

整体的逻辑如上图所示，

在游戏中玩家操控炸弹人可以通过放置炸弹摧毁可破坏的方块，游戏目标为使用炸弹击败所有其他玩家和AI

首先开始游戏后进入游戏界面，通过123选择游戏模式并且按enter进入，

游戏开始之后通过玩家一通过WASD和空格控制，玩家二通过ijkl和回车控制

其余玩家由ai操作

整体结构介绍：

在程序的开始部分会调用这三个函数



在显示主菜单中会根据MainMenu的图片像素进行开始界面绘制

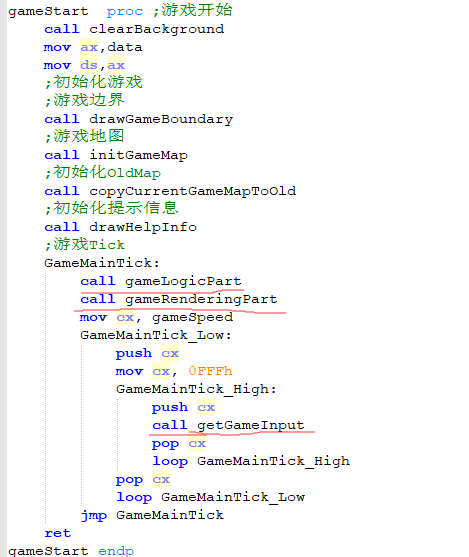


在游戏选择模式中通过循环结合无阻输入来读取数据并且播放音乐



在设置好游戏模式后进入gamestart

在其中有游戏关键部分的Tick 每次Tick会调用逻辑部分，渲染部分和输入

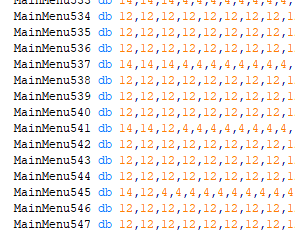


在输入中还是使用无阻尼输入，通过输入设置玩家的一些参数用于之后逻辑计算

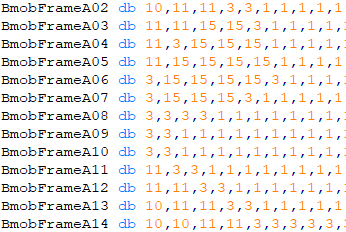
除此之外还有一系列工具函数支持这三个部分的运作，主要包含Debug，rendering，Logic，Music这四大类工具部分

数据介绍：

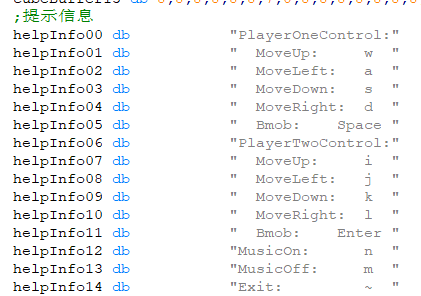
储存的数据有开始界面的图像像素



各种类型方块的像素

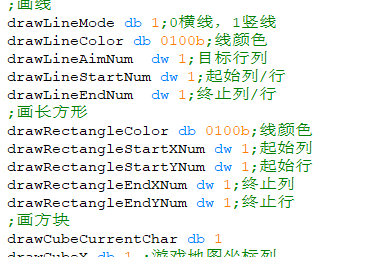


提示信息



游戏地图相关数据 gameMap oldGameMap bmobWaveHeadNum等

工具类储存使用的一些参数



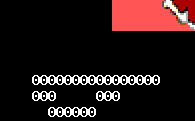
玩家数据和游戏内部设定



工具介绍：

1. DebugTool

可以在任意位置调用显示DebugNum，并且显示他高低位的十进制方便查看



1. Music

Music的频率和每个频率持续时间储存在数据中通过调用不同的PlayMusic函数可以在当前帧播放此频率的声音，这一步是通过调用genFragSound函数 他是通过输入音频进行设置扬声器

3.Rendering部分

渲染部分最基础的是drawPixel和drawChar，用于通过调用中断在指定位置画出像素和字符，之后由此进一步衍生了画线，画矩形，画实体矩形的一些函数drawLine，drawRectangle，drawFillRectangle

对于地图的绘制最常用的一个函数是drawCube，他会根据传入的地方方块类型对该区域16x16个像素进行绘制，这个函数并不是直接输出到屏幕，而是先画到一个16x16的缓存区中，画完之后再进行输出，这是因为玩家的绘制需要在地面的基础上再次绘制，形成叠上去，如果直接重新到屏幕后会出现闪烁的问题

4Logic部分

主要负责一些判断 如果当前位置是不是玩家，当前位置是什么方块，安装按当前方向下一个区域是啥，设置修改地图中的方块，随机数等等

逻辑介绍：

在其中分为游戏结束判断，玩家操作部分，ai操作部分

，炸弹判断，音乐控制

游戏结束判断：通过Playeralive进行游戏结束的判断

玩家控制部分：首先跳过死亡玩家和AI控制，之后进行两种操作的进行，进行移动和放置炸弹，其中用到的原始数据是由input中获得设置在玩家数据中，开进行这些之前进行initPlayerGameParameter 获取储存在每个玩家数据中的当前位置，当前方向等

在放置炸弹的操作中，要先进行是否有放炸弹的操作判断，之后看脚下是否为空地才能放置等判断后 才进行放置操作，先设置脚下Cube为炸弹，再将炸弹初始化储存进PlayerBmob系列参数之中

在移动过程中先看有无请求，再要根据当前位置和方向推测出能否往前移动，能的话就要修改地图上面的玩家字符位置，并且修改当前玩家参数Player系列中的位置，脚下方块，状态等等

在AI操作部分：使用了状态机进行控制，AI有空闲，躲避炸弹，等待炸弹三个状态，在AI空闲的时候会随机选择进行随机移动或者进行炸弹放置

对于随机移动会先随机上下左右任意一个方向，然后判断是否可以前进，如果可以就前进，如果不行就以一定概率就行再次选择方向（如果没再次选择就停在原地）

对于随机炸弹放置首先，放置炸弹后的安全位置为能否躲到左上右上左下右下，所以对于AI随机两步的移动 比如右移加上移，这样可以达到四个角，然后对于随机到的移动路径进行评估，看能否到达安全位置，如果可以就放置炸弹，移动第一步，并且将第二步的移动方向储存，调整状态的躲避炸弹状态，如果不能到达则以一定概率重新随机位置再次判断，如果最后没有随机了就原地不动

如果为躲避炸弹状态就按照记录的第二步移动，并且切换到等待炸弹状态

在等待炸弹状态下，人物会在原地等待，当炸弹爆炸后又会回到空闲状态

对于炸弹部分：首先会进行时间判断，如果没到爆炸时间则减少时间，到了爆炸时间就进行爆炸，对于自身方块和四个方向的路径进行判断，击中玩家就让玩家死亡，击中箱子或者墙或者其他炸弹就停止该方向向前，其余情况下替换空地为按照方向设置的炸弹波纹（4,5,6,7,8）等

为了解决炸弹波纹相互覆盖的问题，有一个bmobWaveHeadBmobNum记录地图上每个位置最顶上的炸弹波纹是哪一个炸弹释放的

在爆炸波纹时间结束之后，对于自身的4个方向进行判断，遇到是自己的波纹，也就是地图上记录num是自己的就将其清除掉，替换为空地，同样遇到障碍也会停止前进，这样如果波纹被其他炸弹覆盖，就会判断不是自己的波纹就不会把别的炸弹波纹清除掉

对于音乐部分其就是通过input的输入控制音乐的开关

渲染介绍：

在渲染部分中主要会调用drawGameMapUpdate，在其中会调用一个函数对比当前地图的老地图，将不同的方块进行重新绘制（如果全地图直接重新绘制帧率很低，画面很卡）同时会有一个gameRenderingFrame参数，表示当前动画的帧数，用于动画帧数切换，这个参数每帧更新，在方块的绘制中就会根据这个参数选择不同的动画像素进行绘制

在update中不仅要绘制不同的方块，还要将一些含有动画的方块必须刷新，同时将人物的头要渲染到人物头上的那个方块中，其中又有一系列复杂的判断计算

对于一个人叠在地板上就用到了CubeBuffer 现将地板绘制再绘制人覆盖，这样就有了人站在地板上的图，再将这样的像素进行输出

渲染部分的图片像素来源方式：待续