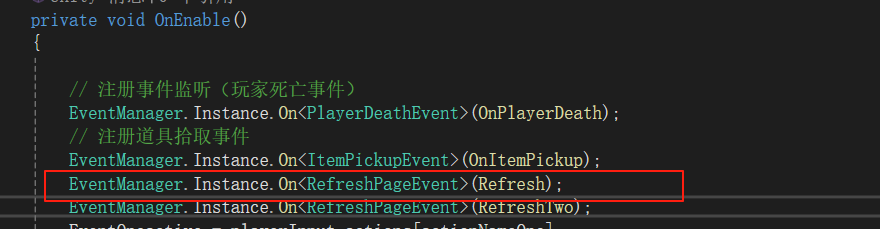
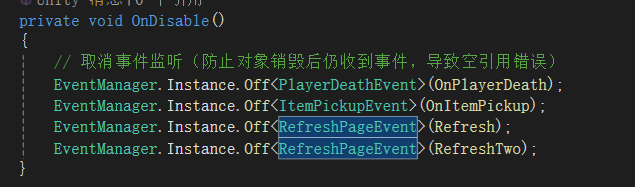
一事件管理类（文档最后有无脑使用流程图）  
主要优化：传统事件类流程多，结构不清晰，最重要的是分布不清晰，所以一般会专门设置管理类去统合管理所有的事件。

这里只讲用法，具体结构和示例可以去scripts脚本下的NewEvent去查看  

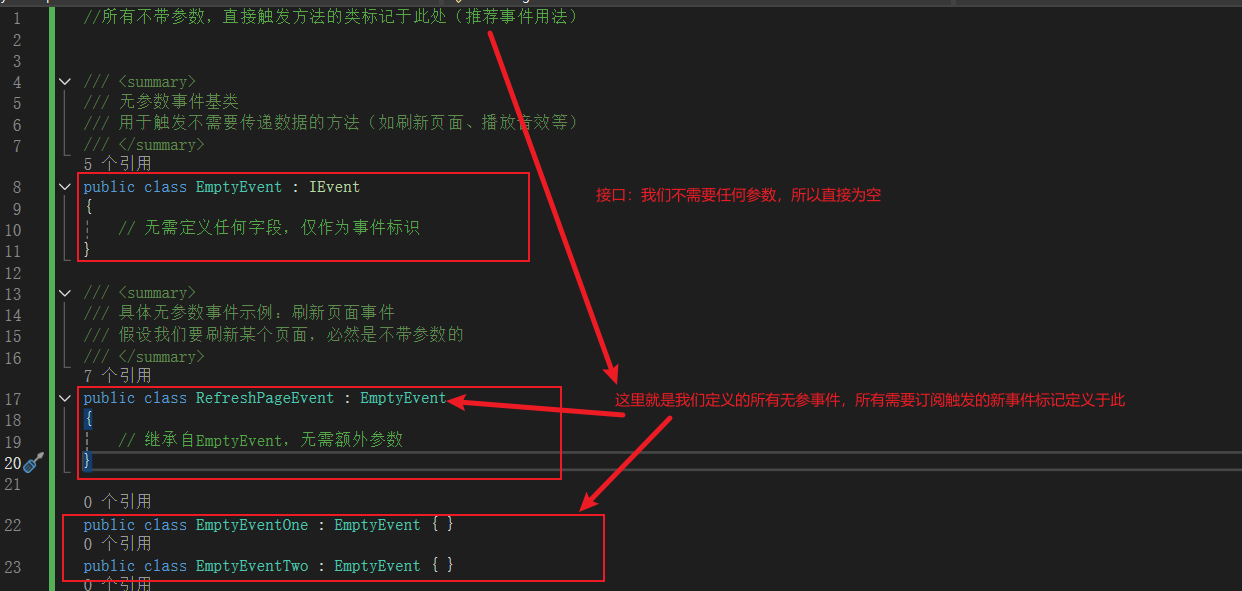
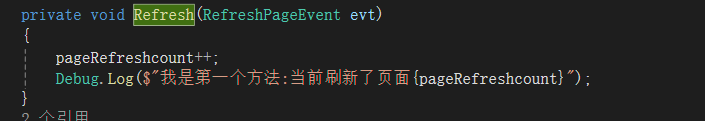
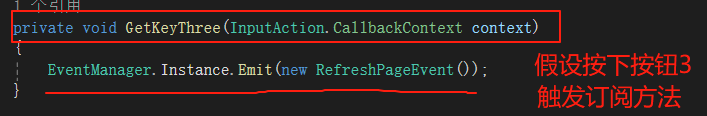

首先假设最简单的一种，玩家完成了任务，在该页面领取了某一物品。此物品的数量在该页面正常显示着，那么领取了之后该页面是不是应该刷新一次，让text文本重新赋值。

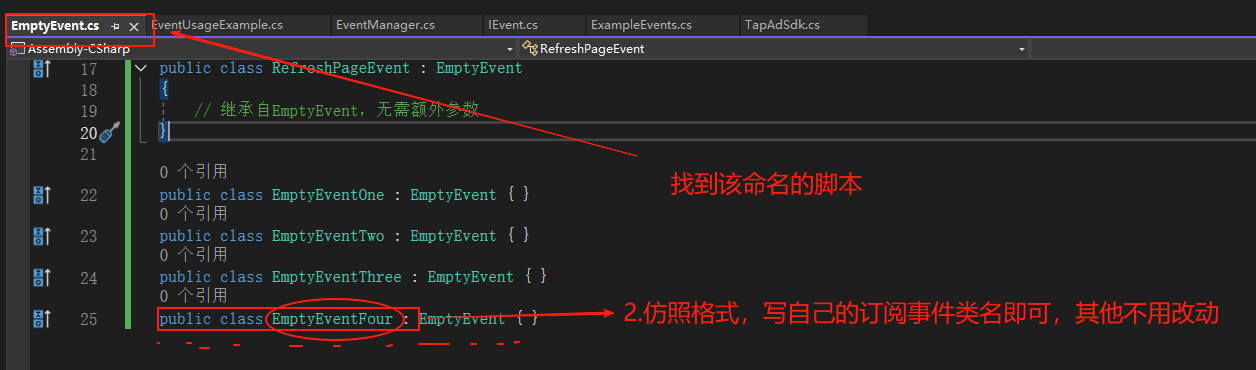
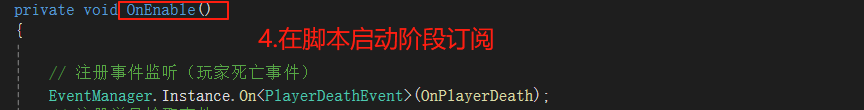
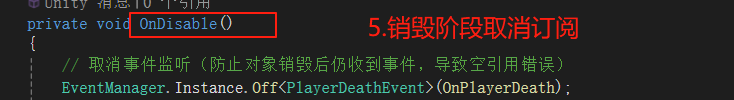
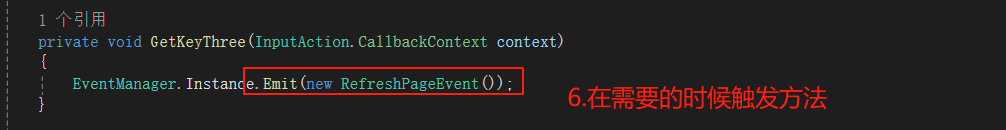
那么后续的事件流程如下，  
  
1.首先是事件的订阅和取消订阅  
在脚本开启的时刻，我们需要订阅我们的刷新方法

EventManager是我们的单例管理类，可以直接调用。这里的On方法就是订阅，<>括号里的是我们定义的事件接口，Refresh是我们当前绑定刷新方法。  
  
  
注意：订阅不是覆盖，一定要取消订阅，



我们直接看<>内和（）里的结构

首先是<RefreshPageEvent>类，它相当于此订阅的：name，我们得通过此类的名字才能寻找到我们所定义的订阅。通过有着这个name名的订阅去触发我们的方法  
  
  
然后是（），我们绑定的方法  
  
这边的本地刷新方法中的行参，主要是为了后续事件中带参的方法，目前只需要记住默认填写EventManager<>中相同的类即可。  
  
2订阅成功后，我们需要调用方法，调用则需要触发此订阅的方法  
  
这里用EventManager.Emit即可触发，注意new 后面的类仍然于我们一开始定义的类name一致。  
  
至此，事件便结束了，此流程让事件订阅触发简化为了四步，并且非常方便于管理所有的事件类  
  
  
  
  
  
  
  
  
  
UI管理框架

事件无脑操作步骤  
1.2  
  
3.  
  
4.  
5.  
  
6.  
  
结束  
  
  
  
  
  
  
  
  
  
UI框架的使用  
直接上视频（8/22晚更新）