

**เกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม**

**Strategy Game using Behavior Tree**

**นาย รัฐภูมิ ม้าเพ็ง**

**570510676**

**การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิทยาศาสตรบัณฑิต**

**สาขาวิชาวิทยาการคอมพิวเตอร์**

**คณะวิทยาศาสตร์ มหาวิทยาลัยเชียงใหม่**

**ปีการศึกษา 2560**

# เกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

## Strategy Game using Behavior Tree

นาย รัฐภูมิ ม้าเพ็ง

570510676

การค้นคว้าอิสระนี้ได้รับการพิจารณาอนุมัติให้นับเป็นส่วนหนึ่งของการศึกษา

ตามหลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์ มหาวิทยาลัยเชียงใหม่

ปีการศึกษา 2560

คณะกรรมการสอบการค้นคว้าอิสระ

..... ประธานกรรมการ

อาจารย์ ดร.จักริน ชวชาติ

..... กรรมการ

อาจารย์ ดร.ประภาพร เตชอังกูร

วันที่.....เดือน.....พ.ศ.....

## กิตติกรรมประกาศ

งานค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาตรี วิทยาศาสตร์บัณฑิต สาขาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ ในระบอบนวิชา 204499 ซึ่งผู้ค้นคว้าได้จัดทำในหัวข้อเรื่อง “เกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม”

การค้นคว้าอิสระนี้สำเร็จลงได้ด้วยความกรุณาจากบุคคลหลายท่าน ขอกราบขอบพระคุณ อาจารย์ ดร.จักริน ชวชาติ อาจารย์ที่ปรึกษาการค้นคว้าอิสระและอาจารย์ ดร.ประภาพร เตชอังกูร อาจารย์กรรมการคุมสอบ ที่กรุณาให้ความรู้ คำแนะนำ คำปรึกษา รวมทั้งสละเวลาตรวจแก้ไขงาน การค้นคว้านี้เสร็จสมบูรณ์ อีกทั้งต้องขอขอบคุณนายนิติภูมิ อุณารมณ ที่ให้คำปรึกษาและริวิวโค้ด บางส่วน ช่วยให้คำแนะนำในการแก้ไขปรับปรุงงานในส่วนของเกมทำงานได้อย่างถูกต้องและมีประสิทธิภาพ

ขอกราบขอบพระคุณอาจารย์ประจำภาควิชาวิทยาการคอมพิวเตอร์ทุกท่านที่มอบความรู้ตลอดการศึกษาในมหาวิทยาลัยเชียงใหม่ ซึ่งส่งผลทำให้การค้นคว้าครั้งนี้สำเร็จลุล่วง ขอขอบคุณเจ้าหน้าที่ประจำภาควิชาวิทยาการคอมพิวเตอร์ทุกท่าน ที่ให้ความสะดวกและความช่วยเหลือในเรื่องต่าง ๆ เป็นอย่างดี

รัฐภูมิ ม้าเพ็ง

570510676

|                           |  |
|---------------------------|--|
| ชื่อหัวข้อการค้นคว้าอิสระ | เกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม         |
| เจ้าของผลงาน              | นาย รัฐภูมิ ม้าเพ็ง รหัสนักศึกษา 570510676   |
| หลักสูตร                  | วิทยาศาสตร์บัณฑิตสาขาวิชาวิทยาการคอมพิวเตอร์ |
| อาจารย์ที่ปรึกษา          | อาจารย์ ดร.จักริน ชวชาติ                     |

### บทคัดย่อ

ปัจจุบันปัญญาประดิษฐ์นั้นมีความนิยมอย่างแพร่หลายในหลายด้าน เช่น ธุรกิจ สุขภาพ และการขนส่ง หลักการพื้นฐานของปัญญาประดิษฐ์อย่างหนึ่งคือต้นไม้การตัดสินใจซึ่งมีการใช้อย่างแพร่หลายและมีการนำไปประยุกต์ในหลายๆ ด้าน ทางผู้พัฒนาสนใจรูปแบบหนึ่งของการประยุกต์ต้นไม้ตัดสินใจมาใช้นั้นคือต้นไม้พฤติกรรม ผู้พัฒนาได้พัฒนาเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม เกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรมเป็นเกมแนวผสมระหว่างเกมกลยุทธ์และเกมเทิร์นเบสเกมถูกออกแบบให้ผู้เล่นควบคุมหุ่นยนต์ของตนด้วยต้นไม้พฤติกรรมแทนการควบคุมแบบปกติ เช่น เม้าส์ คีย์บอร์ดและจอยสติ๊ก โดยต้นไม้พฤติกรรมถูกใช้เพื่อกำหนดลำดับและความสำคัญของการตรวจสอบเงื่อนไขในการแสดงออกของหุ่นยนต์ ซึ่งผู้เล่นสามารถสร้างและปรับปรุงได้อย่างอิสระ เพื่อที่จะบรรลุเป้าหมายของเกมด้วยการเก็บคริสตัลหรือการทำลายคู่ต่อสู้ ผู้เล่นสามารถชนะเกมได้หลายรูปแบบโดยจะแตกต่างกันไปตามต้นไม้พฤติกรรมที่ผู้เล่นสร้างขึ้น การสร้างต้นไม้พฤติกรรมเพื่อควบคุมหุ่นยนต์ถือเป็นการช่วยให้ผู้เล่นเข้าใจพื้นฐานของปัญญาประดิษฐ์นั้นทำงานอย่างไร

|                |   |
|----------------|---|
| <b>Title</b>   | Strategy game using behavior tree       |
| <b>Author</b>  | Rattapoom Mapeng Student ID 570510676   |
| <b>Degree</b>  | Bachelor of Science in Computer Science |
| <b>Advisor</b> | Dr.Jakarin Chawachat                    |

### **Abstract**

Artificial intelligence(AI) is widely used in many fields such as business, health and transportation. The basic principle of artificial intelligence is the decision tree, which is widely used and applied in many ways. We are interested in one form of decision tree, i.e., the behavior tree. Therefore, we developed a strategy game using behavior tree. Strategy game using behavior tree is a hybrid game between strategy game and turn based game. It is designed to allow players to control their robots with behavior tree instead of normal controls such as mouse, keyboard and joystick. Behavior trees are used to determine the order and importance of checking conditions in the expression of the robot. Players can create and improve independently in order to achieve the goal of the game by collecting crystals or destroying the opponents. Players can win different types of games, depending on the player's behavior trees. Creating a behavior tree to control a robot is one way to help players understand the basics of artificial intelligent.

## สารบัญ

|   | หน้า |
|---|------|
| กิตติกรรมประกาศ   | ก    |
| บทคัดย่อ  | ข    |
| ABSTRACT  | ค    |
| สารบัญ  | ง    |
| สารบัญรูป   | ช    |
| สารบัญตาราง   | ญ    |
| บทที่ 1 บทนำ  | 1    |
| 1.1 หลักการและเหตุผล  | 1    |
| 1.2 วัตถุประสงค์ของโครงการ                                    | 2    |
| 1.3 ประโยชน์ที่จะได้รับจากการศึกษาเชิงประยุกต์                | 2    |
| 1.4 ขอบเขตของโครงการ  | 3    |
| บทที่ 2 ทฤษฎีหรือหลักการที่เกี่ยวข้อง                         | 5    |
| 2.1 ต้นไม้การตัดสินใจ   | 5    |
| 2.2 การค้นตามแนวลึก   | 7    |
| 2.3 ต้นไม้พหุติกรรม   | 8    |
| 2.4 กฎและกติกาของเกมกลยุทธ์วางแผนกลยุทธ์โดยใช้ต้นไม้พหุติกรรม | 12   |
| 2.5 โปรแกรม UNITY 3D  | 13   |
| 2.6 เกมและงานวิจัยที่เกี่ยวข้อง                               | 17   |

## สารบัญ (ต่อ)

|                                       | หน้า |
|---------------------------------------|------|
| บทที่ 3 การวิเคราะห์และออกแบบระบบ     | 21   |
| 3.1 การวิเคราะห์โครงสร้างของระบบ      | 21   |
| 3.2 การวิเคราะห์พฤติกรรมของระบบ       | 28   |
| บทที่ 4 การออกแบบจอภาพ                | 40   |
| 4.1 จอเมนูเริ่มต้น                    | 40   |
| 4.2 จอแนะนำวิธีการเล่น                | 41   |
| 4.3 จอเลือกด่าน                       | 42   |
| 4.4 จอการเล่นเกม                      | 43   |
| 4.5 จอสรุปผลการเล่นเกม                | 44   |
| 4.6 จอภาพส่วนประสานต้นไม้พฤติกรรม     | 45   |
| 4.7 จอหน้าต่างการสร้างโหนดใหม่        | 46   |
| 4.8 จอเมนูเลือกโหนด                   | 46   |
| บทที่ 5 การพัฒนาระบบ                  | 47   |
| 5.1 ภาษาและเครื่องมือที่ใช้ในการพัฒนา | 47   |
| 5.2 เทคนิคที่ใช้ในการพัฒนา            | 48   |
| 5.3 การพัฒนาระบบ                      | 49   |
| บทที่ 6 ผลสรุป                        | 52   |
| 6.1 ผลการทดลองโปรแกรม                 | 52   |
| 6.2 ปัญหาที่พบ                        | 52   |
| 6.3 ข้อเสนอแนะ                        | 52   |
| เอกสารอ้างอิง                         | 53   |

## สารบัญ (ต่อ)

|  | หน้า |
|--|------|
| ภาคผนวก  | 55   |
| ภาคผนวก ก คู่มือการติดตั้งเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม | 55   |
| ภาคผนวก ข คู่มือการเล่นเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม    | 56   |

## สารบัญรูป

|   | หน้า |
|---|------|
| รูปที่ 2.1 ตัวอย่างต้นไม้การตัดสินใจ                          | 6    |
| รูปที่ 2.2 ตัวอย่างการทำงานของต้นไม้การตัดสินใจ               | 6    |
| รูปที่ 2.3 ตัวอย่างขั้นตอนการค้นหาตามแนวลึกในไบนารีทรี        | 7    |
| รูปที่ 2.4 ตัวอย่างต้นไม้พฤติกรรม                             | 10   |
| รูปที่ 2.5 ตัวอย่างการทำงานของต้นไม้พฤติกรรม                  | 10   |
| รูปที่ 2.6 ตัวอย่างการทำงานของต้นไม้พฤติกรรมแบบที่สอง         | 11   |
| รูปที่ 2.7 ตัวอย่างการทำงานของต้นไม้พฤติกรรมแบบที่สอง         | 11   |
| รูปที่ 2.8 ตัวอย่างการทำงานระหว่างต้นไม้พฤติกรรมทั้งสองแบบ    | 12   |
| รูปที่ 2.9 โปรแกรม Unity                                      | 13   |
| รูปที่ 2.10 ตัวอย่างการเรียงตัวใน Hierarchy ของ Scence        | 14   |
| รูปที่ 2.11 ตัวอย่างการผูก Script เข้ากับ Object              | 15   |
| รูปที่ 2.12 แผนภาพลำดับการทำงานของ MonoBehaviour              | 16   |
| รูปที่ 2.13 ตัวอย่างระหว่างเล่นเกม Red Alert 2                | 17   |
| รูปที่ 2.14 ตัวอย่างเกมหมากรุก                                | 18   |
| รูปที่ 2.15 ตัวอย่างเกม Lightbot                              | 19   |
| รูปที่ 2.16 ตัวอย่างเกม Code Combat                           | 20   |
| รูปที่ 3.1 ยูสเคสไคอะแกรมของเกมวางแผนกลยุทธ์โดยต้นไม้พฤติกรรม | 23   |
| รูปที่ 3.2 แผนภาพกิจกรรมของระบบ                               | 29   |
| รูปที่ 3.3 แผนภาพกิจกรรมของการเล่นเกม                         | 30   |
| รูปที่ 3.4 Sequence Diagram ของ Select LV                     | 32   |
| รูปที่ 3.5 Sequence Diagram ของ How To                        | 33   |
| รูปที่ 3.6 Sequence Diagram ของ Play Game                     | 34   |
| รูปที่ 3.7 Sequence Diagram ของ Create Tree                   | 35   |
| รูปที่ 3.8 Sequence Diagram ของ Fast Forward                  | 36   |

## สารบัญรูป (ต่อ)

|   | หน้า |
|---|------|
| รูปที่ 3.9 คลาสไดอะแกรมของหุ่นยนต์                    | 37   |
| รูปที่ 3.10 คลาสไดอะแกรมของกระสุน                     | 38   |
| รูปที่ 3.11 คลาสไดอะแกรมของคริสตัล                    | 38   |
| รูปที่ 3.12 คลาสไดอะแกรมของโหนดต่างๆ                  | 38   |
| รูปที่ 3.13 คลาสไดอะแกรมความสัมพันธ์ในระบบ            | 39   |
| รูปที่ 4.1 หน้าจอหลักของการเล่นเกม                    | 40   |
| รูปที่ 4.2 หน้าจอแนะนำวิธีการเล่นเกม                  | 41   |
| รูปที่ 4.3 หน้าจอเลือกด่าน                            | 42   |
| รูปที่ 4.4 หน้าจอหลักของการเล่นเกม                    | 43   |
| รูปที่ 4.5 หน้าจอสรุปผลการเล่นเกม                     | 44   |
| รูปที่ 5.1 การใช้ Prefab มาสร้าง GameObject           | 48   |
| รูปที่ 5.2 การเรียกใช้ Singleton Pattern              | 49   |
| รูปที่ 5.3 การแปลงโหนดที่ใช้ควบคุมและโหนดที่ใช้แสดงผล | 50   |
| รูปที่ 5.4 การวางตำแหน่งของส่วนแก้ไขต้นไม้พฤติกรรม    | 51   |
| รูปที่ ข.1 ตัวเล่นเกม                                 | 56   |
| รูปที่ ข.2 หน้าต่างปรับตั้งค่า                        | 56   |
| รูปที่ ข.3 หน้าจอเมนู                                 | 57   |
| รูปที่ ข.4 หน้าจอเลือกด่าน                            | 57   |
| รูปที่ ข.5 หน้าจอเริ่มเกม                             | 58   |
| รูปที่ ข.6 หุ่นยนต์ของทีมผู้เล่น                      | 58   |
| รูปที่ ข.7 หน้าจอแก้ไขต้นไม้พฤติกรรม                  | 58   |
| รูปที่ ข.8 หน้าต่างเลือกสร้างโหนด                     | 59   |
| รูปที่ ข.9 หน้าจอหลังจากสร้างโหนด                     | 59   |
| รูปที่ ข.10 หน้าจอเมื่อเชื่อมโหนดเข้ากับโหนดราก       | 60   |

## สารบัญรูป (ต่อ)

|   | หน้า |
|---|------|
| รูปที่ ข.11 หน้าจอเมื่อสร้างโหนดเพิ่ม                   | 60   |
| รูปที่ ข.12 หน้าจอเมื่อสร้างโหนดยิงเพิ่ม                | 61   |
| รูปที่ ข.13 หน้าจอเมื่อย้ายโหนดยิง                      | 61   |
| รูปที่ ข.14 หน้าจอเมื่อเชื่อมโหนด Fire กับ ClosestEnemy | 62   |
| รูปที่ ข.15 หน้าจอเมื่อสร้างต้นไม้ตัดสินใจเสร็จ         | 62   |
| รูปที่ ข.16 หน้าจอขณะเกมกำลังเล่น                       | 63   |
| รูปที่ ข.17 หน้าจอสรุปผลการเล่น                         | 63   |

## สารบัญตาราง

|   | หน้า |
|---|------|
| ตารางที่ 2.1 สัญลักษณ์และความหมายของต้นไม้การตัดสินใจ                 | 5    |
| ตารางที่ 2.2 สัญลักษณ์และความหมายของต้นไม้พฤติกรรมในการควบคุมหุ่นยนต์ | 9    |
| ตารางที่ 3.1 สัญลักษณ์ที่ใช้ในยูสเคสไดอะแกรม                          | 22   |
| ตารางที่ 3.2 Use Case Specification ของ Select LV                     | 24   |
| ตารางที่ 3.3 Use Case Specification ของ How To                        | 25   |
| ตารางที่ 3.4 Use Case Specification ของ Play Game                     | 26   |
| ตารางที่ 3.5 Use Case Specification ของ Create Tree                   | 27   |
| ตารางที่ 3.6 Use Case Specification ของ Fast Forward                  | 28   |
| ตารางที่ 3.7 สัญลักษณ์แผนภาพกิจกรรมของระบบ                            | 29   |
| ตารางที่ 3.8 สัญลักษณ์ที่ใช้ในซีควเอนซ์ไดอะแกรม                       | 31   |
| ตารางที่ 3.9 สัญลักษณ์ที่ใช้ในแผนภาพคลาสไดอะแกรม                      | 36   |

# บทที่ 1

## บทนำ

### 1.1 หลักการและเหตุผล

ปัจจุบันการเรียนรู้มีช่องทางหลากหลาย อีกทั้งเด็กและเยาวชนยุคนี้ใกล้ชิดกับเทคโนโลยีเป็นอย่างมาก จึงมีการนำเกมมาเป็นช่องทางหนึ่งสำหรับการเรียนรู้ในเรื่องต่างๆ ทั้งนี้ เกมปริศนา เกมแนวแก้ปัญา เกมแนววางแผนการรบนั้นสามารถนำมาใช้ฝึกลำดับความคิดในการทำงานให้กับผู้เล่นได้

เกมวางแผนกลยุทธ์เป็นเกมประเภทหนึ่งที่โดยทั่วไปจะให้ผู้เล่นควบคุมตัวละครในทีมของตนไปปฏิบัติการกิจตามเป้าหมายของฉากนั้น ๆ โดยส่วนใหญ่เกมแนวกลยุทธ์จะเป็นเกมที่ต้องวางแผนโดยควบคุมกำลังพลให้ทำตามภารกิจที่ได้รับ ผู้ศึกษาจึงเลือกหุ่นยนต์เพื่อเป็นตัวละครหลักที่ผู้เล่นใช้ในการควบคุม โดยที่ผู้เล่นจะต้องออกแบบคำสั่งเพื่อควบคุมหุ่นยนต์โดยใช้ต้นไม้พฤติกรรม ให้เก็บคริสตัลในฉากหรือทำลายหุ่นยนต์ฝ่ายตรงข้าม เมื่อผู้เล่นออกแบบต้นไม้พฤติกรรมของหุ่นยนต์แต่ละตัวเสร็จ ผู้เล่นจะต้องสั่งเริ่มเกม เพื่อให้หุ่นยนต์แต่ละตัวจะปฏิบัติตามคำสั่งที่ได้ออกแบบไว้ในต้นไม้พฤติกรรม เกมจะสิ้นสุดเมื่อคริสตัลทั้งหมดถูกเก็บไป ผู้ชนะคือผู้ที่เก็บคริสตัลได้มากกว่า

ในการออกแบบการควบคุมหุ่นยนต์แต่ละตัวนั้น ทางผู้ศึกษาคิดที่จะนำเอาความรู้บางส่วนทางวิทยาการคอมพิวเตอร์ นั่นคือต้นไม้การตัดสินใจ มาประยุกต์เข้าเป็นส่วนหนึ่งของพื้นฐานในการเล่น เกม โดยผู้เล่นจะต้องออกแบบต้นไม้พฤติกรรมซึ่งมีรูปแบบการทำงานคล้ายกับต้นไม้การตัดสินใจ ผู้เล่นจะต้องออกแบบเงื่อนไขการทำงานให้กับหุ่นยนต์โดยใช้ต้นไม้พฤติกรรม เมื่อเริ่มเกมหุ่นยนต์จะตรวจสอบและปฏิบัติตามเงื่อนไข แต่ถ้าเกิดมีเหตุการณ์ที่ต้องพร้อม ๆ กันหุ่นยนต์จะเลือกทำตามลำดับความสำคัญของเงื่อนไขที่ผู้เล่นสร้างขึ้นก่อน ยกตัวอย่างเช่น หุ่นยนต์ตัวที่หนึ่งมีเงื่อนไข ยิ่งหุ่นยนต์ที่เข้ามาในระยะจากนั้นเก็บคริสตัล และหุ่นยนต์ตัวที่สองมีเงื่อนไข เก็บคริสตัลจากนั้นยิ่งหุ่นยนต์ที่เข้ามาในระยะ หากเริ่มเกมในระยะทางที่เท่ากันในแผนที่ มีความสมมาตรกันหุ่นยนต์ตัวที่หนึ่งและสอง จะเดินไปเส้นทางที่มีคริสตัลอยู่ในขณะที่หุ่นยนต์ตัวที่สอง

จะเดินเข้าใกล้คริสตัลเรื่อย ๆ แต่หุ่นยนต์ตัวที่หนึ่งจะหยุดการเดินไปหาคริสตัลและยิงหุ่นยนต์ตัวที่สองเมื่อหุ่นยนต์ตัวที่สองเริ่มเข้ามาในระยะการยิง เนื่องจากหุ่นยนต์ตัวที่สองจะเดินไปหาคริสตัลต่อไปเนื่องจากความสำคัญของเดินไปเก็บคริสตัลมาก่อนการยิงตอบโต้ เป็นต้น

เกมที่ผู้ศึกษาพัฒนาขึ้นมีจุดประสงค์เพื่อให้ผู้เล่นได้ฝึกทักษะด้านการวางแผนและแก้ไขปัญหาในรูปแบบต่าง ๆ ซึ่งขึ้นกับเงื่อนไขในฉากนั้น ๆ การทำเช่นนี้ถือว่าการฝึกฝนการจัดการกับปัญหาและจัดระบบความคิดให้เป็นระบบระเบียบได้เป็นอย่างดี อีกทั้งยังเป็นการช่วยสร้างความเข้าใจพื้นฐานการทำงานของต้นไม้พฤติกรรมให้กับผู้เล่นอีกด้วย

## 1.2 วัตถุประสงค์ของโครงการ

- 1) พัฒนาเกมที่ควบคุมหุ่นยนต์โดยต้นไม้พฤติกรรม
- 2) พัฒนาเกมที่ส่งเสริมให้ผู้เล่นได้ฝึกทักษะด้านการจัดการแก้ไขปัญหาอย่างเป็นระบบและมีประสิทธิภาพ
- 3) พัฒนาเกมที่สอดแทรกเนื้อหาเกี่ยวกับความรู้ด้านวิทยาการคอมพิวเตอร์เพื่อให้ง่ายต่อการเข้าถึงและสามารถศึกษาได้ด้วยตนเอง

## 1.3 ประโยชน์ที่จะได้รับจากการศึกษาเชิงประยุกต์

- 1) ได้เกมที่ควบคุมหุ่นยนต์โดยใช้ต้นไม้พฤติกรรม
- 2) ได้เกมที่ช่วยฝึกฝนทักษะด้านการจัดการปัญหาและช่วยให้ผู้เล่นสามารถจัดการกับความคิดได้อย่างเป็นระบบและมีประสิทธิภาพ
- 3) ได้เกมที่สอดแทรกความรู้ด้านวิทยาการคอมพิวเตอร์เพื่อให้คนทั่วไปสามารถเข้าถึงและเรียนรู้ด้วยตนเองได้

## 1.4 ขอบเขตของโครงการ

### 1.4.1 ขอบเขตทางสถาปัตยกรรม

- 1) ระบบถูกพัฒนาในรูปแบบของเกมผู้เล่นคนเดียว ทำงานบนระบบปฏิบัติการวินโดวส์ (Windows)
- 2) ฮาร์ดแวร์ (Hardware) ที่ใช้ในการพัฒนาระบบ
  - 2.1) ซีพียู (CPU) อินเทลคอร์ไอเจ็ต 1.80 จิกะเฮิร์ตซ์ (Intel® Core™ i7-4500U CPU @ 1.80GHz)
  - 2.2) หน่วยความจำหลัก (RAM) 12 จิกะไบต์ (Gigabyte)
  - 2.3) หน่วยความจำสำรอง (Hard Disk) ขนาดความจุ 1 เทราไบต์ (Terabyte)
- 3) ซอฟต์แวร์ (Software) ที่ใช้ในการพัฒนาระบบ
  - 3.1) ระบบปฏิบัติการ (Operating System) ที่สามารถใช้พัฒนาระบบคือ วินโดวส์ 10 (Window 10)
  - 3.2) ยูนิตี้เกมเอนจิน 5.6 (Unity 5.6) ใช้สำหรับพัฒนาระบบ
  - 3.3) เครื่องมือที่ใช้ในการพัฒนาโปรแกรม (Integrated Development Environment: IDE) คือ วิซวลสตูดิโอ 2017 คอมมูนิตี้ (Visual Studio 2017 Community)
  - 3.4) ภาษาที่ใช้ในการพัฒนาระบบ คือ ซีชาร์ป (C#)

### 1.4.2 ขอบเขตของระบบงาน

- 1) สามารถควบคุมหุ่นยนต์ด้วยคันไม้พุดติกรรมได้
- 2) สามารถรับข้อมูลคันไม้พุดติกรรมจากผู้เล่นผ่านส่วนต่อประสานได้

### 1.4.3 ขอบเขตของข้อมูล

#### 1) ข้อมูลนำเข้า

- 1.1) เลือกด่านที่ต้องการเล่น
- 1.2) เลือกหุ่นยนต์ที่ต้องการสร้างต้นไม้พฤติกรรม
- 1.3) เลือกสร้างโหนดพฤติกรรม เพื่อสร้างเป็นต้นไม้พฤติกรรม
- 1.4) เลือกย้ายตำแหน่งของโหนดต่าง
- 1.5) เลือกสร้างเส้นเชื่อมความสัมพันธ์ระหว่างโหนด
- 1.6) เลือกเกมเริ่มทำงาน

#### 2) ข้อมูลส่งออก

- 2.1) หุ่นยนต์แสดงพฤติกรรมของหุ่นยนต์ตามต้นไม้พฤติกรรมที่ผู้เล่นได้สร้างไว้  
ผ่านหน้าจอแสดงผล
- 2.2) ผลสรุปเกมเมื่อเกมจบ

## บทที่ 2




### ทฤษฎีหรือหลักการที่เกี่ยวข้อง

การพัฒนาเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม ซึ่งมีส่วนสำคัญคือการควบคุมด้วยต้นไม้พฤติกรรมจึงจำเป็นต้องศึกษาข้อมูลเพิ่มเติม ได้แก่ ต้นไม้การตัดสินใจ การค้นตามแนวลึก ซึ่งเป็นพื้นฐานของต้นไม้พฤติกรรม อีกทั้งยังต้องศึกษาเกี่ยวกับหลักการพื้นฐานของ Unity 3D เพื่อนำความรู้เหล่านี้ไปใช้ในการพัฒนาเกมดังกล่าว ซึ่งมีทฤษฎีและหลักการที่เกี่ยวข้อง ดังนี้

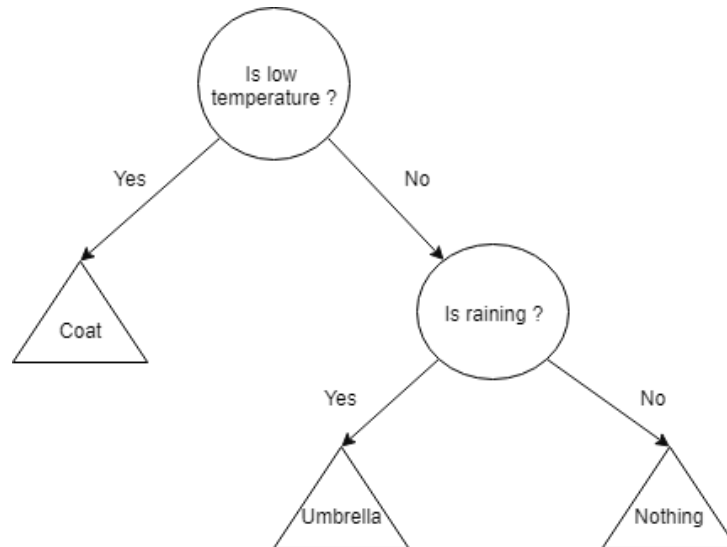
#### 2.1 ต้นไม้การตัดสินใจ

ต้นไม้การตัดสินใจ (Decision Tree) [1] คือแผนภาพที่ใช้ในการแยกประเภทตามเงื่อนไขตามข้อมูลที่เข้ามา โดยจะเลือกเส้นทางไปตามเส้นที่เชื่อมที่ตรงกับคุณลักษณะของข้อมูลนำเข้า ไปยังโหนดตัดสินใจต่อไปจนสิ้นสุดที่ใบของต้นไม้การตัดสินใจ มักจะใช้ในการแยกประเภทของข้อมูลส่วนประกอบของต้นไม้การตัดสินใจมีดังตารางที่ 2.1

ตารางที่ 2.1 สัญลักษณ์และความหมายของต้นไม้การตัดสินใจ

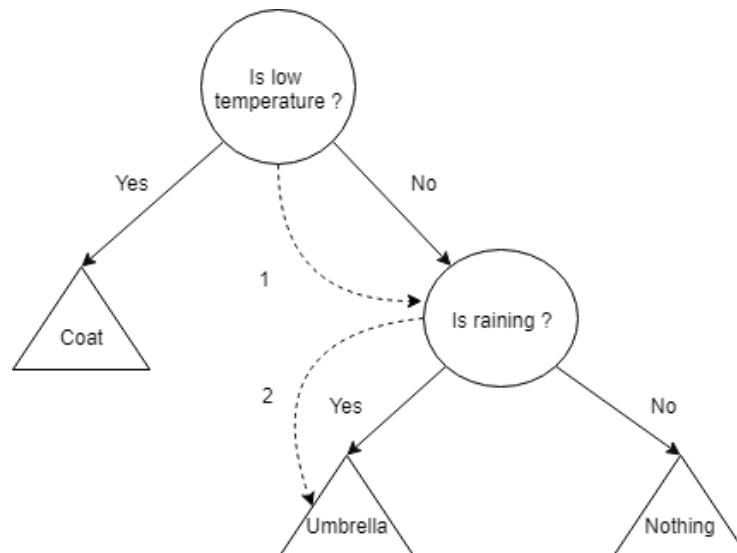
| ชื่อ           | สัญลักษณ์   | ความหมาย   |
|----------------|---|--|
| Condition Node |  | โหนดเงื่อนไขจะใช้เป็นตัวกำหนดการพิจารณาเงื่อนไขในโหนดชั้นถัดไป                         |
| Connection     |  | เส้นเชื่อมใช้ในการเชื่อมโหนดสองโหนดใด ๆ เข้าด้วยกัน และจะบอกค่าการตัดสินใจบนเส้นเชื่อม |
| Leaf Node      |  | โหนดสุดท้ายถูกใช้ในตัดสินใจข้อมูลที่เข้ามาว่าเป็นสิ่งใด                                |

จากตารางที่ 2.1 เพื่อการอธิบายเพิ่มเติมผู้ศึกษาจึงยกตัวอย่างต้นไม้การตัดสินใจดังรูปที่ 2.1



รูปที่ 2.1 ตัวอย่างต้นไม้การตัดสินใจ

จากรูปที่ 2.1 เป็นตัวอย่างของต้นไม้การตัดสินใจที่ช่วยให้ช่วยเลือกสิ่งของที่จะนำติดตัวออกมาจากบ้านตามสภาพอากาศ จะพบว่าหากข้อมูลนำเข้าคือวันที่อุณหภูมิสูงและฝนตก การทำงานของต้นไม้การตัดสินใจจะให้ผลลัพธ์สุดท้ายคือ “ร่ม” โดยจะทำงานดังรูปที่ 2.2 ด้านล่าง



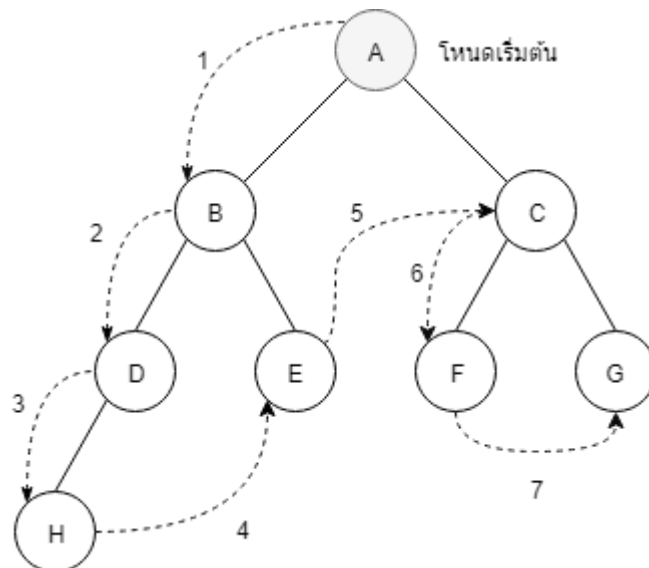
รูปที่ 2.2 ตัวอย่างการทำงานของต้นไม้การตัดสินใจ

จากรูปที่ 2.2 กรณีที่ผู้ใช้นำเข้าข้อมูล “วันที่อุณหภูมิสูงและฝนตก” ต้นไม้การตัดสินใจจะเริ่มทำงานจากโหนดแรกคือโหนดในการตรวจสอบอุณหภูมิ โดยโหนดนี้จะตรวจสอบข้อมูลนำเข้า

ว่า อุณหภูมิต่ำหรือไม่จากกรณีนี้ข้อมูลนำเข้าคือ “วันที่อุณหภูมิสูงและฝนตก” อุณหภูมิสูงจึงทำการตรวจสอบเงื่อนไขเป็นเท็จ เมื่อเงื่อนไขเป็นเท็จต้นไม้การตัดสินใจจะเปลี่ยนไปตรวจสอบโหนดลูกที่มีเงื่อนไขเป็นเท็จ คือโหนดที่ตรวจสอบข้อมูลนำเข้าว่าขณะนี้ฝนตกหรือไม่ ซึ่งข้อมูลนำเข้าคือ “วันที่อุณหภูมิสูงและฝนตก” จะพบว่า ฝนตก จึงทำให้เงื่อนไขนี้เป็นจริง จึงทำการตรวจสอบไปยังโหนดลูกที่มีเงื่อนไขเป็นจริงพบว่าโหนดลูกนั้นเป็นโหนดใบ ต้นไม้ตัดสินใจจึงหยุดตรวจสอบเงื่อนไขและแจ้งผลลัพธ์ว่าต้องนำ “ร่ม” ติดตัวออกจากบ้าน

## 2.2 การค้นหาตามแนวลึก

การค้นหาเชิงลึก (Depth first search) เป็นขั้นตอนวิธีแบบหนึ่งในการท่อง (Traversing) หรือค้นหา (Searching) โหนดในกราฟหรือต้นไม้ หลักการทำงานคือ เริ่มจากโหนดโหนดหนึ่งก่อน จากนั้นทำการค้นหาไปยังโหนดที่เชื่อมต่อกันต่อไปให้ลึกที่สุดหรืออยู่ต่ำที่สุด หากค้นหาต่อไม่ได้ให้ย้อนกลับมายังโหนดก่อนหน้าแล้วเลือกเส้นทางอื่น ทำเช่นเดิมจนค้นหาคอบทุกโหนด



รูปที่ 2.3 ตัวอย่างขั้นตอนการค้นหาตามแนวลึกในไบนารีทรี




จากรูปที่ 2.3 กำหนดให้โหนด A ซึ่งเป็นโหนดรากเป็นโหนดเริ่มต้นจึงนำโหนด A ใส่เข้าไปในสแตคและค้นหาต่อไปยังโหนดลูกของโหนด A ได้แก่ โหนด B ตามลำดับของไบนารีทรี จากนั้นตรวจสอบว่าโหนด B เป็นโหนดใบหรือไม่หากไม่แล้วนำ B ไปใส่ไว้ในสแตคแล้วตรวจสอบไปยังโหนดลูกของโหนด B ได้แก่โหนด D แล้วนำ D ใส่ในสแตคตามลำดับทำซ้ำไปเรื่อย ๆ จนถึงโหนด H ที่เป็นโหนดใบให้ดูที่สแตคว่าตัวที่ต้องค้นต่อไปคือโหนด D เมื่อโหนด D

ไม่มีโหนดลูกแล้วให้นำโหนด D ออกจากสแตกแล้ว ค้นหาตัวที่อยู่ในสแตกต่อไปคือ B ซึ่งโหนด B มีลูกคือโหนด D และโหนด E แต่โหนด D ถูกค้นไปแล้ว ดังนั้นจึงไปยังโหนด E โหนด E เป็นโหนดใบจากนั้นต้องกลับไปดูที่สแตกอีกครั้งคือโหนด B โหนด B ได้ถูกค้นลูกทั้งหมดแล้ว จึงนำโหนด B ออกจากสแตก จากนั้นดูที่สแตกตัวต่อไปคือโหนด A ซึ่งโหนดที่ต้องค้นต่อไปคือ C มีโหนดลูกอยู่ 2 ที่ยังไม่ได้ค้นหาถึงนำโหนด C ไปใส่ไว้ในสแตก จากนั้นเริ่มค้นที่โหนด F ตามหลักของไบนารีทรีโหนด F เป็นโหนดใบแล้วถึงกลับไปดูที่สแตกพบคือโหนด C จากนั้นค้นโหนดลูกที่เหลือคือโหนด G และกลับไปดูอีกสแตกอีกครั้งคือโหนด C แต่ถูกค้นโหนดลูกจนหมดแล้วถึงนำออกจากสแตกแล้วดูในสแตกอีกครั้งคือโหนด A จากโหนด A ซึ่งถูกค้นโหนดลูกจะหมดแล้วจึงนำออกจากสแตกเมื่อสแตกว่างแล้วจึงจบการทำงาน จะได้ว่าการค้นหาตามแนวลึกจะได้ลำดับการค้นหาดังนี้ A B D H E C F และ G

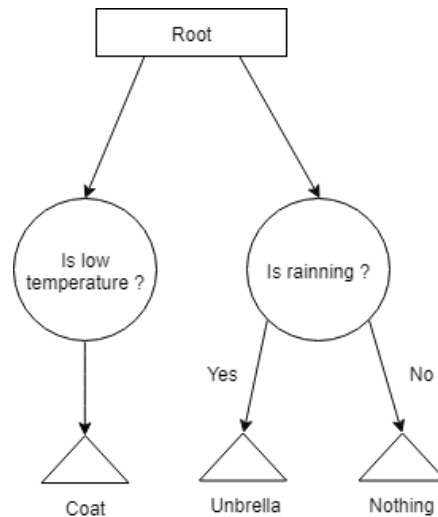
### 2.3 ต้นไม้พหุติกรรม

ในเกมวางแผนกลยุทธ์จะใช้ต้นไม้พหุติกรรมในการควบคุมหุ่นยนต์ซึ่งเป็นต้นไม้ที่มีรูปแบบผสมระหว่างต้นไม้การตัดสินใจและการค้นหาตามแนวลึก โดยต้นไม้พหุติกรรมนั้นจะประกอบด้วย 1) โหนดรากนั้นจะมีความพิเศษคือมีลูกกี่โหนดก็ได้ 2) โหนดในการตัดสินใจโหนดนี้ใช้สำหรับใส่เงื่อนไขต่าง ๆ โดยเงื่อนไขจะเป็นเงื่อนไขแบบสองทางเลือกคือ จริงและเท็จ จึงทำให้มีโหนดมีลูกได้สูงสุดเพียง 2 โหนดเท่านั้น ถ้าหากโหนดลูกมีเพียงโหนดเดียวระบบจะถึงว่าเป็นโหนดของเงื่อนไขจริง โดยจะบอกว่าโหนดที่เชื่อมเป็นเงื่อนไขแบบใดผ่านตัวหนังสือบนเส้นเชื่อมระหว่างโหนด 3) โหนดพหุติกรรมเป็นโหนดที่ใช้ในการเลือกการกระทำของหุ่นยนต์ที่ควบคุม การค้นหาผลลัพธ์ จะทำการค้นหาแนวลึกตามข้อ 2.2 โดยจะเริ่มค้นหาในต้นย่อยในโหนดลูกของโหนดรากเรียงตามลำดับจากซ้ายไปขวา เมื่อค้นหาเสร็จแล้วพบว่าโหนดสุดท้ายไม่ใช่โหนดพหุติกรรม จะทำการค้นหาในต้นย่อยของโหนดลูกตัวถัดไปจนครบ หากไม่สามารถหาโหนดพหุติกรรมได้ จะคืนค่าเป็นค่าว่างเปล่า

ตารางที่ 2.2 สัญลักษณ์และความหมายของต้นไม้พฤติกรรมในการควบคุมหุ่นยนต์

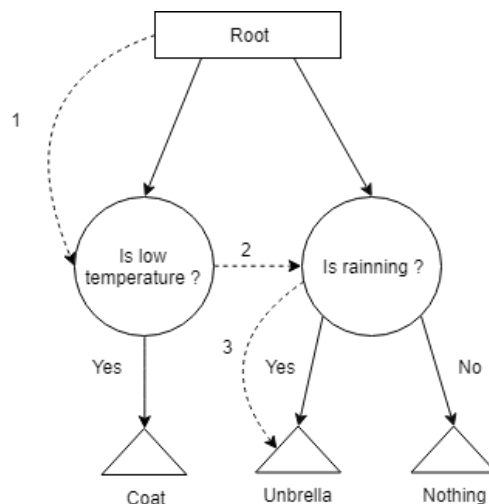
| ชื่อ          | สัญลักษณ์   | ความหมาย  |
|---------------|---|---|
| Root Node     |    | โหนดที่ใช้สำหรับบอกให้เริ่มต้นการค้นหาจากโหนดนี้  |
| Decision Node |    | โหนดตัดสินใจ คือโหนดที่ใช้กำหนดเงื่อนไขโดยจะมีลูกของโหนดได้สูงสุดเพียง 2 โหนดเท่านั้น   |
| Connection    |    | เส้นเชื่อมใช้ในการเชื่อมโหนดเข้าด้วยกัน หากเชื่อมโหนดเป็นโหนดภายใต้การตัดสินใจจะบอกว่าเส้นเชื่อนั้นเป็นการตัดสินใจที่เป็นเงื่อนไขที่ ถูกหรือผิด |
| Action Node   |  | โหนดพฤติกรรมจะใช้ในตัดสินใจข้อมูลที่เข้ามาว่าเป็นสิ่งใด   |

จากตารางที่ 2.2 จะสามารถสร้างต้นไม้พฤติกรรมที่ส่งผลลัพธ์เหมือนกับต้นไม้การตัดสินใจ จากรูปที่ 2.1 ได้ดังรูปที่ 2.4



รูปที่ 2.4 ตัวอย่างต้นไม้พหุติกรรม

จากรูปที่ 2.4 แสดงตัวอย่างของการแปลงต้นไม้การตัดสินใจในจากรูปที่ 2.1 เป็นต้นไม้พหุติกรรม จึงตรวจสอบด้วยการนำข้อมูลนำเข้าจากตัวอย่างเดิมในรูปที่ 2.1 คือ “วันที่อุณหภูมิสูงและฝนตก” จะมีการตรวจสอบลำดับเงื่อนไขดังรูปที่ 2.5

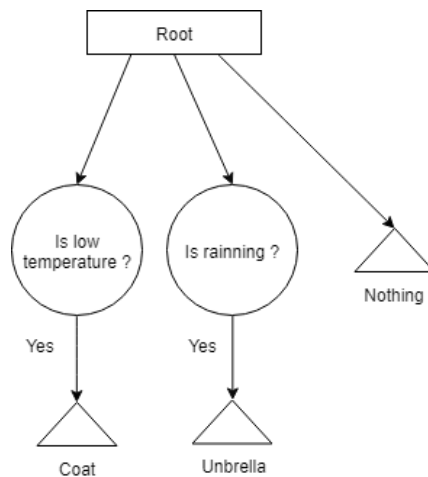


รูปที่ 2.5 ตัวอย่างการทำงานต้นไม้พหุติกรรม

จากรูปที่ 2.5 การทำงานของต้นไม้พหุติกรรมจะเริ่มต้นที่โหนดราก โดยเริ่มค้นหาจากต้นไม้ย่อยของโหนดลูกจากทางซ้ายไปขวาและการค้นหาในต้นไม้ย่อยนั้นจะเป็นการค้นหาตามแนวลึกโดยจะค้นหาตามเงื่อนไข ในกรณีนี้ข้อมูลนำเข้าคือ “วันที่อุณหภูมิสูงและฝนตก” ต้นไม้พหุติกรรมจะเริ่มค้นหาจากโหนดราก โดยโหนดลูกทางซ้ายสุดคือโหนดตัดสินใจ ที่ตรวจสอบว่าข้อมูลนำเข้านั้นคือวันที่อุณหภูมิต่ำหรือไม่ ในกรณีนี้ข้อมูลนำเข้าคือ “วันที่อุณหภูมิสูงและฝนตก” จึงทำให้

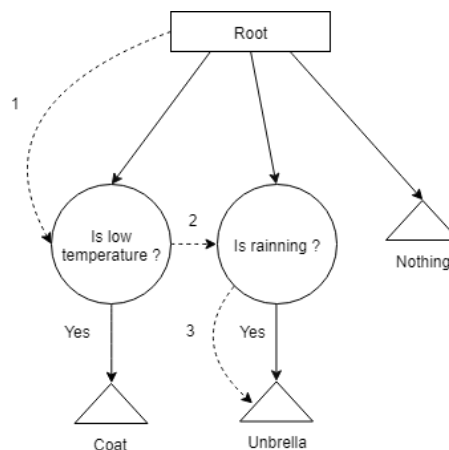
เงื่อนไขเป็นเท็จ ต้นไม้พฤติกรรมจะตรวจสอบลูกของโหนดที่เป็นเท็จ แต่โหนดนี้ไม่มีลูกของโหนดที่เป็นเงื่อนไขเท็จ จึงย้ายการค้นไปโหนดลูกของโหนดรากถัดไป คือโหนดตัดสินใจ โดยมีเงื่อนไขคือ วันที่ฝนตก จากข้อมูลนำเข้า “วันที่อุณหภูมิสูงและฝนตก” จะพบว่าเงื่อนไขถูกต้อง จึงค้นหาต่อไปโหนดลูกของที่เป็นเงื่อนไขจริง พบว่าโหนดลูกนั้นมีชนิดเป็นโหนดพฤติกรรม ทำให้ต้นไม้พฤติกรรมตัดสินใจเลือก “ร่ม” เป็นผลลัพธ์

เนื่องจากต้นไม้พฤติกรรมมีอิสระในการออกแบบดังนั้นจึงมีกรณีที่ต้นไม้พฤติกรรมที่แตกต่างกัน แต่มีผลลัพธ์การทำงานเหมือนกันได้ ดังรูปที่ 2.6



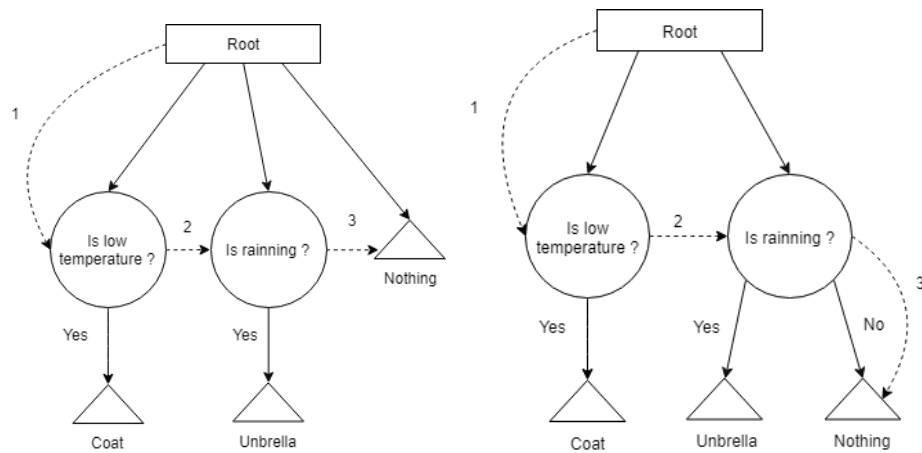
รูปที่ 2.6 ตัวอย่างการทำงานต้นไม้พฤติกรรมแบบที่สอง

จากรูปที่ 2.6 ต้นไม้พฤติกรรมอีกรูปแบบหนึ่งที่มีการทำงานเหมือนกับต้นไม้ในรูปที่ 2.4 โดยมีการทำงานดังรูปที่ 2.7



รูปที่ 2.7 ตัวอย่างการทำงานของต้นไม้พฤติกรรมแบบที่สอง

จากต้นไม้พฤติกรรมรูปที่ 2.4 และ 2.6 หากเปลี่ยนข้อมูลนำเข้าเป็น วันที่อุณหภูมิสูงและฝนไม่ตก จะได้การทำงานในรูปที่ 2.8



รูปที่ 2.8 ตัวอย่างการทำงานระหว่างต้นไม้พฤติกรรมทั้งสองแบบ

จากรูปที่ 2.8 จะเห็นว่าเมื่อเปลี่ยนข้อมูลนำเข้าให้กับต้นไม้พฤติกรรมทั้งสองแบบแล้ว ถึงแม้ว่าลำดับของการค้นหาจะแตกต่างกัน แต่ผลลัพธ์ที่ได้ยังเหมือนเดิม จึงเป็นสาเหตุให้ผู้เล่นสามารถชนะเกมได้ในต้นไม้พฤติกรรมหลายรูปแบบ

## 2.4 กฎและกติกาของเกมกลยุทธ์วางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

เป้าหมายของเกม ผู้เล่นจะต้องควบคุมหุ่นยนต์ของทีมของตนให้เก็บคริสตัลให้มากกว่าทีมตรงข้ามซึ่งแต่ละด้านจะสร้างคริสตัลมาสูงสุด 5 ชิ้น ดังนั้นทีมใดที่เก็บคริสตัลได้ครบ 3 ชิ้นก่อนจะเป็นฝ่ายที่ชนะ หรือสามารถทำลายหุ่นยนต์ฝ่ายตรงข้ามได้ทั้งหมดก็ถือว่าเป็นฝ่ายชนะเช่นกัน แต่ถ้าทั้งสองฝ่ายไม่เหลือหุ่นยนต์ในสนาม ฝ่ายที่เก็บคริสตัลได้มากกว่าจะเป็นฝ่ายชนะ หากมีคริสตัลเท่ากันจะถือว่าเสมอ

### หลักการทั่วไปของเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) หุ่นยนต์จะไม่สามารถรับรู้ถึงพื้นที่ด้านนอกระยะการมองเห็น
- 2) หากหุ่นยนต์ที่เก็บคริสตัลถูกทำลายระหว่างการนำส่งไปยังพื้นที่เก็บจะทำให้คริสตัลตก ณ จุดที่หุ่นยนต์ถูกทำลาย
- 3) ผู้เล่นจะได้คะแนนก็ต่อเมื่อหุ่นยนต์ของฝ่ายตนนำคริสตัลไปในพื้นที่เก็บของตนเองเรียบร้อยแล้ว

- 4) เกลาะสามารถฟื้นฟูได้เป็นอัตรา 2 หน่วย/วินาที หลังจากโดนโจมตีครั้งสุดท้ายเป็นเวลา 3 วินาทีแต่พลังชีวิตไม่สามารถฟื้นฟูได้
- 5) ขณะที่หุ่นยนต์ถอยกลับหุ่นยนต์จะหมุนตัวและเดินได้เร็วกว่าปกติ
- 6) การยิงจะมีความแรงเต็มทีที่ระยะการมองเห็น ในกรณีที่ยิงโดนฝ่ายตรงข้าม นอกกระยะการมองเห็นความแรงจะลดลง แต่หากยิงในระยะการมองเห็นความแรงจะเพิ่มขึ้น
- 7) หากหุ่นยนต์อยู่ระหว่างการถอยหนี หากโดนโจมตีที่ค่าความเสียหายต่ำกว่าร้อยละ 80 กระสุนจะไม่สามารถเจาะเกราะของหุ่นยนต์ตัวนั้นได้
- 8) กรณีที่หุ่นยนต์ถูกโจมตีนอกกระยะการยิงที่ค่าความเสียหายลดลงต่ำกว่าร้อยละ 50 การโจมตีนั้นจะไม่มีผลกับอัตราการฟื้นฟูเกราะ

## 2.5 โปรแกรม Unity 3D

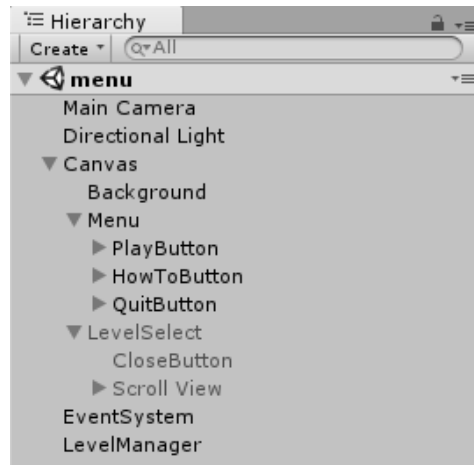


รูปที่ 2.9 โปรแกรม Unity

โปรแกรม Unity [2] เป็น Game Engine หรือโปรแกรมสร้างเกม ที่ใช้สำหรับพัฒนาเกม สามารถพัฒนาได้ทั้งเกม 3 มิติ และเกม 2 มิติ ซึ่งเกมที่พัฒนาด้วย Unity สามารถรองรับการทำงานหลาย Platform ซึ่งผู้พัฒนาเกมสามารถพัฒนาครั้งเดียว แล้วนำไปใช้กับหลาย Platform ได้ แต่อาจจะต้องเปลี่ยนแปลงเกมบางส่วนเพื่อรองรับ Platform นั้น ๆ สำหรับตัวโปรแกรม Unity จะรองรับการใช้งานบน Windows และ OSX โดยจะมี Version Free และ Version Pro นอกจาก Unity จะเป็นโปรแกรมสร้างเกมแล้ว ยังสามารถนำมาประยุกต์ใช้ในงานด้าน สถาปัตยกรรมวิศวกรรม และการก่อสร้าง และงานนำเสนอผลิตภัณฑ์สินค้าต่าง ๆ ได้อีกด้วย

### 2.5.1 พื้นฐานของ Unity 3D

- 1) การเรียงตัวของ Hierarchy ของ Scence

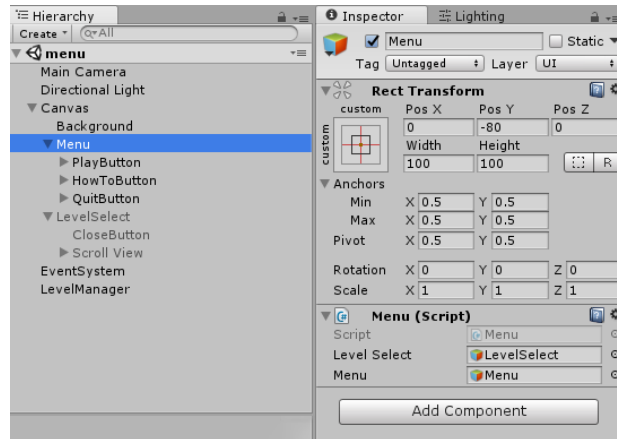


รูปที่ 2.10 ตัวอย่างการเรียงตัวใน Hierarchy ของ Scene

จากรูปที่ 2.10 จะพบว่าใน Unity นั้นมีการเรียงตัวกันคล้าย Layout แต่จะมีการเรียงตัวเป็นจากด้านบน ลงด้านล่างและจะมี UI ที่แสดงผลได้อยู่ 2 ประเภทคือ Canvas และ GameObject ซึ่ง Canvas ส่วนใหญ่จะเป็น UI จะอยู่ในแกนภาพที่มี 2 แกนคือ X และ Y ของหน้าจอ ส่วน GameObject จะอยู่ใน Word Space ของฉากโดยจะมีตำแหน่ง X Y และ Z

สำหรับ Canvas นั้นสามารถเลือกการแสดงผลได้ว่าจะให้อยู่ในรูปแบบของ UI หรือไปแสดงผลใน WordSpace ซึ่งจะสามารถใช้งานร่วมกับ GameObject ได้ตามความเหมาะสม แต่ข้อเสียคือการจัดวางตำแหน่งต้องใช้ตำแหน่ง X และ Y เช่นเดิมซึ่งทำให้ยากต่อการจัดการกับตำแหน่งของ Canvas ส่วนใหญ่หากใช้ใน WordSpace จึงผูกติดกับ GameObject มากกว่า

## 2) การผูก Script ให้กับ Object



รูปที่ 2.11 ตัวอย่างการผูก Script เข้ากับ Object

จากรูปที่ 2.11 GameObject นั้นสามารถเพิ่ม Component ได้หลาย Component เช่น ระบบฟิสิกส์ของ Unity ซึ่ง Script ก็ถือว่าเป็น Component ชนิดหนึ่งใน Unity เมื่อใส่ Script ให้กับ Object นั้น ๆ แล้วจะสามารถเขียน Script เพื่อเรียกใช้งานส่วนต่าง ๆ และติดต่อกับ Object ภายในฉากได้ ในรูปที่ 2.11 ในส่วนของ Script นั้นจะเห็นว่าการเข้าถึง Object 2 ตัวคือ Menu และ LevelSelect ไว้

### 2.5.2 ส่วนประกอบพื้นฐานของ Script ใน Unity

ในการเขียนระบบของ Unity นั้นจะมีการเรียกใช้ Object พื้นฐานที่เรียกว่า MonoBehaviour สำหรับ Object ใดที่มีการสืบทอดคลาสนี้ Unity จะทำการสั่งให้ทำฟังก์ชันดังต่อไปนี้

#### 1) Awake

Awake จะเป็นฟังก์ชันแรกที่ทำงานระหว่างที่ฉากนั้นเริ่มต้นขึ้นโดยปกติแล้วจะใช้ในการจัดการตัวแปรพื้นฐานและหา Script ที่เป็น Component เดียวกันกับตัว Object ที่ผูก Script

#### 2) Start

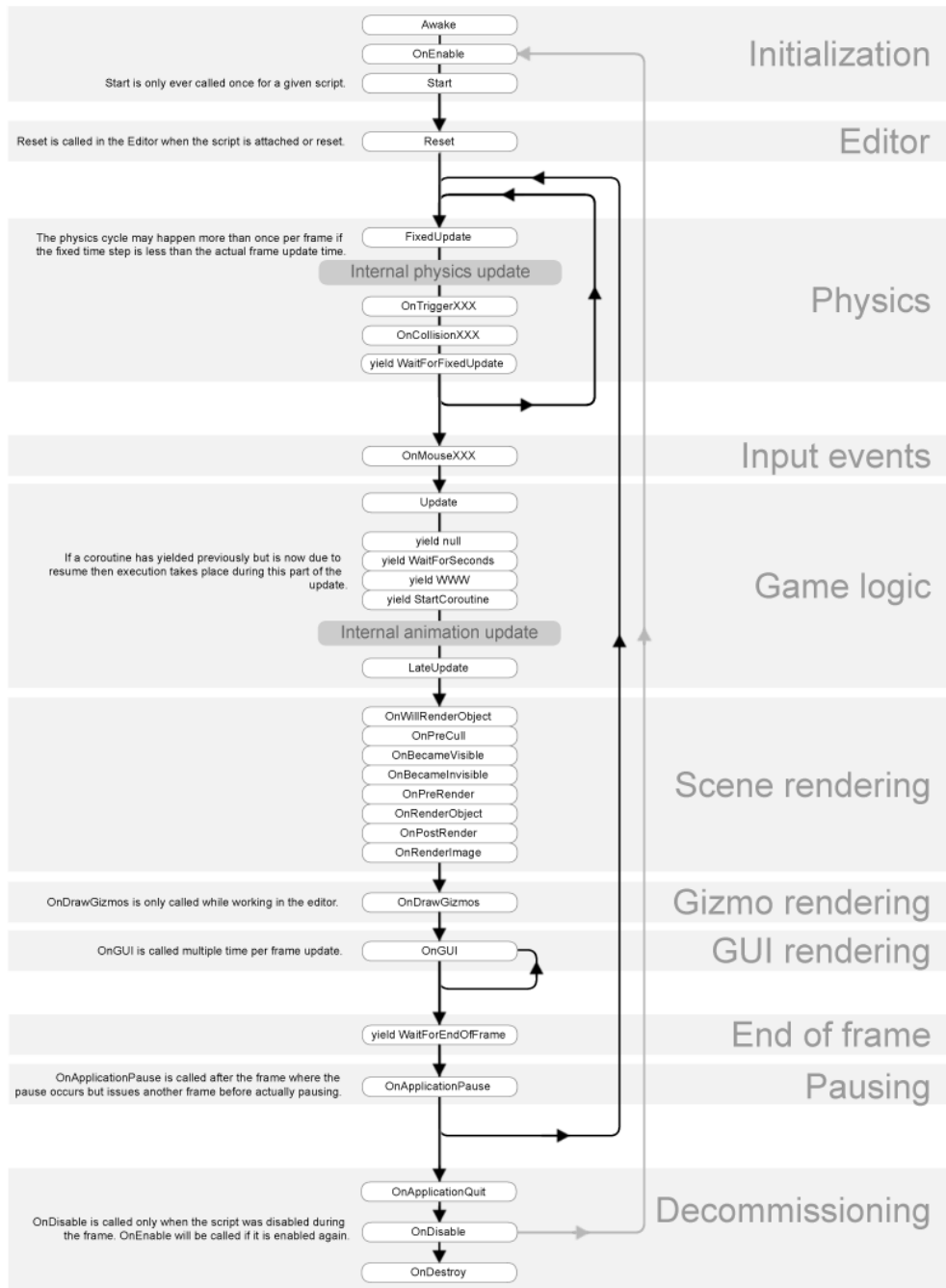
Start เป็นฟังก์ชันที่ 2 รันต่อจาก Awake จะทำการจัดการอ้างอิงถึง Object ที่อยู่ภายนอก

#### 3) FixedUpdate

#### 4) Update

สำหรับ FixedUpdate และ Update จะทำงานวนซ้ำอยู่ตลอดเวลาโดย FixedUpdate จะทำงานเมื่อระบบฟิสิกส์ของ Unity เปลี่ยนแปลงส่วน Update จะงานในทุก ๆ เฟรมที่ เกมประมวลผล

### 2.5.3 ลำดับการทำงานของ Script ใน Unity



รูปที่ 2.12 แผนภาพลำดับการทำงานของ MonoBehaviour

จากรูปที่ 2.12 จะเห็นว่าใน MonoBehavior [3] มีฟังก์ชันที่ทำงานอยู่หลายตัว แต่ฟังก์ชันที่ผู้พัฒนาเลือกใช้และให้ความสำคัญคือ Update เป็นฟังก์ชันที่ทำงานทุก ๆ เฟรมและยังใช้สำหรับตรวจสอบเงื่อนไขของต้นไม้พฤติกรรมที่เปลี่ยนแปลงเพื่อที่จะเลือกพฤติกรรมให้กับหุ่นยนต์

## 2.6 เกมและงานวิจัยที่เกี่ยวข้อง

เกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรมเป็นเกมที่ประยุกต์การควบคุมจากการใช้เมาส์ที่เป็นแบบเรียลไทม์มาเป็นแบบเทอร์นเบส โดยผู้เล่นจะไม่สามารถปรับเปลี่ยนรูปแบบของต้นไม้พฤติกรรมได้ระหว่างที่เกมดำเนินการ โดยการพัฒนาเกมจะต้องศึกษาข้อมูลเกมจากการศึกษาเกมและงานวิจัยที่เกี่ยวข้องดังนี้

### เกมวางแผนกลยุทธ์

#### 2.6.1 Red Alert 2

เกม Red Alert 2 [4] เป็นเกมวางแผนการรบที่มียูนิตในการควบคุม จุดเด่นของเกมนี้อีกคือสามารถควบคุมยูนิตได้ทั้งแบบเป็นกลุ่มและแบบเดี่ยว เพื่อให้ทำตามคำสั่งที่ต้องการได้และสามารถปรับเปลี่ยนคำสั่งได้ตลอดเวลา อีกทั้งยังสามารถสร้างคำสั่งแบบเรียงตามลำดับและทำซ้ำได้ เป้าหมายของเกมนี้อีกคือการทำลายสิ่งปลูกสร้างของฝ่ายตรงข้ามให้หมด โดยมีตัวอย่างหน้าจอระหว่างเล่นเกมดังรูปที่ 2.13



รูปที่ 2.13 ตัวอย่างระหว่างเล่นเกม Red Alert 2

### ความคล้ายคลึงกับเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) เป็นเกมแนววางแผนการรบ
- 2) เป้าหมายในการเอาชนะฝ่ายตรงข้ามโดยการทำลายสิ่งก่อสร้างฝ่ายตรงข้าม

### ความแตกต่างกับเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) มีการควบคุมแบบเรียลไทม์และผู้ใช้ต้องควบคุมด้วยเมาส์
- 2) ชุดคำสั่งไม่สามารถสร้างเงื่อนไขได้ ทำได้เพียงแบบวนซ้ำ
- 3) ทักษะวิสัยในเกมมีจำกัดไม่สามารถเห็นพื้นที่ทั้งหมดได้

#### 2.6.2 หมากรุก

หมากรุก [5] เป็นเกมกระดานที่มุ่งชนะ ฝ่ายตรงข้ามโดยการพยายามกำจัดตัวขุนของฝ่ายตรงข้าม โดยอาศัยลักษณะการเดินที่เฉพาะตัวของหมากแต่ละตัวดังรูปที่ 2.14



รูปที่ 2.14 ตัวอย่างเกมหมากรุก

### ความคล้ายคลึงกับเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) เป็นเกมแนววางแผนการรบที่ต้องคาดการณ์สิ่งที่จะเกิดขึ้นล่วงหน้า
- 2) เป้าหมายในการเอาชนะฝ่ายตรงข้าม
- 3) สามารถมองเห็นภาพรวมของเกมได้

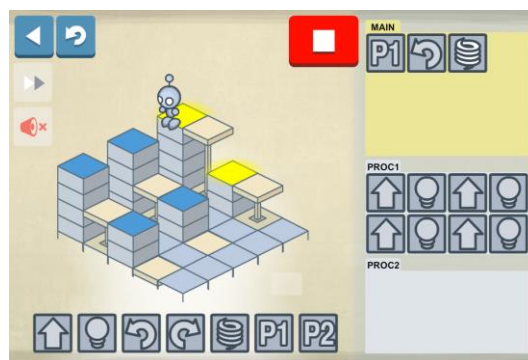
### ความแตกต่างกับเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) หมากแต่ละตัวมีลักษณะการเดินและหน้าที่แตกต่างกันอย่างชัดเจน แต่หุ่นยนต์ในเกมกลยุทธ์นั้นจะมีความสามารถต่างกันไปตามต้นไม้พฤติกรรมที่ผู้เล่นสร้าง
- 2) เป็นเกมแนวเทิร์นเบส
- 3) แนวการเดินจะต้องเดินบนตารางที่จัดไว้และตามความสามารถในการเดินของหมากแต่ละตัว
- 4) ผู้ใช้ต้องออกคำสั่งในการควบคุมเอง

### เกมที่ประยุกต์การควบคุมในรูปแบบอื่น ๆ

#### 2.6.3 Lightbot

เกม Lightbot [6] เป็นเกมแนวแก้ปัญหาที่จะต้องสร้างคำสั่งโดยใช้บล็อกโค้ดเพื่อเปิดไฟให้ครบทุกดวงโดยผู้เล่นสามารถใช้เทคนิค การเวียนเกิดและฟังก์ชันในการแก้ปัญหาได้และทุกขั้นตอนหุ่นยนต์จะแสดงออกอย่างชัดเจนเพื่อให้ผู้เล่นได้สังเกตและแก้ไขจุดที่ผิดพลาด ดังตัวอย่างเกม ดังรูปที่ 2.15



รูปที่ 2.15 ตัวอย่างเกม Lightbot

### ความคล้ายคลึงกับเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) ใช้การควบคุมแบบบล็อกโค้ดและจะแก้ไขได้เมื่อจบการทำงานหรือสั่งหยุด

### ความแตกต่างกับเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) เป้าหมาย Lightbot ต้องการจะเปิดไฟที่มีทั้งหมด
- 2) เกม Lightbot ไม่สามารถใช้การควบคุมแบบสร้างเงื่อนไขได้
- 3) บล็อกโค้ดทำคำสั่งเป็นเส้นตรงไม่สามารถข้ามคำสั่งได้

#### 2.6.4 Code Combat

เกม Code Combat [7] เป็นเกมแนวผจญภัยโดยใช้ตัวละครในการสู้กับปีศาจเพื่อเดินทางไปยังด่านต่อไปมีการใช้ความสามารถต่าง ๆ และไอเทมแต่การควบคุมตัวละคนนั้นไม่ได้ควบคุมโดยการใส่ปุ่มธรรมดาแต่ต้องวางแผนและเขียนโค้ดเพื่อให้ทำตามคำสั่งต่าง ๆ และสามารถใส่คำสั่งที่ค่อนข้างอิสระและหลากหลายในการผ่านด่านดังรูปที่ 2.16



รูปที่ 2.16 ตัวอย่างเกม Code Combat

### ความคล้ายคลึงกับเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) สามารถใช้การควบคุมแบบวนซ้ำและเงื่อนไขได้อย่างอิสระ
- 2) สามารถสร้างฟังก์ชันได้

### ความแตกต่างกับเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) เป็นเกมแนวผจญภัย

## บทที่ 3

### การวิเคราะห์และออกแบบระบบ

ในบทนี้จะเกี่ยวกับการวิเคราะห์และการออกแบบระบบ เป็นขั้นตอนที่สำคัญสำหรับการวางแผนแนวทางในการพัฒนาระบบให้ตรงตามวัตถุประสงค์ตามที่ได้กำหนดไว้ เป็นการวางกรอบของงาน เพื่อเป็นการลดปัญหาหรือข้อผิดพลาดในช่วงของการพัฒนา รวมถึงเป็นการลดความซับซ้อนของระบบ อีกทั้งเป็นการวางกฎกติกาการเล่นของเกม โดยการวิเคราะห์และออกแบบระบบนั้นจะใช้การออกแบบเชิงวัตถุ (Object Oriented Design) ในการมองและเขียนโปรแกรมเชิงวัตถุ โดยใช้แผนภาพยูสเคส (Use Case Diagram) แผนภาพกิจกรรม (Activity Diagram) และ แผนภาพคลาสไดอะแกรม (Class Diagram) ในการออกแบบเกม เพื่อให้เห็นการทำงานของระบบเกมมากขึ้น

#### 3.1 การวิเคราะห์โครงสร้างของระบบ

เกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม เป็นเกมวางแผนกลยุทธ์ที่ออกแบบให้ผู้ใช้ควบคุมหุ่นยนต์แต่ละตัวโดยใช้ต้นไม้พฤติกรรม โดยการสร้างโหนดแต่ละชนิดแล้วนำมาสร้างความสัมพันธ์กัน โดยระบบจะนำเอาโหนดเหล่านั้นไปสร้างต้นไม้พฤติกรรมให้กับหุ่นยนต์ตัวที่เลือกจากนั้น เมื่อเริ่มเกมหุ่นยนต์ก็จะทำตามต้นไม้พฤติกรรมที่ผู้เล่นสร้างไว้ และเมื่อเกมจบลงจะสรุปผลของเกมให้ผู้เล่นทราบ โดยระบบต้องมีความสามารถดังนี้

- 1) ผู้เล่นสามารถสร้างต้นไม้พฤติกรรมให้กับหุ่นยนต์แต่ละตัว
- 2) ผู้เล่นสามารถเพิ่ม ลบ เชื่อมความสัมพันธ์ของโหนด
- 3) ผู้เล่นสามารถแก้ไขต้นไม้พฤติกรรมของหุ่นยนต์
- 4) ผู้เล่นสามารถสั่งให้เวลาในเกมเดินเร็วขึ้นขณะที่เกมเริ่ม
- 5) ผู้เล่นสามารถขีดขวางระหว่างที่เกมเริ่ม

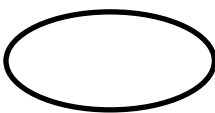
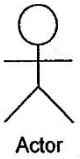


ในการวิเคราะห์โครงสร้างของระบบจะกล่าวถึงการวิเคราะห์ปัญหา และความต้องการของระบบซึ่งแสดงด้วยยูเอ็มแอล (UML: Unified Modeling Language) โดยยูเอ็มแอลคือ โมเดลมาตรฐานที่ใช้หลักการออกแบบเชิงวัตถุ รูปแบบของภาษายูเอ็มแอลจะมีเครื่องหมายซึ่งเป็น

สัญลักษณ์ที่นำไปใช้ในโมเดลต่าง ๆ ยูเอ็มแอลจะมีข้อกำหนดกฎระเบียบต่าง ๆ ในการเขียนโปรแกรม โดยกฎระเบียบต่าง ๆ จะมีความหมายต่อการเขียนโปรแกรม โดยมีไดอะแกรมต่าง ๆ ดังนี้

### 3.1.1 ยูสเคสไดอะแกรม

แผนภาพยูสเคส (Use Case Diagram) คือแผนภาพที่ระบบต้องกระทำทั้งหมด และ แสดงความสัมพันธ์ระหว่างระบบงานและสิ่งที่อยู่นอกระบบงาน เพื่อให้ง่ายต่อการเข้าใจการทำงานของระบบ และการนำมาใช้อีกทั้งยังถูกนำไปใช้เป็นพื้นฐานเพื่อการสร้าง แผนภาพ (Diagram) ชนิดอื่นในขั้นตอนต่อไป โดยสัญลักษณ์ที่ใช้ในยูสเคส [8] แสดงได้ดังตารางที่ 3.1

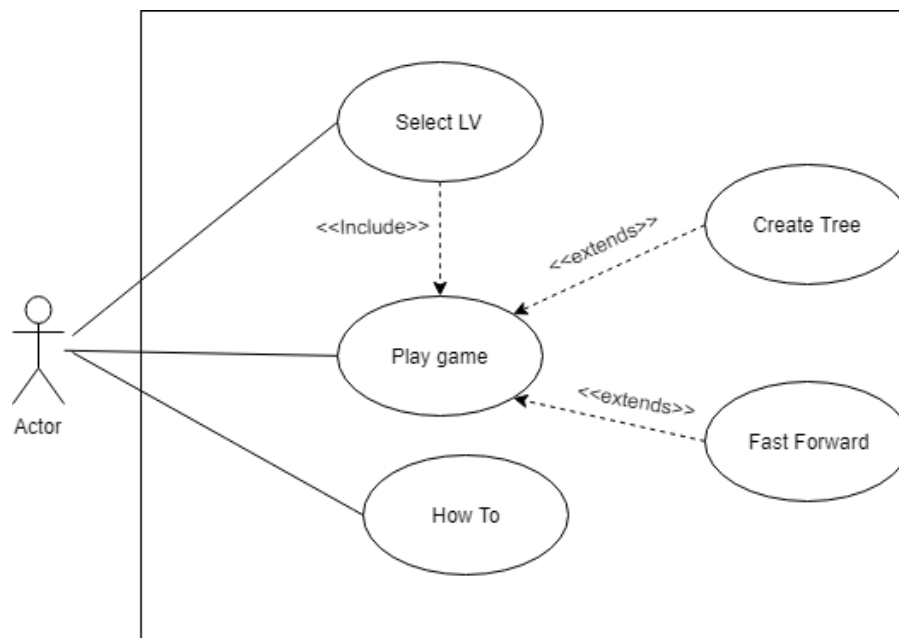
ตารางที่ 3.1 สัญลักษณ์ที่ใช้ในยูสเคสไดอะแกรม

| ชื่อ            | สัญลักษณ์   | ความหมาย   |
|-----------------|---|--|
| Use Case        |   | ส่วนของงาน หรือกระบวนการที่จะเกิดขึ้นกับระบบ   |
| Actor           |  | ผู้เกี่ยวข้องกับระบบ ซึ่งรวมทั้ง Primary Actor และ Stakeholder Actor ที่เป็นมนุษย์ ในที่นี้จะใช้สัญลักษณ์รูปคน (Stick Man Icon) เหมือนกัน พร้อมทั้งเขียนชื่อ Actor ไว้ด้านล่างของสัญลักษณ์ด้วย |
| System Boundary |  | อธิบายขอบเขตภายในระบบซึ่งภายในกรอบจะมีการแสดงส่วนสัมพันธ์ระหว่างผู้ใช้กับระบบ  |
| Connection      |  | เส้นที่ลากเชื่อมต่อระหว่าง Actor กับ Use Case ที่มีปฏิสัมพันธ์กัน  |

ตารางที่ 3.1 สัญลักษณ์ที่ใช้ในยูสเคสไดอะแกรม (ต่อ)

| ชื่อ                 | สัญลักษณ์  | ความหมาย   |
|----------------------|--|--|
| Include Relationship | <pre>       &lt;&lt;Include&gt;&gt;       -----&gt;           </pre> | ความสัมพันธ์ขยายที่เกิดขึ้นในกรณี Use Case หนึ่งสามารถเรียกใช้ Use Case หนึ่งได้   |
| Extends Relationship | <pre>       &lt;&lt;Extend&gt;&gt;       -----&gt;           </pre>  | ความสัมพันธ์แบบขยายหรือเพิ่ม เกิดขึ้นในกรณีที่บาง Use Case ดำเนินกิจกรรมของตนเองไปตามปกติ แต่อาจจะมีเงื่อนไขหรือสิ่งกระตุ้นบางอย่างที่ส่งผลให้กิจกรรมตามปกติของ Use Case นั้นถูกรบกวนจนเบี่ยงเบนไป |

ยูสเคสไดอะแกรมของเกมวางแผนกลยุทธ์โดยต้นไม้พฤติกรรม แสดงได้ดังรูปที่ 3.1



รูปที่ 3.1 ยูสเคสไดอะแกรมของเกมวางแผนกลยุทธ์โดยต้นไม้พฤติกรรม

จากรูปที่ 3.1 สามารถอธิบาย Use Case ได้ดังนี้

ตารางที่ 3.2 Use Case Specification ของ Select LV

|                     |   |
|---------------------|---|
| Use Case Id         | UC01 – Select LV  |
| Scenarios           | ผู้เล่นต้องการเลือกด่านที่เกมกำลังจะเริ่ม                                       |
| Trigger Event       | เลือกเมนู Play Game   |
| Brief Description   | ผู้เล่นจะต้องเลือกด่านที่มีอยู่ในรายการด่านที่สามารถเล่นได้จากหน้าส่วนต่อประสาน |
| Actors              | ผู้เล่น   |
| Related Use Case    | -   |
| Precondition        | ผู้เล่นยังไม่ได้เลือกด่านที่ต้องการเล่น   |
| Postcondition       | ผู้เล่นเลือกด่านที่ต้องการเล่นแล้ว  |
| Flow of Events      | 1) ผู้เล่นทำการเลือกด่านที่ต้องการเล่น  |
| Exception Condition | -   |

ตารางที่ 3.3 Use Case Specification ของ How To

|                     |  |
|---------------------|--|
| Use Case Id         | UC02 – How To  |
| Scenarios           | ผู้เล่นต้องการทราบข้อมูลเกี่ยวกับเกมก่อนเริ่มเล่น  |
| Trigger Event       | เลือกเมนู HowTo  |
| Brief Description   | ผู้เล่นต้องการที่จะทราบข้อมูลเกี่ยวกับตัวละคร แนวเกม สภาพแวดล้อม และสิ่งที่อยู่ภายในเกม  |
| Actors              | ผู้เล่น  |
| Related Use Case    | -  |
| Precondition        | -  |
| Postcondition       | -  |
| Flow of Events      | <ol style="list-style-type: none"> <li>1) ผู้เล่นเข้ามาหน้า How to</li> <li>2) ผู้เล่นเลือกปุ่มคำอธิบายของสิ่งต่าง ๆ ภายในเกม</li> <li>3) คำอธิบายเหล่านั้นปรากฏขึ้นมาบนส่วนต่อประสาน</li> </ol> |
| Exception Condition | -  |

ตารางที่ 3.4 Use Case Specification ของ Play Game

|                     |  |
|---------------------|--|
| Use Case Id         | UC03 – Play Game   |
| Scenarios           | ผู้เล่นต้องการเริ่มเกมที่สร้างขึ้นไม่พบเหตุการณ์เสร็จแล้ว  |
| Trigger Event       | เลือกปุ่ม Play บนส่วนต่อประสาน   |
| Brief Description   | ผู้เล่นต้องการที่จะให้หุ่นยนต์ในเกมทำตามต้นไม้เหตุการณ์ที่ผู้เล่นสร้างไว้  |
| Actors              | ผู้เล่น  |
| Related Use Case    | UC 01 – Select LV<br>UC04 – Create Tree<br>UC05 – Fast Forward   |
| Precondition        | ผู้เล่นสร้างขึ้นไม่พบเหตุการณ์ให้กับหุ่นยนต์เรียบร้อยแล้ว  |
| Postcondition       | สถานะของเกมเปลี่ยนเป็นเริ่มเกม   |
| Flow of Events      | <ol style="list-style-type: none"> <li>1) ผู้เล่นมาจาก UC01 – Select LV</li> <li>2) ผู้เล่นเข้าสู่หน้าจอเกม</li> <li>3) ผู้เล่นเลือกหุ่นยนต์แล้วเข้าสู่ UC04 - CreateTree</li> <li>4) ผู้เล่นกดปุ่ม Play</li> <li>5) ผู้เล่นสามารถเปลี่ยนความเร็วของเวลาผ่าน UC05 - FastForward</li> </ol> |
| Exception Condition | -  |

ตารางที่ 3.5 Use Case Specification ของ Create Tree

|                     |   |
|---------------------|---|
| Use Case Id         | UC04 – Create Tree  |
| Scenarios           | ผู้เล่นต้องสร้างต้นไม้พฤติกรรมให้กับหุ่นยนต์  |
| Trigger Event       | ผู้ใช้เลือกหุ่นยนต์ที่ต้องการสร้างต้นไม้พฤติกรรม  |
| Brief Description   | ผู้เล่นต้องการเข้าสู่ส่วนต่อประสานที่ใช้สำหรับสร้าง ปรับปรุงแก้ไขต้นไม้พฤติกรรม ของหุ่นยนต์ตัวที่เลือก  |
| Actors              | ผู้เล่น   |
| Related Use Case    | -   |
| Precondition        | ผู้เล่นเลือกหุ่นยนต์ที่ต้องการสร้างต้นไม้พฤติกรรม   |
| Postcondition       | หุ่นยนต์ที่เลือกได้รับการแก้ไขต้นไม้พฤติกรรม  |
| Flow of Events      | <ol style="list-style-type: none"> <li>1) ผู้เล่นเข้ามาหน้าแก้ไขต้นไม้พฤติกรรม</li> <li>2) ผู้เล่นเปลี่ยนแปลงต้นไม้พฤติกรรมผ่านส่วนต่อประสาน</li> </ol> |
| Exception Condition | -   |

ตารางที่ 3.6 Use Case Specification ของ Fast Forward

|                     |  |
|---------------------|--|
| Use Case Id         | UC05 – Fast Forward  |
| Scenarios           | ผู้เล่นต้องการปรับความเร็วในการแสดงพฤติกรรม  |
| Trigger Event       | เลือกปุ่ม Fast Forward   |
| Brief Description   | ผู้เล่นต้องการที่จะปรับความเร็วในการแสดงพฤติกรรมของหุ่นยนต์ให้เร็วขึ้น หรือช้าลงเพื่อสังเกต หรือเร่งให้เกมจบเร็วขึ้น |
| Actors              | ผู้เล่น  |
| Related Use Case    | -  |
| Precondition        | ผู้เล่นกดปุ่มเริ่มเกม  |
| Postcondition       | เวลาในระบบถูกปรับตามที่ผู้เล่นกำหนด  |
| Flow of Events      | 1) ผู้เล่นเลือกปุ่ม Fast Forward   |
| Exception Condition | -  |

### 3.2 การวิเคราะห์พฤติกรรมของระบบ

การวิเคราะห์พฤติกรรมของระบบเป็นส่วนของการอธิบายการทำงาน ซึ่งจะใช้ชีทเควซ์ ไดอะแกรมและแผนภาพกิจกรรมในการอธิบายขั้นตอนการทำงาน และการทำงานร่วมกันของวัตถุ ด้วยการติดต่อสื่อสาร มีรายละเอียดดังนี้

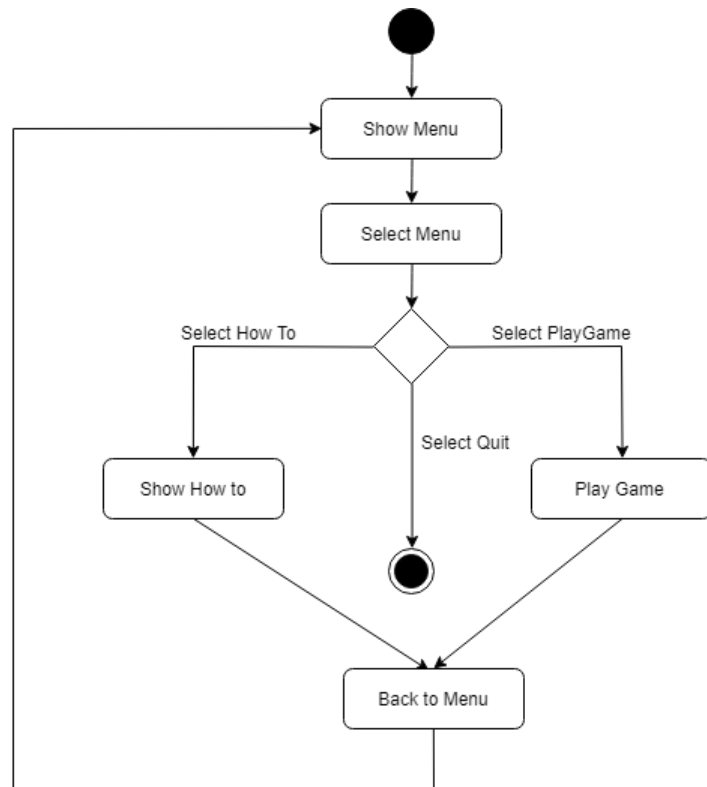
#### 3.2.1 แผนภาพกิจกรรม (Activity Diagram)

แผนภาพกิจกรรมของระบบ [9] เป็นแผนภาพที่แสดงถึงขั้นตอนการทำงานของระบบ โดยประกอบไปด้วยสถานะต่าง ๆ ที่เกิดขึ้นระหว่างการทำงาน และผลจากการทำงานในขั้นตอนต่าง ๆ จากสัญลักษณ์ที่ใช้ในการสร้างแผนภาพคลาสจากตารางที่ 3.7

ตารางที่ 3.7 สัญลักษณ์แผนภาพกิจกรรมของระบบ

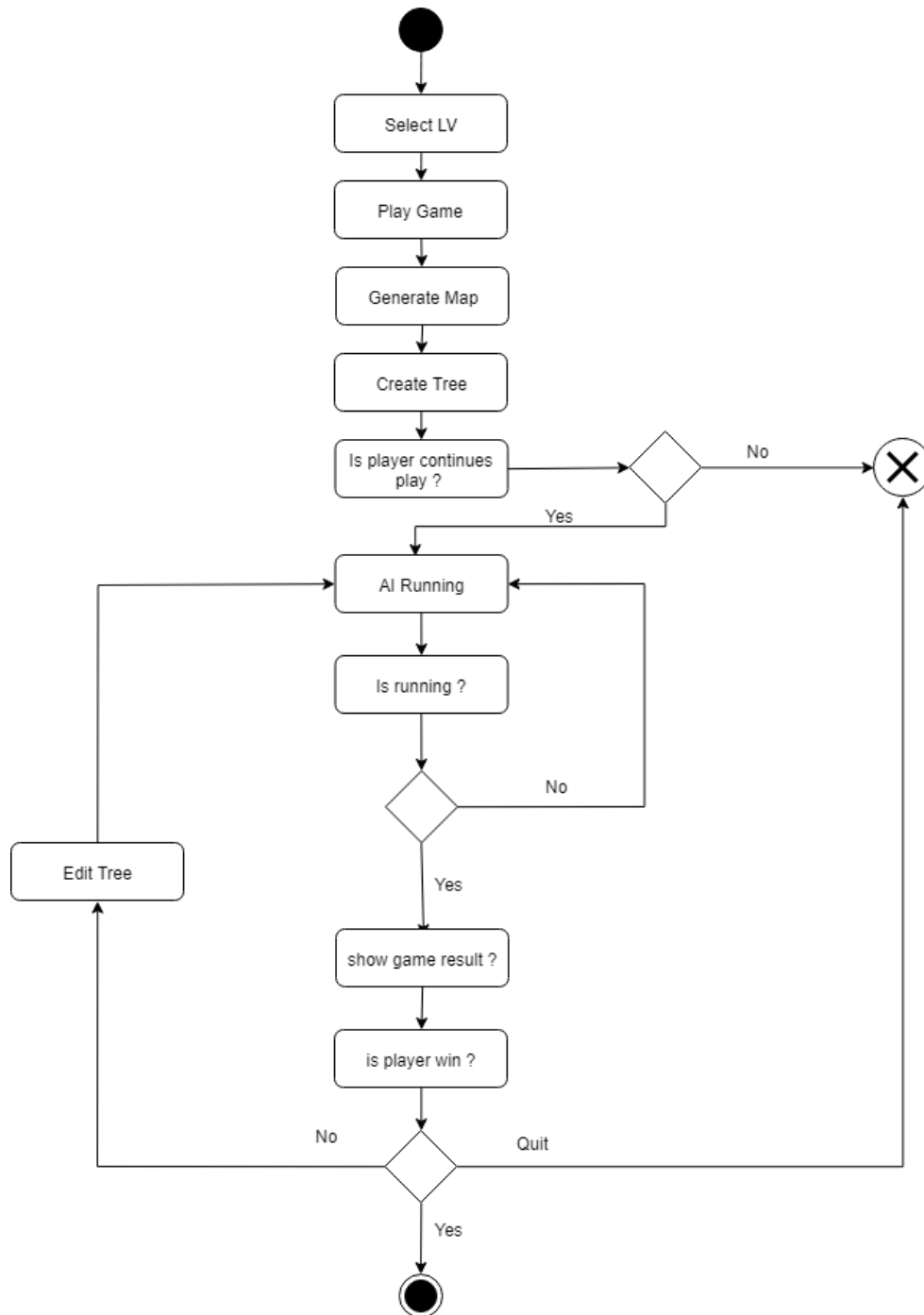
| ชื่อ      | สัญลักษณ์ | ความหมาย   |
|-----------|-----------|--|
| Start     | ●         | สถานะเริ่มต้น                                      |
| Destroy   | ⊗         | สถานะสุดท้ายที่ไม่ใช่ลำดับงานปกติ                  |
| Terminate | ⦿         | สถานะสุดท้าย                                       |
| Activity  | ▭         | กิจกรรมต่าง ๆ ที่เกิดขึ้น                          |
| Control   | →         | แสดงการไหลของข้อมูลในกิจกรรมหนึ่ง ๆ                |
| Condition | ◇         | กรอบการตัดสินใจ เพื่อให้เลือกทิศทางต่อไปของกิจกรรม |

จากตารางที่ 3.7 จะสามารถสร้างแผนภาพกิจกรรมของระบบได้ดังรูปที่ 3.2 และรูปที่ 3.3



รูปที่ 3.2 แผนภาพกิจกรรมของระบบ

จากรูปที่ 3.2 เมื่อเกมถูกเรียกขึ้นจะปรากฏหน้าจอเมนูให้ผู้เล่นได้เลือกเล่น โดยจะมีอยู่ 3 สิ่งหลัก ๆ คือ เล่นเกม อ่านวิธีการเล่น และออกจากเกม หากผู้เล่นเลือก เมนูอ่านวิธีการเล่นจะนำผู้เล่นไปสู่หน้าจอแสดงวิธีการเล่นและข้อมูลต่าง ๆ ภายในเกมหากผู้เล่นเลือก Play จะทำให้ผู้เล่นเข้าสู่แผนภาพกิจกรรมที่ 2 ดังรูปที่ 3.3 และถ้าผู้เล่นเลือกเมนู Quit จะทำให้เกมปิดตัวลง






รูปที่ 3.3 แผนภาพกิจกรรมของการเล่นเกม

จากรูปที่ 3.3 จะสามารถอธิบายแผนภาพกิจกรรมได้ดังนี้ เมื่อผู้เล่นเลือกเล่นเกม ก่อนที่จะเริ่มเล่นผู้เล่นต้องทำการเลือกด่านที่จะเล่นก่อน จากนั้นระบบจะทำการสร้างด่านให้ ในกิจกรรมนี้ผู้เล่นสามารถออกจากเกมได้ หากไม่ผู้เล่นจะต้องสร้างต้นไม้พฤติกรรมให้กับหุ่นยนต์แต่ละตัว จากนั้นจะทำการกดปุ่มเล่นเพื่อให้หุ่นยนต์แต่ละตัวทำตามต้นไม้พฤติกรรมระหว่างนั้นจะมีการนับคะแนนและตรวจสอบว่าผู้เล่นได้อยู่ในเงื่อนไขที่มีหุ่นยนต์หรือไม่ จากนั้นแสดงหน้าสรุปผล ถ้าหากผู้เล่นแพ้หรือเสมอผู้เล่นจะเลือกระหว่างกลับไปแก้ต้นไม้พฤติกรรมและเล่นใหม่อีกครั้งหรือผู้เล่นจะเลือกออกจากเกมก็ได้ หากผู้เล่นชนะเกมจะจบลงแล้วนำผู้เล่นไปด่านต่อไป



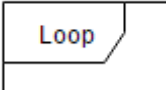
### 3.2.2 ซีควেনซ์ไดอะแกรม (Sequence Diagram)

ซีควেনซ์ไดอะแกรม (Sequence Diagram) [10] เป็นแผนภาพที่สามารถแสดงให้เห็นถึงการปฏิสัมพันธ์ระหว่างออบเจกต์ของคลาสตามลำดับของเวลาเป็นสำคัญ ซึ่งสัญลักษณ์ที่ใช้ในซีควেনซ์ไดอะแกรมแสดงได้ดังตารางที่ 3.8

ตารางที่ 3.8 สัญลักษณ์ที่ใช้ในซีควেনซ์ไดอะแกรม

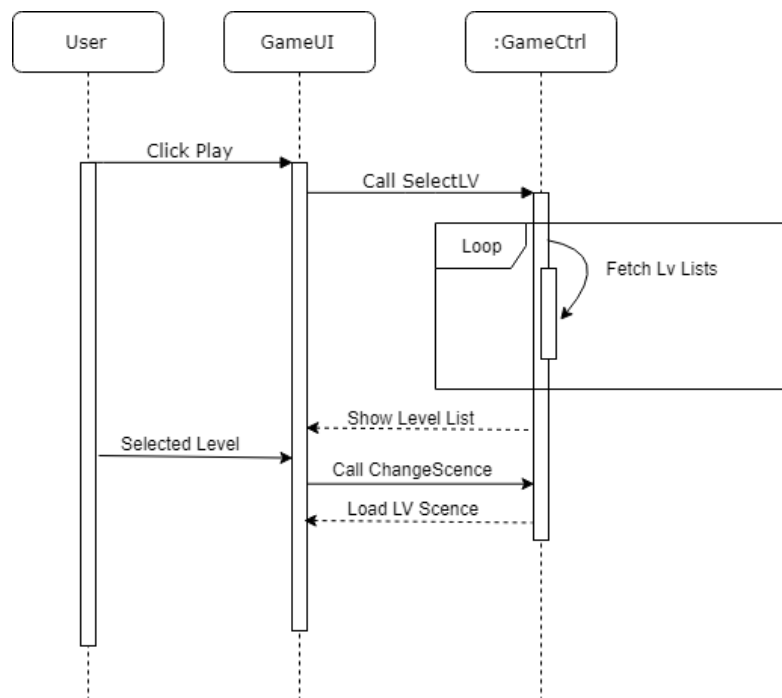
| ชื่อ                        | สัญลักษณ์   | ความหมาย   |
|-----------------------------|---|--|
| Object Class                |  | ชื่อออบเจกต์และคลาส จะบอกว่าออบเจกต์ในคลาสนี้ที่ต้องทำหน้าที่ตอบสนองต่อ Actor ในแต่ละกิจกรรม |
| Lifeline                    |  | เส้นแสดงชีวิตของออบเจกต์หรือคลาส   |
| Focus of control/Activation |  | จุดเริ่มต้นหรือจุดสิ้นสุดของแต่ละกิจกรรมในระหว่างที่มีชีวิตอยู่                              |

ตารางที่ 3.8 สัญลักษณ์ที่ใช้ในซีเควนซ์ไดอะแกรม (ต่อ)

| ชื่อ                     | สัญลักษณ์   | ความหมาย  |
|--------------------------|---|---|
| Message                  |  | คำสั่งหรือฟังก์ชันที่ออบเจกต์หนึ่งส่งให้อีกออบเจกต์หนึ่ง ซึ่งสามารถส่งกลับได้ด้วย |
| Callback/Self Delegation |  | การประมวลผลและคืนค่าที่ได้ภายในออบเจกต์เดียวกัน                                   |
| Loop                     |  | แสดงกรอบส่วนที่มีการเรียกซ้ำ  |

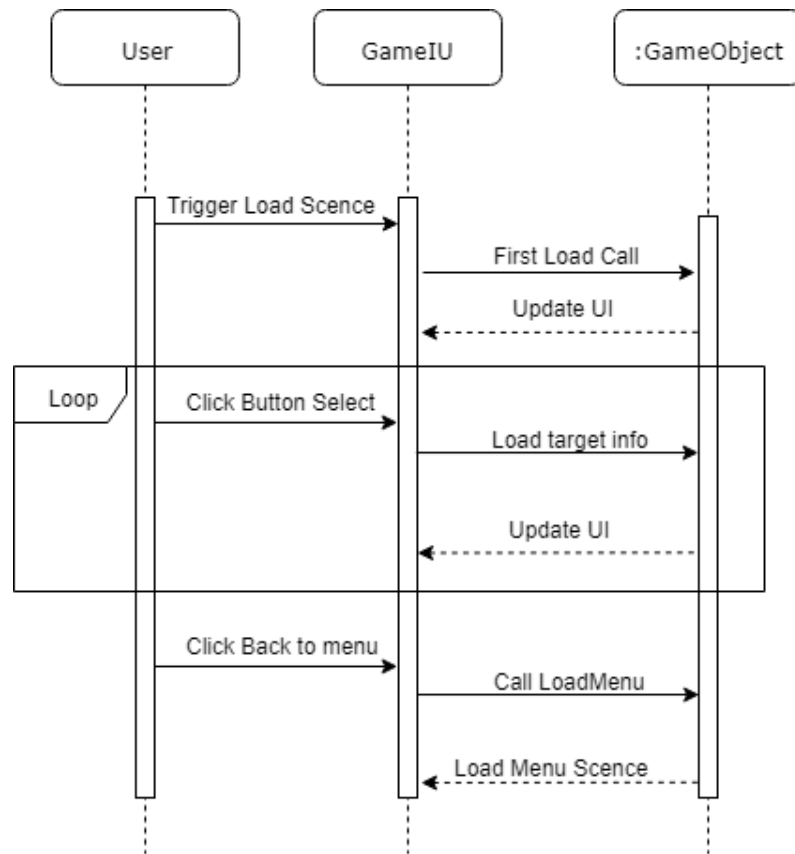
ซีเควนซ์ไดอะแกรมของระบบ มีดังนี้

1) Sequence Diagram ของ Select LV



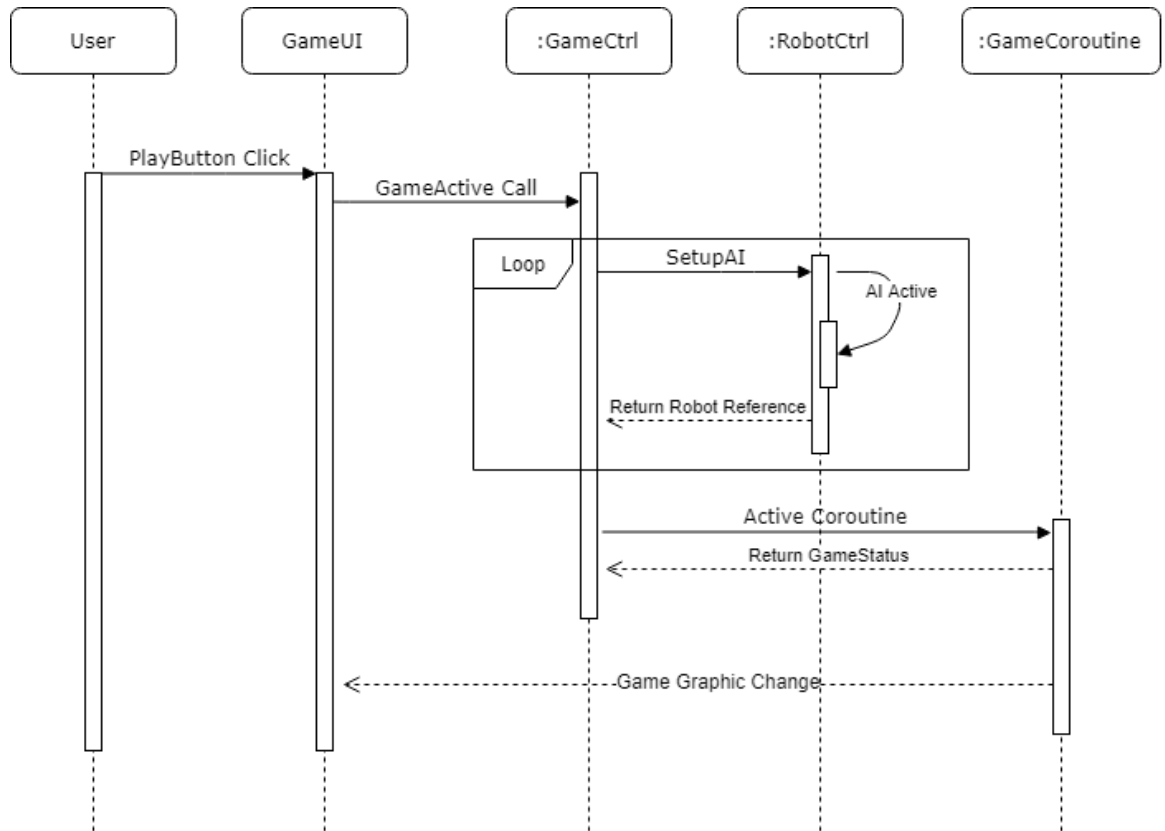
รูปที่ 3.4 Sequence Diagram ของ Select LV

## 2) Sequence Diagram ៗៗ How To



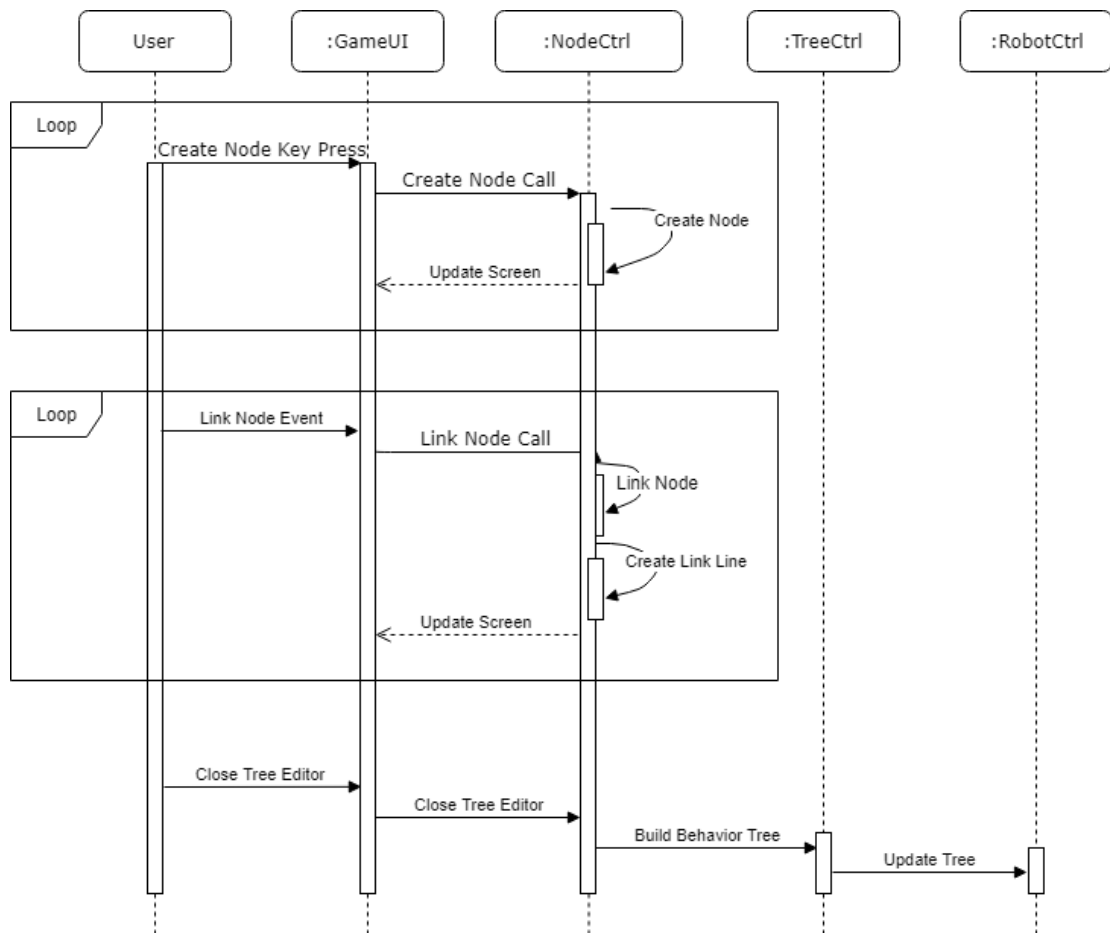
រូប្តី 3.5 Sequence Diagram ៗៗ How To

## 3) Sequence Diagram ของ Play Game



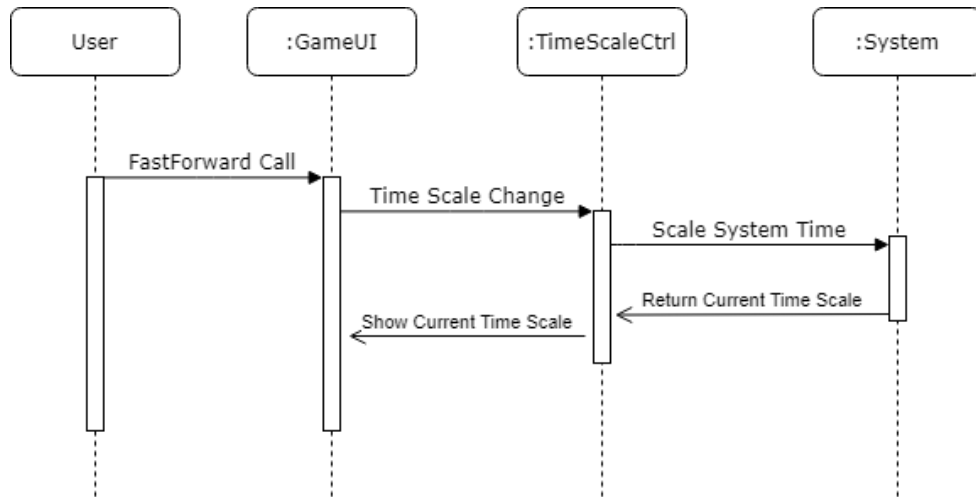
รูปที่ 3.6 Sequence Diagram ของ Play Game

## 4) Sequence Diagram ของ Create Tree



รูปที่ 3.7 Sequence Diagram ของ Create Tree

### 5) Sequence Diagram ของ Fast Forward



รูปที่ 3.8 Sequence Diagram ของ Fast Forward


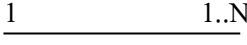
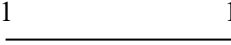
### 3.2.3 แผนภาพคลาสไดอะแกรม (Class Diagram)

แผนภาพคลาส [11] เป็นแผนภาพที่ใช้แสดงคลาสและความสัมพันธ์ระหว่างคลาสของระบบ โดยแผนภาพจะแสดงแอตทริบิวต์ที่ใช้บอกคุณสมบัติของคลาสและโอเปอเรชันที่ใช้บอกหน้าที่การทำงานของคลาส ในระบบหนึ่ง ๆ จะประกอบด้วยคลาสจำนวนมาก แต่ละคลาสต่างทำหน้าที่แตกต่างกันไปโดยจะต้องมีการประสานการทำงานของแต่ละคลาสเข้าด้วยกัน

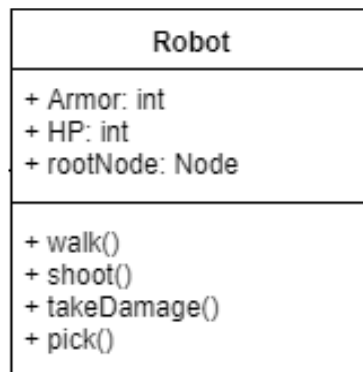
ตารางที่ 3.9 สัญลักษณ์ที่ใช้ในแผนภาพคลาสไดอะแกรม

| ชื่อ        | สัญลักษณ์ | ความหมาย   |
|-------------|-----------|--|
| Class       |           | <ul style="list-style-type: none"> <li>- ชื่อของคลาส</li> <li>- คุณลักษณะของคลาส</li> <li>- กิจกรรมที่สามารถกระทำกับออบเจกต์นั้นได้</li> </ul> |
| Aggregation |           | เป็นความสัมพันธ์ระหว่างคลาสแบบขึ้นต่อกันและมีความเกี่ยวข้องกันเสมอ   |

ตารางที่ 3.9 สัญลักษณ์ที่ใช้ในแผนภาพคลาสไดอะแกรม (ต่อ)

| ชื่อ                    | สัญลักษณ์   | ความหมาย  |
|-------------------------|---|---|
| Composition             |    | เป็นความสัมพันธ์ระหว่างคลาสแบบเป็นส่วนหนึ่ง โดยจะมีคลาสหนึ่งเป็นคลาสใหญ่เป็นหลักและมีคลาสอื่นเป็นส่วนประกอบ |
| one to many association |    | ความสัมพันธ์เชิงโครงสร้าง อธิบายการเชื่อมโยงกันระหว่างวัตถุ ใช้กำหนดความสัมพันธ์แบบหนึ่งต่อหลาย             |
| one to one association  |  | ความสัมพันธ์เชิงโครงสร้าง อธิบายการเชื่อมโยงกันระหว่างวัตถุ ใช้กำหนดความสัมพันธ์แบบหนึ่งต่อหนึ่ง            |

จากตารางที่ 3.9 จะสามารถสร้างคลาสไดอะแกรมของเกมได้ดังต่อไปนี้

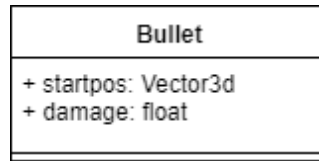


รูปที่ 3.9 คลาสไดอะแกรมของหุ่นยนต์

จากรูปที่ 3.9 หุ่นยนต์นั้นประกอบด้วยแอตทริบิวต์ Armor ซึ่งแทนค่าเกราะและ HP ใช้แทนค่าพลังชีวิตของหุ่นยนต์ และเมธอดที่ใช้ในการแสดงกิริยาต่างๆของหุ่นยนต์คือ วิ่ง ยิง รับค่าเสียหาย และ หยิบ

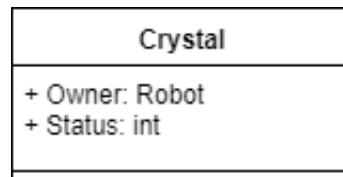
จากคลาสหุ่นยนต์จะเห็นว่าทุกการกระทำที่แสดงออกของหุ่นยนต์นั้น อยู่ในคลาสนี้ทั้งหมด เนื่องจากว่าตัวโหนดจะเป็นเพียงคลาสที่ใช้สั่งให้หุ่นยนต์ทำงานเท่านั้น หากเก็บข้อมูลไว้

ในโหนด จะทำให้ต้องเก็บแอตทริบิวต์ ซ้ำกับที่มีอยู่ในหุ่นยนต์ ดังนั้นจึงทำการเก็บไว้ที่หุ่นยนต์ตัวนั้น ๆ เมื่อต้องการใช้ จะส่งค่าให้กับเมธอดนั้นแทน



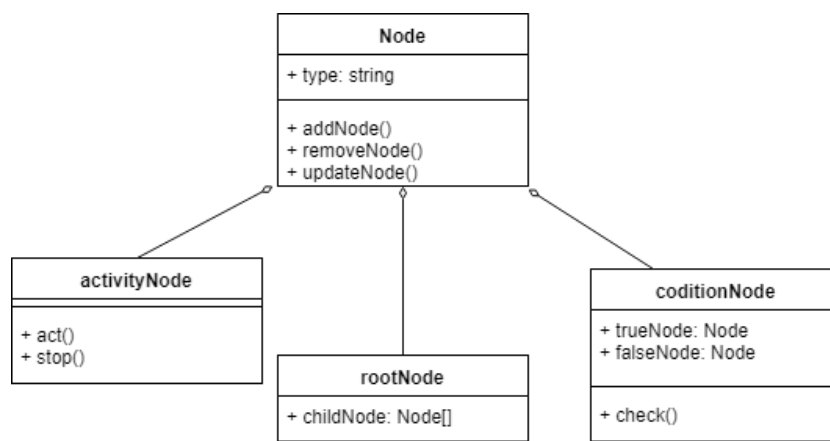
รูปที่ 3.10 คลาสไดอะแกรมของกระสุน

จากรูปที่ 3.10 กระสุนจะถูกสร้างใหม่เมื่อหุ่นยนต์ยิงกระสุนออกไป โดยจะทำการเก็บตำแหน่งและความแรงไว้กับกระสุน โดยการเก็บค่าตำแหน่งนั้นเพื่อไว้ใช้สำหรับคำนวณค่าความเสียหาย จากระยะทางที่ห่างออกไป ตามกฎของเกม



รูปที่ 3.11 คลาสไดอะแกรมของคริสตัล

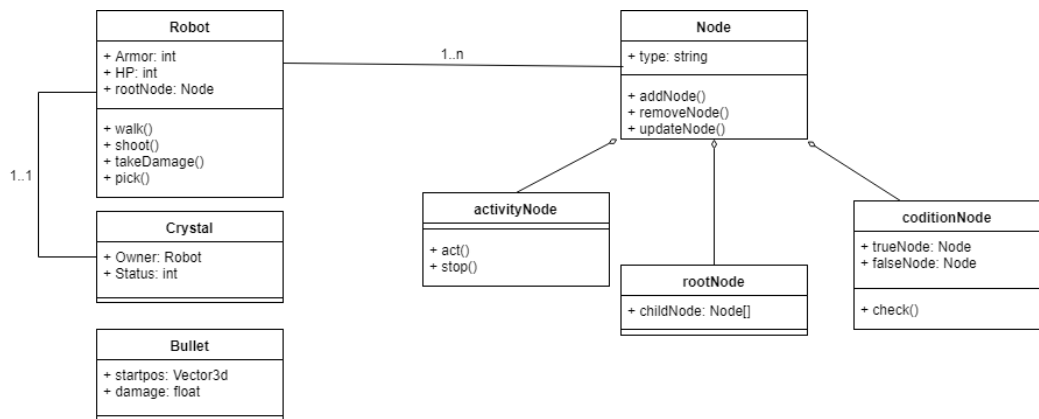
จากรูปที่ 3.11 สำหรับคริสตัลนั้นจะเก็บเพียงแค่สถานะ และเก็บหุ่นยนต์ตัวไหนเป็นเจ้าของเท่านั้น โดยพฤติกรรมอื่น ๆ ที่กระทำกับคลาสนี้จะมีเพียงหุ่นยนต์เท่านั้น ดังนั้นเมื่อเกิดเหตุการณ์ดังกล่าวขึ้น เช่นการเก็บจะทำการส่งค่าคริสตัลไปประมวลผลที่คลาสของหุ่นยนต์



รูปที่ 3.12 คลาสไดอะแกรมของโหนดต่าง ๆ

จากรูปที่ 3.12 แสดงคลาสไดอะแกรมของโหนด ซึ่งโหนดมีหลายประเภทจึงให้การสืบทอดคุณสมบัติกันคุณสมบัติพื้นฐานที่โหนดทั่วไปต้องมีคือแอตทริบิวต์ ชนิด และเมธอด เพิ่ม ลบ และแก้ไข จากนั้นก็จะแยกเป็นโหนดต่าง ๆ คือโหนดกิจกรรมเอาไว้สำหรับเลือกแสดงว่าหุ่นยนต์

จะแสดงกิริยาใดออกมาจะมีเมธอด แสดงกิริยา โหนดรากใช้สำหรับเป็นโหนดเริ่มต้นโดยจะมีแอตทริบิวต์ โหนดลูกซึ่งเก็บเป็นอาเรย์ในการเก็บโหนดต่าง ๆ ตามลำดับ และโหนดการตัดสินใจจะมีแอตทริบิวต์ที่ใช้เก็บที่อยู่ของโหนดเมื่อการตัดสินใจเป็นจริง และเป็นเท็จ และมีเมธอดตรวจสอบเพื่อตรวจสอบว่าเงื่อนไขนั้นถูกต้องหรือไม่ จากนั้นเมื่อนำคลาสมารวมกันจะได้ความสัมพันธ์รูปที่ 3.13



รูปที่ 3.13 คลาสไดอะแกรมความสัมพันธ์ในระบบ

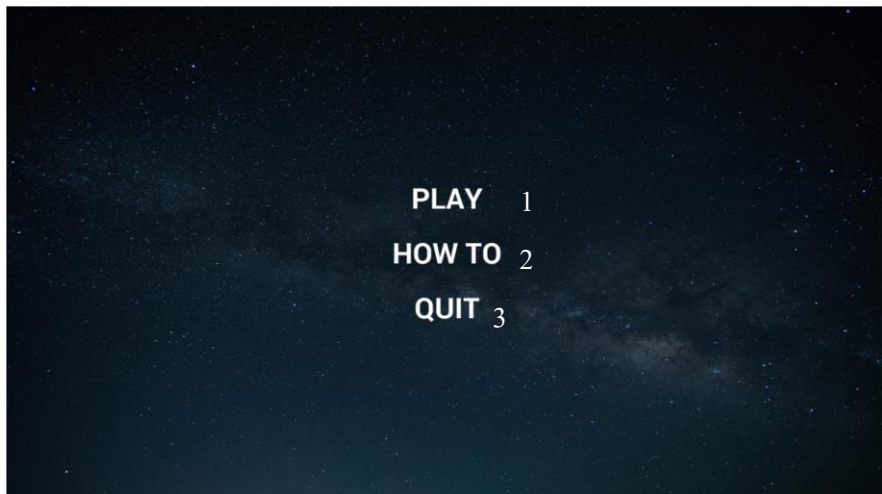
จากรูปที่ 3.13 จะอธิบายได้ว่า คลาสหุ่นยนต์แต่ละตัวนั้นมีต้นไม้พฤติกรรมเป็นของตัวเอง โดยที่หุ่นยนต์หนึ่งตัวมีได้หลายโหนดแล้วโหนดนั้นจะแตกต่างกันไปซึ่งโหนดเริ่มต้นจะเป็นโหนดราก ซึ่งสามารถมีโหนดลูกได้กี่โหนดก็ได้และมีโหนดที่เป็นเงื่อนไขและโหนดกิจกรรมประกอบกันจนเป็นต้นไม้พฤติกรรมของหุ่นยนต์ตัวนั้น ๆ และยังมีคลาสกระสุนที่เก็บตำแหน่งของจุดที่ยิงไว้เพื่อคำนวณความเสียหายเมื่อมีการกระทบกับหุ่นยนต์ตัวนั้น ๆ และสุดท้ายคลาสคริสตัล เป็นเป้าหมายของหุ่นยนต์ที่จะต้องเดินไปเก็บ จะใช้เก็บว่าขณะนี้หุ่นยนต์ที่เก็บคริสตัลไว้จำนวนเท่าใด จากนั้นใช้เพื่อที่จะนับคะแนนเพื่อตรวจสอบว่ามีผู้ชนะในเกมแล้วหรือไม่

## บทที่ 4

### การออกแบบจอภาพ

การออกแบบจอภาพ เป็นส่วนสำคัญของระบบเนื่องจากต้องใช้ในการติดต่อกับผู้เล่นและเป็นผู้ที่ผู้เล่นใช้ในการควบคุมเกมดังนั้นจะต้องออกแบบให้สามารถสื่อสารได้อย่างถูกต้อง และเข้าใจง่ายโดยจะมีการออกแบบดังนี้

#### 4.1 จอเมนูเริ่มต้น



รูปที่ 4.1 หน้าจอหลักของการเล่นเกม

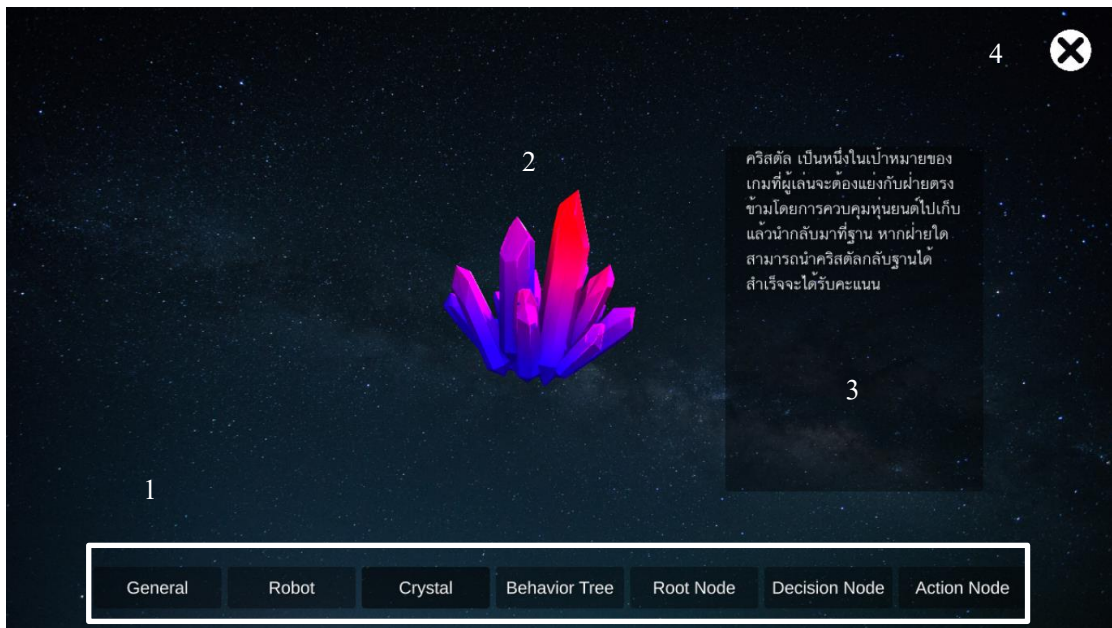
จากรูปที่ 4.1 อธิบายรายละเอียดของจอภาพได้ ดังนี้

ส่วนที่ 1 ปุ่มสำหรับเข้าเล่นเกม ซึ่งปุ่มนี้จะพาผู้เล่นไปยัง จอเลือกด่าน

ส่วนที่ 2 ปุ่มสำหรับไปยังหน้าแนะนำวิธีการเล่นเกมซึ่งปุ่มนี้จะพาผู้เล่นไปยัง จอแนะนำวิธีการเล่น

ส่วนที่ 3 ปุ่มสำหรับปิดเกม เมื่อผู้เล่นกดปุ่มนี้เกมจะปิดลงทันที

## 4.2 จอแนะนำวิธีการเล่น



รูปที่ 4.2 หน้าจอแนะนำวิธีการเล่นเกม

จากรูปที่ 4.2 อธิบายรายละเอียดของจอภาพได้ ดังนี้

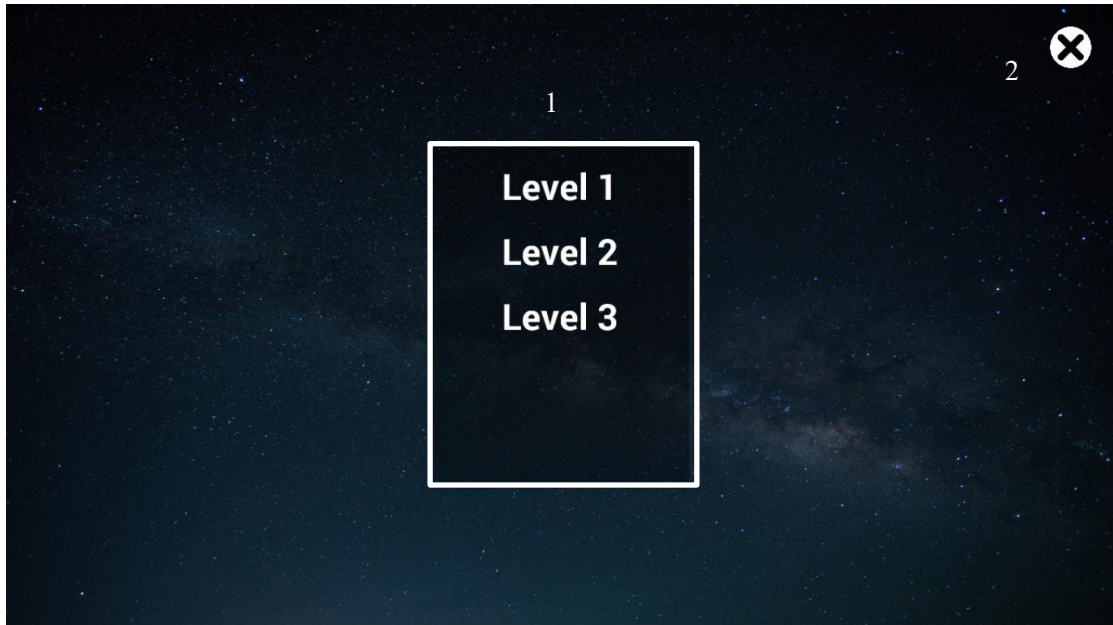
ส่วนที่ 1 กลุ่มของปุ่มในส่วนที่ 2 เมื่อกดไปยังปุ่มเหล่านี้ จอภาพจะทำการไปอัพเดทข้อมูลในส่วนที่ 2 และ 3 ให้ตรงกับข้อมูลของปุ่มที่ผู้เล่นได้กดเลือก

ส่วนที่ 2 แสดงรูปภาพที่ผู้เล่นได้เลือกในส่วนที่ 1

ส่วนที่ 3 กล่องข้อความสำหรับบรรยายข้อความแนะนำ เกี่ยวกับสิ่งที่ผู้เล่นเลือกในส่วนที่ 1

ส่วนที่ 4 ปุ่มปิด จะพาผู้เล่นกลับไปยัง จอหน้าเมนูเริ่มต้น

### 4.3 จอเลือกด่าน



รูปที่ 4.3 หน้าจอเลือกด่าน

จากรูปที่ 4.3 อธิบายรายละเอียดของจอภาพได้ ดังนี้

ส่วนที่ 1 กลุ่มของปุ่มในส่วนนี้จะเป็นรายการของด่านที่ผู้เล่นสามารถเล่นได้ เมื่อกดเลือกจะนำผู้เล่นไปยังหน้าจอหลักการเล่น เกม ของด่านนั้นทันที

ส่วนที่ 2 ปุ่มกลับ จะพาผู้เล่นกลับสู่หน้าจอเมนูเริ่มต้น

#### 4.4 จอการเล่นเกม



รูปที่ 4.4 หน้าจอหลักของการเล่นเกม

จากรูปที่ 4.4 อธิบายรายละเอียดของจอภาพได้ ดังนี้

- ส่วนที่ 1 ปุ่มสำหรับเริ่มเกม เมื่อเลือกหุ่นยนต์จะทำตามต้นไม้พฤติกรรมที่ผู้เล่นสร้างไว้
- ส่วนที่ 2 ปุ่มสำหรับเปลี่ยนความเร็วเวลาสามารถเลือกได้ 3 ระดับคือ ปกติ สองเท่า และสี่เท่า
- ส่วนที่ 3 กล่องข้อความสำหรับบอกคะแนนและความเร็วของเกมที่ใช้อยู่
- ส่วนที่ 4 หุ่นยนต์ของทีมผู้เล่นเมื่อกดจะพาผู้เล่นเข้าสู่หน้าจอแก้ไขต้นไม้พฤติกรรม
- ส่วนที่ 5 จุดกึ่งกลางมีไว้เป็นเส้นแบ่งเสมือน เขตแดนของทั้งสองฝ่ายและเป็นจุดกำเนิดคริสตัล
- ส่วนที่ 6 หุ่นยนต์ของทีมตรงข้าม ใช้ออกตำแหน่งของทีมตรงข้ามว่ามีการเรียงตัวกันอย่างไร
- ส่วนที่ 7 ปุ่มปิด ใช้สำหรับออกจากหน้าจอเกม เพื่อกลับสู่เมนูเริ่มต้น

#### 4.5 จอสรุปผลการเล่นเกม



รูปที่ 4.5 หน้าจอสรุปผลการเล่นเกม

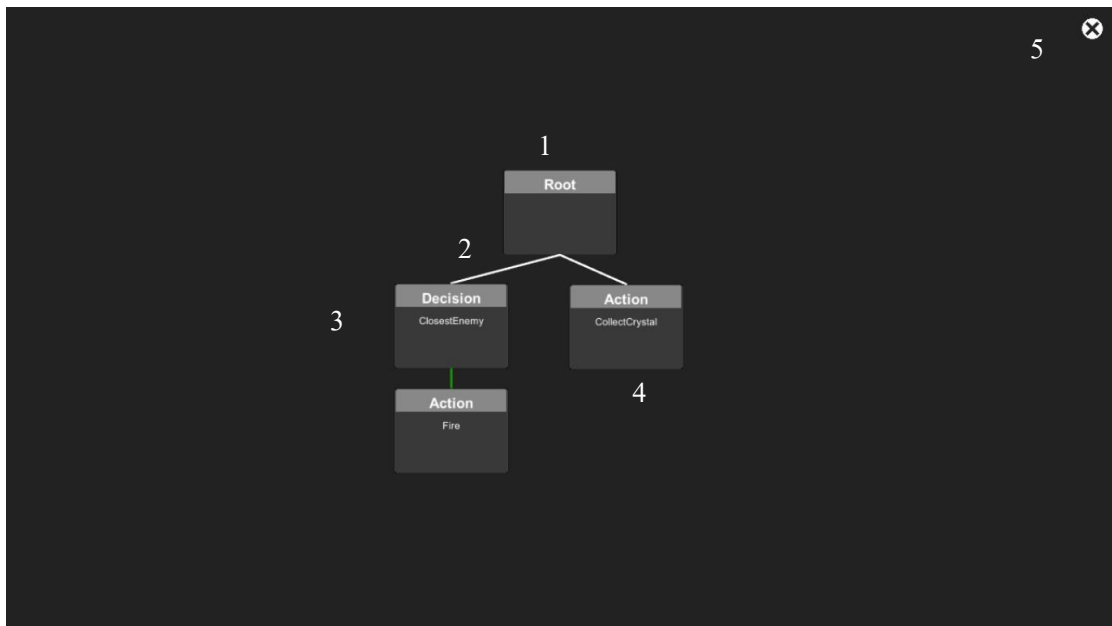
จากรูปที่ 4.5 อธิบายรายละเอียดของจอภาพได้ ดังนี้

ส่วนที่ 1 ข้อความบอกผลเกม

ส่วนที่ 2 ปุ่มสำหรับเล่นด่านนี้อีกครั้ง

ส่วนที่ 3 ปุ่มสำหรับไปยังด่านถัดไป

#### 4.6 จอภาพส่วนประสานต้นไม้พฤติกรรม



รูปที่ 4.6 ภาพส่วนประสานที่ใช้แก้ไขต้นไม้พฤติกรรม

จากรูปที่ 4.6 อธิบายรายละเอียดจอภาพได้ดังนี้

ส่วนที่ 1 โหนดรากโหนดพื้นฐานในหน้าจอแก้ไขต้นไม้พฤติกรรม

ส่วนที่ 2 ปุ่มเส้นเชื่อมใช้บอกว่าโหนดนี้เชื่อมกับโหนดใดอยู่โดยเส้นเชื่อมจะมีทั้งหมด 3 สี ดังนี้

- 1) สีขาว เชื่อมระหว่างโหนดรากกับโหนดใด ๆ
- 2) สีเขียว เชื่อมระหว่างโหนดตัดสินใจกับลูกโหนดตัดสินใจที่เป็นจริง
- 3) สีแดง เชื่อมระหว่างโหนดตัดสินใจกับลูกโหนดตัดสินใจที่เป็นเท็จ

ส่วนที่ 3 โหนดตัดสินใจใช้แสดงให้ผู้เล่นทราบชนิดของโหนดที่ใช้ในการตัดสินใจ

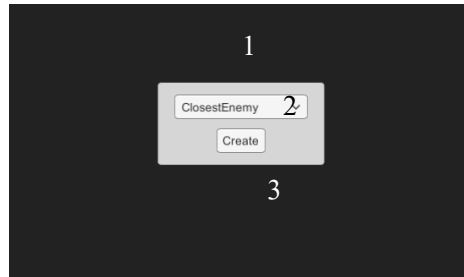
ส่วนที่ 4 โหนดพฤติกรรมใช้แสดงให้ผู้เล่นทราบถึงชนิดของโหนดที่ใช้แสดงพฤติกรรม

ส่วนที่ 5 ปุ่มกลับ จะพาผู้เล่นกลับไปหน้า จอการเล่นเกม

เพิ่มเติม

- 1) โหนดทุกประเภทสามารถย้ายได้ โดยการคลิกเมาส์ค้างที่โหนดแล้วเลื่อนเพื่อย้ายตำแหน่ง
- 2) โหนดทุกประเภทสามารถเลือกได้โดยการคลิกซ้าย 1 ครั้ง และยกเลิกการเลือกด้วยการคลิกขวา
- 3) การคลิกที่พื้นที่ว่างเปล่าจะเป็นการเปิด จอหน้าต่างการสร้างโหนดใหม่

#### 4.7 จอหน้าต่างการสร้างโหนดใหม่



รูปที่ 4.7 ภาพหน้าต่างสร้างโหนดใหม่

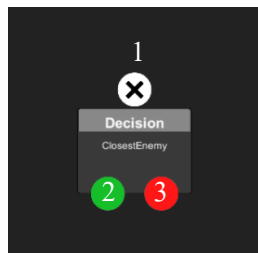
จากรูปที่ 4.7 อธิบายรายละเอียดจอภาพได้ดังนี้

ส่วนที่ 1 หน้าต่างสำหรับสร้างโหนดใหม่

ส่วนที่ 2 ปุ่ม Dropdown สำหรับเลือกโหนดที่ต้องการสร้างใหม่

ส่วนที่ 3 ปุ่มสร้างใช้สำหรับสร้างโหนดที่เลือกไว้ใน Dropdown

#### 4.8 จอเมนูเลือกโหนด



รูปที่ 4.8 ภาพเมนูเลือกโหนด

จากรูปที่ 4.8 อธิบายรายละเอียดจอภาพได้ดังนี้

ส่วนที่ 1 ปุ่มลบสำหรับลบโหนด

ส่วนที่ 2 ปุ่มเชื่อมความสัมพันธ์ ให้กับโหนดลูกที่เป็นจริง

ส่วนที่ 3 ปุ่มเชื่อมความสัมพันธ์ ให้กับโหนดลูกที่เป็นเท็จ

## บทที่ 5

### การพัฒนาระบบ

ในการพัฒนาเกมจำเป็นต้องทราบถึงเครื่องมือและภาษาโปรแกรมที่จำเป็นต้องใช้ในการพัฒนา เพื่อให้เหมาะสมกับลักษณะงานที่กำลังจะพัฒนา ดังนั้น ในบทนี้จะอธิบายเหตุผลว่าเพราะเหตุใดจึงเลือกใช้ภาษาโปรแกรมและเครื่องมือที่ใช้ในการพัฒนาเกมวางแผนกลยุทธ์โดยค้นไม่พบพฤติกรรมและเทคนิคที่ใช้ในการพัฒนาโปรแกรม

#### 5.1 ภาษาและเครื่องมือที่ใช้ในการพัฒนา

##### 5.1.1 ภาษาที่ใช้ในการพัฒนาระบบ

C# เป็นภาษาเขียนโปรแกรมแบบ Multi-paradigm ซึ่งมีรูปแบบภาษาที่ตายตัว และเป็นรูปแบบบังคับในการเขียน มีฟังก์ชัน และยังเป็นภาษาการเขียนโปรแกรมที่มีคุณสมบัติเป็นแบบออบเจกต์ด้วย ซึ่งถูกพัฒนาโดยบริษัท Microsoft ภายใต้อิง .NET framework เนื่องจากภาษา C# เป็นภาษาที่สามารถเขียนโปรแกรมเชิงวัตถุที่ Unity แนะนำให้เป็นภาษาสำหรับพัฒนาเกม

##### 5.1.2 เครื่องมือที่ใช้ในการพัฒนาระบบ

Unity 3D คือ Game Engine ที่ช่วยสร้างเกม 3 มิติ ซึ่งสามารถทำงานได้ 2 แพลตฟอร์ม คือ Windows และ OSX อีกทั้งยังสามารถนำเกมที่พัฒนาเสร็จไปใช้งานได้หลายแพลตฟอร์ม เช่น Windows, OSX, Androids, iOS เป็นต้น

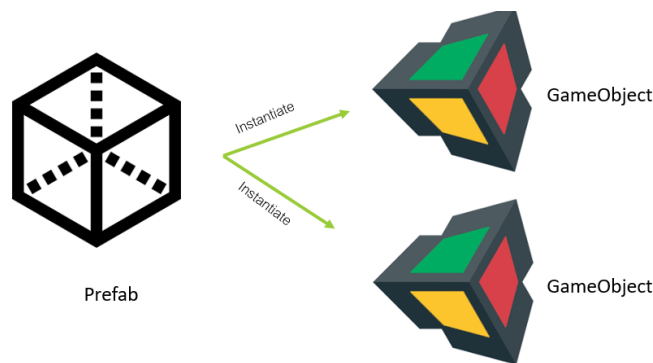
Visual Studio 2017 Community เป็นโปรแกรมประเภท IDE ที่ใช้ในการแก้ไขและปรับแต่งโค้ดจากไมโครซอฟท์ ได้ฟรี โดยจะต้องลงทะเบียนผ่านเว็บก่อน ซึ่งการพัฒนาภาษา C# ใน Unity บน Windows นั้นมีการแนะนำให้ลง Visual Studio เพื่อความสะดวกและเข้ากันได้กับ Unity อีกทั้ง Visual Studio ยังมีเครื่องมือต่าง ๆ ที่สามารถโหลดเพิ่มเติมได้และสามารถ Debug ร่วมกับ Unity ได้ทันที โดยไม่ต้องลงส่วนเสริม

Source Tree โปรแกรมจัดการกับ Git ผ่านส่วนประสานที่สวยงามและใช้งานง่าย ซึ่งจะทำงานร่วมกับ Github หรือจะใช้ Bitbucket ที่เป็น Git Repository Service ของ SourceTree เองได้ และมีความสามารถในการแก้ไขข้อขัดแย้งระหว่างโค้ด

## 5.2 เทคนิคที่ใช้ในการพัฒนา

### 5.2.1 Prefab & Instantiate

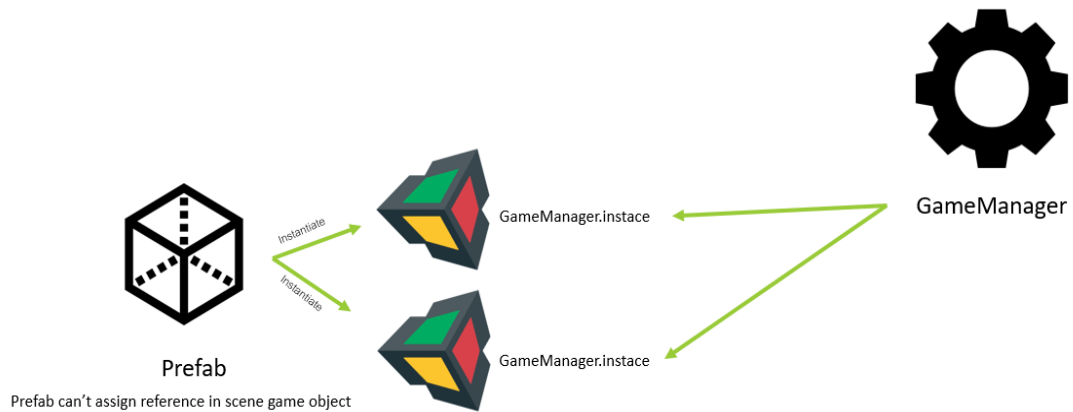
Prefab คือการเก็บ Object ในรูปแบบก้อนของข้อมูลที่จัดการโดย Unity ทำให้สามารถสร้างวัตถุนั้นได้เรื่อย ๆ แต่การที่จะทำให้ Object นั้นแยกตัวกันจะต้องใช้วิธีการ Instantiate เป็นการ Clone Object เพื่อที่จะใช้งานร่วมกับเทคนิค Singleton Object ต่อไป



รูปที่ 5.1 การใช้ Prefab มาสร้าง GameObject

### 5.2.2 Singleton Pattern

ในการสร้าง Node เนื่องจาก Unity ไม่แนะนำให้เก็บ Object ที่ต้องเรียกใช้งานบ่อยไว้ใน GameObject เนื่องจากจะเกิดจากเสียหายได้จากการเปลี่ยนหน้า ดังนั้นจะสร้าง ScriptableObject ขึ้นมาเพื่อเก็บ Node แล้วแยกส่วนของ NodeUI เมื่อต้องการใช้งานจะทำการแมพ Object ทั้งสองชนิดเข้าด้วยกันแล้วเรียกว่า Singleton โดยจะทำให้เมื่อเราเปลี่ยน Object ใหม่แล้วจะทำให้ Object ที่ Map นั้นเปลี่ยนไปด้วย



รูปที่ 5.2 การใช้ Singleton Pattern

จากรูปที่ 5.2 เมื่อใช้ Singleton Pattern จะทำให้ Object ใด ๆ ใน Namespace ที่เรียกใช้สามารถเข้าถึง Object ที่เรียกได้โดยวิธีนี้เป็นลดการผูก Object ไว้กับ GameObject อีกด้วย

### 5.2.3 Weak Reference

การอ้างอิงถึง Object เนื่องจาก C# ปกติการอ้างอิงจะเป็นการ Pass by Reference ทำให้เมื่อเราเปลี่ยน Object นั้นจะทำการเปลี่ยน Object ต้นฉบับด้วย หาก Object ถูกทำลายจะทำให้ไม่สามารถอ้างอิงถึง Object นั้นได้อีกทำให้โปรแกรมเกิดข้อผิดพลาดที่ไม่สามารถทำงานต่อได้ จึงต้องทำการอ้างอิงผ่าน Weak Reference เพื่อที่จะแก้ไขปัญหานี้

## 5.3 การพัฒนาระบบ

### 5.3.1 การแสดงออกของหุ่นยนต์

เมื่อเริ่มพัฒนาระบบผู้พัฒนาทำการพัฒนาทุกการกระทำของหุ่นยนต์ ที่ทุกการกระทำต้องเปลี่ยนไปยังอีกการกระทำได้โดยไม่มีติดขัด เช่น หุ่นยนต์สามารถ เดิน แล้วสามารถเปลี่ยนไปยัง และเปลี่ยนเป็นพฤติกรรมอื่น ๆ ได้

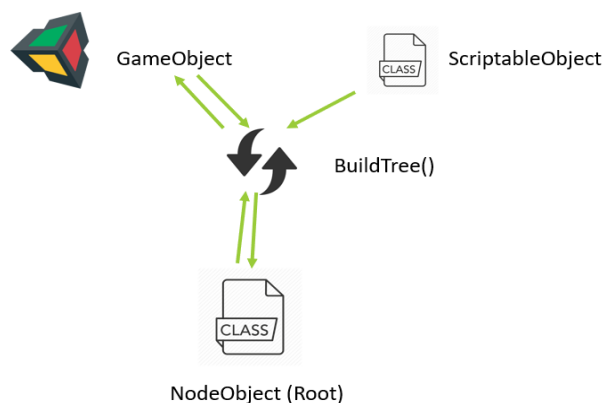
### 5.3.2 ต้นไม้พฤติกรรม

พัฒนาโดยเขียนโน้ตแต่ละรูปแบบขึ้นมาแล้วทดสอบการทำงานว่าสามารถเลือกพฤติกรรมได้ถูกต้องหรือไม่ จากนั้นปรับมาใช้เทคนิค Prefab และ Instantiate เนื่องจาก

Node จะต้องมีการสร้างใหม่ตามหุ่นยนต์และตามที่ใช้รูปแบบ ซึ่งจะทำให้สะดวกกว่าการมาเรียกสร้าง Object โดยการ New Object ขึ้นมาใหม่

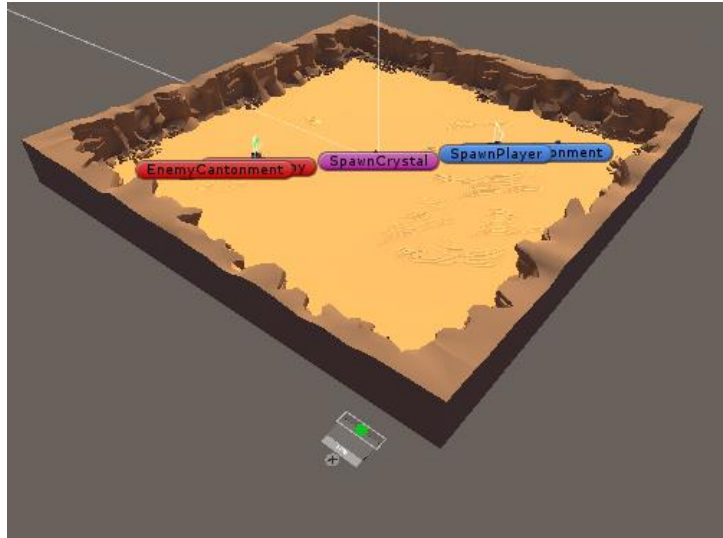
### 5.3.3 โหนดของต้นไม้พฤติกรรมที่แสดงผลและโหนดที่ทำงานในหุ่นยนต์

เนื่องจากโหนดที่แสดงผลในเกมเป็นโหนดแบบ GameObject ซึ่งไม่เหมาะสำหรับการเก็บข้อมูล หากเก็บจะต้องย้าย Object ไปไว้ไกลๆ ซึ่งการทำเช่นนั้นจะทำให้เกมประมวลผลมากขึ้นเนื่องจาก ต้องนำต้นไม้พฤติกรรมเหล่านี้ไปไว้ในที่ ๆ มองไม่เห็น แต่การมีอยู่ของกลุ่ม Object เหล่านี้ส่งผลต่อการเรียกเมธอด Update และ FixedUpdate ของ Unity ซึ่งอาจจะส่งผลเสียต่อความเร็วของระบบได้ ดังนั้นจึงใช้วิธีการแปลงต้นไม้พฤติกรรมที่ใช้สำหรับควบคุมหุ่นยนต์ และต้นไม้พฤติกรรมที่ใช้แสดงผลให้ แปรลงย้อนกลับไป



รูปที่ 5.3 การแปลงโหนดที่ใช้ควบคุมและโหนดที่ใช้แสดงผล

จากรูปที่ 5.3 เมื่อทำการเปิดหน้าต่างต้นไม้พฤติกรรมจะทำการสร้างต้นไม้พฤติกรรมที่แสดงผลผ่านหน้าจอส่วนต่อประสานขึ้นมาใหม่ โดยใช้กล่องตัวที่ 2 ในการฉายภาพและเมื่อทำการออกจากหน้าต่างดังกล่าว จะทำการแปลงโหนดที่ใช้แสดงผลกลับมาเป็นโหนดที่ใช้ควบคุมหุ่นยนต์และนำไปตั้งค่าให้กับหุ่นยนต์ ดังรูปที่ 5.4



รูปที่ 5.4 การวางตำแหน่งของส่วนแก้ไขต้นไม้พฤติกรรม

#### 5.3.4 การส่งข้อมูลระหว่าง GameObject

ดังเดิมแล้ว Unity นั้นสามารถเรียกค้นหา GameObject ได้ผ่าน Tag หรือการ Reference ผ่านตัวแปรแบบ Public แต่ในกรณีที่มีการเลือกแล้วจะให้ระบบจำได้ว่าผู้เล่นเคยเลือก Object นั้นจะต้องนำตัวแปรเหล่านั้นไปฝากไว้กับ Object ซึ่งการทำเช่นนั้นจะทำให้ไม่สามารถสร้าง Object โดยใช้ Prefab ได้เนื่องจาก Prefab Object จะไม่สามารถเก็บค่าที่ Reference กับ Object ที่อยู่ใน Scene ได้ดังนั้นจะต้องใช้ Object หนึ่งสร้างคลาสที่ชื่อว่า GameManager ขึ้นมาเพื่อฝากตัวแปรชั่วคราว จากนั้นเมื่อเริ่มต้นจะต้องทำการเพิ่มค่าอ้างอิงให้กับ GameObject ที่สร้างผ่านเทคนิค Prefab & Instantiate เมื่อ GameObject เริ่มทำงานจะเรียกเมธอด Start เพื่อตั้งค่าตัวแปรอ้างอิงให้กับหุ่นยนต์ตัวนั้น ๆ โดยใช้เทคนิค Singleton Pattern

#### 5.3.5 การเก็บข้อมูลชั่วคราว

เมื่อมีการเก็บข้อมูลผ่าน GameManager โดยหุ่นยนต์แต่ละตัวมีโอกาที่จะถูกทำลายหรือสร้างใหม่ได้ ดังนั้นเมื่อมีการเรียกใช้ Object ที่ไม่มีอยู่จะทำให้เกมไม่สามารถดำเนินต่อไปได้ จึงจะต้องสร้าง Weak Reference ขึ้นมาเพื่อเก็บตัวแปรเหล่านั้นแทนการที่จะอ้างอิงถึงโดยตรง ซึ่งการทำในรูปแบบนี้จะทำให้ผู้พัฒนาทราบถึงการเปลี่ยนแปลงที่อยู่ใน Weak Reference ที่อ้างอิงถึงได้

## บทที่ 6

### ผลสรุป

การทดสอบระบบ เป็นขั้นตอนสำคัญเพื่อให้ทราบว่าระบบที่พัฒนานั้นมีข้อบกพร่องหรือข้อจำกัดใด ซึ่งในบทนี้จะแสดงรายละเอียดของการทดสอบ รายงานปัญหา ข้อสรุปและข้อเสนอแนะ ดังนี้

#### 6.1 ผลการทดลองโปรแกรม

- 1) เกมสามารถสร้างต้นไม้พฤติกรรมให้กับตัวละครได้
- 2) ผู้เล่นแต่ละตัวสามารถแสดงพฤติกรรม ตามต้นไม้พฤติกรรมของตัวเองได้
- 3) เกมสามารถปรับปรุงต้นไม้พฤติกรรมได้
- 4) เกมสามารถเพิ่มความเร็วในการแสดงพฤติกรรมของตัวละครได้

#### 6.2 ปัญหาที่พบ

- 1) เกมที่พัฒนามีความซับซ้อนสูงมาก จึงจำเป็นต้องใช้เวลาในการศึกษาและพัฒนา
- 2) เกมที่พัฒนานั้นมีความแปลกใหม่มาก จึงไม่สามารถหาข้อมูลได้ จำเป็นต้องอาศัยการลองผิดลองถูก ในการพัฒนาทำให้ใช้เวลานานกว่าปกติ และแก้ไขหลายครั้ง

#### 6.3 ข้อสรุปและข้อเสนอแนะ

ข้อสรุป เกมที่พัฒนาสามารถสร้างต้นไม้พฤติกรรม เพื่อควบคุมพฤติกรรมของตัวหุ่นยนต์ได้ และสามารถแก้ไขปรับแต่งต้นไม้เพื่อเปลี่ยนแผนการต่อสู้ได้ สามารถแสดงต้นไม้พฤติกรรมผ่านส่วนต่อประสานได้

ข้อเสนอแนะ ปรับแต่งหน้าจอแสดงผลโหนด และทำเอฟเฟกเพิ่มเติมในส่วนของการสื่อสารกับผู้เล่น แสดงว่าเกิดกิจกรรมใด ๆ ในหน้าโหนดเช่น เพิ่มโหนด A ให้เป็นลูกของโหนด B ตัดความสัมพันธ์ของโหนด A จากโหนด B เป็นต้น เพิ่มโหนดเงื่อนไข และโหนดการกระทำอื่น

## เอกสารอ้างอิง

- [1] scikit-learn, “1. 10. Decision Trees,” 2556. [ออนไลน์]. Available: <http://scikit-learn.org/stable/modules/tree.html>. [วันที่เข้าถึง 12 ธันวาคม 2560].
- [2] Nuck, “[Unity3D] – Software Review : Unity3D & Qualcomm AR,” 2557. [ออนไลน์]. Available: <http://www.codenuke.net/2014/01/software-review-unity3d-qualcomm-ar.html>. [วันที่เข้าถึง 10 พฤศจิกายน 2560].
- [3] Unity Technologies, “Execution Order of Event Functions,” Unity Technologies, [ออนไลน์]. Available: <https://docs.unity3d.com/Manual/ExecutionOrder.html>. [วันที่เข้าถึง 30 เมษายน 2561].
- [4] imdb, “Command & Conquer: Red Alert 2,” 2543. [ออนไลน์]. Available: <https://www.imdb.com/title/tt0252338>. [วันที่เข้าถึง 24 มีนาคม 2560].
- [5] maxoja, “เ อ พ " ห ม า ก ก ร ะ ด า น " ,” 2557. [ออนไลน์]. Available: <http://unitystarterthailand.blogspot.com/2014/03/google-play-httpyoutu.html>. [วันที่เข้าถึง 12 มีนาคม 2560].
- [6] C. S. Education, “Lightbot,” 2558. [ออนไลน์]. Available: <http://lightbot.com/hour-of-code.html>. [วันที่เข้าถึง 16 มิถุนายน 2560].
- [7] C. Combat, “About | CodeCombat,” 2558. [ออนไลน์]. Available: <https://codecombat.com/about#story>. [วันที่เข้าถึง 3 มกราคม 2561].
- [8] ป. มั่นจิต, “หลักการเขียน Use Case Diagram,” 2548. [ออนไลน์]. Available: <http://howtomakediagram.blogspot.com/2015/03/use-case-diagram.html>. [วันที่เข้าถึง 2 มกราคม 2561].

- [9] tutorialspoint, “UML - Activity Diagrams,” 2558. [ออนไลน์]. Available: [https://www.tutorialspoint.com/uml/uml\\_activity\\_diagram.htm](https://www.tutorialspoint.com/uml/uml_activity_diagram.htm). [วันที่เข้าถึง 9 พฤษภาคม 2560].
- [10] Y. Panyawai, “Sequence Diagram,” 2558. [ออนไลน์]. Available: <http://sequencediagram-it56.blogspot.com>. [วันที่เข้าถึง 9 เมษายน 2560].
- [11] visual-paradigm, “UML Class Diagram Tutorial,” 2558. [ออนไลน์]. Available: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/>. [วันที่เข้าถึง 14 กรกฎาคม 2560].

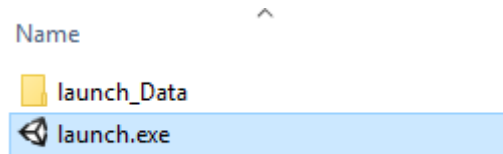
## ภาคผนวก

### ภาคผนวก ก คู่มือการติดตั้งเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) เปิดซีดีของ เกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม
- 2) ถัดลอกแฟ้มข้อมูลชื่อ Release ลงในเครื่อง
- 3) เมื่อเปิดแฟ้มข้อมูลจะพบกับไฟล์ชื่อ launch.exe
- 4) ดับเบิลคลิกไฟล์ launch.exe จะพบหน้าต่างตั้งค่าง่อนเริ่มเกม
- 5) กดปุ่ม Play !

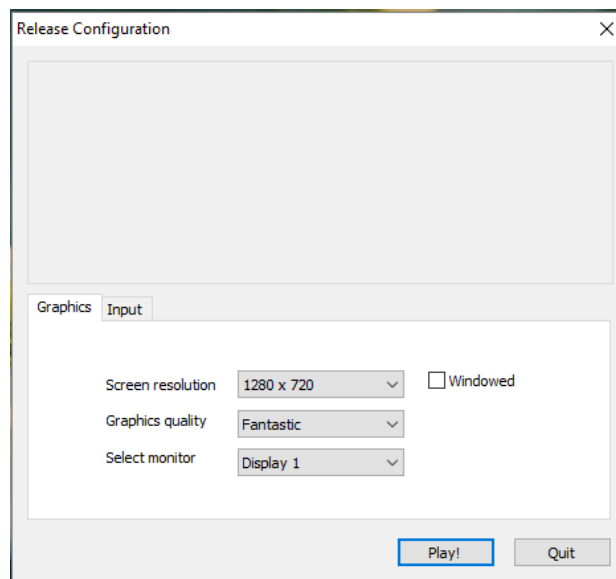
ภาคผนวก ข คู่มือการเล่นเกมวางแผนกลยุทธ์โดยใช้ต้นไม้พฤติกรรม

- 1) เปิดเกมผ่านตัวเล่นชื่อ launch.exe



รูปที่ ข.1 ตัวเล่นเกม

- 2) จะพบหน้าต่างดังรูปที่ ข.2 ให้เลือก Play



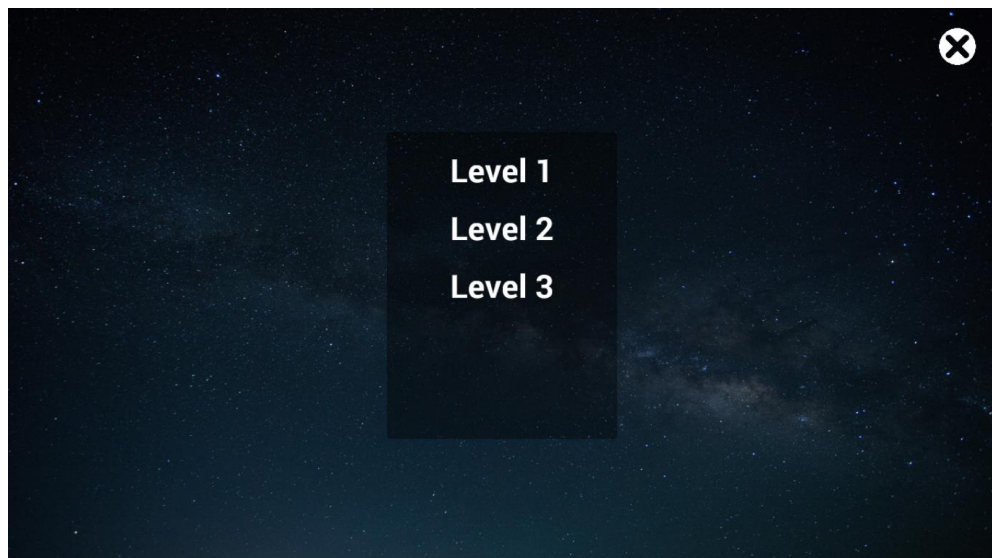
รูปที่ ข.2 หน้าต่างปรับตั้งค่า

- 3) เมื่อเปิดเกมได้แล้วจะพบกลับหน้าจอเมนูเริ่ม



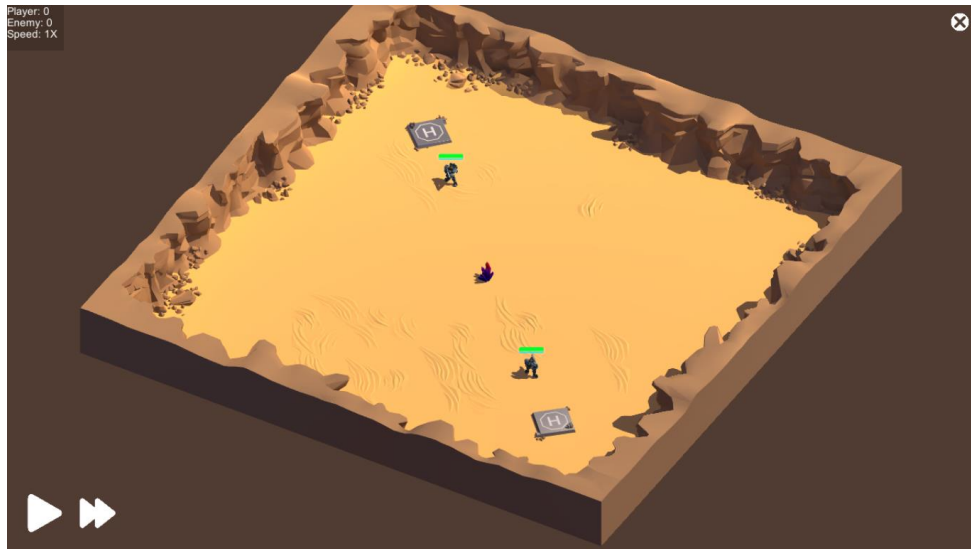
รูปที่ ข.3 หน้าจอเมนู

- 4) ให้ผู้เล่นเลือกปุ่ม Play จากนั้นเกมจะนำผู้เล่นเข้าสู่หน้าเลือกด่าน



รูปที่ ข.4 หน้าเลือกด่าน

- 5) ผู้เล่นเลือกด่านที่ต้องการเล่นจากนั้นจะเข้าสู่หน้าเกม



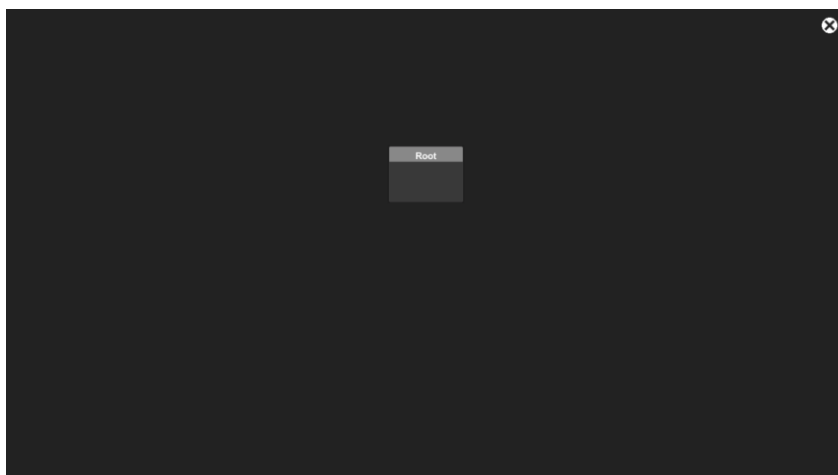
รูปที่ ข.5 หน้าจอเริ่มเกม

- 6) จากนั้นให้คลิกเลือกที่หุ่นยนต์ที่ต้องการจะสร้างต้นไม้พฤติกรรม



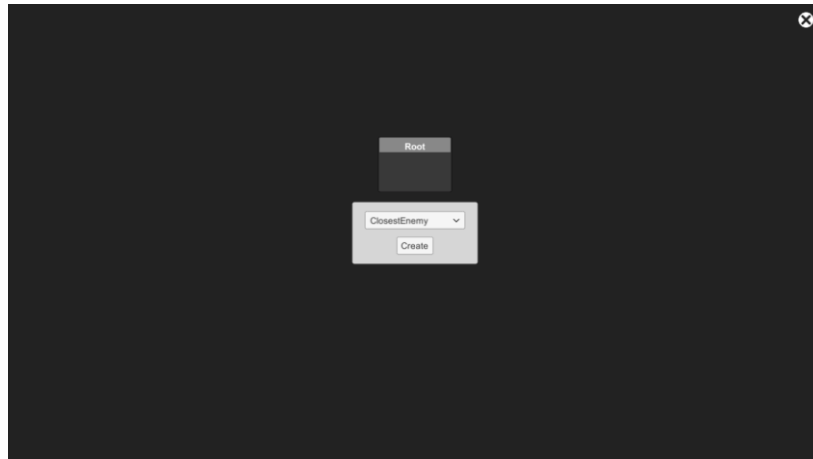
รูปที่ ข.6 หุ่นยนต์ของทีมผู้เล่น

- 7) เกมจะพาผู้เล่นเข้าสู่หน้าปรับแต่งแก้ไขต้นไม้พฤติกรรม



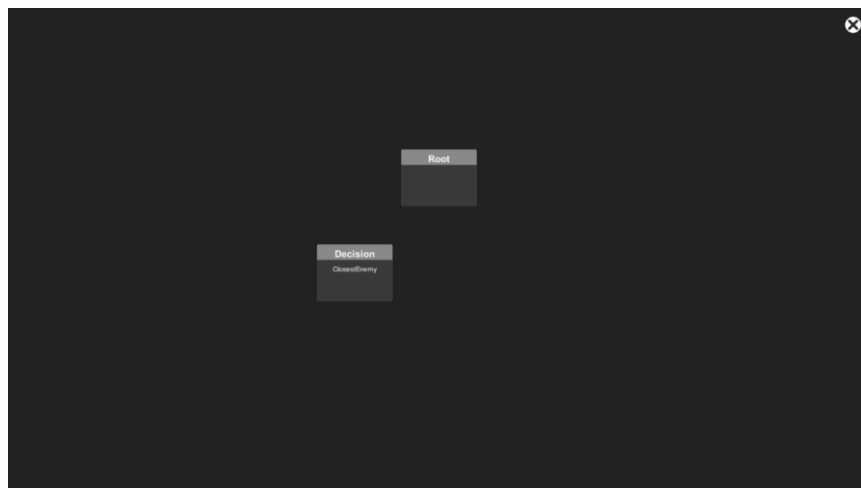
รูปที่ ข.7 หน้าจอแก้ไขต้นไม้พฤติกรรม

- 8) คลิกขวาที่พื้นที่ว่างเพื่อเปิดหน้าต่างเลือกสร้างโหนด



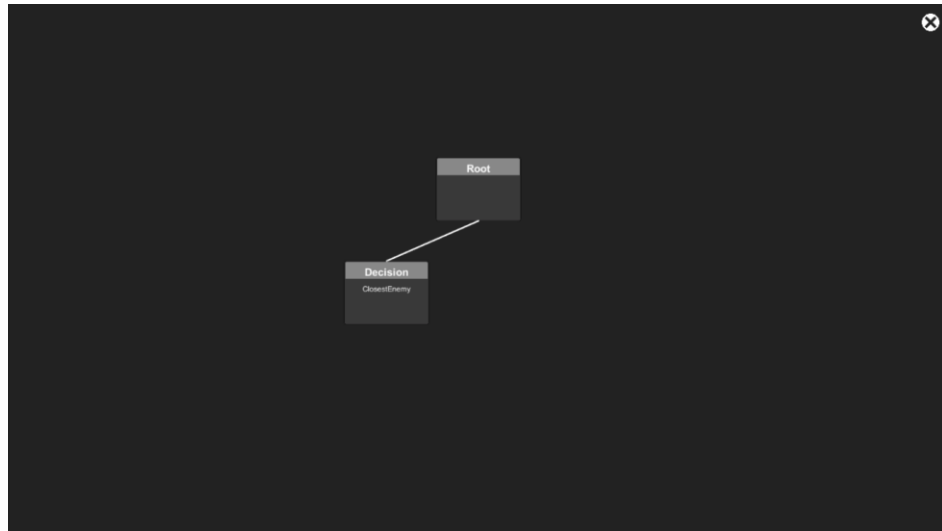
รูปที่ ข.8 หน้าต่างเลือกสร้างโหนด

- 9) เลือก ClosestEnemy จากนั้นเลือก Create



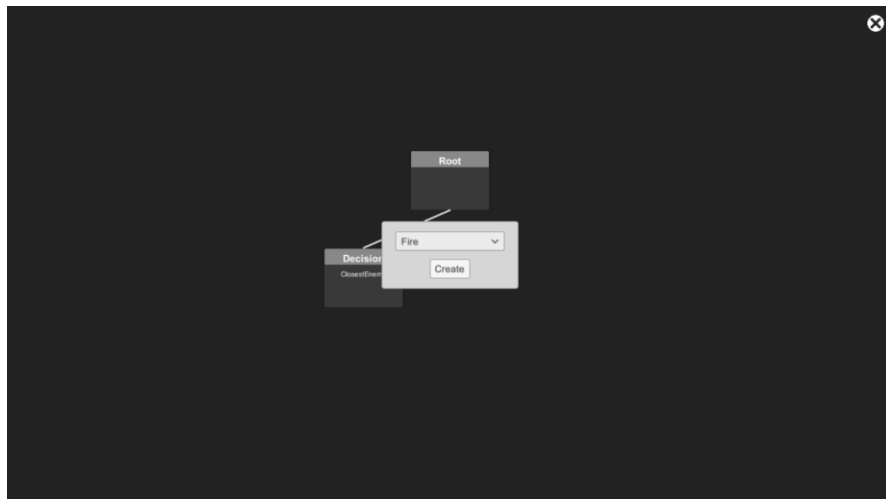
รูปที่ ข.9 หน้าจอหลังจากสร้างโหนด

- 10) จากนั้นเชื่อมโหนดรากเข้ากับโหนดตัดสินใจที่สร้างขึ้น โดยการคลิกที่โหนดราก จากนั้นคลิกที่ปุ่มสีเขียว แล้วเลือกโหนดตัดสินใจที่สร้างขึ้นมาใหม่เพื่อเชื่อม

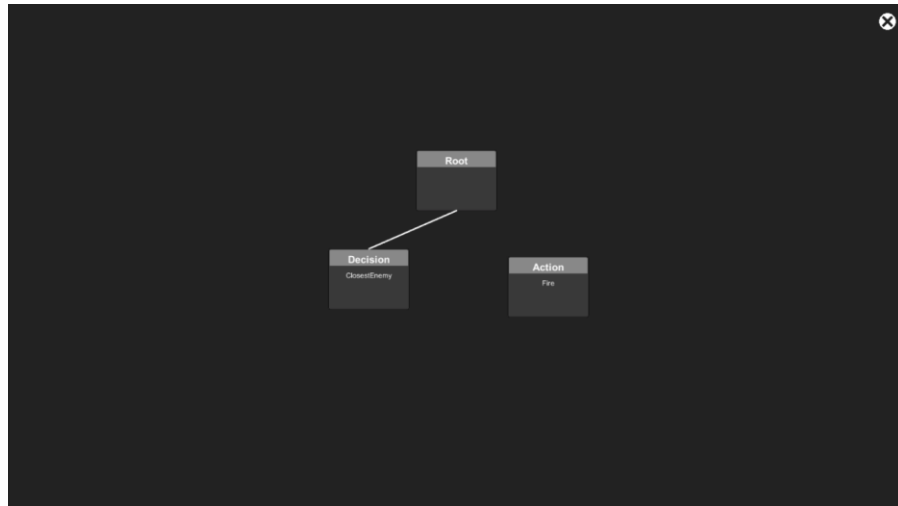


รูปที่ ข.10 หน้าจอเมื่อเชื่อมโหนดเข้ากับโหนดราก

- 11) จากนั้นทำการสร้างโหนดยังเพิ่ม โดยทำการคลิกขวาแล้ว เลือก Dropdown เป็น Fire จากนั้นกด Create

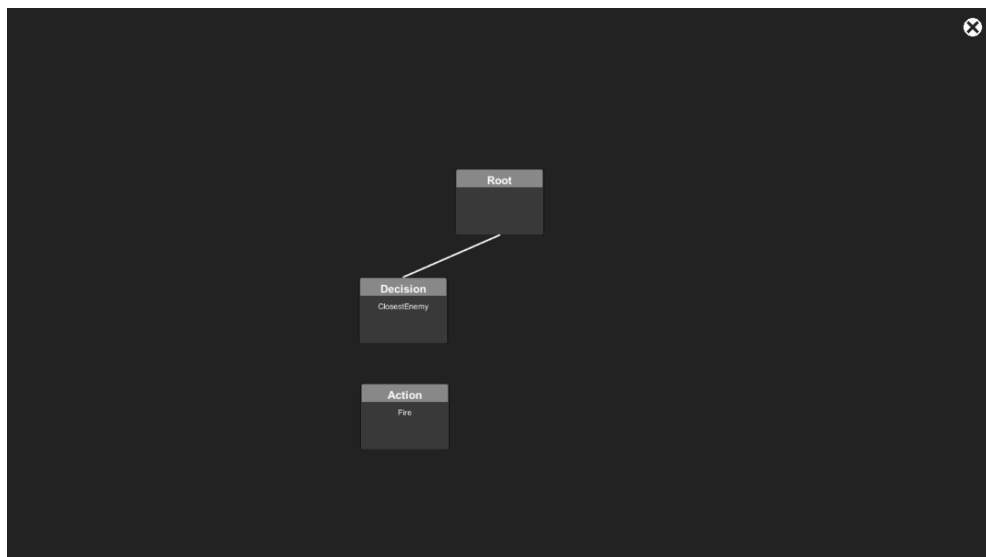


รูปที่ ข.11 หน้าจอเมื่อสร้างโหนดเพิ่ม



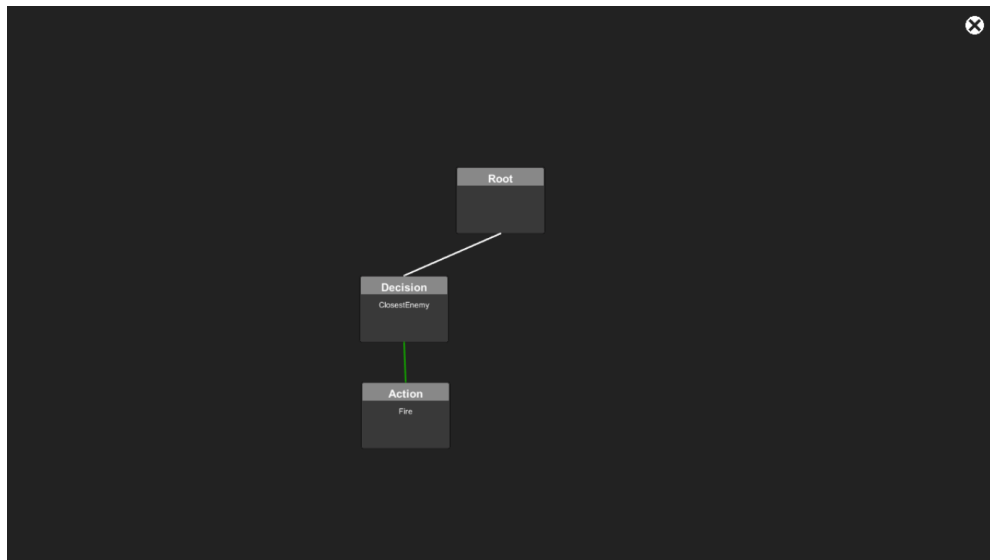
รูปที่ ข.12 หน้าจอเมื่อสร้างโหนดยังเพิ่ม

- 12) ทำการย้ายโหนดยังไปได้ด้านล่างของโหนด ClosestEnemy โดยการคลิกที่โหนด Fire ค้างไว้ แล้วเลือกเมาส์ไปยังตำแหน่งใหม่ที่ต้องการ



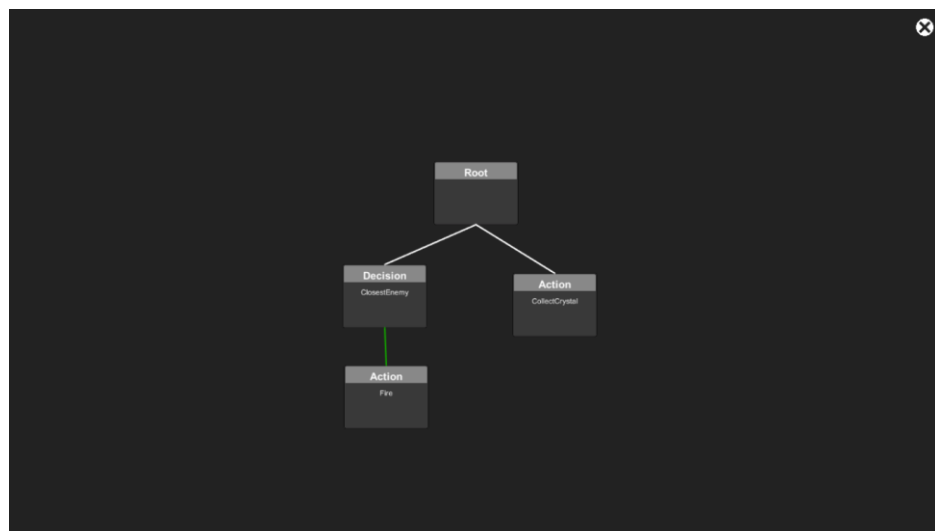
รูปที่ ข.13 หน้าจอเมื่อย้ายโหนดยัง

- 13) ทำการเชื่อมโหนดเงื่อนไขจริงของ ClosestEnemy เข้ากับโหนด Fire โดยการคลิกที่โหนด ClosestEnemy จากนั้นจะปรากฏปุ่มสี่เหลี่ยม และแดง ให้เลือกปุ่มสี่เหลี่ยม แล้วคลิกเลือกที่โหนด Fire จะปรากฏเส้นเชื่อมสี่เหลี่ยมขึ้นมา



รูปที่ ข.14 หน้าจอเมื่อเชื่อมโหนด Fire กับ ClosestEnemy

- 14) จากนั้นสร้างโหนด CollectCrystal เพิ่มแล้วเชื่อมเข้ากับโหนดราก โดยที่โหนด CollectCrystal จะต้องอยู่ถัดจากโหนด ClosestEnemy ดังรูปที่ ข.15



รูปที่ ข.15 หน้าจอเมื่อสร้างต้นไม้ตัดสินใจเสร็จ

- 15) ปิดหน้าต่างแก้ไขต้นไม้ตัดสินใจโดยปุ่ม X มุมขวาบน จากนั้นจะพาผู้เล่นกลับสู่ หน้าจอเกมอีกครั้ง ให้ผู้เล่นกดปุ่มสัญลักษณ์ Play มุมซ้ายล่าง จากนั้นหุ่นยนต์จะแสดงพฤติกรรมตามต้นไม้พฤติกรรมที่สร้างไว้



รูปที่ ข.16 หน้าจอขณะเกมกำลังเล่น

- 16) เมื่อเกมจบลงจะมีหน้าจอสรุปผลขึ้นดังภาพ ผู้เล่นสามารถเลือกสัญลักษณ์ถัดไปเพื่อเล่นด่านใหม่ หรือสัญลักษณ์วนซ้ำเพื่อเล่นด่านเดิมอีกครั้ง



รูปที่ ข.17 หน้าจอสรุปผลการเล่น