大地图优化方案

我们那个项目里主要是采用一个地图切块的方式进行加载。

是写好那个切块的工具，然后让美术那边去对地图进行一个切块以及地图上资源的调整，调整完之后再给我们手里进行逻辑代码的编写。

切块的大小，我记得项目里是64\*64的，然后整个地图尺寸是512\*512的，分了64块，当时地块打ab包，是所有的地块打一个包

/////**可能问的问题**

**（地块的刷新规则，是根据玩家位置的变化进行刷新，根据位置计算出 玩家所在的地块下标，下标发生变化，就刷新地块）**

**（九宫格加载，是在玩家所在的地块发生变化的时候才会刷新，然后如果你在一个地块的边缘，但是并没有跨过边界线，此时玩家前方只有一个地块，玩家视野前方就会出现空地，属于是一个bug，可以不用管，就说因为地图上会有一些树或者建筑物，也可以设置迷雾的效果，阻碍玩家视野，让这个bug不突出）**

**（地图尺寸，512\*512可行 但是因为是大地图【打包后是几百mb的那种】 他可能问这就是整个地图最大尺寸嘛？就说这是某个区域的尺寸，比如野外战斗）**

**（分块一般就是规则形状，然后不规则该如何切块：方案是取地图的最大长宽高，进行切块，然后标识好是可行走区域还是不可行走区域，区域的标识是地编或者主美那边做的 是有个地图编辑器，根据角色的步长和地图的尺寸进行划分，会生成一个配置表，表中是地图区域的一个状态）**

**（打包，如果地图太大 10240\*10240的话 包体就会很大， 如果分成四个区域包打的话，如果玩家，除在中心的话，四个包都会加载到内存 【没有一个比较好的解决方案】，暂定打一个包）**

地图分块之后如果出现缝隙的话，是因为贴图边缘没处理好，或者相邻地图块之间没有正确对齐，如果是地图块没对齐的话，就在地块的边缘增加一些重叠区域，多切几像素，如果是因为贴图的话，就去检查一下uv映射就可以

地图块的管理方案主要是采用九宫格分层进行管理，分层是显示层有两个，一个是脚下那一块地图，一个是周围八个地图，还有就是缓存层，因为加载地块的ab包是整个ab包 都加载了 然后那些地块资源都存在这个缓存层，需要显示的时候就放到那个 显示层，不需要显示的就放入缓存层

当时还准备了另一个方案，就是每个块打一个包，需要显示的加载ab包到缓存里，然后不需要显示的直接卸载掉

选择第一个方案是因为 地块都打一个包占用的内存也并不大，是可以接受的，而且也不用频繁的读取资源，第二个方案就需要一直去读取资源，增大了消耗

然后项目中地图上的资源管理采用的是四叉树进行管理，就是把地图划分为一个层次结构，每个地图块分成四个节点，如果节点里面有资源就划分然后没有的话可以不用划分节点，这里当时发现一个问题，就是有的资源太大，它的模型会横跨好几个节点，应该归属哪个节点？最后讨论出来的解决方案就是，一个是按照他的中心点划分，中心点属于哪个节点就划分给哪个节点，还有一个是这些个节点同时持有他，查找这个资源的话，就算都持有他，但是最后都是指向这个资源，也不会有冲突啥的

然后就是资源显示这一块，每个节点也是给他画一个包围盒，玩家的包围盒碰到之后就显示节点内的资源，但是这样有个问题，如果有个资源横跨好几个节点范围，然后就是 玩家进入资源范围了 但是还没碰到节点的包围盒，资源不会显示，当玩家碰到节点的包围盒之后，玩家眼前就会突然出现物体，是个bug，所以我们当时讨论出来一个方案，就是在资源身上有一个包围盒，如果玩家的包围盒，碰撞到资源的包围盒后，通过四叉树查找到那个资源，给他显示出来

然后地图优化这一块还可以使用LOD进行优化，通过更改模型的精度，减少三角形的数量，降低顶点数，减少几何复杂度，也减少了cpu的负担

还有就是当人物站在高处俯视的时候，需要加载的地方就更多，然后就需要近处的加载清晰，远处的加载模糊，甚至更改成2d贴图，距离玩家远的一些山等 高大的资源，也可以直接整个贴图

贴图分为png格式和jpg格式，png格式可以是透明图，jpg的透明就是白色地图，渲染上白色底图更容易加载，png的占用内存要比jpg要大好多倍