Maquina de refrigerantes com Allegro 5

Generated by Doxygen 1.9.8

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 AnimacaoEntrega Struct Reference

Armazena os dados para a animação de entrega do produto ao cliente.

Data Fields

- ALLEGRO_BITMAP ∗ **bitmap**

  Bitmap do produto sendo entregue.
- float **x**
- float **y**
- float **escala**
- float **alpha**

  Posição, escala e transparência (alpha) atuais.
- float **velocidade_x**
- float **velocidade_escala**

  Velocidades de movimento e de alteração da escala.
- float **escala_final**

  Escala final para a animação.
- bool **ativa**

  Flag que indica se a animação está em andamento.

3.1.1 Detailed Description

Armazena os dados para a animação de entrega do produto ao cliente.

The documentation for this struct was generated from the following file:

- main.c

3.2 AnimacaoQueda Struct Reference

Armazena os dados para a animação de queda de um produto.

Data Fields

- ALLEGRO_BITMAP * **bitmap**

  Bitmap do produto que está caindo.
- float **x**
- float **y**
- float **angulo**

  Posição (x, y) e ângulo de rotação atuais.
- float **velocidade_y**

  Velocidade de queda no eixo Y.
- float **velocidade_angulo**

  Velocidade de rotação.
- float **y_final**

  Posição Y onde a animação termina.
- bool **ativa**

  Flag que indica se a animação está em andamento.

3.2.1 Detailed Description

Armazena os dados para a animação de queda de um produto.

The documentation for this struct was generated from the following file:

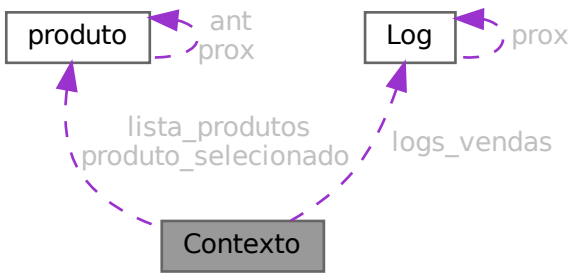
- main.c

3.3 Contexto Struct Reference

Estrutura de contexto que armazena o estado global da aplicação.

`#include <config.h>`

Collaboration diagram for Contexto:



Data Fields

- Produto * lista_produtos
- Log * logs_vendas
- Produto * produto_selecionado

3.3.1 Detailed Description

Estrutura de contexto que armazena o estado global da aplicação.

Agrupa os ponteiros para as principais estruturas de dados utilizadas pela máquina de refrigerantes.

3.3.2 Field Documentation

3.3.2.1 lista_produtos

```
Produto* lista_produtos
```

Ponteiro para a cabeça da lista de produtos.

3.3.2.2 logs_vendas

```
Log* logs_vendas
```

Ponteiro para a cabeça da lista de logs de vendas.

3.3.2.3 produto_selecionado

```
Produto* produto_selecionado
```

Ponteiro para o produto atualmente selecionado para compra.

The documentation for this struct was generated from the following file:

- config.h

3.4 EstatisticasData Struct Reference

Estrutura para armazenar os dados das estatísticas calculadas.

```
#include <estatisticas.h>
```

Data Fields

- int **total_vendas**
- float **valor_arrecadado**
- float **valor_medio**
- int **id_mais_vendido**
- int **qtd_mais_vendido**

3.4.1 Detailed Description

Estrutura para armazenar os dados das estatísticas calculadas.

The documentation for this struct was generated from the following file:

- estatisticas.h

3.5 Fila Struct Reference

Estrutura para representar uma fila circular de propagandas.

`#include <fila.h>`

Data Fields

- char mensagens [MAX_PROPAGANDAS][TAMANHO_MAX_MSG]
- int frente
- int tras
- int tamanho

3.5.1 Detailed Description

Estrutura para representar uma fila circular de propagandas.

3.5.2 Field Documentation

3.5.2.1 frente

`int frente`

Índice da frente da fila.

3.5.2.2 mensagens

`char mensagens[MAX_PROPAGANDAS][TAMANHO_MAX_MSG]`

Array de strings para as mensagens.

3.5.2.3 tamanho

```
int tamanho
```

Número de elementos na fila.

3.5.2.4 tras

```
int tras
```

Índice da traseira da fila.

The documentation for this struct was generated from the following file:

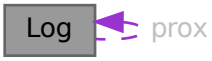
- fila.h

3.6 Log Struct Reference

Estrutura para representar um registro (log) de uma venda.

```
#include <log_vendas.h>
```

Collaboration diagram for Log:



Data Fields

- char timestamp [20]
- int produto_id
- float valor_pago
- float troco
- struct Log * prox

3.6.1 Detailed Description

Estrutura para representar um registro (log) de uma venda.

3.6.2 Field Documentation

3.6.2.1 produto_id

```
int produto_id
```

ID do produto vendido.

3.6.2.2 prox

```
struct Log* prox
```

Ponteiro para o próximo log na lista.

3.6.2.3 timestamp

```
char timestamp[20]
```

Data e hora da transação.

3.6.2.4 troco

```
float troco
```

Valor do troco devolvido.

3.6.2.5 valor_pago

```
float valor_pago
```

Valor total pago pelo cliente.

The documentation for this struct was generated from the following file:

- log_vendas.h

3.7 Pilha Struct Reference

Estrutura para a pilha que armazena as moedas (valores float).

```
#include <pilha.h>
```

Data Fields

- float valores [MAX_PILHA]
- int topo

3.7.1 Detailed Description

Estrutura para a pilha que armazena as moedas (valores float).

3.7.2 Field Documentation

3.7.2.1 topo

```
int topo
```

Índice do topo da pilha.

3.7.2.2 valores

```
float valores[MAX_PILHA]
```

Array para armazenar os valores.

The documentation for this struct was generated from the following file:

- pilha.h

3.8 PilhaEstados Struct Reference

Estrutura para a pilha que armazena o histórico de estados da FSM.

```
#include <pilha.h>
```

Data Fields

- State estados [MAX_PILHA]
- int topo

3.8.1 Detailed Description

Estrutura para a pilha que armazena o histórico de estados da FSM.

3.8.2 Field Documentation

3.8.2.1 estados

```
State estados[MAX_PILHA]
```

Array para armazenar os estados.

3.8.2.2 topo

```
int topo
```

Índice do topo da pilha.

The documentation for this struct was generated from the following file:

- pilha.h

3.9 produto Struct Reference

Estrutura para representar um produto na máquina de refrigerantes.

```
#include <lista_produtos.h>
```

Collaboration diagram for produto:

Data Fields

- int id
- char nome [50]
- float preco
- int estoque
- char imagem_path [100]
- float pos_x_inicial
- float pos_y_inicial
- ALLEGRO_BITMAP * imagem_animacao
- struct produto * prox
- struct produto * ant

3.9.1 Detailed Description

Estrutura para representar um produto na máquina de refrigerantes.

3.9.2 Field Documentation

3.9.2.1 ant

```
struct produto* ant
```

Ponteiro para o produto anterior na lista.

3.9.2.2 estoque

```
int estoque
```

Quantidade disponível em estoque.

3.9.2.3 id

```
int id
```

Identificador único do produto.

3.9.2.4 imagem_animacao

```
ALLEGRO_BITMAP* imagem_animacao
```

Ponteiro para o bitmap da animação carregado.

3.9.2.5 imagem_path

```
char imagem_path[100]
```

Caminho do arquivo de imagem para a animação.

3.9.2.6 nome

```
char nome[50]
```

Nome do produto.

3.9.2.7 pos_x_inicial

```
float pos_x_inicial
```

Posição X inicial para a animação de queda.

3.9.2.8 pos_y_inicial

```
float pos_y_inicial
```

Posição Y inicial para a animação de queda.

3.9.2.9 preco

```
float preco
```

Preço do produto em reais.

3.9.2.10 prox

```
struct produto* prox
```

Ponteiro para o próximo produto na lista.

The documentation for this struct was generated from the following file:

- lista_produtos.h

Chapter 4

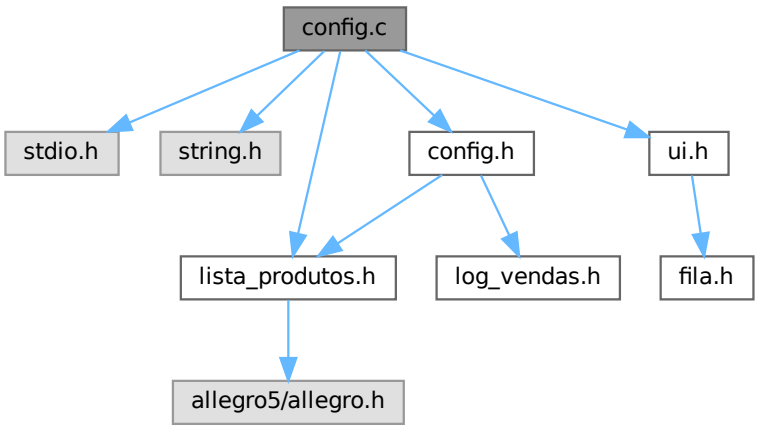
File Documentation

4.1 config.c File Reference

Implementação das funcionalidades do menu de configuração.

```
#include <stdio.h>
#include <string.h>
#include "config.h"
#include "lista_produtos.h"
#include "ui.h"
```

Include dependency graph for config.c:



Functions

- int verificar_senha ()

  Solicita e verifica a senha do administrador.
- void menu_configuracao (Contexto *ctx)

  Exibe e gerencia o menu de configuração.

4.1.1 Detailed Description

Implementação das funcionalidades do menu de configuração.

Note

Este módulo foi projetado para uma versão de console da aplicação. Na versão gráfica final (main.c), toda a interface e lógica de configuração foram reimplementadas com Allegro, e as funções deste arquivo (exceto `verificar_senha`, cuja lógica foi adaptada) não são utilizadas.

4.1.2 Function Documentation

4.1.2.1 menu_configuracao()

```
void menu_configuracao (
            Contexto * ctx )
```

Exibe e gerencia o menu de configuração.

Permite que o administrador adicione, remova e liste produtos. O acesso a este menu é protegido por senha.

Parameters

in,out	*ctx*	Ponteiro para o contexto da aplicação, que será modificado.

4.1.2.2 verificar_senha()

```
int verificar_senha ( )
```

Solicita e verifica a senha do administrador.

Permite até 3 tentativas antes de bloquear o acesso.

Returns

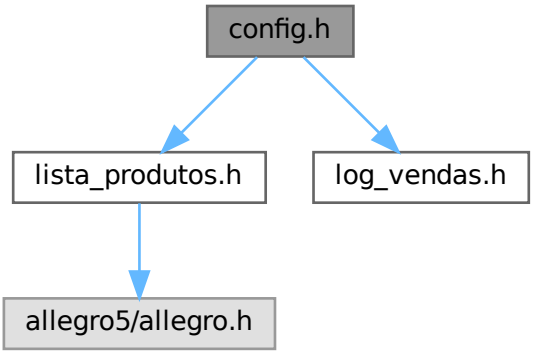
1 se a senha estiver correta, 0 caso contrário.

4.2 config.h File Reference

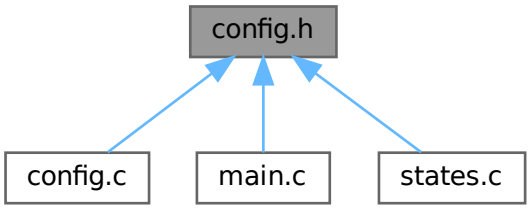
Definições e protótipos para o módulo de configuração da máquina.

```
#include "lista_produtos.h"
#include "log_vendas.h"
```

Include dependency graph for config.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct Contexto

  Estrutura de contexto que armazena o estado global da aplicação.

Macros

- #define **SENHA_PADRAO** "admin123"

  Senha padrão para acesso ao menu de configuração.

Functions

- int verificar_senha ()

  Solicita e verifica a senha do administrador.
- void menu_configuracao (Contexto *ctx)

  Exibe e gerencia o menu de configuração.

4.2.1 Detailed Description

Definições e protótipos para o módulo de configuração da máquina.

Este arquivo contém a estrutura de contexto da aplicação, a senha de administrador e os protótipos das funções relacionadas ao menu de configuração.

4.2.2 Function Documentation

4.2.2.1 menu_configuracao()

```
void menu_configuracao (
            Contexto * ctx )
```

Exibe e gerencia o menu de configuração.

Permite que o administrador adicione, remova e liste produtos. O acesso a este menu é protegido por senha.

Parameters

in,out	*ctx*	Ponteiro para o contexto da aplicação, que será modificado.

4.2.2.2 verificar_senha()

```
int verificar_senha ( )
```

Solicita e verifica a senha do administrador.

- Permite até 3 tentativas antes de bloquear o acesso.

  Returns

  1 se a senha estiver correta, 0 caso contrário.

  Permite até 3 tentativas antes de bloquear o acesso.

  Returns

  1 se a senha estiver correta, 0 caso contrário.

4.3 config.h

Go to the documentation of this file.

```
00001
00010 #ifndef CONFIG_H
00011 #define CONFIG_H
00012
00013 #include "lista_produtos.h"
00014 #include "log_vendas.h"
00015
00019 #define SENHA_PADRAO "admin123"
00020
00027 typedef struct {
00028     Produto* lista_produtos;
```

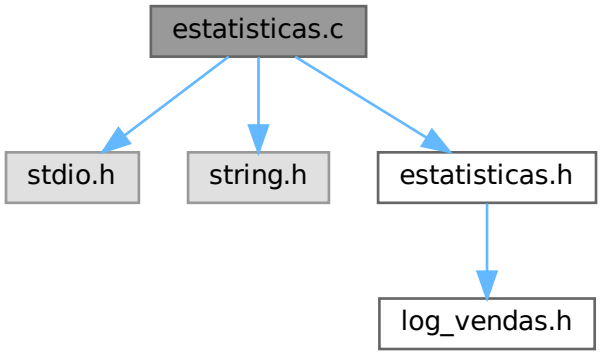
```
00029     Log* logs_vendas;
00030     Produto* produto_selecionado;
00031 } Contexto;
00032
00038 int verificar_senha();
00039
00047 void menu_configuracao(Contexto *ctx);
00048
00049 #endif
```

4.4 estatisticas.c File Reference

Implementação das funções de geração de estatísticas de vendas.

```
#include <stdio.h>
#include <string.h>
#include "estatisticas.h"
```

Include dependency graph for estatisticas.c:



Functions

- void calcular_estatisticas (Log *logs, EstatisticasData *data)
  Calcula as estatísticas com base nos logs e preenche uma struct.
- void salvar_estatisticas_csv (Log *logs, const char *filename)
  Salva as estatísticas de vendas em um arquivo CSV.

4.4.1 Detailed Description

Implementação das funções de geração de estatísticas de vendas.

4.4.2 Function Documentation

4.4.2.1 calcular_estatisticas()

```
void calcular_estatisticas (
            Log * logs,
            EstatisticasData * data )
```

Calcula as estatísticas com base nos logs e preenche uma struct.

Parameters

in	*logs*	Ponteiro para a cabeça da lista de logs de vendas.
out	*data*	Ponteiro para a struct que será preenchida com os dados.

4.4.2.2 salvar_estatisticas_csv()

```
void salvar_estatisticas_csv (
            Log * logs,
            const char * filename )
```

Salva as estatísticas de vendas em um arquivo CSV.

Parameters

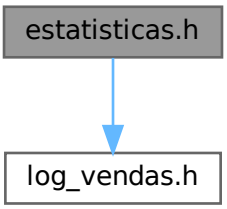
in	*logs*	Ponteiro para a cabeça da lista de logs de vendas.
in	*filename*	O nome do arquivo CSV onde as estatísticas serão salvas.

4.5 estatisticas.h File Reference

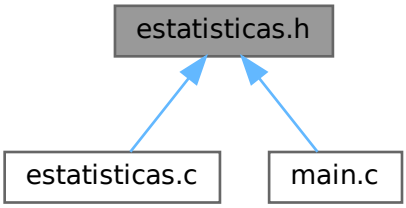
Protótipos e estruturas para as funções de geração de estatísticas.

```
#include "log_vendas.h"
```

Include dependency graph for estatisticas.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct EstatisticasData

  Estrutura para armazenar os dados das estatísticas calculadas.

Functions

- void calcular_estatisticas (Log ∗logs, EstatisticasData ∗data)

  Calcula as estatísticas com base nos logs e preenche uma struct.
- void salvar_estatisticas_csv (Log ∗logs, const char ∗filename)

  Salva as estatísticas de vendas em um arquivo CSV.

4.5.1 Detailed Description

Protótipos e estruturas para as funções de geração de estatísticas.

4.5.2 Function Documentation

4.5.2.1 calcular_estatisticas()

```
void calcular_estatisticas (
            Log * logs,
            EstatisticasData * data )
```

Calcula as estatísticas com base nos logs e preenche uma struct.

Parameters

in	*logs*	Ponteiro para a cabeça da lista de logs de vendas.
out	*data*	Ponteiro para a struct que será preenchida com os dados.

4.5.2.2 salvar_estatisticas_csv()

```
void salvar_estatisticas_csv (
            Log * logs,
            const char * filename )
```

Salva as estatísticas de vendas em um arquivo CSV.

Parameters

in	*logs*	Ponteiro para a cabeça da lista de logs de vendas.
in	*filename*	O nome do arquivo CSV onde as estatísticas serão salvas.

4.6 estatisticas.h

Go to the documentation of this file.

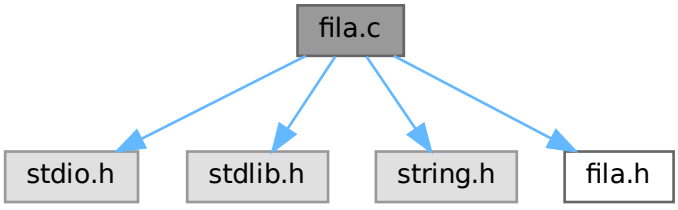
```
00001
00006 #ifndef ESTATISTICAS_H
00007 #define ESTATISTICAS_H
00008
00009 #include "log_vendas.h"
00010
00014 typedef struct {
00015     int total_vendas;
00016     float valor_arrecadado;
00017     float valor_medio;
00018     int id_mais_vendido;
00019     int qtd_mais_vendido;
00020 } EstatisticasData;
00021
00027 void calcular_estatisticas(Log* logs, EstatisticasData* data);
00028
00034 void salvar_estatisticas_csv(Log* logs, const char* filename);
00035
00036 #endif
```

4.7 fila.c File Reference

Implementação da estrutura de dados de fila circular.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "fila.h"
```

Include dependency graph for fila.c:



Functions

- void inicializar_fila (Fila *fila)

  Inicializa a fila.
- void enfileirar (Fila *fila, const char *mensagem)

  Adiciona uma mensagem ao final da fila (enfileirar).
- const char * desenfileirar (Fila *fila)

  Remove e retorna a mensagem da frente da fila (desenfileirar).
- const char * frente_fila (Fila *fila)

  Retorna a mensagem na frente da fila sem removê-la.

4.7.1 Detailed Description

Implementação da estrutura de dados de fila circular.

4.7.2 Function Documentation

4.7.2.1 desenfileirar()

```
const char * desenfileirar (
            Fila * fila )
```

Remove e retorna a mensagem da frente da fila (desenfileirar).

Parameters

in,out	*fila*	Ponteiro para a fila.

Returns

Ponteiro para a mensagem removida, ou NULL se a fila estiver vazia.

4.7.2.2 enfileirar()

```
void enfileirar (
            Fila * fila,
            const char * mensagem )
```

Adiciona uma mensagem ao final da fila (enfileirar).

Parameters

in,out	*fila*	Ponteiro para a fila.
in	*mensagem*	Mensagem a ser adicionada.

4.7.2.3 frente_fila()

```
const char * frente_fila (
            Fila * fila )
```

Retorna a mensagem na frente da fila sem removê-la.

Parameters

in	*fila*	Ponteiro para a fila.

Returns

Ponteiro para a mensagem da frente, ou NULL se a fila estiver vazia.

4.7.2.4 inicializar_fila()

```
void inicializar_fila (
            Fila * fila )
```

Inicializa a fila.

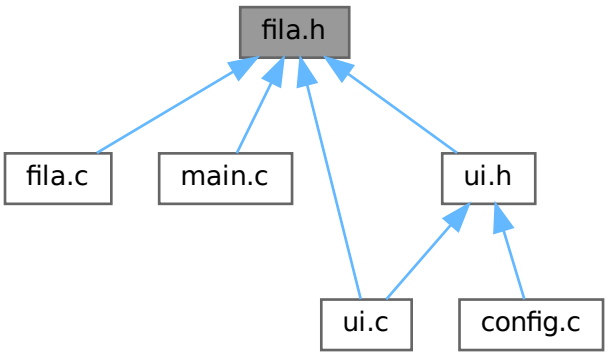
Parameters

out	*fila*	Ponteiro para a fila a ser inicializada.

4.8 fila.h File Reference

Definições e protótipos para uma estrutura de dados de fila circular.

This graph shows which files directly or indirectly include this file:



Data Structures

- struct Fila

  Estrutura para representar uma fila circular de propagandas.

Macros

- #define **MAX_PROPAGANDAS** 10
- #define **TAMANHO_MAX_MSG** 100

Typedefs

- typedef struct Fila **Fila**

  Estrutura para representar uma fila circular de propagandas.

Functions

- void inicializar_fila (Fila *fila)

  Inicializa a fila.
- void enfileirar (Fila *fila, const char *mensagem)

  Adiciona uma mensagem ao final da fila (enfileirar).
- const char * desenfileirar (Fila *fila)

  Remove e retorna a mensagem da frente da fila (desenfileirar).
- const char * frente_fila (Fila *fila)

  Retorna a mensagem na frente da fila sem removê-la.

4.8.1 Detailed Description

Definições e protótipos para uma estrutura de dados de fila circular.

- Usada para gerenciar as propagandas da máquina.

4.8.2 Function Documentation

4.8.2.1 desenfileirar()

```
const char * desenfileirar (
            Fila * fila )
```

Remove e retorna a mensagem da frente da fila (desenfileirar).

Parameters

in,out	*fila*	Ponteiro para a fila.

Returns

Ponteiro para a mensagem removida, ou NULL se a fila estiver vazia.

4.8.2.2 enfileirar()

```
void enfileirar (
            Fila * fila,
            const char * mensagem )
```

Adiciona uma mensagem ao final da fila (enfileirar).

Parameters

in,out	*fila*	Ponteiro para a fila.
in	*mensagem*	Mensagem a ser adicionada.

4.8.2.3 frente_fila()

```
const char * frente_fila (
            Fila * fila )
```

Retorna a mensagem na frente da fila sem removê-la.

Parameters

in	*fila*	Ponteiro para a fila.

Returns

Ponteiro para a mensagem da frente, ou NULL se a fila estiver vazia.

4.8.2.4 inicializar_fila()

```
void inicializar_fila (
            Fila * fila )
```

Inicializa a fila.

Parameters

out	*fila*	Ponteiro para a fila a ser inicializada.

4.9 fila.h

Go to the documentation of this file.

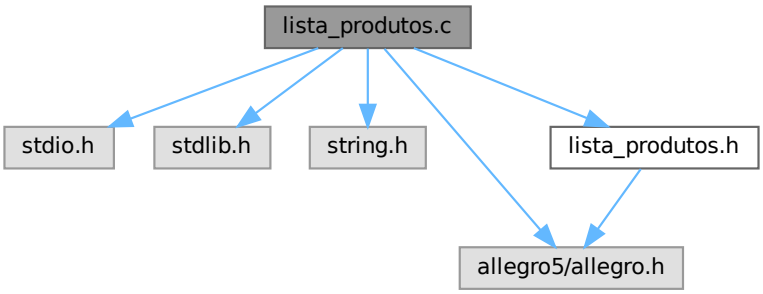
```
00001
00007 #ifndef FILA_H
00008 #define FILA_H
00009
00010 #define MAX_PROPAGANDAS 10
00011 #define TAMANHO_MAX_MSG 100
00012
00016 typedef struct Fila {
00017     char mensagens[MAX_PROPAGANDAS][TAMANHO_MAX_MSG];
00018     int frente;
00019     int tras;
00020     int tamanho;
00021 } Fila;
00022
00027 void inicializar_fila(Fila* fila);
00028
00034 void enfileirar(Fila* fila, const char* mensagem);
00035
00041 const char* desenfileirar(Fila* fila);
00042
00048 const char* frente_fila(Fila* fila);
00049
00050 #endif
```

4.10 lista_produtos.c File Reference

Implementação das funções de gerenciamento da lista de produtos.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <allegro5/allegro.h>
#include "lista_produtos.h"
```

Include dependency graph for lista_produtos.c:



Functions

- Produto * adicionar_produto (Produto *head, int id, const char *nome, float preco, int estoque, const char *img_path, float pos_x, float pos_y)

  Adiciona um novo produto ao início da lista duplamente encadeada.
- Produto * remover_produto (Produto *head, int id)

  Remove um produto da lista pelo seu ID.
- Produto * buscar_produto (Produto *head, int id)

  Busca um produto na lista pelo seu ID.

- void listar_produtos (Produto *head)

  Lista todos os produtos da lista no terminal.
- void liberar_lista_produtos (Produto *head)

  Libera toda a memória alocada para a lista de produtos.
- void salvar_produtos_csv (Produto *head, const char *filename)

  Salva a lista de produtos em um arquivo no formato CSV.
- Produto * carregar_produtos_csv (const char *filename)

  Carrega uma lista de produtos a partir de um arquivo CSV.

4.10.1 Detailed Description

Implementação das funções de gerenciamento da lista de produtos.

4.10.2 Function Documentation

4.10.2.1 adicionar_produto()

```
Produto * adicionar_produto (
            Produto * head,
            int id,
            const char * nome,
            float preco,
            int estoque,
            const char * img_path,
            float pos_x,
            float pos_y )
```

Adiciona um novo produto ao início da lista duplamente encadeada.

Parameters

in	*head*	Ponteiro para a cabeça da lista.
in	*id*	ID do novo produto.
in	*nome*	Nome do novo produto.
in	*preco*	Preço do novo produto.
in	*estoque*	Quantidade em estoque do novo produto.
in	*img_path*	Caminho do arquivo de imagem para a animação.
in	*pos_x*	Posição X inicial da animação.
in	*pos_y*	Posição Y inicial da animação.

Returns

Ponteiro para a nova cabeça da lista.

4.10.2.2 buscar_produto()

```
Produto * buscar_produto (
            Produto * head,
            int id )
```

Busca um produto na lista pelo seu ID.

Parameters

in	*head*	Ponteiro para a cabeça da lista.
in	*id*	ID do produto a ser buscado.

Returns

Ponteiro para o nó do produto encontrado, ou NULL se não for encontrado.

4.10.2.3 carregar_produtos_csv()

```
Produto * carregar_produtos_csv (
            const char * filename )
```

Carrega uma lista de produtos a partir de um arquivo CSV.

Parameters

in	*filename*	Nome do arquivo CSV a ser lido.

Returns

Ponteiro para a cabeça da nova lista de produtos carregada.

4.10.2.4 liberar_lista_produtos()

```
void liberar_lista_produtos (
            Produto * head )
```

Libera toda a memória alocada para a lista de produtos.

Parameters

in	*head*	Ponteiro para a cabeça da lista a ser liberada.

4.10.2.5 listar_produtos()

```
void listar_produtos (
            Produto * head )
```

Lista todos os produtos da lista no terminal.

Parameters

in	*head*	Ponteiro para a cabeça da lista.

4.10.2.6 remover_produto()

```
Produto * remover_produto (
            Produto * head,
            int id )
```

Remove um produto da lista pelo seu ID.

Parameters

in	*head*	Ponteiro para a cabeça da lista.
in	*id*	ID do produto a ser removido.

Returns

Ponteiro para a nova cabeça da lista.

4.10.2.7 salvar_produtos_csv()

```
void salvar_produtos_csv (
            Produto * head,
            const char * filename )
```

Salva a lista de produtos em um arquivo no formato CSV.

Parameters

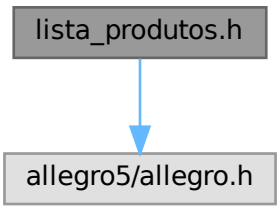
in	*head*	Ponteiro para a cabeça da lista.
in	*filename*	Nome do arquivo CSV de destino.

4.11 lista_produtos.h File Reference

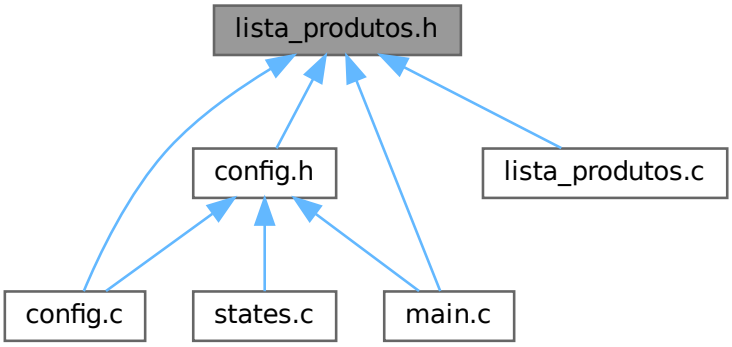
Definições e protótipos para a lista duplamente encadeada de produtos.

```
#include <allegro5/allegro.h>
```

Include dependency graph for lista_produtos.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct produto

  Estrutura para representar um produto na máquina de refrigerantes.

Typedefs

- typedef struct produto **Produto**

  Estrutura para representar um produto na máquina de refrigerantes.

Functions

- Produto ∗ adicionar_produto (Produto ∗head, int id, const char ∗nome, float preco, int estoque, const char ∗img_path, float pos_x, float pos_y)

  Adiciona um novo produto ao início da lista duplamente encadeada.
- Produto ∗ remover_produto (Produto ∗head, int id)

  Remove um produto da lista pelo seu ID.
- Produto ∗ buscar_produto (Produto ∗head, int id)

  Busca um produto na lista pelo seu ID.
- void listar_produtos (Produto ∗head)

  Lista todos os produtos da lista no terminal.
- void liberar_lista_produtos (Produto ∗head)

  Libera toda a memória alocada para a lista de produtos.
- void salvar_produtos_csv (Produto ∗head, const char ∗filename)

  Salva a lista de produtos em um arquivo no formato CSV.
- Produto ∗ carregar_produtos_csv (const char ∗filename)

  Carrega uma lista de produtos a partir de um arquivo CSV.

4.11.1 Detailed Description

Definições e protótipos para a lista duplamente encadeada de produtos.

4.11.2 Function Documentation

4.11.2.1 adicionar_produto()

```
Produto * adicionar_produto (
            Produto * head,
            int id,
            const char * nome,
            float preco,
            int estoque,
            const char * img_path,
            float pos_x,
            float pos_y )
```

Adiciona um novo produto ao início da lista duplamente encadeada.

Parameters

in	*head*	Ponteiro para a cabeça da lista.
in	*id*	ID do novo produto.
in	*nome*	Nome do novo produto.
in	*preco*	Preço do novo produto.
in	*estoque*	Quantidade em estoque do novo produto.
in	*img_path*	Caminho do arquivo de imagem para a animação.
in	*pos_x*	Posição X inicial da animação.
in	*pos_y*	Posição Y inicial da animação.

Returns

Ponteiro para a nova cabeça da lista.

4.11.2.2 buscar_produto()

```
Produto * buscar_produto (
            Produto * head,
            int id )
```

Busca um produto na lista pelo seu ID.

Parameters

in	*head*	Ponteiro para a cabeça da lista.
in	*id*	ID do produto a ser buscado.

Returns

Ponteiro para o nó do produto encontrado, ou NULL se não for encontrado.

4.11.2.3 carregar_produtos_csv()

```
Produto * carregar_produtos_csv (
            const char * filename )
```

Carrega uma lista de produtos a partir de um arquivo CSV.

Parameters

in	*filename*	Nome do arquivo CSV a ser lido.

Returns

Ponteiro para a cabeça da nova lista de produtos carregada.

4.11.2.4 liberar_lista_produtos()

```
void liberar_lista_produtos (
            Produto * head )
```

Libera toda a memória alocada para a lista de produtos.

Parameters

in	*head*	Ponteiro para a cabeça da lista a ser liberada.

4.11.2.5 listar_produtos()

```
void listar_produtos (
            Produto * head )
```

Lista todos os produtos da lista no terminal.

Parameters

in	*head*	Ponteiro para a cabeça da lista.

4.11.2.6 remover_produto()

```
Produto * remover_produto (
            Produto * head,
            int id )
```

Remove um produto da lista pelo seu ID.

Parameters

in	*head*	Ponteiro para a cabeça da lista.
in	*id*	ID do produto a ser removido.

Returns

Ponteiro para a nova cabeça da lista.

4.11.2.7 salvar_produtos_csv()

```
void salvar_produtos_csv (
            Produto * head,
            const char * filename )
```

Salva a lista de produtos em um arquivo no formato CSV.

Parameters

in	*head*	Ponteiro para a cabeça da lista.
in	*filename*	Nome do arquivo CSV de destino.

4.12 lista_produtos.h

Go to the documentation of this file.

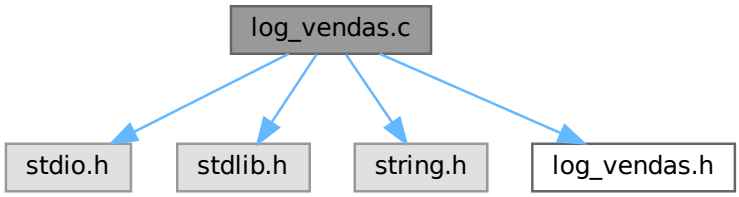
```
00001
00006 #ifndef LISTA_PRODUTOS_H
00007 #define LISTA_PRODUTOS_H
00008
00009 #include <allegro5/allegro.h>
00010
00014 typedef struct produto {
00015     int id;
00016     char nome[50];
00017     float preco;
00018     int estoque;
00020     // --- CAMPOS ADICIONADOS PARA ANIMAÇÃO ---
00021     char imagem_path[100];
00022     float pos_x_inicial;
00023     float pos_y_inicial;
00024     ALLEGRO_BITMAP* imagem_animacao;
00026     struct produto* prox;
00027     struct produto* ant;
00028 } Produto;
00029
00042 Produto* adicionar_produto(Produto* head, int id, const char* nome, float preco, int estoque, const
      char* img_path, float pos_x, float pos_y);
00043
00050 Produto* remover_produto(Produto* head, int id);
00051
00058 Produto* buscar_produto(Produto* head, int id);
00059
00064 void listar_produtos(Produto* head);
00065
00070 void liberar_lista_produtos(Produto* head);
00071
00077 void salvar_produtos_csv(Produto* head, const char* filename);
00078
00084 Produto* carregar_produtos_csv(const char* filename);
00085
00086 #endif
```

4.13 log_vendas.c File Reference

Implementação das funções de gerenciamento de logs de vendas.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "log_vendas.h"
```

Include dependency graph for log_vendas.c:



Functions

- Log * adicionar_log (Log *head, const char *timestamp, int produto_id, float valor_pago, float troco)
  Adiciona um novo log de venda ao início da lista.
- void salvar_logs (const char *filename, Log *head)
  Salva a lista de logs em um arquivo binário.
- Log * carregar_logs (const char *filename)
  Carrega a lista de logs a partir de um arquivo binário.
- void liberar_logs (Log *head)
  Libera toda a memória alocada para a lista de logs.
- void exibir_logs (Log *head)
  Exibe todos os logs de vendas no terminal.

4.13.1 Detailed Description

Implementação das funções de gerenciamento de logs de vendas.

4.13.2 Function Documentation

4.13.2.1 adicionar_log()

```
Log * adicionar_log (
            Log * head,
            const char * timestamp,
            int produto_id,
            float valor_pago,
            float troco )
```

Adiciona um novo log de venda ao início da lista.

Parameters

in	*head*	Ponteiro para a cabeça da lista de logs.
in	*timestamp*	String com a data e hora da venda.
in	*produto_id*	ID do produto vendido.
in	*valor_pago*	Valor inserido pelo cliente.
in	*troco*	Valor do troco.

Returns

Ponteiro para a nova cabeça da lista de logs.

4.13.2.2 carregar_logs()

```
Log * carregar_logs (
            const char * filename )
```

Carrega a lista de logs a partir de um arquivo binário.

Parameters

in	*filename*	Nome do arquivo a ser lido.

Returns

Ponteiro para a cabeça da nova lista de logs carregada.

4.13.2.3 exibir_logs()

```
void exibir_logs (
            Log * head )
```

Exibe todos os logs de vendas no terminal.

Parameters

in	*head*	Ponteiro para a cabeça da lista de logs.

4.13.2.4 liberar_logs()

```
void liberar_logs (
            Log * head )
```

Libera toda a memória alocada para a lista de logs.

Parameters

in	*head*	Ponteiro para a cabeça da lista a ser liberada.

4.13.2.5 salvar_logs()

```
void salvar_logs (
            const char * filename,
            Log * head )
```

Salva a lista de logs em um arquivo binário.

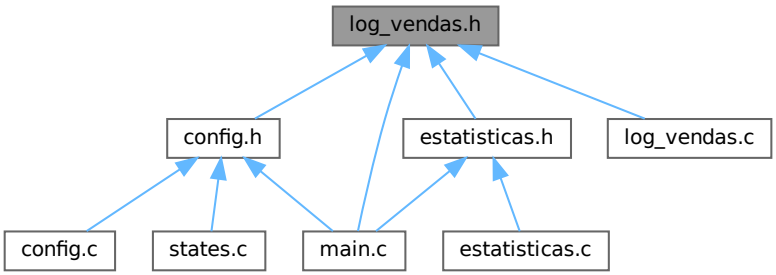
Parameters

in	*filename*	Nome do arquivo de destino.
in	*head*	Ponteiro para a cabeça da lista de logs.

4.14 log_vendas.h File Reference

Definições e protótipos para a lista simplesmente encadeada de logs de vendas.

This graph shows which files directly or indirectly include this file:



Data Structures

- struct Log

  Estrutura para representar um registro (log) de uma venda.

Typedefs

- typedef struct Log **Log**

  Estrutura para representar um registro (log) de uma venda.

Functions

- Log * adicionar_log (Log *head, const char *timestamp, int produto_id, float valor_pago, float troco)

  Adiciona um novo log de venda ao início da lista.
- void salvar_logs (const char *filename, Log *head)

  Salva a lista de logs em um arquivo binário.
- Log * carregar_logs (const char *filename)

  Carrega a lista de logs a partir de um arquivo binário.
- void liberar_logs (Log *head)

  Libera toda a memória alocada para a lista de logs.
- void exibir_logs (Log *head)

  Exibe todos os logs de vendas no terminal.

4.14.1 Detailed Description

Definições e protótipos para a lista simplesmente encadeada de logs de vendas.

4.14.2 Function Documentation

4.14.2.1 adicionar_log()

```
Log * adicionar_log (
            Log * head,
            const char * timestamp,
            int produto_id,
            float valor_pago,
            float troco )
```

Adiciona um novo log de venda ao início da lista.

Parameters

in	*head*	Ponteiro para a cabeça da lista de logs.
in	*timestamp*	String com a data e hora da venda.
in	*produto_id*	ID do produto vendido.
in	*valor_pago*	Valor inserido pelo cliente.
in	*troco*	Valor do troco.

Returns

Ponteiro para a nova cabeça da lista de logs.

4.14.2.2 carregar_logs()

```
Log * carregar_logs (
            const char * filename )
```

Carrega a lista de logs a partir de um arquivo binário.

Parameters

in	*filename*	Nome do arquivo a ser lido.

Returns

Ponteiro para a cabeça da nova lista de logs carregada.

4.14.2.3 exibir_logs()

```
void exibir_logs (
            Log * head )
```

Exibe todos os logs de vendas no terminal.

Parameters

in	*head*	Ponteiro para a cabeça da lista de logs.

4.14.2.4 liberar_logs()

```
void liberar_logs (
            Log * head )
```

Libera toda a memória alocada para a lista de logs.

Parameters

in	*head*	Ponteiro para a cabeça da lista a ser liberada.

4.14.2.5 salvar_logs()

```
void salvar_logs (
            const char * filename,
            Log * head )
```

Salva a lista de logs em um arquivo binário.

Parameters

in	*filename*	Nome do arquivo de destino.
in	*head*	Ponteiro para a cabeça da lista de logs.

4.15 log_vendas.h

Go to the documentation of this file.

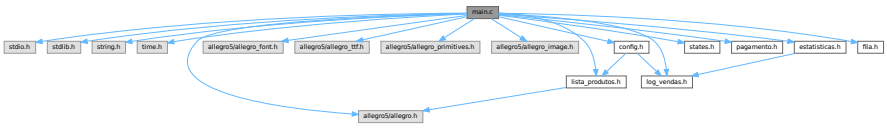
```
00001
00006 #ifndef LOG_VENDAS_H
00007 #define LOG_VENDAS_H
00008
00012 typedef struct Log {
00013     char timestamp[20];
00014     int produto_id;
00015     float valor_pago;
00016     float troco;
00017     struct Log* prox;
00018 } Log;
00019
00029 Log* adicionar_log(Log* head, const char* timestamp, int produto_id, float valor_pago, float troco);
00030
00036 void salvar_logs(const char* filename, Log* head);
00037
00043 Log* carregar_logs(const char* filename);
00044
00049 void liberar_logs(Log* head);
00050
00055 void exibir_logs(Log* head);
00056
00057 #endif
```

4.16 main.c File Reference

Ponto de entrada principal e gerenciador da interface gráfica com Allegro 5.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <allegro5/allegro.h>
#include <allegro5/allegro_font.h>
#include <allegro5/allegro_ttf.h>
#include <allegro5/allegro_primitives.h>
#include <allegro5/allegro_image.h>
#include "config.h"
#include "states.h"
#include "lista_produtos.h"
#include "log_vendas.h"
#include "pagamento.h"
#include "estatisticas.h"
#include "fila.h"
```

Include dependency graph for main.c:



Data Structures

- struct AnimacaoQueda

  Armazena os dados para a animação de queda de um produto.
- struct AnimacaoEntrega

  Armazena os dados para a animação de entrega do produto ao cliente.

Macros

- #define **LARGURA_JANELA** 1280
- #define **ALTURA_JANELA** 720
- #define **FPS** 60.0
- #define **TELA_X** 607
- #define **TELA_Y** 130
- #define **TELA_LARGURA** 159
- #define **TELA_ALTURA** 190
- #define **PROPAGANDA_X** 335
- #define **PROPAGANDA_Y** 110
- #define **PROPAGANDA_LARGURA** 220
- #define **PROPAGANDA_ALTURA** 60
- #define **PORTA_X** 400
- #define **PORTA_Y** 560

Functions

- void desenhar_produtos_estaticos (Produto *lista_produtos, AnimacaoQueda *animacao_atual)

  Desenha os produtos na vitrine da máquina.
- void desenhar_menu_principal (ALLEGRO_FONT *fonte, int indice_selecionado)

  Desenha o menu principal na tela da máquina.
- void desenhar_tela_propaganda (ALLEGRO_FONT *fonte, const char *propaganda)

  Desenha a mensagem de propaganda na área designada.
- void desenhar_tela_produtos (ALLEGRO_FONT *fonte, Produto *lista_produtos, int indice_selecionado)

  Desenha a tela de seleção de produtos.
- void desenhar_tela_pagamento (ALLEGRO_FONT *fonte, Produto *produto_selecionado, float valor_↩ inserido, int botao_pressionado_id)

  Desenha a tela de pagamento para um produto selecionado.
- void desenhar_tela_estatisticas (ALLEGRO_FONT *fonte, Contexto *ctx)

  Desenha a tela de estatísticas de vendas.
- void desenhar_tela_senha (ALLEGRO_FONT *fonte, const char *senha_digitada, bool cursor_visivel)

  Desenha a tela de inserção de senha para acesso à configuração.
- void desenhar_menu_config (ALLEGRO_FONT *fonte, int indice_selecionado)

  Desenha o menu principal de configurações.
- void desenhar_tela_add_produto (ALLEGRO_FONT *fonte, char *id, char *nome, char *preco, char *estoque, char *img_path, char *pos_x, char *pos_y, int campo_ativo, bool cursor_visivel)

  Desenha o formulário para adicionar um novo produto.
- void desenhar_tela_remove_produto (ALLEGRO_FONT *fonte, char *id_str, bool cursor_visivel)

  Desenha a tela para remover um produto.
- void desenhar_tela_atualizar_estoque (ALLEGRO_FONT *fonte, char *id_str, char *estoque_str, int campo_ativo, bool cursor_visivel)

  Desenha a tela para atualizar o estoque de um produto.
- int main ()

  Função principal da aplicação.

4.16.1 Detailed Description

Ponto de entrada principal e gerenciador da interface gráfica com Allegro 5.

Este arquivo inicializa a biblioteca Allegro, cria a janela, gerencia o laço de eventos principal e controla a máquina de estados finitos que rege a aplicação. Toda a lógica de renderização das telas e de interação do usuário (teclado e mouse) está contida aqui.

4.16.2 Function Documentation

4.16.2.1 desenhar_menu_config()

```
void desenhar_menu_config (
            ALLEGRO_FONT * fonte,
            int indice_selecionado )
```

Desenha o menu principal de configurações.

Parameters

fonte	Ponteiro para a fonte a ser usada.
indice_selecionado	Índice da opção de configuração selecionada.

4.16.2.2 desenhar_menu_principal()

```
void desenhar_menu_principal (
            ALLEGRO_FONT * fonte,
            int indice_selecionado )
```

Desenha o menu principal na tela da máquina.

Parameters

fonte	Ponteiro para a fonte a ser usada.
indice_selecionado	Índice da opção de menu atualmente selecionada.

4.16.2.3 desenhar_produtos_estaticos()

```
void desenhar_produtos_estaticos (
            Produto * lista_produtos,
            AnimacaoQueda * animacao_atual )
```

Desenha os produtos na vitrine da máquina.

Parameters

lista_produtos	Ponteiro para a lista de produtos.
animacao_atual	Ponteiro para a animação de queda, para não desenhar o produto que está caindo.

4.16.2.4 desenhar_tela_add_produto()

```
void desenhar_tela_add_produto (
            ALLEGRO_FONT * fonte,
            char * id,
```

```
            char * nome,
            char * preco,
            char * estoque,
            char * img_path,
            char * pos_x,
            char * pos_y,
            int campo_ativo,
            bool cursor_visivel )
```

Desenha o formulário para adicionar um novo produto.

Parameters

fonte	Ponteiro para a fonte a ser usada.
id	String para o campo ID.
nome	String para o campo Nome.
preco	String para o campo Preço.
estoque	String para o campo Estoque.
img_path	String para o caminho da imagem.
pos_x	String para a posição X.
pos_y	String para a posição Y.
campo_ativo	Índice do campo do formulário atualmente ativo.
cursor_visivel	Flag para controlar o piscar do cursor.

4.16.2.5 desenhar_tela_atualizar_estoque()

```
void desenhar_tela_atualizar_estoque (
            ALLEGRO_FONT * fonte,
            char * id_str,
            char * estoque_str,
            int campo_ativo,
            bool cursor_visivel )
```

Desenha a tela para atualizar o estoque de um produto.

Parameters

fonte	Ponteiro para a fonte a ser usada.
id_str	String para o campo ID.
estoque_str	String para o novo valor de estoque.
campo_ativo	Índice do campo do formulário atualmente ativo.
cursor_visivel	Flag para controlar o piscar do cursor.

4.16.2.6 desenhar_tela_estatisticas()

```
void desenhar_tela_estatisticas (
            ALLEGRO_FONT * fonte,
            Contexto * ctx )
```

Desenha a tela de estatísticas de vendas.

Parameters

fonte	Ponteiro para a fonte a ser usada.
ctx	Ponteiro para o contexto da aplicação, contendo os logs de vendas.

4.16.2.7 desenhar_tela_pagamento()

```
void desenhar_tela_pagamento (
            ALLEGRO_FONT * fonte,
            Produto * produto_selecionado,
            float valor_inserido,
            int botao_pressionado_id )
```

Desenha a tela de pagamento para um produto selecionado.

Parameters

fonte	Ponteiro para a fonte a ser usada.
produto_selecionado	Ponteiro para o produto que está sendo comprado.
valor_inserido	Valor monetário já inserido pelo usuário.
botao_pressionado_id	ID do botão sendo pressionado (para feedback visual).

4.16.2.8 desenhar_tela_produtos()

```
void desenhar_tela_produtos (
            ALLEGRO_FONT * fonte,
            Produto * lista_produtos,
            int indice_selecionado )
```

Desenha a tela de seleção de produtos.

Parameters

fonte	Ponteiro para a fonte a ser usada.
lista_produtos	Ponteiro para a lista de produtos a ser exibida.
indice_selecionado	Índice do produto atualmente selecionado.

4.16.2.9 desenhar_tela_propaganda()

```
void desenhar_tela_propaganda (
            ALLEGRO_FONT * fonte,
            const char * propaganda )
```

Desenha a mensagem de propaganda na área designada.

Parameters

fonte	Ponteiro para a fonte a ser usada.
propaganda	String contendo a propaganda a ser exibida.

4.16.2.10 desenhar_tela_remove_produto()

```
void desenhar_tela_remove_produto (
            ALLEGRO_FONT * fonte,
            char * id_str,
            bool cursor_visivel )
```

Desenha a tela para remover um produto.

Parameters

fonte	Ponteiro para a fonte a ser usada.
id_str	String para o campo ID do produto a ser removido.
cursor_visivel	Flag para controlar o piscar do cursor.

4.16.2.11 desenhar_tela_senha()

```
void desenhar_tela_senha (
            ALLEGRO_FONT * fonte,
            const char * senha_digitada,
            bool cursor_visivel )
```

Desenha a tela de inserção de senha para acesso à configuração.

Parameters

fonte	Ponteiro para a fonte a ser usada.
senha_digitada	String com a senha atualmente digitada pelo usuário.
cursor_visivel	Flag para controlar o piscar do cursor.

4.16.2.12 main()

```
int main ( )
```

Função principal da aplicação.

Inicializa o Allegro, carrega recursos, gerencia o laço de eventos (entradas do usuário, timer, redesenho da tela) e, ao final, salva os dados e libera os recursos alocados.

Returns

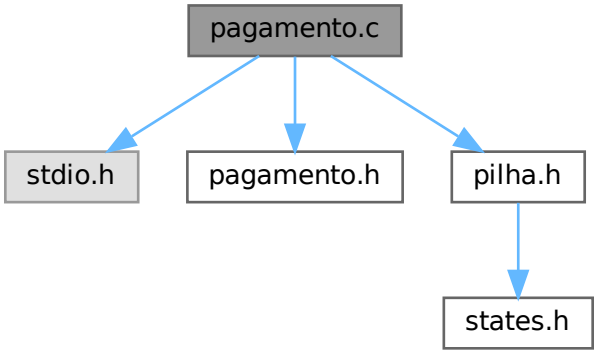
0 em caso de sucesso, -1 em caso de erro de inicialização.

4.17 pagamento.c File Reference

Implementação das funções de pagamento.

```
#include <stdio.h>
#include "pagamento.h"
#include "pilha.h"
```

Include dependency graph for pagamento.c:



Functions

- int inserir_moedas (float *valor_acumulado)

  Coleta as moedas inseridas pelo usuário (versão console).
- float calcular_troco (float valor_inserido, float preco)

  Calcula o troco com base no valor inserido e no preço do produto.
- float desfazer_moeda ()

  Desfaz a inserção da última moeda, usando a pilha de moedas.

4.17.1 Detailed Description

Implementação das funções de pagamento.

Note

Este módulo contém implementações originalmente pensadas para uma versão de console. Na versão gráfica final (main.c), a função `inserir_moedas` é substituída por uma lógica de interface gráfica. A função `calcular_troco` continua sendo utilizada.

4.17.2 Function Documentation

4.17.2.1 calcular_troco()

```
float calcular_troco (
            float valor_inserido,
            float preco )
```

Calcula o troco com base no valor inserido e no preço do produto.

Parameters

in	*valor_inserido*	Valor total inserido pelo usuário.
in	*preco*	Preço do produto.

Returns

Valor do troco, ou um valor negativo se o pagamento for insuficiente.

4.17.2.2 desfazer_moeda()

```
float desfazer_moeda ( )
```

Desfaz a inserção da última moeda, usando a pilha de moedas.

Returns

O valor da moeda removida, ou 0 se não houver moedas para remover.

4.17.2.3 inserir_moedas()

```
int inserir_moedas (
            float * valor_acumulado )
```

Coleta as moedas inseridas pelo usuário (versão console).

Coleta as moedas inseridas pelo usuário.

A função usa um ponteiro para retornar o valor total acumulado. O valor de retorno da função indica o status da operação.

Parameters

out	*valor_acumulado*	Ponteiro para a variável que armazenará o total inserido.

Returns

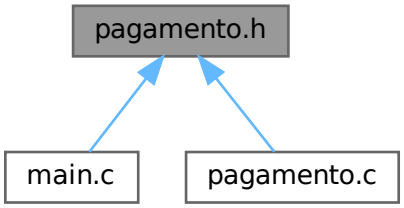
1 se a operação foi concluída com sucesso (usuário digitou 0).

0 se a operação foi cancelada (usuário digitou -2).

4.18 pagamento.h File Reference

Protótipos para as funções relacionadas ao processo de pagamento.

This graph shows which files directly or indirectly include this file:



Functions

- int inserir_moedas (float *valor_acumulado)

  Coleta as moedas inseridas pelo usuário.
- float calcular_troco (float valor_inserido, float preco)

  Calcula o troco com base no valor inserido e no preço do produto.
- float desfazer_moeda ()

  Desfaz a inserção da última moeda, usando a pilha de moedas.

4.18.1 Detailed Description

Protótipos para as funções relacionadas ao processo de pagamento.

4.18.2 Function Documentation

4.18.2.1 calcular_troco()

```
float calcular_troco (
            float valor_inserido,
            float preco )
```

Calcula o troco com base no valor inserido e no preço do produto.

Parameters

in	*valor_inserido*	Valor total inserido pelo usuário.
in	*preco*	Preço do produto.

Returns

Valor do troco, ou um valor negativo se o pagamento for insuficiente.

4.18.2.2 desfazer_moeda()

```
float desfazer_moeda ( )
```

Desfaz a inserção da última moeda, usando a pilha de moedas.

Returns

O valor da moeda removida, ou 0 se não houver moedas para remover.

4.18.2.3 inserir_moedas()

```
int inserir_moedas (
            float * valor_acumulado )
```

Coleta as moedas inseridas pelo usuário.

- A função usa um ponteiro para retornar o valor total acumulado. O valor de retorno da função indica o status da operação.

Parameters

out	*valor_acumulado*	Ponteiro para a variável que armazenará o total inserido.

Returns

1 se a operação foi concluída com sucesso (usuário digitou 0).
0 se a operação foi cancelada (usuário digitou -2).

Coleta as moedas inseridas pelo usuário.

A função usa um ponteiro para retornar o valor total acumulado. O valor de retorno da função indica o status da operação.

Parameters

out	*valor_acumulado*	Ponteiro para a variável que armazenará o total inserido.

Returns

1 se a operação foi concluída com sucesso (usuário digitou 0).
0 se a operação foi cancelada (usuário digitou -2).

4.19 pagamento.h

Go to the documentation of this file.

```
00001
00006 #ifndef PAGAMENTO_H
00007 #define PAGAMENTO_H
00008
00017 int inserir_moedas(float *valor_acumulado);
00018
00025 float calcular_troco(float valor_inserido, float preco);
```

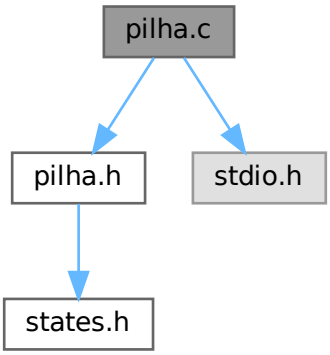
```
00026
00031 float desfazer_moeda();
00032
00033 #endif
```

4.20 pilha.c File Reference

Implementação da estrutura de dados de pilha.

```
#include "pilha.h"
#include <stdio.h>
```

Include dependency graph for pilha.c:



Functions

- void **inicializar_pilha** (Pilha *pilha)
- int **pilha_vazia** (Pilha *pilha)
- int **pilha_cheia** (Pilha *pilha)
- void **empilhar** (Pilha *pilha, float valor)
- float **desempilhar** (Pilha *pilha)
- void **inicializar_pilha_estados** (PilhaEstados *pilha)
- int **pilha_estados_vazia** (PilhaEstados *pilha)
- int **pilha_estados_cheia** (PilhaEstados *pilha)
- void **empilhar_estado** (PilhaEstados *pilha, State estado)
- State **desempilhar_estado** (PilhaEstados *pilha)

4.20.1 Detailed Description

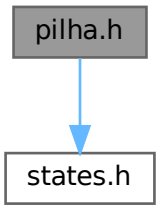
Implementação da estrutura de dados de pilha.

4.21 pilha.h File Reference

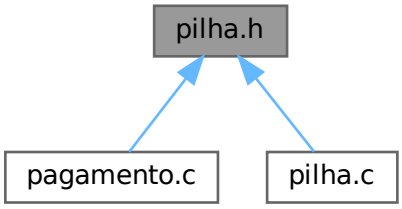
Definições e protótipos para estruturas de dados de pilha.

```
#include "states.h"
```

Include dependency graph for pilha.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct Pilha

  Estrutura para a pilha que armazena as moedas (valores float).
- struct PilhaEstados

  Estrutura para a pilha que armazena o histórico de estados da FSM.

Macros

- #define **MAX_PILHA** 100

Functions

- void **inicializar_pilha** (Pilha *pilha)
- int **pilha_vazia** (Pilha *pilha)
- int **pilha_cheia** (Pilha *pilha)
- void **empilhar** (Pilha *pilha, float valor)
- float **desempilhar** (Pilha *pilha)
- void **inicializar_pilha_estados** (PilhaEstados *pilha)
- int **pilha_estados_vazia** (PilhaEstados *pilha)
- int **pilha_estados_cheia** (PilhaEstados *pilha)
- void **empilhar_estado** (PilhaEstados *pilha, State estado)
- State **desempilhar_estado** (PilhaEstados *pilha)

4.21.1 Detailed Description

Definições e protótipos para estruturas de dados de pilha.

- Contém implementações para uma pilha de floats (usada para moedas) e uma pilha de estados (usada para o histórico de navegação).

4.22 pilha.h

Go to the documentation of this file.

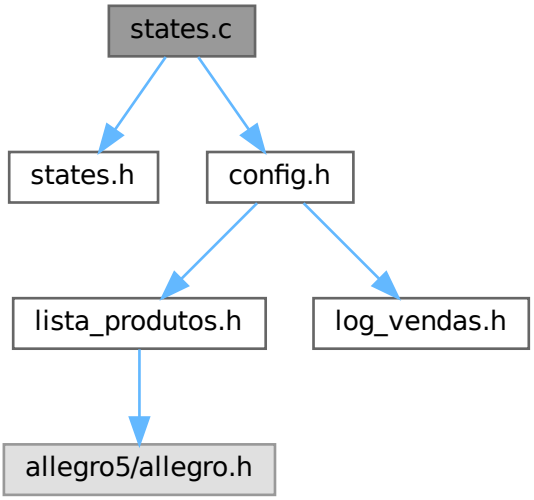
```
00001
00008 #ifndef PILHA_H
00009 #define PILHA_H
00010
00011 #include "states.h"
00012
00013 #define MAX_PILHA 100
00014
00018 typedef struct {
00019     float valores[MAX_PILHA];
00020     int topo;
00021 } Pilha;
00022
00023 void inicializar_pilha(Pilha *pilha);
00024 int pilha_vazia(Pilha *pilha);
00025 int pilha_cheia(Pilha *pilha);
00026 void empilhar(Pilha *pilha, float valor);
00027 float desempilhar(Pilha *pilha);
00028
00032 typedef struct {
00033     State estados[MAX_PILHA];
00034     int topo;
00035 } PilhaEstados;
00036
00037 void inicializar_pilha_estados(PilhaEstados *pilha);
00038 int pilha_estados_vazia(PilhaEstados *pilha);
00039 int pilha_estados_cheia(PilhaEstados *pilha);
00040 void empilhar_estado(PilhaEstados *pilha, State estado);
00041 State desempilhar_estado(PilhaEstados *pilha);
00042
00043 #endif
```

4.23 states.c File Reference

Implementação das funções de lógica para cada estado da FSM.

```
#include "states.h"
#include "config.h"
```

Include dependency graph for states.c:



Functions

- State **estado_menu_principal** (Contexto *ctx)
- State **estado_selecao_produto** (Contexto *ctx)
- State **estado_pagamento** (Contexto *ctx)
- State **estado_configuracao** (Contexto *ctx)

4.23.1 Detailed Description

Implementação das funções de lógica para cada estado da FSM.

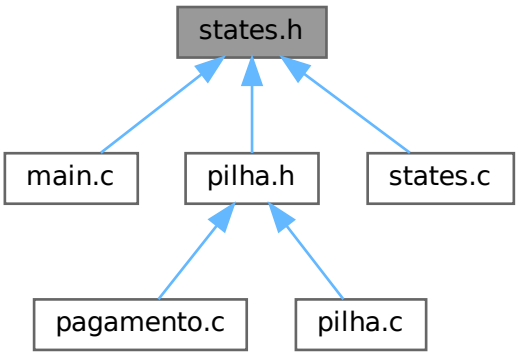
Note

Na versão Allegro, a maior parte da lógica foi movida para o main.c. Estas funções são mantidas para compatibilidade, mas estão vazias.

4.24 states.h File Reference

Definição dos estados da máquina e protótipos das funções de estado.

This graph shows which files directly or indirectly include this file:



Enumerations

- enum State {
  MENU_PRINCIPAL , **SELECAO_PRODUTO** , **PAGAMENTO** , **SAIR** ,
  VOLTAR , **ESTATISTICAS** , **TELA_SENHA** , **CONFIGURACAO_MENU** ,
  CONFIG_LISTAR_PRODUTOS , **CONFIG_ADICIONAR_PRODUTO** , **CONFIG_REMOVER_PRODUTO** ,
  CONFIG_ATUALIZAR_ESTOQUE ,
  ANIMACAO_QUEDA , **ANIMACAO_PORTA** , **ANIMACAO_ENTREGA** }

  Enumeração dos possíveis estados da máquina de estados finitos (FSM).

Functions

- State **estado_menu_principal** ()
- State **estado_selecao_produto** ()
- State **estado_pagamento** ()
- State **estado_configuracao** ()

4.24.1 Detailed Description

Definição dos estados da máquina e protótipos das funções de estado.

4.25 states.h

Go to the documentation of this file.

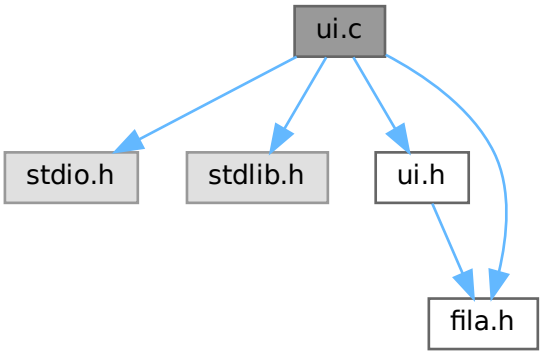
```
00001
00006 #ifndef STATES_H
00007 #define STATES_H
00008
00012 typedef enum {
00013     MENU_PRINCIPAL,
00014     SELECAO_PRODUTO,
00015     PAGAMENTO,
00016     SAIR,
00017     VOLTAR,
00018     ESTATISTICAS,
00019     TELA_SENHA,
00020     CONFIGURACAO_MENU,
00021     CONFIG_LISTAR_PRODUTOS,
00022     CONFIG_ADICIONAR_PRODUTO,
00023     CONFIG_REMOVER_PRODUTO,
00024     CONFIG_ATUALIZAR_ESTOQUE,
00025     ANIMACAO_QUEDA,
00026     ANIMACAO_PORTA,
00027     ANIMACAO_ENTREGA
00028 } State;
00029
00030 // Protótipos das funções de estado
00031 State estado_menu_principal();
00032 State estado_selecao_produto();
00033 State estado_pagamento();
00034 State estado_configuracao();
00035
00036 #endif
```

4.26 ui.c File Reference

Implementação das funções de interface de usuário.

```
#include <stdio.h>
#include <stdlib.h>
#include "ui.h"
#include "fila.h"
```

Include dependency graph for ui.c:



Functions

- void limpar_tela ()

  Limpa a tela do terminal.
- void pausar ()

  Pausa a execução do programa até que o usuário pressione Enter.
- void inicializar_interface (Fila *propagandas)

  Exibe a tela de boas-vindas da aplicação.

4.26.1 Detailed Description

Implementação das funções de interface de usuário.

Note

Estas funções foram criadas para uma versão de console da aplicação e não são utilizadas pela interface gráfica principal implementada em main.c.

4.26.2 Function Documentation

4.26.2.1 inicializar_interface()

```
void inicializar_interface (
            Fila * propagandas )
```

Exibe a tela de boas-vindas da aplicação.

Parameters

in	*propagandas*	Ponteiro para a fila de propagandas, para exibir a primeira.

4.26.2.2 limpar_tela()

```
void limpar_tela ( )
```

Limpa a tela do terminal.

Tenta usar "clear" (Linux/macOS) ou "cls" (Windows).

4.26.2.3 pausar()

```
void pausar ( )
```

Pausa a execução do programa até que o usuário pressione Enter.

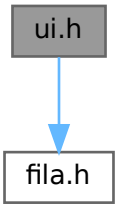
Esta versão corrigida consome quaisquer caracteres residuais no buffer de entrada e espera por um único Enter para continuar.

4.27 ui.h File Reference

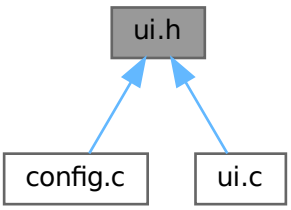
Protótipos para funções de interface de usuário (UI) no terminal.

```
#include "fila.h"
```

Include dependency graph for ui.h:



This graph shows which files directly or indirectly include this file:



Functions

- void limpar_tela ()

  Limpa a tela do terminal.
- void pausar ()

  Pausa a execução do programa até que o usuário pressione Enter.
- void inicializar_interface (Fila *propagandas)

  Exibe a tela de boas-vindas da aplicação.

4.27.1 Detailed Description

Protótipos para funções de interface de usuário (UI) no terminal.

Note

Estas funções foram criadas para uma versão de console da aplicação e não são utilizadas pela interface gráfica principal implementada em main.c.

4.27.2 Function Documentation

4.27.2.1 inicializar_interface()

```
void inicializar_interface (
            Fila * propagandas )
```

Exibe a tela de boas-vindas da aplicação.

Parameters

in	*propagandas*	Ponteiro para a fila de propagandas, para exibir a primeira.

4.27.2.2 limpar_tela()

```
void limpar_tela ( )
```

Limpa a tela do terminal.

Tenta usar "clear" (Linux/macOS) ou "cls" (Windows).

4.27.2.3 pausar()

```
void pausar ( )
```

Pausa a execução do programa até que o usuário pressione Enter.

Útil para permitir que o usuário leia as mensagens antes da tela ser limpa.

Esta versão corrigida consome quaisquer caracteres residuais no buffer de entrada e espera por um único Enter para continuar.

4.28 ui.h

Go to the documentation of this file.

```
00001
00008 #ifndef UI_H
00009 #define UI_H
00010
00011 #include "fila.h"
00012
00017 void limpar_tela();
00018
00023 void pausar();
00024
00029 void inicializar_interface(Fila *propagandas);
00030
00031 #endif
```

Index