**全局模块**

procedure Singleton

begin

if 实例 == null then

尝试在现有实例中寻找

if 找不到实例 then

创建新的实例

为实例添加自己的脚本组件

返回新的实例

end if

end if

返回实例

end

end Singleton

procedure ObjectPool

begin　Spawn

interface 子池名, 实体类型

if 子池表中含有该子池名 then

注册新子池，调用RegisterPool

end if

在子池中取一个实例

激活实例

返回实例

end Spawn

begin　Unspawn

interface 实例

loop until　找到含有该实例的子池或遍历完子池表

if 当前子池不含有该实例 then

找到子池表中下一个子池

end if

end loop

反激活实例

end Unspawn

begin　RegisterPool

interface 子池名，实体类型，池中实例数量

if 子池表中已存在同名子池 then

返回

end if

根据实体类型设置预定路径

根据路径加载Prefab

根据Prefab创建子池，调用子池的构造函数，参数为Prefab

为子池中加入足够数量的实例，调用子池的AddObject函数，参数为池中实例数量

在子池表中加入新子池的记录

end RegisterPool

end ObjectPool

procedure SubPool

begin AddObject

interface 实例数量

loop while 循环次数小于等于实例数量

创建新实例

激活实例

在实例表中加入实例

end loop

end AddObject

begin Spawn

interface 实例数量

loop Until 遍历完实例表

if 实例已激活 then

找下一个实例

else

取出实例

退出循环

end if

end loop

if 循环结束未找到可用实例 then

创建新实例

加入实例表中

取出新创建的实例

end if

激活取出的实例

委托Unity发送广播消息，通知本实例已激活

返回实例

end Spawn

begin Unspawn

interface 实例

if 子池中没有该实例 then

返回

end if

反激活实例

委托Unity发送广播消息，通知本实例已反激活

end Unspawn

end SubPool

procedure LevelLoader

begin LoadLevel

interface 文件名

创建新的Level实例

根据文件名读取对应的level文件

将文件中的字段Name存入Level中

将文件中的字段Road存入Level中

将文件中的字段Money存入Level中

将文件中的字段MonsterGap存入Level中

//读取字典

读取文件中的Dictionary节点下所有Item节点

创建一个新的字典容器dictionary

loop until 遍历完找到的所有Item节点

if 该节点没有属性 then

跳过

end if

在字典容器dictionary中加入新节点的属性name和entity

end loop

//读取障碍物

读取文件中的Holder节点，得到生字符串surroundingStr

除去surroundingStr中的回车符和换行符

除去surroundingStr中的空格

将surroundingStr按','分割成多个子字符串，存入数组surrounding中

loop until 遍历完surrounding数组

if 该字典容器dictionary中存在该字符串 then

Case　字符串 of

预置的"Plate"字符串值：

在可建造防御塔列表中加入当前坐标

预置的""Start""字符串值：

将游戏起点设为当前坐标

预置的""End""字符串值：

将游戏结束点设为当前坐标

default：

从字典容器中得到障碍物

创建新的Pair，以当前障碍物为键，以当前坐标为值

在关卡文件的障碍物列表中加入新创建的Pair

end case

跳过

end if

在字典容器dictionary中加入新字段的属性name和entity

end loop

//读取回合信息

读取文件中的Rounds节点下所有Round节点

loop until 遍历完找到的所有Round节点

if 该节点没有属性 then

跳过

end if

创建新的Pair，以当前节点属性Monster为键，以节点属性Count为值

在关卡文件的回合列表中加入新创建的Pair

end loop

end LoadLevel

end LevelLoader

**实体模块**

1. role类（怪物和基地的父类）

处理死亡

数据还原成初始状态

扔回对象池

判断死亡

IF HP==0

处理死亡

END IF

得到当前血量

Return HP

1. monster类（怪物类）

到达终点事件（参数 MONSTER）

基地扣除MONSTER攻击力血量事件

MONSTER处理死亡

收到攻击（参数 HIT伤害）

HP-=HIT

IF HP>0 THEN HP=0

判断死亡

移动

IF 还未出发 THEN

当前位置=起点位置

求出下一个位置

移动到一下个位置

IF 当前位置==终点 THEN

触发到达终点事件

1. 基地类

基地扣血(DAMEGE)

HP-=DAMEGE；

IF HP<0 THEN HP=0

IF HP==0 THEN GAMEOVER事件

1. 塔类（比较大的父类）

LOAD(进入地图）参数：TOWERID，TILE（位置）

对象池取出

GRT相应TOWER信息

M\_TILE=TILE

处理删除

数据还原成初始状态

扔回对象池

UPDATE

转向LOOK\_AT（MONSTER POSITION)

IF 攻击目标==NULL THEN

取出所有怪物类

FOR 所有怪物类

IF 怪物存活&&怪物距离和塔距离小于当前攻击范围

THEN 攻击目标=怪物BREAK

ELSE 攻击目标已存在

IF 攻击目标怪物存活&&怪物距离和塔距离小于当前攻击范围

IF 上次攻击时间到当前时间大于攻击频率

THEN

ATTACK;

ELSE

攻击目标=null;

转向LOOK\_AT(MONSTER)

IF 目标为空 THEN 角度=0

计算当前塔和怪物直接的角度

根据角度旋转

1. 具体塔类

ATTACK

播放动画

播放声音

1. 子弹类

LOAD(进入地图）参数：BALLID，MONSTER（位置）

对象池取出

GRT相应BALL信息

TARGET=MONSTER

DIRECTION=GET\_DIRECTION

处理爆炸

数据还原成初始状态

扔回对象池

UPDATE

IF 已经爆炸 THEN RETURN

IF 攻击目标存活 THEN

LOOK\_AT

往目标方向移动

//打中目标

IF 当前坐标与目标相同（攻击到怪兽）THEN

爆炸

怪物受伤（HIT)

ELSE 直接爆炸

**界面模块**

begin

interface 触屏事件

if 开始游戏

interface 可选关卡

while 点击关卡 do

interface 游戏界面

interface 触屏事件

if 菜单

interface 菜单对话框

if 继续游戏

继续游戏

else if 返回菜单

end if

end if

else if 暂停

interface 游戏暂停

else if 空地

interface 创建防御塔

else if 防御塔

interrface 升级/删除防御塔

end if

end while

else if 说明

说明文档界面

else if 退出游戏

end