PYGAME.

ЦЕЛЬ: **создать группу объектов. Объект – это огненный шар, который падает на игрока через случайное время. Необходимо использовать принцип ООП.**

ПРИМЕЧАНИЕ:

ПЛАНИРОВАНИЕ

1. Класс FireBall

Добавим в нашу игру какие-нибудь препятствия нашему персонажу, сделаем их динамическими – например, пускай это будут огненные шары, падающие с неба.

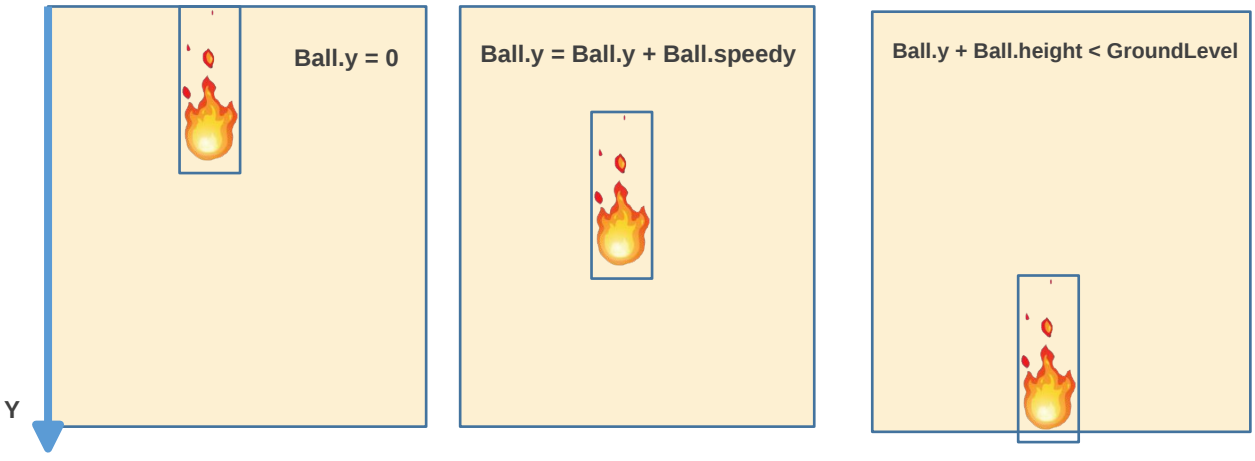
Вопрос ученикам? С чего мы должны начать, чтобы в нашей игре появились огненные шары?

Нам понадобиться для огненных шаров создать класс, который будет описывать наши огненные шары или, точнее говоря, класс который будет описывать огненный шар.

Разумно сделать его наследником класса Sprite, чтобы не прописывать базовые свойства, координаты, изображение и т.д.

Class FireBall(Sprite)

Прежде чем прописывать конструктор и метод update() этого класса, нарисуем схему на координатной плоскости, как будут двигаться огненные шары по координатам:



Как и с другими спрайтами, на огненный шар будет действовать гравитация. Но после того, как координата – **y** будет больше нижнего уровня экрана, будем ее изменять снова на ноль, таким образом возвращая спрайт шара вверх, чтобы он снова начал падать. В итоге будет создаваться впечатление, что с верху падают огненные шары.

Напишем конструктор:

```
class Fireball(Sprite):
    def __init__(self, img, group):
        Sprite.__init__(self, 0, 0, img)
        self.add(group)
```

Сразу будем прописывать добавление в ранее созданную группу для огненных шаров. Это будет эффективно. Будем в программе вызывать лишь метод update() у группы и этот метод выполниться у всех шаров в группе.

Новые команды:

Может возникнуть ситуация, когда изображение спрайта в вашем файле не соответствует нужной ориентации или масштабу. Вот например изображение огненного шара предназначенные для этого урока смотрят вправо, шар как бы летит в правую сторону, а в игре необходимо, чтобы он был повернут вниз, ведь он падает сверху. И само изображение в файле слишком большое, в игре размер шара должен быть меньше.

В **pygame** для решения таких проблем есть много методов, мы рассмотрим два из них:

1) **pygame.transform.scale(self.image, (180, 65))**
   Он возвращает новый объект Image, с новым масштабом, а точнее с новым разрешением в пикселах. В скобках мы задаем необходимый для игры размер в пикселах и pygame изменяет масштаб – подгоняет изображение под эти размеры.
2) **pygame.transform.rotate(self.image, -90)**
   Несложно догадаться, как работает данный метод – он поворачивает изображение на заданный угол, если угол положительный – то по часовой стрелке, если отрицательный, то против часовой стрелки.

Добавим их в конструктор:

```
class Fireball(Sprite):
    def __init__(self, img, group):
        Sprite.__init__(self, 0, 0, img)
        self.add(group)
        self.image = pygame.transform.scale(self.image, (180, 65))
        self.image = pygame.transform.rotate(self.image, -90)
```

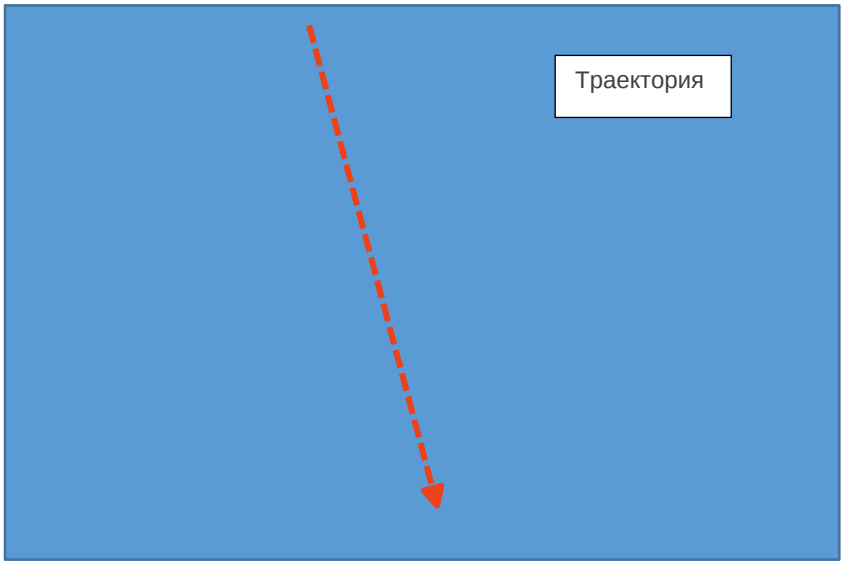
Теперь мы можем создать метод update(), который будет вызываться каждое повторение (итерацию) игрового цикла и обновлять значения координат спрайта. Иначе говоря, будет обновлять его состояние в игре.

```
class Fireball(Sprite):
    def update (self, g):
        if self.rect.y < 700:
            self.rect.y += self.speedy
            self.speedy += g.value        # гравитация
        else:
            self.rect.y = 0
            self.rect.x = random.randint(0, 1200)
```

Как видно, каждый раз, когда спрайт будет ниже нижней границы экрана, его положение по вертикали – координата **y (self.rect.y)** будет устанавливаться в значение **ноль.** И огненный шар снова начнет падать с верху. Также мы сделали так, чтобы шар каждый раз появлялся в случайном положении по горизонтали – то есть по оси **x.**

self.rect.x = random.randint(0, 1200)

! Проблемная задача. Сделать так, чтобы огненный шар летел немного под углом, то есть чуть-чуть наискосок:



А также, чтобы этот угол, каждый раз был случайным.

Сделать начальную скорость шара каждый раз разную, случайной.

2. Анимация огненных шаров

! Проблемная задача: сделать анимацию огненным шарам.

Решение:

```
class Fireball(Sprite):
    def __init__(self, img, group):
        Sprite.__init__(self, 0, 0, 'Images/fireball.png')
        self.add(group)
        #self.visible = False
        self.speedx = random.randint(-5, 5)
        self.animimg = img
        self.cadr = 0
    def update (self, g):
        if self.rect.y < 700:
            self.rect.y += self.speedy
            self.rect.x += self.speedx
            self.speedy += g.value
        else:
            self.rect.y = 0
            self.rect.x = random.randint(0, 1200)
            self.speedx = random.randint(-5, 5)
            self.speedy = random.randint(0, 30)
        self.image = self.animimg[self.cadr // 2 % 4]
        self.cadr += 1
```

```
# анимация огенных шаров
ImgFireBall = [pygame.image.load('Images/fireball.png'),
               pygame.image.load('Images/fireball1.png'),
               pygame.image.load('Images/fireball2.png'),
               pygame.image.load('Images/fireball3.png'),]
i = 0
for img in ImgFireBall:
    ImgFireBall[i] = pygame.transform.rotate(pygame.transform.scale(img, (180, 65)), -90)
    i += 1
```

3. Рефлексия

- Сегодня мы создали препятствия в игре

- создали новый класс

- сделали анимацию

- узнали новые методы