UFRN - UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
IMD - INSTITUTO METRÓPOLE DIGITAL

FRANCISCO JÂNIO BATISTA XAVIER JUNIOR
LUIZ EDUARDO MARQUES SILVA
MANOEL MESSIAS DE BARROS JUNIOR
THAIAN ASSIS MORAES

PROJETO FINAL

Implementação do Jogo Drifts em C++

Natal/RN
2014

FRANCISCO JÂNIO BATISTA XAVIER JUNIOR
LUIZ EDUARDO MARQUES SILVA
MANOEL MESSIAS DE BARROS JUNIOR
THAIAN ASSIS MORAES

PROJETO FINAL
Implementação do Jogo Drifts em C++

Relatório técnico apresentado como requisito parcial
para obtenção de aprovação nas disciplinas de
Estrutura de Dados Básicas I e
Laboratório de Programação I,
no Curso de Bacharelado em Tecnologia da Informação,
na Universidade Federal do Rio Grande do Norte.

Prof. Dr.: João Carlos Xavier Junior
Prof. Dr.: Carlos Eduardo da Silva

Natal / RN
2014

RESUMO

Este trabalho apresenta como foi elaborado a implementação do jogo Drifts, proposto pelas disciplinas de Estrutura de Dados Básicas I e de Laboratório de Programação I, utilizando a linguagem C++ e a biblioteca Allegro 5.0. O objetivo é descrever algumas estruturas utilizadas que foram utilizadas no jogo, e que foram apreendidas nessas disciplinas.

Palavras-Chave: Drifts, C++, Allegro, biblioteca.

SUMÁRIO

1 INTRODUÇÃO

Drifts é um jogo flash desenvolvido pela empresa bobblebrook e está disponível no site: http://www.bobblebrook.com/games/drifts.

Nesse jogo você controla uma bola amarela com o cursor do mouse em uma área de tela. Seu objetivo é conseguir colar sua bola amarela nas bolas verdes que aparecem na tela, evitando tocar as bolas roxas que causam perda de vida. Quando você tiver mais de duas bolas verdes grudadas na sua bola amarela você pode tocar nas bolas azuis que converte as bolas verdes em pontos, quanto mais bolas verdes estiverem coladas na bola amarela maior será a pontuação.

Para a implementação desse jogo utilizando a linguagem de programação C++ e a biblioteca Allegro 5.0, utilizamos o ambiente de desenvolvimento integrado(IDE) Code::Blocks para compilar e executar nosso programa na plataforma Windows.

2 DESENVOLVIMENTO

Para desenvolver o jogo Drifts, nós inicialmente fizemos fluxogramas do comportamento geral do jogo desde a tela inicial até o fim do jogo. A partir desses fluxograma conseguimos ter uma ideia do que seria necessário fazer para criar o jogo. Em seguida, estudamos a biblioteca Allegro 5.0 e vimos o que era possível ser feito utilizando a linguagem C++.

Desenvolvemos esse jogo utilizando alguns contêineres e iteradores padrão, principalmente <list> e <vector>, dessa linguagem e de algumas variáveis e funções da biblioteca Allegro 5.0.

Essa biblioteca possui ferramentas suficientes para desenvolver uma interface gráfica, efeitos sonoros, eventos e timers. Além disso, ela é multi plataforma e funciona com diversos compiladores, o que facilitou nosso projeto, pois um código desenvolvido no Windows funciona também no Linux ou OS X, com pequenas exceções.

Nós modularizamos todo o código, pois dessa forma conseguimos ter um código mais legível com uma melhor manutenção.

2.1 IMPLEMENTAÇÃO

Nesta seção discutiremos a forma como o jogo foi implementado e os recursos utilizados para implementá-lo.

2.1.1 CONSTANTES E ESTRUTURAS

Inicialmente definimos algumas constantes do tipo inteiro para determinar o comportamento e a estrutura do jogo. Dentre elas estão:

1. altura e largura da tela(640x480), essa é a região de tela que nós iremos processar a imagem;
2. FPS = 60, para determinar a taxa de atualização 1/60 segundos;
3. Número de bolas verdes = 20: número indica a quantidade de bolas máximas que irão aparecer na tela;
4. Número de bolas azuis = 10: número que indica a quantidade de bolas azuis máxima que irão aparecer na tela;
5. Número de bolas roxas = 15: esse número é armazenado no conteiner <vector>, pois na medida que a dificuldade aumenta a quantidade de bolas roxas também aumenta;
6. Número de Estrela = 100: representa a quantidade máxima de estrelas no background;
7. Número de planos = 3: serve para indicar as três cores das estrelas no background.

Em seguida, criamos as seguintes estruturas para os objetos presentes no jogo:

1. CursorBola: para indicar a posição da bola amarela, x e y, e o nível, os pontos e a quantidade de vidas;
2. Bolas: representa as bolas verdes, azuis, e roxas, com a posição, x e y, a velocidade, e uma variável para indicar se a bola está ativa ou não;
3. Estrelas: para indicar a posição, x e y, e a velocidade das estrelas no background;
4. Botão: para manipular a posição, x e y, dos botões do jogo.

2.1.2 MÉTODOS E FUNÇÕES

Encontram-se na pasta do jogo, assim como a função principal (main) e o executável do jogo.

2.1.3 SOLUÇÃO ENCONTRADA

Nesta seção explicaremos a estrutura do jogo que implementamos.

- FLUXOGRAMA 1: TELAS DO JOGO

Este fluxograma mostra as telas utilizadas no jogo e como elas se interligam:

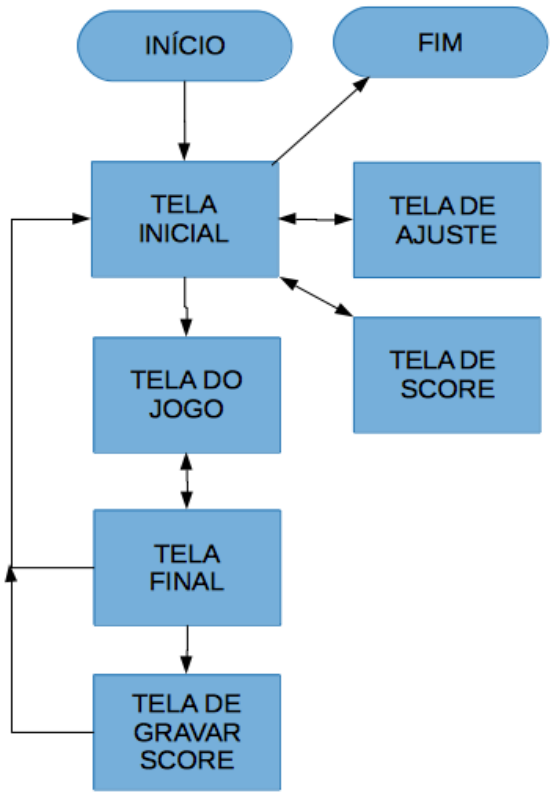

Imagem 1: telas.

Na tela inicial nós implementamos 4 botões: iniciar, ajuste, ranking e sair, ao clicar em um desses botões o usuário é redirecionado para a tela do jogo, tela de ajuste, tela de score ou sair do jogo, respectivamente. A tela inicial possuí também os nossos nomes e o nome que nós adotamos para o jogo: Space Drifts.

As telas de ajuste e de score possuem o mesmo botão que permite voltar a tela inicial. Na tela de ajuste é possível configurar o som para ligado ou desligado. Enquanto que na tela de score é exibido uma tabela contendo um top 10 das melhores pontuações.

A tela do jogo é contém a lógica do jogo, e por ser muito extensa a explicação foi dividida num sub-tópico.

A tela final possui 3 botões: gravar resultado, jogar novamente e sair, ao clicar em um desses botões o usuário é redirecionado para a tela de gravar score, tela do jogo, ou tela inicial, respectivamente. Na tela final também há um som que diz: "Game Over"; e também o nome Game Over.

A tela de gravar score possui uma tabela com o titulo ranking e dois tipos de dados Nome e Pontos, com 10 elementos. Em nome está computer que é preenchido com o nome que o usuário digitar, se nenhum nome for digito ficará um espaço vazio e a pontuação será registrada, caso contrário irá ficar o nome e a pontuação. Mesmo que o jogo seja fechado, a pontuação será mantida.

A tela de score mostra o que foi registrado na tela de gravar score, se nada tiver sido gravado, então aparecerá apenas no campo nome a sting computer e no campo pontos o valor 0.

Para o usuário sair do jogo é preciso que ele clique no botão sair ou no botão x que fica no canto superior esquerdo da tela do jogo.

- FLUXOGRAMA 2: LÓGICA DO JOGO

Este fluxograma mostra a lógica utilizada para implementar a tela do jogo:

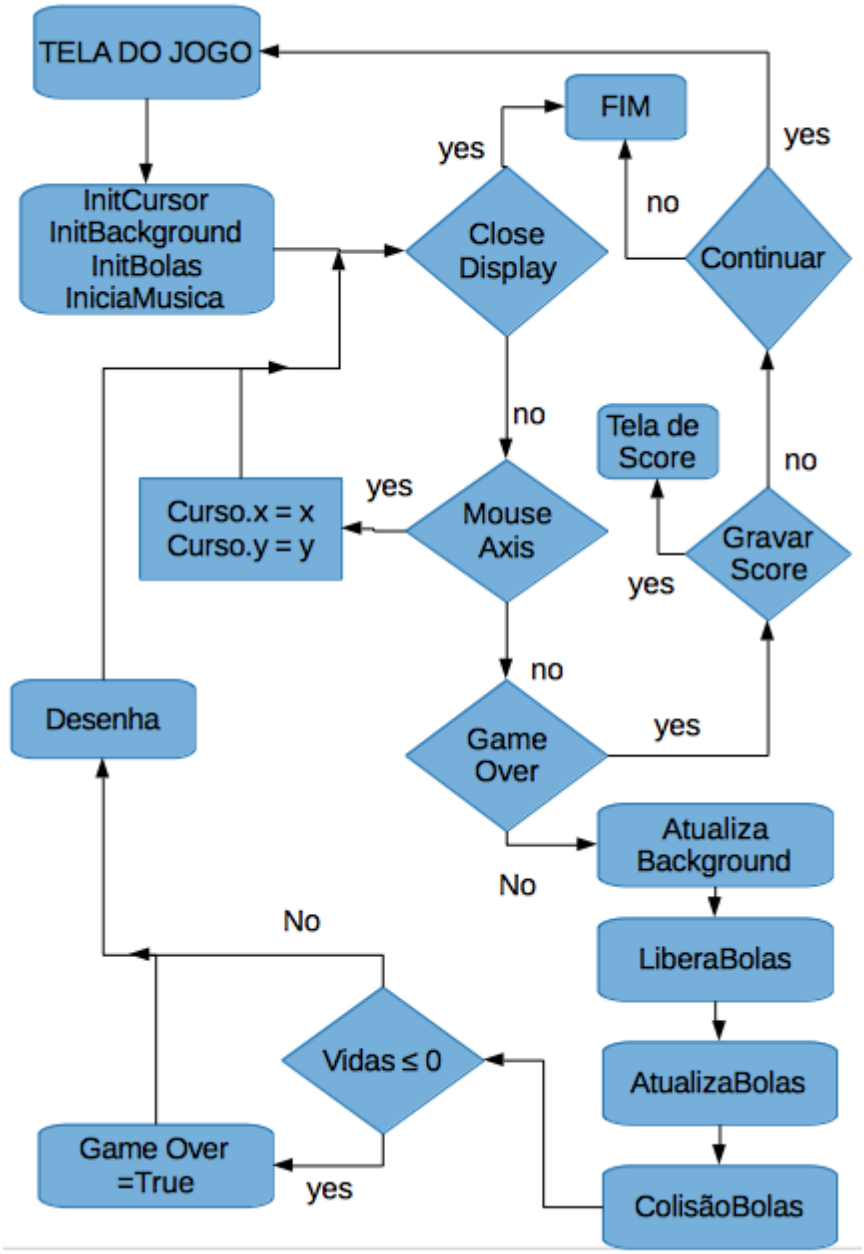

Imagem 2: Lógica do jogo.

Nessa imagem temos a implementação da lógica do jogo quando o usuário clicar no botão iniciar. Primeiramente, inicializamos o cursor(bola amarela, vida, pontuação e nível), as bolas: verdes, azuis, e roxa; a música e o background. Em cada um desses itens, nós utilizamos módulos.

Em seguida, criamos uma variável do tipo evento, ALLEGRO_EVENT, para verificar qual evento foi realizado pelo usuário. Se o evento for:

1. botão x do display acionado, então o jogo será fechado;
2. movimento do mouse, então a bola amarela recebe a posição do mouse;
3. tempo decorrido, então verifica Game Over.

Game Over é uma variável do tipo bool que nós utilizamos para verificar o fim do jogo. Se ela for verdadeira, então paramos a música com uma função do Allegro 5.0 e acionamos um som que diz: "Game Over", em seguida, verificamos qual botão o usuário clicou, se for:

1. gravar score, ele é redirecionado para a tela de gravar score, e sua pontuação é armazenada;
2. continuar, então o jogo ele é redirecionado para a tela do jogo, onde os registros de pontuação, nível e vida serão re-inicializados. Se ele não quiser continuar então ele pode voltar para o menu principal(esse botão está omitido na imagem 2) e sair do jogo ou fechar o jogo no clicando no botão x.

Se Game Over for falso, então atualizamos o background, isso faz com que o background se "mova", liberamos as bolas verdes, azuis, e/ou roxas, atualizamos essas bolas(realiza o movimento delas na tela), e verificamos se alguma dessas bolas colidiu com a bola amarela. Se a bola que colidiu na amarela for:

1. verde, então fazemos com que ela grude na bola amarela utilizando o conteiner <list> e seu iterador. O iterador neste caso serve para verificar qual das bolas verdes foram grudadas.
2. azul, então verificamos a quantidade de bolas verdes, se for maior que 2, então incrementamos a pontuação final com um valor e limpamos o conteiner <list>, senão apenas limpamos o conteiner <list> e seu iterador.
3. roxa, então limpamos o conteiner <list> e seu iterador e decrementamos uma vida.

Depois de verificarmos as colisões, verificamos a quantidade de vida, se ela for menor ou igual a zero, então atribuímos à Game Over o valor true e desenhamos na tela tudo que foi processado, senão apenas desenhamos tudo que foi processado.

2.4 RESULTADOS

Após terminamos a implementação, chegamos aos seguintes resultados:

Imagem 3: tela inicial.

Imagem 4: tela de ajuste.

Imagem 4.1: tela de ajuste.

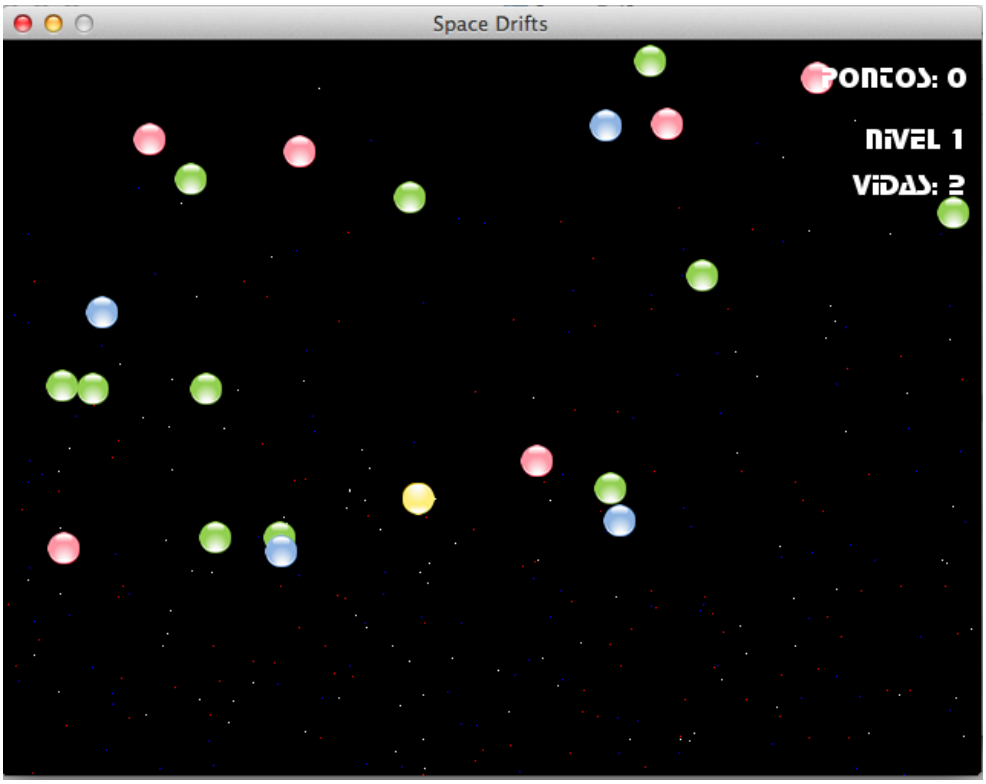

Imagem 5: tela do jogo.

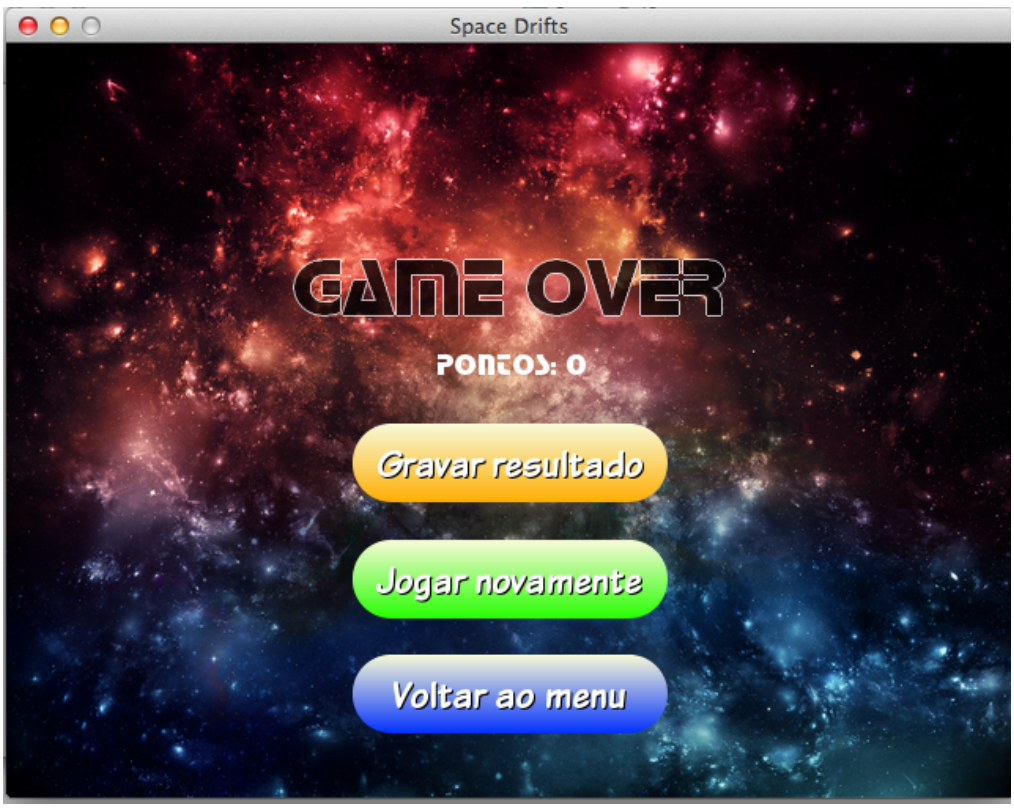

Imagem 6: tela final

Imagem 7: Tela de Score.

3 CONCLUSÃO

Concluímos que a linguagem C++ junto com outras bibliotecas, e uma IDE, constituí uma ferramenta poderosa para criação de jogos e outras aplicações do gênero. E constamos que seria difícil trabalhar isoladamente no desenvolvimento desse jogo.

Vimos que a modularização permitiu rápida manutenção e legibilidade do código. Mas percebemos que em certos momentos o tempo de execução do código ficou mais longo quando comparado com o código não modularizado. Mesmo assim optamos por modularizar por ser mais fácil de trabalhar em equipe.

REFERÊNCIAS

http://www.rafaeltoledo.net/tutoriais-allegro-5/

https://www.allegro.cc

http://www.cplusplus.com

http://www.stageweb.com.br/forum/10-c/21-re-instalando-o-codeblocks-10--allegro-5-no-win-7.html