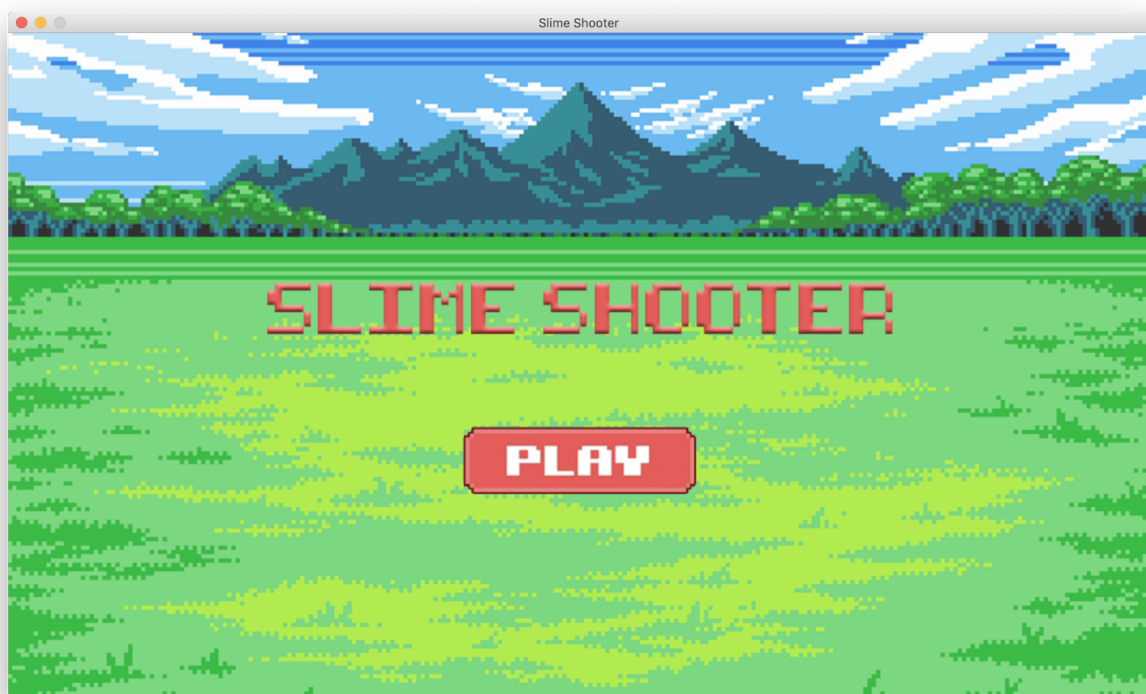


Slime Shooter (Group EiEi)

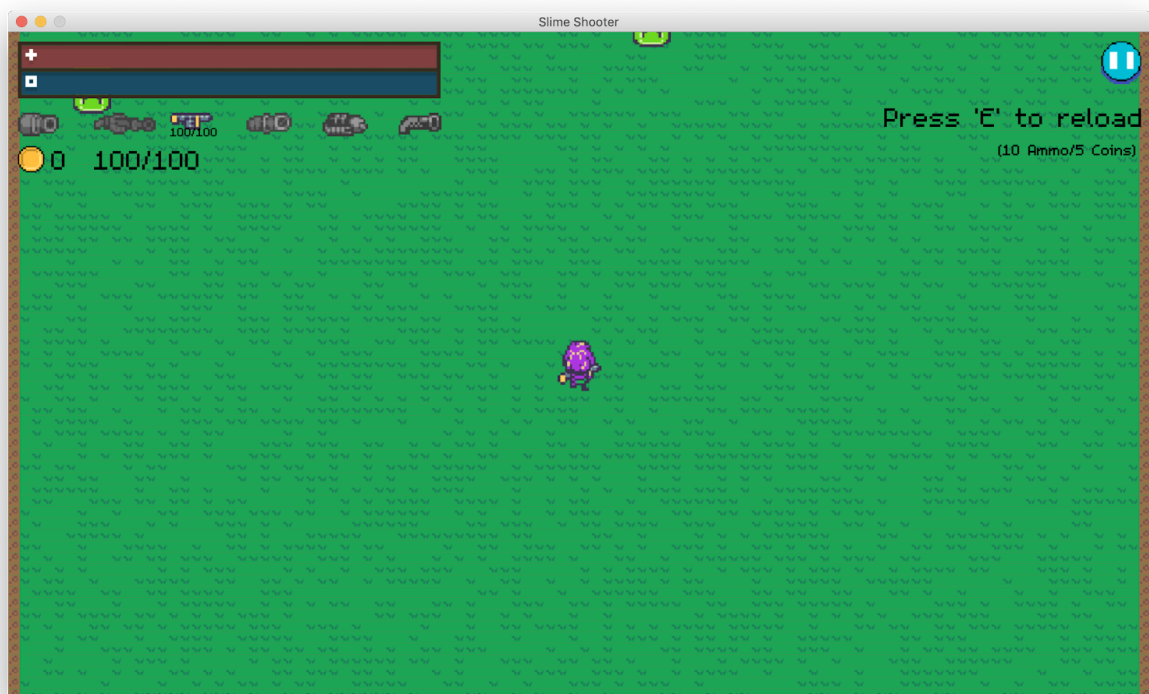
Slime Shooter เป็นเกมแนว 8 bit Top-down Shooter โดยมีจุดมุ่งหมายคือให้ผู้เล่นจัดการสไลม์ที่จะเกิดขึ้นมาเรื่อยๆ ให้ได้จำนวนมากที่สุด ผู้เล่นสามารถใช้อาวุธที่มีให้ทั้งหมด 6 ประเภท ได้แก่ Machine Gun, Flamethrower, Cannon, Matter, Shotgun และ Rocket Launcher โดยอาวุธแต่ละประเภทจะมีคุณสมบัติที่แตกต่างกันไป เมื่อจัดการสไลม์ได้หนึ่งตัวผู้เล่นจะได้รับเหรียญตั้งแต่ 1 ถึง 10 เหรียญ และเนื่องจากในตอนแรกกระสุนที่ให้มาของอาวุธแต่ละประเภทมีอยู่จำกัด ผู้เล่นจำเป็นต้องใช้เหรียญที่สะสมมาซื้อกระสุนเพิ่มเอง ซึ่งอาวุธแต่ละประเภทก็จะมีราคากระสุนที่แตกต่างกัน ส่วนรายละเอียดของคุณสมบัติอาวุธ และราคากระสุนจะกล่าวในย่อหน้าถัดไป

เมื่อเริ่มต้นเปิดเกม ผู้เล่นจะพบกับหน้าต่างเมนูหลัก ดังภาพที่ 1



ภาพที่ 1 หน้าต่างเมนูหลักของเกม

เมื่อผู้เล่นกดปุ่ม “PLAY” เกมจะถูกเริ่มขึ้น โดยมีหน้าต่างดังภาพที่ 2



ภาพที่ 2 หน้าต่างเมื่อกดเริ่มเกม

Graphical User Interface

ในส่วนของ GUI จะประกอบด้วยส่วนหลักๆ ได้แก่ แถบเลือด (สีแดง) / กระสุน (สีน้ำเงิน) จำนวนเหรียญสะสม สถานะอาวุธ และปุ่มหยุดเกมชั่วคราว เมื่อเกมเริ่ม ผู้เล่นจะมีค่าเริ่มต้นต่างๆดังนี้

HP	100
Ammo (and magazine size)	Machine Gun: 200 Flamethrower: 200 Matter: 50 Shotgun: 7 Cannon: 20 Rocket Launcher: 3
Coin	0

นอกจากนี้ HP ของผู้เล่นจะเพิ่มขึ้นเอง (Regenerate) 0.3 หน่วยต่อวินาที

Slimes and how to kill them

สไลม์จะถูกปล่อยออกมาในบริเวณแผนที่อย่างสม่ำเสมอทุกๆประมาณ 2.5 วินาที และจะวิ่งเข้าหาตัวผู้เล่นตลอดเวลา เมื่อสไลม์เข้าชนตัวผู้เล่น HP ของผู้เล่นจะลดลง 20 หน่วย จากนั้นสไลม์จะหายไป ในตอนเริ่มเกม ผู้เล่นจะได้รับอาวุธ Machine Gun เป็นอาวุธเริ่มต้น หน้าทีของผู้เล่นคือให้บังคับทิศทางของกระสุนด้วยเมาส์ จากนั้นกดยิงด้วยปุ่ม Spacebar ผู้เล่นสามารถสลับประเภทปืนได้ (หากได้เก็บปืนอื่นๆมาเพิ่ม) โดยกดปุ่ม Q หรือใช้ mouse wheel โดยปืนและจำนวนกระสุนที่ผู้เล่นมีทั้งหมดจะปรากฏให้เห็นในแถบสถานะด้านบนซ้าย และกดซื้อกระสุนโดยการกดปุ่ม E โดยจะทำการซื้อกระสุนของปืนประเภทที่ผู้เล่นกำลังถืออยู่ โดยมีอัตราราคาบอกทางด้านบนขวา สไลม์จะตายและระเบิดตัวเองเมื่อ HP เหลือ 0 (เริ่มเกิดจะมี HP 100 หน่วยเช่นเดียวกับตัวผู้เล่น) ดังภาพที่ 3 ผู้เล่นจะได้เหรียญเป็นจำนวนสุ่มตั้งแต่ 1 ถึง 10 เหรียญเมื่อจัดการสไลม์ได้หนึ่งตัว



ภาพที่ 3 เมื่อสไลม์ตายจะระเบิดตัวเอง



เกมจะจบเมื่อ HP ของผู้เล่นเหลือ 0 จะมีหน้าต่าง Game Over ขึ้นมาดังภาพที่ 4 ให้คลิกที่ตำแหน่งใดก็ได้บนหน้าจอเพื่อกลับไปยังเมนูหลัก (สำหรับปุ่ม Pause ก็เช่นกัน ให้คลิกที่ตำแหน่งใดก็ได้เพื่อเล่นเกมต่อ)



ภาพที่ 4 หน้าต่าง Game Over และ หน้าต่าง Paused

Power-Ups

ในเกมนี้จะมีการสุ่มดรอปของไอเทม power-up ซึ่งมีทั้งหมด 2 ประเภทคือ Health Box และ Damage Multiplier โดยทุกๆ 2.5 วินาทีจะมีโอกาสปรากฏขึ้นบนแผนที่ประเภทละ 10 เปอร์เซ็นต์ โดยจะปรากฏเพียง 20 วินาที จากนั้นจะหายไปหากผู้เล่นไม่ได้เก็บ และ Power up จะมีผลทันทีเมื่อผู้เล่นเดินไปเก็บ คุณสมบัติของแต่ละประเภทเป็นไปดังตาราง

Power-Up	คุณสมบัติ	โอกาสดรอป
Damage Multiplier 	เพิ่มความแรงของกระสุนปืนทุกประเภทขึ้น 1.5 เท่า เป็นเวลา 5 วินาที	ร้อยละ 10
Health Box 	เพิ่ม HP ของผู้เล่น 30 หน่วย	ร้อยละ 10
















สำหรับ Damage Multiplier ในขณะที่กำลังใช้งาน (5 วินาที) แถบของกระสุนจะเปลี่ยนจากสีน้ำเงินเป็นสีเหลืองดังภาพที่ 5



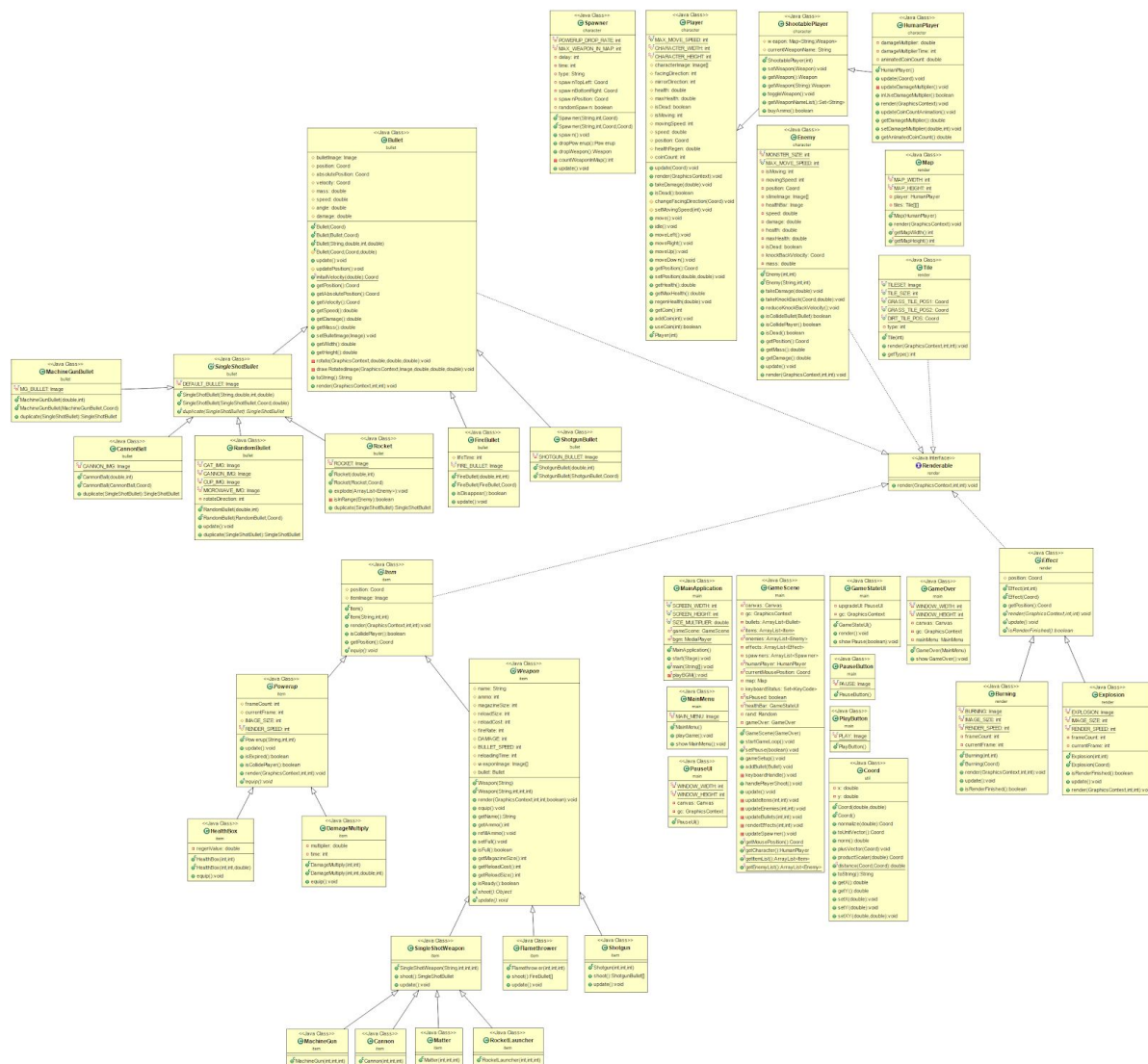
ภาพที่ 5 แถบของกระสุนจะเปลี่ยนเป็นสีเหลืองเป็นระยะเวลา 5 วินาที หลังจากเก็บ Damage Multiplier

Weapon types

เกมนี้จะมีอาวุธอยู่ทั้งหมด 6 ประเภท โดยจะสุ่มดรอปทุกๆ 2.5 วินาทีเช่นเดียวกับ Power ups แต่ละประเภทจะมีคุณสมบัติและราคากระสุนที่แตกต่างกัน หากผู้เล่นเก็บปืนประเภทใดๆได้ จะสามารถใช้อาวุธนั้นได้ตลอด ตราบที่ยังมีกระสุน และหากผู้เล่นเก็บได้ปืนซ้ำ กระสุนของปืนนั้นๆจะถูกเติมจนเต็มทันที ในการเล่น ผู้เล่นจะต้องเลือกใช้ประเภทของอาวุธให้เหมาะสมกับสถานการณ์ และเลือกซื้อกระสุนอย่างรอบคอบ เนื่องจากมีเหรียญอยู่จำกัด คุณสมบัติหลักๆของอาวุธแต่ละประเภทเป็นไปดังตาราง

ประเภทอาวุธ (Drop rate)	ความแรง	เวลาการรีโหลด	คุณสมบัติพิเศษ	ราคากระสุน (เหรียญ/นัด)
Machine Gun (6%)  	10	10	-	10 / 20
Flamethrower (2%)  	15	10	ปล่อยไฟเป็นวงกว้าง ± 30 องศา	12 / 20
Cannon (6%)  	50	30	กระสุนปืนใหญ่จะมีค่า knockback (ผลักสไลม์ให้ถอยหลังไป) แรงกว่ากระสุนประเภทอื่น	10 / 5
Matter (5%)     	(ขึ้นอยู่กับประเภทกระสุน)	30	กระสุนจะถูกปล่อยออกมาจากเครื่องกำเนิดสสารอย่างสุ่ม โดยมีกระสุน 4 ประเภท ได้แก่ แมว (Damage 20) กระสุนปืนใหญ่ (Damage 50) แก้วกาแฟ (Damage 15) ไมโครเวฟ (Damage 30)	7 / 10
Shotgun (5%)  	10	50	กระสุนจะกระจายเป็น 10 นัด เมื่อยิงหนึ่งครั้ง โดยจะกระจายในช่วง ± 15 องศา	12 / 3
Rocket Launcher (2%)  	80	50	กระสุนจะระเบิดเมื่อโดนตัวสไลม์ แรงระเบิดจะส่งผลกระทบไปยังสไลม์รอบๆ ในรัศมี 100 หน่วย	30 / 1

แผนภาพ UML เป็นไปดั่งภาพที่ 6 โดยจะแบ่งออกเป็น 6 package คือ util main bullet item render และ character โดยจะอธิบายแต่ละแพคเกจตามลำดับ



หมายเหตุ : เครื่องหมาย Access Modifier แต่ละประเภทเป็นดังนี้

(FINAL) จะใช้ชื่อตัวแปรเป็น CONSTANT_CASE

1. Package util

ประกอบด้วยคลาส Coord ใช้แทนพิกัด (x, y) หรือเวกเตอร์บนระนาบ 2 มิติ และมีเมธอดที่จำเป็นเกี่ยวกับการคำนวณทางคณิตศาสตร์ของเวกเตอร์

1.1 Class util.Coord

1.1.1 Field

- double x	พิกัดแกน x
- double y	พิกัดแกน y

1.1.2 Constructor

+ Coord()	สร้างพิกัด (0, 0)
+ Coord(double x, double y)	สร้างพิกัด (x, y)

1.1.3 Methods

+ double norm()	คืนค่า magnitude ของเวกเตอร์นี้ (Euclidean norm) คำนวณได้จาก $\sqrt{x^2 + y^2}$
+ void plusVector(Coord rsh)	นำเวกเตอร์ rsh มาบวกกับเวกเตอร์นี้
+ void productScalar(double c)	นำค่าคงที่ c มาคูณกับเวกเตอร์นี้
+ Coord toUnitVector()	คืนค่า Coord ที่เป็นเวกเตอร์หนึ่งหน่วยของเวกเตอร์นี้
+ Coord normalize(double c)	คืนค่า Coord ที่เอาค่าคงที่ c มาคูณกับเวกเตอร์หนึ่งหน่วยของเวกเตอร์นี้
+ <u>double distance(Coord a, Coord b)</u>	หาระยะห่างระหว่างสองเวกเตอร์ คำนวณได้จาก $\sqrt{(\Delta x)^2 + (\Delta y)^2}$
+ String toString()	คืนค่าสตริงในรูปแบบ "Coord(x, y)"
+ void setXY(double x, double y)	Setter โดยมีพารามิเตอร์ทั้งสองค่า
+ getter & setter ของทุกฟิลด์	

2. Package main

ประกอบด้วยคลาสหลักและคลาสที่เกี่ยวข้องกับ GUI รวมไปถึงคลาสที่จัดการกับ game loop

2.1 Class main.MainApplication extends Application (Main Class)

2.1.1 Field

+ <u>int SCREEN_WIDTH</u>	ความกว้างของหน้าต่าง (1200 พิกเซล)
+ <u>int SCREEN_HEIGHT</u>	ความสูงของหน้าต่าง (700 พิกเซล)
+ <u>double SIZE_MULTIPLIER</u>	ค่าคงที่สำหรับขยายขนาดการ render ภาพ ถูกตั้งให้เป็น 1 เท่า
- <u>GameScene gameScene</u>	คลาสของ game loop
- <u>MediaPlayer bgm</u>	ตัวเล่น background music

2.1.2 Method

+ <u>void main(String[] args)</u>	เมธอด main ใช้เพื่อรันโปรแกรม และมีการเรียกใช้เมธอด playBGM() ในเมธอดนี้ด้วย
+ <u>void start(Stage primaryStage)</u>	เมธอดของ javafx โดยจะทำการตั้งขนาดหน้าต่าง ตั้งชื่อ title และเพิ่ม StackPane ของ gameScene mainMenu และ gameOver เข้า root ตามลำดับ พร้อมทั้งตั้งค่าไม่ให้เปลี่ยนขนาดหน้าต่างได้
- <u>void playBGM()</u>	สร้าง Thread ใหม่ให้ทำการดึง resource เพลง และเล่น bgm ไปเรื่อยๆ

2.2 Class main.MainMenu extends StackPane

2.2.1 Field

+ <u>Image MAIN_MENU</u>	ภาพพื้นหลังของเมนูหลัก
--------------------------	------------------------

2.2.2 Constructor

+ MainMenu()	เซตค่าพื้นหลัง และสร้างคลาส PlayButton ใส่เข้าไปใน Pane นี้ พร้อมทั้งใส่เมธอด setOnMouseClicked ให้เรียก playGame() เมื่อกดปุ่ม
--------------	---

2.2.3 Method

+ void playGame()	ทำการซ่อน Pane นี้ และเริ่มเกม
+ void showMainMenu()	แสดง Pane นี้ขึ้นมา หากถูกซ่อนอยู่

2.3 Class main.PlayButton extends ImageView

2.3.1 Field

- <u>Image PLAY</u>	ภาพปุ่ม “PLAY”
---------------------	----------------

2.3.2 Constructor

+ PlayButton()	เซตรูปภาพให้ ImageView นี้ให้เป็น PLAY
----------------	--

2.4 Class main.GameOver extends StackPane

2.4.1 Field

+ <u>int WINDOW_WIDTH</u>	ความกว้างของ Pane (1200 พิกเซล)
+ <u>int WINDOW_HEIGHT</u>	ความสูงของ Pane (700 พิกเซล)
- Canvas canvas	Canvas เอาไว้วาดรูป
- GraphicsContext gc	เครื่องมือวาดรูปของ Canvas
- MainMenu mainMenu	คลาส MainMenu มีไว้สำหรับเรียกใช้เมธอด

2.4.2 Constructor

+ GameOver(MainMenu mainMenu)	กำหนดฟิลด์ mainMenu เรียก GraphicsContext เซตฟอนต์ และทำการวาดรูปหน้าต่าง GameOver พร้อมทั้งใส่เมธอด this.setOnMouseClicked ให้เรียกเมธอด mainMenu.showMainMenu() เมื่อคลิกที่ Pane นี้
-------------------------------	---

2.4.3 Method

+ void showGameOver()	แสดง Pane นี้ขึ้นมา หากถูกซ่อนอยู่
-----------------------	------------------------------------

2.5 Class main.PauseUI extends StackPane

2.5.1 Field

มีทุกอย่างเหมือน 2.4 แต่ไม่มีฟิลด์ mainMenu

2.5.2 Constructor

+ PauseUI()	เหมือน 2.4 แต่วาดหน้าต่าง Paused แทนพร้อมทั้งใส่เมธอด this.setOnMouseClicked ให้เรียกเมธอด GameScene.toggleGamePause() เมื่อคลิกที่ Pane นี้
-------------	---

2.6 Class main.GameSceneUI extends StackPane

2.6.1 Field

+ <u>Image HEALTH_FRAME</u>	ภาพของกรอบหลอด HP /Ammo
+ <u>Image HEALTH_BAR</u>	ภาพแถบของหลอด HP /Ammo
+ <u>Image COIN_UI</u>	รูปเหรียญ
- Canvas canvas	Canvas เอาไว้วาดรูป
- GraphicsContext gc	เครื่องมือวาดรูปของ Canvas

2.6.2 Constructor

+ GameUI()	เหมือน 2.4 แต่วาดหน้าต่าง UI ที่มีปุ่ม Pause แทน พร้อมทั้งใส่เมธอด setOnClick ให้เรียกเมธอด GameScene.toggleGamePause() และแสดง PauseUI ขึ้นมา เมื่อคลิกที่ปุ่ม Pause
------------	--

2.6.3 Method

+ void render()	ใช้ gc วาดรูปหลอดเลือด แถบเลือด / กระสุน และรูปเหรียญ กับจำนวนเหรียญ โดยใช้ข้อมูลผู้เล่นจาก GameScene.getHumanPlayer()
-----------------	--

2.7 Class main.GameScene extends StackPane

2.7.1 Field

- <u>Canvas canvas</u>	Canvas เอาไว้วาดรูป
- GraphicsContext gc	เครื่องมือวาดรูปของ Canvas
- ArrayList<Bullet> bullets	List เก็บกระสุนในแผนที่
- ArrayList<Effect> effects	List เก็บเอฟเฟกต์ในแผนที่
- ArrayList<Spawner> spawners	List เก็บตัวสร้างวัตถุในแผนที่
- <u>ArrayList<Item> items</u>	List เก็บไอเทมในแผนที่
- <u>ArrayList<Enemy> enemies</u>	List เก็บศัตรูในแผนที่
- Map map	คลาส Map สำหรับ render แผนที่
- <u>HumanPlayer humanPlayer</u>	ตัวผู้เล่น
- <u>Coord currentMousePosition</u>	Coord เก็บตำแหน่งปัจจุบันของ cursor
- <u>boolean isPaused</u>	เก็บสถานะการ pause ของเกม เริ่มต้นจะเป็น true
- Set<KeyCode> keyBoardStatus	เซตเก็บสถานะของคีย์บอร์ดว่าผู้เล่นกำลังกดปุ่มไหนอยู่บ้าง
- Random rand	คลาส Random เอาไว้สุ่มเลข
- GameUI healthBar	UI ของเกมทั้งหมด
- GameOver gameOver	คลาส GameOver มีไว้สำหรับเรียกใช้เมธอด

2.7.2 Constructor

+ GameScene(GameOver gameOver)	กำหนดฟิลด์ gameOver เรียกเมธอด gameSetup() สร้างวัตถุใหม่สำหรับฟิลด์ต่างๆ เรียกใช้ GraphicsContext ใส่เมธอด setOnMouseMoved เพื่อเปลี่ยนค่า currentMousePosition เมธอด setOnKeyPressed/Released เพื่อใส่เข้าหรือเอา KeyCode ออกจาก keyboardStatus จากนั้นเรียก startGameLoop()
--------------------------------	--

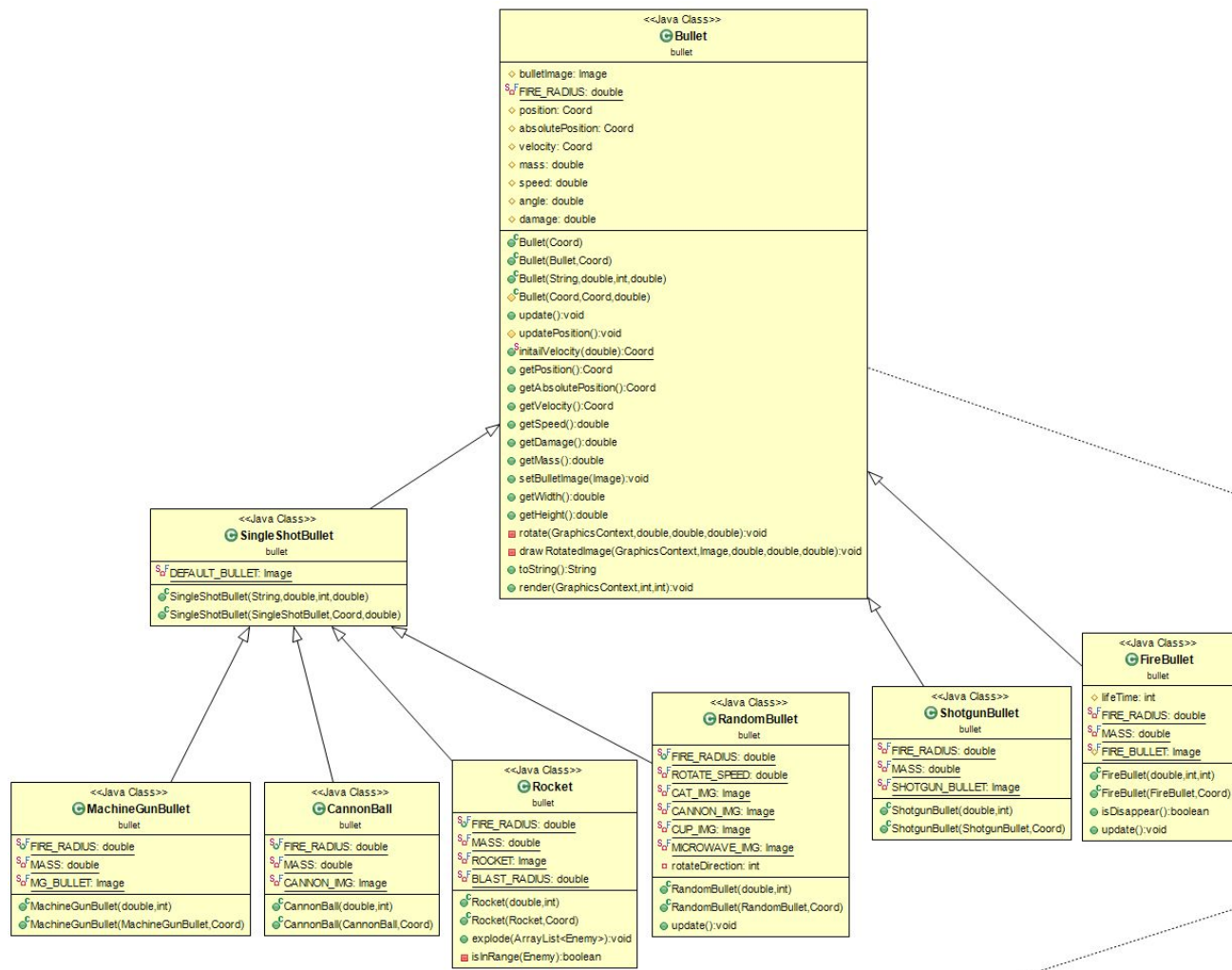
2.7.3 Method

+ void startGameLoop()	สร้าง AnimationTimer ขึ้นมา โดยทุกๆครั้งที่อัปเดต ถ้า isPaused เป็น false ให้เรียกเมธอด update()
+ void gameSetup()	สร้าง ArrayList ใหม่ให้กับฟิลด์ที่เป็น ArrayList ทุกฟิลด์ สร้าง HumanPlayer กับ Map ใหม่ เพิ่ม Spawner ของสโลิม์เข้าไปยัง spawners และให้ isPaused = true
- void keyboardHandle()	ตรวจสอบเงื่อนไขของ keyboardStatus ถ้า <ul style="list-style-type: none"> • มีปุ่ม W ให้เดินขึ้น • มีปุ่ม A ให้เดินไปทางซ้าย • มีปุ่ม S ให้เดินลง • มีปุ่ม D ให้เดินไปทางขวา • มีปุ่ม Space ให้ยิง (handlePlayerShoot) * สำหรับปุ่ม E ให้เช็คเงื่อนไขใน constructor แทนเพราะไม่ต้องการให้กดค้าง
+ void addBullet(Bullet bullet)	เพิ่ม bullet เข้าไปใน List bullets
+ void handlePlayerShoot()	ถ้าผู้เล่นพร้อมที่จะยิง ให้ทำการยิงและเอากระสุนใส่เข้าไปใน bullets
- void updateItems(int startRenderX, int startRenderY)	ตรวจสอบว่าตัวผู้เล่นได้เก็บไอเทมหรือไม่ ถ้าใช่ให้เรียกใช้ไอเทมทันที ถ้าไม่ใช่ให้ update และ render ทุกไอเทมที่อยู่ใน items
- void updateEnemies(int startRenderX, int startRenderY)	ตรวจสอบว่าสโลิม์โดนตัวผู้เล่นหรือไม่ ถ้าใช่ให้เรียก takeDamage() ของผู้เล่นและลบสโลิม์ตัวนั้นทิ้งไป และถ้าตายให้เพิ่มเหรียญไปยังผู้เล่นโดยสุ่ม
- void updateBullets(int startRenderX, int startRenderY)	ถ้ากระสุนโดนตัวสโลิม์ให้เรียก takeDamage ของสโลิม์ และลบกระสุนออกจากลิสต์ ในกรณีที่กระสุนเป็น Rocket ให้เรียก explode ด้วย ถ้าไม่โดนให้ update และ render กระสุนทุกนัดใน bullets ในกรณีที่ เป็น FireBullet ให้เช็คด้วยว่าถึงเวลาที่ต้องหายไปหรือยัง
- void renderEffects(int startRenderX, int startRenderY)	ไล่เช็คทุกเอฟเฟกต์ใน effects ว่าเอฟเฟกต์นั้นถูก render เสร็จหรือยัง ถ้าเสร็จแล้วให้ลบทิ้ง แต่ถ้าไม่ใช่ให้เรียก update และ render
- void updateSpawner()	เรียก update ของ spawner ทุกอันใน

	spawners
+ void update()	เรียก keyboardHandle จากนั้นล้างภาพใน canvas จากนั้นเริ่มวาดโดยการเรียก render ของ map healthBar และเรียก renderEffects renderItem renderEnemies renderBullets humanPlayer.update humanPlayer.render ตามลำดับ จากนั้นตรวจสอบว่าผู้เล่นตายหรือไม่ ถ้าตายให้เรียก gameSetup และแสดงหน้าต่าง gameOver ขึ้นมา
+ void toggleGamePause()	กลับค่า isPaused
+ Coord getMousePosition()	Getter ของ currentMousePosition
+ HumanPlayer getHumanPlayer()	Getter ของ humanPlayer
+ ArrayList<Item> getItemList()	Getter ของ items
+ ArrayList<Enemy> getEnemyList()	Getter ของ enemies

3. Package bullet

ประกอบด้วยคลาสของกระสุนประเภทต่างๆ โดยทุกประเภทจะสืบทอดมาจากคลาส Bullet



ภาพที่ 7 UML Diagram ของส่วน package bullet

3.1 Class bullet.Bullet implements Renderable

3.1.1 Field

# Image bulletImage	รูปภาพของกระสุน เริ่มต้นให้เป็น bulletc.png
# Coord position	พิกัดของกระสุนสัมพันธ์กับหน้าจอ
# Coord absolutePosition	พิกัดของกระสุนเทียบกับตำแหน่งบนแผนที่

# Coord velocity	เวกเตอร์เก็บความเร็วของกระสุน
# double mass	ค่ามวลของกระสุน มีไว้เพื่อคำนวณโมเมนตัมการ knockback อย่างง่ายของสไลม์
# double speed	อัตราเร็วของกระสุน เป็นปริมาณสเกลาร์
# double angle	มุมของกระสุน คิดตามวงกลมหนึ่งหน่วย
# double damage	ความแรงของกระสุน

3.1.2 Constructor

# Bullet(Coord position, Coord velocity, double fireRadius)	กำหนดค่าฟิลด์ต่างๆ และคำนวณค่า absolutePosition จากนั้นทำการคำนวณมุม angle จากค่า velocity
+ Bullet(Coord velocity)	เรียกใช้ constructor ด้านบน โดยให้ position เป็นตำแหน่งกลางจอและ fireRadius = FIRE_RADIUS
+ Bullet(String type, double speed, int damage, double mass)	กำหนดค่าให้ฟิลด์ต่างๆ ส่วน bulletImage ให้เรียกดูรูปภาพจาก bullets/{type}.png
+ Bullet(Bullet bullet, Coord velocity)	Copy constructor ของ bullet แต่มีการแก้ไขฟิลด์ velocity ให้มีค่าเท่ากับพารามิเตอร์ที่ป้อนเข้ามา และ damage ให้มีค่า damage เดิมคูณกับ damageMultiplier ของผู้เล่น

3.1.3 Method

# void updatePosition()	อัปเดตค่า position และ absolutePosition โดยการบวกเวกเตอร์ velocity เข้าไป
+ void update()	เรียก updatePosition()
+ <u>Coord initialVelocity(double speed)</u>	คืนค่า Coord โดยเป็นเวกเตอร์ของผลต่างของพิกัดของเมาส์กับตำแหน่งกลางจอจากนั้นนำไป normalize ด้วยค่า speed จะได้ความเร็วที่แท้จริงของกระสุนนั้นๆ
- void rotate(GraphicsContext gc, double angle, double px, double py)	หมุน gc ด้วยมุม angle และแกนหมุนอยู่ที่ (px, py)
- void	วาดรูป image ที่เอียง angle องศา โดยมี

drawRotatedImage(GraphicsContext gc, Image image, double angle, double tlpX, double tlpY)	จุดศูนย์กลางของภาพอยู่ที่ (tlpx, tlpY) หลักการคือให้หมุน gc ก่อนแล้วค่อยวาด จากนั้นให้หมุน gc กลับที่เดิม
+ void render(GraphicsContext gc, int x, int y) throws Exception	เรียกเมธอด drawRotatedImage(gc, bulletImage, angle, x, y)
+ String toString()	คืนค่าสตริงเป็นพิกัดของ position
setBulletImage(Image bulletImage)	Setter ของ bulletImage
Getter	Getter ของ position absolutePosition velocity speed damage mass width และ height ของ bulletImage

3.2 Class abstract bullet.SingleShotBullet extends Bullet

3.2.1 Field

- <u>double DEFAULT_BULLET</u>	รูปภาพกระสุนเริ่มต้นเป็น bulletc.png
--------------------------------	--------------------------------------

3.2.2 Constructor

+ SingleShotBullet(String name, double speed, double damage, double mass)	เรียก constructor ของคลาสแม่
+ SingleShotBullet(SingleShotBullet bullet, Coord velocity, double fireRadius)	เหมือน Copy constructor ใน 3.1.2 แต่เปลี่ยน bulletImage เป็น DEFAULT_BULLET ด้วย

3.2.3 Method

+ <i>SingleShotBullet duplicate(SingleShotBullet bullet)</i>	Abstract Method คืนค่า SingleShotBullet ที่เหมือนกับพารามิเตอร์
--	---

3.3 Class bullet.CannonBall extends SingleShotBullet

3.3.1 Field

- <u>Image CANNON_IMG</u>	รูปภาพกระสุนปืนใหญ่
---------------------------	---------------------

3.3.2 Constructor

+ CannonBall(double speed, double damage)	เรียก super("cannonball", speed, damage, MASS)
+ CannonBall(CannonBall bullet, Coord velocity)	เหมือน Copy constructor ใน 3.1.2 แต่เปลี่ยน bulletImage เป็น CANNON_IMG ด้วย

3.3.3 Method

+ SingleShotBullet duplicate(SingleShotBullet bullet)	คืนค่า CannonBall ที่เหมือนกับพารามิเตอร์
--	---

3.4 Class bullet.FireBullet extends Bullet

3.4.1 Field

- <u>Image FIRE_BULLET</u>	รูปภาพไฟ
- int lifeTime	เวลาที่อยู่ได้ของกระสุนไฟ ก่อนที่จะหายไป

3.4.2 Constructor

+ FireBullet(double speed, double damage, int lifeTime)	เรียก super("flamethrower_bullet", speed, damage, MASS) และกำหนดฟิลด์ lifeTime
+ FireBullet(FireBullet bullet, Coord velocity)	เหมือน Copy constructor ใน 3.1.2 แต่เปลี่ยน bulletImage เป็น FIRE_BULLET และ กำหนดฟิลด์ lifeTime ให้เหมือนกับ bullet ด้วย

3.4.3 Method

+ boolean isDisappear()	ถ้า lifeTime < 0 ให้คืนค่า true ถ้าไม่ คืน false
+ void update()	เรียก updatePosition() พร้อมทั้งลดค่า lifeTime ลง 1

3.5 Class bullet.RandomBullet extends SingleShotBullet

3.5.1 Field

- <u>double ROTATE_SPEED</u>	อัตราเร็วเชิงมุมของกระสุน มีค่า 5
- <u>Image CAT_IMG</u>	รูปภาพแมว
- <u>Image CANNON_IMG</u>	รูปภาพกระสุนปืนใหญ่
- <u>Image CUP_IMG</u>	รูปภาพแก้วกาแฟ
- <u>Image MICROWAVE_IMG</u>	รูปภาพไมโครเวฟ
- int rotateDirection	ทิศทางการหมุนของกระสุน มีค่า 1 (ตามเข็มนาฬิกา) หรือ -1 (ทวนเข็มนาฬิกา)

3.5.2 Constructor

+ RandomBullet(double speed, double damage)	เรียก super("cat", speed, damage, 1) และกำหนดฟิลด์ rotateDirection ให้เป็น 1
+ RandomBullet(RandomBullet bullet, Coord velocity)	เหมือน Copy constructor ใน 3.1.2 แต่เปลี่ยน bulletImage ไปตามค่าที่ส่งขึ้นมา (0 - 3) โดยมีเงื่อนไขดังนี้ <ul style="list-style-type: none">• ได้ค่า 0 ให้ใช้ CAT_IMG• ได้ค่า 1 ให้ใช้ CANNON_IMG• ได้ค่า 2 ให้ใช้ CUP_IMG• ได้ค่า 3 ให้ใช้ MICROWAVE_IMG ทั้งนี้ให้กำหนด damage และ mass ตามประเภทของกระสุนด้วย จากนั้นให้เปลี่ยนค่า rotateDirection ตามเลขที่ส่งขึ้นมาใหม่ (0 - 1) โดย <ul style="list-style-type: none">• ได้ค่า 0 ให้ใช้ค่า 1• ได้ค่า 1 ให้ใช้ค่า -1

3.5.3 Method

+ void update()	เรียก updatePosition() พร้อมทั้งเพิ่มค่า angle ขึ้นเป็น angle + ROTATE_SPEED * rotateDirection
+ SingleShotBullet duplicate(SingleShotBullet bullet)	คืนค่า RandomBullet ที่เหมือนกับพารามิเตอร์

3.6 Class bullet.Rocket extends SingleShotBullet

3.6.1 Field

- <u>Image ROCKET</u>	รูปภาพกระสุน RPG
- <u>double BLAST_RADIUS</u>	รัศมีการระเบิด มีค่า 100

3.6.2 Constructor

+ Rocket(double speed, double damage)	เรียก super("rocket", speed, damage, MASS)
+ Rocket(Rocket bullet, Coord velocity)	เหมือน Copy constructor ใน 3.1.2 แต่เปลี่ยน bulletImage เป็น ROCKET ด้วย

3.6.3 Method

- boolean isInRange(Enemy e)	ตรวจสอบว่าระยะห่างระหว่างพิกัดของกระสุนกับพิกัดของ e นั้นน้อยกว่า BLAST_RADIUS หรือไม่ ถ้าใช่ให้คืน true ไม่งั้น false
+ void explode(ArrayList<Enemy> enemies)	เช็คสโลม์ทุกตัว ถ้าตัวไหนมีระยะห่างจากกระสุนไม่เกิน BLAST_RADIUS ให้เรียก takeDamage
+ SingleShotBullet duplicate(SingleShotBullet bullet)	คืนค่า Rocket ที่เหมือนกับพารามิเตอร์

3.7 Class bullet.ShotgunBullet extends Bullet

3.7.1 Field

- <u>Image SHOTGUN_BULLET</u>	รูปภาพกระสุน Shotgun
-------------------------------	----------------------

3.7.2 Constructor

+ ShotgunBullet(double speed, double damage)	เรียก super("bulleta", speed, damage, MASS)
+ ShotgunBullet(ShotgunBullet bullet, Coord velocity)	เหมือน Copy constructor ใน 3.1.2 แต่เปลี่ยน bulletImage เป็น SHOTGUN_BULLET ด้วย

3.8 Class bullet.MachineGunBullet extends SingleShotBullet

3.8.1 Field

- <u>Image MG_BULLET</u>	รูปภาพกระสุนปืนกล
--------------------------	-------------------

3.8.2 Constructor

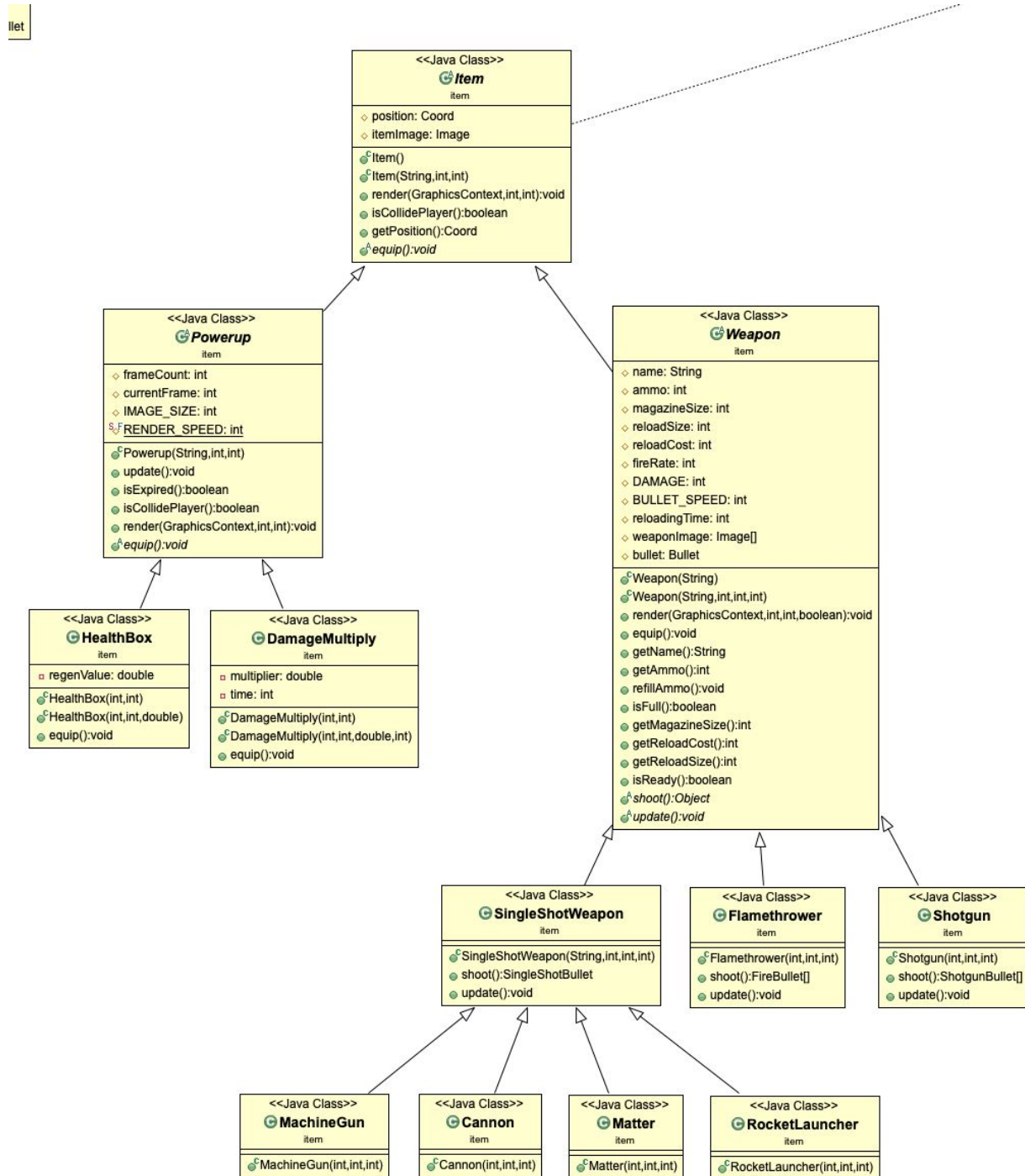
+ MachineGunBullet(double speed, double damage)	เรียก super("cannonball", speed, damage, MASS)
+ MachineGunBullet(MachineGunBullet bullet, Coord velocity)	เหมือน Copy constructor ใน 3.1.2 แต่เปลี่ยน bulletImage เป็น MG_BULLET ด้วย

3.8.3 Method

+ SingleShotBullet duplicate(SingleShotBullet bullet)	คืนค่า MachineGunBullet ที่เหมือนกับพารามิเตอร์
---	---

4. Package item

ประกอบด้วยคลาสของปืนประเภทต่างๆ และคลาสของ Power-Up



ภาพที่ 8 UML Diagram ของส่วน package item

4.1 Class abstract item.Item implements Renderable

4.1.1 Field

# Coord position	พิกัดของไอเทมบนแผนที่
# Image itemImage	รูปภาพของไอเทม

4.1.2 Constructor

+ Item()	เรียก Item("mg", 0, 0)
+ Item(String name, int x, int y)	กำหนดฟิลด์ position และ itemImage โดยถ้า Item เป็น Weapon ให้โหลดภาพจาก weapons/{name}/{name}_side.png ถ้าเป็น PowerUp ให้โหลดจาก powerups/powerup_{name}.png

4.1.3 Method

+ void render(GraphicsContext gc, int x,int y) throws Exception	หากพิกัด (x, y) ไม่ได้อยู่ภายในหน้าจอ ให้ throw new Exception ถ้าอยู่ในจอให้ใช้ gc วาดภาพ itemImage โดยมีจุดศูนย์กลางที่ (x, y)
+ boolean isCollidePlayer()	ตรวจสอบว่าพิกัดตำแหน่งของผู้เล่นอยู่ในกรอบสี่เหลี่ยมของไอเทมนี้หรือไม่ ถ้าใช่ให้คืน true ถ้าไม่คืน false
+ Coord getPosition()	Getter ของ position
+ void equip()	Abstract method ถูกเรียกเมื่อผู้เล่นจะใช้ไอเทมนี้

4.2 Class abstract item.Powerup extends Item

4.2.1 Field

# int frameCount	จำนวนเฟรมที่ถูก render ตั้งแต่สร้างคลาสนี้ขึ้น
# int currentFrame	ภาพปัจจุบัน ใช้ในการ render animation
# int IMAGE_SIZE	ขนาดของรูปไอเทม มีค่า 40
# int RENDER_SPEED	ความเร็วในการเปลี่ยนภาพ มีค่า 7

4.2.2 Constructor

+ Powerup(String name, int x, int y)	เรียก super(name, x, y) และกำหนดฟิลด์ frameCount และ currentFrame เป็น 0
--------------------------------------	--

4.2.3 Method

+ void render(GraphicsContext gc, int x, int y) throws Exception	หากพิกัด (x, y) ไม่ได้อยู่ภายในหน้าจอ ให้ throw new Exception ถ้าอยู่ในจอให้ใช้ gc วาดภาพ itemImage โดยมีจุดศูนย์กลางที่ (x, y) แต่จะเริ่มวาดจากตำแหน่ง (currentFrame * IMAGE_SIZE, 0) เนื่องจากเหตุผลทางการแบ่ง sprite
+ boolean isCollidePlayer()	ตรวจสอบว่าพิกัดตำแหน่งของผู้เล่นอยู่ในกรอบสี่เหลี่ยมของไอเทมนี้หรือไม่ ถ้าใช่ให้คืน true ถ้าไม่คืน false
+ void update()	เพิ่ม frameCount ขึ้น 1 และกำหนดค่า currentFrame = frameCount / RENDER_SPEED % 4
+ boolean isExpired()	ถ้า frameCount > 1200 (20 วินาที) ให้คืนค่าเป็น true ถ้าไม่คืน false
+ void equip()	Abstract method ถูกเรียกเมื่อผู้เล่นจะใช้ไอเทมนี้

4.3 Class item.HealthBox extends Powerup

4.3.1 Field

- double regenValue	ค่า HP ที่เพิ่มเมื่อผู้เล่นใช้งาน
---------------------	-----------------------------------

4.3.2 Constructor

+ HealthBox(int x, int y, double regenValue)	เรียก super("red", x, y) และกำหนดฟิลด์ regenValue
+ HealthBox(int x, int y)	เรียก HealthBox(x, y, 30) นั่นคือค่าเริ่มต้นของ regenValue จะเท่ากับ 30

4.3.3 Method

+ void equip()	เพิ่ม HP ของผู้เล่นขึ้น regenValue หน่วย
----------------	--

4.4 Class item.DamageMultiply extends Powerup

4.4.1 Field

- double multiplier	จำนวนเท่าของ damage ที่เพิ่มขึ้น
- int time	ระยะเวลาที่ส่งผล

4.4.2 Constructor

+ DamageMultiply(int x, int y, double multiplier, int time)	เรียก super("green", x, y) และกำหนดฟิลด์ multiplier และ time
+ DamageMultiply(int x, int y)	เรียก HealthBox(x, y, 1.5, 300) นั่นคือค่าเริ่มต้นของ multiplier คือ 1.5 เท่า เป็นระยะเวลา 300 เฟรม (5 วินาที)

4.4.3 Method

+ void equip()	เซต damageMultiplier ของผู้เล่นเป็น multiplier และกำหนดเวลาเท่ากับ time
----------------	---

4.5 Class abstract item.Weapon extends Item

4.5.1 Field

# String name	ชื่อของอาวุธ
# int ammo	จำนวนกระสุนที่มีอยู่
# int magazineSize	ขนาดของซองกระสุน
# Image[] weaponImage	รูปภาพปืนในมุมต่างๆ 5 มุม ได้แก่ <ul style="list-style-type: none"> • Index 0: down • Index 1: diagdown • Index 2: side • Index 3: diagup • Index 4: up
# Bullet bullet	คลาส Bullet เป็นรูปแบบกระสุนของปืนนี้

4.5.2 Constructor

+ Weapon(String name, int x, int y, int ammo)	กำหนดฟิลด์ต่างๆ ให้ขนาดของกระสุนเท่ากับ ammo และโหลดรูปภาพเข้าอาร์เรย์ของ Image ตามมุมที่กล่าวข้างต้น
+ Weapon(String name)	เรียก Weapon(name, 0, 0, 100)

4.5.3 Method

+ void render(GraphicsContext gc, int facingDirection, int mirrorDirection, boolean renderFirst)	ใช้ gc วาดภาพอาวุธที่ตำแหน่งเหนือกึ่งกลางหน้าจอไป 10 พิกเซล (หาก renderFirst เป็น false จะเป็น 20 พิกเซล) โดยภาพอาวุธจะใช้ภาพจาก weaponImage[facingDirection]
+ void refillAmmo(int amount)	เพิ่มกระสุนเป็นจำนวน amount นัด โดยห้ามเกิน magazineSize
+ boolean isFull()	ตรวจสอบว่ามีกระสุนอยู่เต็มหรือไม่ ถ้าเต็มคืน true ไม่เต็ม false
+ void equip()	เปลี่ยนอาวุธปัจจุบันของผู้เล่นให้เป็นอาวุธนี้
+ <i>Object shoot()</i>	Abstract method สำหรับเรียกยิง
+ <i>boolean isReady()</i>	Abstract method สำหรับตรวจสอบว่าอาวุธพร้อมยิงหรือไม่
+ <i>void update()</i>	Abstract method สำหรับอัปเดตอาวุธ

4.6 Class item.SingleShotWeapon extends Weapon

4.6.1 Field

- <u>int DAMAGE</u>	ความแรงของปืน มีค่าแตกต่างกันไปตามประเภทอาวุธ มีค่า 10
- <u>int BULLET SPEED</u>	ความเร็วของกระสุน มีค่าแตกต่างกันไปตามประเภทอาวุธ มีค่า 10
- <u>double MASS</u>	มวลของกระสุน มีค่าแตกต่างกันไปตามประเภทอาวุธ มีค่า 0.1
# int fireRate	เวลาการรีโหลดกระสุน มีค่าเริ่มต้นเป็น 10

# int reloadingTime	ตัวจับเวลาการรีโหลด มีค่าเป็น 0
---------------------	---------------------------------

4.6.2 Constructor

+ SingleShotWeapon(String name, int x, int y, int ammo)	เรียก super(name, x, y, ammo) และให้ bullet เป็น Bullet("bulletc", BULLET_SPEED, DAMAGE, MASS)
+ SingleShotWeapon(String name)	เรียก super(name) และให้ bullet เป็น Bullet("bulletc", BULLET_SPEED, DAMAGE, MASS)

4.6.3 Method

+ SingleShotBullet shoot()	ถ้าอาวุธพร้อมยิง ให้ลด ammo ลง 1 ปรับ reloadingTime = 0 และคืนค่า SingleShotBullet(bullet.initialVelocity(bullet.getSpeed())) ถ้าไม่พร้อมคืน null
+ boolean isReady()	ถ้ามีกระสุน และ reloadingTime > fireRate ให้คืนค่า true ถ้าไม่คืน false
+ void update()	เพิ่ม reloadingTime ขึ้น 1

4.7 Class item.Cannon extends SingleShotWeapon

4.7.1 Field

- <u>int DAMAGE</u>	ความแรงของปืน มีค่า 50
- <u>int BULLET_SPEED</u>	ความเร็วของกระสุน มีค่า 5

4.7.2 Constructor

+ Cannon(int x, int y, int ammo)	เรียก super("cannon", x, y, ammo) และให้ bullet เป็น CannonBall(BULLET_SPEED, DAMAGE) และปรับ fireRate = 30
+ Cannon()	เรียก super("cannon") และให้ bullet เป็น CannonBall(BULLET_SPEED, DAMAGE) และปรับ fireRate = 30

4.8 Class item.Matter extends SingleShotWeapon

4.8.1 Field

- <u>int DAMAGE</u>	ความแรงของปืน มีค่า 0 เพราะความแรงจะขึ้นอยู่กับกระสุนที่ปล่อยออกมา
- <u>int BULLET_SPEED</u>	ความเร็วของกระสุน มีค่า 5

4.8.2 Constructor

+ Matter(int x, int y, int ammo)	เรียก super("matter", x, y, ammo) และให้ bullet เป็น RandomBullet(BULLET_SPEED, DAMAGE) และปรับ fireRate = 30
+ Matter()	เรียก super("matter") และให้ bullet เป็น RandomBullet(BULLET_SPEED, DAMAGE) และปรับ fireRate = 30

4.9 Class item.RocketLauncher extends SingleShotWeapon

4.9.1 Field

- <u>int DAMAGE</u>	ความแรงของปืน มีค่า 80
- <u>int BULLET_SPEED</u>	ความเร็วของกระสุน มีค่า 20

4.9.2 Constructor

+ RocketLauncher(int x, int y, int ammo)	เรียก super("rocket", x, y, ammo) และให้ bullet เป็น Rocket(BULLET_SPEED, DAMAGE) และปรับ fireRate = 50
+ RocketLauncher()	เรียก super("rocket") และให้ bullet เป็น Rocket(BULLET_SPEED, DAMAGE) และปรับ fireRate = 50

4.10 Class item.Flamethrower extends Weapon

4.10.1 Field

- <u>int DAMAGE</u>	ความแรงของปืน มีค่า 15
- <u>int BULLET_SPEED</u>	ความเร็วของกระสุน มีค่า 3
- <u>int SHOT_COUNT</u>	จำนวนกระสุนที่กระจายออกมาต่อการยิงหนึ่งครั้ง มีค่า 3
- <u>int ACCURACY_ERROR</u>	ความผิดพลาดของการกระจาย เทียบกับมุมที่ผู้เล่นยิง มีค่า $\frac{\pi}{6}$
- <u>int DEFAULT_LIFETIME</u>	เวลาที่กระสุนไฟอยู่ได้ ก่อนที่จะดับ
# int fireRate	เหมือนกับ SingleShotWeapon
# int reloadingTime	เหมือนกับ SingleShotWeapon

4.10.2 Constructor

+ Flamethrower(int x, int y, int ammo)	เรียก super("flamethrower", x, y, ammo) และให้ bullet เป็น FireBullet(BULLET_SPEED, DAMAGE, DEFAULT_LIFETIME) และปรับ fireRate = 10
+ Flamethrower()	เรียก super("flamethrower") และให้ bullet เป็น FireBullet(BULLET_SPEED, DAMAGE, DEFAULT_LIFETIME) และปรับ fireRate = 10

4.10.3 Method

+ FireBullet[] shoot()	ถ้าอาวุธพร้อมยิง ให้ลด ammo ลง SHOT_COUNT นัด ปรับ reloadingTime = 0 จากนั้นทำการคำนวณมุมที่ผู้เล่นยิงแล้ว \pm ค่า โดยสุ่มในช่วง [-ACCURACY_ERROR, ACCURACY_ERROR] แล้วสร้าง Bullet ใหม่ ด้วย velocity ที่เกิดจากมุมใหม่ใส่ไว้ในอาเรย์ ทำเช่นนี้เป็นจำนวน SHOT_COUNT รอบ เมื่อทำครบให้คืนค่าอาเรย์ดังกล่าว ถ้าอาวุธไม่พร้อมยิงให้คืน null
+ boolean isReady()	ถ้ามีกระสุน และ reloadingTime > fireRate ให้คืนค่า true ถ้าไม่คืน false

+ void update()	เพิ่ม reloadingTime ขึ้น 1
-----------------	----------------------------

4.11 Class item.Shotgun extends Weapon

4.11.1 Field

- <u>int DAMAGE</u>	ความแรงของปืน มีค่า 10
- <u>int BULLET_SPEED</u>	ความเร็วของกระสุน มีค่า 13
- <u>int SHOT_COUNT</u>	จำนวนกระสุนที่กระจายออกมาต่อการยิงหนึ่งครั้ง มีค่า 10
- <u>int ACCURACY_ERROR</u>	ความผิดพลาดของการกระจาย เทียบกับมุมที่ผู้เล่นยิง มีค่า $\frac{\pi}{12}$
# int fireRate	เหมือนกับ SingleShotWeapon
# int reloadingTime	เหมือนกับ SingleShotWeapon

4.11.2 Constructor

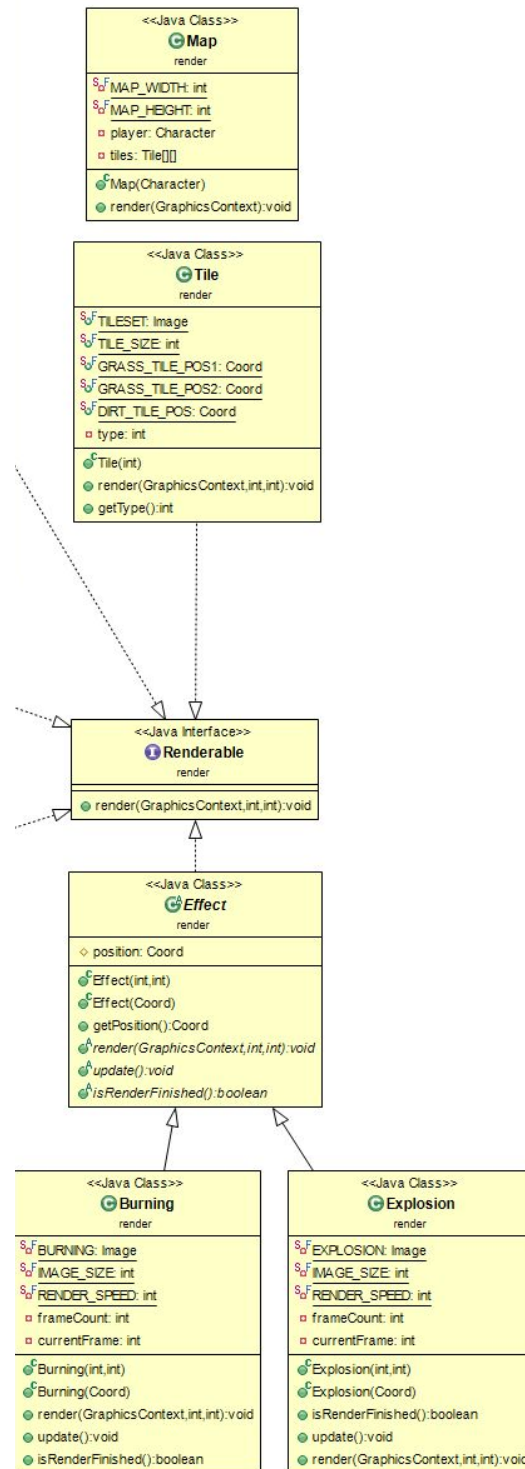
+ Shotgun(int x, int y, int ammo)	เรียก super("shotgun", x, y, ammo) และให้ bullet เป็น ShotgunBullet(BULLET_SPEED, DAMAGE) และปรับ fireRate = 50
+ Shotgun()	เรียก super("shotgun") และให้ bullet เป็น ShotgunBullet(BULLET_SPEED, DAMAGE) และปรับ fireRate = 50

4.11.3 Method

+ ShotgunBullet[] shoot()	เหมือนกัน 4.10 โดยเปลี่ยนนิดาเรย์เป็น ShotgunBullet[] แทน
+ boolean isReady()	ถ้ามีกระสุน และ reloadingTime > fireRate ให้คืนค่า true ถ้าไม่คืน false
+ void update()	เพิ่ม reloadingTime ขึ้น 1

5. Package render

ประกอบด้วยคลาสที่เกี่ยวกับแผนที่ เอฟเฟกต์ต่างๆ รวมถึง interface Renderable



ภาพที่ 9 UML Diagram ของส่วน package render

5.1 Class render.Tile implements Renderable

5.1.1 Field

+ <u>Image TILESET</u>	รูป tileset โหลดจาก res/tileset.png
+ <u>int TILE_SIZE</u>	ขนาด tile แต่ละอัน มีค่า 12 พิกเซล
+ <u>Coord GRASS TILE_POS1</u>	ตำแหน่งการวาด Tile หญ้า (แบบเขียวล้วน)
+ <u>Coord GRASS TILE_POS2</u>	ตำแหน่งการวาด Tile หญ้า (แบบมีหญ้าด้วย)
+ <u>Coord DIRT TILE</u>	ตำแหน่งการวาด Tile ดิน
- int type	ประเภทของ tile โดยมีค่า 0 - 2 <ul style="list-style-type: none">• 0 เป็นแบบเขียวล้วน• 1 เป็นแบบมีหญ้า• 2 เป็นดิน

5.1.2 Constructor

+ Tile(int type)	กำหนดฟิลด์ type
------------------	-----------------

5.1.3 Method

+ void render(GraphicsContext gc, int x, int y) throws Exception	Render ภาพ tile นี้ ที่พิกัด (x, y) โดยตำแหน่งที่เริ่มวาดบน tileset จะขึ้นอยู่กับค่าคงที่ข้างต้น
+ int getType()	Getter ของ type

5.2 Class render.Map

5.2.1 Field

+ <u>int MAP_WIDTH</u>	จำนวน tile ในแนวแกน X
+ <u>int MAP_HEIGHT</u>	จำนวน tile ในแนวแกน Y
- HumanPlayer player	ตัวผู้เล่น
- Tile[][] tiles	เมทริกซ์ของ tile บนแผนที่

5.2.2 Constructor

+ Map(HumanPlayer player)	กำหนดฟิลด์ player และ tiles จากนั้นใช้ Scanner อ่านจำนวนเต็มในไฟล์ res/map_ ที่ละตัวแล้วสร้าง Tile ใหม่ตามประเภทของเลขนั้นๆ
---------------------------	---

5.2.3 Method

+ void render(GraphicsContext gc)	Render tile ทุกอันใน tiles
-----------------------------------	----------------------------

5.3 Class abstract render.Effect implements Renderable

5.3.1 Field

# Coord position	พิกัดของเอฟเฟกต์บนแผนที่
------------------	--------------------------

5.3.2 Constructor

+ Effect(int x, int y)	กำหนดฟิลด์ Coord ด้วยค่า Coord(x, y)
+ Effect(Coord position)	กำหนดฟิลด์ Coord ด้วย position

5.3.3 Method

+ void Coord getPosition()	Getter ของ position
+ void render(GraphicsContext gc, int x, int y) throws Exception	Abstract method ใช้ render effect นี้
+ void update()	Abstract method ใช้ update เฟรมของเอฟเฟกต์
+ boolean isRenderFinished()	Abstract method ใช้ตรวจสอบว่าเอฟเฟกต์นี้ถูก render จบแล้วหรือยัง

5.4 Class render.Burning extends Effect

5.4.1 Field

- <u>Image BURNING</u>	รูปภาพไฟไหม้
- <u>int IMAGE_SIZE</u>	ขนาดของเอฟเฟกต์ มีค่า 40

- <u>int RENDER_SPEED</u>	จำนวนครั้งการอัปเดตก่อนที่จะเปลี่ยนเฟรม มีค่า 5
- int frameCount	จำนวนเฟรมที่อัปเดตหลังจากถูกสร้าง เริ่มที่ 0
- int currentFrame	ตำแหน่งเฟรมปัจจุบัน เริ่มที่ 0

5.4.2 Constructor

+ Burning(int x, int y)	กำหนดฟิลด์ Coord ด้วยค่า Coord(x, y)
+ Burning(Coord position)	กำหนดฟิลด์ Coord ด้วย position

5.4.3 Method

+ void render(GraphicsContext gc, int x, int y) throws Exception	ถ้าตำแหน่งที่จะ render ไม่อยู่ในหน้าต่างให้ throw Exception แต่ถ้าอยู่ในหน้าต่างให้ใช้ gc วาดเอฟเฟกต์นั้น โดยเริ่มวาดตั้งแต่ (currentFrame % 4) * IMAGE_SIZE เนื่องจากเหตุผลทางการแบ่ง sprite
+ void update()	เพิ่มค่า frameCount และ ให้ currentFrame = frameCount / RENDER_SPEED
+ boolean isRenderFinished()	คืนค่า true เมื่อ currentFrame > 20 ถ้าไม่ใช่ให้คืน false

5.5 Class render.Exlosion extends Effect

5.5.1 Field

- <u>Image EXPLOSION</u>	รูปภาพเอฟเฟกต์ระเบิด
- <u>int IMAGE_SIZE</u>	ขนาดของเอฟเฟกต์ มีค่า 64
- <u>int RENDER_SPEED</u>	จำนวนครั้งการอัปเดตก่อนที่จะเปลี่ยนเฟรม มีค่า 5
- int frameCount	จำนวนเฟรมที่อัปเดตหลังจากถูกสร้าง เริ่มที่ 0
- int currentFrame	ตำแหน่งเฟรมปัจจุบัน เริ่มที่ 0

5.5.2 Constructor

+ Explosion(int x, int y)	กำหนดฟิลด์ Coord ด้วยค่า Coord(x, y)
+ Explosion(Coord position)	กำหนดฟิลด์ Coord ด้วย position

5.5.3 Method

+ void render(GraphicsContext gc, int x, int y) throws Exception	เหมือนกับ 5.4
+ void update()	เหมือนกับ 5.4
+ boolean isRenderFinished()	คืนค่า true เมื่อ currentFrame > 16 ถ้าไม่ใช่ให้คืน false

5.6 Interface render.Renderable

+ void render(GraphicsContext gc, int x, int y) throws Exception	Method สำหรับใช้ gc วาดภาพใดๆ บนพิกัด (x, y) เทียบกับหน้าต่าง หาก (x, y) นั้นไม่อยู่ในหน้าต่างควร throw Exception ใหม่
--	--

6. Package character

ประกอบด้วยคลาสตัวละคร นั่นคือ ตัวผู้เล่นและสไลม์ นอกจากนี้ยังมีคลาส spawner รวมอยู่ใน package นี้ด้วย

6.1 Class character.Player

6.1.1 Field

- Image[] characterImage	อาร์เรย์เก็บภาพของตัวละครแต่ละมุมโดยมี 5 มุมได้แก่ <ul style="list-style-type: none">• Index 0: south• Index 1: diagdown• Index 2: side• Index 3: diagup• Index 4: north
- <u>int CHARACTER_WIDTH</u>	ความกว้างของรูปตัวละคร มีค่า 50 พิกเซล
- <u>int CHARACTER_HEIGHT</u>	ความสูงของรูปตัวละคร มีค่า 60 พิกเซล
- <u>int MAX_MOVE_SPEED</u>	ความเร็วของ animation สูงสุด มีค่า 10

- int facingDirection	ทิศทางการหันหน้าของตัวละคร
- int mirrorDirection	ค่าการสะท้อนรูปภาพ มีค่า 1 (ไม่สะท้อน) หรือ -1 (สะท้อนในแกน Y)
- double health	HP ของผู้เล่น (100)
- double maxHealth	HP สูงสุดของผู้เล่น (100)
- boolean isDead	สถานะของผู้เล่น บอกว่าตายหรือยัง (false)
- int isMoving	สถานะของการขยับขาของตัวละคร ถ้ามีค่ามากกว่า 0 หมายความว่าตัวละครกำลังขยับอยู่ (0)
- int movingSpeed	ความเร็วการก้าวขาของตัวละคร ค่านี้จะมีค่าไม่เกิน MAX_MOVE_SPEED (5)
- double speed	อัตราเร็วของการเดินของตัวละคร (1.5)
- Coord position	พิกัด (x, y) ของผู้เล่นบนแผนที่ (300, 300)
- double healthRegen	ค่า regenerate HP ของผู้เล่น (0.005)
- double damageMultiplier	จำนวนเท่าของ damage ปกติของตัวละคร โดยปกติจะมีค่า 1 ยกเว้นเมื่อตัวละครกำลังใช้ Damage Multiplier ค่านี้จะเป็น 1.5 (1)
- int coinCount	จำนวนเหรียญที่ผู้เล่นมี (0)

6.1.2 Constructor

+ HumanPlayer(String name, int type)	กำหนดฟิลด์ต่างๆ
+ HumanPlayer()	HumanPlayer จะชื่อ No Name

6.1.3 Method

+ void regenHealth(int value)	เพิ่มค่า HP ให้ผู้เล่นจำนวน value หน่วย โดย HP ห้ามเกินค่า maxHealth
- void changeFacingDirection(Coord currentMousePosition)	ทำการเปลี่ยนค่า facingDirection (fd) และ mirrorDirection (md) ตามช่วงของมุมระหว่าง currentMousePosition กับกึ่งกลางหน้าต่าง ดังนี้

	<ul style="list-style-type: none"> ช่วง $[0, \frac{\pi}{8})$ fd = 2, md = 1 ช่วง $[\frac{\pi}{8}, \frac{3\pi}{8})$ fd = 3, md = 1 ช่วง $[\frac{3\pi}{8}, \frac{5\pi}{8})$ fd = 4, md = 1 ช่วง $[\frac{5\pi}{8}, \frac{7\pi}{8})$ fd = 3, md = -1 ช่วง $[\frac{7\pi}{8}, \frac{9\pi}{8})$ fd = 2, md = -1 ช่วง $[\frac{9\pi}{8}, \frac{11\pi}{8})$ fd = 1, md = -1 ช่วง $[\frac{11\pi}{8}, \frac{13\pi}{8})$ fd = 0, md = 1 ช่วง $[\frac{13\pi}{8}, \frac{15\pi}{8})$ fd = 1, md = 1 ช่วง $[\frac{15\pi}{8}, 2\pi)$ fd = 2, md = 1
+ void render(GraphicsContext gc)	ใช้ gc วาดภาพตัวละครที่กึ่งกลางหน้าต่าง โดย วาดตาม facingDirection และ mirrorDirection พร้อมทั้งวาดอาวุธของตัว ละครด้วย
+ void takeDamage(double damage)	ลด HP ของตัวละครเป็นจำนวน damage หน่วย และเช็คค่า HP เหลือ 0 หรือไม่ ถ้าใช่ให้รับ isDead = true
+ void move()	เพิ่มค่า isMoving ขึ้น 1
+ void moveLeft()	ลดค่า position แกน X ด้วยค่า speed
+ void moveRight()	เพิ่มค่า position แกน X ด้วยค่า speed
+ void moveUp()	ลดค่า position แกน Y ด้วยค่า speed
+ void moveDown()	เพิ่มค่า position แกน Y ด้วยค่า speed
+ void idle()	ให้ isMoving = 0
+ void update()	เรียกเมธอด changeFacingDirection, regenHealth, updateDamageMultiplier, updateCoinAnimation และ weapon.update ตามลำดับ
+ void addCoin(int amount)	เพิ่มเหรียญให้ผู้เล่น amount เหรียญ
+ boolean useCoin(int amount)	ถ้าจำนวนเหรียญไม่พอให้คืนค่า false ถ้าพอให้ ลดเหรียญผู้เล่น amount เหรียญแล้วคืน true
Getter & Setter ของทุกฟิลด์	

6.2 Class character.ShootablePlayer extend Player

6.2.1 Field

- Map<String, Weapon> weapon	Map ชื่อ และ object อาวุธของผู้เล่น
- String currentWeaponName	ชื่ออาวุธปัจจุบันของผู้เล่น

6.2.2 Constructor

+ HumanPlayer(int type)	กำหนดฟิลด์ต่างๆ
-------------------------	-----------------

6.2.3 Method

+ void update()	เรียกเมธอด changeFacingDirection, regenHealth, updateDamageMultiplier, updateCoinAnimation และ weapon.update ตามลำดับ
+ boolean buyAmmo()	ถ้าอาวุธมีกระสุนไม่เต็มและสามารถซื้อกระสุนตามประเภทอาวุธที่ถืออยู่ได้ ให้ใช้เหรียญและคืนค่า true ถ้าไม่ได้คืน false
+ void toggleWeapon()	สลับอาวุธของผู้เล่น
+ Set<String> getWeaponNameList()	ชื่ออาวุธทั้งหมดของผู้เล่น
Getter & Setter ของทุกฟิลด์	

6.3 Class character.HumanPlayer

6.3.1 Field

- double damageMultiplier	จำนวนเท่าของ damage ปกติของตัวละคร โดยปกติจะมีค่า 1 ยกเว้นเมื่อตัวละครกำลังใช้ Damage Multiplier ค่านี้จะเป็น 1.5 (1)
- int damageMultiplierTime	ระยะเวลาที่เหลืออยู่ในการใช้ไอเทม Damage Multiplier (0)
- double animatedCoinCount	ค่าเหรียญที่แสดงบน GUI ค่านี้จะปรับเข้าสู่ coinCount เสมอ (0)

6.3.2 Constructor

+ HumanPlayer()	กำหนดฟิลด์ต่างๆ
-----------------	-----------------

6.3.3 Method

- void updateDamageMultiplier()	ลดค่า damageMultiplierTime ลง 1 หาก damageMultiplierTime มีค่าเท่ากับ 0 ให้เซตค่า damageMultiplier = 1
+ boolean inUseDamageMultiplier()	ตรวจสอบว่า damageMultiplierTime > 0 หรือไม่ ถ้าใช่ให้คืน true ถ้าไม่คืน false
+ void render(GraphicsContext gc)	ใช้ gc วาดภาพตัวละครที่กึ่งกลางหน้าต่าง โดยวาดตาม facingDirection และ mirrorDirection พร้อมทั้งวาดอาวุธของตัวละครด้วย
+ void update()	เรียกเมธอด changeFacingDirection, regenHealth, updateDamageMultiplier, updateCoinAnimation และ weapon.update ตามลำดับ
+ void updateCoinAnimation()	ปรับค่า animateCoinCount ให้เข้าใกล้ coinCount โดยสามารถเพิ่ม/ลดได้ 0.2 หน่วยต่อการเรียกหนึ่งครั้ง
Getter & Setter ของทุกฟิลด์	

6.4 Class character.Enemy implements Renderable

6.4.1 Field

- <u>int MONSTER_SIZE</u>	ขนาดของสไลม์ มีค่า 40 พิกเซล
- <u>int MAX_MOVE_SPEED</u>	เหมือน 6.1
- int isMoving	เหมือน 6.1
- int movingSpeed	เหมือน 6.1 (3)
- Coord position	เหมือน 6.1

- Image[] slimeImage	รูปของสไลม์ในมุมมองต่างๆ 3 มุม ได้แก่ <ul style="list-style-type: none"> • Index 0: front • Index 1: side • Index 2: back
- Image healthBar	หลอด HP บนหัวสไลม์
- double speed	เหมือน 6.1 (1.1)
- double damage	ความแรงของการโจมตีของสไลม์ (20)
- double health	เหมือน 6.1
- double maxHealth	เหมือน 6.1
- boolean isDead	เหมือน 6.1
- Coord knockBackVelocity	เวกเตอร์ความเร็วการ knockback โดยถ้าไม่ได้ อยู่ในสถานะ knockback ค่านี้จะเป็น (0, 0)
- double mass	มวลของสไลม์ใช้ในการคำนวณโมเมนตัม knockback (1)

6.4.2 Constructor

+ Enemy(String name, int x, int y)	กำหนดฟิลด์ต่างๆ และกำหนด position เป็น Coord(x, y)
+ Enemy(int x, int y)	Enemy ชื่อ slime1

6.4.3 Method

+ void takeDamage(double damage)	เหมือน 6.1
+ void takeKnockBack(Coord bulletVelocity, double bulletMass)	ให้เวกเตอร์ knockBackVelocity บวกเพิ่ม ด้วยเวกเตอร์ ของ bulletVelocity คูณด้วย สเกลาร์ bulletMass / this.mass
+ void reduceKnockBackVelocity()	ลดขนาดของเวกเตอร์ knockBackVelocity ลง 1 แต่ถ้า knockBackVelocity เป็นเวก เตอร์ศูนย์อยู่แล้วก็ไม่ต้องทำอะไร
+ boolean isCollideBullet(Bullet bullet)	ตรวจสอบว่าตำแหน่งของกระสุน bullet นั้นทับ กับกรอบสี่เหลี่ยมของสไลม์หรือไม่ ถ้าใช่ให้คืนค่า true ถ้าไม่คืน false

+ boolean isCollidePlayer()	ตรวจสอบว่าตำแหน่งของผู้เล่น นั้นทับกับกรอบสี่เหลี่ยมของสโลมหรือไม่ ถ้าใช่ให้คืนค่า true ถ้าไม่คืน false
+ void update()	เพิ่มค่า isMoving ขึ้น 1 เรียกเมธอด reduceKnockBackVelocity จากนั้นให้เปลี่ยนตำแหน่งสโลมด้วยทิศทางที่เข้าหาผู้เล่น
+ void render(GraphicsContext gc, int x, int y) throws Exception	ถ้าตำแหน่งที่จะ render ไม่อยู่ในหน้าต่างให้ throw Exception แต่ถ้าอยู่ในหน้าต่างให้ใช้ gc วาดรูปสโลมและหลอด HP
Getter & Setter ของทุกฟิลด์	

6.5 Class character.Spawner

6.5.1 Field

- <u>int POWERUP DROP RATE</u>	โอกาสครอบของ power up มีค่า 20
- <u>int MAX WEAPON IN MAP</u>	จำนวนปืนที่ครอบบนแผนที่ที่มากที่สุด
- int delay	ระยะเวลาในการสุ่ม drop
- int time	เวลาปัจจุบัน เอาไว้เทียบกับ delay ว่าถึงเวลาสุ่มครอบหรือยัง
- String type	ประเภทวัตถุที่ spawner ทำการปล่อย <ul style="list-style-type: none"> • “Enemy” ปล่อยสโลม • “Item” ปล่อย powerup
- Coord spawnTopLeft	ขอบเขตของตำแหน่งในการสุ่มที่จุดบนซ้ายของสี่เหลี่ยม
- Coord spawnBottomRight	ขอบเขตของตำแหน่งในการสุ่มที่จุดล่างขวาของสี่เหลี่ยม
- Coord spawnPosition	พิกัดในการสุ่มครอบ
- boolean randomSpawn	สถานะบอกว่า spawner นี้สุ่มวัตถุแบบสุ่มพิกัดใหม่ทุกครั้ง (true) หรือเป็นแบบพิกัดเดิมตลอด (false)

6.5.2 Constructor

+ Spawner(String type, int delay, Coord position)	กำหนดฟิลด์ต่างๆ และให้ randomSpawn = false
+ Spawner(String type, int delay, Coord topLeft, Coord bottomRight)	กำหนดฟิลด์ต่างๆ และให้ randomSpawn = true

6.5.3 Method

+ Powerup dropPowerup()	สุ่มดรอป powerup ทั้ง 2 ประเภท ประเภทละ 10 เปอร์เซ็นต์ และคืนค่า powerup นั้น ถ้าสุ่มแล้วไม่ได้ให้คืน null
+ Weapon dropWeapon()	สุ่มดรอปอาวุธโดยห้ามให้จำนวนอาวุธบนแผนที่ไม่เกิน MAX_WEAPON_IN_MAP
- int countWeaponInMap()	คืนค่าจำนวนอาวุธที่วางอยู่ในแผนที่
+ void spawn()	ถ้า spawner นี้ใช้การสุ่มพิกัดทุกครั้ง ให้สุ่มพิกัด (x, y) ขึ้นมาโดยพิกัดนี้จะต้องอยู่ในกรอบสี่เหลี่ยมของ spawnTopLeft และ spawnBottomRight จากนั้นเรียก method drop ตามประเภทของ spawner
+ void update()	เพิ่มค่า time ขึ้น 1 ถ้า time > delay แล้วให้เรียกเมธอด spawn จากนั้นเซตค่า time ให้เป็น 0