Module Guide

Super Tetris

Group#: 38

Team Name: Binary

Members:

Tongfei Wang : wangt62

Bowen Yuan : yuanb1

Tim Zhang : zhangj14

December 6, 2017

SFWR ENG 3XA3

McMaster University

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
Nov. 11th	1.0	Upload the context
Nov. 11th	1.1	Modification
Dec. 6th	2.0	Rev_1 We decompose the Module, This whole document is rewrited. Because of this we do not add any color to represent the changing

1 Introduction

Super Tetris is an online game modified from the original Tetris.

To achieve the goal, we are going to add the new features like an internal economic system, experience system, item system and special Tetriminos to increase the complexity. When deleting a line, a certain amount of gold and experience will be awarded to the player. The experience unlocks high-level items while the players can buy the items by the gold they earned. Moreover, there will be more different shapes of Tetriminos in our game as to make the game a little more challenging. Some of the Tetriminos will have a certain type, such as bomb-type, wall-type, double-gold-type and so on. Every type has its own different function when they are deleted.These features will bring another perspective to value the gameplay for the players. The game is developed in javascript and html.

The documentation is organized as follows:
Section 2: The anticipated and unlikely changes of the program.
Section 3: A module hierarchy is presented.
Section 4: The context of the connection between the design and requirements
Section 5: Specified decomposition of the modules
Section 6 : The traceability between requirement and modules and between anticipated change and modules.
Section 7: The relationship between modules

2 Anticipated and Unlikely Changes

This section lists possible changes to the system. According to the likeliness of the change, the possible changes are classified into two categories. Anticipated changes are listed in Section 2.1, and unlikely changes are listed in Section 2.2.

2.1 Anticipated Changes

Anticipated changes are the source of the information that is to be hidden inside the modules. Ideally, changing one of the anticipated changes will only require changing the one module that hides the associated decision. The approach adapted here is called design for change.

AC1: The format of input data.

AC2: UI design

AC3: The falling speed algorithm

AC4: The balance of the internal economy

AC5: The algorithm of the item

2.2 Unlikely Changes

The module design should be as general as possible. However, a general system is more complex. Sometimes this complexity is not necessary. Fixing some design decisions at the system architecture stage can simplify the software design. If these decision should later need to be changed, then many parts of the design will potentially need to be modified. Hence, it is not intended that these decisions will be changed.

UC1: Input/Output devices (Input: Keyboard Output: Monitor)

UC2: The outlook of the tetrimino

UC3: Game mode(Two-player vs. Single player)

UC4: Programming language usage

3 Module Hierarchy

This section provides an overview of the module design. Modules are summarized in a hierarchy decomposed by secrets in Table 2. The modules listed below, which are leaves in the hierarchy tree, are the modules that will actually be implemented.

M1: Index Module
This module is responsible for connecting the system to the website.

M2: Control Module
his module is responsible for initializing the game and linked all the function the game.

M3: Style Module
This module is responsible for the layout of the web page.

M4: detect Module
This module is responsible for checking if the process condition of the game including whether the game should be ended.

M5: drawmatrix Module
This module is responsible for displaying the game.

M6: playerinput Module
This module is responsible for receiving the input of the user.

M7: bomb Module
This module creates a bomb which can be used as an item while playing the game.

M8: randomclear Module
This module makes a function which can be used as an item while playing the game.

M9: slowdrop Module
This module makes a function which can be used as an item while playing the game.

Top Level	Level 1	Level 2	Level 3
	Style Module		
Index Module			
	Control Module	detect Module drawmatrix Module playerinput Module	slowdrop Module randomclear Module bomb Module

Table 2: Module Hierarchy

4 Connection Between Requirements and Design

The design of the system is intended to satisfy the requirements developed in the SRS. In this stage, the system is decomposed into modules. The connection between requirements and modules is listed in Table 3.

5 Module Decomposition

Modules are decomposed according to the principle of "information hiding" proposed by **?**. The *Secrets* field in a module decomposition is a brief statement of the design decision hidden by the module. The *Services* field specifies *what* the module will do without documenting *how* to do it. For each module, a suggestion for the implementing software is given under the *Implemented By* title. If the entry is *OS*, this means that the module is provided by the operating system or by standard programming language libraries. Also indicate if the module will be implemented specifically for the software.

Only the leaf modules in the hierarchy have to be implemented. If a dash (–) is shown, this means that the module is not a leaf and will not have to be implemented. Whether or not this module is implemented depends on the programming language selected.

5.1 Hardware Hiding Modules

Secrets: The data structure and algorithm used to implement the virtual hardware.

Services: This module connects the input and output data. through the interface between the hardware and the software.

5.2 Behaviour-Hiding Module

Secrets: This module includes all the required behaviours.

Services: : This module shall be able to show every visible behaviour of the system specified in the software requirements specification (SRS) documents.

5.2.1 Input Format Module (M6)

Secrets: The format and structure of the input data.

Services: Converts the input data into the data structure used by the playinput module.

5.3 Software Decision Module

Secrets: This module includes all the method, methodology and algorithm of the system.

Services: The module does all the all the algorithm and makes decisions which are not exposed to the users.

6 Traceability Matrix

This section shows two traceability matrices: between the modules and the requirements and between the modules and the anticipated changes.

Req.	Modules
FR1	M1
FR2	M3 M5
FR3	M2
FR4	M2
FR5	M3 M6
FR6	M3 M6
FR7	M3 M5
FR8	M2
FR9	M4
R10	M4
R11	M4
R12	M2
R13	M2
R14	M7 M8 M9

Table 3: Trace Between Requirements and Modules

AC	Modules
AC1	M6
AC2	M3
AC3	M2
AC4	M2
AC5	M7 M8 M9

Table 4: Trace Between Anticipated Changes and Modules

7 Use Hierarchy Between Modules

The project has only two module. The first one is the Interface Module which is developed in HTML. The other one is Function Module and it is developed in Javescript.

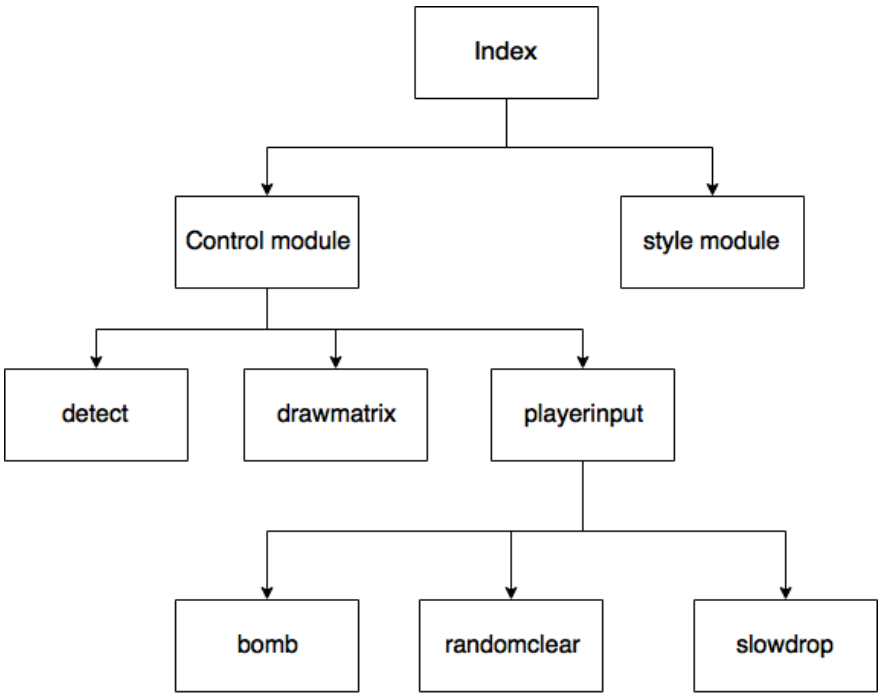

Figure 1: Use hierarchy among modules