协程原理

协程和线程

**网络通信**

**UGUI怎样优化**

**换装**

**聊天系统**

**Shader**

Unity打包原理

如何实现游戏资源的下载，和assetbundie有关

安卓和unity之间如何传递信息

## unity3d 从唤醒到销毁有一段生命周期，请列出系统自己调用的几个重要方法。

Awake->OnEnable->Start-> OnTriggerXXX(Collider other)->OnCollisionXXX (Collision collisionInfo)->Update-> FixedUpdate-> LateUpdate->OnGUI –> Reset –> OnDisable –> OnDestroy

## 如何判断是否需要优化？

在Unity中查看DrawCall参数，Window / Profiler 或者Ctrl+7 快捷键打开 Profiler性能分析器面板。

Drawcall：CPUGPU执行渲染操作，渲染流程采用流水线实现，CPU和GPU并行工作，它们之间通过命令缓冲区连接，CPU向其中发送渲染命令，GPU接收并执行对应的渲染命令。

## 说说你对UI优化思路

使用图集、减少UI之间的覆盖、使用canvas对不同层级的UI进行分层、销毁不使用的UI界面。

## 如何优化内存？

1.压缩自带类库。

2.将暂时不用的以后还需要使用的物体隐藏起来而不是直接Destory掉。

3.释放AssetBundle占用的资源。

4.降级模型的片面数，降级模型的骨骼数量，降低贴图的大小。

5.使用光照贴图，使用多层次细节（LOD），使用遮挡剔除，使用着色器（Shader），使用预设（Prefab）。

6.代码中少产生临时变量。

## 代码书写规范优化性能

1. 减少循环调用
2. 仅在改变是更新显示信息
3. 增加代码更新的延时（分帧执行）

## 如何减少Drawcall？

1.使用Draw Call Batching，静态批处理。

2.通过把纹理打包成图集来尽量减少材质的使用。

## 如何解决过多创建和删除对象带来的卡顿问题（节省内存？

## ）

1、使用预制物体对象 Prefab

2、使用对象池技术，不使用时关闭，使用时打开

## 你认为unity在开发过程中哪些地方比较容易造成内存泄漏和内存泄漏问题？如何避免？

脚本做载入不删除处理时，往往脚本中还有一些不用的变量，容易导致内容泄漏

使用GC进行巡查清理

## LOD是什么，优缺点是什么

LOD 多层次细节，是常用的游戏优化技术。它按照模型的位置和重要程度决定物体渲染的资源分配，降低非重要物体的面数和细节数，从而获得高效率的渲染运算。缺点是增加了内存

## Unity资源加载的有几种方式，简述asset bundie

1. 直接在脚本中public一个对象，然后在监视器面板进行赋值

2. 直接在程序中进行find查找

3. Resource.load

4. 把资源打成Assetbudle，然后用的时候load进来

assetbundle就是对资源的打包处理，同时这种资源格式便于从互联网上下载

## Unity3d中resources目录一般用来放些什么，打包的时候会有什么影响？

resource一般用来放置一些需要动态加载的资源，打包程序的时候会将Resource目录下的全部文件都加密压缩打进包内，这样再想使用assetbundle方式打小包的话就不行了

## 多线程和协程

多线程和协程的区别

协同程序：与多线程下的线程比较类似：有自己的堆栈，自己的局部变量，有自己的指令指针，但与其它协同程序共享全局变量等很多信息。

协程：在任一指定时刻只有一个协同程序在运行，并且这个正在运行的协同程序只在必要时才会被挂起。开辟多个协程开销不大。协程适合对某任务进行分时处理。

缺点：协同程序并非真线程，可能发生堵塞。

线程：同一时间可以同时执行多个线程。开辟多线程开销很大。线程适合多任务同时处理。

## yieid关键字的作用

Unity中通过StartCoroutine开启IEnumerator协程里的yield，在这个协程中执行一些异步操作，比如下载文件，加载文件等，在完成这些操作后，执行我们的回调。

## unity中的UV是什么？

贴图影射到模型表面的依据

U和V分别是图片在显示器水平、垂直方向上的坐标。

UV” 这里是指u,v纹理贴图坐标的简称(它和空间模型的X, Y, Z轴是类似的). 它定义了图片上每个点的位置的信息. 这些点与3D模型是相互联系的, 以决定表面纹理贴图的位置. UV就是将图像上每一个点精确对应到模型物体的表面. 在点与点之间的间隙位置由软件进行图像光滑插值处理. 这就是所谓的UV贴图。