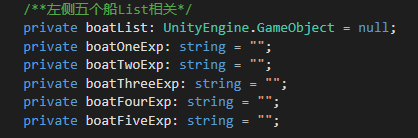
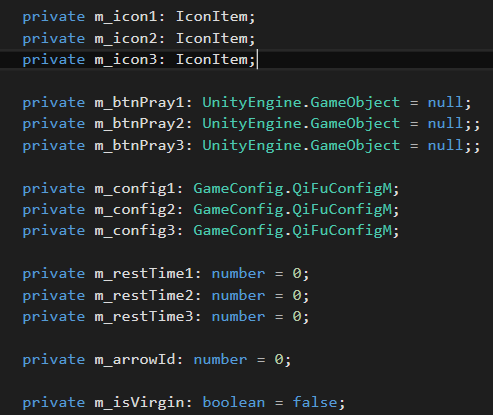
1. 变量命名不规范

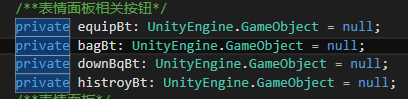
比如用one/two/three… left/right… 等命名，命名应该能一眼看出其含义和用途，并保持连贯性和可读性。比如下图，应改为boatExp1/boatExp2/boatExp3…，再考虑到实际用途，使用声明1个数组变量取代这5个变量更为合适。



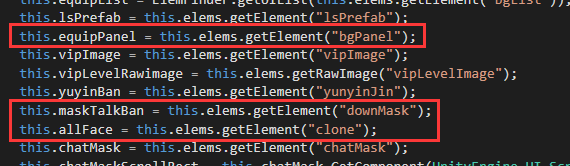
又如下图



再如下图，button的简写一般是btn，而不是bt

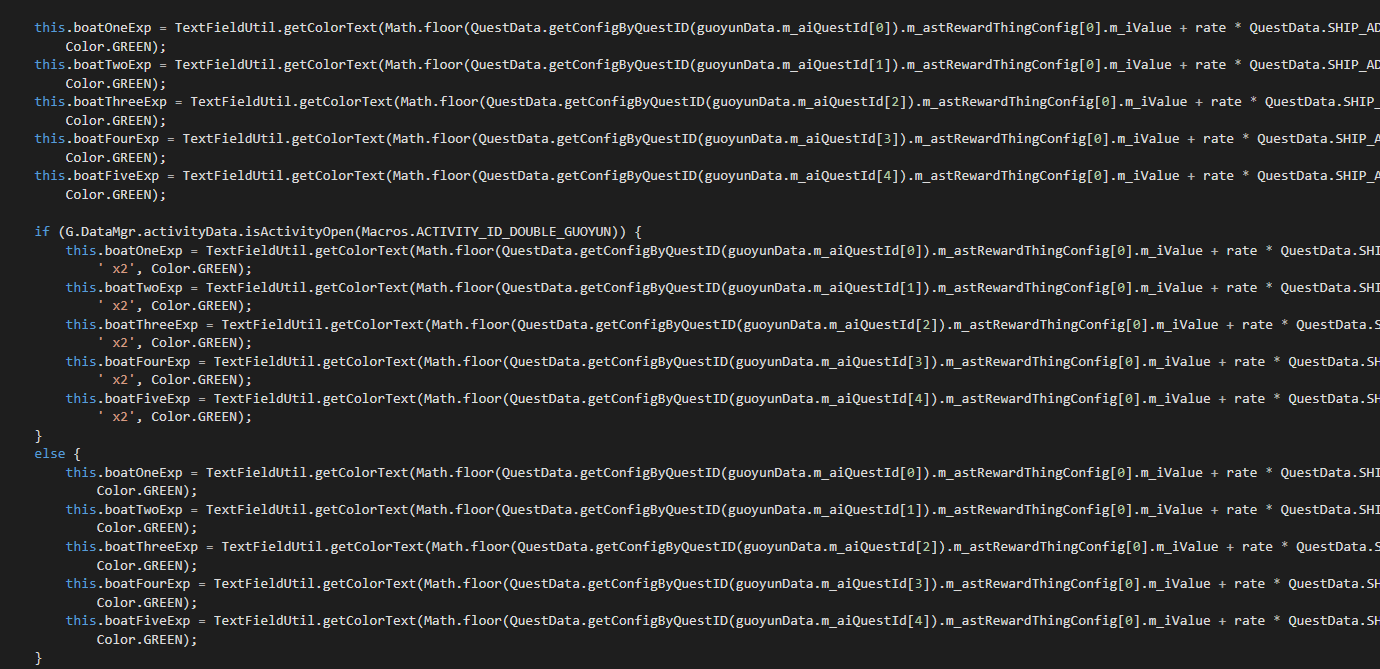


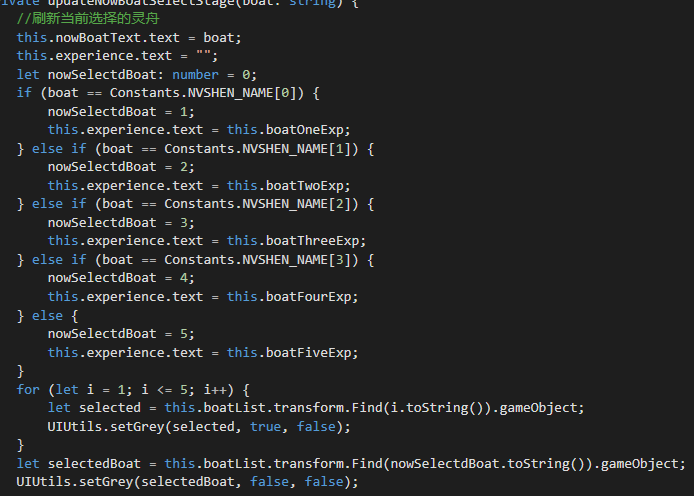
再如下图，程序中的命名与ui中的命名不一致



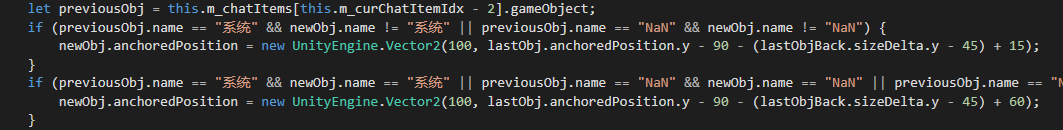
1. 大片雷同代码拷贝粘贴。

如下图，应改为使用循环

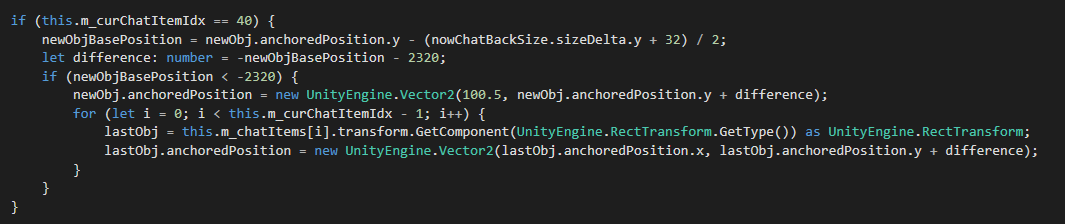




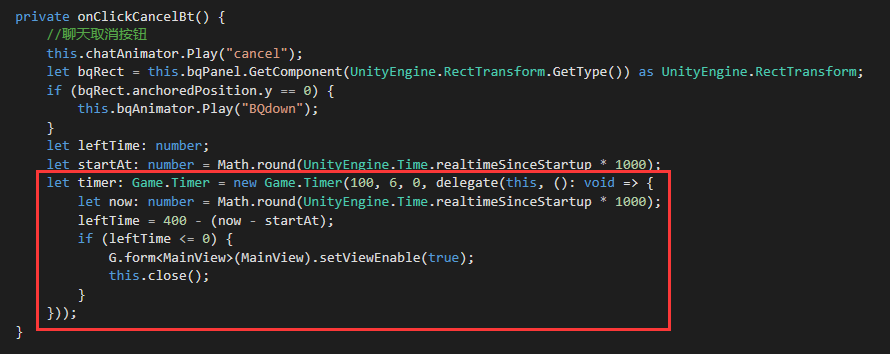
1. 使用GameObject.name记录某些逻辑数据用于后续使用，比较超出一般程序员的想象。如有必要，使用Game.UserData



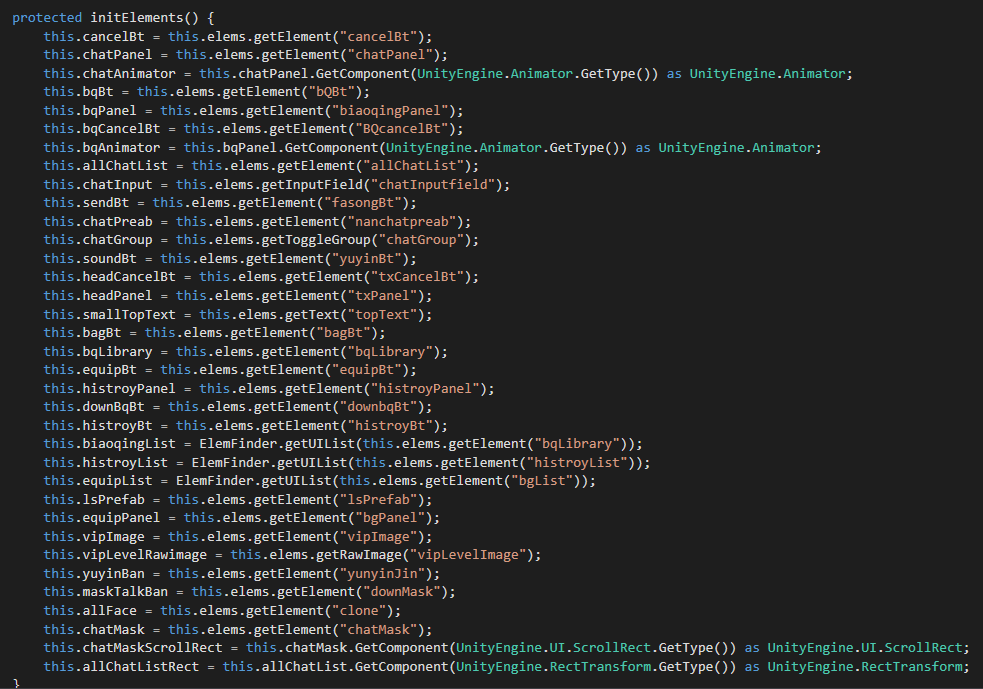
1. 随意使用各种Magic Number，这种代码可维护性极差



1. 随意使用Timer，任何Timer，都必须确保有显式逻辑将其Stop掉。



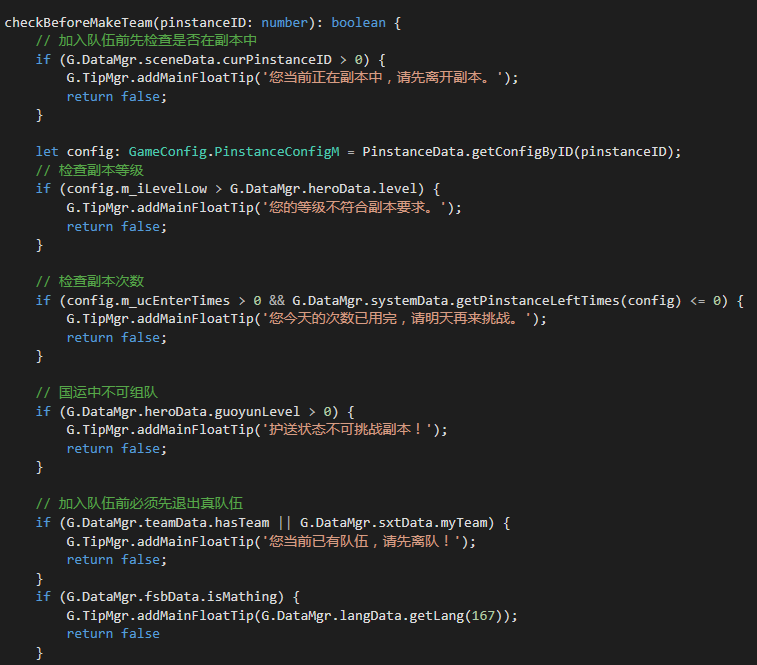
1. 应习惯对代码进行分块排版，增加可读性，反例如下图，黑压压一片，再加上一句注释也没有：



应当如下图，边写逻辑边写注释，说明当时你的逻辑清晰：

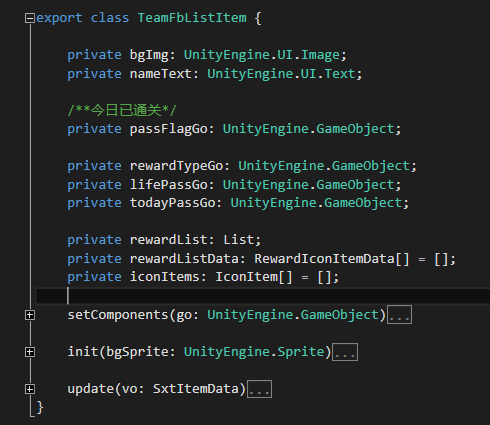


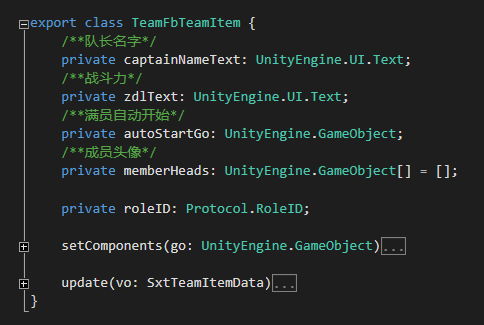
再如下图：



1. 应习惯对某些独立部件进行封装，这样有利于逻辑抽象集中和重复利用，以组队副本为例：







1. 要形成与团队代码风格融合、与原项目代码风格具有一定传承的个人代码风格，且始终如一前后一致，有助于团队其他成员快速理解自己的代码。养成了自己的风格，也说明你形成了较为成熟的逻辑思路。切忌今天这样写，明天这样写，或者大家都这么写，就你这样写，增加别人看代码时的思考成本。比如上述的Item封装，我基本上都遵循了如下风格：setComponents/init/update

