

2020游戏程序设计报告

课程名称： 游戏程序设计

任课老师： 李仕

姓名：赖蔷冰 包效尹

学号：18201103 18072107

专业： 数字媒体技术

## 简介：

**1.游戏基本资料**

·游戏名称： 推箱子

·游戏平台： 电脑

·游戏类型： 逻辑按键推理

**2.游戏概要**

把所有箱子推到对应果实的地方，别把箱子推到死路，要不然游戏就没法进行喽……

**3.游戏界面分析**

3.1游戏开始界面

开始按钮：进入游戏

游戏难度选择：123对应不同难度关卡 可以退回选择关卡

3.2游戏内容界面

·界面操作说明：

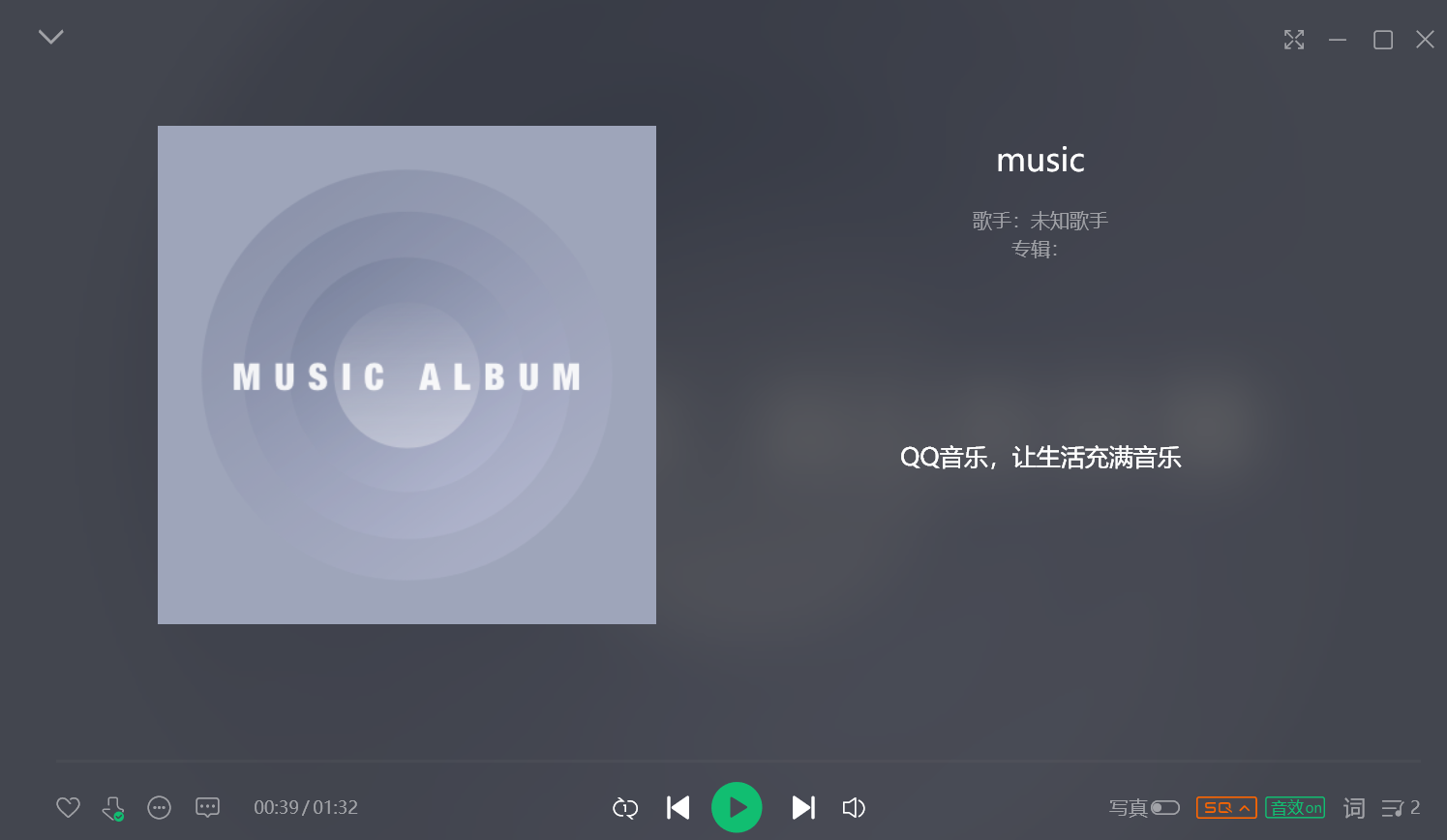
上下左右或者wasd按键：分别控制人物进行上下左右的移动。

设置按钮：点开控制背景音乐大小和重新开始游戏

离开：退出游戏。

·界面说明：

|  |  |  |
| --- | --- | --- |
| 界面 | 截图 | 说明界面内容解说 |
| 初始界面 | 以后补 | 进入游戏的界面  关卡选择和退出游戏 |
| 成功界面 |  |  |
| 失败界面 |  | 一般不会除非有箱子被推到角落 可以选择重新开始 |
| 设置界面 |  | 选择音量或者重新开始 |
| 游戏界面 | 本周主要进行了游戏框架的搭建，并且完成了游戏界面绘制的Draw函数 |  |
|  |  |  |
|  |  |  |

1. **音乐音效**
2. 背景音效<本周找到的音效为1分半的纯音乐，符合游戏气氛，缓解压力，带来愉快的游戏体验>
3. 开始界面的选择按钮音效
4. 推箱子音效
5. 把箱子推到对应地方的音效

**5.游戏世界各元素定义**

（列表对游戏世界包含的各个对象进行说明）

|  |  |  |
| --- | --- | --- |
| 对象 | 作用 | 属性 |
|  | 已经推到正确位置的箱子个数 | 整型 |
|  | 两种箱子 | 精灵对象 |
|  | 砖墙背景 | 精灵对象 |
|  | 墙体图案 | 精灵对象 |
|  | 人物图案《后期有更新皮肤，增加游戏乐趣》 | 精灵对象 |
|  | 果实图案 | 精灵对象 |
|  | 计时器，表示从点好模式后经过的时间 | 计时变量，整型（不一定会有，对这个游戏来说有点多余，除非是计时模式？） |
|  | 关卡的选择 | 按钮 |
|  | 设置 | 按钮 |
|  | 按重来重新开始游戏  按离开结束游戏 | 按钮 |

**6.游戏逻辑实现（举例如下）**

**6.1 初始化游戏世界**

（1）加载图片和音频资源

加载游戏的背景初始界面，推箱子的布置界面，箱子、人物、围墙、果实、空地。

1. 初始化地图:

对游戏中需要的精灵图及纹理的初始化，还有游戏逻辑判断变量的初始化

void initial()

{

window.setFramerateLimit(600); //控制窗口更新频率，每秒设置目标帧数

ViewPos.x = (mode.width - STAGE\_WIDTH \* SCALE) / 2;

ViewPos.y = (mode.height - STAGE\_HEIGHT \* SCALE) / 2;

window.setPosition(ViewPos);

if (!T\_mainBackground.loadFromFile("date/images/background2.png"))

{

*cout* << "没有找到主背景图" << *endl*;

}

S\_mainBackground.setTexture(T\_mainBackground); //设置精灵对象纹理

S\_mainBackground.setScale(SCALE, SCALE); //设置精灵对象缩放

//S\_mainBackground.setOrigin(0, 0);

if (!T\_people.loadFromFile("date/images/people.jpg"))

{

*cout* << "没有找到主人公" << *endl*;

}

S\_people.setTexture(T\_people); //设置精灵对象纹理

S\_people.setScale(SCALE, SCALE); //设置精灵对象缩放

//S\_mainBackground.setOrigin(0, 0);

if (!T\_wall.loadFromFile("date/images/wall.jpg"))

{

*cout* << "没有找到墙体" << *endl*;

}

S\_wall.setTexture(T\_wall); //设置精灵对象纹理

S\_wall.setScale(SCALE, SCALE); //设置精灵对象缩放

//S\_mainBackground.setOrigin(0, 0);

if (!T\_blank.loadFromFile("date/images/blank.jpg"))

{

*cout* << "没有找到地板" << *endl*;

}

S\_blank.setTexture(T\_blank); //设置精灵对象纹理

S\_blank.setScale(SCALE, SCALE); //设置精灵对象缩放

//S\_mainBackground.setOrigin(0, 0);

if (!T\_end.loadFromFile("date/images/end.jpg"))

{

*cout* << "没有找到小球" << *endl*;

}

S\_end.setTexture(T\_end); //设置精灵对象纹理

S\_end.setScale(SCALE, SCALE); //设置精灵对象缩放

//S\_mainBackground.setOrigin(0, 0);

if (!T\_box.loadFromFile("date/images/box.jpg"))

{

*cout* << "没有找到箱子" << *endl*;

}

S\_box.setTexture(T\_box); //设置精灵对象纹理

S\_box.setScale(SCALE, SCALE); //设置精灵对象缩放

//S\_mainBackground.setOrigin(0, 0);

if (!T\_redbox.loadFromFile("date/images/redbox.jpg"))

{

*cout* << "没有找到箱子" << *endl*;

}

S\_redbox.setTexture(T\_redbox); //设置精灵对象纹理

S\_redbox.setScale(SCALE, SCALE); //设置精灵对象缩放

//S\_mainBackground.setOrigin(0, 0);

gameOver = false;

gameQuit = false;

sceneNumber = 2;

}

按照初始函数定义的01345分别导入空地、墙、果实、箱子、人

（3）初始化计时器

计时变量清零。

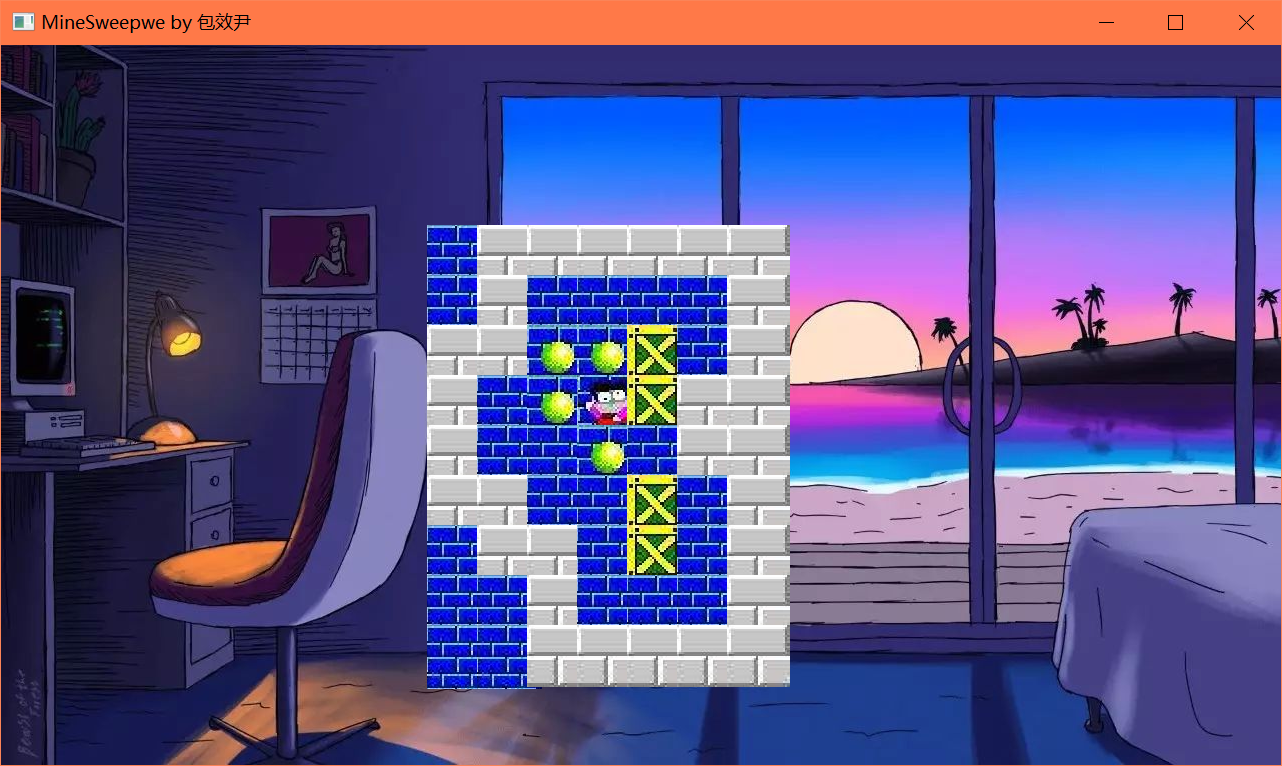
1. 初始化计数器

初始为0

**6.2 游戏循环**

**（1）输入（想并到逻辑一起写……）**

定义三个界面的按键响应以及模式转换 人的方向转换（对应的变量以后补充）



1. **逻辑处理**

根据推箱子的各种要素工具来改变游戏地图的绘制

以人作为游戏最中心的出发点，推箱子则要考虑到两格之后的范围是什么

Case 按键向上：

1.人的前面是空地 新位置刷新人

2.人的前面是果实 新位置刷新人

3.人的前面是墙 无法移动

4.人的前面是箱子

4.1箱子的前面是空地 新位置刷新人 新位置+1刷新箱子

4.2箱子的前面是墙 无法移动

4.3箱子的前面是果实 新位置刷新人 新位置+1刷新箱子 得分计数加一

4.4箱子的前面是箱子 无法移动

5.人的前面是箱子加目的地

5.1再前面是空地 新位置刷新人 新位置+1刷新箱子 得分计数-1

5.2再前面是墙 无法移动

5.3再前面是果实 新位置刷新人 新位置+1刷新箱子

5.4再前面是箱子 无法移动

Case按键向下：

同理

Case按键向左：

Case 按键向右：

对不同的情况进行不同的刷新讨论 用二维数组实现移动（以后具体代码补充）

1. **绘制**

关卡游戏界面设计：

void draw()

{

window.clear();

//绘制游戏开始界面

//绘制游戏主界面

int map[7][9] = {

{0,0,1,1,1,1,0,0,0}, //0 空地

{1,1,1,0,0,1,1,0,0}, //1 墙

{1,0,3,3,0,0,1,1,1}, //3 目的地

{1,0,7,3,7,0,0,0,1}, //4 箱子

{1,5,4,4,3,4,4,0,1}, //5 人

{1,0,0,1,1,0,0,0,1}, //7 箱子+目的地

{1,1,1,1,1,1,1,1,1} }; //8 人+目的地

for (int i = 0; i < 7; i++)

{

for (int j = 0; j < 9; j++)

{

switch (map[i][j])

{

case 0:

/\*putimage(i \* 50, j \* 50, &blank);\*/

S\_blank.setPosition(i \* 50, j \* 50);

window.draw(S\_blank);

break;

case 1:

//putimage(i \* 50, j \* 50, &wall);

S\_wall.setPosition(i \* 50, j \* 50);

window.draw(S\_wall);

break;

case 3:

//putimage(i \* 50, j \* 50, &end);

S\_end.setPosition(i \* 50, j \* 50);

window.draw(S\_end);

break;

case 4:

//putimage(i \* 50, j \* 50, &box);

S\_box.setPosition(i \* 50, j \* 50);

window.draw(S\_box);

break;

case 5:

//putimage(i \* 50, j \* 50, &people);

S\_people.setPosition(i \* 50, j \* 50);

window.draw(S\_people);

break;

case 7:

//putimage(i \* 50, j \* 50, &redbox);

S\_redbox.setPosition(i \* 50, j \* 50);

window.draw(S\_redbox);

break;

case 8:

//putimage(i \* 50, j \* 50, &people);

S\_blank.setPosition(i \* 50, j \* 50);

window.draw(S\_blank);

break;

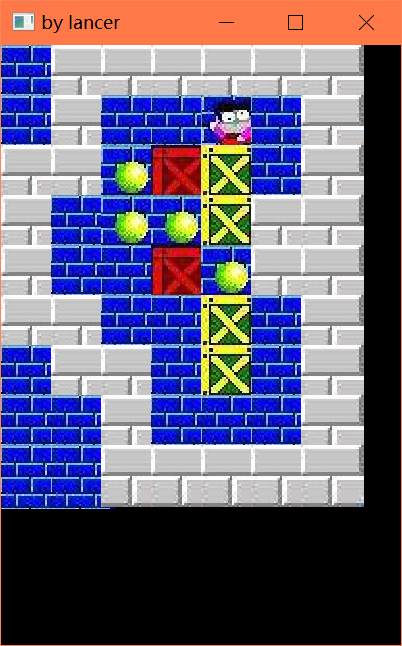
}

}

*printf*("\n");

}

window.display();

}

绘制时钟和计时文本。

绘制剩余已经到达目的地箱子数。

遍历地图状态数组中的每个元素：

对应逻辑代码进行相应的元素刷新

音频输出；

**6.3 游戏难度切换**

不同的关卡（对应不同的箱子个数）

开始界面自己选择

**6.4 游戏结束**

跳出返回开始界面选项：<游戏开始结束的逻辑判断>

int main()

{

initial();

while (window.isOpen() && gameOver == false)

{

switch (sceneNumber)

{

case 1://开始界面

Logic();

Input();

draw();

break;

case 2://游戏界面

Logic();

Input();

draw();

break;

default:

break;

}

}

return 0;

}

## 游戏SWOT分析

|  |  |  |  |
| --- | --- | --- | --- |
| S优势 | 未来想做到达一定积分 更换推箱子的角色（类似新皮肤） | O机会 | 老游戏的经典  大家的童年 |
| W劣势 | 关卡没有重复游玩性 | T威胁 | 需要智商才能玩 |