My Project

Generated by Doxygen 1.8.13

Contents

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

Chapter 2

File Documentation

2.1 CMakeFiles/3.15.2/CompilerIdC/CMakeCCompilerId.c File Reference

Macros

- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define C_DIALECT

Functions

- int main (int argc, char ∗argv[ ])

Variables

- char const ∗ info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const ∗ info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const ∗ info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char ∗ info_language_dialect_default

2.1.1 Macro Definition Documentation

2.1.1.1 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

2.1.1.2 C_DIALECT

```
#define C_DIALECT
```

2.1.1.3 COMPILER_ID

```
#define COMPILER_ID ""
```

2.1.1.4 DEC

```
#define DEC(
              n )
```

Value:

```
('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

2.1.1.5 HEX

```
#define HEX(
              n )
```

Value:

```
('0' + ((n)>>28 & 0xF)), \
  ('0' + ((n)>>24 & 0xF)), \
  ('0' + ((n)>>20 & 0xF)), \
  ('0' + ((n)>>16 & 0xF)), \
  ('0' + ((n)>>12 & 0xF)), \
  ('0' + ((n)>>8  & 0xF)), \
  ('0' + ((n)>>4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

2.1.1.6 PLATFORM_ID

```
#define PLATFORM_ID
```

2.1.1.7 STRINGIFY

```
#define STRINGIFY(
              X ) STRINGIFY_HELPER(X)
```

2.1.1.8 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
              X ) #X
```

2.1.2 Function Documentation

2.1.2.1 main()

```
int main (
             int argc,
             char * argv[] )
```

```
641 {
642   int require = 0;
643   require += info_compiler[argc];
644   require += info_platform[argc];
645   require += info_arch[argc];
646 #ifdef COMPILER_VERSION_MAJOR
647   require += info_version[argc];
648 #endif
649 #ifdef COMPILER_VERSION_INTERNAL
650   require += info_version_internal[argc];
651 #endif
652 #ifdef SIMULATE_ID
653   require += info_simulate[argc];
654 #endif
655 #ifdef SIMULATE_VERSION_MAJOR
656   require += info_simulate_version[argc];
657 #endif
658 #if defined(__CRAYXE) || defined(__CRAYXC)
659   require += info_cray[argc];
660 #endif
661   require += info_language_dialect_default[argc];
662   (void)argv;
663   return require;
664 }
```

2.1.3 Variable Documentation

2.1.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

2.1.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

2.1.3.3 info_language_dialect_default

```
const char* info_language_dialect_default
```

Initial value:

```
=
  "INFO" ":" "dialect_default[" C_DIALECT "]"
```

2.1.3.4 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

2.2 CMakeFiles/3.15.2/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference

Macros

- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define CXX_STD __cplusplus

Functions

- int main (int argc, char *argv[])

Variables

- char const * info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const * info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const * info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char * info_language_dialect_default

2.2.1 Macro Definition Documentation

2.2.1.1 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

2.2.1.2 COMPILER_ID

```
#define COMPILER_ID ""
```

2.2.1.3 CXX_STD

```
#define CXX_STD __cplusplus
```

2.2.1.4 DEC

```
#define DEC(
              n )
```

Value:

```
('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

2.2.1.5 HEX

```
#define HEX(
              n )
```

Value:

```
('0' + ((n)>>28 & 0xF)), \
  ('0' + ((n)>>24 & 0xF)), \
  ('0' + ((n)>>20 & 0xF)), \
  ('0' + ((n)>>16 & 0xF)), \
  ('0' + ((n)>>12 & 0xF)), \
  ('0' + ((n)>>8  & 0xF)), \
  ('0' + ((n)>>4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

2.2.1.6 PLATFORM_ID

```
#define PLATFORM_ID
```

2.2.1.7 STRINGIFY

```
#define STRINGIFY(
              X ) STRINGIFY_HELPER(X)
```

2.2.1.8 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
              X ) #X
```

2.2.2 Function Documentation

2.2.2.1 main()

```
int main (
             int argc,
             char * argv[] )
```

```
622 {
623   int require = 0;
624   require += info_compiler[argc];
625   require += info_platform[argc];
626 #ifdef COMPILER_VERSION_MAJOR
627   require += info_version[argc];
628 #endif
629 #ifdef COMPILER_VERSION_INTERNAL
630   require += info_version_internal[argc];
631 #endif
632 #ifdef SIMULATE_ID
633   require += info_simulate[argc];
634 #endif
635 #ifdef SIMULATE_VERSION_MAJOR
636   require += info_simulate_version[argc];
637 #endif
638 #if defined(__CRAYXE) || defined(__CRAYXC)
639   require += info_cray[argc];
640 #endif
641   require += info_language_dialect_default[argc];
642   (void)argv;
643   return require;
644 }
```

2.2.3 Variable Documentation

2.2.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

2.2.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

2.2.3.3 info_language_dialect_default

```
const char* info_language_dialect_default
```

Initial value:

```
= "INFO" ":" "dialect_default["


  "98"
"]"
```

2.2.3.4 info_platform

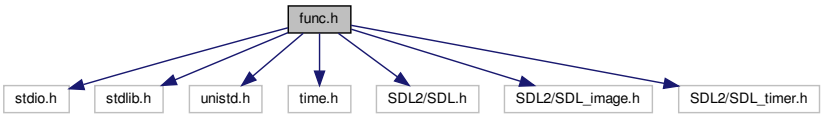
```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

2.3 func.h File Reference

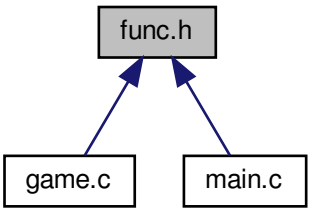
```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <time.h>
#include <SDL2/SDL.h>
#include <SDL2/SDL_image.h>
```

```
#include <SDL2/SDL_timer.h>
```

Include dependency graph for func.h:



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



Macros

- #define heigh 30
- #define width 80

Functions

- void generate ()

  Generates the table based on the selected capacity level.
- void display (char world[width][heigh])
- void rules (int b, int c, char table1[width][heigh], char tableN[width][heigh])

Variables

- int capacity
- char table_t [width][heigh]
- char table_N [width][heigh]

2.3.1 Macro Definition Documentation

2.3.1.1 heigh

```
#define heigh 30
```

2.3.1.2 width

```
#define width 80
```

2.3.2 Function Documentation

2.3.2.1 display()

```
void display (
             char world[width][heigh] )
```

```
32                                      {
33
34     for (int j = 0; j < heigh; j++){
35         for(int i = 0 ;i < width; i++){
36             printf("\033[31m%c", world[i][j]);
37         }
38         printf("\n");
39     }
40 }
```

Here is the caller graph for this function:

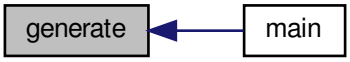
2.3.2.2 generate()

```
generate ( )
```

Generates the table based on the selected capacity level.

```
14                {
15     int life;
16     for (int j=0;j<heigh;j++){
17         for(int i=0;i<width;i++){
18             life=(rand()%9);
19             if (life<capacity) {
20                 table_t[i][j]='0';
21             }
22             else{ table_t[i][j]=' ';
23             }
24         }
25     }
26 }
```

Here is the caller graph for this function:



2.3.2.3 rules()

```
void rules (
              int b,
              int c,
              char table1[width][heigh],
              char tableN[width][heigh] )
```

if cell is alive

Due to the rules

Not generating properly with iy

```
42                                                                  { // Checks how many living
       neighbors this tile currently has.
43     int neighbors = 0;
44     for (int i = -1; i <= 1; i++) {
45         for (int j = -1; j <= 1; j++) {
46             if ((i == 0 && j == 0) || b + i >= width || b + i < 0 || c + j >=
      heigh  || c + j < 0) { //either in current cell or neightb.
47                 continue;
48             }
52             if (table1[b + i][c + j] == '0'){
53                 neighbors++;
54             }
55         }
56     }
```

```
57
61      if (neighbors == 3 || (neighbors == 2 && table1[b][c] == '0')) { //Any live cell with two or three live
        neighbors survives.
62          tableN[b][c] = '0';
63      } else {
64          tableN[b][c] = ' ';
65      }
69      //    if (neighbors == 3 || table1[b][c] == ' ') {//Any dead cell with three live neighbors becomes a
        live cell.
70      //  tableN[b][c] = '0';
71      // } else {
72      //  tableN[b][c] = ' ';
73      // }
74  }
```

Here is the caller graph for this function:

2.3.3 Variable Documentation

2.3.3.1 capacity

```
int capacity
```

2.3.3.2 table_N

```
char table_N[width][heigh]
```

2.3.3.3 table_t

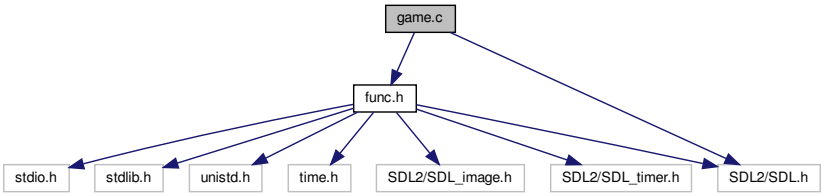
```
char table_t[width][heigh]
```

2.4 game.c File Reference

game Source code of Conway's Game Of Life

```
#include "func.h"
#include <SDL2/SDL.h>
```

Include dependency graph for game.c:



Functions

- void generate ()

  Generates the table based on the selected capacity level.
- void display (char world[width][heigh])
- void rules (int b, int c, char table1[width][heigh], char tableN[width][heigh])

2.4.1 Detailed Description

game Source code of Conway's Game Of Life

2.4.2 Function Documentation

2.4.2.1 display()

```
void display (
             char world[width][heigh] )
```

```
32                                      {
33
34     for (int j = 0; j < heigh; j++){
35         for(int i = 0 ;i < width; i++){
36             printf("\033[31m%c", world[i][j]);
37         }
38         printf("\n");
39     }
40 }
```

Here is the caller graph for this function:

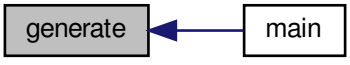
2.4.2.2 generate()

```
void generate ( )
```

Generates the table based on the selected capacity level.

```
14                {
15     int life;
16     for (int j=0;j<heigh;j++){
17         for(int i=0;i<width;i++){
18             life=(rand()%9);
19             if (life<capacity) {
20                 table_t[i][j]='0';
21             }
22             else{ table_t[i][j]=' ';
23             }
24         }
25     }
26 }
```

Here is the caller graph for this function:



2.4.2.3 rules()

```
void rules (
              int b,
              int c,
              char table1[width][heigh],
              char tableN[width][heigh] )
```

if cell is alive

Due to the rules

Not generating properly with iy

```
42                                                                          { // Checks how many living
      neighbors this tile currently has.
43     int neighbors = 0;
44     for (int i = -1; i <= 1; i++) {
45         for (int j = -1; j <= 1; j++) {
46             if ((i == 0 && j == 0) || b + i >= width || b + i < 0 || c + j >=
     heigh  || c + j < 0) { //either in current cell or neightb.
47                 continue;
48             }
52             if (table1[b + i][c + j] == '0'){
53                 neighbors++;
54             }
55         }
56     }
57
61     if (neighbors == 3 || (neighbors == 2 && table1[b][c] == '0')) { //Any live cell with two or three live
      neighbors survives.
62         tableN[b][c] = '0';
63     } else {
64         tableN[b][c] = ' ';
65     }
69     //    if (neighbors == 3 || table1[b][c] == ' ') {//Any dead cell with three live neighbors becomes a
      live cell.
70     //  tableN[b][c] = '0';
71     // } else {
72     //  tableN[b][c] = ' ';
73     // }
74 }
```

Here is the caller graph for this function:

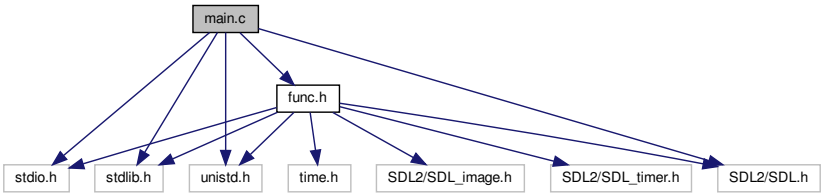
2.5 main.c File Reference

main function Main file of Conway's Game Of Life project

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <SDL2/SDL.h>
```

```
#include "func.h"
```

Include dependency graph for main.c:



Functions

- int main ()

  generating tables and creating the game due to random 7 type of position which the user select

2.5.1 Detailed Description

main function Main file of Conway's Game Of Life project

2.5.2 Function Documentation

2.5.2.1 main()

```
main ( )
```

generating tables and creating the game due to random 7 type of position which the user select

Taking the capacity

Warning

Only range of [1;7] acceptable, otherwise it'll ask again untill get the integer in that range

Generates new table

Warning

draft SDL

Displays the datas in the t and t+1 tables separately

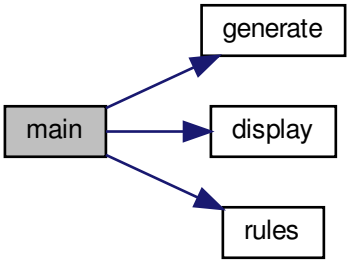
Warning

draft SDL
for compiling with SDL: gcc game.c main.c `sdl2-config --cflags --libs` -lSDL2 -lSDL2_mixer -lSDL2_image -lSDL2_ttfgcc game.c main.c `sdl2-config --cflags --libs` -lSDL2 -lSDL2_mixer -lSDL2_image -lSDL2_ttf

```
18          {
19 // unsigned char copy[width * heigh];
20 //unsigned char pixels[width * heigh * 4];
21     capacity = 0;
22     char start;
23     srand(time(NULL));
24 
29     printf("Please, choose the starting capacity level in [1; 7] range:\n");
30     char buffer[80] = "";
31     for(int i=0; i<7; i++){
32         fgets(buffer, sizeof(buffer), stdin);
33         if (sscanf(buffer, "%d", &capacity) != 1) {
34             capacity = 0;
35         }
36         if (capacity >= 1 && capacity <= 7) {
37             break;
38         }
39         printf("Please, choose the starting capacity level in [1; 7] range:\n");
40     }
41 
45     generate();
46 
50 // SDL_Init(SDL_INIT_VIDEO);
51 // SDL_Window *window = SDL_CreateWindow("life", SDL_WINDOWPOS_UNDEFINED, SDL_WINDOWPOS_UNDEFINED, width,
       heigh, SDL_WINDOW_SHOWN);
52 // SDL_Renderer *renderer = SDL_CreateRenderer(window, -1, SDL_RENDERER_ACCELERATED);
53 // SDL_Texture *texture = SDL_CreateTexture(renderer, SDL_PIXELFORMAT_RGBA32, SDL_TEXTUREACCESS_STREAMING,
       width, heigh);
54 // SDL_Event event;
55 // while (1) {
56 //     if (SDL_PollEvent(&event)) {
57 //       if (event.type == SDL_QUIT) {
58 //         break;
59 //       }
60 //     }
61 
65     display(table_t);
66     printf("\n\n Type '1' to play the game.\n Type '0' to exit.\n");
67     // printf("\033[2J");
68     int turn = 0;
69     int generation;
70     scanf("\n%c", &start);
71     if (start == '1'){
72         for(generation = 0; generation < 150; generation++) {
73             usleep(30000);
74             // printf("\033[2J");
75             if (turn == 0){
76                                 for (int j = 0; j < heigh; j++){
77                     for (int i = 0; i < width; i++){
78                         rules(i, j, table_t, table_N);
79                     }
80                 }
81                 display(table_N);
82                 turn = 1;
83 
84             } else if (turn == 1){
85                 for (int j = 0; j < heigh; j++){
86                     for (int i = 0; i < width; i++){
87                         rules(i, j, table_N, table_t);
88                     }
89                 }
90                 display(table_t);
91                 turn = 0;
92             }
93             usleep(30000);
94             printf("\033[r;cH");
95 
96         }
97     }
98 
102    //    SDL_UpdateTexture(texture, NULL, pixels, width * 16);
103    //    SDL_SetRenderDrawColor(renderer, 0, 0, 0, SDL_ALPHA_OPAQUE);
104    //    SDL_RenderClear(renderer);
105    //    SDL_RenderCopy(renderer, texture, NULL, NULL);
106    //    SDL_RenderPresent(renderer);
107    //    SDL_Delay(5);
108    // }
```

```
109    // SDL_DestroyTexture(texture);
110    // SDL_DestroyRenderer(renderer);
111    // SDL_DestroyWindow(window);
112    // SDL_Quit();
113    // return 0;
118 }
```

Here is the call graph for this function:



Index