Arcade

Sprites e utilização de teclado

Arcade

Sprites e utilização de teclado

Em computação gráfica, um **sprite** (do latim *spiritus*, significando "duende", "fada") é um objeto gráfico bi ou tridimensional que se move em uma tela sem deixar traços de sua passagem (como se fosse um "espírito").

Fonte:

https://pt.wikipedia.org/wiki/Sprite_(computa%C3%A7%C3%A3o_gr%C3%A1fica)

Arcade

Sprites e utilização de teclado

No Arcade temos a classe Sprite (arcade.Sprite), que representa um “sprite” na tela.

A maioria dos jogos gira em torno de sprites.

Link da documentação da classe Sprite:

https://arcade.academy/arcade.html?highlight=sprite#arcade.Sprite

Arcade

Sprites e utilização de teclado

No Arcade temos também a classe SpriteList (arcade.SpriteList), que representa uma lista de “sprites”.

Vamos utilizar esta classe para adicionar e remover nossos sprites no jogo.

Link da documentação da classe Sprite:

https://arcade.academy/arcade.html?highlight=spritelist#arcade.SpriteList

Arcade

Sprites e utilização de teclado

Nesta aula vamos aprender a trabalhar com sprites e movimentá-los na tela utilizando as setas do teclado.

Arcade

Sprites e utilização de teclado

Vamos começar o exemplo importando o arcade e definindo as constantes para o nosso jogo.

```
# importando o arcade
import arcade

# Definindo as constantes
LARGURA_JANELA = 800
ALTURA_JANELA = 600
TITULO = "Jogo de Nave"
ESCALA = 1.0
```

Arcade

Sprites e utilização de teclado

Vamos criar a classe FlyingSprite herdada de arcade.Sprite() que será a classe base para nossos sprites.

Essa classe terá o método update() apenas.

```
# Classes
class FlyingSprite(arcade.Sprite):
    """Classe base para nosso sprite
    """

    def update(self):
        """Atualiza a posição do sprite
        """

        # Move o sprite
        super().update()
```

Arcade

Sprites e utilização de teclado

Vamos criar agora a classe do jogo, herdada de arcade.Window que terá os métodos __init__, setup, on_key_press, on_key_release, on_update e on_draw.

Neste slide você vê o método __init__.

```
class Game(arcade.Window):
    """Nesse jogo vamos movimentar
    nosso sprite dentro da área da janela
    """


    def __init__(self, width: int, height: int, title: str):
        """Inicializa o jogo
        """
        super().__init__(width, height, title)

        # Limpa a lista de sprites
        self.all_sprites = arcade.SpriteList()
```

Arcade

Sprites e utilização de teclado

No método setup da classe Game vamos definir a cor de fundo e vamos configurar nosso player. No método arcade.Sprite estamos passando o arquivo de imagem que será nosso player e uma escala de 1.0, que é o tamanho original da imagem. Você pode alterar esta escala por exemplo para 0.5 para que a imagem tenha a metade do tamanho.

Faça o download do arquivo nave.png nos recursos da aula.

```
def setup(self):
    """Prepara o jogo
    """

    # Define a cor de fundo
    arcade.set_background_color(arcade.color.SKY_BLUE)

    # Configura o player (nave)
    self.player = arcade.Sprite("images/nave.png", ESCALA)
    self.player.center_y = self.height / 2
    self.player.left = 10
    self.all_sprites.append(self.player)
    self.paused = False
```

Arcade

Sprites e utilização de teclado

No método on_key_press da classe Game manipulamos a entrada do teclado.

Veja o que são os dois parâmetros:

symbol: Qual tecla foi pressionada;

modifiers: Quais modificadores foram pressionados (shift, ctrl, alt).

Vamos utilizar as setas do teclado para movimentar nosso player, a tecla “P” para pausar e a tecla “Q” para sair do jogo.

```
def on_key_press(self, symbol: int, modifiers: int):
    if symbol == arcade.key.Q:
        arcade.close_window()

    if symbol == arcade.key.P:
        self.paused = not self.paused

    if symbol == arcade.key.UP:
        self.player.change_y = 250

    if symbol == arcade.key.DOWN:
        self.player.change_y = -250

    if symbol == arcade.key.LEFT:
        self.player.change_x = -250

    if symbol == arcade.key.RIGHT:
        self.player.change_x = 250
```

Arcade

Sprites e utilização de teclado

No método on_key_release da classe Game também manipulamos a entrada do teclado.

Veja o que são os dois parâmetros:

symbol: Qual tecla foi pressionada;

modifiers: Quais modificadores foram pressionados (shift, ctrl, alt).

Este método é acionado quando as teclas são liberadas.

Quando soltamos as teclas de seta para cima ou para baixo, a propriedade change_y é zerada e quando soltamos as teclas de seta para esquerda e direita, a propriedade change_x é zerada.

```
def on_key_release(self, symbol: int, modifiers: int):

    if (symbol == arcade.key.UP or
            symbol == arcade.key.DOWN):
        self.player.change_y = 0

    if (symbol == arcade.key.LEFT or
            symbol == arcade.key.RIGHT):
        self.player.change_x = 0
```

Arcade

Sprites e utilização de teclado

No método on_update da classe Game vamos validar se o jogo foi pausado para não atualizarmos nada na tela.

Caso o jogo esteja em execução, vamos percorrer a lista de Sprites para atualizar todos os sprites.

O parâmetro delta_time é um float que contém o tempo desde a última vez que on_update foi executado.

center_x: Localização do centro do sprite no eixo x.

center_y: Localização do centro do sprite no eixo y.

change_x: Vetor de movimento na direção do eixo x.

change_y: Vetor de movimento na direção do eixo y.

```
def on_update(self, delta_time: float):

    # Se o jogo estiver em pausa, não fazer nada
    if self.paused:
        return

    # Atualiza todos sprites
    for sprite in self.all_sprites:
        sprite.center_x = int(
            sprite.center_x + sprite.change_x * delta_time
        )
        sprite.center_y = int(
            sprite.center_y + sprite.change_y * delta_time
        )
```

Arcade

Sprites e utilização de teclado

No método on_draw da classe Game vamos desenhar os objetos na tela.

Após iniciar com start_render() desenhamos os sprites que estão na lista all_sprites usando o método draw de SpriteList.

Em seguida validamos se a variável de controle “paused” é verdadeira para desenharmos na tela o texto “EM PAUSA”, nas posições x igual a 120, y igual a 0, cor vermelha e tamanho da fonte igual a 100.

```
def on_draw(self):
    """Desenha os objetos do jogo
    """


    arcade.start_render()
    self.all_sprites.draw()

    if self.paused:
        arcade.draw_text("EM PAUSA", 120, 0,
                         arcade.color.BRICK_RED, 100)
```

Arcade

Sprites e utilização de teclado

Agora vamos escrever o ponto de entrada do programa (onde o programa começa).

Para isso criaremos uma variável chamada jogo que será uma instância da classe Game.

Vamos iniciar o jogo informando a largura e altura da janela multiplicada pela constante ESCALA, além do título da janela.

Configuramos o jogo com jogo.setup() e executamos com arcade.run().

```
if __name__ == "__main__":
    jogo = Game(
        int(LARGURA_JANELA * ESCALA),
        int(ALTURA_JANELA * ESCALA),
        TITULO
    )
    jogo.setup()
    arcade.run()
```

Arcade

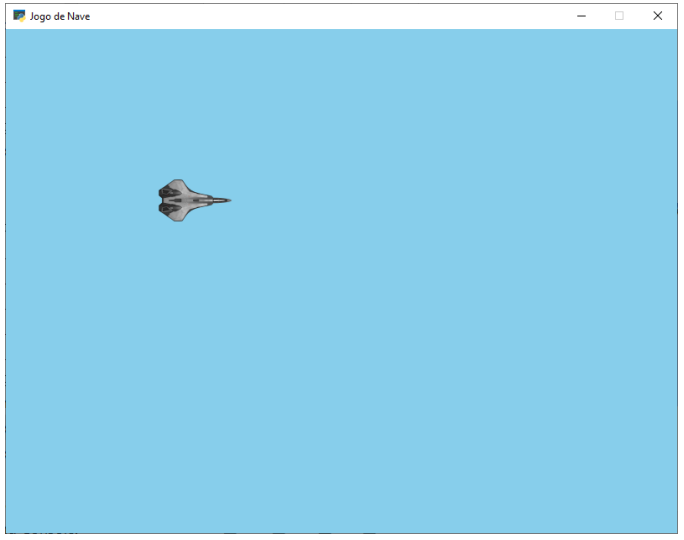
Sprites e utilização de teclado

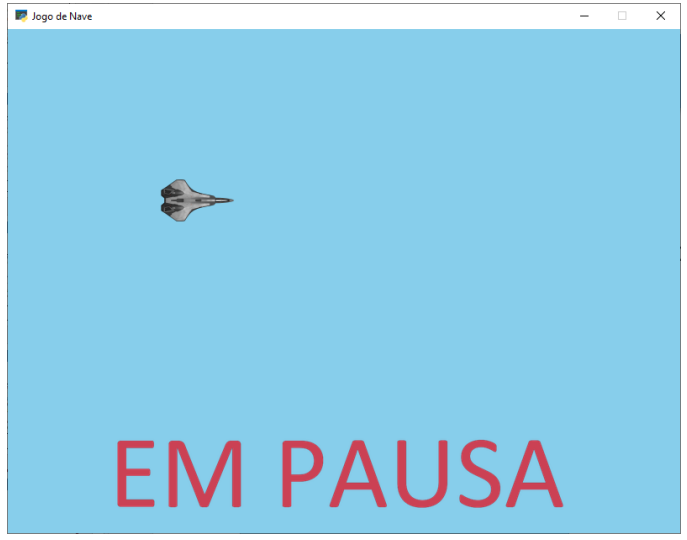
Execute o programa.

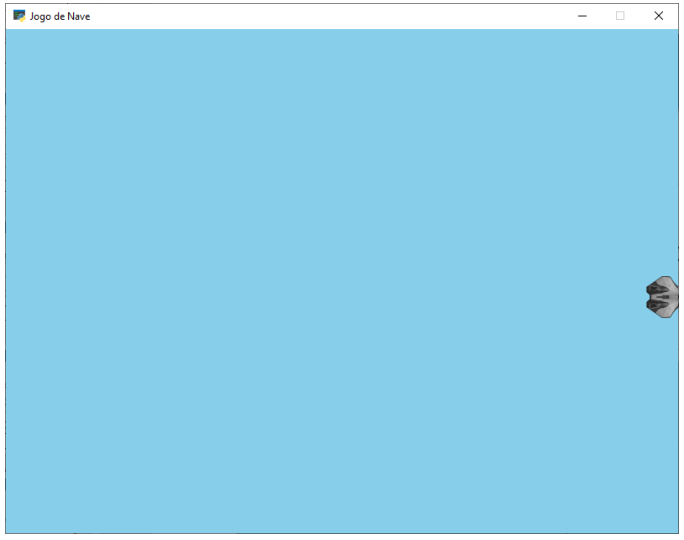
Utilize as setas do teclado para movimentar a nave.

Pressione a tecla “P” para pausar o jogo. Pressione novamente para liberar.

Observe que não fizemos nenhum tratamento para evitar que a nave saia da área da janela.







Arcade

Sprites e utilização de teclado

Para evitar que a nave saia da janela, adicione o código a seguir no método on_update da classe Game.

As propriedades top, right, bottom e left de arcade.Sprite definem a localização do elemento na janela. Podemos consultar e alterar esta localização.

```
if self.player.top > self.height:
    self.player.top = self.height
if self.player.right > self.width:
    self.player.right = self.width
if self.player.bottom < 0:
    self.player.bottom = 0
if self.player.left < 0:
    self.player.left = 0
```

Nesse código estamos validando:

Se a propriedade top da nave for maior que a altura da janela, defina a propriedade top como a altura da janela. Assim a nave vai parar quando subir até o topo da janela.

Se propriedade right da nave for maior que a largura da janela, a propriedade right receberá a largura da janela, evitando que a nave saia pelo lado direito da janela.

Se a propriedade bottom da nave for menor que zero, a propriedade bottom receberá zero. Assim a nave vai parar quando descer até o final da janela.

Se propriedade left da nave for menor que zero, a propriedade left receberá zero, evitando que a nave saia pelo lado esquerdo da janela.

Após esta alteração tente levar a nave até as bordas da janela e veja o resultado.

FIM