Software
Engineering
School

Курсы программирования для
взрослых и детей

Делаем игры на Pygame

Чем мы сегодня займемся

- Рассмотрим, как продвинутые программисты реализуют игровой цикл;
- Узнаем, как обрабатывать клики мышкой и нажатия на клавиатуру;
- Сделаем кликер

Но сначала вопросы!

- Что такое игровой цикл?
- Из каких основных действий он состоит?
- Какие команды Pygame используются для рисования прямоугольника? А круга?

Продвинутый игровой цикл

Более удобной реализацией игрового цикла вместо набора процедур служит класс:

```
class Game():
    def __init__(self):
        # Задание начальных настроек игры...

    def process_input(self):
        # Обработка ввода пользователя...

    def update_game_state(self):
        # Обновление состояния игры...

    def render(self):
        # Отрисовка игры на экран...

    def main_loop(self):
        # Игровой цикл...
```

Игровой цикл

Зачем нам использовать класс?

- Удобство: функциям (методам) не нужно задавать много аргументов – методы класса имеют доступ к его данным, поэтому у всех методов есть одни и те же игровые данные без явного указания;
- Определяя класс, мы явно говорим, что такие-то функции относятся только к игре и обработке её данных – это повышает читаемость кода и обеспечивает более естественную структуру программы

Практика: двигающаяся машина

Игровой цикл: начальные настройки

```
class Game():

    def __init__(self):
        pygame.init()

        self.WINDOW_WIDTH, self.WINDOW_HEIGHT = 800, 800
        self.FPS = 60
        self.main_window = pygame.display.set_mode((self.WINDOW_WIDTH,
                                                     self.WINDOW_HEIGHT))
        self.clock = pygame.time.Clock()
        self.running = True

        self.x_car = 40
        self.y_car = 400
```

Игровой цикл: обработка ввода и обновление состояния

```
class Game():

    ...

    def process_input(self):
        pass

    def update_game_state(self):
        pass
```

Пока они будут пустые...

Игровой цикл: отрисовка машины

```
class Game():

    ...

    def render(self):
        main_window_color = pygame.color.THECOLORS["white"]
        self.main_window.fill(main_window_color)

        self.draw_car(self.x_car, self.y_car)

        pygame.display.update()
```

Игровой цикл: собираем все вместе

```
class Game():

    ...

    def main_loop(self):
        while self.running:
            self.process_input()
            self.update_game_state()
            self.render()
            self.clock.tick(self.FPS)

        pygame.quit()
```

Рисуем машину: кузов

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        # Отрисовка кузова
        body_width = 240
        body_height = 100
        x_body = x_body_start
        y_body = y_body_start
        body_color = pygame.color.THECOLORS["orange"]
        body = pygame.Rect(x_body, y_body, body_width, body_height)
        pygame.draw.rect(self.main_window, color=body_color, rect=body)
```

Рисуем машину: кузов

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        ...

        # Отрисовка кабины
        cabin_width = 100
        cabin_height = 130
        x_cabin = x_body + body_width
        y_cabin = y_body - (cabin_height - body_height)
        cabin_color = pygame.color.THECOLORS["orange"]
        cabin = pygame.Rect(x_cabin, y_cabin, cabin_width, cabin_height)
        pygame.draw.rect(self.main_window, color=cabin_color, rect=cabin)
```

Рисуем машину: окно в кабине

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        ...

        # Отрисовка окна в кабине
        window_width = 60
        window_height = 50
        x_offset_cabin = 20
        y_offset_cabin = 10
        x_window = x_cabin + x_offset_cabin
        y_window = y_cabin + y_offset_cabin
        window_color = pygame.color.THECOLORS["skyblue"]
        window = pygame.Rect(x_window, y_window, window_width, window_height)
        pygame.draw.rect(self.main_window, color=window_color, rect=window)
```

Рисуем машину: окно в кабине

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        ...

        # Отрисовка окна в кабине
        window_width = 60
        window_height = 50
        x_offset_cabin = 20
        y_offset_cabin = 10
        x_window = x_cabin + x_offset_cabin
        y_window = y_cabin + y_offset_cabin
        window_color = pygame.color.THECOLORS["skyblue"]
        window = pygame.Rect(x_window, y_window, window_width, window_height)
        pygame.draw.rect(self.main_window, color=window_color, rect=window)
```

Рисуем машину: окно в кабине

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        ...

        # Отрисовка контура окна
        window_contour_color = pygame.color.THECOLORS["black"]
        window_contour_width = 2
        pygame.draw.rect(self.main_window,
                         color=window_contour_color,
                         rect=window,
                         width=window_contour_width)
```

Рисуем машину: левое колесо

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        ...

        # Настройки для отрисовки сердцевины колеса
        wheel_center_color = pygame.color.THECOLORS["black"]
        wheel_center_radius = 5

        # Отрисовка колеса у кузова
        left_wheel_radius = 25
        x_offset_body = 40
        y_offset_body = 90
        x_left_wheel = x_body + x_offset_body
        y_left_wheel = y_body + y_offset_body
        left_wheel_color = pygame.color.THECOLORS["grey"]
```

Рисуем машину: левое колесо

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        ...

        pygame.draw.circle(
            self.main_window,
            color=left_wheel_color,
            center=(x_left_wheel, y_left_wheel),
            radius=left_wheel_radius
        )
        pygame.draw.circle(
            self.main_window,
            color=wheel_center_color,
            center=(x_left_wheel, y_left_wheel),
            radius=wheel_center_radius
        )
```

Рисуем машину: правое колесо

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        ...

        # Отрисовка колеса у кабины
        right_wheel_radius = 25
        x_offset_cabin = 60
        y_offset_cabin = 120
        x_right_wheel = x_cabin + x_offset_cabin
        y_right_wheel = y_cabin + y_offset_cabin
        right_wheel_color = pygame.color.THECOLORS["grey"]
```

Рисуем машину: правое колесо

```
class Game():

    ...

    def draw_car(self, x_start, y_start):
        ...

        # Отрисовка колеса у кабины
        pygame.draw.circle(
            self.main_window,
            color=right_wheel_color,
            center=(x_right_wheel, y_right_wheel),
            radius=right_wheel_radius
        )
        pygame.draw.circle(
            self.main_window,
            color=wheel_center_color,
            center=(x_right_wheel, y_right_wheel),
            radius=wheel_center_radius
        )
```

Как заставить машину двигаться?

Для того, чтобы машина начала двигаться, необходимо реагировать на события от пользователя – ввод с клавиатуры и нажатие мышки, для этого в Pygame предусмотрен объект **event**, хранящий все события, произошедшие между кадрами игрового цикла

```
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        running = False
```

События клавиатуры

Чтобы проверить тип события, нужно узнать значение свойства **type**, а чтобы проверить какая клавиша была нажата – свойство **key** (для стрелок клавиш – **K_UP** - вверх, **K_DOWN** - вниз, **K_LEFT** - влево, **K_RIGHT** - вправо)

```
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        running = False
    elif event.type == pygame.KEYDOWN:
        if event.key == pygame.K_UP:
            move_up()
```

Если тип события «нажата клавиша»…
…и если это была клавиша «вверх»
…сдвинуться вверх

Добавляем обработку ввода

```
class Game():

    def __init__(self):
        ...

        self.x_move = 0
        self.y_move = 0
```

Добавляем обработку ввода

```
class Game():

    ...

    def process_input(self):
        self.x_move = 0
        self.y_move = 0

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                running = False
            elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_RIGHT:
                    self.x_move += 8
                elif event.key == pygame.K_LEFT:
                    self.x_move -= 8
```

Добавляем обновление состояния игры

```
class Game():

    ...

    def update_game_state(self):
        self.x_car += self.x_move
        self.y_car += self.y_move
```

Домашнее задание

Игра, в которой мы управляем роботом, который умеет перемещаться влево, вправо и прыгать. Во время прыжка робот перемещается на некоторое расстояние вверх, после чего спускается на прежнее положение. Во время прыжка робот ничего не может сделать, сам прыжок длится некоторое время (несколько секунд).

Подсказка: чтобы сделать замедленный эффект прыжка, воспользуйтесь pygame.time.wait(1000) – функция wait принимает кол-во миллисекунд для ожидания; чтобы понять, что мы в «состоянии прыжка» стоит ввести логический флаг is_jumping, указывающий на это...

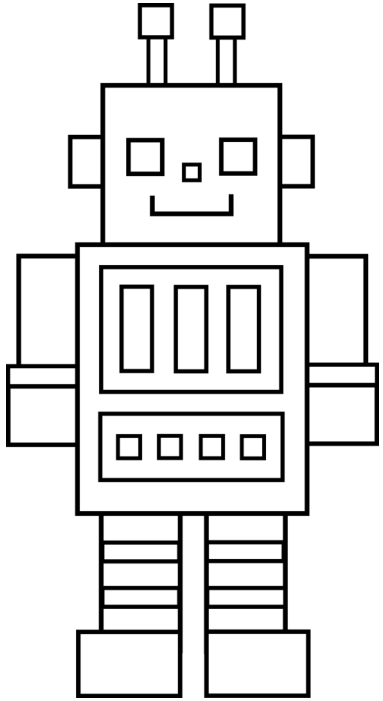

Рисунок робота примерный!!! Можно без сильной детализации

Наш репозиторий

https://github.com/samedit66/pygame_2024/tree/main