辅助类：

枚举类Kind，包括：EMPTY, WALL, FIGHTER, MONSTER, EXPLOSION

枚举类Direction，包括UP,RIGHT,DOWN,LEFT

常量类Constant，包含消息常量和数据常量。

数据库DBHelper，向外提供DBHelper单例，以及预先写好的插入删除查询方法。数据库含一张表，存储id，areaId，level。areaId没啥用，都是默认a，因为只会有一条记录。

主活动MainActivity

成员变量包括GameView变量gameview和DBHelper的单例。

onCreate方法启动背景音乐服务，隐去标题栏和修饰部分，调用welcome（）方法进入初始界面。

welcome（）方法，设置布局，主要是四个ImageButton，分别设置监听方法。

开始游戏按钮的监听方法startGame（）：

设置gameview布局，调用initGame（）方法，查询数据库，设置gameview的level值，如果数据库中没有就设置为1。

initGame（）方法，从布局中获得gameview实例，以及各个按钮的实例，并为这些按钮设置监听器。游戏结束的对话框alertDialog也在这时候创建等待后续show。

游戏规则按钮的监听方法ruleGame（）：

设置rulegame布局，获取返回键实例并设置监听器。

关于游戏按钮监听方法aboutGame（）：

设置aboutGame布局，获取控件实例，然后通过数据库访问获取存储的游戏级别-1作为最高分设置到textview中，并给返回键设置监听器。

设置关卡的按钮监听方法settings（）：

调用setLevel（）方法， 创建一个AlertDialog需要用户数据关卡等级数字，如果确定的话，就删除数据库中现有的记录，并插入输入的记录。然后显示对话框。

退出游戏按钮监听方法quitGame（）：

finish（）；

返回键监听器是所有地方共用的，他的onClick方法会判断gameview实例是否是null，如果是，游戏还没开始，统一调用welcome方法回到初始界面，否则说明在游戏中，那么删除数据库的记录，将当前level作为记录存入，并调用welcome方法回到初始界面，调用gameview.clearGameviewData（）方法清除游戏界面中的数据，最后让gameview=null。

上键监听器方法：

通过gameview.fighter的get方法获得fighter的坐标值x，y（此处为索引坐标），y--，之后通过get方法获得fighter的方向，如果不等于枚举方向的UP，就set方向为UP，然后return。否则y--，判断是否越界（大于等于0）以及gameview.info对应坐标处是否是枚举EMPTY，如果符合这两个移动条件，就重新设置fighter的坐标，并将新位置的info设置为FIGHTER，原位置设为EMPTY。

下左右类似，由于地图设置了终点一定在最下方最右处，只能通过右移得到，所以右键监听还要额外判断位置是否是gameview.end.x和.y，如果是，通过handler发送获胜消息。

攻击键监听方法：

get方法获得fighter的坐标，并调用fighter的fire（）方法。

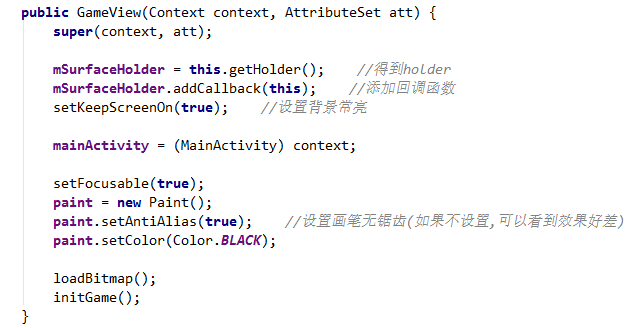
handler，判断msg.what，如果是welcome消息，就调用welcome（）方法，如果是update消息，就判断gameview不为null的时候，setText到textview控件上，把gameview的存储信息的变量值放上去。如果是win消息，提供一个toast提示，并将gameview的level加一，fighter的生命值通过set加一，调用gameview.resumeGame（）方法。如果是lost消息，调用gameview.lostGameDo（）方法。

GameView类

自定义view，继承了SurfaceView，并实现Runnable接口和SurfaceView.Callback接口。

成员变量有各种图和各种需要的数据变量。一个int二维数据存储地图01值，对应的info二维数据存储Kind枚举类表示这个位置所属类型。一个Fighter变量。两个Point变量存储起始点为终点。可分别调用变量的.x和.y获得对应的坐标值。一个MainActivity变量。

构造方法



loadBitmap（）方法，

给bitmap数组赋值，得到勇士图，怪物图，墙图和空位图。

initGame（）方法，

设置游戏级别level=1，通过MyMap的getMap（level）方法给mymap二维数组赋值。MyMap类存储了一个静态三维数组，就是10个二维数组表示地图01值，1墙0空，并通过getMap方法将地图数据传出，level作为三维数组的第一维索引，就可以传出一个二维数组。

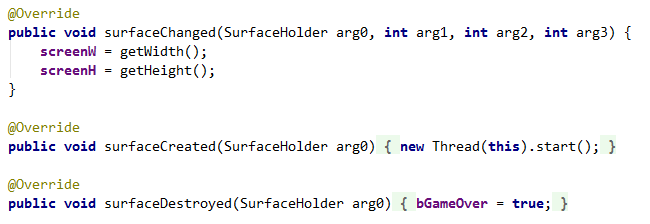
在起始位置上new一个fighter（构造方法传入this，和起始位置x,y的坐标值（索引坐标））并设置info为FIGHTER，然后new出需要的list，有怪兽、爆炸、好坏子弹。然后根据mymap的01值对info数组赋值WALL或者EMPTY。根据预定义常量的怪兽数目，调用对应次数的createMonster方法。

createMonster（）方法

while（true）内，随机生成两个xy坐标，要求在地图范围大小内。如果对应位置的info不是墙，就在那个地方new一个Monster，然后启动MonsterRun线程，设置对应位置info为MONSTER，然后break，否则就继续循环。

createFighter（）方法，同initGame中的操作，用于角色死亡后调用（其实也可以写好在initGame中调用）。

surfaceview的一套操作：

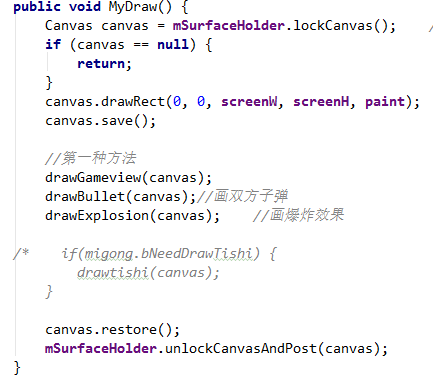


这个view要实现run方法，如下：

通过指示变量bGameOver判断游戏是否结束，来控制一个while循环。循环内一个同步块调用kill（）方法，同步块外调用myDraw（）方法，通过handler发送更新消息，之后sleep一段时间。

kill（）方法：遍历好子弹和坏子弹的列表，分别调用它们的killMonster、killBullet、killFighter方法进行攻击判断。

myDraw（）方法：



drawGameView（）方法：

遍历各个索引坐标点，判断info信息是墙还是空位还是勇士和怪物，并通过canvas.drawBitmap（bitmap，x像素坐标，y像素坐标，paint）画图，再画出起点终点。

drawExplosion（）方法和drawBullet（）方法

遍历爆炸列表和子弹列表，调用列表元素的draw方法。

regetInfo（）方法：

根据当前map数组，fighter和monster的坐标，遍历一遍重新设置info值。

resumeGame（）方法：用于重新开始一局游戏

清空所有列表，重新得到当前等级的map，createOneFighter（），调用3+level次createOnesMonster（）方法，最后调用regetInfo（）方法。

clearGameViewData（）方法

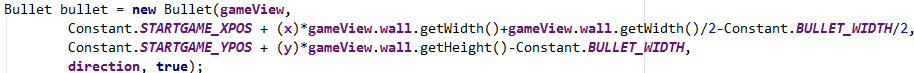
将所有怪物、子弹的生命set为false，并清空列表，令fighter=null，最后bGameOver=true指示游戏结束。

lostGameDo（）方法

设置游戏结束指示bGameOver=true，show一个AlertDialog，如果选择重玩，就clearGameViewData，resumeGame，设置结束指示为false，然后new Thread（this）启动新的游戏线程。否则发送消息回到welcome界面。

Fighter类

创建需要提供gameView，以及索引坐标x，y。包含direction成员变量，默认是右。内含fire（）方法，判断方向前面是否越界，如果不是，就new一个Bullet，启动BulletRun线程，再将这个子弹加入gameView的好子弹列表中。



Bullet类，成员变量有gameview，xy像素坐标，direction，good判断好坏子弹，live判断是否存活。然后loadBitmap载入子弹图，提供draw方法通过canvas.drawBitmap画出子弹。

killMonster（monster）方法：

判断两者的像素坐标差是否小于宽（高）和的一半，如果是，就表示命中，从列表中移除这个好子弹和怪物，同时设置monster和子弹的live都为false用于关闭线程，gameview的成员变量杀敌计数+1，在怪物的位置new一个Explosion并启动ExplosionRun线程，并设置改位置的info为爆炸。

类似的killFighter方法，如果命中还要gameview设置勇士生命数-1，如果小于1了，就发送消息游戏结束，否则就调用gameview的createOneFighter和regetInfo方法。

类似的killBullet方法判断两个子弹的good是否不同，不同才开打，从列表中移除这两个子弹并设置live为false。

BulletRun类

run方法在一个循环内，循环终止条件为子弹的生命live是否是false，每次执行循环，子弹根据方向调整x或y的坐标，调整值在常量中设置，根据位置判断是否越界或撞墙，如果是就从列表删除并设置生命为false，否则就重新设置x，y坐标。然后线程sleep一段时间不让子弹移动过快。

Monster类似Fighter

MonsterRun类

run方法，一个循环判断金鱼的live，如果存活，根据十分之一的概率调用fire（）方法，再根据一定概率修改方向，同时调用move（direction）方法。然后sleep一段时间。

move（）方法，根据传入方向，判断与当前方向是否一致，如果不一致就修改当前方向，return。否则就根据方向移动一个索引坐标，然后修改因此而改变的info。