Computer Games Architecture Report

Li Guo

220060950

Content

[Part 1: 2](#_Toc166260346)

[Assets 2](#_Toc166260347)

[Control of the game character 2](#_Toc166260348)

[Collision detection 2](#_Toc166260349)

[Moving and animated game elements 2](#_Toc166260350)

[Part 2: 3](#_Toc166260351)

[Game world with data-driven approach 3](#_Toc166260352)

[Removal of game elements 4](#_Toc166260353)

[Scoring system 4](#_Toc166260354)

[High-score table 5](#_Toc166260355)

[Part 3: 6](#_Toc166260356)

[Start-screen and game over screen 6](#_Toc166260357)

[NPC opponents 7](#_Toc166260358)

[Overall game-play 7](#_Toc166260359)

[Part 4 8](#_Toc166260360)

[References 8](#_Toc166260361)

# Part 1:

## Assets

All the assets are configured and restored by using MGCB Editor. For all the scene backgrounds, they are using the original template backgrounds and edited for a little bit to fit for different screen concept. For example, according to Figure 1, the background for death screen adds two lines of sentences of “You Died” and “Press Space to restart” to give instruction to the player. The main player and enemies are using the characters in the template. There is a new collectable item added into the game is the star and there is a new type of trap added into the game, which both are saved in the sprite folder.

蓝色的天空

描述已自动生成

Figure 1 Death screen

## Control of the game character

The game character can be controlled by using space, W, A and D. When pressing space or W, the game character can jump with certain height, and during the character is in the air, the jump function will not working. A and D are used for character to go left and right. When the character in the air, A and D are working and character can move left and right in the air.

## Collision detection

For the collision, I have added bounding to all the characters and objects. For the collectable items, when the bounding of the main character is detected to hit the bounding of the collectable item, this item will be removed from the list that saved all the items in and play the sound of collecting this item. For the traps and enemies, when the bounding of the main character hits the bounding the of the trap, the death state will be activated and stop the game state, the death screen will pop up force the game to stop. For the exit point, when the bounding is detected to hit, the win state will be activated, and the game winning screen will pop up by stopping the game state.

## Moving and animated game elements

There are two animated game elements. For the enemies, they can walk left and right on the plane, when the enemy is detected on the edge of the plane, it will stop and turn around. For the collectable items, it will bounce up and down within certain range of 0.18 of the game height. The bouncing speed is set as sin(t\*Br+x\*Bs). The Br means the bounce rate, which is set as 3, the t is every second in the gametime, the x is the number of x ray of each collectable item and the Bs is the bounce sync which is set as -0.75 [2].

文本

描述已自动生成

Figure 2 Bounce code

# Part 2:

## Game world with data-driven approach

The game world is saved in the txt file, which in this file, the game world can be configured by using different kind of indexes to represent different game elements. For example, the plane can be set up by using #, enemies can be set up by using A,B or C for different kinds of enemies, the starting position of main character can be set up by using 1, the collectable items can be set up by using S or G for different kinds of collectable items and the trap can be set up by using T. The coding structure can be seen in Figure 3 and demonstration of txt file can be seen in Figure 4.

文本

描述已自动生成

Figure 3 Game world coding

电子设备的屏幕

低可信度描述已自动生成

Figure 4 game world txt file

## Removal of game elements

For the collectable items when the bounding of the collectable items is detected to hit the bounding of the main character, the collectable items will be removed form the list that save all the collectable items and they will not be displayed in the game world. The Figure 5 shows the code for collision detection and the Figure 6 shows the code for collision response.

文本

描述已自动生成

Figure 5 Collision detection

文本

描述已自动生成

Figure 6 Collision response

## Scoring system

For the scoring system, when the player collect the collectable items, it will add the score to the total score that player already have, each collectable item will add 30 points. And when player finish the game, it will add the score to the total score based on how much time has left, for every second left, it will add 5 points.

文本

描述已自动生成

Figure 7 Collectable items score

文本

描述已自动生成

Figure 8 time remaining score

## High-score table

For the high-score table, all the history scores are collected and saved in a txt file, when the player reaches the exit point, the score will be saved into the txt file. And player can see all the history scores in the score screen by loading the txt file and save all the score into the array and print them on the screen.

The score table structure is set up depending on the number of history scores in the score txt file, and the gap between each score is set as the same to make the structure of the high score table to look comfortable.

文本

描述已自动生成

Figure 9 Saving score

文本

描述已自动生成

Figure 10 Loading Score table

# Part 3:

## Start-screen and game over screen

The start screen contains the title of the game of “If you are man, go down 100 levels”, the keyboard controls instructions, and two buttons of “Start Game” and “Score Board”. The game over screen contains the score of this game and the restart playing the game by going back to start screen.

There are four game states in the game loop, there are Intro, playing, endgame and win. For the start-screen the state is Intro state, and the function of starting the game and open score screen functions are shown in the Figure 11.

文本

描述已自动生成

Figure 11 Intro screen functions

There are two kinds of game over screen, one is the winning screen, and the other is the death screen. For both kinds of screen shows the score of the player has obtained and gives the function when player press SPACE key on keyboard, the game state will change to Intro and pop up the Intro screen. The code functions for changing game states of game over screen are shown in Figure 12.

文本

描述已自动生成

Figure 12 two kinds of game over screen

## NPC opponents

The NPCs can walk on the blocks and when NPCs reaches the edge of the plane that they are on, they will turn around and walk to the other side of the plane. The following Figure 13 shows the code of waiting for change direction of face direction and walking on the plane.

文本

描述已自动生成

figure 13 NPC walk function

## Overall game-play

This game is a Marial-like game, in the intro screen, player can press F1 to see the history score of passing the game. When press enter, the game will start. In the game play scree, the scene will go down automatically, player can press W to jump, A and D to go left and right, and finally when player reaches the exit point, player will win. If player dies when falling off the stage or collides with enemy, there will pop up a death screen. In the death screen, player can see the score they have when the time they die, but the score will not be counted in the history scores. And player can replay by pressing space to enter the intro screen and start replay. If the player collides with the exit point, there will pop up a winning screen. In the winning screen, player can see the final score and this score will be counted into the history scores and player can replay the game by pressing space to enter the intro screen and replay.

# Part 4

My game architecture could be adapted for a client-server model other than peer-to-peer model. According to Glazer, J. and Madhav, S., client-server model is the model that players are connected to the center server and all the data are saved and loaded from the center server. For my game architecture, all the data are saved in the txt file and all the models are saved in the folder, it will be easy to read all the documents easily from the center server and load all the models and txt files. The center server can save all the coding documents and item models and txt files for levels and history scores, then when the server running, the html for the game can read all the documents from the server and process the game world on the browser.

# References

Glazer, J. and Madhav, S., 2015. Multiplayer game programming: Architecting networked games. Addison-Wesley Professional.