《数据库系统原理》课程设计

系统实现报告

题目名称：“物外”校园二手交易平台

学号及姓名： 16061200 陈治齐

16061202 胡俊崧

2018年12月14日

# 系统功能需求分析

用户进行交易的大致模式如下：

1. 卖家将想要出售的商品进行登记，包括商品名，售价，数量，详细描述，商品图片。
2. 用户进入交易平台，搜索自己感兴趣的商品或者通过浏览商品列表找到商品，进行下单购买。
3. 用户下单之后生成订单，之后卖家确认订单并发货，卖家收货之后订单关闭。在未收到货前，用户可以取消订单。

经过分析,对于不同类型的用户，对应的具体功能描述如下：

1. 访客：
2. 浏览、查找商品条目，查看商品详细信息
3. 登陆或注册（填写个人信息姓名、联系方式、地址等）成为会员
4. 会员：
5. 作为买家：
6. 浏览、查找商品条目，查看商品详细信息
7. 收藏商品，为商品点赞
8. 评论商品
9. 查看、修改个人信息
10. 给个人账户充值
11. 查看卖家个人信息，与卖家交流
12. 下单购买，查看物流信息，确认收货或者取消订单
13. 作为卖家：
14. 发布商品，查看、修改、删除商品
15. 查看订单，确认订单以及处理发货
16. 管理员：
17. 浏览商品信息,查看用户信息
18. 接受用户请求，处理订单信息
19. 审核编辑商品信息

# 系统功能结构设计

系统的功能主要有用户管理、商品管理、订单管理和评论管理四大部分，如下图所示。



用户管理部分包括用户的注册和登录，用户名、更换等基本信息的修改，以及账户余额充值。

商品管理分为前台和后台两部分，前台主要有商品的展示和商品的收藏，后台部分包括商品发布、修改、删除以及管理商品的分类。

订单管理有订单创建、确认和取消。评论管理部分，可以发表评论，展示已经发表的评论。

用户可以在商品展示页面收藏或者购买商品，个人中心页面可以查看订单，进行订单的确认或者取消，也可以查看已收藏的商品，在收藏页面购买商品或者取消收藏商品。在商品页面，除了可以看到商品的基本信息外，还可以查看到不同用户对商品的评论，商品被收藏的次数，以及拥有该商品的卖家在平台上的在售商品数和交易数。在后台管理页面，用户可以查看、增加、修改或删除自己的商品，也可以为自己的商品添加新的分类。

# 数据库基本表的定义

1. 创建数据库：数据库名为“db\_course\_design”，使用UTF8编码格式。

**CREATE** **DATABASE** db\_course\_design **DEFAULT** CHARSET UTF8 **COLLATE**

UTF8\_GENERAL\_CI;

1. 用户信息表：用户的主要信息，包括用户编号、用户名，登录时使用的邮箱和密码。要求用户名和邮箱唯一。

**create** **table** **if** **not** **exists** `user\_main` (

`id` int(11) **unsigned** **NOT** NULL auto\_increment,

`name` varchar(20) **NOT** NULL **UNIQUE** **DEFAULT** '',

`email` varchar(50) **NOT** NULL **UNIQUE** **DEFAULT** '',

`password` char(32) **NOT** NULL **DEFAULT** '',

**PRIMARY** **KEY**(`id`)

) **default** charset=utf8;

1. 用户详细信息表：包括用户性别、账户余额、账户创建时间和上一次更新时间。表中id是此表的主码也是外码。

**create** **table** **if** **not** **exists** `user\_detail` (

`id` int(11) **unsigned** **NOT** NULL,

`sex` varchar(6) **DEFAULT** '',

`balance` DOUBLE **NOT** NULL **DEFAULT** 999.9,

`create\_time` int(11) **unsigned** **NOT** NULL **default** 0,

`update\_time` int(11) **unsigned** **NOT** NULL **default** 0,

**PRIMARY** **KEY** (`id`),

**FOREIGN** **KEY** (`id`) **REFERENCES** user\_main(`id`)

) **default** charset=utf8;

1. 商品分类表：包括分类的编号、分类名以及该分类的备注。要求分类名唯一，因此候选码可以是分类号或者分类名。

**CREATE** **TABLE** **IF** **NOT** **EXISTS** category (

category\_no INT **UNSIGNED** **NOT** NULL AUTO\_INCREMENT,

name VARCHAR(20) **NOT** NULL **UNIQUE**,

category\_comment VARCHAR(100) **NOT** NULL,

**PRIMARY** **KEY** (category\_no)

) **default** charset=utf8;

1. 物品信息表：包括物品号、物品名、详细描述、价格、数量、收藏人数和分类编号。要求物品名唯一。物品号是主码，分类编号是外码。

**CREATE** **TABLE** **IF** **NOT** **EXISTS** product (

product\_id INT **UNSIGNED** **NOT** NULL AUTO\_INCREMENT,

product\_name VARCHAR(50) **NOT** NULL **UNIQUE**,

detail TEXT **NOT** NULL,

price DOUBLE **NOT** NULL,

quantity INT **UNSIGNED** **NOT** NULL,

likes INT **UNSIGNED** **NOT** NULL **DEFAULT** 0,

category\_no INT **UNSIGNED** **NOT** NULL,

**PRIMARY** **KEY**(product\_id),

**FOREIGN** **KEY** (category\_no) **REFERENCES** category(category\_no)

) **default** charset=utf8;

1. 收藏表：用户对某件物品的收藏信息，包括物品号和用户号，候选码是全码，物品号、用户号均是外码。要求用户被删除或者产品被删除时，此表中有关信息也被删除。

**CREATE** **TABLE** **IF** **NOT** **EXISTS** favorites (

product\_id INT **UNSIGNED** **NOT** NULL,

user\_id INT **UNSIGNED** **NOT** NULL,

**PRIMARY** **KEY** (product\_id, user\_id),

**FOREIGN** **KEY** (product\_id) **REFERENCES** product(product\_id) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY** (user\_id) **REFERENCES** user\_main(id) **ON** **DELETE** **CASCADE**

) **default** charset=utf8;

1. 用户与产品联系表：此表记录每个用户发布的商品，包括用户号和产品号，候选码是全码，物品号、用户号均是外码。要求用户被删除或者产品被删除时，此表中有关信息也被删除。

**CREATE** **TABLE** **IF** **NOT** **EXISTS** user\_product (

user\_id INT **UNSIGNED** **NOT** NULL,

product\_id INT **UNSIGNED** **NOT** NULL **UNIQUE**,

**PRIMARY** **KEY** (user\_id, product\_id),

**FOREIGN** **KEY** (user\_id) **REFERENCES** user\_main(id) **ON** **DELETE** **CASCADE**,

**FOREIGN** **KEY** (product\_id) **REFERENCES** product(product\_id) **ON** **DELETE** **CASCADE**

) **default** charset=utf8;

1. 物品评论表：有评论编号、用户号、产品号、评论内容、评论时间。评论编号是主码，用户号和产品号均是外码，产品被删除时，有关评论也会被删除。

**CREATE** **TABLE** **IF** **NOT** **EXISTS** product\_comment (

comment\_id INT **UNSIGNED** **NOT** NULL AUTO\_INCREMENT,

user\_id INT **UNSIGNED** **NOT** NULL,

product\_id INT **UNSIGNED** **NOT** NULL,

content VARCHAR(500) **NOT** NULL,

comment\_time DATETIME **NOT** NULL,

**PRIMARY** **KEY** (comment\_id),

**FOREIGN** **KEY** (user\_id) **REFERENCES** user\_main(id),

**FOREIGN** **KEY** (product\_id) **REFERENCES** product(product\_id) **ON** **DELETE** **CASCADE**

) **default** charset=utf8;

1. 订单表：包括订单号、买家用户号、产品号、下单时间、数量、成交价格、订单状态。订单号是主码，买家用户号和产品号是外码。

**CREATE** **TABLE** **IF** **NOT** **EXISTS** orders (

order\_id INT **UNSIGNED** **NOT** NULL AUTO\_INCREMENT,

buyer\_id INT **UNSIGNED** **NOT** NULL,

order\_time DATETIME **NOT** NULL,

product\_id INT **UNSIGNED** **NOT** NULL,

quantity INT **UNSIGNED** **NOT** NULL,

price DOUBLE **NOT** NULL,

status VARCHAR(10) **NOT** NULL,

**PRIMARY** **KEY** (order\_id),

**FOREIGN** **KEY** (buyer\_id) **REFERENCES** user\_main(id),

**FOREIGN** **KEY** (product\_id) **REFERENCES** product(product\_id)

)**default** charset=utf8;

# 触发器的定义与实现

1. 订单状态改变触发器：当订单状态由“未完成”变为“已取消”时，调用取消订单过程，当订单状态由“未完成”变为“已完成”时，调用确认订单过程。订单状态变为“已完成”或者“已取消”后，不能再改变，这一约束由应用程序来保证。

**CREATE** **TRIGGER** t\_order\_status\_changed

AFTER **UPDATE** **ON** orders

**FOR** **EACH** ROW

BEGIN

**IF** new.status='已取消' **AND** old.status='未完成' **THEN**

**CALL** cancel\_order(new.buyer\_id, new.product\_id);

**ELSEIF** new.status='已完成' **AND** old.status='未完成' **THEN**

**CALL** confirm\_order(new.buyer\_id, new.product\_id);

END **IF**;

END;

1. 物品收藏触发器：当添加收藏时，更新对应产品信息表中的收藏数量likes。其值应加1。

**CREATE** **TRIGGER** t\_add\_favorite

AFTER **INSERT** **ON** favorites

**FOR** **EACH** ROW

BEGIN

**UPDATE** product SET likes=likes+1 **WHERE** product\_id=new.product\_id;

END;

1. 取消收藏触发器：当取消收藏时，更新对应产品信息表中的收藏数量likes，其值应减1。

**CREATE** **TRIGGER** t\_cancel\_favorite

AFTER **DELETE** **ON** favorites

**FOR** **EACH** ROW

BEGIN

**UPDATE** product SET likes=likes-1 **WHERE** product\_id=old.product\_id;

END;

# 存储过程的定义与实现

1. 创建订单过程：该过程有4个参数，输入的参数分别是买家用户号、物品号、购买数量，输出的参数是订单创建的结果。

插入订单表项前需要先检查买家账户余额是否足够购买所需物品、物品数量是否满足买家需求，如果任一个条件不成立，则不生成订单，返回出错信息“买光了”或者“余额不足”。如果符号创建订单的条件，则将订单表项插入订单表，更新物品余量和买家账户余额。如果买家收藏了该物品，则将该物品从收藏中删除。

**CREATE** **PROCEDURE** create\_order(**IN** bid INT **UNSIGNED**, **IN** pid INT **UNSIGNED**,

**IN** qty INT **UNSIGNED**, **OUT** msg VARCHAR(10))

BEGIN

**DECLARE** oprice DOUBLE;

**DECLARE** oquantity INT **UNSIGNED**;

**DECLARE** nquantity INT **UNSIGNED**;

**DECLARE** obalance DOUBLE;

**DECLARE** nbalance DOUBLE;

**DECLARE** userid INT **UNSIGNED**;

**SELECT** price, quantity **INTO** oprice, oquantity **FROM** product **WHERE** product\_id = pid;

**SELECT** balance **INTO** obalance **FROM** user\_detail **WHERE** id=bid;

SET nbalance = obalance – oprice \* qty;

SET nquantity = oquantity - qty;

**IF** nquantity < 0 **THEN**

SET msg = '卖光了';

**ELSEIF** nbalance < 0 **THEN**

set msg = '余额不足';

**ELSE**

**INSERT** **INTO** orders **VALUES** (null, bid, **CURRENT\_TIME**(), pid, qty, oprice, '未完成');

**UPDATE** product SET quantity=nquantity **WHERE** product\_id=pid;

**UPDATE** user\_detail SET balance=nbalance **WHERE** id=bid;

**SELECT** user\_id **INTO** userid **FROM** favorites **WHERE** product\_id=pid **AND** user\_id=bid;

**IF** !ISNULL(userid) **THEN**

**DELETE** **FROM** favorites **WHERE** product\_id=pid **AND** user\_id=bid;

END **IF**;

SET msg = 'success';

END **IF**;

END;

1. 取消订单过程：过程有两个输入参数，分别是买家用户号和物品号。可以通过设置订单的状态为“已取消”，由触发器调用此过程完成订单的取消。取消订单将创建订单时修改的产品余量和买家账户余额恢复到之前的状态。

**CREATE** **PROCEDURE** cancel\_order(**IN** bid INT **UNSIGNED**, **IN** pid INT **UNSIGNED**)

BEGIN

**DECLARE** oprice DOUBLE;

**DECLARE** qty INT **UNSIGNED**;

**SELECT** quantity, price **INTO** qty, oprice **FROM** orders **WHERE** buyer\_id=bid **AND** product\_id=pid;

**UPDATE** user\_detail SET balance=balance+oprice\*qty **WHERE** id=bid;

**UPDATE** product SET quantity=quantity+qty **WHERE** product\_id=pid;

END;

1. 确认订单过程：过程有两个输入参数，分别是买家用户号和物品号。可以通过设置订单的状态为“已完成”，由触发器调用此过程完成订单的确认。确认订单将买家支付款加入到卖家账户中。

**CREATE** **PROCEDURE** confirm\_order(**IN** bid INT **UNSIGNED**, **IN** pid INT **UNSIGNED**)

BEGIN

**DECLARE** oprice DOUBLE;

**DECLARE** seller\_id INT **UNSIGNED**;

**DECLARE** qty INT **UNSIGNED**;

**SELECT** quantity, price **INTO** qty, oprice **FROM** orders **WHERE** product\_id=pid **AND** buyer\_id=bid **LIMIT** 1;

**SELECT** user\_id **INTO** seller\_id **FROM** user\_product **WHERE** product\_id=pid;

**UPDATE** user\_detail SET balance=balance+oprice\*qty **WHERE** id=seller\_id;

END;

# 系统实现结果

本系统采用html + css + js + php的结构，前端使用bootstrap响应式与jquery框架，并做了响应式布局，使网站在移动端也能够有良好的浏览体验；后端使用php作为开发语言，采用thinkphp框架以及其mvc设计模式开发，做到界面、控制器和模型分离。前后端采用ajax异步交互传送数据。

从功能实现上来看，本系统在完成了需求分析阶段的基本上所有功能以外，还从细节上对用户的访问动作和范围进行了限制，从一定程度上保证了系统的安全性和数据完整性。

以下是具体实现结果介绍：

1. 进入首页，用户可以在没有登录的情况下浏览分类商品列表，以及商品详情。 用户若点击购买或收藏，会提示没有登录。

图 1首页界面

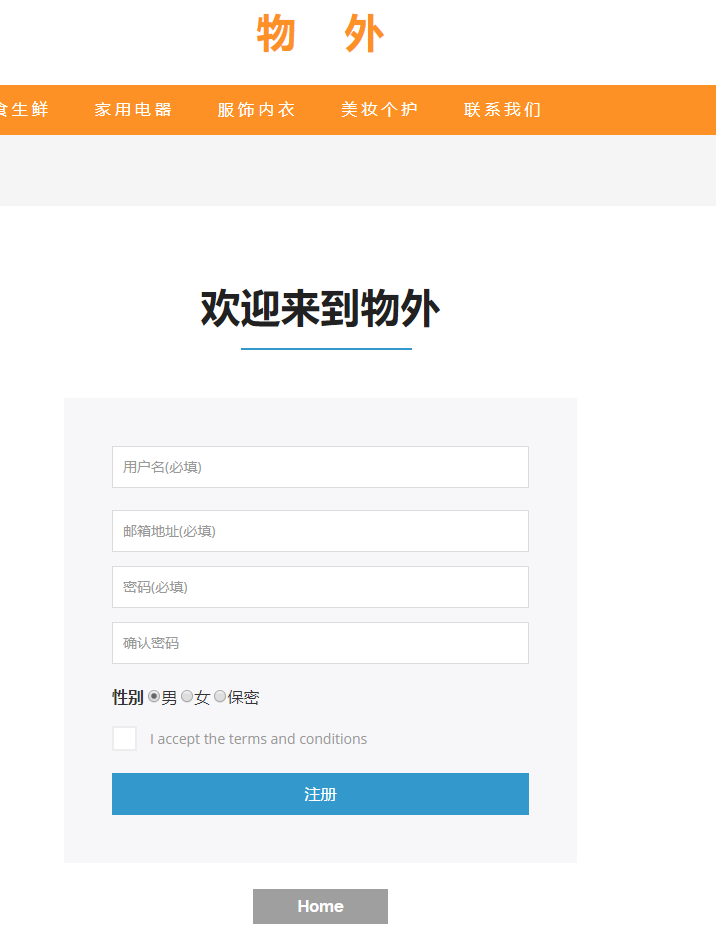
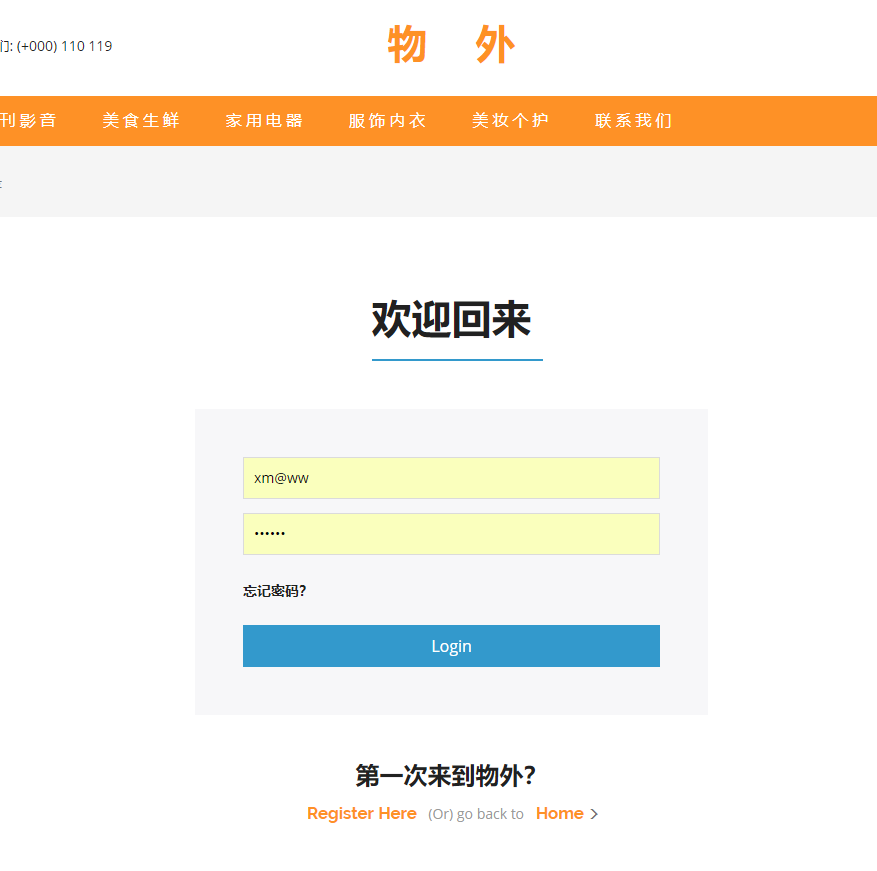
1. 用户可以进入注册界面进行注册，也可以进入登录界面登录，登录完成后自动跳转回到首页，并隐藏注册登录入口。若用户在登陆情况下输入URL强行访问登陆或注册页面会自动跳转至首页。用户登录后产生session会话，除非关闭一定时间或主动点击注销，用户会保持登录。

图 3登陆界面

图 2注册界面

1. 用户登录后能够访问个人中心。在个人中心，用户能够查看自己的基本信息、账户信息，还能够修改基本信息、充值以及修改密码。用户拥有一张默认头像，也可以上传自己的头像。

图 4个人中心界面

1. 用户可以将浏览的商品加入收藏。每个商品会统计收藏数作为商品人气。用户可以通过个人中心查看自己收藏商品的列表。

图 6订单状态界面

图 5收藏商品界面

1. 用户在商品详情页和收藏列表可以购买商品，点击购买后会产生一个订单，用户可以通过个人中心看到自己的订单信息和每个订单的状态，还能够确认订单完成和取消本次订单。
2. 商品详情页展示了卖家的信息，以及所有用户对于该商品的评论，评论显示了评论信息、评论时间以及对应的用户名。用户在登陆状态下能够对商品进行文字评论。

图 7评论列表界面

图 8商品详情界面

1. 用户作为卖家可以进入后台管理，查看所属商品的信息，添加自己的商品，或者修改删除商品。

图 9后台商品列表

# 总结

**胡俊崧**：

通过本次数据库课程设计，我经历了一个完整的前后端系统开发流程。虽然这个系统并不复杂，功能也不尽完善，但还是让我在一个较短的周期内学到了很多知识和技巧。

在这个项目中，我主要负责开发前端界面，也负责了一部分后端的中间交互。在这次课程设计之前我完全不了解从前端到后端再到数据库之间的整个系统如何运作，然而这次课程设计打开了我通往新知识的打门。在开发过程中我学习了从基础的前后端语法知识再到各种框架的结构以及思想，学到了从前端的网页布局、事件动画、网络请求到后端的请求处理、会话控制、数据库访问等web开发的大量知识。

其次，我体会了团队协同开发的重要性，也积累了宝贵的经验。我们使用git托管项目迭代开发，采用MVC设计模式使前后端开发分离，再加上我们分工明确，积极交流，使我们能够在短时间内高效的地协同完成此次项目的搭建。

此外，这次课程设计也让我体会到数据库设计的重要性，一个设计规范的数据库可以在应用中让开发人员省去很多不必要的操作，良好的设计能够保证数据库的安全性和数据完整性，从而使整个项目更加健壮。

总的来说，体验了这样一个完整的项目设计，我受益匪浅。

**陈治齐：**

1. 理论与实践

理论与实践缺一不可，二者相辅相成。这学期数据库是选修课，因此有的同学选了理论课没有选课程设计。我认为，理论和实践相结合或许可以学习得更深入，理解得更透彻。拿我自己的经历举个例子，当学到数据库设计一章时，我还没有开始做课设的数据库设计。阅读这一章的感觉就是没有感觉，书上描述得头头是道，看了之后觉得是那么回事儿，却给人以虚无缥缈的感觉。当我真正开始做设计时，却又觉得无理可依，无据可寻。再次翻开书才发现，原来做法在书上已经写得很清晰了。顺着书上的思路从需求分析、概念结构设计到逻辑结构设计，这一路走下来，对之前学习的内容有了更深入的理解，同时也体会到了理论和实践的微妙差别。然而，这个设计过程不是一帆风顺的。

1. 设计与实现

设计的过程也让我感到苦恼。一来是没有经验，以前从未做过这样的设计，所以对自己的设计感到些许怀疑，总是自问这样做可不可行呢。二来是对所用的语言工具不熟悉，需要现学现用，这也对设计的可行性有一些影响。到了实现时才发现，大部分设计都是可行的，而且最后确实是按照设计那样实现的。只是一些细节之处，设计的时候要么考虑得太复杂，要么考虑得太简单，实现起来有一定难度。最后的结论就是不要忽视设计，尽管有时觉得设计与理论一样“虚”，难以捉摸，实际上不是的，设计从整体的宏观的层面出发，指出了要实现什么，实现才进入细节，考虑怎么实现。如果没有设计作为指导，陷入细节中，容易无法自拔，忘记初心。

1. 合作与交流

与其他核心专业课程的课设不同，数据库课设是由两人共同完成。从前还没有真正的与别人合作过，这是第一次体验。我要感谢我的同伴。首先，在我毫无头绪的时候他做出了表率。俗话说，完事开头难，我不知道该从何做起，如何协作时，我的同伴首先把整个应用的框架建设好，然后我们再进一步完善实现。在这一点上，我要向他学习。第二，在我举步维艰的时候同伴给了我足够的鼓励。我知道虽然他没有明确地表达出来，但我从他的行动能感受到，从细微的言语中体会到。

1. 数据库课设与其他课设的区别

数据库课设别具一格，在思路上就与其他课设不同。像计组、操作系统的课设是“有底线”，只有在底线之上才能通过课程。而数据库课设是“无上限”，可以根据自己的思路，做出各式各样、功能丰富的应用。从这一点上看，数据库课设的出发点就是积极的。

1. 如果我能再做一次

通过这次课设，我再次印证了我的拖延症是存在的。尤其是多人合作时，如果每个人都沉默，可能真得等到临死前才哀鸣。如果我能再做一次，我一定要早早地做准备，提前开始，哪怕前期进度再慢，也比没有进度好，看到进度就消除了焦虑。这学期的任务多，各种DDL集中在一起。另外一门课程，有同学在微信群里问老师能不能延期，声称如果多给一点时间可能做的效果更好。经过多次沉痛的领悟，我认为这个理由有8成是不现实的。所以，更好的办法是提前准备，提早开始。