



Python

Чего мы добьемся сегодня?

- 1 Резюме материала
- 2 Знакомство с модулем Pygame
- 3 Работа с функциями и спрайтами
- 4 Написание космической игры

△
1 0 1
H 1 0 1
M 1 0 1
W 1 0 1
Th 1 0 1

△
1 0 1
1 0 1
1 0 1
1 0 1

△
1 0 1
1 0 1
1 0 1

△
1 0 1
1 0 1
1 0 1
1 0 1

△
1 0 1
1 0 1
1 0 1

△
1 0 1
1 0 1
1 0 1
1 0 1

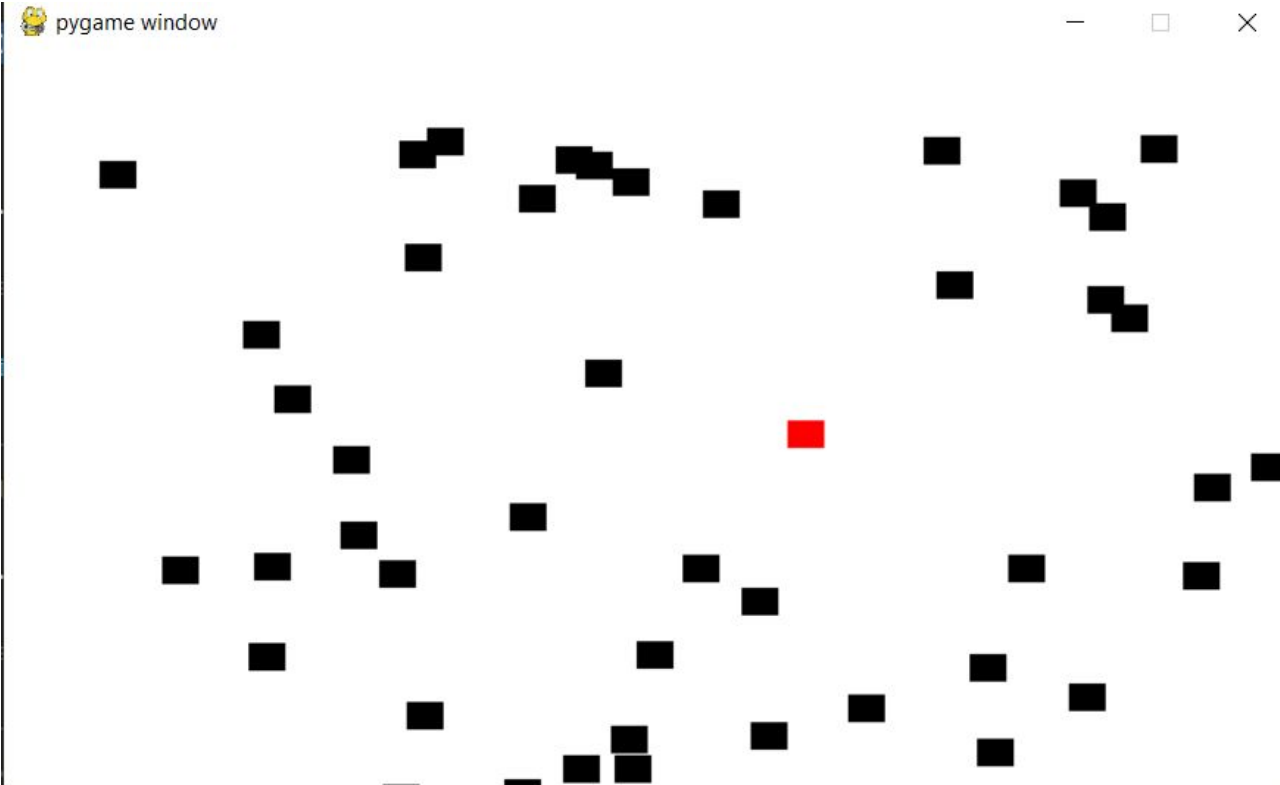
△
1 0 1
1 0 1
1 0 1

△
1 0 1
1 0 1
1 0 1
1 0 1

△
1 0 1
1 0 1
1 0 1

△
1 0 1
1 0 1
1 0 1
1 0 1

Чего мы добьемся сегодня?



1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

1.1.1
1.1.2
1.1.3
1.1.4
1.1.5

Подведем итоги:

widget

App

main loop

init



Установите Pygame

- В терминале пишем `pip install pygame`

```
(base) C:\Users\User\Desktop\python_lessons>pip install pygame
Collecting pygame
  Downloading pygame-2.0.1-cp38-cp38-win_amd64.whl (5.2 MB)
    |████████████████████████████████████████| 5.2 MB 3.2 MB/s
Installing collected packages: pygame
Successfully installed pygame-2.0.1

(base) C:\Users\User\Desktop\python_lessons>
```



Первая программа в Pygame

- Импортируем библиотеку **pygame** и инициализируем ее с помощью функции **init ()**
- Функция **init ()** дает нам доступ ко всем приложениям **pygame**

```
1 # Import and initialize the pygame library
2 import pygame
3 pygame.init()
4
```



Первая программа в Pygame

- Импортируем библиотеку **pygame** и инициализируем ее с помощью функции **init()** и устанавливаем размер окна

```
1 # Import and initialize the pygame library
2 import pygame
3 pygame.init()
4
5 # Set up the drawing window
6 screen = pygame.display.set_mode([500, 500])
7
8 # Run until the user asks to quit
9 running = True
10
```



ОСНОВНОЙ ЦИКЛ

- Если пользователь щелкает на закрыть окно, **running** будет False, и цикл остановится.

```
8 # Run until the user asks to quit
9 running = True
10 while running:
11
12     # Did the user click the window close button?
13     for event in pygame.event.get():
14         if event.type == pygame.QUIT:
15             running = False
16
```




ОСНОВНОЙ ЦИКЛ

- Создайте синий круг на белом экране

```
17     # Fill the background with white
18     screen.fill((255, 255, 255))
19
20     # Draw a solid blue circle in the center
21     pygame.draw.circle(screen, (0, 0, 255), (250, 250), 75)
22
23     # Flip the display
24     pygame.display.flip()
25
26     # Done! Time to quit.
27     pygame.quit()
```



Параметры функции Draw.circle

```
pygame.draw.circle(screen, (0, 0, 255), (250, 250), 75)
```

- параметр 1, где будет создан объект

```
pygame.draw.circle(screen, (0, 0, 255), (250, 250), 75)
```

- параметр 2, RGB-цвет объекта

```
pygame.draw.circle(screen, (0, 0, 255), (250, 250), 75)
```

- параметр 3, положение (x, y) объекта

```
pygame.draw.circle(screen, (0, 0, 255), (250, 250), 75)
```

- параметр 4, радиус объекта

Параметры функции

- Некоторые параметры на разных фигурах отличаются
- К прямоугольнику добавляется длина и ширина в параметре pos.

```
rect = pygame.draw.rect(screen, (255, 0, 0), (250, 250, 50, 50), 6)
```

- И последний параметр – толщина линии

```
rect = pygame.draw.rect(screen, (255, 0, 0), (250, 250, 50, 50), 6)
```



Движение предметов

- Добавляем переменные x и y

```
8 # Pos variables
9 x = 250
10 y = 250
```

- Мы добавляем переменные как параметры в функции

```
23
24 # Draw a solid blue circle in the center
25 pygame.draw.circle(screen, (0, 0, 255), (x, y), 75)
26
```





Ввод с клавиатуры

- Переменная `key_pressed` хранит, какая кнопка была нажата, а с помощью `if` мы это проверим и изменим `x` и `y`

```
21     key_pressed = pygame.key.get_pressed()
22
23     if key_pressed[pygame.K_d]:
24         x += 1
25     if key_pressed[pygame.K_a]:
26         x -= 1
27     if key_pressed[pygame.K_w]:
28         y -= 1
29     if key_pressed[pygame.K_s]:
30         y += 1
31
```



Пределы передвижения

- Не позволяйте персонажу выходить за пределы экрана

```
32         # Limits of move
33         if x > 480:
34             x = 480
35         if x < 20:
36             x = 20
37         if y > 480:
38             y = 480
39         if y < 20:
40             y = 20
```



Вывод текста

- Добавляем переменные font, score и life

```
12 # Variables for text
13 font = pygame.font.SysFont("comicansms", 35)
14 score = 0
15 lives = 3
```

- Добавляем текст на экран

```
50 # Add text to screen
51 score_b = font.render("Score " + str(score), True, (0, 0, 255))
52 screen.blit(score_b, [0, 0])
53 live_b = font.render("Live " + str(lives), True, (255, 0, 0))
54 screen.blit(live_b, [0, 40])
```




Параметры функции

- Функция **render ()** создает текст в соответствии со шрифтом и назначает ему цвет.
- Функция **flash ()** отображает визуальные объекты на экране в заданном положении.

```
47 # Fill the background with white
48 screen.fill((255, 255, 255))
49
50 # Add text to screen
51 score_b = font.render("Score " + str(score), True, (0, 0, 255))
52 screen.blit(score_b, [0, 0])
53 live_b = font.render("Live " + str(lives), True, (255, 0, 0))
54 screen.blit(live_b, [0, 40])
55
```




Спрайты

- С точки зрения программирования, спрайт – это 2D-представление чего-либо на экране.
- По сути, это изображение ругаме, которое предоставляет класс `Sprite`, который предназначен для хранения одного или нескольких графических представлений любого игрового объекта, который вы хотите отобразить на экране.
- Чтобы использовать его, создайте новый класс, наследующий `Sprite`.
- Это позволяет использовать его встроенные методы.

Спрайт и изображения

- Вы можете загружать изображения с компьютера того же размера с Surface ()

```
17 player_img = pygame.image.load(r'C:/Users/User/Desktop/ufo.png')
18 class Player(pygame.sprite.Sprite):
19
20     def __init__(self):
21         pygame.sprite.Sprite.__init__(self)
22         self.image = pygame.Surface((50,50))
23         self.image = player_img
24         self.rect = self.image.get_rect()
```



Группа спрайтов

- Создаем переменную для всех спрайтов
- Мы создаем игрока
- Добавляем игрока в группу спрайтов

```
26 all_sprites = pygame.sprite.Group()
27 player = Player()
28 all_sprites.add(player)
29
```



Группа спрайтов

- Движения персонажа, а точнее прямоугольник вокруг персонажа

```
51 #Player move
52 if key_pressed[pygame.K_d]:
53     player.rect.x += 1
54 if key_pressed[pygame.K_a]:
55     player.rect.x -= 1
56 if key_pressed[pygame.K_w]:
57     player.rect.y -= 1
58 if key_pressed[pygame.K_s]:
59     player.rect.y += 1
60
```

Добавление спрайта на экран

- Добавляем группу спрайтов на экран через функцию `.draw` с указанием экрана
- Примечание: будьте осторожны, любое добавление должно быть между `screen.fill ()` и `pygame.display.flip ()`

```
32 while running:  
33     all_sprites.update()
```

```
82  
83     # Draw the player on screen  
84     all_sprites.draw(screen)  
85
```

Космическая игра

<fun+learning/empowering>(shape future)

<WE CREATE>/EDUCATE/(INSPIRE)NOW.

<fun+learning/empowering>(shape future)

{we generate experience}* <explore+invent>

<WE CREATE>/EDUCATE/(INSPIRE)NOW.

<fun+learning/empowering>(shape future)

<WE CREATE>/EDUCATE/(INSPIRE)NOW.

<fun+learning/empowering>(shape future)

{we generate experience}* <explore+invent>

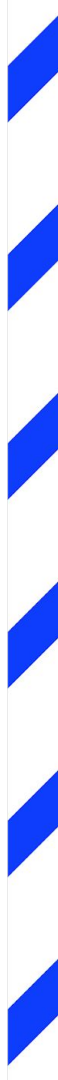
<WE CREATE>/EDUCATE/(INSPIRE)NOW.



Game.py

- Импортируем модули и определяем цвета

```
1 #Import pygame and random
2 import pygame
3 import random
4
5 # Define some colors
6 BLACK = (0, 0, 0)
7 WHITE = (255, 255, 255)
8 GREEN = (0, 255, 0)
9 RED = (255, 0, 0)
```



Класс блока

```
12 class Block(pygame.sprite.Sprite):
13
14     def __init__(self, color, width, height):
15         # Call the parent class (Sprite) constructor
16         super().__init__()
17
18         # Create an image of the block, and fill it with a color.
19         # This could also be an image loaded from the disk.
20         self.image = pygame.Surface([width, height])
21         self.image.fill(color)
22
23         # Fetch the rectangle object that has the dimensions of the image
24         # image.
25         # Update the position of this object by setting the values
26         # of rect.x and rect.y
27         self.rect = self.image.get_rect()
```


Функции блочного класса

```
29     def reset_pos(self):
30         """ Reset position to the top of the screen, at a random x location.
31             Called by update() or the main program loop if there is a collision.
32         """
33         self.rect.y = random.randrange(-300, -20)
34         self.rect.x = random.randrange(0, screen_width)
35
36     def update(self):
37         """ Called each frame. """
38
39         # Move block down one pixel
40         self.rect.y += 1
41
42         # If block is too far down, reset to top of screen.
43         if self.rect.y > 410:
44             self.reset_pos()
```



Класс игрока

- **Player Class** Наследует класс **Block** и переписывает функцию **update**.

```
47 class Player(Block):
48     """ The player class derives from Block, but overrides the 'update'
49     functionality with new a movement function that will move the block
50     with the mouse. """
51
52     def update(self):
53         # Get the current mouse position. This returns the position
54         # as a list of two numbers.
55         pos = pygame.mouse.get_pos()
56
57         # Set the player object to the mouse location
58         self.rect.x = pos[0]
59         self.rect.y = pos[1]
60
```

Основные данные

```
62 # Initialize Pygame
63 pygame.init()
64
65 # Set the height and width of the screen
66 screen_width = 700
67 screen_height = 400
68 screen = pygame.display.set_mode([screen_width, screen_height])
69
70 # This is a list of 'sprites.' Each block in the program is
71 # added to this list. The list is managed by a class called 'Group.'
72 block_list = pygame.sprite.Group()
73
74 # This is a list of every sprite. All blocks and the player block as well.
75 all_sprites_list = pygame.sprite.Group()
76
```

Создание 50 блоков

```
77 for i in range(50):
78     # This represents a block
79     block = Block(BLACK, 20, 15)
80
81     # Set a random location for the block
82     block.rect.x = random.randrange(screen_width)
83     block.rect.y = random.randrange(screen_height)
84
85     # Add the block to the list of objects
86     block_list.add(block)
87     all_sprites_list.add(block)
88
```



Создание игрока

```
89 # Create a red player block
90 player = Player(RED, 20, 15)
91 all_sprites_list.add(player)
92
93 # Loop until the user clicks the close button.
94 done = False
95
96 # Used to manage how fast the screen updates
97 clock = pygame.time.Clock()
98
99 score = 0
100
```



ОСНОВНОЙ ЦИКЛ

```

101     # ----- Main Program Loop -----
102     while not done:
103         for event in pygame.event.get():
104             if event.type == pygame.QUIT:
105                 done = True
106
107         # Clear the screen
108         screen.fill(WHITE)
109
110         # Calls update() method on every sprite in the list
111         all_sprites_list.update()
112

```





Коллизия

- Функция `.spritecollide ()` проверяет коллизии между спрайтами.
- Коллизия – это когда два объекта сталкиваются или перекрываются

```

113         # See if the player block has collided with anything.
114         blocks_hit_list = pygame.sprite.spritecollide(player, block_list, False)
115
116         # Check the list of collisions.
117         for block in blocks_hit_list:
118             score += 1
119             print(score)
120             # Reset block to the top of the screen to fall again.
121             block.reset_pos()
122

```




Коллизия

- Объявляем скорость игры и «рисует» спрайты на экране

```
123         # Draw all the spites
124         all_sprites_list.draw(screen)
125
126         # Limit to 20 frames per second
127         clock.tick(20)
128
129         # Go ahead and update the screen with what we've drawn.
130         pygame.display.flip()
131
132     pygame.quit()
```


Что мы сделали сегодня?

- 1 Резюме материала
- 2 Знакомство с модулем Pygame
- 3 Работа с функциями и спрайтами
- 4 Написание космической игры



<Thank you! />



impact
ACADE-
MIES &
CAMPS

