



PRESS ENTER

Navegantes: Carolina Felix, Paulo Henrique, Gustavo Guidorizzi

Sumário

The background is a dark purple space scene. It features several brown, cratered asteroids of various sizes. Three blue comets with long tails are streaking across the sky. A large, detailed planet with a blue and white surface is visible in the lower right quadrant. Small white dots represent distant stars.

- I. Engine (Allegro 5)
- II. Implementação
 - A. Structs
 - B. Alocação dinâmica e ponteiros
 - C. Arquivos
 - D. Manipulação de Strings



[0]

Allegro 5

Alegria, alegria

Allegro (biblioteca)

15 línguas

Artigo [Discussão](#)

Ler [Editar](#) [Ver histórico](#) [Ferramentas](#)

Allegro é uma [biblioteca livre](#) de [código fonte aberto](#) para o desenvolvimento de [Video games](#).

O objetivo principal é a portabilidade entre as [plataformas](#) em que é feito uso da biblioteca. O mesmo [código-fonte](#) deve compilar e rodar em todas as plataformas suportadas. Um objetivo de curto prazo é a plataforma [64-bits](#).

O seu principal uso é no escopo da [programação](#) de [jogos](#). Atualmente ela possui uma grande comunidade, pois além de possuir diversos recursos nativamente (gráficos [2D](#) e [3D](#) com [OpenGL](#), entrada de dados pelo [teclado](#) e [mouse](#), RLE-Sprites, exibição de vídeos e controle de [som](#)) a [API](#) é bastante extensível fazendo que com existam diversos [addons](#) disponíveis.

A interface pública de acesso da [biblioteca](#) é escrita em [C](#), porém há algumas versões (não-oficiais) para outras [linguagem de programação](#). Internamente seu código fonte é escrito em uma mistura de [C](#), [Assembly](#) (drivers i386), [C++](#) e [Objective-C](#).

Historicamente, um ponto forte do Allegro era sua performance. Uma parcela de seu código fonte era escrita de forma [otimizada](#) em [Assembly](#) para situações em que tempo de processamento é fundamental. Quando seu código fonte deixou de ser escrito exclusivamente para [DOS](#) alguns desses processamentos críticos passaram a ser feitos pelo [sistema operacional](#), como por exemplo, com o uso interno da biblioteca [DirectX](#) no [Windows](#).

Allegro

Allegro

Desenvolvedor Colaboradores do projeto Allegro

Plataforma x86, x64, ARM (Android)

Lançamento 1990 (33–34 anos)

Versão estável 5.2.9.1 (20 de janeiro de 2024; há 4 meses)

Idioma(s) Inglês

Escrito em C

Sistema operacional Multiplataforma

Gênero(s) Biblioteca Gráfica

Licença Allegro 4 Giftware Allegro 5 Licença zlib

Estado do desenvolvimento Ativo

Página oficial [liballeg.org](#)

Repositório [allegro5](#) no [GitHub](#)

Cronologia

Allegro 4

allegro.cc

Manual

Contents

Configuration files
Display
Events
File I/O
Filesystem
Fixed point math
Graphics
Joystick
Keyboard
Memory
Mouse
Path
State
System
Threads
Time
Timer
Transformations
UTF-8
Miscellaneous
Platform-specific
Direct3D
OpenGL

Addons

Audio addon
Audio codecs
Color addon
Font addons
Image I/O addon
Memfile addon
Native dialogs addon
PhysicsFS addon
Primitives addon

Allegro 5.0 reference manual

- Getting started guide

API

- Configuration files
- Display routines
- Events
- File I/O
- Filesystem
- Fixed point math
- Graphics routines
- Joystick routines
- Keyboard routines
- Memory management
- Mouse routines
- Path structures
- State
- System routines
- Threads
- Time
- Timer
- Transformations
- UTF-8 string routines
- Miscellaneous
- Platform-specific
- Direct3D integration
- OpenGL integration

Addons

- Audio addon
- Audio codecs
- Color addon
- Font addons
- Image I/O addon
- Memfile addon
- Native dialogs addon
- PhysicsFS addon
- Primitives addon

Allegro.cc



Funções

Al_draw_text() escreve uma *string* em tela

Al_load_bitmap() carrega uma imagem para tela

Al_play_sample() reproduz um som/música

Al_flip_display() Atualiza e exibe o que está em tela

Al_get_bitmap_width() e **Al_get_bitmap_height()** Largura e Altura

Al_destroy()

• E muitas mais...

Allegro.cc





[1]

Structs

Agrupe seus dados!

Structs

1 Nave



2 Tiro "pei pei"



3 Monstro(s)



Mon: e eu ?

4 Score



```
typedef struct Nave{  
    int x;  
    int y;  
    int velocidade;  
    int borda_x;  
    int borda_y;  
    int pontos;  
    int vida;  
    bool ativo;  
}Nave;
```

```
typedef struct Monstro{  
    int x;  
    int y;  
    int velocidade;  
    int borda_x;  
    int borda_y;  
    int pontuacao;  
    float rotacao;  
    bool ativo;  
}Monstro;
```

```
typedef struct Tiro{  
    int x;  
    int y;  
    int velocidade;  
    bool ativo;  
    int dano;  
}Tiro;
```

```
typedef struct HiScore  
{  
    long int pontuacao[10];  
    int posicao[10];  
    char* nome[10][3];  
} HiScore;
```




[2]

Alocação dinâmica de memória e ponteiros

Põe e tira

```
HiScore* hiscore = (HiScore*)malloc(sizeof(HiScore) * NUM_HISCORES + 1);  
hiscore = armazenaHiscotes(arquivo_inicial, &hiscore);  
char* player_hiscore = "HAI";  
fclose(arquivo_hiscotes_atualizados);
```

```
int* pontuacao = (int*)malloc(sizeof(int));  
if (pontuacao != NULL) {  
    *pontuacao = 0;  
}
```

```
// Encerra o jogo, libera a memória  
free(hiscore);  
free(pontuacao);  
al_destroy_bitmap(bg);  
al_destroy_bitmap(sprite);  
al_destroy_bitmap(disparo);  
al_destroy_font(font);  
al_destroy_display(display);  
al_destroy_event_queue(event_queue);  
al_destroy_sample(sample);  
al_destroy_sample(sample_2);  
al_destroy_sample(sample_3);  
return 0;
```

```
//DEFINIÇÕES DE BITMAP
ALLEGRO_BITMAP* sprite = al_load_bitmap("./nave_resized.png");
if (!sprite) {
    fprintf(stderr, "Falha ao carregar o sprite da nave.\n");
    return -1;
}

ALLEGRO_BITMAP* inimigo = al_load_bitmap("./mon.png");
if (!inimigo) {
    fprintf(stderr, "Falha ao carregar o sprite do inimigo.\n");
    return -1;
}

ALLEGRO_BITMAP* disparo = al_load_bitmap("./disparo.png");
if (!disparo) {
    fprintf(stderr, "Falha ao carregar o sprite do disparo.\n");
    return -1;
}

ALLEGRO_BITMAP* bg = al_load_bitmap("./R.jpg");
if (!bg) { ...

ALLEGRO_BITMAP* bgGameOver = al_load_bitmap("./gameOver.jpg");
if (!bgGameOver) {
    fprintf(stderr, "Falha ao carregar o background do GameOver.\n");
    return -1;
}

ALLEGRO_BITMAP* bgMenu = al_load_bitmap("./bgMenu.png");

if (!bgMenu) {
    fprintf(stderr, "Falha ao carregar o background do Menu.\n");
    return -1;
}
```

```
al_destroy_bitmap(bg);
al_destroy_bitmap(sprite);
al_destroy_bitmap(disparo);
al_destroy_font(font);
al_destroy_display(display);
al_destroy_event_queue(event_queue);
al_destroy_sample(sample);
al_destroy_sample(sample_2);
al_destroy_sample(sample_3);
```

The background is a dark purple space scene. It features several stylized celestial bodies: a large yellow planet with brown spots in the upper left, a smaller yellow planet with brown spots in the upper center, and several brown, irregularly shaped asteroids scattered throughout. Two comets with long, light blue tails are also visible. The background is decorated with numerous small white dots representing stars. In the center, there are three concentric circles in shades of brown and purple.

[3]

Arquivos

Vamos nos lembrar

```

72
73 void
74 gravaHiscores(HiScore** hiscore)
75 {
76
77     FILE* arquivo_hiscores;
78     int i;
79     HiScore* hiscore_local = *hiscore;
80
81     arquivo_hiscores = fopen("hiscores_atuais.txt", "w");
82     for (i = 0; i < NUM_HIScores; i++)
83     {
84         fprintf(arquivo_hiscores, "%d %s %ld\n", i,
85                 (char*) hiscore_local->nome[i], hiscore_local->pontuacao[i]);
86     }
87     fclose(arquivo_hiscores);
88 }

```

```

57 HiScore*
58 armazenaHiscores(FILE* arquivo, HiScore** hiscore)
59 {
60     int i;
61     HiScore* hiscore_local = *hiscore;
62
63     for (i = 0; i < NUM_HIScores; i++)
64     {
65         fscanf(arquivo, "%d %s %ld\n", &hiscore_local->posicao[i],
66                (char*) hiscore_local->nome[i], &hiscore_local->pontuacao[i]);
67         //printf ("%d %s %ld\n", hiscore_local->posicao[i],
68                //      hiscore_local->nome[i], hiscore_local->pontuacao[i]);
69     }
70     return hiscore_local;
71 }

```



[4]

Manipulação de *strings*

Escreva coisas bonitas

```
HiScore* hiscore_local = *hiscore;
//printf("%s", (char*) hiscore_local->nome[9]);

// Tem um bugzinho aqui
for (i = NUM_HISCORES; i > posicao; i--)
{
    (long int) hiscore_local->pontuacao[i] = (long int) hiscore_local->pontuacao[i-1];
    strcpy((char*)hiscore_local->nome[i], (char*)hiscore_local->nome[i-1]);
}

// Inclui o novo hiscore
if (posicao < NUM_HISCORES) {
    hiscore_local->pontuacao[posicao] = *pontuacao;
    strcpy((char*)hiscore_local->nome[posicao], (char*)nome);
}
```

```
HiScore*
armazenaHiscres(FILE* arquivo, HiScore** hiscore)
{
    int i;
    HiScore* hiscore_local = *hiscore;

    for (i = 0; i < NUM_HISCORES; i++)
    {
        fscanf(arquivo, "%d %s %ld\n", &hiscore_local->posicao[i],
            (char*) hiscore_local->nome[i], &hiscore_local->pontuacao[i]);
        //printf ("%d %s %ld\n", hiscore_local->posicao[i],
            //      hiscore_local->nome[i], hiscore_local->pontuacao[i]);
    }
    return hiscore_local;
}
```

Agradecimentos

Agradecimento especial ao nosso colega

Paulo Victor

- por nos ajudar com algumas configurações que foram essenciais para o desenvolvimento do projeto.

Agradecimento especial ao irmão da Carol

Gabriel Felix

por nos ajudar a fazer uma adaptação para a tela inicial do jogo.

GitHub do Código



<https://github.com/OPauloss/Jogo-LP>



Muito obrigado!

free(LP1);

