เกมแอปพลิเคชัน : เกมกระโดดหลบ “Satori : Enso” บน Windows 11

**กลุ่มที่ 99**

เอกสารนี้ได้รับการอนุมัติ  
จากอาจารย์ที่ปรึกษาโครงงานแล้ว

ลงชื่อ ..............................................   
(วันที่ .............................................)  
หลังสอบเสร็จ  
ให้ลบกล่องข้อความนี้

ปริญญา ตุ้มฉาย

นภสินธุ์ ต่อศิริสกุลวงศ์  
  
หมายเหตุ เลขหน้าของหน้านี้คือหน้า ก

ตอนส่งไฟล์รายงาน PDF ใส่ปกหน้าด้วย ปกไม่ต้องมีเลขหน้านะครับ  
<http://pioneer.netserv.chula.ac.th/~achatcha/2301499/Cover.docx>

แนะนำว่าไปรวมไฟล์ (merge) PDF ปก กับ PDF รายงานทีหลังก็ได้  
หรือถ้าใส่ปกมาใน word ก็กำหนดเลขหน้า ก ให้เริ่มที่นี่ ไม่ใช่เริ่มที่ปก

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาวิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2567

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย

Game Application : Dodging Game “Satori : Enso” on Windows 11

Parinya Tumchay   
Noppasin Torsirisakulvong

A Project Submitted in Partial Fulfillment of the Requirements

for the Degree of Bachelor of Science Program in Computer Science

Department of Mathematics and Computer Science

Faculty of Science

Chulalongkorn University

Academic Year 2024

Copyright of Chulalongkorn University

หัวข้อโครงงาน เกมแอปพลิเคชัน : เกมกระโดดหลบ “Satori : Enso” บน Windows 11

โดย นายปริญญา ตุ้มฉาย   
 นายนภสินธุ์ ต่อศิริสกุลวงศ์

สาขาวิชา วิทยาการคอมพิวเตอร์

อาจารย์ที่ปรึกษาโครงงานหลัก อาจารย์ ดร.ทรรปณ์ ปณิธาณะรักษ์

อาจารย์ที่ปรึกษาโครงงานร่วม อาจารย์ โชติรส สุรพลชัย

ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับโครงงานฉบับนี้เป็นส่วนหนึ่ง ของการศึกษาตามหลักสูตรปริญญาบัณฑิต ในรายวิชา 2301499 โครงงานวิทยาศาสตร์ (Senior Project)

หัวหน้าภาควิชาคณิตศาสตร์   
(ศาสตราจารย์ ดร.กฤษณะ เนียมมณี) และวิทยาการคอมพิวเตอร์

คณะกรรมการสอบโครงงาน

อาจารย์ที่ปรึกษาโครงงานหลัก

(อาจารย์ ดร.ทรรปณ์ ปณิธาณะรักษ์)

อาจารย์ที่ปรึกษาโครงงานร่วม

(อาจารย์ โชติรส สุรพลชัย)

กรรมการ

(ศาสตราจารย์ ดร.ไพศาล นาคมหาชลาสินธุ์)

กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.สายชล ใจเย็น)

ปริญญา ตุ้มฉาย, นภสินธุ์ ต่อศิริสกุลวงศ์: เกมแอปพลิเคชัน : เกมกระโดดหลบ “Satori : Enso” บน Windows 11. (Game Application : Dodging Game “Satori : Enso” on Windows 11) อ.ที่ปรึกษาโครงงานหลัก : อ.ดร.ทรรปณ์ ปณิธาณะรักษ์, อ.ที่ปรึกษาโครงงานร่วม : อ.โชติรส สุรพลชัย, xx หน้า.

เกมแอปพลิเคชัน “Satori : Enso” เป็นเกมแนว Platform Fighter ที่ผสมผสานความท้าทายของการหลบกระสุนจำนวนมาก (Bullet Hell) เข้ากับองค์ประกอบของการแก้ปริศนา (Puzzle) อย่างลงตัว ผู้เล่นจะต้องควบคุมตัวละครเพื่อกระโดดหลบอุปสรรคและกระสุนที่พุ่งเข้ามาอย่างต่อเนื่อง พร้อมทั้งต้องไขปริศนาในแต่ละด่านเพื่อเปิดโอกาสในการโจมตีใส่ศัตรู หากสามารถเอาชนะศัตรูได้ ผู้เล่นจึงจะสามารถปลดล็อกและผ่านไปยังด่านถัดไปได้ เกมนี้ถูกพัฒนาขึ้นโดยใช้ซอฟต์แวร์ Unity และภาษา C# ในการเขียนโปรแกรม โดยมีเป้าหมายเพื่อสร้างประสบการณ์การเล่นที่ทั้งสนุก ท้าทาย และฝึกฝนทักษะผู้เล่นในด้านการตอบสนองที่รวดเร็ว การวิเคราะห์สถานการณ์ และการตัดสินใจอย่างมีประสิทธิภาพ นอกจากนี้ยังมุ่งหวังให้เกมสามารถเป็นเครื่องมือในการผ่อนคลายความเครียดจากชีวิตประจำวัน และเสริมสร้างสมาธิให้กับผู้เล่น

หมายเหตุ จำนวนหน้าให้นับทั้งหมด เช่น มี ก – ฎ และ 1 – 50 ก็นับเป็น 61 หน้า  
 (ใส่จำนวนหน้าข้างบนเป็น 61 หน้า)

ภาควิชา คณิตศาสตร์และวิทยาการคอมพิวเตอร์ ลายมือชื่อนิสิต   
 ลายมือชื่อนิสิต

สาขาวิชา วิทยาการคอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาโครงงานหลัก

ปีการศึกษา 2567 ลายมือชื่อ อ.ที่ปรึกษาโครงงานร่วม

# # 6434449023, 6434440223: MAJOR COMPUTER SCIENCE

KEYWORDS : GAMES / BULLET-HELL / PLATFORMER

Parinya Tumchay, Noppasin Torsirisakulvong: Game Application : Dodging Game “Satori : Enso” on Windows 11. ADVISOR : Thap Panitanarak, Ph.D., CO-ADVISOR : Chotiros Surapholchai, xx pp.

*Satori : Enso* is a platform fighter game that uniquely blends the intensity of bullet-hell gameplay with the cognitive challenge of puzzle-solving. The player must skillfully maneuver their character to evade waves of incoming projectiles while solving puzzles that appear throughout each stage. Successfully solving these puzzles allows the player to unleash attacks on the boss enemy. Only by defeating the boss can the player proceed to the next level. This game was developed using Unity and programmed in C#. The main objective is to create an engaging, challenging experience that enhances the player's reflexes, situational awareness, and decision-making skills. Additionally, the game aims to serve as a stress-relieving activity and a tool to promote concentration through fast-paced yet thoughtful gameplay.

Note that the total number of pages ia all pages (i-xi and 1-50 = 61 pages)

Department : Mathematics and Computer Science Student’s Signature

Student’s Signature

Field of Study : Computer Science Advisor’s Signature

Academic Year : 2024 Co-advisor’s Signature

กิตติกรรมประกาศ

ผู้พัฒนาโครงการเกมแอปพลิเคชัน : เกมกระโดดหลบ “Satori : Enso” บน Windows 11 สามารถสำเร็จตามวัตถุประสงค์ เนื่องจากในตลอดระยะเวลาดำเนินการ ผู้พัฒนาได้รับการสนับสนุน ความรู้ คำแนะนำและการช่วยเหลือต่าง ๆ จากคณาจารย์และบุคลากรต่าง ๆ เป็นอย่างดี ผู้พัฒนาจึงขอขอบคุณไว้ ณ ที่แห่งนี้

ขอขอบพระคุณ อาจารย์ ดร.ทรรปณ์ ปณิธาณะรักษ์ และ อาจารย์ โชติรส สุรพลชัย อาจารย์ที่ปรึกษาโครงงาน ที่ได้ให้ความรู้ คำแนะนำต่าง ๆ และยังตรวจสอบขั้นตอนการทำโครงงาน จึงทำให้โครงงานนี้สำเร็จลุล่วงด้วยดี

ขอขอบพระคุณ ศาสตราจารย์ ดร.ไพศาล นาคมหาชลาสินธุ์ และ ผู้ช่วยศาสตราจารย์ ดร.สายชล ใจเย็น กรรมการสอบ ที่ให้คำแนะนำและแนวคิดที่เป็นประโยชน์ในการดำเนินงานโครงงานนี้

ขอขอบพระคุณ ผู้ทดลองเล่นเกม ที่สละเวลาเพื่อทดลองเล่นเกมและมอบความคิดเห็นและข้อเสนอแนะที่เป็นประโยชน์ในการดำเนินงานโครงงานนี้

สุดท้ายนี้ ขอขอบพระคุณในความกรุณาจากทุกท่านที่ไม่ได้กล่าวไว้ ณ ที่นี้ สำหรับความช่วยเหลือและคำแนะนำต่าง ๆ ที่เป็นแรงผลักดันให้ผู้พัฒนาได้พัฒนาโครงงานนี้ให้ประสบความสำเร็จด้วยดี

สารบัญ

หน้า

บทคัดย่อภาษาไทย ง

บทคัดย่อภาษาอังกฤษ จ

[กิตติกรรมประกาศ ช](#_Toc194880402)

[สารบัญ ซ](#_Toc194880403)

[สารบัญตาราง ญ](#_Toc194880404)

[สารบัญภาพ ฎ](#_Toc194880405)

[บทที่ 1 บทนำ 1](#_Toc194880406)

[1.1. ความเป็นมาและเหตุผลของโครงงาน 1](#_Toc194880407)

[1.2. วัตถุประสงค์ของโครงงาน 4](#_Toc194880408)

[1.3. ขอบเขตของโครงงาน 4](#_Toc194880409)

[1.4. ขั้นตอนการดำเนินงาน 4](#_Toc194880410)

[1.5. ประโยชน์ที่คาดว่าจะได้รับ 5](#_Toc194880411)

[1.6. โครงสร้างของรายงาน 5](#_Toc194880412)

[บทที่ 2 ความรู้พื้นฐานและทฤษฎีที่เกี่ยวข้อง 6](#_Toc194880413)

[2.1. ทฤษฎีที่เกี่ยวข้อง 6](#_Toc194880414)

[2.1.1. Object Oriented Programming 6](#_Toc194880415)

[2.1.2. Physics in Game Development 7](#_Toc194880416)

[2.1.3. Mathematics in Game Development 7](#_Toc194880417)

[2.1.4. Rendering and Graphics in Game Development 8](#_Toc194880418)

[2.2. โปรแกรมประยุกต์ที่ใช้ 9](#_Toc194880419)

[2.2.1. Unity 9](#_Toc194880420)

[2.2.2. ภาษา C# 9](#_Toc194880421)

[2.2.3. Git และระบบควบคุมเวอร์ชัน (Version Control System) 10](#_Toc194880422)

[บทที่ 3 การวิเคราะห์และออกแบบ 11](#_Toc194880423)

[3.1. ออกแบบวิธีการเล่น(Gameplay Design) 11](#_Toc194880424)

[3.2. การวิเคราะห์และออกแบบระบบ(System Design) 12](#_Toc194880425)

[3.3. การออกแบบส่วนต่อประสานผู้ใช้ (User Interface Design) 17](#_Toc194880426)

[3.4. การออกแบบด่าน(Level Design) 19](#_Toc194880427)

[บทที่ 4 การพัฒนาและผลการทดสอบ 21](#_Toc194880428)

[1.7. 4.1 ผลของการดำเนินการวิจัยที่ 1 21](#_Toc194880429)

[บทที่ 5 ข้อสรุปและข้อเสนอแนะ 23](#_Toc194880430)

[1.8. 5.1 ข้อสรุป 23](#_Toc194880431)

[รายการอ้างอิง 24](#_Toc194880432)

[ภาคผนวก ก แบบเสนอหัวข้อโครงงาน รายวิชา 2301399 Project Proposal ปีการศึกษา 25xx 26](#_Toc194880433)

[ภาคผนวก ข คู่มือการใช้งาน (ถ้ามี) 29](#_Toc194880434)

[ประวัติผู้เขียน 30](#_Toc194880435)

สารบัญตาราง

หน้า

[ตารางที่ 3.1 กรณีการใช้งาน Start Game 12](#_Toc194880494)

[ตารางที่ 3.2 กรณีการใช้งาน Resolution Setting 13](#_Toc194880495)

[ตารางที่ 3.3 กรณีการใช้งาน Volume Setting 13](#_Toc194880496)

[ตารางที่ 4.1 คะแนนเฉลี่ยของผู้เรียน 21](#_Toc194880497)

สารบัญภาพ

หน้า

[ภาพที่ 1.1 Touhou 14 1](#_Toc194880349)

[ภาพที่ 1.2 Super Smash Bros. Ultimate 1](#_Toc194880350)

[ภาพที่ 3.1 แผนภาพยูสเคสของเกม 12](#_Toc194880351)

[ภาพที่ 3.2 แผนภาพคลาสที่ทำหน้าที่หลักภายในเกม 14](#_Toc194880352)

[ภาพที่ 3.3 ส่วนต่อประสานผู้ใช้ของหน้าจอเมนู 17](#_Toc194880353)

[ภาพที่ 3.4 ส่วนต่อประสานผู้ใช้ของหน้าจอตั้งค่า 18](#_Toc194880354)

[ภาพที่ 3.5 ส่วนต่อประสานผู้ใช้ของหน้าจอสอนการเล่น 18](#_Toc194880355)

[ภาพที่ 3.6 ส่วนต่อประสานผู้ใช้ของหน้าจอเล่นเกม 19](#_Toc194880356)

[ภาพที่ 3.7 การออกแบบด่านในเกมด่านที่หนึ่ง 19](#_Toc194880357)

[ภาพที่ 3.8 การออกแบบด่านในเกมด่านที่สอง 20](#_Toc194880358)

# บทนำ

* 1. ความเป็นมาและเหตุผลของโครงงาน

ในปัจจุบันเกมที่ได้รับความนิยมในหมู่ผู้เล่นนั้นมีหลากหลายประเภทตั้งแต่เกมแอคชันที่ต้้องการการ ตอบสนองที่รวดเร็วไปจนถึงเกมที่ต้องใช้การวางแผนหรือเกมที่ใช้ความคิดสร้างสรรค์ได้อย่างอิสระ อย่างไรก็ตามยังมีกลุ่มผู้เล่นสาย “ Hardcore ” ที่ชื่นชอบเกมที่มีความท้าทายสูงและมีความต้องการเกม ที่สามารถผสมผสานกลไกการเล่นหลายรูปแบบได้อย่างลงตัวเพื่อทดสอบทั้งทักษะทั้งทางการตอบสนองของ ร่างกายและความคิดภายใต้สถานการณ์กดดันไปพร้อมกัน

เกมประเภท Bullet Hell เช่น Touhou Project [1] และ Platform Fighter แบบ Super Smash Bros. [2] ดังรูปที่ 1 และ 2 ตามลำดับ ถือเป็นตัวอย่างที่ชัดเจนของเกมที่มอบความท้าทายเหล่านี้ให้ผู้เล่น ในเกมประเภท Bullet Hell ผู้เล่นจำเป็นต้องหลบกระสุนจำนวนมากที่พุ่งเข้ามาจากทุกทิศทุกทาง อย่างต่อเนื่อง ซึ่งต้องอาศัยทักษะการตอบสนองที่รวดเร็วรวมทั้งการคาดการณ์ล่วงหน้าถึงทิศทางของกระสุน ที่จะเคลื่อนที่และปรากฏออกมาขณะที่เกม Platform Fighter อย่าง Super Smash Bros. จะเน้นการโจมตี และการหลบหลีกการโจมตีของคู่ต่อสู้ภายใต้สภาพแวดล้อมที่สามารถเปลี่ยนแปลงไปได้ ซึ่งต้องใช้ การวางแผนเชิงกลยุทธ์และการตอบสนองที่รวดเร็วในการโต้กลับในการเอาชนะคู่ต่อสู้

ภาพที่ 1.1 Touhou 14

ภาพที่ 1.2 Super Smash Bros. Ultimate

รูปภาพประกอบด้วย เกมพีซี, แอนิเมชัน, วิดีโอเกมกลยุทธ์, การ์ตูนแอนิเมชัน

คำอธิบายที่สร้างโดยอัตโนมัติ

รูปที่ 2 Super Smash Bros. Ultimate

รูปภาพจาก

https://www.bbc.com/news/newsbeat-67407712

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ม่วง, สีม่วง

คำอธิบายที่สร้างโดยอัตโนมัติ

รูปที่ 1 Touhou 14

รูปภาพจากhttps://steamcommunity.com/sharedfiles/filedetails/?l=thai&id=173745051

นอกจากนี้การผสมผสานกลไกของ Bullet Hell และ Platform Fighter เข้าด้วยกันจะนำไปสู่ ประสบการณ์ใหม่ที่จะช่วยตอบสนองความต้องการของกลุ่มผู้เล่น Hardcore ที่ต้องการความท้าทาย ที่หลากหลายมากขึ้น ส่วนการใช้มุมมองและฟิสิกส์แบบ Platform Fighter ช่วยให้ผู้เล่นรู้สึกมีอิสระ ในรูปแบบการเคลื่อนไหวแต่การออกแบบเกมเพลย์ที่ให้เน้นการหลบหลีกกระสุนแบบ Bullet Hell จะเพิ่มระดับความยากขึ้นอย่างมากเนื่องจากผู้เล่นจำเป็นต้องคิดให้ดีก่อนการกระโดดเคลื่อนไหวแต่ละครั้ง และยังจำเป็นต้องคำนึงถึงสภาพ Platform ของด่านเพื่อใช้ในรีเซ็ตแอคชันในการเคลืื่อนไหวสำหรับการหลบ การโจมตีต่อไป ผู้เล่นจะต้องเผชิญหน้ากับการโจมตี ที่ซับซ้อนจากศัตรูหลักซึ่งเป็น Boss ที่จะยิงกระสุนจำนวนมากเข้ามาหาผู้เล่นอยู่ตลอดเวลา ผู้เล่นไม่เพียงแต่จะต้องหลบหลีกแต่ยังต้อง หาจังหวะที่เหมาะสมในการโจมตีสวนกลับซึ่งจะเป็นการผสมผสานระหว่างทักษะการหลบหลีก ความเร็วในการตัดสินใจ และความสามารถในการอ่านสถานการณ์และคาดการณ์ล่วงหน้า ให้ถูกต้องเข้าด้วยกัน

การสร้างเกมที่มีระบบการเล่นผสมผสานเช่นนี้ ไม่เพียงแต่จะช่วยสร้างความสดใหม่ในวงการเกม แต่ยังช่วยให้ผู้เล่นได้รับประสบการณ์การเล่นที่ท้าทายที่มากยิ่งกว่าเดิม ซึ่งจะเป็นสิ่งที่ดึงดูดผู้เล่นกลุ่ม Hardcore ที่แสวงหาความท้าทายที่แปลกใหม่ อีกทั้งยังสามารถดึงดูดกลุ่มผู้เล่น หน้าใหม่ที่สนใจในการ ท้าทายขีดความสามารถของตนเองในการเล่นเกมที่ต้องใช้ทักษะรอบด้าน

โดยโครงงานนี้จะสร้างเกมที่เป็นการผสมผสานทั้ง 2 อย่างเข้าด้วยกันตามที่กล่าวมา โดยจะใช้ Art style เป็นแบบ ภาพขาวดำ เมื่อเข้าเกมมาผู้เล่นจะพบกับหน้าต่างสำหรับเริ่มเกม

A mountain range with clouds and birds flying

AI-generated content may be incorrect.

ภาพที่ 1.3 หน้าต่าง start game

ในหน้าต่างแรกเริ่มจะประกอบไปด้วย ชื่อเกม ปุ่มเริ่มเกม ปุ่มตั้งค่าเกม ดังรูปที่ 3 หลังจากที่ผู้เล่นกดเข้าไปที่ปุ่มเริ่มเกมก็จะมีการเข้าไปยังหน้าเลือกด่านแล้วจึงจะเข้าไปภายในเกม

A screen shot of a video game

AI-generated content may be incorrect.

ภาพที่ 1.4 องค์ประกอบที่จะมีภายในด่าน

เมื่อเข้าสู่ด่านและเริ่มเกม ผู้เล่นจะพบกับองค์ประกอบหลัก ได้แก่ ตัวละครผู้เล่น (Player) ตัวละครบอส (Boss) แท่นเหยียบ (Platform) และกระสุนหลากหลายรูปแบบที่บอสจะยิงเข้ามา (Bullet) ดังแสดงในรูปที่ 4 ผู้เล่นจะต้องเคลื่อนไหวหลบกระสุนโดยใช้ปุ่ม Spacebar หรือ W A S D บนคีย์บอร์ด เพื่อควบคุมการเคลื่อนไหว โดยจะเป็นการกระโดด การเคลื่อนที่ไปทางซ้าย การเคลื่อนที่ลงจากแท่นเหยียบ และการเคลื่อนที่ไปทางขวาตามลำดับ การเคลื่อนไหว การกระโดดและใช้ประโยชน์จากแท่นเหยียบจะช่วย ให้ผู้เล่นหลบหลีกกระสุนหรือโจมตีบอสได้ในสถานการณ์ที่เหมาะสม โดยในขณะที่อยู่กลางอากาศ ผู้เล่นจะสามารถกระโดดซ้ำได้อีกครั้งหนึ่ง อย่างไรก็ตามผู้เล่นจะต้องกลับมาเหยียบแท่นหรือพื้นก่อน เพื่อรีเซ็ตความสามารถนี้ให้พร้อมใช้ในการกระโดดครั้งต่อไป

ระหว่างการเล่นจะมีไอเทมปรากฏขึ้นในจุดต่างๆ ของด่านให้ผู้เล่นเก็บสะสม ซึ่ง เมื่อเก็บแล้วจะได้รับพลังเสริมชั่วคราวที่ช่วยเพิ่มประสิทธิภาพในหลายรูปแบบ เช่น โล่ป้องกันชั่วคราว การหยุดกระสุนชั่วขณะ หรือเพิ่มพลังการโจมตีของผู้เล่น อย่างไรก็ตามนอกจากไอเทมที่จะส่งผลดี กับผู้เล่นแล้วก็ยังมีไอเทมกับดักซึ่งจะผลเชิงลบกับผู้เล่น เช่น การลดความเร็วในการเคลื่อนไหว หรือ การล็อคการเคลื่อนไหวชั่วคราว เช่นกัน ทำให้ผู้เล่นจะต้องระมัดระวังและแยกแยะว่าไอเทมใด เป็นไอเทมที่ควรเก็บในระหว่างการเล่นด้วย

ในการที่จะชนะบอสได้นั้น ผู้เล่นต้องใช้การสังเกตและค้นหาจุดอ่อนของบอสซึ่งกลไก และวิธีการเหล่านี้ถูกซ่อนอยู่ภายในด่าน โดยต้องอาศัยความช่างสังเกตในการระบุ และใช้งานกลไกเสริม ที่สามารถช่วยโจมตีบอสเพื่อเอาชนะได้

A black and white text on a white background

AI-generated content may be incorrect.

ภาพที่ 1.5 หน้าต่าง เมื่อผู้เล่น Game Over

ถ้าหากผู้เล่นถูกโจมตีจากบอสหรือว่าพลัดตกลงจากแท่นเหยียบ ผู้เล่นจะพ่ายแพ้ทันที และหน้าต่างแสดงความพ่ายแพ้จะปรากฏขึ้นมา ดังแสดงในรูปที่ 5 โดยผู้เล่นจะมีเพียงทางเลือกเดียวเท่านั้น คือกลับไปเริ่มใหม่ตั้งแต่หน้าจอเริ่มต้นและเริ่มการผจญภัยอีกครั้ง

* 1. วัตถุประสงค์ของโครงงาน

เพื่อพัฒนาเกม PC ที่มีรูปแบบการเล่นแบบเล่นคนเดียว ซึ่งจะผสมผสานเกมแอคชันที่รวดเร็ว เข้ากับเกมปริศนาเพื่อทดสอบทั้งความสามารถในการตอบสนองและการคิดวิเคราะห์ของผู้เล่น โดยใช้กลไกและมุมมองการเล่นแบบ Platform Fighter ที่ได้รับแรงบันดาลใจจาก Super Smash Brothers และ Bullet Hell จาก Touhou Project เพื่อพัฒนาและทดสอบทักษะทั้งสองด้าน ให้กับผู้เล่น

* 1. ขอบเขตของโครงงาน

1. รองรับระบบปฏิบัติการ Windows 11 Version 23H2 เท่านั้น
2. เป็นเกมเล่นคนเดียวบนคอมพิวเตอร์ส่วนบุคคลเท่านั้น
3. รูปแบบของเกมจะเป็นการผสมผสานระหว่างเกมแอคชันและเกมปริศนา โดยมีการใช้กลไก การหลบหลีกการโจมตีที่ซับซ้อนและการไขปริศนาที่ต้องใช้การวิเคราะห์เชิงลึก
4. กลุ่มเป้าหมายของเกมคือกลุ่มผู้เล่นที่ชอบเล่นเกมที่ยาก
   1. ขั้นตอนการดำเนินงาน

การพัฒนาเกมแอปพลิเคชัน : เกมกระโดดหลบ “Satori : Enso” บน Windows มีขั้นตอนการดำเนินการดังต่อไปนี้

1. วิเคราะห์และออกแบบระบบเกม (System Analysis and Design)
2. การออกแบบเกมเพลย์และเลเวล (Gameplay and Level Design)
3. พัฒนาโปรแกรม (Develop)
4. ทดสอบการทำงาน (Test and Debug)
5. ทดสอบกลุ่มตัวอย่าง (Sample Test)
6. ประเมินและปรับปรุง (Evaluate and Refine)
7. จัดทำรูปเล่มรายงานของโครงงาน (Project Report)
   1. ประโยชน์ที่คาดว่าจะได้รับ

ประโยชน์ต่อผู้พัฒนา

1. เพิ่มทักษะการพัฒนาเกมในสภาพแวดล้อมจริง ตั้งแต่การออกแบบระบบ ไปจนถึงการเขียนโค้ด และการทดสอบเกม
2. ได้ฝึกทักษะด้านการวิเคราะห์และแก้ไขปัญหาทางเทคนิค และการทำงานร่วมกับซอฟต์แวร์ และเครื่องมือพัฒนาเกม
3. ได้รับประสบการณ์ในการจัดการ แบ่งงาน ทำงานร่วมกับผู้อื่น ในโปรเจคที่เสกลใหญ่มากขึ้น
4. เกมที่พัฒนาขึ้นสามารถใช้เป็นต้นแบบสำหรับการพัฒนาเกมที่ผสมผสานทั้งเกมแอคชันและ ปริศนาในอนาคต
5. เกมนี้อาจมีการต่อยอดไปสู่การพัฒนาเพิ่มเติมในเชิงธุรกิจ โดยนำไปวางจำหน่ายในร้านค้า ออนไลน์ อาทิ Steam หรือใช้ในการพัฒนาเกมที่เน้นการฝึกทักษะต่าง ๆ ของผู้เล่นต่อไป

ประโยชน์ต่อผู้เล่น

1. ผู้เล่นสามารถเล่นเพื่อท้าทายตัวเองได้อย่างสนุกสนาน
2. ผู้เล่นสามารถเพิ่มทักษะการตอบสนองและการวางแผนผ่านการเล่นเกมได้
3. ในด้านการศึกษา เกมนี้อาจเป็นต้นแบบสำหรับการวิจัยในเรื่องของพฤติกรรมผู้เล่น ในสถานการณ์ ที่ท้าทายและการตอบสนองต่อความยากในการเล่นเกม
   1. โครงสร้างของรายงาน

บทที่ 2 จะกล่าวองค์ความรู้และทฤษฏีที่ใช้ในการสร้างเกมแอพลิเคชั่นนี้

บทที่ 3 จะกล่าวถึงการวิเคราะห์และออกแบบ ซึ่งจะประกอบไปด้วย การออกแบบเกมเพลย์ การออกแบบระบบและโครงสร้างของเกม การออกแบบ UI การออกแบบด่าน

บทที่ 4 จะกล่าวถึงการพัฒนาและการทดสอบกับกลุ่มตัวอย่าง

บทที่ 5 จะกล่าวถึงข้อสรุป และข้อเสนอแนะ

บทที่ 2  
ความรู้พื้นฐานและทฤษฎีที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงความรู้พื้นฐานและทฤษฎีที่เกี่ยวข้องในการใช้พัฒนาโครงงานเกมแอปพลิเคชัน : เกมกระโดดหลบ “Satori : Enso” บน Windows ทั้งในส่วนของเครื่องมือที่ใช้ในการพัฒนาและองค์ความรู้ที่นำมาประยุกต์ใช้

* 1. ทฤษฎีที่เกี่ยวข้อง
     1. **Object Oriented Programming**

Object-Oriented Programming หรือ OOP คือ การเขียนโปรแกรมเชิงวัตถุ ซึ่งเป็นแบบจำลองในการเขียนโปรแกรมที่เน้นมองสิ่งต่าง ๆ ในระบบให้เป็น Object การทำแบบนี้จะช่วยให้นักพัฒนาสามารถจัดระเบียบโค้ดและเข้าใจการทำงานในภาพรวมมากขึ้น และยังมีอีกส่วนที่สำคัญนั่นก็คือ Class ที่ทำหน้าที่เป็นต้นแบบสำหรับการสร้าง Object โดย Class จะกำหนดคุณสมบัติ (Attributes) และพฤติกรรม (Methods) ที่ Object ของ Class นั้นควรจะมี หลักการทำงานของ OOP มีดังนี้

* + - 1. **Encapsulation**

การซ่อนข้อมูลถือเป็นหลักการพื้นฐานของ OOP โดยจะทำการซ่อนข้อมูลในส่วนของ Class อย่างคุณสมบัติ (Attributes) และพฤติกรรม (Methods) ที่เกี่ยวข้องภายใน Object ให้เป็นแบบ Private ไว้เพื่อป้องกันการเข้าถึงจากภายนอกได้และกำหนดให้สามารถเข้าถึงได้เฉพาะในส่วนที่เรากำหนดให้เป็น Public เท่านั้น

* + - 1. **Abstraction**

Abstraction ต่อยอดมาจาก Encapsulation คือการเปิดเผย Object เฉพาะส่วนที่จำเป็นต่อการใช้งานโดยเราไม่จำเป็นต้องรู้การทำงานหรือรายละเอียดภายในของมัน

* + - 1. **Inheritance**

การสืบทอดคุณสมบัติ (Inheritance) ใน OOP นั้นช่วยให้เราสามารถสร้าง Class ใหม่ได้โดยเอาคุณสมบัติ (Attributes) และพฤติกรรม (Methods) จาก Class ที่มีอยู่แล้วมาสร้างและเพิ่มคุณสมบัติ (Attributes) กับพฤติกรรม (Methods) เฉพาะตัวเข้าไปเพิ่มได้

* + - 1. **Polymorphism**

ช่วยให้สามารถใช้ Object ที่มีคุณสมบัติ (Attributes) และพฤติกรรม (Methods) ที่แตกต่างกันได้ใน Class เดียวกันหรือก็คือตัว Class หลักจะกำหนดคุณสมบัติ (Attributes) และพฤติกรรม (Methods) เอาไว้และให้ Class ย่อยต่าง ๆ ไประบุรายละเอียดต่าง ๆ ของคุณสมบัติ (Attributes) และพฤติกรรม (Methods) เอง

* + 1. **Physics in Game Development**

ฟิสิกส์ในเกมเป็นองค์ประกอบที่สำคัญในการทำให้วัตถุมีพฤติกรรมที่สมจริง เช่น การเคลื่อนที่ แรง และแรงกระทำที่เกิดจากการชนกัน โดยมีทฤษฎีหลักที่เกี่ยวข้องดังนี้

* + - 1. **Dynamics and Kinematics**

การเคลื่อนที่ของวัตถุในเกมมักถูกจำลองโดยใช้ กฎการเคลื่อนที่ของนิวตัน ซึ่งสามารถแบ่งออกเป็นสองส่วนหลัก ได้แก่:

**Kinematics (จลนศาสตร์):** ศึกษาการเคลื่อนที่ของวัตถุโดยไม่คำนึงถึงแรง เช่น การคำนวณระยะทาง ความเร็ว และความเร่ง

**Dynamics (พลศาสตร์):** ศึกษาเกี่ยวกับแรงที่กระทำต่อวัตถุและผลที่เกิดขึ้น เช่น การคำนวณแรงโน้มถ่วงและแรงเสียดทาน

* + - 1. **Collision Detection and Response**

การตรวจจับการชนเป็นองค์ประกอบสำคัญในระบบฟิสิกส์ของเกม ซึ่งสามารถแบ่งออกเป็นสองแนวทางหลัก:

**Discrete Collision Detection (DCD):** ตรวจสอบการชนในแต่ละเฟรมของเกม อาจทำให้เกิดปัญหา "Tunneling" หรือการพลาดการตรวจจับเมื่อวัตถุเคลื่อนที่เร็วเกินไป

**Continuous Collision Detection (CCD):** ใช้วิธีการคำนวณตำแหน่งล่วงหน้าเพื่อตรวจจับการชนอย่างแม่นยำ แม้ในกรณีที่วัตถุเคลื่อนที่เร็ว

* + - 1. **Forces and Motion in Games**

ระบบฟิสิกส์ของเกมมักใช้แนวคิดของแรงที่กระทำต่อวัตถุ เช่น:

* แรงโน้มถ่วง:
* แรงเสียดทาน:
* แรงจากการชนกัน:
  + 1. **Mathematics in Game Development**
       1. **Vector Mathematics**

เวกเตอร์ (Vector) เป็นแนวคิดทางคณิตศาสตร์ที่ใช้ในการแทนค่าปริมาณที่มีทั้งขนาดและทิศทาง ซึ่งในเกมมักใช้เวกเตอร์เพื่อกำหนดตำแหน่ง ทิศทาง และความเร็วของวัตถุสมการที่ใช้บ่อยได้แก่:

* ขนาด (Magnitude)
* ทิศทาง (Direction)

สมการที่ใช้บ่อยได้แก่:

* การหาขนาดของเวกเตอร์:
* การบวกเวกเตอร์:
* การคูณเวกเตอร์กับสเกลาร์:
  + - 1. **Trigonometry in Game Development**

ตรีโกณมิติ (Trigonometry) เป็นศาสตร์ที่ใช้ศึกษาเกี่ยวกับมุมและความสัมพันธ์ระหว่างด้านของรูปสามเหลี่ยม ซึ่งในเกมใช้เพื่อคำนวณมุมการหมุนของวัตถุหรือการเคลื่อนที่ของกระสุนในแนวโค้ง

* หาทิศทางของวัตถุที่เคลื่อนที่:
* คำนวณตำแหน่งของวัตถุในระบบพิกัดเชิงขั้ว:

* + 1. **Rendering and Graphics in Game Development**
       1. **Rendering Pipeline**

การเรนเดอร์ (Rendering) เป็นกระบวนการสร้างภาพกราฟิกโดยการคำนวณและแสดงผลบนหน้าจอ ซึ่งแบ่งออกเป็นหลายขั้นตอน เช่น:

1. **Vertex Processing:** แปลงตำแหน่งของวัตถุจากระบบพิกัดโลกเป็นระบบพิกัดหน้าจอ
2. **Clipping and Culling:** คัดกรองวัตถุที่อยู่นอกขอบเขตหน้าจอเพื่อลดภาระการคำนวณ
3. **Rasterization:** เปลี่ยนรูปทรงเวกเตอร์ให้เป็นพิกเซล
4. **Fragment Processing:** คำนวณสีและการแสดงผลของพิกเซลที่เกี่ยวข้อง
5. **Blending and Post-Processing:** ผสมสีและเอฟเฟ็กต์แสงเงาเพื่อทำให้ภาพดูสมจริง
   * + 1. **Shader Programming**

Shader เป็นโปรแกรมขนาดเล็กที่ใช้คำนวณและกำหนดลักษณะของวัตถุกราฟิกในเกม โดยแบ่งเป็น:

1. **Vertex Shader:** คำนวณตำแหน่งของจุดบนหน้าจอ
2. **Fragment Shader:** คำนวณสีของแต่ละพิกเซล
3. **Post-Processing Shader:** ใช้สร้างเอฟเฟ็กต์ เช่น Bloom, Motion Blur และ Depth of Field
   1. โปรแกรมประยุกต์ที่ใช้
      1. **Unity**

**Unity** เป็นโปรแกรมสำหรับพัฒนาเกม (Game Engine) ที่ได้รับความนิยมอย่างแพร่หลายทั้งในวงการอุตสาหกรรมเกมระดับมืออาชีพและในกลุ่มนักพัฒนาระดับเริ่มต้นจนถึงระดับกลาง โดย Unity มีคุณสมบัติที่โดดเด่นดังนี้:

* รองรับการพัฒนาเกมทั้งแบบ 2 มิติ (2D) และ 3 มิติ (3D)
* มีระบบฟิสิกส์ในตัว ทั้ง Physics 2D และ Physics 3D
* มีระบบจัดการฉาก (Scene) ที่ช่วยให้นักพัฒนาสามารถวางวัตถุ จัดแสง และตั้งค่าต่าง ๆ ได้อย่างสะดวก
* มี Inspector และ Hierarchy ที่ใช้ในการจัดการวัตถุภายในเกม
* รองรับการทำงานแบบ Component-Based Architecture ซึ่งให้นักพัฒนาสามารถเพิ่มความสามารถให้กับวัตถุผ่านการใส่ Component ต่าง ๆ
* มี Asset Store ที่เปิดให้นักพัฒนาสามารถดาวน์โหลดหรือซื้อทรัพยากรมาใช้ได้อย่างง่ายดาย
* สามารถ Build ออกไปยังหลายแพลตฟอร์ม เช่น Windows, Android, iOS, WebGL ฯลฯ
  + 1. **ภาษา C#**

**ภาษา C# (C-Sharp)** เป็นภาษาการเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming) ที่ถูกออกแบบและพัฒนาโดยบริษัท Microsoft โดยมีคุณสมบัติเหมาะสมในการใช้พัฒนาแอปพลิเคชันที่ต้องการความปลอดภัย มีโครงสร้างชัดเจน และสามารถจัดการกับความซับซ้อนได้ดี โดยเฉพาะเมื่อนำมาใช้ร่วมกับ Unity จะทำให้สามารถเขียนสคริปต์เพื่อควบคุมพฤติกรรมต่าง ๆ ภายในเกมได้อย่างยืดหยุ่น

คุณสมบัติของภาษา C# ที่โดดเด่น ได้แก่:

* รองรับแนวคิด OOP อย่างเต็มรูปแบบ เช่น Class, Object, Inheritance, Polymorphism และ Abstraction
* มีระบบจัดการข้อผิดพลาด (Exception Handling) ที่ช่วยเพิ่มความเสถียรในการทำงานของโปรแกรม
* Syntax อ่านง่าย มีความคล้ายคลึงกับภาษากลุ่ม C อื่น ๆ เช่น Java และ C++
* สามารถใช้งานร่วมกับระบบของ Unity ได้อย่างสมบูรณ์ ผ่าน Unity API ที่มีให้ใช้อย่างครอบคลุม

ด้วยคุณสมบัติเหล่านี้ ภาษา C# จึงถือเป็นเครื่องมือสำคัญที่ใช้ในการควบคุมตรรกะต่าง ๆ ภายในเกมและเชื่อมโยงระบบต่าง ๆ เข้าด้วยกันอย่างเป็นระบบ

* + 1. **Git และระบบควบคุมเวอร์ชัน (Version Control System)**

Git เป็นระบบควบคุมเวอร์ชันแบบกระจาย (Distributed Version Control System) ที่ได้รับความนิยมอย่างแพร่หลายในการพัฒนาโปรแกรมและซอฟต์แวร์ รวมถึงงานพัฒนาเกมด้วย โดย Git ช่วยให้สามารถติดตามการเปลี่ยนแปลงของไฟล์ในโปรเจกต์ ทั้งในด้านของซอร์สโค้ด ทรัพยากร และไฟล์คอนฟิกต่าง ๆ ได้อย่างเป็นระบบ ประโยชน์ของ Git สามารถสรุปได้เป็นดังนี้:

* การติดตามเวอร์ชัน (Version History) Git สามารถเก็บบันทึกการเปลี่ยนแปลงของไฟล์ทั้งหมดในแต่ละ commit ทำให้นักพัฒนาสามารถย้อนกลับไปยังสถานะก่อนหน้าได้หากเกิดข้อผิดพลาด
* รองรับการทำงานร่วมกัน (Collaboration) นักพัฒนาหลายคนสามารถทำงานบนโปรเจกต์เดียวกันได้โดยไม่รบกวนกันผ่านระบบ branch, merge และ conflict resolution
* รองรับการใช้งานร่วมกับบริการ Cloud เช่น GitHub, GitLab, Bitbucket ทำให้สามารถจัดเก็บและแชร์โปรเจกต์ผ่านออนไลน์ได้อย่างสะดวก
* ระบบ Branching ที่ยืดหยุ่นช่วยให้นักพัฒนาสามารถสร้างสาขาการทำงานแยกออกจากกัน เช่น สาขาพัฒนา สาขาทดสอบ หรือฟีเจอร์เฉพาะ แล้วนำกลับมารวมเข้ากับระบบหลักได้

# การวิเคราะห์และออกแบบ

ในบทนี้จะกล่าวถึงภาพรวมของการวิเคราะห์ข้อมูลความต้องการของระบบ เพื่อนำมาออกแบบให้อยู่ในรูปแบบโครงสร้างของระบบต่าง ๆ สำหรับการพัฒนาเกมกระโดดหลบ Satori : Enso บนระบบปฎิบัติการ วินโดว์ 11 รวมทั้งอธิบายถึงรูปแบบการเล่น ไปจนถึงระบบต่างๆภายในเกม

* 1. ออกแบบวิธีการเล่น(Gameplay Design)

ลักษณะของวิธีการเล่นถือเป็นองค์ประกอบสำคัญที่ส่งผลโดยตรงต่อแนวทางการพัฒนาเกม โดยเกมนี้ได้รับการออกแบบให้อยู่ในรูปแบบผสมระหว่างแนว Platform Fighter และ Bullet Hell ซึ่งผู้เล่นจะต้องทำการหลบหลีกกระสุนจำนวนมากจากศัตรู พร้อมทั้งทำการแก้ไขปริศนาเพื่อเอาชนะศัตรูและผ่านเข้าสู่ด่านถัดไป

จากรูปแบบการเล่นข้างต้น จึงต้องมีการออกแบบระบบควบคุมตัวละครที่มีความลื่นไหล รวดเร็ว และตอบสนองได้อย่างฉับไว เพื่อให้เหมาะสมกับสถานการณ์ที่ผู้เล่นต้องหลบกระสุนตลอดเวลา กลไกต่าง ๆ เช่น การเคลื่อนไหวกลางอากาศ การแดช หรือการเร่งความเร็วในแนวดิ่ง จึงถูกออกแบบมาเพื่อส่งเสริมประสบการณ์การเล่นที่รื่นไหลและมีความยืดหยุ่นสูงให้ผู้เล่นเลือกใช้จัดการกับสถานการณ์ภายในเกมได้อย่างมีประสิทธิภาพ

เพื่อเสริมสร้างความหลากหลายในการเล่นและเพิ่มความท้าทาย ได้มีการออกแบบระบบไอเทมพิเศษที่สุ่มปรากฏภายในด่าน เช่น ไอเทมเพิ่มความเร็ว ไอเทมเพิ่มพลังกระโดด หรือไอเทมลดขนาดตัวละคร ซึ่งจะส่งผลต่อการเล่นในลักษณะที่แตกต่างกัน ขึ้นอยู่กับสถานการณ์ที่ผู้เล่นพบเจอ

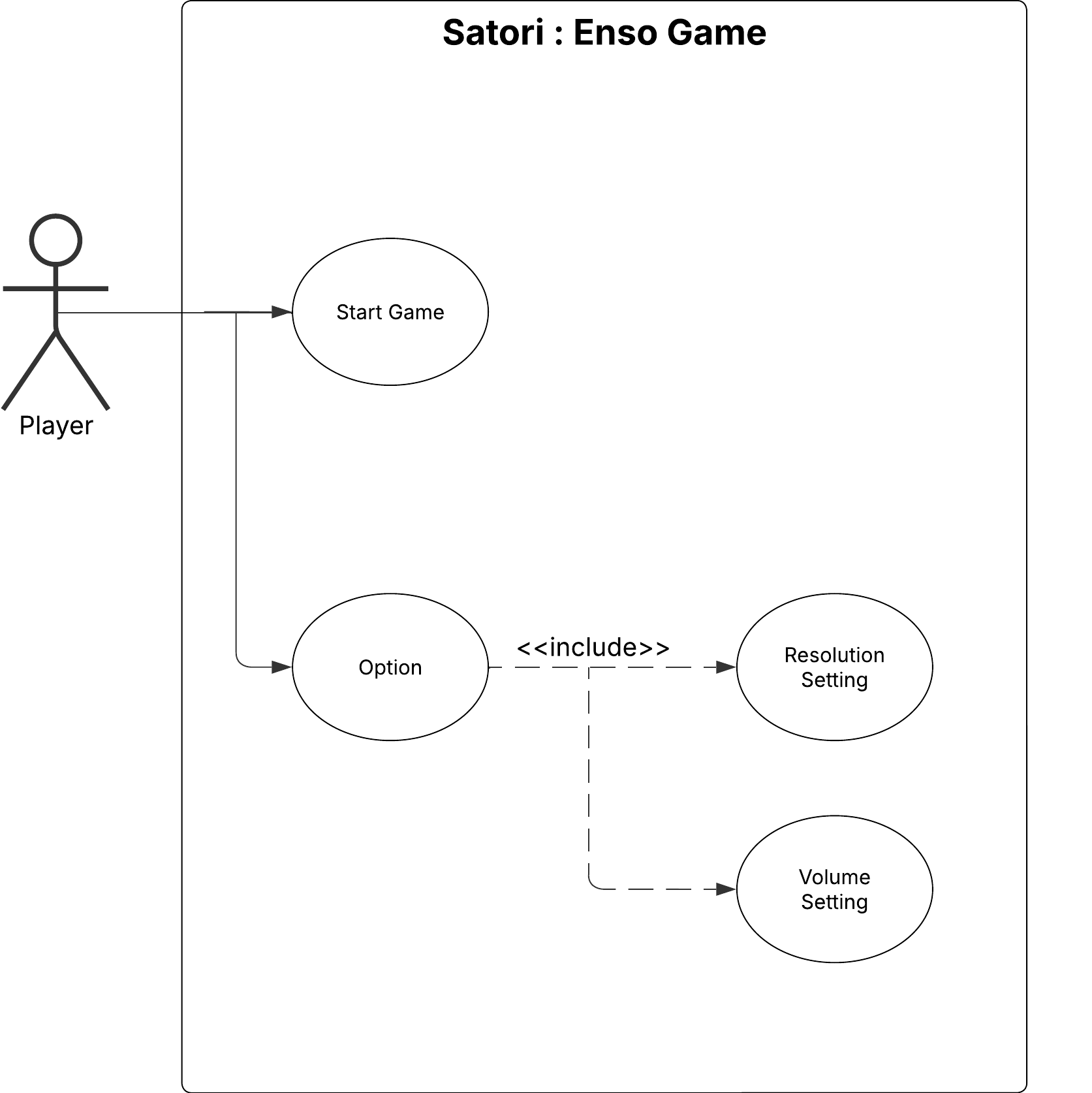
ในด้านการออกแบบกระสุน รูปแบบของกระสุนได้รับการออกแบบให้หลากหลาย เพื่อสร้างความน่าสนใจและกระตุ้นให้ผู้เล่นต้องปรับตัวอยู่ตลอดเวลา เช่น กระสุนที่เคลื่อนที่เป็นเส้นตรง กระสุนที่เคลื่อนที่ตามรูปคลื่นไซน์ หรือกระสุนที่มีการกระเด้งเมื่อชนกับพื้นผิวต่าง ๆ ซึ่งจำเป็นต้องใช้การสังเกตและการตอบสนองที่แม่นยำในการหลบหลีก

องค์ประกอบสุดท้ายที่สำคัญคือ "เป้าหมายของผู้เล่น" ซึ่งในที่นี้คือการเอาชนะศัตรู โดยวิธีการที่ใช้คือการแก้ปริศนาในด่าน ซึ่งระบบปริศนาได้รับการออกแบบไว้หลากหลายรูปแบบ และจะถูกสุ่มเรียกใช้งานในแต่ละช่วงเวลาของเกม เมื่อผู้เล่นสามารถแก้ปริศนาได้สำเร็จจะสามารถสร้างความเสียหายแก่ศัตรู และเมื่อสามารถกำจัดศัตรูได้สำเร็จจึงจะสามารถผ่านเข้าสู่ด่านถัดไปได้

* 1. การวิเคราะห์และออกแบบระบบ(System Design)

การออกแบบภาพรวม โครงสร้างของระบบ ความสามารถของผู้เล่นที่สามารถกระทำการได้ Workflow การทำงานของระบบในการใช้งาน เพื่อให้สามารถพัฒนาได้ตามที่ออกแบบเอาไว้โดยไม่ติดขัด โดยได้ออกแบบไว้เป็นแผนภาพและตารางดังนี้

* + 1. **แผนภาพยูสเคสของเกม**



ภาพที่ 3.1 แผนภาพยูสเคสของเกม

มีรายละเอียดแบบกรณีการใช้งาน (Use Case Template) ดังต่อไปนี้

ตารางที่ 3.1 กรณีการใช้งาน Start Game

|  |  |
| --- | --- |
| ชื่อ | Start Game |
| เป้าหมาย | ผู้เล่นต้องการเริ่มเล่นเกม |
| ผู้ใช้ | ผู้เล่น |
| เงื่อนไขก่อนหน้า | - |
| ลำดับของเหตุการณ์ | 1. ผู้ใช้คลิกปุ่มเข้าเกม หรือกด space bar 2. ระบบแสดงคำแนะนำการเล่น 3. ผู้เล่นเข้าสู่เกมด่านแรก |
| ข้อยกเว้น | - |
| เงื่อนไขภายหลัง | - |

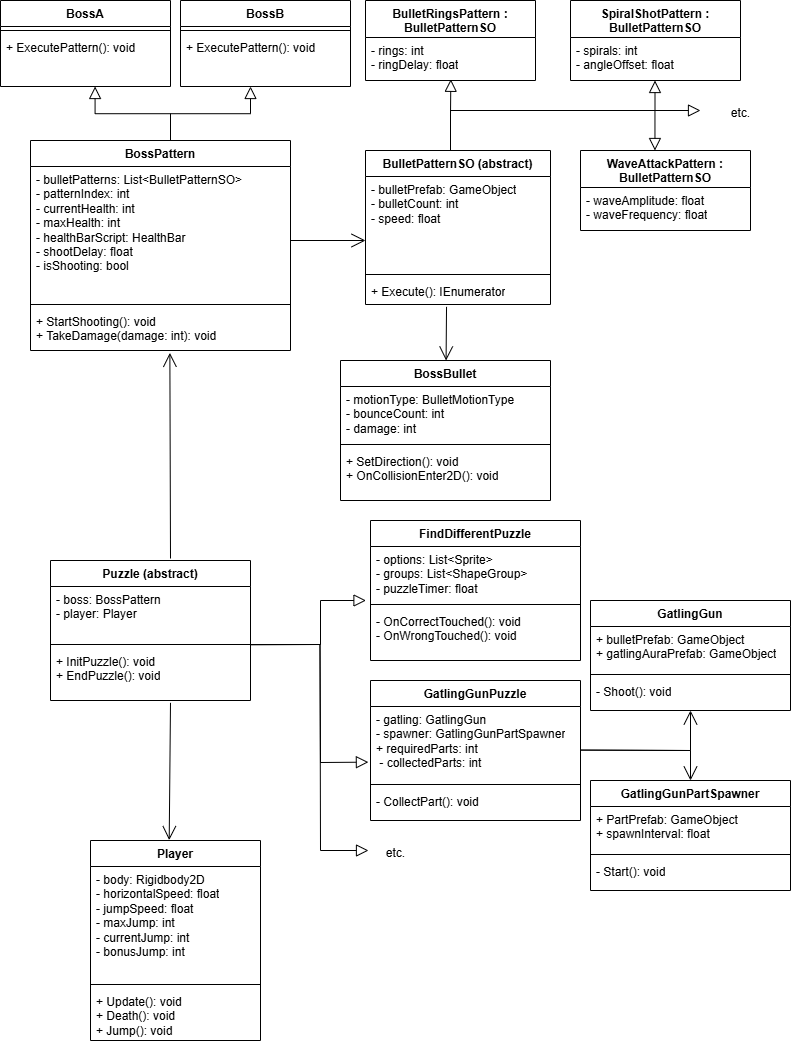
ตารางที่ 3.2 กรณีการใช้งาน Resolution Setting

|  |  |
| --- | --- |
| ชื่อ | Resolution Setting |
| เป้าหมาย | ผู้เล่นต้องการปรับความละเอียดและขนาดของจอแสดงผลเกม |
| ผู้ใช้ | ผู้เล่น |
| เงื่อนไขก่อนหน้า | ผู้เล่นคลิกเข้า Option |
| ลำดับของเหตุการณ์ | 1. ผู้เล่นเลือกความละเอียดของหน้าจอที่ต้องการ 2. ผู้เล่นกดตกลง 3. แสดงข้อความว่าดำเนินการสำเร็จ |
| ข้อยกเว้น | - |
| เงื่อนไขภายหลัง | หน้าจอเกมเปลี่ยนเป็นความละเอียดที่ผู้เล่นเลือก |

ตารางที่ 3.3 กรณีการใช้งาน Volume Setting

|  |  |
| --- | --- |
| ชื่อ | Volume Setting |
| เป้าหมาย | ผู้เล่นต้องการปรับความดังของเสียงในเกม |
| ผู้ใช้ | ผู้เล่น |
| เงื่อนไขก่อนหน้า | ผู้เล่นคลิกเข้า Option |
| ลำดับของเหตุการณ์ | 1. ผู้เล่นเลือกความดังของเสียงที่ต้องการ 2. ผู้เล่นกดตกลง 3. แสดงข้อความว่าดำเนินการสำเร็จ |
| ข้อยกเว้น | - |
| เงื่อนไขภายหลัง | ตัวเกมปรับระดับของเสียงให้เป็นระดับที่ผู้เล่นเลือกไว้ |

* + 1. **การออกแบบคลาสภายในเกม**



ภาพที่ 3.2 แผนภาพคลาสที่ทำหน้าที่หลักภายในเกม

สำหรับหน้าที่และคำอธิบายคลาสจากแผนภาพคลาสข้างต้น มีดังต่อไปนี้

1. **คลาส BossPattern**

เป็นแกนกลางของระบบบอสในเกม มีหน้าที่กำหนดพฤติกรรมการต่อสู้ของบอส เช่น การเลือกรูปแบบการยิงกระสุน (BulletPatternSO), การตั้งค่าพลังชีวิต (HealthBar), การเล่นเสียง และการสั่นหน้าจอเมื่อได้รับความเสียหาย โดย BossPattern จะเป็นคลาสพื้นฐานที่สามารถถูกสืบทอดได้โดยบอสแต่ละประเภท (BossA, BossB)

**หน้าที่หลัก:**

* เรียกใช้แพทเทิร์นการยิงแบบเป็นลำดับ
* รับความเสียหายและปรับค่าพลังชีวิต
* สั่งเล่นเสียงและสั่นหน้าจอเพื่อเพิ่มความรู้สึกในการเล่น

แนวคิดการออกแบบ:

* ใช้หลัก **Inheritance** ให้บอสทุกตัวมีโครงสร้างพื้นฐานร่วมกัน
* สนับสนุนแนวคิด **Composition** โดยการแยกการยิงกระสุนให้เป็น ScriptableObjec

1. **คลาส BossA** และ **คลาส BossB**

เป็นคลาสที่สืบทอดจาก BossPattern เพื่อสร้างพฤติกรรมเฉพาะของบอสแต่ละตัว เช่น ลำดับการยิงกระสุนเฉพาะของตนเอง เอฟเฟกต์ หรือเงื่อนไขในการเข้าสู่เฟสถัดไป

**แนวคิดในการออกแบบ:**

* ใช้ OOP เพื่อสนับสนุนการเพิ่มบอสใหม่ในอนาคต โดยไม่แตะโค้ดกลาง

1. **คลาส BulletPatternSO**

เป็น ScriptableObject แบบนามธรรมที่กำหนดโครงสร้างของรูปแบบการยิงกระสุน โดย บอสแต่ละตัวจะยิงผ่านการเรียกใช้แพทเทิร์นของคลาสนี้แทนการเขียนคำสั่งยิงเอง

**หน้าที่หลัก:**

* นิยามฟังก์ชัน Execute() ที่จะถูก override เพื่อยิงกระสุนในรูปแบบต่าง ๆ

**แนวคิดในการออกแบบ:**

* ใช้ **ScriptableObject** เพื่อให้สามารถออกแบบ/สลับแพทเทิร์นใน Inspector ของ Unity ได้โดยไม่ต้องเขียนโค้ดใหม่
* สนับสนุนแนวคิด **Open-Closed Principle** เพื่อเพิ่มแพทเทิร์นใหม่ได้โดยไม่ต้องแตะโค้ดบอส

1. **คลาส BulletRingsPattern**, **SpiralShotPattern** และ **WaveAttackPattern**

เป็นคลาสที่สืบทอดจาก BulletPatternSO เพื่อกำหนดรูปแบบการยิงกระสุน

* BulletRingsPattern ยิงกระสุนวนเป็นวงกลมหลายชั้น
* SpiralShotPattern ยิงแบบเกลียว หมุนออกจากจุดศูนย์กลาง
* WaveAttackPattern ยิงกระสุนกวาดไปมาเป็นลูกคลื่น

**แนวคิดในการออกแบบ:**

* ใช้ **Polymorphism** เพื่อให้สามารถเรียกใช้ผ่าน BulletPatternSO เดียวกันได้หมด

1. **คลาส BossBullet**

เป็นคลาสที่ควบคุมการเคลื่อนไหวและการชนของกระสุน เช่น ความเร็ว ทิศทาง การกระเด้ง และการตรวจจับการชนกับ Player หรือ Bullet อื่น ๆ

**แนวคิดในการออกแบบ:**

* ใช้แยกจาก BulletPatternSO เพื่อให้สามารถใช้กระสุนเดียวกันได้กับแพทเทิร์นหลายแบบหรือให้แพทเทิร์นแต่ละอันที่เป็นแพทเทิร์นประเภทเดียวกันใช้กระสุนต่างกันได้

1. **คลาส Puzzle**

เป็นคลาสนามธรรมที่ทำหน้าที่เป็นแม่แบบของระบบปริศนา (Puzzle System) ภายในเกม โดยมีเป้าหมายหลักคือการออกแบบโครงสร้างที่สามารถนำไปต่อยอดสร้างปริศนาแต่ละแบบได้อย่างยืดหยุ่นและไม่ซ้ำซ้อน โดยในคลาสนี้จะนิยามเมธอดสำคัญ 2 ตัว คือ InitPuzzle() สำหรับเริ่มต้นปริศนา และ EndPuzzle() สำหรับสิ้นสุดปริศนา นอกจากนี้ยังมีการเชื่อมต่อไปยังคลาสสำคัญอื่น ๆ เช่น

* Player เพื่อควบคุมการตอบสนองของผู้เล่น
* BossPattern เพื่อสั่งให้บอสหยุดยิงหรือทำงานร่วมกับปริศนา

**แนวคิดในการออกแบบ:**

* ใช้ **Abstract Class** เพื่อให้ทุกปริศนาใหม่สามารถสืบทอดและบังคับใช้ฟังก์ชันสำคัญได้
* สนับสนุนหลักการ **Polymorphism** ทำให้ระบบสามารถสลับปริศนาได้แบบ Dynamic โดยไม่ต้องแก้โค้ดส่วนกลาง
* ลด Coupling กับระบบอื่น โดยให้เฉพาะตัว Puzzle เป็นผู้จัดการกระบวนการของมันเอง

1. **คลาส FindDifferentPuzzle และ คลาส GatlingGunPuzzle**

เป็นคลาสที่สืบทอดจากคลาสนามธรรม Puzzle โดยมีจุดประสงค์เพื่อสร้างรูปแบบของปริศนาหลากหลายที่สามารถถูกใช้งานร่วมกับระบบของเกมได้อย่างยืดหยุ่น

**แนวคิดในการออกแบบ:**

* ใช้หลักการ **Inheritance** เพื่อให้ทุกปริศนาใหม่สามารถสืบทอดโครงสร้างพื้นฐานจาก Puzzle
* สนับสนุนแนวคิด **Polymorphism** เพื่อให้ระบบกลาง (เช่น PuzzleSpawner) สามารถเรียกใช้งานได้โดยไม่รู้ว่าเป็นปริศนาแบบใด
* แยก Logic ของแต่ละปริศนาออกจากกัน เพื่อเพิ่มความง่ายในการขยายเพิ่มเติมในอนาคต

1. **คลาส Player**

เป็นหัวใจสำคัญของเกมที่รับผิดชอบการควบคุมตัวละครผู้เล่น ทั้งด้านการเคลื่อนที่ การกระโดด การแดช การรับความเสียหาย รวมถึงการปฏิสัมพันธ์กับสิ่งแวดล้อมต่าง ๆ ภายในเกม เช่น การเก็บ ไอเทม การชนกับศัตรู หรือเข้าสู่ปริศนา

**แนวคิดในการออกแบบ:**

* ใช้หลักการ **Encapsulation** เพื่อควบคุมการเข้าถึงฟังก์ชันภายในอย่างปลอดภัย
* แยกฟังก์ชันต่าง ๆ ออกให้ชัดเจน เช่น การเคลื่อนที่ การแดช การชน ฯลฯ เพื่อให้สามารถ debug และขยายเพิ่มเติมได้ง่าย
  1. การออกแบบส่วนต่อประสานผู้ใช้ (User Interface Design)

การออกแบบในเกมกระโดดหลบ Satori : Enso จะมีการดีไซน์โดยยึดธีมสีขาวดำเป็นหลัก ภายในเกมนั้นจะมีส่วนต่อประสานผู้ใช้อยู่ 4 หน้า ดังนี้

* + 1. **การออกแบบส่วนต่อประสานผู้ใช้ของหน้าเมนู**

หน้าจอเมนูจะประกอบไปด้วย 2 ข้อความ คือ Start และ Option เพื่อให้ผู้เล่นกด หากผู้เล่นกด Start เกมก็จะพาผู้เล่นไปเริ่มเกมทันที หากผู้เล่นกด Option เกมก็จะพาผูเล่นไปหน้าต่างตั้งค่า

รูปภาพประกอบด้วย เมฆ, ดำและขาว, หมอก, ร่าง

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

ภาพที่ 3.3 ส่วนต่อประสานผู้ใช้ของหน้าจอเมนู

* + 1. **การออกแบบส่วนต่อประสานผู้ใช้ของหน้าจอตั้งค่า**

หน้าต่างตั้งค่าจะไปประกอบไปด้วย การตั้งค่าความละเอียดหน้าจอ และ ตั้งค่าระดับความดังของเสียง การตั้งค่าความละเอียด จะเป็นตัวเลือก Dropdown list ให้ผู้เล่นเลือกจากรายการที่เป็นไปได้ การตั้งค่าระดับความความดังของเสียงจะเป็น Slider ให้ผู้เล่นลากไปมาเพื่อปรับระดับ เมื่อกด Apply ก็จะเป็นการตกลง และทำการเซฟการตั้งค่านี้ไปใช้ทันที

ภาพที่ 3.4 ส่วนต่อประสานผู้ใช้ของหน้าจอตั้งค่า

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ดำและขาว, ขาวดำ

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

* + 1. **การออกแบบส่วนต่อประสานผู้ใช้ของหน้าจอสอนการเล่น**

หน้าต่างสอนการเล่นจะแสดงถึงการกระทำต่าง ๆ ที่เป็นไปได้ทั้งหมดของผู้เล่น เพื่อสอนการควบคุมให้กับผู้เล่นที่เริ่มเกมครั้งแรก โดยมีการบอกปุ่มที่ผู้เล่นต้องกดเพื่อออกจากการสอนการเล่น

ภาพที่ 3.5 ส่วนต่อประสานผู้ใช้ของหน้าจอสอนการเล่น

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ตัวอักษร, แผนภาพ

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

* + 1. **การออกแบบส่วนต่อประสานผู้ใช้ของหน้าจอเล่นเกม**

หน้าจอเล่นเกมจะประกอบไปด้วย ตัวละครของผู้เล่น บอส หลอดเลือดของบอส กระสุนของบอส พื้นแท่นเหยียบ ปริศนาภายในเกม เวลาที่เหลืออยู่ของการแก้ปริศนาและไอเทมที่จะสุ่มปรากฎขึ้นมา

ภาพที่ 3.6 ส่วนต่อประสานผู้ใช้ของหน้าจอเล่นเกม

รูปภาพประกอบด้วย ภาพหน้าจอ, ดาราศาสตร์

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

* 1. การออกแบบด่าน(Level Design)

ในเกมกระโดดหลบ Satori : Enso จะประกอบไปด้วยด่าน 2 ด่าน เพื่อให้ผู้เล่นเอาชนะ

* + 1. **ด่านที่หนึ่ง**

โดยด่านแรกเพื่อให้ผู้เล่นค่อยๆได้ปรับตัวกับตัวเกมก่อน จึงมีลักษณะด่านที่เรียบง่าย และศัตรูจะปล่อยกระสุนที่ไม่ค่อยมีความยากมากนัก ดังภาพที่ 3.7

รูปภาพประกอบด้วย ภาพหน้าจอ

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

ภาพที่ 3.7 การออกแบบด่านในเกมด่านที่หนึ่ง

โดยเส้นสีแดงและสีเหลืองคือเส้นทางของกระสุนที่บอสจะปล่อยออกมา และเส้นสีฟ้าและเขียวคือเส้นทางที่บอสจะใช้ในการเคลื่อนที่

* + 1. **ด่านที่สอง**

ในด่านที่สองหลังจากผู้เล่นคุ้นเคยกับเกมแล้ว ผู้พัฒนาจึงทำการเพิ่มความซับซ้อนของด่านให้มากขึ้นโดยการทำแท่นเหยียบที่เคลื่อนไหวได้ กระสุนที่ยิงออกมาจากจุดอื่นนอกเหนือจากบอสและการยิงกระสุนของบอสที่ยุ่งยากมากขึ้น ดังภาพที่ 3.8

รูปภาพประกอบด้วย ภาพหน้าจอ, ไลน์, แผนภาพ

เนื้อหาที่สร้างโดย AI อาจไม่ถูกต้อง

ภาพที่ 3.8 การออกแบบด่านในเกมด่านที่สอง

1. ผลการวิจัย

ในบทนี้จะกล่าวถึงขั้นตอนการพัฒนาระบบ และการทดสอบระบบของ เกมกระโดดหลบ “Satori : Enso” บน Windows

4.1 เทคนิคการพัฒนาระบบ

ผู้จัดทำได้เลือกใช้ภาษา C# ซึ่งเป็นภาษาที่น่าสนใจและง่ายต่อการพัฒนา โดยเครื่องมือที่ใช้พัฒนาโปรแกรมคือ Visual Studio Code ซึ่งมีส่วนเสริมที่หลากหลายและสามารถใช้งานร่วมกับ Github ได้

A screenshot of a computer

AI-generated content may be incorrect.

**ภาพ 4.1 หน้าจอของโปรแกรม Visual Studio Code**

ในการใช้งาน Visual Studio Code ร่วมกับ GitHub ผู้จัดทำได้ติดตั้งส่วนเสริมที่ชื่อว่า "GitHub Pull Requests” ซึ่งช่วยให้สามารถเชื่อมต่อกับบัญชี GitHub ได้โดยตรงจากในโปรแกรม นอกจากนี้ VSCode ยังมีฟีเจอร์ Git ในตัวที่สามารถใช้งานได้อย่างสะดวก เช่น การ commit การ push และการ pull โค้ด รวมไปถึงการดูความเปลี่ยนแปลงของไฟล์ (diff) ได้แบบเรียลไทม์ ซึ่งช่วยให้การจัดการเวอร์ชันของโค้ดเป็นไปอย่างมีประสิทธิภาพและเหมาะสมกับการทำงานเป็นทีม

A screenshot of a computer

AI-generated content may be incorrect.

**ภาพที่ 4.2 การใช้ Github ร่วมกับ Visual Studio Code**

4.2 เทคนิคการพัฒนาโปรแกรม

โค้ด (Code) ต่อไปนี้คือส่วนหนึ่งของเกมกระโดดหลบ “Satori : Enso” บน Windows

ในไฟล์ Player.cs เมื่อผู้เล่นกดอินพุทเดินทางซ้ายหรือเดินทางขวา ความเร็วของผู้เล่นจะเพิ่มขึ้นในทิศทางของอินพุท ตามโค้ด body.velocity = new Vector2(horizontalInput \* (horizontalSpeed + bonusHorizontalSpeed), body.velocity.y); และเมื่อผู้เล่นกดอินพุทกระโดด ที่ if ((Input.GetKeyDown(KeyCode.Space) || Input.GetKeyDown(KeyCode.W) || Input.GetKeyDown(KeyCode.UpArrow)) && CheckJump()) จะเช็คว่าผู้เล่นยังเหลือจำนวนครั้งในการกระโดดหรือไม่ หากยังเหลืออยู่จะทำการกระโดด ดังภาพที่ 4.3

float horizontalInput = Input.GetAxis("Horizontal") \* (CheckMovementReversed() ? -1 : 1);

        body.velocity = new Vector2(horizontalInput \* (horizontalSpeed + bonusHorizontalSpeed), body.velocity.y);

        if (horizontalInput > 0.01f)

            transform.localScale = new Vector3(playerScale.x, playerScale.y, playerScale.z);

        else if (horizontalInput < -0.01f)

            transform.localScale = new Vector3(-playerScale.x, playerScale.y, playerScale.z);

        if ((Input.GetKeyDown(KeyCode.Space) || Input.GetKeyDown(KeyCode.W) || Input.GetKeyDown(KeyCode.UpArrow)) && CheckJump())

        {

            isDiving = false;

            currentJump--;

            body.velocity = new Vector2(body.velocity.x, jumpSpeed + bonusJumpSpeed);

            PlaySound(jumpSound);

            anim.SetBool("OnGround", false);

        }

**ภาพที่ 4.3 โค้ดที่จะทำงานเมื่อผู้เล่นเคลื่อนที่**

เมื่อบอสเริ่มโจมตี บอสจะยิงชุดกระสุนตามเวลาที่กำหนด จากโค้ด yield return new WaitForSeconds(config.delayBeforeStart); จะทำให้รอจนกว่าเวลาไปครบตามที่กำหนด ก่อนที่โค้ด yield return ExecutePattern(config); จะทำให้บอสยิงชุดกระสุน ดังภาพที่ 4.4

protected override IEnumerator AttackPattern()

{

while (true)

{

foreach (var config in bulletPatterns)

{

yield return new WaitForSeconds(config.delayBeforeStart);

yield return ExecutePattern(config);

}

}

}

**ภาพที่ 4.4 โค้ดที่จะทำงานเมื่อบอสเริ่มโจมตี**

หาก Collider ของ Player ชนกับ Object ที่มี Tag คือ Bullet โค้ด if(bonusShieldCount > 0) จะเช็คว่าผู้เล่นมีโล่ที่ได้ไอเทมเหลืออยู่หรือไม่ หากไม่มี โค้ด if (immortal == false) จะเช็คว่าผู้เล่นเป็นอมตะอยู่หรือไม่ หากไม่ ผู้เล่นจะตาย ดังภาพที่ 4.5

if (collision.gameObject.CompareTag("Bullet"))

{

PlaySound(hitSound);

if (bonusShieldCount > 0)

{

Destroy(collision.gameObject);

return;

}

if (immortal == false)

Death();

}

**ภาพที่ 4.5 โค้ดที่จะทำงานเมื่อผู้เล่นชนกระสุน**

เมื่อบอส (Boss) ได้รับความเสียหายจากกลไก พลังชีวิตของบอสจะลดลงตามความเสียหายที่กลไกนั้นสร้าง และแสดงพลังชีวิตที่บอสเหลือผ่านหลอดพลังชีวิต (HealthBar) ตามโค้ด healthBarScript.SetCurrentValue(currentHealth); หากพลังชีวิตบอสกลายเป็น 0 โค้ด Invoke(nameof(Death), 0.5f); จะทำให้บอสตายและเข้าสู่ฉากถัดไปหลังจากเวลาผ่านไป 0.5 วินาที ดังภาพที่ 4.6

public void TakeDamage(float damage)

{

Debug.LogError("Hit " + gameObject.name);

currentHealth -= damage;

Debug.Log("Hp:" + currentHealth);

healthBarScript.SetCurrentValue(currentHealth);

if (shakeCoroutine != null)

StopCoroutine(shakeCoroutine);

shakeCoroutine = StartCoroutine(ShakeDamageEffect(damage));

if (currentHealth <= 0) {

Invoke(nameof(Death), 0.5f); // Delay scene transition

}

}

**ภาพที่ 4.6 โค้ดที่จะทำงานเมื่อบอสได้รับความเสียหาย**

4.3 การทดสอบระบบ

ผู้จัดทำได้ทำการทดสอบระบบเกมกระโดดหลบ “Satori : Enso” บน Windows โดยให้ผู้เล่นทดลองเล่นเกมกระโดดหลบ “Satori : Enso” บน Windows อย่างน้อย 15 นาที ก่อนให้ผู้เล่นประเมินผลตามความพึงพอใจ โดยผู้ทดสอบเป็นทั้งผู้หญิงและผู้ชาย อยู่ในช่วงอายุ 18-25 ปี จำนวนทั้งหมด 11 คน แบ่งเป็นผู้ชาย 8 คน ผู้หญิง 3 คน

4.4 ผลทดสอบ

**4.4.1 ผลการทดลองของผู้ที่เล่นเกม**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ระดับความพึงพอใจโดยรวมต่อเกม | ผลประเมินแบ่งตามระดับความพอใจ (คน) | | | | |
| 1  แย่มาก | 2 | 3 | 4 | 5  ยอดเยี่ยม |
| 0 | 1 | 5 | 3 | 2 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ระดับความสนุกจากการเล่นเกม | ผลประเมินแบ่งตามระดับความสนุก (คน) | | | | |
| 1  ไม่สนุกเลย | 2 | 3 | 4 | 5  สนุกมาก |
| 1 | 1 | 3 | 4 | 1 |
| ระดับความยากในการเอาชนะเกม | ผลประเมินแบ่งตามระดับความยาก (คน) | | | | |
| 1  ยากเกินไป | 2 | 3 | 4 | 5  ง่ายเกินไป |
| 5 | 4 | 1 | 0 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ฉันเข้าใจจุดมุ่งหมายที่ต้องทำในเกมได้อย่างง่ายดาย | ผลประเมินแบ่งตามระดับความพอใจ (คน) | | | | |
| 1  เห็นด้วยอย่างยิ่ง | 2 | 3 | 4 | 5  ไม่เห็นด้วยอย่างยิ่ง |
| 0 | 1 | 5 | 3 | 2 |

**4.4.2 ข้อปรับปรุง**

จากผลสำรวจ สิ่งที่ควรปรับปรุงมีดังต่อไปนี้

* + ระดับความยากมากเกินไป
  + Tutorial เข้าใตยากเกินไป
  + ความเร็วการเคลื่อนของผู้เล่นมากเกินไป ทำให้ยากต่อการควบคุม

บทที่ 5  
ข้อสรุปและข้อเสนอแนะ

จากการพัฒนาเกมกระโดดหลบ “Satori : Enso” บน Windows ผู้จัดทำสามารถสรุปผลการดำเนินโครงงาน รวมถึงปัญหาและอุปสรรคที่พบระหว่างการดำเนินโครงงาน วิธีแก้ปัญหาและข้อเสนอแนะเพื่อเป็นแนวทางสำหรับผู้ที่มีความสนใจหรือผู้ที่จะนำโครงงานนี้ไปพัฒนาต่อไปในอนาคตดังต่อไปนี้

5.1 สรุปผลที่ได้

จากการพัฒนาเกมกระโดดหลบ “Satori : Enso” บน Windows ประโยชน์ที่ได้รับแบ่งออกเป็นประโยชน์สำหรับผู้พัฒนา และประโยชน์สำหรับผู้เล่น

**5.1.1 ผู้พัฒนา**

ผู้พัฒนาได้รับความรู้ในการพัฒนาซอฟท์แวร์บนระบบปฏิบัติการ Windows ได้รับความรู้การทำงานเป็นทีมร่วมกับผู้อื่นผ่านการใช้ Github และ Trello ได้เรียนรู้การเขียนโปรแกรมด้วยภาษา C# ผ่านโปรแกรม Visual Studio Code ได้เรียนรู้การใช้งานโปรแกรมพัฒนาเกม Unity ซึ่งในอนาคต และได้เรียนรู้การใช้งานโปรแกรม Photoshop ในการออกแบบภาพในเกม

**5.1.2 ผู้เล่น**

ผู้เล่นได้รับความสนุกสนานในการเล่นเกม ได้ฝึกความเร็วการตอบสนองในการหลบกระสุน และไหวพริบในการแก้ปริศนา

5.2 ปัญหาและอุปสรรค

**5.2.1 ปัญหาก่อนเริ่มพัฒนา**

ในการเลือกโปรแกรมสำหรับพัฒนาเกมนั้นเป็นเรื่องค่อนข้างยาก เนื่องจากมีโปรแกรมสำหรับพัฒนาเกมที่หลากหลาย จากการศึกษาแต่ละโปรแกรม ผู้พัฒนาได้เลือกใช้ Unity เนื่องจากมีความง่ายต่อการใช้งาน และสามารถใช้งานได้ฟรี และใช้ Visual Studio Code ในการเขียนโปรแกรมเนื่องจากมีส่วนเสริมที่หลากหลายและช่วยจัดระเบียบโค้ดได้ด้วย นอกจากนี้การเลือก Art Style และโปรแกรมออกแบบภาพเพื่อใช้ในเกมก็เป็นเรื่องที่ท้าทายเช่นเดียวกัน ผู้พัฒนาได้เลือก Art Style แบบภาพพู่กัน เนื่องจากง่ายต่อการออกแบบ และช่วยเพิ่มความน่าสนใจของธีมเกม ผู้พัฒนาเลือกใช้ Photoshop ในการออกแบบภาพในเกม เนื่องจากมีประสบการณ์ในการใช้งานอยู่แล้ว และมี Generative AI ช่วยในการออกแบบด้วย

**5.2.2 ปัญหาหลังเริ่มพัฒนา**

ในการทำงานร่วมกันนั้นมีอุปสรรคหลายอย่าง ไม่ว่าจะเป็นการสื่อสารถึงงานที่ต้องทำ งานที่กำลังทำอยู่ และงานที่ทำเสร็จแล้ว ทำให้อาจทำงานที่เพื่อนร่วมงานกำลังทำอยู่หรือทำเสร็จแล้ว ส่งผลให้การทำงานมีประสิทธิภาพน้อย หรือบางครั้งการเขียนโค้ดของเพื่อนร่วมงานอาจไม่สมเหตุสมผลหรือไม่เป็นระเบียบตามความคิดของเพื่อนร่วมงานอีกคน ทำให้ยากต่อการทำความเข้าใจและพัฒนาเพิ่มเติม นอกจากนี้ผู้พัฒนามีความรู้ในการออกแบบภาพในเกมน้อยมาก ทำให้เสียเวลาในการออกแบบนาน

5.3 วิธีการแก้ปัญหา

1. ผู้พัฒนาใช้ Trello ในการสื่อสารขั้นตอนในการทำงานต่าง ๆ ให้เป็นการ์ด และจัดการ์ดให้อยู่ในหมวดหมู่งานที่ต้องทำ งานที่กำลังทำอยู่ หรืองานที่ทำเสร็จแล้ว

2. ผู้พัฒนาใช้ Github ในการระงับไม่ให้ผู้พัฒนาสามารถ Merge Pull Request เข้า Branch main ได้เอง โดยต้องผ่านการ Review ของเพื่อนร่วมงานอีกคนก่อน

3. ฝึกการออกแบบภาพ และใช้ Generative AI ช่วยในการออกแบบภาพ

5.4 ข้อเสนอแนะ

ผู้พัฒนามีข้อเสนอสำหรับผู้ที่ต้องการพัฒนาเกมกระโดดหลบ “Satori : Enso” บน Windows

1. ออกแบบไม่ให้ความยากมากเกินไปสำหรับด่านแรก

2. สร้างระบบ Tutorial เพื่อสอนผู้เล่นในการควบคุมตัวละครในเกมทีละอย่าง เพื่อให้ผู้เล่นเข้าใจได้ง่ายขึ้น

3. ออกแบบภาพไอเทมให้เข้าใจความหมายได้ง่ายขึ้น หรือสร้างฉากเพื่อให้ผู้เล่นอ่านข้อมูลของไอเทมต่าง ๆ ได้

รายการอ้างอิง

การเขียนรายการอ้างอิง ให้ดูตาม “คู่มือการพิมพ์วิทยานิพนธ์ 2548” บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย เลือกใช้แบบใดแบบหนึ่ง (ตามตัวเลขหรือตามชื่อสกุล)

http://www.grad.chula.ac.th/download/thesis.pdf

ถ้าภาษาอังกฤษ ให้ดูตาม “Thesis Handbook 2005”

http://www.grad.chula.ac.th/download/files/30\_Thesis\_Handbook.pdf

[1] A. Amresh, A.R. Carberry, J. Femiani, “Evaluating the effectiveness of flipped classrooms for teaching CS1”, Proceeding of Frontiers in Education Conference, IEEE, pp. 733-735, 2013

Wanvarie, D., Takamura, H., and Okumura, M. 2010. Active learning for sequence labelling with probability re-estimation. In Zhang, B.-T. and Orgun, M. A. (ed.), PRICAI, volume 6230 of Lecture Notes in Computer Science, pp. 681–686. Heidelberg, Germany: Springer.

**ภาคผนวก**

ภาคผนวก ก  
แบบเสนอหัวข้อโครงงาน รายวิชา 2301399 Project Proposal  
ปีการศึกษา 2567

ชื่อโครงงาน (ภาษาไทย) เกมแอปพลิเคชัน : เกมกระโดดหลบ “Satori : Enso” บน Windows 11

ชื่อโครงงาน (ภาษาอังกฤษ) Game Application : Dodging Game “Satori : Enso” on Windows 11

อาจารย์ที่ปรึกษา อาจารย์ ดร.ทรรปณ์ ปณิธาณะรักษ์

ผู้ดำเนินการ 1. ปริญญา ตุ้มฉาย เลขประจำตัวนิสิต 6434449023

2. นภสินธุ์ ต่อศิริสกุลวงศ์ เลขประจำตัวนิสิต 6434440223

สาขาวิชาวิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

หลักการและเหตุผล

ในปัจจุบันเกมที่ได้รับความนิยมในหมู่ผู้เล่นนั้นมีหลากหลายประเภทตั้งแต่เกมแอคชันที่ต้องการการ ตอบสนองที่รวดเร็วไปจนถึงเกมที่ต้องใช้การวางแผนหรือเกมที่ใช้ความคิดสร้างสรรค์ได้อย่างอิสระ อย่างไรก็ตามยังมีกลุ่มผู้เล่นสาย “ Hardcore[[1]](#footnote-1) ” ที่ชื่นชอบเกมที่มีความท้าทายสูงและมีความต้องการเกม ที่สามารถผสมผสานกลไกการเล่นหลายรูปแบบได้อย่างลงตัวเพื่อทดสอบทั้งทักษะทั้งทางการตอบสนองของ ร่างกายและความคิดภายใต้สถานการณ์กดดันไปพร้อมกัน

เกมประเภท Bullet Hell เช่น Touhou Project [1] และ Platform Fighter แบบ Super Smash Bros. [2] ดังรูปที่ 1 และ 2 ตามลำดับ ถือเป็นตัวอย่างที่ชัดเจนของเกมที่มอบความท้าทายเหล่านี้ให้ผู้เล่น ในเกมประเภท Bullet Hell[[2]](#footnote-2) ผู้เล่นจำเป็นต้องหลบกระสุนจำนวนมากที่พุ่งเข้ามาจากทุกทิศทุกทาง อย่างต่อเนื่อง ซึ่งต้องอาศัยทักษะการตอบสนองที่รวดเร็วรวมทั้งการคาดการณ์ล่วงหน้าถึงทิศทางของกระสุน ที่จะเคลื่อนที่และปรากฏออกมาขณะที่เกม Platform Fighter อย่าง Super Smash Bros. จะเน้นการโจมตี และการหลบหลีกการโจมตีของคู่ต่อสู้ภายใต้สภาพแวดล้อมที่สามารถเปลี่ยนแปลงไปได้ ซึ่งต้องใช้ การวางแผนเชิงกลยุทธ์และการตอบสนองที่รวดเร็วในการโต้กลับในการเอาชนะคู่ต่อสู้

รูปภาพประกอบด้วย เกมพีซี, แอนิเมชัน, วิดีโอเกมกลยุทธ์, การ์ตูนแอนิเมชัน

คำอธิบายที่สร้างโดยอัตโนมัติ

รูปที่ 2 Super Smash Bros. Ultimate

รูปภาพจาก

https://www.bbc.com/news/newsbeat-67407712

รูปภาพประกอบด้วย ข้อความ, ภาพหน้าจอ, ม่วง, สีม่วง

คำอธิบายที่สร้างโดยอัตโนมัติ

รูปที่ 1 Touhou 14

รูปภาพจากhttps://steamcommunity.com/sharedfiles/filedetails/?l=thai&id=173745051

นอกจากนี้การผสมผสานกลไกของ Bullet Hell และ Platform Fighter เข้าด้วยกันจะนำไปสู่ ประสบการณ์ใหม่ที่จะช่วยตอบสนองความต้องการของกลุ่มผู้เล่น Hardcore ที่ต้องการความท้าทาย ที่หลากหลายมากขึ้น ส่วนการใช้มุมมองและฟิสิกส์แบบ Platform Fighter ช่วยให้ผู้เล่นรู้สึกมีอิสระ ในรูปแบบการเคลื่อนไหวแต่การออกแบบเกมเพลย์ที่ให้เน้นการหลบหลีกกระสุนแบบ Bullet Hell จะเพิ่มระดับความยากขึ้นอย่างมากเนื่องจากผู้เล่นจำเป็นต้องคิดให้ดีก่อนการกระโดดเคลื่อนไหวแต่ละครั้ง และยังจำเป็นต้องคำนึงถึงสภาพ Platform ของด่านเพื่อใช้ในรีเซ็ตแอคชันในการเคลื่อนไหวสำหรับการหลบ การโจมตีต่อไป ผู้เล่นจะต้องเผชิญหน้ากับการโจมตี ที่ซับซ้อนจากศัตรูประจำด่านที่จะยิงกระสุนจำนวนมากเข้ามาหาผู้เล่นอยู่ตลอดเวลา ผู้เล่นไม่เพียงแต่จะต้องหลบหลีกแต่ยังต้อง หาจังหวะที่เหมาะสมในการโจมตีสวนกลับซึ่งจะเป็นการผสมผสานระหว่างทักษะการหลบหลีก ความเร็วในการตัดสินใจ และความสามารถในการอ่านสถานการณ์และคาดการณ์ล่วงหน้า ให้ถูกต้องเข้าด้วยกัน

การสร้างเกมที่มีระบบการเล่นผสมผสานเช่นนี้ ไม่เพียงแต่จะช่วยสร้างความสดใหม่ในวงการเกม แต่ยังช่วยให้ผู้เล่นได้รับประสบการณ์การเล่นที่ท้าทายที่มากยิ่งกว่าเดิม ซึ่งจะเป็นสิ่งที่ดึงดูดผู้เล่นกลุ่ม Hardcore ที่แสวงหาความท้าทายที่แปลกใหม่ อีกทั้งยังสามารถดึงดูดกลุ่มผู้เล่น หน้าใหม่ที่สนใจในการ ท้าทายขีดความสามารถของตนเองในการเล่นเกมที่ต้องใช้ทักษะรอบด้าน

วัตถุประสงค์

เพื่อพัฒนาเกม PC ที่มีรูปแบบการเล่นแบบเล่นคนเดียว ซึ่งจะผสมผสานเกมแอคชันที่รวดเร็ว เข้ากับเกมปริศนาเพื่อทดสอบทั้งความสามารถในการตอบสนองและการคิดวิเคราะห์ของผู้เล่น โดยใช้กลไกและมุมมองการเล่นแบบ Platform Fighter ที่ได้รับแรงบันดาลใจจาก Super Smash Bros. และ Bullet Hell จาก Touhou Project เพื่อพัฒนาและทดสอบทักษะทั้งสองด้าน ให้กับผู้เล่น

ขอบเขตของโครงงาน

ขอบเขตของปัญหา ข้อจำกัดในการทำโครงงาน เช่น สนใจเฉพาะกรณีที่คำตอบเป็นจำนวนเต็มเท่านั้น หรือ โปรแกรมทำงานได้เฉพาะระบบปฏิบัติการ Android หรือ ศึกษาเฉพาะกลุ่มตัวอย่างที่เป็นเพศชาย เป็นต้น

วิธีการดำเนินงาน

ขั้นตอนการดำเนินงาน

1. วิเคราะห์และออกแบบระบบเกม (System Analysis and Design)

ศึกษาเป้าหมายและความต้องการของกลุ่มเป้าหมาย กำหนดขอบเขตของโปรเจค เช่น จำนวนเลเวล

ที่ต้องการและความซับซ้อนของระบบ รวมทั้งศึกษาและเตรียมการเลือกเครื่องมือที่จะใช้ในการพัฒนา เช่น Unity หรือ Unreal หรือการเลือกใช้เครื่องมือออกแบบกราฟิกต่าง ๆ เพื่อให้กระบวนการพัฒนาง่ายและมีประสิทธิภาพมากขึ้น

1. การออกแบบเกมเพลย์และเลเวล (Gameplay and Level Design)

กำหนดกลไกการเล่น (Game Mechanics) ว่าผู้เล่นจะควบคุมตัวละครหรือโต้ตอบกับสิ่งต่าง ๆ ในเกมอย่างไร เช่น การกระโดด การหลบหลีก หรือการแก้ปริศนา ต่อมาจะออกแบบสตอรี่บอร์ด (Storyboard) เพื่อจัดลำดับเหตุการณ์หรือความท้าทายต่าง ๆ ในเกม และกำหนดจุดเริ่มต้น จุดสิ้นสุด รวมถึงความยากง่ายของแต่ละเลเวล นอกจากนี้ยังมีการออกแบบกราฟิก เช่น การกำหนดธีมของเกมว่าจะใช้สไตล์ภาพแบบไหน (เช่น Pixel Art หรือ Hand-Drawn) พร้อมกับออกแบบเสียงประกอบ เช่น เพลงพื้นหลังและเสียงเอฟเฟกต์ เพื่อสร้างบรรยากาศที่สอดคล้องกับธีมเกม

1. พัฒนาโปรแกรม (Develop)

เขียนโปรแกรมเพื่อเปลี่ยนไอเดียให้เป็นเกมที่เล่นได้จริง โดยจะเริ่มจากการเขียนโค้ดระบบย่อย เช่น การควบคุมตัวละคร ระบบฟิสิกส์ และการชนวัตถุ (Collision Detection) จากนั้นนำระบบเหล่านี้มาประกอบกันในเอนจินเกม รวมถึงพัฒนาส่วนของ UI/UX เช่น เมนูหลักและหน้าจอการตั้งค่า รวมถึงการจัดการทรัพยากร เช่น การนำเข้าไฟล์กราฟฟิก เสียง และฟอนต์ต่าง ๆ เพื่อให้ทุกองค์ประกอบเชื่อมโยงกันอย่างราบรื่นภายในเกม

1. ทดสอบการทำงาน (Test and Debug)

เพื่อให้แน่ใจว่าเกมสามารถทำงานได้ตามที่ออกแบบไว้ เราจะทดสอบฟีเจอร์ทุกอย่าง เช่น การควบคุมตัวละคร การเปลี่ยนฉาก และระบบฟิสิกส์ โดยใช้เครื่องมือ Debug ของเอนจิน เช่น Unity Console เพื่อตรวจสอบข้อผิดพลาดที่เกิดขึ้น และแก้ไขปัญหาเหล่านั้น นอกจากนี้ยังต้องทดสอบประสิทธิภาพของเกม เช่น ตรวจสอบว่ามีปัญหาเรื่องการใช้หน่วยความจำหรือไม่ และเกมสามารถทำงานได้ที่ความเร็วเฟรมเรต (FPS) คงที่หรือไม่

1. ทดสอบกลุ่มตัวอย่าง (Sample Test)

นำเกมไปให้กลุ่มตัวอย่าง เช่น นักเรียน นักศึกษา หรือผู้ที่ชื่นชอบการเล่นเกมเล่น กลุ่มนี้จะทดลองเล่นเกมในสภาพแวดล้อมจริงและให้ข้อเสนอแนะเกี่ยวกับประสบการณ์การเล่น ความสนุก และข้อผิดพลาดต่าง ๆ ที่พวกเขาพบ ข้อเสนอแนะเหล่านี้จะถูกบันทึกและนำไปใช้ในการปรับปรุงเกมให้เหมาะสมยิ่งขึ้น

1. ประเมินและปรับปรุง (Evaluate and Refine)

ทำการวิเคราะห์ว่ามีฟีเจอร์ใดในเกมที่ยังขาดความสมบูรณ์ หรือส่วนใดที่ไม่ตอบโจทย์ผู้เล่น นำผลลัพธ์เหล่านี้มาปรับปรุง เช่น ปรับสมดุลของเกม (Game Balance) เพิ่มความหลากหลายในเกมเพลย์ หรือแก้ไขปัญหาที่ทำให้ผู้เล่นรู้สึกไม่สะดวก รวมถึงการปรับปรุงประสิทธิภาพของเกม เช่น ลดขนาดไฟล์เกม หรือเพิ่มการรองรับการเล่นในอุปกรณ์หลากหลายประเภท

1. จัดทำรูปเล่มรายงานของโครงงาน (Project Report)

จัดทำเอกสารรายงานโครงงานโดยสรุปทุกขั้นตอนของการพัฒนาออกมา

**ตารางการดำเนินงาน**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ขั้นตอนการดำเนินการ** | **เดือน/ปีการศึกษา 2567** | | | | | | | | |
| **ส.ค.** | **ก.ย.** | **ต.ค.** | **พ.ย.** | **ธ.ค.** | **ม.ค.** | **ก.พ.** | **มี.ค.** | **เม.ย.** |
| 1. System Analysis and Design |  |  |  |  |  |  |  |  |  |
| 2. Gameplay and Level Design |  |  |  |  |  |  |  |  |  |
| 3. Development |  |  |  |  |  |  |  |  |  |
| 4. Testing and Debugging |  |  |  |  |  |  |  |  |  |
| 5. Sample Testing |  |  |  |  |  |  |  |  |  |
| 6. Evaluation and Refinement |  |  |  |  |  |  |  |  |  |
| 7. Project Report Preparation |  |  |  |  |  |  |  |  |  |

ประโยชน์ที่คาดว่าจะได้รับ

**ประโยชน์ต่อผู้พัฒนา**

1. เพิ่มทักษะการพัฒนาเกมในสภาพแวดล้อมจริง ตั้งแต่การออกแบบระบบ ไปจนถึงการเขียนโค้ด และการทดสอบเกม
2. ได้ฝึกทักษะด้านการวิเคราะห์และแก้ไขปัญหาทางเทคนิค และการทำงานร่วมกับซอฟต์แวร์ และเครื่องมือพัฒนาเกม
3. ได้รับประสบการณ์ในการจัดการ แบ่งงาน ทำงานร่วมกับผู้อื่น ในโปรเจคที่สเกลใหญ่มากขึ้น
4. เกมที่พัฒนาขึ้นสามารถใช้เป็นต้นแบบสำหรับการพัฒนาเกมที่ผสมผสานทั้งเกมแอคชันและ ปริศนาในอนาคต
5. เกมนี้อาจมีการต่อยอดไปสู่การพัฒนาเพิ่มเติมในเชิงธุรกิจ โดยนำไปวางจำหน่ายในร้านค้า ออนไลน์ อาทิ Steam หรือใช้ในการพัฒนาเกมที่เน้นการฝึกทักษะต่าง ๆ ของผู้เล่นต่อไป

**ประโยชน์ต่อผู้เล่น**

1. ผู้เล่นสามารถเล่นเพื่อท้าทายตัวเองได้อย่างสนุกสนาน
2. ผู้เล่นสามารถเพิ่มทักษะการตอบสนองและการวางแผนผ่านการเล่นเกมได้
3. ในด้านการศึกษา เกมนี้อาจเป็นต้นแบบสำหรับการวิจัยในเรื่องของพฤติกรรมผู้เล่น ในสถานการณ์ ที่ท้าทายและการตอบสนองต่อความยากในการเล่นเกม

อุปกรณ์และเครื่องมือที่ใช้

วางแผนว่าจะต้องใช้วัสดุ อุปกรณ์ ซอฟต์แวร์ อะไรบ้าง มีค่าใช้จ่ายเท่าไหร่ในการทำโครงงานนี้ หากกำหนด spec ของอุปกรณ์ได้ก็ขอให้ระบุด้วย

งบประมาณ

1. ค่า Asset ที่จะใช้ในเกม 3,000 บาท
2. ค่า ChatGPT เพื่อใช้ช่วยในการทำงาน 2,500 บาท
3. ค่าลงเกมใน Steam 3,499 บาท
4. ค่า SSD Hard disk 2,000 บาท

เอกสารอ้างอิง

หนังสือ บทความ หรือเว็บไซต์

หนังสือ บทความ หรือเว็บไซต์

การเขียนรายการอ้างอิง ให้ดูตาม “คู่มือการพิมพ์วิทยานิพนธ์ 2548” บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย

http://www.grad.chula.ac.th/download/thesis.pdf

ถ้าภาษาอังกฤษ ให้ดูตาม “Thesis Handbook 2005”

http://www.grad.chula.ac.th/download/files/30\_Thesis\_Handbook.pdf

ภาคผนวก ข  
คู่มือการใช้งาน  
(ถ้ามี)

ประวัติผู้เขียน

|  |  |
| --- | --- |
| brown-man-icon | นาย/นางสาว  ประวัติ |
|  |  |
| woman | นาย/นางสาว  ประวัติ |

1. กลุ่มผู้ที่ชื่นชอบความท้าทาย และความรู้สึกในการเอาชนะความท้าทายนั้น [↑](#footnote-ref-1)
2. เกมที่มีกระสุนจำนวนมากที่ผู้เล่นจำเป็นที่ต้องหลบ [↑](#footnote-ref-2)