跑酷教程3

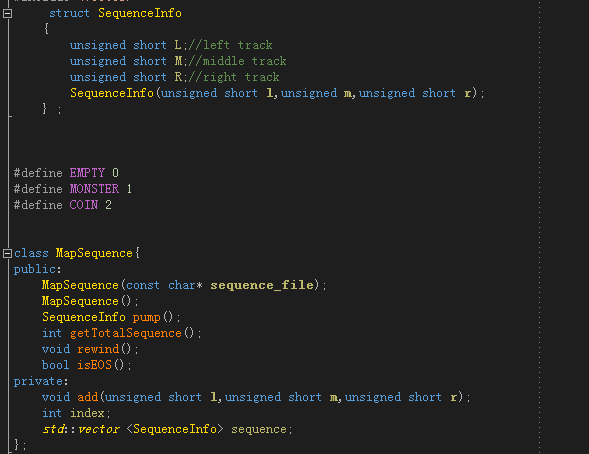
**[教程内所有素材来自网络，如有侵权，请联系删除]**

在第三篇教程这里，我们主要是为了添加障碍物，以及金币，并能让地图按照若干种定式不断的生成出来，为了简化问题，我们这里只分成两种物体，一种是障碍物——当游戏主角与其相撞时就会触发Crash事件，第二种是金币，金币在地图里会不断地旋转，若与游戏主角相碰的话，那么他就会触发GetMoney事件。

首先，我们要考虑的一个问题，是如何过程生成，一种简单的方式是按照给定的概率，在三条不同的跑道上面随机生成出金币和障碍物，这种方法最为简单，但是游戏体验以及收益的计算是不可靠的，因为如果单纯的依赖概率的话，那么不会产生许多很有趣的、连续的金币路径、故意生成比较难的关卡“赛点”等事件。同时，因为单纯的靠概率生成，游戏策划很难计算某一段路的难度，很难把关整个游戏体验。

为了让玩家既能看起来玩一些随机生成、充满想象的地图，又不能影响游戏的体验性，通常的游戏开发时采取的办法式，通过经验或者游戏策划精心的设计，产生若干个模式，然后再来根据概率随机播放这些模式中的其中一个。

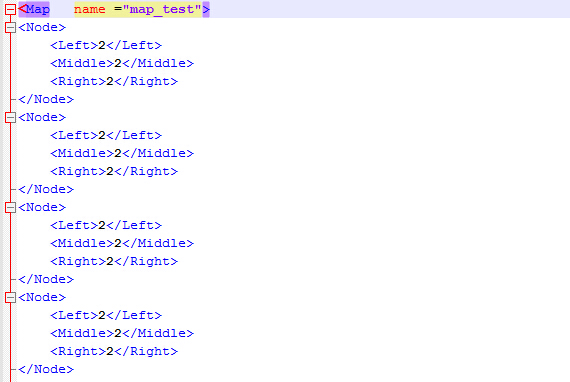
为了构建模式，在这里，我们创建了一个MapSequence类，这个类用于存放每个单个的模式



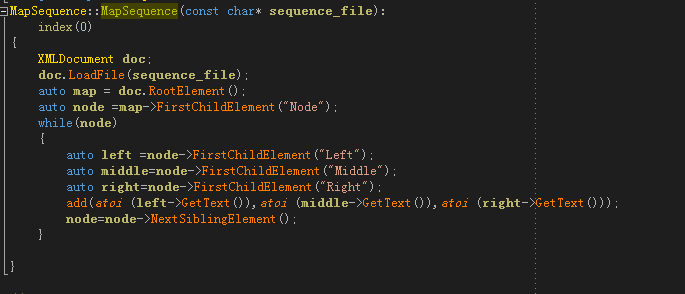
首先，用一个结构体SequenceInfo来储存每次出怪时，左、中、右 赛道会出现的类型，为了简化问题，我们现在暂定有三种类型，什么都没有、金币、障碍物。

SequenceInfo 被放置在MapSequence中的容器内，MapSequence通过构造函数读取一个外部设计好了的模式，并将内容转换成一个个的sequence，这样，模式就构筑好了。

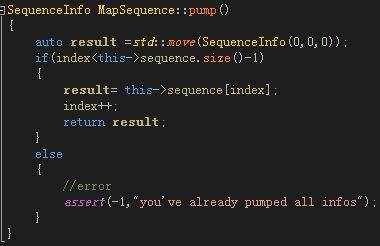
在这里，我们使用XML文件来作为模式的配置文件

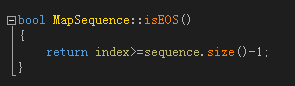


，并使用TinyXML2 来读取他们：



这样一个模式就已经读取完毕了。我们设置了一个Pump函数来每次弹出一个当前模式的情况，并用isEOS来判定是否已经取完了整个Sequence。

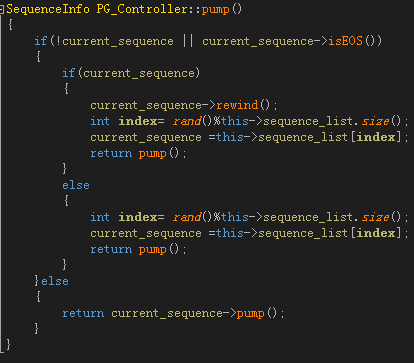




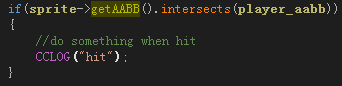
显然要让场景里显示出来，我们就需要将整个MapSequence放入到PG\_Controller中：

C:\Users\ziwen.tang\AppData\Roaming\Tencent\Users\576419632\QQ\WinTemp\RichOle\[]BU71LP%FS1A93)]K]}_}2.png

最终，我们在PG\_Controller处包装一层pump，让它不断的弹出应该出现的，如果弹完了，就递归调用自己本身，然后再重新的弹出:



接下来我们要检测怪物和玩家的碰撞，但是，一个模型通常是一个由几千，几万面组成的一个复杂几何体，如果我们直接将两个模型当做多边形来考虑，那么效率就会非常的低，这里我们使用了一个近似的方法——AABB，这里可以使用Cocos2dX 为Sprite3D提供的一个方法getAABB()；所谓AABB就是物体的轴对齐包围盒，这个包围盒可以想象成一个紧紧的把模型给包住的盒子，这样，我们把对模型的多边形的碰撞的检测，转换成对包围盒的检测，这样做效率就高了不少，同时误差也是在刻意接受的范围内。我们在障碍物的Action里的update里使用障碍物（或者）金币的AABB于主角的AABB检测是否相交：



使用同样的方法，给金币的Action里做一些处理。

这样，整个基本的跑酷demo逻辑就完成了，我们来看看最终的效果：

