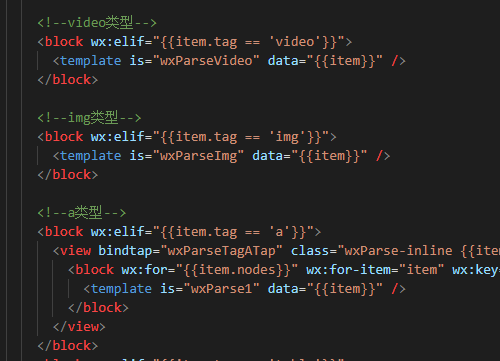
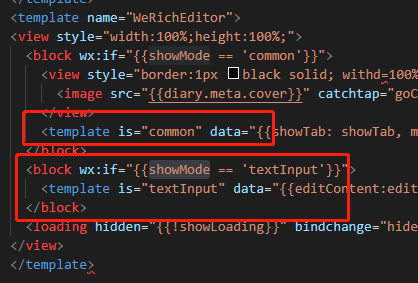
改造进度一20181228：

在网友Hollow的建议下，参考wxParse的结构改写。从分析结构，wxParse有多个JS， 一个wxml，一个wxss在wxml中，显示不同的元素，调用不同的template。

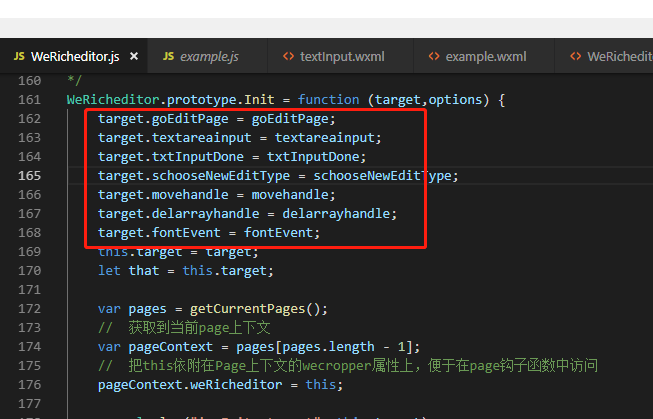


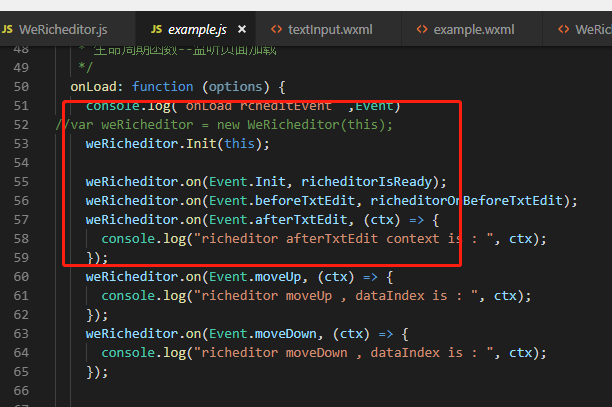
运用一致的方法, 放到本次改造的多页面template，通过 showMode 变量改变Show出的页面。举例，进入文本编辑模式，就改变showMode=textInput ， 那么就是显示对应的画面了。与原来的跳转文字编辑画面相比，没有了左上角的返回按钮。 应该说，使用template的方式，改变画面，都没有返回按钮。



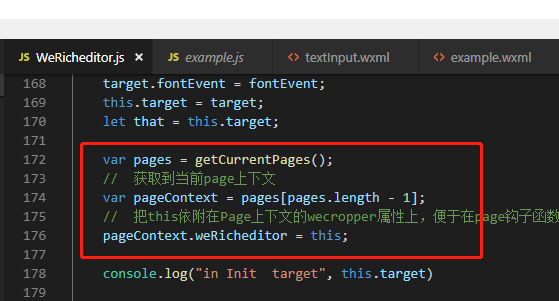
原来写在页面的Function，就需要统一在WeRicheditor.js 暴露出来。因为Function只能访问page中定义的Function，但是我们的function都写WeRicheditor.js。如果不做如下处理，就会报找不到函数。

传入的参数 target 是指在对应页面的This。因为WeRicheditor 是要有this.SetData 的方法将改变的数据会写到Page.Data中。

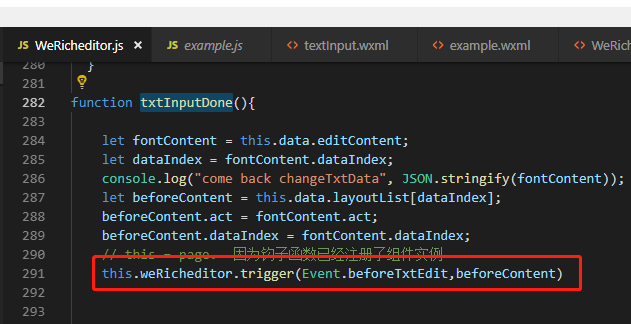




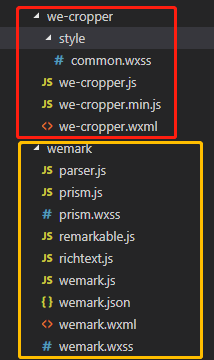
相反地，需要实现触发WeRicheditor的事件，我们需要写在Page 层面函数(txtInputDone)能访由example.wxml页面创建的weRicheditor对象。



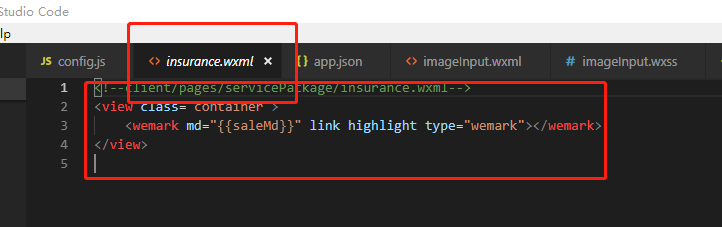
这样，卸载暴露在page层面的方法，通过访问this.weRicheditor，来访问到page new 的weRicheditor实例了。

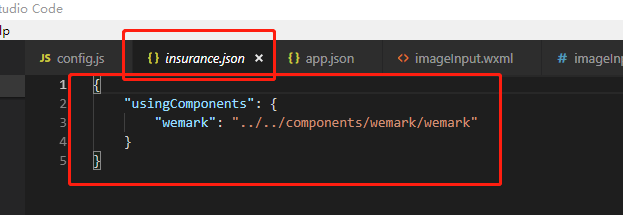


項目引用过的组件分析：



WeMark :

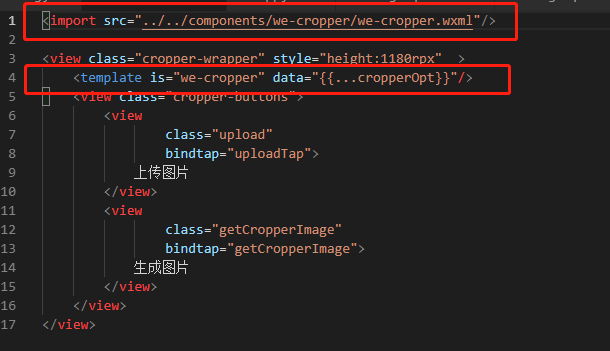


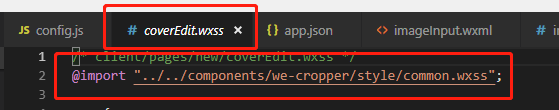


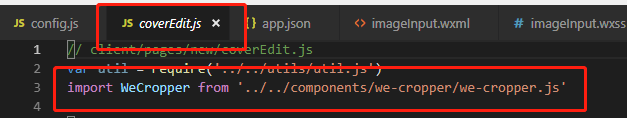
那么， 引用的页面，就可以识别{{saleMd}}变量内的MD，转换成小程序页面

we-cropper

作为template， 引用了对应的WXML和WXSS，就可以显示图片修剪的页面。



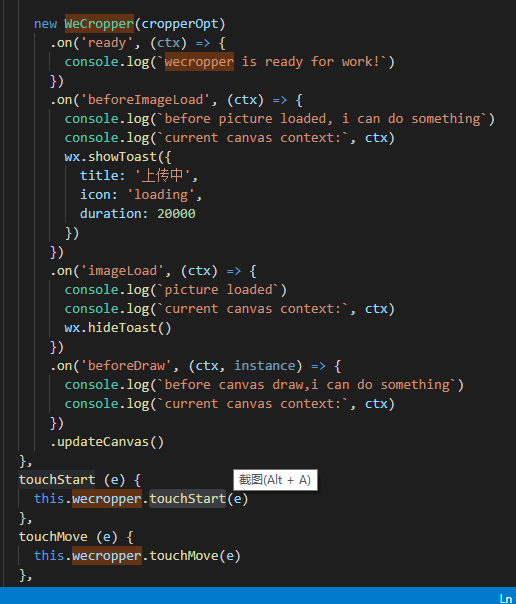
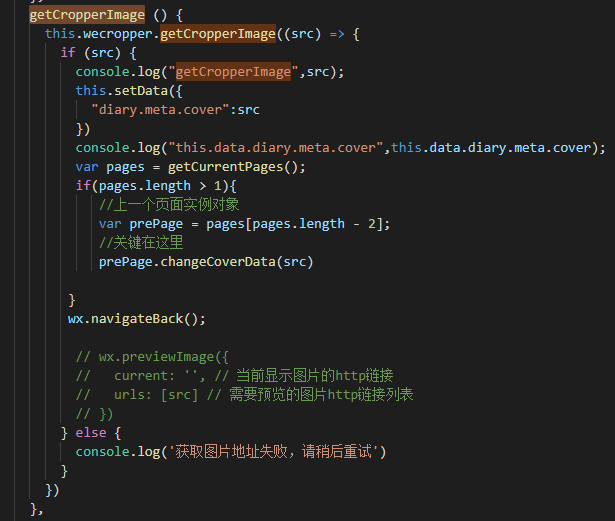




we-cropper分析：

作为补充小程序富文本编辑的开源组件，模仿we-cropper更适合。

**在调用we-cropper涉及了we-cropper的New，事件传递，获取组件内部数据等核心方法。**

补充小程序富文本编辑的开源组件，暂时命名为 We-RichEditor.

演变成修改需求：

1. We-RichEditor.的样式文件，图片文件，JS等，都要在一个文件夹中
2. We-RichEditor 提供的方式是template方式
3. We-RichEditor 提供getRichEditorImage 方法提供图片列表，由调用者上次到自己的图片服务器
4. We-RichEditor提供getRichEditorLayout 方法提供富文本layout
5. We-RichEditor提供template方式提供富文本预览功能（We-RichEditorPreview）

目前版本是0.0.1，属于Init 版本，但是也是能使用。如由问题，请加入MiniWechat技术讨论群 WeChatID： luckydcx

新的目录层次

Example

\components

\We-RichEditor

\image

\style

example.wxml

example.js

组件的全部代码，样式 ，都保存在We-RichEditor下

example.wxml 是调用例子