PYGAME.

ЦЕЛЬ: **Научиться программировать гравитацию и прыжок**

ПРИМЕЧАНИЕ:

ПЛАНИРОВАНИЕ

1. Гравитация

У каждого объекта в игре есть скорость, а с точки зрения физики, скорость можно разбить на отдельные скорости по осям координат. У нашего спрайта, получается вместо одной скорости – две: по оси **x** и по оси **y**.

А гравитация создает ускорение, то есть заставляет скорость за каждую единицу времени увеличиваться на определенное значение (это значение и есть ускорение).

Создадим в классе Sprite два новых свойства:

speedx

speedy

И тогда, чтобы создать гравитацию, нужно добавить в программу команду:

Player.speedy += g

g – ускорение свободного падения

Примечание: в python **x += 1** означает: **x = x + 1**

За исключением некоторых случаев, это когда наш герой на чем-либо стоит, иначе говоря, когда функция collide равна True:

```
if not collide(Player, Ground):
    Player.speedy += g
```

Давайте создадим спрайт – кусочек земли или какая-нибудь платформа, на которой наш герой сможет стоять.

Ground = Sprite(80, 400, 0, 'Images/ground.png')

Установим ее под героем, так, чтобы он начал падать и приземлился прямо на платформу.

! Проблемная задача: Создать класс Константы и создать объект – экземпляр этого класса, константу: g – ускорение свободного падения.

Результат:

```
class Const():
  def __init__(self, value):
    self.value = value
```

Создадим константу – **g** :

g = Const(1)

Наша гравитация равна (1 пиксел в секунду) за секунду.

и напишем код, который будет отвечать за гравитацию.

```
while game:
    clock.tick(24)
    for ev in pygame.event.get ():
        if ev.type == pygame.QUIT:
            game = False
    keys = pygame.key.get_pressed()
    if keys[pygame.K_RIGHT]:
        Player.rect.x += Player.speed
    elif keys[pygame.K_LEFT]:
        Player.rect.x -= Player.speed
    # gravitation
    if not collide(Player.rect, Ground.rect):
        Player.speedy += g.value
    else:
        Player.speedy = 0
    Player.rect.y += Player.speedy


    w.fill((0, 0, 0))
    w.blit(ImgPlayerGo[Ncadr % 5], Player.rect)
    w.blit(Ground.image, Ground.rect)
    pygame.display.update()
    Ncadr += 1
```

Теперь наш герой приземляется на платформу и если с нее уйти, то он падает.

2. ПРЫЖОК

Если смотреть с точки зрения механики, которую мы прописали выше – как работает гравитация, то, что она влияет на вертикальную скорость спрайта, то прыжок – это резкое изменение вертикальной скорости.

Иначе говоря, когда пользователь будет нажимать клавишу пробел, мы должны у спрайта просто изменить скорость в сторону верха нашей игры:

Player.speedy = -10

А лучше добавить в класс Спрайт свойство – jumppower, скорость которую задает прыжок или «сила прыжка», тогда команда будет выглядеть так:

Player.speedy = Player.jumppower

Вы спросите почему отрицательное число, ответ: потому что ось игрек направлена вниз, получается, что прыжок задает спрайту скорость в противоположном направлению оси – **y**.

А все остальное сделает код, который мы написали выше – гравитация

```
if keys[pygame.K_SPACE]:   # проверяем нажатие клавиши пробел
    Player.speedy = Player.jumppower  # задаем скорость по игрек


if not collide(Player, Ground):
    Player.speedy += g.value
else:
    Player.speedy = 0
```

Если запустить программу, то спрайт не будет прыгать…

! Проблемная задача: исправить программу так, чтобы спрайт прыгал.

Решение:

Дело в том, что в момент, когда спрайт стоит на платформе, он ее касается, поэтому функция

collide(Player, Ground) возвращает значение **True,** и выполняется команда:

Player.speedy = 0

В реальности, если предмет задевает опору и его скорость направлена вверх, то он будет двигаться вверх, а вот если скорость направлена вниз и предмет касается опоры, то он останавливается.

```
if keys[pygame.K_SPACE]:
        if Player.speedy == 0: # прыгаем только если скорость равна нулю, как в реальности
            Player.speedy = Player.jumppower


    if not collide(Player, Ground):
        Player.speedy += g.value


    elif Player.speedy > 0:  # если задеваем платформу и скорость направлена вниз, останавливаемся
        Player.speedy = 0
```

! Проблемная задача: создать еще несколько платформ на разной высоте, чтобы можно было по ним прыгать.

! Создать анимацию прыжка

3. Рефлексия

- Сегодня мы научились программировать гравитацию и прыжок.