内容摘要

在知识付费的今天，线上网课已经成为一种有效的营销方式，人们开始习惯于在零散的时间段内对某知识点进行学习，因而越来越多的线下实体班，也纷纷开展了线上的付费网课。然而，开展线上网络班的形式多种多样，人们能在网络上学习的方式也大相径庭，但归根结底，最好是能够以一种高效便捷的方式来进行对知识的获取才符合当下在碎片化的时间中人们能够补充知识点的需求。

为了能够让学生和老师们更好体验到“快捷便利”的知识付费服务，本文设计了一款基于微信小程序的微课系统。其主要内容包括“各类课程”，“购物车” “课程付费”三大顺序模块；学生先会看到“各类课程”，这些课程就是已经上传好的存在于服务器数据库内的课程，点开课程会看到各种目录，教师信息，课程评价等；而“购物车”功能用于多课程同时购买；当选择好课程后进行最后的“课程付费”，就完成了一次便捷的知识付费的过程。

关键词：微信小程序 网课 知识付费 碎片化时间

Abstract

Today, when knowledge is paid for, online courses have become an effective marketing method. People begin to get used to learning certain knowledge points in scattered time periods. As a result, more and more offline entity classes have also started online paid online courses. However, there are various forms of online classes, and the ways people can learn on the Internet are also very different. However, in the final analysis, it is best to acquire knowledge one by one in an efficient and convenient way to meet the current demand that people can supplement knowledge points in the fragmented time.

In order to enable students and teachers to better experience the "fast and convenient" knowledge payment service, this paper designs a micro-lesson system based on WeChat applet. Its main content includes "various courses", "shopping cart" and "course payment" three major sequence modules; Students will see "various courses" first. These courses are already uploaded and exist in the server database. Click on the courses to see various directories, teacher information, course evaluation, etc. The "shopping cart" function is used for simultaneous purchase of multiple courses. When the course is selected and the final "course payment" is made, a convenient process of knowledge payment is completed.

Key words：WeChat applet　Online classes　Knowledge payment　Fragmentation time

目 录

第 1 章 绪论

1.1 研究背景

随着人们对碎片化时间下对知识的汲取需求的增多，以及互联网+时代的发展，网络上出现了一种新型的知识付费模式，即互联网+网课，简称微课或云课堂。这种微课用于人们在碎片化的时间下能够对已学过的知识中的某一知识点进行巩固或补充。

在互联网+网课的强烈冲击下，加上昂贵的地租以及各种成本，大多数线下实体班经济发展大不如前，甚至难以站住脚而走向倒闭；而在网络上开展线上网课所需要的成本最低只需要一台好的录制设备，仅此而已。所以，大多数线下实体班纷纷投入了线上网课教学的洪流中，知识付费的云课堂开始爆炸性增长，最知名的网课平台有 CCTALK、知乎、腾讯课堂、网易云课堂、慕课等。其中 CCTALK 甚至走向了国际化网课教程平台。

1.2 研究的目的及意义

古人云：学不可以已，自古以来人们对知识的渴求就没有停下来过，所以才会有我们如今精彩纷呈的世界存在。随着移动互联网时代的到来，微信小程序成为网络时代最具潜力的一匹多金黑马，而教育行业也是永垂不朽，该微课小程序为当下碎片化时代的人们提供便利快捷的一种学习方式，让人们能够在零散的时间段内能够做到对自己感兴趣或了解的领域进行查漏补缺，充实巩固。

微信小程序是一种即开即用，方便快捷的互联网新兴产品，本次研究的目的在于微信小程序的方便快捷对于知识付费的网课的有效结合，该小程序主要有三大模块，最最重要的“各类课程”模块，在首页上会有各种精挑细选的优质微课，让感兴趣的或想了解的人进行挑选，“购物车”模块用于多课程购买，是考虑到有些人感兴趣的课程多种多样，加入购物车后再三思考后会筛选出自己想要的课程，然后才进行最后的“付费”模块，就可以开始网课的视频学习。

第 2 章 研究现状

2.1 简析互联网+网课产生与发展的原因

传统的课程是以实体班的形式，即实地，实物，设身处地地在某个场所里学习，可能是老师们申请的学校放假时空置的教室，也可能是企业提供已经租赁好的房间，当然也有可能是老师自身的家里，但是无论如何，都是要前往某个地方然后进行上课的；而当人们发现可以在互联网上通过视频面对面地教学上课，其实大多数课程也可以做到和实体班上课时所达成的效果，而且还不用大老远跑到某个地方去上课，这很好的节约了路程的时间，而且如果实体班因为场地的限制，能够教的学生是有限的，就好像你在 50 平米的房间内不可能容的下 300 个学生吧；而在互联网上进行网络课程的教学，可以说很好的打破了这种限制，别说 300 个学生同时教学，3000 个都不在话下（在服务器容许的情况下）。不仅如此，线上网课，还解决了老师地租的成本，学生路程的路费等等，这可以说是互联网+网课带来的便利；既然在网上学习和实体班其实有差不多的成效，那么在网上学习不就可以了吗？因为这种价值观念，所以互联网+网课的产生以及快速发展也就是必然的事情了，这一切都是为了便利和高效以及节约成本。当然，对于一些特殊课程，最好还是去上实体班比较好，例如古典钢琴课，油画班等等，没有名师在你身旁手把手指导，效果是差强人意的。但这种课程只占小部分，大多数课程其实都可以以网络的形式进行教学还不会影响教学质量的。

2.2 网络教学的形式

2.2.1 直播

直播的形式是比较火的，从以前的娱乐直播，到现在的教育直播，因其可以实时互动和反复观看，在线直播受到许多师生的喜爱。例如 CCTALK，腾讯课堂，不过这些教学平台一般直播完后会给付费用户留下录播以便重复观看，会有大概半年或者一年的时间限制，不过聪明的网友一般会把录播的视频自己在录一遍，就变成永久性的了。

2.2.2 录播

录播就不能像直播那样，做到实时互动，不过录播还是有直播无法比拟的优势的，看录播的人群可以针对某一知识点来学习或巩固，而且不像直播，错过了内容就错过了，录播的话可以随时暂停，学习的自由度会相比直播要高，如果是短时间的录播学习那么比较适合那些碎片化时间的人群，他们可以在有限的时间内学习和

巩固某知识点。

2.2.3 直播面授

当多人上课的时候，例如大型教育活动，一个场地可能坐有好几百号人，而那些坐在后面的人可能无法看清前面老师讲的内容，而如果把这些人聚集在多媒体教室里，那么老师的屏幕就可以连接到学生的电脑或手机，就能很好地解决这种问题。而这种方式，就是直播面授了。直播面授这种形式比较少见，因为这种教学方式比较适合学校或培训机构，像云朵课堂就是这种直播面授的方式。

2.2.4 一对一

一对一网课形式就像线下家教一样，只不过由线下变成了线上，这种一对一的线上教育可以节约路程的时间，并保障学生的人身安全，毕竟在路上意外还是有可能发生的。

2.2.5 本章小结

我们已经了解了几种网课教学的形式，大家可以根据个人需要选择自身合适的网课教学模式。不过，本设计主要还是针对在微信小程序上的网课形式，所以还是以录播为主。

第 3 章 小程序分析

3.1 技术可行性

微信小程序诞生了 3 年多，光是在微信能找到的微信小程序就已经多达百万个，其中涵盖了各个领域，有图书类、美食类、交通类、教育类等等。可以肯定的是，微信小程序未来必将成为众多轻应用中的一匹黑马。而目前，微信小程序仍是互联网界讨论的热点，因为它的注册门槛，还有开发成本相比其他软件开发都要低的多，不管你是企业还是个人，只要你想做个小程序，都可以进行简单的尝试，因为腾讯官方提供了关于微信小程序的开发文档，里面的内容可用性很强，而且简单上手。

我在开发本次小程序中，遇到了两个问题，第一个是企业小程序需要营业执照，而我没有，所以本次设计选择了个人小程序，所以也就无法实现支付功能了；第二个是微信小程序的 API 需要获取的数据，因为 API 的数据一般都是别人已经制作好的，而自己需要的 API 没有，所以找到了 easy-mock 这个网站来解决自己需要的 API 的问题。不过因为 easy-mock 的服务器是在国外，所以有时候会出现数据因为网络延迟而无法显示。

3.2 SWOT 分析

3.2.1 Strength 优势

1.微信小程序不需要安装，即开即用，方便快捷。像以往的 APP 程序，一般都要经过下载安装，而微信小程序很好的做到了即开即用，而且还不会占据内存，能够节省内存空间，这是微信小程序相比以往 APP 程序的最大优势。

2.开发和维护的成本相比 APP 程序要低的多，而且开发的门槛也比 APP 程序要低。

3.微信小程序的页面是很简洁的，而且操作起来也比 APP 程序的开发要简单的多。

3.2.2 Weakness 劣势

1.微信小程序目前能支付的方式只支持微信支付和银行卡支付，像支付宝或京东白条目前还不支持。

2.微信小程序直接分享的话只能转发到群聊或者私聊，而无法直接转发到朋友圈。

3.相对于 APP 开发的程序，微信小程序的功能开发是有其局限性的，为了能做到“小”这一点，则必将舍弃某些 APP 才有的功能，必然会影响到部分用户对于微

信小程序的使用体验。

3.2.3 Opportunity 机会

1.微信小程序未来必将成为电商领域的一匹黑马，因为微信是一个巨大的社交平台，微商这种群体也已经层出不穷了。

2.小程序即开即用，不耗内存的特点，更能迎合碎片化人群的喜好，因为碎片化人群本身就不是长时间的使用某一 APP，但是 APP 又确实地在占用着内存，例如交通类程序，要买车票的时候才用，平常都是不怎么用到，卸载了以后要再次用到，就要重新安装，会很麻烦的。

3.2.4 Threat 威胁

微信小程序的网络安全隐患较大。因为即开即用的特点，有些诈骗人员用小程序进行诈骗时，客户无法提供有效的数据进行举报，而且举报了，诈骗人员还可以再开发个新的微信小程序，目前这还是一个问题。

第 4 章 小程序设计与实现

4.1 总体功能布局

此微信小程序的总体布局主要是仿照网易云课堂，突出简洁大方，带有 ins 简约风格。以白色为主要底色，辅以绿色，白色会给人舒服明亮清爽的感觉，再加上绿色护眼的暗示，让人们在浏览课程的时候尽量是以舒服舒适的感觉

4.2 小程序设计所需工具

在项目开发中选用好的工具能使得工作事半功倍。

1.微信开发者工具，是必须要用到的工具，用于对微信小程序的开发；

2.easy-mock，是用于获取 API 接口（因为没有数据库，所以数据要自己写，用于微信小程序云开发时对数据的获取）

4.3 功能结构

|-cloudfunctions 云函数

|-getMyCourse 获取我的课程

|-getCourseInfo 获取课程信息

|-getCart 获取购物车

|-miniprogram 项目模块

|-components 自定义组件

|-box-module 盒子

|-myCourse-module 我的课程

|-special-module 专题

|-utils 工具

|-indexMock 获取主页数据

|-viewContent 文本处理

|-pages 页面

|-account 账号

|-cart 购物车

|-confirm 确认订单

|-courseInfo 课程信息

|-myStudy 我的学习

|-index 首页

|-vant-weapp 有赞 vant 框架组件库

4.4 小程序云开发

4.4.1 微信小程序中自带的数据库

微信小程序有自带的数据库，一般都是以读写 JSON 文件的方法来存入数据库中，还有一种是通过在前端写数据并提交，例如一些评论或者图片的上传等等。在这次关于微课（云课堂）的微信小程序设计中，我是把事先写好数据的 JSON 文件导入了数据库中，而导入的过程中偶尔会出现内容报错，这是因为微信小程序中对于JSON格式的数据库的识别是有别于以前学过的类似于SQL的这种语法语句格式，在微信小程序开发文档下有特别强调，但是还是会出现报错，可以说，格式语法的细节真的是防不胜防，写代码的时候要尤其注意。

然而，各类课程的数据项是多出了我的预料的，所以这里我只定义了我所使用了的一部分。每个页面显示不同的课程。而其他课程其实只是空白页。

图 4.1 和图 4.2 是已存入数据库的数据，都是用 JSON 文件导入的。

4.1-数据库数据 A

"_id":W-rjM0dHASyEa0OB

"discount":70

"id":c3

"image":https://edu-image.nosdn.127.net/fa33d65b-bf6f-4b9c-8cb4-e08d4146c2b7.jpg?imageView&quality=100&crop=0_0…

"isSelected":false

"ownername":罗文益阅读课堂

"price":109

"time":永久有效

"title":25课通识阅读 带你打破知识边界

4.2-数据库数据 B

4. 4. 2 云函数

在创建云函数时，要先安装依赖，在终端中 yarn。创建云函数后要进行上传并部署才能够做到在云端运行。

图 4.3 是云函数简单配置的代码。

```
// 云函数入口文件
const cloud = require('wx-server-sdk')
cloud.init()const db = cloud.database();const course_cart = db.collection('

// 云函数入口函数
exports.main = async (event, context) => {
    let info = course_cart.where({ id: event.id }).get();
    return info
}
```

4.3-云函数配置

javascript 作为单线程语言，虽然逻辑运行时简单直接，但在数据交互的过程中，容易出现数据无法显示的问题，这是因为异步数据交互是一个异步问题，而异步的问题只要用 Promise 就可以解决了，图 4.4 是异步处理的代码

```
const promiseArr = cart_coursesId.map(course => new Promise((resolve, reject) =>{
   wx.cloud.callFunction({
   name: 'getCart',
   data: { id: course}    })
   .then(res =>{
       resolve(res.result.data[0]);
   })
 }))
   Promise
   .all(promiseArr)
   .then(data => {
       console.log(data);
       wx.hideLoading()
       this.setData({
           courses: data
       })
    })
```

4.4-异步处理的代码

4. 4. 3 云课堂设计的部分页面

1.首页-部分课程，如图 4.5

4.5-首页

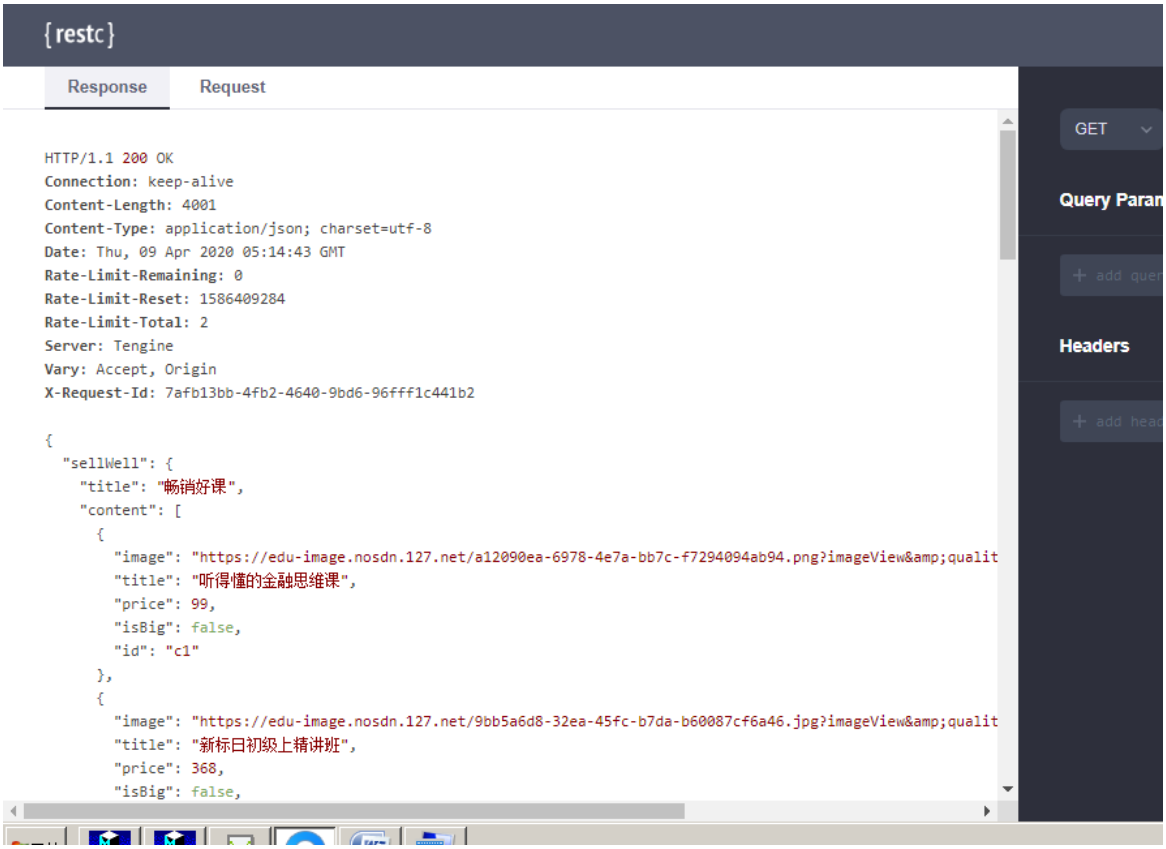
这一部分主要是前端的课程盒子，以及关于课程部分的后端数据的获取。

关于主页的课程盒子，此处使用弹性布局的 justify-content 与 flex-wrap 就能十分轻松地达到效果，使用的是如图 4.6 的这样类型的代码。

```
<view class="divide"></view>
<view class="box-module">
    <text class="box-title">{{data.title}}</text>
    <view class="courses">
        <block wx:for="{{data.content}}" wx:key="index">
            <block wx:if="{{item.isSpecial}}">
                <view class="course_special">
                <image lazy-load="{{true}}" class="course-image " src="{{item.image}}"
                </view>
            </block>
            <block wx:else>
                <view class="{{item.isBig ? 'course_big': 'course'}}">
                    <navigator url="../courseInfo/courseInfo?id={{item.id}}">
                    <image lazy-load="{{true}}" class="course-image " src="{{item.imag
                    <text class="course-title">{{item.title}}</text>
                    <text class="course-price" wx:if="{{item.price > 0}}">¥{{item.pri
                    </navigator>
                </view>
            </block>
        </block>
    </view>
```

4.6-盒子页面布局的代码

而后端对于数据的获取，是运用了云开发中 API 获取数据的方法，用到了一种名为“easy-mock”的网站，在这个网站中，可以自己写 JSON 格式的数据，而且还可以自制接口，达成云开发中对于 API 数据的获取，这一直是困扰我的地方，因为网上存在的 API 一般都是别人已经做好的，对于数据的获取没有那种自主性，而在有了 easy-mock 这个网站后，自主的制作 API 来获取数据就成为了一种可能，如图 4.7



4.7-自制 API 接口

2.我的学习页面，看图 4.8

4.8-“我的学习”页面

图 4.8 的页面已经是我购买后（其实不是真的购买，只是购买后会显示的效果）而存在的课程，是通过对数据 status 的判断来决定显示对应的部分，如下图 4.9。

```
<ul class="status">
    <li>
      <text class="{{status == '1' ? 'active':''}}" bindtap="showStatus" data-status="
    </li>
    <li>
      <text class="{{status == '2' ? 'active':''}}" bindtap="showStatus" data-status="
    </li>
 </ul>


<block wx:if="{{status == '1'}}">
</block>
<block wx:else>
</block>
```

4.9-已购课程的判定的代码

而给相应的标签设置对应的 data-status，再将修改的函数绑定到 bindtap 上，一个最简单的 MVVM 例子就实现了，如下图 4.10 的代码。

```
showStatus: function(e){

    let status = e.currentTarget.dataset.status;
    this.setData({
      status:status
    })
  }
```

4.10-简单 MVVM 例子

而关于课程的获取就是云函数的用武之地了，如下图 4.11

```
onShow:function(){
    wx.cloud.callFunction({
        name: 'getMyCourse',
    }).then(res=>{
        this.setData({
            my_courses: res.result.data
        })
    })
}
```

4.11-课程获取

my_courses 中有各种课程的数据，通过云函数的调用，以及 wx：for 把数据输出到前端，就可以看到课程内容，图 4.12 是前端数据输出的代码。

```
<view wx:for="{{my_courses}}" wx:key="index">
    <myCourse-module image="{{item.image}}" title="{{item.title}}" cid="{{item.id}}">
    </myCourse-module>
</view>
```

4.12-前端数据输出

3.购物车页面，如下图 4.13

4.13-“购物车”页面

说真的购物车页面在设计的时候真的是遇到了很多难题，单单是对数据的获取就直接阻挡我大半的进度，而关于购物车中课程的相关数据，例如对于课程价格的获取，然后是导入如下代码所示的集合中：

totalPrice:0,

selectedId:[],

selectAllStatus: false

并通过对列表中数据的 isSelected 属性判断，然后来计算出总体课程的价格，如下图 4.14 的代码

```
getTotalPrice: function(){
    let courses = this.data.courses;
    let total = 0;
    let selectedId = [];
    for(let i=0;i<courses.length;i++){
        if(courses[i].isSelected){
            total += courses[i].price;
            selectedId.push(courses[i].id)
        }
    }
    this.setData({
        totalPrice:total,
        selectedId
    })
    console.log('[total]',total);
    console.log('[selected]',selectedId);
    return total;
}
```

4.14-总体价格计算

在购物车中选中的部分课程，运用的还是 MVVM 数据的绑定，且选中后重新计算一遍总体课程的价格价，如下图 4.15

```
  selectedList:function(e){
    const index = e.currentTarget.dataset.index;
    const courses = this.data.courses;
    courses[index].isSelected = !courses[index].isSelected;
    this.setData({
        courses
    })
    this.getTotalPrice();
}
```

4.15-部分课程的价格计算

而购物车中的全选，每次进行 selectAll 操作，先将 selectAllStatus 改为！selectAllStatus，（全选=>全不选 || 全不选=>全选 ），之后将所有数据的 isSelected 属性统一为 selectAllStatus 的当前状态，如图 4.16 所示代码。

```
selectAll:function(e){
    let courses = this.data.courses;
    let selectAllStatus = this.data.selectAllStatus;
    selectAllStatus = !selectAllStatus;
    courses.forEach((item, index) => {
        item.isSelected = selectAllStatus
    })
    this.setData({
        courses,
        selectAllStatus
    })
    this.getTotalPrice();
}
```

4.16-全选课程

最后获取的课程数据将传递到下一个界面，即订单页面，将选中的数组 selectedId 作为对象通过 navigator 传入下一个页面。

4.订单页面

先获取从购物车那传递来的数据，然后完成订单，如图 4.17 的逻辑代码

```
let orderIds = options.ids.split(",");
console.log(orderIds)
this.setData({
    userInfo : app.globalData.userInfo,
    orderIds: orderIds
})
```

4.17-完成订单

在完成订单后，将购买的课程存入数据库内，如图 4.18 和图 4.19

```
wx.cloud.init()
const db = wx.cloud.database();
const my_courses = db.collection('my_courses');
```

4.18-存入数据库

```
addMyCourse(){
    const orders = this.data.orders;
    for(let order of orders){
      let myCourse = {
        title:order.title,
        image:order.image,
        id:order.id
    }
    console.log(myCourse);
    my_courses.add({data:myCourse})
    }
}
```

4.19-存入数据库

提交订单之后 数据库中的 my_courses 将会被更新，相应的，打开被购买的课程页面也会被更新，然后查询当前课程是否在购买的课程中，是的话将 isPaid 改为 true，如图 4.20。

```
wx.cloud.callFunction({
  name: 'getMyCourse',
  }).then(res=>{
              for(const my_course of res.result.data){
                  if(options.id === my_course.id){
                      let isPaid = true;
                      this.setData({
                          isPaid
                      })
                  return ;
              }
          }
      })
```

4.20-已购课程的判定

然后如果再次打开已购买的课程就会改成雪莱名言的提示语，如图 4.21

4.21-已改变的温馨提示语

而之前如果未登录的话是会出现“您还没有登录哦-请赶快登陆”这样的提示语，如图 4.22。

4.22-未登录时的温馨提示语

5.对于课程功能完整实现的步骤总结，即达成，选课程，进入购物车，购买的

全过程。

如图 4.23 和图 4.24，我先点开图 4.23 中的第一个课程后，会进入图 4.24 的界面，在图 4.24 中的最下栏有“加入购物车”，以及“加入学习”两个选项，当点击“加入购物车”后会提示添加成功的提示语（即进入后台的数据，就像我们平常在淘宝购物时添加购物车的效果是一样的），而点击“加入学习”将会进入直接越过“购物车”页面，直接进入“完成订单”的购买支付页面。

4.23-首页

4.24-某一课程

如图 4.25 和 4.26，图 4.25 就是“购物车”的页面，就像前文所说的有“全选”，以及“多选”的功能，点击“去结算”会跳转到图 4.26 的“确认订单”页面，即本次微信小程序设计中的最后一步，支付购买功能，因为我所使用的微信小程序是个人版的，所以其实并不具备真正的购买功能。