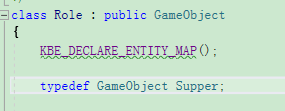
客户端服务器通信方式完善

1、KBE提供的通信方式：

① 创建一个entity类，并继承entity基类。头文件entity类定义KBE\_DECLARE\_ENTITY\_MAP ()宏



在entity的实现文件中定义以下宏

KBE\_BEGIN\_ENTITY\_METHOD\_MAP (Role, Supper)

DECLARE\_REMOTE\_METHOD (方法名，方法地址，参数)

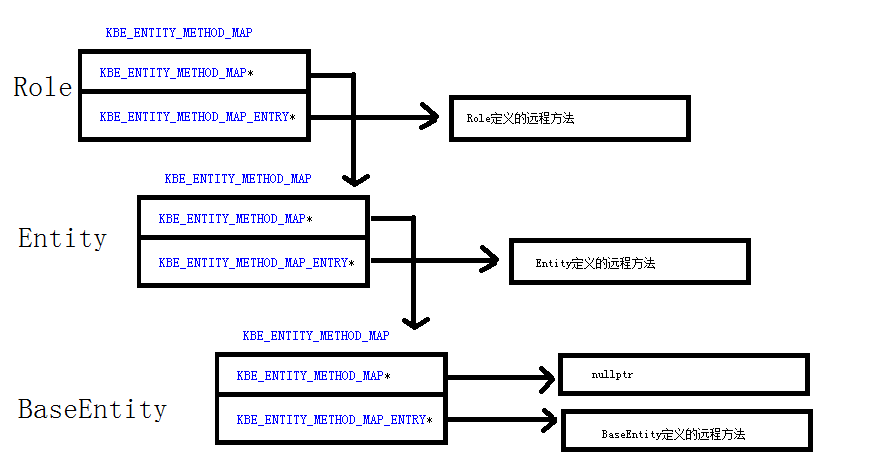
KBE\_END\_ENTITY\_METHOD\_MAP ()

KBE\_BEGIN\_ENTITY\_PROPERTY\_MAP (Role, Supper)

KBE\_END\_ENTITY\_PROPERTY\_MAP ()



以上宏的作用，重写父类的GetThisMethodMap方法。该方法获取一个KBEngine::KBE\_ENTITY\_METHOD\_MAP的结构体，结构体参数有KBE\_ENTITY\_METHOD\_MAP结构体指针（指向父类）、KBE\_ENTITY\_METHOD\_MAP\_ENTER结构体的指针（指向自己）。



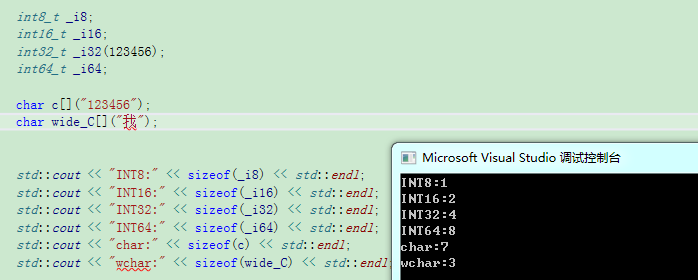
由于远程方法设计成链式结构，如果使用kBE插件方法通信，就会导致玩家RoleEntity “链式”继承。如下方式：Role->RoleSkillEntity->RoleQuestEntity->Entity，中间层使用了继承的方式，子类与父类没有继承关系。而且再次扩展游戏功能，在哪里插入继承关系没有一个好的规则。（entity的client属性更新，与方法设计相同）

2、interface扩展方式1（解决功能扩展问题）

客户端entity与interface是组合关系。通过配置表，将entity的className与Interface实例建立关系。Interface根据功能的扩展，进行添加新的interface。Interface需要重写GetMethodName(TArray<FString> &Method)方法，该方法需要将服务器调用客户端的远程方法名，添加到Method列表中。在创建interface时，会调用GetMethodName获取远程方法名，将远程方法名和interface实例，分别作为字典的key、value值存储在entity的字典中。当服务器一个远程方法过来时，就会在entity的字典中根据方法名获取interface实例，再调用interface对应的远程方法。

————————————————考虑到的问题———————————————————

Interface的扩展方式不支持，int64、uint64、自定义数据。如果传送上述数据，需要服务器将int64、uint64转化为字符串，自定义数据打包为json数据发送给客户端，然后再interface远程方法内解析json数据。这就导致消耗服务器与客户端传输消耗多余的带宽。因为字符串在网络的传输中消耗较大。数据消耗见下图（单位：字节）：



————————————————解决问题的方案—————————————————

1. 参考KBE的通信方式支持，在interface中实现一种机制，通过字符串来调用interface的方法。
2. 将服务器传过来的参数，查看虚幻有没有转化数据的相关支持。