01、GPU的优化（主要通过OverDraw即像素的重复绘制，进行优化）

不可见的UI禁用

边框背景图Image的 FillCenter去勾选

不使用空的Image接受Touch事件

2、CPU的优化(主要通过避免Canvas对网格重建来进行优化)

使用尽可能少的UI元素

将动态的UI元素（频繁改变例如顶点、alpha、坐标、颜色和大小等的元素）与静态的UI元素分离

（比如仍然以背包界面为例，滑动内容会导致界面的变化，

在滑动过程中每帧都会有Mesh的重建操作。把滚动的部分做成单独的Canvas，

这样背景图片和标题等部分就不会被每帧重建了。）

谨慎使用enable和disable

谨慎使用Text BestFit（Best Fit表示让输入的文字自适应文本框的大小，

也就是说在文本框内输入的文字过多的时候，文字的字体大小会缩小，以保证所有文字都能显示出来！）

谨慎使用Canvas的Pixel Perfect（PixelPerfect(完美像素)指的是

一个UI素材本身的像素对应屏幕上一个像素的情况，这种情况下UI素材映射到屏幕上时没有

任何拉伸和压缩，这种情况下UI显示效果非常清晰完美。）

对于不需要接收touch事件的Graphic组件，一定要禁用Raycast Target

3、动态批处理（降低DrawCall）

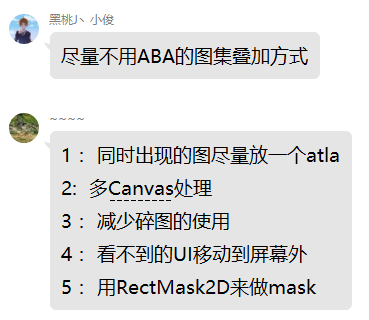
引入图集和对图集进行管理

不同UI元素尽量不交叉，重叠

不同图集的图片尽量不交叉

相同字体的文本控件尽量相邻

避免使用Mask组件



类和结构体的区别

1. 引用类型和值类型的区别
2. 继承的区别，结构体是隐式密封的，不可被继承，可以继承接口不可以继承类
3. 构造函数和析构函数的区别，构造函数是用来初始化成员变量的，结构体可以不使用构造函数去初始化，不可以删除或者重定义默认的构造函数，不能有析构函数
4. 不允许字段的初始化

值类型和引用类型的区别

1. 继承不同，一个继承自System.ValueType
2. 存储位置（值类型和指针总是分配在被定义的地方，他们不一定被分配到栈上。）
3. 变量的存储内容不一样
4. 复制传递方式不一样

堆和栈的区别

1. （内存分配时机）栈是编译期间就分配好的内存空间，因此你的代码中必须就栈的大小有明确的定义；堆是程序运行期间动态分配的内存空间，你可以根据程序的运行情况确定要分配的堆内存的大小

2、（存储有序无序和连续性）存放在栈中时要管存储顺序,保持着先进后出的原则，他是一片连续的内存域，有系统自动分配和维护。

而堆是无序的，他是一片不连续的内存域，有用户自己来控制和释放，如果用户自己不释放的话，当内存达到一定的特定值时，通过垃圾回收器(GC)来回收

3、（垃圾回收速度）栈内存无需我们管理，也不受GC管理。当栈顶元素使用完毕，立马释放。而堆则需要GC(Garbage collection:垃圾收集器)清理