Design Document

Version 2.0 – 2023.12.05

Created 2023.10.26

Group 65

Andrew Nic Cole
Devin Mendes
Chau Nguyen
Amber Richardson

GitLab Repository:

https://mcsscm.utm.utoronto.ca/csc207_20239/group_65

SECTION 1: PROJECT IDENTIFICATION

During this project, we wanted to address accessibility for people with visual impairment (traversability and alt text, and enlarging the image), cognitive impairments (a guiding NPC, statistics, and a record of the player's progress), and people with limited mobility (buttons to move rooms).

Another goal of this project is to add more depth and increase engagement to the game, through new ways to collect objects and move rooms.

SECTION 2: USER STORIES

Name	ID	Owner	Description	Implementation Details	Priority	Effort
Game modes	1.1	Amber	As a developer, I want to create a section on the GUI that lists game modes, so that players can alter how they move rooms.	Add a list of radio buttons that correspond to game modes. When one of the buttons is selected, that game mode is enabled, and its features are added to the game. Additionally, the AdventureGame class will be refactored to implement different movePlayer methods	1	2
Statistics	1.2	Amber	As a player that wants to explore all endings of the game, I want to be able to view statistics related to the rooms I've been in and the possible rooms and objects I've yet to discover.	Add a button that when clicked displays a pop-up window that contains statistics such as the total rooms visited, unique rooms visited, room visited the most, the number of rooms in the game and the number of objects in the game.	1	1
Object Types	1.3	Devin	As a player that wants more ways to interact with the game, I want variety with how I obtain objects so that the game is more interactive.	When an object is created it will be given a random type (runner, treasure, basic) that will make object more interactive upon pick up. Runner: when you try to pick it up 60% chance it will move to another room. Treasure: must solve word puzzle to pick up. Basic: a normal object	1	1
Command Buttons	1.4	Nic	As a player, I want to press a button to select what command to use to play the game without the ability or need to type.	Present one button for each command that the player can use. This can be in addition to the text box or in place of the text box depending on setting toggles.	1	1

			Accessibility: Limited mobility, those who primarily use mouse. Also for language barriers since it is easier to recognise words than recall them.			
Zoom Option	1.5	Chau	As a player with limited vision, I want an option to zoom into the graphic visualization of the room I'm currently in to have the view and understanding of the room as a normal-sighted person.	Add a button below the room image, which upon mouse click will open to a separate ScrollPane pop-up window where user can zoom in/out and move around the room image to see more details.	1	1
Game Mode: Chaotic movement	2.6	Amber	As a player that has become bored of the game, I want to start a new game where I am always sent to random rooms when I attempt to move rooms, so that I can explore new paths within the game.	If this game mode is selected at the start of a game, when the player moves rooms, they are always sent to a random room in the PassageTable, instead of in the direction they specify. The user must have any necessary keys to enter a randomized room.	2	1
Game Mode: Always Trolled	2.7	Devin	As a player that seeks a challenge, I want to face a troll every time I want to move to another room. This increases the difficulty of the game and increases enjoyment of the game.	When this game mode is selected, when the player tries to move to another room, they will always have to face a troll, even if the Passage chosen doesn't contain a troll.	2	2
Informative NPCs	2.8	Nic	As a new player, I want to be told tasks for the game to have a motivation/goal and understands the game faster.	Using the "Talk" command will bring the player to a room where description has the NPC's speech. The room has a forced movement back to the original room.	2	3
Informative Objects (was not implemented)	2.9	Nic	As a player, I want to read a riddle when I pick up a scroll to know what to do next.	The scroll will have a type "informative". Object types will be implemented in (1.5). When the object is picked up, its text will appear and	3	2

				be read in the text box below the room image.		
Visual Path	2.10	Chau	As a player, I want to visualize the overall path of the rooms and objects I have been in to better navigate larger games	Add a button which upon mouse click will open to a separate GridPane. This will get updated with the player's distance (the rooms in the order of visit by player) and displacement (actual progress of players' movement).	2	3
Traversal and Alt Text	3.11	Amber	As a player with low vision, I want to traverse the inter-actable elements of the game with the tab key and read their alt text so that I can access all the features of the game and understand what input I should provide.	Ensure all buttons, text fields and text has alt text, and is traversable with the tab key.	3	2

Acceptance Criteria:

Name	**ID**	**Acceptance Criteria**
Game Modes	1.1	- Buttons appear on the GUI corresponding to each game mode implemented. - A player can only select a game mode on a new game (i.e. they haven't input any commands). - A player can only select one game mode at a time. - If a button is selected, the code corresponding to the selected game mode is used when the player moves rooms. - If the user changes their selection of game mode while still on a new game, the code corresponding to the last selected game mode is used.
Statistics	1.2	- If the button is clicked, a popup window displays the statistics - The total rooms and objects correspond to the number of rooms and objects in the game. - If the user moves rooms, the total rooms visited number is increased by 1

		- If the user moves to a new room they've not yet been in, the unique rooms visited number increases by 1 - If the user moves to a room, and they've visited it more than the room that is currently visited the most, the current room is now the room visited the most - A11yAC: each statistic is traversable using the tab key so it can be read by a screen reader
Game Mode: Chaotic Movement	2.6	- If the user moves rooms, they are moved to a room in the current room's PassageTable. This room is not necessarily in the direction they specified. - If a room in the current room is only accessible with a key, the user is not moved to this room if they do not have the corresponding key. - If a forced room is encountered, the user is moved through rooms based on RegularMovement implementation, until the game ends or they are no longer in a forced room.
Game Mode: Always Trolled	2.7	- If the user tries to move rooms, they will encounter a troll - If the passage, they choose already has a troll they will only encounter one troll - No troll to face when moving because of "Forced" moves
Object Types	1.3	- All objects are given a random type - Runner: when moving to a different room and tell the player where it went - Runner: objects will not move into a room the player can't - Tressure: upon request to pick up item, prompt the player with a random descramble word game - Tressure: if they solve the word, add the object to player inventory, if then dropped and picked up there will be no puzzle - Tressure: if fail, unlimited tries, if abandoned and then picked up prompt the player with a new random descramble word game - Basic: act like a normal object
Command Buttons	1.4	- All command options have buttons that appear - No commands have multiple buttons (Up, North, N should have 1 button) - No command buttons should appear but be unusable in a room - Commands can still be inputting in the text box with synonyms.
Informative NPCs	2.8	- The talk command can be used in rooms with an npc - After using the talk command, the player is sent to a room with the Npc's dialogue - The player is returned to the original room after the dialogue completes - The dialogue hints at what the player should do next.
Informative Objects (not implemented)	2.9	- The scroll can be picked up by clicking its button or inputting a TAKE command - When the scroll is picked up, its message is displayed and read. - The message hints at what the player should do next, something that the player has not already done

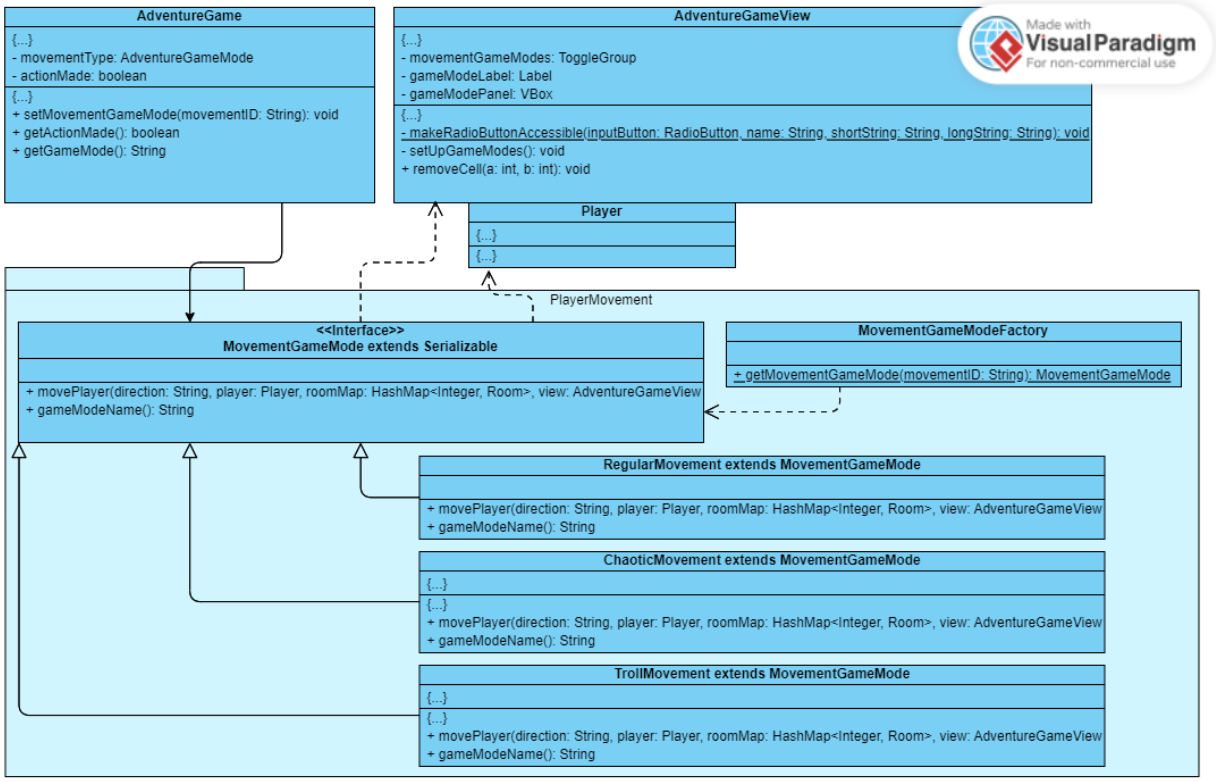
Zoom Option	1.5	- Button appears below the room image - Upon a mouse click, a separate ScrollPane will open with the room image, 2 buttons on the side for zooming in/out, and a scroll bar on the two sides of the image for users to move the zoomed-in image up/down and sideways to get to all corners. - When user close the zoom ScrollPane, the room image should appear as normal
Visual Path	2.10	- Button appears on the top of the GridPane of the game which upon mouse click will open to a separate pop-window. This will get updated with 2 panes: the distance and displacement of the player. In particular: ○ Distance: all the rooms in the exact order of visit by the player. This allows for repetition. e.g., If player visits A -> B -> A, display A -> B -> A ○ Displacement: The actual progress made by the player so far in the game. This does not allow for repetition, and only display the rooms that user has advanced into. e.g., ○ if player moves A -> B -> C, display A -> B -> C. ○ if player moves A -> B -> A, display A. ○ If player moves A -> B -> C -> A -> D, display A -> D
Traverse and Alt Text	3.11	- All interactable nodes on the screen are traversable with appropriate alt text. - The image is traversable and provides a description of the image - The label for the game modes is traversable and informs the user to select a game mode on a new game, or informs the user of the game mode otherwise - The panels containing the objects in the room and the objects in the player's inventory are traversable, and have alt text that informs the player that the following buttons are the objects in the room and inventory, respectfully - When pressing the tab key repeatedly, the order of traversal is left to right, top to bottom of the main game mode panel

SECTION 3: SOFTWARE DESIGN

Design Pattern #1, 1.1: Factory Pattern

Overview: This pattern will be used to implement game modes that will implement new ways to move the player between rooms.

UML Diagram:



Implementation Details: The UML diagram outlines these main components of the Factory Design Pattern

- The MovementGameMode acts as the "Product" interface
- Concrete implementations of MovementGameMode (RegularMovement, ChaoticMovement, and AlwaysTrolled) are the "Concrete Products" that are varying implementations of how the user will move rooms
- The MovementGameModeFactory acts as the Creator, that creates and returns the Products. This implementation returns Products based on their MovementID, that corresponds to an ID of the buttons the user selects on the GUI

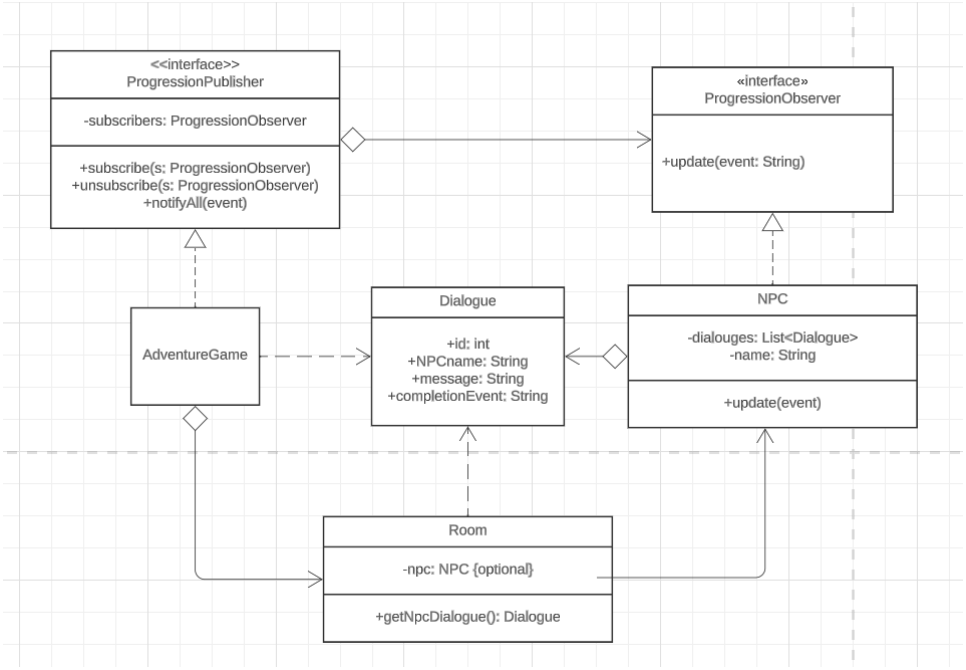
Use for User Story:
The new getter and setter for actionMade will allow the GUI to lock and unlock the movementGameMode ToggleGroup, so that players can select a game mode only when they've not yet moved rooms or interacted with objects. When the players have started the game (entered a movement or interacted with an object), the MovementGameModelFactory first sets up the correct game mode based on the user's selection and this is saved to movementType. Then, any room movements are based off the implementations of the MovementGameMode that is saved in movementType.

Design Pattern # 2, 2.7 Informative NPCs: Observer

Overview: There is an NPC in a room near the start who the player can talk to for hints on how to proceed with the adventure. To effectively guide the player, the npc's message will depend on the player's progress, measured by which of the necessary objects the player

has found.

UML Diagram:



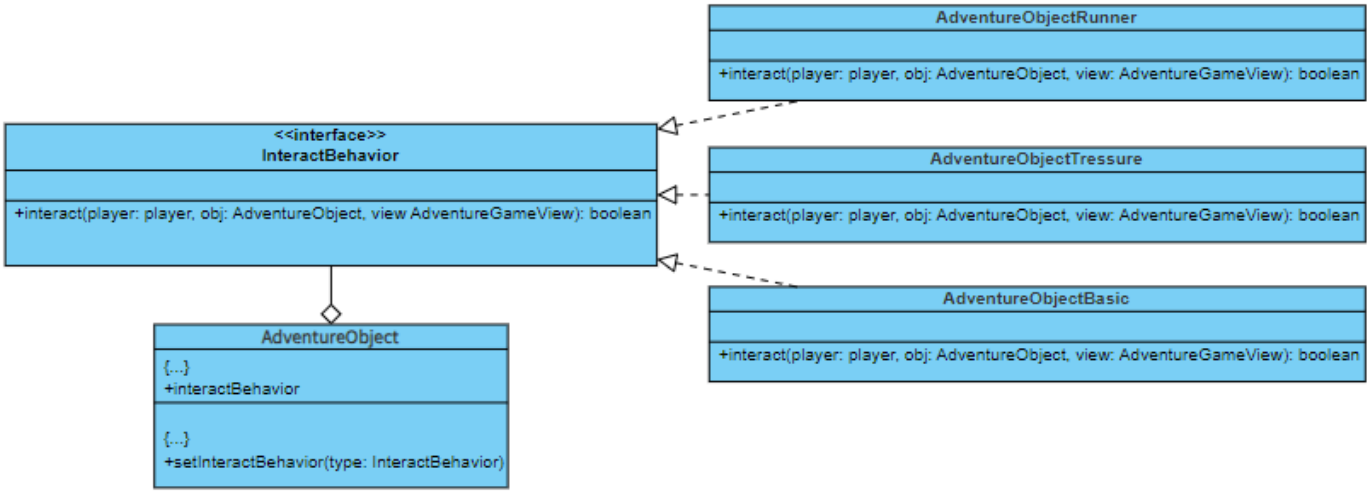
https://refactoring.guru/design-patterns/observer

Implementation Details:
For NPCs that are intended to help guide the player, they all need to be updated on what the player has done to know what to advise them to do next. For this reason, the observer pattern is used. When a player reaches a room or picks up an object, the AdventureGame class will update all of the NPCs since they are ProgressionObservers. Since the game follows a linear progression, the Dialogues are stored in a priority queue based on their order in the game progression. When an NPC is updated with an event, it removes any pieces of advice with that completionEvent.
When a player is in a room with an NPC, they will be able to use the TALK command which displays text of the NPC's advice and plays an audio saying the same thing.

Use for User Story:
As the player progresses, they reach NPCs who will give them an idea of what to do next. The players are also able to return to NPCs for new advice when they have progressed but got stuck.

Design Pattern # 3, 1.4 Object Types: Strategy

Overview: Upon the creation of an object, it will be given a random type. When the player tries to take the object, the object will interact with the player depending on its type.



Implementation Details: The UML diagram outlines these main components

- The InteractBehavior interface that contains the method: interact
- Three concrete classes Basic, Tressure and Runner that all implement the InteractBehavior interface
- The *AdventureObject* class and how it uses the InteractBehavior interface

The interface "InteractBehavior" contains one method interact, each object type implements and defines how it will interact will the player. All interact will return true if the player is able to take the item and false if they can't. The "AdventureObject" class contains an attribute "interactBehavior" that defines what object type it is and method "setInteractBehavior" that take in a InteractBehavior object and changes how the object interacts according to "interactBehavior".

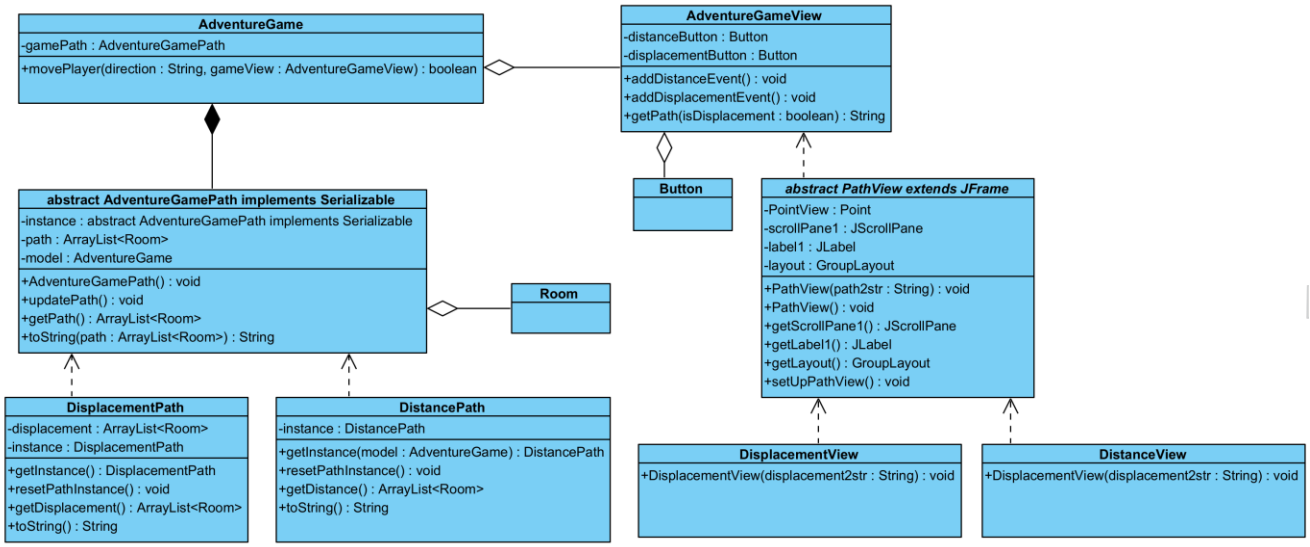
Use for User Story:

When the player takes objects, a behavior can be assigned to it, to give the player a new way to interact with it. This will add variety to the interaction of gathering objects.

Design Pattern 4, #2.9: Command

Overview: Whenever player moves to a new room, the visual path will be updated accordingly to record the player's movement and progress.

UML Diagram:



Implementation Details: The UML Diagram outlines these main components:

- The following new concrete classes in addition to the original AdventureGame: abstract class PathView and child classes DisplacementView, DistanceView; abstract class AdventureGamePath and child classes DisplacementPath, DistancePath
- The AdventureGameView, AdventureGame and Room classes and how they interact with the new classes.

Use for User Story:

1. AdventureGamePath:

- The AdventureGamePath is an abstract class implementing the Serializable interface that has a composition relationship with the AdventureGame class. It creates and stores the path that users have been on as they play the game. Thus, an AdventureGamePath object is dependent upon the AdventureGame object to get information on whether the player has moved to a new room to update path, or restarting an entire new game to restart the entire path
-
- **Child Class DistancePath inheritting from AdventureGamePath:** this class is responsible for the logic behind the "Journey thus far" button: it displays all the rooms that the user has been in every time they move rooms, regardless of whether that room has been visited before.
- **Child Class DisplacementPath inheritting from AdventureGamePath:** this class is responsible for the logic behind the "Progress thus far" button: it displays only the rooms that the user has never been in as they progress through the game. Therefore, if user enters a room that they have been in before, all rooms between the last instance and the latest instance of the duplicated room will not be recorded.

This is demonstrated in the getDisplacement() method, which clearly illustrates the "undo" command and thus reflect the **Command** design pattern.

- Method **"toString"** returns a string representation of the path. This method creates a string with HTML wraps compatible with the JLabel of the PathView class.
- **Accessibility:** both of the child classes helps to record the journey ("distance") that users have made so far, as well as the actual progress ("displacement") that users have made. This would tremendously help those with cognitive impairment to better visualize the scope of the game, better memorize the movements around rooms, and avoid those that lead to the undoing of progress. Therefore, this would promote engagement with the game and make the overall gaming experience more enjoyable.

2. PathView

- PathView is an abstract class extending JavaFX JFrame. It creates the popup window when user clicks on the "Journey thus far" or "Progress thus far" buttons. This class has an association relationship with the AdventureGameView, as the AdventureGameView contains Button objects with methods addDistanceEvent() and addDisplacementEvent() that call on the PathView object.
- Child classes DisplacementView and DistanceView inherits most of the GUI implementation from PathView, with the minor exception of denoting whether it's Journey or Progress. Therefore, it's a better design choice to implement all the inheritable methods in PathView instead of only writing method signatures in an interface.
- **Use for User Story and Accessbility:**
  - PathView extends JFrame, which allows users to have multiple popup windows open. This helps when users want to have a side-by-side comparison between all the movements they have made via the "Journey thus far" window and the actual progress they have made via the "Progress thus far" window. Therefore, this feature promotes accessibility for those with visual and cognitive impairments (e.g., memory issues).
  - The formatting of the string representation as implemented in method pathView() is set to be large, which helps those with visual impairment. In addition, since the large formatting leads to overflow when the path length exceeds that of the popup window' dimension, a JscrollPane containing the JLabel displaying the text is implemented. This allows users to easily scroll through the entire path, helping those with visual impairment.