CMPT 276 D100

FALL 2021

GROUP 21 - PHASE 2

APPROACH

- To implement the skeleton of the game first (i.e., creating a simplified version of the game with basic functions) with a working back-end that can be represented textually.
- Holding meetings roughly every week to discuss responsibility, future vision of the game, and problems each group member faces.

For the first week or two, our team has not conducted team meetings frequently compared to last week, as we did not have a clear goal at the time. Over time, however, we began to have a better understanding of what needs to be implemented, how much time must be invested in order to develop certain functions and methods, and how certain methods should be handled. In the beginning, we only have a simple 2D-Text Maze to represent our board and the movable object within the maze is the main character. As we progress, we added collision interaction between objects, images to represent the board, then menu, start up button, and end screen. All of these changes are part of the discussion we held weekly, which helped us organize the ordering of what needs to be done; this approach allowed us to progress systematically, ensuring we are not starting too late into the project.

ADJUSTMENTS TO PHASE 1

- Ditching StaticAgent/DynamicAgent
    - Was too unnecessarily complex, so the idea got scratched
- 2D array of Cell objects with MazeObject lists instead of 2D array of MazeObjects
    - Cell object provides ‘depth’ to the array. This solves problems of collision and interaction between objects.

Agent

- Agent
    - Remove StaticAgent and DynamicAgent
    - Let Class MainCharacter and Enemy extend Agent directly
    - Added variables moveSpeed nad movable
- Direction
    - Added enum variable Direction
- Enemy
    - Class Enemy extends Agent
    - Remove EnemyA and EnemyB
    - Just set the moving Enemy
- MainCharacter
    - Class MainCharacter extends Agent
    - Added two variables: maxHealth and characterScore
    - Getter and Setter methods for all variables

Display

- Assets
  - Location of game assets
  - Image resources
- HealthBar
  - HealthBar extends StackPane
  - Display mainCharacter HP
  - Set HP font

Fixed

- BonusReward
  - BounsReward extends Reward
  - Add LifeSpam() return lifeSpam
- EventTile
  - EvenTile extends Wall
  - Set “StartTile” and “EndTile”
- Lootable
  - Lootable extends MazeObject
- Reward
  - Reward extends Lootable
  - Set score
- Space
  - Space extends MazeObject
- Wall

Game

- Cell
- Clock
  - Clock extends Pane
  - Clock function displays as minute and seconds “0 m 0 s”
  - Added getDuration( ) that return seconds
- Maze
  - Added some list variables to store Agent, Lootable and EventTile
  - Added a method check collision with rewards
  - Added a method check collision with enemies
  - Added methods that remove reward and enemy
  - Added a method to judge collision with walls
  - Added a method that let enemy move
  - Added a method that return a cell decided by one input cell
- MazeObject
  - Remove some unneeded variables and methods
- Game
  - Class Game Extends JavaFX Application
  - Added two scene: one scene displays the game, the other displays the Pause Page
  - Added controller, which allows player to control main character with arrow keys

- start
    - New class start Extends JavaFx Application
    - Start Page for the main Menu of the game
    - Added three buttons (“Start” “Exit” “Introduction”)
- EndPage
    - New class EndPage Extends JavaFx Application
    - End Page for the end (failure) of the game
    - Added two buttons (“Restart” “Back to Main Menu”)
- WinPage
    - New class WinPage Extends JavaFx Application
    - WinPage for the winning of the Game
    - Add two buttons (“Restart ” “Back to main Menu”)
- IntroductionText
    - New class IntroductionText Extends JavaFx Application
    - Introduction Page for the introduction of the game
    - Added one button (“Back”)
    - Added one Text variable for message

Levels

- Level
- LevelOne

MANAGEMENT PROCESS & RESPONSIBILITIES

Management Process

- Discuss responsibilities and roles, and setting deadline for certain functions on certain dates (roles varies between meeting) through regular group meeting

Responsibilities

- To complete assigned task by the next meeting
- Debugging the game through playing it

EXTERNAL LIBRARIES

- JavaFX
    - To implement player interaction with the game and to create said game’s window(s).

Through the use of JavaFX, we were able to implement many important functions our game needed; for instance, the buttons, windows, panes, animation, timer, and many more.

QUALITY CONTROL

- Regular discord meetings including possible screen sharing
- Regular communication about git activities & delegating tasks
- Regular group chat for any minor / major problems

CHALLENGES

- Learning JavaFX on a limited time frame
- Translating 2D array data into a GUI with events
- Task conflicts between group members
- Connect different scenes
- Require us to learn new programming knowledge and apply it immediately in the project

One of the biggest challenges in phase 2 is the nature of working in gitlab. Quite frequently, group members will face merge conflicts due to different implementations and interpretations; in some cases, new bugs are created or found due to additional coding. Not to mention, when a team member pushes a brand new code, sometimes it is required for other team members to learn, debug, and understand the new code fully in order to fix any current or potential bugs, which is time-consuming, especially when the deadline is considered. Furthermore, there are instances where group members have clashing tasks that wastes a lot of time.

Another big challenge is learning GUI. As our group has little to no experience of coding with GUI, all of us are required to learn new concepts of it. This was especially hard, considering that GUI is not a main topic in our lectures and we have to learn both the lecture materials and GUI; this challenge did become easier over time as our group experiments and learns about GUI, but it was a time consuming process which adds to the difficulty of phase 2.

All of these challenges require a considerable amount of time, which makes meeting the deadline really difficult. We believe that even though we met the deadline, numerous tests and debugging must go into phase 3 before the game feels complete, as we have not gone into extensive testing yet.