Shoot it

Вашему вниманию предоставляется шутер «Shoot it» подготовленный Юдиной Дарьей, студенткой группы БСБО-09-22.

Введение:

Игра "Shoot it" была разработана как современная интерпретация классического шутера DOOM, сохраняя при этом стилистику и атмосферу оригинала. Игроки встречаются с графикой, характерной для тех времён, узнаваемыми моделями и мотивами песен, что создаёт ностальгическое ощущение.

Разработка:

Игра была создана на языке программирования Python с использованием библиотеки Pygame, что позволило воссоздать классический игровой процесс с добавлением современных элементов. Для создания трёхмерных моделей было разработано множество алгоритмов, основанных на сложных математических вычислениях, что позволило достичь желаемого эффекта визуализации.

```
1 usage   DariaYudina004
def mini_map(self, player):
    self.sc_map.fill(BLACK)
    map_x, map_y = player.x // MAP_SCALE, player.y // MAP_SCALE
    pygame.draw.line(self.sc_map, YELLOW, start_pos: (map_x, map_y), end_pos: (map_x + 12 * math.cos(player.angle),
                                                                               map_y + 12 * math.sin(player.angle)), width: 2)
    pygame.draw.circle(self.sc_map, RED, center: (int(map_x), int(map_y)), radius: 5)
    for x, y in mini_map:
        pygame.draw.rect(self.sc_map, DARKBROWN, rect: (x, y, MAP_TILE, MAP_TILE))
    self.sc.blit(self.sc_map, MAP_POS)
```

Рисунок 1. Программная реализация мини карты

В углу левом нижнем углу перед пользователем располагается мини карта. На ней можно проследить не только все находящиеся на сцене предметы и объекты, так же она в реальном времени отображает местонахождение и передвижение игрока.

```python
@njit(fastmath=True, cache=True)
def ray_casting_npc_player(npc_x, npc_y, blocked_doors, world_map, player_pos):
    ox, oy = player_pos
    xm, ym = mapping(ox, oy)
    delta_x, delta_y = ox - npc_x, oy - npc_y
    cur_angle = math.atan2(delta_y, delta_x)
    cur_angle += math.pi

    sin_a = math.sin(cur_angle)
    sin_a = sin_a if sin_a else 0.000001
    cos_a = math.cos(cur_angle)
    cos_a = cos_a if cos_a else 0.000001

    # verticals
    x, dx = (xm + TILE, 1) if cos_a >= 0 else (xm, -1)
    for i in range(0, int(abs(delta_x)) // TILE):
        depth_v = (x - ox) / cos_a
        yv = oy + depth_v * sin_a
        tile_v = mapping(x + dx, yv)
        if tile_v in world_map or tile_v in blocked_doors:
            return False
        x += dx * TILE

    # horizontals
    y, dy = (ym + TILE, 1) if sin_a >= 0 else (ym, -1)
    for i in range(0, int(abs(delta_y)) // TILE):
        depth_h = (y - oy) / sin_a
        xh = ox + depth_h * cos_a
        tile_h = mapping(xh, y + dy)
        if tile_h in world_map or tile_h in blocked_doors:
            return False
        y += dy * TILE
    return True
```

Рисунок 2. Просчет траектории полета пули от игрока к NPC

```python
1 usage  DariaYudina004
def npc_action(self):
    for obj in self.sprites.list_of_objects:
        if obj.flag == 'npc' and not obj.is_dead:
            if ray_casting_npc_player(obj.x, obj.y,
                                      self.sprites.blocked_doors,
                                      world_map, self.player.pos):
                obj.npc_action_trigger = True
                self.npc_move(obj)
            else:
                obj.npc_action_trigger = False
```

Рисунок 3. Скрипт отвечающий за анимации персонажей на сцене

Если персонаж не мертв и, если он не встретился с игроком, у него проигрывается анимация бездействия, если встреча произошла, начинается анимация атаки и если этот самый NPC потерпел поражения п ход вступает анимация смерти.

```
1 usage   DariaYudina004
def npc_move(self, obj):
    if abs(obj.distance_to_sprite) > TILE:
        dx = obj.x - self.player.pos[0]
        dy = obj.y - self.player.pos[1]
        obj.x = obj.x + 1 if dx < 0 else obj.x - 1
        obj.y = obj.y + 1 if dy < 0 else obj.y - 1
```

Рисунок 4. Код алгоритма передвижения NPC

```
1 usage   DariaYudina004
def check_win(self):
    if not len([obj for obj in self.sprites.list_of_objects if obj.flag == 'npc' and not obj.is_dead]):
        pygame.mixer.music.stop()
        pygame.mixer.music.load('sound/win.mp3')
        pygame.mixer.music.play()
        while True:
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    exit()

            self.drawing.win()
```

Рисунок 5. Проверка окончания игры

Игра продолжается до тех пор, пока игрок не выследит и не одолеет всех врагов. Если игрок остался жив, а каждый из монстров разгуливающий по сцене погиб, игра оканчивается, перед пользователем появляется уведомление о его победе, а также начинается торжественная мелодия

Игровой процесс:

При входе в игру игроков встречает динамический экран с выбором начать игру или выйти. После начала игры пользователь попадает в разрушенное и токсичное пространство, где ему предстоит столкнуться с опасными монстрами. Игра продолжается до тех пор, пока на поле боя не останется один игрок. После уничтожения последнего NPC появляется уведомление о победе с возможностью выхода из игры.

Заключение:

"Shoot it" представляет собой уникальное сочетание классического и современного дизайна, предлагая игрокам новый взгляд на любимую игру. Разработка требовала значительных усилий, особенно в части программирования и математических расчётов, но результат оправдал все ожидания, предоставив игрокам захватывающий и ностальгический опыт.