GeekSchool

Программирование на Python

Урок 9

Новая игра и ООП

Немного повторим прошлый урок

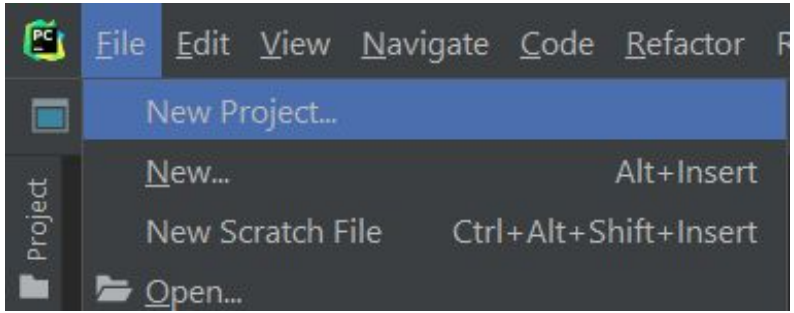
Что будет на уроке сегодня?

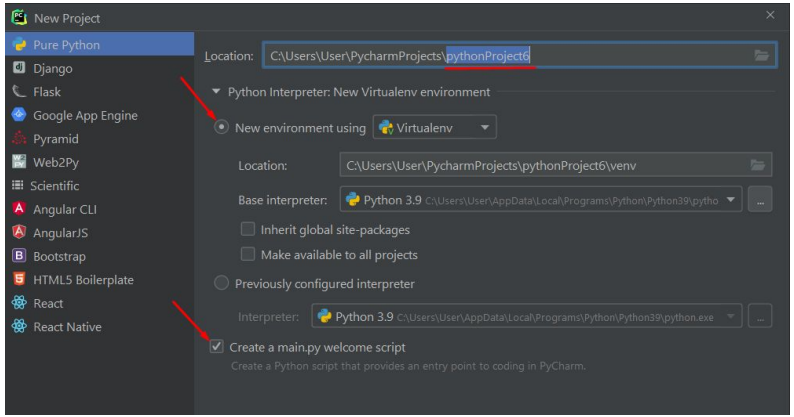
- Познакомимся с понятием ООП
- Создадим многофайловый проект
- Научимся объединять код из нескольких файлов
- Создадим классы игрока, мобов
  и всех остальных объектов в игре
- Добавим логику движения игрока

Создаем
и настраиваем
новый проект

Создаем новый проект

Для новой игры нам потребуется новый проект. Создаем его выбрав пункт File — New Project. Убеждаемся, что выбраны все необходимые пункты. Можно также переименовать его любым именем.

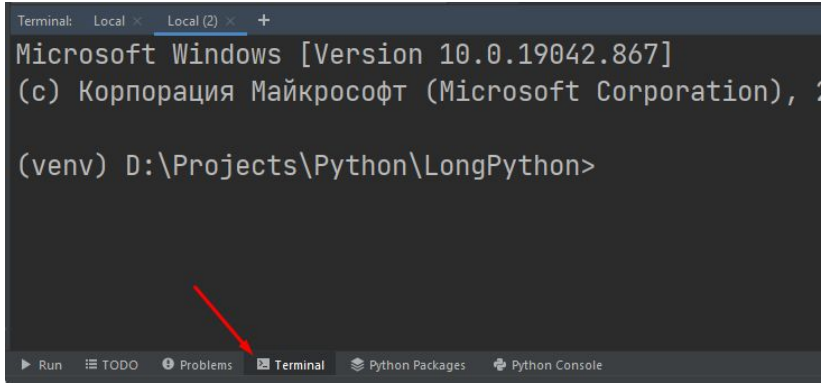


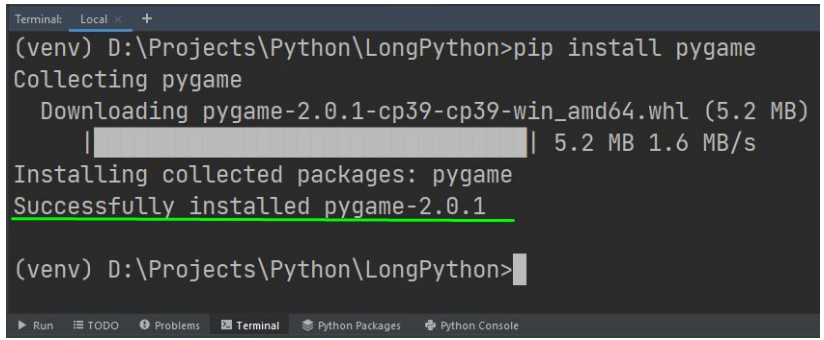


Создаем новый проект

После создания проекта, необходимо сразу установить модуль pygame Точно таким же способом как и в предыдущем проекте

```
pip install pygame
```





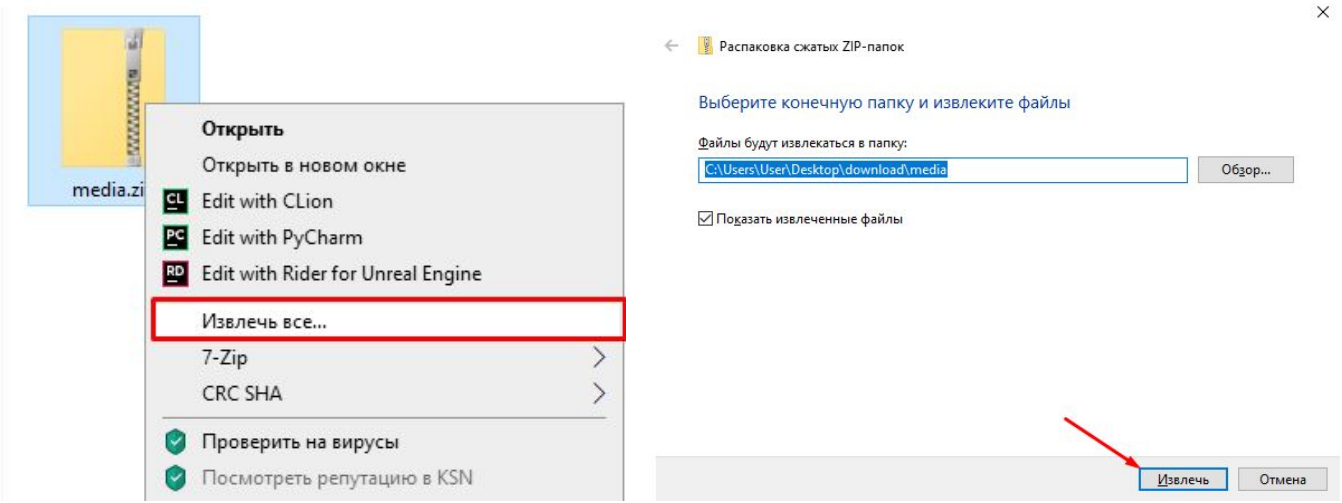
Создаем новый проект

Дальше внутри файла main.py удаляем весь стандартный код и вставляем на его место код из шаблона. Скопировать его можно по ссылке ниже: https://pastebin.com/xmwS5car

Добавляем ресурсы в проект

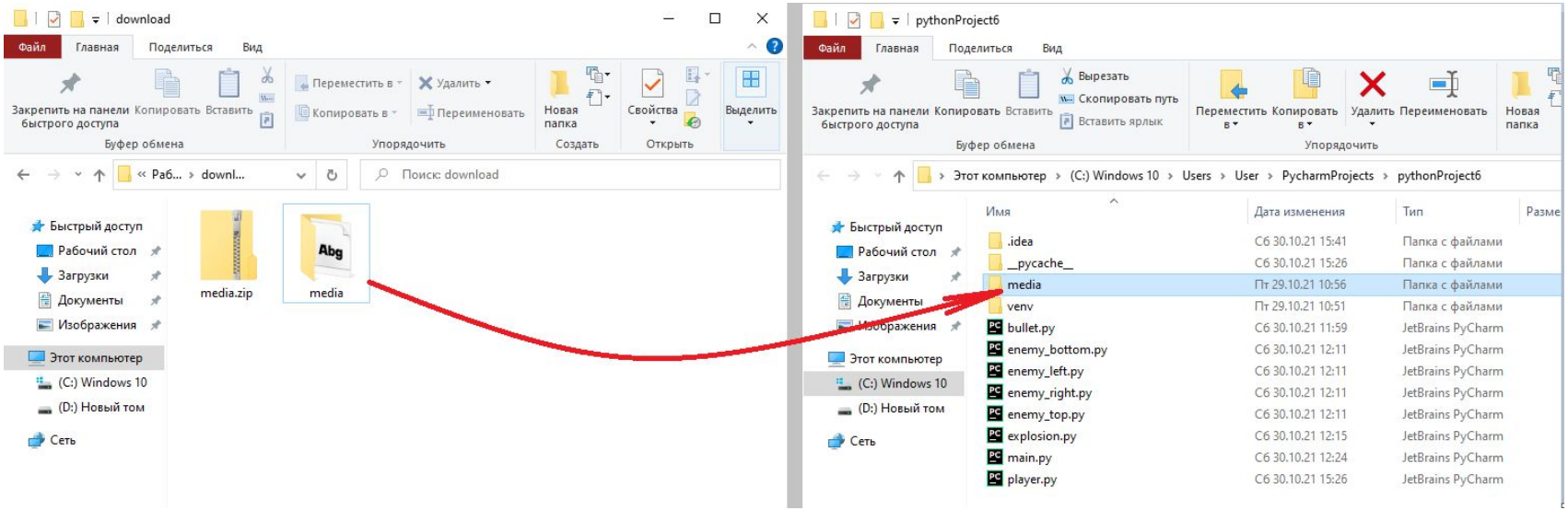
Необходимо скачать архив с ресурсами по ссылке:
https://drive.google.com/file/d/13Q9y8TyVUAVkAdK69BHXHP4rbX7BzXPp

Распаковываем наш архив:



Добавляем ресурсы в проект

Переносим распакованную папку в проект с игрой



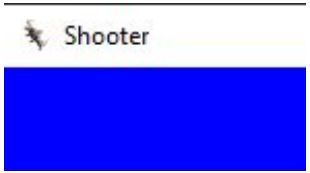
Добавляем ресурсы в проект

Теперь необходимо создать переменные, в которых будут храниться данные пути.
Создаем их до игрового цикла:

```
snd_dir = 'media/snd/'             # Путь до папки со звуками
img_dir = 'media/img/'             # Путь до папки со спрайтами
```

Протестируем наши пути, изменив иконку игры:

```
icon = pygame.image.load(img_dir + 'icon.png')        # Загружаем файл с иконкой
pygame.display.set_icon(icon)                         # Устанавливаем иконку в окно
```



Создаем группу спрайтов

Спрайтов в игре может быть очень много, то управлять ими со временем становится не удобно. Чтобы решить эту проблему, разработчики решили объединить все спрайты в одну группу. И давать команды не каждому спрайту, а сразу целым группам спрайтов. Создадим общую группу спрайтов. Сделать это нужно до игрового цикла:

```
all_sprites = pygame.sprite.Group()                    # Создаем группу для спрайтов
```

Затем запустим выполнение действий у всех спрайтов:

```
while run:                          # Начинаем бесконечный цикл
    timer.tick(fps)                     # Контроль времени (обновление игры)
    all_sprites.update()                # Выполняем действия всех спрайтов в группе
```

Создаем группу спрайтов

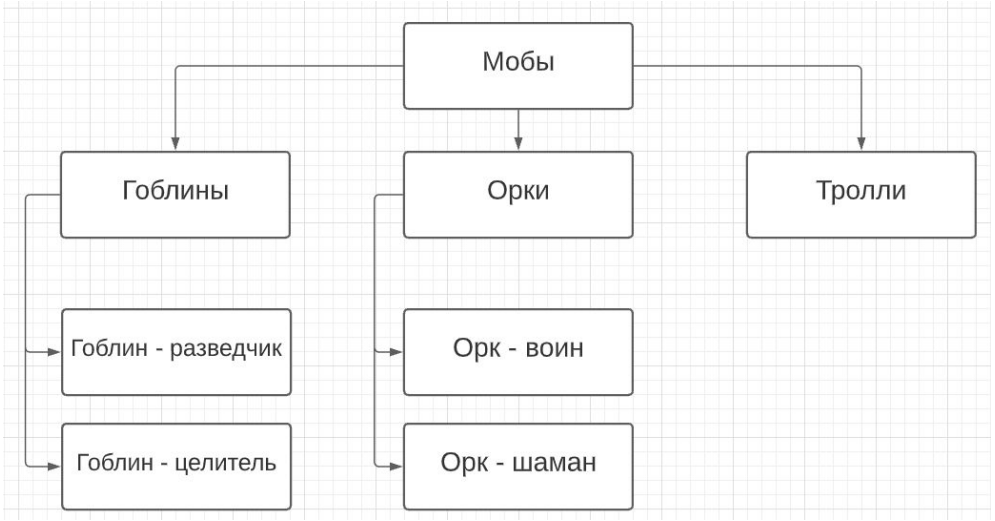
Затем отрисуем сразу все спрайты в группе. Добавим команду после заливки экрана:

```
    screen.fill(CYAN)                    # Заливка заднего фона
    all_sprites.draw(screen)             # Отрисовываем все спрайты
    pygame.display.update()              # Обновляем экран
```

Добавляем класс игрока

Знакомство с классами

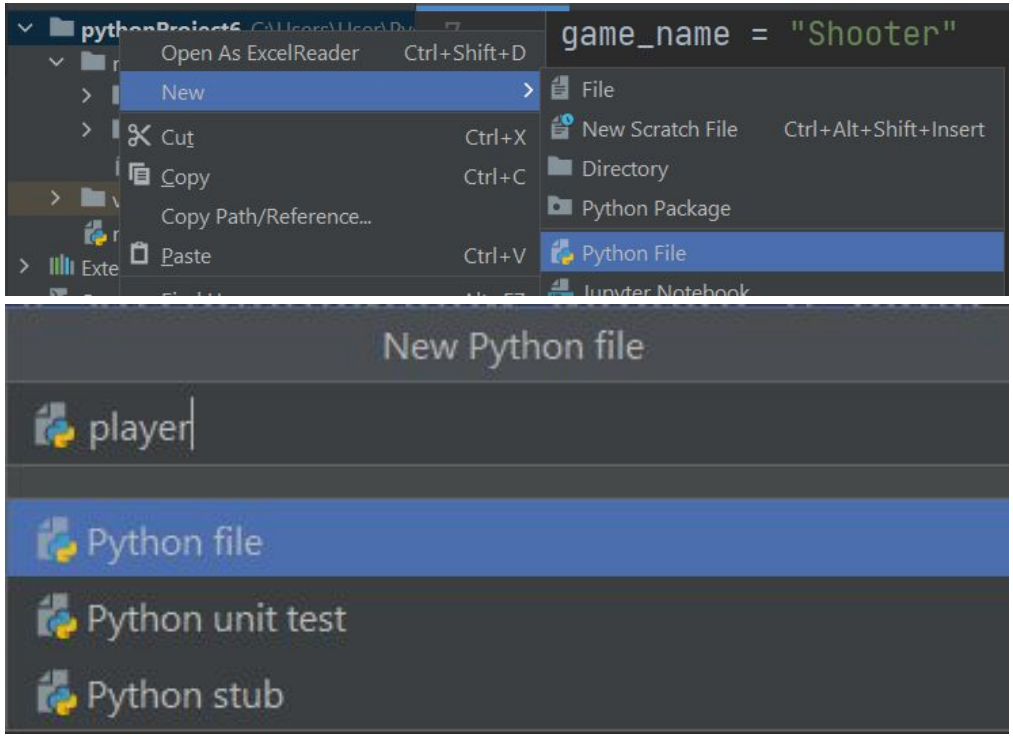
Классы — это удобный способ описать сразу несколько объектов нашей игры в одном месте. Например можно создать класс «мобы» и внутри описать, что у всех мобов есть здоровье, защита, золото, которое будет выпадать. Также мы можем описать, что все мобы умеют ходить и атаковать игрока.



Создаем класс игрока

Классы еще удобны тем, что их можно создавать в отдельных файлах, а потом подключать также как обычные модули. Давайте создадим отдельный файл в нашем проекте. Кликнем правой кнопкой мыши по названию проекта в дереве и выберем пункты **New — Python File.**

Назовем наш файл **player**.



Создаем класс игрока

Появится пустой файл, в котором мы и будем писать наш код. В этот файл нужно скопировать строчки с импортом pygame и с указанием путей расположения медиа файлов. Они нам здесь тоже пригодятся:

```
import pygame
snd_dir = 'media/snd/'             # Путь до папки со звуками
img_dir = 'media/img/'             # Путь до папки со спрайтами
width = 1366                       # ширина игрового окна
height = 768                       # высота игрового окна
```

Ниже создадим сам класс:

```
class Player(pygame.sprite.Sprite):
```

Мы только что создали класс игрока. А внутри будем сейчас описывать, что будет у нашего игрока **(свойства класса)** и что он будет уметь делать **(методы класса).**

Создаем класс игрока

Первое, что нужно в создании класса — специальная функция __init__(), которая называется конструктором. Внутри данной функции код будет запущен один раз при создании нового объекта этого класса (нашего игрока). А значит, там удобно создать все переменные с параметрами игрока. И в дальнейшем, если захотим что-то добавить (жизни, патроны и т.д.), то необходимо будет сначала создать соответствующую переменную:

```
class Player(pygame.sprite.Sprite):
    def __init__(self):             # Конструктор, где указываем что будет у объекта
        pygame.sprite.Sprite.__init__(self)                          # Игрок - спрайт
        self.image = pygame.image.load(img_dir + 'player/player.png')
        self.rect = self.image.get_rect()
```

Первая строка, `pygame.sprite.Sprite.__init__(self)` говорит что наш игрок будет являться спрайтом. Именно благодаря этой строке Pygame понимает что обращаться с нашим игроком нужно как со спрайтом.

Добавляем игрока на сцену

Чтобы добавить нашего игрока на сцену, необходимо переключиться на основной файл и там где подключаем модуль pygame, подключить файл, содержащий класс игрока:

```
import pygame
from player import Player                    # Из файла player подключаем класс Player
pygame.init()                                # Инициализируем модуль pygame
```

Это даст нам возможность обращаться к классу Player так, как будто мы его создали прямо в этом файле! Чтобы создать игрока, нужно до игрового цикла выполнить команды по созданию объекта класса и добавлению созданного объекта в группу спрайтов:

```
player = Player()                            # Создаем игрока класса Player
all_sprites.add(player)                      # Добавляем игрока в группу спрайтов
```

Перерыв

10 мин

Настройка класса игрока

Устанавливаем спрайт в центре

Переходим в файл с классом игрока и после строчек где получили рамку, пишем в координаты X и Y рамки соответствующие значения (не забудьте о self в самом начале):

```
self.image = pygame.image.load(img_dir + 'player/player.png')
self.rect = self.image.get_rect()
self.rect.x = width/2
self.rect.y = height/2
```

Если очень хочется разместить игрока ровно в центре, то можно вместо отдельных координат X и Y поменять координаты центра:

```
self.image = pygame.image.load(img_dir + 'player/player.png')
self.rect = self.image.get_rect()
self.rect.center = [width/2, height/2]
```

Все остальные спрайты

Самостоятельно добавьте классы

- ? Класс bullet (создаем файл, импортируем в main)
- ? Класс explosion (создаем файл, импортируем в main)
- ? Класс enemy_left (создаем файл, импортируем в main, добавляем ко всем спрайтам)
- ? Класс enemy_right (создаем файл, импортируем в main, добавляем ко всем спрайтам)
- ? Класс enemy_bottom (создаем файл, импортируем в main, добавляем ко всем спрайтам)
- ? Класс enemy_top (создаем файл, импортируем в main, добавляем ко всем спрайтам)

Класс Bullet

```
import pygame

snd_dir = 'media/snd/'                    # Путь до папки со звуками
img_dir = 'media/img/'                    # Путь до папки со спрайтами
width = 1366                              # ширина игрового окна
height = 768                              # высота игрового окна

# Создаем класс пули
class Bullet(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)  # Пуля - спрайт

        self.image = pygame.image.load(img_dir + 'bullet.png')
        self.rect = self.image.get_rect()
```

Класс Explosion

```
import pygame

snd_dir = 'media/snd/'                    # Путь до папки со звуками
img_dir = 'media/img/'                    # Путь до папки со спрайтами
width = 1366                              # ширина игрового окна
height = 768                              # высота игрового окна


class Explosion(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.image.load(img_dir + 'explosion/0.png')
        self.rect = self.image.get_rect()
```

Класс EnemyLeft

```
import pygame
import random

snd_dir = 'media/snd/'  # Путь до папки со звуками
img_dir = 'media/img/'  # Путь до папки со спрайтами
width = 1366                            # ширина игрового окна
height = 768                            # высота игрового окна

# Создаем класс врага слева
class EnemyLeft(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)  # Враг - спрайт

        self.image = pygame.image.load(img_dir + 'enemy_left/1.png')
        self.rect = self.image.get_rect()
        self.rect.x = 0                             # По горизонтали - слева
        self.rect.y = random.randint(0, height)   # По вертикали случайное положение
```

Класс EnemyRight

```
import pygame
import random

snd_dir = 'media/snd/'  # Путь до папки со звуками
img_dir = 'media/img/'  # Путь до папки со спрайтами
width = 1366                            # ширина игрового окна
height = 768                            # высота игрового окна

# Создаем класс врага справа
class EnemyRight(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)  # Враг - спрайт

        self.image = pygame.image.load(img_dir + 'enemy_right/1.png')
        self.rect = self.image.get_rect()
        self.rect.x = width - 100                # По горизонтали - справа
        self.rect.y = random.randint(0, height)   # По вертикали случайное положение
```

Класс EnemyBottom

```
import pygame
import random

snd_dir = 'media/snd/'  # Путь до папки со звуками
img_dir = 'media/img/'  # Путь до папки со спрайтами
width = 1366                            # ширина игрового окна
height = 768                            # высота игрового окна

# Создаем класс врага снизу
class EnemyBottom(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)  # Враг - спрайт

        self.image = pygame.image.load(img_dir + 'enemy_bottom/1.png')
        self.rect = self.image.get_rect()
        self.rect.x = random.randint(0, width)  # По горизонтали - случайное положение
        self.rect.y = height - 100              # По вертикали - снизу
```

Класс EnemyTop

```
import pygame
import random

snd_dir = 'media/snd/'  # Путь до папки со звуками
img_dir = 'media/img/'  # Путь до папки со спрайтами
width = 1366                            # ширина игрового окна
height = 768                            # высота игрового окна

# Создаем класс врага сверху
class EnemyTop(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)  # Враг - спрайт

        self.image = pygame.image.load(img_dir + 'enemy_top/1.png')
        self.rect = self.image.get_rect()
        self.rect.x = random.randint(0, width)  # По горизонтали - случайное положение
        self.rect.y = 0                         # По вертикали - сверху
```

Результат

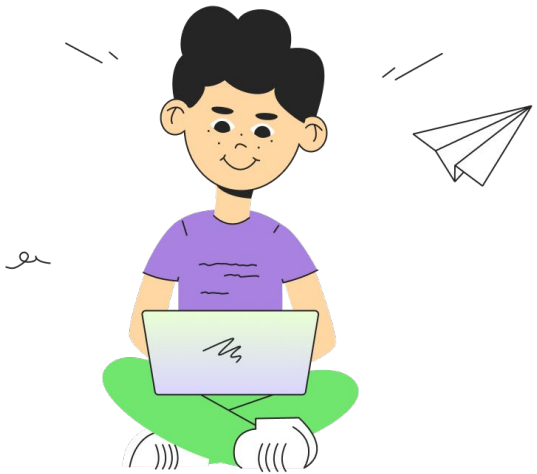
Весь проект с готовыми файлами можно скачать здесь:
https://github.com/ronmount/gb_shooter/archive/refs/heads/lesson1.zip

Итоги

- Изучили принципы ООП
- Узнали как создавать многофайловый проект
- Научились управлять несколькими спрайтами одной командой
- Создали и настроили классы игрока, врагов и остальных объектов
- Добавили возможность управления игроком

На следующем занятии:

- Настроим правильное движение игроку
- Напишем логику респауна мобов
- Сделаем случайное перемещение мобам
- Узнаем про повороты и векторы



Немного повторим

Для чего нужны классы?

Можно ли создать классы в одном файле?

Чем отличаются параметры классов (свойства) от действий классов (методов)?

Для чего нужен self?

Ваши вопросы

Спасибо
за внимание

Домашнее задание

Заполни, пожалуйста,
форму обратной связи по уроку

Напоминание для преподавателя

- **Проверить заполнение Журнала**
- **Заполнить форму Т22**