面向对象程序设计 课程报告

［姓名］沈韵泓［学号］3200104392

---

1 个人贡献

在本次Project开发的过程中：作为组长，我承担了总体架构设计、任务分配及进度跟进、文档整理及UI规范制定与图片素材整理的相关工作。而作为组员，完成了准备界面的功能设计与实现、21点前端界面的实现。

2 总结

2.1 先了解实施难度，后指定开发计划

受个人时间精力限制，本组仅在最初确定选题及希望实现的高级功能前对相关资料进行了收集与较为深入的了解，最终确立的选择与预期大体相符。但由于开发周期较短，在本人细化并设计具体实现方案时未考虑到每一部分的具体实现难度，如发牌时扑克平移加旋转的动画、页面切换时的淡入淡出效果、人物贴图的表情差分实现.....导致在这些技术细节上耗费了大量时间成本，却没有得到有效的产出。因此，在后续开发中，应当事先对方案中各技术细节的实施难度预先进行了解，并有所取舍，从而在有限的开发周期中按时完成最小可用模型，并在之后进行迭代。

2.2 窗口切换闪烁问题及其解决

在初期实现准备界面时，采用了多窗口切换的方式进行开发。然而在测试过程中发现：由于导入了图片素材，Qt在即时读入时会导致较长的延时；而预先加载所有窗口并设置其可见性的方案会在用户移动窗口后导致明显的位移，造成不良的使用体验。

在查询相关资料后，决定采用单页面多图层方案进行替代，该方案在使用初期也遇到了一定阻力。由于存在多窗口共用的组件，准备界面在实现初期是单独对各个组件设置其可见性的，导致代码大量重复、并在较大跨度的界面切换时易出现个别控件未隐藏的现象。

在21点游戏界面的实现中，采用了以frame控件代替成员函数实现分图层显示效果的方案。通过设置frame组件的可见性，完成对整个卡片页内容可见性的调整，有效解决了部分控件漏设的问题。对于部分多界面共用控件，选择不将其放入frame控件，并单独设置其可见性。

最终，以frame控件模拟多图层，以实现单窗口开发的方案有效解决的Qt进行窗口切换时的闪烁问题，并避免了因控件数量繁多而遗漏的情况。这一方案在现实玩家手牌卡组时同样适用。

2.3 使用Timer完成线性的游戏流程

为实现过场动画、AI操作、庄家操作与最终结算过程的自动进行，最初采用sleep命令使程序在一定时间间隔后继续执行。但由于sleep指令实际实现的停顿时间与预定时间存在出入，后决定采用timer控件实现定时间间隔的动画效果。

由于Qt中的Timer控件会为程序新增一个线程执行相关代码，导致执行过程由线性变为并发，与预期设计存在出入，甚至出现timer间互相调用导致申请线程数超过上限导致程序崩溃的情况。最终采用了在timer发出timeout信号时将timer设为stop，并调用其他成员函数推动游戏流程的解决方案。

2.4 控件数组的取舍

在进行扑克牌绘制时，采用了以单张卡片作为一个frame，包含用于显示点数的label*1、显示花色的label*4和显示底纹的label*1，存在大量结构相似的结构。采用控件数组将使得对卡组可将状况的批量设置变得更为便捷：只需要使用一个for循环，而不需要使用大量代码通过unique的控件名对对应控件进行设置。

然而，由于Qt框架的限制，预先在UI设计界面上绘制的控件是无法设置为控件数组的。在程序段中新建对象虽然能降低设置可见属性时的代码重复度，但违背了使用控件预先加载图片资源降低延迟的初衷。因此在最终的程序实现中并未采用控件数组，而是以控件名称作为标识进行控件的相关设置。

3 感受

由于是首次担任Project项目组长， 感受到了作为项目负责人的不易：如何进行项目设计、如何合理分配任务、如何有效沟通并跟进进度都成了亟待解决的问题。在本次项目过程中采用了钉钉作为交流工具，并通过共享文档记录各组的进度以方便进度记录与组员沟通。由于部分组员存在Git使用不熟练的情况，本次项目并未采用git进行版本管理，也没有建立远程仓库，而是在约定规范并完成三部分开发后（准备页面、21点、德州扑克）后人工整合，为最终的整合阶段造成了很多困难，需要在以后的开发过程中加以避免。

作为主要负责前端页面实现的开发者，首先在开发过程中体会到了C++与C的巨大区别：开发初期由于对C++理解不够透彻，往往在外部调用类的私有成员导致报错，最终通过对相关类的深入了解避免了该情况的发生。庄家、玩家与AI类之间类似的结构与继承关系使得函数调用在形式上具有极高的一致性，避免了C语言中需要以不同函数名区分对于不同自定义结构内容的操作，极大的提高了开发效率。同时，在开发过程中对Qt中的信号槽机制、UI机制及Timer控件的相关使用进行了解，收获颇丰。

由于个人能力限制，本次21点游戏的流程主要通过timer和button的点击事件实现线性推进。并由于复用界面（过渡页）等元素的存在导致跳转逻辑较为凌乱。希望能通过之后的学习增进对界面逻辑的理解，设计并实现逻辑更为清晰的流程触发与推进模式。