CPSC 2720 WINTER 2024

Beyond Isolation Game

P2_Group-E

Matthew MacMaster, Gurpreet Singh, Sayahang Rai

April 5th, 2024

TABLE OF CONTENTS

INTRODUCTION

PROJECT MANAGEMENT

TEAM ROLES

Team Member	**Design - Draft**	**Design – Final**	**Implementation - Basic**	**Implementation - Final**
Matthew	Phase Lead	Design Lead	QA Lead	Reporting Lead
Gurpreet	Design Lead	QA Lead	Reporting Lead	Phase Lead
Saya	QA Lead	Reporting Lead	Phase Lead	Design Lead

RISK MANAGEMENT

Early on we had much more variety for the npc class, building it very similar to the item class with a tradeNpc, a puzzle npc and more but the implementation became too vast and after discussing with Nichole it was clear we needed to address it, as it would have affected deadlines and overall code functionality. So, we ended up cutting a lot for the sake of deadlines and implemented a simpler variant that can do some of those things together.

DEVELOPMENT PROCESS

CODE REVIEW PROCESS

-This process overall was a combined effort between ensuring we follow eachothers expectations of the classes via communication and the uml we done. We maintained branches separately for the most part, creating new ones only after major merges into main which only happened a select few times throughout the project.

COMMUNICATION TOOLS

-For communication we used discord as it allows for code snippets and easy communication.

CHANGE MANAGEMENT

-Our overall implementation of the game has most of our ideas put into it and implemented, along the way we ensured it was implemented in a manner to allow ease of addition of more item types, as well as a relatively addition to allow for more players and even playing as the opponent if someone wishes to without it much to rewrite.

CLASS DESCRIPTIONS

Characters: An overall abstraction where we split possibly playable characters, and non playable characters allowing us to split actions like movement for characters we do not wish to be able to move.

Players: A derived form of characters, this splits our differences between main player and opponent. This allows for us to easily allow more players in the game, or more opponents along with being able to develop them separately, for example the move options are player input for main player but are randomly selected for opponent.

Npc:This is a derived class of character and NOT player, as it removes the major actions that can be taken normally by players but allows for inventory to still be derived.

Main Player: This acts as the user and is the only way the game prompts the player for inputs, gives an action list to which it will reference their attributes, as well as their current environment.

Opponent: Functionally identical to main player with minor changes to allow for automatic play without user input. Functions as the main threat in the game, outside of resource needs.

Item: A interface for the creation of subclasses, allows for ease of new items to be made and allow small modifications to existing classes without rewriting them individually.

Inventory: A means to store items, as well as interact with this array in a means to use, store, and trade items.

Oxygen: A derived class that sets the users oxygen to full

Pipe:A derived class that allows the user choose an adjacent location, to relocate the opponent

FlamePart: Innately doesn't do anything but if 5 are detected in a user's inventory, it removes them and adds a Flame Item.

FlameItem: Major point of defeating the alien, as it's the only damage weapon in the game.

Environment:An overall abstraction for our rooms of the map, allowing us to let the derive needed attributes like giving them inventories, but also have it be mandatory to give them actions via virtual functions so the player can interact with the rooms depending on there respective type via polymorphism.

4 Env types(hazard,puzzle,general,npc]: General serves as a no action, puzzle serves as a location for input answers for whatever puzzle requirement is present, npc contains a npc to which the action depends on the npc assigned to that location, and hazard triggers a negative effect on the player.

Game Manager: Responsible for constructing all pieces that are needed, setup the maps routes, as well as randomly generate the items and assign them to rooms while also making any mandatory pieces like the flame parts in specific locations.