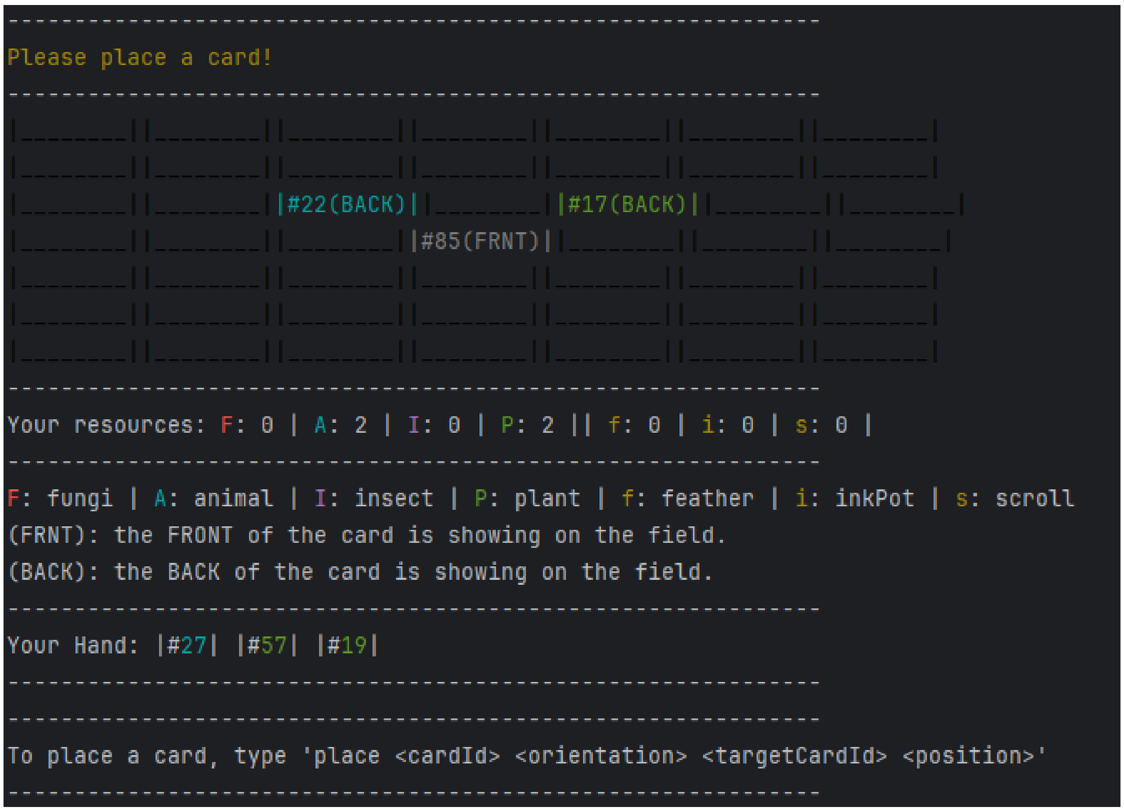

```
//GOLD CARD #51
Resources needed to place this card:
F : 0  P : 2  A : 0  I : 1
Earned placement points:
1 point for each feather
```

Figure 3 A card when it's printed with the info command.



```
-------------------------------------------------------------
Please place a card!
-------------------------------------------------------------
|________||________||________||________||________||________||________|
|________||________||________||________||________||________||________|
|________||________||#22(BACK)||________||#17(BACK)||________||________|
|________||________||________||#85(FRNT)||________||________||________|
|________||________||________||________||________||________||________|
|________||________||________||________||________||________||________|
|________||________||________||________||________||________||________|
-------------------------------------------------------------
Your resources: F: 0 | A: 2 | I: 0 | P: 2 || f: 0 | i: 0 | s: 0 |
-------------------------------------------------------------
F: fungi | A: animal | I: insect | P: plant | f: feather | i: inkPot | s: scroll
(FRNT): the FRONT of the card is showing on the field.
(BACK): the BACK of the card is showing on the field.
-------------------------------------------------------------
Your Hand: |#27| |#57| |#19|
-------------------------------------------------------------
-------------------------------------------------------------
To place a card, type 'place <cardId> <orientation> <targetCardId> <position>'
-------------------------------------------------------------
```

Figure 4 The CLI version of the gameboard.