微信小程序O2O商城实操手册

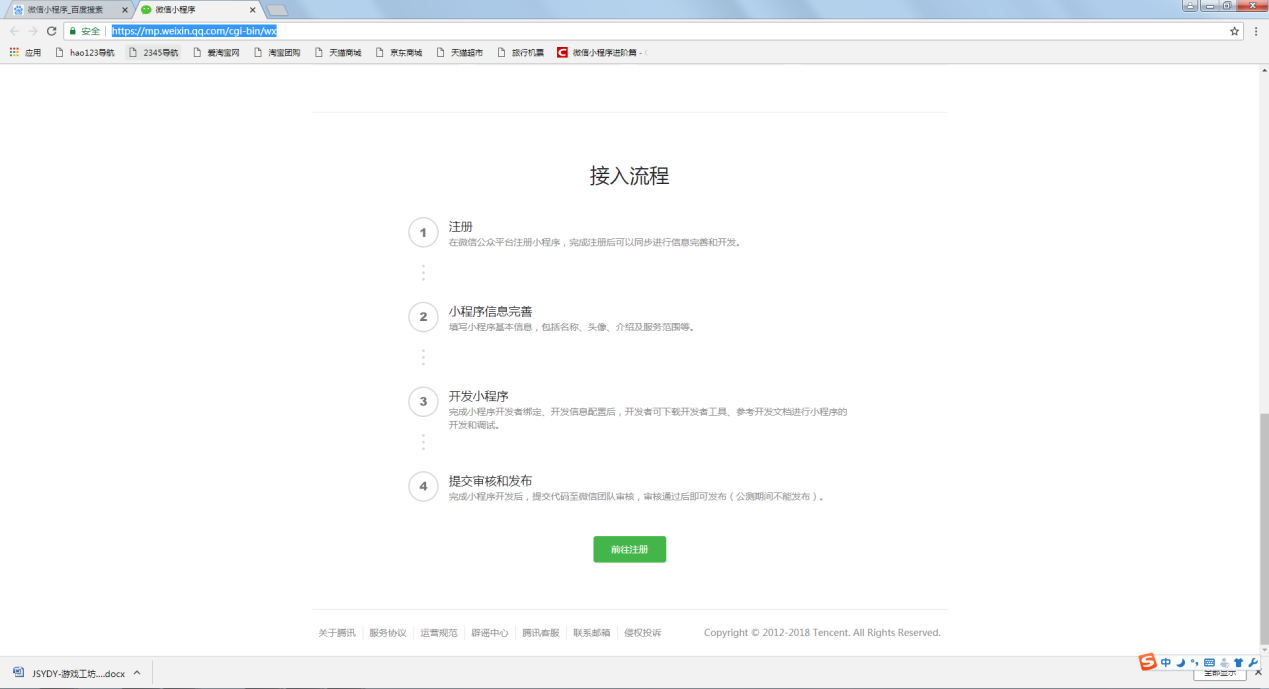
# 单元1：项目准备

## 任务1.1 注册微信小程序账号

### 1.客户端

（1）进入微信公众平台小程序模块：https://mp.weixin.qq.com/cgi-bin/wx。

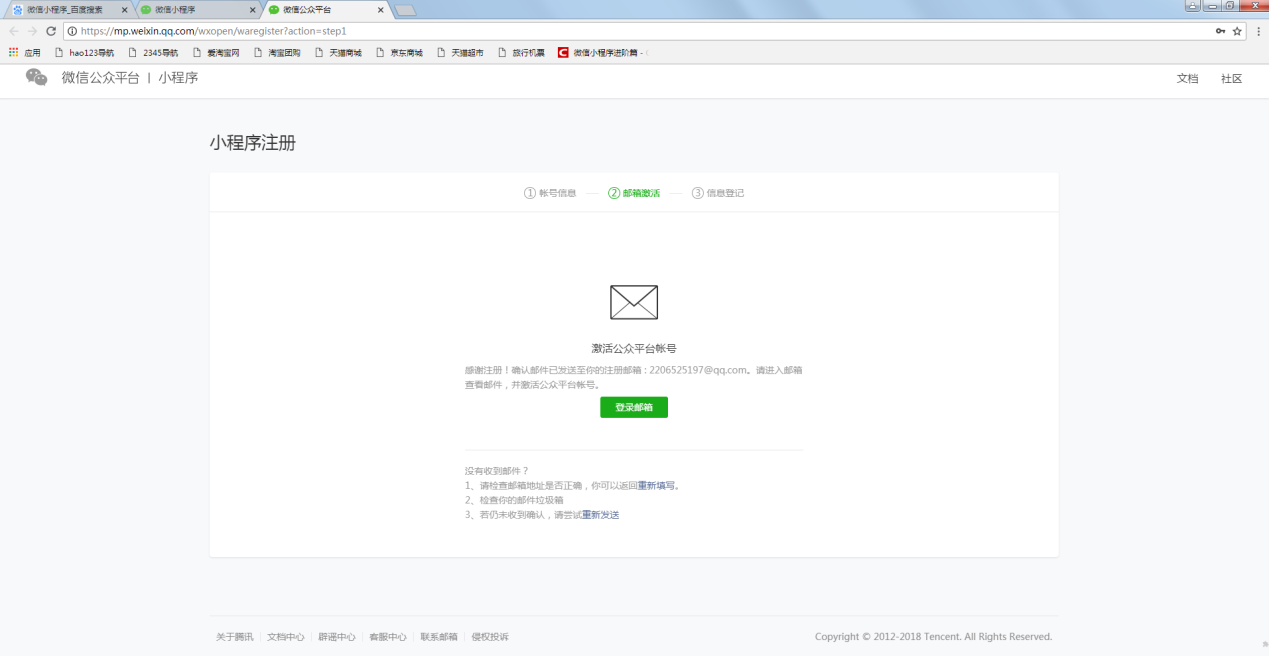
在页面最下方点击前往注册按钮。



（2）根据页面要求，填写完注册信息。



（3）根据页面提示，前往注册邮箱进行激活公众平台操作。

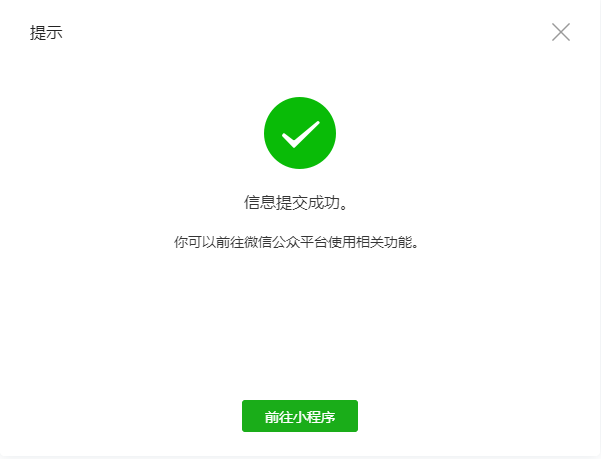


（4）激活邮箱以后，进入信息登记页面。选择合适的主体类型，进行相关信息的登记。后续操作以“个人”为主体类型进行演示。



点击页面最下方的继续按钮。

（5）根据提示，完成账号的申请。点击前往小程序按钮进入小程序配置页面。



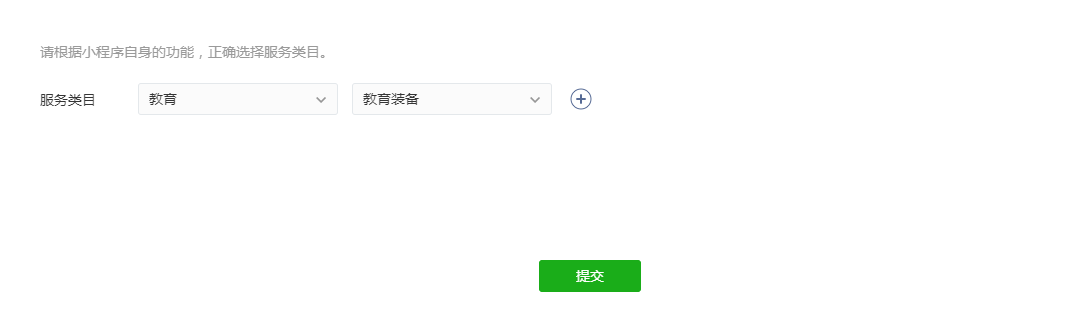
（6）点击填写按钮进入填写小程序信息页面。



（7）根据页面提示，完成信息的填写。



**注意：因为本项目虽然为小游戏，但是是以小程序开发方式实现开发，所以这里不能选择游戏类目，我们这以教育类目为例。**



（8）点击提交，就完成了小程序信息的录入。

### 2.商户端

与客户端相同的办法申请商户端的微信小程序账号。

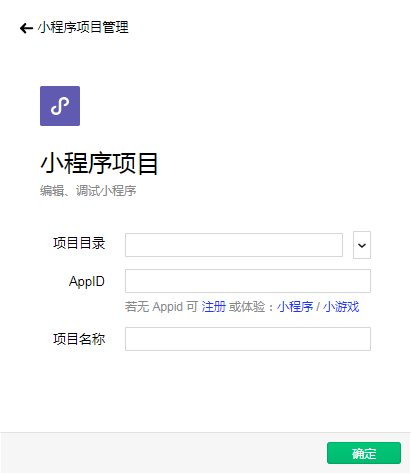
## 任务1.2 新建小程序

### 1.客户端

（1）选择小程序项目。



（2）根据提示填写项目信息。



**注意：AppID可以直接选择小程序/小游戏选项，但不能进行小程序发布的操作。**

查看AppID，进入开发者公众平台，点击如下图所示按钮。



下图所示就是我们需要用的AppID。

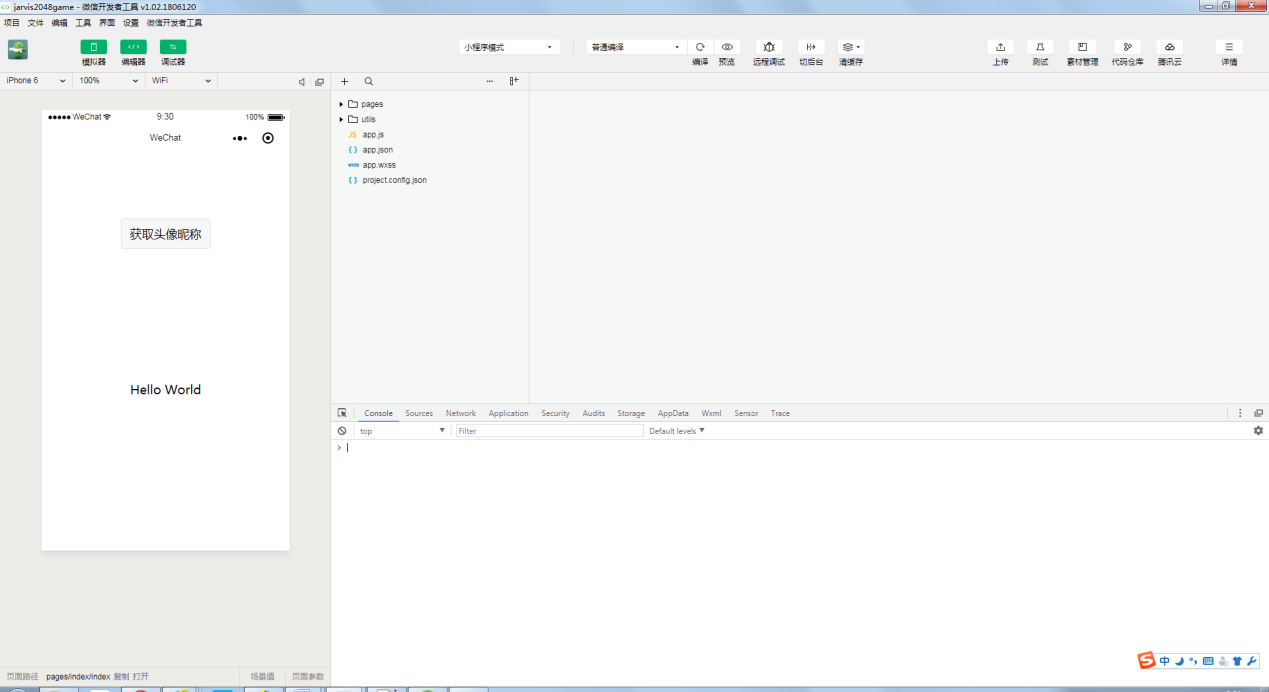


（3）当填写完所有信息之后，会出现下图所示样子。



**注意：勾选并且选择简历普通快速启动模板。**

（4）点击确认按钮，新建的第一个项目就完成了。



（5）整理初始代码，删除部分文件及代码

① 打开app.js文件，删除自带代码，只保留以下代码。

//app.js

App({

})

② 删除logs文件包。

③ 打开app.json文件，删除"pages/logs/logs"，保留以下代码。

{

"pages":[

"pages/index/index"

],

"window":{

"backgroundTextStyle":"light",

"navigationBarBackgroundColor": "#fff",

"navigationBarTitleText": "WeChat",

"navigationBarTextStyle":"black"

}

}

④ 删除index.wxml和index.wxss中所有代码。

⑤ 打开index.js文件，删除自带代码，只保留以下代码。

//index.js

//获取应用实例

const app = getApp()

Page({

})

### 2.商户端

与客户端相同的办法创建商户端的小程序项目。

## 任务1.3 服务器接口布置

所需资源：

1.项目包：wxxcx.war

2.数据库sql语句文件：dxshop.sql

准备工作：

1.安装好虚拟机，CentOS系统或者订购好腾讯云服务器CentOS，并且启动。

2.使用Xshell或者其他工具，能连接上服务器系统

3.使用WinSCP或者其他的FTP、SFTP客户端工具，能连接上服务器系统

4.使用Navicat或其他数据库连接工具，能连接数据库

实施：

一、进入linux系统进行处理：

（1）在线下载docker

yum update #保持最新的

yum install -y epel-release

yum install docker-io -y # 安装docker

systemctl enable docker.service # 开机启动

systemctl start docker # 启动docker服务

注意：

如果启动有问题，需要关闭SELinux，有时候还要关闭防火墙。

关闭SELinux(永久)：

打开 selinux 配置文件：vim /etc/selinux/config

修改 selinux 配置文件

将SELINUX=enforcing改为SELINUX=disabled，保存后退出，并且重启服务器

防火墙：

systemctl stop firewalld

systemctl disable firewalld

shutdown -r now #重新启动

systemctl status docker #查看是否启动状态

（2）docker安装Tomca、mysql 并进行相应的配置准备

① 安装tomcat

查找服务器的tomcat信息：

docker search tomcat



下载下来官方的镜像Starts最高的那个

docker pull docker.io/tomcat

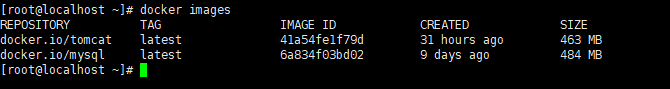
② 按同步骤安装mysql

docker pull docker.io/mysql

③ sql数据的导入

查看docker所有的镜像

docker images

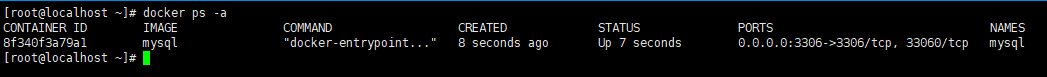


首先启动mysql

docker run -itd --name mysql -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=123456 mysql

查看运行情况：

docker ps –a



然后进入mysql容器：

docker exec -it 8f340f3a79a1 bash

（8f340f3a79a1为容器的ID）

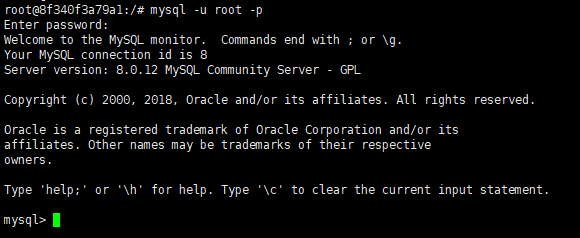
命令行的前面变成如下，表明已经进入容器：



进行相应的配置：

mysql –u root –p

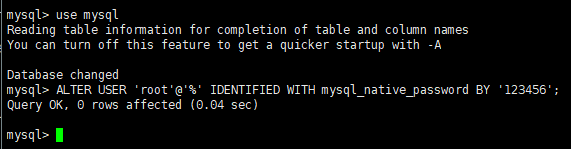
输入密码：123456，进入mysql，如图：



配置，允许远程使用root账户连接：

user mysql

ALTER USER 'root'@'%' IDENTIFIED WITH mysql\_native\_password BY '123456';



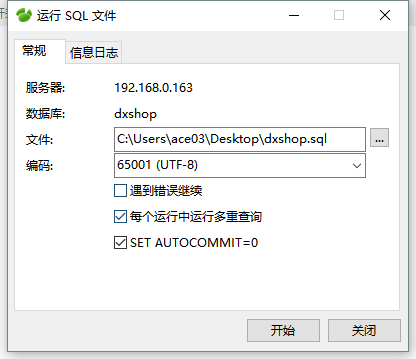
这样子，我们就可以使用mysql的连接工具，进行连接，然后进行相应的操作了。

创建数据库，已经插入测试数据：

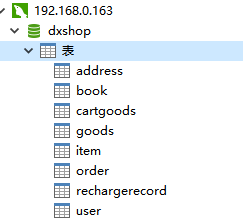


连接成功之后，创建数据库dxshop

对于数据库dxshop运行sql文件，dxshop.sql



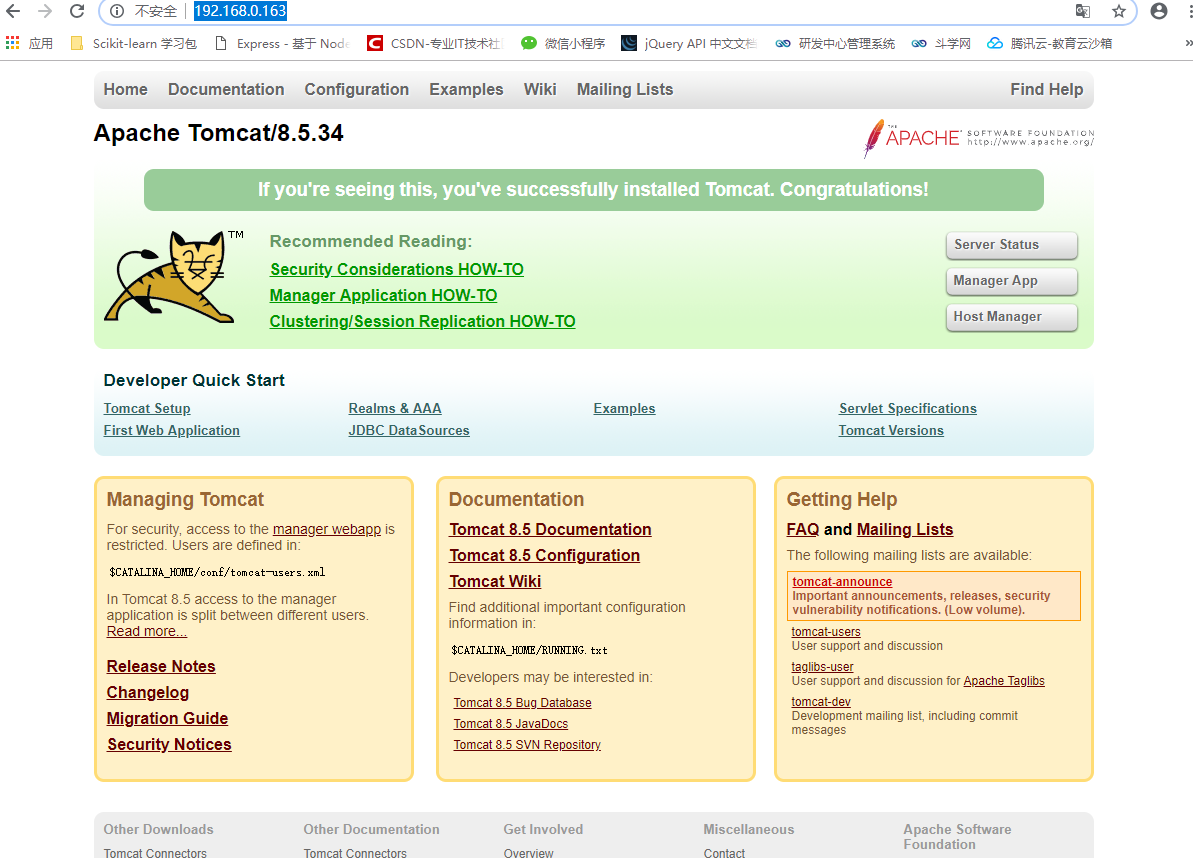
运行成功之后，有如下的数据表结构：



④ 启动tomcat，连接上mysql

docker run --name myweb -p 80:8080 -d --link mysql:db tomcat

浏览器输入对应的网址，看下tomcat是否运行正常：



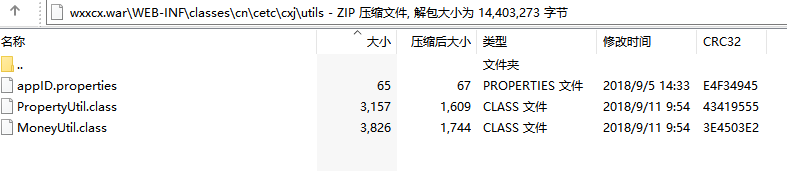
（3）将项目包上传tomcat容器中:

① 修改war包中的appID.properties配置文件，然后上传到centos系统的/usr/loacl中。

使用压缩文件打开war文件，找到appID.properties进行编辑，文件路径为：

wxxcx.war\WEB-INF\classes\cn\cetc\cxj\utils

修改文件内容，将APPID和SECRET修改为自己微信小程序对应的。



（双击指定文件，进行编辑保存即可）

② 将war包上传至tomcat容器：+

docker cp /usr/local/wxxcx.war 11d5a8405f66:/usr/local/tomcat/webapps

（11d5a8405f66是tomcat的容器ID）



查看，是否已经成功上传：

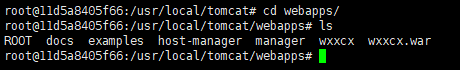
docker exec -it 11d5a8405f66 bash



cd webapps/

ls

如图，我们的项目包已经正常传送过去了。



③ 重启下我们tomcat容器：

docker restart 11d5a8405f66

浏览器中在原先的地址里面，加入我们想项目名，看到如图就说明已经部署完毕。



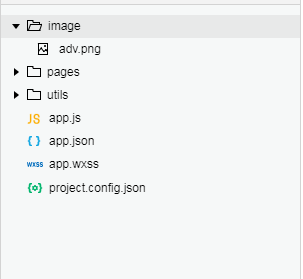
# 单元2 ：个人信息模块

## 任务2.1 授权登陆

### 1. 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）将资源包里image文件夹整个拷贝至项目中，项目结构如下：



（3）打开index.json文件，添加如下代码，禁止页面滑动：

{

"disableScroll":true

}

（4）打开app.json文件，修改navigationBarTitleText属性值为斗学商城，更新代码如下：

"navigationBarTitleText": "斗学商城"

（5）打开index.wxml文件，绘制启动页，添加如下代码：

<view class='container'>

<image class='img\_bg' src='../../image/adv.png' mode='scaleToFill'></image>

<button class='bt\_go' open-type="getUserInfo" bindgetuserinfo="getUserInfo" hidden='{{isTrue}}'>立即体验</button>

</view>

（6）打开index.wxss，为启动页添加样式，添加如下代码：

.img\_bg{

width: 100vw;

height:100vh

}

.bt\_go{

background: rgb(223, 10, 10);

color:#fff;

position: absolute;

bottom: 37vh;

padding: 0 50rpx;

border-radius: 10rpx;

}

（7）修改app.wxss文件，删除 padding: 200rpx 0;代码。

（8）打开app.json文件，添加网络请求超时设置， 顺序添加如下代码：

"networkTimeout": {

"request": 10000

"uploadFile":10000

},

（9）打开app.js文件，设置服务器域名地址，userInfo: null代码下添加如下代码：

host:'http://192.168.0.117:8080/wxxcx/'

***注意：此处需替换成自己布置的服务器域名地址。***

（10）添加按钮显示隐藏变量，在Page({代码下面添加如下代码：

data: {

isTrue:false

}，

（11）增加登陆请求方法，先获取用户信息，然后调用服务器登陆接口。增加login方法，添加如下代码：

login:function(){

wx.getUserInfo({

success: (u) => {

var that=this;

wx.login({

success: function (res) {

wx.showNavigationBarLoading()

wx.request({

url: getApp().globalData.host+'user/login',

method:'POST',

data: {

code: res.code,

name: u.userInfo.nickName,

headImage: u.userInfo.avatarUrl

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideNavigationBarLoading()

console.log(res.data)

if (res. data.code==200){

getApp().globalData.userInfo = res.data.data

console.log('登陆成功')

}else{

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

that.setData({

isTrue: false

})

}

},

fail:function(res){

wx.hideNavigationBarLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

that.setData({

isTrue: false

})

}

})

}

})

}

});

},

（12）为按钮添加getUserInfo单击事件，添加如下代码：

getUserInfo: function () {

this.setData({

isTrue: true

})

this.login();

}

（13）在onLoad事件监听中添加如下代码,实现用户的自动登陆：

onLoad: function () {

// 获取用户信息

wx.getSetting({

success: res => {

if (res.authSetting['scope.userInfo']) {

// 已经授权，可以直接调用 getUserInfo 获取头像昵称，不会弹框

this.setData({

isTrue: true

})

this.login();

}

}

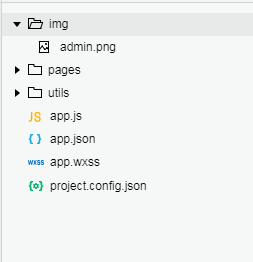
})

},

### 2. 商户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）将资源包里img文件夹整个拷贝至项目中，项目结构如下：



（3）打开index.json文件，添加如下代码，禁止页面滑动：

{

"disableScroll":true

}

（4）打开app.json文件，修改navigationBarTitleText属性值为斗学商城，更新代码如下：

"navigationBarTitleText": "斗学商城"

（5）修改app.wxss文件，删除 padding: 200rpx 0;代码。

（6）打开app.json文件，添加网络请求超时设置， 顺序添加如下代码：

"networkTimeout": {

"request": 10000

},

（7）打开app.js文件，设置服务器域名地址，userInfo: null代码下添加如下代码：

host:'http://192.168.0.117:8080/wxxcx/'

***注意：此处需替换成自己布置的服务器域名地址。***

（8）打开index.wxml文件，绘制登陆页面，添加如下代码：

<view class='container'>

<image class='title\_img' src='../../img/admin.png'></image>

<form bindsubmit="formSubmit">

<view class='login'>

<view class='user\_psw line'>

<text>用户名：</text>

<input placeholder='请输入用户名' placeholder-class='placeholder' name='user'></input>

</view>

<view class='user\_psw'>

<text>密码：</text>

<input placeholder='请输入密码' placeholder-class='placeholder' password name='psd'></input>

</view>

</view>

<button class='bt\_login'formType="submit">登陆</button>

</form>

</view>

（9）打开index.wxss文件，为登陆页面添加样式文件，添加如下代码：

/\*\*index.wxss\*\*/

.title\_img{

width: 100vw;

height: 130rpx;

margin-top: 15vh;

}

.container{

background-color: rgb(213, 223, 227);

height: 100vh;

justify-content:start;

}

.login{

background-color: #fff;

margin: 140rpx 40rpx 100rpx 40rpx;

}

.user\_psw{

padding: 20rpx;

display: flex;

align-items:center;

}

.user\_psw text{

font-size: 30rpx

}

.placeholder{

font-size: 30rpx

}

.line{

border-bottom: 1px solid rgb(182, 175, 175)

}

.bt\_login{

background: blue;

color:#fff;

width: 450rpx;

}

（10）打开index.js文件，为登陆按钮添加点击事件方法，登陆成功的时候记录下账号密码，实现下次自动登陆。在page字段下添加如下代码：

formSubmit: function (e) {

console.log('form发生了submit事件，携带数据为：', e.detail.value)

var name = e.detail.value.user;

var psd = e.detail.value.psd;

if (name=='jarvis' && psd=='123456'){

wx.setStorageSync('isLogin', true);

console.log('登陆成功')

}else{

wx.showToast({

title: "账号或者密码错误",

icon: 'none',

duration: 1000

})

}

}

***注意：此处实现的是模拟登陆，验证信息自由替换。后期也可做成网络接口请求模式。此处操作仅作参考。***

（11）添加onLoad方法，实现自动登陆效果，添加如下代码：

onLoad: function (options) {

if (wx.getStorageSync("isLogin")){

console.log('自动登陆成功')

}

}

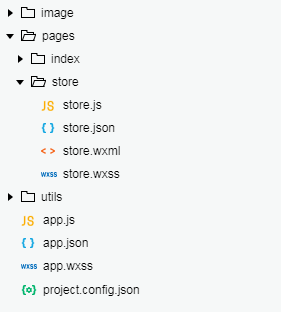
## 任务2.2 个人信息页面

### 客户端

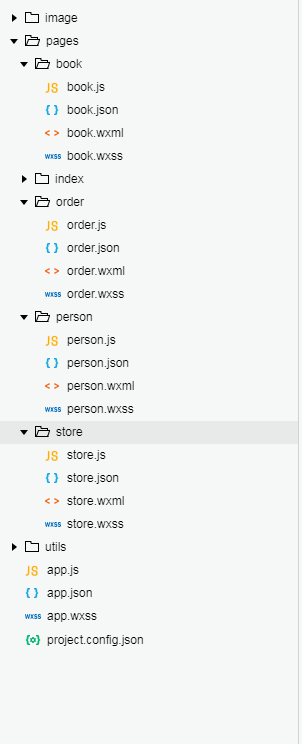
（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）构造底部tab页效果

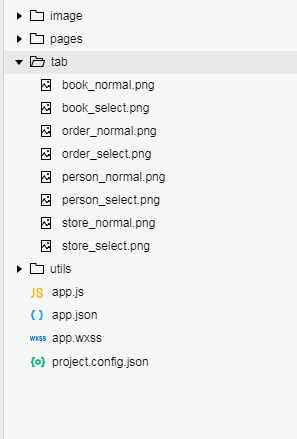
① 新建pages/store目录，在该目录下新建store页面。项目结构如下图：



② 相同的办法新建book、order、person三个页面。项目结构如下图所示：



③ 将资源包里tab文件夹拷贝到项目目录下，项目结构如下：



④ 打开app.json文件，添加tab页设置，顺序添加如下代码：

"tabBar": {

"borderStyle": "white",

"backgroundColor": "#ffffff",

"selectedColor": "#81c7d1",

"color": "#a6a6a6",

"list": [

{

"pagePath": "pages/store/store",

"iconPath": "tab/store\_normal.png",

"selectedIconPath": "tab/store\_select.png",

"text": "商店"

},

{

"pagePath": "pages/book/book",

"iconPath": "tab/book\_normal.png",

"selectedIconPath": "tab/book\_select.png",

"text": "预约"

},

{

"pagePath": "pages/order/order",

"iconPath": "tab/order\_normal.png",

"selectedIconPath": "tab/order\_select.png",

"text": "订单"

},

{

"pagePath": "pages/person/person",

"iconPath": "tab/person\_normal.png",

"selectedIconPath": "tab/person\_select.png",

"text": "账户"

}

]

}

（3）打开index.js文件，添加登陆成功之后跳转代码，在console.log('登陆成功')代码下面添加如下代码：

wx.switchTab({

url: '../store/store'

})

（4）打开person.json文件，添加如下代码，禁止页面滑动：

{

"disableScroll":true

}

（5）打开person.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '个人中心'

})

（6）打开person.wxml文件，编写个人中心页面代码，替换源代码如下：

<view class='container'>

<image class='title\_bg' src='../../image/profile\_bg.png'></image>

<view class='title'>

<image class="userinfo\_avatar" src='{{userInfo.headImage}}' mode="cover"></image>

<text class='user\_name'>{{userInfo.name}}</text>

</view>

<view class='item'>

<view class='item\_left'>

<image class='img\_left' src='../../image/balance.png'></image>

<text class='text\_left'>余额</text>

</view>

<view class='item\_right' bindtap='pay\_click'>

<text class='right\_balance'>￥{{userInfo.balance}}</text>

<text class='right\_recharge'>充值</text>

</view>

</view>

<view class='item' bindtap='consumption\_click'>

<view class='item\_left'>

<image class='img\_left' src='../../image/consumption.png'></image>

<text class='text\_left'>消费记录</text>

</view>

<image class='img\_more' src='../../image/right.png'></image>

</view>

<view class='item' bindtap='recharge\_click'>

<view class='item\_left'>

<image class='img\_left' src='../../image/recharge.png'></image>

<text class='text\_left'>充值记录</text>

</view>

<image class='img\_more' src='../../image/right.png'></image>

</view>

<view class='item' bindtap='address\_click'>

<view class='item\_left'>

<image class='img\_left' src='../../image/address.png'></image>

<text class='text\_left'>收货地址</text>

</view>

<image class='img\_more' src='../../image/right.png'></image>

</view>

</view>

（7）打开person.wxss文件，为个人中心页面添加样式，添加如下代码：

/\* pages/person/person.wxss \*/

.title\_bg{

width: 100%;

height: 200rpx;

}

.title{

width: 100%;

height: 120rpx;

position: absolute;

top:40rpx;

display: flex;

align-items:center;

}

.userinfo\_avatar {

width: 120rpx;

height: 120rpx;

border-radius: 50%;

margin-left: 40rpx;

}

.user\_name{

margin-left: 30rpx;

color:white;

}

.item{

width: 100%;

display: flex;

justify-content:space-between;

border-bottom: 1rpx solid rgb(218, 201, 201);

align-items:center;

}

.img\_more{

margin: 30rpx;

width: 50rpx;

height: 50rpx;

}

.item\_left{

display: flex;

align-items:center;

}

.img\_left{

width: 50rpx;

height: 50rpx;

margin-left: 20rpx;

}

.text\_left{

margin-left: 20rpx;

font-size: 35rpx;

}

.item\_right{

margin: 30rpx;

height: 50rpx;

}

.right\_balance{

font-size: 35rpx;

}

.right\_recharge{

font-size: 35rpx;

color: red;

border: 1px solid red;

padding: 5rpx;

margin-left: 10rpx;

}

（8）打开person.js文件，定义页面数据，在data字段下添加如下代码：

userInfo:null,

（9）定义获取网络数据的方法，添加loadData方法，代码如下：

loadData:function(){

wx.showNavigationBarLoading()

var that = this;

wx.request({

url: getApp().globalData.host + 'user/userInfo',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideNavigationBarLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('请求成功')

that.setData({

userInfo: res.data.data

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideNavigationBarLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

（10）调用网络请求方法，在onLoad方法里添加如下代码：

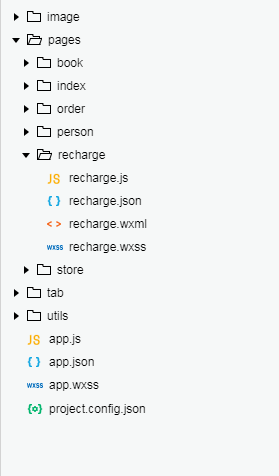
this.loadData()

## 任务2.3 充值记录

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/recharge目录，在该目录下新建recharge页面。项目结构如下图：



（3）打开person.js文件，添加充值记录按钮页面跳转方法，添加如下代码：

recharge\_click:function(){

wx.navigateTo({

url: '../recharge/recharge'

})

},

（4）打开recharge.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '充值记录'

})

（5）打开recharge.wxml文件，绘制充值记录列表页面，替换如下代码：

<view class='content' wx:for="{{list}}" wx:key="id">

<text >{{item.rechargeTime}}</text>

<text class='money'>+￥{{item.money}}</text>

</view>

（6）打开recharge.wxss文件，为充值记录页面添加样式文件，添加如下代码：

.content{

display: flex;

justify-content:space-between;

border-bottom: 1px solid rgb(201, 191, 191)

}

text{

font-size: 35rpx;

padding: 30rpx;

}

.money{

color: red

}

（7）打开recharge.js文件，为页面添加绑定数据，在data字段下添加如下代码：

list:null

（8）请求网络数据，在onLoad方法中添加网络请求，代码如下：

wx.showNavigationBarLoading()

var that = this;

wx.request({

url: getApp().globalData.host + 'recharge/getRecord',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideNavigationBarLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('请求成功')

that.setData({

list: res.data.data

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideNavigationBarLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

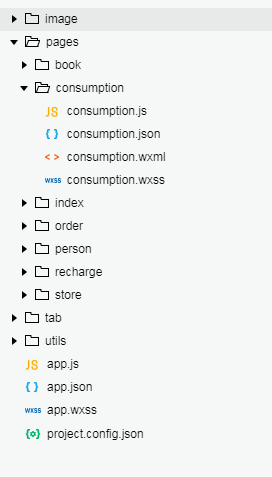
})

## 任务2.4 消费记录

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/consumption目录，在该目录下新建consumption页面。项目结构如下图：



（3）打开person.js文件，添加充值记录按钮页面跳转方法，添加如下代码：

consumption\_click:function(){

wx.navigateTo({

url: '../consumption/consumption'

})

},

（4）打开consumption.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '消费记录'

})

（5）打开consumption.wxml文件，绘制消费记录页面，添加如下代码：

<!--pages/consumption/consumption.wxml-->

<wxs module="m1">

var msg = function(val){

var status;

if(val==0){

status='已完成'

}else if(val==1){

status='待发货'

}else if(val==2){

status='待收货'

}

return status

}

module.exports.getStaus = msg;

</wxs>

<view class="swiper-tab">

<view class="tab-list-left {{currentTab==0 ? 'on' : ''}}" data-current="0" bindtap="swichNav">时间排序</view>

<view class="tab-list-right {{currentTab==1 ? 'on' : ''}}" data-current="1" bindtap="swichNav">金额排序 </view>

</view>

<view class="content {{index%2==0 ? 'bg':''}}" wx:for="{{list}}" wx:key="id">

<view class='item'>

<text class='order'>订单号：<text class='order\_no'>{{item.orderNum}}</text></text>

<text class='time'>{{item.createTime}}</text>

</view>

<view class='item'>

<text class='money'>-￥{{item.money}}</text>

<text class='staus\_{{item.status}}'>{{m1.getStaus(item.status)}}</text>

</view>

</view>

（6）打开consumption.wxss文件，为消费记录页面添加样式，添加如下代码：

.content{

display: flex;

justify-content:space-between;

border-bottom: 1px solid rgb(245, 245, 245);

align-items: center;

}

.bg{

background-color: rgb(245, 245, 245)

}

.money{

color: rgb(19, 148, 19);

margin-right: 20rpx;

}

.staus\_0{

margin-right: 20rpx;

text-align: right;

font-size: 25rpx;

}

.staus\_1{

margin-right: 20rpx;

text-align: right;

font-size: 25rpx;

color: red

}

.staus\_2{

margin-right: 20rpx;

text-align: right;

font-size: 25rpx;

color: blue

}

.item{

display: flex;

flex-direction: column;

margin: 20rpx;

}

.order{

font-size: 30rpx;

}

.order\_no{

font-size: 30rpx;

color: blue;

}

.time{

font-size: 20rpx;

margin-top: 10rpx;

}

.swiper-tab {

text-align: center;

line-height:10vh;

height: 10vh;

margin-top: auto

}

.tab-list-left {

font-size: 30rpx;

display: inline-block;

width: 25%;

border: 1px solid #bbada0;

line-height: 60rpx;

color: #bbada0;

}

.tab-list-right {

line-height: 60rpx;

font-size: 30rpx;

display: inline-block;

width: 25%;

border: 1px solid #bbada0;

color: #bbada0;

}

.on {

color: #fff;

background-color: #bbada0;

}

（7）打开consumption.js文件，添加页面绑定数据，在data字段下添加如下代码：

list:null,

currentTab:0,

valueBy:"createTime",

orderBy:"0"

（8）新增loadData网络请求方法，根据状态获取网络数据，添加如下代码：

loadData:function(){

wx.showNavigationBarLoading()

var that = this;

wx.request({

url: getApp().globalData.host + 'order/show',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

orderBy: this.data.orderBy,

valueBy: this.data.valueBy

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideNavigationBarLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('请求成功')

that.setData({

list: res.data.data

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideNavigationBarLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

（9）新增按钮单击事件方法，根据按钮状态设定请求参数，添加代码如下：

swichNav:function(e){

var val;

var order;

if (e.target.dataset.current == 1) {

val='money'

order="1"

} else {

val ='createTime'

order = "0"

}

this.setData({

currentTab: e.target.dataset.current,

valueBy: val,

orderBy: order

})

this.loadData();

}

（10）进入页面用默认参数请求数据，在onLoad方法中添加代码如下：

this.loadData();

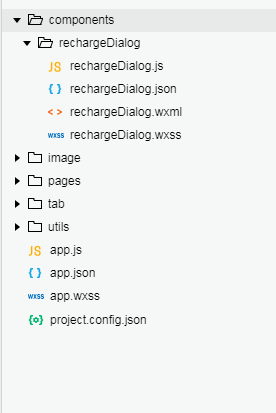
## 任务2.5 充值接口

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建自定义充值对话框页面

① 新建components/rechargeDialog目录，在该目录下新建rechargeDialog的Component构造器用来自定义充值对话框。项目结构如下图：



***注意：这边新建的是Component构造器。***

② 打开rechargeDialog.wxml文件，为绘制自定义充值对话框添加如下代码：

<view class='content' hidden='{{isShown}}'catchtap='cancel\_click' >

<view class='dialog' catchtap='none\_click'>

<form bindsubmit="formSubmit">

<view class='ip\_money'>

<text >输入金额：</text>

<view class='ip\_money right'>

<input placeholder='0' placeholder-class='placeholder' value='{{value}}' name='money' type='number'></input>

<text >元</text>

</view>

</view>

<view class='money\_value'>

<text catchtap='bt\_click' data-value="50">50</text>

<text catchtap='bt\_click' data-value="100">100</text>

<text catchtap='bt\_click' data-value="200">200</text>

</view>

<view class='money\_value'>

<text catchtap='bt\_click' data-value="500">500</text>

<text catchtap='bt\_click' data-value="800">800</text>

<text catchtap='bt\_click' data-value="1000">1000</text>

</view>

<view class='view\_foot'>

<button formType="submit">{{ confirmText }}</button>

</view>

</form>

</view>

</view>

③ 打开rechargeDialog.wxss文件，为自定义充值对话框添加样式，添加如下代码：

.content{

background: rgba(0, 0, 0, 0.3);

width: 100vw;

height: 100vh;

display: flex;

align-items:center;

justify-content:center;

z-index: 1000;

position: absolute;

top: 0;

left: 0;

}

.dialog{

background: rgb(245, 245, 245);

width: 70%;

height: 40%;

border-radius: 10rpx;

position: relative;

}

.ip\_money{

padding: 10rpx;

display: flex;

align-items:center;

justify-content:space-between;

background: white;

margin: 60rpx 20rpx 40rpx 20rpx;

}

.right{

padding: 0rpx;

margin:0rpx;

}

.ip\_money text{

font-size: 30rpx;

margin-right: 10rpx;

}

.ip\_money input{

text-align: right;

width:230rpx;

margin-right: 20rpx;

font-size: 30rpx;

}

.placeholder{

font-size: 30rpx

}

.money\_value{

display: flex;

justify-content:space-between;

margin: 20rpx 10rpx;

}

.money\_value text{

color: white;

background: red;

margin: 10rpx;

flex: 1;

text-align: center;

border-radius: 5rpx;

}

.view\_foot{

display: flex;

margin-top: 10rpx;

align-items:center;

justify-content:space-between;

position: absolute;

bottom: 0;

width: 100%;

}

.view\_foot button{

flex: 1;

text-align: center;

background: #fff;

}

button::after{

border: none;

}

④ 添加组件的属性列表，在properties字段下添加如下代码：

confirmText: {

type: String,

value: '确认'

},

⑤ 添加组件的初始化数据，在data字段下添加如下代码：

value:0,

isShown:true

⑥ 添加自定义组件所需要使用的方法，以下内容在methods字段下添加：

//面板金额按钮

bt\_click: function (e) {

this.setData({

value: e.target.dataset.value

})

},

//对话框点击事件，无任何事件响应

none\_click: function () {

},

//取消按钮事件，点击非对话框区域事件

cancel\_click:function(){

this.triggerEvent("cancelEvent")

},

//确认按钮事件

formSubmit: function (e) {

console.log("formSubmit")

this.triggerEvent("confirmEvent", { data\_value: e })

},

//隐藏弹框

hideDialog() {

this.setData({

isShown: !this.data.isShown,

value:0

})

},

//展示弹框

showDialog: function () {

this.setData({

isShown: !this.data.isShown,

})

},

（3）引用自定义充值对话框组件

① 打开person.json文件，添加自定义组件路径，在"disableScroll": true后面添加如下代码：

"usingComponents": {

"dialog": "/components/rechargeDialog/rechargeDialog"

}

② 打开person.wxml文件，引用自定义组件，在最后一行代码</view>前添加如下代码：

<dialog id="dialog"

confirmText="确认"

bind:cancelEvent="cancelClick"

bind:confirmEvent="confirmClick">

</dialog>

③ 获取自定义对话框组件对象，在onReady方法中添加如下代码：

this.dialog = this.selectComponent("#dialog");

④ 打开person.js文件，添加隐藏对话框方法，添加如下代码：

cancelClick:function(){

this.dialog.hideDialog();

}

⑤ 添加确认按钮时间监听，判断输入金额是否大于0，如果金额大于0，则调用模拟充值接口，充值成功之后调用loadData方法刷新页面数据。如果金额小于等于0，则给出错误提示。添加如下代码：

confirmClick: function (e) {

var value = e.detail.data\_value.detail.value.money;

if (value>0){

wx.showLoading({

title: '加载中',

mask:true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'recharge/record',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

money: value

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('充值成功')

that.dialog.hideDialog();

wx.showToast({

title: "充值成功",

icon: 'success',

duration: 1000

})

that.loadData();

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}else{

wx.showToast({

title: '请输入正确金额',

icon: 'none',

duration: 1000

})

}

}

⑥ 点击充值按钮，弹出自定义对话框，添加如下代码：

pay\_click:function(){

this.dialog.showDialog();

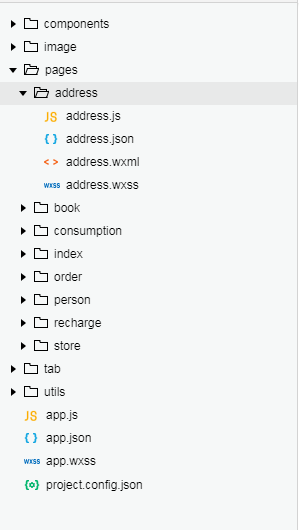
}

## 任务2.6 收货地址

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/address目录，在该目录下新建address页面。项目结构如下图：



（3）打开person.js文件，添加收货地址按钮页面跳转方法，添加如下代码：

address\_click: function () {

wx.navigateTo({

url: '../address/address'

})

},

（4）打开address.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '收货地址'

})

（5）打开address.wxml文件，绘制收货地址列表页面，添加如下代码：

<view class='container'>

<view class='content' wx:for="{{list}}" wx:key="id" data-index="{{index}}" bindtap='item\_click' >

<view class='view\_item'>

<text>{{item.name}}</text>

<text>{{item.phone}}</text>

</view>

<view class='address'>

<text >{{item.province+item.city+item.district+item.detail}}</text>

</view>

<view class='view\_item line'>

<view class='view\_bottom\_item' data-index="{{index}}" catchtap='default\_click'>

<image src="../../image/{{item.defaultFlag==true?'img\_selected':'img\_unselect'}}.png"></image>

<text class="{{item.defaultFlag==true?'on':''}}">{{item.defaultFlag==true?'默认地址':'设为默认地址'}}</text>

</view>

<view class='view\_bottom\_item'>

<view class='view\_bottom\_item' data-index="{{index}}" catchtap='edit\_click'>

<image src='../../image/img\_edit.png'></image>

<text>编辑</text>

</view>

<view class='view\_bottom\_item' data-index="{{index}}" catchtap='delete\_click'>

<image src='../../image/img\_delete.png'></image>

<text >删除</text>

</view>

</view>

</view>

</view>

<button class='add' type='primary' catchtap='add\_click'>添加新地址</button>

</view>

（6）打开address.wxss文件，为守护地址列表添加样式，添加如下代码：

.content{

display: flex;

flex-direction: column;

width: 100%;

}

.view\_item{

display: flex;

justify-content: space-between;

}

.line{

border-bottom:14rpx solid rgb(238, 238, 238);

}

.view\_item text{

margin: 20rpx;

}

.address{

width: 100%;

padding: 0 20rpx 20rpx 20rpx;

border-bottom:1rpx solid rgb(238, 238, 238);

box-sizing:border-box;

}

.address text{

word-break: break-all;

}

.view\_bottom\_item{

display: flex;

align-items: center;

padding-left: 20rpx;

}

.view\_bottom\_item image{

width: 40rpx;

height: 40rpx;

}

.view\_bottom\_item text{

margin-left: 10rpx;

}

.on{

color: orange;

}

.bottom\_right{

margin-left: 20rpx;

}

.add{

width: 70%;

margin: 20rpx 0;

}

（7）为页面绑定数据，在data字段下添加如下代码：

list:null

（8）为网络请求列表数据封装方法，添加loadData方法，代码如下：

loadData:function(){

wx.showNavigationBarLoading()

var that = this;

wx.request({

url: getApp().globalData.host + 'address/all',

method: 'POST',

header: {

'content-type': 'application/json' // 默认值

},

data: {

userId: getApp().globalData.userInfo.userId,

},

success: function (res) {

wx.hideNavigationBarLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('请求成功')

that.setData({

list: res.data.data

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideNavigationBarLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

（9）调用网络请求方法，在onShow方法中调用如下代码：

this.loadData();

（10）封装设置默认地址的网络请求方法，传入参数为列表索引和数据id。设置成功之后，更新本地数据。添加changeDefault方法，如下代码:

changeDefault:function(index,id){

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'address/changeDefault',

method: 'POST',

data: {

addrId: id

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('请求成功')

var listInfo = that.data.list;

for (var i = 0; i < listInfo.length;i++){

listInfo[i].defaultFlag=false;

}

listInfo[index].defaultFlag=true;

that.setData({

list: listInfo

})

wx.showToast({

title: "操作成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

（11）为设置默认地址按钮添加单击事件，添加default\_click方法，如下代码:

default\_click:function(e){

var index = e.currentTarget.dataset.index;

var id = this.data.list[index].addrId;

var defaultFlag = this.data.list[index].defaultFlag;

if (defaultFlag!=true){

this.changeDefault(index,id);

}

}

（12）封装删除地址的网络请求方法，传入参数为数据id。删除成功以后，重新请求列表数据获取最新的列表数据。添加deleteAddress方法，如下代码:

deleteAddress:function(id){

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'address/delete ',

method: 'POST',

data: {

addrId: id

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('删除成功')

that.loadData();

wx.showToast({

title: "删除成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

（13）为删除地址按钮添加单击事件，添加delete\_click方法，如下代码:

delete\_click: function (e) {

var index = e.currentTarget.dataset.index;

var id = this.data.list[index].addrId;

var that = this

wx.showModal({

title: '提示',

content: '是否删除该条地址？',

success: function (res) {

if (res.confirm) {

that.deleteAddress(id)

}

}

})

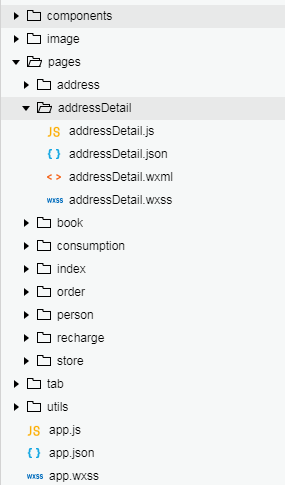
}

## 任务2.7 收货地址

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/addressDetail目录，在该目录下新建addressDetail页面。项目结构如下图：



（3）打开address.js文件，添加编辑收货地址页面跳转edit\_click方法，页面间传递当前选中地址信息参数。添加如下代码：

edit\_click: function (e) {

var index = e.currentTarget.dataset.index;

var addressBean = JSON.stringify(this.data.list[index])

wx.navigateTo({

url: '../addressDetail/addressDetail?address=' + addressBean,

})

},

（4）打开address.js文件，添加新增收货地址页面跳转add\_click方法，页面间传递空值，方便统一处理。添加如下代码：

add\_click:function(){

var addressBean = JSON.stringify(null)

wx.navigateTo({

url: '../addressDetail/addressDetail?address=' + addressBean,

})

},

（5）打开addressDetail.js文件，动态设置当前页面标题。在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '收货地址'

})

（6）打开addressDetail.wxml文件，绘制收货地址详情页面，添加如下代码：

<view class='container'>

<form bindsubmit='subfn'>

<view class='view\_item'>

<text>收货人</text>

<input placeholder='请输入收货人姓名' placeholder-class='placeholder' name='name' value='{{address.name}}'></input>

</view>

<view class='view\_item'>

<text>联系电话</text>

<input placeholder='请输入联系电话' placeholder-class='placeholder' name='phone' value='{{address.phone}}'></input>

</view>

<view class="view\_item\_between">

<text class='value\_left'>所在地区</text>

<picker mode="region" bindchange="bindRegionChange" value="{{region}}" name='city'>

<view class="pick">

<text>{{city}}</text>

<image class='img\_more' src='../../image/right.png'></image>

</view>

</picker>

</view>

<view class='view\_address'>

<textarea placeholder='请填写详细地址' placeholder-class='placeholder' name='address' value='{{address.detail}}'></textarea>

</view>

<view class="view\_item\_between">

<text class='value\_left'>设为默认</text>

<switch checked='{{address.defaultFlag}}' name='default'/>

</view>

<button class='add' type='primary' form-type='submit'>提交</button>

</form>

</view>

（7）打开addressDetail.wxss文件，为收货地址详情页添加样式文件，添加如下代码：

.view\_item{

display: flex;

width: 100%;

align-items: center;

border-bottom: 1px solid rgb(245, 245, 245);

}

.view\_item text{

margin: 20rpx;

width: 250rpx;

}

.view\_item input{

width: 100%;

margin-right: 20rpx;

}

.view\_item\_between{

justify-content: space-between;

display: flex;

width: 100%;

align-items: center;

border-bottom: 1px solid rgb(245, 245, 245);

}

.value\_left{

margin: 20rpx;

width: 250rpx;

}

.img\_more{

width: 50rpx;

height: 50rpx;

}

.pick{

display: flex;

align-items: center;

}

.view\_address{

padding: 20rpx;

width:100%;

box-sizing:border-box;

border-bottom: 15rpx solid rgb(245, 245, 245);

}

.view\_address textarea{

width:100%;

height: 150rpx;

}

.add{

width: 70%;

margin-top: 100rpx;

}

（8）添加页面绑定数据，在data字段下添加如下代码：

address:null,

city: "请选择",

region: [],

isAdd:true, //是否是新增地址

（9）获取上一页面传递过来的数据，在onLoad方法中添加如下代码：

var addressBean = JSON.parse(options.address);

var address\_detail;

var region\_info=[];

var isAdd\_info=true;

if (addressBean!=null){

address\_detail = addressBean.province + addressBean.city + addressBean.district

region\_info.push(addressBean.province);

region\_info.push(addressBean.city);

region\_info.push(addressBean.district);

isAdd\_info=false;

}else{

address\_detail='请选择'

isAdd\_info=true;

}

this.setData({

address: addressBean,

city: address\_detail,

region:region\_info,

isAdd: isAdd\_info

})

console.log(this.data.address)

（10）为区域选择添加数值变化监听事件：添加如下代码：

bindRegionChange: function (e) {

console.log('picker发送选择改变，携带值为', e.detail.value)

this.setData({

region: e.detail.value,

city: e.detail.value[0] + e.detail.value[1] + e.detail.value[2]

})

},

（11）封装网络请求submitData方法，进行表单的提交操作。传入参数为页面表单数据，添加如下代码：

submitData:function(value){

var type=this.data.isAdd;

var addrId;

if (this.data.isAdd){

type ='add'

addrId=''

}else{

type = 'updateAddr'

addrId = this.data.address.addrId;

}

wx.showLoading({

title: '加载中',

mask:true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'address/' + type,

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

name: value.name,

province: value.city[0],

city: value.city[1],

district: value.city[2],

detail: value.address,

phone: value.phone,

defaultFlag: value.default,

addrId: addrId

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('操作成功')

wx.navigateBack()

wx.showToast({

title: "操作成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

（12）为确认按钮添加subfn单击事件，获取页面表单数据，进行简单的表单验证，调用submitData方法，上传或者添加地址信息。添加如下代码：

subfn:function(e){

console.log('form发生了submit事件，携带数据为：', e.detail.value)

var info = e.detail.value;

if (info.name==''){

wx.showToast({

title: "请填写收货人",

icon: 'none',

duration: 1000

})

return;

}

if (info.phone == '') {

wx.showToast({

title: "请填写联系电话",

icon: 'none',

duration: 1000

})

return;

}

if (this.data.city == '请选择') {

wx.showToast({

title: "请选择所在地区",

icon: 'none',

duration: 1000

})

return;

}

if (info.address == '') {

wx.showToast({

title: "请填写详细地址",

icon: 'none',

duration: 1000

})

return;

}

this.submitData(info)

},

# 单元3：商店模块

## 任务3.1 列表页

### 客户端

（1）打开store.wxml文件，绘制商店主页面，添加如下代码：

<!--pages/store/store.wxml-->

<view class='container'>

<scroll-view scroll-y class='scroll' class='view\_list' bindscrolltolower='loadMore' bindscrolltoupper='reflash'>

<block wx:for="{{list}}">

<view class='list\_item' data-index="{{index}}" bindtap='item\_click'>

<view class='list\_item\_left'>

<image class="logo" mode="scaleToFill" src='{{item.image}}'></image>

<view class='list\_item\_middle'>

<text>{{item.name}}</text>

<text class='price'>￥{{item.price}}</text>

</view>

</view>

<text class='add' data-index="{{index}}" catchtap='add'>购买</text>

</view>

</block>

</scroll-view>

<view class='view\_bottom'>

<view class='view\_bottom\_item'>

<image src='../../image/shopping.png'></image>

<text>{{total\_num}}件商品</text>

<text class='money\_total'>￥{{total\_money}}</text>

</view>

<view class='view\_bottom\_item'>

<button class='clear' bindtap='clear'>清空</button>

<button class='submit' bindtap='check\_click'>结算</button>

</view>

</view>

</view>

（2）打开store.wxss文件，为商店主页面添加样式文件，添加如下代码：

/\* pages/store/store.wxss \*/

.view\_bottom{

display: flex;

justify-content: space-between;

width: 100%;

align-items: center;

border-top: 1rpx solid rgb(245, 245, 245);

border-bottom: 1rpx solid rgb(245, 245, 245);

position: absolute;

bottom: 0;

height: 10vh;

}

.view\_bottom image{

width: 50rpx;

height: 50rpx;

padding: 20rpx;

}

.view\_bottom\_item{

display: flex;

align-items: center;

}

.view\_bottom\_item text{

font-size: 30rpx;

}

.money\_total{

color: orange;

margin-left: 20rpx;

}

.clear{

margin: 20rpx;

width: 120rpx;

font-size: 30rpx;

}

.submit{

margin: 20rpx;

width: 120rpx;

background: red;

color: white;

font-size: 30rpx;

}

.view\_list{

height: 90vh;

}

.list\_item{

display: flex;

align-items: center;

justify-content: space-between;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.logo {

width: 120rpx;

height: 120rpx;

border-radius: 20rpx;

padding: 20rpx;

flex-shrink:0;

}

.add{

padding: 15rpx 30rpx;

background: rgb(126, 126, 238);

color: white;

font-size: 30rpx;

margin-right: 20rpx;

border-radius: 10rpx;

flex-shrink:0;

}

.list\_item\_left{

display: flex;

align-items: center;

}

.list\_item\_middle{

display: flex;

flex-direction: column;

}

.list\_item\_middle text{

word-break:break-all;

}

.price{

color: orange;

margin-top: 20rpx;

}

（3）打开store.js文件，动态设置当前页面标题。在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '商店'

})

（4）打开store.js文件，为页面添加绑定数据，在data字段下添加如下代码：

list: [],

total\_money: 0,

total\_num: 0,

cart:[]

（5）在Page({代码上面定义分页需要的变量，添加如下代码：

var currentPage=1 //当前页数

var pageSize= 10 //每页显示数量

（6）添加网络请求接口loadData方法，请求商品列表数据，添加如下代码:

loadData: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'goods/getGoodsByPage',

method: 'POST',

data: {

page: currentPage,

pageSize: pageSize

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

var result = res.data.data;

var list\_info = that.data.list;

for (var i = 0; i < result.length;i++){

result[i].image = getApp().globalData.host + result[i].image;

list\_info.push(result[i]);

}

console.log(list\_info)

that.setData({

list: list\_info

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（7）添加网络请求接口loadCart方法，请求购物车相关数据，添加如下代码：

loadCart: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'cart/getAll',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

that.setData({

total\_money: res.data.data.total\_money,

total\_num: res.data.data.total\_num,

cart: res.data.data.goods\_list

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（8）调用上述2个方法，在页面显示时请求页面所需数据，在onShow方法中中添加如下代码：

currentPage=1;

this.setData({

list: []

})

this.loadData();

this.loadCart();

（9）新增下拉刷新reflash方法，初始化currentPage为1，将list列表制空。添加如下代码：

reflash:function(){

console.log("reflash ------")

currentPage=1;

this.setData({

list: []

})

this.loadData();

},

（10）新增上拉加载loadMor方法，currentPage参数自加加，添加如下代码：

loadMore : function () {

console.log("loadMore ------")

currentPage++;

this.loadData();

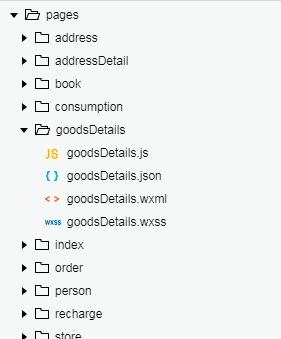
},

## 任务3.2 商品详情

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/goodsDetails目录，在该目录下新建goodsDetails页面。项目结构如下图：



（3）打开store.js文件，添加收货地址按钮页面跳转方法，添加如下代码：

item\_click:function(e){

var index = e.currentTarget.dataset.index;

var goodsId = this.data.list[index].goodsId;

wx.navigateTo({

url: '../goodsDetails/goodsDetails?goodsId=' + goodsId,

})

}

（4）打开goodsDetails.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '商品详情'

})

（5）添加goodsId参数，在Page({代码上面添加如下代码：

var goodsId;

接收上一页面传递过来的值，在onLoad方法中添加如下代码：

goodsId = options.goodsId;

console.log("goodsId:" + goodsId)

（6）打开goodsDetails.wxml文件，绘制商品详情页面，添加如下代码：

<!--pages/goodsDetails/goodsDetails.wxml-->

<view class='container'>

<scroll-view class='view\_scroll'scroll-y>

<view class='content'>

<view>商品名称:</view>

<text>{{goodsDetail.name}}</text>

<view class='line'></view>

</view>

<view class='content'>

<view>封面图片：</view>

<image src='{{goodsDetail.image}}' mode='widthFix'></image>

<view class='line'></view>

</view>

<view class='content'>

<view>商品简介:</view>

<text>{{goodsDetail.detail}}</text>

<view class='line'></view>

</view>

<view class='time'>

<view>上架时间:</view>

<text>{{goodsDetail.upTime}}</text>

<view class='line'></view>

</view>

</scroll-view>

<view class='view\_bottom'>

<view class='view\_bottom\_item'>

<text>单价：</text>

<text class='price'>￥{{goodsDetail.price}}</text>

</view>

<view class='view\_bottom\_item'>

<button class='add' bindtap='add'>加入购物车</button>

<button class='submit' bindtap='check\_click'>购买</button>

</view>

</view>

</view>

（7）打开goodsDetails.wxss文件，为商品详情页添加样式文件，添加如下代码：

.view\_bottom{

display: flex;

justify-content: space-between;

width: 100%;

align-items: center;

border-top: 1rpx solid rgb(245, 245, 245);

border-bottom: 1rpx solid rgb(245, 245, 245);

position: absolute;

bottom: 0;

height: 10vh;

}

.view\_bottom\_item{

display: flex;

align-items: center;

margin-left: 20rpx;

}

.view\_bottom\_item text{

font-size: 30rpx;

}

.price{

color: orange;

margin-left: 20rpx;

}

.add{

margin: 20rpx;

font-size: 30rpx;

height: 6vh;

line-height: 6vh;

text-align: center

}

.submit{

margin: 20rpx;

background: red;

color: white;

font-size: 30rpx;

height: 6vh;

line-height: 6vh;

text-align: center

}

.view\_scroll{

height: 90vh;

}

.content{

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

display: flex;

flex-direction: column;

align-items: center;

}

.time{

display: flex;

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.content view{

width: 100%;

}

.content text{

width: 100%;

}

.content image{

width: 100%;

margin-top: 10rpx;

}

（8）打开goodsDetails.js文件，添加页面绑定数据，在data字段下添加如下代码：

goodsDetail:null

（9）添加网络请求接口方法，根据上一页面传递过来的goodsId获取商品详情信息，在console.log("goodsId:" + goodsId)代码下面添加如下代码：

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'goods/goodsDetails',

method: 'POST',

data: {

goodsId: goodsId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

var result = res.data.data;

result.image = getApp().globalData.host + result.image;

that.setData({

goodsDetail: result

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

## 任务3.3 购物车操作

### 客户端

（1）打开store.js文件，添加清空购物车的clearData网络请求方法，添加如下代码：

clearData:function(){

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'cart/delete',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('删除成功')

that.loadCart();

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

（2）添加clear单击事件方法，弹出警告提示框，点击确认的时候调用网络请求，添加如下代码：

clear:function(){

var that=this;

wx.showModal({

title: '提示',

content: '是否清空购物车？',

success: function (res) {

if (res.confirm) {

that.clearData()

}

}

})

}

（3）为商品列表的购买按钮添加单击事件add单击事件，先获取按钮上的index值，根据此值获取goodsId，然后进行网络接口请求，添加如下代码：

add:function(e){

var index = e.currentTarget.dataset.index;

var goodsId = this.data.list[index].goodsId;

console.log("goodsId:" + goodsId)

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'cart/add',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

goodsId: goodsId

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('添加成功')

wx.showToast({

title: "添加成功",

icon: 'none',

duration: 1000

})

that.loadCart();

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

（4）打开goodsDetails.js文件，为页面上添加到购物车按钮增加add单击事件。添加如下代码：

add: function (e) {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'cart/add',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

goodsId: goodsId

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('添加成功')

wx.showToast({

title: "添加成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

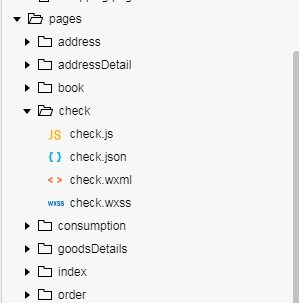
}

## 任务3.4 结算页面

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/check目录，在该目录下新建check页面。项目结构如下图：



（3）打开store.js文件，添加结算按钮check\_click单击事件方法，判断购物车是否有商品存在，页面间传递当前购物车列表参数。添加如下代码：

check\_click: function (e) {

if (this.data.cart.length==0){

wx.showToast({

title: "购物车无数据",

icon: 'none',

duration: 1000

})

return

}

var cartBean = JSON.stringify(this.data.cart)

wx.navigateTo({

url: '../check/check?cart=' + cartBean + '&flag=0'

,

})

},

（4）打开goodsDetails.js文件，添加购买按钮check\_click单击事件方法，页面间传递空值，方便统一处理。添加如下代码：

check\_click:function(){

var list=[];

list.push({

goodsId: goodsId,

num:1,

userId: getApp().globalData.userInfo.userId

})

var cartBean = JSON.stringify(list)

wx.navigateTo({

url: '../check/check?cart=' + cartBean + '&flag=0',

})

},

（5）打开check.js文件，动态设置当前页面标题。在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '结算'

})

（6）打开check.wxml文件，绘制结算页面，添加如下代码：

<!--pages/check/check.wxml-->

<view class='container'>

<form bindsubmit="formSubmit">

<scroll-view scroll-y class='content'>

<text class='title'>收货地址：</text>

<view class='view\_item' bindtap='address\_click'>

<view wx:if="{{hasAddress}}" class='view\_address'>

<view class='view\_item\_left'>

<text>{{address.name}}</text>

<text>{{address.phone}}</text>

</view>

<view class='address'>

<text >{{address.province+address.city+address.district+address.detail}}</text>

</view>

</view>

<text wx:else class='no\_address'>没有默认地址，请先添加</text>

<image class='more' src='../../image/right.png'></image>

</view>

<text class='title'>商品列表：</text>

<block wx:for="{{list}}">

<view class='list\_item' data-index="{{index}}" bindtap='item\_click'>

<view class='list\_item\_left'>

<image class="logo" mode="scaleToFill" src='{{item.goods.image}}'></image>

<view class='list\_item\_middle'>

<text>{{item.goods.name}}</text>

<view class='item\_price'>

<text class='price'>￥{{item.goods.price}}</text>

<text>\*{{item.number}}</text>

</view>

</view>

</view>

</view>

</block>

<view class='total\_price'>

<text >总价：{{total\_price}}</text>

</view>

<text class='title'>备注：</text>

<view class='view\_mark'>

<textarea placeholder='请填写备注' placeholder-class='placeholder' name='mark'></textarea>

</view>

</scroll-view>

<view class='check'>

<button type='primary' formType="submit">结算</button>

</view>

</form>

</view>

（7）打开check.wxss文件，为结算页面添加样式文件，添加如下代码：

.view\_item{

display: flex;

justify-content: space-between;

width: 100vw;

align-items: center;

border-bottom:1rpx solid rgb(238, 238, 238);

}

.view\_item\_left{

display: flex;

justify-content: space-between;

width: 90vw;

}

.view\_item\_left text{

margin: 20rpx;

}

.address{

width: 90vw;

margin: 0 0 20rpx 20rpx;

box-sizing:border-box;

word-break: break-all

}

.view\_address{

width: 90vw;

}

.more{

width: 50rpx;

height: 50rpx;

}

.no\_address{

margin: 20rpx;

color: red

}

.title{

width: 100%;

margin-left: 20rpx;

}

.list\_item{

display: flex;

align-items: center;

justify-content: space-between;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.logo {

width: 120rpx;

height: 120rpx;

border-radius: 20rpx;

padding: 20rpx;

}

.list\_item\_left{

display: flex;

align-items: center;

}

.list\_item\_middle{

display: flex;

flex-direction: column;

width: 70vw

}

.list\_item\_middle text{

white-space: nowrap;

text-overflow:ellipsis;

display: block;

overflow: hidden;

}

.price{

color: orange;

}

.item\_price{

display: flex;

justify-content: space-between;

align-items: center;

margin-top: 20rpx;

}

.content{

display: flex;

flex-direction: column;

width: 100%;

height: 90vh;

}

.check{

position: absolute;

bottom: 0;

width: 100%;

text-align: center;

display: flex;

align-items: center;

justify-content: center;

height: 10vh;

}

.check button{

width: 70%;

}

.total\_price{

margin: 20rpx;

text-align: right

}

.total\_price text{

color: green

}

.view\_mark{

padding: 20rpx;

width:100%;

box-sizing:border-box;

border-bottom: 15rpx solid rgb(245, 245, 245);

}

.view\_mark textarea{

width:100%;

height: 150rpx;

}

（8）打开check.js文件，添加页面绑定数据，在data字段下添加如下代码：

list:[],

address:null,

total\_price:0,

hasAddress:false

（9）定义上一页面传递过来的参数，在Page({代码上面添加如下代码：

var cartBean;

var flag; //0 购物车 1 详情页

（10）接收上一页面传递过来的参数，获取网络接口数据，根据返回值判断是否存在默认收货地址。在onLoad方法中添加如下代码：

flag = options.flag;

cartBean = JSON.parse(options.cart);

console.log(cartBean)

console.log(flag)

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'order/settle',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

goods\_list: cartBean

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

if (res.data.data.address==null){

that.setData({

hasAddress:false

})

}else{

that.setData({

hasAddress: true,

address: res.data.data.address

})

}

var list\_info = res.data.data.good\_list;

for (var i = 0; i < list\_info.length;i++){

list\_info[i].goods.image = getApp().globalData.host + list\_info[i].goods.image

}

that.setData({

list: list\_info,

total\_price: res.data.data.total\_money

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

## 任务3.5 支付页面

### 客户端

（1）打开check.js文件，添加地址选择修改address\_click单击事件，跳转到收货地址列表，传递check值为1的参数，添加如下代码：

address\_click:function(){

wx.navigateTo({

url: '../address/address?check=1',

})

},

（2）打开address.js文件，添加isFromCheck参数，用来表示是否是从结算页面跳转过来，在Page({代码上面添加如下代码：

var isFromCheck=0;

（3）在onLoad方法中接收上一页面传递过来的参数，添加如下代码：

var checkBean = options.check;

console.log("checkBean:" + checkBean)

isFromCheck = checkBean;

（4）添加列表item\_click单击事件，如不是从结算页面跳转过来，则return此次单击事件，否则将该条address更新至结算页面，添加如下代码：

item\_click:function(e){

if (isFromCheck!=1){

return;

}

var index = e.currentTarget.dataset.index;

var address = this.data.list[index];

console.log("item\_click")

var pages = getCurrentPages()

var prevPage = pages[pages.length - 2] //上一个页面

prevPage.setData({

address: address

})

wx.navigateBack({

delta: 1

})

}

（5）打开check.js文件，添加结算按钮formSubmit单击事件，首先获取备注输入框信息，请求成功以后返回上一页。添加如下代码：

//确认按钮事件

formSubmit: function (e) {

console.log(e)

var mark=e.detail.value.mark;

if(mark==''){

mark='暂无备注'

}

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

console.log(cartBean)

wx.request({

url: getApp().globalData.host + 'order/create',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

money: that.data.total\_price+"",

addrId: that.data.address.addrId,

goods\_list: cartBean,

message: mark,

flag:flag

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('结算成功')

wx.navigateBack({

delta: 1,

success:()=>{

wx.showToast({

title: "结算成功",

icon: 'none',

duration: 1000

})

}

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

}

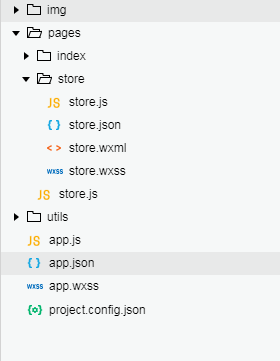
## 任务3.6 列表页

### 商户端

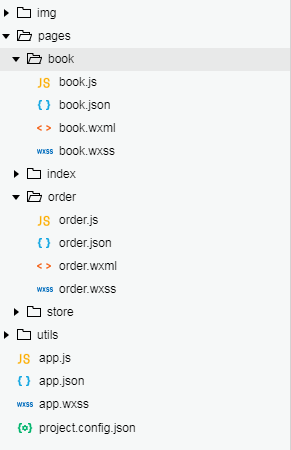
（1）打开商户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）构造底部tab页效果

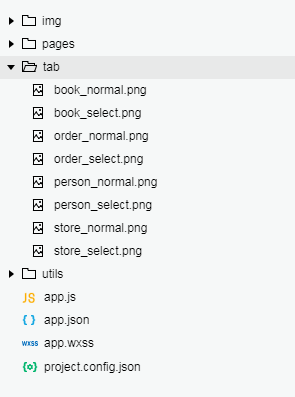
① 新建pages/store目录，在该目录下新建store页面。项目结构如下图：



② 相同的办法新建book、order两个页面。项目结构如下图所示：



③ 将资源包里tab文件夹拷贝到项目目录下，项目结构如下：



④ 打开app.json文件，添加tab页设置，顺序添加如下代码：

"tabBar": {

"borderStyle": "white",

"backgroundColor": "#ffffff",

"selectedColor": "#81c7d1",

"color": "#a6a6a6",

"list": [

{

"pagePath": "pages/store/store",

"iconPath": "tab/store\_normal.png",

"selectedIconPath": "tab/store\_select.png",

"text": "商店"

},

{

"pagePath": "pages/book/book",

"iconPath": "tab/book\_normal.png",

"selectedIconPath": "tab/book\_select.png",

"text": "预约"

},

{

"pagePath": "pages/order/order",

"iconPath": "tab/order\_normal.png",

"selectedIconPath": "tab/order\_select.png",

"text": "订单"

}

]

}

（3）打开index.js文件，添加登陆成功跳转到首页功能，在console.log('登陆成功')代码后面添加如下代码：

wx.switchTab({

url: '../store/store'

})

（4）打开store.wxml文件，绘制商店主页面，添加如下代码：

<view class='container'>

<scroll-view scroll-y class='view\_list' bindscrolltolower='loadMore' bindscrolltoupper='reflash' >

<block wx:for="{{list}}">

<view class="list\_item {{item.isTouchMove ? 'trastram\_left' : 'trastram\_right'}}" data-index="{{index}}" bindtap='item\_click' bindtouchstart="touchstart" bindtouchmove="touchmove">

<view class='list\_item\_left'>

<image class="logo" mode="scaleToFill" src='{{item.image}}'></image>

<view class='list\_item\_middle'>

<text>{{item.name}}</text>

<text class='price'>￥{{item.price}}</text>

</view>

</view>

<text class='delete' data-index="{{index}}" catchtap='delete\_click'>删除 </text>

</view>

</block>

</scroll-view>

<view class='view\_bottom'>

<button class='add' type='primary' bindtap='add\_click'>添加商品</button>

</view>

</view>

（5）打开store.wxss文件，为商店主页面添加样式文件，添加如下代码：

.view\_list{

height: 90vh;

}

.list\_item{

display: flex;

align-items: center;

justify-content: space-between;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.trastram\_left{

transform: translateX(-180rpx);

transition: all 1s;

}

.trastram\_right{

transform: translateX(0px);

transition: all 1s;

}

.delete{

background: red;

color: white;

position: absolute;

right: -180rpx;

width: 180rpx;

height: 100%;

text-align: center;

display: flex;

align-items: center;

justify-content: center;

}

.list\_item\_left{

display: flex;

align-items: center;

width: 100%

}

.logo {

width: 120rpx;

height: 120rpx;

border-radius: 20rpx;

padding: 20rpx;

flex-shrink:0;

}

.list\_item\_middle{

display: flex;

flex-direction: column;

}

.list\_item\_middle text{

word-break:break-all;

padding: 0 20rpx;

}

.price{

color: orange;

margin-top: 20rpx;

}

.add{

width: 70%;

margin: 20rpx 0;

}

.view\_bottom{

display: flex;

justify-content: center;

width: 100%;

align-items: center;

border-top: 1rpx solid rgb(245, 245, 245);

border-bottom: 1rpx solid rgb(245, 245, 245);

position: absolute;

bottom: 0;

height: 10vh;

}

（6）打开store.js文件，动态设置当前页面标题。在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '商店'

})

（7）打开store.js文件，为页面添加绑定数据，在data字段下添加如下代码：

list:[],

startX: 0, //开始坐标

startY: 0,

（8）在Page({代码上面定义分页需要的变量，添加如下代码：

var currentPage=1 //当前页数

var pageSize= 10 //每页显示数量

（9）添加网络请求接口loadData方法，请求商品列表数据，为每一条数据添加isTouchMove属性。添加如下代码:

loadData: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'goods/getGoodsByPage',

method: 'POST',

data: {

page: currentPage,

pageSize: pageSize

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

var result = res.data.data;

var list\_info = that.data.list;

for (var i = 0; i < result.length; i++) {

result[i].image = getApp().globalData.host + result[i].image;

result[i].isTouchMove=false; //添加滑动属性

list\_info.push(result[i]);

}

console.log(list\_info)

that.setData({

list: list\_info

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（10）调用加载列表网络请求方法，在页面显示时请求页面所需数据，在onShow方法中中添加如下代码：

currentPage=1;

this.setData({

list: []

})

this.loadData();

（11）新增下拉刷新reflash方法，初始化currentPage为1，将list列表制空。添加如下代码：

reflash:function(){

console.log("reflash ------")

currentPage=1;

this.setData({

list: []

})

this.loadData();

},

（12）新增上拉加载loadMor方法，currentPage参数自加加，添加如下代码：

loadMore : function () {

console.log("loadMore ------")

currentPage++;

this.loadData();

},

（13）添加计算滑动角度angle方法，传入参数为起始坐标和终点坐标。代码如下：

angle: function (start, end) {

var \_X = end.X - start.X,

\_Y = end.Y - start.Y

//返回角度 /Math.atan()返回数字的反正切值

return 360 \* Math.atan(\_Y / \_X) / (2 \* Math.PI);

},

（14）监听滑动手势，根据起始位置和终点位置坐标算出滑动角度，循环遍历出滑动数据，进行侧滑显示。添加如下两个方法代码：

//手指触摸动作开始 记录起点X坐标

touchstart: function (e) {

//开始触摸时 重置所有删除

this.data.list.forEach(function (v, i) {

if (v.isTouchMove)//只操作为true的

v.isTouchMove = false;

})

this.setData({

startX: e.changedTouches[0].clientX,

startY: e.changedTouches[0].clientY,

list: this.data.list

})

},

//滑动事件处理

touchmove: function (e) {

var that = this,

index = e.currentTarget.dataset.index,//当前索引

startX = that.data.startX,//开始X坐标

startY = that.data.startY,//开始Y坐标

touchMoveX = e.changedTouches[0].clientX,//滑动变化坐标

touchMoveY = e.changedTouches[0].clientY,//滑动变化坐标

//获取滑动角度

angle = that.angle({ X: startX, Y: startY }, { X: touchMoveX, Y: touchMoveY });

that.data.list.forEach(function (v, i) {

v.isTouchMove = false

//滑动超过30度角 return

if (Math.abs(angle) > 30) return;

if (i == index) {

if (touchMoveX > startX) //右滑

v.isTouchMove = false

else //左滑

v.isTouchMove = true

}

})

//更新数据

that.setData({

list: that.data.list

})

},

（15）添加删除商品deleteGoods网络请求方法，传入参数为列表索引，删除成功以后根据索引删除列表显示数据，更新列表。代码如下：

deleteGoods:function(index){

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'goods/deleteById',

method: 'POST',

data: {

goodsId: that.data.list[index].goodsId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('删除成功')

var list = that.data.list;

//移除列表中下标为index的项

list.splice(index, 1);

//更新列表的状态

that.setData({

list: list

});

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（16）添加删除按钮delete\_click单击事件，调用deleteGoods网络请求方法，添加代码如下：

//点击删除按钮事件

delete\_click: function (e) {

var that = this

wx.showModal({

title: '提示',

content: '是否删除？',

success: function (res) {

if (res.confirm) {

//获取列表中要删除项的下标

var index = e.target.dataset.index;

that.deleteGoods(index)

}

}

})

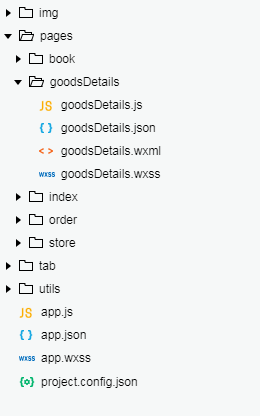
},

## 任务3.7 详情页

### 商户端

（1）打开商户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）新建pages/goodsDetails目录，在该目录下新建goodsDetails页面。项目结构如下图：



（3）打开store.js文件，添加收货地址按钮页面跳转方法，添加如下代码：

item\_click:function(e){

var index = e.currentTarget.dataset.index;

var goodsId = this.data.list[index].goodsId;

wx.navigateTo({

url: '../goodsDetails/goodsDetails?goodsId=' + goodsId,

})

}

（4）打开goodsDetails.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '商品详情'

})

（5）添加goodsId参数，在Page({代码上面添加如下代码：

var goodsId;

接收上一页面传递过来的值，在onLoad方法中添加如下代码：

goodsId = options.goodsId;

console.log("goodsId:" + goodsId)

（6）打开goodsDetails.wxml文件，绘制商品详情页面，添加如下代码：

<!--pages/goodsDetails/goodsDetails.wxml-->

<view class='container'>

<scroll-view class='view\_scroll'scroll-y>

<view class='content'>

<view>商品名称:</view>

<text>{{goodsDetail.name}}</text>

<view class='line'></view>

</view>

<view class='content'>

<view>封面图片：</view>

<image src=' {{imgHost}}{{goodsDetail.image}}' mode='widthFix'></image>

<view class='line'></view>

</view>

<view class='content'>

<view>商品简介:</view>

<text>{{goodsDetail.detail}}</text>

<view class='line'></view>

</view>

<view class='time'>

<view>上架时间:</view>

<text>{{goodsDetail.upTime}}</text>

<view class='line'></view>

</view>

</scroll-view>

<view class='view\_bottom'>

<view class='view\_bottom\_item'>

<text>单价：</text>

<text class='price'>￥{{goodsDetail.price}}</text>

</view>

<view class='view\_bottom\_item'>

<button class='edit' bindtap='edit\_click'>编辑</button>

</view>

</view>

</view>

（7）打开goodsDetails.wxss文件，为商品详情页添加样式文件，添加如下代码：

.view\_bottom{

display: flex;

justify-content: space-between;

width: 100%;

align-items: center;

border-top: 1rpx solid rgb(245, 245, 245);

border-bottom: 1rpx solid rgb(245, 245, 245);

position: absolute;

bottom: 0;

height: 10vh;

}

.view\_bottom\_item{

display: flex;

align-items: center;

margin-left: 20rpx;

}

.view\_bottom\_item text{

font-size: 30rpx;

}

.price{

color: orange;

margin-left: 20rpx;

}

.edit {

margin: 20rpx;

background: red;

color: white;

font-size: 30rpx;

height: 6vh;

line-height: 6vh;

text-align: center

}

.view\_scroll{

height: 90vh;

}

.content{

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

display: flex;

flex-direction: column;

align-items: center;

}

.time{

display: flex;

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.content view{

width: 100%;

}

.content text{

width: 100%;

}

.content image{

width: 100%;

margin-top: 10rpx;

}

（8）打开goodsDetails.js文件，添加页面绑定数据，在data字段下添加如下代码：

goodsDetail:null，

imgHost: getApp().globalData.host

（9）添加网络请求接口方法，根据上一页面传递过来的goodsId获取商品详情信息，在onShow方法中添加如下代码：

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'goods/goodsDetails',

method: 'POST',

data: {

goodsId: goodsId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

var result = res.data.data;

that.setData({

goodsDetail: result

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

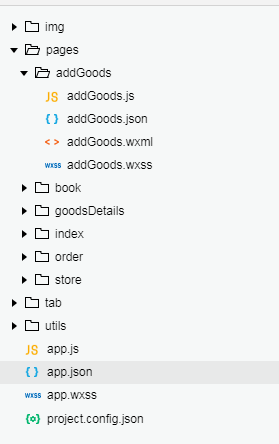
})

## 任务3.8 商品

### 商户端

（1）打开商户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）新建pages/addGoods目录，在该目录下新建addGoods页面。项目结构如下图：



（3）打开goodsDetails.js文件，添加编辑商品按钮页面跳转方法，添加如下代码：

edit\_click: function (e) {

var goodsDetailBean = JSON.stringify(this.data.goodsDetail)

wx.navigateTo({

url: '../addGoods/addGoods?goodsDetail=' + goodsDetailBean,

})

},

（4）打开store.js文件，添加新增商品按钮页面跳转方法，添加如下代码：

var goodsDetailBean = JSON.stringify(null)

wx.navigateTo({

url: '../addGoods/addGoods?goodsDetail=' + goodsDetailBean,

})

（5）打开addGoods.js文件，添加上一页面传递goodsId参数，在Page({代码之前添加如下代码：

var goodsDetail={};

（6）获取上一次页面传递参数，根据情况设置不同的标题，在onLoad方法中添加如下代码：

var goodsDetail = JSON.parse(options.goodsDetail);

console.log("goodsDetail:" + goodsDetail)

if (goodsDetail==null){

wx.setNavigationBarTitle({

title: '添加商品'

})

}else{

wx.setNavigationBarTitle({

title: '编辑商品'

})

}

（7）打开addGoods.wxml文件，绘制添加和编辑商品页，添加如下代码：

<!--pages/addGoods/addGoods.wxml-->

<view class='container'>

<form bindsubmit="formSubmit" class='formContaner'>

<scroll-view class='view\_scroll'scroll-y>

<view class='content'>

<view>商品名称:</view>

<input placeholder='请输入商品名称' name='name' value='{{goodsDetail.name}}'></input>

<view class='line'></view>

</view>

<view class='content'>

<view>封面图片：</view>

<image wx:if="{{noImage}}" src='../../img/add.jpg' mode='widthFix' bindtap='changeImg'></image>

<image wx:else src='{{imgHost}}{{imgUrl}}' mode='widthFix' bindtap='changeImg'></image>

<view class='line'></view>

</view>

<view class='content'>

<view>商品简介:</view>

<textarea placeholder='请输入商品简介' placeholder-class='placeholder' name='detail' value='{{goodsDetail.detail}}'></textarea>

<view class='line'></view>

</view>

</scroll-view>

<view class='view\_bottom'>

<view class='view\_bottom\_item'>

<text>单价：</text>

<input class='price' placeholder='请输入单价' name='price' type='number' value='{{goodsDetail.price}}'></input>

</view>

<view class='view\_bottom\_item'>

<button class='edit' type='primary' formType="submit">确定</button>

</view>

</view>

</form>

</view>

（8）打开addGoods.wxss文件，为添加和编辑商品页添加样式文件，添加如下代码：

/\* pages/addGoods/addGoods.wxss \*/

.view\_bottom{

display: flex;

justify-content: space-between;

width: 100%;

align-items: center;

border-top: 1rpx solid rgb(245, 245, 245);

border-bottom: 1rpx solid rgb(245, 245, 245);

position: absolute;

bottom: 0;

height: 10vh;

}

.formContaner{

width: 100%

}

.view\_bottom\_item{

display: flex;

align-items: center;

margin-left: 20rpx;

}

.view\_bottom\_item input{

font-size: 30rpx;

}

.price{

color: orange;

margin-left: 20rpx;

}

.edit{

margin: 20rpx;

background: red;

color: white;

font-size: 30rpx;

height: 6vh;

line-height: 6vh;

text-align: center

}

.view\_scroll{

height: 90vh;

}

.content{

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

display: flex;

flex-direction: column;

align-items: center;

}

.content view{

width: 100%;

}

.content input{

width: 100%;

}

.content textarea{

width: 100%;

}

.content image{

width: 100%;

margin-top: 10rpx;

}

（9）打开addGoods.js文件，添加页面绑定数据，在data字段下添加如下代码：

goodsDetail:null,

noImage:false，

imgHost: getApp().globalData.host，

imgUrl:""

（10）修改onLoad方法，在该方法中获取上一页面传递来的数据进行页面填充，替换所有代码如下：

var goodsDetail = JSON.parse(options.goodsDetail);

console.log("goodsDetail:" + goodsDetail)

var noImage=false;

if (goodsDetail==null){

wx.setNavigationBarTitle({

title: '添加商品'

})

goodsDetail={

image:"",

goodsId:''

};

noImage=true;

}else{

wx.setNavigationBarTitle({

title: '编辑商品'

})

noImage=false;

}

this.setData({

goodsDetail: goodsDetail,

noImage: noImage,

imgUrl: goodsDetail.image

})

（11）添加changeImg方法，用来添加或者重新选择封面图片，使用微信小程序自带的图片选择和上传文件的API，添加如下代码：

changeImg:function(){

var that=this;

wx.chooseImage({

success(res) {

wx.showLoading({

title: '加载中',

mask: true

})

const tempFilePaths = res.tempFilePaths

wx.uploadFile({

url: getApp().globalData.host + 'goods/uploadImage', //仅为示例，非真实的接口地址

filePath: tempFilePaths[0],

name: 'multFile',

header: {

'content-type': 'multipart/form-data' // 默认值

},

success(res) {

wx.hideLoading()

var data = JSON.parse(res.data);

console.log(data)

if (data.code == 200) {

console.log('上传成功')

that.setData({

imgUrl: data.data,

noImage:false

})

} else {

console.log('上传失败')

wx.showToast({

title: data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

console.log("fail")

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

},

})

}

})

},

（12）封装网络请求submitData方法，进行表单的提交操作。传入参数为页面表单数据，添加如下代码：

submitData: function (value) {

var goodsId = this.data.goodsDetail.goodsId;

console.log(goodsId)

var type=''

if (goodsId=='') {

type = 'add'

} else {

type = 'update'

}

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'goods/' + type,

method: 'POST',

data: {

goodsId: goodsId,

name: value.name,

detail: value.detail,

price: value.price,

image: this.data.imgUrl

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('操作成功')

wx.navigateBack()

wx.showToast({

title: "操作成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（13）为确认按钮添加formSubmit单击事件，获取页面表单数据，进行简单的表单验证，调用submitData方法，编辑或者添加商品信息。添加如下代码：

//确认按钮事件

formSubmit: function (e) {

console.log('form发生了submit事件，携带数据为：', e.detail.value)

var info = e.detail.value;

if (info.name == '') {

wx.showToast({

title: "请输入商品名称",

icon: 'none',

duration: 1000

})

return;

}

if (this.data.imgUrl == '') {

wx.showToast({

title: "请选择封面图片",

icon: 'none',

duration: 1000

})

return;

}

if (info.detail == '') {

wx.showToast({

title: "请输入商品简介",

icon: 'none',

duration: 1000

})

return;

}

if (info.price == '') {

wx.showToast({

title: "请输入商品单价",

icon: 'none',

duration: 1000

})

return;

}

this.submitData(info)

},

# 单元4 预约模块

## 任务4.1 预约列表

### 客户端

（1）打开book.wxml文件，绘制预约主页面，添加如下代码：

<view class='container'>

<view class='content' wx:for="{{list}}" wx:key="id" data-index="{{index}}" bindtap='item\_click'>

<view class='item\_left'>

<text>{{item.bookItem}}</text>

<text class='time'>到店时间：{{item.comeTime}}</text>

</view>

<view class='circle circle\_{{item.status}}'></view>

</view>

<button class='add' type='primary' catchtap='add\_click'>添加预约</button>

</view>

（2）打开book.wxss文件，为预约主页面添加样式，添加如下代码：

.add{

width: 70%;

margin: 20rpx 0;

}

.content{

width: 100%;

display: flex;

justify-content: space-between;

align-items: center;

border-bottom: 1px solid rgb(187, 178, 178);

padding: 20rpx

}

.item\_left text{

word-break:break-all;

}

.circle{

width: 20rpx;

height: 20rpx;

border-radius: 20rpx;

background: #000;

margin: 20rpx;

flex-shrink:0;

}

.circle\_0{

background: red

}

.circle\_1{

background: rgb(44, 8, 173)

}

.circle\_2{

background: rgb(12, 245, 109)

}

.circle\_3{

background: rgb(181, 6, 235)

}

.item\_left{

display: flex;

flex-direction: column;

margin-left: 20rpx;

}

.time{

font-size: 35rpx;

margin-top: 10rpx;

}

（3）打开book.js文件，动态设置当前页面标题。在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '预约'

})

（4）打开book.js文件，为页面添加绑定数据，在data字段下添加如下代码：

list:[],

（5）添加网络请求接口loadData方法，请求预约列表数据，添加如下代码:

loadData: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'book/getAllById',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

that.setData({

list: res.data.data

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（6）调用网络请求方法，在页面显示时请求页面所需数据，在onShow方法中中添加如下代码：

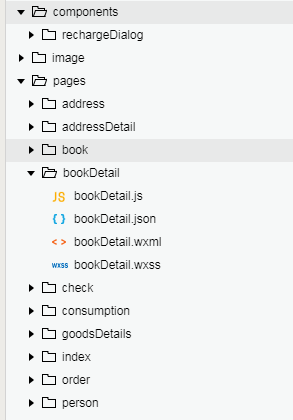
this.loadData();

## 任务4.2 预约详情

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/bookDetails目录，在该目录下新建bookDetails页面。项目结构如下图：



（3）打开book.js文件，添加预约详情按钮页面跳转方法，添加如下代码：

item\_click:function(e){

var index = e.currentTarget.dataset.index;

var bookId =this.data.list[index].bookId;

wx.navigateTo({

url: '../bookDetails/bookDetails?bookId=' + bookId ,

})

},

（4）打开bookDetails.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '预约详情'

})

（5）添加bookId参数，在Page({代码上面添加如下代码：

Var bookId;

接收上一页面传递过来的值，在onLoad方法中添加如下代码：

bookId = options.bookId;

console.log("bookId:" + bookId)

（6）打开bookDetails.wxml文件，绘制预约详情页面，添加如下代码：

<!--pages/bookDetails/bookDetails.wxml-->

<wxs module="m1">

var msg = function(val){

var status;

if(val==0){

status='已过期'

}else if(val==1){

status='已接受'

}else if(val==2){

status='已拒绝'

}else if(val==3){

status='未处理'

}

return status

}

module.exports.getStaus = msg;

</wxs>

<view class='container'>

<scroll-view class='view\_scroll'scroll-y>

<view class='status'>

<view>预约申请:</view>

<text class='status\_{{details.status}}'>{{m1.getStaus(details.status)}}</text>

</view>

<view class='content'>

<view>服务项目:</view>

<text>{{details.bookItem}}</text>

</view>

<view class='time'>

<view>预约时间:</view>

<text>{{details.bookTime}}</text>

</view>

<view class='time'>

<view>到店时间:</view>

<text>{{details.comeTime}}</text>

</view>

<view class='content'>

<view>留言:</view>

<text>{{details.message}}</text>

</view>

</scroll-view>

<view class='view\_bottom'hidden='{{details.isHidden}}'>

<text>预约已处理或者已失效</text>

<button class='delete' bindtap='delete\_click' >删除</button>

</view>

</view>

（7）打开bookDetails.wxss文件，为预约详情页添加样式文件，添加如下代码：

.view\_bottom{

display: flex;

justify-content: space-between;

width: 100%;

align-items: center;

border-top: 1rpx solid rgb(245, 245, 245);

border-bottom: 1rpx solid rgb(245, 245, 245);

position: absolute;

bottom: 0;

height: 10vh;

}

.view\_bottom text{

margin-left: 20rpx;

}

.delete{

margin: 20rpx;

background: red;

color: white;

font-size: 30rpx;

height: 6vh;

line-height: 6vh;

text-align: center

}

.view\_scroll{

height: 90vh;

}

.content{

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

display: flex;

flex-direction: column;

align-items: center;

}

.time{

display: flex;

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.status{

display: flex;

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

justify-content: space-between;

}

.status\_0{

color: red

}

.status\_1{

color: rgb(44, 8, 173)

}

.status\_2{

color: rgb(12, 245, 109)

}

.status\_3{

color: rgb(181, 6, 235)

}

.content view{

width: 100%;

}

.content text{

width: 100%;

}

（8）打开bookDetails.js文件，添加页面绑定数据，在data字段下添加如下代码：

details:null

（9）添加网络请求接口方法，根据上一页面传递过来的bookId获取预约详情信息，在console.log("bookId:" + bookId)代码后面添加如下代码：

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'book/details',

method: 'POST',

data: {

bookId: bookId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

var result = res.data.data;

if (result.status==0){

result.isHidden = false;

}else{

result.isHidden = true;

}

that.setData({

details: result

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

（10）封装删除预约信息的网络请求方法，传入参数为数据id。删除成功以后，返回上一页面。添加delete\_data方法，如下代码:

delete\_data:function(){

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'book/delete',

method: 'POST',

data: {

bookId: bookId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('删除成功')

wx.navigateBack({

delta: 1,

success: () => {

wx.showToast({

title: "删除成功",

icon: 'none',

duration: 1000

})

}

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（11）为删除预约按钮添加单击事件，添加delete\_click方法，如下代码:

delete\_click: function () {

var that=this;

wx.showModal({

title: '提示',

content: '是否删除该条预约信息？',

success: function (res) {

if (res.confirm) {

that.delete\_data()

}

}

})

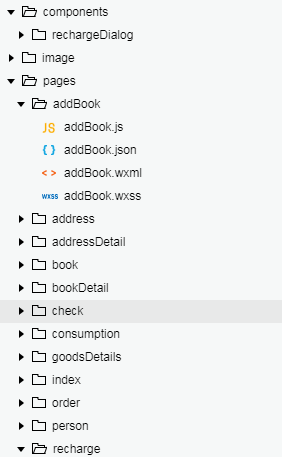
},

## 任务4.3 添加预约

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/addBook目录，在该目录下新建addBook页面。项目结构如下图：



（3）打开book.js文件，添加预约按钮页面跳转方法，添加如下代码：

add\_click:function(){

wx.navigateTo({

url: '../addBook/addBook',

})

},

（4）打开addBook.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '添加预约'

})

（5）打开addBook.wxml文件，绘制添加预约主页面，添加如下代码：

<view class='container'>

<form bindsubmit="formSubmit" class='formContaner'>

<scroll-view class='view\_scroll'scroll-y>

<view class='content'>

<view>服务项目:</view>

<input placeholder='请输入服务项目名称' name='name'></input>

</view>

<view class='content'>

<view>预约日期:</view>

<picker mode="date" value="{{date}}" start="{{start\_date}}" end="2020-01-01" bindchange="bindDateChange" name="date">

<view>当前选择: {{date}}</view>

</picker>

</view>

<view class='content'>

<view>预约时间:</view>

<picker mode="time" value="{{time}}" start="09:00" end="21:00" bindchange="bindTimeChange" name="time">

<view> 当前选择: {{time}}</view>

</picker>

</view>

<view class='content'>

<view>留言:</view>

<textarea placeholder='请输入留言' placeholder-class='placeholder' name='message'></textarea>

<view class='line'></view>

</view>

</scroll-view>

<view class='view\_bottom'>

<button class='sumbit' type='primary' formType="submit">确定</button>

</view>

</form>

</view>

（6）打开addBook.wxss文件，为添加预约主页面添加样式，添加如下代码：

.view\_bottom{

display: flex;

justify-content: center;

width: 100vw;

align-items: center;

border-top: 1rpx solid rgb(245, 245, 245);

border-bottom: 1rpx solid rgb(245, 245, 245);

position: absolute;

bottom: 0;

height: 10vh;

}

.formContaner{

width: 100%

}

.sumbit{

margin: 20rpx;

color: white;

font-size: 30rpx;

height: 6vh;

line-height: 6vh;

text-align: center;

width: 70vw

}

.view\_scroll{

height: 90vh;

}

.content{

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

display: flex;

flex-direction: column;

align-items: center;

}

.content picker{

width: 100%;

}

.content view{

width: 100%;

}

.content input{

width: 100%;

}

.content textarea{

width: 100%;

}

（7）为页面添加绑定数据，在data字段下添加如下代码：

start\_date:'',

date:'',

time:''

（8）获取当前时间，设置最早预约时间，在onLoad方法中添加如下代码：

var date = new Date();

var seperator1 = "-";

var seperator2 = ":";

var month = date.getMonth() + 1;

var strDate = date.getDate();

var minite = date.getMinutes();

var hour = date.getHours();

if (month >= 1 && month <= 9) {

month = "0" + month;

}

if (strDate >= 0 && strDate <= 9) {

strDate = "0" + strDate;

}

if (minite >= 0 && minite <= 9) {

minite = "0" + minite;

}

if (hour >= 0 && hour <= 9) {

hour = "0" + hour;

}

this.setData({

start\_date: date.getFullYear() + seperator1 + month + seperator1 + strDate,

date: date.getFullYear() + seperator1 + month + seperator1 + strDate,

time: hour + seperator2 + minite

})

（9）为时间和日期选择器添加滑动改变时间监听，添加代码如下：

bindDateChange: function (e) {

console.log('picker发送选择改变，携带值为', e.detail.value)

this.setData({

date: e.detail.value

})

},

bindTimeChange: function (e) {

console.log('picker发送选择改变，携带值为', e.detail.value)

this.setData({

time: e.detail.value

})

},

（10）封装网络请求submitData方法，进行表单的提交操作。传入参数为页面表单数据，添加如下代码：

submitData: function (value) {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'book/add ',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

comeTime: value.date + " " + value.time+":00",

bookItem : value.name,

message : value.message,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('添加成功')

wx.navigateBack()

wx.showToast({

title: "添加成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（11）为确认按钮添加formSubmit单击事件，获取页面表单数据，进行简单的表单验证，调用submitData方法，添加预约信息。添加如下代码：

formSubmit:function(e){

console.log('form发生了submit事件，携带数据为：', e.detail.value)

var info = e.detail.value;

if (info.name == '') {

wx.showToast({

title: "请输入服务项目名称",

icon: 'none',

duration: 1000

})

return;

}

if (info.message == '') {

wx.showToast({

title: "请输入留言",

icon: 'none',

duration: 1000

})

return;

}

this.submitData(info)

},

## 任务4.4 预约列表

### 商户端

（1）打开商户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）打开book.wxml文件，绘制预约列表主页面，添加如下代码：

<scroll-view class='scroll' scroll-y bindscrolltolower='loadMore' bindscrolltoupper='reflash' >

<view class='content' wx:for="{{list}}" wx:key="id" data-index="{{index}}" bindtap='item\_click'>

<view class='item\_left'>

<text>{{item.bookItem}}</text>

<text class='time'>到店时间：{{item.comeTime}}</text>

</view>

<view class='circle circle\_{{item.status}}'></view>

</view>

</scroll-view>

（3）打开book.wxss文件，为预约列表主页面添加样式文件，添加如下代码：

/\* pages/store/store.wxss \*/

.scroll{

height: 100vh;

}

.content{

display: flex;

justify-content: space-between;

align-items: center;

border-bottom: 1px solid rgb(187, 178, 178);

padding: 20rpx;

}

.item\_left text{

word-break:break-all;

}

.circle{

width: 20rpx;

height: 20rpx;

border-radius: 20rpx;

background: #000;

margin: 20rpx;

flex-shrink:0;

}

.circle\_0{

background: red

}

.circle\_1{

background: rgb(44, 8, 173)

}

.circle\_2{

background: rgb(12, 245, 109)

}

.circle\_3{

background: rgb(181, 6, 235)

}

.item\_left{

display: flex;

flex-direction: column;

margin-left: 20rpx;

}

.time{

font-size: 35rpx;

margin-top: 10rpx;

}

（4）打开book.js文件，动态设置当前页面标题。在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '预约'

})

（5）打开book.js文件，为页面添加绑定数据，在data字段下添加如下代码：

list:[],

（6）在Page({代码上面定义分页需要的变量，添加如下代码：

var currentPage=1 //当前页数

var pageSize= 10 //每页显示数量

（7）添加网络请求接口loadData方法，请求商品列表数据，添加如下代码：

loadData: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'book/getAllBooks',

method: 'POST',

data: {

page: currentPage,

pageSize: pageSize,

flag:true

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res)

if (res.data.code == 200) {

console.log('查询成功')

var result = res.data.data;

var list\_info = that.data.list;

for (var i = 0; i < result.length; i++) {

list\_info.push(result[i]);

}

that.setData({

list: list\_info

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（8）调用网络请求方法，在页面显示时请求页面所需数据，在onShow方法中中添加如下代码：

currentPage=1;

this.setData({

list:[]

})

this.loadData();

（9）新增下拉刷新reflash方法，初始化currentPage为1，将list列表制空。添加如下代码：

reflash:function(){

console.log("reflash ------")

currentPage=1;

this.setData({

list: []

})

this.loadData();

},

（10）新增上拉加载loadMor方法，currentPage参数自加加，添加如下代码：

loadMore : function () {

console.log("loadMore ------")

currentPage++;

this.loadData();

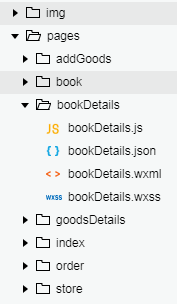
},

## 任务4.5 详情页

### 商户端

（1）打开商户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）新建pages/bookDetails目录，在该目录下新建bookDetails页面。项目结构如下图：



（3）打开book.js文件，添加预约详情按钮页面跳转方法，添加如下代码：

item\_click:function(e){

var index = e.currentTarget.dataset.index;

var bookId =this.data.list[index].bookId;

wx.navigateTo({

url: '../bookDetails/bookDetails?bookId=' + bookId ,

})

},

（4）打开bookDetails.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '预约详情'

})

（5）添加bookId参数，在Page({代码上面添加如下代码：

Var bookId;

接收上一页面传递过来的值，在onLoad方法中添加如下代码：

bookId = options.bookId;

console.log("bookId:" + bookId)

（6）打开bookDetails.wxml文件，绘制预约详情页面，添加如下代码：

<!--pages/bookDetails/bookDetails.wxml-->

<wxs module="m1">

var msg = function(val){

var status;

if(val==0){

status='已过期'

}else if(val==1){

status='已接受'

}else if(val==2){

status='已拒绝'

}else if(val==3){

status='未处理'

}

return status

}

module.exports.getStaus = msg;

</wxs>

<view class='container'>

<scroll-view class='view\_scroll'scroll-y>

<view class='status'>

<view>预约申请:</view>

<text class='status\_{{details.status}}'>{{m1.getStaus(details.status)}}</text>

</view>

<view class='content'>

<view>服务项目:</view>

<text>{{details.bookItem}}</text>

</view>

<view class='time'>

<view>预约时间:</view>

<text>{{details.bookTime}}</text>

</view>

<view class='time'>

<view>到店时间:</view>

<text>{{details.comeTime}}</text>

</view>

<view class='content'>

<view>留言:</view>

<text>{{details.message}}</text>

</view>

</scroll-view>

<view class='view\_bottom' hidden='{{details.isHidden}}'>

<text>预约待处理</text>

<view class='view\_bottom\_left'>

<button class='accept' data-current="1" bindtap='change\_click' >接受</button>

<button class='refused' data-current="2" bindtap='change\_click' >拒绝</button>

</view>

</view>

</view>

（7）打开bookDetails.wxss文件，为预约详情页添加样式文件，添加如下代码：

.view\_bottom{

display: flex;

justify-content: space-between;

width: 100%;

align-items: center;

border-top: 1rpx solid rgb(245, 245, 245);

border-bottom: 1rpx solid rgb(245, 245, 245);

position: absolute;

bottom: 0;

height: 10vh;

}

.view\_bottom\_left{

display: flex;

justify-content: space-between;

}

.view\_bottom text{

margin-left: 20rpx;

}

.refused{

margin: 20rpx;

background: red;

color: white;

font-size: 30rpx;

height: 6vh;

line-height: 6vh;

text-align: center

}

.accept{

margin: 20rpx;

background: green;

color: white;

font-size: 30rpx;

height: 6vh;

line-height: 6vh;

text-align: center

}

.view\_scroll{

height: 90vh;

}

.content{

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

display: flex;

flex-direction: column;

align-items: center;

}

.time{

display: flex;

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.status{

display: flex;

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

justify-content: space-between;

}

.status\_0{

color: red

}

.status\_1{

color: rgb(44, 8, 173)

}

.status\_2{

color: rgb(12, 245, 109)

}

.status\_3{

color: rgb(181, 6, 235)

}

.content view{

width: 100%;

}

.content text{

width: 100%;

}

（8）打开bookDetails.js文件，添加页面绑定数据，在data字段下添加如下代码：

details:null

（9）添加网络请求接口方法，根据上一页面传递过来的bookId获取预约详情信息，在console.log("bookId:" + bookId)代码后面添加如下代码：

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'book/details',

method: 'POST',

data: {

bookId: bookId,

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

var result = res.data.data;

if (result.status==3){

result.isHidden = false;

}else{

result.isHidden = true;

}

that.setData({

details: result

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

（10）添加接收和拒绝按钮的单击事件change\_click方法，添加如下代码：

change\_click:function(e){

var status=e.target.dataset.current;

console.log("status:" + status)

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'book/update',

method: 'POST',

data: {

bookId: bookId,

status: status

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('操作成功')

wx.navigateBack()

wx.showToast({

title: "操作成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

# 单元5 订单模块

## 任务5.1 列表

### 1.客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）打开order.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '订单列表'

})

（3）打开order.wxml文件，绘制订单列表页面，添加如下代码：

<!--pages/order/order.wxml-->

<view class="swiper-tab">

<view class="tab-list-left {{currentTab==1 ? 'on' : ''}}" data-current="1" bindtap="swichNav">待发货</view>

<view class="tab-list-right {{currentTab==2 ? 'on' : ''}}" data-current="2" bindtap="swichNav">待收货 </view>

<view class="tab-list-right {{currentTab==3 ? 'on' : ''}}" data-current="3" bindtap="swichNav">已完成 </view>

</view>

<view class="content" wx:for="{{list}}" wx:key="id" data-index="{{index}}" bindtap='item\_click'>

<view class='item'>

<text class='order'>订单号：<text class='order\_no'>{{item.orderNum}}</text></text>

<text class='time'>{{item.createTime}}</text>

</view>

<view class='item\_right'>

<text class='money'>￥{{item.money}}</text>

<view hidden="{{currentTab!=2}}">

<button class='confirm' data-index="{{index}}"

catchtap ='confirm\_click'>确认收货</button>

</view>

</view>

</view>

（4）打开consumption.wxss文件，为消费记录页面添加样式，添加如下代码：

/\* pages/order/order.wxss \*/

.content{

display: flex;

justify-content:space-between;

border-bottom: 1px solid rgb(245, 245, 245);

align-items:center;

}

.money{

color: rgb(19, 148, 19);

}

.item{

display: flex;

flex-direction: column;

margin: 20rpx;

}

.item\_right{

display: flex;

flex-direction: column;

text-align: right;

align-items:flex-end;

margin: 20rpx;

}

.confirm{

font-size: 20rpx;

width: 150rpx;

background: red;

color: white;

}

.order{

font-size: 30rpx;

}

.order\_no{

font-size: 30rpx;

color: blue;

}

.time{

font-size: 20rpx;

margin-top: 10rpx;

}

.swiper-tab {

text-align: center;

line-height:10vh;

height: 10vh;

margin-top: auto

}

.tab-list-left {

font-size: 30rpx;

display: inline-block;

width: 25%;

border: 1px solid #bbada0;

line-height: 60rpx;

color: #bbada0;

}

.tab-list-right {

line-height: 60rpx;

font-size: 30rpx;

display: inline-block;

width: 25%;

border: 1px solid #bbada0;

color: #bbada0;

}

.on {

color: #fff;

background-color: #bbada0;

}

（5）打开order.js文件，添加页面绑定数据，在data字段下添加如下代码：

list:null,

currentTab:1,

（6）新增loadData网络请求方法，根据状态获取网络数据，添加如下代码：

loadData: function () {

wx.showNavigationBarLoading()

var that = this;

wx.request({

url: getApp().globalData.host + 'order/showByStatus',

method: 'POST',

data: {

userId: getApp().globalData.userInfo.userId,

status: this.data.currentTab

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideNavigationBarLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('请求成功')

that.setData({

list: res.data.data

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideNavigationBarLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（7）新增按钮单击事件方法，根据按钮状态设定请求参数，添加代码如下：

swichNav: function (e) {

this.setData({

currentTab: e.target.dataset.current,

})

this.loadData();

},

（8）进入页面用默认参数请求数据，在onShow方法中添加代码如下：

this.setData({

list:[]

})

this.loadData();

### 2.商户端

（1）打开商户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）打开order.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '订单列表'

})

（3）打开order.wxml文件，绘制订单列表页面，添加如下代码：

<view class="swiper-tab">

<view class="tab-list-left {{currentTab==1 ? 'on' : ''}}" data-current="1" bindtap="swichNav">待发货</view>

<view class="tab-list-right {{currentTab==2 ? 'on' : ''}}" data-current="2" bindtap="swichNav">待收货 </view>

<view class="tab-list-right {{currentTab==3 ? 'on' : ''}}" data-current="3" bindtap="swichNav">已完成 </view>

</view>

<view class="content" wx:for="{{list}}" wx:key="id" data-index="{{index}}" bindtap='item\_click'>

<view class='item'>

<text class='order'>订单号：<text class='order\_no'>{{item.orderNum}}</text></text>

<text class='time'>{{item.createTime}}</text>

</view>

<view class='item\_right'>

<text class='money'>￥{{item.money}}</text>

<view hidden="{{currentTab!=1}}">

<button class='confirm' data-index="{{index}}"

catchtap ='confirm\_click'>确认发货</button>

</view>

</view>

</view>

（4）打开consumption.wxss文件，为消费记录页面添加样式，添加如下代码：

/\* pages/order/order.wxss \*/

.content{

display: flex;

justify-content:space-between;

border-bottom: 1px solid rgb(245, 245, 245);

align-items:center;

}

.money{

color: rgb(19, 148, 19);

}

.item{

display: flex;

flex-direction: column;

margin: 20rpx;

}

.item\_right{

display: flex;

flex-direction: column;

text-align: right;

align-items:flex-end;

margin: 20rpx;

}

.confirm{

font-size: 20rpx;

width: 150rpx;

background: red;

color: white;

}

.order{

font-size: 30rpx;

}

.order\_no{

font-size: 30rpx;

color: blue;

}

.time{

font-size: 20rpx;

margin-top: 10rpx;

}

.swiper-tab {

text-align: center;

line-height:10vh;

height: 10vh;

margin-top: auto

}

.tab-list-left {

font-size: 30rpx;

display: inline-block;

width: 25%;

border: 1px solid #bbada0;

line-height: 60rpx;

color: #bbada0;

}

.tab-list-right {

line-height: 60rpx;

font-size: 30rpx;

display: inline-block;

width: 25%;

border: 1px solid #bbada0;

color: #bbada0;

}

.on {

color: #fff;

background-color: #bbada0;

}

（5）打开order.js文件，添加页面绑定数据，在data字段下添加如下代码：

list:null,

currentTab:1,

（6）新增loadData网络请求方法，根据状态获取网络数据，添加如下代码：

loadData: function () {

wx.showNavigationBarLoading()

var that = this;

wx.request({

url: getApp().globalData.host + 'order/showAllByStatus',

method: 'POST',

data: {

status: this.data.currentTab

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideNavigationBarLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('请求成功')

that.setData({

list: res.data.data

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideNavigationBarLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（7）新增按钮单击事件方法，根据按钮状态设定请求参数，添加代码如下：

swichNav: function (e) {

this.setData({

currentTab: e.target.dataset.current,

})

this.loadData();

},

（8）进入页面用默认参数请求数据，在onShow方法中添加代码如下：

this.setData({

list:[]

})

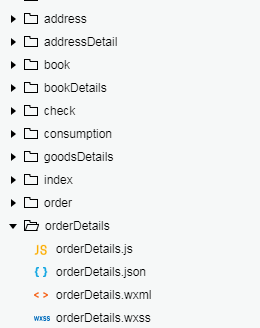
this.loadData();

## 任务5.2 订单详情

### 客户端

（1）打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）新建pages/orderDetails目录，在该目录下新建orderDetails页面。项目结构如下图：



（3）打开order.js文件，添加预约详情按钮页面跳转方法，添加如下代码：

item\_click: function (e) {

var index = e.currentTarget.dataset.index;

var orderNum = this.data.list[index].orderNum;

wx.navigateTo({

url: '../orderDetails/orderDetails?orderNum=' + orderNum,

})

},

（4）打开orderDetails.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '订单详情'

})

（5）添加orderNum参数，在Page({代码上面添加如下代码：

var orderNum;

接收上一页面传递过来的值，在onLoad方法中添加如下代码：

orderNum = options. orderNum;

console.log("orderNum:" + orderNum)

（6）打开orderDetails.wxml文件，绘制订单详情页面，添加如下代码：

<wxs module="m1">

var msg = function(val){

var status;

if(val==1){

status='待发货'

}else if(val==2){

status='待收货'

}else if(val==3){

status='已完成'

}

return status

}

module.exports.getStaus = msg;

</wxs>

<scroll-view scroll-y class='content'>

<view class='order\_no'>

<text>订单编号：{{orderDetail.order.orderNum}}</text>

<text class='status'>{{m1.getStaus(orderDetail.order.status)}}</text>

</view>

<view class='time'>创建时间：{{orderDetail.order.createTime}}</view>

<text class='title'>收货地址：</text>

<view class='view\_item' bindtap='address\_click'>

<view class='view\_item\_left'>

<text>{{orderDetail.address.name}}</text>

<text>{{orderDetail.address.phone}}</text>

</view>

<view class='address'>

<text >{{orderDetail.address.province+orderDetail.address.city+orderDetail.address.district+orderDetail.address.detail}}</text>

</view>

</view>

<text class='title'>商品列表：</text>

<block wx:for="{{orderDetail.item\_list}}">

<view class='list\_item' data-index="{{index}}" bindtap='item\_click'>

<view class='list\_item\_left'>

<image class="logo" mode="scaleToFill" src='{{item.image}}'></image>

<view class='list\_item\_middle'>

<text>{{item.name}}</text>

<view class='item\_price'>

<text class='price'>￥{{item.price}}</text>

<text>\*{{item.num}}</text>

</view>

</view>

</view>

</view>

</block>

<view class='total\_price'>

<text >总价：{{orderDetail.order.money}}</text>

</view>

<text class='title'>备注：</text>

<view class='address'>{{orderDetail.order.message}}</view>

<button class='submit' hidden='{{orderDetail.order.status!=2}}' catchtap='submit\_click'>确认收货</button>

</scroll-view>

（7）打开orderDetails.wxss文件，为订单详情页添加样式文件，添加如下代码：

/\* pages/orderDetail/orderDetail.wxss \*/

.view\_item{

display: flex;

justify-content: space-between;

flex-direction: column;

width: 100vw;

border-bottom:1rpx solid rgb(238, 238, 238);

}

.time{

padding: 20rpx;

border-bottom:1rpx solid rgb(238, 238, 238);

}

.view\_item\_left{

display: flex;

justify-content: space-between;

margin: 20rpx;

}

.address{

margin: 0 0 20rpx 20rpx;

box-sizing:border-box;

word-break: break-all

}

.title{

width: 100%;

margin: 20rpx;

}

.list\_item{

display: flex;

align-items: center;

justify-content: space-between;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.logo {

width: 120rpx;

height: 120rpx;

border-radius: 20rpx;

padding: 20rpx;

}

.list\_item\_left{

display: flex;

align-items: center;

}

.list\_item\_middle{

display: flex;

flex-direction: column;

width: 70vw

}

.list\_item\_middle text{

white-space: nowrap;

text-overflow:ellipsis;

display: block;

overflow: hidden;

}

.price{

color: orange;

}

.item\_price{

display: flex;

justify-content: space-between;

align-items: center;

margin-top: 20rpx;

}

.content{

display: flex;

flex-direction: column;

width: 100%;

}

.check{

position: absolute;

bottom: 0;

width: 100%;

text-align: center;

display: flex;

align-items: center;

justify-content: center;

height: 10vh;

}

.check button{

width: 70%;

}

.total\_price{

margin: 20rpx;

text-align: right

}

.total\_price text{

color: green

}

.status{

color: blue

}

.order\_no{

display: flex;

justify-content: space-between;

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.submit{

width: 70%;

margin-bottom: 50rpx;

margin-top: 50rpx;

}

（8）打开orderDetails.js文件，添加页面绑定数据，在data字段下添加如下代码：

orderDetail:null

（9）新增loadData网络请求方法，根据状态获取网络数据，添加如下代码：

loadData: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'order/details',

method: 'POST',

data: {

orderNum: orderNum

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

var orderDetais = res.data.data;

for (var i = 0; i < orderDetais.item\_list.length; i++) {

orderDetais.item\_list[i].image = getApp().globalData.host + orderDetais.item\_list[i].image;

}

that.setData({

orderDetail: orderDetais

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

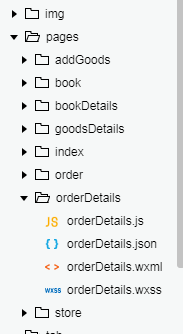
（10）添加网络请求接口方法，在console.log("orderNum:" + orderNum)代码后面添加如下代码：

this.loadData();

### 商户端

（1）打开商户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）新建pages/orderDetails目录，在该目录下新建orderDetails页面。项目结构如下图：



（3）打开order.js文件，添加预约详情按钮页面跳转方法，添加如下代码：

item\_click: function (e) {

var index = e.currentTarget.dataset.index;

var orderNum = this.data.list[index].orderNum;

wx.navigateTo({

url: '../orderDetails/orderDetails?orderNum=' + orderNum,

})

},

（4）打开orderDetails.js文件，动态设置当前页面标题，在onReady方法中添加如下代码：

wx.setNavigationBarTitle({

title: '订单详情'

})

（5）添加orderNum参数，在Page({代码上面添加如下代码：

var orderNum;

接收上一页面传递过来的值，在onLoad方法中添加如下代码：

orderNum = options. orderNum;

console.log("orderNum:" + orderNum)

（6）打开orderDetails.wxml文件，绘制订单详情页面，添加如下代码：

<!--pages/orderDetail/orderDetail.wxml-->

<wxs module="m1">

var msg = function(val){

var status;

if(val==1){

status='待发货'

}else if(val==2){

status='待收货'

}else if(val==3){

status='已完成'

}

return status

}

module.exports.getStaus = msg;

</wxs>

<scroll-view scroll-y class='content'>

<view class='order\_no'>

<text>订单编号：{{orderDetail.order.orderNum}}</text>

<text class='status'>{{m1.getStaus(orderDetail.order.status)}}</text>

</view>

<view class='time'>创建时间：{{orderDetail.order.createTime}}</view>

<text class='title'>收货地址：</text>

<view class='view\_item' bindtap='address\_click'>

<view class='view\_item\_left'>

<text>{{orderDetail.address.name}}</text>

<text>{{orderDetail.address.phone}}</text>

</view>

<view class='address'>

<text >{{orderDetail.address.province+orderDetail.address.city+orderDetail.address.district+orderDetail.address.detail}}</text>

</view>

</view>

<text class='title'>商品列表：</text>

<block wx:for="{{orderDetail.item\_list}}">

<view class='list\_item' data-index="{{index}}" bindtap='item\_click'>

<view class='list\_item\_left'>

<image class="logo" mode="scaleToFill" src='{{item.image}}'></image>

<view class='list\_item\_middle'>

<text>{{item.name}}</text>

<view class='item\_price'>

<text class='price'>￥{{item.price}}</text>

<text>\*{{item.num}}</text>

</view>

</view>

</view>

</view>

</block>

<view class='total\_price'>

<text >总价：{{orderDetail.order.money}}</text>

</view>

<text class='title'>备注：</text>

<view class='address'>{{orderDetail.order.message}}</view>

<button class='submit' hidden='{{orderDetail.order.status!=1}}' catchtap='submit\_click'>确认发货</button>

</scroll-view>

（7）打开orderDetails.wxss文件，为订单详情页添加样式文件，添加如下代码：

/\* pages/orderDetail/orderDetail.wxss \*/

.view\_item{

display: flex;

justify-content: space-between;

flex-direction: column;

width: 100vw;

border-bottom:1rpx solid rgb(238, 238, 238);

}

.time{

padding: 20rpx;

border-bottom:1rpx solid rgb(238, 238, 238);

}

.view\_item\_left{

display: flex;

justify-content: space-between;

margin: 20rpx;

}

.address{

margin: 0 0 20rpx 20rpx;

box-sizing:border-box;

word-break: break-all

}

.title{

width: 100%;

margin: 20rpx;

}

.list\_item{

display: flex;

align-items: center;

justify-content: space-between;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.logo {

width: 120rpx;

height: 120rpx;

border-radius: 20rpx;

padding: 20rpx;

}

.list\_item\_left{

display: flex;

align-items: center;

}

.list\_item\_middle{

display: flex;

flex-direction: column;

width: 70vw

}

.list\_item\_middle text{

white-space: nowrap;

text-overflow:ellipsis;

display: block;

overflow: hidden;

}

.price{

color: orange;

}

.item\_price{

display: flex;

justify-content: space-between;

align-items: center;

margin-top: 20rpx;

}

.content{

display: flex;

flex-direction: column;

width: 100%;

}

.check{

position: absolute;

bottom: 0;

width: 100%;

text-align: center;

display: flex;

align-items: center;

justify-content: center;

height: 10vh;

}

.check button{

width: 70%;

}

.total\_price{

margin: 20rpx;

text-align: right

}

.total\_price text{

color: green

}

.status{

color: blue

}

.order\_no{

display: flex;

justify-content: space-between;

padding: 20rpx;

border-bottom: 1rpx solid rgb(245, 245, 245);

}

.submit{

width: 70%;

margin-bottom: 50rpx;

margin-top: 50rpx;

}

（8）打开orderDetails.js文件，添加页面绑定数据，在data字段下添加如下代码：

orderDetail:null

（9）新增loadData网络请求方法，根据状态获取网络数据，添加如下代码：

loadData: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'order/details',

method: 'POST',

data: {

orderNum: orderNum

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('查询成功')

var orderDetais = res.data.data;

for (var i = 0; i < orderDetais.item\_list.length; i++) {

orderDetais.item\_list[i].image = getApp().globalData.host + orderDetais.item\_list[i].image;

}

that.setData({

orderDetail: orderDetais

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（10）添加网络请求接口方法，在console.log("orderNum:" + orderNum)代码后面添加如下代码：

this.loadData();

## 任务5.3 订单操作

### 客户端

**（1）**打开客户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2ouser.zip的项目。

（2）打开order.js文件，添加确认发货按钮confirm\_click单击事件，添加如下代码：

confirm\_click:function(e){

console.log(e);

var index = e.target.dataset.index//当前索引

var orderNum = this.data.list[index].orderNum

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'order/updateStatus',

method: 'POST',

data: {

orderNum: orderNum,

status : '3'

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('修改成功成功')

wx.showToast({

title: '发货成功',

icon: 'none',

duration: 1000

})

that.loadData();

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（3）打开orderDetail.js文件，添加确认发货按钮submit \_click单击事件，添加如下代码：

submit\_click: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'order/updateStatus',

method: 'POST',

data: {

orderNum: orderNum,

status: '3'

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('修改成功成功')

wx.navigateBack()

wx.showToast({

title: "发货成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

### 2.商户端

**（1）**打开商户端源代码，可以打开本地上一章节完成的项目，也可以打开资源包里o2oadmin.zip的项目。

（2）打开order.js文件，添加确认发货按钮confirm\_click单击事件，添加如下代码：

confirm\_click:function(e){

console.log(e);

var index = e.target.dataset.index//当前索引

var orderNum = this.data.list[index].orderNum

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'order/updateStatus',

method: 'POST',

data: {

orderNum: orderNum,

status : '2'

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('修改成功成功')

wx.showToast({

title: '发货成功',

icon: 'none',

duration: 1000

})

that.loadData();

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},

（3）打开orderDetail.js文件，添加确认发货按钮submit \_click单击事件，添加如下代码：

submit\_click: function () {

wx.showLoading({

title: '加载中',

mask: true

})

var that = this;

wx.request({

url: getApp().globalData.host + 'order/updateStatus',

method: 'POST',

data: {

orderNum: orderNum,

status: '2'

},

header: {

'content-type': 'application/json' // 默认值

},

success: function (res) {

wx.hideLoading()

console.log(res.data)

if (res.data.code == 200) {

console.log('修改成功成功')

wx.navigateBack()

wx.showToast({

title: "发货成功",

icon: 'none',

duration: 1000

})

} else {

wx.showToast({

title: res.data.message,

icon: 'none',

duration: 1000

})

}

},

fail: function (res) {

wx.hideLoading()

wx.showToast({

title: res.errMsg,

icon: 'none',

duration: 1000

})

}

})

},