Bilkent University

Department of Computer Engineering

CS319
Object Oriented Software Engineering
Final Report

Katamino

Group 2-F

Sena Er	21502112
Zeynep Sonkaya	21501981
Hüseyin Orkun Elmas	21501364
Utku Görkem Ertürk	21502497
Bartu Atabek	21602229

1. Introduction

In this report, we aim to discuss the implementation process, the design decisions, the changes during implementation and provide a user's guide for Katamino v2.0. In the

remaining of this section, we will discuss the implementation process and current status of the implementation.

After the design process of our game, we started implementing the game. In the design report, we decomposed our system into subsystems.The subsystems were useful for distributing the work. We assigned subsystems as tasks. However, some of the subsystems were dependent on each other; therefore, some subsystems were assigned for multiple tasks or multiple people were assigned tasks that were related to a subsystem. Yet, this was minimal because we valued loose coupling in our subsystem design.

We made our meetings face to face when possible, in other cases we used Skype. In these meetings, we took design choices about implementation and structure of our game. We had initial meetings to create tasks and after that most meetings were about specific tasks, therefore the meetings were between assignees of the tasks. This resulted in more efficient meetings.

In this stage of our implementation of Katamino v2.0, we have implemented the single player mode of the game.

Currently game includes, movement of pentominoes such as drag and drop, rotations and lips. The algorithms for detecting clashing pentominoes and detection of a full board is implemented. Also, a stopwatch to measure game time, and to calculate score is implemented. In GUI subsystem; main menu, level selection, player selection, high scores and credits screens are implemented. Moreover, save and load mechanisms for player selection and board are implemented. Furthermore, settings menu and mechanism to enable sound adjustment is implemented.

The other two modes, namely multiplayer and custom board modes are not implemented in this version. And for single player version; support for hints and solutions are not implemented.

In terms of user interface, components of the game could be integrated into fxml files. However, for the sake of the simplicity and understandability we separated our components from the main project and built a component library. In the contemporary design of the project imports the external component library which is internal work product.

2. Design Changes

In our design report, we have decided used some model and controller classes.After implementation we have decided some are not necessary and some should be redesigned.

First we thought both GameBoardController and GameController should be separated for the sake of MVC design. However when we start to implementation we realized that GameBoard is a supplementary model for our game so that we still separate GameBoard and GameModel but we combined GameBoardController into GameController. So we removed GameBoardController as a separate class.

As an another design change, we removed Serializable interface. This is because we want to change and easily understand setting files of the game so that we determined to save game files as Json file. Also this make possible for players to create their own maps. Moreover we no longer need to make whole classes implement serializable interface for the serialization process. Because of this change our FileManager class changed as well. FileManager uses external Json serialization library called Gson and Interior methods of the FileManager adapted to Json file usage instead of Java's Serializable interface.

In first phases of our project we planned our player selection menu will be in the usage of limited players. So that player selection menu will have only 3 users. However, we changed this limitation by changing UI design of the player selection menu. In updated version of the menu we used spinner which compatible with our UI components.

One of design changes is about the Timer that used in GameModel. We want to record pasted time for updating score of a player during game which is can not be succeeded without modifying java.util.Timer. We have decided use an Stopwatch which is a Class implemented for counting up.So a new class called Stopwatch is added to our design.

Another changed is occured about our understanding of the game after implementation about katamino game control system. We had a combined class called GameController which controls a one level and also it has been thought as it controls one game flow of an player. For maintaining information about levels and creating a flow we have decided to add a GameManager class as an controller of a separate GameContoller. By separating one class to two class in design we have gained control of a single level and the main game separately.

For the first iteration,we have different Levels that have different initial stages(not yet played Katamino games) and solutions of them. We have not included the solution information to our Level class in the design stage but in implementation Level class will contain the information about initial stage and solution. Level will have related information passed by FileManager which get the information from the json file. For this relationship we had to modify FileManager and Level classes.

In our object design we have not included an enum class for Cell information which is about state of the cell as emptyNotOnBoard, pentominoNotOnBoard, pentominoOnBoard, blocked and emptyOnBoard. However for readability purposes we have need it so we have included an enum class called CellType with mentioned information.

3. Lessons Learnt

During our implementation we have seen that design and analyses stages are really important and useful for implementation stage. When we start implementing Katamino, we have used many information that has been already discussed in the previous stages. We have not thought everything in some parts and those caused problem in the implementation part. We know that change can occur anytime of the programing and why we implement it we have witnessed it from first-hand experience. We have reconsidered our design and learned that not everything could be determined in the earlier stages however many things could be determined in the earlier stages and used in later stages.

Another lesson we learnt throughout the implementation phase is group work is essential and also division of tasks could be hard while tasks are sometimes vague. We had an system architecture that separates our tasks into layers. However in every layer some classes are really connected to each other that they had hard time to coordinate assignment division and used different approaches for task assignment. We have seen being a little bit flexible is useful in a group work while keeping the balance of equal division of labor.

4. User's Guide

4.1 System Requirements & Installation

Katamino can be runned in all platforms and does not need any installation. It is a single executable JAR. However, in order to be able to run the JAR, computer should have Java Runtime Environment installed.

4.2 How to Use

4.2.1 Game Overview

Katamino 2.0 is a tile based 2D puzzle game which is styled with a retro modern user interface, created for the desktop platform. After the initialization of the application the user is presented with a welcome screen which prompts them with options such as choosing a gameplay mode i.e. single-player, or multiplayer and changing the settings of the game. Players will be able to save their progress and compare their scores with other players.

4.2.2 Menu Operations

4.2.2.1 Starting a New Classic Single Player Game

Player can start a new game by simply clicking on the new game button in main menu screen. After clicking the button, users will be asked to choose their player from the menu and if not present they can create a new player by entering a username. Afterwards the player chooses between the different game modes and after selecting “Classic (Arcade) Mode the player will be represented with the available game levels. After selecting a level from the menu the game automatically starts.

4.2.2.2 Displaying Highscore Board

Users can view the current high scores of the players sorted by their scores by clicking on the High Score button in the main menu. After clicking the button, high scores will be displayed in the screen.

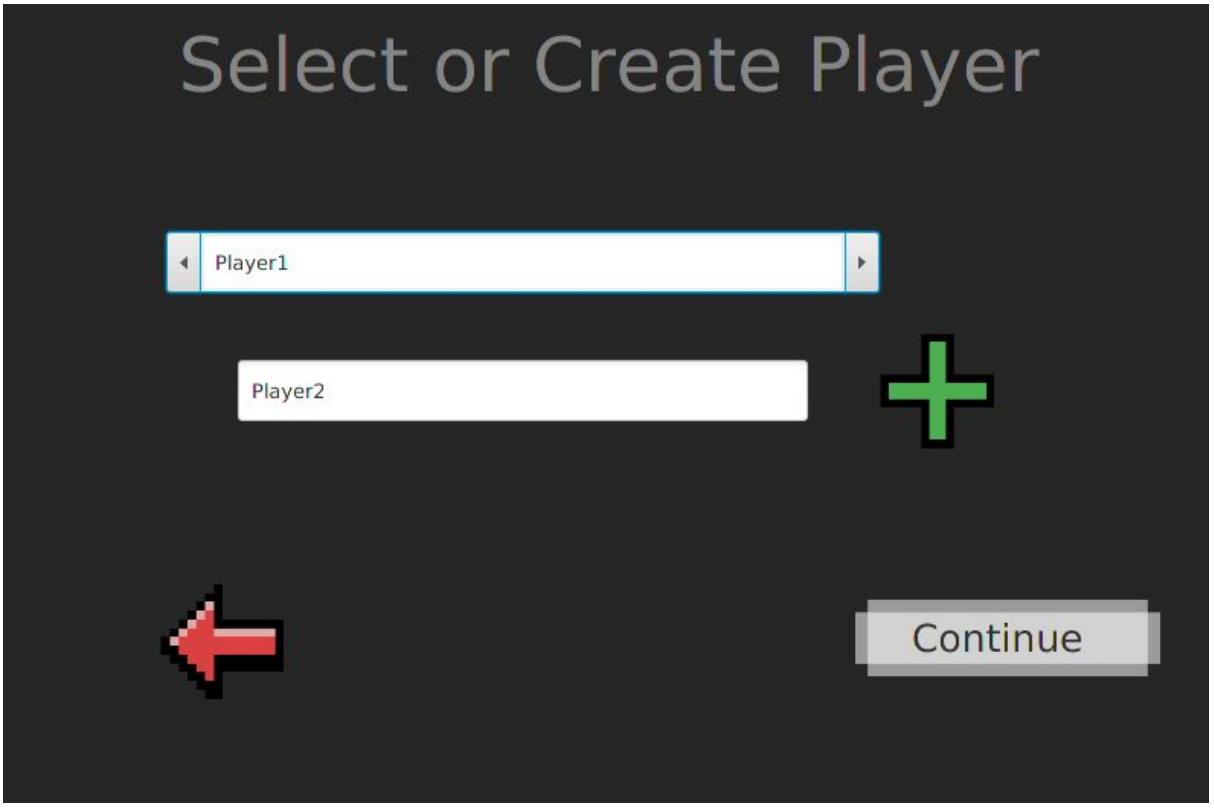
4.2.2.3 Viewing Credits

Player can view credits by clicking on the label text at the bottom of the window. After clicking the button, creators of the game will be displayed in the screen.

4.2.3 Single Player Arcade Gameplay Operations

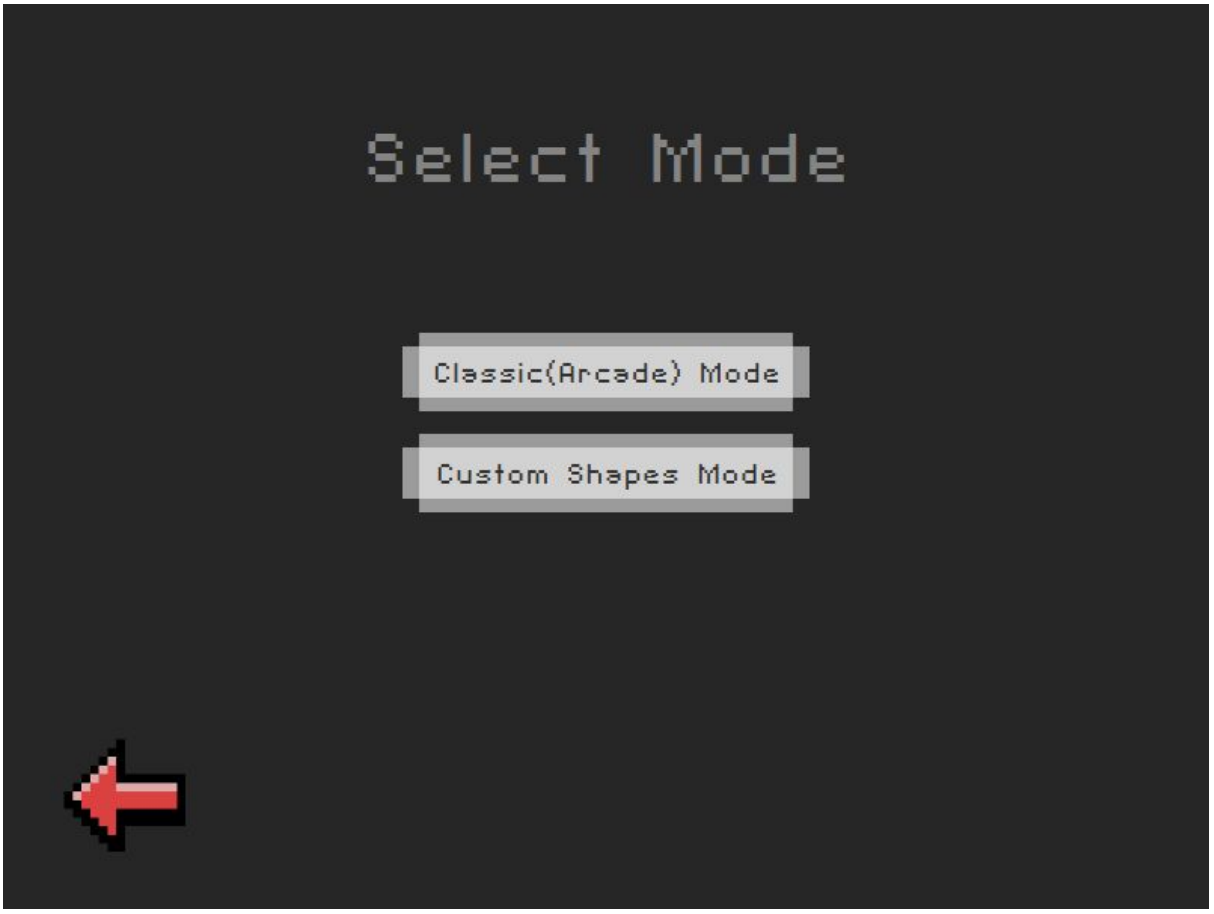
4.2.3.1 Player Selection Menu

After selecting the single player option the user will be presented with the saved players of the game. Users can slide through the players and choose theirs by selecting it or they could create a new player by entering their name and click the '+' button. After a player is selected they will be guided to the game mode selection menu.



4.2.3.2 Game Mode Selection Menu

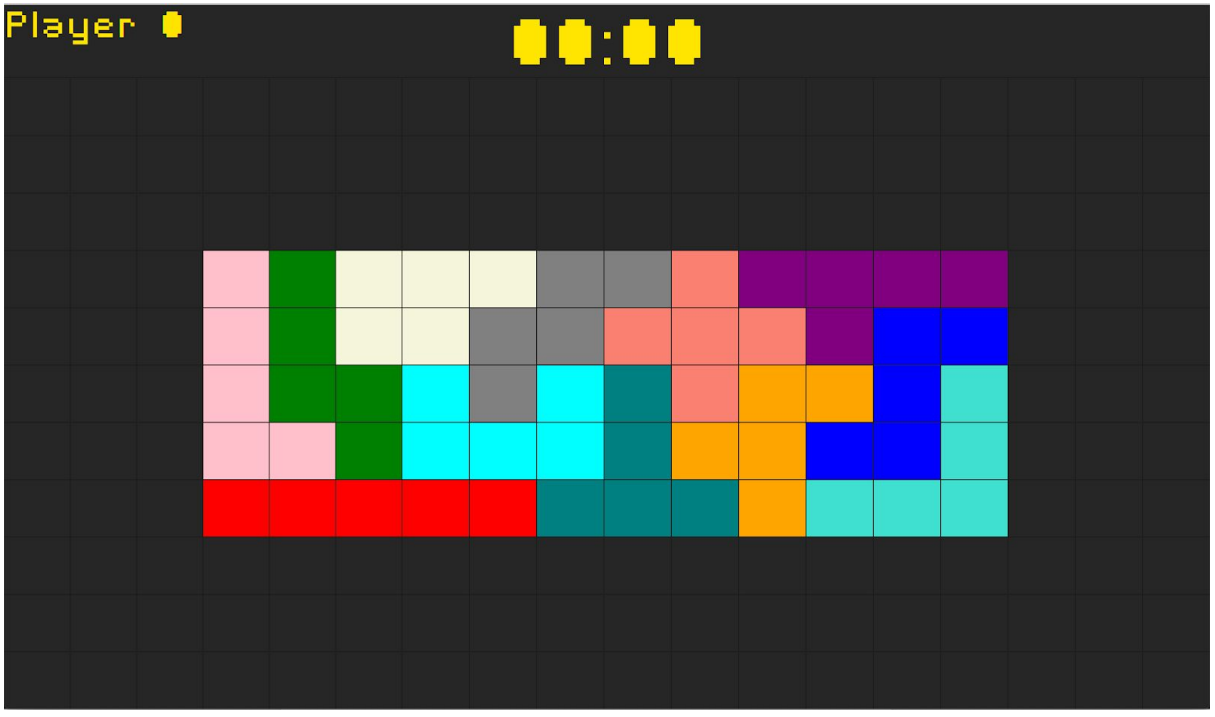
The players presented with two gameplay options; Classic (Arcade) mode and Custom Shapes mode. When clicking either one of them, the player will be presented with the game levels according to their selection.



4.2.3.3 Level Selection Menu

The players can choose from the available game levels presented. Some of the levels will be locked and can be seen as greyed out from the interface. As they progress throughout the game more levels will become available and unlocked. When a level selected by clicking their icon the selected level will be loaded to the game board and the game will start.

4.2.3.3 Gameboard Operations



This is the main game screen. This screen will be displayed after level selection is done.The player select pentominoes available for the specific level and using drag and drop method

the player should fill the board. Also, there is a stopwatch for time tracking. The main goal is filling the board with pentominoes.