PROJECT PYGAME

5. PUNBALL

<u>กลุ่ม วีชิมิชิมิ</u>

โปรเจคนี้ถูกพัฒนาขึ้นด้วย Pygame โดยใช้เทคนิคการเขียนโปรแกรมเชิงวัตถุ (OOP) ซึ่งช่วยให้โครงสร้างโค้ดมีความเป็นระบบและจัดการง่าย เราได้นำแนวคิดนี้มาใช้ในการสร้าง Class สำหรับจัดการองค์ประกอบต่าง ๆ ของเกม


Shooting Star: INTERGALACTIC FRONTIER
tutorial
15
10
4
RESTART
x2

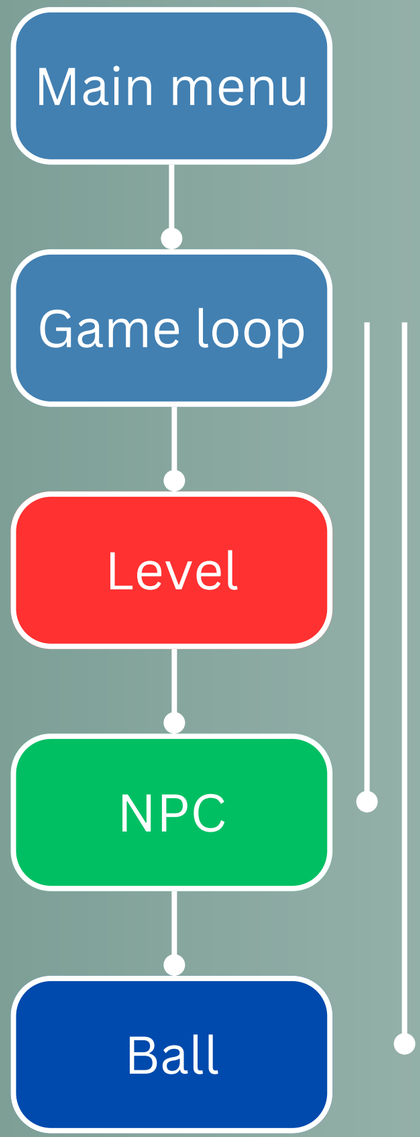
PUNBALL

GAME RULES

1. เริ่มจากผู้เล่นคลิ๊กเมาส์เพื่อยิงลูกบอลไปยังจุดต่างๆ
2. ผู้เล่นจะต้องคำนวณกะระยะและวิถีการเด้งของลูกบอลในมุมต่าง
3. ผู้เล่นต้องสร้างความเสียหายแก่ศัตรูที่อยู่ในแผนที่หรือด่านนั้นๆ จากลูกบอลที่โดนตัวศัตรู
4. ผู้เล่นสามารถเพิ่มจำนวนลูกบอลที่เด้งอยู่ในแผนที่ได้โดยการทำให้ลูกบอลเด้งผ่านจุดที่กำหนด
5. ผู้เล่นจะชนะเกมนี่ได้ก็ต่อเมื่อผู้เล่นสามารถผ่านด่านแต่ละด่านได้จนมาถึงด่านสุดท้ายโดยในแต่ละด่านนั้นผู้เล่นจะต้องจำกัดศัตรูภายในด่านให้หมดโดยการใช้ลูกบอล

PUN BALL

PROGRAM DESIGNS



Punball.py

```python
import pygame
import sys
import random
import math

# Global constants
BLACK = (0, 0, 0)
WHITE = (255, 255, 255)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLUE = (0, 0, 255)



class MainMenu:
    def __init__(self):
        pygame.init()
        self.WIDTH, self.HEIGHT = 1280, 720
        self.screen = pygame.display.set_mode((self.WIDTH, self.HEIGHT))
        pygame.display.set_caption("Shooting Star: INTERGALACTIC FRONTIER")
        self.clock = pygame.time.Clock()
        self.FPS = 60

        # Load assets
        self.background = pygame.image.load(r"background.png")
        self.background = pygame.transform.scale(self.background, (3840, 2160))
        self.background_width, self.background_height = self.background.get_size()

        self.logo = pygame.image.load("logo.png")

        # Buttons
        self.play_button_1 = pygame.image.load("play1.png")
        self.play_button_1 = pygame.transform.scale(self.play_button_1, (300, 120))
        self.play_button_2 = pygame.image.load("play2.png")
        self.play_button_2 = pygame.transform.scale(self.play_button_2, (300, 120))
        self.play_button_1_rect = self.play_button_1.get_rect()

```

PUN BALL

PROGRAM DESIGNS

Main menu

Game loop

Level

NPC

Ball

STEP-1

MAIN MENU

เป็นคลาสแรกที่ถูกเรียกใช้งานเมื่อเริ่มต้นโปรแกรม โดยมีหน้าที่แสดงผล เมนูหลัก ของเกม ซึ่งประกอบด้วยการตั้งค่าและจัดการปุ่มต่าง ๆ เช่น

- ปุ่มเปิด-ปิดเสียง เพื่อควบคุมเสียงประกอบในเกม
- ปุ่มเริ่มเกม ซึ่งเมื่อกดแล้วจะเรียกใช้งานคลาสที่รับผิดชอบในการเริ่มต้นเกม
- ปุ่มออกจากเกม

PUN BALL

PROGRAM DESIGNS

Main menu

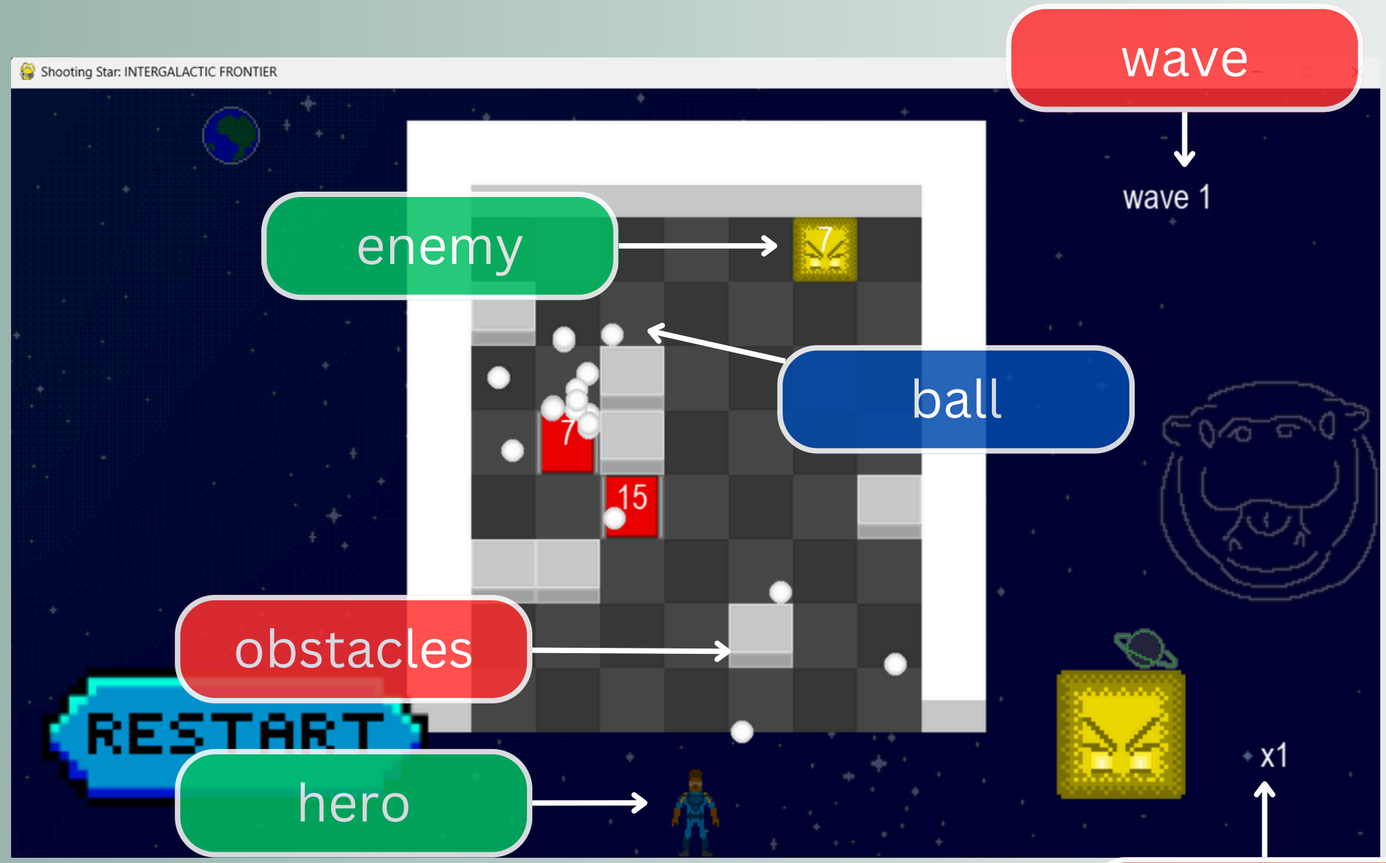
Game loop

Level

NPC

Ball

STEP-2



GAME CLASS

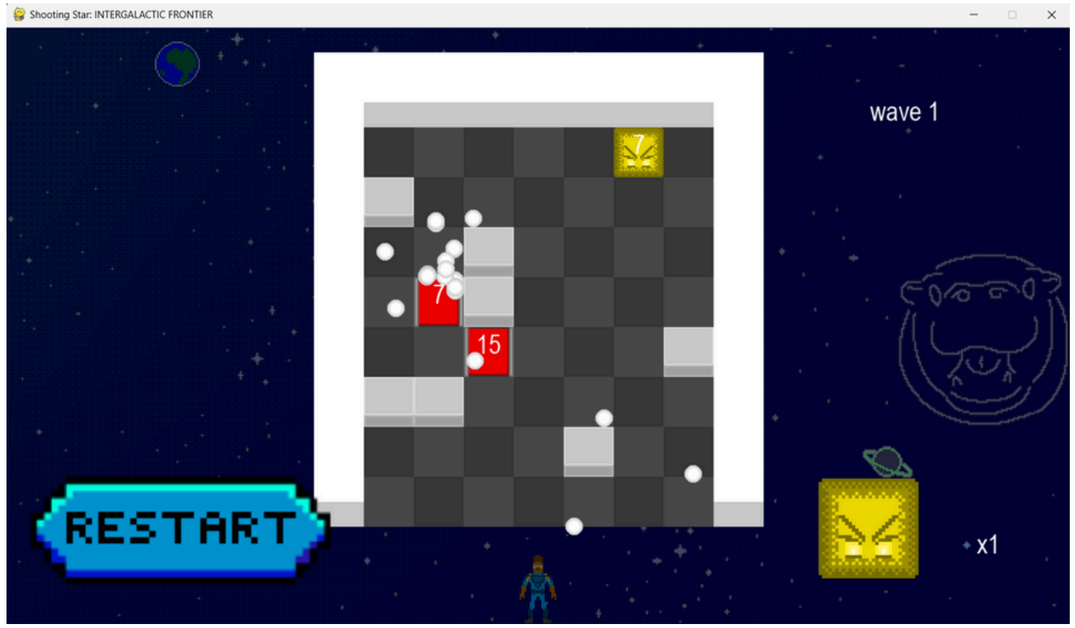
ที่ใช้รันเกมและหน้าจอ

Data

- MainMenu
- GameLoop

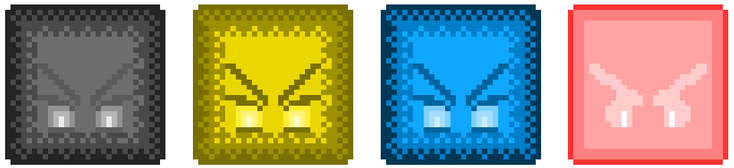
Methods

- __init__()
- update_background()
- main_menu()
- run_game()



GAME CLASS

องค์ประกอบที่จำเป็นของเกมหลักเพื่อให้เกมเล่นได้



Data

- Hero
- Monster(Easy, Medium, Hard, BOSS)
- Ball

Methods

- __init__()
- fire()
- shoot()
- update()
- take_damage()
- is_dead()
- move()
- check_collision()
- get_rect()

GAME CLASS

องค์ประกอบเสริมของเกมเพื่อความสวยงาม

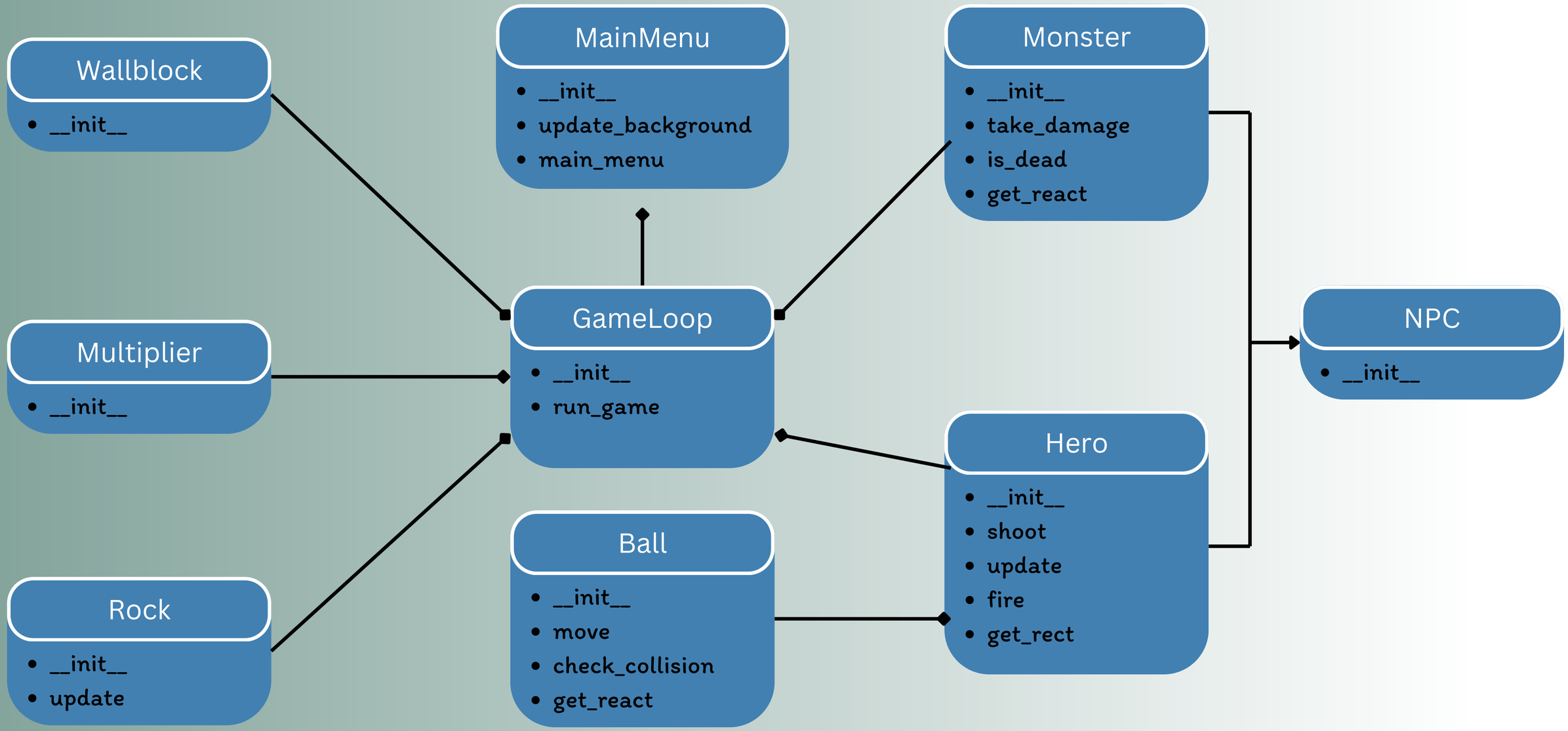
Data

- Wallblock
- Multiplier
- Rock

Methods

- __init__()
- update()

CLASS DIAGRAM

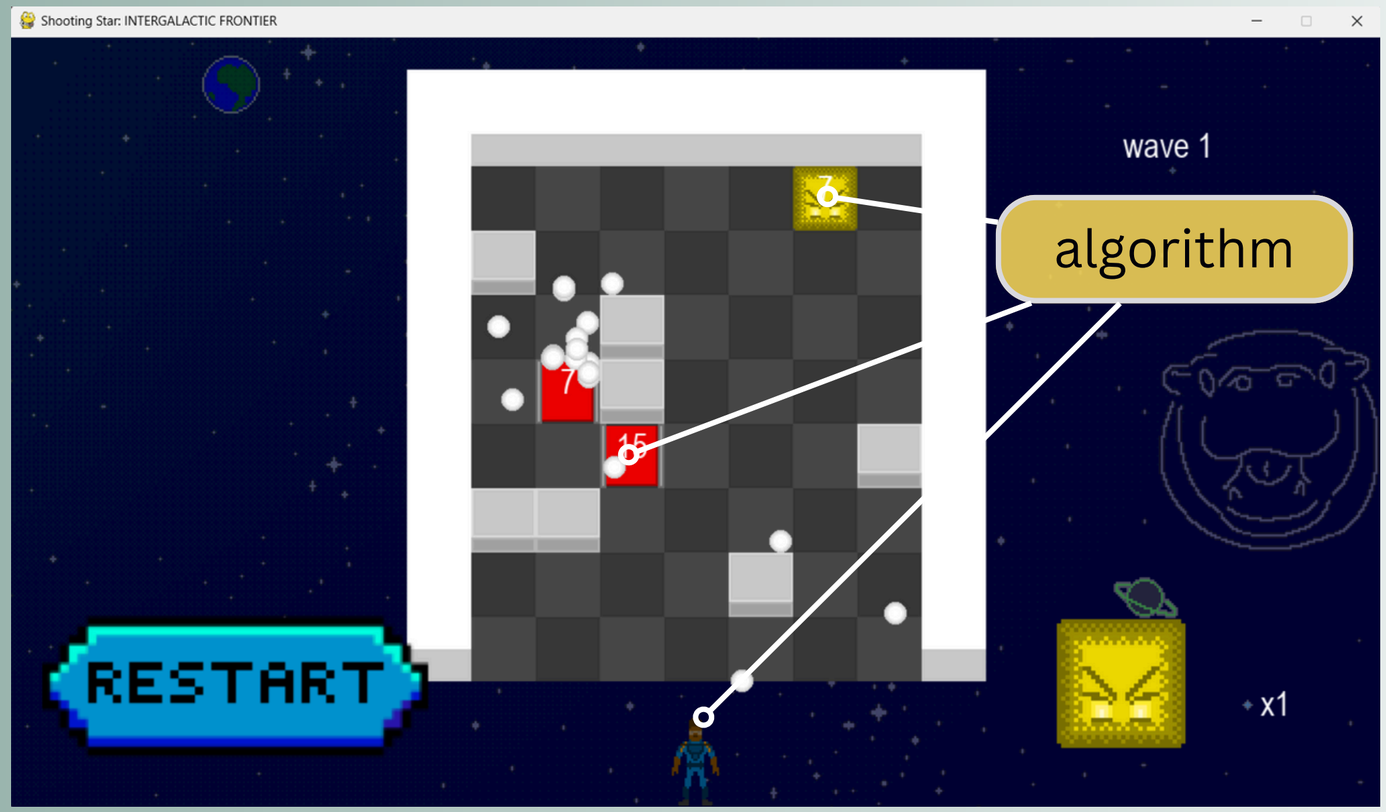


PUN BALL

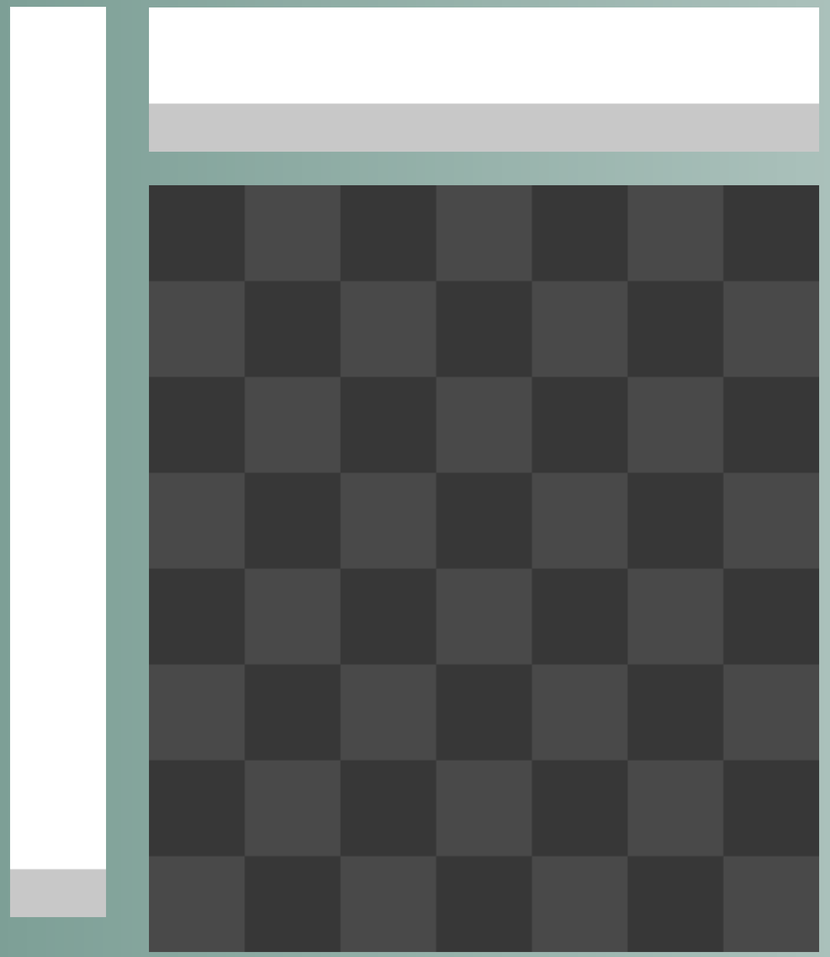
PROGRAM DESIGNS

STEP-3

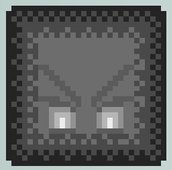
- Main menu
- Game loop
- Level
- NPC
- Ball



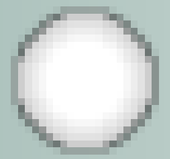
SPRITES



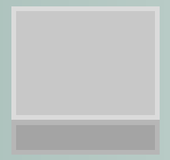
มอนส์เตอร์



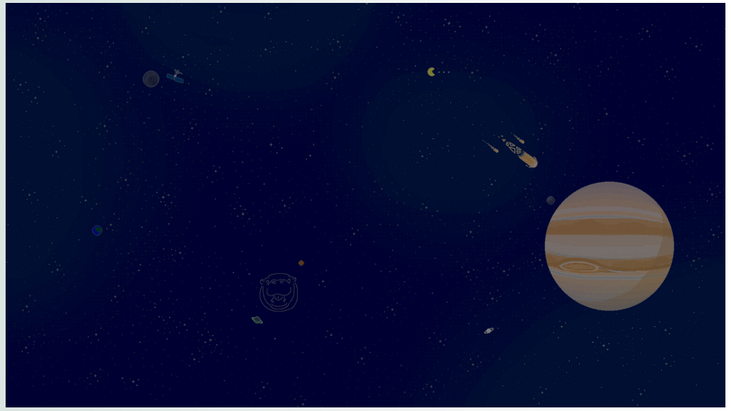
บอล



กำแพง

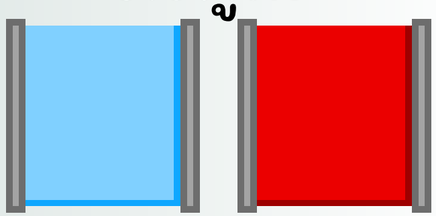


ฮีโร่



พื้นหลัง

ตัวคูณ

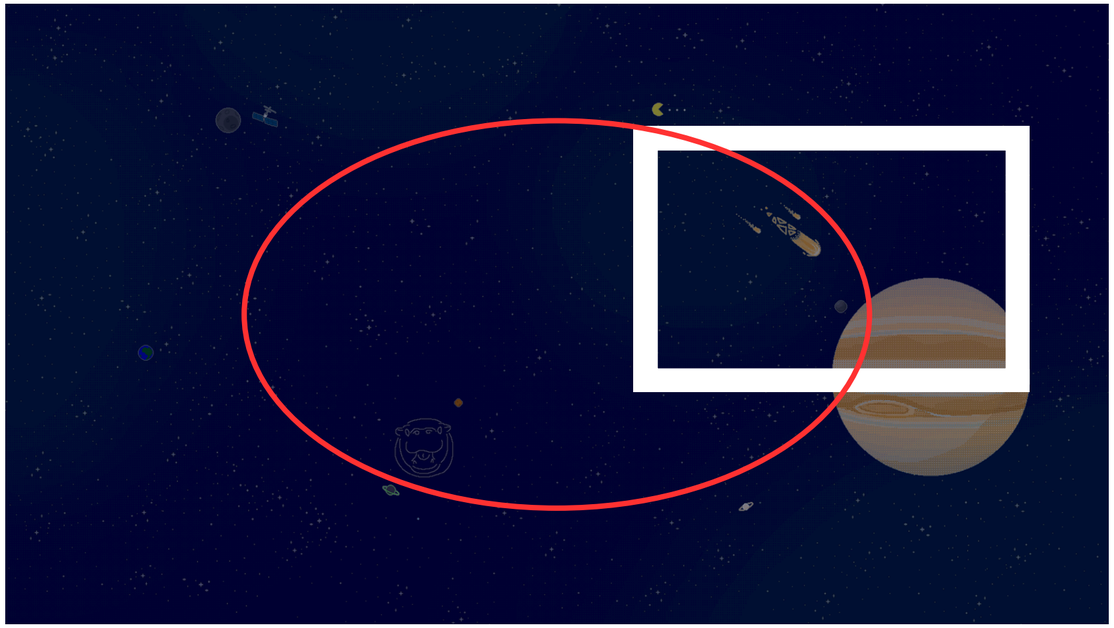


โปรแกรมที่ใช้

BACKGROUND

```
199     def run_game(self):
200         running = True
201         a = 1280  # Semi-major axis
202         b = 720  # Semi-minor axis
203         center_x = self.WIDTH // 2
204         center_y = self.HEIGHT // 2
205         angle = 59
206 
207         while running:
208             self.clock.tick(self.FPS)
209 
210             # Calculate background movement
211             bg_x = center_x + a * math.cos(angle) - self.background_width / 2
212             bg_y = center_y + b * math.sin(angle) - self.background_height / 2
213             angle += 0.001
214 
215             # Draw background
216             self.WIN.blit(self.background, (bg_x, bg_y))
```

จะได้มุมใหม่ทุกๆเฟรม



HERO

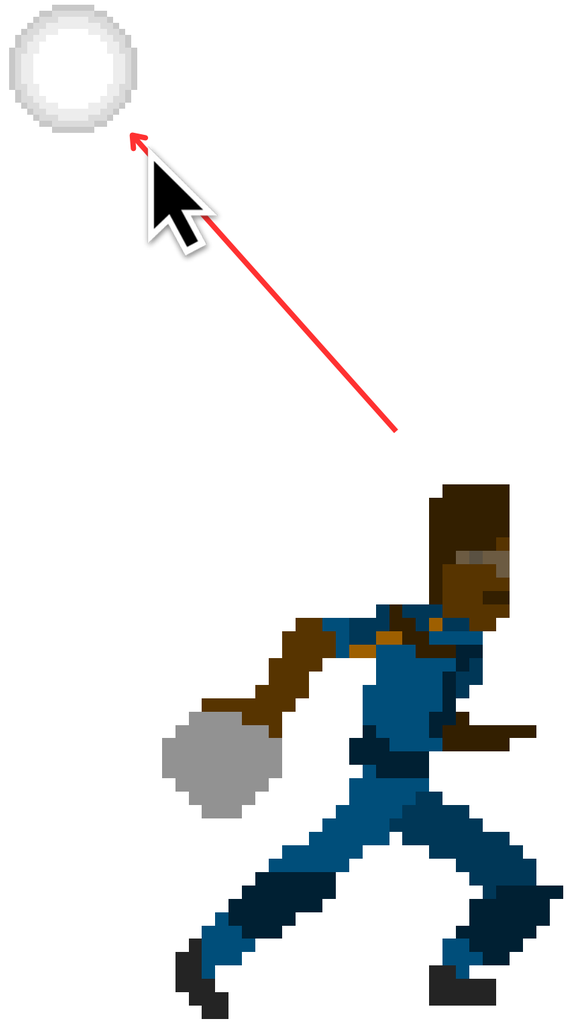
```
501         self.balls = []
502         self.can_shoot = True
503         self.shots_fired = 0
504         self.shoot_delay = 200
505         self.last_shot_time = 0
506         self.shooting = False
507
508     def shoot(self, target_x, target_y, texture):
509         if self.can_shoot and not self.shooting and len(self.balls) == 0:
510             self.shooting = True
511             self.shots_fired = 0
512             self.last_shot_time = pygame.time.get_ticks()
513             self.fire(target_x, target_y, texture)
514
515     def update(self):
516         current_time = pygame.time.get_ticks()
517         if self.shooting and self.shots_fired < 10:
518             if current_time - self.last_shot_time >= self.shoot_delay:
519                 target_x, target_y = pygame.mouse.get_pos()
520                 self.fire(target_x, target_y, self.balls[0].texture)
521                 self.last_shot_time = current_time
522                 self.shots_fired += 1
523             if self.shots_fired >= 10:
524                 self.shooting = False
525
526     def fire(self, target_x, target_y, texture):
527         ball = Ball(self.x, self.y - 20, target_x, target_y, texture)
528         self.balls.append(ball)
```

สร้างลิสต์บอลในคลาสฮีโร่

คำนวณทิศลูกบอลตามตำแหน่งเมาส์

เพิ่มลูกบอลเข้าลิสต์

target_x, target_y = pygame.mouse.get_pos()



BALL

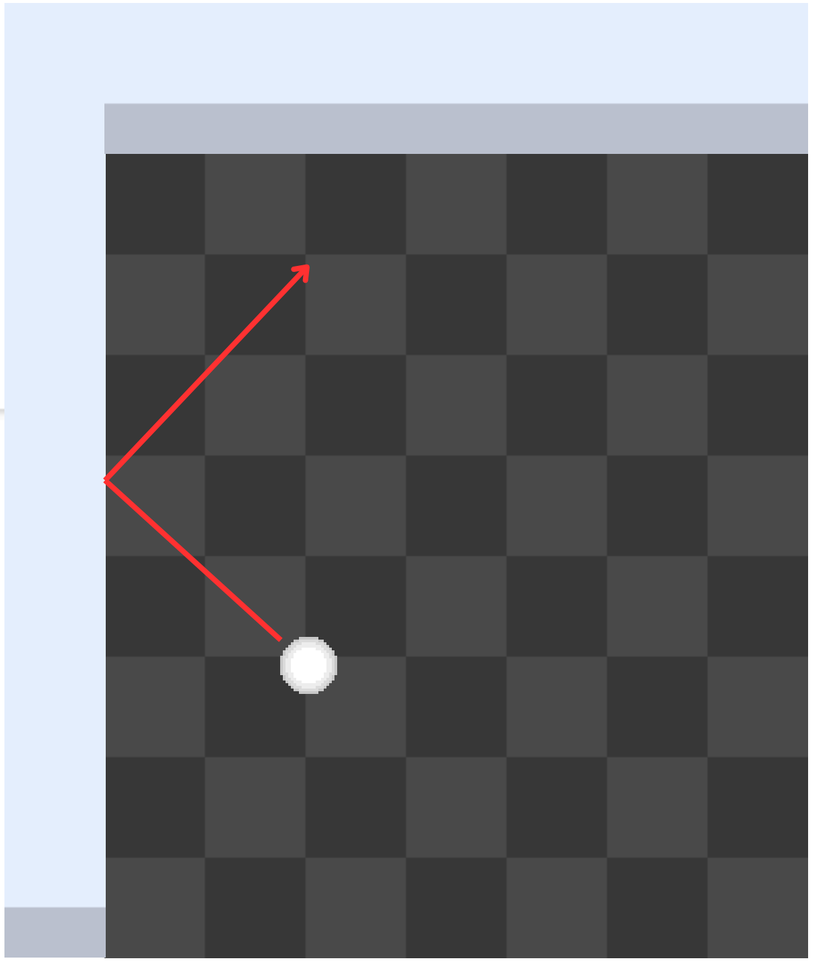
```
569  class Ball(pygame.sprite.Sprite):
588      def move(self):
589          if self.active:
590              self.x += self.x_vel
591              self.y += self.y_vel
592
593              if self.x <= 0 or self.x >= 1280:
594                  self.x_vel *= -1
595              if self.x <= 430 + 5 or self.x >= 850 - 20: # wall left and right
596                  self.x_vel *= -1
597              if self.y <= 0:
598                  self.y_vel *= -1
599              if self.y <= 120:
600                  self.y_vel *= -1
601              elif self.y >= 720:
602                  self.active = False
603
```

อัปเดตตำแหน่งบอล

ความเร็วแกน x จะคูณ -1
เมื่อแตะกำแพงซ้าย/ขวา

ความเร็วแกน y จะคูณ -1
เมื่อแตะกำแพงบน

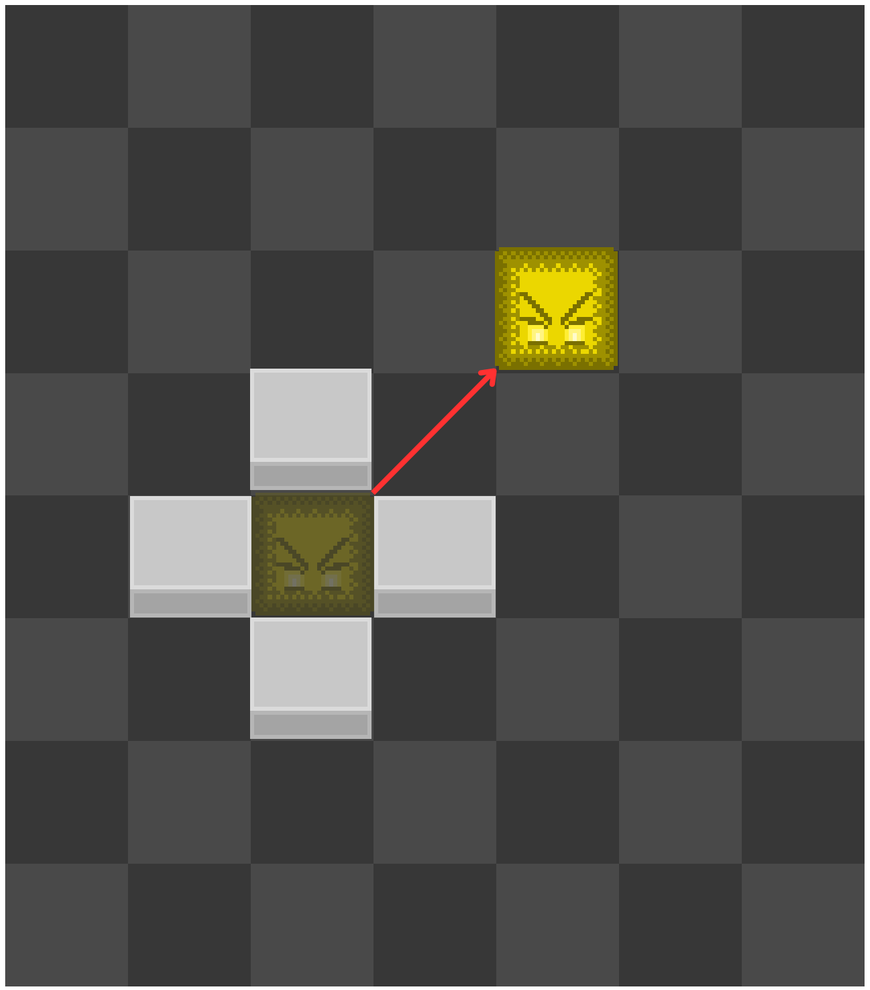
ลบบอลทิ้งเมื่อบอลออกด้านล่าง



MONSTER BEHAVIOR

```
341                 # monster moves each turn
342                 if len(self.hero.balls) == 0 and self.monster.movetile == True:
343                     grid = (random.randint(0, 6), random.randint(0, 7))
344                     self.monster.x = grid[0]*60 + 430
345                     self.monster.y = grid[1]*60 + 120
346                     self.monster.movetile = False
```

```
'''
    monsters position is determined by grid [7x8]
    (0,0)(1,0)(2,0)(3,0)(4,0)(5,0)(6,0)
    (0,1)(1,1)(2,1)(3,1)(4,1)(5,1)(6,1)
    (0,2)(1,2)(2,2)(3,2)(4,2)(5,2)(6,2)
    (0,3)(1,3)(2,3)(3,3)(4,3)(5,3)(6,3)
    (0,4)(1,4)(2,4)(3,4)(4,4)(5,4)(6,4)
    (0,5)(1,5)(2,5)(3,5)(4,5)(5,5)(6,5)
    (0,6)(1,6)(2,6)(3,6)(4,6)(5,6)(6,6)
    (0,7)(1,7)(2,7)(3,7)(4,7)(5,7)(6,7)

    Formula: (x*60)+430, (y*60)+120
'''
```

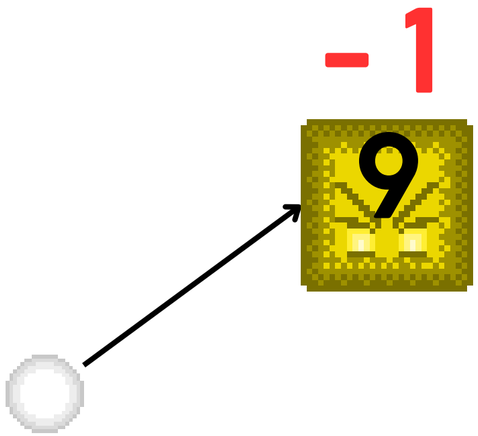


BALL & MONSTER

```python
def check_collision(self, monster):
    monster_rect = monster.get_rect()
    ball_rect = self.get_rect()

    if self.collision_cooldown == 0 and ball_rect.colliderect(monster_rect):
        overlap_left = ball_rect.right - monster_rect.left
        overlap_right = monster_rect.right - ball_rect.left
        overlap_top = ball_rect.bottom - monster_rect.top
        overlap_bottom = monster_rect.bottom - ball_rect.top

        min_overlap = min(overlap_left, overlap_right, overlap_top, overlap_bottom)

        if min_overlap == overlap_left:
            self.x = monster_rect.left - self.width
            self.x_vel *= -1
        elif min_overlap == overlap_right:
            self.x = monster_rect.right
            self.x_vel *= -1
        elif min_overlap == overlap_top:
            self.y = monster_rect.top - self.height
            self.y_vel *= -1
        elif min_overlap == overlap_bottom:
            self.y = monster_rect.bottom
            self.y_vel *= -1

        monster.take_damage(1)
```

บอลจะเด้งถ้าโดนมอนส์เตอร์

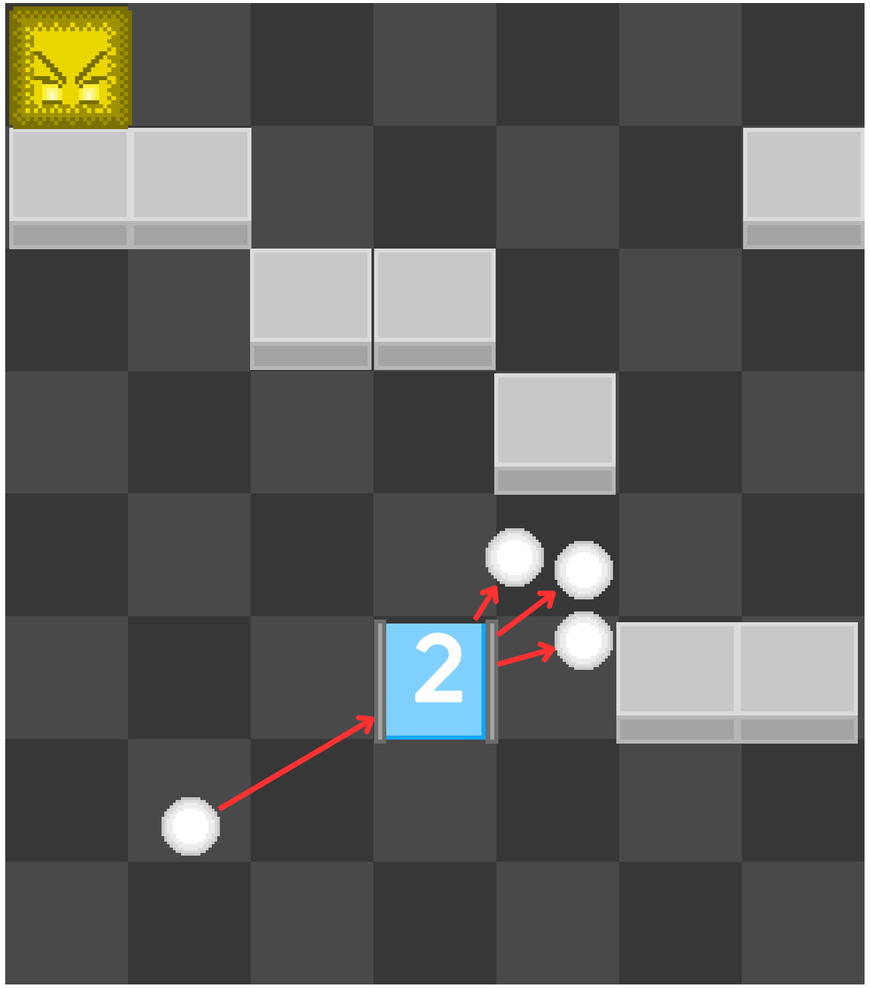
มอนส์เตอร์รับ 1 ดาเมจ



OBSTACLE

```
class Multiplier(pygame.sprite.Sprite):
-----------------------------------------
class Wallblock(pygame.sprite.Sprite):
    def __init__(self, texture):
        super().__init__()
        self.image = pygame.image.load(texture)
        self.rect = self.image.get_rect()
        grid = (random.randint(0, 6), random.randint(0, 6))
        self.rect.x = grid[0]*60 + 430
        self.rect.y = grid[1]*60 + 120
```

กำแพงและบล็อคเพิ่มกระสุนจะสุ่มจุดใหม่ทุกๆ wave

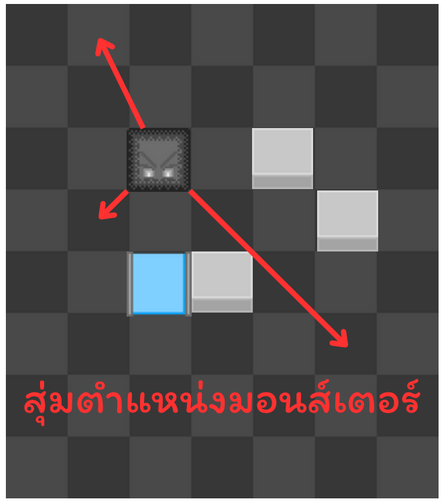



```
281             for multiplier in self.multiplier_list:
282                 ball_rect = ball.get_rect()
283                 multiplier_rect = multiplier.rect
284                 if len(self.hero.balls) == 0 and multiplier.active == False:
285                     multiplier.image = pygame.image.load("multiplier.png")
286                     multiplier.active = True
287 
288                 if ball_rect.colliderect(multiplier_rect) and multiplier.active == True:
289                     # Create new ball with slightly modified velocity
290                     for i in range(multiplier.multiply):
291                         dx = ball.x_vel + random.uniform(-50,50)
292                         dy = ball.y_vel + random.uniform(-50,50)
293                         new_ball = Ball(ball.x, ball.y, dx, dy, self.ball_texture)
294                         self.hero.balls.append(new_ball)
295                     multiplier.image = pygame.image.load("multiplier_inactive.png")
296                     multiplier.active = False
```

ลูกบอลจะเพิ่มขึ้นเมื่อผ่านตัวคูณ

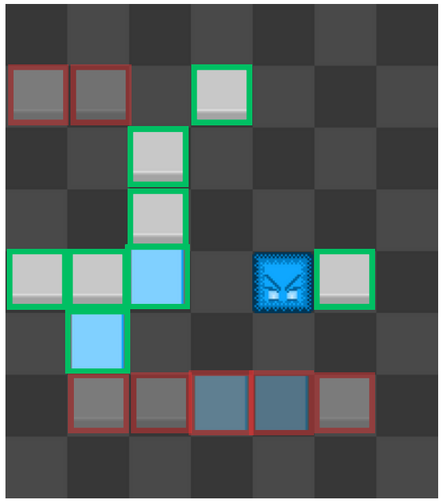
RANDOM STAGE

tutorial

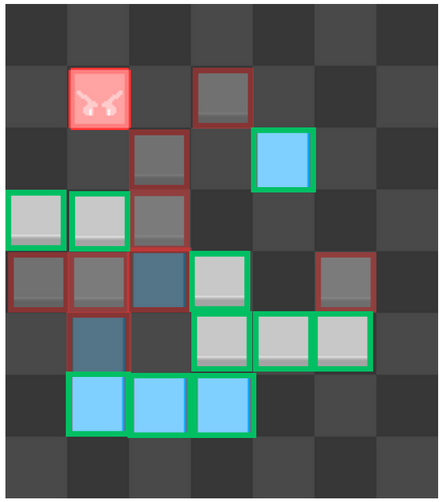


wave 1

wave 2



BOSS



```
grid = (random.randint(0, 6), random.randint(0, 6))
self.rect.x = grid[0]*60 + 430
self.rect.y = grid[1]*60 + 120
```

โค้ดที่ใช้สุ่มตำแหน่งสิ่งของต่างๆ
พบในทุกคลาสที่เป็นสิ่งกิดขวาง