层级：网络层-》业务逻辑层-》页面层，层级之间，上层提供接口给下层调用，比如：页面层可以调用业务逻辑层的接口，业务逻辑层可以调用网络层的接口，但是有一点必须注意：下层是绝对不能调用上层的接口的，像业务逻辑层是不允许调用页面的接口的，不同层级之间的调用，只能通过事件来进行处理

网络层

网络层核心代码都用C#写好，Lua里面就需要一个NetworkManager 来做交互，提供发送数据和解析数据的接口

业务逻辑层

业务逻辑层一般会分成三层：Info层-》Library层-》Manager层

Info层：只保存数据和提供返回数据的接口，不能跟其他层级做交互

Library层：只做逻辑处理，比如对Info层的数据做排序，筛选等等，同样不能跟其他层级做交互

Manager层：对外公开的接口，提供UI层或者其他Manager层调用的接口，Manager层之间可以互相调用，但是UI层只能调用Manager层的接口，Manager层不能调用UI层的接口，当网络有数据更新的时候，就触发一个事件，通知UI层有数据更新，然后调用Manager层的数据进行更新

备注：业务逻辑层分开三个层是为了代码扩展性更强，维护性更好，拿常用的聊天模块来说：

一般来说，聊天模块里面必须存在的两个参数：用户名和聊天内容，这两个数据都是保存在info层级的，Library只是负责传值而已，当聊天模块越来越复杂的时候，比如新增用户VIP等级，用户等级，用户称号等等，那么只需要添加Info层的数据即可，如果现在要开发新的项目，也有聊天系统，那么只需要把Info层和Libray拷贝过去即可，就不需要重复编写这部分的代码

页面层

页面层只是用于显示业务逻辑层的数据的，从Manager层里面取需要显示的数据，常用的逻辑最好不要在页面层实现，比如上面说到的排序和筛选功能，其他的模块都是可能会用到的，直接写在library层会让页面层的逻辑更清晰，也更好理解