

Projeto Final - Avaliação A3

[Super Mario Bros]

Relatório técnico apresentado na UC Usabilidade, desenvolvimento web e jogos pelo Prof. MSc Flávio Henrique da Silva.

Curitiba

2025

INTEGRANTES DO GRUPO

Anna Clara Aguemi - 172215425

Miguel Otto Wolff - 172211098

Pablo Gabriel Cavallari - 172214836

SUMÁRIO

1 INTRODUÇÃO 4

2 DESENVOLVIMENTO 5

2.1 Divisão das Tarefas 5

2.2 Estrutura do Projeto 6

2.3 Explicação da Aplicação/Software 7

2.4 Orientações de execução da Aplicação/Software 7

2.5 Repositório 7

3 CONCLUSÃO 7

4 REFERÊNCIAS 9

1 INTRODUÇÃO

Este projeto tem como objetivo o desenvolvimento de um jogo baseado no clássico Super Mario Bros, utilizando a biblioteca Pygame. A proposta incluiu a criação e implementação de mecânicas características do jogo original, com a obrigatoriedade de pelo menos três fases distintas.

5

2 DESENVOLVIMENTO

Foi utilizada a biblioteca Pygame para o desenvolvimento deste projeto. Nele, são abordados os seguintes arquivos de código: main.py, game.py, menu.py, obj.py e enemies.py, além dos arquivos presentes na pasta Assets, que são utilizados para a construção do mundo e dos sons do jogo.

O arquivo main.py é responsável por inicializar o jogo e mantê-lo em execução em um loop contínuo, evitando que a janela seja encerrada. Ele também detecta o clique no botão "X" da janela para encerrar o jogo.

O arquivo `game.py` cria o cenário e gerencia a lógica do funcionamento do mundo, como, por exemplo, as colisões e a movimentação do jogador. É nesse arquivo que são criadas as fases, oferecendo diferentes desafios ao jogador.

O arquivo `menu.py` controla os menus do jogo, incluindo a tela final exibida ao concluir o jogo. O arquivo `obj.py` define a lógica dos objetos e do personagem Mario, que são utilizados em `game.py`. Já o `enemies.py` trata da lógica dos inimigos, também utilizados em `game.py`.

2.1 Divisão das Tarefas

A criação do arquivo `main.py` foi realizada de forma colaborativa por todos os alunos durante as aulas.

O arquivo `menu.py` foi desenvolvido pelo aluno Pablo Gabriel Cavallari, responsável pela criação das classes `Menu` e `FinalScreen`, além de realizar ajustes no `main.py` para correção de bugs relacionados à implementação dessas classes.

O arquivo `enemies.py` foi desenvolvido pela aluna Anna Clara Aguemi, que implementou a lógica das animações dos inimigos e realizou correções de bugs durante a integração dos inimigos no `game.py`.

Os arquivos `game.py` e `obj.py` foram desenvolvidos pelo aluno Miguel Otto Wolff, responsável pela criação das fases no `game.py`, bem como pela implementação das colisões com blocos e demais objetos do jogo.

6

2.2 Estrutura do Projeto

O projeto foi desenvolvido utilizando a biblioteca Pygame, com o objetivo de criar um jogo de plataforma. A estrutura principal do jogo está centrada na classe `Main`, responsável por gerenciar o fluxo da aplicação. Por meio do método `draw`, são renderizados o menu inicial, o jogo em si e a tela final. O método `events` trata os eventos da janela, como o encerramento da aplicação ou as interações com o jogo, menu e tela final.

O método `encerra_jogo` invoca a classe `FinalScreen`, possibilitando a exibição da tela final e a manipulação de seus eventos. Além disso, altera o estado do jogo para o modo finalizado. O método `run` contém o loop principal da aplicação, iniciando a lógica do jogo com a instância `self.game = Game(self.size_x, self.size_y)`, prosseguindo com as atualizações necessárias até a exibição da tela final.

Na estrutura `if __name__ == "__main__":` até a chamada `main.run()`, ocorre a criação da janela e a inicialização da classe `Main`, que gerencia os eventos principais da aplicação.

No arquivo `menu.py`, encontram-se as classes `Menu` e `FinalScreen`. A classe `Menu` contém os métodos `tela_comandos` e `tela_fases`, responsáveis por apresentar ao jogador as instruções e a escolha de fase. Já a classe `FinalScreen` reúne os métodos que controlam a transição para a tela final e o encerramento do jogo.

O arquivo `enemies.py` define três classes de inimigos: `Goomba`, `Bowser` e `KoopaTroopa`. A classe `KoopaTroopa` inclui o método `load_frames`, que permite a animação dos diferentes tipos de `Koopa Troopas`, e o método `morrer`, que foi implementado em todas as classes de inimigos para excluir seus retângulos de colisão após a derrota.

No arquivo `obj.py`, estão implementadas diversas classes que compõem os elementos do jogo. A classe `Mario` gerencia os estados do personagem, seus power-ups e animações. As classes `Cogumelo`, `Estrela` e `FlorDeFogo` controlam a criação, animação e efeitos dos itens correspondentes. As classes `Bloco` e `QuestionBlock` cuidam da lógica de geração de itens e servem como plataformas para o personagem.

7

A classe `Fireball` centraliza a lógica das bolas de fogo, que são disparadas após Mario obter a `FlorDeFogo`. A classe `Coin` anima as moedas, verifica sua coleta e removê-las do cenário. As moedas também podem surgir de um `QuestionBlock`.

A classe `Flag` atua como um marcador de fim de fase, sendo uma das implementações mais simples, utilizada apenas para exibir a bandeira no mundo do jogo.

No arquivo `game.py`, encontram-se as constantes que definem os elementos do cenário e o método `carregar_fases`, responsável pela criação de três fases distintas. Isso permite alterar o ambiente com base na seleção de fase. O método `update` gerencia a lógica de colisão com blocos, inimigos e power-ups, além de atualizar o estado do Mario de acordo com os itens coletados.

2.3 Explicação da Aplicação/Software

Baixar arquivo .zip do código do repositório do github conforme o link disponibilizado.

Extrair todos os arquivos, na pasta extraída navegar até a pasta `dist` e executar o arquivo `"Super Mario Bros.exe"`

2.4 Orientações de execução da Aplicação/Software

Ter python na versão 3.13.2 64-bit instalado, biblioteca `pygame` instalada na versão 2.6.1.

2.5 Repositório

Link do repositório github do projeto: <https://github.com/MiguelWolff/A3-Usabilidade-desenvolvimento-web-e-mobile/tree/main>

3 CONCLUSÃO

Este projeto proporcionou uma rica experiência prática no desenvolvimento de jogos com `Pygame`, exigindo dos integrantes o domínio de conceitos como lógica de programação, estruturação de fases, tratamento de eventos e animações. A divisão clara das tarefas permitiu uma colaboração eficiente e produtiva, resultando em um jogo funcional e alinhado com a proposta da disciplina.

8

REFERÊNCIAS

ENCYCLOPEDIA – Cores e combinações. Encycolorpedia, [s.d.]. Disponível em: <https://encycolorpedia.pt/>. Acesso em: 06 junho 2025.

SUPER MARIO BROTHERS SPRITES. NES Maps, [s.d.]. Disponível em: <https://nesmaps.com/maps/SuperMarioBrothers/sprites/SuperMarioBrothersSprites.html>. Acesso em: 19 maio 2025.

SPLIT IMAGE ONLINE. Ezgif, [s.d.]. Disponível em: <https://ezgif.com/split>. Acesso em: 21 maio 2025.

PYGAME DOCUMENTATION. Pygame, [s.d.]. Disponível em: <https://www.pygame.org/docs/>. Acesso em: 07 abril 2025.