3.3 方块自动下落_物联网 / 嵌入式工程师 - 慕课网

> 慕课网慕课教程 3.3 方块自动下落涵盖海量编程基础技术教程，以图文图表的形式，把晦涩难懂的编程专业用语，以通俗易懂的方式呈现给用户。

```
#include<unistd.h>

unsigned int alarm (unsigned int seconds);
功能：设置一个闹钟时间，经过seconds秒后，发送一个SIGALRM信号给当前进程。
      多次调用会刷新闹钟时间。
参数：
@seconds    设置经过seconds秒后发送SIGALRM信号
返回值：如果调用此alarm（）前，进程已经设置了闹钟时间，
       则返回上一个闹钟时间的剩余时间，否则返回0。



#include <sys/time.h>

int setitimer(int which, const struct itimerval *new_value,struct itimerval *old_value);
功能：设置内核定时器
参数：
@which
         ITIMER_REAL    用系统实时的时间计算,时间减1，减到0发送SIGALRM信号
         ITIMER_VIRTUAL 当前进程用户态计数，  计数完成发送SIGVTALRM信号
         ITIMER_PROF 用户态和内核态同时计数，计数完成发送SIGPROF信号
@new_value  当前设置的新值
         struct itimerval
         {
            struct timeval it_interval;
            struct timeval it_value;
         };


         struct timeval
         {
            long tv_sec;
            long tv_usec;
         };
@old_value   保存旧的状态


示例用法：

struct itimerval  val = {{0,500000},{1,0}};

setitimer{ITIMER_REAL, &val, NULL};



#include <signal.h>

typedef void (*sighandler_t)(int);

sighandler_t signal(int signum, sighandler_t handler);
功能：  采用异步的方式捕捉signum的信号，并调用信号处理函数来处理。
参数：
@signum       信号
@handler      对信号的处理方式
              SIG_IGN                  忽略信号
              SIG_DFL                  默认处理
              void (*handler)(int)     用户设置信号处理函数，自行处理
返回值：
成功，返回信号处理函数的地址.
失败，返回SIG_ERR.



#include <stdio.h>
#include <unistd.h>
#include <signal.h>
#include <sys/time.h>

void sig_handler(int signum)
```

```
{
    printf("recv signal\n");
    return ;
}

int main(int argc, const char *argv[])
{

    if(signal(SIGALRM,sig_handler) == SIG_ERR)
    {
        printf("signal call is error!\n");
        return -1;
    }




    struct itimerval val = {{5,0},{1,0}};



    setitimer(ITIMER_REAL,&val,NULL);

    while(1);

    return 0;
}



recv signal
recv signal
recv signal
recv signal



int main()
{
        init_game_ui();


        signal(SIGALRM,sig_handler);


        alarm_us(tm);

        while(1);
        return 0;
}



void sig_handler(int signum)
{

     move_down(dynamic_num,dynamic_mode);

}



int move_down(int num,int mode)
{
    erase_last_shape(num,mode,dynamic_x,dynamic_y);
    dynamic_y++;
    print_mode_shape(num,mode,dynamic_x,dynamic_y,dynamic_color);
}



void alarm_us(int n)
{
        struct itimerval value;


        value.it_value.tv_sec = 0;
        value.it_value.tv_usec = n;


        value.it_interval.tv_sec = 0;
        value.it_interval.tv_usec = n;

        setitimer(ITIMER_REAL,&value,NULL);
}
```

```
#include "user_print.h"

int next_num = 0;
int next_mode = 0;
int next_color = 0;


int init_x = 24;
int init_y = 6;


int next_x = 46;
int next_y = 8;


int dynamic_x = 0;
int dynamic_y = 0;


int dynamic_num = 0;
int dynamic_mode = 0;
int dynamic_color = 0;


int tm = 800000;

int score_x = 45;
int score_y = 18;
int level_x = 45;
int level_y = 22;

int shape[7][4][18] =
{
    {
            {1,1,0,0, 1,1,0,0, 0,0,0,0, 0,0,0,0, 2,2},
            {1,1,0,0, 1,1,0,0, 0,0,0,0, 0,0,0,0, 2,2},
            {1,1,0,0, 1,1,0,0, 0,0,0,0, 0,0,0,0, 2,2},
            {1,1,0,0, 1,1,0,0, 0,0,0,0, 0,0,0,0, 2,2},
    },
    {
            {1,0,0,0, 1,0,0,0, 1,0,0,0, 1,0,0,0, 3,0},
            {1,1,1,1, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,3},
            {1,0,0,0, 1,0,0,0, 1,0,0,0, 1,0,0,0, 3,0},
            {1,1,1,1, 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,3},
    },
    {
            {0,1,0,0, 1,1,1,0, 0,0,0,0, 0,0,0,0, 1,2},
            {1,0,0,0, 1,1,0,0, 1,0,0,0, 0,0,0,0, 2,1},
            {1,1,1,0, 0,1,0,0, 0,0,0,0, 0,0,0,0, 1,2},
            {0,1,0,0, 1,1,0,0, 0,1,0,0, 0,0,0,0, 2,1}
    },
    {
            {1,1,0,0, 0,1,1,0, 0,0,0,0, 0,0,0,0, 1,2},
            {0,1,0,0, 1,1,0,0, 1,0,0,0, 0,0,0,0, 2,1},
            {1,1,0,0, 0,1,1,0, 0,0,0,0, 0,0,0,0, 1,2},
            {0,1,0,0, 1,1,0,0, 1,0,0,0, 0,0,0,0, 2,1},
    },
    {
            {0,1,1,0, 1,1,0,0, 0,0,0,0, 0,0,0,0, 1,2},
            {1,0,0,0, 1,1,0,0, 0,1,0,0, 0,0,0,0, 2,1},
            {0,1,1,0, 1,1,0,0, 0,0,0,0, 0,0,0,0, 1,2},
            {1,0,0,0, 1,1,0,0, 0,1,0,0, 0,0,0,0, 2,1},
    },
    {
            {0,0,1,0, 1,1,1,0, 0,0,0,0, 0,0,0,0, 1,2},
            {1,0,0,0, 1,0,0,0, 1,1,0,0, 0,0,0,0, 2,1},
            {1,1,1,0, 1,0,0,0, 0,0,0,0, 0,0,0,0, 1,2},
            {1,1,0,0, 0,1,0,0, 0,1,0,0, 0,0,0,0, 2,1}
    },
    {
            {1,0,0,0, 1,1,1,0, 0,0,0,0, 0,0,0,0, 1,2},
            {1,1,0,0, 1,0,0,0, 1,0,0,0, 0,0,0,0, 2,1},
            {1,1,1,0, 0,0,1,0, 0,0,0,0, 0,0,0,0, 1,2},
            {0,1,0,0, 0,1,0,0, 1,1,0,0, 0,0,0,0, 2,1}},
};



#ifndef _USER_PRINT_H_
#define _USER_PRINT_H_

extern int next_num;
extern int next_mode;
extern int next_color;

extern int next_num;
```

```
extern int next_mode;
extern int next_color;


extern int next_x;
extern int next_y;


extern int init_x;
extern int init_y;


extern  int dynamic_x;
extern  int dynamic_y;


extern  int dynamic_num;
extern  int dynamic_mode;
extern  int dynamic_color;


extern int shape[7][4][18];

extern void print_mode_shape(int n,int m,int x,int y,int c);
extern void print_next_shape();
extern void erase_last_shape(int n,int m,int a,int b);
extern int move_down(int num,int mode);

#endif



#include <stdio.h>
#include <sys/time.h>
#include <stdlib.h>
#include <signal.h>
#include "user_print.h"




void print_mode_shape(int n,int m,int x,int y,int c)
{
    int i = 0;
    int xx = x;
    int yy = y;
        for(i = 0;i < 16;i++)
        {

                if(i != 0 && i%4 == 0)
                {
                        yy += 1;
                        xx = x;
                }

                if(shape[n][m][i] == 1){
                        printf("\033[%d;%dH\033[%dm[]\033[0m",yy,xx,c);
                }
                xx += 2;
        }
        fflush(NULL);
}




void erase_last_shape(int n,int m,int a,int b)
{
    int i = 0;
    int xx = a;
    int yy = b;

    for(i = 0;i < 16;i++){
            if(i != 0 && i%4 == 0){
                    yy++;
                    xx = a;
            }
            if(shape[n][m][i] == 1){
                    printf("\033[%d;%dH  \033[0m",yy,xx);
            }
            xx += 2;
    }
    fflush(NULL);
```

```
}


void print_next_shape()
{

    erase_last_shape(next_num,next_mode,next_x,next_y);

    next_num = random()%7;
    next_mode = random()%4;
    next_color = random()%7 + 40;


    print_mode_shape(next_num,next_mode,next_x,next_y,next_color);

    fflush(NULL);
}

int move_down(int num,int mode)
{
    erase_last_shape(num,mode,dynamic_x,dynamic_y);
    dynamic_y++;
    print_mode_shape(num,mode,dynamic_x,dynamic_y,dynamic_color);
}
```

```
#ifndef _USER_CONTROL_H_
#define _USER_CONTROL_H_

extern int getch();
extern void alarm_us(int n);

#endif
```

```
#include "user_control.h"
#include "user_print.h"
#include <stdio.h>
#include <termios.h>
#include <signal.h>
#include <time.h>
#include <sys/time.h>
#include <stdlib.h>


int getch()
{
        struct termios tm,tm_old;

        tcgetattr(0,&tm_old);

        cfmakeraw(&tm);

        tcsetattr(0,0,&tm);

        int ch = getchar();

        tcsetattr(0,0,&tm_old);

        return ch;
}


void alarm_us(int n)
{
        struct itimerval value;


        value.it_value.tv_sec = 0;
        value.it_value.tv_usec = n;


        value.it_interval.tv_sec = 0;
        value.it_interval.tv_usec = n;

        setitimer(ITIMER_REAL,&value,NULL);
}
```

```
#include <stdio.h>
#include <termios.h>
#include <signal.h>
#include <time.h>
#include <sys/time.h>
#include <stdlib.h>
```

```
#include "user_print.h"
#include "user_control.h"


extern int tm;

extern int score_x;
extern int score_y;
extern int level_x;
extern int level_y;

void print_start_ui()
{

        printf("\33[2J");
        int i;

        for(i = 0;i < 47;i++){
                printf("\33[%d;%dH\33[43m \33[0m",5,i+10);
                printf("\33[%d;%dH\33[43m \33[0m",30,i+10);
        }

        for(i = 0;i < 26;i++){
                printf("\33[%d;%dH\33[43m  \33[0m",i+5,10);
                printf("\33[%d;%dH\33[43m  \33[0m",i+5,40);
                printf("\33[%d;%dH\33[43m  \33[0m",
                                i+5,56);
        }

        for(i=0;i < 17;i++){
                printf("\33[%d;%dH\33[43m \33[0m",12,40+i);
        }



        printf("\33[%d;%dH分数:\33[0m",score_y,score_x);

        printf("\33[%d;%dH等级:\33[0m",level_y,level_x);

        fflush(NULL);
}

void init_game_ui()
{

         print_start_ui();


        getch();


        srand(time(NULL));

        dynamic_num = random()%7;
        dynamic_mode = random()%4;
        dynamic_color = random()%7+40;

        dynamic_x = init_x;
        dynamic_y = init_y;


        print_mode_shape(dynamic_num,dynamic_mode,dynamic_x,dynamic_y,dynamic_color);

        print_next_shape();
        printf("\33[?25l");
}


void sig_handler(int signum)
{

     move_down(dynamic_num,dynamic_mode);

}

int main()
{
    init_game_ui();


    signal(SIGALRM,sig_handler);


    alarm_us(tm);

    while(1);
```

```
    return 0;
}
```

全文完