大作业文档

虚基类及接口：

baseWarrior:

武士基类，所有武士应继承自该基类

变量：

std::string name; 名字

int health ：生命值

int power ：基础攻击力

bool ifCounterattack = true;：是否反击

int campID 阵营ID

int destID 当前目标敌人阵营ID

int maxArmsNum = 1；：最大持有武器数

std::vector<baseArms\*> arms; 武器列表

初始化接口

/// \*必须重载\* 返回默认血量的引用（记得为它设置静态变量）

virtual int& DefaultHealth();

返回该类型默认的生命值，请为它声明一个静态变量并且为它赋初始值

为了初始化武士所设置的接口

/// \*必须重载\* 返回默认攻击力的引用（记得为它设置静态变量）

virtual int& DefaultAttack();

返回该类型默认的基础攻击力，请为它声明一个静态变量并且为它赋初始值

为了初始化武士所设置的接口

/// \*必须重载\* 工厂函数，返回一个指向新new的warrior对象的指针

virtual baseWarrior\* factionWar();

返回一个新创建的对象

请为它初始化一个名字

示例：

nomelWarrior类中：

baseWarrior\* nomelWarrior::factionWar()

{

baseWarrior\* war = new nomelWarrior();

war->name = "nomel";

return war;

}

接口：

/// 每个游戏刻调用

virtual void Updata() {}

游戏时间每“一秒”调用一次

/// 每个Updata调用前调用

virtual void UpdataStart() {}

/// 每个Updata调用后调用

virtual void UpdataEnd() {}

/// 每个小时结束时调用

virtual void OnHourEnd() {}

/// 每个小时开始时调用

virtual void OnHourStart() {}

/// 武士诞生时调用

virtual void Start() {}

在武士初始化结束时调用，可以在此初始化一些设置

/// 武士成功移动结束后调用

virtual void OnMoved() {}

武士成功从一个城市移动到另一个城市时调用

/// 武士主动发动进攻前调用

/// <param name="目标武士"></param>

virtual void OnAttack(baseWarrior\*) {}

/// 武士击败敌人时调用

/// <param name="击败的武士"></param>

virtual void OnKillEnemy(baseWarrior\*) {}

/// 武士反击前调用

/// <param name="反击目标武士"></param>

virtual void OnCounterAttack(baseWarrior\*) {}

/// 武士死亡时调用

/// <param name="击杀者"></param>

virtual void OnDead(baseWarrior\*) {}

/// 武士被投掷物射击时调用

/// <param name=""></param>

virtual void OnBeShoot(baseMissile\*) {}

/// 武士获得战斗胜利时调用

virtual void OnWinAttack() {}

所在城市的敌人都被杀死时调用，在OnAfterAttack之前，可在此标记武士是否获得胜利

/// 在战斗结束后调用（winattack之后）

virtual void OnAfterAttack() {}

战斗结束时调用，可在此处理武士战斗结束时的事件