

ДЕТСКАЯ ОНЛАЙН-ШКОЛА ПРОГРАММИРОВАНИЯ

# HELLO WORLD



#ШПАРГАЛОЧКИ



# ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ PYTHON

## Базовый уровень

Материалы подготовлены отделом методической разработки

Больше полезных материалов и общения в нашем комьюнити в Telegram: [https://t.me/hw\\_school](https://t.me/hw_school)






# Создание игры “Contra” (часть 1)





Чтобы реализовать смену фона, когда персонаж уходит за край окна, нужно загрузить несколько текстур в список, а также сохранить индекс текущей текстуры:

```
self.background_textures = []
```

```
for i in range(1,16): 
```

```
    self.background_textures.append(arcade.load_texture(f'background/Map{i}  
.png'))
```

```
self.texture_index = 0
```



И отрисовать соответствующую текстуру:

**def on\_draw(self):**

```
    arcade.draw_texture_rectangle(SCREEN_WIDTH / 2, SCREEN_HEIGHT / 2,  
    SCREEN_WIDTH, SCREEN_HEIGHT,  
    self.background_textures[self.index_texture])
```



Смена текстуры будет происходить, когда главный герой зайдет за правый или левый край окна. Кроме проверки его координат нужна еще и проверка, что он находится не на первой (при движении влево) и не на последней (при движении вправо) текстуре.



На примере движения вправо:

```
def back_left(self):
```

```
    if self.left > SCREEN_WIDTH:
```

```
        self.center_x = 0
```

```
        return True
```

```
    return False
```

```
# в update
```



```
    if self.bill.back_left():
```

```
        if self.index_texture < len(self.background_textures)-2:
```

```
            self.index_texture += 1
```



Чтобы включать анимацию персонажа только при движении, нужно создать логическую переменную. Она будет отвечать за состояние персонажа - движется он или нет:

```
self.is_walk = False
```

```
# в update
```

```
if self.game:
```

```
    self.bill.update()
```

```
        if self.is_walk:
```

```
            self.bill.update_animation(delta_time)
```





Состояние персонажа будет меняться при нажатии на клавишу (на примере левой):

```
def on_key_press(self, key, modifiers):
```

```
    if key == arcade.key.LEFT:
```

```
        self.is_walk = True
```

И при отпускании клавиши:

```
def on_key_release(self, key, modifiers):
```

```
    if key == arcade.key.LEFT or key == arcade.key.RIGHT:
```

```
        self.is_walk = False
```



**arcade.PhysicsEnginePlatformer** - класс, облегчающий разработку платформера благодаря встроенным методам, обеспечивающим взаимодействие игрока и платформ.

При создании объекта обязательно нужно указать спрайт игрока, спрайтлист с платформами и гравитацию, которая воздействует на игрока:

```
self.engine = arcade.PhysicsEnginePlatformer(self.bill, self.lines, GRAVITY)
```

А также движок нужно обновлять в классе окна:

**# в update**

```
self.engine.update()
```





**can\_jump** - метод класса **arcade.PhysicsEnginePlatformer**, проверяющий, есть ли под персонажем платформа.

**jump** - метод **arcade.PhysicsEnginePlatformer**, отвечающий за прыжок персонажа, принимает скорость (силу) прыжка.

**def on\_key\_press(self, key, modifiers):**

...



**if key == arcade.key.UP:**

**if self.engine.can\_jump():**

**self.engine.jump(JUMP)**



**enumerate** - функция, которая позволяет перебрать список и возвращает каждый его элемент в паре с индексом.

Например:

```
my_list = ['a', 'b', 'c', 'd']
```

```
for i, letter in enumerate(my_list):
```

```
    print(i, letter) # 0 a, 1 b, 2 c, 3 d
```



**abs** - функция, которая возвращает модуль числа (и отрицательное, и положительное число вернутся как положительные):

**abs(-5) # 5**

**abs(5) # 5**

