

**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E
TECNOLOGIA DE SÃO PAULO**

JOÃO PEDRO VIANA RODRIGUES

DRAGON TYPING

**CAMPOS DO JORDÃO
2025**

RESUMO

Este relatório descreve o desenvolvimento do "Dragon Typing", um jogo de digitação com temática medieval, criado utilizando a biblioteca Raylib em C++. O projeto visa proporcionar uma experiência interativa e desafiadora, onde o jogador deve digitar palavras que aparecem aleatoriamente na tela antes que o tempo se esgote. O jogo incorpora elementos visuais e sonoros para aumentar a imersão. A implementação foi realizada sob uma abordagem orientada a objetos, visando modularidade e manutenibilidade do código.

INTRODUÇÃO

Este trabalho apresenta o processo de concepção e desenvolvimento do jogo "Dragon Typing", um software interativo que desafia as habilidades de digitação do usuário. O objetivo principal do projeto é criar uma experiência de jogo envolvente e funcional, utilizando os princípios da programação orientada a objetos e a biblioteca gráfica Raylib.

Objetivos:

- Desenvolver um jogo de digitação em C++ utilizando a biblioteca Raylib.
- Implementar um sistema de gameplay que inclua geração aleatória de palavras, controle de tempo, pontuação e sistema de vidas.
- Aplicar conceitos de programação orientada a objetos para estruturar o código de forma modular e escalável.
- Integrar elementos visuais e sonoros para aprimorar a imersão e a experiência do usuário.

Metodologia

O desenvolvimento do "Dragon Typing" iniciou-se com a definição das considerações iniciais e o planejamento da arquitetura do projeto. O uso da linguagem C++ e da biblioteca Raylib foi motivada pelas aulas apresentadas na matéria de Programação Orientada a Objetos, que se mostrou muito robusta e de boa performance.

Considerações Iniciais sobre o Projeto: O jogo foi concebido como um desafio de digitação rápido, onde o jogador deve transcrever palavras que aparecem na tela. A temática de dragões e fantasia foi escolhida para tornar a experiência mais envolvente. Aspectos como a legibilidade do texto, o tempo de resposta da interface e o feedback claro ao usuário foram prioritários.

Ferramentas Utilizadas:

- **Linguagem de Programação:** C++
- **Compilador:** GCC/g++ (MinGW-w64, versão 6.3.0)
- **Biblioteca Gráfica e de Áudio:** Raylib (versão 5.5, MinGW-w64)
- **Ambiente de Desenvolvimento/Edição:** Visual Studio Code
- **Ferramentas de Imagem:** ChatGPT e Google Gemini
- **Ferramentas de Áudio:** Pixabay e FL Studio.

Descrição do Projeto: O jogo "Dragon Typing" é estruturado em cinco telas principais para gerenciar o fluxo do jogador:

- **Tela de Menu:** É a tela inicial. Apresenta o título do jogo e a imagem de fundo. O jogador pode selecionar o nível de dificuldade (Fácil, Normal, Difícil) e optar por iniciar o jogo (indo para a Entrada de Nome) ou verificar o placar.
- **Tela de Entrada de Nome:** Pede ao jogador que digite um nome de usuário. Esse nome é armazenado e utilizado para registrar a pontuação no placar de líderes.
- **Tela de Gameplay:** Onde a ação principal ocorre. Possui um fundo temático, um temporizador regressivo (cuja velocidade depende da dificuldade), contador de vidas e pontuação. As palavras a serem digitadas aparecem em posições aleatórias na tela.
- **Tela de Game Over:** Exibida quando o jogador perde todas as vidas. Nesta tela, a pontuação final é salva no placar de líderes e é exibida uma mensagem de "Fim de Jogo". O jogador pode pressionar ENTER para retornar ao Menu Principal.
- **Tela de Placar (High Score):** Carrega e exibe as 10 maiores pontuações a partir de um arquivo local. O jogador pode visualizar o placar a partir do menu e retornar a ele.

A lógica do jogo é gerenciada por classes:

- **Game**: Gerencia os estados do jogo, o loop principal, a transição entre telas e a inicialização/finalização de recursos.
- **Player**: Controla as vidas e a pontuação do jogador.
- **WordManager**: Responsável por carregar palavras de um arquivo words.txt, selecioná-las aleatoriamente, gerenciar o tempo de digitação de cada palavra e processar a entrada do usuário.
- **Validação de Entrada**: A entrada do usuário é validada para aceitar apenas caracteres alfabéticos (e espaços, se a palavra for composta) e convertida para maiúsculas para comparação consistente.
- **Randomização Visual**: As palavras são exibidas em coordenadas X e Y aleatórias dentro de uma área segura da tela para aumentar o desafio.
- **Áudio**: O jogo inclui música de fundo e efeitos sonoros para eventos como acertos e perdas de vida.

Resultados Obtidos

O desenvolvimento do "Dragon Typing" resultou em um protótipo funcional que demonstra a aplicação dos conceitos de programação em um ambiente de jogo interativo. Foram superados desafios como a configuração do ambiente de desenvolvimento C++ e Raylib, a compilação de múltiplos arquivos, a integração de assets visuais e sonoros, e a implementação de lógica de jogo orientada a objetos.

Capturas de Tela do Jogo:

Tela de Menu Inicial:



Tela de Gameplay:



Tela de Placar de Líderes:



Tela de GameOver:



Conclusão

O projeto "Dragon Typing" foi concluído com sucesso, resultando em um jogo de digitação interativo. A utilização da abordagem orientada a objetos demonstrou ser eficaz na organização do código, facilitando a depuração e a adição de novos recursos. A integração da biblioteca Raylib permitiu um desenvolvimento ágil da parte gráfica e de áudio, proporcionando uma experiência visualmente e sonora. O aprendizado e a superação dos desafios de compilação e linkagem em um ambiente C++ complexo foram pontos cruciais do desenvolvimento.

Sugestões para Possíveis Melhorias:

- **Power-ups e Penalidades:** Introduzir itens no jogo que concedem bônus de tempo, vida ou que gerem efeitos negativos.
- **Animações do Dragão:** Animar o dragão do fundo para reagir ao desempenho do jogador (ex: mais furioso quando o tempo está acabando, ou comemorando um acerto).
- **Controles de Volume:** Adicionar sliders ou opções no menu para que o jogador possa ajustar o volume da música e dos efeitos sonoros.
- **Feedback Visual Aprimorado:** Desenvolver efeitos visuais mais elaborados para acertos críticos ou erros, como explosões de fogo ou texto flutuante.
- **Compatibilidade com Acentuação:** Explorar métodos para lidar com caracteres acentuados em português, caso se deseje usar palavras com essa característica futuramente.
- **Refinamento da UI/UX:** Melhorar a interface do usuário com elementos mais gráficos para o HUD (barras de vida/tempo) e transições de tela mais suaves.

8. Referências Bibliográficas

- **RAYLIB.** *raylib: a simple and easy-to-use library to enjoy videogames programming.* Disponível em: <https://www.raylib.com/>. Acesso em: [Data do seu acesso, ex: 08 jun. 2025].
- **C++ Reference.** *cppreference.com.* Disponível em: <https://en.cppreference.com/w/cpp>. Acesso em: [Data do seu acesso, ex: 08 jun. 2025].