My Project

Generated by Doxygen 1.8.17

1 File Index 1

1.1 File List . . . 1

2 File Documentation 3

2.1 Source.c File Reference . . . 3

2.2 warcaby.c File Reference . . . 3

2.2.1 Function Documentation . . . 4

2.2.1.1 damka() . . . 4

2.2.1.2 damka_nastepne_bicie() . . . 4

2.2.1.3 graj() . . . 4

2.2.1.4 nowy_wiersz() . . . 4

2.2.1.5 odczytaj_pola() . . . 5

2.2.1.6 rozloz_pionki_gracza() . . . 5

2.2.1.7 sprawdz_koniec() . . . 5

2.2.1.8 sprawdz_wielokrotny_pionek() . . . 5

2.2.1.9 sprawdzaj_ruch() . . . 5

2.2.1.10 utworz_nowa_szachownice() . . . 6

2.2.1.11 wykonaj_ruch() . . . 6

2.2.1.12 wyswietlaj_pionki() . . . 6

Index 7

Chapter 1

File Index

1.1 File List

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

Source.c 3
warcaby.c 3
warcaby.h . . . . . **??**

Chapter 2

File Documentation

2.1 Source.c File Reference

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

Functions

- int **main** (void)

2.2 warcaby.c File Reference

```
#include <stdio.h>
#include <allegro5/allegro5.h>
#include "warcaby.h"
```

Macros

- #define **k** 64

Functions

- void nowy_wiersz (int tab[N][N], int pole_pierwsze, int pole_drugie, int wiersz)
- void utworz_nowa_szachownice (int tablica[N][N])
- void wyswietlaj_pionki (int tab[N][N], ALLEGRO_BITMAP *czarny, ALLEGRO_BITMAP *bialy, ALLEGRO↩ _BITMAP *damka_b, ALLEGRO_BITMAP *damka_cz)
- void rozloz_pionki_gracza (int tab[N][N], int gracz, int rzad_startowy)
- void wykonaj_ruch (int tab[N][N], int pola[N])
- bool sprawdz_koniec (int tab[N][N], int gracz_pionek, int gracz_damka)
- bool damka_nastepne_bicie (int tab[N][N], int pola[N], int przeciwnik, int d_przeciwnik)
- bool damka (int tab[N][N], int pola[N], int przeciwnik, int d_przeciwnik)
- bool sprawdz_wielokrotny_pionek (int tab[N][N], int pola[N], int przeciwnik, int d_przeciwnik)
- int sprawdzaj_ruch (int tab[N][N], int pola[N], int obecny_gracz)
- bool odczytaj_pola (int punkty[N], int szachownica[N][N], int obecny_gracz)
- bool graj ()

2.2.1 Function Documentation

2.2.1.1 damka()

```
bool damka (
            int tab[N][N],
            int pola[N],
            int przeciwnik,
            int d_przeciwnik )
```

Sprawdza poprawność ruchu damki, sprawdza czy zbija tylko jednego przeciwnika, czy obecny gracz nie poruszły się nie swoją damka, zbija ewentualnyego przeciwnika wywołuje funkcje wykonującą ruch, a następnie wywołuje funkcje, która sprawdzi możliwe następne bicie.

2.2.1.2 damka_nastepne_bicie()

```
bool damka_nastepne_bicie (
            int tab[N][N],
            int pola[N],
            int przeciwnik,
            int d_przeciwnik )
```

Funkcja sprawdza możliwość kolejnego zbicia dla damki dla danego gracza, korzysta z współrzędnych na których znajdzie się damka, zwraca true gdy jest taka możliwość

2.2.1.3 graj()

```
bool graj ( )
```

Funkcja zarząda wszystkimi pozostałymi funkcjami, generuje obrazy, pobiera i wczytuje bitmapy pobiera wpółrzędne myszki, wywołuje funkcje odpowiedzialne za ruchy i wyświetlanie obrazu, kończy grę

2.2.1.4 nowy_wiersz()

```
void nowy_wiersz (
            int tab[N][N],
            int pole_pierwsze,
            int pole_drugie,
            int wiersz )
```

Funkcja wypełnia dany wiersz na przemian polami czarnymi i białymi, otrzymuje na przemian argumenty, które decydują o kolejności wypłenienia.

2.2.1.5 odczytaj_pola()

```
bool odczytaj_pola (
            int punkty[N],
            int szachownica[N][N],
            int obecny_gracz )
```

Odczytanie współrzeędnych myszki(pierwsze cztery elementy tablicy punkty) obliczenie z tego pól na szachownicy - elementy 4-7 , następnie wywołuje funkcję sprawdzającą ruch, dla otrzymanych wyyników odpowiednio decyduję jakie zakończenie ruchu wybrać, lub czy jest możliwy jeszcze jeden ruch gracza w tej kolejce, wtedy zwraca false

2.2.1.6 rozloz_pionki_gracza()

```
void rozloz_pionki_gracza (
            int tab[N][N],
            int gracz,
            int rzad_startowy )
```

Funkcja rozkłada pionki dla danego gracza na szachownicy, rozkłada po 3 rzędy pionków, tylko na polach czarnych.

2.2.1.7 sprawdz_koniec()

```
bool sprawdz_koniec (
            int tab[N][N],
            int gracz_pionek,
            int gracz_damka )
```

Sprawdza koniec gry, czyli zwraca true jeśli nie napotka pionka lub damki dla danego gracza.

2.2.1.8 sprawdz_wielokrotny_pionek()

```
bool sprawdz_wielokrotny_pionek (
            int tab[N][N],
            int pola[N],
            int przeciwnik,
            int d_przeciwnik )
```

Funkcja sprawdza możliwość kolejnego zbicia dla pionka w tej samej kolejce danego gracza, jeśli jest możliwe zwraca true.

2.2.1.9 sprawdzaj_ruch()

```
int sprawdzaj_ruch (
            int tab[N][N],
            int pola[N],
            int obecny_gracz )
```

Sprawdza poprawność ruchu pionka, gdy jest to damka wywołuje odpowiednią funkcję, zbija pionka jeśli jest taka możliwość

2.2.1.10 utworz_nowa_szachownice()

```
void utworz_nowa_szachownice (
            int tablica[N][N] )
```

Funkcja utworz_nowa_szachownice tworzy szachownice, wypełniają ją na przemian polami białymi i czarnymi wykorzystuje do tego funkcję nowy_wiersz

2.2.1.11 wykonaj_ruch()

```
void wykonaj_ruch (
            int tab[N][N],
            int pola[N] )
```

Wykonuje ruch dla otrzymanych współrzędnych oraz sprawdza czy nie trzeba zmienić pionka w damkę gdy osiągnie ostatni rząd szachownicy.

2.2.1.12 wyswietlaj_pionki()

```
void wyswietlaj_pionki (
            int tab[N][N],
            ALLEGRO_BITMAP * czarny,
            ALLEGRO_BITMAP * bialy,
            ALLEGRO_BITMAP * damka_b,
            ALLEGRO_BITMAP * damka_cz )
```

Wyświetla pionki na szachownicy, wykorzystując bibliotekę allegro.

Index

damka
    warcaby.c, 4
damka_nastepne_bicie
    warcaby.c, 4

graj
    warcaby.c, 4

nowy_wiersz
    warcaby.c, 4

odczytaj_pola
    warcaby.c, 4

rozloz_pionki_gracza
    warcaby.c, 5

Source.c, 3
sprawdz_koniec
    warcaby.c, 5
sprawdz_wielokrotny_pionek
    warcaby.c, 5
sprawdzaj_ruch
    warcaby.c, 5

utworz_nowa_szachownice
    warcaby.c, 5

warcaby.c, 3
    damka, 4
    damka_nastepne_bicie, 4
    graj, 4
    nowy_wiersz, 4
    odczytaj_pola, 4
    rozloz_pionki_gracza, 5
    sprawdz_koniec, 5
    sprawdz_wielokrotny_pionek, 5
    sprawdzaj_ruch, 5
    utworz_nowa_szachownice, 5
    wykonaj_ruch, 6
    wyswietlaj_pionki, 6
wykonaj_ruch
    warcaby.c, 6
wyswietlaj_pionki
    warcaby.c, 6