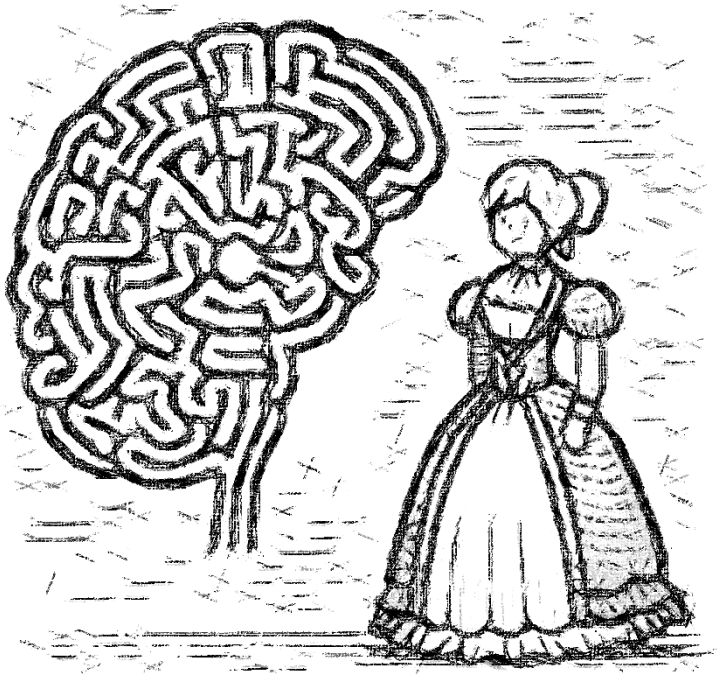
MMI716 – Game Programming Patterns

Project Final Phase Report



Burcu Alakuş
2459410

Table of Contents

List of Figures

1. Introduction

This final report provides a comprehensive overview of the completed game programming project. It begins with a summary of the initial concept and objectives set in Phase I, highlighting the core ideas that guided the development process. The report then details the key game components used, such as physics and audio, and how they enhance the overall gameplay experience. Additionally, it discusses the design patterns applied throughout the project, explaining their roles and benefits within the game's architecture. The report also includes an assessment of the project's progress, highlighting the milestones achieved and any challenges encountered. Finally, it reflects on the lessons learned and assesses the overall success of the project.

2. Project Summary

The game is a first-person exploration game centred around a female painter who has lost her memory. Players navigate through various rooms, each capturing a memory. The primary objective is to find specific items that trigger her memories, blending puzzle-solving and storytelling to provide an emotional exploration of her past. The game delves into themes of memory, identity, and creation, offering deep gameplay and significant emotional depth. Due to the lack of available assets related to the painter’s theme, the narrative has been adjusted. The protagonist is now a governess, not a painter. This change allows for better utilisation of existing resources while maintaining the core elements of puzzle-solving and storytelling.

3. Game Components

In the discussion of my game components, I will go into the interactive elements that constitute my game’s environment, such as the house generation system, which leverages a binary space partitioning and flyweight patterns to intricately layout houses with rooms and corridors, and the

integration of physics engines for realistic object interactions, as well as assets, ghosts, environmental components that breathe life into the game.

3.1. House Creator

I've implemented a procedural generation technique using a house generator based on the binary space partitioning (BSP) algorithm in my game. I was not aware that BSP is also a design pattern until we have learnt about it in the classroom. This method is a systematic approach to dynamically create a complex layout of a house complete with rooms and corridors.

Binary space partitioning is a recursive division of space. Imagine it as continually splitting a space into subsections using imaginary, straight lines (or planes in 3D). In the context of my game, this space represents the interior of a house. The initial space is the entire area of the house, which the BSP algorithm divides into smaller sections, each potentially becoming a room or a corridor. The one of the outputs of this script is as in Figure 1, I have chosen this layout as my house plan. The parameters for creating a house can be seen in Figure 2.

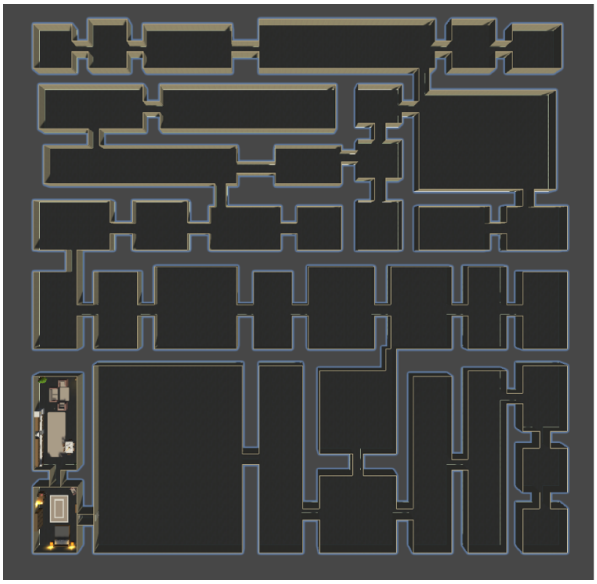


Figure 1. Top view of the house.

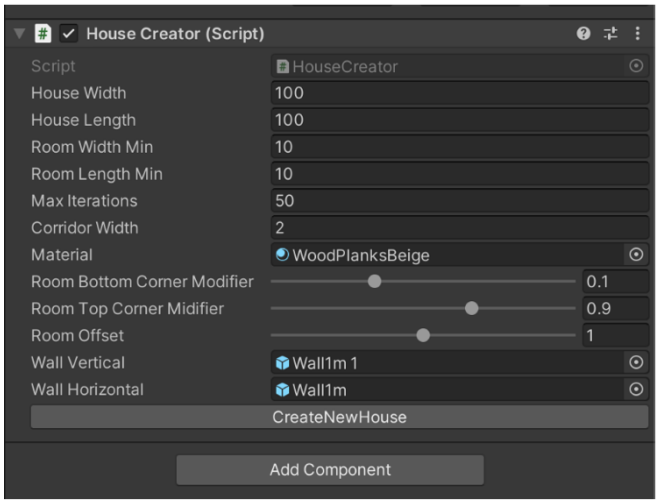


Figure 2. HouseCreator script parameter view in Unity UI.

By setting parameters such as the minimum and maximum size of a room, I can control the outcome of the generation to ensure that the spaces created are usable within the game's context — not too small to move around in and not so large that they don't resemble realistic rooms. Once the

algorithm has partitioned the space into a satisfactory layout of rooms and corridors, the maze generator comes into play. It carves out pathways between the rooms, ensuring there are no inaccessible areas. These pathways are the corridors of the house, and the logic of the maze ensures that they interconnect the rooms in a complex, yet navigable, manner.

3.2. Assets

In crafting the environmental and aesthetic dimensions of my game, I've incorporated a suite of no-cost assets that greatly enhance the visual richness and ambience. These assets, sourced from communal pools and open-access platforms, encompass textures and room designs that contribute distinct atmospheres to various game settings. I've employed available 3D models (a sample is in Figure 3) to furnish the game with interactive elements and structures, while leveraging tile sets for creating diverse terrains, thus bypassing the need for extensive bespoke graphical design.

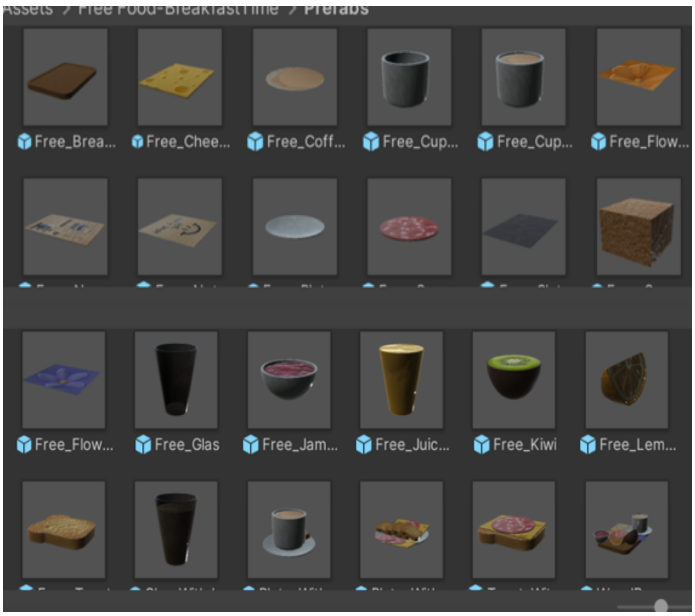


Figure 3. Assets sample from Unity UI

3.3. Player

The player component in the game is equipped with a suite of scripts that manage various aspects of gameplay. The FPS Controller script (see Figure 4) adjusts the player's movement and camera control, setting how it walks, runs, jumps, and looks around within the game's environment. There's also an inventory script that handles the items (see Figure 5) the player collects, storing them

and enabling their use at key moments. An item collector script allows for interaction with collectable objects, adding them to the player's inventory. Lastly, a flashlight controller script manages a flashlight's functionality, critical for navigation and interaction within darker areas of the game. Together, these scripts define the player's interaction with the virtual world, making for a cohesive and engaging gameplay experience.

Figure 4. Player script parameters in Unity UI

Figure 5. Some items in the game with 'collectable' tag

3.4. Ghosts

In the game, an engaging component of the environment is the presence of ghosts (see Figure 6), which are programmed to enhance the thrill and challenge of gameplay. These spectral entities are designed to materialize when the player acquires certain items, triggering their appearance as if awakened by these objects. Once active, they pursue the player, adding a layer of urgency and danger

to the navigation of the game's spaces. The ghosts emit eerie sounds that contribute to the haunting atmosphere, heightening the sense of suspense and immersion. However, their presence isn't permanent; the ghosts are coded to vanish when the player has either put enough distance between themselves and these apparitions or after a set duration, ensuring that the game remains balanced between challenge and playability. This mechanic not only adds depth to the gameplay but also serves as a motivator for players to continue exploring and interacting with the game world, pushing forward to uncover more of the game's mysteries.

Figure 6. Different ghosts to spawn

3.5. Doors & Keys

Within the game's design, a classic yet effective gameplay mechanic is the incorporation of locked doors and the requisite keys needed for their opening. This component plays a strategic role in guiding the progression through the game. Some rooms within the game environment are inaccessible from the outset, locked, prompting a search for the keys (see Figure 7). This search encourages exploration and interaction with the game's environment as players must navigate through the rooms and corridors, all while potentially evading hazards such as ghosts. This mechanic not only serves to structure the player's journey through the game but also adds layers of engagement and problem-

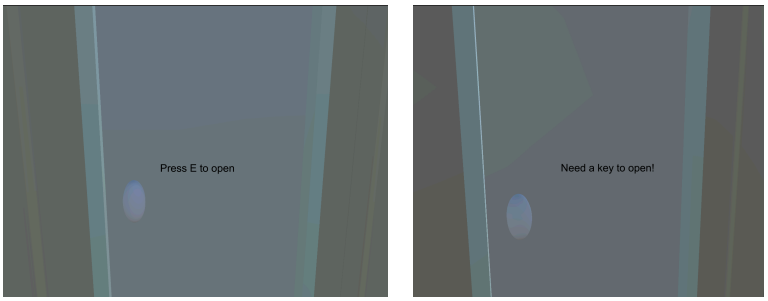


Figure 7. Door state instruction in gameplay

solving, as players must remember which doors they have encountered and return once they have found the correct keys. It’s a balance between exploration and reward, providing a satisfying sense of accomplishment as previously locked areas become accessible and reveal their secrets.

3.6. Environment Lighting

To craft a distinctive atmosphere within my game, I've chosen to forgo the use of a traditional skybox material. Instead, I've embraced pervasive fog throughout the game environment. This design decision amplifies the enigmatic and perhaps claustrophobic ambience, focusing the player's attention on the immediate surroundings and creating a sense of isolation. The absence of a skybox means there are no distant vistas or celestial bodies to provide orientation or distraction, thereby heightening the player’s immersion in the game world. The fog not only obscures distant objects, adding a layer of mystery and suspense but also plays a practical role in optimizing game performance by reducing the rendering load of distant scenery. This combines aesthetic intent and technical realism, ensuring that the players remain grounded in the game's immediate narrative and visual experience (see Figure 8).

Figure 8. A screen showing how no skybox material and fog adds mystery to the environment.

3.7. Audio

The auditory dimension of my game is carefully constructed with a selection of sounds that are integral to the gaming experience. The opening of doors is accompanied by distinctive audio cues that not only signal a transition from one space to another but also enhance the realism of the environment. Scary sounds attributed to the ghosts provide an unsettling ambience, contributing to the tension and suspense that are central to the game's horror theme. Collecting items triggers a specific sound,

providing immediate auditory feedback to the player, reinforcing the action, and contributing to a rewarding experience. Initially, I planned to incorporate varied walking and running sounds that would change depending on the floor type the player traverses. This feature aimed to immerse players further into the game world by reflecting the physical properties of the environment and enriching the overall soundscape with a more dynamic and responsive audio system. However, due to time and resource constraints, I was only able to implement a single walking sound for the game. This limitation affects the audio diversity but ensures that basic sound functionality remains in place.

4. Design Patterns

4.1. Flyweight + Binary Space Partitioning

In the architectural design of my game, I have applied the flyweight pattern to the creation of the house interiors, such as walls and floors. This design pattern is crucial for optimizing memory usage and performance, particularly when dealing with many similar objects within the game's world. By using binary space partitioning pattern, I can efficiently generate a complex layout of rooms and corridors. This system recursively divides a given space into smaller sections based on predefined size constraints, ensuring that the resulting spaces are neither too small nor excessively large for the gameplay requirements.

The flyweight pattern approach is further refined by cloning a single prefab—an archetype of an object in the game—for various instances of walls and floors throughout the house. While each instance is a clone of the original prefab, its position is uniquely determined by the binary space partitioning process, allowing for a variety of room shapes and corridor connections while maintaining a consistent look and feel. This not only conserves resources but also allows for a diverse and engaging game environment to emerge from a relatively limited set of base components. Through this method, I'm able to create an intricate and memory-efficient house layout that is both functional for gameplay and varied enough to maintain the player's interest.

4.2. Prototype

For the dynamic appearance of adversaries within my game, I've implemented the Prototype design pattern through the 'Ghost Spawner' system. This system utilizes a ghost prefab, a template for ghost entities, which can be instantiated multiple times within the game world. The Ghost Spawner is configured to track the player's position and inventory, spawning ghosts within a certain radius once the player has collected specific items, such as a newspaper.

The spawn conditions, managed by the Ghost Spawner script (see Figure 9), ensure that ghosts only materialize under specific circumstances, adding an element of strategy to the gameplay. Players must consider the repercussions of collecting items that might trigger the appearance of these spectral foes. The Prototype pattern allows for the efficient creation of each ghost instance with unique parameters while avoiding the overhead of creating each from scratch. This method not only provides a reliable way to diversify the challenges faced by the player but also streamlines the game development process.

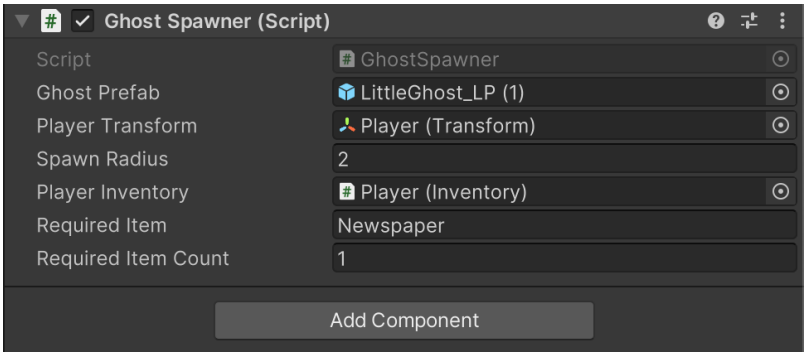


Figure 9. Ghost Spawner script parameters in the Unity UI

4.3 State

In my game project, I used the state pattern to manage door behaviour effectively. This pattern allows doors to switch between different states, like open, closed, locked, and unlocked, using separate state objects. For example, a door can move from a closed to an open state when the player interacts with it, or from a locked to an unlocked state if the player has a key (see Figure 10). By implementing the state pattern, I made the code more organized and easier to maintain. It also made it simple to add

new door states or behaviours in the future. This approach improves the overall gameplay by providing a flexible and efficient way to handle door interactions.

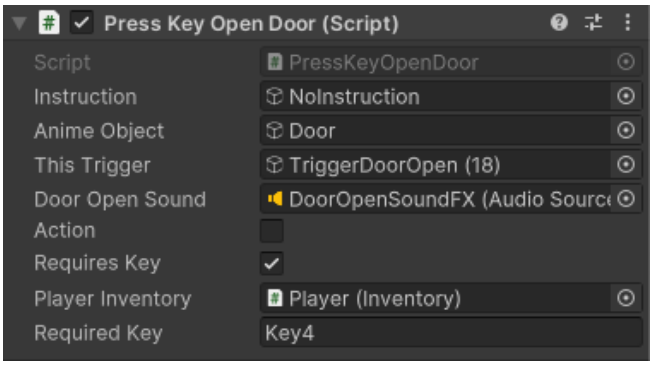


Figure 10. Door script parameters in the Unity UI

5. Progress and Completion Evaluation

The progress section of my report highlights the strides made in various aspects of the game's development. Initially, I completed the technical components, including the foundational coding structures and menus, and planned to implement the command pattern for efficient in-game command processing. The layout and design phase saw the room interiors being arranged, and necessary assets for these spaces integrated.

The current version of the game is playable and complete enough to explore, although it is not in the desired final state. The menu and "How to Play?" sections are added, but the "Settings" tab is not yet activated, which is where the command pattern implementation will come into play (see Figure

Figure 11. Opening screen & menu

11). Several design patterns, including the state pattern for doors, have been implemented to enhance the game's architecture and player experience.

Initially, I aimed to design over 20 rooms, but due to time constraints and a lack of assets, I have completed 11 rooms. The interior of these rooms has been carefully arranged, though not fully completed as originally planned shown in Figure 12. Despite these limitations, the game's core mechanics and narrative are intact, providing a solid foundation for future expansion.

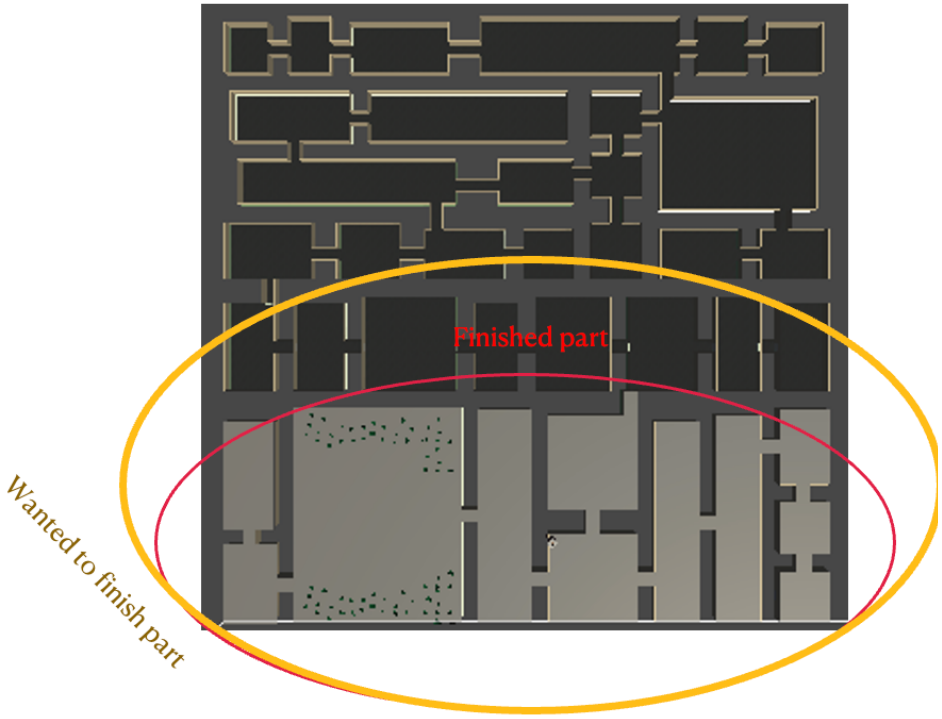


Figure 12. Room completion assessment

The story and main gameplay elements are now complete, although there have been significant changes due to resource constraints. Originally, the protagonist was envisioned as a painter, but due to the unavailability of appropriate assets, this role evolved into a governess, influenced by Anne Brontë's "Agnes Grey." This shift has not only aligned better with available resources but has also added a new dimension to the narrative.

Overall, I have completed the planned content. However, having finished the more challenging aspects, I am optimistic about progressing faster with the remaining parts. The current state of the game provides a cohesive and engaging experience for players, with ample room for further development and enhancement.