/\* A conio implementation for Mingw/Dev-C++.

\*

\* Written by:

\* Hongli Lai <hongli@telekabel.nl>

\* tkorrovi <tkorrovi@altavista.net> on 2002/02/26.

\* Andrew Westcott <ajwestco@users.sourceforge.net>

\*

\* Offered for use in the public domain without any warranty.

\*/

#ifndef \_CONIO\_C\_

#define \_CONIO\_C\_

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <windows.h>

#include <string.h>

#include "conio.h"

#ifdef \_\_cplusplus

extern "C" {

#endif

static int \_\_BACKGROUND = BLACK;

static int \_\_FOREGROUND = LIGHTGRAY;

void

clrscr ()

{

DWORD written;

FillConsoleOutputAttribute (GetStdHandle (STD\_OUTPUT\_HANDLE),

\_\_FOREGROUND + (\_\_BACKGROUND << 4), 2000, (COORD) {0, 0},

&written);

FillConsoleOutputCharacter (GetStdHandle

(STD\_OUTPUT\_HANDLE), ' ',

2000, (COORD) {0, 0}, &written);

gotoxy (1, 1);

}

void

clreol ()

{

COORD coord;

DWORD written;

CONSOLE\_SCREEN\_BUFFER\_INFO info;

GetConsoleScreenBufferInfo (GetStdHandle (STD\_OUTPUT\_HANDLE),

&info);

coord.X = info.dwCursorPosition.X;

coord.Y = info.dwCursorPosition.Y;

FillConsoleOutputCharacter (GetStdHandle (STD\_OUTPUT\_HANDLE),

' ', info.dwSize.X - info.dwCursorPosition.X, coord, &written);

gotoxy (coord.X, coord.Y);

}

void

delline()

{

COORD coord;

DWORD written;

CONSOLE\_SCREEN\_BUFFER\_INFO info;

GetConsoleScreenBufferInfo (GetStdHandle (STD\_OUTPUT\_HANDLE),

&info);

coord.X = info.dwCursorPosition.X;

coord.Y = info.dwCursorPosition.Y;

FillConsoleOutputCharacter (GetStdHandle (STD\_OUTPUT\_HANDLE),

' ', info.dwSize.X \* info.dwCursorPosition.Y, coord, &written);

gotoxy (info.dwCursorPosition.X + 1,

info.dwCursorPosition.Y + 1);

}

int

\_conio\_gettext (int left, int top, int right, int bottom,

char \*str)

{

int i, j, n;

SMALL\_RECT r;

CHAR\_INFO buffer[25][80];

r = (SMALL\_RECT) {left - 1, top - 1, right - 1, bottom - 1};

ReadConsoleOutput (GetStdHandle (STD\_OUTPUT\_HANDLE),

(PCHAR\_INFO) buffer, (COORD) {80, 25}, (COORD) {0, 0}, &r);

lstrcpy (str, "");

for (i = n = 0; i <= bottom - top; i++)

for (j = 0; j <= right - left; j++)

{

str[n] = buffer[i][j].Char.AsciiChar;

n++;

}

str[n] = 0;

return 1;

}

void

gotoxy(int x, int y)

{

COORD c;

c.X = x - 1;

c.Y = y - 1;

SetConsoleCursorPosition (GetStdHandle(STD\_OUTPUT\_HANDLE), c);

}

void

puttext (int left, int top, int right, int bottom, char \*str)

{

int i, j, n;

SMALL\_RECT r;

CHAR\_INFO buffer[25][80];

memset (buffer, 0, sizeof (buffer));

r = (SMALL\_RECT) {left - 1, top - 1, right - 1, bottom - 1};

for (i = n = 0; i <= bottom - top; i++)

for (j = 0; j <= right - left && str[n] != 0; j++)

{

buffer[i][j].Char.AsciiChar = str[n];

buffer[i][j].Attributes = FOREGROUND\_BLUE | FOREGROUND\_GREEN | FOREGROUND\_RED;

n++;

}

WriteConsoleOutput (GetStdHandle (STD\_OUTPUT\_HANDLE),

(CHAR\_INFO \*) buffer, (COORD) {80, 25},

(COORD) {0, 0}, &r);

}

void

\_setcursortype (int type)

{

CONSOLE\_CURSOR\_INFO Info;

if (type)

Info.dwSize = type;

else

{

Info.bVisible = 0 ;

Info.dwSize = 100;

}

SetConsoleCursorInfo (GetStdHandle (STD\_OUTPUT\_HANDLE),

&Info);

}

void

textattr (int \_attr)

{

SetConsoleTextAttribute (GetStdHandle(STD\_OUTPUT\_HANDLE), \_attr);

}

void

textbackground (int color)

{

\_\_BACKGROUND = color;

SetConsoleTextAttribute (GetStdHandle (STD\_OUTPUT\_HANDLE),

\_\_FOREGROUND + (color << 4));

}

void

textcolor (int color)

{

\_\_FOREGROUND = color;

SetConsoleTextAttribute (GetStdHandle (STD\_OUTPUT\_HANDLE),

color + (\_\_BACKGROUND << 4));

}

int

wherex ()

{

CONSOLE\_SCREEN\_BUFFER\_INFO info;

GetConsoleScreenBufferInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &info);

return info.dwCursorPosition.X + 1;

}

int

wherey ()

{

CONSOLE\_SCREEN\_BUFFER\_INFO info;

GetConsoleScreenBufferInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &info);

return info.dwCursorPosition.Y + 1;

}

#ifdef \_\_cplusplus

}

#endif

#endif /\* \_CONIO\_C\_ \*/