1、自我介绍

评价：自我介绍过长，有几年工作经验，开发个几个项目，最后一个介绍，我主要做了什么功能。

2、热更新是编写的

资源热更

使用AssetBundle或YooAsset打包资源，通过版本号对比服务器资源，下载差异文件

示例：将模型、场景等资源打包为.ab文件，运行时通过AssetBundle.LoadFromFile动态加载

代码热更

ILRuntime：

① 创建热更项目编译为DLL；

② 主工程通过AppDomain.LoadAssembly加载DLL；

③ 使用适配器解决跨工程接口调用

HybridCLR：

① 配置HybridCLRSettings划分AOT和热更程序集；

② 通过Assembly.Load加载补丁文件

初始化与执行

启动时检查版本，从服务器下载更新包。

加载热更DLL/Lua脚本，替换原有逻辑。例如ILRuntime通过appDomain.Invoke("类名", "方法名")调用热更代码

3、差异包是怎么生成的。

版本号比对

客户端启动时向服务器请求最新版本号（如1.1.0.3），与本地版本号（如1.1.0.2）对比。若服务器版本更高，则触发差异包生成流程3。

MD5校验文件对比

客户端本地存储的资源配置表（如Files.txt）记录每个资源文件的路径、MD5值和文件大小34。

服务器端提供最新的资源配置表，通过逐项对比MD5值，筛选出MD5不一致的文件，生成差异文件列表14。

增量打包

根据差异文件列表，将变化的资源或代码模块（如DLL、AssetBundle）单独打包为差异包。例如：

Huatuo方案：仅编译修改后的C#代码生成差异DLL1；

AssetBundle方案：仅重新打包变化的资源（如UI贴图、Lua脚本）34。

4、assetbundle包是怎么进行管理的。

1. 资源加载怎么加载的？

6、怎么异步加载的？

开携程 ，两个参数，一个是资源路径，一个回调函数

携程用异步函数

7、携程底层实现

迭代器

目的是模拟多线程效果，实际上只有线程。

while(true){

dostring()

-- yield wait(0.1)

dostring()

}

8、加载assetbunle有哪些方式？

a、www w.assetbundle

b、assetbundle.loadfromfile

c、loadfrommemory

d、loadfromcacheordown

9、assetbundle压缩方式有那几种，有什么区别？

LZ4,Lzma

10、assetbundle打包策略，怎么打包?

把公共的部分进行单独打包，同时使用的资源可以打在一个包里，这个包尽量不要太大，

a、首先是UI分模块进行打包。比如说背包界面用到的prefab和材质和贴图就可以单独

打在一个包里，比如排行帮用到的prefab和材质和贴图就可以打在一个包里，

b、其次是每个人物模型，材质和动作是打在一个包里

c、所有的声音打在一个包里（不超过10M）

d、每个特效和他所依赖的材质贴图打成一个包。

e、代码可以分功能打包(所有的代码也打在一个包里)（不超过3M）

f、每个场景单独打成一个包。

公用的资源单独打包，防止重复打包

同时使用的包尽量打在一起

把小图集合成大图集打成一个包。(Icon)

11、assetbundle资源有多大，总共有个多少个包？

MMORPG 1个G左右，资源600,700个。

12、一个asssetbundle包的大小大概是多少？

2M左右

13、更新，资源加载顺序

先加载大资源，再加载小资源

14、当资源更新时出现断网怎么处理

实现断点续传

这个类UnityWebRequest

15、如何自身代码更新

更新模块用lua实现，就修改Lua代码，如果更新代码用c#写，用热修复功能（tolua,xlua）

16、xlua热修复底层怎么实现的？

注入-->将带有[hotfix]标签的函数修改为如果lua函数不为空，执行lua函数，否则执行原函数--》xlua.hotfix(fc,flua)将

lua函数赋值。

17、xlua和tolua有什么区别

tolua停止更新,xlua还在更新阶段

xlua在编辑下不需要wrap,在运行时wrap，开发效率比较高

xlua有热修复功能

xlua性能比tolua性能高

18、用c#如何进行热更新

ILrunTime

19、ILRunTime如何进行热更新的

将游戏分为两个部分，Unity和Hotfix。其中主要的游戏逻辑部分也就是可能需要热更的部分都写在Hotfix当中，

然后上传到web服务器，unity启动的时候会下载Dll,并装载DLL

20、ILRunTime需要注意的

a、跨域继承（从dll工程继承Unity工程类 需要再unity注册适配器）

b、Dll工程里使用事件，委托，需要在unity工程注册事件委托，如果事件委托不是action/fun还需要写转换器

c、在dll工程里调用Unity工程类相应函数，最好生产wrap文件（CLR绑定）否则会使用反射降低性能

21、lua代码如何进行打包更新的？

可以将一个文件夹下的所有文件打到一个包里，包名以文件的路径名做为包名，包/改成\_

可以将所有的lua文件打成assetbundle包

22、打包后如何加载lua assetbundle包

程序启动时将所有的ab包加载到内存，通过字典来实现，key为路径，value为文件所对应的内容

都执行文件的时候直接将路径/替换为\_，然后根据替换后的字符串直接从字典取文件内容，然后再把文件内存加载执行。

23、lua再unity中文件后缀名。

不能以.lua为后缀名

24、lua的面向对象是怎么实现的？

25、\_\_index和\_\_newindex执行流程

\_\_index是当访问一个表里的字段，如果这个表里不存在对应的对应的字段，他会首先找到这个表的元表，如果元表存在，就访问这个元表的\_\_index字段，rawgettable

—newindex是当设置一个表里的字段，如果这个表里不存在对应的对应的字段，他会首先找到这个表的元表，如果元表存在，就执行这个元表的\_\_newindex字段，rawsettable

26、什么是lua闭包？什么时候使用闭包

商城初始化Item时，需要给Item里的购买按钮需要传一个函数，这个函数就需要使用外部局部变量

27、lua使用时候尽量使用区别变量不要使用全局变量。

28、lua优化

1、使用lua定义变量的时候尽量使用local，不要使用全局，因为local直接从lua堆栈取值，而全局变量从\_G表把值放入到堆栈，存在内存拷贝。使用全局表安全性比较差，模块耦合性比较高

2、Lua的表分为数组部分和哈希部分，数组部分存储元素值，而哈希部分存储键值。所以哈希占用的内存比数组占用的存储空间高，所以尽量用数组，不要用哈希。

3、使用的表时预先填充，避免rehash

4、使用..的时候，用table来模拟buff,再使用table.concat

5、3R原则，3R原则

减量化（reducing）减少内存占用，最好用数组，不要用hash存，减少table的使用

再利用（reusing）重复使用旧对象，避免创建新对象。

和再循环（recycling）停止垃圾回收后执行程序，程序执行完后，再启动垃圾回收

6、使用luajit（不能IOS）可以提供性能

7、瓶颈部分使用c++而不使用lua

29、tolua优化有哪些

a、尽量不调用对象的成员变量和成员方法，这样会出现频繁的出栈入栈，和类型转换，我们写一个工具类，在工具内写一个静态方法，实现对象方法和成员变量调用。

b、函数参数最好不要用vector3,四元数等类型，最好用float,int 类型。

c、Userdata使用完后要设置为nil否则mono内存泄漏

d、程序启动时将所有的ab包里的文件加载到内存，通过字典来实现，key为路径，value为文件所对应的内容

都执行文件的时候直接将路径/替换为\_，然后根据替换后的字符串直接从字典取文件内容，然后再把文件内存加载执行。

e、编写lua内存检测工具，可以查看垃圾对象和非垃圾对象，如果是非垃圾对象可以查看他引用的c#的对象。

30、tolua原理

首先将类导到lua表里，c#对象就对应了lua 的userdata ，userdata存的值实际上就是ObjectTranslate返回的索引值。

userdata将lua表作为元表。这样当通过userdata调用方法时，就会去找元表对应的方法，并把userdata传过去,根据

userdata值再ObjectTranslate里取到对应的对象，然后调用对象对应的方法。

31、接android sdk用的那个版本？

6.0（API level 23）

32、接过哪些平台？

9游，360，华为，小米，应用宝，豌豆荚,taptap,OPPO,VIVO,百度

33、unity和android 是怎么交互的？

unity调用andorid

通过new一个com.unity3d.player.UnityPlayer的AndroidJavaClass，然后通过androidjavaClass得到currentActivity（当前界面）的AndroidJavaObject这种类型的

变量，然后可以通过call方法调用android中定义的方法

android调用unity 通过 UnityPlayer.UnitySendMessage方法调用unity对象的方法

34、说下android的开发工具

android studio

eclipse

35、接sdk是怎么接的？

第一种方式：

是直接从unity项目导出android项目，然后通过android studio 打开android项目，然后是将平台sdk包导入到android项目，接下来是初始化sdk,登录，支付,注销，退出。主要是调用

平台sdk里的初始化，登录，支付等接口，并注册初始化，登录，支付，,注销，退出的回调函数。接完后再打包成apk文件。函数调用主要参考平台接sdk的文档

第二种方式：

将平台SDK的jar包放到unity 里的Plugins,然后再unity里调用jar包里的初始化，登录，支付，,注销，退出等相关API,并注册初始化，登录，支付，,注销，退出的回调函数，然后再unity中直接打包成

apk. 函数调用主要参考平台接sdk的文档

36、接sdk一种框架

先出一个母包，然后通过工具接对应平台sdk.

然后封装jar包，把jar包里的函数名进行统一。unity调用统一后的文件名。

先出一个母包，打包成一个母包的apk,然后通过工具将具体平台sdk对应的jar包和配置文件文件放到母包解压后对应的文件夹下，然后将文件夹压缩成apk。

37、ios包出过没有？

接过，主要是将项目导成xcode项目。然后通过xcode接入登录和支付，然后编译成ipa

38、xcode使用的语言是什么语言

object-c 和c语言比较类似。

39、在开发项目中遇到过什么问题？怎么解决的？

a、我在开发xxx项目的时候，系统比较比较复杂，策划案都差不多20页。刚开始就是不知如何去分析。

主要是先抓重点，看系统有多少个界面，从界面效果图入手，分析效果图，然后再结合策划文档来整理需求，不明白的和策划去沟通。沟通好之后，

就针对功能的重要等级进行排序，先做重要的功能，再做其他稍微不重要的功能。

b、我在开发xxx项目的时候写网络框架的时候，需要进行多线程编程，多线程对共享数据需要进行同步，并且在网络编程时还需要进行粘包拆包的处理，如果处理过程中，

少一个字节就会出现问题。

主要是采用分而治之的方式，将复杂的功能细分成多个小功能，然后看小功能是否比较复杂，如果比较复杂，再将小功能进行细分，直到小功能比较容易解决为止。

这样如果把所有的小功能都完成了，实际上就完成了这个复杂的功能。再做小功能的时候，可以写测试用例。

c、帧同步框架，浮动点的问题。

用两个整数，一个表示整数部分，一个表示小数部分。

d、刚开始接sdk.遇到各种报错，不知怎么解决。

刚开始用baidu查询问题，定位不准，一般用google引擎输入英文单词很快就可以找到解决问题的办法。

e、我在开发xxxx游戏中进行优化时发现mono内存泄漏，mono内存一直增加。

这种问题一般通过profiler和framedebug，不好查出问题的原因，我们最后写了一个内存检测工具。内存检测工具主要是统计和查询游戏运行过程中分别的对象，并可以导出，就可以

根据导出文件找到哪些对象没有被释放。然后分析没有被释放的原因。

40、puremvc

puremvc是对MVC的进一步封装，通过代理封装M层，通过中介封装V,命令层就是C程，mvc可以通过外观进行相互

访问。可以通过工具直接将UI（prefab）生成对应的UI类,这样可以提高开发效率。

41、tcp与udp的区别，http,https

1、tcp是可靠的，不能丢包，udp不可靠

2、tcp有顺序的，udp无顺序的

3、tcp没有udp快

4、udp可以实现快速重传

42、网络丢包怎么处理

43、UDP怎么防止丢包

模拟tcp（kcp协议）

发冗余包

将消息编号

43、发送数据包的大小

不要超过mtu（最大传输单元）

44、socket底层

socket底层主要是通过发送和接收缓存，调用send函数的时候从，自定义缓存拷贝到socket的发送缓存

调用recv函数的时候，将socket的接收缓存的数据拷贝到自定义缓存。

45、网络框架怎么搭建的

网络框架是基于tcp的，主要处理消息的发送，接收，和断线重连，主要采用TCP协议，发送消息时将要发送的消息放入发送消息队列里，

主线程再update里从消息队列里取出消息进行异步发送，当异步发送函数进行回调时，再从消息队列里取出

消息再进行发送。

接收数据时也采用了异步，接收到数据后进行拆分包，然后把消息放到队列里，再主线程里从队列取出消息，然后

通过消息id找到对应的消息处理函数进行调用。这里需要注意线程同步（因为主线程和接收线程公用了同一个缓冲区）

断线重连：客户端每隔5秒钟向服务器发心跳包，正常情况下服务器会返回心跳包，当服务器再10秒内没有返回心跳包，将会进行重连，并重连三次，如果三次未重连上，49

直接回到登录界面，如果连接上，重新走登录流程（将登录流程用到的消息和服务器再走一遍）这样可以获取最新数据（通过会话登录，然后把没有发完的数据通过新的socket发送）

46、状态同步原理

客户端表现效果（播放动作，声音和特效），服务器进行运算，举个例子火球释放 首先是客户端进行条件判断（玩家自身状态判断，有些减益buff(炫晕，冰冻)，技能CD时间）

--》客服端发送开始释放技能消息（发送技能ID)--》服务器收到释放技能消息--》服务器进行验证--》给客户端返回消息(可以释放技能，客户端播放动作和特效)

==》服务器创建火球，并开始移动==》通知客户端（客户端创建火球，并且开始根据速度移动）==》服务器进行碰撞检测，产生伤害==》通知客户端

47、帧同步原理

帧同步是基于相同的输入 + 相同的处理函数 ==相同的输出,客户端进行逻辑运算和表现效果，客户端每个30毫秒向服务器转发操作，服务器每个60毫秒进行转发消息到所有的客户端操作

客户端收到转发消息后首先进行逻辑运算，将运算的结果放入渲染队列，客户端通过渲染队列进行客户端表现效果。

48、状态同步，帧同步网络延迟

49、设计模式（PureMVC）

单例，puremvc里用到的设计模式代理，中介，命令，外观，观察者（消息中心），工厂模式（简单工厂，抽象工厂，工厂方法）

https://www.cnblogs.com/heliusKing/p/11577837.html

50、lua基本数据类型

number,nil,string,bool,function,userdata,table,最常用的是table

51、string,stringbuilder的区别

stringbuilder再拼接时不会产生临时字符串

52、protobuf,xml,json的区别

protobuf可以跨语言c++,lua,java

protobuf序列化和反序列化效率高

protobuf对数据进行了压缩

xml,json用于网页比较好，但数据冗余

protobuf c#性能有问题 优化可以无gc的protobuf

52、用于那些打包平台

pc,android,iOS

53、常用的数据结构

链表，队列，数组，二叉树

54、ecs

基于数据编程，和unity基于组件开发差不多，e表示entity-->GameObject,c(Data)----->组件，S(系统）

55、常用的插件有哪些?

SimpleLOD，EasyTouch,NGUI,DoTween,behaviour design

56、基于四叉树的大地图

57、行为树

https://blog.csdn.net/linxinfa/article/details/72937709

58、欧拉角，四元数的优缺点

59、UI适配

UGUI是调整canvas scaler的属性进行适配。model改成scale with screen size.

NGUI 是屏幕实际分辨率和标准分辨率的宽高比进行对比来矫正相机的正交尺寸。如果实际的分辨率比标准分辨率的

宽高比大，出现左右黑边，如果小的话，会出现上下黑边。

60、高速移动物体碰撞检测失败

连接运动物体上一帧和当前帧的位置形成一个向量，然后做碰撞检测。

61、 meshcollider与其他collider的不同

两种碰撞器各有利弊：MeshCollider的精度非常高，但是计算量大。

BoxCollider的 精度低，但是计算量少

62、2018新的动画系统

2D Animation2D

63、c#委托，事件 的区别

委托和事件绑定都不受限制，但是事件在调用时只能在本类进行调用。

65、u3d物理部分

66、您有什么想问的？

什么类型的项目，项目进度。

可以问下你没回答对的问题。

67、什么是反射

在程序运行时，动态获取和加载程序集，并且可以获取到程序集的信息

在程序集中，包含模块（Module），模块包含类型，类型包含属性，方法和，字段，我们可以查看到一个程序集的路径，命名空间，类的属性，方法和字段。我们还可以对其进行操作。

举例

using System.Reflection;

namespace ReflectionDllDemo

{

class Program

{

static void Main(string[] args)

{

var asm = Assembly.LoadFile(@"d:\3\_code\DotNet\DllDemo\DllDemo\bin\Debug\DllDemo.dll");

var type = asm.GetType("DllDemo.ClassGreenerycn");

var instance = asm.CreateInstance("DllDemo.ClassGreenerycn");

type.GetProperty("Name").SetValue(instance, "http://greenerycn.cnblogs.com", null);

type.GetProperty("IsTest").SetValue(instance, true, null);

var method = type.GetMethod("Hello");

method.Invoke(instance, null);

}

}

}

68、action、func 有什么区别

69、优先使用组合而不使用继承

高内聚，低耦合

70、接口和抽象类的异同是什么？

1、都不能被实例化。

2、接口的实现类和抽象类的子类只有全部实现了接口或者抽象类中的方法后才可以被实例化。

不同点：

1、接口只能定义抽象方法不能实现方法，抽象类既可以定义抽象方法，也可以实现方法。

2、单继承，多实现。接口可以实现多个，只能继承一个抽象类。

3、接口强调的是功能，抽象类强调的是所属关系。

4、接口中的所有成员变量 为public static final， 静态不可修改，当然必须初始化。接口中的所有方法都是public abstract 公开抽象的。而且不能有构造方法。抽象类就比较自由了，和普通的类差不多，可以有抽象方法也可以没有，可以有正常的方法，也可以没有。

71、 为何要创造接口和抽象类吗

https://zhuanlan.zhihu.com/p/78957036

接口和抽象类之间的区别在于他们的目的不同，抽象类是为了提供继承，而接口是为了提供一种规范，目的是为了降低耦合，接口比抽象类抽象程度更高。

72、什么时候用struct,什么时候用class

sturct主要是表示数据，用来抽象，继承继承，否则只表示数据就用struct

73、ref与out有什么区别

74、程序设计的原则

单一原则，开闭原则，里氏替换原则，依赖倒转，迪米特法则,接口隔离

75、使用过哪些设计模式？

单例，简单工厂，抽象工厂，PureMvc（外观，代理，中介，命令，观察者模式）

76、简单工厂，抽象工厂，工厂方法

简单工厂就是根据类型生成对应的对象，抽象工厂是一个类一个工厂类，这些工厂继承一个抽象基类。

https://blog.csdn.net/jerry11112/article/details/80618420

77、设计模式有多少种？

23种

78、观察者模式？

观察者模式在游戏中的消息中心使用，它通过委托，事件来绑定消息和函数，通过派发消息就可以调用所有绑定某消息的具体函数。

79、抽象工厂，和工厂方法有什么不一样？

抽象工厂是一个工厂可以生产多个类的对象，工厂方法一个工厂只能生产一个类。

80、点乘与叉乘有什么区别

点乘的结果是一个标量（数值），叉乘的结果是一个向量

可以通过点乘的结果求两个向量的夹角。

dot(a,b) = |A|\*|B|\*cos0 0表示向量A与B夹角

|cross(a,b)| = |A|\*|B|\*sin0

81、欧拉角与四元数，旋转矩阵的区别

都可以表示方向，但是欧拉角造成万向节死锁

82、常用数据结构

队列，栈，链表，哈希表、树、图

83、常用的排序算法

冒泡，插入排序，希尔，堆排序，快速排序，选择排序，归并排序,基数排序，桶排序

84、快速排序算法时间复杂度，最坏情况什么时候出现？

选择基数时选择了最大值和最小值的情况，会出现最坏情况。

85、堆排序算法

时间复杂度o(nlgn) 空间复杂度

86、c#List.sort() 底层是怎么实现的？

当元素比较少时用快速排序，元素多的时候用堆排序。

87、c#里的array ,arraylist,Lists<>,LinkedList有什么区别？

88、字典dictionary的底层实现

一个数组和一个单链表，这个数组是桶，链表是存放数据的，主要是解决哈希冲突的，底层用的用数组实现。

89、字典dictionary如何解决碰撞的？

拉链法

90、字典dictionary怎么存储一个key,value的

根据hash算法求key的hash值，然后根据hash%桶的大小 求桶的编号，然后找一个空闲的元素存入k,v值，并将该元素插入到链表中。

91、从字典dictionary中查找

根据hash算法求key的hash值，然后根据hash%桶的大小 求桶的编号，然后再从单链表中查找key值一样的。

92、使用字典dictionary需要主要什么？

开辟一个足够大的空间

93、rehash的时候会产生什么问题

原来的数组成为了垃圾

原来的元素需要重新计算的hash值并插入到新的链表

94、链表与数组的优缺点

数组是插入元素慢，查询快，链表是查询慢，插入或者删除比较快。

95、两个有序数组排成一个有序数组。

双指针，然后对比两个数组元素大小，将小的放到新数组并进行指针+1，直到遍历完数组的元素，如果某个数组还有剩余元素，将剩余元素放到新的数组。

96、二叉树的遍历方式有哪些？

深度优先（先序，中序，后序）,广度优先（层次遍历）

97、二叉树递归与非递归

https://blog.csdn.net/u013728021/article/details/84067019

98、ArrayList与list<T>有什么区别

ArrayList元素类型是object，就存在拆装箱的问题，引用类型存在堆里，就有可能引起GC

99、怎么打assetbundle包的？

把公共的部分进行单独打包，同时使用的资源可以打在一个包里，这个包尽量不要太大，

a、首先是UI分模块进行打包。比如说背包界面用到的prefab和材质和贴图就可以单独

打在一个包里，比如排行帮用到的prefab和材质和贴图就可以打在一个包里，

b、其次是每个人物模型，材质和动作是打在一个包里

c、所有的声音打在一个包里（不超过10M）

d、每个特效和他所依赖的材质贴图打成一个包。

e、代码可以分功能打包(所有的代码也打在一个包里)（不超过3M）

f、每个场景单独打成一个包。

100、如何更新assetbundle包的？

首先是生成差异包（assetbundle,文本文件(需要更新的文件)），然后将差异包上传服务器，程序启动时客户端需要计算出差异的小版本，

然后依次更新差异版本，把差异版本放到P目录。如果差异版本比较多的，直接更新最新版本。如果更新过程中出现断网的时候，启动断线重连。

(HttpWebRequest).

101、地图编辑器是干啥的？

地图编辑器主要是给服务器导场景配置数据用的，比如场景碰撞信息,把客户端的 NavMesh数据导出给服务器使用，navmesh提供了NavMesh.CalculateTriangulation接口，这个接口可以获取navmesh的数据，

顶点和索引。除了场景碰撞信息外，还有NPC的信息（位置信息）如果是商店NPC，可以配置商店NPC可售卖的物品。如果是战斗NPC可以配置掉落ID，模型，生命值，攻击值，

等等。还可以配置其他的可破坏物品，等等。

102、地图编辑器如何实现的？

首先是一个表对应一个数据类，这些类是通过代码自动生成，数据类里定义了数据表里所有列对应的变量和一个函数GetLineCsv。然后定义个ConfigManger类对所有的表进行

读取和存储管理，然后再这个类里定义了一个字典对特定的表进行管理，key就是行号，value就是该行数据， 读取的时候是从一个大字典取数据，大字典的实现是这样实现的，key表示表的名字，value就是这个表的数据，value也是通过字典来实现的，key表示行号，value表示这一行的数据，其中value通过字典来实现的，其中key表示列，

value表示列所对应的数据。存储的时候将表里的数据存储到对应的文件里。

103、代码怎么自动生成的？

就是指定表所在的路径，列举出该路径里所有的表，然后对每个表生成对应的类，主要是根据这个表里的字段，生成对应类的字段。

104、表是什么格式？

csv

105、技能编辑是如何实现的？

MMO如何实现?

首先根据策划需要将技能分为普通技能，吟唱技能，通道技能，再编辑器编辑技能的时候需要设置技能类型，然后不同的技能分为不同阶段，普通技能分为技能释放，释放结束。吟唱分为开始吟唱，吟唱中，吟唱结束，吟唱攻击阶段，技能结束，通道节能分为释放技能，持续释放，释放结束。

然后针对不同的几能阶段创建时间线，然后再时间线上传建不同的效果（播放动作，特效，声音，屏幕震动效果），配置完成后保存为csv格式。然后服务器发对应消息后，根据技能ID找到配置表读取对应的字段进行播放。

回合制：

通过创建时间线，然后再时间线上添加不同的效果（播放动作，特效，声音，屏幕震动效果）

，

106、udp与tcp的区别

tcp是基于连接的（（3次握手），断开（4次挥手））,udp是基于无连接的

tcp可靠，udp不可靠的(不会丢数据)

tcp接收数据是有序的，udp是无序的

tcp在弱网环境下不能实现快速重传，udp能实现快速重传(KCP)

107、帧 同步框架。

客户端每33毫秒将当前帧的所有操作发给服务器，服务器每66毫秒转发所有客户端操作，客户端收到所有操作后，

进行逻辑计算，将逻辑运算结果放到渲染队列里，客户端根据渲染队列再进行渲染。

108、帧同步断线重连。（kcp也有连接的）

每隔5秒钟向服务器发送心跳包，服务器收到心跳包后会返回信息，如果没有返回信息，将隔10秒钟进行断线重连，

总共3次，如果3次不成功，提示网络异常，点击确定回到登录界面。

如果连接上服务器后

把登录消息向服务器再发送一遍，从而获得最新数据。

把原来没有发送过的数据再发送一遍。--》 需要快速播放（折半）

109、人物在帧同步下怎么实现？

向服务器发送移动操作（type=1,x,y轴值）-->服务器转发--》客户端计算(位置）--》渲染队列-->设置位置。

110、帧同步出现网络延迟如何处理？

需要快速播放（折半）

111、状态同步框架

112、状态同步断线重连

113、状态同步人物移动如何实现。

摇杆移动的时候每0.1秒向服务器发移动消息（移动的方向，位置，时间戳）并进行移动，服务器收到消息后，服务器进行验证

位置，并进行模拟移动。如果验证通过就进行转发，否则向客户端发送位置矫正消息，客户端收到矫正消息后会

瞬移到矫正位置。（客户端先行）（客户端后行模拟帧同步）

114、状态同步网络延迟如何处理？

115、战斗框架主要是实现了什么东西

帧同步框架，客户端动作播放，客户端特效播放。客户端进行碰撞检测，伤害计算。（AABB算法）

116、资源管理器是怎么实现的？

资源管理实现了对资源的加载与卸载管理，首先是从assetbundle包加载资源,加载完资源后,需要卸载assetbundle

,然后对加载出来的资源进行引用计算管理，当引用计算为0的时候把加载出来的资源也释放掉。

资源管理器还提供了同步和异步的加载方式

举例：

void LoadAsset(string filename,assetName,Type type){

assetbundle ab = assetbundle.loadfromfile(filename)

Type tex= ab.LoadAsset<Type>(assetName)

ab.unload(false)

return tex

}

public void LoadPrefab(string res, LuaFunction func)

{

StartCoroutine(AsyncLoadPrefab<GameObject>(res, func));

}

IEnumerator AsyncLoadPrefab<T>(string res, LuaFunction func) where T : UObject

{

ResourceRequest rr = Resources.LoadAsync<T>(res);

yield return rr;

if(rr.isDone)

{

func.Call(rr.asset);

func.Dispose();

func = null;

}

}

117、mvc框架是怎么搭建的？

m表示数据,v表示视图，c表示控制，M和V不打交道，是通过C进行交互。MVC都可以是单例，V可以包含多个UI

当客户端收到消息后，通过消息ID，找到对应的消息函数，消息函数首先修改M

然后通过消息中心通知控制层C，C收到消息后，再通知V刷新界面。

118、texturepacker?

美术出散图，程序开发期间使用散图，打包时将使用工具合并散图，打包成一个图集，并将UI里相应的Sprite修改为打包图集上的Sprite.

工具的功能--》1、调用texturePacker.exe将小图片生成一张图集，并且生成一个xml.2、根据xml来生成图片的sprite ,3，删除xml-->将UI里的sprite替换为图集合并后后的sprite

119、AStar算法大概思路

Astar算法主要是通过广度优先算法，然后根据F= G + H这个公式从优先队列求下一步最优解。G表示起点到当前节点的距离，H表示当前节点到终点的距离估计值。H有两种算法曼哈顿算法和

欧几里得算法。

120、Astar具体实现：

通过open列表和close列表来实现，首先将起点放到open列表里，然后判断Open是否为空，如果为空表示搜索失败，如果终点在open列表里，表示搜寻完毕。否则从

open里取F的最小值节点将他放到close列表，并将 此节点的邻近节点放到Open列表，并设置其父节点。如此循环，直到终点在Open列表里结束循环，然后将找出从终点节点

的父节点，再找到此节点的父节点，直到起点，然后倒序排序。

121、Astar寻路优化

减少格子数，比如说原来是200\*200，减少为100\*100

用WayPoint 可以格式数

用navmesh(unity）

122 、Open和close列表就是优先队列（二叉堆）

122、迪杰斯特拉算法（穷举算法）

123、离职原因

最后一个项目离职原因

项目上线数据不好，再加上舆情，项目组解散。然后公司把我掉去其他项目组，去这个项目组之后，这个项目也接近尾声，活不多。

前几个项目离职原因：

想有提升空间。

124、项目状态：

项目已经上线，主要是后期做一些维护工作。

125、客户端有多少人

8个人（游戏逻辑5，渲染优化2，战斗系统1（主程））

126、服务器4个人，c++，Java语言。数据库:mysql

127、策划有多少人？5个人 系统策划（3） 数值策划（1），剧情策划（1）

128、美术有多少人？15人（公用）

129、项目组总共有多少人？32

130、客户端游戏逻辑系统 ：登录，创建角色，人物移动，主界面，背包，任务，新手引导，红点系统，装备系统，充值系统，签到，好友聊天系统，工会系统，商城，邮件，

组队。。。。。

131、在项目中做得比较好的是哪些部分？

热更新 打包（差异包） 加载时（将脚本文件加载内存 ）

网络框架 双缓冲。（发消息，介绍消息，断线重连）

基于mvc的UI框架 （首先收消息-->刷新界面）

优化方面（C#内存检测工具，lua内存检测）

TexturePacker扩展（提高开发消息）

地图编辑器

技能编辑器

红点系统，新手引导系统

人物移动

状态同步

帧同步框架

无限滑动列表

图文混排

HUD（不使用UGUI,使用OpenGL底层）

Astar寻路

132、比较有大的进步有哪些？

优化

编码的能力有提升（不加注释，没有规范）

知识面的提升（以前只会UI功能，现在会assetbundle打包，编辑器扩展，shader的编写）

编写框架能力的提升（网络框架，UI框架，战斗框架）

解决问题的能力

沟通能力的能力（策划和服务器）

对MMorgp有了更深的理解。

133、找工作更关注那些因素？

个人发展空间（职位，能力得到提升）

公司的文化（可以看出公司怎么样）

项目团队（制作人比较出名（有成功项目经验））

公司福利待遇

134、那些技术提升？

ecs这方面的项目没有接触过，想接触下ecs方面的项目，现在只是了接

shader方面会写一些效果，也想深入研究下。

135、业余时间怎么提升自己的能力？

看论坛，看知乎，csdn，uwa 大拿的技术文章。

github找一些项目看看,平时自己写一些工具上传到github（断点续传工具，资源打包工具,文件查重工具）。

关注unity官方网站关注最新的技术。

136、目前关注那些工作机会？

刚开始找，一聊就聊上你们这家公司的。

137、目前关注那些公司的工作机会？

刚开始找，一聊就聊上你们这家公司的。

139、上一家薪资情况

结合公司招聘简历上标准的薪酬。只说税前

140、为什么离职

最后一个项目离职原因

项目上线数据不好，再加上舆情，项目组解散。然后公司把我掉去其他项目组，去这个项目组之后，这个项目也接近尾声，活不多。

前几个项目离职原因：

想有提升空间。

141、有什么想要了解的？（面试快结束了，没有答好的问题再请教下）

项目类型 ：

项目进度：

142、UI底层

https://www.cnblogs.com/chenggg/p/11184784.html

首先是UGUI的控件渲染是由四边形（2个三角形）组成的mesh来实现的，顶点信息包含（位置，UV坐标，顶点颜色），当

引起顶点位置变化的属性被改变时放到重建布局队列里，否则放到重建渲染元素队列里。然后下次Unity将会调用SendWillRenderCanvases

这个方法，SendWillRenderCanvases方法会调用PerformUpdate方法，PerformUpdate方法会调用重建元素的rebuild函数，rebuild函数会判断是否更新了几何信息和渲染信息

如果更新了几何信息就会调用方法UpdateGeometry,UpdateGeometry里会调用OnpopulateMesh进行顶点数数据填充，否则调用UpdateMaterial，然后是进行drawcall合并，首先是深度排序，

然后是材质排序，贴图排序，最后进行drawcall合并，合并后提交给gpu进行渲染。

143、LOD的资源打包是怎么做的？

可以将lod公用的材质，贴图打成一个包,lod再分别打包。

144、制作的过程中，制作一个预设，这个预设怎么和你的Lua关联，怎么去写业务逻辑？

1、一个prefab上挂上luabehaviour，luabehaviour指定lua文件。

2、一个lua文件对应一个prefab,然后lua的初始化函数里和prefab进行绑定。

145、UGUI有一些组件，你们自己有写自己的组件吗？

写过，图文混排控件，他继承text类，通过这个控件可以将文字与图片，表情，按钮，超链接进行混排。

实现主要是通过继承text类，然后通过定义表签，1表示图片，2表示表情，3表示按钮，4表示超链接，当给控件赋值时

解析文本里的表情，通过quad对标签进行占位，然后将占位后的字符串赋值给基类的text属性。赋值时，将会引起text进行

重构，重构会调用onPupluateMesh这个函数，然后可以再这个函数里设置标签的位置，然后再根据位置创建具体的控件，比如

标签类型为1就再具体的位置创建图片。

146、UGUI按钮事件源码

首先是从摄像机和鼠标点击位置发一个射线和UI控件做碰撞检测，然后从近到远排序，检测控件是否设置了碰撞，如果是

就得到了碰撞检测到的控件。接下来从检测控件上找到所有实现了EventTrigger接口的方法，然后调用方法实现。

147、NGUI为什么比UGUI性能差？

UGUI底层drawcall合并用c++写的,NGUI使用c#写的

NGUI每个元素每帧都更新，UGUI只有在变化时才更新。

148、使用Unity什么版本开发有限？

5.6.0

149、lua是什么版本？

5.1,5.3

150、内存检测工具

在游戏运行时，可以统计当前使用的资源比如说texture,materail,mesh,文本，以及堆内存统计，并且将使用的这些对象名字导出到文本中。

151、你平时看什么书？

c#本质论，c#高级编程，游戏编程精粹，《GOF设计模式》，《数学之美》、《浪潮之巅》

152、什么优先队列

153、堆排序

154、快速排序

155、List.Sort如何排序

156、为什么使用对排序。

157、客户端如何分配工作。

主程分工--》客户端，服务器，策划讨论--》客户端，服务器对协议--》做功能--》联调。

158、.net cg,unity cg, lua cg

159、什么判断一个对象是垃圾。

160、介绍下你的项目

项目名称，类型，你自己做的功能，

161、Sprite Packer打包图集有什么问题？

散图的加载问题

图集的拼接算法问题

162、Sprite Atlas有什么问题？

Unity编辑器状态下运行UI全部白图

163、Dot(A,B) = |A|.B|.Cos<A,B> A的模乘以B的模再乘以他们夹角的cos值

几何意见：

1、可以求<A,B>夹角

2、可以求成A向量在B向量的投影

3、可以在游戏里判断怪物在人的前方或者后方

4、表示光照强度

164、叉乘 CrossA,B= |A|\*|B|\*Sin<A,B>

几何意见：

1、计算出两个向量的夹角。

2、用来判断一个向量在另一个向量的左侧还是右侧，通过叉乘后向量的方向判断向量的位置

165、欧拉角和四元数(一个向量+一个旋转角度)有什么区别

vector3 quaterion

166、什么是矩阵，倒置矩阵

大学（线性代数）

167、矩阵MVP

M是将局部坐标-》世界坐标

V 是将世界坐标--》相机坐标

P是将相机坐标--》屏幕坐标

168、unity GC

采用标记清除整理算法，首先采用内存池的方式，内存只增加不减少。首先从内存池中找到空闲的空间，如果空间不够，进行

垃圾回收整理，再分配空间，如果空间不够，就进行新的内存分配。

169、.net gc

采用三代算法。

170、lua gc

标记清除（5.1之前采用黑白清除，5.1之后是黑白灰）

171、怎么判断一个对象是垃圾

可达性分析（从根节点(root)（堆栈，\_G表，注册表）开始遍历）

不可达

172、UGUI什么情况下相邻并且使用同一个材质和贴图不能进行合批

1、mask,rectmask

2、元素重叠并且有透明效果。

173、UI特效

1、使用粒子系统，然后将canvas调整为相机模式，再设置render的sortlayer属性可以调整层级关系。

2、直接用UGUI image控件，通过animation生成动画模拟特效。层级比较好控制。

3、使用UI粒子控件（Unity Shuriken）

https://blog.csdn.net/wwlcsdn000/article/details/81319137

174、UI icon合并图集

t1\_icon1

将背包Icon合并成一张1024\*1024大小的图集，如果有剩余，将剩余的图标再进行打包成另外一个图集。。。直到把所有的ICon都打到图集里。

175、materail 与sharedmaterial

这两个属性用法是一样的，但是从效率上来说最好用sharedMaterial，它是共享材质，无论如何操作材质的属性（如更换颜色或者更换shader），内存中只会占用一份。

但是如果要是用material的话，每次更换属性的时候Unity就会自动new一份新的material作用于它。

它 直到Application.LoadLevel() 或者Resources.UnloadUnusedAssets();的时候才会施放内存。

所以material就有可能会造成内存泄漏，那么我们干脆就不要使用它。

但是在代码中如果直接用render.sharedMaterial的话，你会发现在编辑器开发模式下，运行一会儿游戏本地的.material文件凡是修改了的都变化了，如果这些文件都在svn管理中，那么他们都会变成红叹号，表示文件已经被修改。这样太危险了，一不小心上传了怎么办。 为了解决这个问题，可以用一个简单的方法，每次获取material的时候根据平台而定。

public static Material GetMaterial(Renderer render)

{

#if UNITY\_EDITOR

return render.material;

#else

return render.sharedMaterial;

#endif

}

176、AB包运行模式和非AB包运行模式。

AB包运行模式一般是游戏正式运行时采用的模式，而非AB包模式是开发期间采用的模式，如果开发期间采用AB包模式的话

效率极低。所有开发期间采用非AB包模式。

177、许多玩家同时攻击boss,或者屏幕里出现多个怪。

分帧处理，将接收的消息缓存在本地，然后，每一帧处理一定数量的消息，而不是收到消息后一次性处理完所有消息。

178、什么是弱表（weak）

弱表就是弱引用

https://www.cnblogs.com/sifenkesi/p/3850760.html

178、异或操作符

相同为0不相同为1

179、设置GameObject的激活与隐藏

1.通过Active隐藏子物体在Unity3d 版本4.0之前是可以使用SetActiveRecursively方法的

2、把物体移动到视野范围之外。

3、修改Layer层

4、meshrender设置为false

180、删除GameObject的方法

Destroy方法

DestroyImmediate方法

DontDestroyOnLoad方法

对象池不真正销毁，而放在对象池里

181、创建游戏对象的三种方法

1.new GameObject()

2.GameObject.Instance

3.GameObject.CreatePrimitative

182、删除脚本的方法

removeComponent

183、脚本的生命周期函数

awake onenable,start,update lateupate

184、查找物体的方法

GameObject.Find

Transform.Find

GameObject.FindWithTag

GameObject.FindGameObjectsWithTag

Resources.FindObjectsOfTypeAll

https://blog.csdn.net/s1314\_JHC/article/details/80811834

185、GC的产生原因和消除的办法

GC产生的原因是内存空间不够时

消除方法：

foreach(5.6)

使用stringbuilder

list,dictionary初始化足够长度

对象池

186、降低drawcall

A、打包图集

C、UI避免使用图集交叉

E、材质合并工具(就是将两个同shader，贴图不同的material将他们的贴图进行合并，并生成新的材质球使用新的贴图，需要注意改变原理mesh uv坐标需要进行修改)要注意合并后贴图大小

B、Unity静态合批和动态合批（少于300个面，900个顶点），不能使用缩放

G、使用 烘培（Baking）不使用实时光照， 可以减少drawcall

D、SimpleLOD工具

F 、头顶血条优化，将血条和文字在一个shader里处理，不用UGUI的图片和文字，自己写了一套组件用来处理图片和文字，避免文字和图片改变时进行合批。

187、UGUI 文本的背景随着文本的内容来改变背景的大小

188、meta文件

189、sprite 与image

https://blog.csdn.net/a435931517/article/details/84564155

sprite通过增加顶点裁剪掉透明部分，所以sprite比image顶点数多。但是image有透明部分，所以会产生overdraw

image在canvas下就能显示

190、就是有一个图素更改了 我怎么做到不动预设体去改变预设体里面的图片？

可以使用unity2018.3这个版本就是prefab有新功能,prefab可以嵌套prefab,当小的prefab修改后，大的prefab自动修改。

可以利用这个功能，将图片放在小的prefab里，然后再把他放到大的prefab里。

191、2018 prefab新功能

1、prefab可以嵌套

2、可以使用prefab变体（派生类）

192、lua如何调用update函数

1、可以通过事件UpdateBeate：Add添加Update函数

2、编写一个cs脚本叫luabehaviour，通过luabehaviour里的update函数调用lua函数。

193、客户端美术规范

1、图片大小格式png（透明）,tga,jpg(不透明)

2、图片不要太大（1024\*1024）

3、做特效的时候粒子数，贴图数都有限制

4、做粒子特效有LOD

4、人数骨骼数20-30

5、人物面数（3000左右--9000），npc(2000左右）

6、UI背景图片使用九宫格

194、怎么判断一个点在三角形内

通过叉乘和点乘来实现，首先计算三个向量，然后求三个向量两两的叉乘，然后根据叉乘结果求点乘，如果点乘结果都大于0

表示在内部。

195、为什么Unity是单线程的