
scroll-shooter

Выпуск 1.0.0

Arthur Fedoruk, Mikhail Aldushonkov, Elizaveta Petukhova, Ale

дек. 15, 2022

Contents:

1	API Reference	1
1.1	classes	1
1.2	config	6
1.3	main	7
1.4	menu	10
1.5	setup	10
1.6	conf	10
2	Indices and tables	13
	Содержание модулей Python	15

This page contains auto-generated API reference documentation¹.

1.1 classes

1.1.1 Module Contents

Classes

<i>Bullet</i>	Класс-родитель пуль
<i>Ship</i>	Класс-родитель кораблей
<i>EnemyBullet</i>	Класс вражеских пуль
<i>EnemyShip</i>	Класс вражеских кораблей
<i>LaserBeam</i>	Класс, отвечающий за луч лазера
<i>AllyBullet</i>	Класс дружественных пуль
<i>AllyShip</i>	Класс дружеских кораблей (корабля?)
<i>LineEnemy</i>	Класс врагов, которые не стоят на месте, а двигаются по горизонтальной прямой
<i>CircleEnemy</i>	Класс врагов, которые не стоят на месте, а двигаются по окружности
<i>Asteroid</i>	Класс астероидов, летящих навстречу игроку
<i>Drop</i>	Класс того дропа ('плюшек'), который падает с
<i>Buff</i>	Класс баффа, которым может обладать корабль игрока
<i>Shield</i>	Класс баффа Shield
<i>Background</i>	Класс заднего фона игры
<i>AboutInfo</i>	Класс меню 'About'
<i>Boss</i>	Класс босса игры
<i>BossBullet</i>	Особые пули, которые выпускает босс

¹ Created with sphinx-autoapi

Functions

<i>game_over()</i>	Функция окончания игры когда игрок был сражен
--------------------	---

Attributes

<i>powerup_images</i>	powerup_images - Used skins (dictionary)
<i>music</i>	music - Used music (dictionary)
<i>drop_group</i>	drop_group - Group hierarchy
<i>laser_group</i>	
<i>enemy_bullet_group</i>	
<i>ally_bullet_group</i>	
<i>enemy_ship_group</i>	
<i>ally_ship_group</i>	
<i>asteroid_group</i>	
<i>groups</i>	
<i>keys_down</i>	keys_down = {"w": 0, "a": 0, "s": 0, "d": 0} - Словарь, используемый для перемещения игрока
<i>game_state</i>	

classes.powerup_images
 powerup_images - Used skins (dictionary)

classes.music
 music - Used music (dictionary)

classes.drop_group
 drop_group - Group hierarchy

classes.laser_group

classes.enemy_bullet_group

classes.ally_bullet_group

classes.enemy_ship_group

classes.ally_ship_group

classes.asteroid_group

classes.groups

```

classes.keys_down
    keys_down = {«w»: 0, «a»: 0, «s»: 0, «d»: 0} - Словарь, используемый для перемещения игрока
classes.game_state = startscreen

class classes.Bullet(x, y, picture_path, direction, enemy, damage=1, v=1)
    Bases: pygame.sprite.Sprite
    Класс-родитель пуль
    update()
        Изменяет положение пули. Удаляет из группы при вылезании за экран
class classes.Ship(x, y, picture_path, enemy, speed=1, lives=1)
    Bases: pygame.sprite.Sprite
    Класс-родитель кораблей
    update()
        Функция изменения состояния корабля
    move()
        Перемещает корабль (переписан в AllyShip, наследуется EnemyShip)
    hit()
        Проверка столкновения дружеского корабля с вражескими пулями, и удаление корабля при нулевом количестве жизней
    update_buffs()
        Функция обновления состояния баффов, переписана в AllyShip
class classes.EnemyBullet(x, y, direction, picture_path='images/enemy_bullet.png',
                        damage=ENEMY_BULLET_DAMAGE)
    Bases: Bullet
    Класс вражеских пуль
class classes.EnemyShip(picture_path='images/enemy_ship.png', lives=ENEMY_SHIP_LIVES)
    Bases: Ship
    Класс вражеских кораблей
    update()
        Функция изменения состояния корабля
    shoot()
        Корабль стреляет: создает EnemyBullet, вылетающую из корабля
    hit()
        Проверка столкновения вражеского корабля с дружественными пулями, и удаление корабля при нулевом количестве жизней. Начисление очков.
class classes.LaserBeam(x, y, picture_path='images/laser_beam.png', damage=LASER_DAMAGE)
    Bases: pygame.sprite.Sprite
    Класс, отвечающий за луч лазера
    update()
        Обновляет положение луча

```

```
class classes.AllyBullet(x, y, direction, picture_path='images/ally_bullet.png')
    Bases: Bullet
    Класс дружественных пуль

class classes.AllyShip(x=MAX_X / 2, y=MAX_Y * 3 / 4, picture_path='images/ally_ship.png')
    Bases: Ship
    Класс дружеских кораблей (корабля?)

    move()
        Функция перемещения корабля

    hit()
        Проверка столкновения дружеского корабля с вражескими пулями, и удаление корабля при
        нулевом количестве жизней, а также наложение эффекта баффа

    update_buffs()
        Функция, обновляющая состояние баффов игрока

    start_shooting()
        Начало стрельбы по нажатию кнопки мыши

    stop_shooting()
        Прекращение стрельбы по отпусканию кнопки мыши

    shooting()
        Собственно стрельба (зависит от SHOOTING_COEF)

    shoot()
        Корабль стреляет по-разному в зависимости от того, какое значение self.shooting_style

    normal_shot()
        Обычный выстрел

    double_shot()
        Двойной выстрел

    triple_shot()
        Тройной выстрел

classes.game_over()
    Функция окончания игры когда игрок был сражен

class classes.LineEnemy
    Bases: EnemyShip
    Класс врагов, которые не стоят на месте, а двигаются по горизонтальной прямой

    move()
        Функция перемещения

class classes.CircleEnemy
    Bases: EnemyShip
    Класс врагов, которые не стоят на месте, а двигаются по окружности

    move()
        Функция перемещения
```



```
class classes.Asteroid(picture_path='images/asteroid.png', group=asteroid_group)
    Bases: pygame.sprite.Sprite
    Класс астероидов, летящих навстречу игроку
    move()
        Функция движения астероида
    hit()
        Функция столкновения астероида с дружественными пулями
    update()
        Функция обновления состояния астероида
class classes.Drop(x, y, group=drop_group)
    Bases: pygame.sprite.Sprite
    Класс того дропа („плюшек“), который падает с поверженных врагов (с некоторой вероятностью)
    update()
        Обновление положения летящего дропа. Удаляет из группы при вылезании за экран
class classes.Buff(state)
    Класс баффа, которым может обладать корабль игрока
    default_state
        shield: «not applied». other states: «applied» shooting_style: «normal». other states: «double»,
        «triple», «laser» score_factor: «x1». other states: «x2» heal: «not applied». other states: «applied»
    update()
        Функция того, как „тикает“ таймер баффа
    apply(state, time)
        Функция присваивания баффу определенного состояния на определенное время
class classes.Shield(state, ship, picture_path='images/shield.png')
    Bases: Buff
    Класс баффа Shield
    update()
        Функция обновления состояния щита: обновление таймера и положения
class classes.Background(picture_path='images/back.png')
    Класс заднего фона игры
    update()
        Функция прокручивания фона
class classes.AboutInfo(picture_path='images/test_about.jpg')
    Класс меню „About“
    update()
        Обновление изображения
class classes.Boss(font)
    Bases: EnemyShip
    Класс босса игры
```

`move()`

Функция движения босса по восьмерке

`shoot()`

Функция стрельбы босса особыми пулями: класса `BossBullet`

`update()`

Функция обновления состояния босса

`health_bar()`

Полоска жизни босса

`class classes.BossBullet(x, y, direction, image, damage)`

Bases: *EnemyBullet*

Особые пули, которые выпускает босс

`update()`

Изменяет положение пули. Удаляет из группы при вылезании за экран

1.2 config

1.2.1 Module Contents

`config.MAX_X = 700`

`config.MAX_Y = 720`

`config.FPS = 60`

`config.BORDER_X = 36`

`config.BORDER_Y = 28`

`config.SPAWN_SECONDS = 5`

`config.SHOOTING_COEF = 10`

`config.BUFF_DURATION = 5`

`config.DROP_CHANCE = 1`

`config.INTENSITY = 30`

`config.BOSS_SCORE = 20`

`config.DEFAULT_SPEED`

`config.ALLY_LIVES = 5`

`config.BOSS_LIVES = 100`

`config.ENEMY_BULLET_DAMAGE = 1`

`config.LASER_DAMAGE = 0.15`

`config.ENEMY_SHIP_LIVES = 1`

`config.ASTEROID_LIVES = 5`

1.3 main

1.3.1 Module Contents

Functions

<i>initial_set()</i>	Создает игрока и изначальных врагов
<i>update()</i>	calls the update method of all objects
<i>restart_game()</i>	Перезапускает игру сначала
<i>draw()</i>	calls the draw method of all objects
<i>start_shooting()</i>	all allies start shooting
<i>stop_shooting()</i>	all allies stop shooting
<i>shooting()</i>	all allies shoot
<i>react_on_keys</i> (pygame_event)	processes a KEYUP/KEYDOWN event and updates keys_down
<i>react_on_menu_keys</i> (menu_event)	Реакция на нажатие кнопок меню
<i>spawn()</i>	Эта функция создаёт нам 4 типа врагов раз в FPS * SPAWN_SECONDS тиков
<i>textbar()</i>	Игровая информация на экране в правом верхнем углу
<i>buff_text()</i>	Функция, выводящая информацию о баффах
<i>boss_arrival()</i>	Предупреждает, что скоро будет мясо
<i>boss_is_here()</i>	Функция, вызывающая босса в конце игры - когда игрок набирает BOSS_SCORE (см. config)

Attributes

screen

background

about_image

ARIAL_18

ARIAL_25

ARIAL_45

BIG_OLD_FONT

VERY_BIG_OLD_FONT

spawn_timer

clock

finished

boss_here

ost_game

ost_boss

ost_menu

boss_timer

game_music

`main.screen`

`main.background`

`main.about_image`

`main.ARIAL_18`

`main.ARIAL_25`

`main.ARIAL_45`

`main.BIG_OLD_FONT`

`main.VERY_BIG_OLD_FONT`

`main.spawn_timer = 0`

```
main.clock
main.finished = False
main.boss_here = 0
main.ost_game = 0
main.ost_boss = 0
main.ost_menu = 0
main.boss_timer = 0
main.initial_set()
    Создает игрока и изначальных врагов
main.update()
    calls the update method of all objects
main.restart_game()
    Перезапускает игру сначала
main.draw()
    calls the draw method of all objects
main.start_shooting()
    all allies start shooting
main.stop_shooting()
    all allies stop shooting
main.shooting()
    all allies shoot
main.react_on_keys(pygame_event)
    processes a KEYUP/KEYDOWN event and updates keys_down function for wasd moving
main.react_on_menu_keys(menu_event)
    Реакция на нажатие кнопок меню Навигация по кнопкам - стрелки вверх/вниз Активация вы-
    бранного пункта меню - Enter Quit - завершает игру, если игрок находится на стартовом экране.
    В противном случае возвращает на стартовый экран
main.spawn()
    Эта функция создаёт нам 4 типа врагов раз в FPS * SPAWN_SECONDS тиков
main.textbar()
    Игровая информация на экране в правом верхнем углу healthbar - показывает жизни игрока
    scorebar - показывает набранные игроком очки
main.buff_text()
    Функция, выводящая информацию о баффах
main.boss_arrival()
    Предупреждает, что скоро будет мясо
main.boss_is_here()
    Функция, вызывающая босса в конце игры - когда игрок набирает BOSS_SCORE (см. config)
main.game_music
```

1.4 menu

1.4.1 Module Contents

Classes

<i>Menu</i>	Класс меню. Пока что кнопки только Start и Quit.
-------------	--

Attributes

<i>OLD_FONT</i>	Це шрифт
<i>menu_is_here</i>	

menu.OLD_FONT

Це шрифт

class menu.Menu

Класс меню. Пока что кнопки только Start и Quit. `_option_surfaces` - поверхности для пунктов меню `_callbacks` - что делают кнопки `_current_option_index` - выбранный пункт меню

`append_option(option, callback)`

Добавляет пункт меню `option` - отображаемый текст кнопки в меню `callback` - функция кнопки

`switch_menu_index(direction)`

Переключение между пунктами меню.

`activate_menu_option()`

Делаем «тык» в пункт меню

`check_current_index()`

`drawmenu(surf, x, y, option_y_padding)`

menu.menu_is_here

1.5 setup

1.6 conf

1.6.1 Module Contents

conf.project = scroll-shooter

conf.copyright = 2022, Arthur Fedoruk, Mikhail Aldushonkov, Elizaveta Petukhova, Aleksandr Gaborak

```
conf.author = Arthur Fedoruk, Mikhail Aldushonkov, Elizaveta Petukhova, Aleksandr Gaborak
conf.release = 1.0.0
conf.extensions = ['autoapi.extension']
conf.autoapi_dirs = ['C:/Users/alevg/PycharmProjects/scroll-shooter']
conf.templates_path = ['_templates']
conf.exclude_patterns = ['_build', 'Thumbs.db', '.DS_Store']
conf.language = ru
conf.html_theme = alabaster
conf.html_static_path = ['_static']
```

Indices and tables

- `genindex`
- `modindex`
- `search`

C

`classes`, 1
`conf`, 10
`config`, 6

m

`main`, 7
`menu`, 10

S

`setup`, 10