Coffee Machine

Generated by Doxygen 1.9.4

Chapter 1

Coffee Machine

1.1 Introduction

There is a documentation for the coffee machine program who can be find `here` or on `Github`.
Made by Jules F. alias Seluj78

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

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

Chapter 4

Data Structure Documentation

4.1 struct_coins Struct Reference

Coin structure.

```
#include <main.h>
```

Data Fields

- float **value**

  Value of a coin.
- unsigned int **number**

  Number of coins remaining.

4.1.1 Detailed Description

Coin structure.

Structure in order to store coins from file for the program

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

- main.h

4.2 struct_drinks Struct Reference

Drink structure.

```
#include <main.h>
```

Data Fields

- char **name** [20]

  Name of the coffee.
- float **price**

  Coffee's price.
- int **number**

  Number of drinks remaining.

4.2.1 Detailed Description

Drink structure.

Structure in order to store drinks from file for the program

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

- main.h

4.3 struct_given Struct Reference

Money given structure.

```
#include <main.h>
```

Data Fields

- int **c05**

  number of 0.5 coins
- int **c1**

  number of 1 coins
- int **c2**

  number of 2 coins
- int **c5**

  number of 5 coins

4.3.1 Detailed Description

Money given structure.

Structure in order to store coins entered by the user

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

- main.h

4.4 struct_user Struct Reference

User's choice structure.

```
#include <main.h>
```

Data Fields

- struct_drinks **d**

    Drink chosen by the user.
- int **type**

    Type of the drink by its digit in the list.
- float **money**

    Total amount of money that the user gives.
- struct_given **given**

    All coins that the user gives in order to increase amount of coins in the machine.

4.4.1 Detailed Description

User's choice structure.

Structure in order to store the choice of the user

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

- main.h

Chapter 5

File Documentation

5.1 administration.c File Reference

Administration source file.

```
#include "main.h"
```

Functions

- bool code_check (int a, int b)
- int administration_menu ()
- bool enter_code ()
- struct_drinks ∗ add_drink (struct_drinks ∗drink, int ∗nb_type)
- struct_drinks ∗ remove_drink (struct_drinks ∗drink, int ∗nb_type)
- struct_drinks ∗ administration (struct_drinks ∗drink, int ∗nb_type)

5.1.1 Detailed Description

Administration source file.

Source file contains all functions' definitions related to administrator mode

Author

Jules F.

Date

May 2022

5.1.2 Function Documentation

5.1.2.1 add_drink()

```
struct_drinks ∗ add_drink (
            struct_drinks ∗ drink,
            int ∗ nb_type )
```

Function that add a drink in the drink list, it modify *drink* and the number of drinks

Parameters

drink	where to add the new drink
nb_type	number of drink to modify

Returns

return drinks data

5.1.2.2 administration()

```
struct_drinks * administration (
                struct_drinks * drink,
                int * nb_type )
```

Main part of administration mode, call all others functions

Parameters

drink	drinks data
nb_type	number of types of drink

Returns

return drinks data

5.1.2.3 administration_menu()

```
int administration_menu ( )
```

Print menu for administration mode

Returns

return choice of user

5.1.2.4 code_check()

```
bool code_check (
                int a,
                int b )
```

Function to check if *a* is equal to *b*

Parameters

a	first parameter
b	second parameter

Returns

return true if there equal, false if not

5.1.2.5 enter_code()

```
bool enter_code ( )
```

Function that ask the user to enter the code

Returns

return the code given

5.1.2.6 remove_drink()

```
struct_drinks * remove_drink (
              struct_drinks * drink,
              int * nb_type )
```

Function that remove a drink in the list, it modify drink and the number of drinks

Parameters

drink	where to remove a drink
nb_type	number of drink to modify

Returns

return drinks data

5.2 administration.h File Reference

Administration header file.

Functions

- bool code_check (int a, int b)
- int administration_menu ()
- bool enter_code ()
- struct_drinks ∗ add_drink (struct_drinks ∗drink, int ∗nb_type)
- struct_drinks ∗ remove_drink (struct_drinks ∗drink, int ∗nb_type)
- struct_drinks ∗ administration (struct_drinks ∗drink, int ∗nb_type)

5.2.1 Detailed Description

Administration header file.

Header file that contains all prototypes of function related to administration mode

Author

Jules F.

Date

May 2022

5.2.2 Function Documentation

5.2.2.1 add_drink()

```
struct_drinks ∗ add_drink (
            struct_drinks ∗ drink,
            int ∗ nb_type )
```

Function that add a drink in the drink list, it modify *drink* and the number of drinks

Parameters

drink	where to add the new drink
nb_type	number of drink to modify

Returns

return drinks data

5.2.2.2 administration()

```
struct_drinks * administration (
             struct_drinks * drink,
             int * nb_type )
```

Main part of administration mode, call all others functions

Parameters

drink	drinks data
nb_type	number of types of drink

Returns

return drinks data

5.2.2.3 administration_menu()

```
int administration_menu ( )
```

Print menu for administration mode

Returns

return choice of user

5.2.2.4 code_check()

```
bool code_check (
             int a,
             int b )
```

Function to check if *a* is equal to *b*

Parameters

a	first parameter
b	second parameter

Returns

return true if there equal, false if not

5.2.2.5 enter_code()

```
bool enter_code ( )
```

Function that ask the user to enter the code

Returns

return the code given

5.2.2.6 remove_drink()

```
struct_drinks * remove_drink (
              struct_drinks * drink,
              int * nb_type )
```

Function that remove a drink in the list, it modify drink and the number of drinks

Parameters

drink	where to remove a drink
nb_type	number of drink to modify

Returns

return drinks data

5.3 administration.h

Go to the documentation of this file.

```
#ifndef COFFEE_MACHINE_ADMINISTRATION_H
#define COFFEE_MACHINE_ADMINISTRATION_H

bool code_check(int a, int b);

int administration_menu();

bool enter_code();

struct_drinks *add_drink(struct_drinks *drink, int *nb_type);

struct_drinks *remove_drink(struct_drinks *drink, int *nb_type);

struct_drinks *administration(struct_drinks *drink, int *nb_type);

#endif //COFFEE_MACHINE_ADMINISTRATION_H
```

5.4 drinks.c File Reference

Drinks source file.

```
#include "main.h"
```

Functions

- int drink_menu (struct_drinks drink[ ], const int nb_drink, struct_user ∗user)
- bool check_drink (struct_user user)

5.4.1 Detailed Description

Drinks source file.

Source file that contains all definitions of functions that are related to drinks

Author

Jules F.

Date

May 2022

5.4.2 Function Documentation

5.4.2.1 check_drink()

```
bool check_drink (
             struct_user user )
```

Check if the drink chosen is out of stock or not

Parameters

user	user data

Returns

true if all OK and false if the drink is out of stock

5.4.2.2 drink_menu()

```
int drink_menu (
             struct_drinks drink[ ],
             int nb_drink,
             struct_user ∗ user )
```

Print a menu who ask the user to enter which drink he wants

Parameters

drink	drinks available
nb_drink	number of drink available
user	data of the user's drinks

Returns

return 0 if nothing wrong, 1 if the user wants to enter in administration mode, 2 if the user chooses to reset data, 2 if the user chooses to quit and 36 if there is a problem

5.5 drinks.h File Reference

Drinks header file.

Functions

- int drink_menu (struct_drinks drink[ ], int nb_drink, struct_user ∗user)
- bool check_drink (struct_user user)

5.5.1 Detailed Description

Drinks header file.

Header file that contains all prototypes of functions that are related to drinks

Author

Jules F.

Date

May 2022

5.5.2 Function Documentation

5.5.2.1 check_drink()

```
bool check_drink (
            struct_user user )
```

Check if the drink chosen is out of stock or not

Parameters

user	user data

Returns

true if all OK and false if the drink is out of stock

5.5.2.2 drink_menu()

```
int drink_menu (
            struct_drinks drink[ ],
            int nb_drink,
            struct_user * user )
```

Print a menu who ask the user to enter which drink he wants

Parameters

drink	drinks available
nb_drink	number of drink available
user	data of the user's drinks

Returns

return 0 if nothing wrong, 1 if the user wants to enter in administration mode, 2 if the user chooses to reset data, 2 if the user chooses to quit and 36 if there is a problem

5.6 drinks.h

Go to the documentation of this file.

```
#ifndef COFFEE_MACHINE_DRINKS_H
#define COFFEE_MACHINE_DRINKS_H

int drink_menu(struct_drinks drink[], int nb_drink, struct_user *user);

bool check_drink(struct_user user);

#endif //COFFEE_MACHINE_DRINKS_H
```

5.7 main.c File Reference

Main file.

`#include "main.h"`

Functions

- int **main** (void)

5.7.1 Detailed Description

Main file.

Main file of the program, it contains only main function

Author

Jules F.

Date

May 2022

5.8 main.h File Reference

Main header file.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <stdbool.h>
#include "drinks.h"
#include "money.h"
#include "processing.h"
#include "administration.h"
```

Data Structures

- struct struct_drinks

  Drink structure.
- struct struct_coins

  Coin structure.
- struct struct_given

  Money given structure.
- struct struct_user

  User's choice structure.

Macros

- #define **number_coin** 4

5.8.1 Detailed Description

Main header file.

Header file that contains all structures and includes. This file is the only file that is include in others files

Author

Jules F.

Date

May 2022

5.9 main.h

Go to the documentation of this file.

```
#ifndef COFFEE_MACHINE_MAIN_H
#define COFFEE_MACHINE_MAIN_H

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <stdbool.h>

#define number_coin 4

typedef struct {
    char name[20];
    float price;
    int number;
} struct_drinks;

typedef struct {
    float value;
    unsigned int number;
} struct_coins;

typedef struct {
    int c05;
    int c1;
    int c2;
    int c5;
} struct_given;

typedef struct {
    struct_drinks d;
    int type;
    float money;
    struct_given given;
} struct_user;

#include "drinks.h"
#include "money.h"
#include "processing.h"
#include "administration.h"

#endif //COFFEE_MACHINE_MAIN_H
```

5.10 money.c File Reference

Money source file.

```
#include "main.h"
```

Functions

- int coin_add (struct_user ∗user)
- bool check_change (struct_user user, struct_coins coins[ ])

5.10.1 Detailed Description

Money source file.

Source file that contains all functions' definitions related to money

Author

Jules F.

Date

May 2022

5.10.2 Function Documentation

5.10.2.1 check_change()

```
bool check_change (
            struct_user user,
            struct_coins coins[ ] )
```

Check if the machine can give change

Parameters

user	user's data
coins	coins' data reduce with coin given to user and increase with coin taken from user

Returns

return true if there is enough change and false if there is a problem

5.10.2.2 coin_add()

```
int coin_add (
            struct_user ∗ user )
```

Ask the user to enter the right amount of money needed for the drink chosen

Parameters

user	a pointer to user's data

Returns

return 0 if the user wants to cancel, 1 if user gives the right amount of money and 2 if the user gives more money than needed

5.11 money.h File Reference

Money header file.

Functions

- int coin_add (struct_user ∗user)
- bool check_change (struct_user user, struct_coins coins[ ])

5.11.1 Detailed Description

Money header file.

Header file that contains all functions' prototypes related to money

Author

Jules F.

Date

May 2022

5.11.2 Function Documentation

5.11.2.1 check_change()

```
bool check_change (
            struct_user user,
            struct_coins coins[ ] )
```

Check if the machine can give change

Parameters

user	user's data
coins	coins' data reduce with coin given to user and increase with coin taken from user

Returns

return true if there is enough change and false if there is a problem

5.11.2.2 coin_add()

```
int coin_add (
            struct_user * user )
```

Ask the user to enter the right amount of money needed for the drink chosen

Parameters

user	a pointer to user's data

Returns

return 0 if the user wants to cancel, 1 if user gives the right amount of money and 2 if the user gives more money than needed

5.12 money.h

Go to the documentation of this file.

```
#ifndef COFFEE_MACHINE_MONEY_H
#define COFFEE_MACHINE_MONEY_H

int coin_add(struct_user *user);

bool check_change(struct_user user, struct_coins coins[]);

#endif //COFFEE_MACHINE_MONEY_H
```

5.13 processing.c File Reference

Processing source file.

```
#include "main.h"
```

Functions

- struct_drinks ∗ initialisation (struct_drinks ∗drinks, struct_coins ∗coins, int ∗number_of_coffee)
- void print_info ()
- void saving (struct_drinks ∗drinks, struct_coins coins[number_coin], int number_of_coffee)
- void reset (struct_drinks ∗drinks, struct_coins ∗coins, int number_of_coffee)
- float parse_float (char str[5])
- void clear_buffer ()

5.13.1 Detailed Description

Processing source file.

Source file that contains all others functions' definition that are not related to drinks, coins or administration

Author

Jules F.

Date

May 2022

5.13.2 Function Documentation

5.13.2.1 clear_buffer()

```
void clear_buffer ( )
```

Function that clear the input file because sometimes there still have a character and most of the time it is *\n*

5.13.2.2 initialisation()

```
struct_drinks ∗ initialisation (
             struct_drinks ∗ drinks,
             struct_coins ∗ coins,
             int ∗ number_of_coffee )
```

Initialize the program, check and read files

Parameters

drinks	structure of drinks, the function save all drinks from file **drinks** to this variable
coins	structure of coins, the function save all coins from file **coins** to this variable
number_of_coffee	number of types of coffee in the file **drinks**

Returns

return a pointer to all drinks saved in variable drinks

5.13.2.3 parse_float()

```
float parse_float (
             char str[5] )
```

That function is an update of the function strlen

Parameters

str	string to convert in float

Returns

return float number or -1 if there is a problem

5.13.2.4 print_info()

```
void print_info ( )
```

Just print information when the program start

5.13.2.5 reset()

```
void reset (
             struct_drinks * drinks,
             struct_coins * coins,
             int number_of_coffee )
```

Reset number of coffee and amount of coins in the machine

Parameters

drinks	drinks to reset
coins	coins to reset
number_of_coffee	number of types of coffee

5.13.2.6 saving()

```
void saving (
            struct_drinks * drinks,
            struct_coins coins[number_coin],
            int number_of_coffee )
```

Save drinks, coins and number of coffee in file for the next program's start

Parameters

drinks	drinks to save
coins	coins to save
number_of_coffee	number of types of coffee to save

5.14 processing.h File Reference

Processing header file.

Functions

- struct_drinks ∗ initialisation (struct_drinks ∗drinks, struct_coins ∗coins, int ∗number_of_coffee)
- void print_info ()
- void saving (struct_drinks ∗drinks, struct_coins coins[number_coin], int number_of_coffee)
- void reset (struct_drinks ∗drinks, struct_coins ∗coins, int number_of_coffee)
- float parse_float (char str[5])
- void clear_buffer ()

5.14.1 Detailed Description

Processing header file.

Header file that contains all others functions' prototypes that are not related to drinks, coins or administration

Author

Jules F.

Date

May 2022

5.14.2 Function Documentation

5.14.2.1 clear_buffer()

```
void clear_buffer ( )
```

Function that clear the input file because sometimes there still have a character and most of the time it is *\n*

5.14.2.2 initialisation()

```
struct_drinks * initialisation (
             struct_drinks * drinks,
             struct_coins * coins,
             int * number_of_coffee )
```

Initialize the program, check and read files

Parameters

drinks	structure of drinks, the function save all drinks from file **drinks** to this variable
coins	structure of coins, the function save all coins from file **coins** to this variable
number_of_coffee	number of types of coffee in the file **drinks**

Returns

return a pointer to all drinks saved in variable drinks

5.14.2.3 parse_float()

```
float parse_float (
             char str[5] )
```

That function is an update of the function strlen

Parameters

str	string to convert in float

Returns

return float number or -1 if there is a problem

5.14.2.4 print_info()

```
void print_info ( )
```

Just print information when the program start

5.14.2.5 reset()

```
void reset (
             struct_drinks * drinks,
             struct_coins * coins,
             int number_of_coffee )
```

Reset number of coffee and amount of coins in the machine

Parameters

drinks	drinks to reset
coins	coins to reset
number_of_coffee	number of types of coffee

5.14.2.6 saving()

```
void saving (
             struct_drinks * drinks,
             struct_coins coins[number_coin],
             int number_of_coffee )
```

Save drinks, coins and number of coffee in file for the next program's start

Parameters

drinks	drinks to save
coins	coins to save
number_of_coffee	number of types of coffee to save

5.15 processing.h

Go to the documentation of this file.

```
#ifndef COFFEE_MACHINE_PROCESSING_H
#define COFFEE_MACHINE_PROCESSING_H

struct_drinks *initialisation(struct_drinks *drinks, struct_coins *coins, int *number_of_coffee);

void print_info();

void saving(struct_drinks *drinks, struct_coins coins[number_coin], int number_of_coffee);

void reset(struct_drinks *drinks, struct_coins *coins, int number_of_coffee);

float parse_float(char str[5]);

void clear_buffer();

#endif //COFFEE_MACHINE_PROCESSING_H
```

Index