

TUGAS BESAR PROGRAMMAN BERORIENTASI OBJEK

Kelompok 7
(Dorblezees)

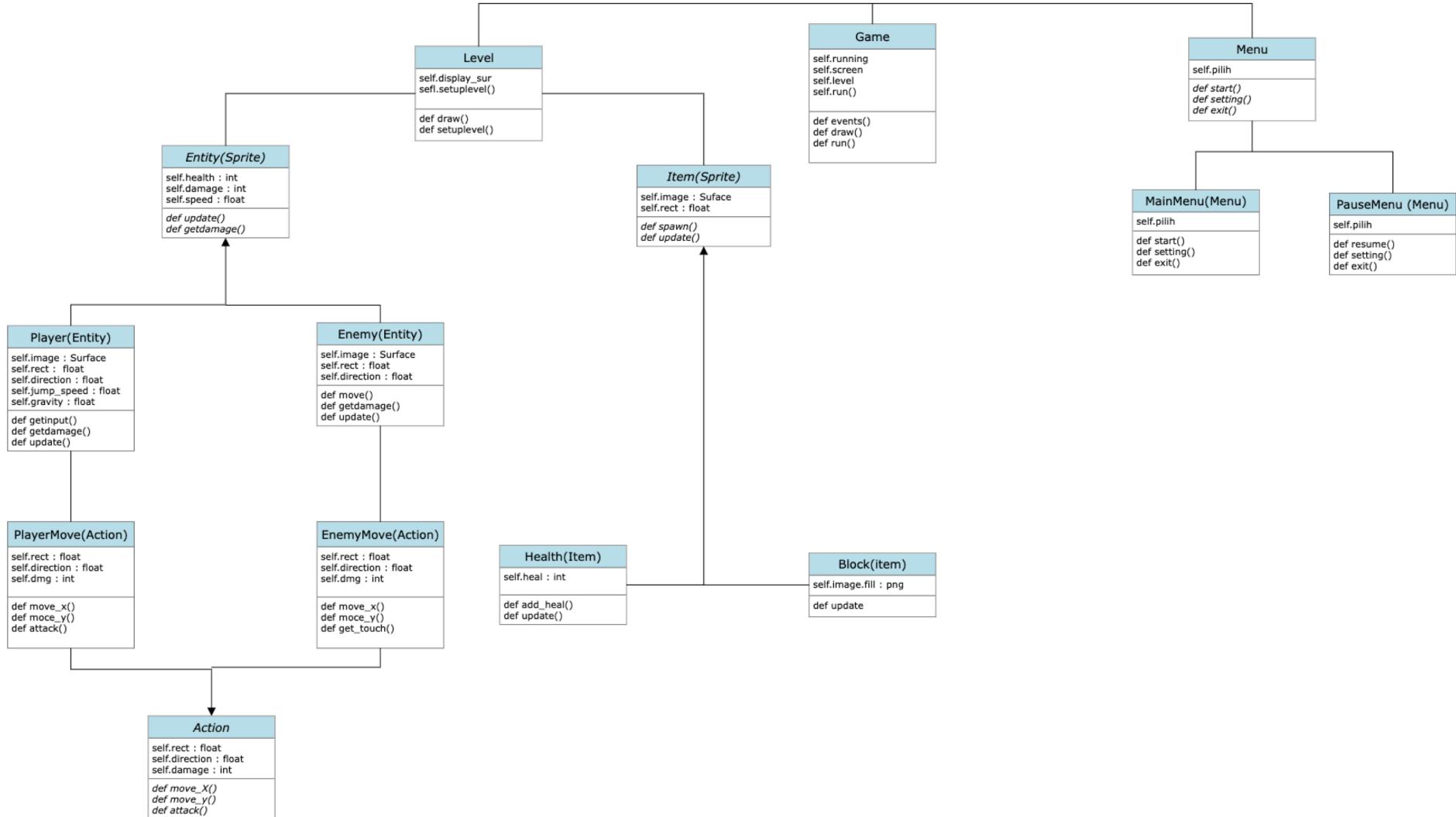
ANGGOTA

- **Muhammad Elang Permadani** (120140194) Project Leader
- **Hendri Aldi Zulfan** (120140186) Programmer
- **Daffa Sandri Ramadhan** (120140193) Character Designer
- **Muhammad Nur Aziz** (120140175) Game Designer
- **Bagus Ardin Saputra** (120140176) Audio Enginer
- **Reyhan Gandaresta** (120140183) Level Editor

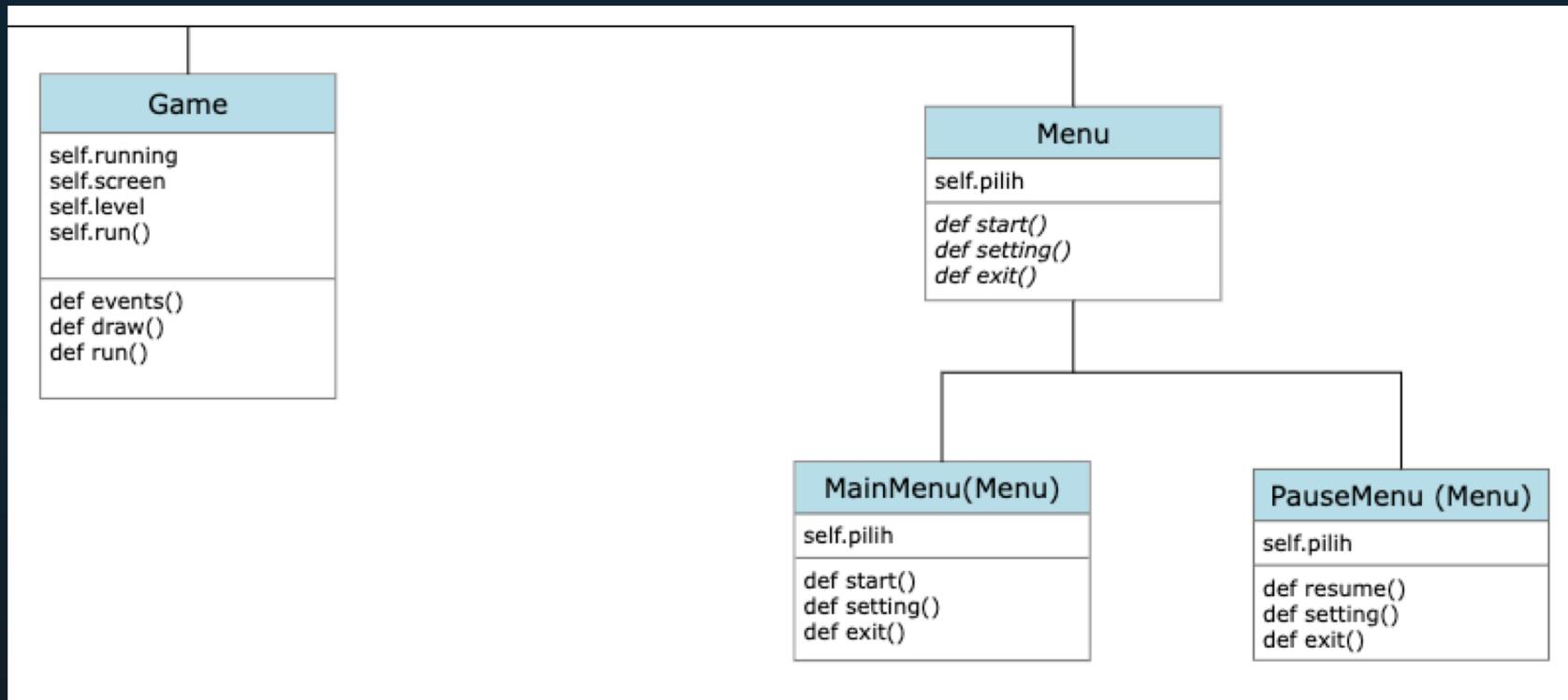
DESAIN UML CLASS DIAGRAM



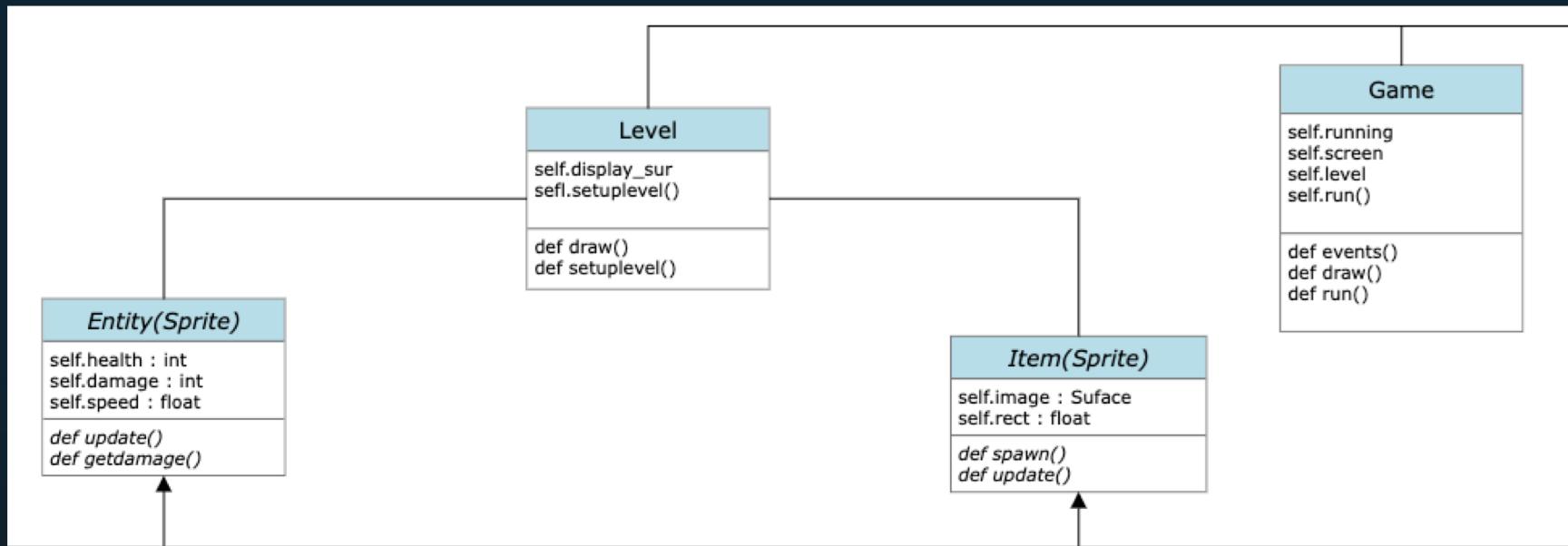
**Secara
Garis
Besar**



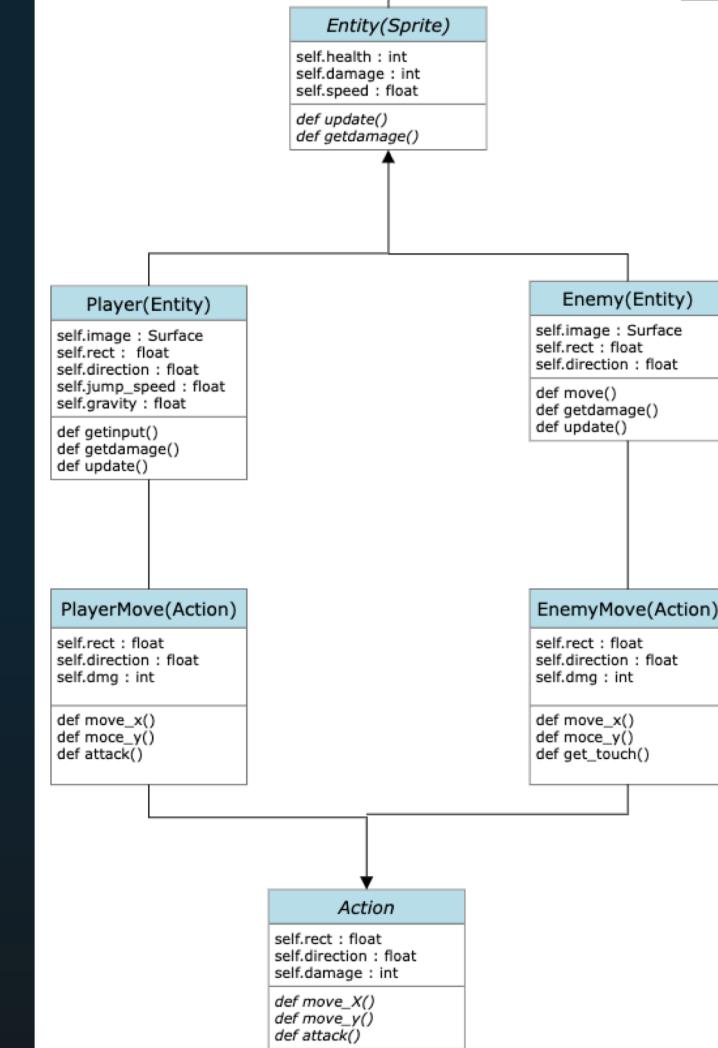
Lebih Rinci Bagian 1



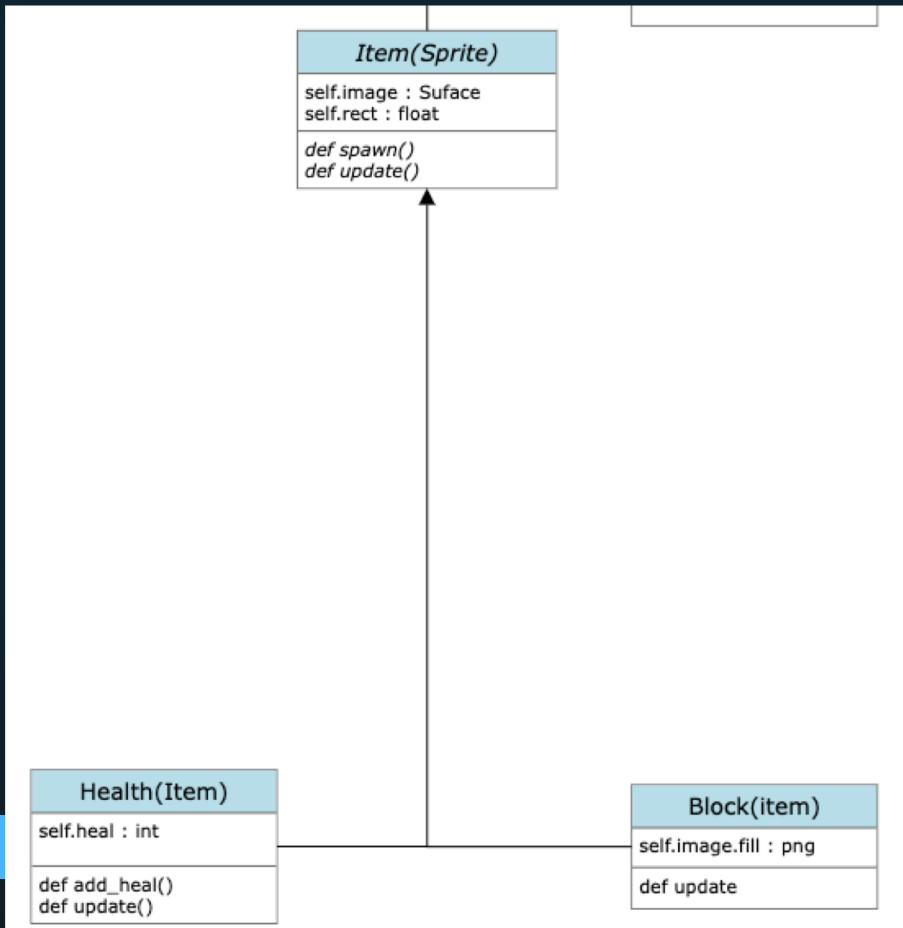
Lebih Rinci Bagian 2



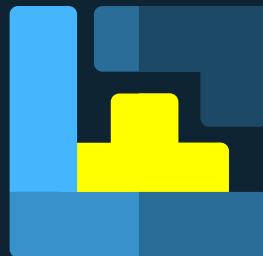
Lebih Rinci Bagian 3



Lebih Rinci Bagian 4



IMPLEMENTASI KELAS UTAMA



class Game

Class Game berfungsi sebagai class parent paling utama pada game ini, tujuannya adalah untuk menampung hampir semua perkondisian secara garis besar yang ada pada game, contohnya seperti, ketika game pertama kali dibuka, ketika game memilih level yang akan dimainkan, ketika menampilkan menu, ketika game telah menang, ketika game telah kalah, dan lain lain.

class Game

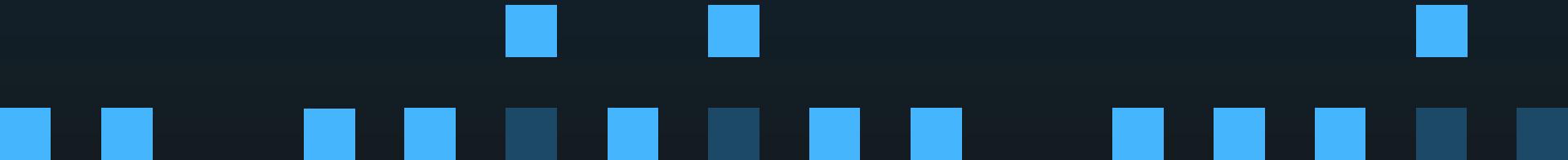
Contoh Implementasinya adalah sebagai berikut, yaitu ketika health dari player sudah <= nol

Note : (source code masih dalam tahap pengembangan)

```
● ● ●  
1 class Game:  
2     def __init__(self):  
3         self.running = True  
4         self.run()  
5  
6     def events(self):  
7         pass  
8  
9     def draw(self):  
10        pass  
11  
12    def game_over(self):  
13        if self.level.Player.health <= 0:  
14            self.running = False  
15  
16    def run(self):  
17        while self.running:  
18            self.game_over()  
19            self.events()  
20            self.draw()
```

class Game

Contoh hasil implementasi class Game yang diharapkan :



class Menu

Class Menu disini akan berfungsi sebagai class parent dari submenu yang ada contohnya seperti menu yang akan muncul pada awal ketika game pertama kali dibuka, dan juga menu yang akan muncul ketika mem-pause game pada saat game sedang berlangsung. Class menu sendiri akan ditampung secara langsung oleh Class Game yang mana sebagai pondasi awal untuk menampilkan menu

class Menu

Contoh Implementasinya adalah sebagai berikut, yaitu ketika Awal Game pertama kali dibuka

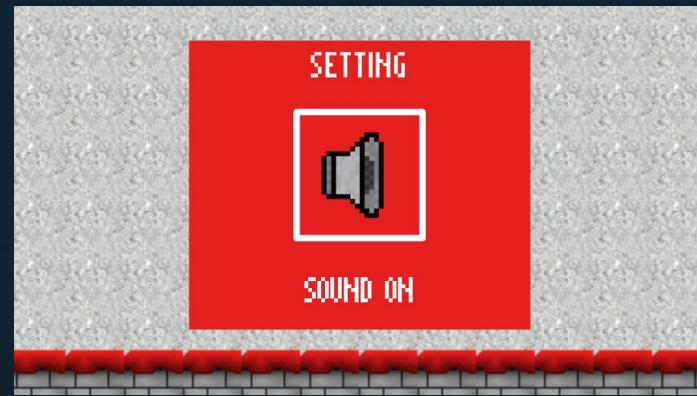
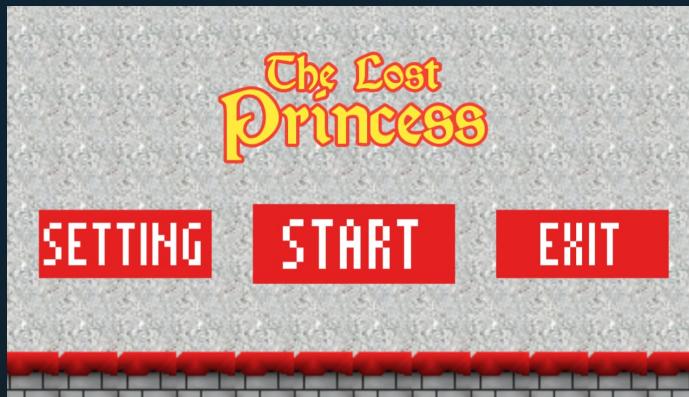
Note : (source code masih dalam tahap pengembangan)

```
1 class Menu:  
2     def __init__(self, surface, submenu):  
3         self.surface = surface  
4         self.submenu = []  
5         self.submenu.append(submenu)  
6         self.tampil()  
7  
8     def tampil(self):  
9         pass
```

```
1 class Game:  
2     def __init__(self):  
3         self.screen = pygame.display.set_mode((SC_HEIGHT, SC_WIDTH))  
4         self.menu_awal = Menu(self.screen, ['Start', 'Setting', 'Quit'])  
5         self.run()  
6  
7     def events(self):  
8         pass  
9  
10    def draw(self):  
11        pass  
12  
13    def game_over(self):  
14        pass  
15  
16    def run(self):  
17        self.menu_awal.tampil()  
18        while self.running:  
19            self.events()  
20            self.clock.tick(self.fps)  
21            self.draw()
```

class Menu

Contoh hasil implementasi class menu yang diharapkan :



class Level

Class Level berfungsi untuk menampung apa yang akan ditampilkan ketika game sedang berjalan, seperti : Map, Player, Enemy, Item, Obstacle, dan lain-lain. Class level sendiri akan ditampung oleh class Game yang mana kita tau class Game adalah class yang akan menampung hampir keseluruhan alur jalannya game.

class Level

Contoh Implementasinya adalah sebagai berikut, yaitu untuk menampilkan map yang sudah dirancang sedemikian rupa

Note : (source code masih dalam tahap pengembangan)

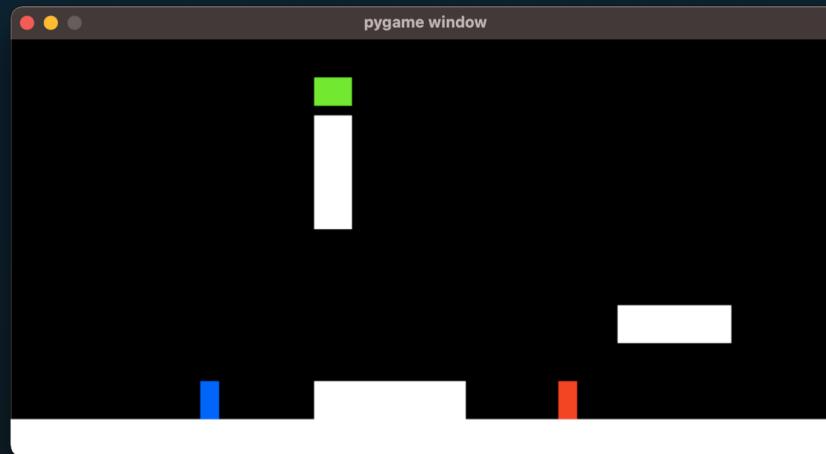


```
1 LEVEL_SET =[  
2 ' ',  
3 ' ',  
4 ' T',  
5 ' #',  
6 ' #',  
7 ' #',  
8 ' ',  
9 ' #####',  
10 ' P #### E',  
11 ' #####',  
12 ' #####',]
```

```
1 class Level:  
2     def __init__(self, level, surface):  
3         self.display_sur = surface  
4         self.setuplevel(level)  
5  
6     def setuplevel(self, layout):  
7         self.Block = pygame.sprite.Group()  
8         self.Health = pygame.sprite.Group()  
9         self.Player = pygame.sprite.GroupSingle()  
10        self.Enemy = pygame.sprite.GroupSingle()  
11  
12        for row_index, row in enumerate(layout):  
13            for cell_index, cell in enumerate(row):  
14                x = cell_index * TILE_SIZE  
15                y = row_index * TILE_SIZE  
16                if cell == '#':  
17                    tile = Block((x, y), TILE_SIZE)  
18                    self.Block.add(tile)  
19                if cell == 'P':  
20                    char = Player((x, y))  
21                    self.Player.add(char)  
22                if cell == 'E':  
23                    enemy = Enemy((x, y))  
24                    self.Enemy.add(enemy)  
25                if cell == 'T':  
26                    health = Health((x, y), TILE_SIZE)  
27                    self.Health.add(health)  
28  
29        def draw(self):  
30            self.Block.update(0)  
31            self.Block.draw(self.display_sur)  
32            self.Health.update(0)  
33            self.Health.draw(self.display_sur)  
34            self.Enemy.update()  
35            self.Enemy.draw(self.display_sur)  
36            self.Player.update()  
37            self.Player.draw(self.display_sur)
```

class Level

Contoh hasil implementasi class Level yang diharapkan :



class Entity

Class Entity berfungsi untuk membuat suatu objek yang bias bergerak dan juga melakukan sesuatu seperti berjalan, menyerang, lompat, dan lain-lain. Contoh penggunaan class entity terdapat pada objek seperti Player dan Enemy. Class entity ini sendiri akan di tamping oleh class Level yang mana kita tau Class Level adalah Class yang akan menampilkan apapun ketika game sedang berjalan.

class Entity

Contoh Implementasinya adalah sebagai berikut, yaitu untuk menggerakan Enemy

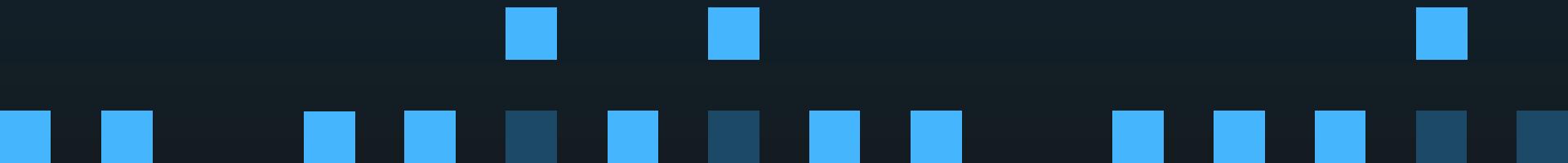
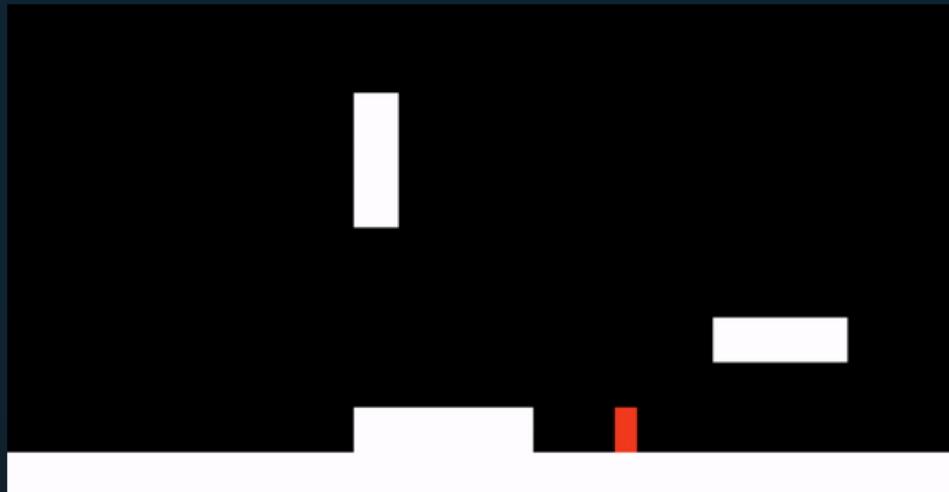
Note : (source code masih dalam tahap pengembangan)

```
● ● ●  
1 class Entity(pygame.sprite.Sprite):  
2     def __init__(self, health, damage, speed):  
3         super().__init__()  
4         self.health = health  
5         self.damage = damage  
6         self.speed = speed  
7  
8     def update(self):  
9         pass
```

```
● ● ●  
1 class Enemy(Entity):  
2     def __init__(self, pos):  
3         super().__init__(20, 3, 1)  
4         self.image = pygame.Surface((16, 32))  
5         self.image.fill('red')  
6         self.rect = self.image.get_rect(topleft = pos)  
7         self.direction = pygame.math.Vector2(0, 0)  
8  
9     def move(self):  
10        EnemyMove(self.direction, self.rect, 544)  
11  
12     def update(self):  
13         self.move()  
14         self.rect.x += self.direction.x * self.speed
```

class Entity

Contoh hasil implementasi class Entity yang diharapkan :



class Item

Fungsi dari Class Item tidak jauh berbeda dengan class Entity, hanya saja class item tidak bias menampung objek yang bias melakukan sesuatu seperti bergerak. Contoh penggunaan class item ada pada Block yang akan tersusun sedemikian rupa untuk membentuk Map yang akan dipilih, dan juga Item yang bias di ambil oleh player guna mendapatkan tambahan statistik.

class Item

Contoh Implementasinya adalah sebagai berikut, yaitu untuk menyusun Banyak Block untuk membuat sebuah Map

Note : (source code masih dalam tahap pengembangan)

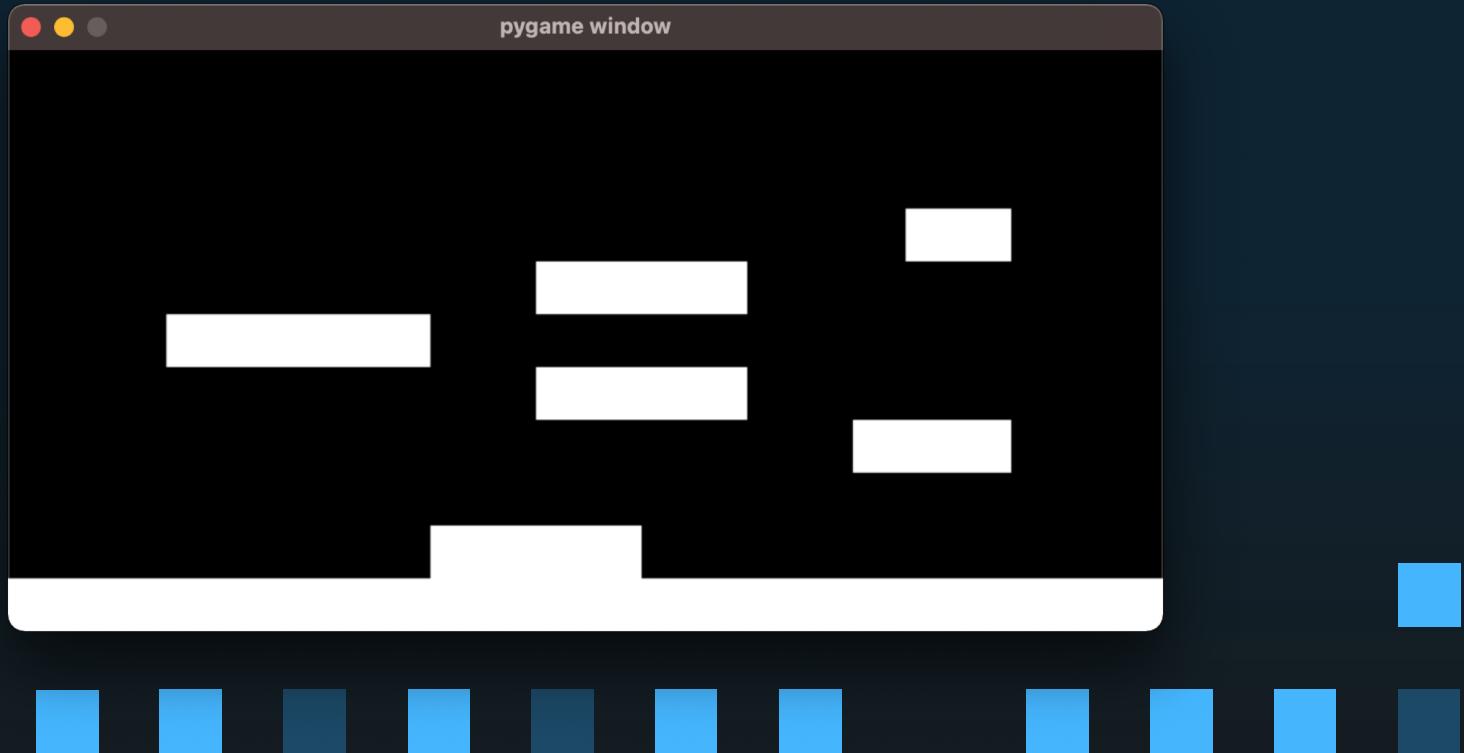
```
● ○ ●
1 class Item(pygame.sprite.Sprite):
2     def __init__(self, pos, size):
3         super().__init__()
4         self.image = pygame.Surface((size, size))
5         self.rect = self.image.get_rect(topleft = pos)
6
7     def spawn(self):
8         pass
9
10    def update(self):
11        pass
```

```
● ○ ●
1 LEVEL_SET = [
2     '',
3     '',
4     '',
5     '      ##',
6     '     ###',
7     '#####',
8     '###',
9     '##',
10    '###',
11    '#####',
12    '#####',]
```

```
● ○ ●
1 class Block(Item):
2     def __init__(self, pos, size):
3         super().__init__(pos, size)
4         self.image.fill('white')
5
6     def update(self, x_shift):
7         self.rect.x += x_shift
```

class Item

Contoh hasil implementasi class Item yang diharapkan :



class Action

Fungsi dari class Action adalah untuk menggerakan objek dari class Entity seperti Player dan juga Enemy. Class Action ini sendiri di tampung oleh class Entity yang membutuhkannya.

class Action

Contoh Implementasinya adalah sebagai berikut, yaitu untuk menggerakan Enemy

Note : (source code masih dalam tahap pengembangan)



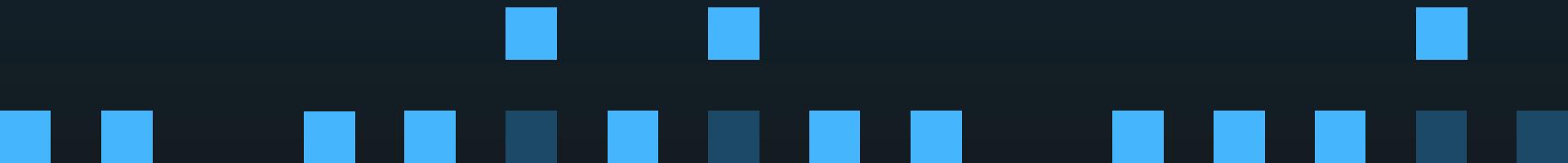
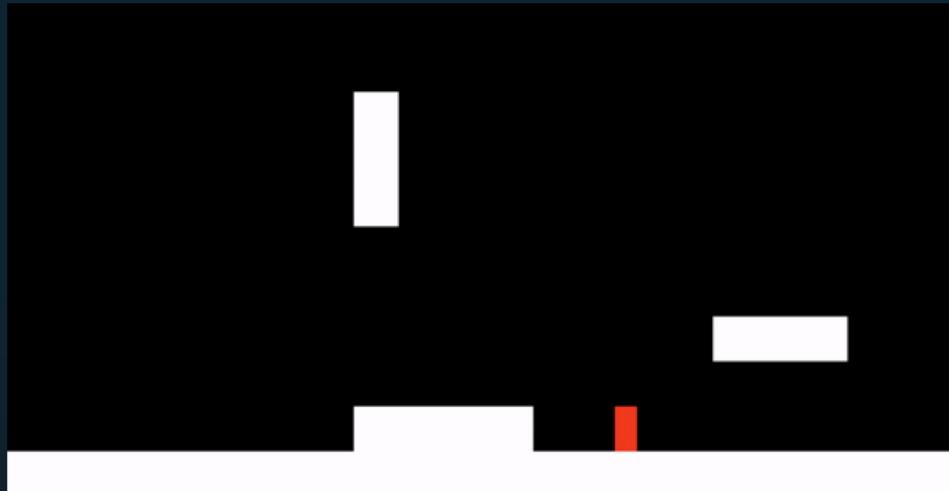
```
1 class Action:  
2     def __init__(self, direction):  
3         self.direction = direction  
4  
5     def move_x(self):  
6         pass  
7  
8     def move_y(self):  
9         pass
```



```
1 class EnemyMove(Action):  
2     def __init__(self, direction, rect, pos):  
3         super().__init__(direction)  
4         self.rect = rect  
5         self.move_x(pos)  
6  
7     def move_x(self, pos):  
8         if self.rect.x >= pos:  
9             self.direction.x = -1  
10            print(self.rect.x)  
11        if self.rect.x <= 384:  
12            self.direction.x = 1  
13            print(self.rect.x)
```

class Action

Contoh hasil implementasi class Action yang diharapkan :



**Terima
Kasih**