39 发个朋友圈：开发实现发表笔记页面

更新时间：2019-09-20 18:30:54

> 勤学如春起之苗，不见其增，日有所长。
>
> ——陶潜

上一节我们完成了发表笔记接口，本节将完成发表笔记的页面开发，完整实现发表笔记功能。

在第二节中，我们已经使用“分类拆解法”的拆解步骤，将页面进行了详细拆解。本节我们将按照“分类拆解法”的编程步骤，一步一步完成如图 7 所示页面的开发。

图 7 发表笔记页面

1. 编程步骤

在上一节我们已经在 NoteService 数据服务中编写好了发表笔记接口 postNote，本节直接进行页面编写即可。

2.1 定义页面子部件及其排列顺序

在本章第二节我们已经完成了图 7 的页面拆解工作。页面可以拆解为 3 个子部件：

- 子部件 1 ：顶部操作按钮栏
- 子部件 2 ：文字输入栏
- 子部件 3 ：图片选择栏

首先在 WXML 页面模板中定义 3 个子部件的容器：

```
<!-- 设置页面功能区域为白色背景 -->
<view class="bg-white padding-lr-sm padding-top-sm">
 <!-- 使用 Form 组件进行表单提交 -->
 <form bindsubmit="postFormSubmit">
  <!-- 子部件 1 ：顶部操作按钮栏 -->
  <view class="weui-flex padding-bottom-lg">
  </view>
  <view class="margin-lr-sm">
   <!-- 子部件 2 ：文字输入栏 -->
   <view>
   </view>
   <!-- 子部件 3 ：图片选择栏 -->
   <view class="weui-uploader">
   </view>
  </view>
 </form>
</view>
```

发表笔记页面是一个表单提交页面，可以使用小程序官方的 Form 组件来提交表单，绑定提交事件 postFormSubmit 。

子部件 1 的两个按钮左右排列，可以用 WeUI 的 Flex 组件实现布局。

在 WeUI 中我们可以找到 Uploader 组件，基于该组件简单修改即可实现类似发微信朋友圈时选择图片的样式效果。

2.2 实现子部件 2：文字输入栏

在本章第二节已经将子部件 2 拆解为一系列的显示元素：

多行文本输入框

输入界面，事件为键盘输入事件，数据为用户输入内容和输入内容字数。

键盘输入事件的事件响应为：更新输入内容字数。

输入内容字数提示

单条内容交互界面，无事件，数据为输入内容字数。

输入内容字数需要在 data 中定义：

```
data: {
  inputLength: 0 //多行文本输入框中输入内容字数
},
```

多行文本输入框使用小程序官方组件 textarea 实现，在 Form 表单提交时获取 textarea 中的内容（name="content"），textarea 的官方文档位置为：“组件” -> “表单组件” -> “textarea”。

输入内容字数提示的界面可以使用 WeUI 的 Input 组件中 "文本域" 的样式来实现。

```
<!-- 子部件 2：文字输入栏 -->
<view>
  <textarea name="content" bindinput='bindTextareaInput' maxlength="1000" auto-focus="true" auto-height="true" placeholder='说说此刻的想法...' />
  <view class="weui-textarea-counter">{{ inputLength }}/1000</view>
</view>
```

多行文本输入框需要添加键盘输入事件 bindinput='bindTextareaInput'，在输入框内容变化时更新输入内容字数：

```
/**
 * 在输入框内容变化时更新输入内容字数
 */
bindTextareaInput: function(e) {
  this.setData({
    inputLength: e.detail.value.length
  });
},
```

2.3 实现子部件 3：图片选择栏

在本章第二节已经将子部件 3 拆解为一系列的显示元素：

图片选择

输入界面，事件为用户点击事件，数据为用户选择图片。

添加图片按钮的用户点击事件为：用户从手机相册中选择最多 9 张图片，或用户使用相机拍照。

已选择图片的用户点击事件为：全屏显示用户点击的图片。

图片最多可以选择 9 张。

已选图片数量提示

单条内容交互界面，无事件，数据为用户选择图片数。

子部件 3 可以直接移植微信官方的小程序示例中图片 API 示例页面的代码实现（示例中的样式是使用的 WeUI 的 Uploader 组件）。

微信官方的小程序示例源代码下载方法请回顾第三章第一节 “2.4 使用小程序组件开发页面”，图片 API 示例页面源代码在 `/miniprogram/page/API/pages/image` 目录中（页面在微信开发者工具的模拟器中依次点击底部 “接口” 菜单 -> “媒体” -> “图片” 查看）。

在 data 中定义数组 `imageList` 来存放用户已选择的图片：

```
data: {
  imageList: [], //用户已选择的图片
},
```

参照小程序示例源代码实现图 7 的界面：

```
<!-- 子部件 3 ： 图片选择栏 -->
<view class="weui-uploader">
    <view class="weui-uploader__hd">
        <view class="weui-uploader__title"></view>
    </view>
    <view class="weui-uploader__bd">
        <!-- 缩略图显示用户已选择的图片 -->
        <view class="weui-uploader__files">
            <block wx:for="{{imageList}}" wx:key="*this" wx:for-item="image">
                <view class="weui-uploader__file">
                    <image class="weui-uploader__img" src="{{image}}" data-src="{{image}}" bindtap="previewImage"></image>
                </view>
            </block>
        </view>
        <!-- 添加图片按钮 -->
        <view class="weui-uploader__input-box">
            <view class="weui-uploader__input" bindtap="chooseImage"></view>
        </view>
    </view>
    <!-- 显示当前已选择几张图片 -->
    <view class="weui-uploader__info text-right">{{imageList.length}}/9</view>
</view>
```

参照小程序示例源代码实现添加图片按钮的用户点击事件和已选择图片的用户点击事件：

```
/**
 * 添加图片按钮的用户点击事件
 */
chooseImage() {
  const that = this
  //使用图片 API 的方法 wx.chooseImage 来让用户选择图片
  wx.chooseImage({
    sourceType: ['camera', 'album'], //用户可以拍照、也可以从相册选择图片
    sizeType: ['compressed'], //用户选择的图片压缩存储
    count: 8, //最多选择9张图片
    success(res) {
      that.setData({
        imageList: res.tempFilePaths
      })
    }
  })
},

/**
 * 已选择图片的用户点击事件
 */
previewImage(e) {
  const current = e.target.dataset.src
  //使用图片 API 的方法 wx.previewImage 全屏显示用户点击的图片
  wx.previewImage({
    current,
    urls: this.data.imageList
  })
},
```

2.4 实现子部件 1 ：顶部操作按钮栏

在本章第二节已经将子部件 1 拆解为一系列的显示元素：

取消按钮

静态界面，事件为用户点击事件，无数据。

取消按钮在页面顶部左侧。

用户点击事件的事件响应为：返回上一页面。

发表按钮

交互界面，事件为用户点击事件，数据为用户输入内容字数和用户选择的图片数。

发表按钮在页面顶部右侧。

只有当用户输入文字内容（输入内容字数大于 0）或用户选择图片（用户选择的图片数大于 0）后，发表按钮才处于可点击状态。

用户点击事件的事件响应为：在笔记记录表保存子部件 2 与子部件 3 中用户输入的内容，在成长值获取记录表中记录用户发表笔记获得成长值，并在用户表中更新用户当前总成长值。

可使用 WeUI 的 Flex 组件实现两个按钮的左右排列。

取消按钮的点击事件是返回上一页，使用 navigator 组件即可。

发表按钮的可点击状态通过 `disabled` 属性来控制，使用 Form 来进行表单提交，必须设置发表按钮的 `form-type="submit"`。

```
<!-- 子部件 1 ：顶部操作按钮栏 -->
<view class="weui-flex__item text-left">
  <navigator open-type='navigateBack'>取消</navigator>
</view>
<view class="weui-flex__item text-right">
  <button form-type="submit" disabled='{{ imageList.length <=0 && inputLength <=0 }}' class="weui-btn mini-btn" type="primary" size="mini">发表</but
ton>
</view>
```

点击发表按钮后会触发 Form 的表单提交事件 `postFormSubmit` ，在事件中调用上一节编写完成的发表笔记接口实现发表笔记的完整功能。

由于上传图片到云存储和云函数执行都需要一定时间，在调用发表笔记接口期间最好类似 “发表中，请等待” 这样的提示语。

此外，回想我们发微信朋友圈的场景，在我们发完一条朋友圈后，页面自动回到了朋友圈首页。这样良好的用户体验在我们的小程序中也应该具备。

由以上分析，我们可以实现表单提交事件 `postFormSubmit` 的完整代码：

```
/**
 * 发表笔记
 */
postFormSubmit: function(e) {
  //在调用发表笔记接口前提示发表中
  wx.showLoading({
    title: '发表中',
  })
  //从 textarea 中获取用户输入的文字内容
  var content = e.detail.value.content
  //调用发表笔记接口
  noteService.postNote(
    content, //用户输入的文字内容
    this.data.imageList, //用户选择的图片
    //处理表笔记接口返回结果的回调函数
    function() {
      //在发表笔记接口调用结束后关闭发表中的页面提示
      wx.hideLoading();
      //类似微信发朋友圈的用户体验,发表笔记成功后:
      //首先提示用户笔记发表成功
      wx.showToast({
        title: '发表笔记成功',
        icon: 'success',
        duration: 1500,
      });
      //然后在1.5秒后自动返回笔记列表页面
      setTimeout(function() {
        wx.navigateBack({
          delta: 1,
        })
      }, 1500)
    }
  )
},
```

请参照前面章节的内容自行添加数据服务引用。

由于篇幅所限，完整的 **WXML** 页面模板代码请查阅专栏源代码 **postnote.wxml** ，完整的 **JS** 逻辑代码见 **postnote.js** ，具体代码位置见图 **8**。

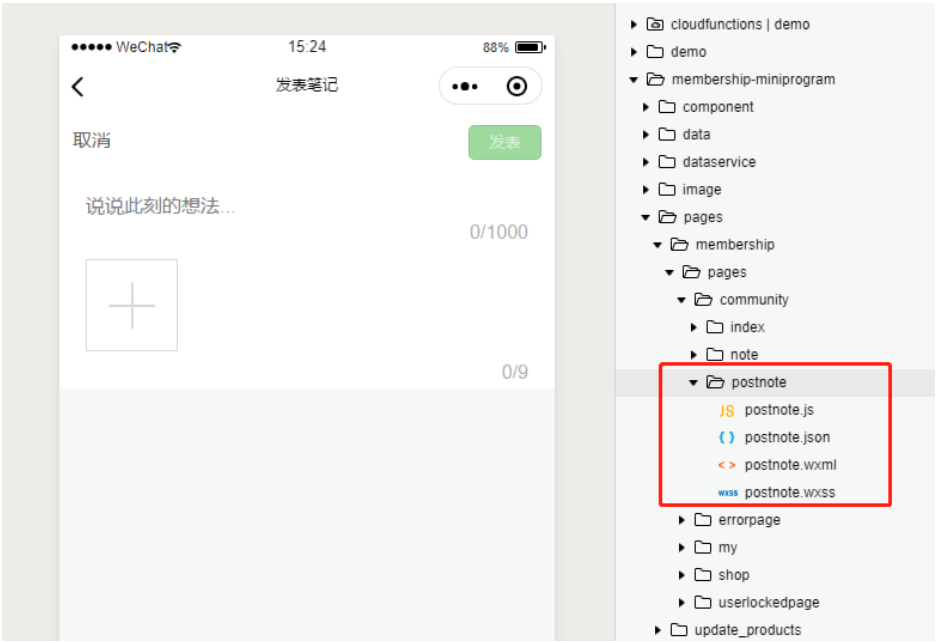
3. 专栏源代码

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 8 红框标出的位置。

图 8 本节源代码位置



下节预告

下一节，我们将开发实现 UGC 社区首页，显示用户发表的笔记。

实践环节

实践是通往大神之路的唯一捷径。

本节实操内容：

- 编写代码完成图 7 所示的页面，如碰到问题，请阅读本专栏源代码学习如何实现。

}

38 开发实现发表笔记接口

40 朋友圈刷起来：开发实现 UGC 社区首页面