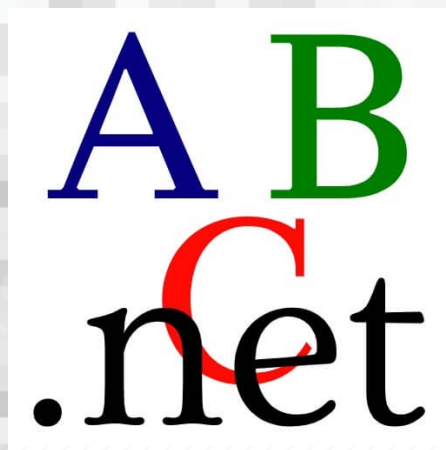


Архангельский колледж телекоммуниканий

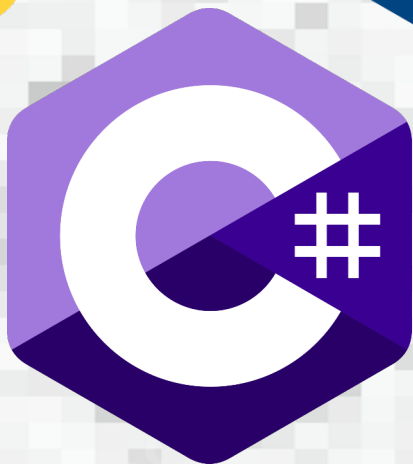
Разработка игры платформера на python

Абрамова
Полина Александровна

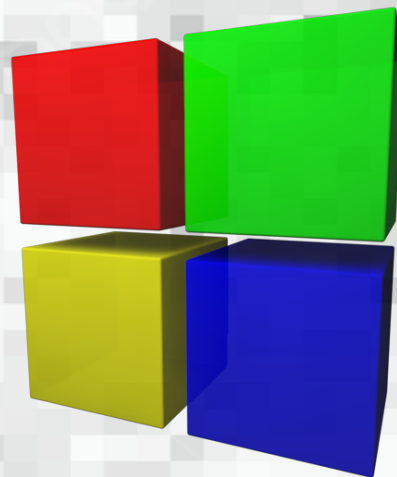
Языки программирования



HTML



Среды программирования



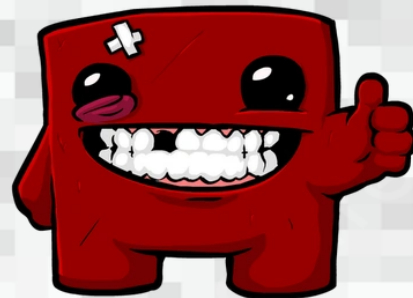
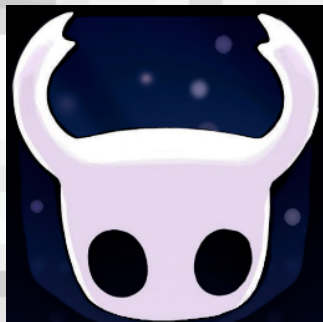
PyCharm

— это кроссплатформенная интегрированная среда разработки для языка программирования Python, разработанная компанией JetBrains на основе IntelliJ IDEA. Предоставляет пользователю комплекс средств для написания кода и визуальный отладчик.

Жанр платформер

В платформерах игровой процесс построен вокруг прыжков. Это может быть акробатика с видом сбоку или от третьего лица. Вы скачете по тем самым платформам, совершаете рывки, отталкиваетесь от стен, собираете бонусы. Запас жизней обычно ограничен, потеряли все — откатываетесь к старту уровня. Чтобы пополнить количество жизней, необходимо хватать какие-нибудь колечки и фрукты. Путь к цели преграждают пропасти и противники.

Платформеры



MARIO
003150

× 15

WORLD
-1

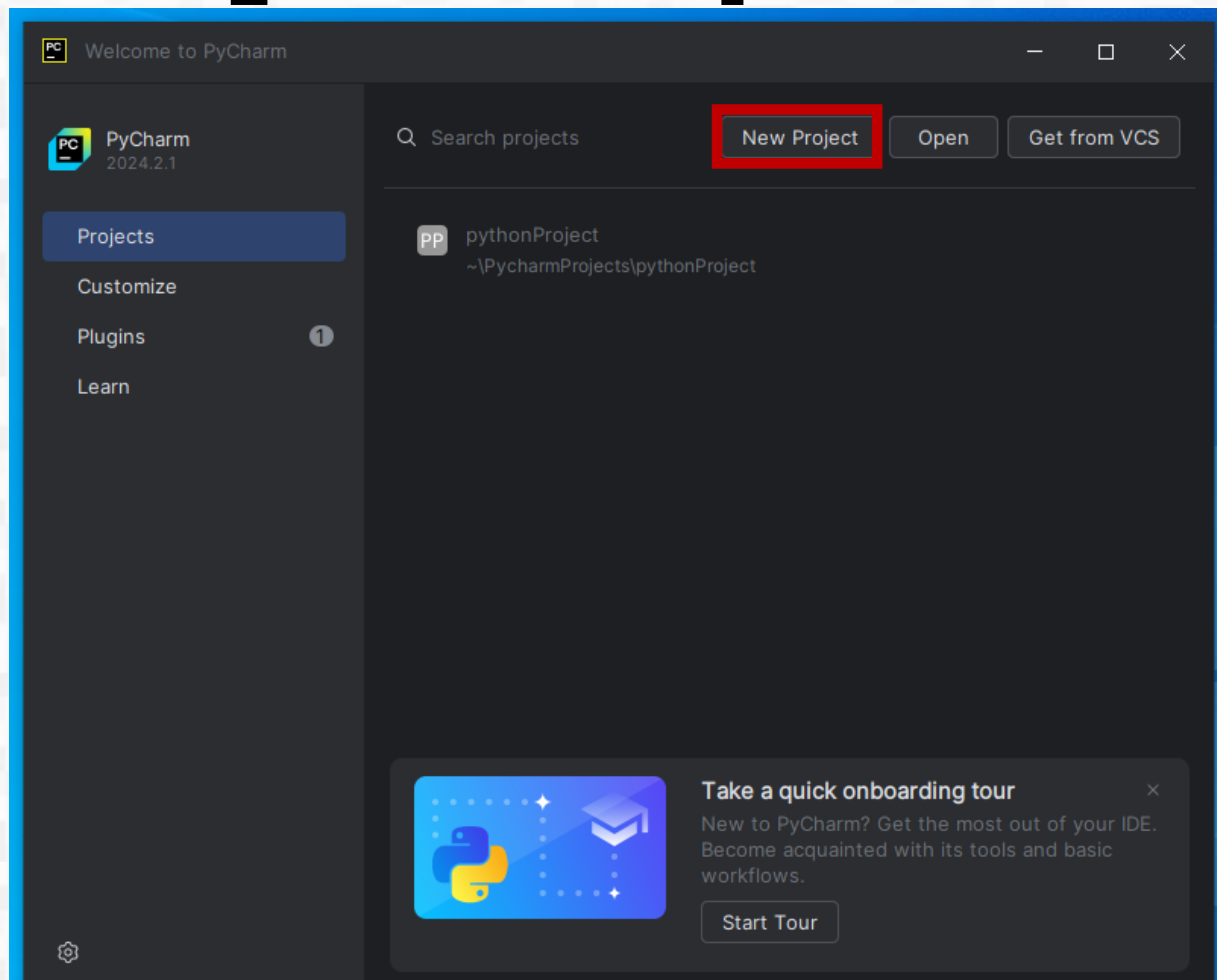
TIME
888







Создание проекта





New Project



- Pure Python
- Python
- Django
- > Other

Name:

platformer

Location:

C:\Users\221\PycharmProjects



Project will be created in: C:\Users\221\PycharmProjects\platformer



Create Git repository



Create a welcome script

Interpreter type:

Project venv

Base conda

Custom environment

Python version:



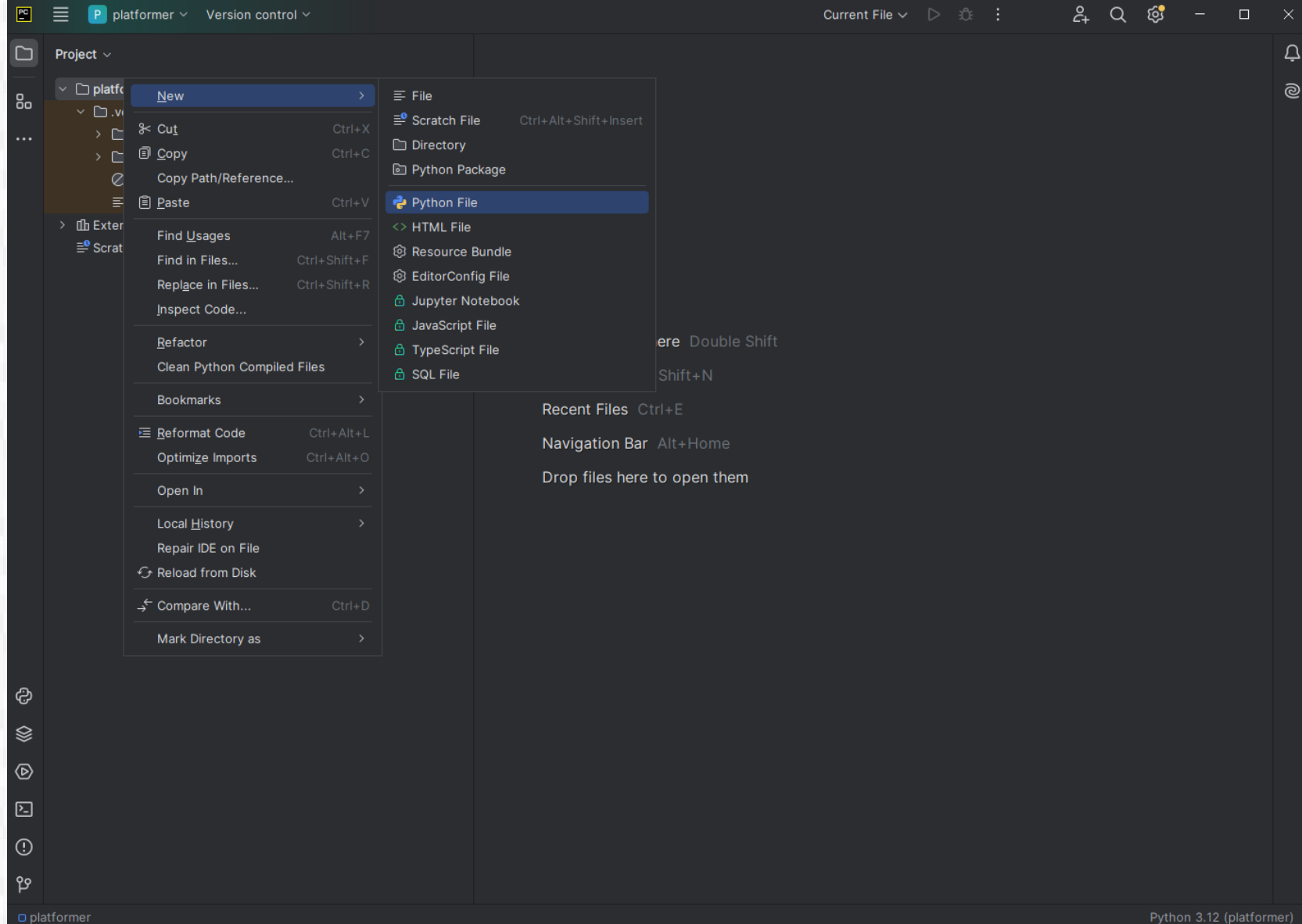
C:/Program Files/Python312/python.exe detected in the system



Python virtual environment will be created in the project root: C:\Users\221\PycharmProjects\platformer\venv

Create


Cancel




New Python file

 game|

 Python file

 Python unit test

 Python stub

more

nv.cfg

/

ories

and Consoles

I. Подключение библиотек

```
import pygame  
import sys
```

2. Инициализация pygame

```
pygame.init()
```


3. Настройки экрана

```
# Настройки экрана
```

```
WIDTH, HEIGHT = 800, 600
```

```
screen = pygame.display.set_mode((WIDTH,  
HEIGHT))
```

```
pygame.display.set_caption("Платформер")
```

4. Установка фона

```
# ФОН
```

```
background = pygame.image.load('background.png')  
background_width, background_height =  
background.get_size()  
  
scaled_background =  
pygame.transform.scale(background, (WIDTH, HEIGHT))
```

5. Указание цветов

Цвета

WHITE = (255, 255, 255)

BLUE = (0, 0, 255)

PLATFORM_COLOR = (29, 171, 68)

BLACK = (0, 0, 0)

6. Создание игрока

```
# Игрок
```

```
player_width, player_height = 70, 70
```

```
player_x, player_y = WIDTH // 2, HEIGHT - player_height - 50
```

```
player_speed = 5
```

```
player_jump_speed = 15
```

```
player_gravity = 1
```

```
player_velocity_y = 0
```

```
is_jumping = False
```

7. Изображение игрока

```
# Импорт изображения
```

```
player_image = pygame.image.load('Mario.png').convert_alpha()
```

```
scaled_image = pygame.transform.scale(player_image, (player_width,  
player_height))
```

8. Создание платформ

```
# Платформы
```

```
platforms = [  
    pygame.Rect(0, HEIGHT - 50, WIDTH, 50),  
    pygame.Rect(200, 400, 200, 20),  
    pygame.Rect(500, 300, 150, 20),  
    pygame.Rect(100, 200, 150, 20),  
]
```

9. Игровой цикл

```
clock = pygame.time.Clock()
running = True
while running:
    screen.blit(scaled_background,
                (0, 0))
```

9.1. Обработка событий

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


9.2. Управление игроком

```
# Управление игроком
```

```
keys = pygame.key.get_pressed()
```

```
if keys[pygame.K_LEFT] and player_x > 0:
```

```
    player_x -= player_speed
```

```
if keys[pygame.K_RIGHT] and player_x < WIDTH - player_width:
```

```
    player_x += player_speed
```

```
if keys[pygame.K_SPACE] and not is_jumping:
```

```
    is_jumping = True
```

```
    player_velocity_y = -player_jump_speed
```

9.3. Гравитация и прыжок

```
player_rect = pygame.Rect(player_x, player_y,  
player_width, player_height)  
  
for platform in platforms:  
    if not player_rect.colliderect(platform) and  
is_jumping != True:  
        is_jumping = True  
        break
```

9.4. Прыжок

```
if is_jumping:
    player_y += player_velocity_y
    player_velocity_y += player_gravity
    for platform in platforms:
        if player_rect.colliderect(platform) and player_velocity_y > 0:
            is_jumping = False
            player_y = platform.y - player_height + 1
            player_velocity_y = 0
            break
    if player_y > HEIGHT:
        player_y = HEIGHT - player_height
        is_jumping = False
        player_velocity_y = 0
```

9.5. Добавление платформ

```
# Отрисовка платформ
```

```
for platform in platforms:
```

```
    pygame.draw.rect(screen, GREEN, platform)
```

9.6. Отрисовка персонажа

```
if is_jumping:
    player_y += player_velocity_y
    player_velocity_y += player_gravity
    # Проверка столкновений с платформами
    for platform in platforms:
        if player_rect.colliderect(platform) and player_velocity_y > 0:
            is_jumping = False
            player_y = platform.y - player_height + 1
            player_velocity_y = 0
            break
    # Если игрок падает за пределы экрана
    if player_y > HEIGHT:
        player_y = HEIGHT - player_height
        is_jumping = False
        player_velocity_y = 0
```

9.7. Обновление кадра

Обновление экрана

```
pygame.display.flip()
```

```
clock.tick(60)
```

10. Конец игры

```
# Завершение игры  
pygame.quit()  
sys.exit()
```

Задание

- Попробуйте изменить спрайт (картинку) игрока.
- Попробуйте изменить размер и положение платформ.

Что можно добавить в игру?

- Враги
- Здоровье
- Различные препятствия
- Способности
- Конец игры