5. 封装通用列表组件

通用列表组件开发（一）

先看一下首页的预期效果

通过对列表数据的类型、结构的分析，我们来封装列表组件。

在 components 目录创建 f-list 目录，然后新建 f-list.vue 组件。

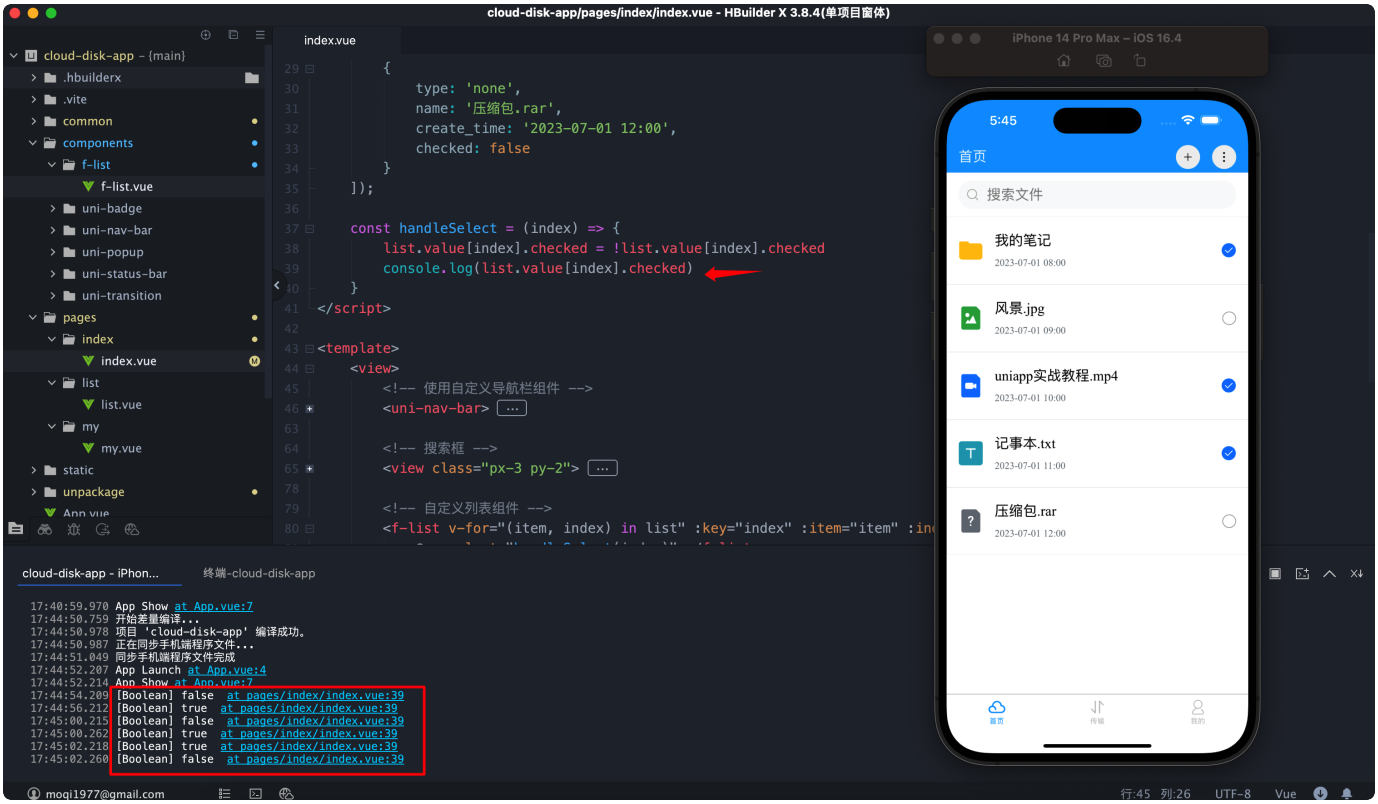
```vue
<script setup>
    import {
        computed,
        reactive
    } from "vue"

    const props = defineProps({
        item: Object,
        index: [Number, String],
        checked: Boolean
    })

    // 注意这里的icons是对象，不是数组哦
    const icons = reactive({
        dir: {
            icon: 'icon-file-b-2',
            color: 'text-warning'
        },
        image: {
            icon: 'icon-file-b-6',
            color: 'text-success'
        },
        video: {
            icon: 'icon-file-b-9',
            color: 'text-primary'
        },
        text: {
            icon: 'icon-file-s-7',
            color: 'text-info'
        },
        none: {
            icon: 'icon-file-b-8',
            color: 'text-muted'
        }
    });

    // 计算属性
    const iconClass = computed(() => {
        // 取得传入的 item 元素的 type 属性（dir、image、video等）
        let item = icons[props.item.type] //根据这个类型，到当前的 icons 对象取得相应的属性（又是一个对象），这里用的是类似数组下标的方式访问对象的属性
        // 取得 item 对象的 icon 属性和 color 属性，拼接成一个叠加样式返回，类似：'icon-file-b-9 text-primary'
        return `${item.icon} ${item.color}`;
    })

    // 定义需要向父组件抛出的事件
    const emits = defineEmits(['my-select'])

    // 向父组件抛出事件
    const onSelect = () => {
        emits('my-select')
    }
</script>

<template>
    <!-- 左中右三部分 -->
    <view class="p-3 flex align-center border bottom-border">
        <!-- 左侧：用计算属性得出动态样式，显示不同类型的文件图标 -->
        <text class="iconfont" :class="iconClass" style="font-size:60rpx"></text>

        <!-- 中间：渲染由父组件传入的对象中的名称和时间属性 -->
        <view class="flex flex-column ml-3" style="line-height: 1.2;">
            <text class="font-md">{{props.item.name}}</text>
            <text class="font-sm text-muted mt-2">{{props.item.create_time}}</text>
        </view>

        <!-- 右侧：根据传入的对象中的checked属性，进行条件渲染 -->
        <view class="ml-auto flex align-center justify-center" @click="onSelect">
            <!-- 未选中，画一个灰色的圆圈 -->
            <text v-if="!props.item.checked" style="width: 30rpx;height: 30rpx;border:1px solid #999;"
                class="rounded-circle"></text>
            <!-- 选中 ，用字体图标 -->
            <text v-else class="iconfont icon-xuanze-yixuan text-primary" style="font-size: 20px;"></text>
        </view>
    </view>
</template>
```

通用列表组件开发（二）

在 index页面 中定义一个数组，然后使用封装好的 f-list 组件，传入数组，并接收组件中抛出的事件进行处理。

```vue
<script setup>
    import { ref } from "vue"
    const list = ref([{
            type: 'dir',
            name: '我的笔记',
            create_time: '2023-07-01 08:00',
            checked: true
        },
        {
            type: 'image',
            name: '风景.jpg',
            create_time: '2023-07-01 09:00',
            checked: true
        },
        {
            type: 'video',
            name: 'uniapp实战教程.mp4',
            create_time: '2023-07-01 10:00',
            checked: true
        },
        {
            type: 'text',
            name: '记事本.txt',
            create_time: '2023-07-01 11:00',
            checked: false
        },
        {
            type: 'none',
            name: '压缩包.rar',
            create_time: '2023-07-01 12:00',
            checked: false
        }
    ]);

    const handleSelect = (index) => {
        list.value[index].checked = !list.value[index].checked
    }
</script>

<template>
    <view>
        <!-- 使用自定义导航栏组件 -->
        <uni-nav-bar>
          ...
        </uni-nav-bar>

        <!-- 搜索框 -->
        <view class="px-3 py-2">
            ...
        </view>

        <!-- 自定义列表组件 -->
        <f-list v-for="(item, index) in list" :key="index" :item="item" :index="index"
            @my-select="handleSelect(index)"></f-list>
    </view>
</template>
```

效果




```
de1a39f (HEAD -> main) 封装通用列表组件
d50e186 加入搜索框
fa87564 自定义导航栏（二）
1047f35 自定义导航栏(一)
f349715 项目基础结构搭建
(END)
```