

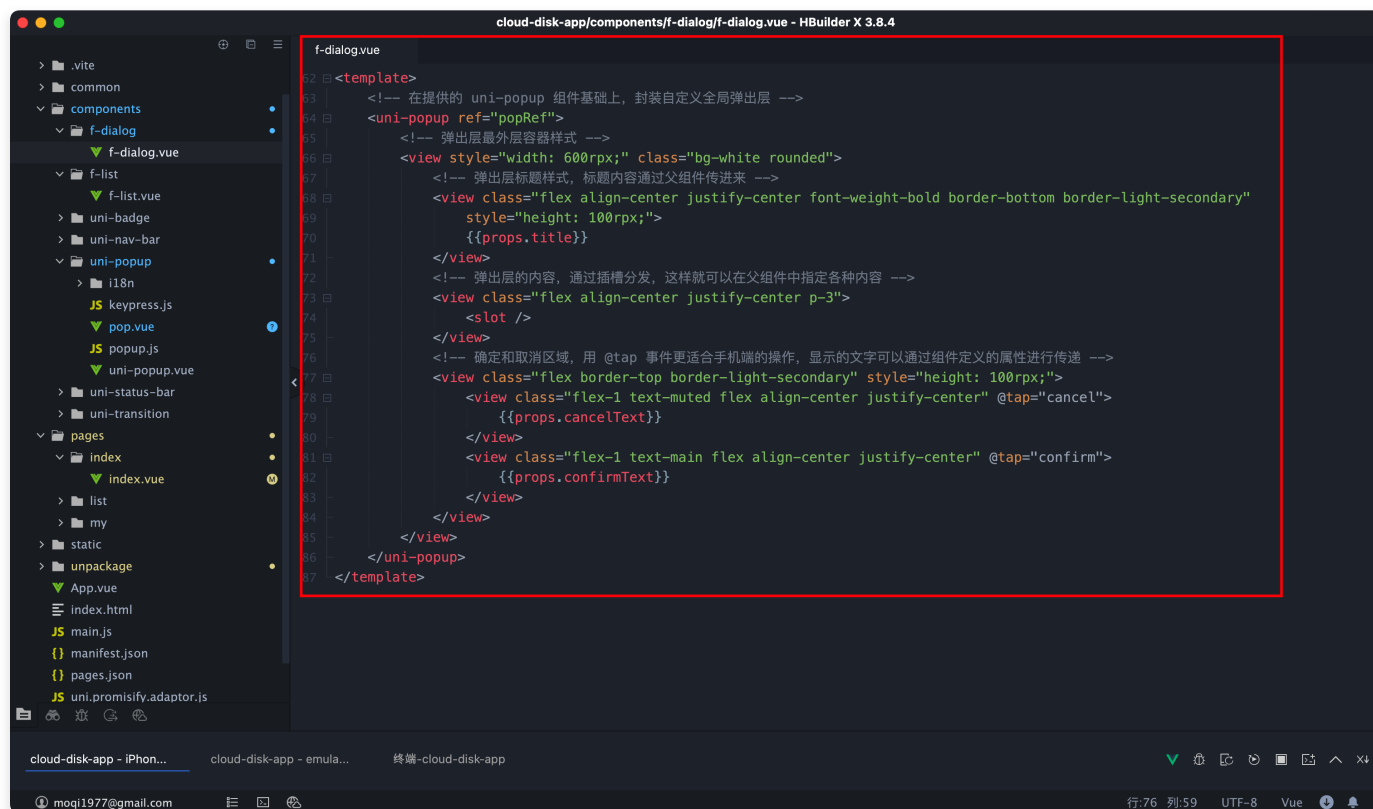
7. 重命名和删除功能

删除功能实现

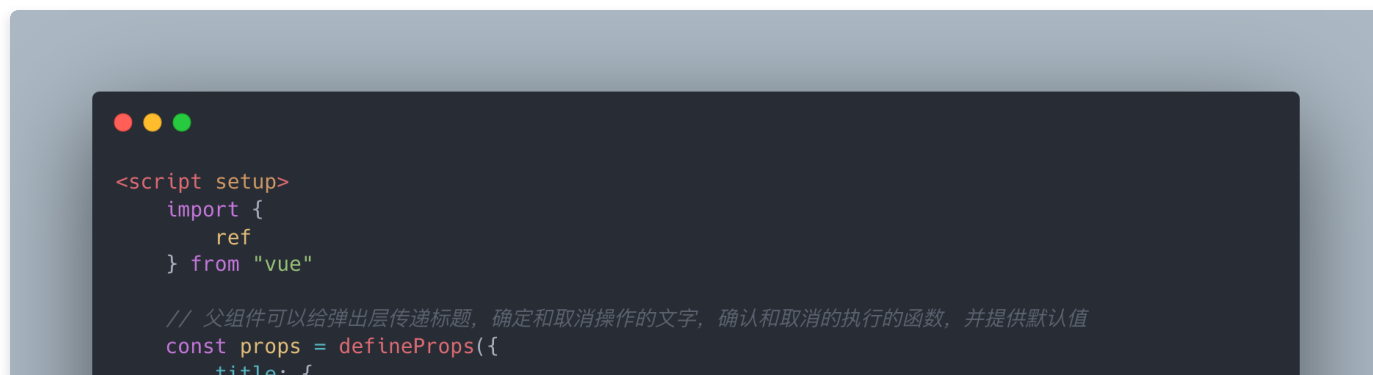
重命名功能实现

删除功能实现

我们先来封装自己的全局弹出层组件 `f-dialog.vue`，按照 `easycom` 规范建立组件文件，编写模版部分代码



然后编写其脚本部分代码，主要做法详见代码注释，如下：



```

        type: String,
        default: '提示'
      },
      cancelText: {
        type: String,
        default: '取消'
      },
      confirmText: {
        type: String,
        default: '确定'
      },
      onConfirm: {
        type: Function,
        default: null
      },
      onCancel: {
        type: Function,
        default: null
      }
    })

    // 根据 template 中的 ref 获取弹出层元素
    const popRef = ref(null)

    // 打开弹出层
    const showPopup = () => {
      popRef.value.open()
    };

    // 关闭弹出层
    const hidePopup = () => {
      popRef.value.close()
    };

    // 点击确认执行的函数
    const confirm = () => {
      // 这里的判断起到容错作用, 因为如果 props.onConfirm 不是函数, 就不能执行了
      if (typeof props.onConfirm === 'function') {
        // 执行属性中定义的, 由父组件传入的确认操作函数
        props.onConfirm();
      }
      // 关闭弹出层
      hidePopup();
    };

    const cancel = () => {
      if (typeof props.onCancel === 'function') {
        // 执行属性中定义的, 由父组件传入的取消操作函数
        props.onCancel();
      }
      hidePopup();
    };

    // 向父组件暴露弹出层的打开和关闭方法, 省去了 emit
    defineExpose({
      showPopup,
      hidePopup
    })
  }
</script>

```

在 index 页面需要执行删除功能的时候, 我们希望弹出对话框进行确认

- 先在 index 页面的 template 部分使用 f-dialog 组件，因为后面还有重命名等其他对话框，所以用 ref 命名区分清楚，并给删除对话框的确认和取消操作指定对应的处理函数。

```
139 <template>
140 <view>
141   <!-- 选中个数为0 -->
142   <uni-nav-bar v-if="checkedList.length === 0"> ...
143
144   <!-- 选中个数不为0 -->
145   <uni-nav-bar v-else> ...
146
147   <!-- 搜索框 -->
148   <view class="px-3 py-2"> ...
149
150   <!-- 自定义列表组件 -->
151   <f-list v-for="(item, index) in list" :key="index" :item="item" :index="index"
152     @my-select="handleSelect(index)"></f-list>
153
154   <!-- 底部操作条 -->
155   <!-- 选中元素个数大于0，才会出现操作条 -->
156   <view v-if="checkedList.length > 0"> ...
157
158   <!-- 删除对话框 -->
159   <f-dialog ref="deleteDialogRef" :onConfirm="handleDeleteConfirm" :onCancel="handleCancel">是否删除选中的文件? </f-dialog>
160
161 </view>
162 </template>
```

- 为底部操作条的每个 item 绑定处理事件，添加红框标注代码

```
<!-- 底部操作条 -->
<!-- 选中元素个数大于0，才会出现操作条 -->
<view v-if="checkedList.length > 0">
  <!-- 设置操作条容器样式：高度、颜色，固定在底部，flex布局，垂直方向为拉升方式 -->
  <view style="height: 115rpx;" class="flex align-stretch bg-main text-white fixed-bottom">
    <!-- 根据操作菜单元素个数等分容器，要么四等分，要么二等分，行高的修改可以让图标和文字之间的距离变得合理，点击变色通过 :hover-class 实现 -->
    <view class="flex-1 flex flex-column align-center justify-center" style="line-height: 1.5;"
      v-for="(item, index) in actions" :key="index" hover-class="bg-hover-primary"
      @click="handleBottomEvent(item)">
      <text class="iconfont" :class="item.icon"></text>
      {{ item.name }}
    </view>
  </view>
</view>
```

- 在 script 部分添加具体的处理事件 handleBottomEvent，根据 item 进行判断，执行不同的操作

* index.vue

```
57 +   const actions = computed(() => { ...  
88  
89   // 获得删除对话框元素  
90   const deleteDialogRef = ref(null)  
91  
92   // 底部操作条处理（根据传入的 item 进行判断，执行不同的操作）  
93   const handleBottomEvent = (item) => {  
94     switch (item.name) {  
95       case '删除':  
96         deleteDialogRef.value.showPopup()  
97         break;  
98       default:  
99         break;  
100     }  
101   }  
102  
103   const handleDeleteConfirm = () => {  
104     //对 list 进行过滤，留下未被选中的元素（选中的即被删除）  
105     list.value = list.value.filter(item => !item.checked);  
106     uni.showToast({  
107       title: '删除成功',  
108       icon: 'success'  
109     });  
110   }  
111  
112   const handleCancel = () => {  
113     console.log('取消');  
114     // 执行取消后的逻辑  
115   }  
116 </script>  
117
```

效果



```
commit 9f1f5fdb40d0a99aeac58d202c703b31c4287972 (HEAD -> main)
Author: mxqu <moqi1977@gmail.com>
Date: Sun Jul 2 16:58:41 2023 +0800
```

弹出层组件封装 + 删除功能

重命名功能实现

先在页面中增加一个重命名对话框组件，通过 ref 的值来和删除对话框进行区分，并且为它中间的 slot 插槽（输入框）使用 v-model 绑定重命名的值，注意红框部分代码：

```
index.vue
60 * <uni-nav-bar v-else> [...]
69
70 <!-- 搜索框 -->
71 * <view class="px-3 py-2"> [...]
84
85 <!-- 自定义列表组件 -->
86 <f-list v-for="(item, index) in list" :key="index" :item="item" :index="index"
87 @my-select="handleSelect(index)"></f-list>
88
89 <!-- 底部操作条 -->
90 <!-- 选中元素个数大于0，才会出现操作条 -->
91 * <view v-if="checkedList.length > 0"> [...]
103
104 <!-- 删除对话框 -->
105 <f-dialog ref="deleteDialogRef" :onConfirm="handleDeleteConfirm" :onCancel="handleCancel">是否删除选中的文件? </f-dialog>
106
107
108 <!-- 重命名对话框 -->
109 <f-dialog ref="renameDialogRef" :onConfirm="handleRenameConfirm" :onCancel="handleCancel">
110 <input type="text" v-model="renameValue" class="flex-1 bg-light rounded px-2" style="height: 95rpx;"
111 placeholder="重命名">
112 </f-dialog>
113
114 </view>
115 </template>
```

然后到底部操作条事件中增加“重命名”的 case 分支

```
index.vue x
57 * const actions = computed(() => { [...]
88
89 // 获得删除对话框元素
90 const deleteDialogRef = ref(null)
91
92 // 获得重命名对话框元素
93 const renameDialogRef = ref(null)
94
95 // 底部操作条处理（根据传入的 item 进行判断，执行不同的操作）
96 const handleBottomEvent = (item) => {
97   switch (item.name) {
98     case '删除':
99       deleteDialogRef.value.showPopup()
100       break;
101     case '重命名':
102       // 重命名只能对单个文件进行，checkedList数组中只有一个选中的元素，checkedList.value[0]
103       renameValue.value = checkedList.value[0].name
104       renameDialogRef.value.showPopup()
105       break;
106     default:
107       break;
108   }
109 }
110 }
```

定义重命名输入框的 v-model 变量，并对重命名的确认操作编写处理函数

```
index.vue x
107         break;
108     }
109 }
110
111 const handleDeleteConfirm = () => { ...
119
120 const handleCancel = () => { ...
124
125 // 重命名相关
126 const renameValue = ref('')
127
128 const handleRenameConfirm = () => {
129     if (renameValue.value === '') {
130         return uni.showToast({
131             title: '文件名不能为空',
132             icon: 'none'
133         })
134     }
135     // 更新该元素的 name 值
136     checkedList.value[0].name = renameValue.value
137     renameDialogRef.value.hidePopup()
138 }
139 </script>
140
```

效果



```
commit 8dd3610e4da0d07b1d9d1b0db8ff400e314b28a2 (HEAD -> main)
Author: mqxu <moqi1977@gmail.com>
Date: Sun Jul 2 17:24:49 2023 +0800
```

重命名功能实现