33 让用户掏钱：开发实现商城商品详情页面

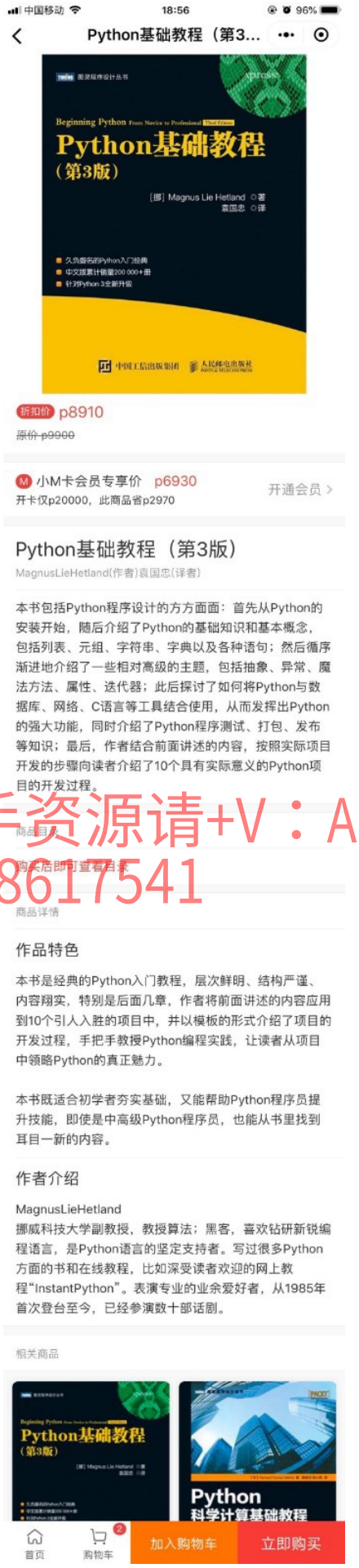
更新时间：2019-09-04 13:57:17

> 为中华之崛起而读书。
>
> ——周恩来

我们在上一节实现了商城支付接口，本节将实现商品详情的展示，并调用支付接口完成商品购买流程。

在第二节中，我们已经设计了商品详情的页面效果，完整的商品详情页面如图 14 所示。

图 14 商品详情页面



接下来，我们根据业务设计、功能设计和页面效果图，按照“分类拆解法”的步骤实现商品详情页面。

1. 拆解步骤

在本节仅给出拆解结果，拆解过程请回顾第二章第三节对“分类拆解法”的详细讲解内容。

请各位同学自己动手实践，按照拆解步骤拆解，结合第一节的业务设计、第二节的功能设计，拆解本节图 14 的商品详情页面，然后将自己的拆解结果与本节列出的拆解结果进行对照总结。这是本节内容的实践环节之一。

商品详情页面可以由上到下拆解为 8 个子部件，具体拆解结果如下：

子部件 1 ：商品大图栏

子部件 1 的显示元素包括：

商品大图

单条内容交互界面、无事件、数据为商品信息的封面大图字段

商品大图水平居中显示

商品信息在页面加载时，通过其他页面跳转到商品详情页面时传递的商品 ID 从云数据库的商品信息表中获取

子部件 2 ：商品价格栏

子部件 2 的数据包括商品信息（价格字段）、用户信息（当前总成长值字段、付费会员到期时间字段）、用户等级与等级特权表

子部件 2 的显示元素包括：

非付费会员商品折扣价

单条内容交互界面，无事件

如果用户不是付费会员（付费会员到期时间小于当前时间），且用户等级有购物折扣（等级 1 用户没有购物折扣特权）时，显示非付费会员商品折扣价，折扣价 = 商品原价 * 用户当前等级享受的购物折扣率

非付费会员商品折扣价同时显示折扣价与原价，分两行显示：

第一行，折扣价放大红字显示，左边有红底白字的“限时特价”标记

第二行，原价灰色删除线显示

付费会员商品折扣价

单条内容交互界面，无事件

如果用户是付费会员（付费会员到期时间大于等于当前时间），显示付费会员商品折扣价，折扣价 = 商品原价 * 付费会员享受的购物折扣率

付费会员商品折扣价同时显示折扣价与原价，分两行显示：

第一行，折扣价放大红字显示，左边有红底白字的“会员专享价”标记

第二行，原价灰色删除线显示

商品原价

单条内容交互界面，无事件

如果用户不是付费会员（付费会员到期时间小于当前时间），且用户等级没有购物折扣 （等级 1 用户没有购物折扣特权）时，仅显示商品原价

商品原价左边有红底白字的 “限时特价” 标记，商品原价放大红字显示

子部件 3 ：付费会员促销栏

子部件 3 的显示元素包括：

小 M 卡会员专享价

单条内容交互界面，事件为用户点击事件，数据为商品原价、付费会员商品折扣价

小 M 卡会员专享价仅在用户不是付费会员（付费会员到期时间小于当前时间）时显示

小 M 卡会员专享价的显示内容包括：

左侧第一行，放大红字显示付费会员商品折扣价

左侧第二行，推广文案 “开卡仅 p20000，此商品省 pXXXX” （XXXX 为商品原价 - 付费会员商品折扣价）

右侧 “开通会员” 引导标记

用户点击事件的事件响应为，页面跳转到付费会员首页面

子部件 4 ：商品简介栏

子部件 4 的显示元素包括：

商品名称

单条内容交互界面，无事件，数据为商品信息的书名与作者字段

书名第一行黑色大字体显示，作者第二行灰色小字体显示

商品描述

单条内容交互界面，无事件，数据为商品信息的本书简介字段

子部件 5 ：商品目录栏

子部件 5 的显示元素包括：

标题

静态界面，无事件，无数据

提示

静态界面，无事件，无数据

红色字体显示提示 “购买后即可查看目录”

子部件 6 ：商品详情栏

子部件 6 的显示元素包括：

标题

静态界面，无事件，无数据

作品特色

单条内容交互界面，无事件，数据为商品信息的本书特色字段

第一行黑色大字体显示 “作品特色”，第二行显示本书特色字段内容

作者介绍

单条内容交互界面，无事件，数据为商品信息的作者介绍字段

第一行黑色大字体显示 “作者介绍”，第二行显示作者介绍字段内容

子部件 7 ：相关商品栏

子部件 7 的显示元素包括：

商品列表

多条数据的交互界面，事件为用户屏幕滑动事件与用户点击屏幕事件，数据为：与当前商品属于同一分类的商品信息数组

在页面加载时，从商品信息表中查找商品分类与当前商品相同的商品信息列表

用户下滑屏幕到页面底部事件的事件响应为：从商品信息表中分页获取相同分类的下一页商品信息数据，并追加到商品信息数组末尾。如果商品信息表中相同分类的所有数据都获取完毕，用户下滑屏幕将不再获取分页数据

商品信息数组中的商品数据以左右两列卡片式瀑布流的方式显示，一个卡片显示一个商品信息，每个卡片显示商品的缩略图、商品名称（黑色）、商品简介（灰色）、商品原价（红色）

用户点击屏幕事件的事件响应为：在用户点击商品卡片后，页面跳转到商品详情页面，需要向商品详情页面传递商品 ID

商品信息表的数据需要从云数据库中获取

子部件 8 ：底部操作栏

底部操作栏固定在页面底部显示，不随屏幕滑动变化

子部件 8 的显示元素包括：

首页按钮

静态界面，事件为用户点击事件，无数据

按钮包含图标和文字

用户点击事件的事件响应为：页面跳转到商城首页

购物车按钮

单条内容交互界面，事件为用户点击事件，数据为微信客户端的购物车缓存

按钮包含图标和文字，如果微信客户端的购物车缓存中有商品，则在该按钮上显示缓存中的商品数量，数字在图标的右上角用红色标记显示

用户点击事件的事件响应为：页面跳转到购物车页面

加入购物车按钮

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：将当前商品信息加入微信客户端的购物车缓存中，提示用户加入购物车成功，并更新购物车按钮右上角的数字

购买商品按钮

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：调用服务端的支付接口，并在页面中显示支付接口返回的支付结果

用户购买商品要支付的积分为子部件 2 中放大红字显示的价格

除上述的 8 个子部件外，与第七章第四节类似，我们还需要两个子部件用于显示支付结果：

- 子部件 9 ：显示支付成功结果
- 子部件 10 ：显示支付失败结果

此外，在专栏的源代码中还设计了一个购买后的环节：当用户未购买商品时，无法查看目录；当用户已购买商品后，在商品详情页面中不再显示商品价格栏、付费会员促销栏和底部菜单栏，而显示商品的详细目录。

因此，还需要一个子部件来显示购买后的商品详情：

- 子部件 11 ：显示购买后的商品详情

2. 数据服务

商品详情页面需要从商品信息表根据商品 ID 获取商品信息，需要从商品信息表获取与当前商品分类一致的商品信息列表，还需要从商品购买记录表中查询用户是否已购买商品。

根据分类获取商品列表的数据服务已在本章第三节中实现。

根据商品 ID 获取商品信息的数据服务需要在 ProductService.js 的 `ProductService` 类中添加一个方法 `getProductByIndex` 来实现：

```
/**
 * 根据商品 ID 从商品信息表获取商品信息
 * @method getProductByIndex
 * @for ProductService
 * @param {string}} index 商品 ID
 * @param {function} successCallback(product) 处理数据查询结果的回调函数
 * product示例数据：
 * {
 *   index: "2599",
 *   bookname: "Python深度学习",
 *   author: "[美]弗朗索瓦•肖莱...",
 *   price: "119.00",
 *   desc: "本书由Keras之父...",
 *   feature: ""本书在当前的...",
 *   authorinfo: "【作者简介】...",
 *   firstcategory: "计算机",
 *   secondcategory: "",
 *   thirdcategory: "",
 *   coverimg: "cloud://xxx",
 *   smallcoverimg: "cloud://xxx",
 *   catelog: [{
 *     name: "前言",
 *     index: "23171"
 *   },],
 *   smallcoverimgwidth: 231,
 *   smallcoverimgheight: 290,
 *   coverimgwidth: 700,
 *   coverimgheight: 879
 * }
 */
getProductByIndex(index, successCallback) {
  //执行数据库查询
  db.collection('product').where({
    index: index
  }).get()
  .then(res => {
    if (res.data.length > 0) {
      //回调函数处理数据查询结果
      typeof successCallback == "function" && successCallback(res.data[0])
    } else {
    //在商品信息表中没有查找到商品 ID 对应的商品信息时，不应该显示商品详情页面，而应该跳转出错页面
      wx.redirectTo({
        url: '../../errorpage/errorpage'
      })
    }
  })
  .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)
  })
}
```


当方法返回值是一个对象时，建议在方法返回值的注释中编写对象的示例，方便自己或其他人在调用该方法时能快速明白该方法的用法，如上述代码所示。

从商品购买记录表中查询用户是否已购买商品，需要新建一个数据服务 `PaidService` 提供 `getPaidInfo` 方法，该方法请各位同学自行编写代码实现，具体实现代码可参考专栏源代码的 `PaidService.js` 文件，具体代码位置见本节末尾图 15。

3. 编程步骤

在商品详情页面实现支付结果显示，需要在显示支付结果子部件时隐藏其他子部件，即：

- 显示子部件 1 - 8 时隐藏子部件 9 、子部件 10、子部件 11
- 显示子部件 9 时隐藏子部件 1 - 8 与子部件 10、子部件 11
- 显示子部件 10 时隐藏子部件 1 - 8 与子部件 9、子部件 11

因此，我们需要两个标志 `showPaySuccess` 和 `showPayFaild` 来控制显示哪些子部件。

同时还需要一个标志 `isPaid` 来控制用户打开已购买商品的商品详情页面，只显示子部件 11，隐藏其他子部件。

从数据库读取用户、商品信息等需要一定时间，还需要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。

子部件 3 ：付费会员促销栏仅在用户不是小 M 卡会员时显示，因此需要一个数据标志 `isMembershipExpired` 来标识用户是否小 M 卡会员。

```
data: {
  showPaySuccess: false, //是否显示支付成功结果
  showPayFaild: false, //是否显示支付失败结果
  isPaid: false, //用户是否已购买该商品
  inited: false, //是否已从数据库读取数据
  isMembershipExpired: true, //用户是否是小M卡会员（还在会员有效期中）
},
```

由于商品详情页需要读取的数据较多，且页面各子部件隐藏显示的逻辑较复杂，我们应该首先理清数据读取的顺序与读取内容。

数据内容包括：


```
data: {
  myInfo: {}, //用户信息，在支付时需要根据当前可用积分判断用户是否有足够积分购买商品
  price: 0, //商品原价
  discount: 1, //用户会员等级或小M卡会员对应的折扣率
  discountedPrice: 0, //用户实际支付的商品价格
  membershipPrice: 0, //用于子部件 3 ：付费会员促销栏显示的付费会员商品折扣价
  productInfo: {}, //商品信息
},
```

数据读取的顺序为：

无论用户是否购买商品，都需要显示商品的详细信息，因此首先读取的数据应该是商品信息：

```
/**
 * 生命周期函数--监听页面加载
 */
onLoad: function(options) {
  //从数据库读取数据，options.index为页面跳转时传递的商品 ID
  this.getProductByIndex(options.index)
},

/**
 * 从商品信息表查询商品 ID 对应的商品信息
 */
getProductByIndex(productIndex) {
  var that = this
  //调用数据服务获取商品信息
  productService.getProductByIndex(
    productIndex,
    //回调函数
    function(productInfo) {
      //设置商品数据
      that.setData({
        productInfo: productInfo,
      })
      //查询用户是否已购买商品
      that.getPaidInfo(productIndex)
    })
},
```

第二步需要读取的是用户是否已购买该商品的信息，如果用户已购买该商品则只显示子部件 11 ，因此不需要再计算商品价格：

```
/**
 * 从商品购买记录表中查询用户是否已购买商品
 */
getPaidInfo(productIndex) {
  var that = this
  //调用数据服务获取用户购买商品记录
  paidService.getPaidInfo(
    productIndex,
    //回调函数
    function(paidInfo) {
      if (paidInfo.length > 0) {
        //如果在商品购买记录表中查询到购买记录，显示已购买界面
        that.setData({
          isPaid: true //设置用户已购买商品标志
        })
      }
      else{
        //如果用户还未购买商品，则需要计算商品价格信息
        that.getProductPrice()
      }
    })
},
```


如果用户未购买商品，则需要读取用户信息计算用户享受的折扣率，并计算用户实际支付价格：

```
/**
 * 根据用户等级与是否付费会员计算商品价格信息
 */
getProductPrice() {
  var that = this
  //调用数据服务获取用户的用户等级与付费会员有效期
  levelService.getLevelList(
    //获取成长体系中的所有成长等级回调函数
    function(levelList) {
      var levels = levelList
      userService.getUserInfo(
        //获取用户信息回调函数
        function(userinfo) {
          var myInfo = userinfo
          //获取用户等级
          var myLevel = levels.filter(e => e.minGrowthValue <= myInfo.growthValue && myInfo.growthValue <= e.max
GrowthValue)[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()
          //设置用户是否是小M卡会员的标志
          that.setData({
            myInfo: myInfo,
            isMembershipExpired: isMembershipExpired,
          })
          if (!isMembershipExpired) {
            //如果用户是在有效期的小M卡会员，设置小M卡会员折扣
            that.setData({
              discount: MEMBERSHIPDISCOUNT
            })
          } else if (myLevel.bonus.length == 3) {
            //如果用户不是小M卡会员，设置用户当前等级对应的折扣
            that.setData({
              discount: myLevel.bonus[1].discount
            })
          }
          //设置商品价格数据
          that.setData({
            price: parseInt(that.data.productInfo.price),
            //根据折扣计算商品折扣价
            discountedPrice: Math.ceil(parseInt(that.data.productInfo.price) * that.data.discount),
            //用于子部件 3 ：付费会员促销栏显示的付费会员商品折扣价
            membershipPrice: Math.ceil(parseInt(that.data.productInfo.price) * MEMBERSHIPDISCOUNT)
          })
          //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,
          })
          //在用户未购买该商品时，需要显示子部件 7 ：相关商品栏的第一页分页数据
          if (!that.data.isPaid) {
            that.getProductList(true)
          }
        })
    })
},
```


请参照前面章节的内容自行添加数据服务引用。

3.1 定义页面子部件及其排列顺序

在 WXML 页面模板中我们需要定义在什么条件下显示哪些子部件：

```
<!-- 当从数据库读取到数据后，用户未购买该商品，并且不显示支付结果时，才显示子部件 1 - 8 -->
<view wx:if="{{inited && !showPaySuccess && !showPayFaild && !isPaid}}">
 <view class="bg-white">
  <!-- 子部件 1 ： 商品大图栏 -->
  <view class="text-center">
  </view>
  <!-- 子部件 2 ： 商品价格栏 -->
  <view class="padding-lr padding-bottom-sm">
  </view>
 </view>
 <!-- 子部件 3 ： 付费会员促销栏 当用户不是小M卡会员时才显示 -->
 <view wx:if="{{isMembershipExpired}}" class="weui-panel weui-panel_access">
 </view>
 <!-- 子部件 4 ： 商品简介栏 -->
 <view class="weui-panel">
 </view>
 <!-- 子部件 5 ： 商品目录栏 -->
 <view class="weui-panel">
 </view>
 <!-- 子部件 6 ： 商品详情栏 -->
 <view class="weui-panel">
 </view>
 <!-- 子部件 7 ： 相关商品栏 -->
 <view class="weui-panel">
 </view>
 <!-- 子部件 8 ： 底部操作栏 -->
 <view class="cu-bar bg-white tabbar border shop bottom_pay_button">
 </view>
</view>

<!-- 当支付接口返回支付成功时，显示支付成功结果 -->
<!-- 子部件 9 ： 显示支付成功结果 -->
<view wx:if="{{showPaySuccess}}" class="weui-msg bg-white padding">
</view>

<!-- 当支付接口返回支付失败时，显示支付失败结果 -->
<!-- 子部件 10 ： 显示支付失败结果 -->
<view wx:if="{{showPayFaild}}" class="weui-msg bg-white padding">
</view>

<!-- 如用户已购买该商品，则显示购买后界面 -->
<!-- 子部件 11 ： 显示购买后的商品详情 -->
<view wx:if="{{isPaid}}">
</view>
```


使用 WeUI 的 Panel 组件实现子部件之间由灰色色块区隔的效果，使用 WeUI 的 Msg 组件的 成功提示页与失败提示页可以快速的实现支付成功与支付失败结果的显示。

3.2 实现子部件 1 ： 商品大图栏

商品大图使用小程序官方媒体组件 image 显示，显示模式为 `mode="widthFix"` （缩放模式，宽度不变，高度自动变化，保持原图宽高比不变）：

```
<!-- 子部件 1 ： 商品大图栏 -->
<view class="text-center">
  <image mode="widthFix" src="{{ productInfo.coverimg ? productInfo.coverimg : '' }}"></image>
</view>
```

3.3 实现子部件 2 ： 商品价格栏

商品价格栏有三个显示元素，根据不同的条件显示不同的显示元素，这可以使用 `block` 标签配合条件渲染 `wx:if` 来实现；红底白字的 “限时特价” 等标记使用 WeUI 的 Badge 组件实现：

```
<!-- 子部件 2 ： 商品价格栏 -->
<view class="padding-lr padding-bottom-sm">
  <!-- 非付费会员商品折扣价 如果用户不是付费会员，且用户等级有购物折扣时显示 -->
  <block wx:if="{{isMembershipExpired && discount < 1}}">
    <view class="weui-badge weui-badge_space">折扣价</view>
    <view class="weui-badge_after text-xl text-red">
      p{{discountedPrice}}
    </view>
    <view>
      <text class="line_through text-sm"> 原价 p{{price}} </text>
    </view>
  </block>
  <!-- 付费会员商品折扣价 用户是小M卡会员时显示 -->
  <block wx:if="{{!isMembershipExpired}}">
    <view class="weui-badge weui-badge_space">M</view>
    <view class="weui-badge_after">
      小M卡会员专享价
      <text class="text-red"> p{{ discountedPrice }}</text>
    </view>
    <view>
      <text class="line_through text-sm"> 原价 p{{price}} </text>
    </view>
  </block>
  <!-- 商品原价 如果用户不是付费会员，且用户等级没有购物折扣时显示-->
  <block wx:if="{{isMembershipExpired && discount == 1}}">
    <view class="weui-badge weui-badge_space">限时特价</view>
    <view class="weui-badge_after text-xl text-red">
      p{{discountedPrice}}
    </view>
  </block>
</view>
</view>
```

3.4 实现子部件 3 ： 付费会员促销栏

付费会员促销栏的小 M 卡会员专享价的显示样式使用 WeUI 的 Panel 组件与 Badge 组件实现，页面跳转使用 Navigator 实现：

```
<!-- 子部件 3 ： 付费会员促销栏 当用户不是小M卡会员时才显示 -->
<view wx:if="{{isMembershipExpired}}" class="weui-panel weui-panel_access">
  <view class="weui-panel__bd">
    <navigator url="../../my/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">
            小M卡会员专享价
            <text class="text-lg text-red margin-left-sm">p{{ membershipPrice }}</text>
          </view>
          <view class="text-sm">
            {{ '开卡仅p20000，此商品省p'+ (productInfo.price - membershipPrice) }}
          </view>
        </view>
        <view class="weui-cell__ft weui-cell__ft_in-access">开通会员</view>
      </view>
    </navigator>
  </view>
</view>
```

3.5 实现子部件 4 - 子部件 6

子部件 4 ： 商品简介栏 、子部件 5 ： 商品目录栏与子部件 6 ： 商品详情栏的实现均较简单，只需使用 WeUI 的 Panel 组件显示商品信息的部分字段内容。

请各位同学自行编写代码实现，具体实现代码可参考专栏源代码的 `product.wxml` 文件，具体代码位置见本节末尾图 15。

3.6 实现子部件 7 ：相关商品栏

相关商品栏的代码实现与本章第三节商品列表栏几乎相同，不再复述。

相关商品栏的实现代码请参考：

- 本章第三节中商品列表栏的代码，不使用自定义组件，直接在页面中编写全部代码
- 专栏源代码，使用自定义组件实现

3.7 实现子部件 8 - 子部件 11

在子部件 8 ：底部操作栏中需要显示购物车中的商品数量，因此需要添加一个数据：

```
data: {
  cart: [], //购物车数据
},
```

并在页面加载时读取购物车数据，购物车数据存储在微信客户端的数据缓存中，使用小程序官方提供的数据缓存 API 读取（数据缓存 API 的使用方法请回顾本章第四节内容）：

```
/**
 * 生命周期函数--监听页面加载
 */
onLoad: function(options) {
  //读取购物车数据
  this.initCart()
},

  /**
 * 页面加载时读取用户本地缓存的购物车数据
 */
initCart: function() {
  var value = wx.getStorageSync('cart')
  if (value) {
    this.setData({
      cart: value
    });
  } else {
    this.setData({
      cart: []
    });
  }
},
```


在第七章第三节中我们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）。

在商品详情页同样存在该特殊场景：

当用户点击购物车按钮，在购物车中删除一些商品，再点击小程序左上角的返回按钮返回商品详情页面时，应该显示最新的购物车中的商品数量。

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购物车中的商品数量：

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

点击加入购物车按钮后，需要将当前页面的商品添加到微信客户端的购物车数据缓存中，并提示用户加入购物车成功：

```
/**
 * 添加商品到购物车
 */
addCart: function() {
  //从缓存读取购物车数据
  var value = wx.getStorageSync('cart');
  if (value) {
    //如果缓存中已经存在购物车数据
    if (value.filter(e => e.index == this.data.productInfo.index).length <= 0) {
//如果购物车数据中没有当前页面的商品，就将当前页面的商品添加到购物车数据的末尾
      value.unshift(this.data.productInfo);
    }
  } else {
    //如果缓存中不存在购物车数据，新建购物车数据并添加当前页面的商品
    value = [];
    value.push(this.data.productInfo);
  }
  //更新显示数据
  this.setData({
    cart: value
  });
  //将最新的购物车数据更新到本地缓存
  wx.setStorage({
    key: "cart",
    data: value,
  })
  //提示用户已加入购物车,2秒后隐藏提示
  wx.showToast({
    title: '已加入购物车',
    icon: 'success',
    duration: 2000
  });
},
```


底部操作栏的样式可以在 ColorUI 的操作条组件中找到，简单修改 ColorUI Demo 小程序的代码即可实现 WXML 页面模板：

```
<!-- 子部件 8： 底部操作栏 -->
<view class="cu-bar bg-white tabbar border shop bottom_pay_button">
  <navigator open-type="switchTab" url="../index/index">
    <view class="action">
      <view class="icon-home">
      </view>
      首页
    </view>
  </navigator>
  <navigator url="../cart/cart">
    <view class="action">
      <view class="icon-cart">
        <view wx:if="{{cart.length > 0}}" class="cu-tag badge">{{cart.length}}</view>
      </view>
      购物车
    </view>
  </navigator>
  <view bindtap='addCart' class="bg-orange submit">加入购物车</view>
  <view bindtap='onPayButtonClick' class="bg-red submit">立即购买</view>
</view>
```

底部操作栏固定在页面底部，可使用 `position` 与 `z-index` 样式属性实现：

```
/* 底部支付按钮样式 */
.bottom_pay_button {
  position: fixed;
  z-index: 9999;
  bottom: 0px;
  width: 100%;
}
```

子部件 8 中购买商品按钮点击事件的实现逻辑与实现代码，与第七章第五节付费会员支付按钮完全一致，区别仅在于调用的支付接口名称不同。

请各位同学参考第七章第五节的代码自行实现购买商品按钮的点击事件 `onPayButtonClick` ，也可参考专栏源代码的 `product.js` 文件，具体代码位置见本节末尾图 15。

子部件 9 与 子部件 10 的实现请参考第七章第五节 “2.7 实现子部件 6 与 子部件 7 ：显示支付结果” 或专栏源代码。

子部件 11 仅显示商品各字段信息，具体实现代码请阅读专栏源代码的 `product.wxml` 文件。

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 product.wxml 与 product.wxss ，完整的 JS 逻辑代码见 product.js ，具体代码位置见本节末尾图 15 。

4. 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 15 红框标出的位置。

图 15 本节源代码位置

下节预告

下一节，我们将实现购物车页面及购物车合并支付功能。

实践环节

实践是通往大神之路的唯一捷径。

本节实操内容：

- 请结合第二章第三节对“分类拆解法”的详细讲解内容，拆解本节图 14 的商品详情页面，然后将自己的拆解结果与本节列出的拆解结果进行对照总结。
- 编写代码完成商品详情页面，如碰到问题，请阅读本专栏源代码学习如何实现。

}

32 准备好收银机：开发实现商城支付接口

34 让用户买买买：开发实现商城购物车页面