2023年数据库课程设计体会与感悟 数据库课程设计心得体会(通用5篇)

我们在一些事情上受到启发后，应该马上记录下来，写一篇心得感悟，这样我们可以养成良好的总结方法。那么你知道心得感悟如何写吗？以下是小编帮大家整理的心得感悟范文，欢迎大家借鉴与参考，希望对大家有所帮助。

数据库课程设计体会与感悟篇一

一周的课程设计结束了，在这次的课程设计中不仅检验了我所学习的知识，也培养了我如何去把握一件事情，如何去做一件事情，又如何完成一件事情的方法和技巧。在设计过程中，和同学们相互探讨，相互学习，相互监督。我学会了运筹帷幄，学会了宽容，学会了理解，也学会了做人与处世，这次课程设计对我来说受益良多。

课程设计是我们专业课程知识综合应用的实践训练，着是我们迈向社会，从事职业工作前一个必不少的过程。“千里之行始于足下”，通过这次课程设计，我深深体会到这句千古名言的真正含义。我今天认真的进行课程设计，学会脚踏实地迈开这一步，就是为明天能稳健地在社会大潮中奔跑打下坚实的基础。我这次设计的科目是数据结。

数据结构，是一门研究非数值计算的程序设计问题中计算机的操作对象（数据元素）以及它们之间的关系和运算等的学科，而且确保经过这些运算后所得到的新结构仍然是原来的结构类型。“数据结构”在计算机科学中是一门综合性的专业基础课。数据结构是介于数学、计算机硬件和计算机软件三者之间的一门核心课程。数据结构这一门课的内容不仅是一般程序设计（特别是非数值性程序设计）的基础，而且是设计和实现编译程序、操作系统、数据库系统及其他系统程序的重要基础。通过这次模具设计，我在多方面都有所提高。

在界面设置中使用函数调用while□其中文本显示颜色和背景颜色都可以任意按照自己的喜好，任意改变，但改变的时候必须采用标准英文大写，同时在制作显示菜单的窗口，大小根据菜单条数设计。最后采用printf输出程序设计界面。

这次的程序软件基本上运行成功，可以简单的建立链式循环链表，并进行输出，及循环语句的运用和选择语句的控制。由于时间和知识上的限制，使得程序规模相对较小，即功能还不很全面，应用也不很普遍。原来c语言可是涉及很多知识，而不是枯燥无聊的简单的代码部分而已，利用c语言方面的知识，我们可以设计出更完善的软件。

通过这次的课程设计，更是让我深刻认识到自己在学习中的不足，同时也找到了克服这些不足的方法，这也是一笔很大的资源。在以后的时间中，我们应该利用更多的时间去上机实验，加强自学的能力，多编写程序，相信不久后我们的编程能力都会有很大的提高能设计出更多的更有创新的作品。

数据库课程设计体会与感悟篇二

两个星期时间非常快就过去了，这两个星期不敢说自己有多大进步，获得了多少知识，但起码是了解了项目开发部分过程。虽说上过数据库上过管理信息系统等相关课程，但是没有亲身经历过相关设计工作细节。这次实习证实提供了一个很好机会。

通过这次课程设计发现这其中需要很多知识我们没有接触过，去图书馆查资料时候发现我们前边所学到仅仅是皮毛，还有很多需要我们掌握东西我们根本不知道。同时也发现有很多已经学过东西我们没有理解到位，不能灵活运用于实际，不能很好用来解决问题，这就需要我们不断大量实践，通过不断自学，不断地发现问题，思考问题，进而解决问题。在这个过程中我们将深刻理解所学知识，同时也可以学到不少很

实用东西。从各种文档阅读到开始需求分析、概念结构设计、逻辑结构设计、物理结构设计。亲身体验了一回系统设计开发过程。很多东西书上写很清楚，貌似看着也很简单，思路非常清晰。但真正需要自己想办法去设计一个系统时候才发现其中难度。经常做到后面突然就发现自己一开始设计有问题，然后又回去翻工，在各种反复中不断完善自己想法。

我想有这样问题不止我一个，事后想想是一开始着手做时候下手过于轻快，或者说是根本不了解自己要做这个系统是给谁用。因为没有事先做过仔细用户调查，不知道整个业务流程，也不知道用户需要什么功能就忙着开发，这是作为设计开发人员需要特别警惕避免，不然会给后来工作带来很大的麻烦，甚至可能会需要全盘推倒重来。所以以后课程设计要特别注意这一块设计。

按照要求，我们做是机票预订系统。说实话，我对这个是一无所知，没有订过机票，也不知道航空公司是怎么一个流程。盲目开始设计下场我已经尝过了，结果就是出来一个四不像设计方案，没有什么实际用处。没有前期调查，仅从指导书上那几条要求着手是不够。

在需求分析过程中，我们通过上网查资料，去图书馆查阅相关资料，结合我们生活经验，根据可行性研究结果和客户要求，分析现有情况及问题，采用client/server结构，将机票预定系统划分为两个子系统：客户端子系统，服务器端子系统。在两周时间里，不断地对程序及各模块进行修改、编译、调试、运行，其间遇到很多问题：由于忘记了一些java语言规范使得在调试过程中一些错误没有发现，通过这次课程设计，我对调试掌握得更加熟练了，意识到了程序语言规范性以及我们在编程时要有严谨态度，同时在写程序时如有一定量注释，既增加了程序可读性，也可以使自己在读程序时更容易。

我们学习并应用了sql语言，对数据库创建、修改、删除方法有了一定了解，通过导入表和删除表、更改表学会了对于表

一些操作，为了建立一个关系数据库信息管理系统，必须得经过系统调研、需求分析、概念设计、逻辑设计、物理设计、系统调试、维护以及系统评价一般过程，为毕业设计打下基础。

很多事情不是想象中那么简单，它涉及到各种实体、属性、数据流程、数据处理等等。很多时候感觉后面设计根本无法继续，感觉像是被前面做各种图限制了。在做关系模型转换时候碰到有些实体即可以认为是实体又可以作为属性，为了避免冗余，尽量按照属性处理了。

物理结构设计基本没有碰到问题，这一块和安全性、完整性不觉就会在物理结构设计中添加一些安全设置：主键约束□check约束□default定义等。最后才做索引部分，对一些比较经常使用搜索列，外键上建立索引，这样可以明显加快检索速度，最后别忘记重要安全性设置，限制用户访问权限，新建用户并和数据库用户做相应映射。

不管做什么，我们都要相信自己，不能畏惧，不能怕遇到困难，什么都需要去尝试，有些你开始认为很难事在你尝试之后你可能会发现原来她并没有你以前觉得那样，自己也是可以。如果没有自信，没有目标，没有信心就不可能把事情做好，当其他人都在迷茫时候，自己一定要坚信目标，大学毕业出去即面临找工作，从学习这个专业，到以后从事这方面工作都需要不断地去学习去实践，这次实践可以给我们敲一个警钟，我们面临毕业，面临择业，需要这些实践经验，在困难面前要勇于尝试，这是这次课程设计给我最大感想！

以上基本是这次实习体会了，设计进行非常艰难，编码非常不容易，才发现做一个项目最重要不在于如何实现，而是实现之前需求分析和模块设计。创新很难，有些流行系统其实现并不难，难在于对市场分析和准确定位。设计，是一个任重道远过程。

数据库课程设计体会与感悟篇三

由于平时接触的都是一些私人项目，这些项目大都是一些类库，其他人的交流相对可以忽略不计，因此也就不考虑规范化的文档。实际上从学习的经历来看，我们接触的知识体系都是属于比较老或比较传统的，与现在发展迅速的it行业相比很多情况已不再适用，尤其是当开源模式逐渐走近开发者后更是如此。

虽然这次是一个数据库课程设计，由于本人在选择项目的时候是本着对自己有实际应用价值的角度考虑的，所以其中也涉及到一些数据库以外的设计。对于ooa/ood的开发模式有时不免要提出一些疑问□uml是设计阶段的工具，而它基本涵盖了软件设计的方方面面，也就是说按照这一软件工程的正常流程，在动手写第一句代码之前，开发人员已经非常熟悉软件产品了，这对于相当有经验的架构师一类人说可能会很容易，但是我们作为学生，连足够的编码经验都没有，却首先被教授并要求先ooa再oop□这样直接导致的。问题就是文档与编码对不上号，在修改代码的时候基本不会再去审查文档和先前的分析。甚至根本就是现有代码再有文档，即便是这种情况，代码与文档还是不对应。不可否认，在传统软件工程的详细设计之前的项目过程中还是有很多利于项目开发的部分的。所以我就一直在寻找适合我针对探究型项目的开发模式，这次的项目也算是一次尝试，当然这个过程并不会太短。

回到数据库设计上了，这次的数据库设计我是严格按照数据库建模的步骤来进行的，老实说我并没有感觉这样的流程对开发带来多大的帮助，反倒是觉得将思维转化为图表很浪费时间。总体上来说这次的项目也不是很大，而且在数据库的设计上比较保守，也就是说实际上数据库设计还可以再完善完善的。随着我对计算机领域的拓宽和加深，我也会静下心来思考在接触计算机之前的行为，很多次我能深切感觉到，

其实我的大脑(未于别人比较)本身就是在使用一种更接近关系数据库的方式来记忆，所以我很可恨自然的设计出符合三范式的表结构来，即便我不知道这些范式的确切含义。可能就像范式不太容易用通俗易懂的方式解释一样，在让工具用图标表述我的思维时费了一番力气。

从我作为项目的提出人和实现者来看，这是个失败的项目，结合几次教学项目的的实践，发现这也已经不是第一次了。主观原因占多数，比如，尝试新的开发方式，根据设计花了太多的时间来抽象出公用的库而忽略业务逻辑。就这次项目而言，失败的原因有以下几点：

使用了新的开发环境(vim)□这是首次在脱离高级ide的情况下编码。

使用了新的开发语言(python□actionscript3)□因为我一直比较喜欢学以致用，而且这样的数据驱动型软件的整套自实现的库都已经完成了，但是由于语言本身的差异，迁移时问题很多，当发现这一点是，已没有多少有效剩余时间了。编码流程的不妥，我比较喜欢从底层的库开始开发，因为一旦库测试通过，将很容易将它放到不同的表示层下。但如果库没有测试成功，将导致整个项目没有任何可视化模型，所以这次的项目无法提交可运行的代码。

实践目的的不同，我轻易不放弃锻炼的机会，事实上，有机会就一定要比以前有所突破，总是照搬以前的做法还不如就不做呢。这个前提是因为现在能完全用来的学习的时间比较多，等到工作时再这样做的可能性就很小了，因此当然要抓紧机会了。不过还有一个隐藏原因，总以为自己很了不起，其实遇到的问题数跟人的能力是成正比的。

数据库课程设计体会与感悟篇四

由于平时接触的都是一些私人项目，这些项目大都是一些类

库，其他人的交流相对可以忽略不计，因此也就不考虑规范化的文档。实际上从学习的经历来看，我们接触的知识体系都是属于比较老或比较传统的，与现在发展迅速的it行业相比很多情况已不再适用，尤其是当开源模式逐渐走近开发者后更是如此。

虽然这次是一个数据库课程设计，由于本人在选择项目的时候是本着对自己有实际应用价值的角度考虑的，所以其中也涉及到一些数据库以外的设计。对于ooa/ood的开发模式有时不免要提出一些疑问□uml是设计阶段的工具，而它基本涵盖了软件设计的方方面面，也就是说按照这一软件工程的正常流程，在动手写第一句代码之前，开发人员已经非常熟悉软件产品了，这对于相当有经验的架构师一类人说可能会很容易，但是我们作为学生，连足够的编码经验都没有，却首先被教授并要求先ooa再oop□这样直接导致的问题就是文档与编码对不上号，在修改代码的时候基本不会再去审查文档和先前的分析。甚至根本就是现有代码再有文档，即便是这种情况，代码与文档还是不对应。不可否认，在传统软件工程的详细设计之前的项目过程中还是有很多利于项目开发的部分的。所以我就一直在寻找适合我——针对探究型项目——的开发模式，这次的项目也算是一次尝试，当然这个过程并不会太短。

回到数据库设计上了，这次的数据库设计我是严格按照数据库建模的步骤来进行的，老实说我并没有感觉这样的流程对开发带来多大的帮助，反倒是觉得将思维转化为图表很浪费时间。总体上来说这次的项目也不是很大，而且在数据库的设计上比较保守，也就是说实际上数据库设计还可以再完善完善的。随着我对计算机领域的拓宽和加深，我也会静下心来思考在接触计算机之前的行为，很多次我能深切感觉到，其实我的大脑（未于别人比较）本身就是在使用一种更接近关系数据库的方式来记忆，所以我很可恨自然的设计出符合三范式的表结构来，即便我不知道这些范式的确切含义。可

能就像"范式不太容易用通俗易懂的方式解释"一样，在"让工具用图标表述我的思维"时费了一番力气。

从我作为项目的提出人和实现者来看，这是个失败的项目，结合几次教学项目的的实践，发现这也已经不是第一次了。主观原因占多数，比如，尝试新的开发方式，根据设计花了太多的时间来抽象出公用的库而忽略业务逻辑。就这次项目而言，失败的原因有以下几点：

使用了新的开发环境□vim□□这是首次在脱离高级ide的情况下编码。

使用了新的开发语言□python□actionscript3□□因为我一直比较喜欢"学以致用"，而且这样的"数据驱动型"软件的整套自实现的库都已经完成了，但是由于语言本身的差异，迁移时问题很多，当发现这一点是，已没有多少有效剩余时间了。

编码流程的不妥，我比较喜欢从底层的库开始开发，因为一旦库测试通过，将很容易将它放到不同的表示层下。但如果库没有测试成功，将导致整个项目没有任何可视化模型，所以这次的项目无法提交"可运行的代码"。

实践目的的不同，我轻易不放弃锻炼的机会，事实上，有机会就一定要比以前有所突破，总是照搬以前的做法还不如就不做呢。这个前提是因为现在能完全用来的学习的时间比较多，等到工作时再这样做的可能性就很小了，因此当然要抓紧机会了。不过还有一个隐藏原因，总以为自己很了不起，其实"遇到的问题数跟人的能力是成正比的"。

数据库课程设计体会与感悟篇五

两个星期的时间非常快就过去了，这两个星期不敢说自己有多大的进步，获得了多少知识，但起码是了解了项目开发的

部分过程。虽说上过数据库上过管理信息系统等相关的课程，但是没有亲身经历过相关的设计工作细节。这次实习证实提供了一个很好的机会。

通过这次课程设计发现这其中需要的很多知识我们没有接触过，去图书馆查资料的时候发现我们前边所学到的仅仅是皮毛，还有很多需要我们掌握的东西我们根本不知道。同时也发现有很多已经学过的东西我们没有理解到位，不能灵活运用于实际，不能很好的用来解决问题，这就需要我们不断的大量的实践，通过不断的自学，不断地发现问题，思考问题，进而解决问题。在这个过程中我们将深刻理解所学知识，同时也可以学到不少很实用的东西。

从各种文档的阅读到开始的需求分析、概念结构设计、逻辑结构设计、物理结构设计。亲身体验了一回系统的设计开发过程。很多东西书上写的很清楚，貌似看着也很简单，思路非常清晰。但真正需要自己想办法去设计一个系统的时候才发现其中的难度。经常做到后面突然就发现自己一开始的设计有问题，然后又回去翻工，在各种反复中不断完善自己的想法。

我想有这样的问题不止我一个，事后想想是一开始着手做的时候下手过于轻快，或者说是根本不了解自己要做的这个系统是给谁用的。因为没有事先做过仔细的用户调查，不知道整个业务的流程，也不知道用户需要什么功能就忙着开发，这是作为设计开发人员需要特别警惕避免的，不然会给后来的工作带来很大的麻烦，甚至可能会需要全盘推倒重来。所以以后的课程设计要特别注意这一块的设计。

按照要求，我们做的是机票预订系统。说实话，我对这个是一无所知的，没有订过机票，也不知道航空公司是怎么一个流程。盲目开始设计的下场我已经尝过了，结果就是出来一个四不像的设计方案，没有什么实际用处。没有前期的调查，仅从指导书上那几条要求着手是不够的。

在需求分析过程中，我们通过上网查资料，去图书馆查阅相关资料，结合我们的生活经验，根据可行性研究的结果和客户的要求，分析现有情况及问题，采用client/server结构，将机票预定系统划分为两个子系统：客户端子系统，服务器端子系统。在两周的时间里，不断地对程序及各模块进行修改、编译、调试、运行，其间遇到很多问题：由于忘记了一些java语言的规范使得在调试过程中一些错误没有发现，通过这次课程设计，我对调试掌握得更加熟练了，意识到了程序语言的规范性以及我们在编程时要有严谨的态度，同时在写程序时如有一定量的注释，既增加了程序的可读性，也可以使自己在读程序时更容易。

我们学习并应用了sql语言，对数据库的创建、修改、删除方法有了一定的了解，通过导入表和删除表、更改表学会了对于表的一些操作，为了建立一个关系数据库信息管理系统，必须得经过系统调研、需求分析、概念设计、逻辑设计、物理设计、系统调试、维护以及系统评价的一般过程，为毕业设计打下基础。

很多事情不是想象中的那么简单的，它涉及到的各种实体、属性、数据流程、数据处理等等。很多时候感觉后面的设计根本无法继续，感觉像是被前面做的各种图限制了。在做关系模型转换的时候碰到有些实体即可以认为是实体又可以作为属性，为了避免冗余，尽量按照属性处理了。

物理结构设计基本没有碰到问题，这一块和安全性、完整性不觉就会在物理结构设计中添加一些安全设置：主键约束□check约束□default定义等。最后才做索引的部分，对一些比较经常使用搜索的列，外键上建立索引，这样可以明显加快检索的速度，最后别忘记重要的安全性设置，限制用户访问权限，新建用户并和数据库用户做相应的映射。

不管做什么，我们都要相信自己，不能畏惧，不能怕遇到困难，什么都需要去尝试，有些你开始认为很难的事在你尝试

之后你可能会发现原来她并没有你以前觉得的那样，自己也是可以的。如果没有自信，没有目标，没有信心就不可能把事情做好，当其他人都在迷茫的时候，自己一定要坚信目标，大学毕业出去即面临找工作，从学习这个专业，到以后从事这方面的工作都需要不断地去学习去实践，这次实践可以给我们敲一个警钟，我们面临毕业，面临择业，需要这些实践经验，在困难面前要勇于尝试，这是这次课程设计给我的最大感想！

以上基本是这次实习的体会了，设计进行的非常艰难，编码非常不容易，才发现做一个项目最重要的不在于如何实现，而是实现之前的需求分析和模块设计。创新很难，有些流行的系统其实现并不难，难的在于对市场的分析和准确定位。设计，是一个任重道远的过程。

数据库课程设计大赛的尘嚣渐渐远去，怀着对这次大赛的些许不舍，怀着对当初课程设计开始时候的豪情万丈的决心的留恋，怀着通过这次课程设计积累的信心与斗志，我开始写这篇文章，为自己的足迹留下哪怕是微不足道但是对自己弥足珍贵的痕迹并期望与大家共勉。

首先，让我的记忆追溯到大二暑假，在老大的指引下()，我接触到microsoft 产品。那个时候我已经学过vc和asp□因为windows程序设计实验的课的关系，接触过vb□但是没有专门去学他，因为习惯了c++里面的class□int□觉得vb 的sub□var 看着就不是很顺心。我是一个好奇心很强的人，突然看到了一个号称“.net是用于创建下一代应用程序的理想而又现实的开发工具”，而且主推c#语言，由于对c语言的一贯好感，我几乎是立刻对他产生了兴趣。我就开始了对c#的学习，任何语言都不是孤立存在的，所以数据交互是很重要的，暑假的时候我把我们这学期的课本数据库系统概论看了一遍。我记得以前用c语言编程的时候，数据是在内存中申请空间，譬如使用数组等等。很耗费内存空间。这个时候就是

数据库站出来的时候啦，于是我又装上了sql server2019□以前学asp的时候用的是access□那个时候只是照着人家做，理论是什么也不是很清楚。

通过一个暑假的学习，基本搞清楚了理论方面的东西，具体怎么用也不是很清楚。但是这为这学期的课程设计打下了铺垫。

来到学校后，随着这学期的数据库课程大赛开始了，我有一个看法就是我自己应该具备的能力不是我会多少，而是我应该具备快速学会东西的能力。遇到什么就学什么。我们有时候很容易被一些专业名词说吓着，包括什么建模，软件工程，数据分析，数据挖掘等等。我身边就有很多同学被这些纸老虎所唬住，而没有勇气去接触他们，总是说这个太难了之类的退堂鼓的话，他们低估了自己的潜力同时也压抑住了他们自己的好奇心。其实都是纸老虎，又不是什么国家科研难题，只是去用一些工具，发明工具是很难，但是用一个工具就容易多了□just do it!我记得我做这个数据库之前，我们老师说要做好前期分析，我就在网上搜索用什么分析工具好。最后我选择了roseuml建模工具。在此之前，我脑袋里面没有软件建模的思想，什么uml建模对我而言就是一张空白的纸。但是真正接触后并没有想象的那么难，有什么不懂的上网去搜索，这是一个信息横流的世界，有google□baidu就没有不能解决的知识难题。以及后来的数据库分析的时候用到的powerdesigner也是一样。

开发的时候我想过用什么架构□c/s模式?模式有很多，怎么选择?我就上网搜索现在最流行的架构是什么。结果搜到了mvc架构，就是你啦。我决定用这个架构，不会，没关系，咱学□just do it!前期工作准备好后，加以实践。这个时候我更加深入的了解了利用操纵数据库的知识。并且对数据库里面的存储过程有了比较深入的了解。经过大概2个多星期的奋斗，数据集的图书馆管理系统。并最后非常荣幸的获得了大赛的

一等奖以及以及新技术应用奖。

与其临渊羡鱼，不如退而结网。这次数据库课程设计给我的最大的印象就是如果自己有了兴趣，就动手去做，困难在你的勇气和毅力下是抬不了头的。从做这个数据库开始无论遇到什么困难，我都没有一丝的放弃的念头。出于对知识的渴望，出于对新技术的好奇，出于对一切未知的求知。我完成了这次数据库课程设计，不过这只是我学习路上的驿站，的核心技术就是xml[至少微软是这么宣传的]，我会继续学习它，包括jave公司的j2ee我也很想试试，语言本来就是相通的□just do it!语言并不重要毕竟它仅仅是工具，用好一个工具并不是一件值得为外人道的事情，主要是了解学习思想。古语说的好：学无止境啊！

我很庆幸我参加了这次数据库大赛，让我确实打开了眼界。

由于平时接触的都是一些私人项目，这些项目大都是一些类库，其他人的交流相对可以忽略不计，因此也就不考虑规范化的文档。实际上从学习的经历来看，我们接触的知识体系都是属于比较老或比较传统的，与现在发展迅速的it行业相比很多情况已不再适用，尤其是当开源模式逐渐走近开发者后更是如此。

虽然这次是一个数据库课程设计，由于本人在选择项目的时候是本着对自己有实际应用价值的角度考虑的，所以其中也涉及到一些数据库以外的设计。对于ooa/ood的开发模式有时不免要提出一些疑问□uml是设计阶段的工具，而它基本涵盖了软件设计的方方面面，也就是说按照这一软件工程的正常流程，在动手写第一句代码之前，开发人员已经非常熟悉软件产品了，这对于相当有经验的架构师一类人说可能会很容易，但是我们作为学生，连足够的编码经验都没有，却首先被教授并要求先ooa再oop□这样直接导致的问题就是文档与编码对不上号，在修改代码的时候基本不会再去审查文档和

先前的分析。甚至根本就是现有代码再有文档，即便是这种情况，代码与文档还是不对应。不可否认，在传统软件工程的详细设计之前的项目过程中还是有很多利于项目开发的部分的。所以我就一直在寻找适合我——针对探究型项目——的开发模式，这次的项目也算是一次尝试，当然这个过程并不会太短。

回到数据库设计上了，这次的数据库设计我是严格按照数据库建模的步骤来进行的，老实说我并没有感觉这样的流程对开发带来多大的帮助，反倒是觉得将思维转化为图表很浪费时间。总体上来说这次的项目也不是很大，而且在数据库的设计上比较保守，也就是说实际上数据库设计还可以再完善完善的。随着我对计算机领域的拓宽和加深，我也会静下心来思考在接触计算机之前的行为，很多次我能深切感觉到，其实我的大脑(未于别人比较)本身就是在使用一种更接近关系数据库的方式来记忆，所以我很可恨自然的设计出符合三范式的表结构来，即便我不知道这些范式的确切含义。可能就像“范式不太容易用通俗易懂的方式解释”一样，在“让工具用图标表述我的思维”时费了一番力气。

从我作为项目的提出人和实现者来看，这是个失败的项目，结合几次教学项目的的实践，发现这也已经不是第一次了。主观原因占多数，比如，尝试新的开发方式，根据设计花了太多的时间来抽象出公用的库而忽略业务逻辑。就这次项目而言，失败的原因有以下几点：

使用了新的开发环境(vim)□这是首次在脱离高级ide的情况下编码。

使用了新的开发语言(python□actionscript3)□因为我一直比较喜欢“学以致用”，而且这样的“数据驱动型”软件的整套自实现的库都已经完成了，但是由于语言本身的差异，迁移时问题很多，当发现这一点是，已没有多少有效剩余时间了。

编码流程的不妥，我比较喜欢从底层的库开始开发，因为一旦库测试通过，将很容易将它放到不同的表示层下。但如果库没有测试成功，将导致整个项目没有任何可视化模型，所以这次的项目无法提交“可运行的代码”。

实践目的的不同，我轻易不放弃锻炼的机会，事实上，有机会就一定要比以前有所突破，总是照搬以前的做法还不如就不做呢。这个前提是因为现在能完全用来的学习的时间比较多，等到工作时再这样做的可能性就很小了，因此当然要抓紧机会了。不过还有一个隐藏原因，总以为自己很了不起，其实“遇到的问题数跟人的能力是成正比的”。