1. MergeDragHighlight

处理逻辑：合成物拖拽时，相同的元素高亮

包含方法如下

public static void SetHighlightMatches(this ElementEntity dragEntity,bool highlight, bool isShowArrow = false)

public static void SetHighlight(this ElementEntity elementEntity, bool highlight, bool isShowArrow = false)

问题：

这种添加方法，破坏了类的封装性

框架设计中一个类应该提供什么方法，是经过需求判断和思考的

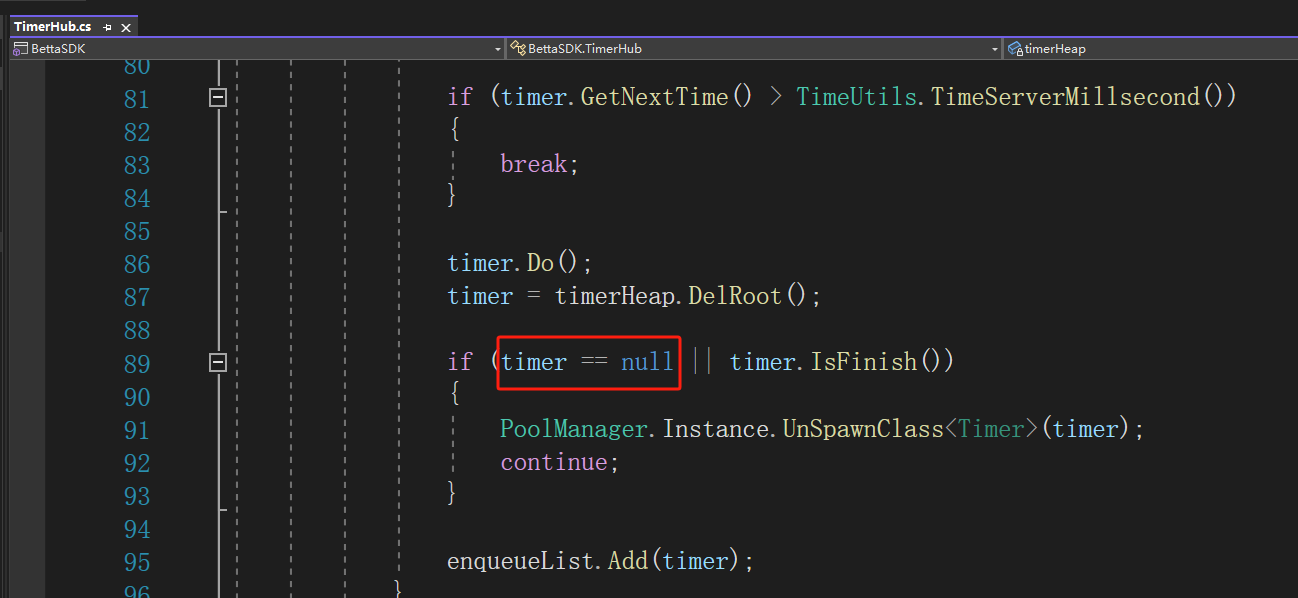
不应该在自己逻辑中随便给原有类添加扩展方法，

尽量少加这种方法，这种行为修改了原有类提供的方法

这种方案多见于使用第三方库，不方便修改源码，有特别想给某个类添加方法

这里把 MergeDragHighlight 作为工具类更好一些，而不是给 ElementEntity 添加扩展方法

2.TimerHub



内存池使用问题

timer 为空，不能放回到内存池

3.GridEntityLogic

GridState \_state;

public void TransitionState(int toState)

{

\_stateMachine.TransitionState(toState);

\_state = (GridState)toState;

\_entity.Data.SetGridState(\_state);

if (\_entity.View is GridEntityView view)

{

view.SetSprite((GridState)toState);

}

}

状态机使用问题

TransitionState 方法中

如果是所有状态都需要执行的方法，可以实现一个共用基类，然后在基类状态中实现方法

如果是某些状态单独处理的方法，每个状态有 OnEnter OnExecute OnExit 生命周期方法，在生命周期方法中执行状态独有的逻辑

GridState \_state; 变量，此处不需要记录当前状态

如果从状态机获取，\_stateMachine.CurrentState.State; 获取的就是当前状态

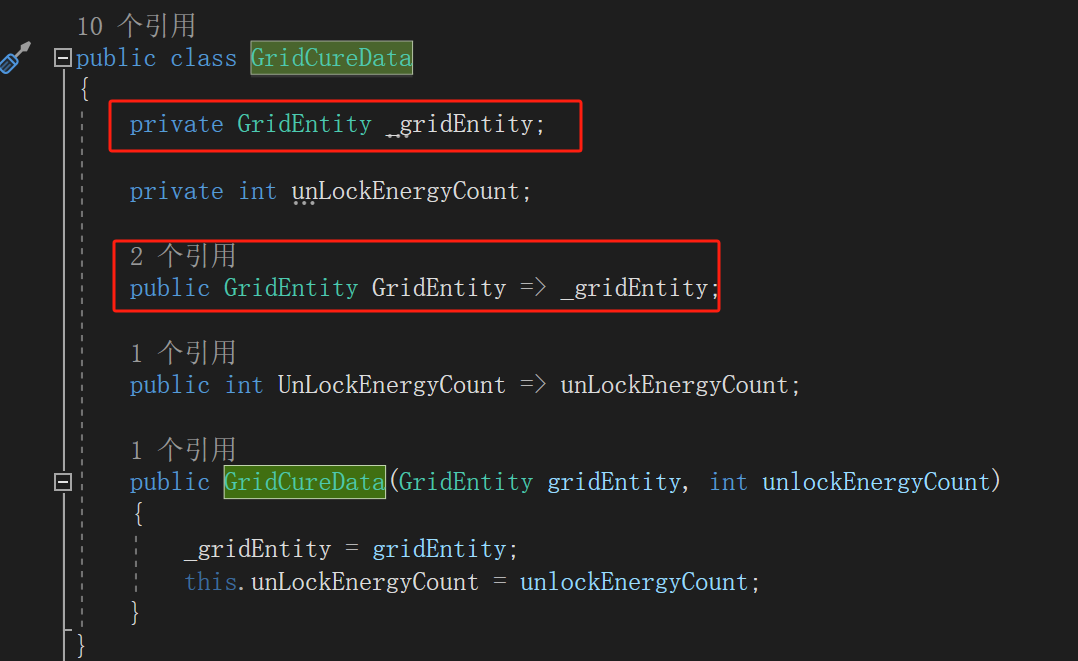
如果从 Data 获取，\_state = \_entity.Data.GetGridState();

一个变量最好不要存在多处

4.EnergyEntity

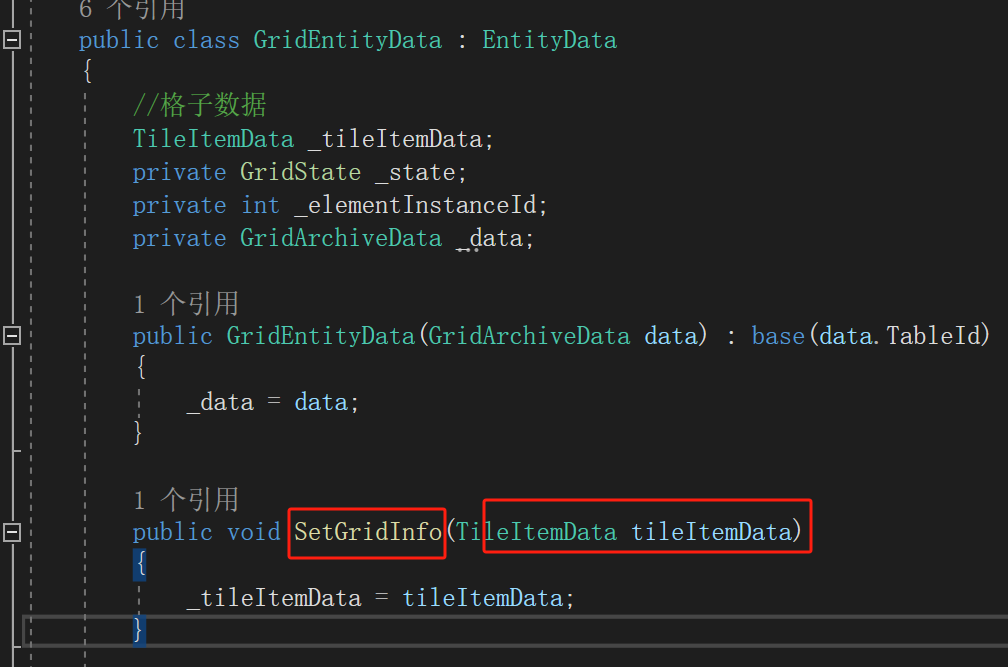
虽然添加了 EnergyEntityLogic 、EnergyEntityView，并没逻辑实现， 逻辑与表现的逻辑都在 EnergyEntity

5.GridCureData



Data 应该是存数据的，存 Entity 不合理

6.GridEntityData



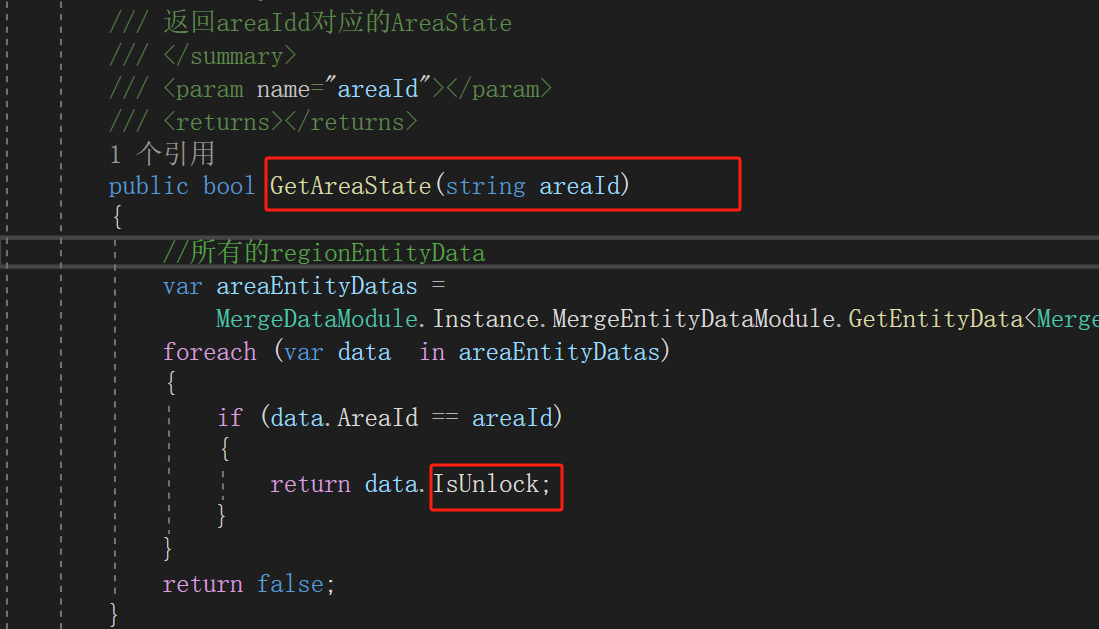
方法名尽量见名知意，这个更像是 GridEntityData，SetItemData



格子状态应该是从 \_tileItemData需要能量和当前能量判断的，尽量不要再记录状态值

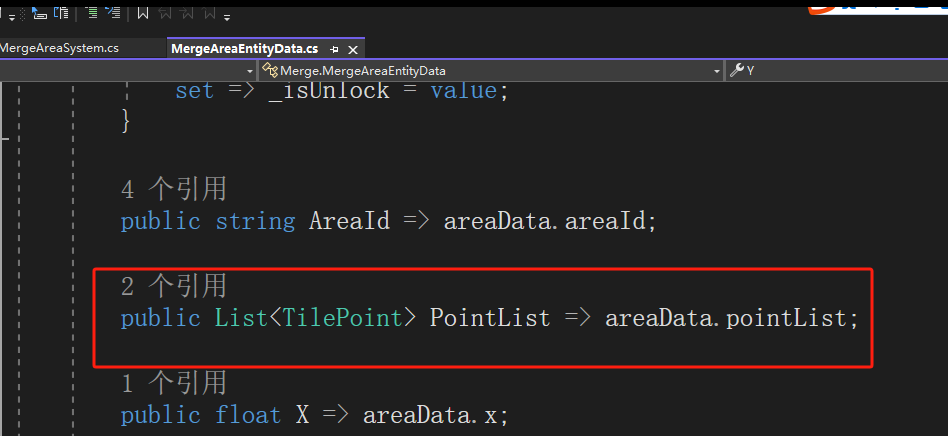
理论上修改的是 Grid 对应的 tileItemData 数据，不应该有 SetGridState 方法

7.MergeAreaSystem



方法名 GetAreaState 返回的是 IsUnlock, 感觉方法名与实现不匹配

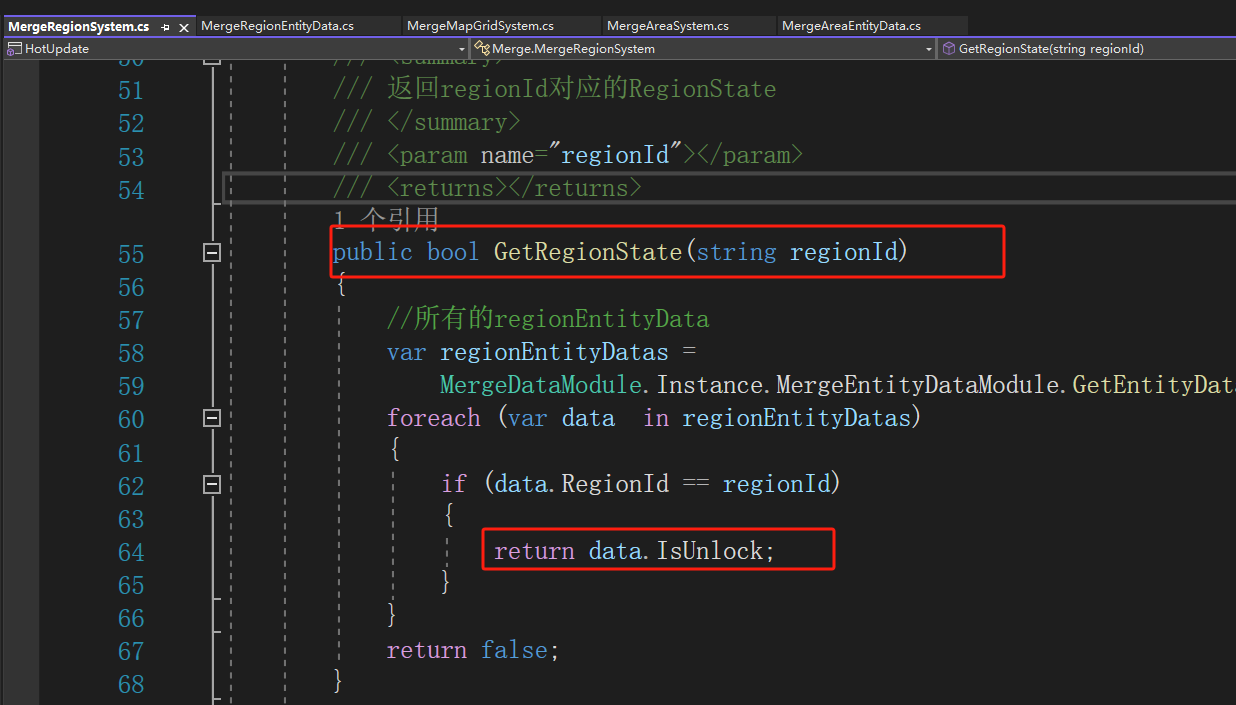
8.MergeAreaEntityData



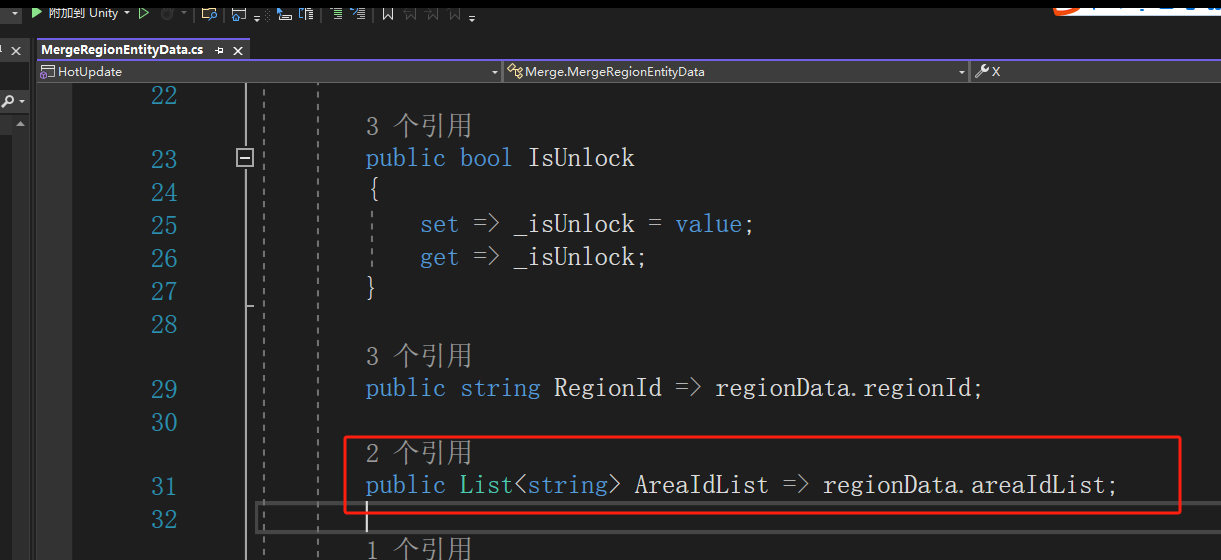
修改为 public IReadOnlyList<TilePoint> PointList => areaData.pointList; 更好

理论上这里是只读的，使用IReadOnlyList ，如果返回 List 外部代码可能会误操作，删除数据

9.MergeRegionEntityData



方法名与实现不太匹配



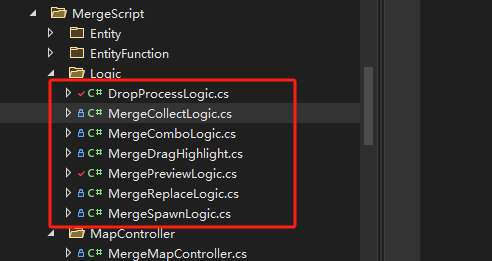
修改同 MergeAreaEntityData

10.ElementEntityFactory

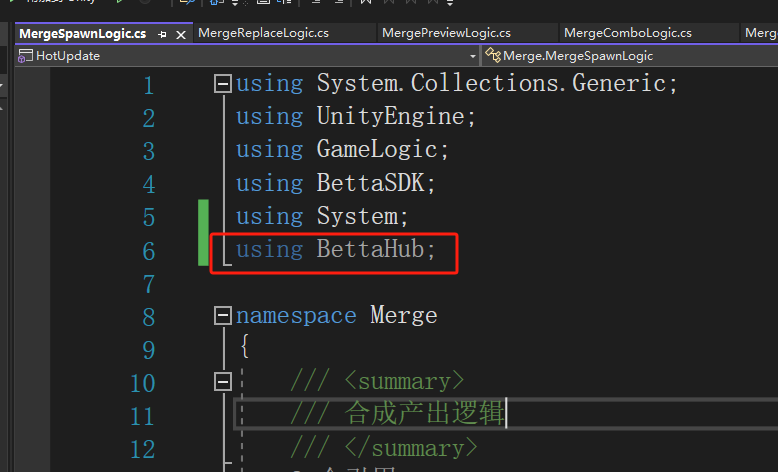
这个添加的工厂并没有写实现，也没有被利用

11.

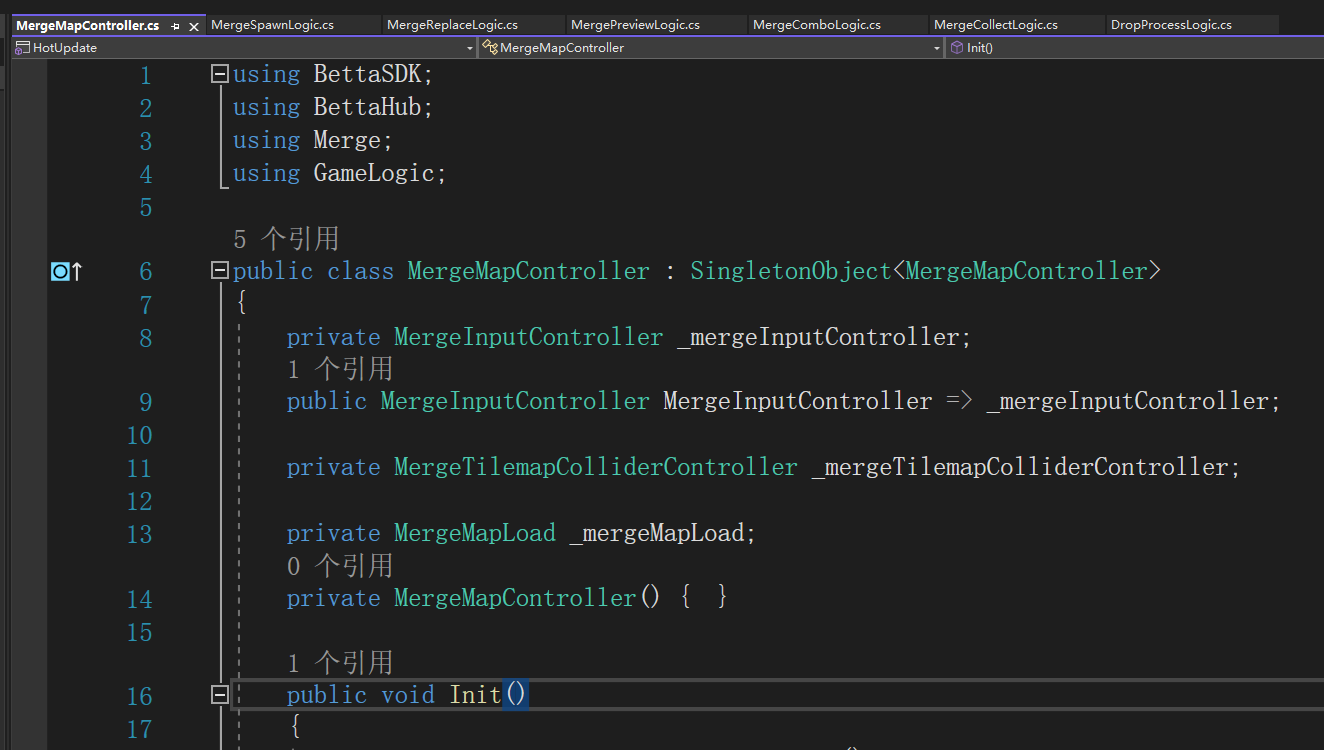
这些都需要整理，最终添加到 MergeSystem

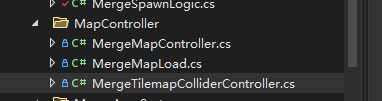


12.如下没有使用的 using 可以删掉



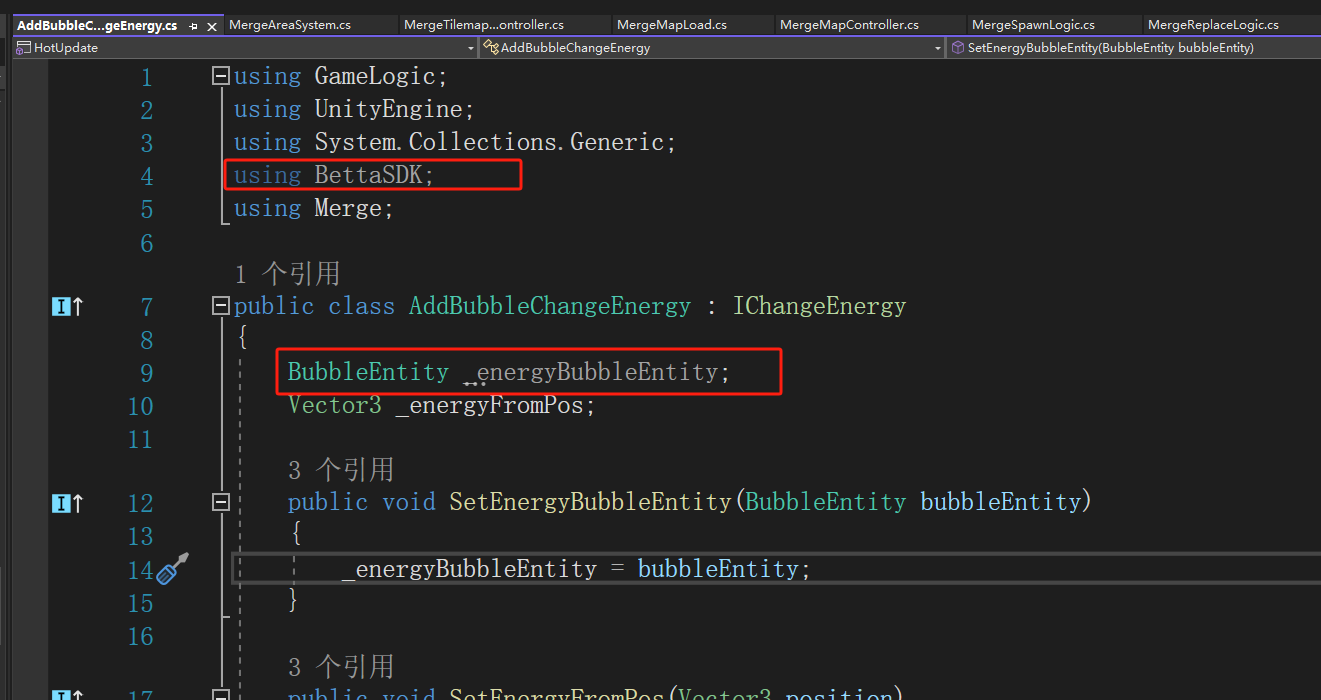
13.MergeMapController





这些代码要整合到 MergeSystem

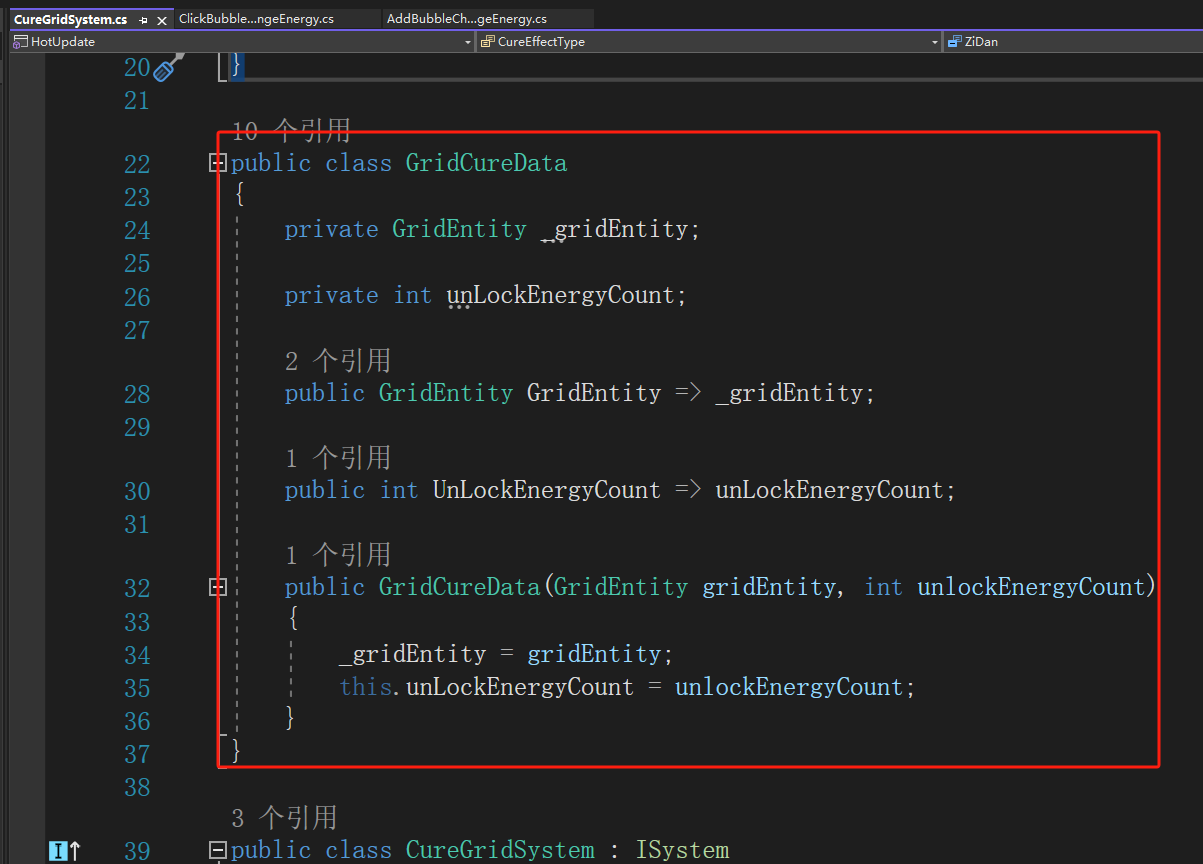
14.AddBubbleChangeEnergy



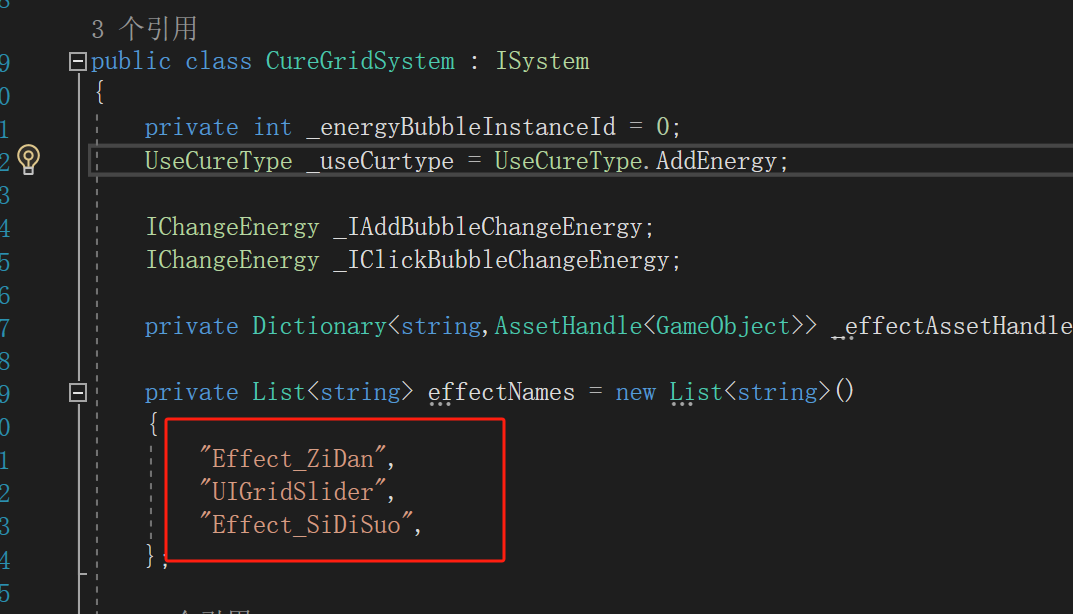
AddBubbleChangeEnergy 更像是 方法名，作为类名有点不太合适

ClickBubbleChangeEnergy 同上

15.CureGridSystem

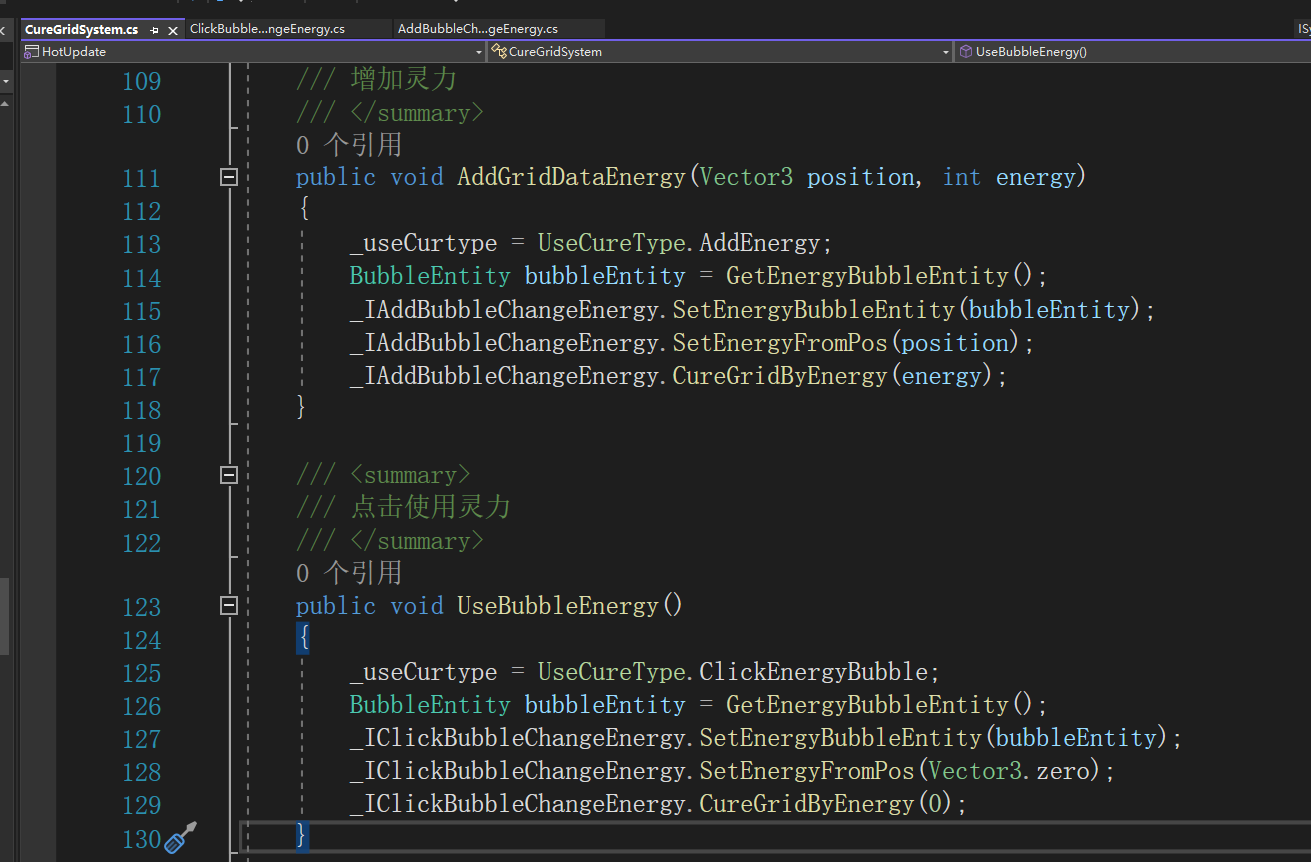


Data 尽量在合适的位置创建文件夹

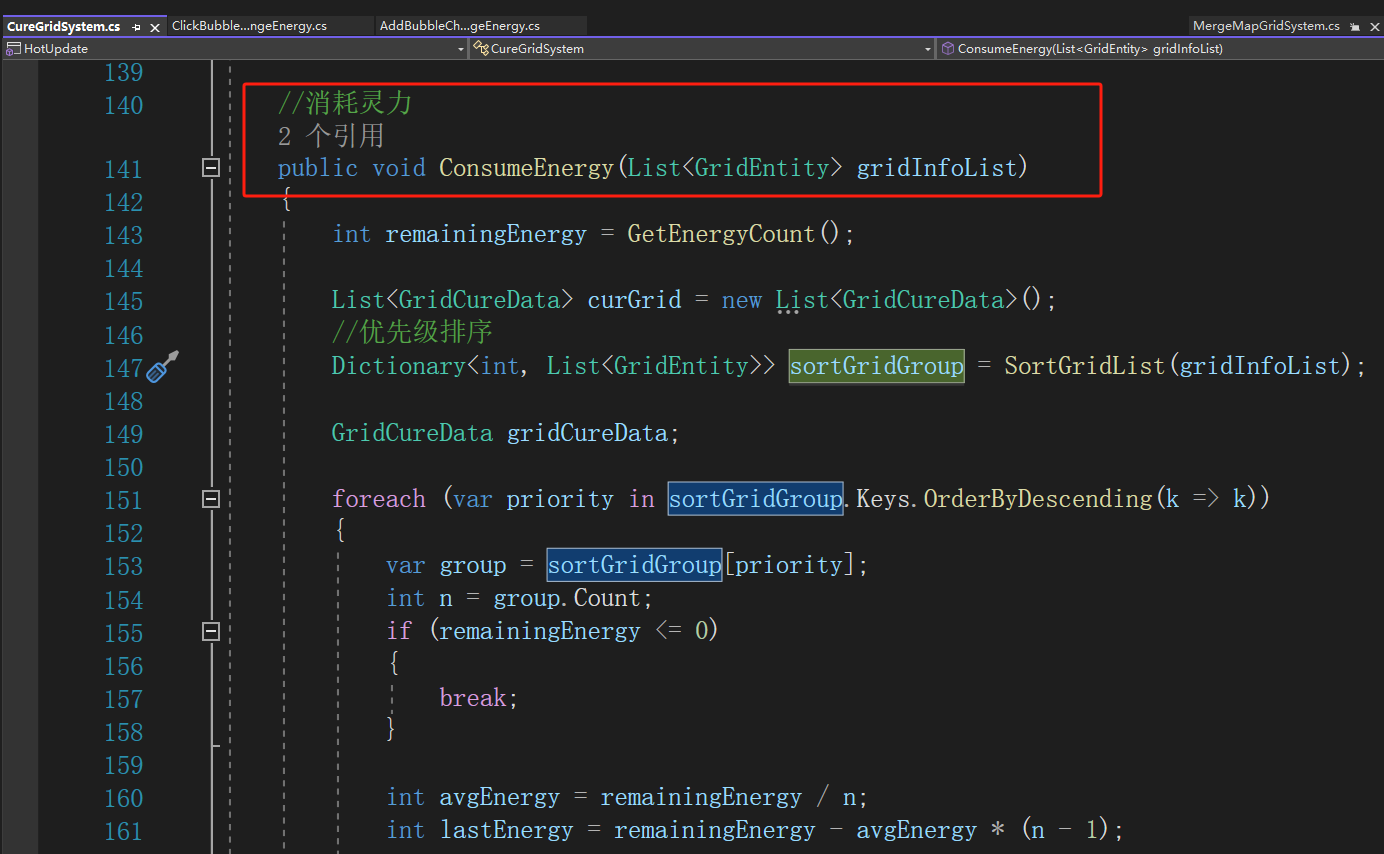


代码逻辑中尽量不要加配置

可以添加到 GameConstants 或者 后续要添加的常量表



这部分实现再斟酌一下



这个方法的逻辑实现效率不高，考虑是否可以使用 大顶堆来帮助实现逻辑

CureGridSystem 中既有逻辑代码，又有表现代码，考虑拆分

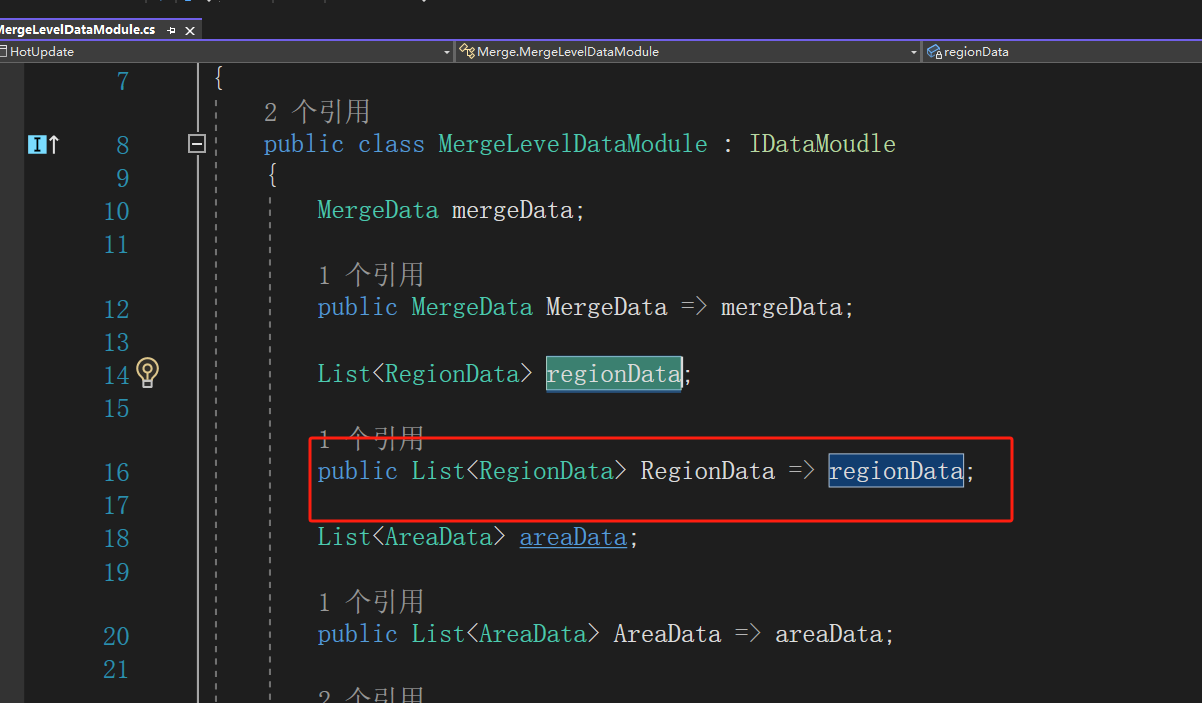
16.MergeDataModule

合成数据模块设计了，后续三合中的数据存储、获取，尽量转移到这里

17.MergeLayerSystem

这个添加的必要性，是否可以省略，添加到何止的代码中

18.MergeLevelDataModule



数据的获取需要最终确认逻辑

这里最好返回的是 IReadOnlyList

这里边的逻辑还需要完善

19.MergeMaterialSystem

代码优化整理

20.UIGridSlider

这个的实现应该是 UIBubble

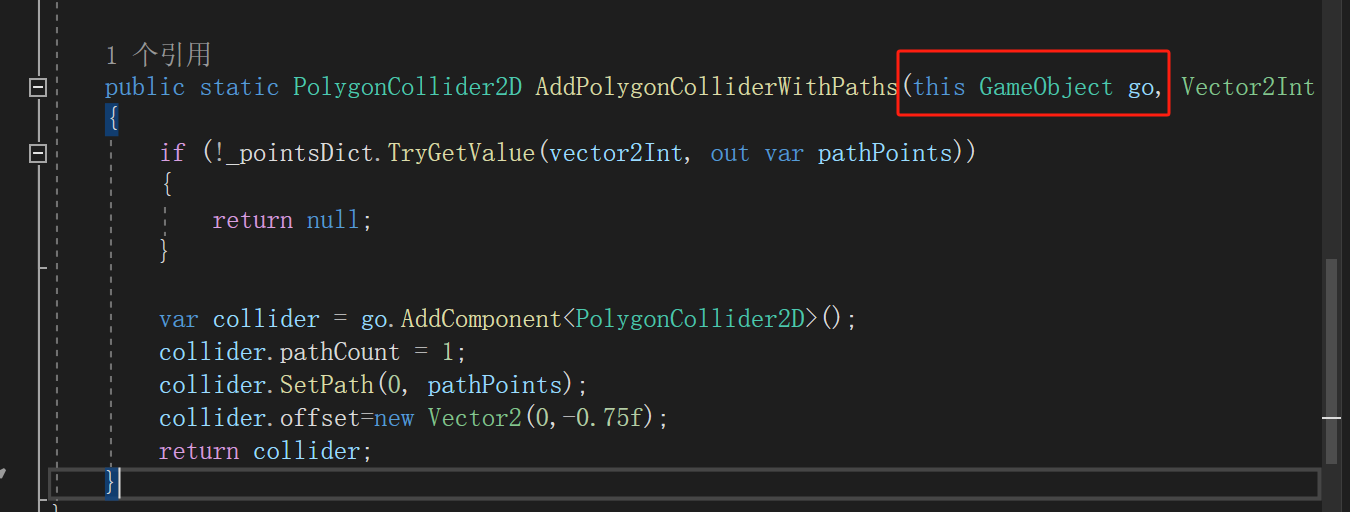
UIBubble 这个名字有些歧义，需要考虑换一个名字

20.EntityUtils

private static Dictionary<Vector2Int, Vector2[]> \_pointsDict = new Dictionary<Vector2Int, Vector2[]>

这个名字跟用途不匹配，其实是不同 Size，对应的格子四个顶点相对于地块的坐标

这种常量考虑添加到 GameConstants 或者 常量表



不太推荐使用这种扩展方法

21.MergeUtils

这个工具类目前没有任何实现，考虑是否有必要保留