面试技巧

1. **技术方面**
2. **MVC模式。**

ui框架，表现层，交互层，特效层等，通过一个脚本去控制一个模型应处于的位置。

1. **Android与Unity交互。**

使用Unity的SendMessage自己Unity自带的jo.call去调用安卓代码交互的，与IOS则是实现方法调用的。

1. **打包流程 sdk接入。**

接过微信登陆，微信分享，短信验证等第三方SDK，以及高德定位等原生SDK。

1. **断线重连。**

通过心跳包检测，也可以在线程处监听socket是否断开，另外收发消息的时候，以及从消息队取消息的时候也要做异常判断。

1. **内存优化。**

比如UGUI中把一个对象要使用的图集放入一张图集中，这样消耗的draw call比较低。内存消耗包括：1.资源内存占用；2.引擎模块自身内存占用；3.托管堆内存占用。

1. **TCP与UDP区别。**
2. TCP面向连接（如打电话要先拨号建立连接）;UDP是无连接的，即发送数据之前不需要建立连接。
3. TCP提供可靠的服务。也就是说，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达;UDP尽最大努力交付，即不保证可靠交付。
4. TCP面向字节流，实际上是TCP把数据看成一连串无结构的字节流;UDP是面向报文的，UDP没有拥塞控制，因此网络出现拥塞不会使源主机的发送速率降低（对实时应用很有用，如IP电话，实时视频会议等）。
5. 每一条TCP连接只能是点到点的;UDP支持一对一，一对多，多对一和多对多的交互通信。
6. TCP首部开销20字节;UDP的首部开销小，只有8个字节。
7. TCP的逻辑通信信道是全双工的可靠信道，UDP则是不可靠信道。
8. **stringformat造成的消耗，与string的区别。**

StringBuilder与String直接相加的执行效率都比String.format高， 而StringBuilder的执行效率要比String直接相加要高点。

1. **一张图集里用多张图片渲染造成的影响，为什么要用多张图集？**
2. 先把散＝小图打包成图集，然后再把所有散图拷贝在Resources文件夹下，这样运行时就能用Resources.load了。
3. 先把小图打成图集，然后把所有小图关联在prefab上，拷贝在Resources文件夹下，这样运行时也能用Resources.load了。
4. **protocolbuffer为什么会比json快等等？**

我曾遇到要解析40M json的需求。在PC端，使用litjson需要解析10秒钟。但是将相同的内容通过protobuf再导出成bytes。只要17M。缩小了2.5倍左右。但是读取速度只要0.8秒，还包括了数据解析后的处理。

0.1秒和0.008秒可能给人差别不大，但是10秒和0.8秒的差别真的是天壤地别。

1. **一般做什么优化？**

使用协程、对象池等

1. **有没有写过什么插件方便打包？**

将图集打包，然后生成预制体。

[MenuItem ("MyMenu/AtlasMaker")]

static private void MakeAtlas()

{

string spriteDir = Application.dataPath +"/Resources/Sprite";

if(!Directory.Exists(spriteDir)){

Directory.CreateDirectory(spriteDir);

}

DirectoryInfo rootDirInfo = new DirectoryInfo (Application.dataPath +"/Atlas");

foreach (DirectoryInfo dirInfo in rootDirInfo.GetDirectories()) {

foreach (FileInfo pngFile in dirInfo.GetFiles("\*.png",SearchOption.AllDirectories)) {

string allPath = pngFile.FullName;

string assetPath = allPath.Substring(allPath.IndexOf("Assets"));

Sprite sprite = Resources.LoadAssetAtPath<Sprite>(assetPath);

GameObject go = new GameObject(sprite.name);

go.AddComponent<SpriteRenderer>().sprite = sprite;

allPath = spriteDir+"/"+sprite.name+".prefab";

string prefabPath = allPath.Substring(allPath.IndexOf("Assets"));

PrefabUtility.CreatePrefab(prefabPath,go);

GameObject.DestroyImmediate(go);

}

}

}

1. **设计模式等等，在什么地方适合使用此模式？**
2. 迭代器模式：用于协程。
3. 访问模式：比如ExtendMonobehave为访问者基类。
4. 命令模式：写一个模块管理器，如果需要调用另一个模块的函数，则通过发送一个消息给模块管理器来发送信息，如果模块不存在就缓存信息，等有的时候再发送过去。这属于静态解耦，就是编译上解耦，即使另一个模块还没有的时候也能运行。
5. **基础逻辑。**
6. 斐波那契数列的递归写法：

static int Fn(int n)  
        {  
            if (n <= 0)  
            {  
                throw new ArgumentOutOfRangeException();  
            }  
  
            if (n == 1||n==2)  
            {  
                return 1;  
            }  
            return checked(Fn(n - 1) + Fn(n - 2)); // when n>46 memory will  overflow  
        }

1. **公司方面**
   1. 公司是否是培训等不适合公司。
   2. 公司目前的状况，包括公司组成、公司的底蕴、公司人员、团队的稳定性、是否有技术大牛、了解岗位要求、了解自身不足、晋升空间。
   3. 公司的发展前景，上线项目，上线项目可玩性，运营渠道。
2. **薪资方面**
3. 如果是棋牌初创公司什么都没有而且只有一个人的话，聊的可以大概12K。
4. 如果是棋牌公司有一定的规模大概10K。
5. 如果是其他种类的可以说一个大概9K-9.5K，或者9K-10K，可以顺便问问公司的期望薪资。
6. **提问**
7. 这个岗位是新开设的还是原岗位上的人离职了？
8. 关于这个岗位，您觉得最重要的工作内容是什么？
9. 您对这个行业未来的发展怎么看？
10. 加班吗？加班是否有工资？出行方式？房租？福利情况？年终奖、分红？五险一金？五险一金是否全部交？转正几个月？能否提前？薪资能否跟着资历有上升空间？薪资发放是否准时？是否提供补贴之类的？我的上下级？节假日以及作息时间？

