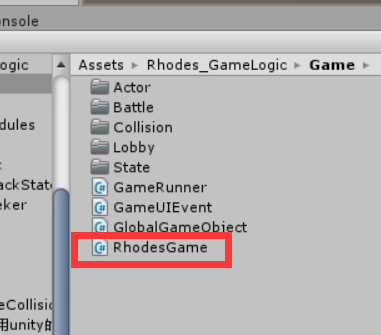
此文件将说明一些开发者之间的约定。包括一些方便开发者使用的工具类和工具对象的使用方式。

# 项目代码简介

这里简单地说一下阅读和理解项目代码的大致顺序。

主要的游戏逻辑存在于Assets/Rhodes\_GameLogic/Game这个路径下面。以下以此路径为根目录进行介绍。

游戏的顶层类是RhodesGame类（文件名同名）



它是一个单例。在游戏启动后，理论上所有的游戏对象都可以从这个单例中取到。

比如如果你要求地图上(3,4)这个节点里的全部敌人的话，代码就是

RhodesGame.Instance.battle.mapNodeCenter[3,4,ActorType.ANY];

与laya版本不同的是，RhodesGame的update函数永远不会调用Actor任何组件的update函数。在这个项目中，对象之间的树状结构还存在，但是update函数的树状调用结构就莫得了。

要注意的是，RhodesGame类不是一个MonoBehavior，引擎并不执行它的update函数。它的update函数其实是在GameRunner类（文件名同名，上图）里面被调用的。GameRunner类的Start函数其实就跟以前laya版本的Main.ts里的构造函数一个意思，是项目代码的起点。模块的初始化啥的活都是在里面干。

GameUIEvent暂时不管，这个是处理用户输入的类，现在还用不到。

GlobalGameObject只是一个工具类，见[下文](#_GlobalGameObject)。

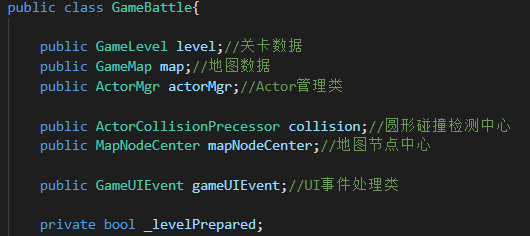
接下来看State文件夹，这个是管理游戏状态的类，场景的切换也是通过这个类完成。现在还不需要看，里面也没有内容。现在游戏基本上是直接进战斗状态的，可以看一眼GameStateBattle里面都update了些啥。

接下来可以看一眼GameBattle类。RhodeGame是“游戏”的类，GameBattle则是“关卡”的类。关卡的任何即时数据都存储在GameBattle中。访问这个类就是

RhodesGame.Instance.battle

就取到了。

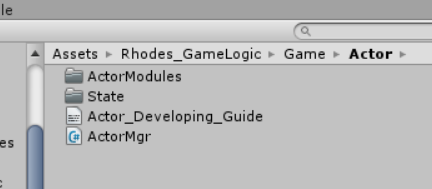
这个类里面包含以下的东西



其中ActorCollisionProcessor、GameLevel、gameUIEvent和GameMap类暂时没有在用，只需要知道他们存在就行。

ActorMgr是管理所有Actor对象（干员、敌人、衍生物）的类，mapNodeCenter则是记录地图节点信息的类。mapNodeCenter只需要看一眼它的取值函数就行，用法就是把你要查的坐标和Actor类型输进去，它会把那个坐标上的所有对应Actor的Reference全部返回给你，上面已经有示例代码了。

然后现在讲目前的开发重点，Actor和Actor模块

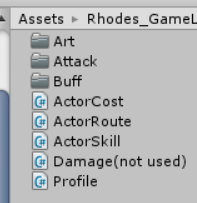


ActorMgr目前其实没有在用，后续实装根据既有的数据部署敌人时可能会用到。现在我们主要的工作是把Actor这个预制件做好，事后再完善它的管理类。顺便一提Actor的预制件在Resources里面，不在这里。

Actor\_Developing\_Guide可以看一下，不过里面的内容其实就一句话：我们在用黑板模式。

State是决定Actor行为模式的状态机，不过现在还没有内容，不用看。

最重要的部分是ActorModules里面的代码。

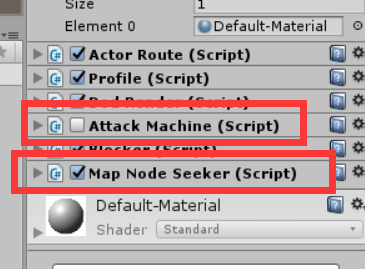


Actor的Profile类是Actor的黑板类。一切Actor的即时数据都存储在这个类里面，包括位置坐标、攻击力防御力、部署位置、攻击类型等等数据。

我们写模块的一个基础的要求就是，ActorModule不得访问任何其他ActorModule。需要获取数据时必须通过Profile间接访问。

比如说我一个ActorRoute（路径管理类，敌人用的）和一个DodRender（可视化用的）都想看Actor的坐标，那这个坐标都存在Profile里，他们互相不需要知道对方存在。

这里面比较大的一个思路变化是Seeker这个类。这个类是用来索敌的，确定哪些敌人处于攻击范围内。它在laya版原本是AtkMachine（攻击状态机）的一个成员。原本我想的是，索敌器嘛，那应该是攻击类的一部分，就跟瞄准镜装在枪上是一样的，就说瞄准镜是枪的一部分。但是现在这个思路不一样了，现在Seeker和AtkMachine是两个互不相关的Modules

（注意，虽然这图里Attack Machine在上面，但执行顺序是MapNodeSeeker先走。详情见Edit>>Project Settings>>Script Execution Order）

那AtkMachine要看现在要打谁，去哪看呢？就去Profile里面看。Profile的这个内容就是为解耦Seeker和AttackMachine准备的。Seeker每帧就倒腾这个变量，AttackMachine需要看索敌结果的时候也是看这个变量，但他们互相不能知道对方存在。



Profile中有一项比较特殊的属性是dataName这一行。这个字符串的内容是Assets/Resources/TestJson下的json文件路径。一个json文件是一种敌人/干员的全部信息。目前此功能仅用于调试。若需要对Actor进行调试，可以通过书写json文件在这里读取的方式修改它的某种属性。

一些工具类

# GlobalGameObject

路径：Assets/Rhodes\_GameLogic/Game/GlobalGameObject

此类用于访问全局唯一的游戏对象

属性:

public static GameObject ObjectInfo

用于显示角色信息的文本框

public static GameObject ObjInfoOriginal

用于显示角色信息的文本框的父级节点，相关的object装在这个节点里面。这只是一个用来分层的对象，相当于一个文件夹

# ObjectInfo : GameObject

用于展示运行时的角色信息的文本框，debug用