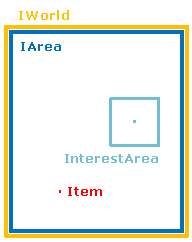
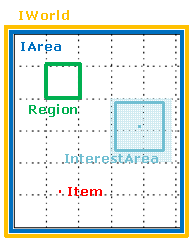
分享一个cocos2d-x集成Photon的MMO(大型多人在线)游戏引擎

首先给大家介绍一下MMO的一些概念

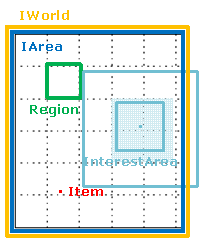
虚拟世界（IWorld）是一个虚拟的坐标系的空间。在这个虚拟空间的实体被称为Item。客户端定义了他们感兴趣的area（位置和范围），并且只能“看见”InterestArea边界内的items。



计算哪些items进入了interest areas非常消耗CPU的资源，特别是items和interest areas频繁移动的话。加快计算的一个简单的方法：虚拟空间划分为固定区域，然后计算出哪些区域与interest area重叠。客户端应收到这些重叠区域的items的所有事件。最简单的算法是使用贴图算法(被称为Square Tile Algorithm)。



当有item进入客户端的interest area里，客户端将接收所有由item发出（或通过）的事件。由于一个移动在interest area边缘的item会频繁得产生订阅和未订阅得变化，以此引入了一个更大的interest area额外的半径： 订阅过的items跨出了这个半径将会退订，客户端将不再接受到这个item的事件。



Interest areas有两个interest阈值。Items进入内半径变得可见; items离开外半径变得不可见。这种优化可以减少频繁的可见变化

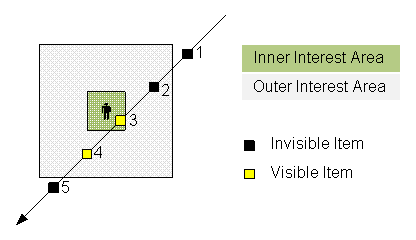
1. 不可见item,超出范围

2. 不可见item在interest area之外

3. 不可见item进入interest area，变为可见

4. 可见item离开interest area的内半径

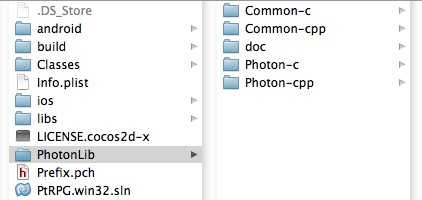
5. 可见item离开interest area的外半径，变为不可见。



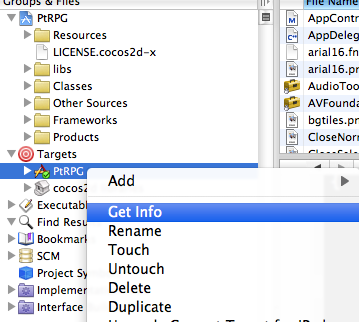
然后给大家介绍下Photon网游的引擎

Photon在Cocos2d-x中的配置

(1) 把Photon-iPhone\_SDK中的Common-c, Common-cpp，Photon-c，Photon-cpp 文件夹拷到你工程的目录中



**注意:以下操作都是在Targets中的Get Info中进行**



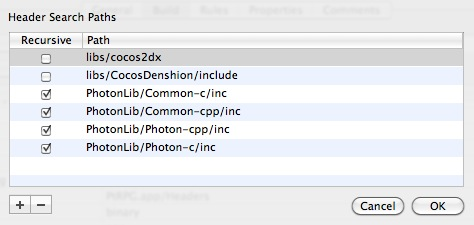
(2) "Header Search Paths" 中添加

PhotonLib/Common-c/inc

PhotonLib/Common-cpp/inc

PhotonLib/Photon-cpp/inc

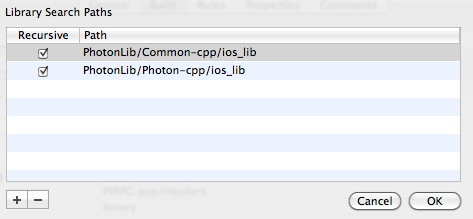
PhotonLib/Photon-c/inc



(3) "Library Search Paths" 中添加

PhotonLib/Common-cpp/ios\_lib

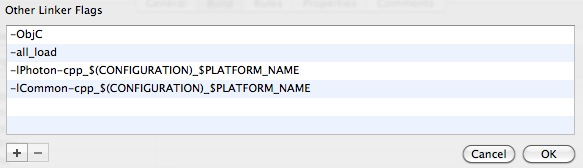
PhotonLib/Photon-cpp/ios\_lib



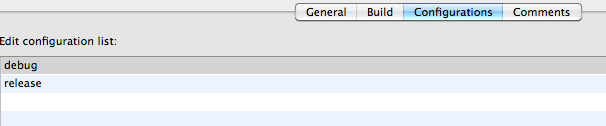
(4) "Other Linker Flags"中添加

-lPhoton-cpp\_$(CONFIGURATION)\_$PLATFORM\_NAME

-lCommon-cpp\_$(CONFIGURATION)\_$PLATFORM\_NAME



(5) 把Configurations中的Debug和Release改成小写的debug和release



(6) 最后在代码中加入#include "LitePeer.h"

Photon工作流程

(1).创建一个LitePeer类的实例对象

LitePeer \* m\_pLitePeer = new LitePeer(this);// 该类继承了public PhotonListener

(2).调用Connect()方法来尝试连接服务器，并通过实现IPhotonPeerListener.PeerStatusCallback来得到连接状态的通知，成功的状态是StatusCode.Connect。

(3).周期性的调用Service()方法来获取事件，同时发送"命令"(比如每秒十次)

(4).调用OpJoin()来加入游戏，通过实现OperationResult()来得到连接状态的通知，成功的状态是LiteOpCode.Join

(5).调用OpRaiseEvent()向服务器发送数据

(6).通过实现IPhotonPeerListener.EventAction()来获取服务器的数据更新

(7).调用LitePeer.OpLeave()离开游戏，若OperationResult()返回LiteOpCode.Leave则表示服务器端已经确认成功离开。

(8).调用Disconnect()以断开连接，通过PeerStatusCallback()得到连接状态的通知，StatusCode.Disconnect为成功断开连接。