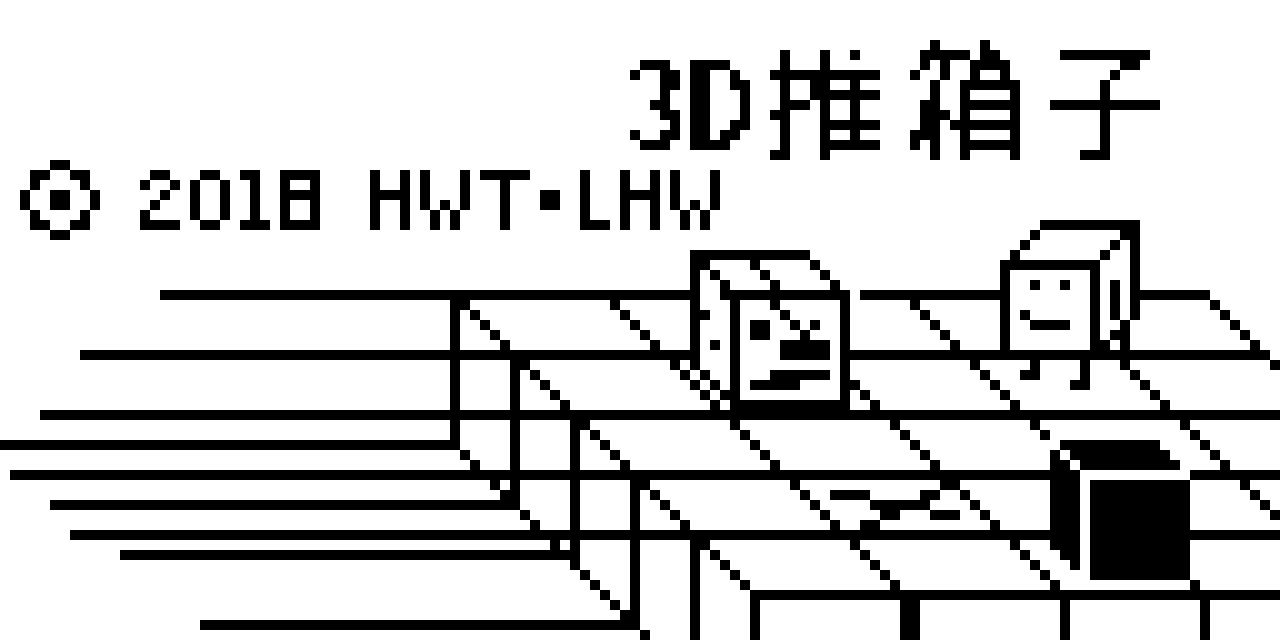
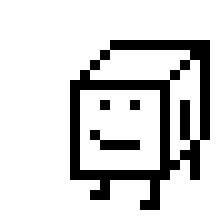
# The Sokoban



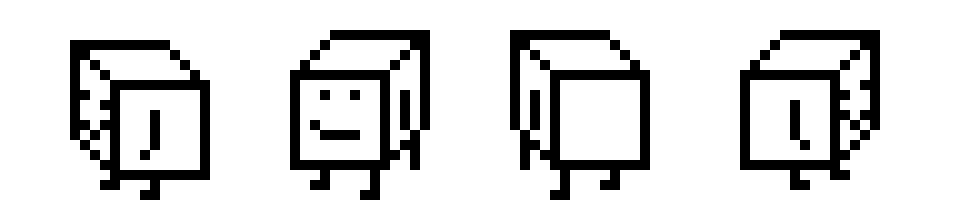


## boxtagHow to play: the control box man pushes the delivery box to the designated place

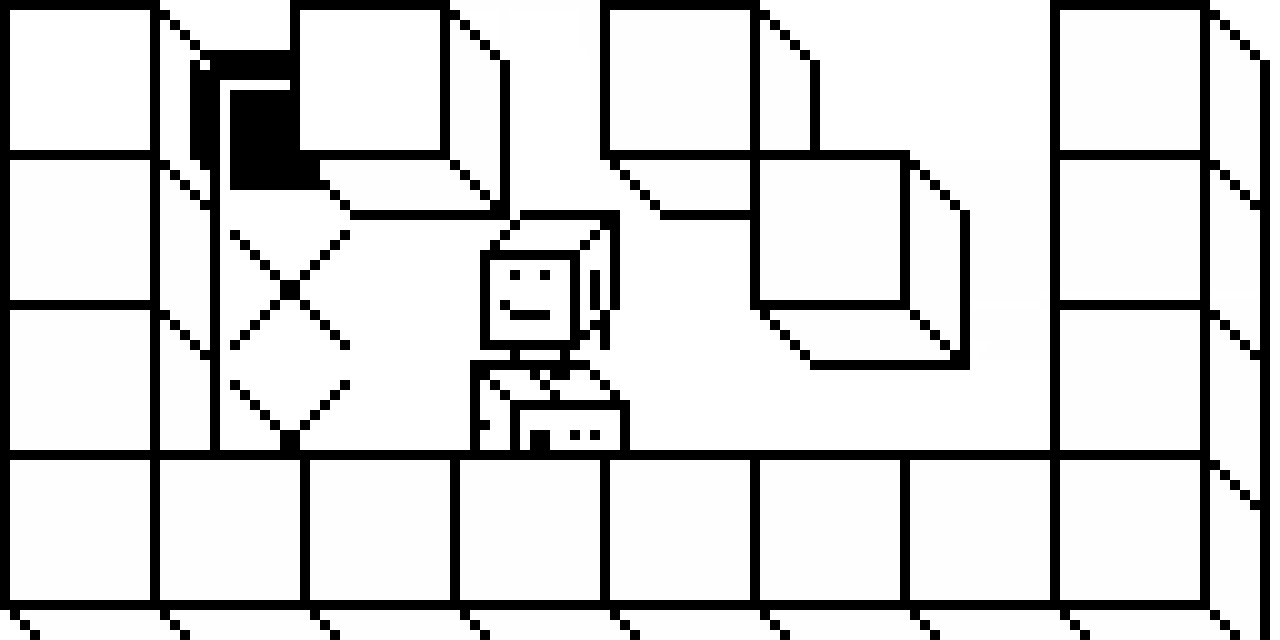
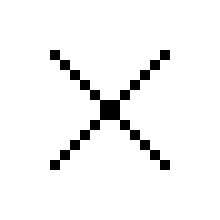
**Key control villain**

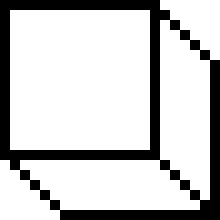
## A + Up and down keys jump level

## B key resets the level



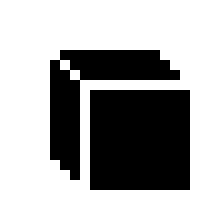
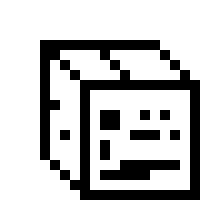
## introduce:





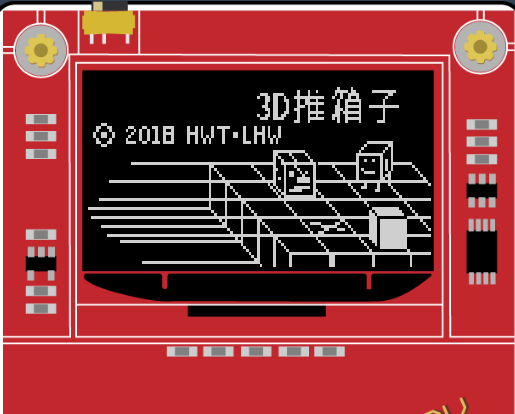
**WALL TAB**

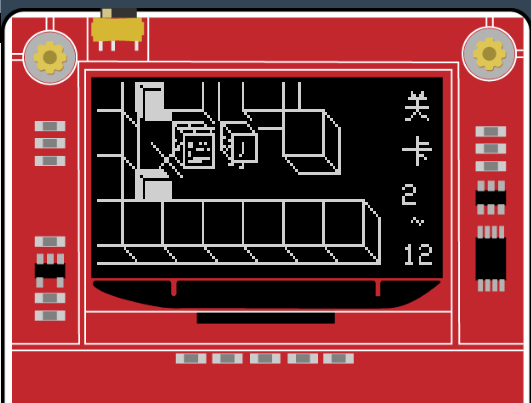
# 



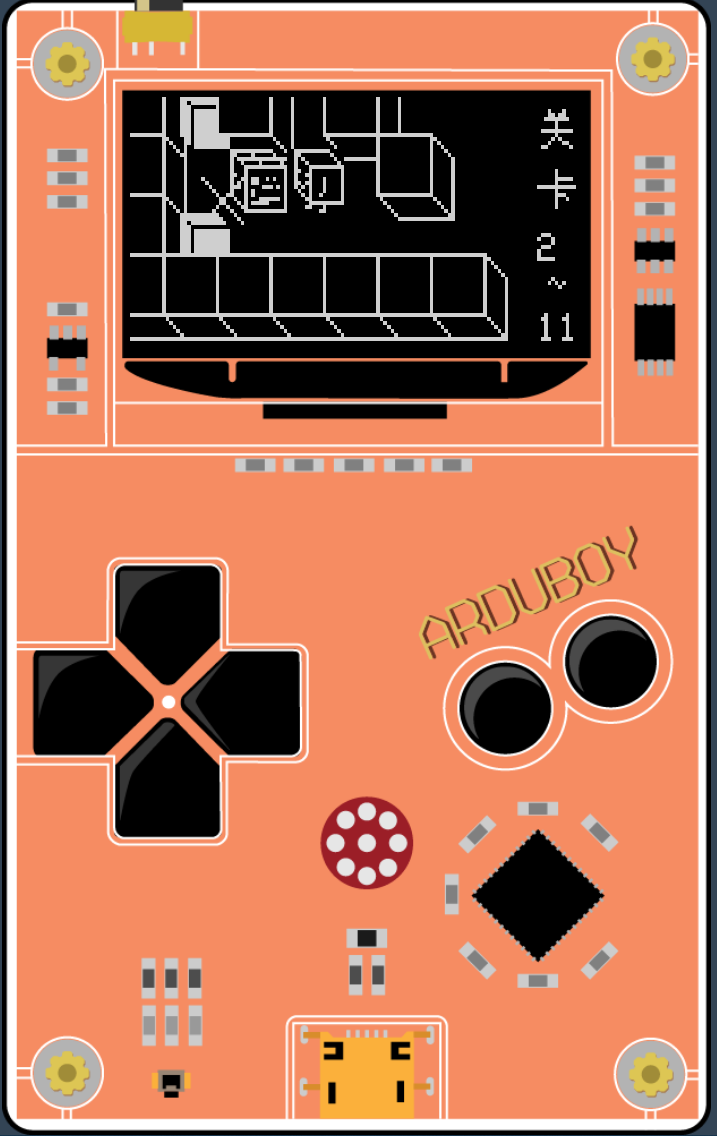
**BOX**

# screenshots:





# Key description:



**Control**

**Of**

**The**

**mobile**

**JUMP LEVEL**

**B：RESET**

**A：ENTER**

# 游戏代码:

#include <Arduboy2.h>

Arduboy2 arduboy**;**

/\*=========================================================

位图

=========================================================\*/

//启动画面

const uint8\_t BOOT**[]** PROGMEM **=** **{**

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,**

篇幅过长，省略

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x84**,** 0x10**,** 0x00**,** 0x30**,** 0x00**,** 0x20**,** 0x00**,** 0x80

**};**

//墙的贴图

const uint8\_t WALL**[]** PROGMEM **=** **{**

0xff**,** 0xff**,** 0x00**,** 0x80**,** 0x01**,** 0x80**,** 0x80**,** 0x01**,** 0x40**,** 0x80**,** 0x01**,** 0x20**,** 0x80**,** 0x01**,** 0x10**,** 0x80**,** 0x01**,** 0x08**,** 0x80**,** 0x01**,** 0x04**,** 0x80**,** 0x01**,** 0x04**,** 0x80**,** 0x01**,** 0x04**,** 0x80**,** 0x01**,** 0x04**,** 0x80**,** 0x01**,** 0x04**,** 0x80**,** 0x01**,** 0x04**,** 0x80**,** 0x01**,** 0x04**,** 0x80**,** 0x01**,** 0x04**,** 0x80**,** 0x01**,** 0x04**,** 0xff**,** 0xff**,** 0x04**,** 0x40**,** 0x00**,** 0x84**,** 0x20**,** 0x00**,** 0x44**,** 0x10**,** 0x00**,** 0x24**,** 0x08**,** 0x00**,** 0x14**,** 0x04**,** 0x00**,** 0x0c**,** 0x03**,** 0xff**,** 0xfc

**};**

//箱子的贴图

const uint8\_t BOX**[]** PROGMEM **=** **{**

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x0f**,** 0xff**,** 0x00**,** 0x0c**,** 0x20**,** 0x80**,** 0x0a**,** 0x10**,** 0x40**,** 0x09**,** 0x08**,** 0x20**,** 0x08**,** 0xff**,** 0xf0**,** 0x08**,** 0x80**,** 0x10**,** 0x0c**,** 0x80**,** 0x10**,** 0x08**,** 0xb2**,** 0x90**,** 0x08**,** 0xb0**,** 0x10**,** 0x0a**,** 0x87**,** 0x50**,** 0x08**,** 0xa0**,** 0x10**,** 0x08**,** 0xa0**,** 0x10**,** 0x04**,** 0x8f**,** 0xd0**,** 0x02**,** 0xbe**,** 0x10**,** 0x01**,** 0x80**,** 0x10**,** 0x00**,** 0xff**,** 0xf0**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00

**};**

//打开的箱子贴图

const uint8\_t OBOX**[]** PROGMEM **=** **{**

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x03**,** 0xff**,** 0x00**,** 0x05**,** 0xff**,** 0x80**,** 0x06**,** 0xff**,** 0xc0**,** 0x07**,** 0x00**,** 0x00**,** 0x07**,** 0x7f**,** 0xe0**,** 0x07**,** 0x7f**,** 0xe0**,** 0x07**,** 0x7f**,** 0xe0**,** 0x07**,** 0x7f**,** 0xe0**,** 0x07**,** 0x7f**,** 0xe0**,** 0x07**,** 0x7f**,** 0xe0**,** 0x07**,** 0x7f**,** 0xe0**,** 0x03**,** 0x7f**,** 0xe0**,** 0x01**,** 0x7f**,** 0xe0**,** 0x00**,** 0x7f**,** 0xe0**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00

**};**

//小人朝上的贴图

const uint8\_t ManUp**[]** PROGMEM **=** **{**

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x01**,** 0xff**,** 0x80**,** 0x01**,** 0x80**,** 0x40**,** 0x01**,** 0x40**,** 0x20**,** 0x01**,** 0x20**,** 0x10**,** 0x01**,** 0x1f**,** 0xf8**,** 0x01**,** 0x10**,** 0x08**,** 0x01**,** 0x50**,** 0x08**,** 0x01**,** 0x50**,** 0x08**,** 0x01**,** 0x50**,** 0x08**,** 0x01**,** 0x50**,** 0x08**,** 0x00**,** 0x90**,** 0x08**,** 0x00**,** 0xd0**,** 0x08**,** 0x00**,** 0xb0**,** 0x08**,** 0x00**,** 0x9f**,** 0xf8**,** 0x00**,** 0x08**,** 0x40**,** 0x00**,** 0x08**,** 0xc0**,** 0x00**,** 0x18**,** 0x00**,** 0x00**,** 0x00**,** 0x00

**};**

//小人朝下的贴图

const uint8\_t ManDown**[]** PROGMEM **=** **{**

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x1f**,** 0xf8**,** 0x00**,** 0x20**,** 0x18**,** 0x00**,** 0x40**,** 0x28**,** 0x00**,** 0x80**,** 0x48**,** 0x01**,** 0xff**,** 0x88**,** 0x01**,** 0x00**,** 0x88**,** 0x01**,** 0x24**,** 0xa8**,** 0x01**,** 0x00**,** 0xa8**,** 0x01**,** 0x00**,** 0xa8**,** 0x01**,** 0x40**,** 0xa8**,** 0x01**,** 0x3c**,** 0x90**,** 0x01**,** 0x00**,** 0xb0**,** 0x01**,** 0x00**,** 0xd0**,** 0x01**,** 0xff**,** 0x90**,** 0x00**,** 0x21**,** 0x00**,** 0x00**,** 0x61**,** 0x00**,** 0x00**,** 0x03**,** 0x00**,** 0x00**,** 0x00**,** 0x00

**};**

//小人朝左的贴图

const uint8\_t ManLeft**[]** PROGMEM **=** **{**

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x01**,** 0xff**,** 0x80**,** 0x01**,** 0x80**,** 0x40**,** 0x01**,** 0x40**,** 0x20**,** 0x01**,** 0x20**,** 0x10**,** 0x01**,** 0x1f**,** 0xf8**,** 0x01**,** 0x90**,** 0x08**,** 0x01**,** 0x30**,** 0x08**,** 0x01**,** 0x11**,** 0x08**,** 0x01**,** 0xb1**,** 0x08**,** 0x01**,** 0x51**,** 0x08**,** 0x00**,** 0x91**,** 0x08**,** 0x00**,** 0x52**,** 0x08**,** 0x00**,** 0x30**,** 0x08**,** 0x00**,** 0x1f**,** 0xf8**,** 0x00**,** 0x31**,** 0x00**,** 0x00**,** 0x03**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00

**};**

//小人朝右的贴图

const uint8\_t ManRight**[]** PROGMEM **=** **{**

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x1f**,** 0xf8**,** 0x00**,** 0x20**,** 0x18**,** 0x00**,** 0x40**,** 0x28**,** 0x00**,** 0x80**,** 0x48**,** 0x01**,** 0xff**,** 0x88**,** 0x01**,** 0x00**,** 0x98**,** 0x01**,** 0x00**,** 0xc8**,** 0x01**,** 0x08**,** 0x88**,** 0x01**,** 0x08**,** 0xd8**,** 0x01**,** 0x08**,** 0xa8**,** 0x01**,** 0x08**,** 0x90**,** 0x01**,** 0x04**,** 0xa0**,** 0x01**,** 0x00**,** 0xc0**,** 0x01**,** 0xff**,** 0x80**,** 0x00**,** 0x08**,** 0xc0**,** 0x00**,** 0x0c**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00

**};**

//“关卡”中文字

const uint8\_t TCP**[]** PROGMEM **=** **{**

0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x04**,** 0x40**,** 0x06**,** 0xc0**,** 0x0f**,** 0xe0**,** 0x01**,** 0x00**,** 0x01**,** 0x00**,** 0x1f**,** 0xf0**,** 0x01**,** 0x00**,** 0x02**,** 0x80**,** 0x04**,** 0x60**,** 0x18**,** 0x10**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x01**,** 0x00**,** 0x01**,** 0x00**,** 0x01**,** 0xf0**,** 0x01**,** 0x00**,** 0x3f**,** 0xf8**,** 0x01**,** 0x00**,** 0x01**,** 0xc0**,** 0x01**,** 0x20**,** 0x01**,** 0x00**,** 0x01**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00**,** 0x00

**};**

/\*=========================================================

地图

=========================================================\*/

const PROGMEM byte MAP**[]** **=** **{**

//0->空白

//1->墙

//2->人

//3->箱子

//4->位置

//5->放到正确位置的箱子

//6->在标签上的人

//每个地图大小16\*16 可以自行添加

//第1关

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 1**,** 1**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 1**,** 4**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 1**,** 1**,** 1**,** 1**,** 0**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 1**,** 4**,** 0**,** 3**,** 3**,** 1**,** 1**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 1**,** 1**,** 1**,** 2**,** 0**,** 3**,** 4**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 1**,** 3**,** 1**,** 1**,** 1**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 1**,** 4**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 1**,** 1**,** 1**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0**,** 0 **,**

#### 节约篇幅 省略剩下关卡

**};**

/\*=========================================================

变量

=========================================================\*/

byte CP **=** 1**;** //关卡

byte ACP **=** **(sizeof(**MAP**)** **/** **sizeof(**MAP**[**0**]))** **/** 256**;** //计算一共有多少关卡，规定每关地图为16x16 ，每由256个数据组成

int RMAP**[**256**];** //运行地图 当载入当前关卡时，会读取内存把地图放进去

int LMAP**[**28**];** //视野地图 显示区域内的地图，设置这个是为了能够快速刷新

int Atmp**;** //数据缓存

int Btmp**;** //数据缓存

int Ctmp**;** //数据缓存

byte DX**;**

byte DY**;**

byte P **=** 0**;** //玩家在地图中的位置

char PlayerD **=** **-**16**;** //最后方向 默认-16 也就是下

boolean LA **=** **false;** //行动是否合法

boolean WIN **=** **false;** //检测是否通关

int X **=** 1**;**

int Y **=** 1**;**

/\*=========================================================

只循环一次

=========================================================\*/

void setup**()** **{**

arduboy**.**boot**();**

arduboy**.**drawSlowXYBitmap**(**0**,** 0**,** BOOT**,** 128**,** 64**,** 1**);**

arduboy**.**display**();**

delay**(**1500**);**

//加载地图

load\_cp**();** //把地图载入内存

load\_LMAP**();**//载入视野地图

draw**();** //渲染画面

**}**/\*=========================================================

不停循环

=========================================================\*/

void loop**()** **{**

LA **=** **false;** //是否刷新移动 否

**if** **(**arduboy**.**pressed**(**A\_BUTTON**))** **{** //如果A键按下

**if** **(**arduboy**.**pressed**(**UP\_BUTTON**))** **{** //上键与A键同时按下

**if** **(**CP **<** ACP**)** CP**++;** //如果不是最后一关 让关卡数+1

Load**();** //载入关卡函数

**}**

**if** **(**arduboy**.**pressed**(**DOWN\_BUTTON**))** **{** //下键与A键同时按下

**if** **(**CP **>** 1**)** CP**--;** //如果不是第一关 让关卡数-1

Load**();** //载入关卡函数

**}**

**}**

**if** **(**arduboy**.**pressed**(**B\_BUTTON**))** **{** //如果B键按下 （B键作用是重置当前关卡）

load\_cp**();** //把当前关卡地图载入内存

load\_LMAP**();** //计算视野地图

draw**();** //渲染画面

loop**();** //回到游戏循环 开始新一轮游戏

**}**

//键盘控制玩家部分

**if** **(**arduboy**.**pressed**(**UP\_BUTTON**))** **{**//上键激活

PlayerD **=** **-**16**;**//设置玩家方向 至于为什么是这个下面有讲 不用着急

**if** **(**RMAP**[**P **-** 16**]** **==** 0 **||** RMAP**[**P **-** 16**]** **==** 3 **||** RMAP**[**P **-** 16**]** **==** 4 **||** RMAP**[**P **-** 16**]** **==** 5 **||** RMAP**[**P **-** 16**]** **==** 6**)** **{**//检查对应位置是否可以走

LA **=** **true;**//移动合法 计算移动 是

**}** **else** **{**

draw**();**

**}**

**}** **else** **{**

**if** **(**arduboy**.**pressed**(**DOWN\_BUTTON**))** **{**//下键激活

PlayerD **=** 16**;**//设置玩家方向 至于为什么是这个下面有讲 不用着急

**if** **(**RMAP**[**P **+** 16**]** **==** 0 **||** RMAP**[**P **+** 16**]** **==** 3 **||** RMAP**[**P **+** 16**]** **==** 4 **||** RMAP**[**P **+** 16**]** **==** 5 **||** RMAP**[**P **+** 16**]** **==** 6**)** **{** //检查对应位置是否可以走

LA **=** **true;**//移动合法 计算移动 是

**}** **else** **{**

draw**();**

**}**

**}** **else** **{**

**if** **(**arduboy**.**pressed**(**LEFT\_BUTTON**))** **{**//左键激活

PlayerD **=** **-**1**;** //设置玩家方向 至于为什么是这个下面有讲 不用着急

**if** **(**RMAP**[**P **-** 1**]** **==** 0 **||** RMAP**[**P **-** 1**]** **==** 3 **||** RMAP**[**P **-** 1**]** **==** 4 **||** RMAP**[**P **-** 1**]** **==** 5 **||** RMAP**[**P **-** 1**]** **==** 6**)** **{**//检查对应位置是否可以走

LA **=** **true;**//移动合法 计算移动 是

**}** **else** **{**

draw**();**

**}**

**}** **else** **{**

**if** **(**arduboy**.**pressed**(**RIGHT\_BUTTON**))** **{** //右键激活

PlayerD **=** 1**;**//设置玩家方向 至于为什么是这个下面有讲 不用着急

**if** **(**RMAP**[**P **+** 1**]** **==** 0 **||** RMAP**[**P **+** 1**]** **==** 3 **||** RMAP**[**P **+** 1**]** **==** 4 **||** RMAP**[**P **+** 1**]** **==** 5 **||** RMAP**[**P **+** 1**]** **==** 6**)** **{** //检查对应位置是否可以走

LA **=** **true;** //移动合法 计算移动 是

**}** **else** **{**

draw**();**

**}**

**}**

**}**

**}**

**}**

/\*=========================================================

移动逻辑判断

=========================================================\*/

**if** **(**LA **==** **true)** **{**

//行动合法

//如果脚下是空的

**if** **(**RMAP**[**P**]** **==** 2**)** **{**

**if** **(**RMAP**[**P **+** PlayerD**]** **==** 0**)** **{**

//如果前方是空气

RMAP**[**P **+** PlayerD**]** **=** 2**;**

RMAP**[**P**]** **=** 0**;**

**}** **else** **{**

//如果前方是标记

**if** **(**RMAP**[**P **+** PlayerD**]** **==** 4**)** **{**

RMAP**[**P **+** PlayerD**]** **=** 6**;**

RMAP**[**P**]** **=** 0**;**

**}** **else** **{**

//如果前面不是标记而是标记上有箱子

**if** **(**RMAP**[**P **+** PlayerD**]** **==** 5**)** **{**

//如果前方标签上的箱子前面是空气

**if** **(**RMAP**[**P **+** 2 **\*** PlayerD**]** **==** 0**)** **{**

RMAP**[**P **+** 2 **\*** PlayerD**]** **=** 3**;**

RMAP**[**P **+** PlayerD**]** **=** 6**;**

RMAP**[**P**]** **=** 0**;**

**}** **else** **{**

//如果前方标签上的箱子前面是标签

**if** **(**RMAP**[**P **+** 2 **\*** PlayerD**]** **==** 4**)** **{**

RMAP**[**P **+** 2 **\*** PlayerD**]** **=** 5**;**

RMAP**[**P **+** PlayerD**]** **=** 6**;**

RMAP**[**P**]** **=** 0**;**

**}**

**}**

**}** **else** **{** //前方是箱子

**if** **(**RMAP**[**P **+** PlayerD**]** **==** 3**)** **{**

//箱子前面是空气

**if** **(**RMAP**[**P **+** 2 **\*** PlayerD**]** **==** 0**)** **{**

RMAP**[**P **+** 2 **\*** PlayerD**]** **=** 3**;**

RMAP**[**P **+** PlayerD**]** **=** 2**;**

RMAP**[**P**]** **=** 0**;**

**}** **else** **{**

**if** **(**RMAP**[**P **+** 2 **\*** PlayerD**]** **==** 4**)** **{**

//箱子前面是标记

RMAP**[**P **+** 2 **\*** PlayerD**]** **=** 5**;**

RMAP**[**P **+** PlayerD**]** **=** 2**;**

RMAP**[**P**]** **=** 0**;**

**}**

**}**

**}**

**}**

**}**

**}**

**}** **else** **{**

//如果前面不是空气并且脚下有标记

**if** **(**RMAP**[**P**]** **==** 6**)** **{**

**if** **(**RMAP**[**P **+** PlayerD**]** **==** 0**)** **{**

//如果前方是空气

RMAP**[**P **+** PlayerD**]** **=** 2**;**

RMAP**[**P**]** **=** 4**;**

**}** **else** **{**

//如果前方依然是是标记

**if** **(**RMAP**[**P **+** PlayerD**]** **==** 4**)** **{**

RMAP**[**P **+** PlayerD**]** **=** 6**;**

RMAP**[**P**]** **=** 4**;**

**}** **else** **{**

//如果前方标签上的箱子前面是空气

**if** **(**RMAP**[**P **+** 2 **\*** PlayerD**]** **==** 0**)** **{**

RMAP**[**P **+** 2 **\*** PlayerD**]** **=** 3**;**

RMAP**[**P **+** PlayerD**]** **=** 2**;**

RMAP**[**P**]** **=** 4**;**

**}** **else** **{**

//如果前方标签上的箱子前面是标签

**if** **(**RMAP**[**P **+** 2 **\*** PlayerD**]** **==** 4**)** **{**

RMAP**[**P **+** 2 **\*** PlayerD**]** **=** 5**;**

RMAP**[**P **+** PlayerD**]** **=** 6**;**

RMAP**[**P**]** **=** 4**;**

**}** **else** **{**

//如果前方只有箱子，箱子前面但是脚下有标签

**if** **(**RMAP**[**P **+** PlayerD**]** **==** 3**)** **{**

//箱子前面是空气

**if** **(**RMAP**[**P **+** 2 **\*** PlayerD**]** **==** 0**)** **{**

RMAP**[**P **+** 2 **\*** PlayerD**]** **=** 3**;**

RMAP**[**P **+** PlayerD**]** **=** 2**;**

RMAP**[**P**]** **=** 4**;**

**}** **else** **{**

//箱子前面是标签

**if** **(**RMAP**[**P **+** 2 **\*** PlayerD**]** **==** 4**)** **{**

RMAP**[**P **+** 2 **\*** PlayerD**]** **=** 5**;**

RMAP**[**P **+** PlayerD**]** **=** 2**;**

RMAP**[**P**]** **=** 4**;**

**}**

**}**

**}**

**}**

**}**

**}**

**}**

**}**

**}**

load\_LMAP**();**  //刷新视野地图

draw**();**  //渲染画面

**}**

**if** **(**WIN **==** **true)** win**();**  //检测到通关 调用通关函数

**}**

/\*=========================================================

显示函数

=========================================================\*/

void draw**()** **{**

arduboy**.**clear**();** //清除显存

arduboy**.**drawSlowXYBitmap**(**112**,** 0**,** TCP**,** 16**,** 32**,** 1**);**//关卡中文字体

arduboy**.**setCursor**(**114**,** 34**);** //设置光标

arduboy**.**print**(**CP**);** //显示当前关卡

arduboy**.**setCursor**(**117**,** 45**);** //设置光标

arduboy**.**print**(**"~"**);** //显示分隔符

arduboy**.**setCursor**(**114**,** 54**);** //设置光标

arduboy**.**print**(**ACP**);** //显示一共有多少关

DrawMap**();** //渲染游戏地图

arduboy**.**display**();** //把画面显示到OLED上

**}**/\*=========================================================

载入当前关卡

=========================================================\*/

void load\_cp**()** **{**

Atmp **=** 0**;** //初始化写入地图内存的位置

Ctmp **=** **(**CP **-** 1**)** **\*** 256**;** //设置读取地图结束位置

**for** **(**Btmp **=** Ctmp**;** Btmp **<=** Ctmp **+** 255**;** Btmp**++)** **{** //循环读取

RMAP**[**Atmp**]** **=** pgm\_read\_byte\_near**(**MAP **+** Btmp**);** //读取相对应位置的地图，写入当前关卡地图内存

Atmp**++;** //地图内存地址位置+1

**}**

**}**/\*=========================================================

载入当前区块视野地图

=========================================================\*/

void load\_LMAP**()** **{**

/\*=========================================================

遍历内存检查玩家位置和通关情况

=========================================================\*/

WIN **=** **true;** //假设玩家通关

**for** **(**int i **=** 0**;** i **<=** 255**;** i**++)** **{** //遍历关卡地图内存;0到255有256个数字,因为规定一关地图大小为16x16所以为256

**if** **(**RMAP**[**i**]** **==** 2**)** **{** //当前是玩家所在的位置 重设玩家所在地图位置

P **=** i**;**

**}** **else** **{**

**if** **(**RMAP**[**i**]** **==** 6**)** P **=** i**;** //当前是玩家所在的位置 重设玩家所在地图位置

**}**

**if** **(**RMAP**[**i**]** **==** 3**)** WIN **=** **false;** //检测到没有放在指定位置的箱子 表明玩家没有通关

**}**

Atmp **=** P **-** 19**;** //视野地图最开始的位置 是关卡地图(16x16大地图)中 玩家位置-19的地方，可以参考下图颜色加深位置（119-110=19）说白了，这个是用来放视野地图最开始的方块在大地图中的位置

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 |
| 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |
| 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 |
| 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 |
| 96 | 97 | 98 | 99 | 100 | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 |
| 112 | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | 123 | 124 | 125 | 126 | 127 |
| 128 | 129 | 130 | 131 | 132 | 133 | 134 | 135 | 136 | 137 | 138 | 139 | 140 | 141 | 142 | 143 |
| 144 | 145 | 146 | 147 | 148 | 149 | 150 | 151 | 152 | 153 | 154 | 155 | 156 | 157 | 158 | 159 |
| 160 | 161 | 162 | 163 | 164 | 165 | 166 | 167 | 168 | 169 | 170 | 171 | 172 | 173 | 174 | 175 |
| 176 | 177 | 178 | 179 | 180 | 181 | 182 | 183 | 184 | 185 | 186 | 187 | 188 | 189 | 190 | 191 |
| 192 | 193 | 194 | 195 | 196 | 197 | 198 | 199 | 200 | 201 | 202 | 203 | 204 | 205 | 206 | 207 |
| 208 | 209 | 210 | 211 | 212 | 213 | 214 | 215 | 216 | 217 | 218 | 219 | 220 | 221 | 222 | 223 |
| 224 | 225 | 226 | 227 | 228 | 229 | 230 | 231 | 232 | 233 | 234 | 235 | 236 | 237 | 238 | 239 |
| 240 | 241 | 242 | 243 | 244 | 245 | 246 | 247 | 248 | 249 | 250 | 251 | 252 | 253 | 254 | 255 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 |

说明：绿色指的是16x16的关卡地图RMAP

蓝色指的是视野地图LMAP

橙色指的是玩家位置

int i **=** 0**;** //i为视野地图LMAP的位置 初始化为0

**for** **(**Btmp **=** P **-** 19**;** Btmp **<=** P **+** 35**;** Btmp**++)** **{** //Btmp为大地图位置

**if** **(**Btmp **>=** Atmp **+** 7**)** **{**

Btmp **+=** 9**;**

Atmp **+=** 16**;**

**}**

LMAP**[**i**]** **=** RMAP**[**Btmp**];** //把大地图蓝色标记的载入视野地图

i**++;** //视野地图位置+1

**}**

**}**

/\*=========================================================

渲染地图

=========================================================\*/

void DrawMap**()** **{**

DX **=** 0**;**

DY **=** 0**;**

**for** **(**byte i **=** 0**;** i **<=** 27**;** i**++)** **{** //遍历视野地图

//0->空白

//1->墙

//2->人

//3->箱子

//4->位置

//5->放到正确位置的箱子

//6->在标签上的人

**if** **(**LMAP**[**i**]** **==** 1**)** **{** //当前位置是墙

//画墙

u8g**.**setColorIndex**(**0**);** //设置背景色为黑色

u8g**.**drawBox**(**DX **-** 6**,** DY **-** 6**,** 16**,** 16**);** //画一个黑色的长方形覆盖画面

u8g**.**drawLine**(**DX**,** DY **+** 10**,** DX**,** DY **+** 14**);** //把多出来的线条去掉

u8g**.**drawLine**(**DX **+** 10**,** DY**,** DX **+** 14**,** DY**);** //把多出来的线条去掉

u8g**.**setColorIndex**(**1**);** //设置背景色为白色

u8g**.**drawBitmapP**(**DX **-** 6**,** DY **-** 6**,** 3**,** 22**,** WALL**);**//显示墙的图片

**}**

**if** **(**LMAP**[**i**]** **==** 2 **||** LMAP**[**i**]** **==** 6**)** **{** //当前是玩家的位置

DrawPlayer**();** //渲染玩家

**}**

**if** **(**LMAP**[**i**]** **==** 3**)** **{** //如果当前位置是箱子

DrawBox**();**

**}**

**if** **(**LMAP**[**i**]** **==** 5**)** **{** //当前位置是放到标记上的箱子

u8g**.**drawBitmapP**(**DX **-** 6**,** DY **-** 6**,** 3**,** 22**,** OBOX**);**

**}**

**if** **(**LMAP**[**i**]** **==** 4**)** **{** //当前位置是标记(“X”)

DrawTag**();**

**}**

//计算坐标 进行移位

DX **+=** 15**;**

**if** **(**DX **>** 90**)** **{**

DX **=** 0**;**

DY **+=** 15**;**

**}**

**}**

**}**

/\*=========================================================

绘图函数

=========================================================\*/

void DrawPlayer**()** **{**

//画玩家

u8g**.**setColorIndex**(**0**);** //设置背景色为黑色

u8g**.**drawBox**(**DX **-** 1**,** DY **-** 2**,** 16**,** 17**);** //画一个黑色的长方形覆盖指定位置画面

u8g**.**setColorIndex**(**1**);** //设置背景色为白色

//-16为上 16为下 -1为左 +1为右 这样设定是因为地图是一个16x16的一维数组 为了获得相对应位置的地图方块信息要进行地图位置地址移位

**if** **(**PlayerD **==** **-**16**)** **{** //如果玩家方向为上

u8g**.**drawBitmapP**(**DX **-** 6**,** DY **-** 6**,** 3**,** 22**,** ManUp**);** //玩家向上图片

**}** **else** **{**

**if** **(**PlayerD **==** 16**)** **{** //如果玩家方向为下

u8g**.**drawBitmapP**(**DX **-** 6**,** DY **-** 6**,** 3**,** 22**,** ManDown**);** //玩家向下图片

**}** **else** **{**

**if** **(**PlayerD **==** **-**1**)** **{** //如果玩家方向为左

u8g**.**drawBitmapP**(**DX **-** 6**,** DY **-** 6**,** 3**,** 22**,** ManLeft**);** //玩家向左图片

**}** **else** **{** //如果玩家方向不是上 也不是下 更不是左 （右）

u8g**.**drawBitmapP**(**DX **-** 6**,** DY **-** 6**,** 3**,** 22**,** ManRight**);** //玩家向右图片

**}**

**}**

**}**

**}**

void DrawBox**()** **{**

//画箱子

u8g**.**drawBitmapP**(**DX **-** 6**,** DY **-** 6**,** 3**,** 22**,** BOX**);** //在当前位置渲染箱子图片

**}**

void DrawTag**()** **{**

//画标记

u8g**.**drawLine**(**DX **+** 5**,** DY **+** 5**,** DX **+** 16**,** DY **+** 16**);** //画一条斜线

u8g**.**drawLine**(**DX **+** 16**,** DY **+** 5**,** DX **+** 5**,** DY **+** 16**);** //画一条斜线 两条斜线组成一个”X”

**}**

/\*=========================================================

下一关

=========================================================\*/

void win**()** **{**

u8g**.**firstPage**();**

**do** **{**

DrawMap**();** //先渲染地图

u8g**.**setFont**(**u8g\_font\_courB12**);** //设置字体

u8g**.**setColorIndex**(**0**);** //设置背景色为黑色

u8g**.**drawBox**(**3**,** 12**,** 86**,** 30**);** //画一个黑色的长方形 用于覆盖画面

u8g**.**setColorIndex**(**1**);** //设置字体为白色

u8g**.**setPrintPos**(**11**,** 32**);** //设置下次绘图的坐标

u8g**.**print**(**"You Win"**);** //打印 你赢了

**}** **while** **(** u8g**.**nextPage**()** **);**

delay **(**1000**);** //等待一秒

**if** **(**CP **<** ACP**)** **{** //如果当前关卡少于总关卡

CP**++;** //那么关卡数+1

**}** **else** **{** //否则如果是最后一关

CP **=** 1**;** //切换回去第一关

**}**

Load**();** //载入地图函数

**}**

void Load**()** **{** //载入函数

load\_cp**();** //把地图载入内存

load\_LMAP**();** //计算视野地图

draw**();** //渲染画面

loop**();** //回到最开始的循环 开始游戏

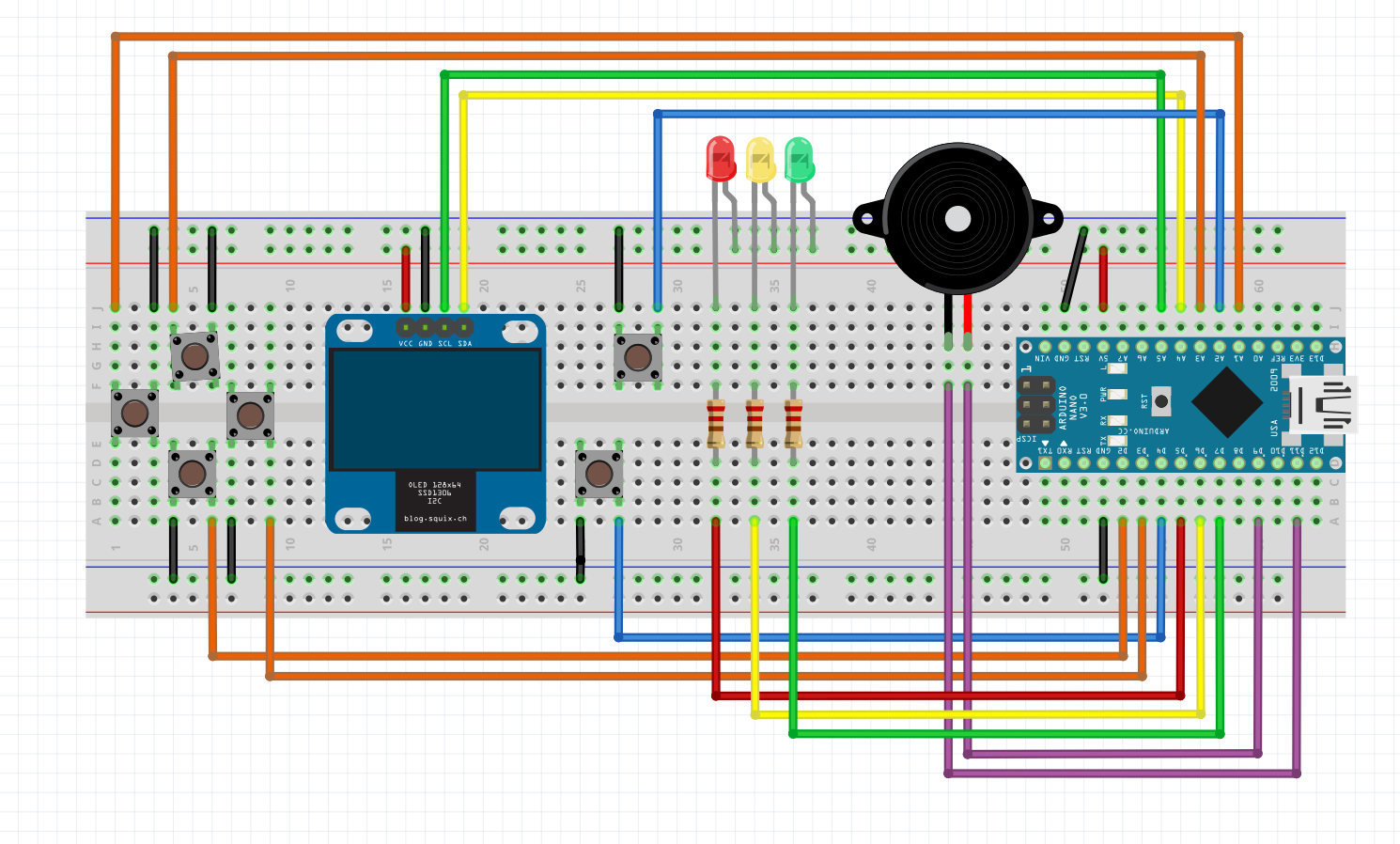
**}**

# 如何搭建在Arduino UNO上:

游戏采用了Arduboy相关库文件 ，想在UNO上运行很有可能无法通过编译，不过国外Arduboy论坛上[MasterOfGizmo](https://community.arduboy.com/u/MasterOfGizmo)移植相关库文件，使这个有可能！

**详细看: <https://community.arduboy.com/t/arduboy-on-arduino-nano-i-c-display>**

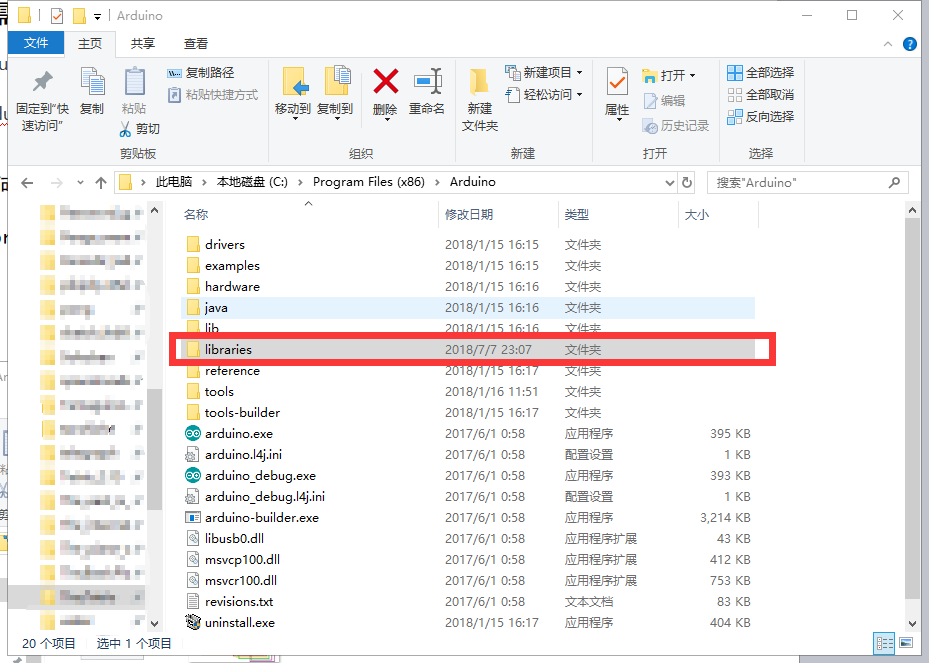
## 电路连线:



扬声器正       ===>9   
扬声器负       ===>11  
红色LED        ===>5  
蓝色LED        ===>6  
绿色LED        ===>7        (由于Atmega328p 少了一个PWM的缘故 绿色led无法调亮度)  
上键           ===>17 A3  
下键            ===>2  
左键            ===>15 A1  
右键            ===>3  
A键            ===>4  
B键          ===>16 A2  
OLED\_SCL      ===>19 A5  
OLED\_SDA    ===>18 A4

所需要的库文件:  
Arduboy2   GITHUB: [https://github.com/harbaum/Arduboy2](https://github.com/harbaum/Arduboy2" \t "https://www.arduino.cn/_blank)  
 ArduboyTones GITHUB: [https://github.com/harbaum/ArduboyTones](https://github.com/harbaum/ArduboyTones" \t "https://www.arduino.cn/_blank)

##### 如何安装库文件：把库文件解压在Arduino IDE安装目录下的libraries文件夹，并重启Arduino IDE



### **这个和真正的ARDUBOY有什么不同：** 真正的ARDUBOY采用Arduino leonardo 使用的是ATmega32u4处理器 而这个采用的是UNO或NANO 使用的是ATmega328 处理器

|  |  |  |
| --- | --- | --- |
|  | **ATmega328** | **ATmega32u4** |
| **运行频率** | 16  Mhz | 16  Mhz |
| **RAM** | 2    KB | 2.5 KB |
| **Flash** | 32   KB | 32 KB |
| **EEPROM** | 1 KB | 1 KB |

### 

### 很显然，通过对比，我们不难发现 ATmega328 的RAM少了0.5 KB,这会导致一些游戏无法运行

**由于RAM不足 所以很多游戏运行不了，而且因为芯片不一样 一些游戏需要库无法正常编译通过，进而无法运行相关Arduboy的程序**