44 我的，都是我的：开发实现个人中心页面

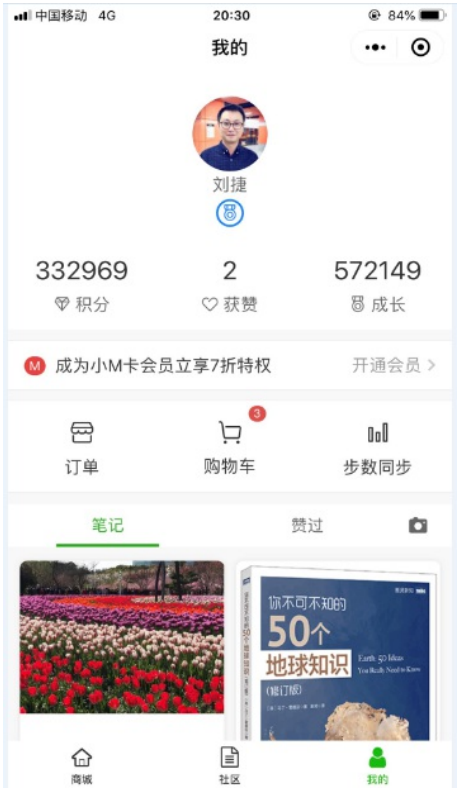
更新时间：2019-10-09 09:08:39

> 理想必须要人们去实现它，它不但需要决心和勇敢而且需要知识。
>
> ——吴玉章

以第二节整理的功能需求为基础，我们即可着手实现个人中心页面。

在上一节中，我们已经使用“分类拆解法”的拆解步骤，对个人中心页面（如图 2 所示）进行了详细拆解。本节我们将按照“分类拆解法”的编程步骤，完成页面代码编写。

图 2 个人中心页面



1. 数据服务

通过梳理上一节“2. 与其它功能模块的关系维度”中需要使用其它功能模块的数据，我们可以一一对应的找到在前面章节已实现的数据服务来提供相应数据：

用户成长体系

用户的当前总成长值：调用用户表数据服务 `UserService` 的 `getUserInfo` 方法获取

用户当前等级：调用用户等级与等级特权表数据服务 `LevelService` 的 `getLevelList` 方法获取用户等级列表后，根据用户的当前总成长值在用户等级列表中查询得到

积分体系

用户的当前可用积分：调用用户表数据服务 `UserService` 的 `getUserInfo` 方法获取

付费会员体系

用户的付费会员有效期：调用用户表数据服务 `UserService` 的 `getUserInfo` 方法获取

商城

用户的购物车商品数量：使用小程序官方提供的数据缓存 API 读取微信客户端的购物车缓存

UGC 社区

用户发表的笔记列表：调用笔记信息表数据服务 `NoteService` 的 `getNoteList` 方法获取

用户点赞的笔记列表：调用点赞记录表数据服务 `UpvoteService` 的 `getMyUpvoteList` 方法获取

在已实现的数据服务中唯一找不到的方法是：从点赞记录表中获取笔记总获赞数（用户发表的所有笔记的点赞数之和）。

因此，我们需要在点赞记录表数据服务 `UpvoteService` 中添加一个方法 `getMyBeUpvotedNum` 来获取用户发表笔记的总获赞数：

```
/**
 * 从数据库获取用户发表笔记的总获赞数
 * @method getMyBeUpvotedNum
 * @for UpvoteService
 * @param {function} successCallback(num) 处理数据查询结果的回调函数
 * num: 用户发表笔记的总获赞数
 */
getMyBeUpvotedNum(successCallback) {
  //在点赞记录表中查找作者OpenId字段中值为用户OpenId的记录数，该数字即用户发表笔记的总获赞数
  db.collection('upvote').where({
      autherOpenid: app.globalData.openid
    })
    .count()
    .then(res => {
      //回调函数处理数据查询结果，res.total即用户发表笔记的总获赞数
      typeof successCallback == "function" && successCallback(res.total)
    })
    .catch(err => {
      //访问数据库失败 的 系统异常处理
      //跳转出错页面
      wx.redirectTo({
        url: '../../errorpage/errorpage'
      })
      console.error(err)
    })
}
```

2. 编程步骤

在编写好数据服务后，就可以根据拆解结果一步步完成页面的代码编写。

个人中心页面读取的数据较多，需要设置一个从数据库获取完数据的标志 `inited` ，当从数据库获取到数据后再显示页面。：

```
data: {
  inited: false, //是否已从数据库读取数据
},
```

2.1 定义页面子部件及其排列顺序

个人中心页面可以拆解为 4 个子部件：

- 子部件 1 ：用户信息栏
- 子部件 2 ：小 M 卡会员入口栏
- 子部件 3 ：常用功能菜单栏
- 子部件 4 ：笔记栏

首先在 WXML 页面模板中定义 4 个子部件的容器 ：

```
<!-- 当从数据库获取完全部数据后，再显示页面-->
<view wx:if="{{inited}}">
 <!-- 子部件 1 ：用户信息栏 设置白色背景-->
 <view class="bg-white padding-top-lg">
 </view>
 <!-- 子部件 2 ：小M卡会员入口栏 使用 WeUI 的 panel -->
 <view class="weui-panel weui-panel_access">
 </view>
 <!-- 子部件 3 ：常用功能菜单栏 使用 WeUI 的 panel -->
 <view class="weui-panel weui-panel_access">
 </view>
 <!-- 子部件 4 ：笔记栏 使用 WeUI 的 navbar -->
 <view class="weui-tab margin-top-sm">
 </view>
</view>
```

2.2 实现子部件 1 ：用户信息栏

在本章第二节已经将子部件 1 拆解为一系列的显示元素：

用户头像

交互界面，无事件，数据为微信用户头像

用户昵称

交互界面，无事件，数据为微信用户昵称

用户的等级图标

交互界面，无事件，数据为用户当前等级

用户等级是计算值，根据用户当前总成长值，从用户等级与等级特权表中计算得出

当前可用积分

交互界面，事件为用户点击事件，数据为用户表中用户当前可用积分字段

用户点击事件的事件响应为：页面跳转到积分体系页面（在第六章第四节开发实现）

笔记总获赞数

交互界面，无事件，数据为用户发表的所有笔记的点赞数之和

用户发表的所有笔记的点赞数之和是计算值，从点赞记录表中查询得出

当前总成长值

交互界面，事件为用户点击事件，数据为用户表中用户当前总成长值字段

用户点击事件的事件响应为：页面跳转到用户成长体系页面（在第五章第四节开发实现）

首先我们要根据 “1. 数据服务” 的整理结果定义 data 中的数据：

```
data: {
  myLevel: {}, //用户等级
  myInfo: {}, //用户信息
  myBeUpvotedNum: 0, //用户发表笔记的总获赞数
  isMembershipExpired: true, //用户是否是小M卡会员（还在会员有效期中）
  cart: [], //购物车数据
},
```

然后调用数据服务在页面加载时获取数据：

```
/**
 * 生命周期函数--监听页面加载
 */
onLoad: function(options) {
  var that = this
  //获取数据库中的用户数据
  this.setUserData()
  //读取微信客户端缓存的购物车数据
  this.initCart()
},

/**
 * 从微信客户端的购物车数据缓存中读取购物车商品数据
 */
initCart: function() {
  var value = wx.getStorageSync('cart');
  if (value) {
    this.setData({
      cart: value
    });
  } else {
    //这里比第八章第七节的 initCart 方法多了一个判断，原因见下文特殊场景
    this.setData({
      cart: []
    });
  }
},

/**
 * 调用数据服务获取数据库中的用户数据
 */
setUserData: function() {
  var that = this
  //调用数据服务获取用户信息、用户等级、笔记总获赞数等数据
  levelService.getLevelList(
    //获取成长体系中的所有成长等级回调函数
    function(levelList) {
      //用户等级列表
      var levels = levelList
      userService.getUserInfo(
        //获取用户信息回调函数
        function(userinfo) {
          //用户信息
          var myInfo = userinfo
          upvoteService.getMyBeUpvotedNum(
            //获取用户发表笔记的总获赞数回调函数
            function(num) {
              //获取用户等级
              var myLevel = levels.filter(e => e.minGrowthValue <= myInfo.growthValue && myInfo.growthValue <= e.maxGrowthValue)[0]
              //根据付费会员有效期计算用户是否是小M卡会员
              var isMembershipExpired = myInfo.memberExpDate < new Date()
              //设置用户数据
              that.setData({
                myLevel: myLevel,
                myInfo: myInfo,
                myBeUpvotedNum: num, //用户发表笔记的总获赞数
                isMembershipExpired: isMembershipExpired
              })
              //在读取完所有数据后，设置从数据库读取数据完成标志
              that.setData({
                inited: true
              })
            })
        })
    })
},
```

在第七章第三节和第八章第六节中我们均提到过一个特殊场景：由于小程序的页面路由机制，点击小程序左上角的返回按钮后只是将前一页面出栈显示出来，并不会重新加载页面（即不会触发 **onLoad** 事件）。

在个人中心页面同样存在特殊场景：

- 当用户从个人中心进入微信运动同步页面，在微信运动同步页面同步微信运动步数后（此时用户的积分和成长值已经发生变化），再点击小程序左上角的返回按钮返回个人中心页面时，应该显示最新的积分与成长值信息；
- 当用户从个人中心进入购物车页面，在购物车中删除商品后再返回个人中心页面，应该显示最新的购物车商品数量。

因此，我们需要增加一个页面显示事件的定义，在页面显示时重新获取数据：

```
/**
 * 生命周期函数--监听页面显示
 */
onShow: function() {
  //页面显示，需要刷新数据
  this.setUserData()
  this.initCart()
},
```

在完成显示数据获取的 JS 逻辑后，编写 WXML 页面模板代码实现页面数据显示。

用户头像、用户昵称、用户等级图标的显示可基于第五章第四节 “2.2 实现子部件 1 ：用户的当前用户等级与成长值显示” 的代码进行修改实现。

当前可用积分、笔记总获赞数、当前总成长值的水平排列效果使用 WeUI 的 Flex 组件实现，图标可以从 ColorUI 中找到，页面跳转用 Navigator。

WXML 页面模板代码如下：

```
<!-- 子部件 1 ：用户信息栏 设置白色背景-->
<view class="bg-white padding-top-lg">
 <view class="text-center padding-bottom-sm">
  <!-- 用户头像 使用open-data显示用户头像 -->
  <view class="cu-avatar xl round bg-gray">
   <open-data class="userAvatarUrl" type="userAvatarUrl"></open-data>
  </view>
  <!-- 用户昵称 使用open-data显示用户头像 -->
  <view>
   <open-data type="userNickName"></open-data>
  </view>
  <!-- 用户的等级图标 -->
  <view class="cu-avatar sm round bg-white line-{{ myLevel.icon }} solids">
   <text class="icon-medal"> </text>
  </view>
 </view>
 <!-- 使用 WeUI 的 Flex 实现水平排列 -->
 <view class="weui-flex">
  <!-- 当前可用积分 -->
  <view class="weui-flex__item text-center padding-tb-sm">
   <navigator url="../point/point">
    <view class="text-xxl text-black">{{myInfo.point}}</view>
    <view><text class="icon-choiceness"></text> 积分</view>
   </navigator>
  </view>
  <!-- 笔记总获赞数 -->
  <view class="weui-flex__item text-center padding-tb-sm">
   <view class="text-xxl text-black">{{myBeUpvotedNum}}</view>
   <view><text class="icon-like"></text> 获赞</view>
  </view>
  <!-- 当前总成长值 -->
  <view class="weui-flex__item text-center padding-tb-sm">
   <navigator url="../level/level">
    <view class="text-xxl text-black">{{myInfo.growthValue}}</view>
    <view><text class="icon-medal"></text> 成长</view>
   </navigator>
  </view>
 </view>
</view>
```

2.3 实现子部件 2 ：小 M 卡会员入口栏

子部件 2 包含一个显示元素：

小 M 卡会员菜单

交互界面，事件为用户点击事件，数据为用户表中用户付费会员有效期字段

如果用户不是小 M 卡会员，显示非小 M 卡会员的内容提示：成为小 M 卡会员立享 7 折特权

如果用户是小 M 卡会员，显示小 M 卡会员的内容提示：我的小 M 卡

用户是否小 M 卡会员根据付费会员有效期是否大于当前时间判断

用户点击事件的事件响应为：页面跳转到付费会员首页（在第七章第三节开发实现）

小 M 卡会员菜单的实现与第八章第六节 “3.4 实现子部件 3 ：付费会员促销栏” 类似：

```
<!-- 子部件 2 ：小M卡会员入口栏 使用 WeUI 的 panel -->
<view class="weui-panel weui-panel_access">
 <view class="weui-panel__bd">
  <navigator url="../membershipcard/membershipcard">
   <view class="weui-cell weui-cell_access weui-cell_hide_line">
    <view class="weui-cell__bd">
     <view class="weui-badge weui-badge_space">M</view>
     <view class="weui-badge_after">
      {{ isMembershipExpired ? '成为小M卡会员立享7折特权' : '我的小M卡'}}
     </view>
    </view>
    <view class="weui-cell__ft weui-cell__ft_in-access">{{ isMembershipExpired ? '开通会员' : ''}}</view>
   </view>
  </navigator>
 </view>
</view>
```

2.4 实现子部件 3 ：常用功能菜单栏

在本章第二节已经将子部件 3 拆解为一系列的显示元素：

订单列表入口

静态界面，事件为用户点击事件，无数据

用户点击事件的事件响应为：页面跳转到订单列表页面（在第八章第八节开发实现）

购物车入口

交互界面，事件为用户点击事件，数据为微信客户端的购物车缓存

如果微信客户端的购物车缓存中有商品，则在该入口的右上角用红色标记显示购物车缓存中的商品数量

用户点击事件的事件响应为：页面跳转到购物车页面（在第八章第七节开发实现）

微信运动步数同步入口

静态界面，事件为用户点击事件，无数据

用户点击事件的事件响应为：页面跳转到微信运动同步页面（在第六章第三节开发实现）

子部件 3 的水平排列效果使用 WeUI 的 Flex 组件实现，图标可以从 ColorUI 中找到，页面跳转用 Navigator。

购物车右上角红色标记显示商品数量，可以以第八章第六节底部操作栏的样式为基础，添加自定义样式 cart_badge 实现。

WXML 页面模板代码如下：

```
<!-- 子部件 3：常用功能菜单栏 使用 WeUI 的 panel -->
<view class="weui-panel weui-panel_access">
 <view class="weui-panel__bd">
  <!-- 使用 WeUI 的 Flex 实现水平排列 -->
  <view class="weui-flex">
   <!-- 订单列表入口 -->
   <view class="weui-flex__item text-center padding-tb-sm">
    <navigator url="../order/order">
     <view class="text-xxl">
      <text class="icon-shop"></text> </view>
     <view>订单</view>
    </navigator>
   </view>
   <!-- 购物车入口 -->
   <view class="weui-flex__item text-center padding-tb-sm">
    <navigator url="../../shop/cart/cart">
     <view class="text-xxl text-center">
      <!-- 右上角红点以第八章第六节底部操作栏的样式为基础，添加自定义样式cart_badge实现 -->
      <view class="icon-cart cart_badge">
       <view wx:if="{{cart.length > 0}}" class="cu-tag badge">{{cart.length}}</view>
      </view>
     </view>
     <view>购物车</view>
    </navigator>
   </view>
   <!-- 微信运动步数同步入口 -->
   <view class="weui-flex__item text-center padding-tb-sm">
    <navigator url="../walkingstepsync/walkingstepsync">
     <view class="text-xxl">
      <text class="icon-rank"></text> </view>
     <view>步数同步</view>
    </navigator>
   </view>
  </view>
 </view>
</view>
```

2.5 子部件 4：笔记栏

子部件 4 就是去掉了发现菜单的 UGC 社区首页，具体实现代码请参考第九章第五节。

由于篇幅所限，包含完整样式的 WXML 页面模板代码请查阅专栏源代码 index.wxml 与 index.wxss，完整的 JS 逻辑代码见 index.js，具体代码位置见本节末尾图 3。

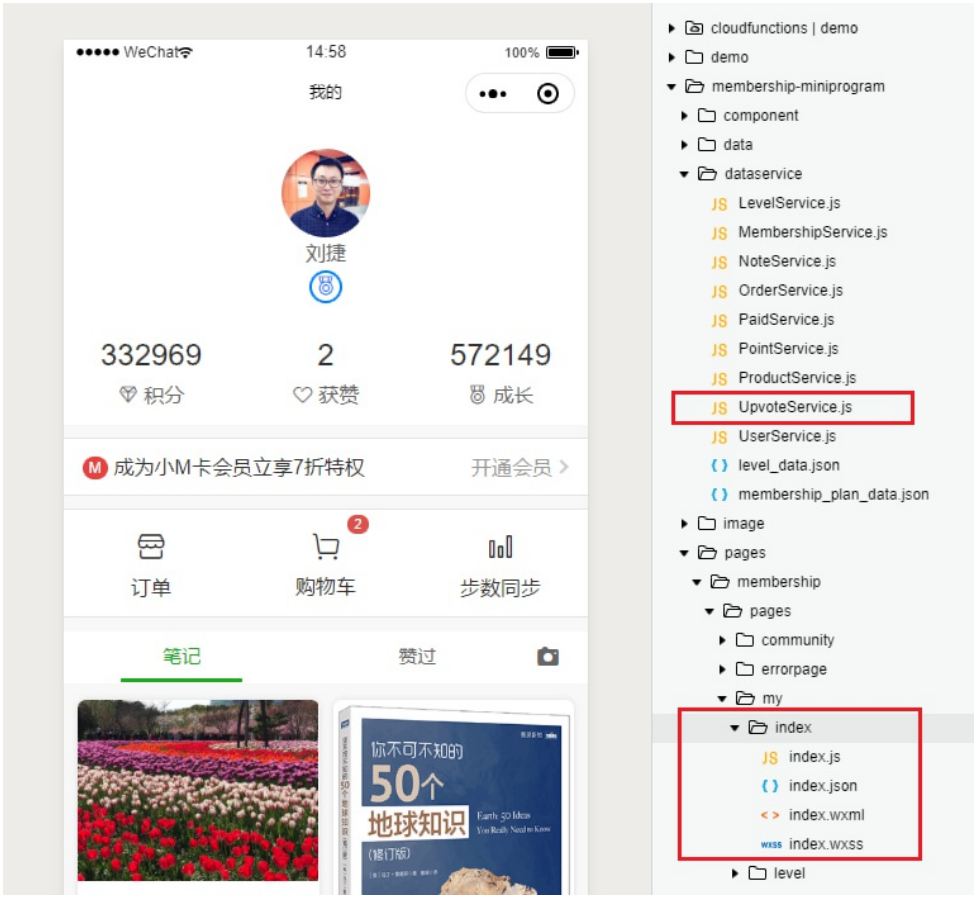
3. 专栏源代码

本专栏源代码已上传到 GitHub，请访问以下地址获取：

https://github.com/liujiec/Membership-ECommerce-Miniprogram

本节源代码内容在图 3 红框标出的位置。

图 3 本节源代码位置



下节预告

下一节，我们将让项目具备为运营服务的能力（发送模板消息）和为营销服务的能力（自定义转发内容）。

实践环节

实践是通往大神之路的唯一捷径。

本节实操内容：

- 编写代码完成个人中心页面，如碰到问题，请阅读本专栏源代码学习如何实现。

}

43 个人中心功能设计

45 无小程序不营销：自定义转发与模板消息