Documentação do Jogo

<Ping Pong!>

Autores: Gustavo Pereira e Renan Mendes

Belo Horizonte

<julho de 2023>.

Documentação do Sistema

Introdução

Escopo do Jogo

Nome do jogo e de seus componentes principais

Nome do jogo: Ping Pong!

Componentes:

- Paddle (Raquete): Representa os dois jogadores no jogo, cada um com sua própria raquete que se move para cima e para baixo.
- Ball (Bola): A bola que quica entre as raquetes dos jogadores.
- Score (Pontuação): Mantém o controle da pontuação de cada jogador.
- Tela Inicial: Exibe uma tela de menu inicial antes do jogo começar.
- Tela de Vitória: Mostra quem ganhou o jogo quando um dos jogadores atinge a pontuação de vitória (WINNING_SCORE).
- Background (Fundo): Imagens de fundo utilizadas no jogo.
- Eventos: O jogo reage aos eventos do teclado (setas para cima e para baixo, 'W' e 'S', 'Enter' e 'Esc') e eventos de timer para atualização contínua.
- Colisões: Verifica colisões entre a bola e as raquetes dos jogadores para que a bola possa quicar e alterar sua direção adequadamente.
- Texto: Usa uma fonte para exibir informações como pontuações e mensagens de "Match Point!".

Análise do público-alvo

- **Jogadores casuais:** O jogo é uma versão digital simplificada do clássico jogo de pingue-pongue (também conhecido como tênis de mesa). É adequado para jogadores casuais que buscam uma experiência de jogo simples e rápida. Não requer conhecimentos complexos ou habilidades avançadas, o que torna o jogo acessível para pessoas de diferentes idades e níveis de habilidade.
- **Entusiastas de jogos retrô:** Com seu estilo de jogo clássico de Pong, o jogo pode atrair entusiastas de jogos retrô que têm apreço por jogos mais simples e nostálgicos. A escolha de gráficos 2D e controles simples pode evocar sentimentos de nostalgia para os jogadores que cresceram jogando jogos antigos.

Pré-requisitos

Dispositivo computacional (PC, notebook), Sistema operacional Windows.

Objetivos educacionais e específicos do jogo

Objetivo educacional geral:

- Proporcionar entretenimento educativo enquanto prática e aprimora habilidades cognitivas, como reflexos, coordenação motora e tomada de decisões rápidas.

Objetivos educacionais específicos:

- Coordenação motora: O jogo requer que os jogadores controlem as raquetes (paddle) para rebater a bola, melhorando a coordenação entre a visão e os movimentos das mãos.
- Concentração e atenção: Os jogadores precisam acompanhar a bola em movimento, permanecer alertas e focados para responder rapidamente aos seus movimentos.
- Agilidade mental: Os jogadores devem tomar decisões rápidas sobre a direção em que devem mover suas raquetes para rebater a bola e evitar que ela passe por eles.

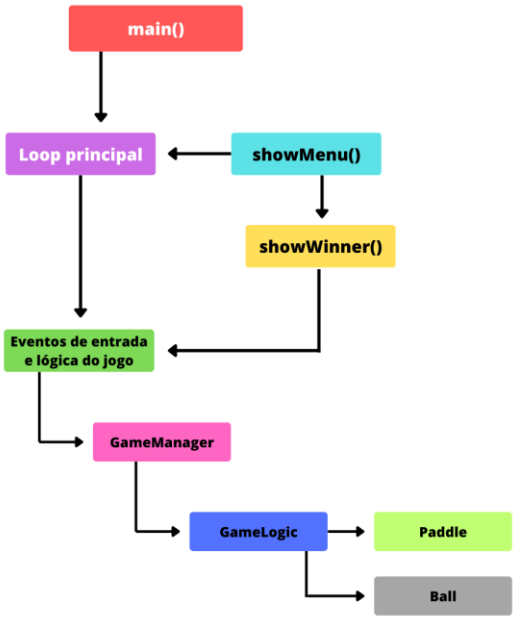
Itens de Avaliação

- **Pontuação:** Acompanhar a pontuação dos jogadores, premiando-os por cada ponto marcado. Isso pode ajudar a medir o progresso e o sucesso dos jogadores no jogo.
- **Tempo de reação:** Medir o tempo de reação dos jogadores para responder aos movimentos da bola. Quanto mais rápido eles conseguirem rebater a bola, melhor será sua pontuação nesse item.
- **Precisão:** Avaliar a precisão dos jogadores ao acertar a bola com a raquete. Isso pode ser medido pela porcentagem de vezes que eles conseguem rebater a bola com sucesso em relação ao número total de tentativas.

Estratégia de apresentação

- Introdução: Apresentando o jogo Ping Pong!
- Público-alvo: Introduzindo o nosso público-alvo.
- Objetivos educacionais.
- Recursos e funcionalidades: Jogabilidade e controles.
- Avaliação: Pontuação do jogo, jogo competitivo simples.
- Pré-requisitos: Listar os pré-requisitos.
- Demonstração.
- Conclusão: Resumo do projeto.

Arquitetura do Sistema



Funcionalidades do produto

Número	*Funcionalidade do sistema*
1	O jogo apresenta um menu inicial onde os jogadores podem iniciar o jogo ou sair do jogo.
2	O jogo oferece um modo de jogo onde dois jogadores podem competir entre si no clássico jogo de pingue-pongue.
3	s jogadores controlam as raquetes usando as teclas de seta (cima e baixo) e as teclas W e S no teclado, tornando o jogo acessível e fácil de jogar.
4	O jogo acompanha a pontuação de cada jogador à medida que eles marcam pontos durante o jogo.
5	O jogo tem um sistema para determinar o vencedor, onde o primeiro jogador a alcançar a pontuação definida como WINNING_SCORE (10, no código fornecido) ganha a partida.

Usuários e sistemas externos

Descrição

Número	*Usuários*	*Definição*
1	Bruno	Quer um game simples para jogar no tédio.
2	Rafael	Busca um jogo “retro” para relembrar sua infância.
3	Luana	Precisa de um game simples para jogar com sua filha.

Documentação do código

Documentação da Estrutura de dados geral do software

- Matrizes e Vetores:
- No código fornecido, não há uso direto de matrizes ou vetores para armazenar dados. No entanto, a linguagem C++ permite a criação e manipulação de matrizes e vetores para armazenar valores em uma estrutura tabular ou unidimensional, respectivamente. Essas estruturas podem ser utilizadas para armazenar informações adicionais, como uma matriz para representar o campo de jogo ou um vetor para armazenar dados históricos do jogo.
- Registros e Estruturas:
- Os registros não são diretamente utilizados no código fornecido. No entanto, a estrutura **`Paddle`** e **`Ball`** são exemplos de estruturas de dados personalizadas que armazenam informações relacionadas a raquetes e bola, respectivamente. As estruturas são usadas para agrupar variáveis relacionadas em uma única entidade, facilitando a manipulação e organização dos dados.
- Classes:
- Embora o código fornecido não inclua explicitamente o uso de classes, o conceito de classes é fundamental para a programação orientada a objetos em C++. O código pode ser organizado e estendido usando classes, onde cada objeto representa uma entidade com seu próprio conjunto de propriedades (variáveis) e comportamentos (métodos). Classes podem ser usadas para representar entidades adicionais no jogo, como jogadores, placar ou ambiente de jogo.
- Armazenamento e Manipulação de Dados:
- No código fornecido, os dados são armazenados em variáveis individuais dentro das estruturas **`Paddle`** e **`Ball`**. Essas variáveis são acessadas e manipuladas diretamente para atualizar as posições das raquetes e da bola. Os dados são armazenados na memória durante a execução do programa e podem ser modificados conforme necessário.

Função <checkCollision>

- `bool checkCollision(float x1, float y1, int width1, int height1, float x2, float y2, int width2, int height2)`
- Descrição: Verifica se ocorre colisão entre dois retângulos.
- Parâmetros:
- `x1` (float): Coordenada x do primeiro retângulo.
- `y1` (float): Coordenada y do primeiro retângulo.
- `width1` (int): Largura do primeiro retângulo.
- `height1` (int): Altura do primeiro retângulo.
- `x2` (float): Coordenada x do segundo retângulo.
- `y2` (float): Coordenada y do segundo retângulo.
- `width2` (int): Largura do segundo retângulo.
- `height2` (int): Altura do segundo retângulo.
- Retorno: Um valor booleano indicando se ocorre colisão (true) ou não (false).
- `void showWinner(ALLEGRO_FONT *font, const char *winnerText, ALLEGRO_BITMAP *background, ALLEGRO_DISPLAY *display, ALLEGRO_EVENT_QUEUE *event_queue)`

- Descrição: Exibe a tela de vitória para o jogador vencedor.
- Parâmetros:
- `font` (ALLEGRO_FONT*): Ponteiro para o objeto de fonte utilizado para exibir o texto.
- `winnerText` (const char*): Texto a ser exibido na tela de vitória.
- `background` (ALLEGRO_BITMAP*): Ponteiro para o objeto de imagem de fundo a ser exibido.
- `display` (ALLEGRO_DISPLAY*): Ponteiro para o objeto de display onde a tela de vitória será renderizada.
- `event_queue` (ALLEGRO_EVENT_QUEUE*): Ponteiro para a fila de eventos do Allegro.
- Retorno: Nenhum (void).

Testes do sistema

Casos de testes do sistema: função <nome da função>

Número	*Varáveis de Entrada*	*Valores válidos*	*Resultado Esperado*	*Valores inválidos*	*Resultado Esperado*
1	x1 = 50, y1 = 50, width1 = 30, height1 = 40, x2 = 60, y2 = 60, width2 = 20, height2 = 25	N/A (válido)	true	N/A (válido)	true
2	x1 = 10, y1 = 10, width1 = 100, height1 = 50, x2 = 0, y2 = 0, width2 = 50, height2 = 50	N/A (válido)	true	N/A (válido)	false
3	x1 = 0, y1 = 0, width1 = 50, height1 = 50, x2 = 100, y2 = 100, width2 = 50,	N/A (válido)	false	N/A (válido)	true

	height2 = 50				
4	x1 = 0, y1 = 0, width1 = 0, height1 = 0, x2 = 0, y2 = 0, width2 = 0, height2 = 0	N/A (válido)	false	N/A (válido)	false