Tampere University
Tampere University of Applied Sciences

Nysse Game - Programming 3

Programming 3 autumn 2020

Minh Hoang Nguyen
Thao Huynh Lam Phuong

Description

This project aims to implement a simple gameplay with basic features which can allow controlling figure's movement with arrow keys, and interactions with and other game objects such as coins and buses. The game rule is simple. Players try to earn as much points as they can before the end of the day (in-game time starts at 5.30 am, ends at 11.30 pm), while avoiding the buses in traffic. The coins are generated randomly on the city's map. Buses, on the other hand, regularly changes their positions according to the arranged schedule, which also change their number of passengers, shown above the bus icon.

User Manual

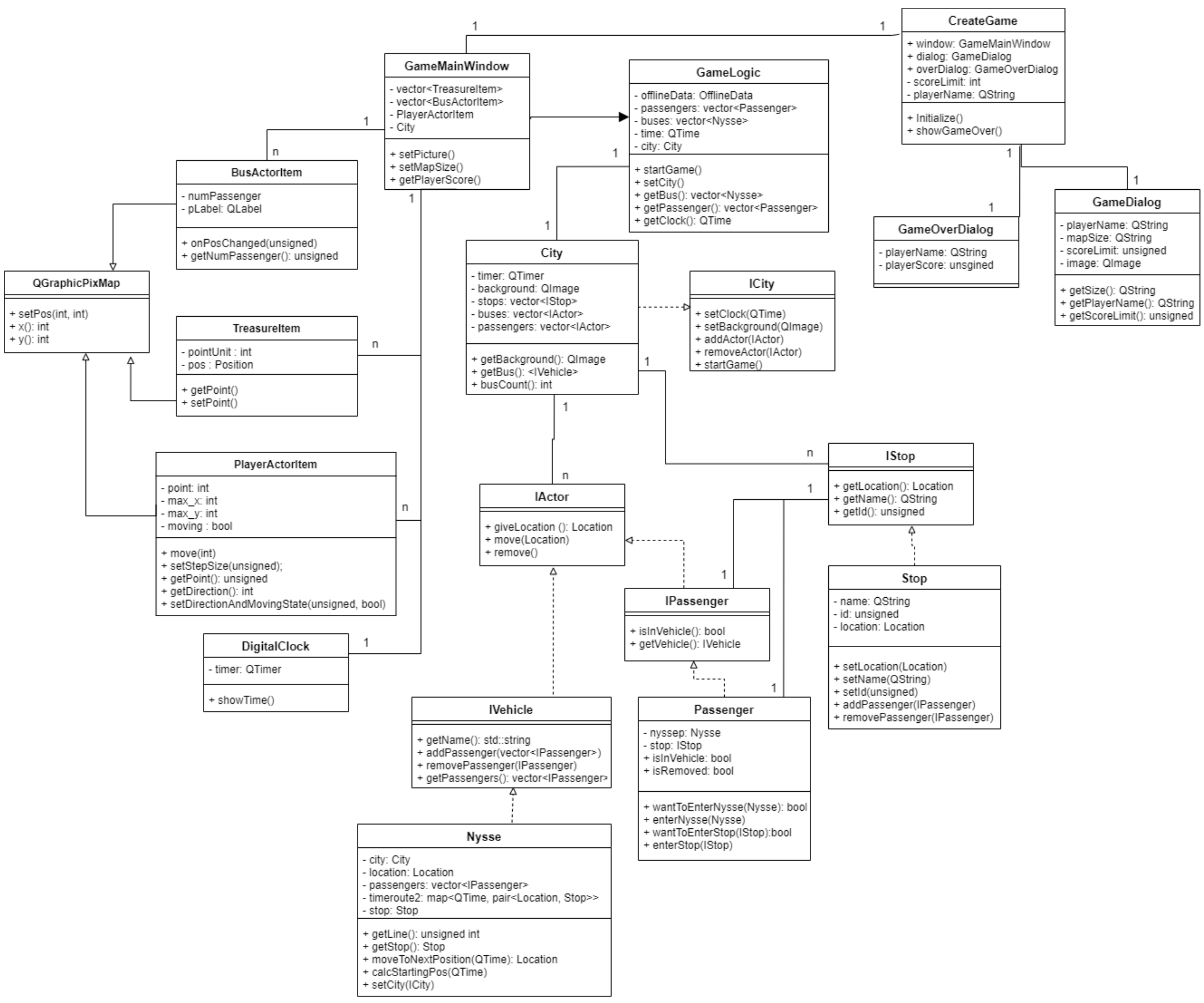
- The game starts after player have finished editing fields in start dialog. The game asks users with name, the preferred score limit and the options for map size as small or large.
- The player starts the game by clicking the player character in the main window. The player can choose to navigate the character by using arrow buttons on their keyboards or arrow buttons in the main window of the game.
- The player moves the character with the arrow keys to move toward coins figure to collect them. Depending on size of the pile of coins, players can earn up to 40 more points.
- The players are to avoid buses coming. On getting hit, 10*(no of passengers) point will be subtracted.
- The game ends at 23:30 (in-game time) or when the player reaches the score-limit that he/she set at the beginning of the game.

Game features:

- Even screen updates: The game is updated to show the real-time location of all buses, coins.
- Minimal screen update. When the actor character is moved, its surroundings are updated in the game area instead of drawing the entire map.
- Even movement of the playable figure. Instead of immediately jumping from one place to another, the character goes to a place with a suitable speed.
- Passenger amounts. The amounts of passengers on the busses and the bus stops is shown on the map as numbers over the icon.
- Following the game state. Scores collected during the game are shown in real time in the main window.
- Own feature as follows:
    - Music played during the game.
    - Different sound effects play when the character hits the bus or collecting coins.
    - Digital clock display in-game time.

- The player can choose the target score limit for him/her at the beginning of the game.
- The player can choose the size map which he/she prefers.
- The actor character can collect coins to increase the scores.
- The speed of the figure increases as player earns more points.

Class diagram



Classes Responsibilities

The key responsibilities are allocated between GameMainWindow class and GameLogic class. GameLogic contains the rules of how each individual game object (Nysse, Passenger, City, and Stop) are behaved in the game implementation. The key function advance(), in particular, demonstrates how the buses, passengers, and stops will updates on Timer.timeout() event. GameMainWindow, on the other hand, generates all of the graphical items (PlayerActor, BusActor, TreasureItem) and initiate signal-slot connections among them. By implementing the GameLogic, the location property of the bus items is generated according to the time-location schedule, which was collected from the json data file with OfflineData. The CreateGame class is responsible for initializing the communication between the GameMainWindow and dialog windows.

Unit Test

Statistic class was implemented to run with unit-test, although all other classes that were implemented by our team were commented in doxy format. The UnitTest subproject included test cases for the documented functions in Statistic class. For future development, we can run the UnitTest project as pipeline by creating a gitlab-ci.yml config file.

Workload division

Both members participate in planning and designing the boiler plate version of the game as well as the documenting the project.

With regards to allocation of programming,

> Minh is responsible for designing the game logic, property of each graphical objects in the game, the timing of gameplay and the player controls configuration.
>
> Thao is responsible for designing graphic user interface, unit tests, and sound effects of the game.

The effective communication between members plays a huge role to overcome some challenges that arise during the implementation.