FVS Extension Report

Extension summary

The project was extended to include obstacles, scoring and quantifiable goals as per the requirements for assessment 3, we also made general enhancements that fixed several existing bugs and improved the general playability of the game.

Challenges encountered

- There is a bug with the existing custom made scene mechanism that results in a red background when more than one dialogs are pushed onto the screen. We weren't able to fix it due to not entirely understanding the management of the scenes and lack of documentation (since it was custom made by the previous group). When contacting the previous group regarding this bug they told us there is not an obvious and easy fix so we decided to leave it at the buggy state as it is right now.
- The Save Game and Load Game functionality required substantial modification as they did not function correctly. The code for this has been removed and the GUI adjusted as necessary. There was no documentation available for how the feature was intended to function so it would have been hard to meet the original idea of the feature.
- The Leaderboard had not been implemented after the previous Assessment, however a scene for displaying it was available. This was removed as the feature could not be implemented in a meaningful way. This would have also required using the local file system to store information. As the current functionality doing this did not function correctly it was decided not to proceed with this feature.

Modifications

- **Obstacles**

To implement the obstacles and the warning system, we tried to reuse the already existing code to minimise time and errors which would occur with refactoring, therefore since the underlying code was not entirely brilliant it was difficult to extend the codebase in a satisfactory way. We tried to keep the extension as simple and as loosely coupled as possible within the time-constraint and without reworking the underlying codebase.

Created 3 new classes:
Abstract Obstacle for common behavior, StationObstacle and JunctionObstacle.

ShopScene

Modified the shop (ShopScene) to accommodate buying obstacles, used the existing framework for buying resources. Created new background in the shop when buying obstacles. There were challenges regarding this new background image in terms of the old image with a misleading name being shared across multiple scenes. That resulted in temporarily having wrong backgrounds at wrong places in the shop.

Player

Created a method buyObstacle in the Player class, added two boolean members trainCrashed and trainDelayed for allowing to notify the player of the situation at the start of their next turn.

Train

Train class got two new members int stationsPassed and boolean delayed, former used for warning system logic and the latter for delaying a train at a station with an obstacle.
The logic for delaying and crashing a train is done in the updatePosition method which calls corresponding trainDelayed() or trainCrashed() methods in the same class. The updatePosition method was quite coupled even before adding the necessary lines for the obstacle logic. The addition of additional coupling and complexity in that method is justified with

the time-limit we have for this assessment and with not entirely understanding the existing code to be able to refactor it perfectly. To be fair, this was the only place in the code to accommodate the obstacle logic without a major rework.
The trainDelayed method sets flags at the right places to notify the player of the delay and the trainCrashed function calls methods in the GameScene to remove the corresponding train and obstacle from the game and also setting a flag to notify the user.

RouteLocation (and Station)

The superclass(RouteLocation) of Junction and Station got self-explanatory methods setObstacle, getObstacle and hasObstacle which are being called in various places. The Station class needed a more specific setObstacle method so its behavior was overridden. Station class also got additional setObstacleTurns and decrementObstacleTurns which are used for automatically removing the obstacle after 10 turns (could be any amount of turns).

GameScene

The GameScene class was already quite the “god class” and we proceeded in the same manner because of the time constraints and not being confident enough in our understanding of the codebase to be able to break it up into clear and loosely coupled bits.

We introduced a yet another member to the class called obstacles which is a list which contains all the active obstacles in the game. This was necessary for hiding the obstacles in the routing mode and making the stations/junctions visible and clickable. updateObstacles method was created to accommodate the auto-expiring functionality of StationObstacles, the method simply loops through all the RouteLocations and decrements the life-period of an obstacle if appropriate.

generateObstacle method is basically an obstacle factory that is called by buyObstacle method mentioned before, it is responsible for creating an obstacle and distributing it to all the right places so it would be included in the rendering and all the logic mechanisms.

player1Active() and player2Active() (horrible design but didn’t change it due to time constraints) were modified to look at the flags in the Player class and to push dialogs onto the screen which would notify the player about a crash and/or a delay that had happened to their trains. Depending on the flags we use crashDialog() or delayedDialog() in the same class to push the dialogs on the screen.

removeObstacle method was also added to the class to remove obstacles when they expire or when something crashes into it. removeTrain method was added for removing trains when they crash.

Detail of change made
Full implementation of two types of obstacles. JunctionObstacle - place non-expiring “explosive” at a junction, when a train crashes into it then the train and the obstacle are removed. Junction will be still usable. StationObstacle - can be placed on a Station, it is basically maintenance work at a station and it causes all passing through trains to be delayed by 1 turn. This obstacle is removed from the map automatically after 10 turns.
Reason for change made
The game needed 2 types of obstacles based on the requirements.
Affected elements
In increasing order (classes): Player, RouteLocation, Station, ShopScene, Train, GameScene Created: Obstacle, JunctionObstacle, StationObstacle
Git commits
● d6d4ab1 (Buy Obstacles) ● 1c01dbe (Obstacles at Junctions) ● 10584d5 (Trains crash with obstacles)

• ca93e30 (Station Obstacles)
Relevant testing
• System tests 1, 2, 4 & 8 • testPlaceObstacle unit test

- **Warning system**

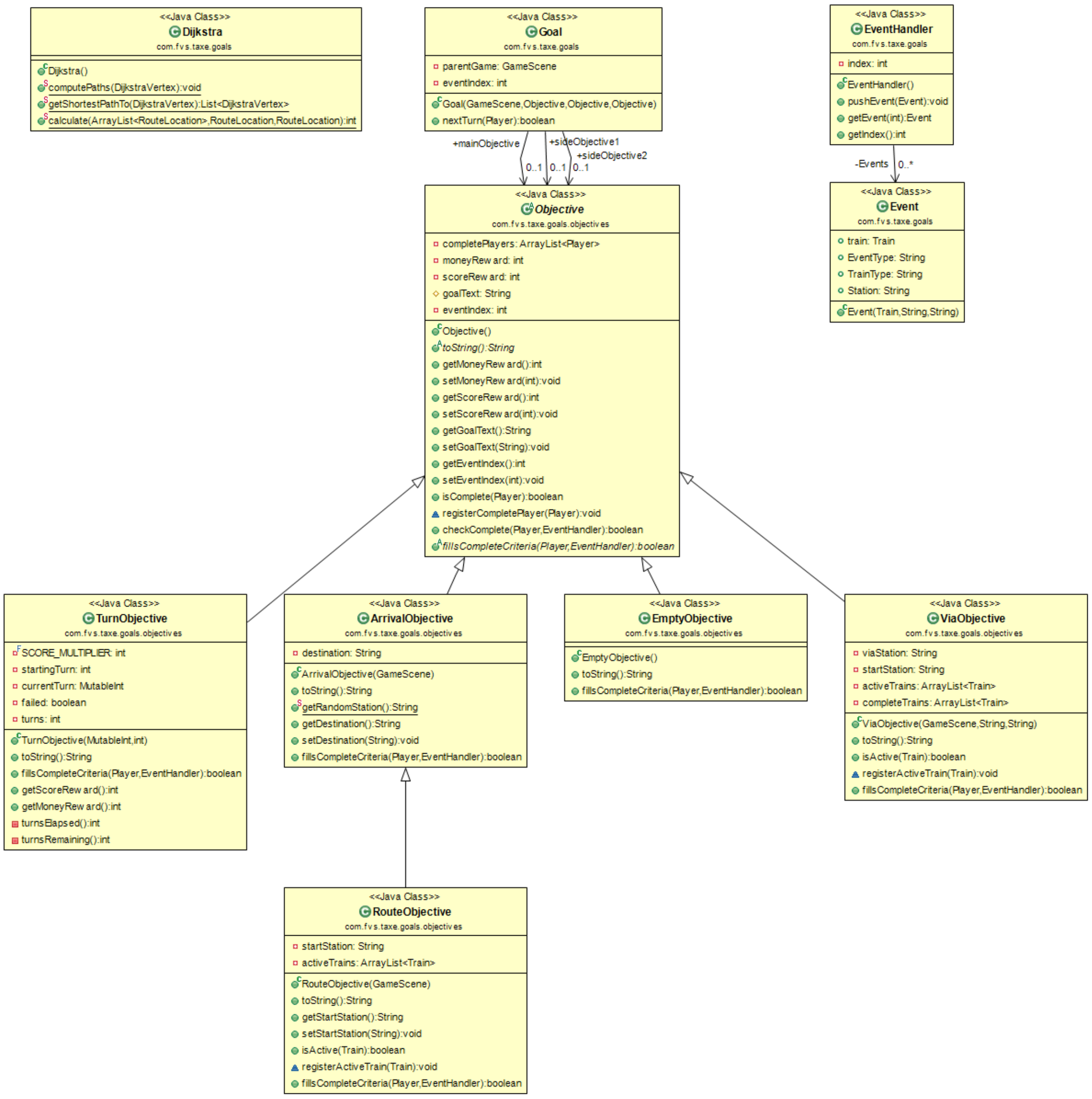
For this feature we tried to keep the code simple and put methods in places they should belong to, did not modify much of the existing code, just extended the code with additional methods.

In GameScene class player1Active() and player2Active() methods call checkObstacles() method which pushes a dialog when the player has an obstacle on the route for any of their trains. The checkObstacle() calls obstacleDialog() in the same class when appropriate. The checkObstacle() method also calls hasObstacleOnRoute() method in the Train class which we added.

Also, when ending the route selection we notify the player instantly when the newly selected route has an obstacle on it. This is done in the endSelectingRoute() method.

Detail of change made
Warning system implemented which warns the player when routing a train through an obstacle and/or at the beginning of each turn when already moving trains now have an obstacle on their route.
Reason for change made
Required by the requirements
Affected elements
Train, GameScene
Git commits
• f9f9035 (Warning when routing) • 9c29545 (Warning at start of each turn) • 3ef320c (Improve warnings)
Relevant testing
• System tests 3 & 7

- **Quantifiable goals**



(UML diagram for implementing quantifiable goals)

Detail of change made
The aim was to add quantifiable goals. The system that ADB had set up was a Goal contains a main Objective and two optional side Objectives. We added a new type of Objective called TurnObjective, which would complete and give the player extra score & money for completing the goal if they did it within X number of turns. ViaObjective was also added, which completed and gave the player additional score based on if they have been via a certain station. TurnObjective needed to know how many turns had passed since the the game had began. In terms of it being an objective, it needs to know the turn count when the objective was created, and the current turn count when it checks if it (the objective) is complete. There was an integer in GameScene which has the value of the current turn count, but when

this is passed to TurnObjective it will be passed by value and this value will never change. Changing the type of this to MutableInt[1], ensures it is passed by reference, and the objective will have means of getting the current number of turns passed.

ViaObjective is separate to, but strongly based on, RouteObjective. It employs the same detection algorithms to calculate when the via station has been passed through, and if it has been passed through after having gone through the starting station (if required).

RouteObjective would sometimes create a route asking for the player to send a train from Berlin to Berlin for example. This was an existing bug from when we inherited the code but we felt needed fixing given that we were extending the functionality of goals. The constructor of RouteObjective was generating two unique stations, but wasn't storing the second station as the destination, it turns out this value was being set by the superclasses' constructor.

Reason for change made

The Objective class and some of its methods were changed to abstract. This makes the contract of what a new subclass of Objective has to implement much clearer.

Each Objective subclass contained a public static method called generate() which contained code which instantiated the the class it was contained it. The previous group's reasoning for this was unclear so we refactored by removing these methods and moved any initialisation data it contained to the subclasses' constructor.

Affected elements

- Objective and all its subclasses, goal and slight changes to GameScene
- GameScene numberTurns type changed to MutableInt

Git commits

- X to X bug fix commit
- Objective refactor

Relevant testing

- Unit test in GoalTest class
- System tests 5 & 6

- **Scoring**

Detail of change made

Addition of scoring methods to all objectives and goal to reward player (including Dijkstra's algorithm to ensure appropriate scores).

ArrivalObjective and RouteObjective have scores based on Dijkstra's Algorithm[2]. This was implemented using a freely available version of the algorithm online, with all stations and junctions being converted to vertices and the connections between them as weighted, directed edges. The score and monetary reward are both based on the minimum distance calculated from the source station (Taken as the initial station for Arrival Objectives) to the destination station. The monetary reward is simply this minimum value. The score reward has a multiplier applied based on if it stipulates a source station or not.

TurnObjective offers scoring based on the number of turns it has taken to complete a goal. For every turn it takes the player to complete the main objective the TurnObjective score is reduced. Monetary reward is offered as an all or nothing based on if the objective is met.

[1] Apache Commons MutableInt, [2] Dijkstra's Algorithm

ViaObjective is scored using the difference between the two routes multiplied by 2. This increases the competitiveness and playability by allowing the tradeoff between speed and points. In case that both paths are the same length a score of 50 is awarded. No Money reward is offered.
The overarching Goal class, which contains the objectives, assesses the overall score for the goal and gives the total score and points which are then added to the players current totals.
Reason for change made
Scoring is a requirement of this assessment. More complex methods of scoring than required were implemented to improve the playability and enjoyment of the game.
Affected elements
All classes within the objective package, Goal and GameScene
Git commits
● a2ce260 (Dijkstra)
● 570efe7 (Allocation of scores to goals)
● 6e6f976 (Update player score)
● f819aad (Prevent score from going to max for arrival goals)
● a92cdd9 (Via Objective)
Relevant testing
● System tests 5 & 6
● testScore unit test

- **General enhancements**

The following changes were made as general enhancements…

Detail of change made
Disabled window resizing by preventing dragging window size and using maximise system commands.
Reason for change made
Background graphics from the inherited codebase were scaling with the window (sometimes causing distortion) but the hard-coded click boundaries were not, resulting in graphics and click regions being offset from one another. This was the simplest and fastest fix to the problem.
Affected elements
DesktopLauncher
Git commits
● 2602567 (Prevent resize)
Relevant testing
● System test 9

Detail of change made
Automatically determine the correct button to use, instead of a hardcoded “In Use” button, to provide some meaningful feedback. The screen provides information as to if the Train is “In Use” (Owned and on a track) or “Available” (Not owned).
Reason for change made
The screen was using purely hardcoded values, meaning the screen was provided no actual information. It was intended that the screen would offer a full overview of what the player owned and how the resources were being used. It, however, proved difficult to do this because pre-rendered graphics are difficult to update and it would have required modifying several classes to expose the required information for more detailed information to be made available. It was decided that this would take too long. It was decided that the Obstacles screen within the same Scene would not be implemented due to it taking a long time to implement, due to graphics development being required, and the information provided would not offer any great improvement over what is already visible on the map or via the shop window.
Affected elements
CurrentResourcesScene
Git commits
• ae63c93 (Current Resources)
Relevant testing
• System test 10

Detail of change made
Bitmap fonts were being generated and then disposed multiple times per scene load. After a font was generated (which took ~50ms) it wasn’t being re-used, and in some scenes different fonts are required upwards of 20 times. The change was to memoize the fonts. So the second time the game requests a font which has already been generated, the method looks in the dictionary using a Tuple of font size and colour as the key and returns the previously generated font.
Reason for change made
The game was loading very slowly, when the player switched scenes or even tabs within a scene (e.g. tabs of complete/incomplete goals) there was a delay of 500ms+ which affected playability.
Affected elements
Label::genericFont() method.
Git commits
• Memoize Font commit
Relevant testing
We added a simple Stopwatch class (core/src/util) to measure the time spent on generating fonts before and after the change. We also performed a system test as a group to check that everyone perceived the speed increase. It was observed that the time taken to switch between goals tabs reduced from 0.9 to 0.01 seconds.

Detail of change made
Improved name entry.
Reason for change made
This screen is one of the first seen by the player and we felt it was important that it worked intuitively in order to make a positive first impression.
Affected elements
The EditText class was amended to only clear the text box when filled with the default 'Name' text, to prevent the erasure of user-entered text. A keycode shortcut was added so that when the tab button was entered, the focus switches to player 2's name entry box. (Note: as this is the only instance in which the EditText class is used, it is now assumed that this EditText class refers specifically to a name entry text box and modifications may wish to be made if used elsewhere)
Git commits
● 0c2681d
Relevant testing
● System test 11

Detail of change made
Removal of carriages from trains
Reason for change made
When weighing up the amount of work required to make the carriages work properly, and then to ensure that they continued to work with our extensions (in particular obstacles); we found that the amount of work would be disproportionate to the visual benefits of having carriages.
Affected elements
Total removal of Carriage and CarriageTest classes. Deletion of carriage image files from assets. Removal of all references to carriages in TrainTest, GameSceneTest, Player, AISprite, Event and Train classes.
Git commits
● 9d41382 ● 046d643 ● 65d262d ● 2f6b218
Relevant testing
● System test 12

Detail of change made
Added proper buttons for routing.
Reason for change made
The original routing controls were very hard to see and didn't fit into the style of the GUI, as these controls are frequently used we decided to change them to make them more user friendly.

Affected elements
Added 3 new button texture images to assets, updated GameScene class.
Git commits
● 3320bc7
Relevant testing
● System test 13

Detail of change made
Enhanced the map quality and made game screens consistent.
Reason for change made
The original map image had been heavily compressed and was of visible poor quality, to make the game more visually appealing to the user we made a higher quality version using the Photoshop file provided to us. Additionally, upon opening menus, the control icons and the map in the background were inconsistent, to add consistency and attention to detail we replaced those images too.
Affected elements
Updated map image and all game screen images in assets.
Git commits
● ae4f3c3 ● 56b29a3
Relevant testing
● Observed

Detail of change made
Made the in-game currency consistent
Reason for change made
We found that the mismatch of dollars and ‘cr’, assumedly an abbreviation of ‘credits’, was inconsistent but also easily confused with the scoring system of ‘pts’, or ‘points’.
Affected elements
Unfortunately the custom font already used exclusively throughout the game, ‘GOST’, did not contain a Euro symbol. However as this seemed the most logical in-game currency for our TaxE game we added a custom made Euro symbol, in place of the dollar symbol, to the font - this also saved having to change the font in images used throughout. All instances of ‘cr’ were therefore replaced, affecting the Goal, GoalsScene and ShopScene classes. Additionally the money icon was altered in assets.
Git commits
● 2aa4312
Relevant testing
● System test 14

Detail of change made
Introduce a winning condition for the Game (tested at the end of each turn) based on if a player has above 6000 points. When a player wins they are notified by an onscreen message (regardless of whose turn it is). The game then ends and returns to the Main Menu where a new game can be started.
Reason for change made
The game did not previously end, which seriously impacted on the playability of the game and left players lacking an end motivation. Introducing this makes the game more playable and enjoyable. The game ends when the message is shown as allowing players to continue in an endless mode would have meant that it would be harder to include information on the Leaderboard. It was decided not to implement the Leaderboard at this stage as it relies on some pre-rendered graphics which were not suitable for the ending method chosen. It would also have required data being saved on the users machine and accessible to the game. This would have required considerable time to implement, given the current save and load game systems to not function correctly.
Affected elements
GameScene
Git commits
● fa073d8
Relevant testing
● winTest Unit test

Detail of change made
As per the specification, modify the game so that a player acquires 2 resources on each turn. Each player now receives an additional 5 fuel and 5€ at the start of each turn. This will also prevent the possible situation where a player can have no money or fuel and so is unable to proceed.
Reason for change made
The game did not previously meet the specification that 2 resources should be acquired by the player each turn. It is important that the game meets the initial specification and so this change was made.
Affected elements
GameScene
Git commits
● 5369e21
Relevant testing
● testTurnResources Unit test