吃豆人 - 类设计图

注：single表示单例模式创建的类

by KevinsBobo https://kevins.pro

CGameCrtl - single

- m_gameUI: CGameUI
- m_map: CGameMap
- m_moveObj: IGameElement**
- m_ghoObj: IGameElement**
- m_player: CPlayer*

// 游戏开始
+ gameStart(): void
// 游戏逻辑开始
+ gameLoop(): void
// 游戏结束
+ gameStop(): int
// 初始化游戏数据
+ initGameData(): void
// 改变鬼的方向
+ changeGhostAct(): void
// 寻路算法
+ findAction(CPostion, CPostion): int
// 检查状态
+ checkStatus(): void
// 当玩家吃掉超级豆
+ whenPlayerEatSuperPean(): void
// 当玩家碰撞恐惧时的鬼
+ whenPlayerEatGhost(): void
// 当玩家碰撞正常鬼
+ whenGhostEatPlayer(): void
// 当玩家吃完所有豆子
+ whenAllPeanBeEat(): void
// 检查玩家分数、升级鬼
+ checkPlayerScore(): void

CGame - （提供全局变量、宏定义和枚举）

```
// 全局变量
const clock_t g_nInitGhoSpead = 400;
const clock_t g_nInitPlayerSpead = 200;
const int g_nMapRow = 31;
const int g_nMapCol = 28;
const int g_nPlayer = 1;
const int g_nGhost = 4;
const int g_nPean = 300;
CGameMap* g_gameMap = NULL;
const int g_nFirstGhoRow = 14;
const int g_nFirstGhoCol = 12;
const int g_nPlayerRow = 23;
const int g_nPlayerCol = 13;
int g_nPlayerLife = 3;
// 玩家已得分数
int g_nAddScore = 0;
// 玩家吃鬼分数
int g_nPlayerEatGhostScore = 50;
// 玩家吃鬼加分倍率
const float g_fMagScore = 1.5;
// 玩家被吃标志
int g_isBeEat = 0;
// 吃掉一只鬼
int g_nGhostBeEat = 0;
// 玩家吃掉豆子的数量（包括超级豆），最多300
int g_nEatPeanNum = 1;
// 超级豆被吃标志，吃一个加1
int g_nSuperPeanBeEat = 0;
// 吃超级豆分数
const int g_nSuperPeanBeEatScore = 1;
// 吃豆子分数
const int g_nPeanBeEatScore = 1;
// 鬼恐惧时间
const clock_t g_nGhostFearTime = 10000;
// 鬼开始恐惧时间
clock_t g_nFearStartTime = 0;
// 鬼等级
int g_nGhostLevel = 1;

// 宏定义
#define MAPROW 31
#define MAPCOL 28
#define NPLAYER 1
#define NGHOST 4
#define NCLASS 14 // 游戏元素数量
#define PLAYEREATGHOSTSCORE 50

// 枚举
// 方向
enum gameAction
{
actUp = 0,
actDown,
actLeft,
actRight
};
// 位置
enum gamePostion
{
posRow = 0,
posCol
};
// 碰撞状态
enum gameCrash
{
craBeHit = -1,
craNo = 0,
craHitAn,
craPass
};
// 所有元素
enum gameElement
{
itemPean = 0,
itemWall,
itemSurperPean,
itemRoad,
itemGhoHome,
itemGhoNormal,
itemGhoRed,
itemGhoYell,
itemGhoBlue,
itemGhoPink,
itemGhoFear,
itemGhoTremble,
itemGhoDie,
itemPlayer
};
```

CGameMap - single

- m_map[MAP_ROW * MAP_COL + 4 + 1]: IGameElement*

// 初始化地图
+ initMap(): void
// 重载[]运算符
+ operator[](int): IGameElement**
// 获取需要移动的对象
+ getMoveObj(): IGameElement**
// 调用传入UI对象函数指针输出地图中的每个对象
+ show(CGameUI*, echoMapElement): void

CGameUI - single

- m_szElement[4 + 4 + 1]

// 对外接口
// 打印地图元素
+ echoMap(int, int, int): void
// 输出游戏信息
+ echoGameInfo(int, int, int): void
// 在消息区域显示消息
+ echoGameMessage(char*): void

// 内部接口
+ echoMapByPrintf(int, int, int): void
+ echoMapByApi(int , int , int): void

CPostion

- nPosRow: int
- nPosCol: int

+ CPostion(int, int)
+ CPostion(CPostion&)
+ operator=(CPostion&): CPostion&
+ operator+(CPostion&): CPostion
+ operator-(CPostion&): CPostion
+ operator*(int): CPostion
+ operator==(CPostion&): int
+ operator[](int): int&
// 修正坐标（防止越界）
+ amend(): void
// 设置坐标
+ set(int, int): void
// 获取这个方向下个坐标
+ getActionPostion(int): CPostion
// 根据方向修改坐标
+ changePostion(int): void

IGameElement

// 获取对象坐标
+ operator[](int): int&
// 与其他对象比较碰撞等级
+ operator==(IGameElement&): const int
// 获取坐标对象
+ getPos(): CPostion&
// 获取类型
+ getType(): const int
// 设置类型
+ changeTypet(): void
// 移动
+ move(): int
// 改变方向
+ changeAction(): int
// 碰撞检测
+ isCrash(): const int
// 撞到比自己碰撞等级高的
+ beHit(int): void
// 撞到比自己碰撞等级低的
+ hitAnthor(int): void
// 创建自身，此方法为后面扩展做准备
+ creatSelf(): IGameElement*

CGameElement

m_postion: CPostion
m_nType: int
m_nAction: int

CStaticObj

CRoad

CPean

CSurpean

CGhoHome

CWall

CMoveObj

m_nSpeed: clock_t
m_preMoveClock: clock_t

// 更新时间
+ updateClock(): int
// 移动
+ move(): void
// 获取速度
+ getSpeed(): clock_t
// 设置速度
+ setSpeed(int): void
// 回位
+ backPos(): void
// 获取方向
+ getAction(): int

CPlayer

- m_nScore: int
- m_nLife: int

+ getScore(): int
+ setScore(): void
+ addScore(): void
+ getLife(): int
+ setLife(): void

CGhost

- m_nWhatGhost

+ move(): void
+ toFear(): void
+ getWhatGhost(): int