[总： 2](#_Toc372062909)

[结构： 2](#_Toc372062910)

[数据层 2](#_Toc372062911)

[表现层： 2](#_Toc372062912)

[Item： 2](#_Toc372062913)

[Tips： 2](#_Toc372062914)

[逻辑层 3](#_Toc372062915)

[role 3](#_Toc372062916)

[控件的联动 3](#_Toc372062917)

[正式召唤 4](#_Toc372062918)

[头文件 4](#_Toc372062919)

[成员函数 5](#_Toc372062920)

# 总：

客户端是gui。

客户端的作用定位是表现层+保证输入合法。

客户端仅保留最少限度的游戏信息，说白了就是作为一个玩家能知道的东西。牌堆、别人手牌什么的都不维护，所以需各玩家互动的东西都做不了，需要服务器配合。

# 结构：

数据层dataInterface、表现层gui、逻辑层logic，各有一个extern的实例指针，在哪都可以引用

## 数据层

就是card、player、team没什么好讲的。。向表现层、逻辑层提供数据咯。dataInterface是所有数据的访问接口，里面有手牌、玩家信息等等。

要注意的是gui里player只作为显示的需要，跟角色技能什么的没有关系

## 表现层：

首先是QT要求的三层结构View、Scene、Item，在gui里分别对应RoomView、RoomScene跟各种Item。（如果你对找不到GraphicItem感到奇怪的话，那是因为GraphicObject继承了GraphicItem，而我们肯定要用QObject的东西，所以直接用GraphicObject了）

### Item：

Item级别内可以设置父子Item。最上层的是XXArea，一般作为管理旗下的子Item类，如HandArea管理手牌CardItem、PlayerArea管理玩家PlayerItem、ButtonArea管理特殊行动跟技能按钮、DecisionArea判断区就是Yes or No还有宣告不能行动（这个区域的按钮的操作很多时候不可逆、不能反悔）、TeamArea战绩区、TipArea提示框、LogArea记录栏。所有的XXArea类的引用都可以在gui里找到。

### Tips：

写角色要用到的一般是：

HandArea、PlayerArea

限制可选牌、玩家，有提供一些过滤的操作；控制能选多少牌、玩家（setQuota），选多会自动反选第一个Item；获取被选中的Item；直接反映数据层信息，注意不要直接在删除手牌Item，请从数据接口删除。

ButtonArea

限制可选按钮，这个区域的按钮都是同一时间只能选一个，多选前面的会自动反选

DecisionArea

什么时候亮哪个按钮、请尽量把No跟伤害摸牌挂钩：应战、魔蛋时点取消是承受伤害，除非你有盾；虚弱时点取消是摸牌。这个区域的按钮一旦选择（大多，“取消”有时例外）就会向服务器发信息，所以无法反悔

TipArea

支持下拉选择，详细请看role的合成

PS：对父Item的操作会反映到子Item，如屏蔽父Item，子Item也会被屏蔽。养成个好习惯，当你不需要操作的时候，把所有控件屏蔽掉（gui->setEnable(0)，屏蔽所有Area，从而屏蔽所有Item），以免走火。。。

## 逻辑层

重点，数据层、表现层相当于零件，怎么用零件拼出个有意义的“情景”就是逻辑层了。写角色改的也是这里。

服务器跟客户端只能通过字符串交流，所以逻辑层就是生成、解读这些字符串。详细看通信协议。注意：生成角色之前是logic在跟服务器交流、生成角色后就是角色在跟服务器交流。

### role

role是所有角色的基类，定义了所有角色的通用“情景”（攻击、应战、法牌、治疗、特殊行动、宣告不能行动）。有特殊情况的角色需要自己的类里覆写这些函数。所有角色都有一个decipher解码器跟onOKClicked、onCancelClicked两个encipher。写角色一般要涉及的是：

decipher

根据收到的命令执行相应的“情景”

encipher

容易发现不同时期点“确定”“取消”的含义是不同的，所以role还要维护一个state值。state1-9已被用。为避免冲突以后角色的state等于技能的发送信号。两个encipher根据state生成并发送命令给服务器。

cardAnalysis

见联动部分

### 控件的联动

之前提过客户端要保证输入的合理性，所以当用户按下“确定”时，各参数（选定的牌、玩家）都应该准备好，不难推出“确定”必然是要最后才激活的，也就是说控件的激活有一定的顺序。

最完整的流程是这样：**激活可选技能、（选完技能后）激活可选牌、（选完牌后）激活可选人、（选完人后）激活确定。**当然实际情况有些步骤可能没有，如弃牌就只有激活可选牌、（选完牌后）激活确定。通用情景已经写好了，但当新角色有特殊的流程时必须扩展这一部分。具体可参考弓女的贯穿射击、狙击。

PS：选完技能后将直接通过gui指针激活下一步可选的控件（牌/人/确定），但选完牌、选完人不一样，选完CardItem后，HandArea会发出CardReady信号，而CardReady信号是跟Role的CardAnalyse绑定；选完PlayerItem后，PlayerArea将发出PlayerReady信号，而playerReady信号跟Role的PlayerAnalyse绑定。这样考虑是为了将所有跟角色技能有关的东西，都放进角色类里，降低HandArea、PlayerArea跟角色的耦合。

# 正式召唤

核心思想就是：所有跟角色技能有关的东西都在该角色的类里。如没特殊情况可直接调用父类Role的函数，如一般人的治疗响应。但像神官就要覆写cure函数，死灵、红莲之流估计早在服务器就会把治疗ban掉。

### 头文件

为方便并行开发请自行在logic目录下新建你的新角色类。考虑到大伙的英文水准，主要是我的英文水准，为避免奇怪、难以理解的词，请尽量用两个字简称的拼音来命名你的新角色。新类要继承父类Role。

一些常用的成员函数：

1）两个SLOT： onOkClicked、onCancelClicked

2）像之前提过的流程：激活可选技能、（选完技能后）激活可选牌、（选完牌后）激活可选人、（选完人后）激活确定。如果你的角色在选牌后有特殊的要求，如弓女的贯穿射击：选完法术牌后直接激活确定按钮，你还需要覆写cardAnalyse SLOT。playerAnalyse同理。

3）如果你的角色有非响应类技能的话，一般都要在回合开始时给出那些技能能否发动的判定，这就需要覆写normal，同理如果你角色有机会加1法术行动，并且有自己独特的法术（非中毒、虚弱之流）的话，你也需要覆写magicAction、attackOrMagic。这三个都是普通函数，不需要是SLOT

4）无法一言以盖全部，但凡你有想实现的东西，请先查看Role有没有相应的函数。然后如果想改Role的话，请跟我（图叔）说一声。

一般来讲，为了清晰，请为每一个技能建立一个成员函数，并以技能全称的拼音命名。技能分两种：响应类跟非响应类。响应类包括启动。响应类的技能将通过提示框、确定、取消按钮三者来呈现；非响应类的技能将通过技能按钮呈现（跟特殊行动一样归于ButtonArea）。请把所有非响应类的技能定义为SLOT（如弓女的闪光陷阱）。

### 成员函数

角色的构建函数需要调用makeConnection进行常规的signal-slot绑定。当你的角色有非响应类技能时，还需要在这里新建按钮，并完成新按钮跟处理函数的绑定（buttonSelected绑技能函数、buttonUnselected绑onCancelClicked），具体参见弓女的狙击、闪光陷阱。

技能函数用来实现你需要的界面跟限制用户的输入：提示框的内容、能选的牌/能选的人、牌的quota、人的quota、确定、取消按钮，注意控件要有序的激活，不要一下子把所有控件开放，不然参数没全就能点确定按钮的话，会报错的。。。怎么个有序法请参靠前面的流程，然后非响应类的技能请让取消按钮具有返回的功能。

因为要有序的开放控件，一个技能牵扯的就不只是它的技能函数，至少还有onOKClicked、onCancelClicked，还有可能有cardAnalysis，当然响应类的话，就还有deciper。它们之前需要协同工作，而且都是复用的，所以要维护一个state值。请把这个state的赋值放在技能函数里。为避免冲突state值一律使用角色技能的命令号。

deciper、onOKClicked、onCancelClicked、cardAnalysis都应在开始时调用父类role对应的方法。然后再实现自己的内容。所有的命令解读放在deciper，所有的命令发送放在onOKClicked、onCancelClicked。删除手牌请在onOKClicked里通过dataInterface删除。onOKClicked、onCancelClicked后适当的加上reset。

PS：这里请慎重使用reset，因为你会把父类的结果给重设了

最后是约定：为了避免混乱，角色技能的命令号一律用roleID+n，n为01、02、03、04、。。。如弓女是301、302、。。。