รายงานผลโครงการและงานวิจัย

เรื่อง

เกม PLANT & FLEE

นายภุชิสสะ แซ่ลี้

นายศุภกิตติ์ แพเมือง

โครงการนี้เป็นส่วนหนึ่งของวิชาโครงการ รหัส 30128-8501

หลักสูตร ประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.)

สาขาวิชาเทคโนโลยีคอมพิวเตอร์

ภาคเรียนที่ 2 ปีการศึกษา 2566

วิทยาลัยเทคนิคจันทบุรี สถาบันกการอาชีวศึกษาภาคตะวันออก

ใบรับรองอนุปริญญานิพนธ์
สาขาเทคโนโลยีคอมพิวเตอร์ วิทยาลัยเทคนิคจันทบุรี

หัวข้อวิจัย	:	เกม PLANT & FLEE
ผู้ดำเนินการวิจัย	:	นายภุชิสสะ แซ่ลี้
		นายศุภกิตติ์ แพเมือง
ที่ปรึกษา	:	นายฉัตรพฤกษ์ สีทิม
		นางสาวประภาพร บันเทา
สาขาวิชา	:	เทคโนโลยีคอมพิวเตอร์
สาขางาน	:	คอมพิวเตอร์ระบบเครือข่าย
ปี พ.ศ.	:	2566

สาขาวิชาเทคโนโลยีคอมพิวเตอร์ ให้นับอนุปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรประกาศนียบัตรชีพขั้นสูง สาขาวิชาเทคโนโลยีคอมพิวเตอร์

.....หัวหน้าสาขาวิชาเทคโนโลยีคอมพิวเตอร์
(นายศัตราวุธ ศรีชาติ)

คณะกรรมการสอบโครงการ

.....ประธานกรรมการ
(นายฉัตรพฤกษ์ สีทิม)

.....กรรมการ
(นายปิยพงษ์ เกตุภูงา)

.....กรรมการ
(นางสาวประภาพร บันเทา)

หัวข้อวิจัย เกม PLAN & FLEE
ผู้ดำเนินการวิจัย นายภุชิสสะ แซ่ลี้
นายศุภกิตติ์ แพเมือง
ที่ปรึกษา นายฉัตรพฤกษ์ สีทิม
นางสาวประภาพร บันเทา
สาขาวิชา เทคโนโลยีคอมพิวเตอร์
สาขางาน คอมพิวเตอร์ระบบเครือข่าย
ปีการศึกษา 2566

บทคัดย่อ

การทำโครงการเกม PLAN & FLEE ครั้งนี้มีวัตถุประสงค์เพื่อ 1. เพื่อสร้างเกม PLAN & FLEE 2. เพื่อศึกษาการใช้โปรแกรม Unity ,Aseprite ,Pro Motion NG และ ภาษา C# 3. เพื่อความพึงพอใจของผู้เล่นเกม PLAN & FLEE 4. เพื่อหาประสิทธิภาพของเกม PLAN & FLEE

จากการดำเนินโครงการและทดสอบโครงการ ได้ผลว่า การประเมินความพึงพอใจของผู้เล่นเกมมีค่ามากสุดในด้านที่เกม PLANT & FLEE คือ ออกแบบสวยงาม โดยมีค่าเฉลี่ย $\bar{x}$ เท่ากับ 4.7 อยู่ในระดับดีมาก และมีค่าน้อยสุดในด้านที่เกม PLANT & FLEE คือ ความเหมาะสมของเวลาในการเล่น โดยมีค่าเฉลี่ย $\bar{x}$ เท่ากับ 4.2 อยู่ในระดับดี โดยมีค่าเฉลี่ยความพึงพอใจรวมทุกด้าน $\bar{x}$ เท่ากับ 4.54 อยู่ในระดับดีมาก และมีค่า $S.D.$ เท่ากับ 0.56

Research Title	Game PLANT & FLEE	
Research	Mr. Phuchitsa	Saeli
	Mr. Supakit	Paemuang
Research Consultants	Mr. Chattapluek	Seethim
	Ms. Prapaporn	Banthao
Organization	Technology computer departmeny	
	Chanthaburi technical collage	
Academic Year	2023	

Abstract

The objectives of this PLAN & FLEE game project are: 1. To create the PLAN & FLEE game. 2. To study the use of Unity, Aseprite, Pro Motion NG and C# language. 3. To the satisfaction of PLAN & FLEE game players. 4. To find out the efficiency of the game PLAN & FLEE

From project implementation and testing, it was found that the highest satisfaction rating from players is in the aspect of beautiful design in the game PLANT & FLEE, with an average value ($\bar{x}$) of 4.7, which is considered excellent. The lowest rating is in the aspect of suitability of playtime in the game PLANT & FLEE, with an average value ($\bar{x}$) of 4.2, still at a good level. The overall average satisfaction across all aspects ($\bar{x}$) is 4.54, which is considered excellent. The standard deviation (S.D.) is 0.56.

กิตติกรรมประกาศ

โครงงานวิจัยฉบับนี้สำเร็จลุล่วงไปด้วยดี คณะผู้จัดทำได้รับความช่วยเหลือดูแลเอาใจใส่เป็นอย่างดีจากหลาย ๆ ฝ่าย โดยเฉพาะอาจารย์ที่ปรึกษา คือ นายฉัตรพฤกษ์ สีทิม และนางสาวประภาพร บันเทา ในการแนะนำ ตรวจแก้ไข ให้ข้อเสนอแนะ ติดตามความก้าวหน้าในการดำเนินการวิจัย คณะผู้จัดทำรู้สึกซาบซึ้งในความกรุณาของอาจารย์เป็นอย่างยิ่ง และขอขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ขอขอบคุณอาจารย์ นายฉัตรพฤกษ์ สีทิม นางสาวประภาพร บันเทา และนายปิยพงษ์ เกตุภูงา กรรมการสอบโครงการวิจัยนี้ที่ได้กรุณาให้ข้อเสนอแนะ แก้ไข และให้แนวคิดต่าง ๆ ที่เป็นประโยชน์คณะผู้จัดทำ ขอขอบคุณเพื่อนๆ ที่ช่วยให้คำแนะนำดีๆ เกี่ยวกับการเลือกคำ และเกี่ยวกับโครงการชิ้นนี้ด้วย

สุดท้ายนี้หวังว่าโครงการฉบับนี้จะเป็นประโยชน์ต่อผู้อ่านและผู้ศึกษาไม่มากก็น้อย ประโยชน์และคุณค่าของโครงการฉบับนี้ ผู้จัดทำขอมอบให้แก่ บิดา มารดา และครูอาจารย์วิทยาลัยเทคนิคจันทบุรี ที่ได้สั่งสอนอบรมมาจากอดีตจนปัจจุบันตลอดจนผู้มีพระคุณทุกท่าน

คณะผู้จัดทำ

สารบัญ

สารบัญ(ต่อ)

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

สารบัญภาพ(ต่อ)

หน้า

สารบัญภาพ(ต่อ)

หน้า

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญของโครงการ

ในปัจจุบันนี้ถือได้ว่าเกมมีบทบาทสำคัญสำในชีวิตประจำวันมากไม่ว่าจะเล่นเกมเพื่อความบันเทิงหรือเล่นเกมเพื่อความสนุกสำหรับเยาวชนนั้น เกมเข้าถึงได้ง่ายไม่ว่าจะเป็นเกมมือถือหรือการเล่นเกมผ่านทางเว็บไซต์ทำให้มีเกมหลายประเภทของเกมที่สามารถเลือกเล่นได้ตามความสนใจ การที่เกมเข้าถึงได้ง่ายนั้นทำให้มีเกมที่หลากหลายในทั้งเรื่องของเนื้อหาหรือเกมประเภทต่างๆ เป็นปัจจัยสำคัญที่ทำให้เกมเป็นหนึ่งในกิจกรรมที่เยาวชนหลายคนตัดสินใจเลือกในการใช้เวลาว่างของตนในการเลือกเล่นเกม อย่างไรก็ตามการมีคัดกรองและเลือกเล่นเกมที่มีคุณภาพเนื้อหาและเกมแนวประเภทต่างๆ ที่มีทั้งเกมแนวประอินดี้ที่เป็นเกมเข้าถึงได้ง่ายเล่นและไม่ยากเหมาะสำหรับผู้เล่นทุกวัยแต่เกมแนวเอาชีวิตนั้นกลับตรงกันข้ามกับเกมอินดี้เลยเพราะเกมแนวเอาชีวิตรอดเป็นที่มีความท้าทายและเล่นยากนอกจากนี้ เกมเอาชีวิตรอดมักจะมีบทลงโทษที่รุนแรงกว่าหากล้มเหลวเช่น สูญเสียสิ่งของทั้งหมดหรือมีแค่ชีวิตเดียวทำเยาวชนส่วนมากเข้าไม่ถึง ทั้งนี้ที่กล่าวมาถือเป็นบันดาลใจให้จัดทำพัฒนาเกม Plant&Flee ขึ้นมา

เนื่องจากในปัจจุบันนั้นเกมแนวเอาชีวิตรอดไม่ค่อยได้รับความนิยมในหมู่ผู้เล่นเยาวชนเพราะเป็นเกมที่เล่นยากและเพราะมีบทลงโทษที่รุนแรงต่อผู้เล่นผู้จัดทำโครงการจึงมีความคิดที่จะสร้างเกมโดยทำเกมปลูกผักที่มาจากเกมแนวอินดี้และผสมความท้าทายในการเอาตัวรอดจากเกมแนวเอาชีวิตรอดโดยจะลดความซับซ้อนของเกมและ การปรับปรุงกลไกการเล่นเกมและมุ่งเน้นไปที่การสร้างประสบการณ์ที่สมดุลและสนุกสนาน เกมเหล่านี้สามารถดึงดูดทั้งนักเล่นเกมที่มีประสบการณ์และผู้เล่นเยาวชน โดยทำให้กลไกการเล่นเกมง่ายขึ้น สิ่งนี้สามารถทำให้เกมเอาชีวิตรอดเข้าถึงได้ง่ายขึ้นสำหรับผู้เล่นที่อาจรู้สึกหวาดกลัวกับความซับซ้อนของเกมเอาชีวิตรอดแบบดั้งเดิมและช่วยพัฒนาทักษะการเล่นเกมของผู้เล่น เนื่องจากผู้จัดทำโครงงานมีความสนใจและชื่นชอบในการเล่นเกมและได้มีการศึกษาโค้ดและการใช้งานโปรแกรมในการสร้างเกมผู้จัดทำจึงมีจุดมุ่งหมายในการสร้างเกมนี้ขึ้นมา

จากเหตุผลข้างต้นจึงเป็นสาเหตุให้ผู้จัดทำโครงงานมีความต้องการที่จะสร้างเกม Plant&Flee ที่ยังคงความสนุกสนานและความท้าทายของเกมแนวเอาชีวิตรอดแบบดั้งเดิมและลดความซับซ้อนของเกม ซึ่งผสมผสานองค์ประกอบของการทำฟาร์มและการบริหารเวลาเข้าด้วยกันผู้เล่นจะต้องเพาะปลูกทำฟาร์มของตนและยังต้องเอาชีวิตรอดจากสัตว์ประหลาดและภัยคุกคาม โดยเพิ่มองค์ประกอบของกลยุทธ์และ

ความตื่นเต้นให้กับการเล่นเกมทำให้เล่นสนุกและผู้เล่นเข้าใจไม่ยากเหมาะกับผู้เล่นทั่วไป หรือในหมู่ผู้เล่นเยาวชนที่ชื่นชอบเกมอินดี้หรือเกมแนวเพาะปลูกทำฟาร์มผู้จัดทำโครงการจึงใช้โปรแกรม Unity2D และภาษา C# ขึ้นมาเพื่อต้องการสร้างเกมได้ง่ายและไม่มีความซับซ้อน ด้วยการสร้างเกมที่มีทั้งความท้าทายและความสนุกสนานในขณะที่ยังคงนำเสนอรูปแบบการเล่นและการเล่าเรื่องที่เป็นเอกลักษณ์ซึ่งเป็นเกมอินดี้ เพื่อดึงดูดผู้เล่นเยาวชนให้มาลองเปิดใจเล่นเกมแนวเอาชีวิตรอดมากขึ้น

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อสร้างเกม Plant & Flee

1.2.2 เพื่อศึกษาการใช้โปรแกรม Unity ,Aseprite ,Pro Motion NG และ ภาษา C#

1.2.3 เพื่อความพึงพอใจของผู้เล่นเกม Plant & Flee

1.2.4 เพื่อหาประสิทธิภาพของเกม Plant & Flee

1.3 ขอบเขตของโครงการ

1.3.1 ประชากรและกลุ่มตัวอย่าง

1.3.1.1 ประชากร คือ นักศึกษาแผนกเทคโนโลยีคอมพิวเตอร์

1.3.1.2 กลุ่มตัวอย่าง คือ นักศึกษาชั้นปวส.2

1.3.2 ตัวแปรที่ปรึกษา

1.3.2.1 ตัวแปรต้น เกม Plant & Flee

1.3.2.2 ตัวแปรตาม ประสิทธิภาพของเกม ด้านความสวยงาม ด้านความเสถียร และความน่าสนใจ ของเกม Plant & Flee

1.3.3 เครื่องมือที่ใช้ในโครงการ

1.3.3.1 แบบสอบถามความพึงพอใจของผู้เล่นเกม Plant & Flee

1.3.3.2 แบบวัดประสิทธิภาพของเกม Plant & Flee

1.3.4 วิธีการเก็บรวบรวมข้อมูล

1.3.4.1 กำหนดกลุ่มประชากรและกลุ่มตัวอย่าง

1.3.4.2 ศึกษาข้อมูลเบื้องต้น

1.3.4.3 ออกแบบและพัฒนาเกม Plant & Flee

1.3.4.4 ทดสอบเล่นเกม

1.3.4.5 ทดสอบเพื่อปรับปรุงข้อบกพร่อง

1.3.4.6 ทดสอบโดยกลุ่มตัวอย่าง

1.3.4.7 นำแบบสอบถามความพึงพอใจให้กลุ่มผู้ทดลองได้ลงความคิดเห็น

1.3.4.8 เก็บรวบรวมแบบสอบถามทั้งหมดและนำมาประมวลให้สมบูรณ์

1.3.5 สถิติที่ใช้ในการวิเคราะห์ข้อมูล

1.3.5.1 ค่าเฉลี่ย โดยใช้สูตรค่าเฉลี่ย

$$\bar{x} = \frac{\sum x}{n}$$

$\bar{x}$ = ค่าเฉลี่ยของคะแนน

$\sum x$ = ผลรวมของคะแนน

n = จำนวน

1.3.5.2 ส่วนเบี่ยงเบนมาตรฐานโดยใช้สูตร

$$S.D. = \frac{\sqrt{n\sum x^2 - (\sum x)^2}}{n(n-1)}$$

$S.D.$ แทน ส่วนเบี่ยงเบนค่ามาตราฐาน

x แทน คะแนนแต่ละคน

n แทน จำนวนคน

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 ได้เกม Plant & Flee

1.4.2 ได้ใช้โปรแกรม Unity ,Aseprite ,Pro Motion NG และ ภาษา C#

1.4.3 ได้ความพึงพอใจของผู้เล่นเกมอยู่ในระดับ ดี

1.4.4 ได้เกม Plant & Flee ที่มีความแม่นยำร้อยละ 80 ขึ้นไป

1.5 สมมติฐานของโครงการและวิจัย

ได้เกมที่มีประสิทธิภาพ ที่สามารถมอบความบันเทิง ความสนุกสนาน และความตื่นเต้นได้

1.6 นิยามศัพท์

1.6.1 เกม คือ เป็นลักษณะของกิจกรรมของมนุษย์เพื่อประโยชน์อย่างใดอย่างหนึ่ง เช่น เพื่อความสนุกสนานบันเทิง เพื่อฝึกทักษะ และเพื่อการเรียนรู้ เป็นต้น และในบางครั้งอาจใช้เพื่อประโยชน์ทางการศึกษาได้

1.6.2 แนว single player คือ วีดีโอเกมที่มีการนำเข้าคำสั่งและการบังคับต่างๆจากผู้เล่นเพียงคนเดียวตลอดตั้งแต่ต้นจนจบเกม ในบางเกมที่เล่นได้หลายคนอาจมี โหมดผู้เล่นเดี่ยว (single-player mode) ติดตั้งเข้ามาด้วยสำหรับการเล่นเพียงคนเดียว

1.6.3 แนว survival horror คือ เป็นประเภทย่อยของวิดีโอเกมแอ็กชันผจญภัยมีแรงบันดาลใจจากบันเทิงคดีสยองขวัญ แม้ว่าจะมีการต่อสู้เป็นส่วนหนึ่งในระบบการเล่น แต่ผู้เล่นจะรู้สึกมีพละกำลังน้อยกว่าเกมแนวแอ็กชันทั่วไป เนื่องจากเกมจะจำกัดกระสุนปืน พลังชีวิต ความเร็ว หรือปัจจัยอื่น ๆ

1.6.4 แนว farming คือวิธีการเล่นอย่างหนึ่งที่ผู้เล่นทำอะไรบางอย่างซ้ำๆ เพื่ออัปสกิลคาแรกเตอร์ เช่น อาจจะเป็นการตีมอนสเตอร์ไปเรื่อยๆ เก็บไอเทม หรือเก็บเงิน แล้วแต่รูปแบบของเกมนั้นๆ

1.6.5 มุมมอง top down คือ เป็นการมองผ่านมุมมองที่ใหญ่ที่สุดก่อนแล้วค่อยเป็นการวิเคราะห์ภาพรวมสถานการณ์ระดับโลก การเปลี่ยนแปลงต่าง ๆ ว่าจะเกิดผลอะไรตามมาบ้าง แล้วจึงมองให้เป็นภาพลึก ๆ เพื่อพิจารณาถึงปัจจัยอื่นในการเลือกหุ้นที่ขนาดเล็กลงมาตามลำดับ

1.6.6 ภาพ 2 มิติ คือภาพที่แสดงรูปทรงของภาพใน 2 มิติคือ ความกว้าง และความสูง หรือความกว้างและความยาว โดยจะมองเห็นเป็นลักษณะของพื้นที่ เช่น รูปสามเหลี่ยม สี่เหลี่ยม ห้าเหลี่ยม หกเหลี่ยม วงกลม หรือรูปทรงอื่น ๆ

1.6.7 Pixel คือ จุดภาพที่แสดงบนจอแสดงผล ในภาพ 1 ภาพจะมีจุดภาพมากมายเรียงต่อกันรวมกันเป็นภาพขึ้นมา ซึ่งจอภาพที่มีจำนวน Pixel มากนั้น จะมีความละเอียดของภาพมากเช่นกัน สามารถคำนวณ จากจำนวนพิกเซลแนวนอน x จำนวนแนวตั้ง เช่น 1284 x 1024 = 1,310,720 Pixel แต่นิยมเขียนแบบย่อเป็น 1.3 MP แทน เนื่องจาก MP (megapixel) เป็นหน่วยนับหลักล้านของพิกเซลนั่นเอง

1.6.8 Unity คือ ซอฟต์แวร์แบบข้ามแพลตฟอร์ม ใช้เพื่อสำหรับการพัฒนาซอฟต์แวร์และการจำลองต่างๆ จำพวกเช่น วีดีโอเกม, อุตสาหกรรมยานยนต์, การขนส่ง, ภาพยนตร์, แอนิเมชัน, สถาปัตยกรรม, วิศวกรรมศาสตร์, วิศวกรรมก่อสร้าง, วิศวกรรมการบินและอวกาศ, การพนัน, และอื่นๆ ยูนิตี นักพัฒนา Unity Technologies

1.7 ระยะเวลาที่ใช้ในการจัดทำโครงงาน

ระยะเวลาในการวิจัย เป็นเวลาทั้งสิ้น 18 สัปดาห์ตั้งแต่วันที่ 16 ตุลาคม พ.ศ. 2566 ถึงวันที่ 15 กุมภาพันธ์ พ.ศ.2567

ตารางที่ 1.1 ระยะเวลาที่ใช้ในการจัดทำโครงงาน

ขั้นตอนการดำเนินงาน	สัปดาห์																	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1.กำหนดการโครงการ	◄	►																
2.เสนอโครงการ		◄	►															
3.ศึกษารายละเอียดข้อมูล				◄	►	►												
4.ออกแบบระบบ						◄	►	►	►	►								
5.ดำเนินการทำระบบ								◄	►	►	►	►	►	►	►			
6.นำเสนอระบบ															◄	►		
7.ประเมินและสรุปผล																◄	►	
8.จัดทำเอกสารโครงงาน																	◄	►

1.8 งบประมาณและทรัพยากร

ตารางที่ 1.2 งบประมาณที่ใช้ในการจัดทำโครงงาน

ที่	รายการวัสดุ	จำนวน	หน่วย	ราคาบาท
1	ค่าโปรแกรม Aseprite	1	289	289
2	ค่าโปรแกรม Pro Motion NG	1	469	469
รวม				758.-

บทที่ 2

เอกสารและงานวิจัยที่เกี่ยวข้อง

โครงการ Plant & Flee เป็นเกมที่มีวัตถุประสงค์เพื่อสร้างเกม Plant & Flee เพื่อเสริมสร้างทักษะการสร้างเกมการวางแผนและจัดการทรัพยากรต่างๆและการใช้ภาษาC# ในการเขียนโปรแกรม เพื่อให้โครงการบรรลุวัตถุประสงค์ และเป็นไปตามขอบเขตที่กำหนดไว้ คณะผู้จัดทำได้ทำการศึกษาทฤษฎีและเนื้อหาที่เกี่ยวข้อง ดังนี้

2.1 Unity

2.2 ภาษา C#

2.3 Aseprite

2.4 Storyboard

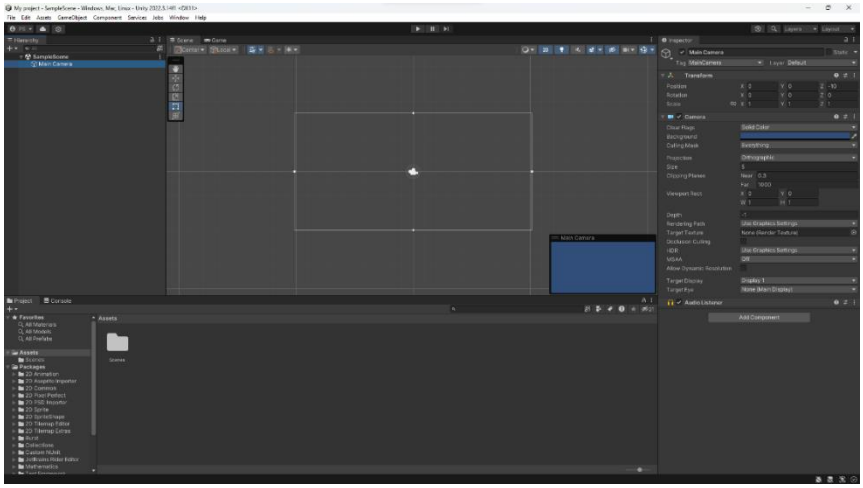
2.5 เอกสารและงานวิจัยที่เกี่ยวข้อง

2.1 Unity (https://th.wikipedia.org/wiki/ยูนิตี_(เกมเอนจิน))

ยูนิตี (Unity) คือซอฟต์แวร์แบบข้ามแพลตฟอร์ม ใช้เพื่อสำหรับการพัฒนาซอฟต์แวร์และการจำลองต่างๆ จำพวกเช่น วีดีโอเกม, อุตสาหกรรมยานยนต์, การขนส่ง, ภาพยนตร์, แอนิเมชัน สถาปัตยกรรม, วิศวกรรมศาสตร์, วิศวกรรมก่อสร้าง, วิศวกรรมการบินและอวกาศ, การพนัน, และอื่นๆ ยูนิตี เอนจินเปิดตัวครั้งแรกในปี พ.ศ. 2548 โดยมีจุดประสงค์เพื่อให้การสร้างเกมเข้าถึงง่ายขึ้น โดยในปีต่อมา ยูนิตี ได้รับรางวัลรองชนะเลิศในหมวดการใช้งานที่ดีที่สุดของ Mac OS X Graphics ในงาน Apple Design Awards ของ แอปเปิล และปัจจุบันยูนิตี ได้ขยายการรองรับไปยังแพลตฟอร์มอื่นๆ เพิ่มเติมมากกว่า 18 แพลตฟอร์ม

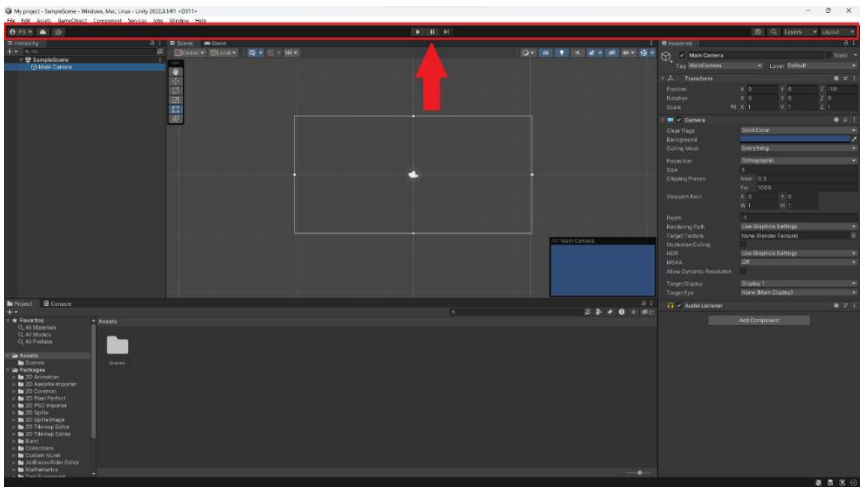
รูปภาพที่ 2.1 Unity Logo

2.1.1 ส่วนประกอบหน้าต่าง Interface ของโปรแกรม Unity แบ่งออก ดังนี้



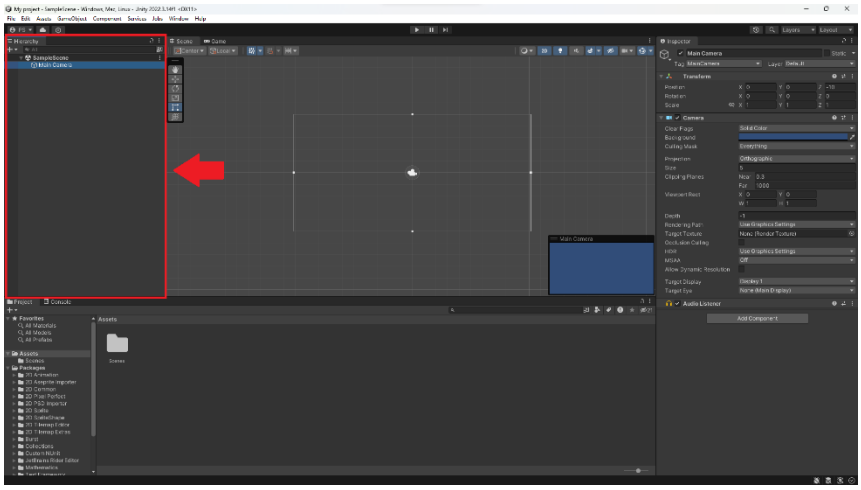
รูปภาพที่ 2.2 Interface ของ Unity

1) ดังรูปภาพที่ 2.3 Toolbar เป็นแถบเครื่องมือที่มีไว้เข้าถึงคุณลักษณะการทำงานที่สำคัญที่สุด ด้านซ้ายมีเครื่องมือพื้นฐานสำหรับการจัดการมุมมองภาพและวัตถุภายในภาพ อยู่ตรงกลางคือการควบคุมการเล่นหยุดชั่วคราวและขั้นตอน ปุ่มทางด้านขวาจะทำให้คุณสามารถเข้าถึง Unity Cloud Services และบัญชี Unity ของคุณตามด้วยเมนูการมองเห็นของเลเยอร์และในที่สุดเมนูเค้าโครงของโปรแกรมแก้ไข (ซึ่งให้รูปแบบอื่น ๆ สำหรับหน้าต่างตัวแก้ไขและช่วยให้คุณสามารถบันทึกกำหนดเองของคุณเองได้ รูปแบบ) ดังนี้



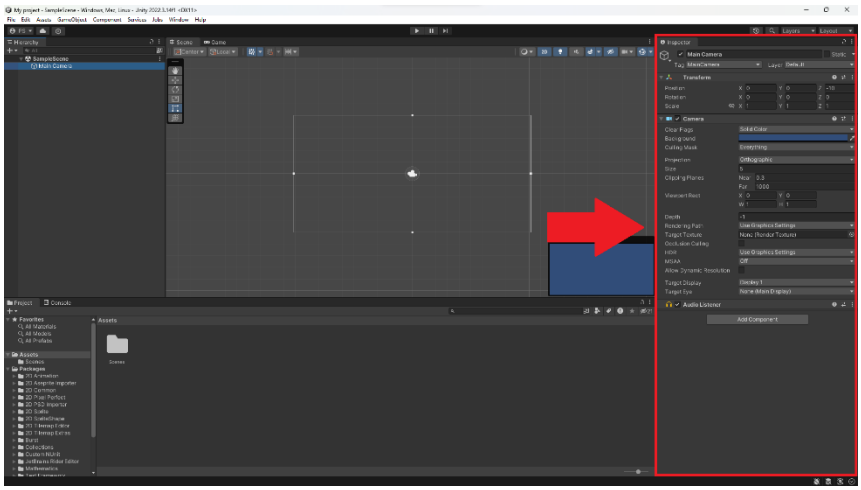
รูปภาพที่ 2.3 Toolbar

2) ดังรูปภาพที่ 2.4 Hierarchy คือส่วนที่บอกลำดับชั้น ของ Object ต่างๆ ที่อยู่ใน Scene นั้นๆซึ่งมีทั้ง Object แบบเดี่ยว และ Object ที่เป็นแม่ลูกกัน ซึ่ง เมื่อมีการจัดการอะไรบางอย่างกับ Object แม่ Object ที่เป็นลูกนั้นก็จะมีการเปลี่ยนแปลงตามไปด้วย การสร้าง Object มีวิธีการคือลาก Object ต่างๆ ที่อยู่ ใน Project มาใส่ไว้ในส่วนของ Hierarchy หลังจาก นั้นจะปรากฏวัตถุที่ลากจาก Project มาวางบน Hierarchy ปรากฏขึ้นบน Scene ซึ่ง Object ต่างๆ เหล่านี้สามารถเพิ่ม/แก้ไข/ลบ ได้โดยไม่ กระทบกับ Object ที่อยู่ใน Project ดังนี้



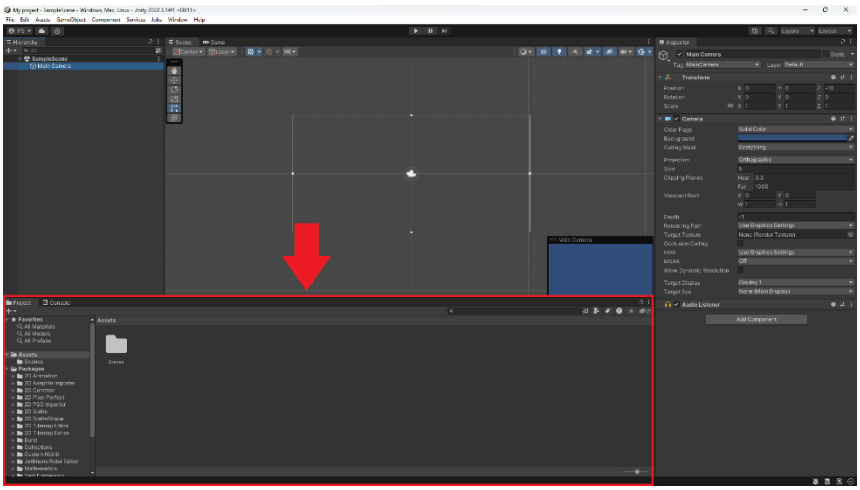
รูปภาพที่ 2.4 Hierarchy

3) ดังรูปภาพที่ 2.5 Inspector เป็นส่วนที่บ่งบอกถึงคุณสมบัติต่างๆ ของ Object ซึ่งสามารถจัดการคุณสมบัติต่างๆ ของ Object ได้ในกรอบของ Inspector ดังนี้



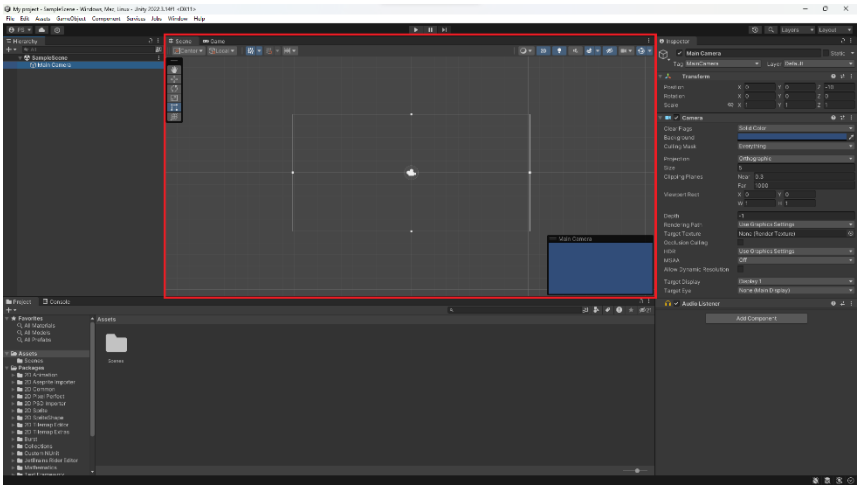
รูปภาพที่ 2.5 Inspector

4) ดังรูปภาพที่ 2.6 Project เป็นส่วนที่ใช้ในการเก็บทรัพยากร ต่างๆก่อนนำไปสร้างเกม เช่น - สคริปท์ต่างๆ ที่ใช้กำหนดควบคุมตัวเกม – 3D โมเดล ใช้เป็นตัวละครหรือวัตถุต่างๆ ในเกม – Textures หรือ พื้นผิวต่างๆ - ไฟล์เสียง หรือวีดีโอ – Prefabs – อื่นๆ ดังนี้



รูปภาพที่ 2.6 Project

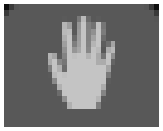
5) ดังรูปภาพที่ 2.7 Scene เป็นส่วนที่บ่งบอกว่าในฉากที่กำลังทำงาน มี Object อะไรบ้าง สามารถจัดการ Object ต่างๆ เช่น กล้อง แสง เอฟเฟค หรือโมเดล 3 มิติ ได้จากส่วนนี้ ดังนี้



รูปภาพที่ 2.7 Scene

เครื่องมือของโปรแกรม Unity มีดังนี้

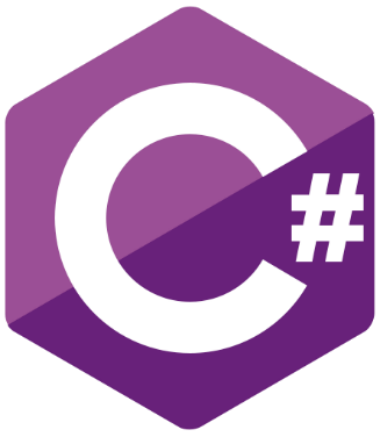
ตารางที่ 2.1 ตารางเครื่องมือของโปรแกรม Unity

รูปภาพ	ชื่อเครื่องมือ	คำอธิบาย
	Pan	Pan Hotkey หรือคีย์ลัดคือ Q เป็นเครื่องมือสำหรับจัดการมุมมองในโปรแกรม Unity3D ถ้ากดปุ่ม middle mouse ค้างไว้ จะเป็นการเคลื่อนย้าย viewport ทั้งหมด แต่ถ้ากดปุ่ม right mouse ค้างไว้จะเป็นการเคลื่อนย้าย viewport โดยที่ยังรักษาจุดศูนย์กลางอยู่ที่มุมมองของกล้อง
	Move	Move Hotkey หรือคีย์ลัดคือ W เป็นเครื่องมือสำหรับปรับตำแหน่งของ Object ทั้งหมดใน Viewport สามารถใช้เครื่องมือนี้ในการเคลื่อนย้ายตำแหน่ง x,y,z ได้
	Rotate	Rotate Hotkey หรือคีย์ลัดคือ E เป็นเครื่องมือสำหรับปรับองศาของ Object ทั้งในแนวแกน x, y และ z
	Scale	Scale Hotkey หรือคีย์ลัดคือ R เป็นเครื่องมือสำหรับปรับขนาดของ Object แบบค่า scale โดยยังคงค่า width และ height ไว้
	Rect Tool	Rect Tool Hotkey หรือคีย์ลัดคือ T เป็นเครื่องมือสำหรับปรับขนาดของ Object แบบแบบกับ scale แต่จะเปลี่ยนแปลงเฉพาะค่า width และ height ค่า scale จะยังคงเป็น 1 เสมอ

2.2 ภาษา C# (https://marcuscode.com/lang/csharp/introduction)

C# เป็นภาษาเขียนโปรแกรมแบบ multi-paradigm ซึ่งมีรูปแบบภาษาที่ตายตัว และเป็นรูปแบบบังคับในการเขียน มีฟังก์ชัน และยังเป็นภาษาการเขียนโปรแกรมที่มีคุณสมบัติเป็นแบบออบเจ็คด้วย ซึ่งมันถูกพัฒนาโดยบริษัท Microsoft ภายใต้ .NET framework โดยในการพัฒนาภาษา C# นี้ มีความตั้งใจให้มันเขียนง่าย ทันสมัย เป็นโปรแกรมเพื่อวัตถุประสงค์ทั่วไปและเป็นแบบออบเจ็ค C# เป็นภาษาเขียนโปรแกรมเพื่อวัตถุประสงค์ทั่วไป การพัฒนานั้นนำทีมโดย Anders Hejlsberg และเวอร์ชันล่าสุดคือ C# 6.0 ซึ่งถูกเผยแพร่ในปี 2015

ประวัติของภาษา C# ในระหว่างการพัฒนาของ .NET Framework คลาสและไลบรารี่ต่างๆ ถูกเขียนขึ้นโดยระบบการจัดการโค้ดสำหรับการคอมไพล์ที่เรียกว่า Simple Managed C (SMC) ในเดือนมกราคม 1999 Anders Hejlsberg ได้ก่อตั้งทีมเพื่อสร้างภาษาใหม่ในเวลานั้น ที เรียกว่า Cool ซึ่งเป็นคำย่อของ "C-like Object Oriented Language" ในเวลาที่โครงการ .NET ของ Microsoft ถูกเผยแพร่ในเดือนกรกฎาคม 2000 ในการประชุมของกลุ่มนักพัฒนามืออาชีพ ภาษาได้ถูกเปลี่ยนชื่อเป็น C# และคลาสไลบรารี่และ ASP.NET ได้ถูกเชื่อมเข้ากับ C#ชื่อของภาษา "C Sharp" นั้นมีแรงบรรดารใจมาจากเครื่องหมายของเพลง โดย Sharp (#) นั้นเป็นสิ่งที่บ่งบอกว่าโน้ตของดนตรีที่ถูกเขียนขึ้นนั้นเป็นสิ่งที่ทำให้เสียงของดนตรีสูงขึ้น ซึ่งนี่จะคล้ายกับภาษา C++ โดย "++" ซึ่งหมายถึงค่าของตัวแปรจะเพิ่มขึ้นหนึ่ง นอกจากนี้มันยังเป็นภาษาที่หมายถึงการเพิ่มความสามารถของภาษา C++



รูปภาพที่ 2.8 C# Programing language

2.2.1 ชนิดข้อมูล (Data Type) ใน C#

C# กำหนดชนิดของข้อมูลไว้หลากหลายชนิดเพื่อรองรับการจัดการข้อมูลหลายๆ ประเภท ดังนี้

ตารางที่ 2.2 ตารางชนิดข้อมูลใน C#

ชนิดข้อมูล	คำอธิบาย
char	อักขระเดี่ยว เช่น a
bool	ค่าความจริง เป็นไปได้สองค่าคือ true หรือ false
byte	จำนวนเต็มไม่มีเครื่องหมาย ตั้งแต่ 0 ถึง 255
int	จำนวนเต็มมีเครื่องหมาย ตั้งแต่ -2147483648 ถึง 2147483647
uint	จำนวนเต็มไม่มีเครื่องหมาย ตั้งแต่ 0 ถึง 4294967295
long	จำนวนเต็มมีเครื่องหมาย ตั้งแต่ -9223372036854775808 ถึง 9223372036854775807
ulong	จำนวนเต็มไม่มีเครื่องหมาย ตั้งแต่ 0 ถึง 18446744073709551615
float	จำนวนจริง (มีทศนิยมได้) เช่น 3.14159
double	จำนวนจริงที่เก็บความละเอียดมากเป็นสองเท่า
string	ข้อความ (สายอักขระ) เช่น Hello

2.2.2 ตัวดำเนินการทางคณิตศาสตร์

ตัวดำเนินการสำหรับคำนวณทางคณิตศาสตร์เป็นตัวดำเนินการที่ใช้มากที่สุดในการเขียนโปรแกรมคอมพิวเตอร์ ตัวดำเนินการทางคณิตศาสตร์มีทั้งหมด 7 ตัว ดังตารางต่อไปนี้

ตารางที่ 2.3 ตัวดำเนินการทางคณิตศาสตร์

สัญลักษณ์	**ตัวดำเนินการ**	**ตัวอย่างและความหมาย**
+	บวก (add)	X = a + b; นำค่า a บวกกับ b แล้วก็บไว้ใน x
-	ลบ (subtract)	X = a - b; นำค่า a ลบกับ b แล้วก็บไว้ใน x
*	คูณ (multiply)	X = a * b; นำค่า a คูณกับ b แล้วก็บไว้ใน x
/	หาร (divide)	X = a / b; นำค่า a หารด้วย b แล้วก็บไว้ใน x
%	หารเอาเศษ (modula)	X = a % b; นำเศษจากการหาร a ด้วย b แล้วก็บไว้ใน x
++	เพิ่มค่า	x++ = เพิ่มค่า x ขึ้นหนึ่งค่า
--	ลดค่า	x-- = ลดค่า x ลงหนึ่งค่า

นิพจน์ทางคณิตศาสตร์นั้นสามารถเกิดจากตัวดำเนินการหลาย ๆ ตัวมากระทำกับตัวแปรหรือค่าคงที่ได้ และสามารถใส่ตัวดำเนินการไปพร้อมกับการประกาศตัวแปรได้ แต่ประเภทของข้อมูลที่ประกาศตัวแปรนั้นจะต้องสอดคล้องกับผลลัพธ์ที่ได้จากการประมวลผลของนิพจน์นั้น

2.2.3 ตัวดำเนินการเปรียบเทียบ

ตัวดำเนินการเปรียบเทียบใช้สำหรับนำค่าของตัวแปรหรือค่าคงที่จำนวนสองตัวมาเปรียบเทียบกัน ผลลัพธ์ที่ได้จะเป็นค่าทางตรรกะ (bool) คือมีค่าเป็นจริง (true) หรือเท็จ (false) ตัวดำเนินการประเภทนี้มี 6 ตัว ดังตารางต่อไปนี้

ตารางที่ 2.4 ตัวดำเนินการเปรียบเทียบ

สัญลักษณ์	ตัวดำเนินการ	ตัวอย่างและความหมาย
==	เท่ากับ	a == b; เป็นจริง ถ้าค่าในตัวแปร a เท่ากับตัวแปร b
>	มากกว่า	a > b; เป็นจริง ถ้าค่าในตัวแปร a มากกว่าตัวแปร b
>=	มากกว่าหรือเท่ากับ	a >= b; เป็นจริง ถ้าค่าในตัวแปร a มากกว่าหรือเท่ากับตัวแปร b
<	น้อยกว่า	a < b; เป็นจริง ถ้าค่าในตัวแปร a น้อยกว่าตัวแปร b
<=	น้อยกว่าหรือเท่ากับ	a <= b; เป็นจริง ถ้าค่าในตัวแปร a น้อยกว่าหรือเท่ากับตัวแปร b
!=	ไม่เท่ากับ	a != b; เป็นจริง ถ้าค่าในตัวแปร a ไม่เท่ากับค่าในตัวแปร b

ในภาษา C# ตัวดำเนินการเปรียบเทียบสามารถใช้กับข้อมูลประเภทตัวเลขได้ทุกประเภท รวมไปถึงข้อมูลแบบ float, double, decimal และ char โดยการเปรียบเทียบข้อมูลแบบ char นั้นจะเป็นการนำค่ารหัส Unicode มาเปรียบเทียบกัน

2.2.4 ตัวดำเนินการทางตรรกศาสตร์

ตัวดำเนินการทางตรรกศาสตร์ (Logical Operators) เป็นการนำตัวแปรหรือค่าคงที่สองค่ามาประมวลผลทางตรรกศาสตร์ต่อกัน ผลลัพธ์ที่ได้จะเป็นจริง (true) หรือเท็จ (false) เท่านั้น ตัวดำเนินการประเภทนี้บางตัวยังสามารถประมวลผลในระดับบิตได้อีกด้วย สัญลักษณ์ของตัวดำเนินการแสดงได้ ดังตารางต่อไปนี้

ตารางที่ 2.5 ตัวดำเนินการทางตรรกศาสตร์

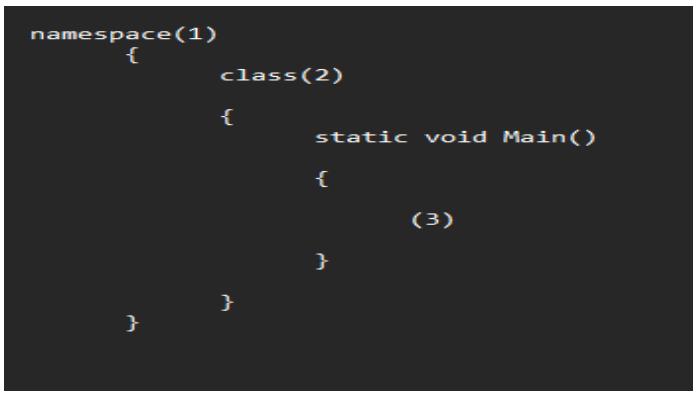
สัญลักษณ์	ตัวดำเนินการ	**ตัวอย่างและความหมาย**
!	NOT	!a กลับค่าลอจิกของ a ถ้าหาก a เป็นจริง ผลลัพธ์จะเป็น เท็จ
&	AND	A & b ดำเนินการ AND ระหว่าง a กับ b
|	OR	a | b ดำเนินการ OR ระหว่าง a กับ b
^	XOR	a ^ b ดำเนินการ exclusive ระหว่าง a กับ b
&&	Short-circuit AND	a && b ดำเนินการ AND แบบ short-circuit AND
||	Short-circuit OR	a || b ดำเนินการ OR แบบ short-circuit OR

2.2.5 ดังรูปภาพที่ 2.9 โครงสร้างของภาษา C# ขั้นพื้นฐานมีรายละเอียด ดังนี้

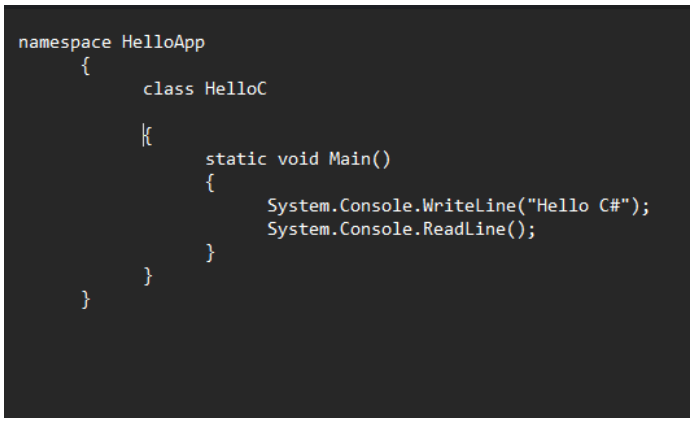
- หมายเลข (1) เป็นการระบุชื่อของ namespace ใช้ในการกำหนดขอบเขตให้กับคลาสต่างๆ รวมถึงใช้ในการจัดโครงสร้างของโปรแกรมขนาดใหญ่ให้เป็นสัดส่วนอีกด้วย โดยเฉพาะอย่างยิ่งในการเขียนโปรแกรมคอมพิวเตอร์ที่ซับซ้อนโดยมีผู้เขียนโปรแกรมหลายคน นอกจากนี้ การกำหนด namespace ยังช่วยป้องกันปัญหาการตั้งชื่อคลาสหรือค่าคงที่อื่นๆ ซ้ำกันได้

- หมายเลข (2) เป็นการระบุชื่อของ class ใช้ในการกำหนดส่วนประกอบต่างๆ ที่จะนำไปสร้าง Object

- หมายเลข (3) เป็นการระบุพื้นที่สาหรับคำสั่งต่างๆ ที่ผู้เขียนโปรแกรมต้องการให้คอมพิวเตอร์ปฏิบัติตาม



รูปภาพที่ 2.9 โครงสร้างของภาษา C#



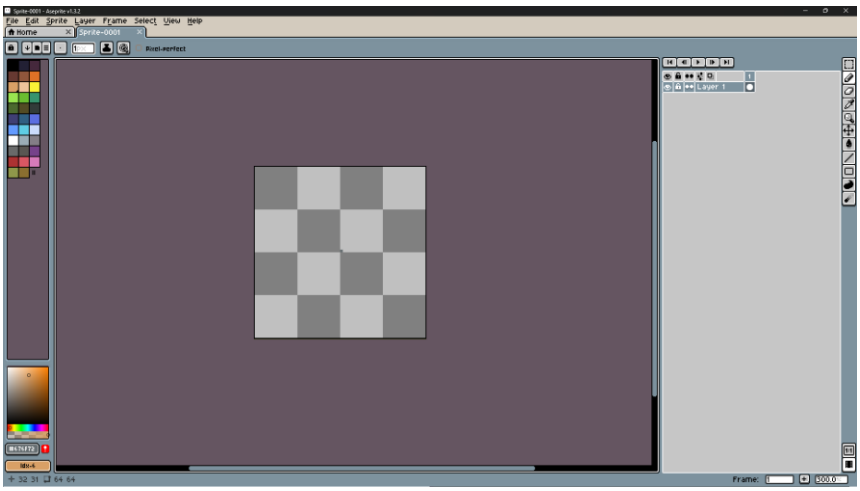
รูปภาพที่ 2.10 ตัวอย่างโครงสร้างของภาษา C#

2.3 Aseprite (https://en.wikipedia.org/wiki/Aseprite)

Aseprite เป็นโปรแกรมแก้ไขรูปภาพที่เป็นกรรมสิทธิ์และมีแหล่งที่มา ซึ่งออกแบบมาเพื่อการวาดภาพแบบพิกเซลอาร์ตและแอนิเมชั่นเป็นหลัก มันทำงานบน Windows, macOS และ Linux และมีเครื่องมือต่าง ๆ สำหรับการแก้ไขรูปภาพและแอนิเมชั่น เช่น เลเยอร์, เฟรม, การรองรับไทล์แมป, อินเทอร์เฟซบรรทัดคำสั่ง, สคริปต์ Lua และอื่น ๆ อีกมากมาย พัฒนาโดย Igara Studio S.A. และนำโดยนักพัฒนา David, Gaspar และ Martín Capello สามารถดาวน์โหลด Aseprite เป็นฟรีแวร์ หรือซื้อบน Steam หรือ Itch.io ซอร์สโค้ดและไบนารีของ Aseprite ได้รับการเผยแพร่ภายใต้สิทธิ์การใช้งาน EULA การศึกษา และสิทธิ์การใช้งานที่เป็นกรรมสิทธิ์ของ Steam

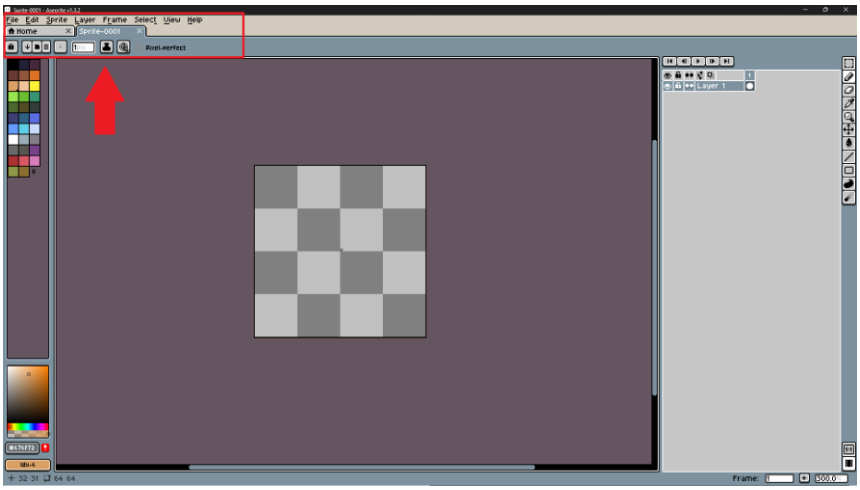
รูปภาพที่ 2.11 Aseprite Logo

2.3.1 ส่วนประกอบหน้าต่าง Interface ของโปรแกรม Aseprite แบ่งออก ดังนี้



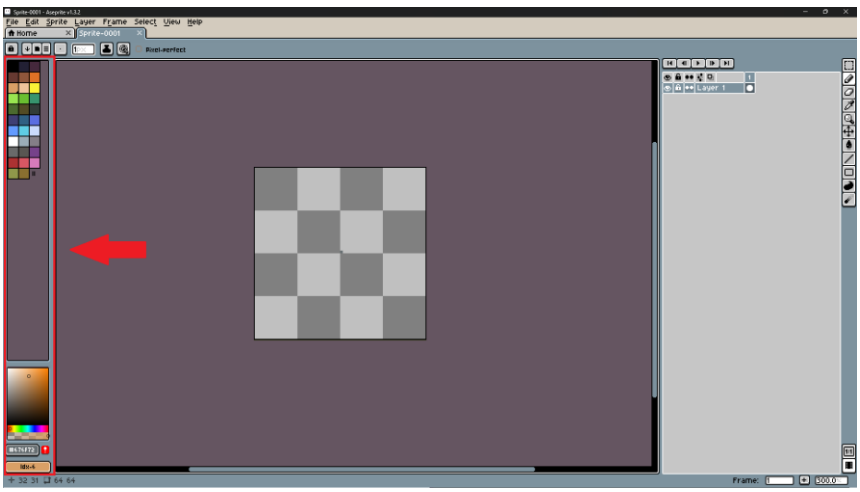
รูปภาพที่ 2.12 Interface ของ Aseprite

1) ดังรูปภาพที่ 2.13 Toolbar เป็นแถบเครื่องมือที่มีไว้เข้าถึงคุณลักษณะการทำงานที่สำคัญที่สุด เป็นพื้นฐานสำหรับการจัดการสิ่งต่างๆภายในโปรแกรม เช่น File, Edit, Sprite, Layer, Frame, Select, View, Help และแสดงไฟล์ต่างๆที่เปิดไว้ พร้อมเครื่องมืออำนวยความสะดวก ดังนี้



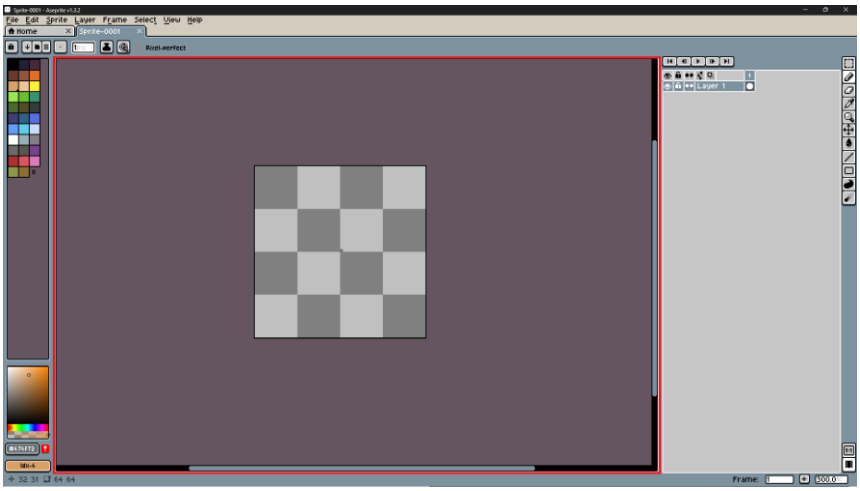
รูปภาพที่ 2.13 Toolbar

2) ดังรูปภาพที่ 2.14 Color เป็นแถบเครื่องมือที่มีไว้เข้าถึงการใช้งานสีของโปรแกรม โดยสามารถใช้งานสีต่างๆได้จากส่วนนี้ และยังสามารถเพิ่มลบสีได้อีกด้วย และสามารถเพิ่มสีได้ด้วยการกรอก code สี เป็นต้น ดังนี้



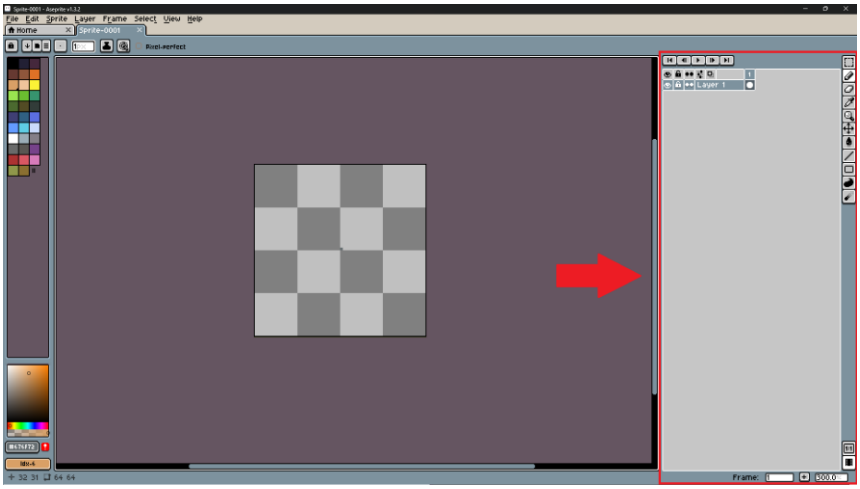
รูปภาพที่ 2.14 Color

3) ดังรูปภาพที่ 2.15 Middle part เป็นที่ใช้ในการวาด ออกแบบภาพ pixel ของโปรแกรม Aseprite ดังนี้



รูปภาพที่ 2.15 Middle part

4) ดังรูปภาพที่ 2.16 Object เป็นส่วนที่แสดงเครื่องมือต่างของโปรแกรม ที่ใช้คู่กับส่วน Middle part ในการวาด ลบ เติมสี ย้ายวัตถุ และยังรวมไปถึงการสร้าง Frame ต่างๆก็ยังอยู่ในส่วนนี้อีกด้วย ดังนี้



รูปภาพที่ 2.16 Object

เครื่องมือของโปรแกรม Aseprite มีดังนี้

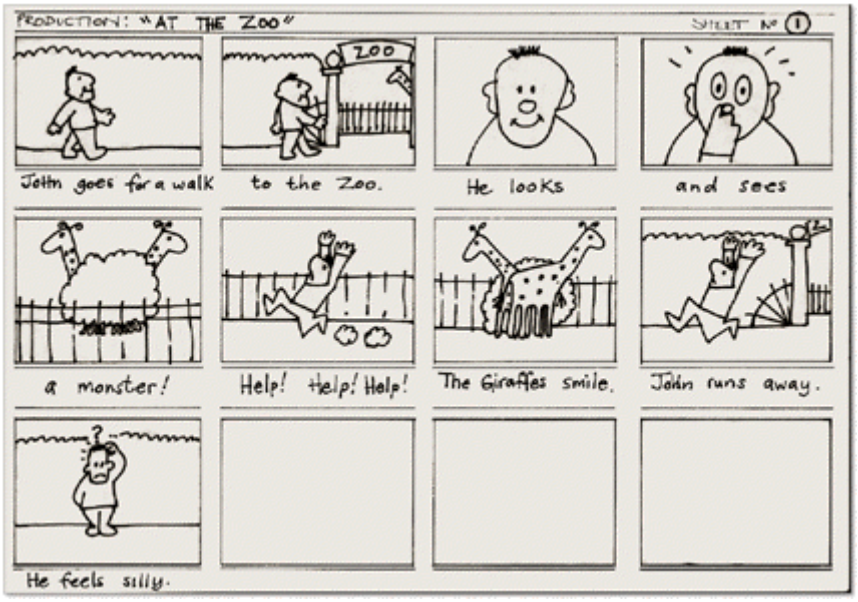
ตารางที่ 2.6 ตารางเครื่องมือของโปรแกรม Aseprite

รูปภาพ	ชื่อเครื่องมือ	คำอธิบาย
	Rectangular Marquee	ใช้สร้างการเลือกพื้นที่ในภาพที่มีรูปร่างของสี่เหลี่ยม (rectangular selection) หรือกล่องสี่เหลี่ยมบนภาพ การเลือกนี้ทำให้คุณสามารถทำตามการกระทำต่าง ๆ บนพื้นที่ที่คุณเลือกได้, เช่น การคัดลอก, ตัด, หรือปรับแต่งสีเฉพาะในบริเวณนั้น
	Pencil	ใช้สำหรับการวาดแบบพิกเซล (pixel art) หรือการแก้ไขภาพที่มีลักษณะของพิกเซลอาร์ต โดยเครื่องมือนี้จะสร้างเส้นที่มีขอบขนานกับแกน x และ y ของภาพ
	Eraser	ใช้เพื่อลบสีหรือพิกเซลที่คุณเลือกในภาพ ซึ่งทำให้สามารถทำการลบส่วนของภาพหรือปรับแต่งภาพได้ตามต้องการ
	Eyedropper	ใช้ในการเลือกสีจากภาพเพื่อนำมาใช้ในการวาด, แก้ไข, หรือปรับแต่งสี โดยคุณสามารถเลือกสีที่คลิกไปในภาพเพื่อนำมาใช้ในการวาดหรือปรับแต่งภาพ
	Zoom	ใช้ในการขยายและย่อภาพ เพื่อให้คุณสามารถดูรายละเอียดของภาพหรือแก้ไขส่วนที่เล็กมีความเรียบเรียงมากขึ้น หรือในกรณีการสร้าง pixel art, จะช่วยให้สามารถทำงานในระดับขอบของพิกเซลได้
	Move	ใช้ในการย้าย (move) หรือเลื่อนทั้งภาพหรือวัตถุในแต่ละเฟรมของ animation. การใช้เครื่องมือนี้ช่วยให้คุณสามารถปรับตำแหน่งของวัตถุหรือภาพได้ตามต้องการ เพื่อสร้างการเคลื่อนไหวหรือแสดงผลลัพธ์ที่ต้องการใน animation
	Paint Bucket	ใช้ในการเติมสีลงบนพื้นที่ที่คุณคลิกเลือกในภาพ หรือในกรณีที่มีสีที่เหมือนกันหรือคล้ายกัน, Paint Bucket จะเติมสีลงบนพื้นที่ที่ปิดรอบกัน
	Line	ใช้ในการวาดเส้นตรงหรือเส้นต่อเนื่องระหว่างจุดที่คุณคลิก เครื่องมือนี้มีประโยชน์ในการสร้างขอบของรูปหรือในการวาดลายเส้นบนภาพ
	Rectangle	ใช้ในการวาดสี่เหลี่ยมหรือกล่องบนภาพ หรือในกรณีของ Aseprite, กล่องนี้จะมีขอบขนานกับแกน x และ y ของภาพ. เครื่องมือ Rectangle มีประโยชน์ในการสร้างขอบของภาพ, การสร้างรูปต่าง ๆ, หรือการทำ pixel art
	Contour	ใช้สำหรับการสร้างขอบของรูปร่าง หรือการเพิ่มขอบในรูปร่างที่มีอยู่แล้ว
	Blur	ใช้ในการทำให้ภาพหรือส่วนของภาพเบลอลง โดยลดความชัดของพิกเซลและทำให้สีหรือรูปร่างที่คุณเลือกมีลักษณะเนื้อหรือขอบที่นุ่มนวลขึ้น. การใช้เครื่องมือ Blur นี้มักจะเป็นประโยชน์ในการสร้างเอฟเฟคหรือทำให้ภาพดูนุ่มนวล

2.4 Storyboard (https://www.cotactic.com/blog/5-tricks-to-create-storyboard/)

การเขียนสตอรี่บอร์ด (Storyboard) คือ การนำเรื่องราวมาถ่ายทอดออกมาในรูปแบบภาพ โดยใช้ภาพวาด ภาพถ่าย หรือภาพกราฟิก เรียงต่อกันเป็นลำดับเพื่อบอกเล่าเรื่องราว มักใช้ในการผลิตภาพยนตร์ แอนิเมชั่น โฆษณา และวิดีโออื่นๆ สตอรี่บอร์ดช่วยให้ผู้สร้างสื่อสามารถวางแผนและสื่อสารเรื่องราวได้อย่างมีประสิทธิภาพ ช่วยให้เห็นภาพรวมของเรื่องราว ลำดับเหตุการณ์ มุมกล้อง และองค์ประกอบอื่นๆ ของสื่อได้อย่างชัดเจน นอกจากนี้ยังช่วยให้ผู้สร้างสื่อสามารถทดสอบไอเดียใหม่ๆ และปรับปรุงเรื่องราวได้ก่อนที่จะนำไปผลิตจริง

การเขียน Storyboard คือการเรียงลำดับภาพในความคิดออกมา ก่อนที่จะถ่ายจริง ซึ่งเขียนออกมาเป็นฉากเรียงลำดับ 1,2,3,.... ซึ่งทำให้เห็นภาพของเรื่องราวที่จะเล่า ปัญหาส่วนใหญ่ของคนที่จะเริ่มทำคือ ไม่รู้ว่าจะเริ่มจากตรงไหนมีองค์ประกอบอะไรที่ต้องใส่เข้าไปบ้าง



รูปภาพที่ 2.17 ตัวอย่าง Storyboard

2.4.1 เทคนิค วิธีการทำ Storyboard

1) เทคนิคหา KEY MESSAGE ให้โดนใจคนดู

- ทำ RESEARCH กับกลุ่มคนดู ว่าเขาเป็นใคร อายุเท่าไหร่ มีความชอบหรือสนใจ ซึ่ง Insight การทำ Storyboard เหล่านี้จะเป็นจุดที่คอยดึงคนเข้ามาดู Video ของเรามากขึ้น เช่น โฆษณาของ K PLUS ชื่อ FACE OFF นั้นหยิบ Insight ที่ว่าคนไทยส่วนใหญ่ไม่ชอบการเปลี่ยนแปลงจากสิ่งที่คุ้นเคย ซึ่ง แอป KPLUS ในตอนนั้นกำลังจะ Update ครั้งใหญ่ โฆษณา FACE OFF จึงสื่อสารว่าถึงหน้าตาจะเปลี่ยนไปแต่คุณภาพในการใช้งานยังเหมือนเดิม

- ปัญหา ของคนดู บางคนอาจจะไม่รู้ตัวว่ากำลังประสบปัญหาเรื่องเล็กน้อยในชีวิต เราต้องสังเกตพฤติกรรม เพื่อนำจุดนั้นมาทำ Key Message และตอบ Pain Point ของคนดูได้ เช่น โฆษณา AirPay ที่ได้จับ Pain Point “คนข้างหน้าเรื่องเยอะเสมอ” ทั้งๆที่เราอยากจะแค่จองตั๋วหนัง แต่ต้องมารอคนจ่ายบิลค่าน้ำค่าไฟ ซึ่งเป็น Pain Point ที่เจอได้ทุกวัน และ AirPay ก็นำเสนอตัวเองเป็น Solution ในการแก้ปัญหานี้

- ความแตกต่าง การบอกว่าเราแตกต่าง แปลกใหม่กว่าคนอื่นอย่างไร ก็เป็นอีกหนึ่ง Key Message ที่ใช้กันเยอะ เพราะถ้ายิ่งทำออกมาได้ชัดมากเท่าไหร่ ก็ยิ่งสร้างความน่าสนใจได้มากขึ้นเท่านั้น เช่น โฆษณาของ Sunsilk ชื่อ The Hair Talk ที่บอกเล่าเรื่องผ่าน LGBT กับครอบครัว ซึ่ง Sunsilk ใช้ความแตกต่างในเรื่องของยาสระผมที่คนส่วนใหญ่คิดว่าเป็นแบรนด์สำหรับผู้หญิงเท่านั้น

2) STORYTELLING

- เล่าตามเหตุการณ์ปกติ เป็นการเล่าเรื่อง จาก 1 ไป 2 ไป 3 ปกติ ซึ่งการเล่าเรื่องแบบนี้มีข้อดีคือ ทำให้คนเข้าใจเนื้อหาได้ง่าย การทำ Storyboard ไม่ต้องคิดเยอะ แต่ถ้าเนื้อหาของเรานั้นไม่ได้มีความน่าสนใจพอ ก็จะทำให้คนดูอาจจะเบื่อ ดูไม่จบ แต่เทคนิคนี้ถือว่าเป็นพื้นฐานที่ดี คือการเล่าเรื่องให้คนเข้าใจ ตัวอย่างโฆษณาของ Kleenex Thailand ชื่อ Tiny Doll ที่ใช้เทคนิคการเล่าตามเหตุการณ์ปกติ แต่เพิ่มความน่าสนใน Video ด้วยการถ่ายแบบ Long Take

- การเรียงลำดับเรื่องใหม่ การเล่าเรื่องแบบนี้ สามารถกระโดดข้ามไปมาได้ จะเอาตอนท้ายเรื่องมาไว้ต้นเรื่อง หรือ ต้นเรื่องไปไว้กลางเรื่องก็ได้ เทคนิคนี้จะมีความยืดหยุ่นในการเล่าเรื่อง เพราะมีความอิสระในการตีความใหม่ทั้งหมด แต่จะมีข้อเสียคือถ้าลำดับเรื่องไม่ดี ก็จะทำให้คนดูไม่เข้าใจเนื้อหาที่อยากจะสื่อไป โฆษณา My Beautiful Woman ของ Wacoal ใช้เทคนิคการเล่าเรื่องแบบตัดสลับกันในตอนท้าย

3) MOOD AND TONE

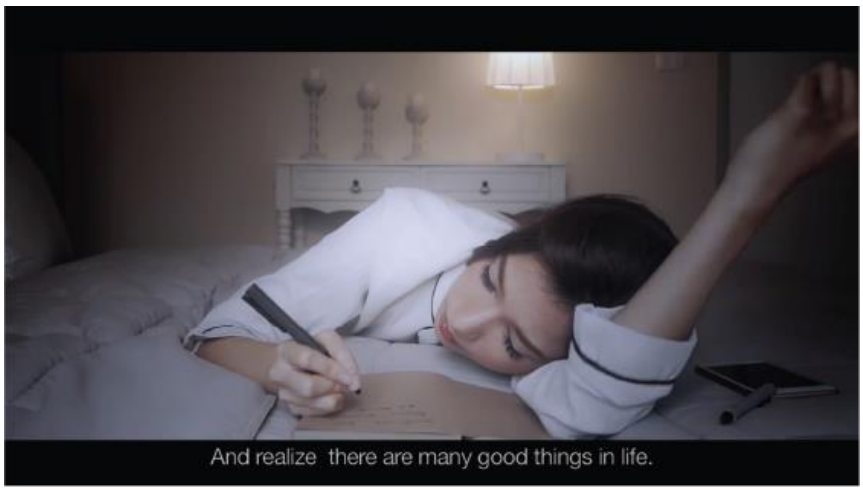
- MOOD คือ อารมรณ์ของ Video เช่น สนุกสนาน เศร้า ร่าเริง ความสงบ ประหลาดใจ โกรธ

- TONE คือ สีที่ใช้ในวีดีโอ โดยสีจะบอกความรู้สึกที่เรารู้กัน สีโทนเย็นหรือสีโทนร้อน ซึ่งขึ้นกับว่าต้องการจะสื่อออกมาในทิศทางไหน เช่น ถ้าอยากให้ภาพใน Video ดูสนุกสดใส อาจจะต้องใช้สีโทนร้อนเพื่อทำให้คนดูรู้สึกถึงความมีชีวิตชีวา ไม่หยุดนิ่ง แต่กลับกันเราอยากจะให้ Video รู้สึกน่ากลัว ลึกลับ อาจจะต้องใช้สีโทนเย็นเพื่อให้รู้สึกสุขุมขึ้น นิ่งขึ้น

4) SHOT REFERENCES

- เทคนิคการหา SHOT REFERENCES นั้นคือการเลือกภาพที่สื่อความหมายที่เราอยากสื่อมากที่สุด เพราะต่อให้เป็นท่าทางเดียวกัน แต่มุมกล้องต่างกันบริบทไม่เหมือนกัน ก็ทำให้อารมณ์ของภาพเปลี่ยนไป ซึ่งในส่วนนี้รวมถึงแสงเงาของภาพเข้าไปด้วย จากตัวอย่างจะเป็นฉากการเขียนสมุดบันทึก ซึ่งเป็นท่าทางเดียวกัน แต่ด้วยมุมกล้องและแสง ทำให้มีอารมณ์ที่ต่างกันสิ้นเชิง

รูปภาพที่ 2.18 ตัวอย่าง SHOT REFERENCES ที่ 1



รูปภาพที่ 2.19 ตัวอย่าง SHOT REFERENCES ที่ 2

- งานที่ควรนำมาเป็น REFERENCES การหา Shot Reference นั้นควรหาจากงาน โฆษณา / Music Video / หนัง / Filmmaker เพราะงานประเภททาง Production จะมีการคิดเรื่องแสง เงา ท่าทาง เพื่อให้ฉากสวยงามอยู่แล้ว เหมาะแก่การนำมาเป็น Reference ได้เลยโดยไม่ต้องไปคิดเอาเอง

5) TRANSITION ที่ใช้เพิ่มสีสันในงาน VIDEO

- CUT คือการเปลี่ยนจากภาพหนึ่งไปยังอีกภาพหนึ่งแบบทันทีทันใด โดยไม่ใช้เทคนิคใดๆ ในการเชื่อมต่อภาพ ลักษณะที่ปรากฏออกมาจึงเป็นภาพที่ต่อด้วยภาพ จะถ่ายทอดความรู้สึกฉับไว นอกจากนี้เรายังรับรู้สิ่งต่างๆ อย่างตรงไปตรงมา

- FADE คือการเปลี่ยนภาพ โดยภาพจะแทนที่ด้วยฉาก โดยการ Fade นี้สามารถแบ่งออกเป็น Fade In และ Fade Out

Fade In ใช้สำหรับการเริ่มต้นของเรื่องราว หรือเหตุการณ์

Fade Out คือภาพหลักจะค่อยๆ จางลงกลายเป็นภาพสีดำ

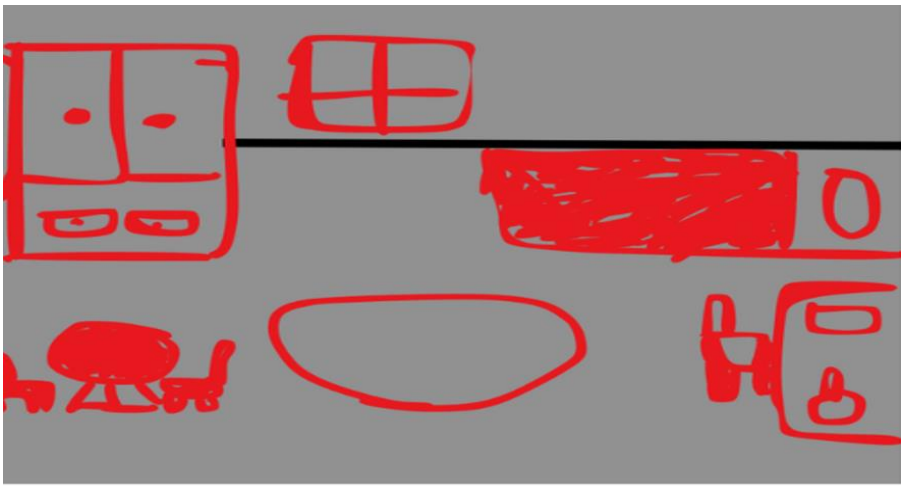
- DISSOLVE คือการเปลี่ยนภาพ โดยภาพแรกค่อยๆจางไปสู่อีกภาพ โดยที่การเปลี่ยนแปลงนั้นจะไม่ได้เกิดขึ้นแบบทันทีทันใดเหมือนการ Cut เทคนิคจะช่วยสร้างความรู้สึกที่ราบรื่น และนุ่มนวล โดยจะใช้เพื่อเชื่อมโยงเรื่องราว

- WIPE การปาด คือหน้าจอจะมีลักษณะของการปาดภาพที่ต้องการจะเปลี่ยน ทับลงบนภาพที่ปรากฏอยู่ การใช้ Wipe อาจสร้างความรู้สึกไม่สมจริงของเรื่องราว ซึ่งส่วนใหญ่จะถูกใช้เพื่อแบ่งเรื่องราวทั้งหมดออกเป็นส่วนๆ อย่างชัดเจน

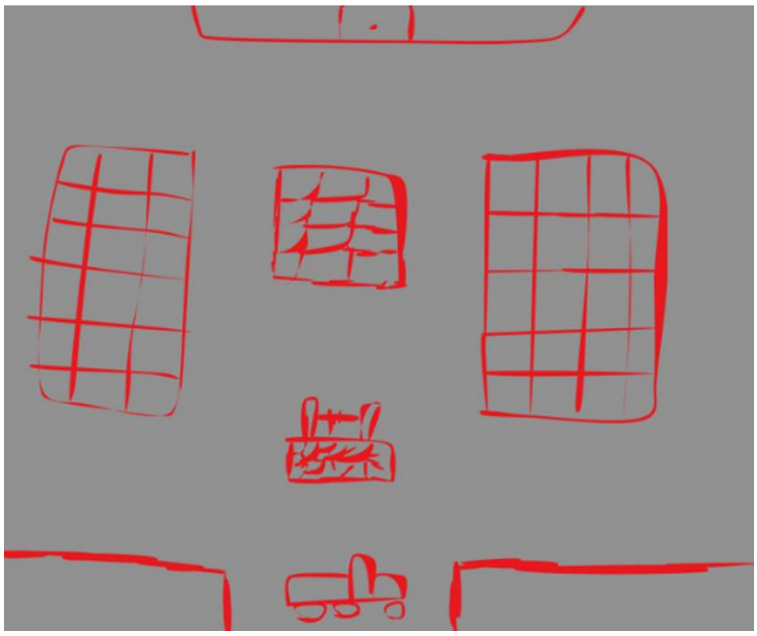
- DIGITAL EFFECT เป็นเทคนิค Transition แบบใหม่ ที่นำมาใช้กับงานที่เน้นความน่าสนใจของภาพ เช่น งานโฆษณา หรือ Motion Graphics เทคนิคช่วยเพิ่มความน่าสนใจให้กับงาน หากเป็นงานประเภทที่ต้องการดึงดูดความสนใจสูง

2.4.2 Storyboard ของ Project

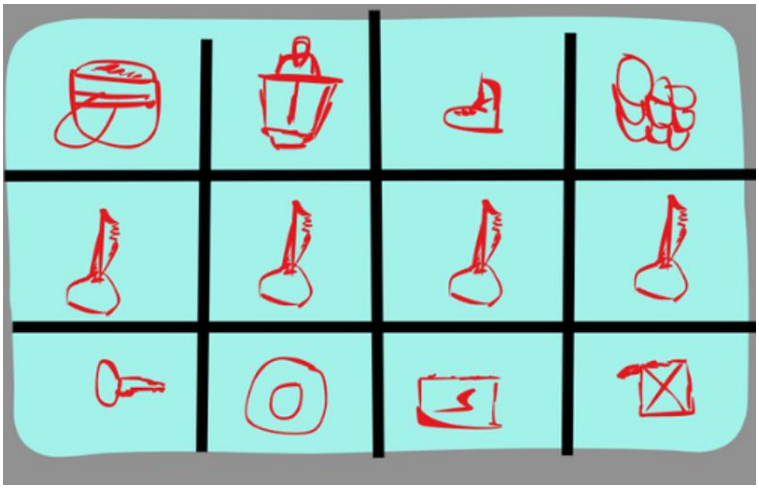
รูปภาพที่ 2.20 หน้าเมนูเกม



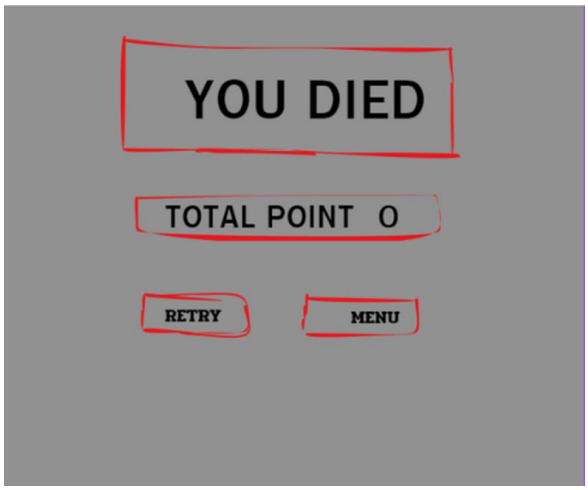
รูปภาพที่ 2.21 หน้าภายในบ้าน



รูปภาพที่ 2.22 ภายนอกบ้าน



รูปภาพที่ 2.23 หน้าต่างระบบไอเทม



รูปภาพที่ 2.24 หน้าต่างผู้เล่นตาย

รูปภาพที่ 2.25 หน้าต่างผู้เล่นชนะ

2.5 เอกสารและงานวิจัยที่เกี่ยวข้อง

นายนัฐวุฒิ หวลละห้อย นายภูวนาท แสงฤทธิ์ และนายรัฐพล สุขสว่างการจัดทำโครงการครั้งนี้มีวัตถุประสงค์เพื่อสร้างเกมสามมิติผีในฝัน ด้วยโปรแกรม Unity (Ghost in Dream 3D Game) ศึกษาความพึงพอใจ ของนักเรียนประกาศนียบัตรวิชาชีพ สาขาวิชาคอมพิวเตอร์ธุรกิจวิทยาลัยเทคนิคระยองที่มีต่อเกมสามมิติ ในฝันด้วยโปรแกรม Unity (Ghost in Dream 3D Game) และเผยแพร่เกมสามมิติผีในฝัน ด้วยโปรแกรม Unity (Ghost in Dream 3D Game) ผ่านโครงการประกวดโครงการวิชาชีพ ขมรมวิชาชีพคอมพิวเตอร์ธุรกิจ กลุ่มตัวอย่างที่ใช้คือนักเรียนประกาศนียบัตรวิชาชีพ สาขาคอมพิวเตอร์ธุรกิจ วิทยาลัยเทคนิคระยอง จำนวน 59 คนเครื่องมือที่ใช้ในการวิเคราะห์ข้อมูล ได้แก่ เกมสามมิติผีในฝันด้วยโปรแกรม Unity (Ghost in Dream 3D Game) และแบบสอบถาม ความพึงพอใจของนักเรียนประกาศนียบัตรวิชาชีพ สาขาคอมพิวเตอรธ์รุกิจ วิทยาลัยเทคนิคระยอง ที่มีต่อเกมสามมิติในฝันด้วยโปรแกรม Unity (Ghost in Dream 3D Game) สถิติที่ใช้ในการวิเคราะห์ข้อมูล คือ ค่าสถิติร้อยละ ค่าเฉลี่ย และส่วนเบี่ยงเบนมาตรฐาน

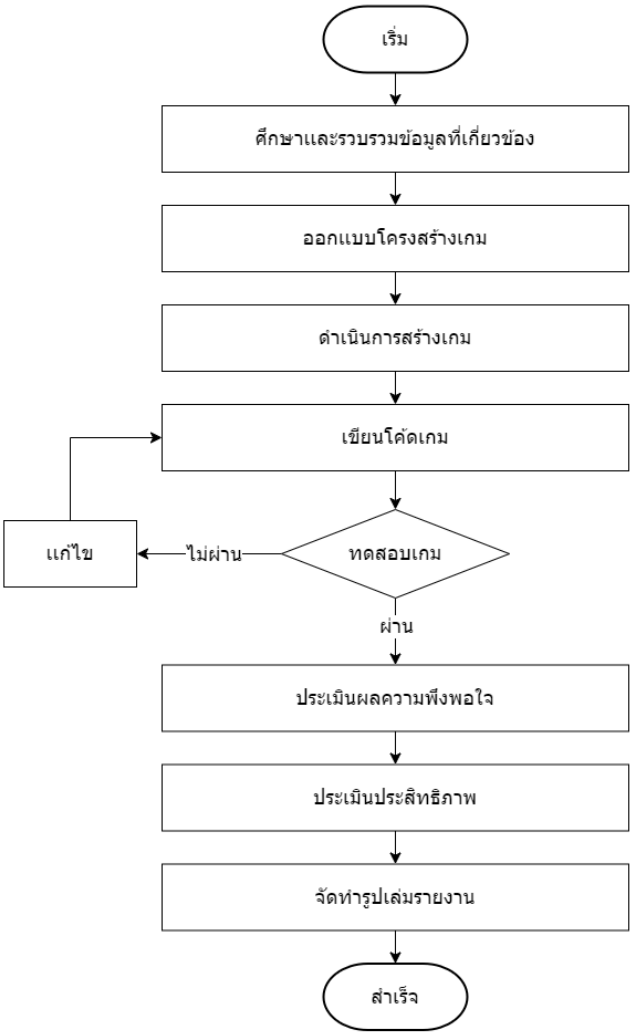
อภินันท์ ภิรมย์ ภาคย์ สธนเสาวภาคย์ และภัทราวัลย์ คำปลิว งานวิจัยนี้มีวัตถุประสงค์เพื่อ 1) พัฒนาเกมสามมิติเรื่อง เกมทางออกอยู่ไหน 2) ศึกษาความพึงพอใจของนักศึกษา ที่ มีต่อเกมสามมิติ เรื่อง เกมทางออกอยู่ไหน กลุ่มเป้าหมาย นักศึกษาสาขาวิศวกรรมคอมพิวเตอร์และสาขาเครือข่ายคอมพิวเตอร์คณะวิศวกรรมศาสตร์ มหาวิทยาลัยราชภัฏมหาสารคาม จำนวน 80 คน เครื่องมือที่ใช้ในการวิจัย ได้แก่ แบบสอบถามความพึงพอใจของนักศึกษา สถิติที่ใช้ในการวิจัย คือ ค่าเฉลี่ย และส่วนเบี่ยงเบนมาตรฐานผลการวิจัยสรุปได้ดังนี้ 1) ได้เกมสามมิติเรื่อง เกมทางออกอยู่ไหน ประกอบด้วยเกาะในเกมจำนวน 3 เกาะ โดยแต่ละเกาะจะมีมอนสเตอร์และความยากแตกต่างกันออกไป โดยเกาะที่ 1 จะง่ายที่สุด และมีมอนสเตอร์คือ ซอมบี้สำหรับ มุมมองในเกมจะเป็นมุมมองของบุคคลที่หนึ่ง ผู้วิจัยได้นำมอนสเตอร์จำนวน 4 ชนิดที่มาจากใน Unity 3D คือ มอนสเตอร์ เอเลี่ยน มอนสเตอร์กบกลายพันธุ์ ซอมบี้ และก็อบบิน 2) นักศึกษาสาขาวิศวกรรมคอมพิวเตอร์และสาขาเครือข่ายคอมพิวเตอร์คณะวิศวกรรมศาสตร์ มหาวิทยาลัยราชภัฏมหาสารคาม มีความพึงพอใจต่อเกมสามมิติเรื่องเกมทางออกอยู่ไหนโดยรวมอยู่ในระดับมาก

รัชพล วรรณวิจิตร โครงงานนี้ทำขึ้นมาโดยมีจุดมุ่งหมายเพื่อโปรโมทสถาบันเทคโนโลยีไทย-ญี่ปุ่น และเพื่อ เป็นแนวทางสำหรับผู้ที่มีความสนใจ ต้องการศึกษาเกี่ยวกับการสร้างเกม 3D โดยแบ่งเป็นขั้นตอนการเลือกอาคารที่จะนำมาใช้ การจัดวางองค์ประกอบให้ดูน่ากลัว การทำ NavMesh ให้ NPC เดิน การจัดการกับแสงเงา และความรู้เบื่องต้นเกี่ยวกับการ optimize ตัวเกม และสามารถนำเทคนิคจาก โครงงานนี้ไปใช้ต่อยอดเพื่อพัฒนาผลงานตัวเองได้

บทที่ 3

วิธีการดำเนินงาน

ในการจัดทำโครงการ Plant & Flee คณะผู้จัดทำได้ดำเนินการตามขั้นตอนดังนี้



รูปภาพที่ 3.1 แผนผังแสดงการดำเนินโครงการ

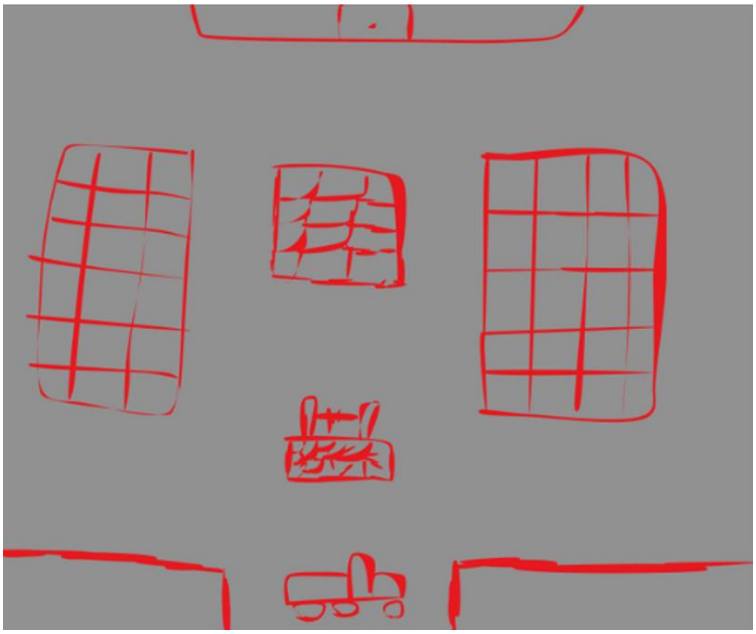
3.1 ศึกษาและรวบรวมข้อมูลที่เกี่ยวข้อง

ดำเนินการศึกษาและรวบรวมข้อมูลที่เกี่ยวข้องของ โปรแกรมที่ใช้ในการจัดทำโครงการในครั้งนี้ Unity คือ โปรแกรมที่มีความสามารถหลากหลายได้แก่สร้างเกม 2 มิติการสร้างเกม 3 มิติการสร้าง AR, VR สามารถส่งแอพพลิเคชั่นได้ทั้งระบบ Windows, iOS และ Android ภาษา C# คือ ภาษาเขียนโปรแกรมแบบ multi-paradigm ซึ่งมีรูปแบบภาษาที่ตายตัว และเป็นรูปแบบบังคับในการเขียน มีฟังก์ชันและยังเป็นภาษาการเขียนโปรแกรมที่มีคุณสมบัติเป็นแบบออบเจ็คด้วย Aseprite คือ โปรแกรมแก้ไขรูปภาพที่เป็นกรรมสิทธิ์และมีแหล่งที่มา ซึ่งออกแบบมาเพื่อการวาดภาพแบบพิกเซลอาร์ตและแอนิเมชั่นเป็นหลัก จึงได้ทำการค้นคว้าข้อมูลต่างๆ ที่เกี่ยวข้อง เพื่อนำมาใช้ในการจัดทำโครงงาน

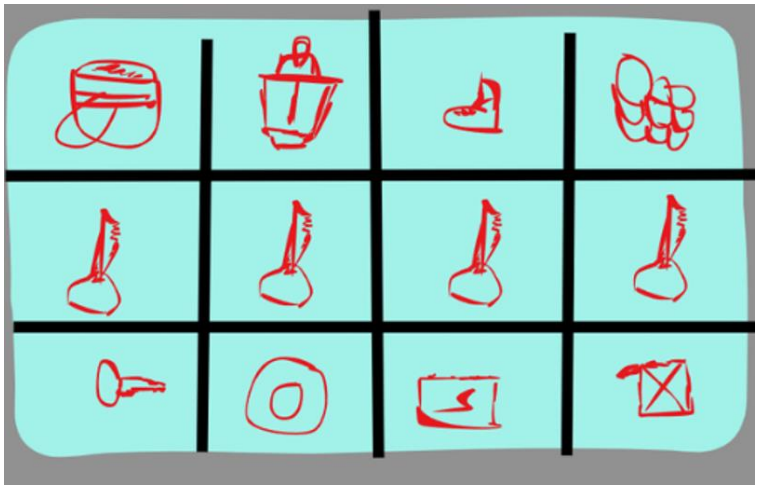
3.2 เขียน Storyboard

3.2.1 Storyboard ของ Project

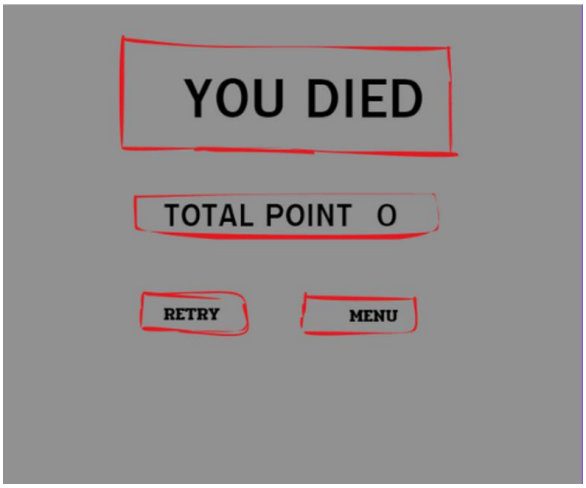
รูปภาพที่ 3.2 ภาพหน้าเมนูเกม



รูปภาพที่ 3.3 ภายนอกบ้าน



รูปภาพที่ 3.4 หน้าต่างระบบไอเทม

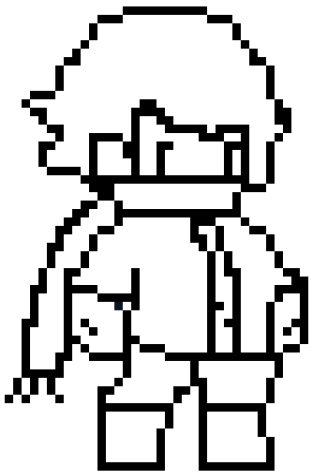


รูปภาพที่ 3.5 หน้าต่างเมื่อผู้เล่นแพ้

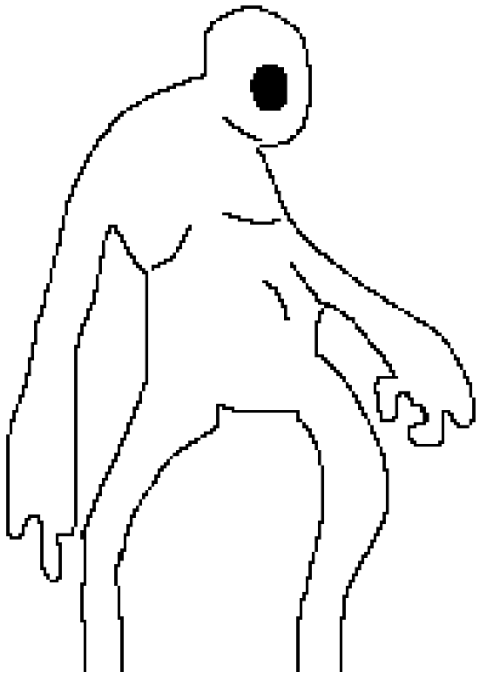
รูปภาพที่ 3.6 หน้าต่างเมื่อผู้เล่นชนะ

3.3 ออกแบบโครงสร้างเกม

3.3.1 ออกแบบตัวละคร



รูปภาพที่ 3.7 ออกแบบตัวละครหลัก



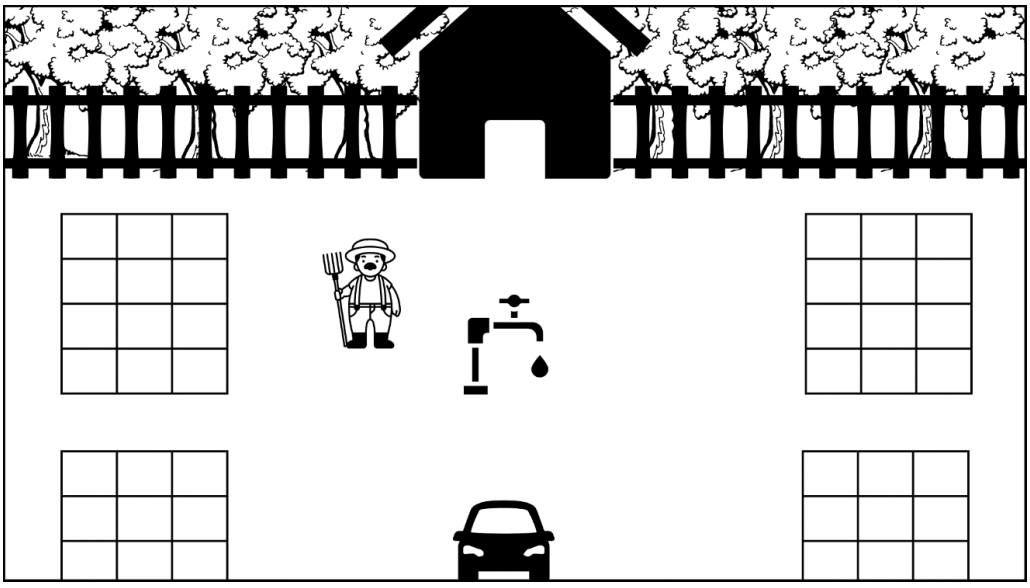
รูปภาพที่ 3.8 ออกแบบ Monster ตัวที่ 1 เดินซ้า

รูปภาพที่ 3.9 ออกแบบ Monster ตัวที่ 2 เดินเร็ว

3.3.2 ออกแบบหน้าเมนูของเกม

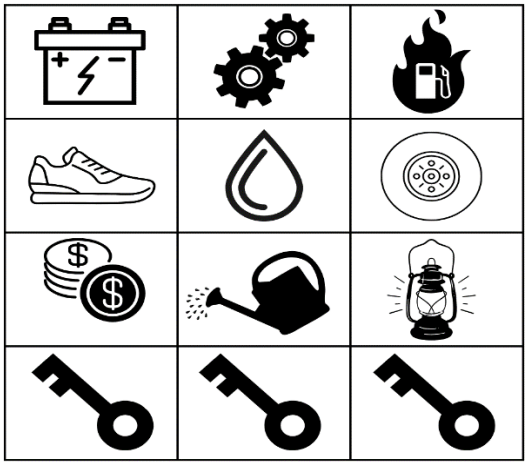
รูปภาพที่ 3.10 ออกแบบหน้าเมนู

3.3.3 ออกแบบรูปแบบการเล่นเกม



รูปภาพที่ 3.11 ออกแบบรูปแบบการเล่นเกม

3.3.4 ออกแบบระดับแลกไอเทม



รูปภาพที่ 3.12 ออกแบบระบบแลกไอเทม

3.3.5 ออกแบบหน้าต่างเมื่อผู้เล่นแพ้

รูปภาพที่ 3.13 ออกแบบหน้าต่างเมื่อผู้เล่นแพ้

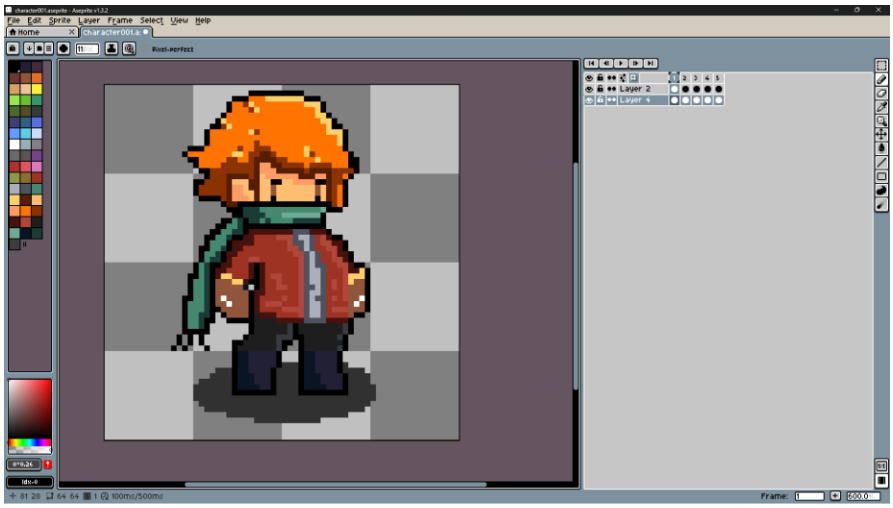
3.3.6 ออกแบบหน้าต่างเมื่อผู้เล่นชนะ

รูปภาพที่ 3.14 ออกแบบหน้าต่างเมื่อผู้เล่นชนะ

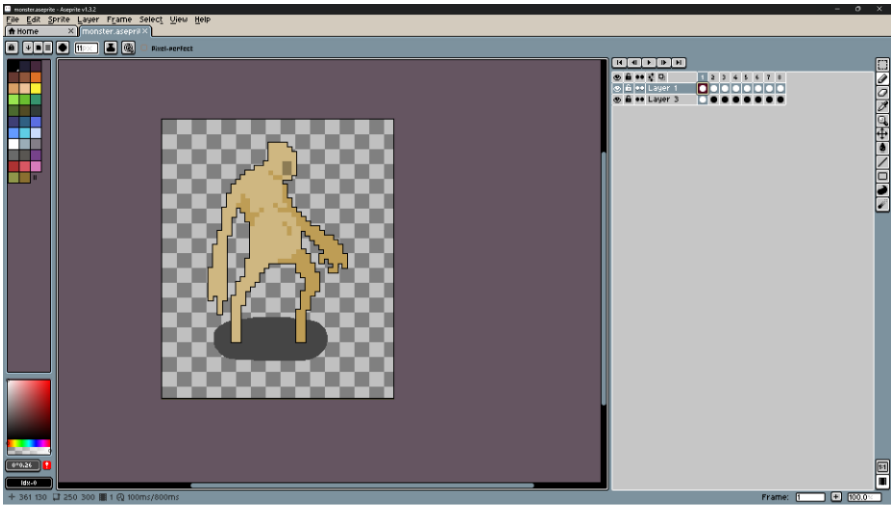
3.4 ดำเนินการสร้างเกม

3.4.1 สร้างตัวละคร

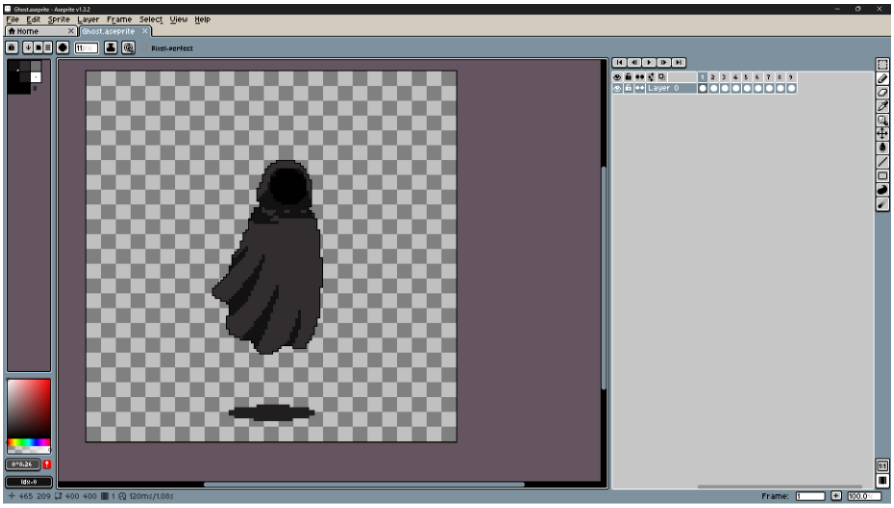
รูปภาพที่ 3.15 สร้างตัวละคร



รูปภาพที่ 3.16 สร้างตัวละครหลัก

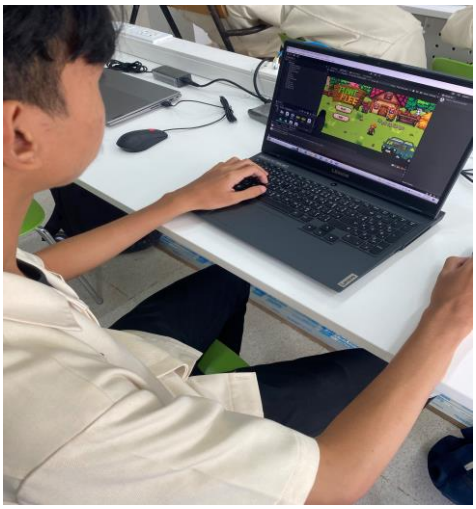


รูปภาพที่ 3.17 สร้าง Monster ตัวที่ 1 เดินช้า

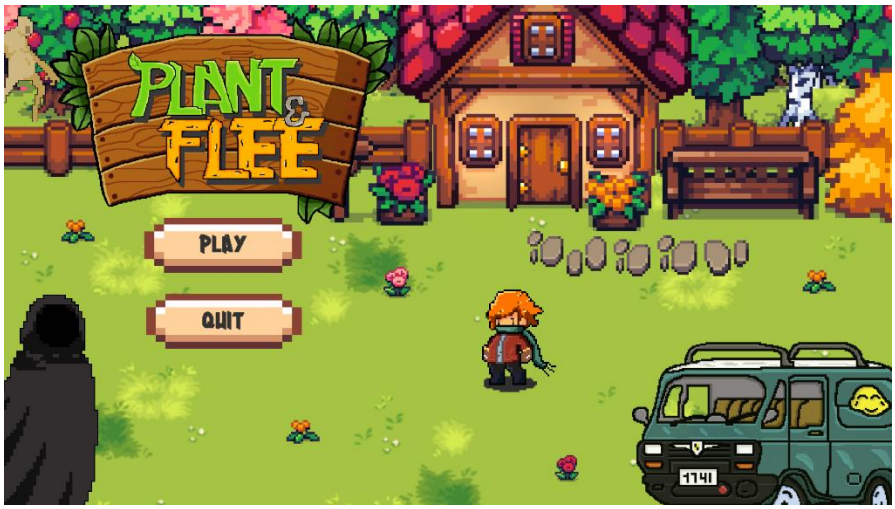


รูปภาพที่ 3.18 สร้าง Monster ตัวที่ 2 เดินเร็ว

3.4.2 สร้างหน้าเมนูของเกม



รูปภาพที่ 3.19 สร้างหน้าเมนูของเกม



รูปภาพที่ 3.20 หน้าเมนูของเกม PLANT & FLEE

3.4.3 ทำระบบการเล่นเกม

รูปภาพที่ 3.21 คู่มือการเล่น

รูปภาพที่ 3.22 ระบบในบ้าน

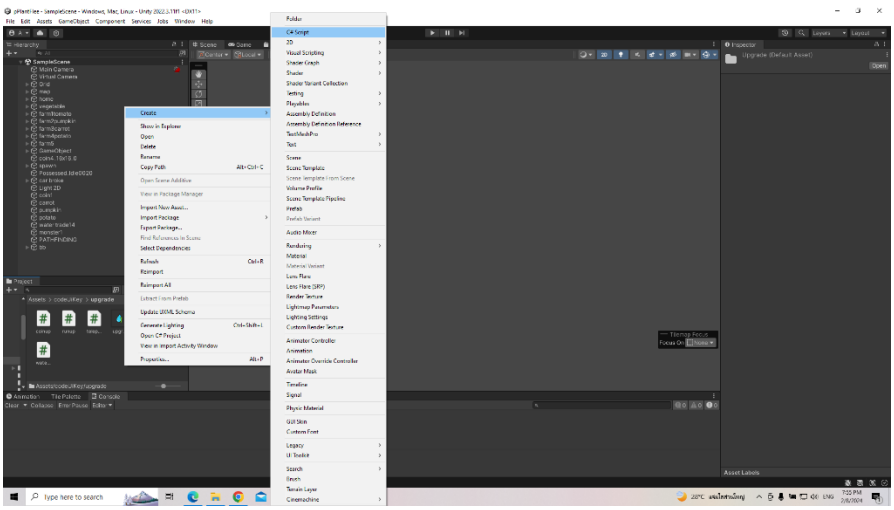
รูปภาพที่ 3.23 ระบบแลกไอเทม

รูปภาพที่ 3.24 ระบบการเติมน้ำ

รูปภาพที่ 3.25 ระบบการปลูก

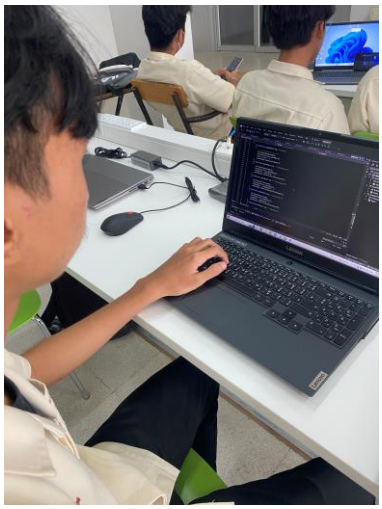
3.5 การเขียนโค้ด

3.5.1 สร้างไฟล์ C# Script ขึ้นมา เพื่อป้อนโค้ดเข้าสู่เกม

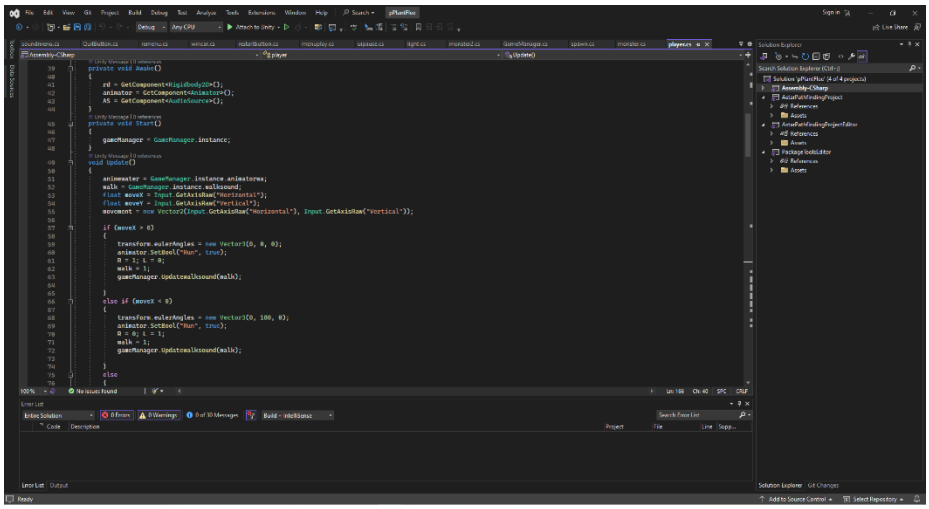


รูปภาพที่ 3.26 การสร้างไฟล์ C# Script

3.5.2 เขียนโค้ด



รูปภาพที่ 3.27 เขียนโค้ดเกม



รูปภาพที่ 3.28 หน้าต่างเขียนโค้ดเกม

3.6 ทดสอบการทำงาน

3.6.1 หลังจากเข้าสู่เกมเป็นที่เรียบร้อยแล้ว ให้กดปุ่ม Play เพื่อเริ่มเกม

3.6.2 จากนั้นตัวเกมจะแสดงตัวละครที่แมพ และจะทำการให้ผู้เล่นทำการปลูกผักเพื่อเอาตัวรอดออกจากแมพ ด้วยการนำ Point ที่ได้จากการปลูกผักมาแลกไอเทม และชิ้นส่วนรถยนต์เพื่อที่ชนะเกม โดยทุกตอนกลางคืนจะมี Monster มาก่อกวนผู้เล่น

3.6.3 หากผู้เล่นโดน Monster จะทำให้แพ้ทันที และต้องเริ่มเล่นใหม่เท่านั้น

3.7 การประเมินความพึงพอใจ

แบบประเมินความพึงพอใจของผู้เล่นจากการเล่น PLAN & FLEE เป็นแบบมาตราส่วนประมาณค่า 5 ระดับ ของลิเคิร์ท(Likert) มีคำถามจำนวน 5 ข้อ โดยกำหนดระดับความคิดเห็นแต่ละช่วงคะแนนและความหมายดังนี้

ระดับ 1 หมายถึง ปรับปรุง

ระดับ 2 หมายถึง พอใช้

ระดับ 3 หมายถึง ปานกลาง

ระดับ 4 หมายถึง ดี

ระดับ 5 หมายถึง ดีมาก

สำหรับการให้ความหมายของค่าที่วัดได้ ผู้วิจัยได้กำหนดเกณฑ์ที่ใช้ในการให้ความหมายโดยได้จากแนวคิดของเบสท์ (Best 1986) การให้ความหมายโดยการให้ค่าเฉลี่ย ดังนี้

1.00 - 1.49 หมายถึง ปรับปรุง

1.50 - 2.49 หมายถึง พอใช้

2.50 - 3.49 หมายถึง ปานกลาง

3.50 - 4.49 หมายถึง ดี

4.50 - 5.00 หมายถึง ดีมาก

ผู้ทดลองใช้งานจะได้ประเมินคุณภาพของโปรแกรม โดยใช้แบบสอบถามเพื่อประเมินความคิดเห็น

ผลที่ได้จากการประเมินของผู้ทดลองใช้งานหลักการทางสถิติเข้ามาช่วยในการสรุปการประเมินเกมที่ได้สร้างขึ้น โดยคำนวณหาค่าเฉลี่ยและค่าเบี่ยงเบนมาตรฐานของระดับความติดในแต่ละด้าน

สูตรหาค่าเฉลี่ย

3.7.1 ค่าเฉลี่ย โดยใช้สูตรค่าเฉลี่ย

$$\bar{x} = \frac{\sum x}{n}$$

เมื่อ $\bar{x}$ แทน ค่าคะแนนเฉลี่ย

$\sum x$ แทน ผลรวมของคะแนนทั้งหมด

n แทน จำนวนครั้งที่ทดสอบ

3.7.2 ส่วนเบี่ยงเบนมาตรฐานโดยใช้สูตร

$$S.D. = \frac{\sqrt{n\sum x^2 - (\sum x)^2}}{n(n-1)}$$

เมื่อ $S.D.$ แทน ส่วนเบี่ยงเบนค่ามาตราฐาน

x แทน คะแนนของแต่ละครั้ง

n แทน จำนวนครั้งที่ทดสอบ

บทที่ 4

ผลการทดลอง

โครงงานครั้งนี้ คณะผู้จัดทำได้ทำการสร้างเกม PLAN & FLEE มีการเก็บรวบรวมข้อมูลผลการทดลองดังนี้

4.1 ผลการทดลองเล่นเกม PLANT & FLEE

4.2 ผลการประเมินหาคุณภาพของเกม PLANT & FLEE

4.1 ผลการทดลองเล่นเกม PLANT & FLEE

การทดลองเปิดใช้งานครั้งที่ 1

ตัวละครตกแมพ

การดำเนินการแก้ไข ปรับ Gravity Scale เป็น 0

การทดลองเปิดใช้งานครั้งที่ 2

ตัวละครเดินทะลุแมพ

การดำเนินการแก้ไข กำหนด Box Collider 2D

การทดลองเปิดใช้งานครั้งที่ 3

ตัวละครหมุดเมื่อเดินชน

การดำเนินการแก้ไข ติ๊กถูกที่ Freeze Rotatione Z ใน Rigidbody2D

การทดลองเปิดใช้งานครั้งที่ 4

ตัวละครไม่สารมารถหันหน้าตรงข้ามได้

การดำเนินการแก้ไข ใช้ transform.eulerAngles = new Vector3(0, 180, 0);

การทดลองเปิดใช้งานครั้งที่ 5

ตัวละครไม่มีเสียง

การดำเนินการแก้ไข เพิ่ม Audio Source ให้ตัวละกำหนดเสียงเองได้

การทดลองเปิดใช้งานครั้งที่ 6

ตัวละครเก็บ coin ไม่ได้

การดำเนินการแก้ไข กำหนด Tag coin ให้เหรียญ

การทดลองเปิดใช้งานครั้งที่ 7

ตัวละครไม่สามารถตายได้เมื่อมอนสเตอร์เข้าใกล้

การดำเนินการแก้ไข เมื่อมอนสเตอร์เข้าใกล้ Tag player ให้ Destroy();

การทดลองเปิดใช้งานครั้งที่ 8

ระยะเวลาผักโตไม่เท่ากันขึ้นอยู่กับความเร็วของเครื่องคอมพิวเตอร์

การดำเนินการแก้ไข ใช้ Time.deltatime เพื่อให้ผักโตเท่ากัน

การทดลองเปิดใช้งานครั้งที่ 9

มอนสเตอร์เดินติด object ภายในแมพ

การดำเนินการแก้ไข ใช้ Pathfinding เพื่อให้มอนสเตอร์ค้นหาเส้นทางเองเพื่อตามผู้เล่น

การทดลองเปิดใช้งานครั้งที่ 10

แมพไม่สามารถมืดและสว่างได้

การดำเนินการแก้ไข ใช้ Global light 2D เพื่อให้เกมมีระบบกลางวันและกลางคืน

4.2 ผลการประเมินหาคุณภาพของเกม PLANT & FLEE

จากการทดลองเกมโดยการมีผู้ใช้งาน จำนวน 10 ท่าน เป็นคุณภาพของเกมผลการทดสอบแสดงให้เห็นว่าเมื่อนำคะแนนเฉลี่ยของแต่ละหัวข้อมาผ่านระเบียบวิธีการทางสถิติเพื่อหาค่าเฉลี่ย (Mean) จะพบว่าค่าเฉลี่ยที่ได้อยู่ที่ 4.54 ดังนั้นเกมที่ได้ทการสร้างขึ้นมาอยู่ในระดับ ดีมาก

ตารางที่ 4.1 ผลการประเมินความพึงพอใจจากผู้ใช้งาน

ผลการผระเมินความพึงพอใจ	ค่าเฉลี่ย($\bar{X}$)	ค่าเบี่ยงเบนมาตรฐาน ($S.D.$)	ระดับความคิดเห็น
1.ออกแบบสวยงาม	4.7	0.48	ดีมาก
2.ใช้งานง่าย ไม่ซับซ้อน	4.6	0.51	ดีมาก
3.มีความน่าใช้งาน	4.5	0.52	ดีมาก

ผลการผระเมินความพึงพอใจ	ค่าเฉลี่ย($\bar{X}$)	ค่าเบี่ยงเบนมาตรฐาน ($S.D.$)	ระดับความคิดเห็น
4.ความเหมาะสมของเวลาในการเล่น	4.2	0.79	ดี
5.ประโยชน์ที่ได้จากเกม	4.7	0.48	ดีมาก
ค่าเฉลี่ยรวม	4.54	0.56	ดีมาก

จากตารางที่ 4.1 เป็นการประเมินความพึงพอใจของเกม PLANT & FLEE การออกแบบสวยงามมีค่าเฉลี่ยสูงสุดอยู่ที่ $\bar{X}$ = 4.7 $S.D.$ = 0.48 อยู่ในเกณฑ์ระดับดีมาก ต่อมาการใช้งานง่ายไม่ซับซ้อนมีค่าเฉลี่ยสูงสุดอยู่ที่ $\bar{X}$ = 4.6 $S.D.$ = 0.51 อยู่ในเกณฑ์ระดับดีมาก มีความน่าใช้งานมีค่าเฉลี่ยสูงสุดอยู่ที่ $\bar{X}$ = 4.5 $S.D.$ = 0.52 อยู่ในเกณฑ์ระดับดีมาก ความเหมาะสมของเวลาในการเล่นมีค่าเฉลี่ยสูงสุดอยู่ที่ $\bar{X}$ = 4.2 $S.D.$ = 0.79 อยู่ในเกณฑ์ระดับดี และสุดท้ายประโยชน์ที่ได้จากเกมมีค่าเฉลี่ยสูงสุดอยู่ที่ $\bar{X}$ = 4.2 $S.D.$ = 0.79 อยู่ในเกณฑ์ระดับดีมาก โดยรวมแล้วมีความพึงพอใจเฉลี่ยอยู่ที่ $\bar{X}$ = 4.54 $S.D.$ = 0.56 และค่าเบี่ยงเบนมาตรฐานใกล้เคียงกับศูนย์ ซึ่งอยู่ในระดับ ดีมาก

4.3 ผลการประเมินประสิทธิภาพของเกม PLANT & FLEE

จากการการศึกษาการทดลองประสิทธิภาพของ PLANT & FLEE จึงมีการบันทึกผลการทดลองในตารางหาประสิทธิภาพดังตาราง ที่ 4.2 ดังนี้

ตารางที่ 4.2 ผลการประเมินประสิทธิภาพของเกม PLANT & FLEE

แบบประเมินประสิทธิภาพ	การทดสอบครั้งที่										ร้อยละ
	1	2	3	4	5	6	7	8	9	10	
1.ส่วนเนื้อหา											
1.1 ความเด่นชัดของตัวละคร	✓	✓	✓	✓	×	✓	✓	✓	×	✓	80
1.2 ความเข้าใจง่ายของเป้าหมายของเกม	×	✓	✓	✓	✓	×	✓	✓	✓	✓	80
1.3 ความเข้าใจง่ายของเหตุการณ์	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	90
2.ส่วนอุปกรณ์											
2.1 ความเข้าใจง่ายของหน้าจอแสดงผล	✓	×	×	✓	✓	✓	✓	✓	✓	✓	80
2.2 ความดึงดูดใจของดนตรีประกอบ	✓	✓	×	✓	✓	✓	✓	✓	×	✓	80

แบบประเมินประสิทธิภาพ	การทดสอบครั้งที่										ร้อยละ
	1	2	3	4	5	6	7	8	9	10	
3. ส่วนของการเล่น											
3.1 ความครบถ้วนของคำอธิบายวิธีการเล่น	✓	✓	✓	✓	×	✓	✓	✓	✓	✓	90
3.2 ความเข้าใจง่ายของระบบของเกม	✓	✓	✓	×	✓	✓	✓	✓	✓	✓	90
รวมทั้งหมด (ร้อยละ)											84.3

จากตารางที่ 4.2 เป็นการประเมินประสิทธิภาพของเกม PLANT & FLEE ดังนี้ ส่วนที่ 1 ส่วนเนื้อหาสูงสุดอยู่ที่ $\bar{X}$ = 83.3% $S.D.$ = 0.38 ซึ่งอยู่ในเกณฑ์ระดับที่ 5 คือ ดีมาก ส่วนที่ 2 ส่วนอุปกรณ์สูงสุดอยู่ที่ $\bar{X}$ = 80% $S.D.$ = 0.41 ซึ่งอยู่ในเกณฑ์ระดับที่ 5 คือ ดีมาก และส่วนที่ 3 ส่วนของการเล่นสูงสุดอยู่ที่ $\bar{X}$ = 90% $S.D.$ = 0.30 ซึ่งอยู่ในเกณฑ์ระดับที่ 5 คือ ดีมาก ประสิทธิภาพของเกม PLANT & FLEE โดยรวมมีค่าเฉลี่ยอยู่ที่ร้อยละ $\bar{X}$ = 84.3 $S.D.$ = 0.36 ซึ่งอยู่ในเกณฑ์ระดับที่ 5 คือ ดีมาก

บทที่ 5

สรุปผล ปัญหาอุปสรรค และข้อเสนอแนะ

5.1 สรุปผล

โครงการนี้มีวัตถุประสงค์เพื่อสร้างเกม PLANT & FLEE ให้สำเร็จลุล่วง เพื่อให้เสริมสร้างทักษะในการวางแผนจัดการทรัพยากร และการเอาตัวรอดจากอุปสรรคต่างๆที่ไม่เหมือนกันส่งผลให้เกิดระยะเวลาจบเกมที่ต่างกันขึ้นอยู่กับการจัดการทรัพยากรภายในเกม

เกม PLANT & FLEE สามารถให้ผู้เล่นวางแผนจัดการทรัพยากรและเอาตัวรอดในทุกๆคืนขึ้นอยู่กับการวางแผนของผู้เล่นเพื่อจบเกม

จากการประเมินความพึงพอใจโดยมีผู้ประเมิน จำนวน 10 ท่าน สรุปได้ว่าเกม PLANT & FLEE มีคุณภาพอยู่ในระดับดี สามารถใช้งานในขอบเขตที่กำหนดไว้

5.2 ปัญหาและอุปสรรค

5.2.1 ระยะเวลาผักโตไม่เท่ากันขึ้นอยู่กับความเร็วของเครื่องคอมพิวเตอร์

5.2.2 มอนสเตอร์เดินติด object ภายในแมพ

5.3 แนวทางแก้ไข

5.3.1 ใช้ Time.deltatime เพื่อให้ผักโตเท่ากันไม่ว่าเครื่องคอมพิวเตอร์จะมีประสิทธิภาพเท่าใดก็ตาม

5.3.2 ใช้ Pathfinding เพื่อให้มอนสเตอร์ค้นหาเส้นทางเองเพื่อตามผู้เล่น

5.4 ข้อเสนอแนะ

5.4.1 ควรเพิ่มหน้า setting

5.4.2 ควรเพิ่มมอนสเตอร์มากกว่านี้

บรรณานุกรม

ปุจฉา วิสัชนา. (2561). **Unity.** สืบค้น 28 มกราคม 2567
จาก https://th.wikipedia.org/wiki/ยูนิตี_(เกมเอนจิน)

ภาษา C#. (2559). **ภาษา C#.** สืบค้น 28 มกราคม 2567
จาก https://marcuscode.com/lang/csharp/introduction

Igara Studio S.A. (2544). **Aseprite.** สืบค้น 28 มกราคม 2567
จาก https://en.wikipedia.org/wiki/Aseprite

การเขียนสตอรี่บอร์ด (STORYBOARD). (2563). **STORYBOARD.** สืบค้น 30 มกราคม 2567
จาก https://www.cotactic.com/blog/5-tricks-to-create-storyboard/

ภาคผนวก

ภาคผนวก ก.

แบบขออนุมัติโครงการ

ภาคผนวก ข.

ผลการประเมินจากผู้ใช้งาน

แบบประเมินประสิทธิภาพของเกม PLANT & FLEE

ผลการทำและหาคุณภาพของเกม PLANT & FLEE

แบบประเมินประสิทธิภาพ	การทดสอบครั้งที่										ร้อยละ
	1	2	3	4	5	6	7	8	9	10	
1.ส่วนเนื้อหา											
1.1 ความเด่นชัดของตัวละคร											
1.2 ความเข้าใจง่ายของเป้าหมายของเกม											
1.3 ความเข้าใจง่ายของเหตุการณ์											
2.ส่วนอุปกรณ์											
2.1 ความเข้าใจง่ายของหน้าจอแสดงผล											
2.2 ความดึงดูดใจของดนตรีประกอบ											
3. ส่วนของการเล่น											
3.1 ความครบถ้วนของคำอธิบายวิธีการเล่น											
3.2 ความเข้าใจง่ายของระบบของเกม											
รวมทั้งหมด (ร้อยละ)											

ผลความพึงพอใจของการใช้เกม PLANT & FLEE

ผลการผระเมินความพึงพอใจ	ค่าเฉลี่ย($\bar{X}$)	ค่าเบี่ยงเบนมาตรฐาน (S.D.)	ระดับความคิดเห็น
1.ออกแบบสวยงาม	4.7	0.48	ดีมาก
2.ใช้งานง่าย ไม่ซับซ้อน	4.6	0.51	ดีมาก
3.มีความน่าใช้งาน	4.5	0.52	ดีมาก
4.ความเหมาะสมของเวลาในการเล่น	4.2	0.79	ดี
5.ประโยชน์ที่ได้จากเกม	4.7	0.48	ดีมาก
ค่าเฉลี่ยรวม	4.54	0.56	ดีมาก

ภาคผนวก ค.

คู่มือการใช้งาน เกม PLANT & FLEE

คู่มือการใช้งาน และควบคุมต่างๆของเกม PLANT & FLEE

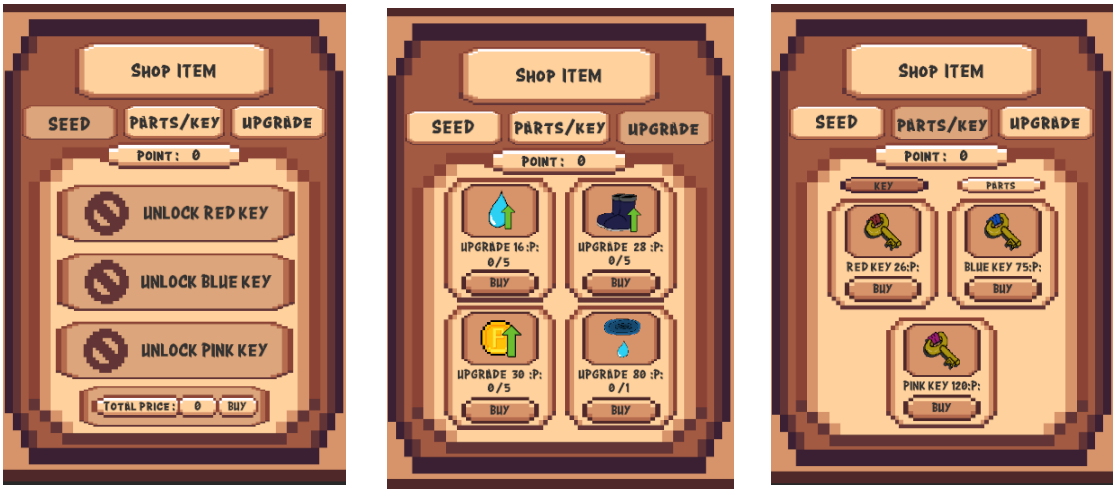
1.การใช้งาน หรือควบคุมระบบต่างๆของเกม PLANT & FLEE

1. การควบคุมตัวละคร

ภาพที่ ค-1 การควบคุมตัวละคร

กด	WASD	เพื่อควบคุมตัวละคร
กด	F	ที่แปลงผักเพื่อปลูกผัก
กด	SPACEBAR	เพื่อรดน้ำผัก
กด	F	ที่ผักที่โตแล้วเพื่อเก็บ
กด	SPACEBAR	ที่ปั้มน้ำเพื่อเติมน้ำให้กับผู้เล่น
กด	F	ที่หัวกวางเพื่อเปิดหน้าแลก item
กด	TAP	เพื่อซื้อค่าน้ำต้องแลก item ใน shop ก่อน
กด	i	แก้ตัวละครบัคเมื่อปลูกผักแล้วเดินไม่ได้
กด	O	เพื่อเพิ่ม point+1000 ให้ผู้เล่นใช้เพื่อทดสอบเกม

1.2 เมื่อเข้าหน้าต่าง item shop จะมี 3 หมวดหมู่ให้เลือก



ภาพที่ ค-2 หน้าต่าง item shop

หน้า SEED เอาไว้ให้ผู้เล่นแลกเมล็ดผักต่างๆ

หน้า PARTS/KEY เอาไว้ให้ผู้เล่นแลกกุญแจเอาไว้ปลดแปลงผักหรือชิ้นส่วนรถยนต์

หน้า UPGRADE เอาไว้อัพความสามารถของผู้เล่น

1.3 หน้า upgrade ความสามารถต่างๆ

ภาพที่ ค-3 หน้าต่าง upgrade ความสามารถ item ต่างๆ

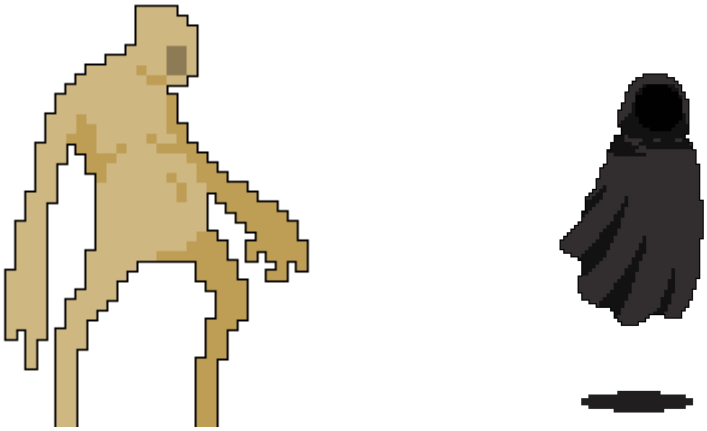
Itemที่ 1 อัพค่าสูงสุดของน้ำผู้เล่น

Itemที่ 2 อัพค่าความเร็วของน้ำผู้เล่น

Itemที่ 3 อัพค่า point ของผักเมื่อเก็บ

Itemที่ 4 อัพเพื่อให้ผู้เล่นกด Tap สามารถซื้อน้ำได้โดยไม่ต้องใช้ปั้ม

1.4 การเอาตัวรอดเมื่อเจอมอนสเตอร์ของเกม PLANT & FLEE



ภาพที่ ค-4 มอนสเตอร์ของเกม PLANT & FLEE

มอนสเตอร์ตัวที่ 1 มอนสเตอร์จะเกิดเมื่อถึงเวลาตอนกลางคืนมอนสเตอร์จะเดินตามผู้เล่นตลอดทั้งคืนผู้เล่นจะต้องเดินหลบเลี่ยงไปเรื่อยๆจนถึงเช้ามอนสเตอร์จะหายไปเอง

มอนสเตอร์ตัวที่ 2 มอนสเตอร์จะเกิดเมื่อถึงเวลาตอนกลางคืนเมื่อมอนสเตอร์เกิดจะมีเสียงเตือนผู้เล่นตอนมีเสียงเตือนผู้เล่น ผู้เล่นจะต้องเข้าไปหลบในบ้านจะต้องมีเสียงมอนสเตอร์ชนประตูผู้เล่นถึงจะออกมาได้

1.5 การจบของเกม PLANT & FLEE

ภาพที่ ค-5 รถในเกม PLANT & FLEE

ผู้เล่นจะต้องซื้อชิ้นส่วน Parts ใน shop ให้ครบทั้ง 4 ชิ้น เมื่อผู้เล่นซื้อชิ้นส่วนครบแล้วให้ผู้เล่นวิ่งมาที่รถเพื่อจบเกม

2.คุณสมบัติของ เกม PLANT & FLEE

2.1 เพื่อฝึกการจัดการทรัพยากร

2.2 เพื่อฝึกการวางแผนจัดการต่างๆ

2.3 เพื่อฝึกไหวพริบและอุปสรรคที่ต่างกัน

3.การติดตั้ง เกม PLANT & FLEE

3.1 เข้าไฟล์ exe

3.2 เลือกตำแหน่งที่ต้องการติดตั้ง

3.3 กดติดตั้ง

3.4 เข้าไฟล์ที่ติดตั้งเลือกไฟล์ชื่อ PLANT & FLEE เพื่อเล่น

ภาคผนวก ง.

โค้ดที่ใช้ใน เกม PLANT & FLEE

โค้ดที่ใช้ในเกม PLANT & FLEE

การเคลื่อนที่ตัวละคร

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class player : MonoBehaviour
{
    Rigidbody2D rd;
    public float speed = 0.5f;
    Vector2 movement;
    Animator animator;
    int R = 1;
    int L = 0;
    private float time = 0;
    private bool timeb = false;
    private int water = 0;
    private AudioSource AS;
    public AudioClip coinget;
    public int point = 0;
    public int watermax = 0;
    public int shop = 0;
    public int imgstop = 0;
    public int seedcarrot = 0;
    public int seedpumpkin = 0;
    public int seedpotato = 0;
    private GameManager gameManager;
    public float speedplayer = 0f;

    public AudioClip moveSound;
    public float moveSoundVolume = 0.5f;
    private bool isMoving = false;
    public int walk =0;
    public int dropwater = 0;
    public int dropwatermax = 0;
    public int stoptab = 0;
    public int animewater = 0;
    public GameObject uidead;
    private void Awake()
    {
        rd = GetComponent<Rigidbody2D>();
        animator = GetComponent<Animator>();
        AS = GetComponent<AudioSource>();
    }
    private void Start()
    {
        gameManager = GameManager.instance;
    }
    void Update()
    {
        animewater = GameManager.instance.animatorwa;
        walk = GameManager.instance.walksound;
        float moveX = Input.GetAxisRaw("Horizontal");
        float moveY = Input.GetAxisRaw("Vertical");
        movement = new Vector2(Input.GetAxisRaw("Horizontal"),
Input.GetAxisRaw("Vertical"));
```

```
if (moveX > 0)
{
    transform.eulerAngles = new Vector3(0, 0, 0);
    animator.SetBool("Run", true);
    R = 1; L = 0;
    walk = 1;
    gameManager.Updatewalksound(walk);

}
else if (moveX < 0)
{
    transform.eulerAngles = new Vector3(0, 180, 0);
    animator.SetBool("Run", true);
    R = 0; L = 1;
    walk = 1;
    gameManager.Updatewalksound(walk);

}
else
{
    timeb = false;
    time = 0;
    animator.SetBool("waterp", false);
    animator.SetBool("Run", false);
    walk = 0;
    gameManager.Updatewalksound(walk);
}
if (R == 1 && L == 0)
{
    if (moveY > 0)
    {
        transform.eulerAngles = new Vector3(0, 0, 0);
        animator.SetBool("Run", true);
        walk = 1;
        gameManager.Updatewalksound(walk);

    }
    if (moveY < 0)
    {
        transform.eulerAngles = new Vector3(0, 0, 0);
        animator.SetBool("Run", true);
        walk = 1;
        gameManager.Updatewalksound(walk);

    }
}
else if (R == 0 && L == 1)
{

    if (moveY > 0)
    {
        transform.eulerAngles = new Vector3(0, 180, 0);
        animator.SetBool("Run", true);
        walk = 1;
        gameManager.Updatewalksound(walk);

    }
    if (moveY < 0)
    {
```

```
            transform.eulerAngles = new Vector3(0, 180, 0);
            animator.SetBool("Run", true);
            walk = 1;
            gameManager.Updatewalksound(walk);

        }
    }
    water = GameManager.instance.animatorwa;
    speed = GameManager.instance.speed;
    speedplayer = GameManager.instance.speedplayer;
    if(animewater == 1)
    {
        animator.SetBool("Run", false);
        walk = 0;
        gameManager.Updatewalksound(walk);
        speed = 0;
    }
    else
    {
        speed = speed + speedplayer;
    }

    // Debug.Log(time);
    shop = GameManager.instance.shopstop;
    imgstop = GameManager.instance.imgstop;
    if(shop == 1)
    {
        animator.SetBool("Run", false);
        speed = 0;
        walk = 0;
        gameManager.Updatewalksound(walk);
    }
    if(imgstop == 1)
    {
        animator.SetBool("Run", false);
        speed = 0;
        walk = 0;
        gameManager.Updatewalksound(walk);
    }
    rd.velocity = movement * speed;
    point = GameManager.instance.point;
    seedcarrot = GameManager.instance.seedcarrot;
    seedpumpkin = GameManager.instance.seedpumpkin;
    seedpotato = GameManager.instance.seedpotato;
    dropwater = GameManager.instance.waterValue;
    stoptab = GameManager.instance.stoptab;
    dropwatermax = GameManager.instance.pumpmax;
    float horizontalMovement = Input.GetAxis("Horizontal");
    float verticalMovement = Input.GetAxis("Vertical");
    isMoving = (horizontalMovement != 0f || verticalMovement != 0f);
    if(Input.GetKeyDown(KeyCode.I))
    {
        animewater = 0;
        gameManager.Updateanimatorwa(animewater);
    }
    if(Input.GetKeyDown(KeyCode.O))
    {
        point = point + 1000;
        gameManager.Updatepoint(point);
```

```
        }
    }
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "coin")
        {
            AS.PlayOneShot(coinget);
            Destroy(collision.gameObject);
            point = point + 1;
            gameManager.Updatepoint(point);
        }
        if (collision.gameObject.tag == "pointcarrot")
        {
            AS.PlayOneShot(coinget);
            Destroy(collision.gameObject);
            seedcarrot = seedcarrot + 1;
            gameManager.Updateseedcarrot(seedcarrot);
        }
        if (collision.gameObject.tag == "pointpumpkin")
        {
            AS.PlayOneShot(coinget);
            Destroy(collision.gameObject);
            seedpumpkin = seedpumpkin + 1;
            gameManager.Updateseedpumpkin(seedpumpkin);
        }
        if (collision.gameObject.tag == "pointpotato")
        {
            AS.PlayOneShot(coinget);
            Destroy(collision.gameObject);
            seedpotato = seedpotato + 1;
            gameManager.Updateseedpotato(seedpotato);
        }
        if (collision.gameObject.tag == "dropwater")
        {
            AS.PlayOneShot(coinget);
            Destroy(collision.gameObject);
            dropwater = dropwater + 5;
            gameManager.UpdateWater(dropwater);
            if(dropwater > dropwatermax)
            {
                dropwater = dropwatermax;
                gameManager.UpdateWater(dropwater);
            }
        }
        if (collision.gameObject.tag == "monster")
        {
            uidead.SetActive(true);
        }
    }
    void OnCollisionEnter2D(Collision2D collision)
    {

    }
}
```

ปลูกผักมะเขือเทศ

```
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
//using UnityEditorInternal;
using UnityEngine;

public class plant0 : MonoBehaviour
{
    public GameObject growthStages1;
    public GameObject growthStages2;
    public GameObject growthStages3;
    public GameObject growthStages4;
    public GameObject growthStages5;
    public GameObject floor;
    private bool isPlanting = false;
    private float time = 0f;
    public int water = 0;
    public int point = 0;
    public int seed = 0;
    private bool iswater = false;
    private bool stop = false;
    private bool waterstop = false;
    private bool grow = false;
    private float timegrow = 0;
    private int roundedTime = 0;
    private GameObject currentPlant;
    private GameManager gameManager;
    public AudioClip plantsound;
    public AudioClip watersound;
    public AudioClip growsound;
    public AudioClip plantget;
    private AudioSource AS;
    private Animator animator;
    private float timewater;
    private int animatorwater = 0;
    private bool stoptime = false;
    private bool stopsac = false;
    public int pointm = 0;
    public int rdm = 0;
    public int coin = 0;
    public GameObject coinPrefab;
    private GameObject go;
    void Start()
    {
        gameManager = GameManager.instance;
        AS = GetComponent<AudioSource>();
        animator = GetComponent<Animator>();
    }

    // Update is called once per frame
    void Update()
    {
        water = GameManager.instance.waterValue;
        point = GameManager.instance.point;
        seed = GameManager.instance.Seedtomato;
        animatorwater = GameManager.instance.animatorwa;
        pointm = GameManager.instance.pointmax;
```

```
        coin = GameManager.instance.cointomato;
        if (Input.GetKeyDown(KeyCode.F) && isPlanting == false && seed >
0)
        {
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("planttomato0"))
                {
                    stop = false;
                    isPlanting = true;
                    Vector3 spawnPosition = collider.transform.position;
                    currentPlant = Instantiate(growthStages1,
spawnPosition, Quaternion.identity);
                    AS.PlayOneShot(plantsound);
                    seed--;
                    gameManager.UpdateSeedtomato(seed);
                }
            }
        }
        if (isPlanting == true && water >= 0)
        {
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("planttomato0"))
                {
                    if (water == 0)
                    {
                        waterstop = true;
                    }
                    else if (water > 0)
                    {
                        waterstop = false;
                    }
if (Input.GetKeyDown(KeyCode.Space) && iswater == false && waterstop ==
false && stop == false)
                    {
                        water--;
                        iswater = true;
                        gameManager.UpdateWater(water);
                        AS.PlayOneShot(watersound);
                        grow = false;
                        timegrow = 0;
                    }
if (Input.GetKeyDown(KeyCode.Space) || stoptime == true){
                        stoptime = true;
if (iswater == true && stoptime == true && stopsac == false)
                        {
                            timewater += Time.deltaTime;
                            animator.SetBool("waterp", true);
                            animatorwater = 1;
                            gameManager.Updateanimatorwa(animatorwater);
                            if (timewater >= 1.1)
                            {
                                animator.SetBool("waterp", false);
                                animatorwater = 0;
```

```
gameManager.Updateanimatorwa(animatorwater);
                            }
                        }
                    }
                    if(stoptime == true)
                    {
                        timewater += Time.deltaTime;
                    }
                    if (timewater >= 1.3)
                    {
                        stoptime = false;
                        stopsac = true;
                        timewater = 0;
                    }
                }
            }
            if (iswater == true && stop == false)
            {
                time += Time.deltaTime;
                roundedTime = Mathf.RoundToInt(time);
               // timewater += Time.deltaTime;
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, new Color(181f / 255f, 138f /
255f, 138f / 255f));
            }
            if (roundedTime == 11)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages2,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 12;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 22)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages3,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 23;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 33)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
```

```
                currentPlant = Instantiate(growthStages4,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 34;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 44)
            {
                stop = true;
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages5,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                timewater = 0;
                stopsac = false;
                foreach (Collider2D collider in colliders)
                {
                    if (collider.CompareTag("planttomato0"))
                    {
                        if (Input.GetKeyDown(KeyCode.F))
                        {
                            Destroy(currentPlant);
                            rdm = Random.Range(1, 3);
                            seed = seed + rdm;
                            gameManager.UpdateSeedtomato(seed);
                            isPlanting = false;
                            time = 0;
                            roundedTime = 0;
                            pointm = pointm + coin;
                            gameManager.Updatepointmax(pointm);
                            AS.PlayOneShot(plantget);
                            for (int i = 0; i < coin; i++)
                            {
Vector2 randomOffset = new Vector2(Random.Range(-4f, 4f), Random.Range(-
4f, 4f));
Vector2 coinPosition = (Vector2)transform.position + randomOffset;
go = Instantiate(coinPrefab, coinPosition, Quaternion.identity);
                            }
                        }
                    }
                }
            }
            //Debug.Log(timewater);
            if (grow == true)
            {
                timegrow++;
                if (timegrow == 10)
                {
                    AS.PlayOneShot(growsound);
                }
            }
        }
    }
    void SetColor(Renderer renderer, Color newColor)
    {
```

```
        renderer.material.color = newColor;
    }
}
```

ปลูกผักแครอท

```
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
//using UnityEditorInternal;
using UnityEngine;

public class plantcarrot0: MonoBehaviour
{
    public GameObject growthStages1;
    public GameObject growthStages2;
    public GameObject growthStages3;
    public GameObject growthStages4;
    public GameObject growthStages5;
    public GameObject floor;
    private bool isPlanting = false;
    private float time = 0f;
    public int water = 0;
    public int point = 0;
    public int seed = 0;
    private bool iswater = false;
    private bool stop = false;
    private bool waterstop = false;
    private bool grow = false;
    private float timegrow = 0;
    private int roundedTime = 0;
    private GameObject currentPlant;
    private GameManager gameManager;
    public AudioClip plantsound;
    public AudioClip watersound;
    public AudioClip growsound;
    public AudioClip plantget;
    private AudioSource AS;
    private Animator animator;
    private float timewater;
    private int animatorwater = 0;
    private bool stoptime = false;
    private bool stopsac = false;
    public int pointm = 0;
    public int rdm = 0;
    public int coin = 0;
    public GameObject coinPrefab;
    private GameObject go;
    void Start()
    {
        gameManager = GameManager.instance;
        AS = GetComponent<AudioSource>();
        animator = GetComponent<Animator>();
    }

    // Update is called once per frame
    void Update()
    {
```

```
        water = GameManager.instance.waterValue;
        point = GameManager.instance.point;
        seed = GameManager.instance.seedcarrot;
        animatorwater = GameManager.instance.animatorwa;
        pointm = GameManager.instance.pointmax;
        coin = GameManager.instance.coincarrot;
        if (Input.GetKeyDown(KeyCode.F) && isPlanting == false && seed >
0)
        {
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("plantcarrot0"))
                {
                    stop = false;
                    isPlanting = true;
                    Vector3 spawnPosition = collider.transform.position;
                    currentPlant = Instantiate(growthStages1,
spawnPosition, Quaternion.identity);
                    AS.PlayOneShot(plantsound);
                    seed--;
                    gameManager.Updateseedcarrot(seed);
                }
            }
        }
        if (isPlanting == true && water >= 0)
        {
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("plantcarrot0"))
                {
                    if (water == 0)
                    {
                        waterstop = true;
                    }
                    else if (water > 0)
                    {
                        waterstop = false;
                    }
if (Input.GetKeyDown(KeyCode.Space) && iswater == false && waterstop ==
false && stop == false)
                    {
                        water--;
                        iswater = true;
                        gameManager.UpdateWater(water);
                        AS.PlayOneShot(watersound);
                        grow = false;
                        timegrow = 0;
                    }
if (Input.GetKeyDown(KeyCode.Space) || stoptime == true)
                    {
                        stoptime = true;
if (iswater == true && stoptime == true && stopsac == false)
                        {
                            timewater += Time.deltaTime;
                            animator.SetBool("waterp", true);
```

```
                            animatorwater = 1;
                            gameManager.Updateanimatorwa(animatorwater);
                            if (timewater >= 1.1)
                            {
                                animator.SetBool("waterp", false);
                                animatorwater = 0;

gameManager.Updateanimatorwa(animatorwater);
                            }
                        }
                    }
                    if (stoptime == true)
                    {
                        timewater += Time.deltaTime;
                    }
                    if (timewater >= 1.3)
                    {
                        stoptime = false;
                        stopsac = true;
                        timewater = 0;
                    }
                }
            }
            if (iswater == true && stop == false)
            {
                time += Time.deltaTime;
                roundedTime = Mathf.RoundToInt(time);
                // timewater += Time.deltaTime;
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, new Color(181f / 255f, 138f /
255f, 138f / 255f));
            }
            if (roundedTime == 11)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages2,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 12;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 22)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages3,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 23;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 33)
```

```
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages4,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 34;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 44)
            {
                stop = true;
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages5,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                timewater = 0;
                stopsac = false;
                foreach (Collider2D collider in colliders)
                {
                    if (collider.CompareTag("plantcarrot0"))
                    {
                        if (Input.GetKeyDown(KeyCode.F))
                        {
                            Destroy(currentPlant);
                            isPlanting = false;
                            time = 0;
                            roundedTime = 0;
                            pointm = pointm + coin;
                            gameManager.Updatepointmax(pointm);
                            AS.PlayOneShot(plantget);
                            for (int i = 0; i < coin; i++)
                            {
Vector2 randomOffset = new Vector2(Random.Range(-4f, 4f), Random.Range(-
4f, 4f));
Vector2 coinPosition = (Vector2)transform.position + randomOffset;
go = Instantiate(coinPrefab, coinPosition, Quaternion.identity);
                            }
                        }
                    }
                }
            }
            //Debug.Log(timewater);
            if (grow == true)
            {
                timegrow++;
                if (timegrow == 10)
                {
                    AS.PlayOneShot(growsound);
                }
            }
        }
    }
```

```
    void SetColor(Renderer renderer, Color newColor)
    {
        renderer.material.color = newColor;
    }
}
```

ปลูกผักฟักทอง

```
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
//using UnityEditorInternal;
using UnityEngine;

public class plantkin0 : MonoBehaviour
{
    public GameObject growthStages1;
    public GameObject growthStages2;
    public GameObject growthStages3;
    public GameObject growthStages4;
    public GameObject growthStages5;
    public GameObject floor;
    private bool isPlanting = false;
    private float time = 0f;
    public int water = 0;
    public int point = 0;
    public int seed = 0;
    private bool iswater = false;
    private bool stop = false;
    private bool waterstop = false;
    private bool grow = false;
    private float timegrow = 0;
    private int roundedTime = 0;
    private GameObject currentPlant;
    private GameManager gameManager;
    public AudioClip plantsound;
    public AudioClip watersound;
    public AudioClip growsound;
    public AudioClip plantget;
    private AudioSource AS;
    private Animator animator;
    private float timewater;
    private int animatorwater = 0;
    private bool stoptime = false;
    private bool stopsac = false;
    public int pointm = 0;
    public int rdm = 0;
    public int coin = 0;
    public GameObject coinPrefab;
    private GameObject go;
    void Start()
    {
        gameManager = GameManager.instance;
        AS = GetComponent<AudioSource>();
        animator = GetComponent<Animator>();
    }

    // Update is called once per frame
```

```
    void Update()
    {
        water = GameManager.instance.waterValue;
        point = GameManager.instance.point;
        seed = GameManager.instance.seedpumpkin;
        animatorwater = GameManager.instance.animatorwa;
        pointm = GameManager.instance.pointmax;
        coin = GameManager.instance.coinpumpkin;
        if (Input.GetKeyDown(KeyCode.F) && isPlanting == false && seed >
0)
        {
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("plantpumpkin0"))
                {
                    stop = false;
                    isPlanting = true;
                    Vector3 spawnPosition = collider.transform.position;
                    currentPlant = Instantiate(growthStages1,
spawnPosition, Quaternion.identity);
                    AS.PlayOneShot(plantsound);
                    seed--;
                    gameManager.Updateseedpumpkin(seed);
                }
            }
        }
        if (isPlanting == true && water >= 0)
        {
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("plantpumpkin0"))
                {
                    if (water == 0)
                    {
                        waterstop = true;
                    }
                    else if (water > 0)
                    {
                        waterstop = false;
                    }
if (Input.GetKeyDown(KeyCode.Space) && iswater == false && waterstop ==
false && stop == false)
                    {
                        water--;
                        iswater = true;
                        gameManager.UpdateWater(water);
                        AS.PlayOneShot(watersound);
                        grow = false;
                        timegrow = 0;
                    }
if (Input.GetKeyDown(KeyCode.Space) || stoptime == true)
                    {
                        stoptime = true;
if (iswater == true && stoptime == true && stopsac == false)
                        {
```

```
                            timewater += Time.deltaTime;
                            animator.SetBool("waterp", true);
                            animatorwater = 1;
                            gameManager.Updateanimatorwa(animatorwater);
                            if (timewater >= 1.1)
                            {
                                animator.SetBool("waterp", false);
                                animatorwater = 0;

gameManager.Updateanimatorwa(animatorwater);
                            }
                        }
                    }
                    if (stoptime == true)
                    {
                        timewater += Time.deltaTime;
                    }
                    if (timewater >= 1.3)
                    {
                        stoptime = false;
                        stopsac = true;
                        timewater = 0;
                    }
                }
            }
            if (iswater == true && stop == false)
            {
                time += Time.deltaTime;
                roundedTime = Mathf.RoundToInt(time);
                // timewater += Time.deltaTime;
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, new Color(181f / 255f, 138f /
255f, 138f / 255f));
            }
            if (roundedTime == 11)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages2,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 12;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 22)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages3,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 23;
                timewater = 0;
                stopsac = false;
```

```
            }
            if (roundedTime == 33)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages4,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 34;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 44)
            {
                stop = true;
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages5,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                timewater = 0;
                stopsac = false;
                foreach (Collider2D collider in colliders)
                {
                    if (collider.CompareTag("plantpumpkin0"))
                    {
                        if (Input.GetKeyDown(KeyCode.F))
                        {
                            Destroy(currentPlant);
                            isPlanting = false;
                            time = 0;
                            roundedTime = 0;
                            pointm = pointm + coin;
                            gameManager.Updatepointmax(pointm);
                            AS.PlayOneShot(plantget);
                            for (int i = 0; i < coin; i++)
                            {
Vector2 randomOffset = new Vector2(Random.Range(-4f, 4f), Random.Range(-
4f, 4f));
Vector2 coinPosition = (Vector2)transform.position + randomOffset;
go = Instantiate(coinPrefab, coinPosition, Quaternion.identity);
                            }
                        }
                    }
                }
            }
            //Debug.Log(timewater);
            if (grow == true)
            {
                timegrow++;
                if (timegrow == 10)
                {
                    AS.PlayOneShot(growsound);
                }
            }
```

```
        }
    }
    void SetColor(Renderer renderer, Color newColor)
    {
        renderer.material.color = newColor;
    }
}
```

ปลูกผักมันฝรั่ง

```
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
//using UnityEditorInternal;
using UnityEngine;

public class plantpotato0 : MonoBehaviour
{
    public GameObject growthStages1;
    public GameObject growthStages2;
    public GameObject growthStages3;
    public GameObject growthStages4;
    public GameObject growthStages5;
    public GameObject floor;
    private bool isPlanting = false;
    private float time = 0f;
    public int water = 0;
    public int point = 0;
    public int seed = 0;
    private bool iswater = false;
    private bool stop = false;
    private bool waterstop = false;
    private bool grow = false;
    private float timegrow = 0;
    private int roundedTime = 0;
    private GameObject currentPlant;
    private GameManager gameManager;
    public AudioClip plantsound;
    public AudioClip watersound;
    public AudioClip growsound;
    public AudioClip plantget;
    private AudioSource AS;
    private Animator animator;
    private float timewater;
    private int animatorwater = 0;
    private bool stoptime = false;
    private bool stopsac = false;
    public int pointm = 0;
    public int rdm = 0;
    public int coin = 0;
    public GameObject coinPrefab;
    private GameObject go;
    void Start()
    {
        gameManager = GameManager.instance;
        AS = GetComponent<AudioSource>();
        animator = GetComponent<Animator>();
    }
```

```
    // Update is called once per frame
    void Update()
    {
        water = GameManager.instance.waterValue;
        point = GameManager.instance.point;
        seed = GameManager.instance.seedpotato;
        animatorwater = GameManager.instance.animatorwa;
        pointm = GameManager.instance.pointmax;
        coin = GameManager.instance.coinpotato;
        if (Input.GetKeyDown(KeyCode.F) && isPlanting == false && seed >
0)
        {
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("plantPotato0"))
                {
                    stop = false;
                    isPlanting = true;
                    Vector3 spawnPosition = collider.transform.position;
                    currentPlant = Instantiate(growthStages1,
spawnPosition, Quaternion.identity);
                    AS.PlayOneShot(plantsound);
                    seed--;
                    gameManager.Updateseedpotato(seed);
                }
            }
        }
        if (isPlanting == true && water >= 0)
        {
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("plantPotato0"))
                {
                    if (water == 0)
                    {
                        waterstop = true;
                    }
                    else if (water > 0)
                    {
                        waterstop = false;
                    }
if (Input.GetKeyDown(KeyCode.Space) && iswater == false && waterstop ==
false && stop == false)
                    {
                        water--;
                        iswater = true;
                        gameManager.UpdateWater(water);
                        AS.PlayOneShot(watersound);
                        grow = false;
                        timegrow = 0;
                    }
if (Input.GetKeyDown(KeyCode.Space) || stoptime == true)
                    {
                        stoptime = true;
```

```
if (iswater == true && stoptime == true && stopsac == false)
                        {
                            timewater += Time.deltaTime;
                            animator.SetBool("waterp", true);
                            animatorwater = 1;
                            gameManager.Updateanimatorwa(animatorwater);
                            if (timewater >= 1.1)
                            {
                                animator.SetBool("waterp", false);
                                animatorwater = 0;

gameManager.Updateanimatorwa(animatorwater);
                            }
                        }
                    }
                    if (stoptime == true)
                    {
                        timewater += Time.deltaTime;
                    }
                    if (timewater >= 1.3)
                    {
                        stoptime = false;
                        stopsac = true;
                        timewater = 0;
                    }
                }
            }
            if (iswater == true && stop == false)
            {
                time += Time.deltaTime;
                roundedTime = Mathf.RoundToInt(time);
                // timewater += Time.deltaTime;
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, new Color(181f / 255f, 138f /
255f, 138f / 255f));
            }
            if (roundedTime == 11)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages2,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 12;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 22)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages3,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 23;
```

```
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 33)
            {
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages4,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                time = 34;
                timewater = 0;
                stopsac = false;
            }
            if (roundedTime == 44)
            {
                stop = true;
                Renderer floorRenderer = floor.GetComponent<Renderer>();
                SetColor(floorRenderer, Color.white);
                iswater = false;
                Destroy(currentPlant);
                currentPlant = Instantiate(growthStages5,
currentPlant.transform.position, Quaternion.identity);
                grow = true;
                timewater = 0;
                stopsac = false;
                foreach (Collider2D collider in colliders)
                {
                    if (collider.CompareTag("plantPotato0"))
                    {
                        if (Input.GetKeyDown(KeyCode.F))
                        {
                            Destroy(currentPlant);
                            isPlanting = false;
                            time = 0;
                            roundedTime = 0;
                            pointm = pointm + coin;
                            gameManager.Updatepointmax(pointm);
                            AS.PlayOneShot(plantget);
                            for (int i = 0; i < coin; i++)
                            {
Vector2 randomOffset = new Vector2(Random.Range(-4f, 4f), Random.Range(-
4f, 4f));
Vector2 coinPosition = (Vector2)transform.position + randomOffset;
go = Instantiate(coinPrefab, coinPosition, Quaternion.identity);
                            }
                        }
                    }
                }
            }
            //Debug.Log(timewater);
            if (grow == true)
            {
                timegrow++;
                if (timegrow == 10)
                {
                    AS.PlayOneShot(growsound);
```

```
                }
            }
        }
    }
    void SetColor(Renderer renderer, Color newColor)
    {
        renderer.material.color = newColor;
    }
}
```

ปั๊มน้ำ

```
using System.Collections;
using System.Collections.Generic;
using System.Drawing;
using System.Reflection;
using UnityEngine;

public class punp : MonoBehaviour
{
    // Start is called before the first frame update
    private GameManager gameManager;
    public int water = 15;
    public float time = 5;
    private bool timetrue = false;
    public int pump = 0;
    private AudioSource AS;
    public AudioClip soundpump;
    private bool sp = false;
    public int max = 0;
    void Start()
    {
        gameManager = GameManager.instance;
        AS = GetComponent<AudioSource>();
        //AS.clip = soundpump;
    }
    // Update is called once per frame
    void Update()
    {
       water = GameManager.instance.waterValue;
        pump = GameManager.instance.pumpwater;
        max = GameManager.instance.pumpmax;
            Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
            foreach (Collider2D collider in colliders)
            {
                if (collider.CompareTag("watermm"))
                {
                    if (Input.GetKey(KeyCode.Space) && timetrue == false)
                    {
                        if (water < max)
                        {
                            timetrue = true;
                            time = GameManager.instance.waterValue;
                            sp = true;
                        }
                    }
                    if (timetrue == true)
```

```
                    {
                        time += Time.deltaTime * 2;
                        gameManager.UpdateWater((int)time);
                        pump = 1;
                        gameManager.Updatepumpwater(pump);

                        if (time >= max)
                        {
                            timetrue = false;
                            pump = 0;
                            gameManager.Updatepumpwater(pump);
                            AS.Stop();
                        }
                    }
                }
            }
            if(sp == true)
        {
            AS.PlayOneShot(soundpump);
            sp = false;
        }
        //Debug.Log(time);
    }
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "box")
        {
            timetrue = false;
            pump = 0;
            gameManager.Updatepumpwater(pump);
            AS.Stop();
        }
    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class pumpani : MonoBehaviour
{
    // Start is called before the first frame update
    Animator animator;
    private bool time = false;
    public int pump = 0;
    void Start()
    {
        animator = GetComponent<Animator>();
    }

    // Update is called once per frame
    void Update()
    {
        pump = GameManager.instance.pumpwater;
        if(pump == 1)
        {
            time = true;
        }
```

```
        if(pump == 0)
        {
            time = false;
        }
        if(time == true)
        {
            animator.SetBool("waterpump", true);
        }
        if (time == false)
        {
            animator.SetBool("waterpump", false);
        }
    }
}
```

หน้าแลก item

```
using UnityEngine;

public class ShopEntrance : MonoBehaviour
{
    public GameObject shopPanel;
    public int numF = 0;
    private GameManager gameManager;
    public int stopf = 0;
    public GameObject uiplayer;
    private AudioSource AS;
    public AudioClip soundui;
    private void Start()
    {
        gameManager = GameManager.instance;
        AS = GetComponent<AudioSource>();
    }
    private void Update()
    {
        numF = GameManager.instance.shopstop;
        stopf = GameManager.instance.starttrade;
        Collider2D[] colliders =
Physics2D.OverlapCircleAll(transform.position, 1f);
        foreach (Collider2D collider in colliders)
        {
            if (collider.CompareTag("shop"))
            {
                if (Input.GetKeyDown(KeyCode.F) && stopf == 0)
                {
                    numF++;
                    AS.PlayOneShot(soundui);
                }
                if(numF == 0)
                {
                    uiplayer.SetActive(true);
                }
                if(numF == 1)
                {
                    shopPanel.SetActive(true);
                    gameManager.Updateshopstop(numF);
                    uiplayer.SetActive(false);
                }
```

```
                if (numF == 2)
                {
                    shopPanel.SetActive(false);
                    numF = 0;
                    gameManager.Updateshopstop(numF);
                }
            }
        }
    }

    private void OpenShop()
    {

           // shopPanel.SetActive(true);
    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class seedsound : MonoBehaviour
{
    public GameObject button;
    private AudioSource AS;
    public AudioClip click;
    public bool soundstop = false;
    public int time = 0;
    public int numF = 0;
    public bool soundstat = false;
    public int timeF = 0;
    void Start()
    {
        AS = GetComponent<AudioSource>();
    }

    // Update is called once per frame
    void Update()
    {
        numF = GameManager.instance.shopstop;
        if(numF == 1 && soundstat == false)
        {
            timeF++;
            AS.enabled = false;
        }
        if (timeF == 10)
        {
            soundstat = true;
            timeF = 0;
            AS.enabled = true;
        }
        if(!button.activeSelf && soundstop == false)
        {
            time++;
        }
        if(button.activeSelf) {
            soundstop = false;
```

```
        }
        if(time == 5) {
            soundstop = true;
            time = 0;
            AS.PlayOneShot(click);
        }
    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class upsound : MonoBehaviour
{
    public GameObject button;
    private AudioSource AS;
    public AudioClip click;
    public bool soundstop = false;
    public int time = 0;
    public int numF = 0;
    public bool soundstat = false;
    public int timeF = 0;
    void Start()
    {
        AS = GetComponent<AudioSource>();
    }

    // Update is called once per frame
    void Update()
    {
        numF = GameManager.instance.shopstop;
        if (numF == 1 && soundstat == false)
        {
            timeF++;
            AS.enabled = false;
        }
        if (timeF == 10)
        {
            soundstat = true;
            timeF = 0;
            AS.enabled = true;
        }
        if (!button.activeSelf && soundstop == false)
        {
            time++;
        }
        if (button.activeSelf)
        {
            soundstop = false;
        }
        if (time == 5)
        {
            soundstop = true;
            time = 0;
            AS.PlayOneShot(click);
        }
    }
```

```
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class keysound : MonoBehaviour
{
    public GameObject button;
    private AudioSource AS;
    public AudioClip click;
    public bool soundstop = false;
    public int time = 0;
    public int numF = 0;
    public bool soundstat = false;
    public int timeF = 0;
    void Start()
    {
        AS = GetComponent<AudioSource>();
    }

    // Update is called once per frame
    void Update()
    {
        numF = GameManager.instance.shopstop;
        if (numF == 1 && soundstat == false)
        {
            timeF++;
            AS.enabled = false;
        }
        if (timeF == 10)
        {
            soundstat = true;
            timeF = 0;
            AS.enabled = true;
        }
        if (!button.activeSelf && soundstop == false)
        {
            time++;
        }
        if (button.activeSelf)
        {
            soundstop = false;
        }
        if (time == 5)
        {
            soundstop = true;
            time = 0;
            AS.PlayOneShot(click);
        }
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class soundkeyparts : MonoBehaviour
{
    public GameObject buyredkey;
    public GameObject buybluekey;
    public GameObject buypinkkey;
    public GameObject item1;
    public GameObject item2;
    public GameObject item3;
    public GameObject item4;
    private AudioSource AS;
    public AudioClip click;
    public bool soundstop1 = false;
    public bool soundstop2 = false;
    public bool soundstop3 = false;
    public bool soundstop4 = false;
    public bool soundstop5 = false;
    public bool soundstop6 = false;
    public bool soundstop7 = false;
    public int time1 = 0;
    public int time2 = 0;
    public int time3 = 0;
    public int time4 = 0;
    public int time5 = 0;
    public int time6 = 0;
    public int time7 = 0;
    public int numF = 0;
    public bool soundstat = false;
    public int timeF = 0;
    void Start()
    {
        AS = GetComponent<AudioSource>();
    }

    // Update is called once per frame
    void Update()
    {
        if (!buyredkey.activeSelf && soundstop1 == false)
        {
            time1++;
        }
        if(time1 == 5)
        {
            time1 = 0;
            soundstop1 = true;
            AS.PlayOneShot(click);
        }
        if (!buybluekey.activeSelf && soundstop2 == false)
        {
            time2++;
        }
        if (time2 == 5)
        {
            time2 = 0;
```

```
            soundstop2 = true;
            AS.PlayOneShot(click);
        }
        if (!buypinkkey.activeSelf && soundstop3 == false)
        {
            time3++;
        }
        if (time3 == 5)
        {
            time3 = 0;
            soundstop3 = true;
            AS.PlayOneShot(click);
        }
        if (!item1.activeSelf && soundstop4 == false)
        {
            time4++;
        }
        if (time4 == 5)
        {
            time4 = 0;
            soundstop4 = true;
            AS.PlayOneShot(click);
        }
        if (!item2.activeSelf && soundstop5 == false)
        {
            time5++;
        }
        if (time5 == 5)
        {
            time5 = 0;
            soundstop5 = true;
            AS.PlayOneShot(click);
        }
        if (!item3.activeSelf && soundstop6 == false)
        {
            time6++;
        }
        if (time6 == 5)
        {
            time6 = 0;
            soundstop6 = true;
            AS.PlayOneShot(click);
        }
        if (!item4.activeSelf && soundstop7 == false)
        {
            time7++;
        }
        if (time7 == 5)
        {
            time7 = 0;
            soundstop7 = true;
            AS.PlayOneShot(click);
        }
    }
}


using System.Collections;
using System.Collections.Generic;
```

```
using UnityEngine.UI;
using UnityEngine;

public class carrotminus : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int numbercarrot = 0;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        gameManager = GameManager.instance;
    }

    private void OnButtonClick()
    {
        numbercarrot = GameManager.instance.carrotnum;
        numbercarrot--;
        gameManager.Updatecarrotnum(numbercarrot);
        if (numbercarrot < 0)
        {
            numbercarrot = 0;
            gameManager.Updatecarrotnum(numbercarrot);
        }
    }
     void Update()
    {

    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class Sseed : MonoBehaviour
{
    private Button myButton;
    private AudioSource AS;
    public AudioClip sound;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        AS = GetComponent<AudioSource>();
    }

    private void OnButtonClick()
    {
       AS.PlayOneShot(sound);
    }
}


using System.Collections;
using System.Collections.Generic;
```

```
using UnityEngine.UI;
using UnityEngine;

public class carrotplus : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int numbercarrot = 0;

    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        gameManager = GameManager.instance;
    }

    private void OnButtonClick()
    {
        numbercarrot = GameManager.instance.carrotnum;
        numbercarrot++;
        gameManager.Updatecarrotnum(numbercarrot);
    }
    private void Update()
    {

    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class Sseed : MonoBehaviour
{
    private Button myButton;
    private AudioSource AS;
    public AudioClip sound;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        AS = GetComponent<AudioSource>();
    }

    private void OnButtonClick()
    {
       AS.PlayOneShot(sound);
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;
```

```
public class pumpkinminus : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int numberpumpkin = 0;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        gameManager = GameManager.instance;
    }

    private void OnButtonClick()
    {
        numberpumpkin = GameManager.instance.pumpkinnum;
        numberpumpkin--;
        gameManager.Updatepumpkinnum(numberpumpkin);
        if (numberpumpkin < 0)
        {
            numberpumpkin = 0;
            gameManager.Updatepumpkinnum(numberpumpkin);
        }
    }
    void Update()
    {

    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class Sseed : MonoBehaviour
{
    private Button myButton;
    private AudioSource AS;
    public AudioClip sound;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        AS = GetComponent<AudioSource>();
    }

    private void OnButtonClick()
    {
       AS.PlayOneShot(sound);
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
```

```
public class potatominus : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int numberpotato = 0;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        gameManager = GameManager.instance;
    }

    private void OnButtonClick()
    {
        numberpotato = GameManager.instance.potatonum;
        numberpotato--;
        gameManager.Updatepotatonum(numberpotato);
        if (numberpotato < 0)
        {
            numberpotato = 0;
            gameManager.Updatepotatonum(numberpotato);
        }
    }
    void Update()
    {

    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class Sseed : MonoBehaviour
{
    private Button myButton;
    private AudioSource AS;
    public AudioClip sound;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        AS = GetComponent<AudioSource>();
    }

    private void OnButtonClick()
    {
       AS.PlayOneShot(sound);
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
```

```
public class potatoplus : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int numberpotato = 0;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        gameManager = GameManager.instance;
    }

    private void OnButtonClick()
    {
        numberpotato = GameManager.instance.potatonum;
        numberpotato++;
        gameManager.Updatepotatonum(numberpotato);
    }
    void Update()
    {

    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class Sseed : MonoBehaviour
{
    private Button myButton;
    private AudioSource AS;
    public AudioClip sound;
    private void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        AS = GetComponent<AudioSource>();
    }

    private void OnButtonClick()
    {
       AS.PlayOneShot(sound);
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;
public class buttonbuy : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int point = 0;
```

```
public int pricecarrot = 0;
public int pricepumpkin = 0;
public int pricepotato = 0;
public int numcarrot = 0;
public int numpumpkin = 0;
public int numpotato = 0;
private int sumcarrot = 0;
private int sumpumpkin = 0;
private int sumpotato = 0;
public int seedcarrot = 0;
public int seedpumpkin = 0;
public int seedpotato = 0;
public int totalprice = 0;
public int numF = 0;
public GameObject uishop;
public int start = 0;
public bool buy = false;
private AudioSource AS;
public AudioClip error;

void Start()
{
    gameManager = GameManager.instance;
    myButton = GetComponent<Button>();
    myButton.onClick.AddListener(OnButtonClick);
    AS= GetComponent<AudioSource>();
}

private void OnButtonClick()
{
   if (point >= totalprice && buy == true)
    {
        buy = false;
        point = point - totalprice;
        gameManager.Updatepoint(point);
        totalprice = 0;
        gameManager.Updatetotalprice(totalprice);
        uishop.SetActive(false);
        numF = 0;
        gameManager.Updateshopstop(numF);
        start = 1;
        gameManager.Updatestarttrade(start);
    }
    else
    {
        AS.PlayOneShot(error);
    }
}
void Update()
{
    point = GameManager.instance.point;
    numcarrot = GameManager.instance.carrotnum;
    numpumpkin = GameManager.instance.pumpkinnum;
    numpotato = GameManager.instance.potatonum;
    totalprice = GameManager.instance.totalprice;
    seedcarrot = GameManager.instance.seedcarrot;
    seedpumpkin = GameManager.instance.seedpumpkin;
    seedpotato = GameManager.instance.seedpotato;
    numF = GameManager.instance.shopstop;
```

```
        start = GameManager.instance.starttrade;
        sumcarrot = numcarrot * pricecarrot;
        sumpumpkin = numpumpkin * pricepumpkin;
        sumpotato = numpotato * pricepotato;
        totalprice = sumcarrot + sumpumpkin + sumpotato;
        gameManager.Updatetotalprice(totalprice);
        if (totalprice > 0)
        {
            buy = true;
        }
        else
        {
            buy = false;
        }
    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class redkey : MonoBehaviour
{
    private Button myButton;
    private AudioSource AS;
    public AudioClip buy;
    private GameManager gameManager;
    public int point = 0;
    public int price = 0;
    public int key = 0;
    public GameObject unlock;
    void Start()
    {

        myButton = GetComponent<Button>();
        AS = GetComponent<AudioSource>();
        gameManager = GameManager.instance;
        myButton.onClick.AddListener(OnButtonClick);
    }
    void Update()
    {
        point = GameManager.instance.point;
        key = GameManager.instance.redkey;

    }
    private void OnButtonClick()
    {
        if(point >= price)
        {
            point = point - price;
            key = 1;
            gameManager.Updatepoint(point);
            gameManager.Updateredkey(key);
            gameObject.SetActive(false);
            unlock.SetActive(false);
        }
```

```
    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class bluekey : MonoBehaviour
{
    private Button myButton;
    private AudioSource AS;
    public AudioClip buy;
    private GameManager gameManager;
    public int point = 0;
    public int price = 0;
    public int key = 0;
    public GameObject unlock;
    void Start()
    {
        myButton = GetComponent<Button>();
        AS = GetComponent<AudioSource>();
        gameManager = GameManager.instance;
        myButton.onClick.AddListener(OnButtonClick);
    }
    void Update()
    {
        point = GameManager.instance.point;
        key = GameManager.instance.bluekey;

    }
    private void OnButtonClick()
    {
        if (point >= price)
        {
            point = point - price;
            key = 1;
            gameManager.Updatepoint(point);
            gameManager.Updatebluekey(key);
            gameObject.SetActive(false);
            unlock.SetActive(false);
        }

    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using static Unity.Collections.AllocatorManager;

public class pinkkey : MonoBehaviour
{
    private Button myButton;
    private AudioSource AS;
    public AudioClip buy;
```

```
    private GameManager gameManager;
    public int point = 0;
    public int price = 0;
    public int key = 0;
    public GameObject unlock;
    void Start()
    {
        myButton = GetComponent<Button>();
        AS = GetComponent<AudioSource>();
        gameManager = GameManager.instance;
        myButton.onClick.AddListener(OnButtonClick);
    }
    void Update()
    {
        point = GameManager.instance.point;
        key = GameManager.instance.pinkkey;

    }
    private void OnButtonClick()
    {
        if (point >= price)
        {
            point = point - price;
            key = 1;
            gameManager.Updatepoint(point);
            gameManager.Updatepinkkey(key);
            gameObject.SetActive(false);
            unlock.SetActive(false);
        }

    }
}

using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;

public class battery : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int point = 0;
    public int price = 0;
    public int item = 0;
    void Start()
    {
        myButton = GetComponent<Button>();
        gameManager = GameManager.instance;
        myButton.onClick.AddListener(OnButtonClick);
    }
    void Update()
    {
        point = GameManager.instance.point;
        item = GameManager.instance.item1;
    }
    private void OnButtonClick()
    {
```

```
        if (point >= price)
        {
            point = point - price;
            item = 1;
            gameManager.Updatepoint(point);
            gameManager.Updateitem1(item);
            gameObject.SetActive(false);
        }

    }
}

using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;

public class wheel : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int point = 0;
    public int price = 0;
    public int item = 0;
    void Start()
    {
        myButton = GetComponent<Button>();
        gameManager = GameManager.instance;
        myButton.onClick.AddListener(OnButtonClick);
    }
    void Update()
    {
        point = GameManager.instance.point;
        item = GameManager.instance.item2;
    }
    private void OnButtonClick()
    {
        if (point >= price)
        {
            point = point - price;
            item = 1;
            gameManager.Updatepoint(point);
            gameManager.Updateitem2(item);
            gameObject.SetActive(false);
        }

    }
}

using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;

public class petrol : MonoBehaviour
{
    private Button myButton;
```

```
    private GameManager gameManager;
    public int point = 0;
    public int price = 0;
    public int item = 0;
    void Start()
    {
        myButton = GetComponent<Button>();
        gameManager = GameManager.instance;
        myButton.onClick.AddListener(OnButtonClick);
    }
    void Update()
    {
        point = GameManager.instance.point;
        item = GameManager.instance.item3;
    }
    private void OnButtonClick()
    {
        if (point >= price)
        {
            point = point - price;
            item = 1;
            gameManager.Updatepoint(point);
            gameManager.Updateitem3(item);
            gameObject.SetActive(false);
        }

    }
}

using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;

public class gear : MonoBehaviour
{
    private Button myButton;
    private GameManager gameManager;
    public int point = 0;
    public int price = 0;
    public int item = 0;
    void Start()
    {
        myButton = GetComponent<Button>();
        gameManager = GameManager.instance;
        myButton.onClick.AddListener(OnButtonClick);
    }
    void Update()
    {
        point = GameManager.instance.point;
        item = GameManager.instance.item4;
    }
    private void OnButtonClick()
    {
        if (point >= price)
        {
            point = point - price;
            item = 1;
```

```
            gameManager.Updatepoint(point);
            gameManager.Updateitem4(item);
            gameObject.SetActive(false);
        }

    }
}

//using System;
using System.Collections;
using System.Collections.Generic;
//using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;
public class bk : MonoBehaviour
{
    private Button myButton;
    public GameObject key;
    public GameObject paets;
    public GameObject buttinpart;
    void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
    }

    private void OnButtonClick()
    {
        key.SetActive(true);
        paets.SetActive(false);

        buttinpart.SetActive(true);
        gameObject.SetActive(false);
    }
    void Update()
    {

    }
}

//using System;
using System.Collections;
using System.Collections.Generic;
//using Unity.VisualScripting;
using UnityEngine;
using UnityEngine.UI;
public class bp : MonoBehaviour
{
    private Button myButton;
    public GameObject key;
    public GameObject paets;
    public GameObject buttonkey;
    void Start()
    {
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
    }

    private void OnButtonClick()
```

```
    {
        key.SetActive(false);
        paets.SetActive(true);

        buttonkey.SetActive(true);
        gameObject.SetActive(false);
    }
    void Update()
    {

    }
}

using System.Collections;
using System.Collections.Generic;
using System.Net.NetworkInformation;
using UnityEngine;
using UnityEngine.UI;

public class waterup : MonoBehaviour
{
    private GameManager gameManager;
    private Button myButton;
    public Text textprice;
    public Text textup;
    public Text textmax;
    public int price = 16;
    public int num = 0;
    public int point = 0;
    public bool max = false;
    public int maxpump = 0;
    private bool up1 = false;
    private bool up2 = false;
    private bool up3 = false;
    private bool up4 = false;
    private bool up5 = false;
    private AudioSource AS;
    public AudioClip soundbuy;
    public AudioClip soundmax;
    private void Start()
    {
        gameManager = GameManager.instance;
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        AS = GetComponent<AudioSource>();
    }

    private void OnButtonClick()
    {
        point = GameManager.instance.point;
        maxpump = GameManager.instance.pumpmax;
        if (max == false)
        {
            if (point >= 16 && num == 0)
            {
                num++;
                AS.PlayOneShot(soundbuy);
            }
            else if (point >= 25 && num == 1)
```

```
{
    num++;
    AS.PlayOneShot(soundbuy);
}
else if (point >= 44 && num == 2)
{
    num++;
    AS.PlayOneShot(soundbuy);
}
else if (point >= 60 && num == 3)
{
    num++;
    AS.PlayOneShot(soundbuy);
}
else if (point >= 76 && num == 4)
{
    num++;
    AS.PlayOneShot(soundbuy);
    max = true;
}
else
{
    AS.PlayOneShot(soundmax);
}
if (num == 1)
{
    if (point >= 16 && up1 == false)
    {
        point = point - 16;
        gameManager.Updatepoint(point);
        price = 25;
        maxpump = maxpump + 5;
        gameManager.Updatepumpmax(maxpump);
        textprice.text = "" + price;
        up1 = true;
    }
}
if (num == 2)
{
    if (point >= 25 && up2 == false)
    {
        point = point - 25;
        gameManager.Updatepoint(point);
        price = 44;
        maxpump = maxpump + 5;
        gameManager.Updatepumpmax(maxpump);
        textprice.text = "" + price;
        up2 = true;
    }
}
if (num == 3)
{
    if (point >= 44 && up3 == false)
    {
        point = point - 44;
        gameManager.Updatepoint(point);
        price = 60;
        maxpump = maxpump + 7;
        gameManager.Updatepumpmax(maxpump);
```

```
                    textprice.text = "" + price;
                    up3 = true;
                }
            }
            if (num == 4)
            {
                if (point >= 60 && up4 == false)
                {
                    point = point - 60;
                    gameManager.Updatepoint(point);
                    price = 76;
                    maxpump = maxpump + 7;
                    gameManager.Updatepumpmax(maxpump);
                    textprice.text = "" + price;
                    up4 = true;
                }
            }
            if (num == 5)
            {
                if (point >= 76 && up5 == false)
                {
                    point = point - 76;
                    gameManager.Updatepoint(point);
                    maxpump = maxpump + 11;
                    gameManager.Updatepumpmax(maxpump);
                    textprice.text = "";
                    textmax.text = "MAX";
                    up5 = true;
                }
            }
        }
        textup.text = "" + num;
    }

}

using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class runup : MonoBehaviour
{
    private GameManager gameManager;
    private Button myButton;
    public Text textprice;
    public Text textup;
    public Text textmax;
    public int price = 35;
    public int num = 0;
    public int point = 0;
    public bool max = false;
    public float speed = 0;
    private bool up1 = false;
    private bool up2 = false;
    private bool up3 = false;
    private bool up4 = false;
    private bool up5 = false;
    private AudioSource AS;
```

```
public AudioClip soundbuy;
public AudioClip soundmax;
private void Start()
{
    gameManager = GameManager.instance;
    myButton = GetComponent<Button>();
    myButton.onClick.AddListener(OnButtonClick);
    AS = GetComponent<AudioSource>();
}

private void OnButtonClick()
{
    point = GameManager.instance.point;
    speed = GameManager.instance.speedplayer;
    if (max == false)
    {
        if (point >= 28 && num == 0)
        {
            num++;
            AS.PlayOneShot(soundbuy);
        }
        else if (point >= 56 && num == 1)
        {
            num++;
            AS.PlayOneShot(soundbuy);
        }
        else if (point >= 80 && num == 2)
        {
            num++;
            AS.PlayOneShot(soundbuy);
        }
        else if (point >= 118 && num == 3)
        {
            num++;
            AS.PlayOneShot(soundbuy);
        }
        else if (point >= 160 && num == 4)
        {
            num++;
            AS.PlayOneShot(soundbuy);
            max = true;
        }
        else
        {
            AS.PlayOneShot(soundmax);
        }
        if (num == 1)
        {
            if (point >= 28 && up1 == false)
            {
                point = point - 28;
                gameManager.Updatepoint(point);
                price = 56;
                speed = speed+ 0.2f;
                gameManager.Updatespeedplayer(speed);
                textprice.text = "" + price;
                up1 = true;
            }
        }
```

```
            if (num == 2)
            {
                if (point >= 56 && up2 == false)
                {
                    point = point - 56;
                    gameManager.Updatepoint(point);
                    price = 80;
                    speed = speed + 0.5f;
                    gameManager.Updatespeedplayer(speed);
                    textprice.text = "" + price;
                    up2 = true;
                }
            }
            if (num == 3)
            {
                if (point >= 80 && up3 == false)
                {
                    point = point - 80;
                    gameManager.Updatepoint(point);
                    price = 118;
                    speed = speed + 0.5f;
                    gameManager.Updatespeedplayer(speed);
                    textprice.text = "" + price;
                    up3 = true;
                }
            }
            if (num == 4)
            {
                if (point >= 118 && up4 == false)
                {
                    point = point - 118;
                    gameManager.Updatepoint(point);
                    price = 160;
                    speed = speed + 0.7f;
                    gameManager.Updatespeedplayer(speed);
                    textprice.text = "" + price;
                    up4 = true;
                }
            }
            if (num == 5)
            {
                if (point >= 160 && up5 == false)
                {
                    point = point - 160;
                    gameManager.Updatepoint(point);
                    speed = speed + 0.7f;
                    gameManager.Updatespeedplayer(speed);
                    textprice.text = "";
                    textmax.text = "MAX";
                    up5 = true;
                }
            }
        }
        textup.text = "" + num;
    }
}


using System.Collections;
```

```
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class coinup : MonoBehaviour
{
    private GameManager gameManager;
    private Button myButton;
    public Text textprice;
    public Text textup;
    public Text textmax;
    public int price = 35;
    public int num = 0;
    public int point = 0;
    public bool max = false;
    public int coincarrot = 0;
    public int coinpumpkin = 0;
    public int coinpotato = 0;
    public int cointomato = 0;
    private bool up1 = false;
    private bool up2 = false;
    private bool up3 = false;
    private bool up4 = false;
    private bool up5 = false;
    private AudioSource AS;
    public AudioClip soundbuy;
    public AudioClip soundmax;
    private void Start()
    {
        gameManager = GameManager.instance;
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        AS = GetComponent<AudioSource>();
    }

    private void OnButtonClick()
    {
        point = GameManager.instance.point;
        coincarrot = GameManager.instance.coincarrot;
        coinpumpkin = GameManager.instance.coinpumpkin;
        coinpotato = GameManager.instance.coinpotato;
        cointomato = GameManager.instance.cointomato;
        if (max == false)
        {
            if (point >= 30 && num == 0)
            {
                num++;
                AS.PlayOneShot(soundbuy);
            }
            else if (point >= 63 && num == 1)
            {
                num++;
                AS.PlayOneShot(soundbuy);
            }
            else if (point >= 100 && num == 2)
            {
                num++;
                AS.PlayOneShot(soundbuy);
            }
```

```
else if (point >= 150 && num == 3)
{
    num++;
    AS.PlayOneShot(soundbuy);
}
else if (point >= 240 && num == 4)
{
    num++;
    AS.PlayOneShot(soundbuy);
    max = true;
}
else
{
    AS.PlayOneShot(soundmax);
}
if (num == 1)
{
    if (point >= 30 && up1 == false)
    {
        point = point - 30;
        gameManager.Updatepoint(point);
        price = 63;
        coincarrot = coincarrot + 4;
        coinpumpkin = coinpumpkin + 4;
        coinpotato = coinpotato + 4;
        gameManager.Updatecoincarrot(coincarrot);
        gameManager.Updatecoinpumpkin(coinpumpkin);
        gameManager.Updatecoinpotato(coinpotato);
        cointomato = cointomato + 4;
        gameManager.Updatecointomato(cointomato);
        textprice.text = "" + price;
        up1 = true;
    }
}
if (num == 2)
{
    if (point >= 63 && up2 == false)
    {
        point = point - 63;
        gameManager.Updatepoint(point);
        price = 100;
        coincarrot = coincarrot + 7;
        coinpumpkin = coinpumpkin + 7;
        coinpotato = coinpotato + 7;
        gameManager.Updatecoincarrot(coincarrot);
        gameManager.Updatecoinpumpkin(coinpumpkin);
        gameManager.Updatecoinpotato(coinpotato);
        cointomato = cointomato + 7;
        gameManager.Updatecointomato(cointomato);
        textprice.text = "" + price;
        up2 = true;
    }
}
if (num == 3)
{
    if (point >= 100 && up3 == false)
    {
        point = point - 100;
        gameManager.Updatepoint(point);
```

```
                    price = 150;
                    coincarrot = coincarrot + 11;
                    coinpumpkin = coinpumpkin + 11;
                    coinpotato = coinpotato + 11;
                    gameManager.Updatecoincarrot(coincarrot);
                    gameManager.Updatecoinpumpkin(coinpumpkin);
                    gameManager.Updatecoinpotato(coinpotato);
                    cointomato = cointomato + 11;
                    gameManager.Updatecointomato(cointomato);
                    textprice.text = "" + price;
                    up3 = true;
                }
            }
            if (num == 4)
            {
                if (point >= 150 && up4 == false)
                {
                    point = point - 150;
                    gameManager.Updatepoint(point);
                    price = 240;
                    coincarrot = coincarrot + 17;
                    coinpumpkin = coinpumpkin + 17;
                    coinpotato = coinpotato + 17;
                    gameManager.Updatecoincarrot(coincarrot);
                    gameManager.Updatecoinpumpkin(coinpumpkin);
                    gameManager.Updatecoinpotato(coinpotato);
                    cointomato = cointomato + 17;
                    gameManager.Updatecointomato(cointomato);
                    textprice.text = "" + price;
                    up4 = true;
                }
            }
            if (num == 5)
            {
                if (point >= 240 && up5 == false)
                {
                    point = point - 240;
                    gameManager.Updatepoint(point);
                    coincarrot = coincarrot + 23;
                    coinpumpkin = coinpumpkin + 23;
                    coinpotato = coinpotato + 23;
                    gameManager.Updatecoincarrot(coincarrot);
                    gameManager.Updatecoinpumpkin(coinpumpkin);
                    gameManager.Updatecoinpotato(coinpotato);
                    cointomato = cointomato + 23;
                    gameManager.Updatecointomato(cointomato);
                    textprice.text = "";
                    textmax.text = "MAX";
                    up5 = true;
                }
            }
        }
        textup.text = "" + num;
    }
}


using System.Collections;
using System.Collections.Generic;
```

```
using UnityEngine;
using UnityEngine.UI;

public class teleportwater : MonoBehaviour
{
    private GameManager gameManager;
    private Button myButton;
    public Text textprice;
    public Text textup;
    public Text textmax;
    public int price = 140;
    public int num = 0;
    public int point = 0;
    public bool max = false;
    private bool up1 = false;
    private AudioSource AS;
    public AudioClip soundbuy;
    public AudioClip soundmax;
    public int water = 0;
    private void Start()
    {
        gameManager = GameManager.instance;
        myButton = GetComponent<Button>();
        myButton.onClick.AddListener(OnButtonClick);
        AS = GetComponent<AudioSource>();
    }

    private void OnButtonClick()
    {
        point = GameManager.instance.point;
        water = GameManager.instance.itemwater;
        if (max == false)
        {
            if (point >= price && num == 0)
            {
                num++;
                AS.PlayOneShot(soundbuy);
                max = true;
            }
            else
            {
                AS.PlayOneShot(soundmax);
            }
            if (num == 1)
            {
                if (point >= price && up1 == false)
                {
                    point = point - price;
                    gameManager.Updatepoint(point);
                    water = 1;
                    gameManager.Updateitemwater(water);
                    textprice.text = "" + price;
                    up1 = true;
                    textprice.text = "";
                    textmax.text = "MAX";
                }
            }
        }
        textup.text = "" + num;
```

```
    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class boxstart : MonoBehaviour
{
    private GameManager gameManager;
    Animator animator;
    public int startbox = 0;
    public bool box = false;
    public float time = 0;
    public bool drop = false;
    public int seedcarrot = 0;
    public int seedpumpkin = 0;
    public int seedpotato = 0;
    public GameObject coincarrot;
    public GameObject coincarpumpkin;
    public GameObject coincarpotato;
    private GameObject go1;
    private GameObject go2;
    private GameObject go3;
    private Collider2D boxCollider2D;
    public int startdrop = 0;
    void Start()
    {
        gameManager = GameManager.instance;
        animator = GetComponent<Animator>();
        boxCollider2D = GetComponent<Collider2D>();
    }

    // Update is called once per frame
    void Update()
    {
        startbox = GameManager.instance.startbox;
        seedcarrot = GameManager.instance.carrotnum;
        seedpumpkin = GameManager.instance.pumpkinnum;
        seedpotato = GameManager.instance.potatonum;
        startdrop = GameManager.instance.startdrop;
       if (startbox == 1)
        {
            box = true;
            //boxCollider2D.isTrigger = false;
        }
       if (box == true)
        {
            //animator.SetBool("box", true);
            time += Time.deltaTime;
        }
        if (time >= 1.5 && drop == false)
        {
            for (int i = 0; i < seedpumpkin; i++)
            {
                Vector2 randomOffset = new Vector2(Random.Range(-3f, 3f),
Random.Range(-4f, 0f));
```

```
                Vector2 coinPosition = (Vector2)transform.position +
randomOffset;
                go1 = Instantiate(coincarpumpkin, coinPosition,
Quaternion.identity);
            }
            for (int i = 0; i < seedpotato; i++)
            {
                Vector2 randomOffset = new Vector2(Random.Range(-3f, 3f),
Random.Range(-4f, 0f));
                Vector2 coinPosition = (Vector2)transform.position +
randomOffset;
                go2 = Instantiate(coincarpotato, coinPosition,
Quaternion.identity);
            }
            for (int i = 0; i < seedcarrot; i++)
            {
                Vector2 randomOffset = new Vector2(Random.Range(-3f, 3f),
Random.Range(-4f, 0f));
                Vector2 coinPosition = (Vector2)transform.position +
randomOffset;
                go3 = Instantiate(coincarrot, coinPosition,
Quaternion.identity);
            }
            seedpotato = 0;
            seedcarrot = 0;
            seedpumpkin = 0;
            startdrop = 1;
            gameManager.Updatestartdrop(startdrop);
            gameManager.Updatecarrotnum(seedcarrot);
            gameManager.Updatepotatonum(seedpotato);
            gameManager.Updatepumpkinnum(seedpumpkin);
            drop = true;
        }
        if(time >= 2)
        {
            boxCollider2D.isTrigger = true;
            startbox = 0;
            gameManager.Updatestartdrop(startbox);
        }
        if (time >= 3.1)
        {
            time = 0;
            //animator.SetBool("box", false);
            box = false;
            drop = false;
        }
    }
}
```

มอนสเตอร์ตัวที่ 1

```
using System.Collections;
using System.Collections.Generic;
using Unity.VisualScripting;
using UnityEngine;

public class spawn : MonoBehaviour
{
```

```
private GameManager gameManager;
public int setspawn = 0;
public int spawnmonster = 0;
public float time = 0;
public float settime = 0;
public int setnight = 0;
public bool spawnstart = false;
public bool stopspawn = false;
public bool one = false;
public int respawn = 0;
private AudioSource AS;
public AudioClip soundwait;
public bool sound = false;
public bool soundplay = false;
public float timesound = 0;
void Start()
{
    gameManager = GameManager.instance;
    AS = GetComponent<AudioSource>();
}

void Update()
{
    setspawn = GameManager.instance.setspawnmonster;
    spawnmonster = GameManager.instance.spawnmonster;
    setnight = GameManager.instance.night;
    respawn = GameManager.instance.respawn;
    if(setnight == 1)
    {
        spawnstart = true;
    }
    else
    {
        sound = false;
        soundplay = false;
        spawnstart = false;
        stopspawn = false;
        one = false;
        respawn = 0;
        gameManager.Updaterespawn(respawn);
        spawnmonster = 0;
        gameManager.Updatespawnmonster(spawnmonster);
        setspawn = 0;
        gameManager.Updatesetspawnmonster(setspawn);
        time = 0;
        settime = 0;
    }
    if (spawnstart == true && stopspawn == false)
    {
        time += Time.deltaTime;
        if (time >= 30)
        {
            setspawn = Random.Range(1, 4);
            gameManager.Updatesetspawnmonster(setspawn);
            time = 0;
        }
        if(setspawn == 1 && stopspawn ==false)
        {
            sound = true;
```

```
                time = 0;
                settime += Time.deltaTime;
                if(settime >= 14)
                {
                    sound = false;
                    soundplay = false;
                    stopspawn = true;
                    settime = 0;
                }
            }
        }
        if(stopspawn == true && one == false)
        {
            one = true;
            spawnmonster = Random.Range(1, 3);
            gameManager.Updatespawnmonster(spawnmonster);
        }
        if (respawn == 1)
        {
            soundplay = false;
            stopspawn = false;
            one = false;
            respawn = 0;
            gameManager.Updaterespawn(respawn);
            spawnmonster = 0;
            gameManager.Updatespawnmonster(spawnmonster);
            setspawn = 0;
            gameManager.Updatesetspawnmonster(setspawn);
        }
        if(sound == true && soundplay ==false)
        {
            AS.PlayOneShot(soundwait);
            soundplay = true;
        }
    }
}


using System.Drawing;
using UnityEngine;

public class MonsterController : MonoBehaviour
{
    public GameObject uidead;
    public float speed = 5f;
    private Transform player;
    private GameManager gameManager;
    private Transform door;
    public int home = 0;
    public bool dead = false;
    private Transform respawn;
    public int spawn = 0;
    private Collider2D myCollider;
    private AudioSource AS;
    public AudioClip sounddoor;
    public AudioClip playerdead;
    void Start()
    {
        player = GameObject.FindGameObjectWithTag("Player").transform;
```

```
        door = GameObject.FindGameObjectWithTag("door").transform;
        respawn = GameObject.FindGameObjectWithTag("Respawn").transform;
        gameManager = GameManager.instance;
        myCollider = GetComponent<Collider2D>();
        AS = GetComponent<AudioSource>();
    }

    void Update()
    {
        home = GameManager.instance.home;
        spawn = GameManager.instance.respawn;
        if(home == 0 && dead == false) {
            if (player != null)
            {
                Vector3 direction = player.position - transform.position;
                transform.position =
Vector2.MoveTowards(transform.position, player.position, speed *
Time.deltaTime);
                if (direction.x < 0)
                {
                    transform.eulerAngles = new Vector3(0, 180, 0);
                }
                else
                {
                    transform.eulerAngles = new Vector3(0, 0, 0);
                }
            }
        }
        if(home == 1 && dead == false)
        {
            if (door != null)
            {
                Vector3 direction = door.position - transform.position;
                transform.position =
Vector2.MoveTowards(transform.position, door.position, speed *
Time.deltaTime);
                if (direction.x < 0)
                {
                    transform.eulerAngles = new Vector3(0, 180, 0);
                }
                else
                {
                    transform.eulerAngles = new Vector3(0, 0, 0);
                }
            }
        }
        if(dead == true)
        {
            if (respawn != null)
            {
                Vector3 direction = respawn.position - transform.position;
                transform.position =
Vector2.MoveTowards(transform.position, respawn.position, speed *
Time.deltaTime);
                if (direction.x < 0)
                {
                    transform.eulerAngles = new Vector3(0, 180, 0);
                }
                else
```

```
                {
                    transform.eulerAngles = new Vector3(0, 0, 0);
                }
            }
        }
    }
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "door")
        {
            dead = true;
            AS.PlayOneShot(sounddoor);
        }
        if (collision.gameObject.tag == "Player")
        {
            Destroy(collision.gameObject);
            AS.PlayOneShot(playerdead);
            uidead.SetActive(true);
        }
        if (collision.gameObject.tag == "Respawn")
        {
            Destroy(gameObject);
            spawn = 1;
            gameManager.Updaterespawn(spawn);
            dead = false;
        }
        if (collision.gameObject.tag == "boxmonster")
        {
            myCollider.isTrigger = true;
        }
    }
}
```

มอนสเตอร์ตัวที่ 2

```
using Pathfinding;
using UnityEngine;
using UnityEngine.Experimental.GlobalIllumination;

public class MonsterController2 : MonoBehaviour
{
    public AIController targetplayer;
    public setaiplayer setplayer;
    public targetbox tarbox;
    public setaibox setbox;
    public targethome tarhome;
    public setaihome sethome;
    private AudioSource AS;
    public AudioClip sound;
    public AudioClip playerdead;
    public AudioClip soundmonster1;
    public AudioClip soundmonster2;
    public AudioClip soundmonster3;
    public AudioClip soundmonster4;
    public GameObject uidead;
    public int night = 0;
    public int home =0;
    public float time = 0;
    public float timesound = 0;
```

```
public bool stopsound = false;
public int rndsound = 0;
void Start()
{
    AS = GetComponent<AudioSource>();
    targetplayer = GetComponent<AIController>();
    setplayer = GetComponent<setaiplayer>();
    tarbox = GetComponent<targetbox>();
    setbox = GetComponent<setaibox>();
    tarhome = GetComponent<targethome>();
    sethome = GetComponent<setaihome>();
}

void Update()
{
    night = GameManager.instance.night;
    home = GameManager.instance.home;
    if(night == 1 )
    {
        time += Time.deltaTime;
        if( time >= 5)
        {
            timesound += Time.deltaTime;
           /* if (!AS.isPlaying)
            {
                AS.clip = sound;
                AS.Play();
            }*/
            if (stopsound == false)
            {
                stopsound = true;
                AS.PlayOneShot(soundmonster1);
            }
            if (timesound >= 12)
            {
                timesound = 0;
                rndsound = Random.Range(1, 3);
                if(rndsound == 1)
                {
                    rndsound = 0;
                    AS.PlayOneShot(soundmonster2);
                }
                if (rndsound == 2)
                {
                    rndsound = 0;
                    AS.PlayOneShot(soundmonster3);
                }
                if (rndsound == 3)
                {
                    rndsound = 0;
                    AS.PlayOneShot(soundmonster4);
                }
            }
        }
        if( home == 1)
        {
            tarbox.enabled = true;
            setbox.enabled = true;
            targetplayer.enabled = false;
```

```
                setplayer.enabled = false;
                tarhome.enabled = false;
                sethome.enabled = false;
            }
            if( home == 0)
            {
                tarbox.enabled = false;
                setbox.enabled = false;
                targetplayer.enabled = true;
                setplayer.enabled = true;
                tarhome.enabled = false;
                sethome.enabled = false;
            }
        }
        else
        {
            time = 0;
            tarbox.enabled = false;
            setbox.enabled = false;
            targetplayer.enabled = false;
            setplayer.enabled = false;
            tarhome.enabled = true;
            sethome.enabled = true;
        }
    }
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "Player")
        {
            Destroy(collision.gameObject);
            AS.PlayOneShot(playerdead);
            uidead.SetActive(true);
        }
        if (collision.gameObject.tag == "Respawn")
        {
            AS.Stop();
            stopsound = false;
            timesound = 0;
        }
    }
}


using UnityEngine;
using Pathfinding;

public class AIController : MonoBehaviour
{
    public Transform target;

    private Seeker seeker;
    private AIPath aiPath;

    void Start()
    {
        seeker = GetComponent<Seeker>();
        aiPath = GetComponent<AIPath>();
        if (seeker == null || aiPath == null)
```

```
            Debug.LogError("Seeker or AIPath not found on " +
gameObject.name);

        if (target == null)
            Debug.LogError("Target not assigned to " + gameObject.name);

        if (seeker.IsDone())
        {
            seeker.StartPath(transform.position, target.position,
OnPathComplete);
        }
    }

    private void OnPathComplete(Path p)
    {
        if (!p.error)
        {
            aiPath.SetPath(p);
        }
    }

    void Update()
    {

        if (aiPath.desiredVelocity.x > 0.01f)
        {
            transform.eulerAngles = new Vector3(0, 0, 0);
        }
        else if (aiPath.desiredVelocity.x < -0.01f)
        {
            transform.eulerAngles = new Vector3(0, 180, 0);
        }
    }
}


using UnityEngine;
using System.Collections;

namespace Pathfinding
{
    [UniqueComponent(tag = "ai.destination")]

    public class setaibox : VersionedMonoBehaviour
    {
        public Transform target;
        IAstarAI ai;

        void OnEnable()
        {
            ai = GetComponent<IAstarAI>();
            if (ai != null) ai.onSearchPath += Update;
        }

        void OnDisable()
        {
            if (ai != null) ai.onSearchPath -= Update;
        }
        void Update()
```

```
        {
            if (target != null && ai != null) ai.destination =
target.position;
        }
    }
}


using UnityEngine;
using Pathfinding;

public class targetbox : MonoBehaviour
{
    public Transform target;

    private Seeker seeker;
    private AIPath aiPath;

    void Start()
    {
        seeker = GetComponent<Seeker>();
        aiPath = GetComponent<AIPath>();
        if (seeker == null || aiPath == null)
            Debug.LogError("Seeker or AIPath not found on " +
gameObject.name);

        if (target == null)
            Debug.LogError("Target not assigned to " + gameObject.name);

        if (seeker.IsDone())
        {
            seeker.StartPath(transform.position, target.position,
OnPathComplete);
        }
    }

    private void OnPathComplete(Path p)
    {
        if (!p.error)
        {
            aiPath.SetPath(p);
        }
    }

    void Update()
    {
        if (aiPath.desiredVelocity.x > 0.01f)
        {
            transform.eulerAngles = new Vector3(0, 0, 0);
        }
        else if (aiPath.desiredVelocity.x < -0.01f)
        {
            transform.eulerAngles = new Vector3(0, 180, 0);
        }
    }
}


using UnityEngine;
```

```
using Pathfinding;

public class targethome : MonoBehaviour
{
    public Transform target;

    private Seeker seeker;
    private AIPath aiPath;

    void Start()
    {
        seeker = GetComponent<Seeker>();
        aiPath = GetComponent<AIPath>();
        if (seeker == null || aiPath == null)
            Debug.LogError("Seeker or AIPath not found on " +
gameObject.name);

        if (target == null)
            Debug.LogError("Target not assigned to " + gameObject.name);
        if (seeker.IsDone())
        {
            seeker.StartPath(transform.position, target.position,
OnPathComplete);
        }
    }

    private void OnPathComplete(Path p)
    {
        if (!p.error)
        {
            aiPath.SetPath(p);
        }
    }

    void Update()
    {

        if (aiPath.desiredVelocity.x > 0.01f)
        {
            transform.eulerAngles = new Vector3(0, 0, 0);
        }
        else if (aiPath.desiredVelocity.x < -0.01f)
        {
            transform.eulerAngles = new Vector3(0, 180, 0);
        }
    }
}
```

ปุ่ม ESC

```
using UnityEngine;

public class uiesc : MonoBehaviour
{
    private bool isPaused = false;
    public GameObject ui;
    public GameObject uistat;
    public int num = 0;
```

```
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.Escape))
        {
            num++;
            if (num == 1)
            {
                ui.SetActive(true);
                uistat.SetActive(false);
            }
            else
            {
                ui.SetActive(false);
                uistat.SetActive(true);
                num = 0;
            }
        }
    }


}


using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class RestartButton : MonoBehaviour
{
    private Button restartButton;
    private GameManager gameManager;

    void Start()
    {

        restartButton = GetComponent<Button>();
        gameManager = FindObjectOfType<GameManager>();

        restartButton.onClick.AddListener(RestartGame);
    }

    void RestartGame()
    {
        Time.timeScale = 1f;
        gameManager.RestartGame();
        SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex,
LoadSceneMode.Single);
    }
}


using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class remenu : MonoBehaviour
```

```
{
    private GameManager gameManager;
    private Button restartButton;
    public string targetSceneName = "Menu";
    void Start()
    {
        restartButton = GetComponent<Button>();
        gameManager = FindObjectOfType<GameManager>();
        restartButton.onClick.AddListener(RestartGame);
    }

    void RestartGame()
    {
        Time.timeScale = 1f;
        gameManager.RestartGame();
        SceneManager.LoadScene(targetSceneName);
    }
}
```

Ui bar

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class bar : MonoBehaviour
{
    public int pointshop = 0;
    public int seedtomato = 0;
    public int pointui = 0;
    public int seedcarrot = 0;
    public int seedpumpkin = 0;
    public int seedpotato = 0;
    public int pointmax = 0;
    public Text textpointshop;
    public Text textpointui;
    public Text texttomato;
    public Text textpotato;
    public Text textcarrot;
    public Text textpumpkin;
    public Text textuideadpointmax;
    public Text textuiwinpointmax;

    private GameManager gameManager;
    void Start()
    {
        gameManager = GameManager.instance;
    }

    void Update()
    {
        pointshop = GameManager.instance.point;
        seedtomato = GameManager.instance.Seedtomato;
        pointui = GameManager.instance.point;
        seedcarrot = GameManager.instance.seedcarrot;
        seedpumpkin = GameManager.instance.seedpumpkin;
        seedpotato = GameManager.instance.seedpotato;
```

```
        pointmax = GameManager.instance.pointmax;
        textpointui.text = "" + pointui;
        textcarrot.text = "" + seedcarrot;
        textpumpkin.text = "" + seedpumpkin;
        textpotato.text = "" + seedpotato;
        textpointshop.text = "" + pointshop;
        texttomato.text = "" + seedtomato;
        textuideadpointmax.text = "TOTAL POINT: " + pointmax;
        textuiwinpointmax.text = "TOTAL POINT: " + pointmax;
    }
}
```

GameManager

```
//using UnityEditor.ShaderGraph.Internal;
using UnityEngine;
using UnityEngine.SceneManagement;

public class GameManager : MonoBehaviour
{
    public GameObject main;
    public static GameManager instance;
    public int waterValue = 15;
    public int point = 0;
    public int Seedtomato = 5;
    public int animatorwa = 0;
    public int pointmax = 0;
    public int cointomato = 7;
    public int pumpwater = 0;
    public int seedpumpkin = 0;
    public int seedcarrot = 0;
    public int seedpotato = 0;
    public int seedbeans = 0;
    public int coinpumpkin = 23;
    public int coincarrot = 13;
    public int coinpotato = 36;
    public int coinbeans = 20;
    public int shopstop = 0;
    public int redkey = 0;
    public int bluekey = 0;
    public int pinkkey = 0;
    public int imgstop = 0;
    public int carrotnum = 0;
    public int pumpkinnum = 0;
    public int potatonum = 0;
    public int totalprice = 0;
    public int starttrade = 0;
    public int startbox = 0;
    public int pumpmax = 15;
    public float speed = 3.0f;
    public float speedplayer = 0f;
    public int startdrop = 0;
    public int item1 = 0;
    public int item2 = 0;
    public int item3 = 0;
    public int item4 = 0;
    public int itemwater = 0;
    public int spawnwater = 0;
    public int tab = 0;
```

```
public int stoptab = 0;
public int walksound = 0;

public int setspawnmonster = 0;
public int spawnmonster = 0;
public int night = 0;
public int home = 0;
public int respawn = 0;
private void Awake()
{
    if (instance == null)
    {
        instance = this;
        DontDestroyOnLoad(gameObject);
    }
}
public void UpdateWater(int newWater)
{
    waterValue = newWater;
}
public void Updatepoint(int newpoint)
{
    point = newpoint;
}
public void UpdateSeedtomato(int newSeedtomato)
{
    Seedtomato = newSeedtomato;
}
public void Updateanimatorwa(int newanimatorwa)
{
    animatorwa = newanimatorwa;
}
public void Updatepointmax(int newpointmax)
{
    pointmax = newpointmax;
}
public void Updatepumpwater(int newpumpwater)
{
    pumpwater = newpumpwater;
}
public void Updateseedpumpkin(int newseedpumpkin)
{
    seedpumpkin = newseedpumpkin;
}
public void Updateseedcarrot(int newseedcarrot)
{
    seedcarrot = newseedcarrot;
}
public void Updateseedpotato(int newseedpotato)
{
    seedpotato = newseedpotato;
}
public void Updateseedbeans(int newseedbeans)
{
    seedbeans = newseedbeans;
}
public void Updateshopstop(int newshopstop)
{
    shopstop = newshopstop;
```

```
}
public void Updateredkey(int newredkey)
{
    redkey = newredkey;
}
public void Updatebluekey(int newbluekey)
{
    bluekey = newbluekey;
}
public void Updatepinkkey(int newpinkkey)
{
    pinkkey = newpinkkey;
}
public void Updateimgstop(int newimgstop)
{
    imgstop = newimgstop;
}
public void Updatecarrotnum(int newcarrotnum)
{
    carrotnum = newcarrotnum;
}
public void Updatepumpkinnum(int newpumpkinnum)
{
    pumpkinnum = newpumpkinnum;
}
public void Updatepotatonum(int newpotatonum)
{
    potatonum = newpotatonum;
}
public void Updatetotalprice(int newtotalprice)
{
    totalprice = newtotalprice;
}
public void Updatestarttrade(int newstarttrade)
{
    starttrade = newstarttrade;
}
public void Updatestartbox(int newstartbox)
{
    startbox = newstartbox;
}
public void Updatepumpmax(int newpumpmax)
{
    pumpmax = newpumpmax;
}
public void Updatespeed(float newspeed)
{
    speed = newspeed;
}
public void Updatespeedplayer(float newspeedplayer)
{
    speedplayer = newspeedplayer;
}
public void Updatecoincarrot(int newcoincarrot)
{
    coincarrot = newcoincarrot;
}
public void Updatecoinpumpkin(int newcoinpumpkin)
{
```

```
    coinpumpkin = newcoinpumpkin;
}
public void Updatecoinpotato(int newcoinpotato)
{
    coinpotato = newcoinpotato;
}
public void Updatecointomato(int newcointomato)
{
    cointomato = newcointomato;
}
public void Updatestartdrop(int newstartdrop)
{
    startdrop = newstartdrop;
}
public void Updateitem1(int newitem1)
{
    item1 = newitem1;
}
public void Updateitem2(int newitem2)
{
    item2 = newitem2;
}
public void Updateitem3(int newitem3)
{
    item3 = newitem3;
}
public void Updateitem4(int newitem4)
{
    item4 = newitem4;
}
public void Updatewalksound(int newwalksound)
{
    walksound = newwalksound;
}
public void Updateitemwater(int newitemwater)
{
    itemwater = newitemwater;
}
public void Updatespawnwater(int newspawnwater)
{
    spawnwater = newspawnwater;
}
public void Updatetab(int newtab)
{
    tab = newtab;
}
public void Updatestoptab(int newstoptab)
{
    stoptab = newstoptab;
}
public void Updatesetspawnmonster(int newsetspawnmonster)
{
    setspawnmonster = newsetspawnmonster;
}
public void Updatespawnmonster(int newspawnmonster)
{
    spawnmonster = newspawnmonster;
}
public void Updatenight(int newnight)
```

```
    {
        night = newnight;
    }
    public void Updatehome(int newhome)
    {
        home = newhome;
    }
    public void Updaterespawn(int newrespawn)
    {
        respawn = newrespawn;
    }
    public void RestartGame()
    {
        DestroyPlayerCharacter();
        SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex,
LoadSceneMode.Single);
    }
    void DestroyPlayerCharacter()
    {
        GameObject player = GameObject.FindGameObjectWithTag("Player");
        if (player != null)
        {
            Destroy(player);
        }
        GameObject trade1 = GameObject.FindGameObjectWithTag("shop");
        if (trade1 != null)
        {
            Destroy(trade1);
        }
        GameObject UI = GameObject.FindGameObjectWithTag("uiall");
        if (UI != null)
        {
            Destroy(UI);
        }
        GameObject waterpump =
GameObject.FindGameObjectWithTag("waterpumpdel");
        if (waterpump != null)
        {
            Destroy(waterpump);
        }
        GameObject box = GameObject.FindGameObjectWithTag("boxdel");
        if (box != null)
        {
            Destroy(box);
        }
        GameObject Ghost1 = GameObject.FindGameObjectWithTag("monster");
        if (Ghost1 != null)
        {
            Destroy(Ghost1);
        }
    waterValue = 15;
    point = 0;
    Seedtomato = 5;
    animatorwa = 0;
    pointmax = 0;
    cointomato = 7;
    pumpwater = 0;
    seedpumpkin = 0;
    seedcarrot = 0;
```

```
    seedpotato = 0;
    seedbeans = 0;
    coinpumpkin = 23;
    coincarrot = 13;
    coinpotato = 36;
    coinbeans = 20;
    shopstop = 0;
    redkey = 0;
    bluekey = 0;
    pinkkey = 0;
    imgstop = 0;
    carrotnum = 0;
    pumpkinnum = 0;
    potatonum = 0;
    totalprice = 0;
    starttrade = 0;
    startbox = 0;
    pumpmax = 15;
    speed = 3.0f;
    speedplayer = 0f;
    startdrop = 0;
    item1 = 0;
    item2 = 0;
    item3 = 0;
    item4 = 0;
    itemwater = 0;
    spawnwater = 0;
    tab = 0;
    stoptab = 0;
    walksound = 0;

    setspawnmonster = 0;
    spawnmonster = 0;
    night = 0;
    home = 1;
    respawn = 0;
}
}
```

ประวัติผู้จัดทำ

ประวัติผู้จัดทำโครงการ

ชื่อ นายภุชิสสะ แซ่ลี้

เกิดเมื่อวันที่ 6 เดือน มกราคม พ.ศ. 2547

ที่อยู่ 3/4 หมู่9 ต.หนองตาคง อ.โป่งน้ำร้อน จ.จันทบุรี

หมายเลขโทรศัพท์ 082-4101248

E-Mail : Phuchitsa96855@gmail.com

การศึกษา

ชั้นประถมศึกษา จากโรงเรียนบ้านจางวาง

ชั้นมัธยมศึกษาตอนต้น จากโรงเรียนหนองตาคงพิทยาคาร

ชั้นประกาศนียบัตรวิชาชีพ จากวิทยาลัยเทคนิคจันทบุรี

ขณะนี้กำลังศึกษาอยู่ในชั้นประกาศนียบัตรวิชาชีพชั้นสูงที่วิทยาลัยเทคนิคจันทบุรี

ประวัติผู้จัดทำโครงการ

ชื่อ นายศุภกิตติ์ แพเมือง

เกิดเมื่อวันที่14 เดือน พฤษภาคม พ.ศ. 2547

อยู่ที่ 51/1 หมู่2 ต.อ่างคีรี อ.มะขาม จ.จันทบุรี

หมายเลขโทรศัพท์ 084-2793591

E-Mail : artkonart32@gmail.com

การศึกษา

ชั้นประถมศึกษา จากโรงเรียนวัดบ้านอ่าง

ชั้นมัธยมศึกษาตอนต้น จากโรงเรียนมะขามสรรเสริญ

ชั้นประกาศนียบัตรวิชาชีพ จากวิทยาลัยเทคนิคจันทบุรี

ขณะนี้กำลังศึกษาอยู่ในชั้นประกาศนียบัตรวิชาชีพชั้นสูงที่วิทยาลัยเทคนิคจันทบุรี