

# Space Shooter

Максим Харламов

Дарья Шевченко

# Объем кода + комментарии + классы

```
screen.blit(enter, enterR)

pygame.display.flip()
clock.tick(60)
if is_level:
    end_game = True # флаг того, как закончилась игра
    # выходом или заходом в уровень
    music_play = (
        True # флаг того, что можно снова играть музыку победы/
        проигрыша
    )
    is_levels_window = False
    # r = False
    score = start_level(d[level_n])
    update(name, score)
    win = score > 0 # победил человек или нет
if is_settings_window:
    # окно с настройками
```

```
class Enemy(pygame.sprite.Sprite):
    def __init__(self, type, pos_x, pos_y):
        super().__init__(enemy_group)
        self.type, image, self.bullet_filename, duration, self.speeds, self.
            points = enemies[type]
        self.image = load_image(image)
        self.mask = pygame.mask.from_surface(self.image)
        self.rect = self.image.get_rect().move(pos_x, pos_y)
        self.rect.x -= self.rect.w // 2
        self.rect.y -= self.rect.h // 2

        EVENT = pygame.USEREVENT + len(events) + 1
        pygame.time.set_timer(EVENT, duration)
```

```
class Present(pygame.sprite.Sprite):
    """
    types:
    0) меняет тип игрока на 0
    1) меняет тип игрока на 1
    2) меняет тип игрока на 2
    3) увеличивает скорость пуля
    4) уменьшает задержку между пулями
    5) увеличивает скорость игрока
    """

    def __init__(self, x, y, type):
        if not (0 <= type < len(presents)):
            return
        super().__init__(presents_group)
```

```
class Animation(pygame.sprite.Sprite):
    # в папке должны быть файлы 1.png, 2.png, 3.png ...
    def __init__(self, x, y, folder,
        n=10):
        super().__init__(animation_group)
        self.images = [load_image(f"{folder}\\{i}.png", None,
            in range(1, n + 1))]
        self.index = -1
```

```
278         animation_group.draw(screen)
279
280         if not enemy_group.sprites():
281             running = False
282
283         pygame.display.flip()
284         clock.tick(FPS)
285         return score if len(events) == 1 else 0
286
287
288 if __name__ == "__main__":
289     print(start_level("q.txt"))
290
```



```
482         pygame.display.flip()
483         clock.tick(60)
484
485         pygame.quit()
486
487
488 start_window()
489
```

# requirements.txt



The screenshot shows the GitHub interface for a file named `requirements.txt` in the repository `Space-Shooter`. The file is owned by `MaksimKharlamov` and was last committed by `d6622a5` yesterday. The file's content is displayed as `pygame` on line 1. The interface includes navigation tabs for `Code` and `Blame`, and a status bar indicating the file's size (7 Bytes) and a suggestion to use GitHub Copilot. Action buttons for `Raw`, `Copy`, `Download`, `Edit`, and `Diff` are also visible.

Space-Shooter / requirements.txt

MaksimKharlamov requirements.txt d6622a5 · yesterday History

Code Blame 1 lines (1 loc) · 7 Bytes Code 55% faster with GitHub Copilot Raw Copy Download Edit Diff

```
1  pygame
```

Стартовое  
ОКНО

# Space Shooter

Войти

Играть

Настройки

# Финальное окно

Unknown

Миссия проиграна

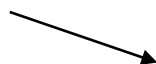
0 - Очки

Назад

# Подсчет результатов

Dasha

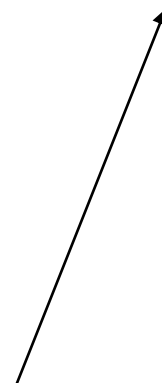
Your Score: 60



Dasha

Mission Accomplished

60 - Score



Dasha

Your Score: 120

# Спрайты + collide

```
self.rect.x += self.vx
self.rect.y += self.vy
if pygame.sprite.spritecollide(self, player_group, False, pygame.
    sprite.collide_mask):
    presents_group.remove(self)
    if self.type == 3:
```

```
def update(self):
    global running
    if pygame.sprite.spritecollide(self, enemy_bullets, False, pygame.
        sprite.collide_mask):
        running = False
```

```
player_group = pygame.sprite.Group()
enemy_group = pygame.sprite.Group()
bullet_group = pygame.sprite.Group()
player_bullets = pygame.sprite.Group()
enemy_bullets = pygame.sprite.Group()
presents_group = pygame.sprite.Group()
animation_group = pygame.sprite.Group()
```

```
player_group.update()
player_group.draw(screen)
```

```
bullet_group.update()
bullet_group.draw(screen)
```

```
enemy_group.update()
enemy_group.draw(screen)
```

```
presents_group.update()
presents_group.draw(screen)
```

```
animation_group.update()
animation_group.draw(screen)
```

# Анимация

```
class Animation(pygame.sprite.Sprite):
    # в папке должны быть файлы 1.png, 2.png, 3.png ...
    def __init__(self, x, y, folder,
                 n=10):
        super().__init__(animation_group)
        self.images = [load_image(f"{folder}\\{i}.png", None, 100, 100) for i
                        in range(1, n + 1)]
        self.index = -1
        self.x = x
        self.y = y

    def update(self):
        self.index += 1
        if self.index >= len(self.images):
            animation_group.remove(self)
            return
        self.image = self.images[self.index]
        self.rect = self.image.get_rect().move(self.x, self.y)
        self.rect.x -= self.rect.w // 2
        self.rect.y -= self.rect.h // 2
```



# Несколько уровней

Level 1

Level 2

Level 3

# Хранение данных

```
connection = sqlite3.connect("data/shooter_bd.db")
cursor = connection.cursor()
cursor.execute(
    """
    CREATE TABLE IF NOT EXISTS Players (
    id INTEGER PRIMARY KEY,
    name TEXT NOT NULL,
    score INTEGER
    );
    """
)
connection.commit()
```

task4.py x | game.py x | q.txt x

0..1..2

.....

.....

..@.....

.....