# 🔥HarmonyOS-硅谷租房项目

HarmonyOS-硅谷租房项目

# 1. 项目介绍

## 1.1. 项目描述

这是一个基于 **鸿蒙 API12** 开发的移动端租房 App，用户可以使用该应用搜索租房列表、查看房屋详情、预约租房等。

## 1.2. 项目截图

|  |
| --- |
| **首页** |
| 想看 |
| 服务 |
| 发现 |
| 我的 |
| 租房列表 |
| 租房详情 |
| 登录-手机号 |
| 登录-验证码 |
| 预约租房列表 |

## 1.3. 学习目标

* 掌握 ArkUI 组件的使用，合理搭建页面布局

* 掌握封装租房业务组件&公共组件

* 掌握移动端屏幕适配最佳实践

* 掌握前后端交互技术，封装企业级的请求函数

* 掌握租房项目核心业务流程

## 1.4. 项目资料

1. UI 设计稿

1. 项目代码

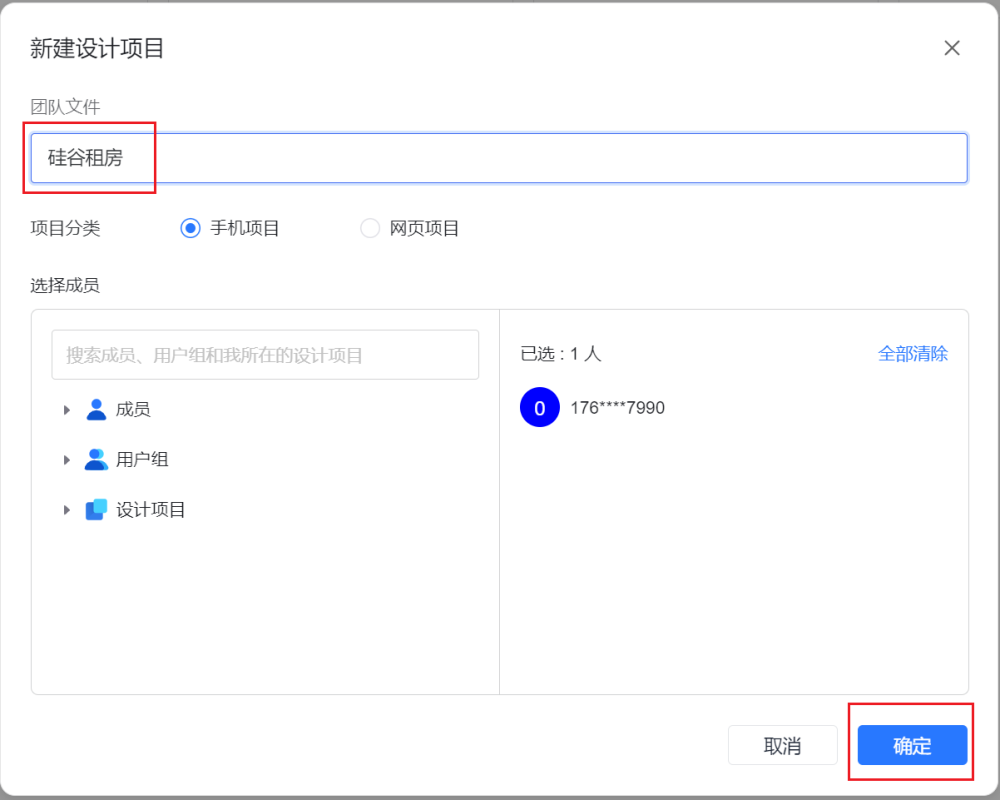
1. 服务器

1. Apifox 接口 JSON 文件

# 2. 项目准备

## 2.1. 将本地 UI 设计稿导入蓝湖

1. 注册并登录蓝湖，来到团队界面，新建设计项目

1. 前往 MasterGo 上传设计稿



1. 点击导入文件，开始上传设计稿

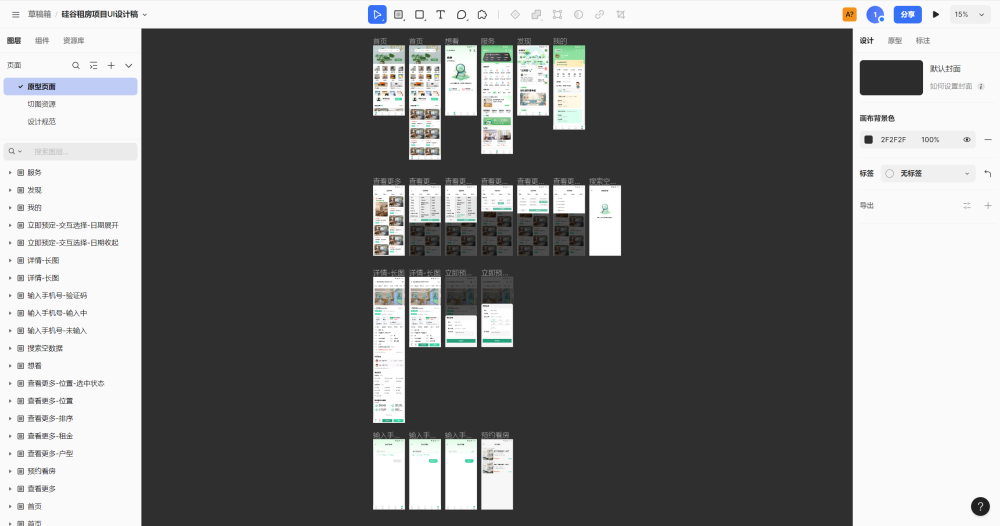




1. 等待导入，导入结束后即可查看设计稿



1. 双击硅谷租房进入设计稿

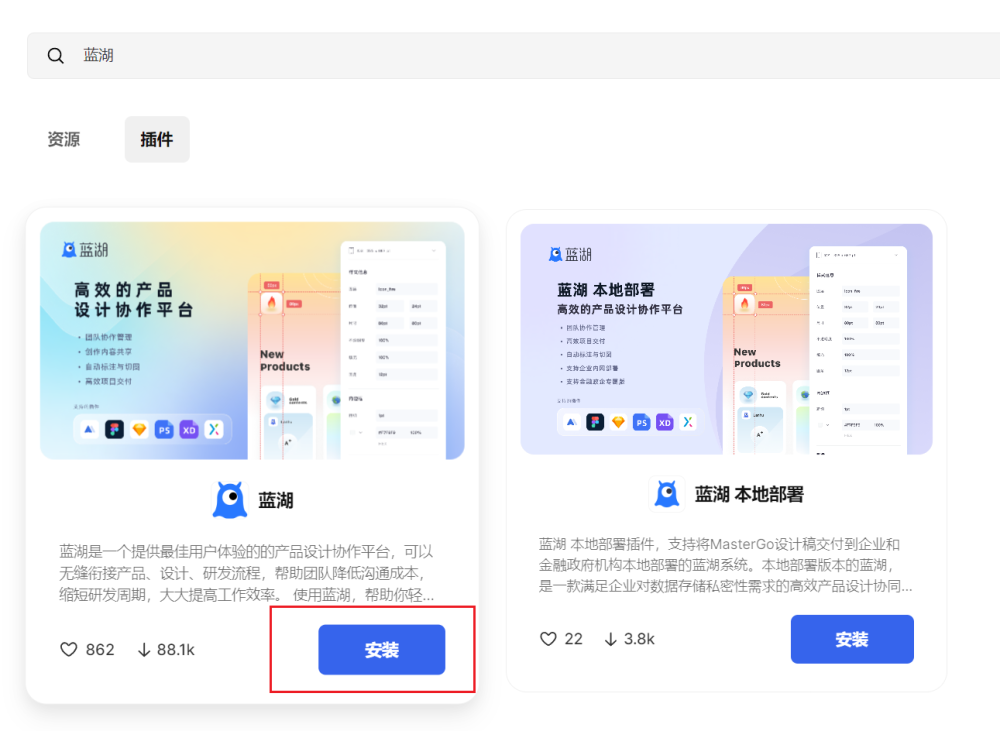
1. 在头部导航点击插件 - 获取更多插件



1. 输入"蓝湖"，按下回车开始搜索



1. 安装蓝湖插件



1. 安装完成，前往工作台



1. 选择硅谷租房项目进入，此时就能选择蓝湖插件了



1. 登录蓝湖



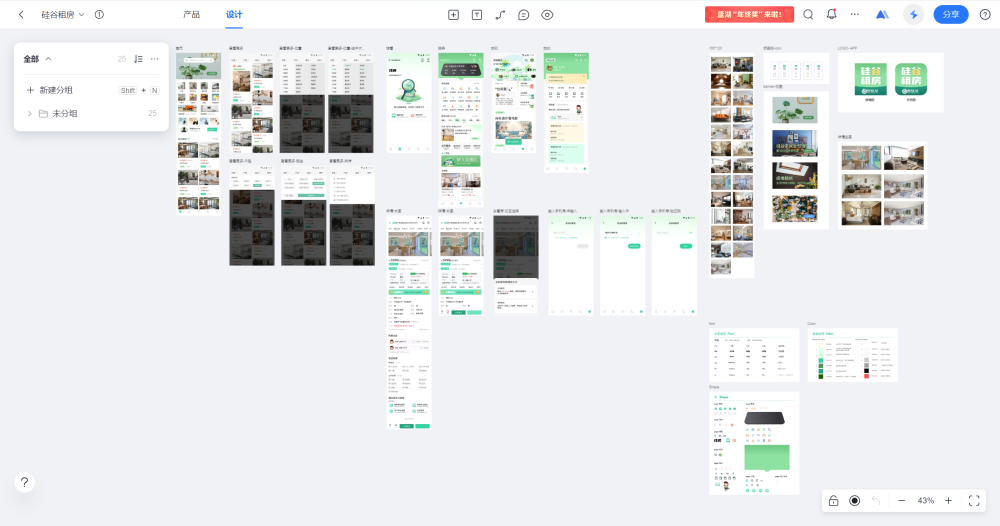
1. 选择该页全部容器，点击上传

注意：左侧三个页面都需要全部上传

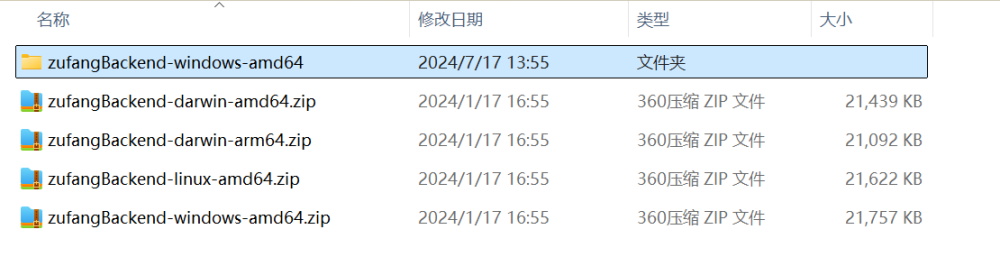


1. 点击查看设计，即可回到蓝湖，此时就能看到所有设计稿了~



## 2.2. 安装&启动服务器

1. 解压对应服务器



1. 双击运行（下图是运行后的效果，多了一些文件）



1. 需要注意的是，启动的服务器不能关闭，关闭服务就不能访问了

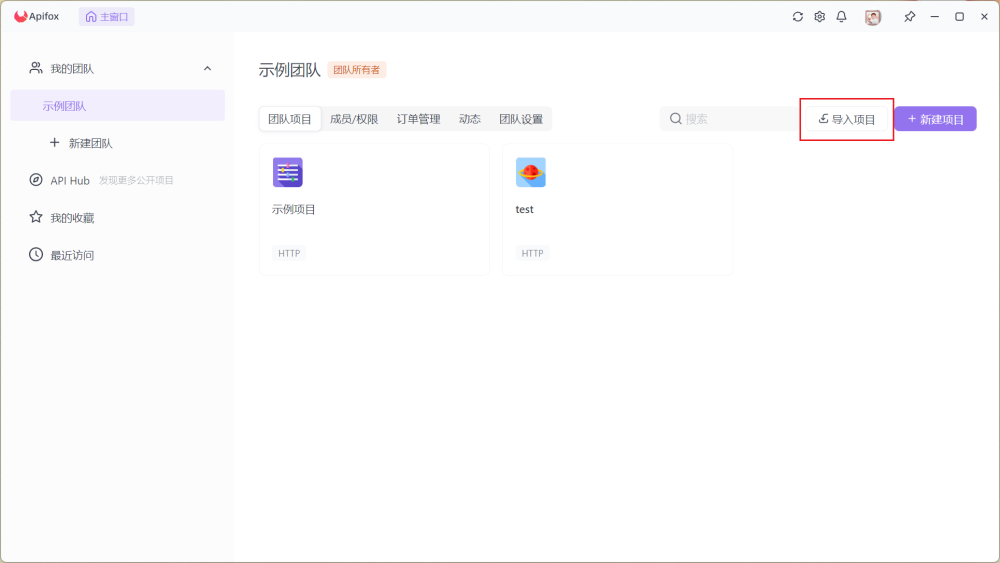
## 2.3. 使用 Apifox 测试接口

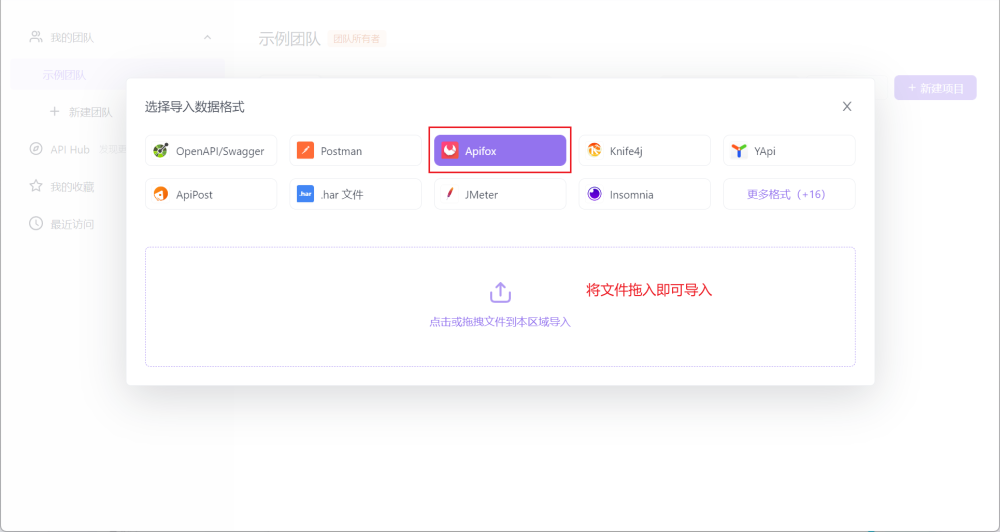
1. 软件下载地址

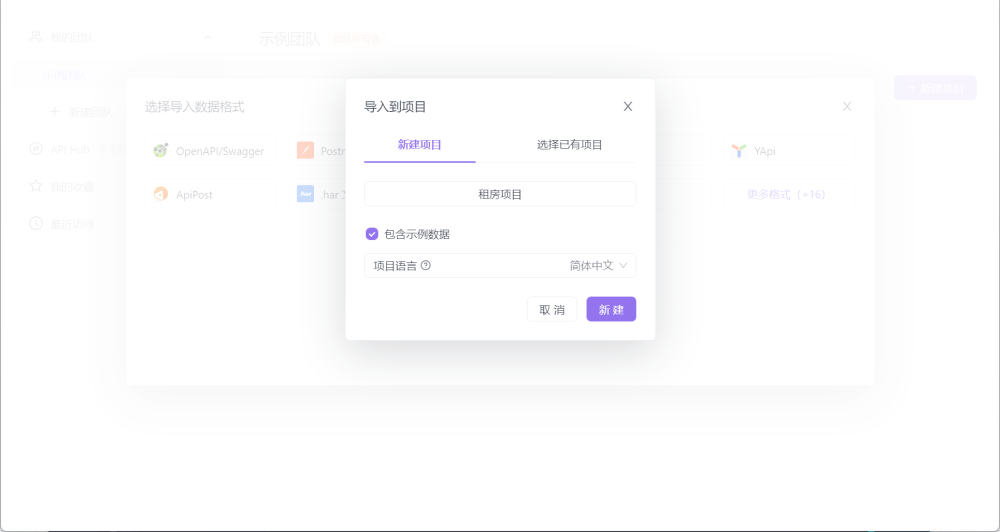
地址：[https://apifox.com](https://apifox.com/?utm_source=baidu_pinzhuan&utm_medium=sem&utm_campaign=pinzhuan&utm_content=pinzhuan&utm_term=apifox)

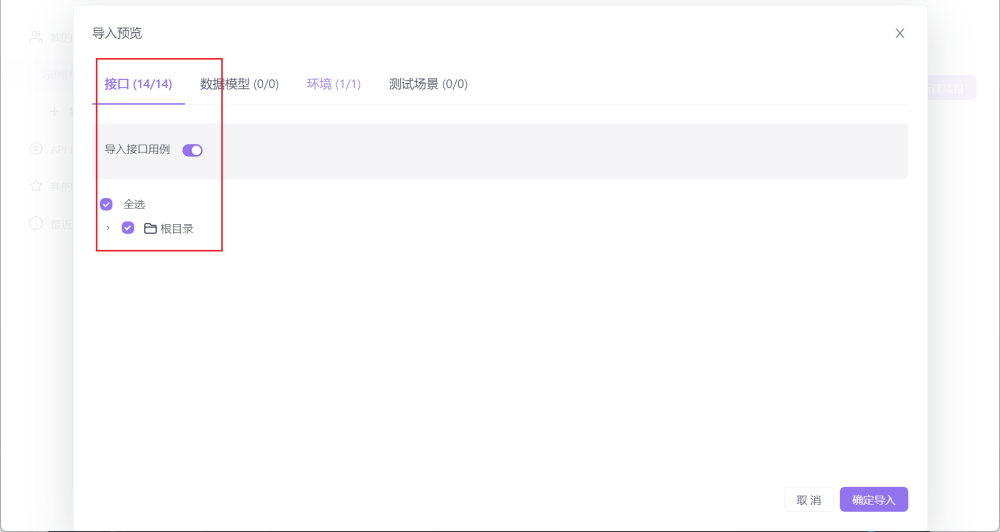
选择适合自己电脑的软件版本下载并安装

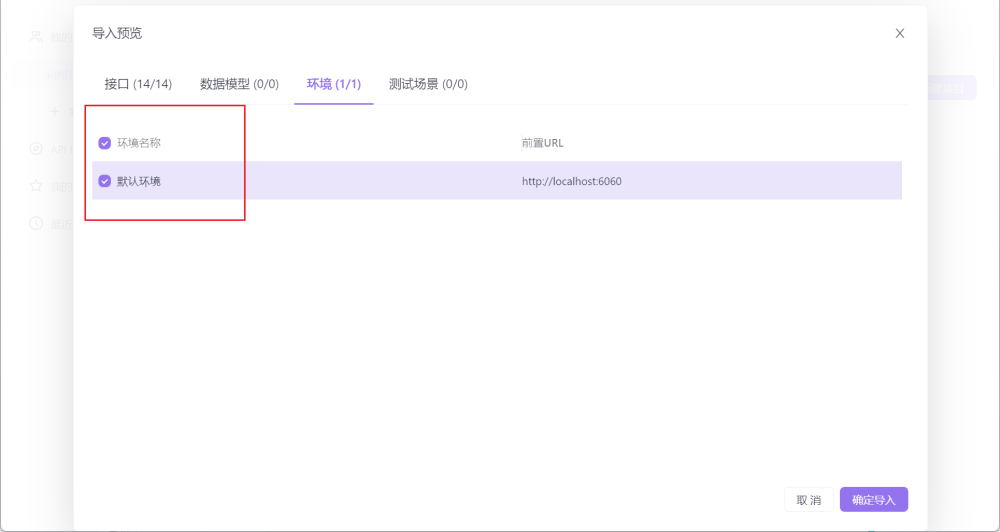
1. 导入接口 json 文件到 Apifox 中





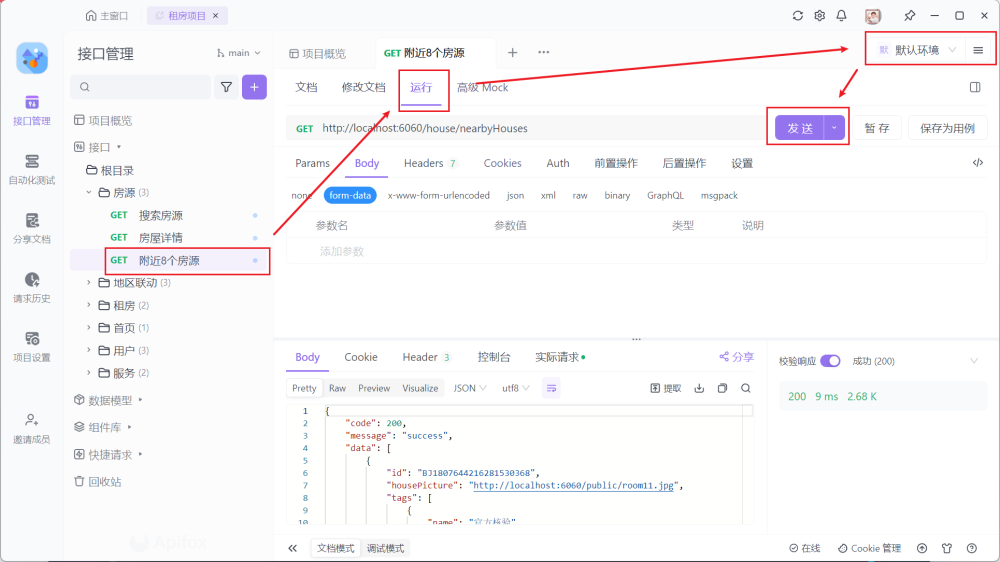






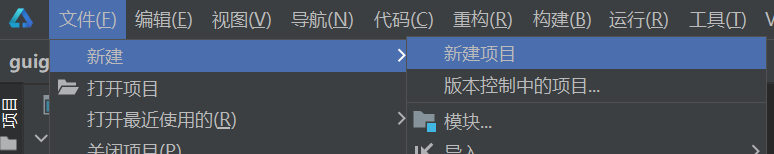
1. 测试接口

注意：记得启动服务器！

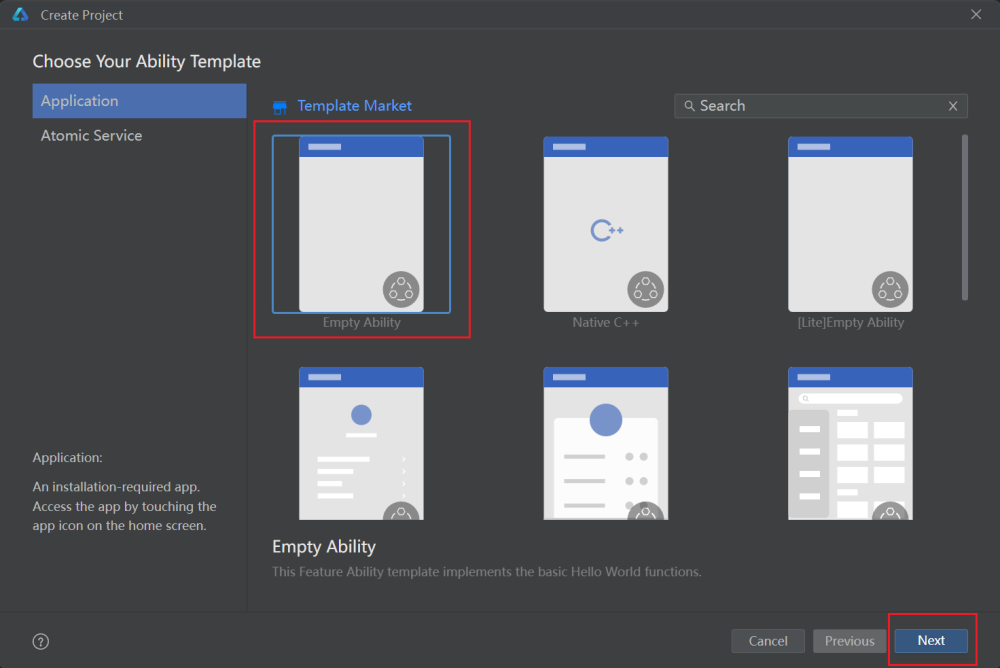


## 2.4. 创建空白项目

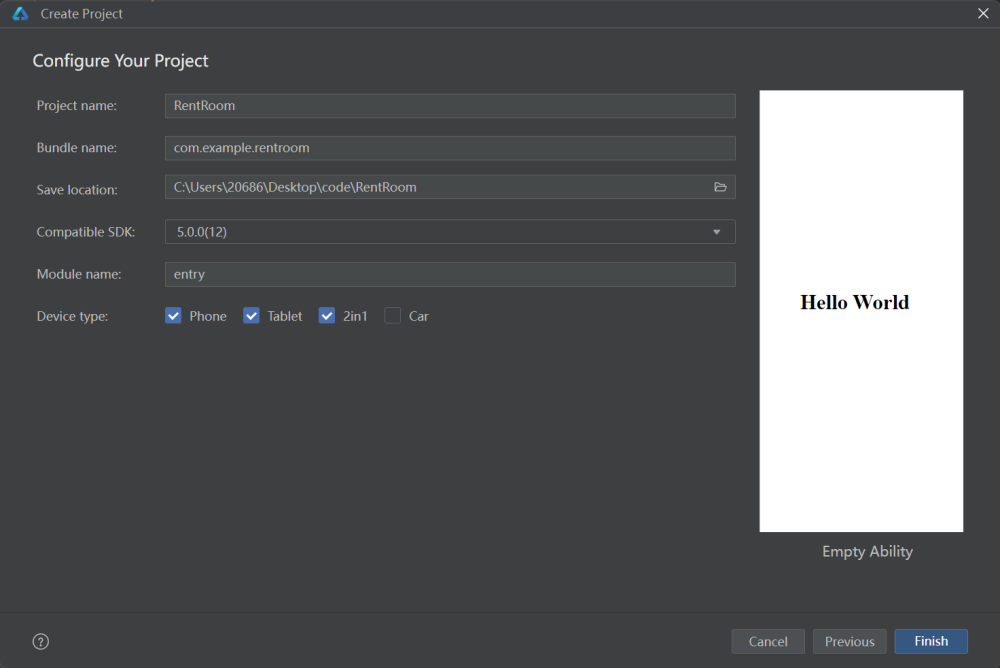
1. 通过 **DevEco Studio** 选择新建项目



1. 选择 **Empty Ability > Next**



1. 填写内容，其他默认值即可。点击 **Finish** 即可完成项目的创建

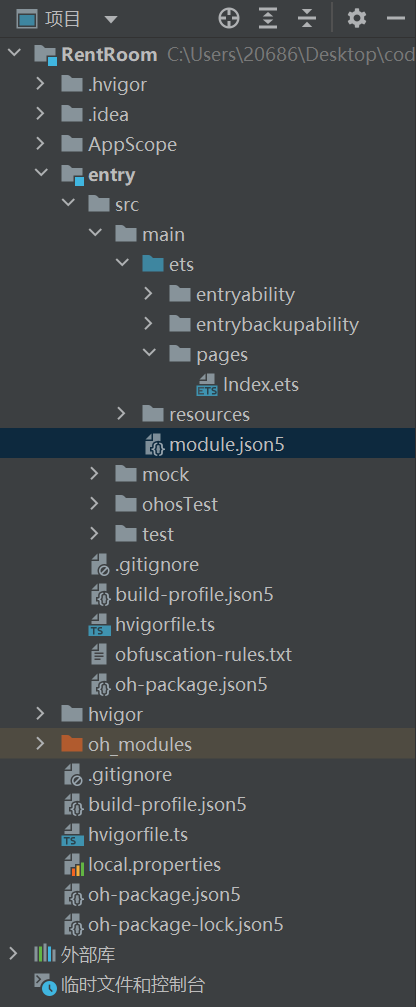


* Project name：项目名称

* Bundle name：应用的包名

* Save location：项目代码存储的路径

1. 创建好的项目目录如下所示



## 2.5. 设置项目图标和标题

### 2.5.1. 修改项目图标

引入图片资源，放入指定目录，覆盖掉之前图片即可

目录：**entry > src > main > resources > base > media**

图片：[📎startIcon.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512738410-edae9d4b-48a0-4d4f-bbda-f0dfc39dd5dc.png)[📎foreground.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512738616-a31394a0-15d1-4af6-a67b-1d779c2da442.png)[📎background.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512738727-bcb71ff9-cdd1-4366-846d-039d733c1841.png)

### 2.5.2. 修改项目标题

需要修改 **EntryAbility\_label** 字段的值。同时因为项目没有做国际化语言配置，所以直接使用中文即可。

1. 文件：**entry > src > main > resources > base > element > string.json**

{  
 "string": [  
 {  
 "name": "module\_desc",  
 "value": "module description"  
 },  
 {  
 "name": "EntryAbility\_desc",  
 "value": "description"  
 },  
 {  
 "name": "EntryAbility\_label",  
 "value": "硅谷租房"  
 }  
 ]  
}

1. 文件：**entry > src > main > resources > zh\_CN > element > string.json**

{  
 "string": [  
 {  
 "name": "module\_desc",  
 "value": "模块描述"  
 },  
 {  
 "name": "EntryAbility\_desc",  
 "value": "硅谷租房描述"  
 },  
 {  
 "name": "EntryAbility\_label",  
 "value": "硅谷租房"  
 }  
 ]  
}

1. 文件：**entry > src > main > resources > en\_US > element > string.json**

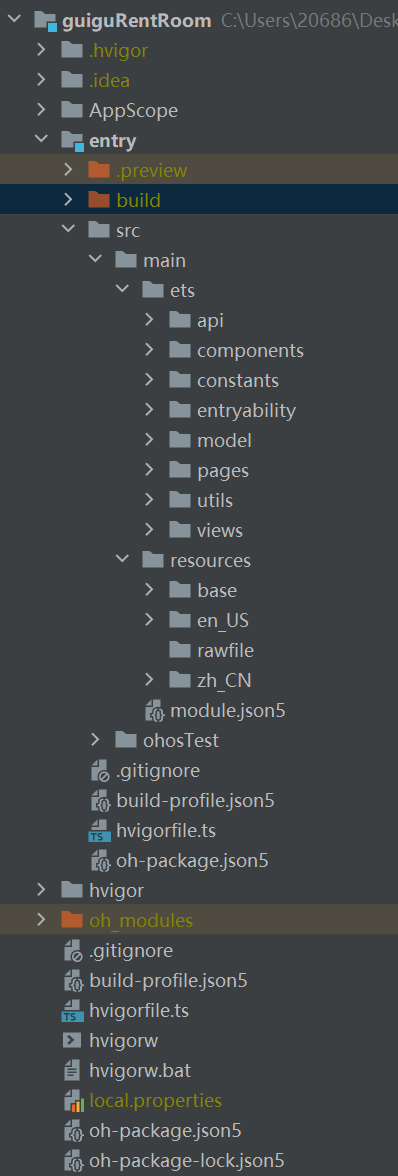
{  
 "string": [  
 {  
 "name": "module\_desc",  
 "value": "module description"  
 },  
 {  
 "name": "EntryAbility\_desc",  
 "value": "description"  
 },  
 {  
 "name": "EntryAbility\_label",  
 "value": "RentRoom"  
 }  
 ]  
}

实际应用名称和图标并不是只能这么写，它可以进行配置：**entry > src > main > module.json5**



其中，icon 就是配置应用图标，label 就是配置应用名称。

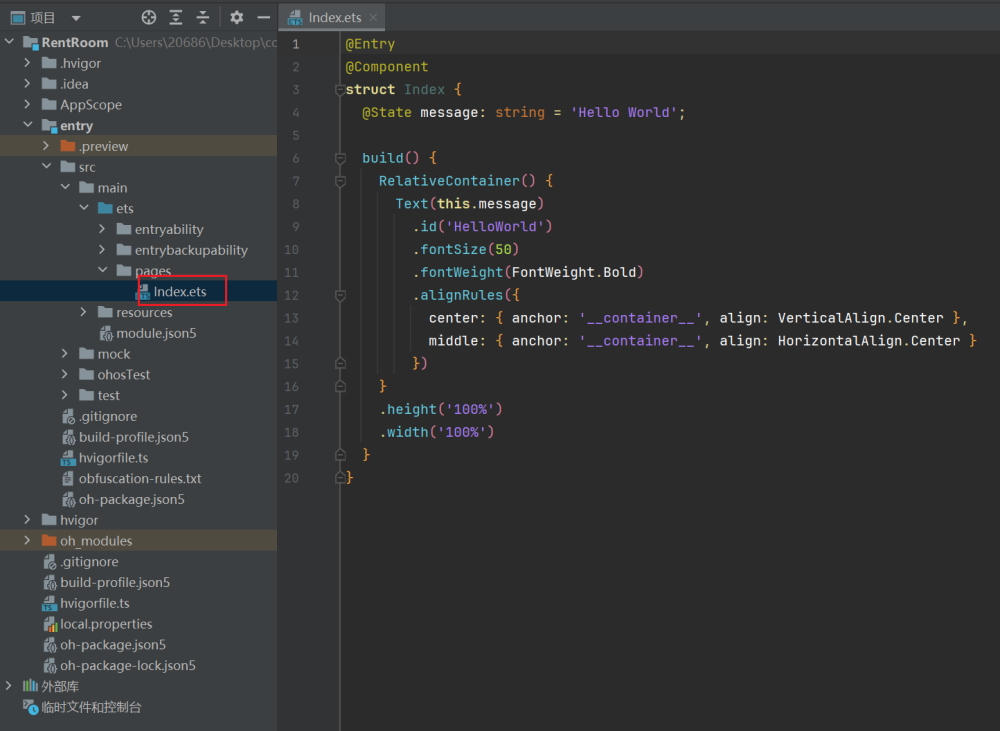
## 2.6. 项目目录说明



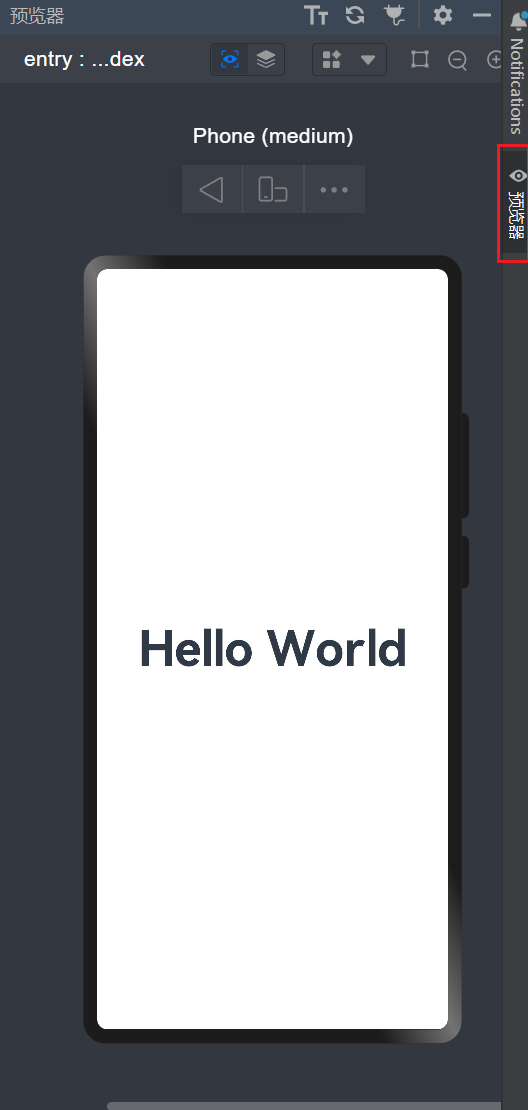
├─ .hvigor -------------------------------- 构建生成的缓存等信息  
├─ .idea ---------------------------------- 项目的配置信息：编译配置、文件编码信息等  
├─ AppScope ------------------------------- 应用/服务的全局公共资源目录  
 | ├─ resource ---------------------------- 应用/服务的全局资源  
 | └─ app.json5 --------------------------- 应用/服务的全局配置信息  
├─ entry ---------------------------------- 应用/服务模块，编译构建生成一个 HAP  
 | ├─ .preview ---------------------------- 预览项目编译、配置信息等  
 | ├─ build ------------------------------- 预览项目编译、配置信息等  
 | ├─ src --------------------------------- 项目源码目录  
 | | ├─ main  
 | | | ├─ ets --------------------------- 用于存放 ArkTS 源码  
 | | | | ├─ api ------------------------ 接口函数目录  
 | | | | ├─ components ----------------- 公共组件目录  
 | | | | ├─ constants ------------------ 常量目录  
 | | | | ├─ entryability --------------- 应用/服务的入口  
 | | | | | └─ EntryAbility.ts  
 | | | | ├─ model ---------------------- 类型目录  
 | | | | ├─ pages ---------------------- 应用/服务包含的页面  
 | | | | ├─ utils ---------------------- 工具函数目录  
 | | | | └─ views ---------------------- 页面中组件目录  
 | | | ├─ module.json5 ------------------ Stage 模型模块配置文件，主要包含 HAP 的配置信息、应用在具体设备上的配置信息以及应用的全局配置信息。  
 | | | └─ resources --------------------- 用于存放应用/服务所用到的资源文件  
 | | | ├─ base  
 | | | | ├─ element  
 | | | | | ├─ color.json ----------- 颜色  
 | | | | | ├─ float.json ----------- 浮点型  
 | | | | | └─ string.json ---------- 字符串  
 | | | | ├─ media ------------------- 图片、视频、音频等  
 | | | | └─ profile  
 | | | | └─ main\_pages.json ------ 页面 page 的路径配置信息，所有需要进行路由跳转的 page 页面都要在这里进行配置。  
 | | | ├─ en\_US ---------------------- 英文语言  
 | | | ├─ rawfile -------------------- 任意格式资源  
 | | | └─ zh\_CN ---------------------- 中文语言  
 | | └─ ohosTest ------------------------- 单元测试目录  
 | ├─ .gitignore -------------------------- 模块的 Git 忽略文件  
 | ├─ build-profile.json5 ----------------- 模块配置信息，包括签名、产品配置等  
 | ├─ hvigorfile.ts ----------------------- 模块构建配置任务脚本  
 | ├─ obfuscation-rules ------------------- 代码混淆配置文件  
 | └─ oh-package.json5 ------------------- 模块第三方包声明文件的入口及包名  
├─ hvigor --------------------------------- 构建配置文件信息，是一款全新基于 TS 实现的前端构建任务编排工具  
 | ├─ hvigor-config.json5  
 | └─ hvigor-wrapper.js  
├─ oh\_modules ----------------------------- 用于存放三方库依赖信息  
├─ .gitignore ----------------------------- Git 提交时的忽略文件  
├─ build-profile.json5 -------------------- 应用级配置信息，包括签名、产品配置等  
├─ hvigorfile.ts -------------------------- 构建配置任务脚本  
├─ hvigorw  
├─ hvigorw.bat  
├─ local.properties ----------------------- 存放私有属性路径，比如 Nodejs 路径、Huawei SDK 路径  
├─ oh-package.json5 ----------------------- 配置第三方包声明文件的入口及包名  
└─ oh-package-lock.json5

## 2.7. 项目预览

1. 选择 **entry > src > main > ets > pages > Index.ets** 文件

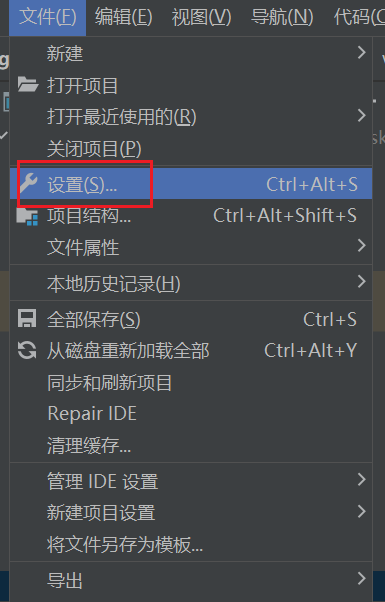


1. 点击右边的预览器开始预览

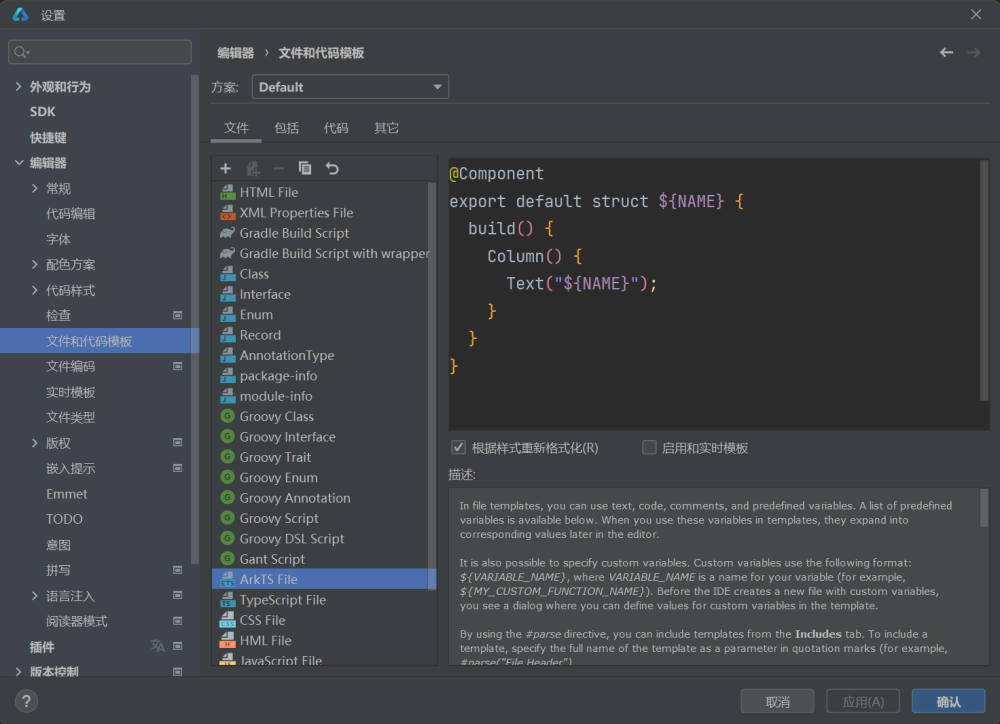


## 2.8. 配置项目代码模板

1. 选择 **文件 > 设置**



1. 搜索 **代码模板**，选择 **ArkTS File**，输入模板代码**。**



@Component  
export default struct ${NAME} {  
 build() {  
 Column() {  
 Text("${NAME}");  
 }  
 }  
}

1. 点击应用和确定保存，将来新建 ArkTS 文件时，就会有代码模板了

## 2.9. 定义项目常量

1. 在 **src > main > ets > constants 目录下定义**

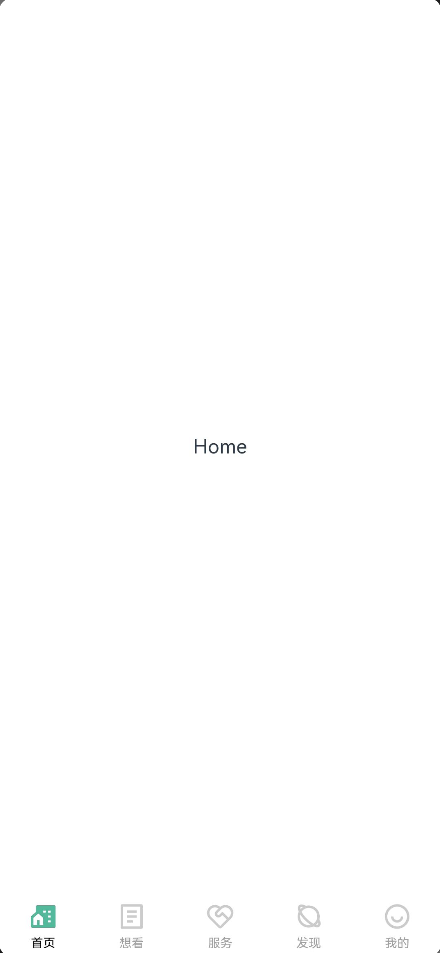
// 边距  
export const PADDING\_S = 10;  
export const PADDING = 16;  
export const PADDING\_L = 20;  
  
// 圆角  
export const BORDER\_RADIUS\_S = 4;  
export const BORDER\_RADIUS = 8;  
export const BORDER\_RADIUS\_L = 12;  
  
// 阴影  
export const SHADOW\_RADIUS = 7.8;

1. 在 **src > main > resources > base > element** 目录下定义

{  
 "color": [  
 {  
 "name": "start\_window\_background",  
 "value": "#FFFFFF"  
 },  
 {  
 "name": "bg\_gray",  
 "value": "#f7f7f7"  
 },  
 {  
 "name": "bg\_gray\_second",  
 "value": "#f9f9f9"  
 },  
 {  
 "name": "gray\_second",  
 "value": "#666666"  
 },  
 {  
 "name": "primary",  
 "value": "#36d1a1"  
 },  
 {  
 "name": "shadow",  
 "value": "#24000000"  
 },  
 {  
 "name": "black",  
 "value": "#000"  
 },  
 {  
 "name": "white",  
 "value": "#fff"  
 },  
 {  
 "name": "gray",  
 "value": "#999999"  
 }  
 ]  
}

# 3. TabBar 页面功能

## 3.1. 页面效果图



## 3.2. 创建标签页组件

1. 使用 **右键 > 新建 > ArkTS File** 来创建组件
2. **新建 > Page** 和 **新建 > ArkTS File** 有什么区别 **？**

* **新建 > Page** 会创建组件，同时自动在 **entry >** **src > main > resources > base > profile > main\_pages.json** 中注册路由，将来可以通过 [**@ohos.router**](https://developer.huawei.com/consumer/cn/doc/harmonyos-references-V2/js-apis-router-0000001478061893-V2)来进行路由跳转

* **新建 > ArkTS File** 不会注册路由，只会创建组件

1. 为什么这里需要 **新建 > ArkTS File** 来创建组件？

因为 Tab 标签页是不需要进行路由跳转，直接点击切换即可，不需要注册路由

1. 如果我不进行路由跳转，能不能用 **新建 > Page** 方式创建组件？

实际上可以，但是不好

1. 创建组件如下

注意：组件首字母大写

@Component  
export default struct Home {  
 build() {  
 Row() {  
 Text('Home')  
 }  
 }  
}

@Component  
export default struct See {  
 build() {  
 Row() {  
 Text('See')  
 }  
 }  
}

@Component  
export default struct Service {  
 build() {  
 Row() {  
 Text('Service')  
 }  
 }  
}

@Component  
export default struct Discover {  
 build() {  
 Row() {  
 Text('Discover')  
 }  
 }  
}

@Component  
export default struct My {  
 build() {  
 Row() {  
 Text('My')  
 }  
 }  
}

## 3.3. 引入图片资源

从 UI 设计稿中下载 TabBar 的切图，移动到项目 **entry > src > main > resources > base > media** 中

[📎tabbar\_discover.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512738827-7cf2456b-595f-4926-a804-6d1758ca1b23.svg)[📎tabbar\_discover\_active.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512738968-03789b29-ae43-4f53-8e8b-c7c6244fb54a.svg)[📎tabbar\_home.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512739148-0b797fe6-8ca3-45a4-91b0-764708fc57fc.svg)[📎tabbar\_home\_active.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512739264-e0c65aba-270c-4648-b76a-a64f46209f7b.svg)[📎tabbar\_my.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512739498-a0b6a071-feb0-4924-8f82-ccb421e7df36.svg)[📎tabbar\_my\_active.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512739729-47625462-fe40-44ee-a443-293d6bc40feb.svg)[📎tabbar\_see.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512739909-e0a8dd00-abb4-48d8-8c79-9f229ab55a25.svg)[📎tabbar\_see\_active.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512739999-51462a11-8f7c-44b9-a1ac-d2895ca7ae74.svg)[📎tabbar\_service.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512740449-818b2e28-0a1a-4931-be57-0d9a375365a5.svg)[📎tabbar\_service\_active.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512740558-f5fbf5a4-89c9-4cd7-a6b6-f2812f483f31.svg)

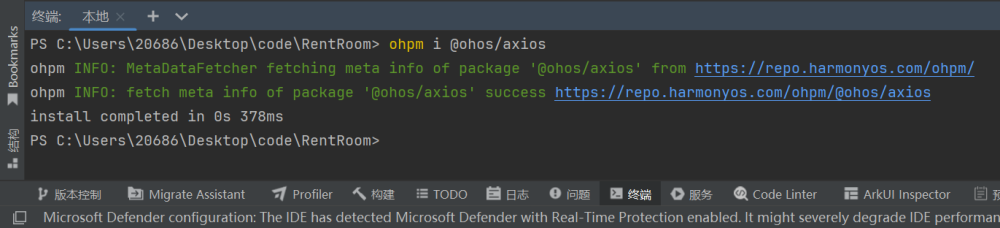
注意：图片名称只能包含：英文、数字和下划线。一旦包含其他命名的图片，即使你没有使用，运行也会报错。

## 3.4. 搭建 TabBar 布局

import Home from './Home';  
import See from './See';  
import Service from './Service';  
import Discover from './Discover';  
import My from './My';  
  
@Entry  
@Component  
struct Index {  
  
 // 当前 tab 高亮的下标  
 @State currentIndex: number = 0;  
  
 // 生成 tab 的函数  
 @Builder TabBuilder(index: number, text: string, icon: Resource, activeIcon: Resource) {  
 Column() {  
 // 项目中常用的颜色、大小等会定义成常量，将来方便复用和统一调整。而不常用的大小颜色就不需要定义了，直接写死即可。  
 Image(this.currentIndex === index ? activeIcon : icon).width(28)  
 Text(text).fontSize(10).fontColor(this.currentIndex === index ? $r('app.color.black') : '#a0a0a0').height(15)  
 }.width('100%').height('100%').justifyContent(FlexAlign.Center)  
 }  
  
 build() {  
 Column() {  
 Tabs({ barPosition: BarPosition.End, index: this.currentIndex }) {  
 TabContent() {  
 Home()  
 }.tabBar(this.TabBuilder(0, '首页', $r('app.media.tabbar\_home'), $r('app.media.tabbar\_home\_active')))  
  
 TabContent() {  
 See()  
 }.tabBar(this.TabBuilder(1, '想看', $r('app.media.tabbar\_see'), $r('app.media.tabbar\_see\_active')))  
  
 TabContent() {  
 Service()  
 }.tabBar(this.TabBuilder(2, '服务', $r('app.media.tabbar\_service'), $r('app.media.tabbar\_service\_active')))  
  
 TabContent() {  
 Discover()  
 }.tabBar(this.TabBuilder(3, '发现', $r('app.media.tabbar\_discover'), $r('app.media.tabbar\_discover\_active')))  
  
 TabContent() {  
 My()  
 }.tabBar(this.TabBuilder(4, '我的', $r('app.media.tabbar\_my'), $r('app.media.tabbar\_my\_active')))  
 }  
 .vertical(false)  
 .barMode(BarMode.Fixed)  
 .onChange((index: number) => {  
 this.currentIndex = index  
 })  
 .barHeight(50) // 设置 tab 导航栏的高度，默认 56  
 .scrollable(false)  
 .width('100%')  
 .height('100%')  
 .backgroundColor($r('app.color.white'))  
 }.height('100%')  
 }  
}

# 4. 请求函数封装

1. 下载依赖



ohpm i @ohos/axios

1. 定义 **utils > http > http.ets** 文件

import axios, { AxiosError, AxiosResponse, InternalAxiosRequestConfig } from '@ohos/axios';  
  
const request = axios.create({  
 baseURL: 'http://172.17.0.195:6060', // 通过 ipconfig 查询当前电脑的 ip，使用自己电脑的 ip 地址  
 timeout: 20000,  
})  
  
request.interceptors.request.use(  
 (config: InternalAxiosRequestConfig) => {  
 // 未来需要添加 token  
 // config.headers.token = token;  
 return config;  
 }  
)  
  
request.interceptors.response.use(  
 (response: AxiosResponse) => {  
 if (response.data.code === 200) {  
 return response.data.data;  
 } else {  
 return Promise.reject(response.data.message);  
 }  
 },  
 (error: AxiosError) => {  
 return Promise.reject(error.message);  
 }  
)  
  
export default class Http {  
 get<T>(url: string, params?: Object) {  
 return request.get<null, T>(url, {  
 params  
 })  
 }  
  
 post<T>(url: string, data?: Object) {  
 return request.post<null, T>(url, data)  
 }  
  
 put<T>(url: string, data?: Object) {  
 return request.put<null, T>(url, data)  
 }  
  
 delete<T>(url: string, params?: Object) {  
 return request.delete<null, T>(url, {  
 params  
 })  
 }  
};

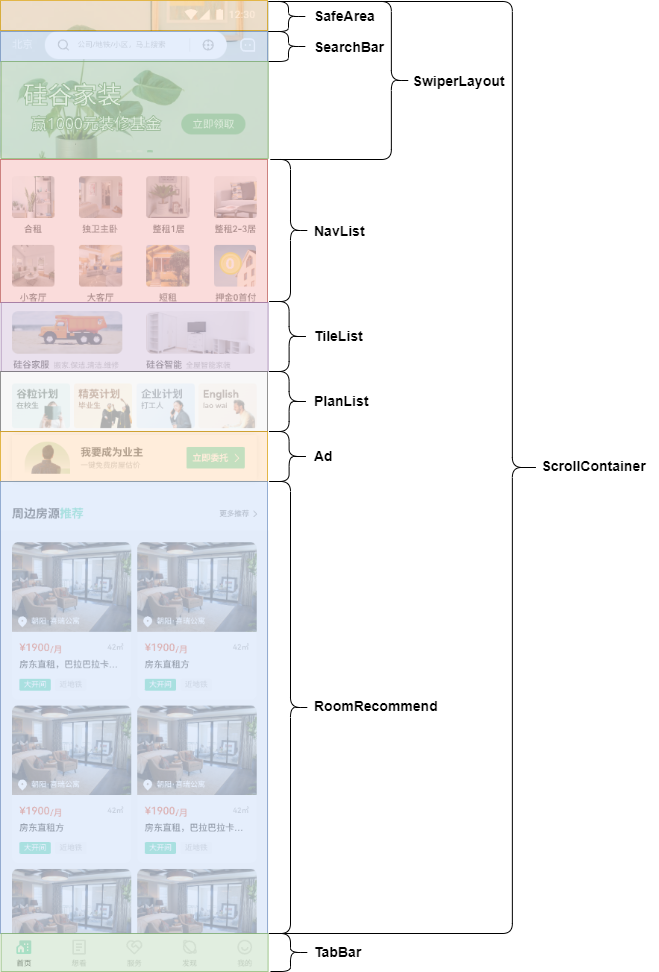
import Http from './http';  
  
export const http = new Http();

1. 申请网络权限

{  
 "module": {  
 // ...  
 "requestPermissions":[ // 申请权限  
 { "name" : "ohos.permission.INTERNET" }, // 使用网络  
 ]  
 }  
}

# 5. 首页页面功能

## 5.1. 布局分析



## 5.2. SwiperLayout 组件功能

### 5.2.1. 搭建组件布局

1. 测试接口得到数据

[  
 {  
 "id": 1,  
 "name": "banner1",  
 "imageURL": "http://localhost:6060/public/home\_banner\_1.jpg"  
 },  
 {  
 "id": 2,  
 "name": "banner2",  
 "imageURL": "http://localhost:6060/public/home\_banner\_2.jpg"  
 },  
 {  
 "id": 3,  
 "name": "banner3",  
 "imageURL": "http://localhost:6060/public/home\_banner\_3.jpg"  
 },  
 {  
 "id": 4,  
 "name": "banner4",  
 "imageURL": "http://localhost:6060/public/home\_banner\_4.jpg"  
 }  
]

1. 定义类型文件 **model > HomeData**

export interface IBannerItem {  
 "id": number  
 "name": string  
 "imageURL": string  
}  
  
export type IBannerList = IBannerItem[]

1. 搭建 **views > Home > SwiperLayout** 组件

import type { IBannerItem, IBannerList } from '../../model/HomeData'  
  
@Component  
export default struct SwiperLayout {  
 @State bannerList: IBannerList = [  
 {  
 "id": 1,  
 "name": "banner1",  
 "imageURL": "http://localhost:6060/public/home\_banner\_1.jpg"  
 },  
 {  
 "id": 2,  
 "name": "banner2",  
 "imageURL": "http://localhost:6060/public/home\_banner\_2.jpg"  
 },  
 {  
 "id": 3,  
 "name": "banner3",  
 "imageURL": "http://localhost:6060/public/home\_banner\_3.jpg"  
 },  
 {  
 "id": 4,  
 "name": "banner4",  
 "imageURL": "http://localhost:6060/public/home\_banner\_4.jpg"  
 }  
 ]  
  
 build() {  
 Swiper() {  
 ForEach(this.bannerList, (banner: IBannerItem) => {  
 Image(banner.imageURL).width('100%').height(175)  
 }, (item: IBannerItem) => item.id + '')  
 }  
 .loop(true)  
 .indicator(  
 Indicator.dot()// 设置圆形导航点样式  
 .color('#CCCBCB')  
 .selectedColor($r('app.color.primary'))  
 )  
 }  
}

1. **Home** 组件使用 **SwiperLayout**

import SwiperLayout from '../views/Home/SwiperLayout'  
  
@Component  
export default struct Home {  
 build() {  
 Column() {  
 SwiperLayout()  
 }.height('100%')  
 }  
}

### 5.2.2. 发送请求，获取数据

1. 定义接口返回值类型 **model > HomeData**

export interface IHomeData {  
 "bannerList": IBannerList  
 "navList": INavList  
 "tileList": ITileList  
 "planList": IPlanList  
 "adPicture": string  
}  
  
// 计划列表类型  
export interface IPlanItem {  
 "id": number  
 "imageURL": string  
}  
  
export type IPlanList = IPlanItem[]  
  
// 瓷片列表类型  
export interface ITileItem {  
 "id": number  
 "imageURL": string  
 "title": string  
 "sub\_title": string  
}  
  
export type ITileList = ITileItem[]  
  
// 导航列表类型  
export interface INavItem {  
 "id": number  
 "title": string  
 "imageURL": string  
}  
  
export type INavList = INavItem[]  
  
// 轮播图类型  
export interface IBannerItem {  
 "id": number  
 "name": string  
 "imageURL": string  
}  
  
export type IBannerList = IBannerItem[]

1. 定义接口函数 **api > home**

import { http } from '../utils/http';  
import type { IHomeData } from '../model/HomeData';  
  
// 获取首页数据  
export const getHomeDataApi = (): Promise<IHomeData> => {  
 return http.get<IHomeData>('/home/info');  
}

1. **Home** 组件发送请求获取数据

import SwiperLayout from '../views/Home/SwiperLayout';  
  
import { getHomeDataApi } from '../api/home';  
import type { IBannerList, INavList, IPlanList, ITileList } from '../model/HomeData';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: IBannerList = [];  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = homeData.bannerList;  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 build() {  
 Column() {  
 SwiperLayout()  
 }.height('100%')  
 }  
}

### 5.2.3. 父子组件通信

父子组件通信，所以使用 @Prop 方案通信

1. 父页面 **Home** 将 **bannerList** 数据通信给子页面 **SwiperLayout**

import SwiperLayout from '../views/Home/SwiperLayout';  
  
import { getHomeDataApi } from '../api/home';  
import type { IBannerList, INavList, IPlanList, ITileList } from '../model/HomeData';  
  
@Component  
 export default struct Home {  
 // 首页数据  
 @State bannerList: IBannerList = [];  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = homeData.bannerList;  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 build() {  
 Column() {  
 SwiperLayout({ bannerList: this.bannerList })  
 }.height('100%')  
 }  
 }

1. 子页面 **SwiperLayout** 声明接受 @Prop 数据并使用

import type { IBannerItem, IBannerList } from '../../model/HomeData'  
  
@Component  
export default struct SwiperLayout {  
 @Prop bannerList: IBannerList  
  
 build() {  
 Swiper() {  
 ForEach(this.bannerList, (banner: IBannerItem) => {  
 Image(banner.imageURL).width('100%').height(175)  
 }, (item: IBannerItem) => item.id + '')  
 }  
 .loop(true)  
 .indicator(  
 Indicator.dot()// 设置圆形导航点样式  
 .color('#CCCBCB')  
 .selectedColor($r('app.color.primary'))  
 )  
 }  
}

### 5.2.4. 使用 LazyForEach

1. 定义 BannerListDataSource 数据

// https://developer.huawei.com/consumer/cn/doc/harmonyos-guides-V5/arkts-rendering-control-lazyforeach-V5  
export class BannerListDataSource implements IDataSource {  
 // 内部用来触发UI更新对象  
 listeners: DataChangeListener[] = []  
 // 管理数据  
 bannerList: IBannerList = []  
  
 // 实现了 IDataSource 接口  
 // 接口要求我们必须定义四个方法：getData、totalCount、registerDataChangeListener、unregisterDataChangeListener  
 // 读取对应下标的数据  
 getData(index: number): IBannerItem {  
 return this.bannerList[index]  
 }  
  
 // 返回数据总长度  
 totalCount(): number {  
 return this.bannerList.length  
 }  
  
 // 保存单个数据  
 setData(item: IBannerItem) {  
 // 只会更新数据，不会更新UI  
 this.bannerList.push(item);  
 // 更新UI  
 this.notifyDataAdd(this.bannerList.length - 1);  
 }  
  
 // 更新所有数据  
 setList(list: IBannerList) {  
 this.bannerList = list;  
 this.notifyDataReload();  
 }  
  
 // 该方法为框架侧调用，为LazyForEach组件向其数据源处添加listener监听  
 registerDataChangeListener(listener: DataChangeListener): void {  
 if (this.listeners.indexOf(listener) < 0) {  
 console.info('add listener');  
 this.listeners.push(listener);  
 }  
 }  
  
 // 该方法为框架侧调用，为对应的LazyForEach组件在数据源处去除listener监听  
 unregisterDataChangeListener(listener: DataChangeListener): void {  
 const pos = this.listeners.indexOf(listener);  
 if (pos >= 0) {  
 console.info('remove listener');  
 this.listeners.splice(pos, 1);  
 }  
 }  
  
 // 通知LazyForEach组件需要重载所有子组件  
 notifyDataReload(): void {  
 this.listeners.forEach(listener => {  
 listener.onDataReloaded();  
 })  
 }  
  
 // 通知LazyForEach组件需要在index对应索引处添加子组件  
 notifyDataAdd(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataAdd(index);  
 })  
 }  
  
 // 通知LazyForEach组件在index对应索引处数据有变化，需要重建该子组件  
 notifyDataChange(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataChange(index);  
 })  
 }  
  
 // 通知LazyForEach组件需要在index对应索引处删除该子组件  
 notifyDataDelete(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataDelete(index);  
 })  
 }  
  
 // 通知LazyForEach组件将from索引和to索引处的子组件进行交换  
 notifyDataMove(from: number, to: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataMove(from, to);  
 })  
 }  
}  
  
  
export interface IHomeData {  
 "bannerList": IBannerList  
 "navList": INavList  
 "tileList": ITileList  
 "planList": IPlanList  
 "adPicture": string  
}  
  
export interface IBannerItem {  
 "id": number  
 "name": string  
 "imageURL": string  
}  
  
export type IBannerList = IBannerItem[]  
  
export interface INavItem {  
 "id": number  
 "title": string  
 "imageURL": string  
}  
  
export type INavList = INavItem[]  
  
export interface ITileItem {  
 "id": number  
 "imageURL": string  
 "title": string  
 "sub\_title": string  
}  
  
export type ITileList = ITileItem[]  
  
export interface IPlanItem {  
 "id": number  
 "imageURL": string  
}  
  
export type IPlanList = IPlanItem[]

1. Home 组件获取数据后，使用 BannerListDataSource 更新数据

import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource } from '../models/HomeData';  
  
import SwiperLayout from '../views/Home/SwiperLayout';  
  
@Component  
export default struct Home {  
 @State bannerList: BannerListDataSource = new BannerListDataSource()  
 // 获取首页数据函数  
 getHomeData = async () => {  
 const result = await getHomeDataApi()  
 this.bannerList.setList(result.bannerList);  
 }  
  
 // 生命周期函数：初始化渲染时触发  
 aboutToAppear(): void {  
 this.getHomeData()  
 }  
  
 build() {  
 Scroll() {  
 Column() {  
 // 轮播图组件（使用prop通信）  
 SwiperLayout({ bannerList: this.bannerList })  
 }  
 .width('100%')  
 }  
 .width('100%')  
 .height('100%')  
 .scrollBar(BarState.Off)  
 .align(Alignment.TopStart)  
 }  
}

1. 子组件 SwiperLayout 声明接受并使用

import { BannerListDataSource, IBannerItem } from '../../models/HomeData';  
  
@Component  
export default struct SwiperLayout {  
 @Prop bannerList: BannerListDataSource;  
  
 build() {  
 Swiper() {  
 LazyForEach(this.bannerList, (banner: IBannerItem) => {  
 Image(banner.imageURL).width('100%').height('100%').objectFit(ImageFit.Fill)  
 }, (banner: IBannerItem) => banner.id + '')  
 }  
 .width('100%')  
 .height(211 - 36) // 暂时减去状态栏高度  
 .autoPlay(true) // 自动轮播  
 .indicator( // 设置圆点导航点样式  
 new DotIndicator()  
 .color($r('app.color.indicator\_color'))  
 .selectedColor($r('app.color.indicator\_active\_color'))  
 )  
 }  
}

1. 此时存在一个问题，轮播图出不来。这里我们需要使用 @Observed 和 @ObjectLink

// https://developer.huawei.com/consumer/cn/doc/harmonyos-guides-V5/arkts-rendering-control-lazyforeach-V5  
@Observed  
export class BannerListDataSource implements IDataSource {  
 // 内部用来触发UI更新对象  
 listeners: DataChangeListener[] = []  
 // 管理数据  
 bannerList: IBannerList = []  
  
 // 实现了 IDataSource 接口  
 // 接口要求我们必须定义四个方法：getData、totalCount、registerDataChangeListener、unregisterDataChangeListener  
 // 读取对应下标的数据  
 getData(index: number): IBannerItem {  
 return this.bannerList[index]  
 }  
  
 // 返回数据总长度  
 totalCount(): number {  
 return this.bannerList.length  
 }  
  
 // 保存单个数据  
 setData(item: IBannerItem) {  
 // 只会更新数据，不会更新UI  
 this.bannerList.push(item);  
 // 更新UI  
 this.notifyDataAdd(this.bannerList.length - 1);  
 }  
  
 // 更新所有数据  
 setList(list: IBannerList) {  
 this.bannerList = list;  
 this.notifyDataReload();  
 }  
  
 // 该方法为框架侧调用，为LazyForEach组件向其数据源处添加listener监听  
 registerDataChangeListener(listener: DataChangeListener): void {  
 if (this.listeners.indexOf(listener) < 0) {  
 console.info('add listener');  
 this.listeners.push(listener);  
 }  
 }  
  
 // 该方法为框架侧调用，为对应的LazyForEach组件在数据源处去除listener监听  
 unregisterDataChangeListener(listener: DataChangeListener): void {  
 const pos = this.listeners.indexOf(listener);  
 if (pos >= 0) {  
 console.info('remove listener');  
 this.listeners.splice(pos, 1);  
 }  
 }  
  
 // 通知LazyForEach组件需要重载所有子组件  
 notifyDataReload(): void {  
 this.listeners.forEach(listener => {  
 listener.onDataReloaded();  
 })  
 }  
  
 // 通知LazyForEach组件需要在index对应索引处添加子组件  
 notifyDataAdd(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataAdd(index);  
 })  
 }  
  
 // 通知LazyForEach组件在index对应索引处数据有变化，需要重建该子组件  
 notifyDataChange(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataChange(index);  
 })  
 }  
  
 // 通知LazyForEach组件需要在index对应索引处删除该子组件  
 notifyDataDelete(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataDelete(index);  
 })  
 }  
  
 // 通知LazyForEach组件将from索引和to索引处的子组件进行交换  
 notifyDataMove(from: number, to: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataMove(from, to);  
 })  
 }  
}  
  
  
export interface IHomeData {  
 "bannerList": IBannerList  
 "navList": INavList  
 "tileList": ITileList  
 "planList": IPlanList  
 "adPicture": string  
}  
  
export interface IBannerItem {  
 "id": number  
 "name": string  
 "imageURL": string  
}  
  
export type IBannerList = IBannerItem[]  
  
export interface INavItem {  
 "id": number  
 "title": string  
 "imageURL": string  
}  
  
export type INavList = INavItem[]  
  
export interface ITileItem {  
 "id": number  
 "imageURL": string  
 "title": string  
 "sub\_title": string  
}  
  
export type ITileList = ITileItem[]  
  
export interface IPlanItem {  
 "id": number  
 "imageURL": string  
}  
  
export type IPlanList = IPlanItem[]

import { BannerListDataSource, IBannerItem } from '../../models/HomeData';  
  
@Component  
export default struct SwiperLayout {  
 @ObjectLink bannerList: BannerListDataSource;  
  
 build() {  
 Swiper() {  
 LazyForEach(this.bannerList, (banner: IBannerItem) => {  
 Image(banner.imageURL).width('100%').height('100%').objectFit(ImageFit.Fill)  
 }, (banner: IBannerItem) => banner.id + '')  
 }  
 .width('100%')  
 .height(211 - 36) // 暂时减去状态栏高度  
 .autoPlay(true) // 自动轮播  
 .indicator( // 设置圆点导航点样式  
 new DotIndicator()  
 .color($r('app.color.indicator\_color'))  
 .selectedColor($r('app.color.indicator\_active\_color'))  
 )  
 }  
}

## 5.3. SearchBar 组件功能

1. 下载切图到 **resource > base > media** 目录中

[📎search.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512740817-e8a88fca-0fe6-42bc-a942-07a83719f6c6.svg)

[📎position.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512741048-ced336a4-f3ea-47d6-bc48-39e2fbcac7d8.svg)

[📎message.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512741165-8e50acb7-8ca2-4926-9ef9-a705b10494d5.svg)

1. 搭建 **views > Home > SearchBar** 组件

import { PADDING } from '../../constants/size'  
  
@Component  
export default struct SearchBar {  
 build() {  
 Row() {  
 Text("北京").fontSize(14).fontColor($r('app.color.white'))  
 Row() {  
 Image($r("app.media.search")).imageStyles(18).margin({ right: 10 })  
 // layoutWeight(1) 占剩下的空间的1份（占据剩下空间）  
 TextInput({ placeholder: '公司/地铁/小区，马上搜索' })  
 .layoutWeight(1)  
 .height('100%')  
 .fontSize(12)  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: 12 })  
 .caretColor($r('app.color.gray'))// 光标颜色  
 .padding(0)  
 .backgroundColor($r('app.color.white'))  
 .stateStyles({  
 // 状态样式  
 pressed: textInputStyles // 按下状态的样式  
 })  
  
 Column() {  
 }.width(1).height(18).backgroundColor('#D7D7D7').margin({ left: 10 })  
  
 Image($r("app.media.position")).imageStyles(18).margin({ left: 16 })  
 }  
 .width(244)  
 .height(38)  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: PADDING, right: PADDING })  
 .border({  
 width: 1,  
 radius: 52,  
 color: '#D7D7D7'  
 })  
  
 // 图片需要通过 fillColor 修改颜色，只有 svg 图片能用  
 Image($r("app.media.message")).imageStyles(24).fillColor($r('app.color.white'))  
 }  
 .width('100%')  
 .padding({  
 left: PADDING,  
 right: PADDING,  
 top: 4,  
 bottom: 4  
 })  
 .justifyContent(FlexAlign.SpaceBetween)  
 }  
}  
  
@Extend(Image)  
function imageStyles(size: number) {  
 .width(size).height(size).objectFit(ImageFit.Fill)  
}  
  
@Styles  
function textInputStyles() {  
 .backgroundColor('#fff')  
}

1. **Home** 使用 **SearchBar** 组件

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
  
import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource, INavList, IPlanList, ITileList } from '../model/HomeData';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: BannerListDataSource = new BannerListDataSource([]);  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = new BannerListDataSource(homeData.bannerList);  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 SwiperLayout({ bannerList: this.bannerList })  
 }  
 .height('100%')  
 .align(Alignment.TopStart)  
  
 SearchBar()  
 }  
 .height('100%')  
 .alignContent(Alignment.TopStart)  
 }  
}

## 5.4. NavList 组件功能

### 5.4.1. 父子组件通信

因为请求已经在 Home 父组件发送了，所以直接通信数据过来直接使用即可

1. 定义基本子组件 **views > Home > NavList**

@Component  
export default struct NavList {  
 build() {  
 Column() {  
 Text("NavList");  
 }  
 }  
}

1. **Home** 父组件通信 **navList** 给 **NavList** 子组件

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
import NavList from '../views/Home/NavList';  
  
import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource, INavList, IPlanList, ITileList } from '../model/HomeData';  
import { PADDING } from '../constants/size';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: BannerListDataSource = new BannerListDataSource([]);  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = new BannerListDataSource(homeData.bannerList);  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 SwiperLayout({ bannerList: this.bannerList })  
 Column() {  
 NavList({ navList: this.navList })  
 }.padding({ left: PADDING, right: PADDING })  
 }  
 .justifyContent(FlexAlign.Start)  
 }  
 .height('100%')  
 .align(Alignment.TopStart)  
  
 SearchBar()  
 }  
 .height('100%')  
 .alignContent(Alignment.TopStart)  
 }  
}

1. **NavList** 子组件声明接受 **navList** 数据

import type { INavList } from '../../model/HomeData'  
  
@Component  
export default struct NavList {  
 @Link navList: INavList  
  
 build() {  
 Column() {  
 Text("NavList");  
 }  
 }  
}

### 5.4.2. 搭建组件布局

import type { INavItem, INavList } from '../../model/HomeData'  
  
@Component  
export default struct NavList {  
 @Link navList: INavList  
  
 build() {  
 Grid() {  
 ForEach(this.navList, (nav: INavItem) => {  
 GridItem() {  
 Column({ space: 8 }) {  
 Image(nav.imageURL).width('100%').height(56).objectFit(ImageFit.Fill)  
 Text(nav.title).fontSize(12)  
 }  
 }.width(58)  
 }, (nav: INavItem) => nav.id + '')  
 }  
 .columnsTemplate('1fr 1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr')  
 .rowsGap(14)  
 .columnsGap(33)  
 .height(170)  
 .margin({  
 top: 24  
 })  
 }  
}

## 5.5. TileList 组件功能

### 5.5.1. 父子组件通信

因为请求已经在 Home 父组件发送了，所以直接通信数据过来直接使用即可

1. 定义基本子组件 **views > Home > TileList**

@Component  
export default struct TileList {  
 build() {  
 Column() {  
 Text("TileList");  
 }  
 }  
}

1. **Home** 父组件通信 **tileList** 给 **TileList** 子组件

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
import NavList from '../views/Home/NavList';  
import TileList from '../views/Home/TileList';  
  
import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource, INavList, IPlanList, ITileList } from '../model/HomeData';  
import { PADDING } from '../constants/size';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: BannerListDataSource = new BannerListDataSource([]);  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = new BannerListDataSource(homeData.bannerList);  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 SwiperLayout({ bannerList: this.bannerList })  
 Column() {  
 NavList({ navList: this.navList })  
 TileList({ tileList: this.tileList })  
 }.padding({ left: PADDING, right: PADDING })  
 }  
 .justifyContent(FlexAlign.Start)  
 }  
 .height('100%')  
 .align(Alignment.TopStart)  
  
 SearchBar()  
 }  
 .height('100%')  
 .alignContent(Alignment.TopStart)  
 }  
}

1. **TileList** 子组件声明接受 **tileList** 数据

import type { ITileList } from '../../model/HomeData'  
  
@Component  
export default struct TileList {  
 @Prop tileList: ITileList  
  
 build() {  
 Column() {  
 Text("TileList");  
 }  
 }  
}

### 5.5.2. 搭建组件布局

import type { ITileItem, ITileList } from '../../model/HomeData';  
import { BORDER\_RADIUS } from '../../constants/size';  
  
@Component  
export default struct TileList {  
 @Prop tileList: ITileList;  
   
 build() {  
 Row() {  
 ForEach(this.tileList, (tile: ITileItem) => {  
 Column({ space: 8 }) {  
 Image(tile.imageURL).width('100%').height(57).borderRadius(BORDER\_RADIUS)  
 Row({ space: 5 }) {  
 Text(tile.title).fontSize(12)  
 Text(tile.sub\_title).fontSize(10).fontColor($r('app.color.gray'))  
 }.justifyContent(FlexAlign.Start).width('100%')  
 }.width(148)  
 }, (tile: ITileItem) => tile.id + '')  
 }.width('100%').height(79).margin({ top: 10 }).justifyContent(FlexAlign.SpaceBetween)  
 }  
}

## 5.6. PlanList 组件功能

### 5.6.1. 父子组件通信

因为请求已经在 Home 父组件发送了，所以直接通信数据过来直接使用即可

1. 定义基本子组件 **views > Home > PlanList**

@Component  
export default struct PlanList {  
 build() {  
 Column() {  
 Text("PlanList");  
 }  
 }  
}

1. **Home** 父组件通信 **planList** 给 **PlanList** 子组件

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
import NavList from '../views/Home/NavList';  
import TileList from '../views/Home/TileList';  
import PlanList from '../views/Home/PlanList';  
  
import { getHomeData } from '../api/home'  
import type { IBannerList, INavList, ITileList, IPlanList } from '../model/HomeData'  
  
import { PADDING } from '../constants/size'  
  
@Component  
export default struct Home {  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 // 获取首页数据功能  
 @State bannerList: IBannerList = [];  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
  
 async getHomeData() {  
 const homeData = await getHomeData()  
 this.bannerList = homeData.bannerList;  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 SwiperLayout({ bannerList: $bannerList })  
 Column() {  
 NavList({ navList: $navList })  
 TileList({ tileList: $tileList })  
 PlanList({ planList: $planList })  
 }.padding({ left: PADDING, right: PADDING }).backgroundColor($r('app.color.white'))  
 }  
 }  
 .width('100%')  
 .height('100%')  
 .align(Alignment.TopStart)  
  
 SearchBar();  
 }  
 .width('100%')  
 .height('100%')  
 .alignContent(Alignment.TopStart)  
 }  
}

1. **PlanList** 子组件声明接受 **planList** 数据

import type { IPlanList } from '../../model/HomeData'  
  
@Component  
export default struct PlanList {  
 @Prop planList: IPlanList  
  
 build() {  
 Column() {  
 Text("PlanList");  
 }  
 }  
}

### 5.6.2. 搭建组件布局

import type { IPlanItem, IPlanList } from '../../model/HomeData';  
  
@Component  
export default struct PlanList {  
 @Prop planList: IPlanList;  
  
 build() {  
 Row() {  
 ForEach(this.planList, (plan: IPlanItem) => {  
 Image(plan.imageURL).width(78).height(60).objectFit(ImageFit.Fill)  
 }, (plan: IPlanItem) => plan.id + '')  
 }.width('100%').margin({ top: 18 }).justifyContent(FlexAlign.SpaceBetween)  
 }  
}

## 5.7. Ad 组件功能

### 5.7.1. 父子组件通信

因为请求已经在 Home 父组件发送了，所以直接通信数据过来直接使用即可

1. 定义基本子组件 **views > Home > Ad**

@Component  
export default struct Ad {  
 build() {  
 Column() {  
 Text("Ad");  
 }  
 }  
}

1. **Home** 父组件通信 **adPicture** 给 **Ad** 子组件

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
import NavList from '../views/Home/NavList';  
import TileList from '../views/Home/TileList';  
import PlanList from '../views/Home/PlanList';  
import Ad from '../views/Home/Ad';  
  
import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource, INavList, IPlanList, ITileList } from '../model/HomeData';  
import { PADDING } from '../constants/size';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: BannerListDataSource = new BannerListDataSource([]);  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = new BannerListDataSource(homeData.bannerList);  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 SwiperLayout({ bannerList: this.bannerList })  
 Column() {  
 NavList({ navList: this.navList })  
 TileList({ tileList: this.tileList })  
 PlanList({ planList: this.planList })  
 Ad({ adPicture: this.adPicture })  
 }.padding({ left: PADDING, right: PADDING })  
 }  
 .justifyContent(FlexAlign.Start)  
 }  
 .height('100%')  
 .align(Alignment.TopStart)  
  
 SearchBar()  
 }  
 .height('100%')  
 .alignContent(Alignment.TopStart)  
 }  
}

1. **Ad** 子组件声明接受 **adPicture** 数据

@Component  
export default struct Ad {  
 @Prop adPicture: string;  
   
 build() {  
 Column() {  
 Text("Ad");  
 }  
 }  
}

### 5.7.2. 搭建组件布局

import { SHADOW\_RADIUS } from '../../constants/size';  
  
@Component  
export default struct Ad {  
 @Prop adPicture: string;  
  
 build() {  
 Image(this.adPicture)  
 .width('100%')  
 .height(61)  
 .objectFit(ImageFit.Fill)  
 .margin({ top: 9 })  
 .shadow({  
 radius: SHADOW\_RADIUS,  
 color: $r('app.color.shadow'),  
 offsetX: 0,  
 offsetY: 0  
 })  
 }  
}

## 5.8. RoomRecommend 组件功能

### 5.8.1. 发送请求获取数据

1. 定义类型文件 **model > RoomRecommendDataSource**

export interface ITagItem {  
 "name": string  
}  
  
export type ITagList = ITagItem[]  
  
  
export interface IRoomRecommendItem {  
 "id": string  
 "housePicture": string  
 "tags": ITagList  
 "houseTitle": string  
 "address": string  
 "rentArea": string  
 "rentPriceUnit": string  
 "rentPriceListing": string  
}  
  
  
export type IRoomRecommendList = IRoomRecommendItem[]  
  
  
export class RoomRecommendDataSource implements IDataSource {  
 roomRecommendList: IRoomRecommendList = []  
  
 constructor(roomRecommendList: IRoomRecommendList) {  
 this.roomRecommendList = roomRecommendList;  
 }  
  
 getData(index: number) {  
 return this.roomRecommendList[index];  
 }  
  
 totalCount() {  
 return this.roomRecommendList.length;  
 }  
  
 registerDataChangeListener() {  
 }  
  
 unregisterDataChangeListener() {  
 }  
}

1. 定义接口函数 **api > home**

import { http } from '../utils/http';  
import type { IHomeData } from '../model/HomeData';  
import type { IRoomRecommendList } from '../model/RoomRecommendDataSource';  
  
// 获取首页数据  
export const getHomeDataApi = (): Promise<IHomeData> => {  
 return http.get<IHomeData>('/home/info');  
}  
  
// 获取首页周边房源数据  
export const getRoomRecommendListApi = (): Promise<IRoomRecommendList> => {  
 return http.get<IRoomRecommendList>('/house/nearbyHouses')  
}

1. 定义并使用组件，发送请求更新数据 **views > RoomRecommend**

import { RoomRecommendDataSource } from '../../model/RoomRecommendDataSource';  
import { getRoomRecommendListApi } from '../../api/home';  
  
@Component  
export default struct RoomRecommend {  
 // 获取推荐房源列表  
 @State roomRecommendDataSource: RoomRecommendDataSource = new RoomRecommendDataSource([])  
  
 aboutToAppear() {  
 this.getRoomRecommendList();  
 }  
  
 async getRoomRecommendList() {  
 const roomRecommendList = await getRoomRecommendListApi();  
 this.roomRecommendDataSource = new RoomRecommendDataSource(roomRecommendList)  
 }  
  
 build() {  
 Column() {  
 Text('RoomRecommend')  
 }.width('100%')  
 }  
}

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
import NavList from '../views/Home/NavList';  
import TileList from '../views/Home/TileList';  
import PlanList from '../views/Home/PlanList';  
import Ad from '../views/Home/Ad';  
import RoomRecommend from '../views/Home/RoomRecommend';  
  
import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource, INavList, IPlanList, ITileList } from '../model/HomeData';  
import { PADDING } from '../constants/size';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: BannerListDataSource = new BannerListDataSource([]);  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = new BannerListDataSource(homeData.bannerList);  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 SwiperLayout({ bannerList: this.bannerList })  
   
 Column() {  
 NavList({ navList: this.navList })  
 TileList({ tileList: this.tileList })  
 PlanList({ planList: this.planList })  
 Ad({ adPicture: this.adPicture })  
 }.padding({ left: PADDING, right: PADDING })  
  
 RoomRecommend()  
 }  
 .justifyContent(FlexAlign.Start)  
 }  
 .height('100%')  
 .align(Alignment.TopStart)  
 .scrollBar(BarState.Off)  
  
 SearchBar()  
 }  
 .height('100%')  
 .alignContent(Alignment.TopStart)  
 }  
}

### 5.8.2. 搭建组件布局

1. 下载切图到 **resource > base > media** 目录中

[📎arrow\_right.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512741299-837eccc1-1cec-42c5-8ea2-6b1605e93086.svg)

[📎location.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512741557-6aff6403-c1d4-4179-9b82-c459ce23752a.svg)

[📎room\_text\_bg.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512741772-c69f7e22-dd9e-415e-a13c-2cd5db9151b9.png)

1. 搭建 **views > Home > RoomRecommend** 组件

import { IRoomRecommendItem, ITagItem, RoomRecommendDataSource } from '../../model/RoomRecommendDataSource'  
import { getRoomRecommendListApi } from '../../api/home'  
  
import { BORDER\_RADIUS, PADDING } from '../../constants/size'  
  
@Component  
export default struct RoomRecommend {  
 // 获取推荐房源列表  
 @State roomRecommendDataSource: RoomRecommendDataSource = new RoomRecommendDataSource([])  
  
 aboutToAppear() {  
 this.getRoomRecommendList();  
 }  
  
 async getRoomRecommendList() {  
 const roomRecommendList = await getRoomRecommendListApi();  
 this.roomRecommendDataSource = new RoomRecommendDataSource(roomRecommendList)  
 }  
  
 // 头部导航  
 @Builder  
 RoomNav() {  
 Row() {  
 Text() {  
 Span('周边房源')  
 Span('推荐').fontColor($r('app.color.primary'))  
 }.fontSize(18)  
  
 Row({ space: 6 }) {  
 Text('更多推荐').fontColor($r('app.color.gray\_second')).fontSize(12)  
 Image($r('app.media.arrow\_right')).width(4).height(8)  
 }  
 }  
 .width('100%')  
 .height(67)  
 .justifyContent(FlexAlign.SpaceBetween)  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: PADDING, right: PADDING, top: 23 })  
 }  
  
 // 房源列表  
 @Builder  
 RoomList() {  
 if (this.roomRecommendDataSource.roomRecommendList.length) {  
 Grid() {  
 LazyForEach(this.roomRecommendDataSource, (room: IRoomRecommendItem) => {  
 GridItem() {  
 Stack() {  
 Column() {  
 Image(room.housePicture).width('100%').height(120).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: BORDER\_RADIUS,  
 topRight: BORDER\_RADIUS  
 })  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(16).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(14).fontColor('#E03810')  
 }.height(22)  
  
 Text(room.rentArea + '㎡').fontSize(12).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text(room.houseTitle)  
 .fontSize(14)  
 .margin({ top: 1 })  
 .height(19)  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: 6 }) {  
 ForEach(room.tags.slice(0, 2), (tag: ITagItem, index) => {  
 Text(tag.name)  
 .padding({  
 top: 1,  
 bottom: 1,  
 left: 6,  
 right: 6  
 })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(12)  
 }, (tag: ITagItem) => tag.name)  
 }.width('100%').margin({ top: 9 })  
 }.padding(12).alignItems(HorizontalAlign.Start).height(91)  
 }.width('100%')  
  
 Image($r('app.media.room\_text\_bg')).width('100%').height(120)  
 Row({ space: 6 }) {  
 Image($r('app.media.location')).width(12).height(14)  
 Text(room.address).fontSize(12).fontColor($r('app.color.white'))  
 }.margin({ left: 8, top: 99 })  
 }.alignContent(Alignment.TopStart).backgroundColor($r('app.color.white'))  
 }.height(210)  
 }, (room: IRoomRecommendItem) => room.id)  
 }  
 .columnsTemplate('1fr 1fr')  
 .columnsGap(8)  
 .rowsGap(8)  
 .width('100%')  
 .padding({  
 left: PADDING,  
 right: PADDING,  
 top: 8,  
 bottom: 8  
 })  
 .backgroundColor($r('app.color.bg\_gray'))  
 .borderRadius(BORDER\_RADIUS)  
 }  
 }  
  
 build() {  
 Column() {  
 this.RoomNav()  
 this.RoomList()  
 }.width('100%')  
 }  
}

## 5.9. 滚动时改变头部导航背景色和颜色

实现思路：

1. 计算 scrollY 滚动的距离

1. 根据 scrollY 滚动的距离来更新背景色和文字颜色

* 当距离小于等于 10 时，背景色是透明的，文字颜色是白色

* 当距离大于 10 时，背景色是白色，文字颜色是黑色

为什么是 10 呢？经过打印 scrollY 发现，滚动距离到达顶部可能不一定是 0，可能是比较小的正数或负数。所以给一点误差空间。

1. **Home** 组件计算 **scrollY** 滚动的距离和颜色，传递给 **SearchBar** 组件使用

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
import NavList from '../views/Home/NavList';  
import TileList from '../views/Home/TileList';  
import PlanList from '../views/Home/PlanList';  
import Ad from '../views/Home/Ad';  
import RoomRecommend from '../views/Home/RoomRecommend';  
  
import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource, INavList, IPlanList, ITileList } from '../model/HomeData';  
import { PADDING } from '../constants/size';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: BannerListDataSource = new BannerListDataSource([]);  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
 scrollY: number = 0;  
 @State bgColor: string = 'rgba(255, 255, 255, 0)'  
 @State fontColor: string = 'rgb(255, 255, 255)'  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = new BannerListDataSource(homeData.bannerList);  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 // 处理滚动  
 handleScroll = (xOffset: number, yOffset: number) => {  
 this.scrollY += yOffset;  
 if (this.scrollY < 0) {  
 this.scrollY = 0;  
 }  
 this.calcColor()  
 }  
 // 计算颜色  
 calcColor = () => {  
 // 滚动计算的 scrollY 的值到达顶部可能不是 0，给点误差  
 if (this.scrollY > 10) {  
 this.bgColor = 'rgba(255, 255, 255, 1)'  
 this.fontColor = 'rgb(0, 0, 0)'  
 } else {  
 this.bgColor = 'rgba(255, 255, 255, 0)'  
 this.fontColor = 'rgb(255, 255, 255)'  
 }  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 SwiperLayout({ bannerList: this.bannerList })  
  
 Column() {  
 NavList({ navList: this.navList })  
 TileList({ tileList: this.tileList })  
 PlanList({ planList: this.planList })  
 Ad({ adPicture: this.adPicture })  
 }.padding({ left: PADDING, right: PADDING })  
  
 RoomRecommend()  
 }  
 .justifyContent(FlexAlign.Start)  
 }  
 .height('100%')  
 .align(Alignment.TopStart)  
 .scrollBar(BarState.Off)  
 .onDidScroll(this.handleScroll)  
  
 SearchBar({ bgColor: this.bgColor, fontColor: this.fontColor })  
 }  
 .height('100%')  
 .alignContent(Alignment.TopStart)  
 }  
}

1. **SearchBar** 组件声明接受颜色并使用

import { PADDING } from '../../constants/size';  
  
@Component  
export default struct SearchBar {  
 @Prop bgColor: string;  
 @Prop fontColor: string;  
  
 build() {  
 Row() {  
 Text("北京").fontSize(14).fontColor(this.fontColor)  
 Row() {  
 Image($r("app.media.search")).imageStyles(18).margin({ right: 10 })  
 // layoutWeight(1) 占剩下的空间的1份（占据剩下空间）  
 TextInput({ placeholder: '公司/地铁/小区，马上搜索' })  
 .layoutWeight(1)  
 .height('100%')  
 .fontSize(12)  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: 12 })  
 .caretColor($r('app.color.gray'))// 光标颜色  
 .padding(0)  
 .backgroundColor($r('app.color.white'))  
 .stateStyles({  
 // 状态样式  
 pressed: textInputStyles // 按下状态的样式  
 })  
  
 Column() {  
 }.width(1).height(18).backgroundColor('#D7D7D7').margin({ left: 10 })  
  
 Image($r("app.media.position")).imageStyles(18).margin({ left: 16 })  
 }  
 .width(244)  
 .height(38)  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: PADDING, right: PADDING })  
 .border({  
 width: 1,  
 radius: 52,  
 color: '#D7D7D7'  
 })  
  
 // 图片需要通过 fillColor 修改颜色，只有 svg 图片能用  
 Image($r("app.media.message")).imageStyles(24).fillColor(this.fontColor)  
 }  
 .width('100%')  
 .padding({  
 left: PADDING,  
 right: PADDING,  
 top: 4,  
 bottom: 4  
 })  
 .justifyContent(FlexAlign.SpaceBetween)  
 .backgroundColor(this.bgColor)  
 }  
}  
  
/\*  
 @Styles: 用来定义复用样式  
 @Extend: 用来扩展组件样式（只有这个组件能用）  
  
 @Styles 和 @Extend 都可以用来复用样式，有什么区别：  
 1. @Styles 只支持通用属性和通用事件。@Extend 即通用属性和通用事件，也支持定义组件私有属性和私有事件  
 2. @Styles 不支持传参。@Extend 支持传参（更灵活）  
 3. @Styles 可以定义组件内或全局。@Extend 只支持  
  
 注意：全局指的是当前文件内生效。不是真正的全局  
 \*/  
  
@Extend(Image)  
function imageStyles(size: number) {  
 .width(size).height(size).objectFit(ImageFit.Fill)  
}  
  
@Styles  
function textInputStyles() {  
 .backgroundColor('#fff')  
}

## 5.10. 封装 ScrollContainer 公共组件

1. 封装基本 **components >** **ScrollContainer** 组件

import { IColor } from '../model/HomeData';  
  
@Component  
export default struct ScrollContainer {  
 @BuilderParam contentBuilder: () => void = this.defaultBuilder  
 @BuilderParam navBuilder: ($$: IColor) => void = this.defaultBuilder  
 scrollY: number = 0;  
 @State bgColor: string = 'rgba(255, 255, 255, 0)'  
 @State fontColor: string = 'rgb(255, 255, 255)'  
  
 @Builder  
 defaultBuilder() {  
 }  
  
 // 处理滚动  
 handleScroll = (scrollX: number, scrollY: number) => {  
 // 更新并修正 scrollY 的值  
 this.scrollY += scrollY;  
 if (this.scrollY < 0) {  
 this.scrollY = 0  
 }  
 this.calcColor();  
 }  
 // 计算颜色  
 calcColor = () => {  
 if (this.scrollY > 10) {  
 this.bgColor = 'rgba(255, 255, 255, 1)'  
 this.fontColor = 'rgb(0, 0, 0)'  
 } else {  
 this.bgColor = 'rgba(255, 255, 255, 0)'  
 this.fontColor = 'rgb(255, 255, 255)'  
 }  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 // 渲染内容  
 this.contentBuilder()  
 }.align(Alignment.TopStart)  
 }  
 .height('100%')  
 .scrollBar(BarState.Off)  
 .align(Alignment.TopStart)  
 .backgroundColor($r('app.color.bg\_gray\_second'))  
 .padding({ bottom: 10 })  
 .onDidScroll(this.handleScroll)  
  
 Column() {  
 // 渲染头部导航  
 this.navBuilder({ fontColor: this.fontColor, bgColor: this.bgColor })  
 }.backgroundColor(this.bgColor)  
  
 }.height('100%').alignContent(Alignment.TopStart)  
 }  
}

1. **Home** 组件使用 **ScrollContainer** 组件

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
import NavList from '../views/Home/NavList';  
import TileList from '../views/Home/TileList';  
import PlanList from '../views/Home/PlanList';  
import Ad from '../views/Home/Ad';  
import RoomRecommend from '../views/Home/RoomRecommend';  
import ScrollContainer from '../components/ScrollContainer';  
  
import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource, IColor, INavList, IPlanList, ITileList } from '../model/HomeData';  
import { PADDING } from '../constants/size';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: BannerListDataSource = new BannerListDataSource([]);  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
 scrollY: number = 0;  
 @State bgColor: string = 'rgba(255, 255, 255, 0)'  
 @State fontColor: string = 'rgb(255, 255, 255)'  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = new BannerListDataSource(homeData.bannerList);  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 // 处理滚动  
 handleScroll = (xOffset: number, yOffset: number) => {  
 this.scrollY += yOffset;  
 if (this.scrollY < 0) {  
 this.scrollY = 0;  
 }  
 this.calcColor()  
 }  
 // 计算颜色  
 calcColor = () => {  
 // 滚动计算的 scrollY 的值到达顶部可能不是 0，给点误差  
 if (this.scrollY > 10) {  
 this.bgColor = 'rgba(255, 255, 255, 1)'  
 this.fontColor = 'rgb(0, 0, 0)'  
 } else {  
 this.bgColor = 'rgba(255, 255, 255, 0)'  
 this.fontColor = 'rgb(255, 255, 255)'  
 }  
 }  
  
 @Builder  
 contentBuilder() {  
 SwiperLayout({ bannerList: this.bannerList })  
 Column() {  
 NavList({ navList: this.navList })  
 TileList({ tileList: this.tileList })  
 PlanList({ planList: this.planList })  
 Ad({ adPicture: this.adPicture })  
 }.padding({ left: PADDING, right: PADDING })  
  
 RoomRecommend()  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 SearchBar({ bgColor: $$.bgColor, fontColor: $$.fontColor });  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder  
 })  
 }  
}

1. 定义 **model > IColor** 类型

export interface IHomeData {  
 "bannerList": IBannerList  
 "navList": INavList  
 "tileList": ITileList  
 "planList": IPlanList  
 "adPicture": string  
}  
  
// 计划列表类型  
export interface IPlanItem {  
 "id": number  
 "imageURL": string  
}  
  
export type IPlanList = IPlanItem[]  
  
// 瓷片列表类型  
export interface ITileItem {  
 "id": number  
 "imageURL": string  
 "title": string  
 "sub\_title": string  
}  
  
export type ITileList = ITileItem[]  
  
// 导航列表类型  
export interface INavItem {  
 "id": number  
 "title": string  
 "imageURL": string  
}  
  
export type INavList = INavItem[]  
  
// 轮播图类型  
export interface IBannerItem {  
 "id": number  
 "name": string  
 "imageURL": string  
}  
  
export type IBannerList = IBannerItem[]  
  
export class BannerListDataSource implements IDataSource {  
 private list: IBannerList = []  
  
 constructor(list: IBannerList) {  
 this.list = list  
 }  
  
 totalCount(): number {  
 return this.list.length  
 }  
  
 getData(index: number): IBannerItem {  
 return this.list[index]  
 }  
  
 registerDataChangeListener(listener: DataChangeListener): void {  
 }  
  
 unregisterDataChangeListener() {  
 }  
}  
  
export interface IColor {  
 fontColor: string;  
 bgColor: string;  
}

# 6. 移动端适配

## 6.1. 问题：请求和图片访问失败

手机模拟器运行代码，发现页面报错，请求失败：

问题原因：在模拟器或真机设备发送请求或访问图片，都需要开启网络访问权限才能使用

解决方案：开启网络权访问限

{  
 "module": {  
 // ...  
 // 前面内容省略  
 "requestPermissions":[  
 {  
 "name" : "ohos.permission.INTERNET",  
 }  
 ]  
 }  
}

## 6.2. 手机端适配

### 6.2.1. 分析问题

手机模拟器访问页面发现存在几个问题：

1. 头部状态栏显示没有全覆盖

1. **TileList** 留白太多



产生原因：

1. 第 1 问题原因是：默认应用不会全屏展示，所以状态栏没有渲染内容

1. 第 2 问题原因是：**模拟器设备** 比 **previewer 预览器** 更宽，而我们元素是设置固定大小，这样留白自然就大了

解决方案：

1. 需要设置应用全屏展示

1. 需要根据设备宽度，将元素大小转化为合适大小，让元素在不同设备也能等比例呈现

### 6.2.2. 解决问题1

1. 在 **entry > src > main >** **ets > entryability > EntryAbility.ts** 完成以下功能：

* 设置应用全屏展示

* 设置状态栏颜色

* 获取并缓存状态栏高度

import { AbilityConstant, UIAbility, Want } from '@kit.AbilityKit';  
import { hilog } from '@kit.PerformanceAnalysisKit';  
import { window } from '@kit.ArkUI';  
  
export default class EntryAbility extends UIAbility {  
 onCreate(want: Want, launchParam: AbilityConstant.LaunchParam): void {  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onCreate');  
 }  
  
 onDestroy(): void {  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onDestroy');  
 }  
  
 async onWindowStageCreate(windowStage: window.WindowStage) {  
 // Main window is created, set main page for this ability  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onWindowStageCreate');  
  
 windowStage.loadContent('pages/Index', (err) => {  
 if (err.code) {  
 hilog.error(0x0000, 'testTag', 'Failed to load the content. Cause: %{public}s', JSON.stringify(err) ?? '');  
 return;  
 }  
 hilog.info(0x0000, 'testTag', 'Succeeded in loading the content.');  
 });  
  
 const windowClass = await windowStage.getMainWindow();  
 // 1. 设置全屏  
 windowClass.setWindowLayoutFullScreen(true);  
 // 2. 设置状态栏颜色  
 windowClass.setWindowSystemBarProperties({  
 statusBarContentColor: '#FFFFFF',  
 navigationBarContentColor: '#FFFFFF'  
 })  
 // 3. 获取状态栏高度并存储起来  
 const avoidArea = windowClass.getWindowAvoidArea(window.AvoidAreaType.TYPE\_SYSTEM);  
 AppStorage.setOrCreate('windowTopHeight', avoidArea.topRect.height);  
 AppStorage.setOrCreate('windowBottomHeight', avoidArea.bottomRect.height);  
 }  
  
 onWindowStageDestroy(): void {  
 // Main window is destroyed, release UI related resources  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onWindowStageDestroy');  
 }  
  
 onForeground(): void {  
 // Ability has brought to foreground  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onForeground');  
 }  
  
 onBackground(): void {  
 // Ability has back to background  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onBackground');  
 }  
}

此时页面可以通过 AppStorage 获取屏幕的参数：windowTopHeight 状态栏高度

1. 定义获取高度函数

需要注意的是：获取到的 windowTopHeight 单位是 px，而我们布局单位是 vp，需要使用 px2vp 函数转化使用

// 获取状态栏高度  
export const getWindowTopHeight = () => {  
 return px2vp(AppStorage.get('windowTopHeight') || 0)  
}  
  
// 获取底部高度  
export const getWindowBottomHeight = () => {  
 return px2vp(AppStorage.get('windowBottomHeight') || 0)  
}

1. **ScrollContainer** 设置状态栏高度

import { IColor } from '../model/HomeData';  
import { getWindowTopHeight } from '../utils/responsive';  
  
@Component  
export default struct ScrollContainer {  
 @BuilderParam contentBuilder: () => void = this.defaultBuilder  
 @BuilderParam navBuilder: ($$: IColor) => void = this.defaultBuilder  
 scrollY: number = 0;  
 @State bgColor: string = 'rgba(255, 255, 255, 0)'  
 @State fontColor: string = 'rgb(255, 255, 255)'  
  
 @Builder  
 defaultBuilder() {  
 }  
  
 // 处理滚动  
 handleScroll = (scrollX: number, scrollY: number) => {  
 // 更新并修正 scrollY 的值  
 this.scrollY += scrollY;  
 if (this.scrollY < 0) {  
 this.scrollY = 0  
 }  
 this.calcColor();  
 }  
 // 计算颜色  
 calcColor = () => {  
 if (this.scrollY > 10) {  
 this.bgColor = 'rgba(255, 255, 255, 1)'  
 this.fontColor = 'rgb(0, 0, 0)'  
 } else {  
 this.bgColor = 'rgba(255, 255, 255, 0)'  
 this.fontColor = 'rgb(255, 255, 255)'  
 }  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 // 渲染内容  
 this.contentBuilder()  
 }.align(Alignment.TopStart)  
 }  
 .height('100%')  
 .scrollBar(BarState.Off)  
 .align(Alignment.TopStart)  
 .backgroundColor($r('app.color.bg\_gray\_second'))  
 .padding({ bottom: 10 })  
 .onDidScroll(this.handleScroll)  
  
 Column() {  
 // 渲染头部导航  
 this.navBuilder({ fontColor: this.fontColor, bgColor: this.bgColor })  
 }.backgroundColor(this.bgColor).padding({ top: getWindowTopHeight() })  
  
 }.height('100%').alignContent(Alignment.TopStart)  
 }  
}

1. **SwiperLayout** 轮播图增加高度

import type { BannerListDataSource, IBannerItem } from '../../model/HomeData';  
  
@Component  
export default struct SwiperLayout {  
 @StorageProp('windowTopHeight') windowTopHeight: number = 0;  
 @Prop bannerList: BannerListDataSource  
  
 build() {  
 if (this.bannerList.totalCount()) {  
 Swiper() {  
 LazyForEach(this.bannerList, (banner: IBannerItem) => {  
 Image(banner.imageURL).width('100%').height(175 + px2vp(this.windowTopHeight))  
 }, (item: IBannerItem) => item.id + '')  
 }  
 .loop(true)  
 .indicator(  
 Indicator.dot()// 设置圆形导航点样式  
 .color('#CCCBCB')  
 .selectedColor($r('app.color.primary'))  
 )  
 }  
 }  
}

1. 设置底部高度

import Home from './Home';  
import See from './See';  
import Service from './Service';  
import Discover from './Discover';  
import My from './My';  
  
import { getWindowBottomHeight } from '../utils/responsive';  
  
@Entry  
 @Component  
 struct Index {  
 // 当前 tab 高亮的下标  
 @State currentIndex: number = 0;  
  
 // 生成 tab 的函数  
 @Builder  
 TabBuilder(index: number, text: string, icon: Resource, activeIcon: Resource) {  
 Column() {  
 // 项目中常用的颜色、大小等会定义成常量，将来方便复用和统一调整。而不常用的大小颜色就不需要定义了，直接写死即可。  
 Image(this.currentIndex === index ? activeIcon : icon).width(28)  
 Text(text)  
 .fontSize(10)  
 .fontColor(this.currentIndex === index ? $r('app.color.black') : '#a0a0a0')  
 .height(rvp(15))  
 }.width('100%').height('100%').justifyContent(FlexAlign.Center)  
 }  
  
 build() {  
 Column() {  
 Tabs({ barPosition: BarPosition.End, index: this.currentIndex }) {  
 TabContent() {  
 Home()  
 }.tabBar(this.TabBuilder(0, '首页', $r('app.media.tabbar\_home'), $r('app.media.tabbar\_home\_active')))  
  
 TabContent() {  
 See()  
 }.tabBar(this.TabBuilder(1, '想看', $r('app.media.tabbar\_see'), $r('app.media.tabbar\_see\_active')))  
  
 TabContent() {  
 Service()  
 }.tabBar(this.TabBuilder(2, '服务', $r('app.media.tabbar\_service'), $r('app.media.tabbar\_service\_active')))  
  
 TabContent() {  
 Discover()  
 }.tabBar(this.TabBuilder(3, '发现', $r('app.media.tabbar\_discover'), $r('app.media.tabbar\_discover\_active')))  
  
 TabContent() {  
 My()  
 }.tabBar(this.TabBuilder(4, '我的', $r('app.media.tabbar\_my'), $r('app.media.tabbar\_my\_active')))  
 }  
 .vertical(false)  
 .barMode(BarMode.Fixed)  
 .onChange((index: number) => {  
 this.currentIndex = index  
 })  
 .barHeight(50 + getWindowBottomHeight()) // 设置 tab 导航栏的高度，默认 56  
 .scrollable(false)  
 .width('100%')  
 .height('100%')  
 .backgroundColor($r('app.color.white'))  
 }.height('100%')  
 }  
 }

1. 滚动时修改状态栏颜色

import { window } from '@kit.ArkUI';  
import { IColor } from '../model/HomeData';  
import { getWindowTopHeight } from '../utils/responsive';  
  
@Component  
export default struct ScrollContainer {  
 @BuilderParam contentBuilder: () => void = this.defaultBuilder  
 @BuilderParam navBuilder: ($$: IColor) => void = this.defaultBuilder  
 scrollY: number = 0;  
 @State bgColor: string = 'rgba(255, 255, 255, 0)'  
 @State fontColor: string = 'rgb(255, 255, 255)'  
 // 设置 window 状态栏颜色  
 setWindowStatusColor = async () => {  
 const windowClass = await window.getLastWindow(getContext())  
 windowClass.setWindowSystemBarProperties({  
 statusBarContentColor: this.fontColor,  
 navigationBarContentColor: this.fontColor  
 })  
 }  
  
 @Builder  
 defaultBuilder() {  
 }  
  
 // 处理滚动  
 handleScroll = (scrollX: number, scrollY: number) => {  
 // 更新并修正 scrollY 的值  
 this.scrollY += scrollY;  
 if (this.scrollY < 0) {  
 this.scrollY = 0  
 }  
 this.calcColor();  
 this.setWindowStatusColor();  
 }  
 // 计算颜色  
 calcColor = () => {  
 if (this.scrollY > 10) {  
 this.bgColor = 'rgba(255, 255, 255, 1)'  
 this.fontColor = '#000000'  
 } else {  
 this.bgColor = 'rgba(255, 255, 255, 0)'  
 this.fontColor = '#ffffff'  
 }  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 // 渲染内容  
 this.contentBuilder()  
 }.align(Alignment.TopStart)  
 }  
 .height('100%')  
 .scrollBar(BarState.Off)  
 .align(Alignment.TopStart)  
 .backgroundColor($r('app.color.bg\_gray\_second'))  
 .padding({ bottom: 10 })  
 .onDidScroll(this.handleScroll)  
  
 Column() {  
 // 渲染头部导航  
 this.navBuilder({ fontColor: this.fontColor, bgColor: this.bgColor })  
 }.backgroundColor(this.bgColor).padding({ top: getWindowTopHeight() })  
  
 }.height('100%').alignContent(Alignment.TopStart)  
 }  
}

### 6.2.3. 解决问题2

1. 获取屏幕宽高并存储起来

import { AbilityConstant, UIAbility, Want } from '@kit.AbilityKit';  
import { hilog } from '@kit.PerformanceAnalysisKit';  
import { window } from '@kit.ArkUI';  
  
export default class EntryAbility extends UIAbility {  
 onCreate(want: Want, launchParam: AbilityConstant.LaunchParam): void {  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onCreate');  
 }  
  
 onDestroy(): void {  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onDestroy');  
 }  
  
 async onWindowStageCreate(windowStage: window.WindowStage) {  
 // Main window is created, set main page for this ability  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onWindowStageCreate');  
  
 windowStage.loadContent('pages/Index', (err) => {  
 if (err.code) {  
 hilog.error(0x0000, 'testTag', 'Failed to load the content. Cause: %{public}s', JSON.stringify(err) ?? '');  
 return;  
 }  
 hilog.info(0x0000, 'testTag', 'Succeeded in loading the content.');  
 });  
  
 const windowClass = await windowStage.getMainWindow();  
 // 1. 设置全屏  
 windowClass.setWindowLayoutFullScreen(true);  
 // 2. 设置状态栏颜色  
 windowClass.setWindowSystemBarProperties({  
 statusBarContentColor: '#FFFFFF',  
 navigationBarContentColor: '#FFFFFF'  
 })  
 // 3. 获取状态栏高度并存储起来  
 const avoidArea = windowClass.getWindowAvoidArea(window.AvoidAreaType.TYPE\_SYSTEM);  
 AppStorage.setOrCreate('windowTopHeight', avoidArea.topRect.height);  
 // 4. 获取屏幕宽高并存储起来  
 const properties = windowClass.getWindowProperties()  
 AppStorage.setOrCreate('windowHeight', properties.windowRect.height);  
 AppStorage.setOrCreate('windowWidth', properties.windowRect.width);  
 }  
  
 onWindowStageDestroy(): void {  
 // Main window is destroyed, release UI related resources  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onWindowStageDestroy');  
 }  
  
 onForeground(): void {  
 // Ability has brought to foreground  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onForeground');  
 }  
  
 onBackground(): void {  
 // Ability has back to background  
 hilog.info(0x0000, 'testTag', '%{public}s', 'Ability onBackground');  
 }  
}

1. 定义计算响应式单位的工具函数

// 获取状态栏高度  
export const getWindowTopHeight = () => {  
 return px2vp(AppStorage.get('windowTopHeight') || 0)  
}  
  
// 获取底部高度  
export const getWindowBottomHeight = () => {  
 return px2vp(AppStorage.get('windowBottomHeight') || 0)  
}  
  
// 获取屏幕宽度，进行缓存  
let windowWidth: number;  
  
export function getWindowWidth() {  
 if (windowWidth) {  
 return windowWidth;  
 }  
 // 获取到的屏幕宽度是px单位  
 const storageWindowWidth = AppStorage.get<number>('windowWidth');  
 if (storageWindowWidth) {  
 // 将屏幕宽度转化为vp单位  
 windowWidth = px2vp(storageWindowWidth)  
 } else {  
 // 预览器没法获取屏幕宽度，所以读取的到值为0，会进入这里，我们给默认值360  
 windowWidth = 360  
 // 预览器平板测试使用这个值  
 // windowWidth = 640  
 }  
 return windowWidth  
}  
  
// 将元素按照原来比例进行转化  
export function rvp(val: number) {  
 /\*  
 \* 设计稿总宽度：360 (固定值)  
 \* 设计稿某个元素大小：120 -> val  
 \* 实际设备屏幕宽度：400 -> 通过 window.getWindowProperties().windowRect.width 方式获取, 得到单位是 px，需要转 vp 计算  
 \* 元素实际大小：x  
 \* x / 400 = val / 360  
 \*  
 \* 解方程式得出: x = val / 360 \* 实际设备屏幕宽度  
 \*/  
 return val / 360 \* getWindowWidth()  
}

1. 改造所有代码，让元素大小都通过 **responsiveVP(rvp)** 函数进行转化

import Home from './Home';  
import See from './See';  
import Service from './Service';  
import Discover from './Discover';  
import My from './My';  
  
import { getWindowBottomHeight, rvp } from '../utils/responsive';  
  
@Entry  
@Component  
struct Index {  
 // 当前 tab 高亮的下标  
 @State currentIndex: number = 0;  
  
 // 生成 tab 的函数  
 @Builder  
 TabBuilder(index: number, text: string, icon: Resource, activeIcon: Resource) {  
 Column() {  
 // 项目中常用的颜色、大小等会定义成常量，将来方便复用和统一调整。而不常用的大小颜色就不需要定义了，直接写死即可。  
 Image(this.currentIndex === index ? activeIcon : icon).width(rvp(28))  
 Text(text)  
 .fontSize(rvp(10))  
 .fontColor(this.currentIndex === index ? $r('app.color.black') : '#a0a0a0')  
 .height(rvp(15))  
 }.width('100%').height('100%').justifyContent(FlexAlign.Center)  
 }  
  
 build() {  
 Column() {  
 Tabs({ barPosition: BarPosition.End, index: this.currentIndex }) {  
 TabContent() {  
 Home()  
 }.tabBar(this.TabBuilder(0, '首页', $r('app.media.tabbar\_home'), $r('app.media.tabbar\_home\_active')))  
  
 TabContent() {  
 See()  
 }.tabBar(this.TabBuilder(1, '想看', $r('app.media.tabbar\_see'), $r('app.media.tabbar\_see\_active')))  
  
 TabContent() {  
 Service()  
 }.tabBar(this.TabBuilder(2, '服务', $r('app.media.tabbar\_service'), $r('app.media.tabbar\_service\_active')))  
  
 TabContent() {  
 Discover()  
 }.tabBar(this.TabBuilder(3, '发现', $r('app.media.tabbar\_discover'), $r('app.media.tabbar\_discover\_active')))  
  
 TabContent() {  
 My()  
 }.tabBar(this.TabBuilder(4, '我的', $r('app.media.tabbar\_my'), $r('app.media.tabbar\_my\_active')))  
 }  
 .vertical(false)  
 .barMode(BarMode.Fixed)  
 .onChange((index: number) => {  
 this.currentIndex = index  
 })  
 .barHeight(rvp(50) + getWindowBottomHeight()) // 设置 tab 导航栏的高度，默认 56  
 .scrollable(false)  
 .width('100%')  
 .height('100%')  
 .backgroundColor($r('app.color.white'))  
 }.height('100%')  
 }  
}

import SwiperLayout from '../views/Home/SwiperLayout';  
import SearchBar from '../views/Home/SearchBar';  
import NavList from '../views/Home/NavList';  
import TileList from '../views/Home/TileList';  
import PlanList from '../views/Home/PlanList';  
import Ad from '../views/Home/Ad';  
import RoomRecommend from '../views/Home/RoomRecommend';  
import ScrollContainer from '../components/ScrollContainer';  
  
import { getHomeDataApi } from '../api/home';  
import { BannerListDataSource, IColor, INavList, IPlanList, ITileList } from '../model/HomeData';  
import { PADDING } from '../constants/size';  
import { rvp } from '../utils/responsive';  
  
@Component  
export default struct Home {  
 // 首页数据  
 @State bannerList: BannerListDataSource = new BannerListDataSource([]);  
 @State navList: INavList = [];  
 @State tileList: ITileList = [];  
 @State planList: IPlanList = [];  
 @State adPicture: string = '';  
 scrollY: number = 0;  
 @State bgColor: string = 'rgba(255, 255, 255, 0)'  
 @State fontColor: string = 'rgb(255, 255, 255)'  
  
 // 组件生命周期  
 aboutToAppear() {  
 this.getHomeData()  
 }  
  
 async getHomeData() {  
 const homeData = await getHomeDataApi()  
 this.bannerList = new BannerListDataSource(homeData.bannerList);  
 this.navList = homeData.navList;  
 this.tileList = homeData.tileList;  
 this.planList = homeData.planList;  
 this.adPicture = homeData.adPicture;  
 }  
  
 // 处理滚动  
 handleScroll = (xOffset: number, yOffset: number) => {  
 this.scrollY += yOffset;  
 if (this.scrollY < 0) {  
 this.scrollY = 0;  
 }  
 this.calcColor()  
 }  
 // 计算颜色  
 calcColor = () => {  
 // 滚动计算的 scrollY 的值到达顶部可能不是 0，给点误差  
 if (this.scrollY > 10) {  
 this.bgColor = 'rgba(255, 255, 255, 1)'  
 this.fontColor = 'rgb(0, 0, 0)'  
 } else {  
 this.bgColor = 'rgba(255, 255, 255, 0)'  
 this.fontColor = 'rgb(255, 255, 255)'  
 }  
 }  
  
 @Builder  
 contentBuilder() {  
 SwiperLayout({ bannerList: this.bannerList })  
 Column() {  
 NavList({ navList: this.navList })  
 TileList({ tileList: this.tileList })  
 PlanList({ planList: this.planList })  
 Ad({ adPicture: this.adPicture })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
  
 RoomRecommend()  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 SearchBar({ bgColor: $$.bgColor, fontColor: $$.fontColor });  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder  
 })  
 }  
}

import { window } from '@kit.ArkUI';  
import { IColor } from '../model/HomeData';  
import { getWindowTopHeight, rvp } from '../utils/responsive';  
  
@Component  
export default struct ScrollContainer {  
 @BuilderParam contentBuilder: () => void = this.defaultBuilder  
 @BuilderParam navBuilder: ($$: IColor) => void = this.defaultBuilder  
 scrollY: number = 0;  
 @State bgColor: string = 'rgba(255, 255, 255, 0)'  
 @State fontColor: string = 'rgb(255, 255, 255)'  
 // 设置 window 状态栏颜色  
 setWindowStatusColor = async () => {  
 const windowClass = await window.getLastWindow(getContext())  
 windowClass.setWindowSystemBarProperties({  
 statusBarContentColor: this.fontColor,  
 navigationBarContentColor: this.fontColor  
 })  
 }  
  
 @Builder  
 defaultBuilder() {  
 }  
  
 // 处理滚动  
 handleScroll = (scrollX: number, scrollY: number) => {  
 // 更新并修正 scrollY 的值  
 this.scrollY += scrollY;  
 if (this.scrollY < 0) {  
 this.scrollY = 0  
 }  
 this.calcColor();  
 this.setWindowStatusColor();  
 }  
 // 计算颜色  
 calcColor = () => {  
 if (this.scrollY > 10) {  
 this.bgColor = 'rgba(255, 255, 255, 1)'  
 this.fontColor = '#000000'  
 } else {  
 this.bgColor = 'rgba(255, 255, 255, 0)'  
 this.fontColor = '#ffffff'  
 }  
 }  
  
 build() {  
 Stack() {  
 Scroll() {  
 Column() {  
 // 渲染内容  
 this.contentBuilder()  
 }.align(Alignment.TopStart)  
 }  
 .height('100%')  
 .scrollBar(BarState.Off)  
 .align(Alignment.TopStart)  
 .backgroundColor($r('app.color.bg\_gray\_second'))  
 .padding({ bottom: rvp(10) })  
 .onDidScroll(this.handleScroll)  
  
 Column() {  
 // 渲染头部导航  
 this.navBuilder({ fontColor: this.fontColor, bgColor: this.bgColor })  
 }.backgroundColor(this.bgColor).padding({ top: getWindowTopHeight() })  
  
 }.height('100%').alignContent(Alignment.TopStart)  
 }  
}

import type { BannerListDataSource, IBannerItem } from '../../model/HomeData';  
import { getWindowTopHeight, rvp } from '../../utils/responsive';  
  
@Component  
export default struct SwiperLayout {  
 @Prop bannerList: BannerListDataSource  
  
 build() {  
 if (this.bannerList.totalCount()) {  
 Swiper() {  
 LazyForEach(this.bannerList, (banner: IBannerItem) => {  
 Image(banner.imageURL).width('100%').height(rvp(175) + getWindowTopHeight())  
 }, (item: IBannerItem) => item.id + '')  
 }  
 .loop(true)  
 .indicator(  
 Indicator.dot()// 设置圆形导航点样式  
 .color('#CCCBCB')  
 .selectedColor($r('app.color.primary'))  
 )  
 }  
 }  
}

import type { INavItem, INavList } from '../../model/HomeData'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct NavList {  
 @Link navList: INavList  
  
 build() {  
 Grid() {  
 ForEach(this.navList, (nav: INavItem) => {  
 GridItem() {  
 Column({ space: rvp(8) }) {  
 Image(nav.imageURL).width('100%').height(rvp(56)).objectFit(ImageFit.Fill)  
 Text(nav.title).fontSize(rvp(12))  
 }  
 }.width(rvp(58))  
 }, (nav: INavItem) => nav.id + '')  
 }  
 .columnsTemplate('1fr 1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr')  
 .rowsGap(rvp(14))  
 .columnsGap(rvp(33))  
 .height(rvp(170))  
 .margin({  
 top: rvp(24)  
 })  
 }  
}

import { PADDING } from '../../constants/size';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct SearchBar {  
 @Prop bgColor: string;  
 @Prop fontColor: string;  
  
 build() {  
 Row() {  
 Text("北京").fontSize(rvp(14)).fontColor(this.fontColor)  
 Row() {  
 Image($r("app.media.search")).imageStyles(rvp(18)).margin({ right: rvp(10) })  
 // layoutWeight(1) 占剩下的空间的1份（占据剩下空间）  
 TextInput({ placeholder: '公司/地铁/小区，马上搜索' })  
 .layoutWeight(1)  
 .height('100%')  
 .fontSize(rvp(12))  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: rvp(12) })  
 .caretColor($r('app.color.gray'))// 光标颜色  
 .padding(0)  
 .backgroundColor($r('app.color.white'))  
 .stateStyles({  
 // 状态样式  
 pressed: textInputStyles // 按下状态的样式  
 })  
  
 Column() {  
 }.width(rvp(1)).height(rvp(18)).backgroundColor('#D7D7D7').margin({ left: rvp(10) })  
  
 Image($r("app.media.position")).imageStyles(rvp(18)).margin({ left: rvp(16) })  
 }  
 .width(rvp(244))  
 .height(rvp(38))  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 .border({  
 width: rvp(1),  
 radius: rvp(52),  
 color: '#D7D7D7'  
 })  
  
 // 图片需要通过 fillColor 修改颜色，只有 svg 图片能用  
 Image($r("app.media.message")).imageStyles(rvp(24)).fillColor(this.fontColor)  
 }  
 .width('100%')  
 .padding({  
 left: rvp(PADDING),  
 right: rvp(PADDING),  
 top: rvp(4),  
 bottom: rvp(4)  
 })  
 .justifyContent(FlexAlign.SpaceBetween)  
 .backgroundColor(this.bgColor)  
 }  
}  
  
/\*  
 @Styles: 用来定义复用样式  
 @Extend: 用来扩展组件样式（只有这个组件能用）  
  
 @Styles 和 @Extend 都可以用来复用样式，有什么区别：  
 1. @Styles 只支持通用属性和通用事件。@Extend 即通用属性和通用事件，也支持定义组件私有属性和私有事件  
 2. @Styles 不支持传参。@Extend 支持传参（更灵活）  
 3. @Styles 可以定义组件内或全局。@Extend 只支持  
  
 注意：全局指的是当前文件内生效。不是真正的全局  
 \*/  
  
@Extend(Image)  
function imageStyles(size: number) {  
 .width(size).height(size).objectFit(ImageFit.Fill)  
}  
  
@Styles  
function textInputStyles() {  
 .backgroundColor('#fff')  
}

import type { ITileItem, ITileList } from '../../model/HomeData';  
import { BORDER\_RADIUS } from '../../constants/size';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct TileList {  
 @Prop tileList: ITileList;  
  
 build() {  
 Row() {  
 ForEach(this.tileList, (tile: ITileItem) => {  
 Column({ space: rvp(8) }) {  
 Image(tile.imageURL).width('100%').height(rvp(57)).borderRadius(BORDER\_RADIUS)  
 Row({ space: rvp(5) }) {  
 Text(tile.title).fontSize(rvp(12))  
 Text(tile.sub\_title).fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.justifyContent(FlexAlign.Start).width('100%')  
 }.width(rvp(148))  
 }, (tile: ITileItem) => tile.id + '')  
 }.width('100%').height(rvp(79)).margin({ top: rvp(10) }).justifyContent(FlexAlign.SpaceBetween)  
 }  
}

import type { IPlanItem, IPlanList } from '../../model/HomeData';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct PlanList {  
 @Prop planList: IPlanList;  
  
 build() {  
 Row() {  
 ForEach(this.planList, (plan: IPlanItem) => {  
 Image(plan.imageURL).width(rvp(78)).height(rvp(60)).objectFit(ImageFit.Fill)  
 }, (plan: IPlanItem) => plan.id + '')  
 }.width('100%').margin({ top: rvp(18) }).justifyContent(FlexAlign.SpaceBetween)  
 }  
}

import { SHADOW\_RADIUS } from '../../constants/size';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct Ad {  
 @Prop adPicture: string;  
  
 build() {  
 Image(this.adPicture)  
 .width('100%')  
 .height(rvp(61))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(9) })  
 .shadow({  
 radius: SHADOW\_RADIUS,  
 color: $r('app.color.shadow'),  
 offsetX: 0,  
 offsetY: 0  
 })  
 }  
}

import { IRoomRecommendItem, ITagItem, RoomRecommendDataSource } from '../../model/RoomRecommendDataSource'  
import { getRoomRecommendListApi } from '../../api/home'  
  
import { BORDER\_RADIUS, PADDING } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct RoomRecommend {  
 // 获取推荐房源列表  
 @State roomRecommendDataSource: RoomRecommendDataSource = new RoomRecommendDataSource([])  
  
 aboutToAppear() {  
 this.getRoomRecommendList();  
 }  
  
 async getRoomRecommendList() {  
 const roomRecommendList = await getRoomRecommendListApi();  
 this.roomRecommendDataSource = new RoomRecommendDataSource(roomRecommendList)  
 }  
  
 // 头部导航  
 @Builder  
 RoomNav() {  
 Row() {  
 Text() {  
 Span('周边房源')  
 Span('推荐').fontColor($r('app.color.primary'))  
 }.fontSize(rvp(18))  
  
 Row({ space: rvp(6) }) {  
 Text('更多推荐').fontColor($r('app.color.gray\_second')).fontSize(rvp(12))  
 Image($r('app.media.arrow\_right')).width(rvp(4)).height(rvp(8))  
 }  
 }  
 .width('100%')  
 .height(rvp(67))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(23) })  
 }  
  
 // 房源列表  
 @Builder  
 RoomList() {  
 if (this.roomRecommendDataSource.roomRecommendList.length) {  
 Grid() {  
 LazyForEach(this.roomRecommendDataSource, (room: IRoomRecommendItem) => {  
 GridItem() {  
 Stack() {  
 Column() {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: BORDER\_RADIUS,  
 topRight: BORDER\_RADIUS  
 })  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(rvp(16)).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(rvp(14)).fontColor('#E03810')  
 }.height(rvp(22))  
  
 Text(room.rentArea + '㎡').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text(room.houseTitle)  
 .fontSize(rvp(14))  
 .margin({ top: rvp(1) })  
 .height(rvp(19))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag: ITagItem, index) => {  
 Text(tag.name)  
 .padding({  
 top: rvp(1),  
 bottom: rvp(1),  
 left: rvp(6),  
 right: rvp(6)  
 })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(rvp(12))  
 }, (tag: ITagItem) => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(91))  
 }.width('100%')  
  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(12))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(rvp(12)).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }.alignContent(Alignment.TopStart).backgroundColor($r('app.color.white'))  
 }.height(rvp(210))  
 }, (room: IRoomRecommendItem) => room.id)  
 }  
 .columnsTemplate('1fr 1fr')  
 .columnsGap(rvp(8))  
 .rowsGap(rvp(8))  
 .width('100%')  
 .padding({  
 left: rvp(PADDING),  
 right: rvp(PADDING),  
 top: rvp(8),  
 bottom: rvp(8)  
 })  
 .backgroundColor($r('app.color.bg\_gray'))  
 .borderRadius(BORDER\_RADIUS)  
 }  
 }  
  
 build() {  
 Column() {  
 this.RoomNav()  
 this.RoomList()  
 }.width('100%')  
 }  
}

## 6.3. 扩展

### 6.3.1. 识别手机、平板、电脑的方案

1. 定义 BreakpointSystem 类

import mediaQuery from '@ohos.mediaquery'  
  
class Breakpoint {  
 name: string = '';  
 size: number = 0;  
 mediaQueryListener?: mediaQuery.MediaQueryListener  
}  
  
export class BreakpointSystem {  
 private currentBreakpoint: string = 'sm'  
 // sm 对应 phone  
 // md 对应 pad  
 // lg 对应 pc 匹配不上也是 pc  
 private breakpoints: Breakpoint[] = [{ name: 'sm', size: 768 },  
 { name: 'md', size: 992 }, { name: 'lg', size: 1200 }]  
  
 private updateCurrentBreakpoint(breakpoint: string) {  
 if (this.currentBreakpoint !== breakpoint) {  
 this.currentBreakpoint = breakpoint  
 AppStorage.Set<string>('currentBreakpoint', this.currentBreakpoint)  
 console.log('on current breakpoint: ' + this.currentBreakpoint)  
 }  
 }  
  
 public register() {  
 this.breakpoints.forEach((breakpoint: Breakpoint, index) => {  
 let condition: string = ''  
 if (index === this.breakpoints.length - 1) {  
 condition = "(" + breakpoint.size + "vp<=width" + ")"  
 } else {  
 condition = "(" + breakpoint.size + "vp<=width<" + this.breakpoints[index + 1].size + "vp)"  
 }  
 breakpoint.mediaQueryListener = mediaQuery.matchMediaSync(condition)  
 breakpoint.mediaQueryListener.on('change', (mediaQueryResult) => {  
 if (mediaQueryResult.matches) {  
 this.updateCurrentBreakpoint(breakpoint.name)  
 }  
 })  
 })  
 }  
  
 public unregister() {  
 this.breakpoints.forEach((breakpoint: Breakpoint) => {  
 if (breakpoint.mediaQueryListener) {  
 breakpoint.mediaQueryListener.off('change')  
 }  
 })  
 }  
}

1. 首页页面注册 BreakpointSystem

import { BreakpointSystem, BreakPointType } from '../common/BreakpointSystem'  
  
@Entry  
@Component  
struct Home {  
 @StorageProp('currentBreakpoint') currentBreakpoint: string = 'sm'  
 private breakpointSystem: BreakpointSystem = new BreakpointSystem()  
 // 注册  
 aboutToAppear() {  
 this.breakpointSystem.register()  
 }  
  
 aboutToDisappear() {  
 this.breakpointSystem.unregister()  
 }  
  
 build() {  
 Column() {  
 Text('测试')  
 }  
 }  
}

1. 此时其他页面定义 @StorageProp('currentBreakpoint') currentBreakpoint: string = 'sm' 方式，最终就能通过 this.currentBreakpoint 得到

### 6.3.2. 识别横竖屏方案

@Entry  
@Component  
struct Home {  
 // 识别横屏的方法  
 listener = mediaquery.matchMediaSync('(orientation: landscape)')  
  
 onPortrait(mediaQueryResult) {  
 if (mediaQueryResult.matches) {  
 console.log('当前是横屏')  
 } else {  
 console.log('当前是竖屏')  
 }  
 }  
  
 aboutToAppear() {  
 this.listener.on('change', this.onPortrait.bind(this))  
 }  
  
 build() {  
 Column() {  
 Text('mediaquery')  
 }  
 }  
}

# 7. 想看页面功能

## 7.1. 布局分析



## 7.2. 引入图片资源

[📎bag.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512742107-9c67b983-989f-4d1f-8f2f-132c3a10c9e5.svg)

[📎journey.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512742416-20e16a20-324b-4ef2-a8c7-e4aa2069f565.svg)

[📎find\_room.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512742557-01569e31-a82e-45e8-bc07-644c9652f961.png)

[📎blank.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512742711-8315311f-c311-4d52-b3b5-087a9a9bae30.png)

[📎bus.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512742925-c4a273fd-30f2-4a36-adc2-334454715aa6.svg)

[📎map.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512743071-0108328f-62e9-486c-90da-016e84275794.svg)

## 7.3. 背景线性渐变色

import ScrollContainer from '../components//ScrollContainer'  
  
import { PADDING } from '../constants/size'  
import { rvp } from '../utils/responsive'  
  
@Component  
export default struct See {  
 @Builder  
 navBuilder() {  
 }  
  
 @Builder  
 contentBuilder() {  
 Column() {  
 }.width('100%').height('100%').padding({ left: rvp(PADDING), right: rvp(PADDING) }).linearGradient({  
 angle: 180,  
 direction: GradientDirection.Top,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ],  
 repeating: false,  
 })  
 }  
  
 build() {  
 ScrollContainer({  
 navBuilder: this.navBuilder,  
 contentBuilder: this.contentBuilder  
 })  
  
 }  
}

## 7.4. NavBar 组件功能

1. 搭建 **views > See > NavBar** 组件

import { rvp } from '../../utils/responsive'  
import { PADDING } from '../../constants/size'  
  
@Component  
export default struct NavBar {  
 build() {  
 Row() {  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.bag')).width(rvp(16)).height(rvp(16))  
 Text('填写通勤地址').fontSize(rvp(14))  
 }  
  
 Row({ space: rvp(28) }) {  
 Column({ space: rvp(1) }) {  
 Image($r('app.media.message')).width(rvp(20)).height(rvp(20)).fillColor($r('app.color.black'))  
 Text('消息').fontSize(rvp(12))  
 }  
  
 Column({ space: rvp(1) }) {  
 Image($r('app.media.journey')).width(rvp(20)).height(rvp(20))  
 Text('行程').fontSize(rvp(12))  
 }  
 }  
 }  
 .width('100%')  
 .height(rvp(44))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }  
}

1. **See** 使用 **NavBar** 组件

import ScrollContainer from '../components//ScrollContainer'  
import NavBar from '../views/See/NavBar'  
  
import { PADDING } from '../constants/size'  
import { rvp } from '../utils/responsive'  
  
@Component  
export default struct See {  
 @Builder  
 navBuilder() {  
 NavBar()  
 }  
  
 @Builder  
 contentBuilder() {  
 Column() {  
 }.width('100%').height('100%').padding({ left: rvp(PADDING), right: rvp(PADDING) }).linearGradient({  
 angle: 180,  
 direction: GradientDirection.Top,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ],  
 repeating: false,  
 })  
 }  
  
 build() {  
 ScrollContainer({  
 navBuilder: this.navBuilder,  
 contentBuilder: this.contentBuilder  
 })  
  
 }  
}

## 7.5. FindRoom 组件功能

1. 搭建 **views > See > FindRoom** 组件

import { BORDER\_RADIUS\_S, SHADOW\_RADIUS } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct FindRoom {  
 build() {  
 Column() {  
 Image($r('app.media.find\_room')).width(rvp(58)).height(rvp(23))  
 Text('发现你想看的房子').fontSize(rvp(14)).margin({ top: rvp(20) })  
 Column({ space: rvp(10) }) {  
 Image($r('app.media.blank')).width(rvp(234)).height(rvp(221)).objectFit(ImageFit.Fill)  
 Text('暂无想看房源，试试如下找房方式').fontSize(rvp(18))  
 }.margin({ top: rvp(6) }).width('100%')  
  
 Row() {  
 Row({ space: rvp(5) }) {  
 Image($r('app.media.bus')).width(rvp(34)).height(rvp(34)).objectFit(ImageFit.Fill)  
 Column({ space: rvp(5) }) {  
 Text('通勤找房').fontSize(rvp(16))  
 Text('按公司通勤时长找房').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 }.alignItems(HorizontalAlign.Start)  
 }  
  
 Row({ space: rvp(5) }) {  
 Image($r('app.media.map')).width(rvp(34)).height(rvp(34)).objectFit(ImageFit.Fill)  
 Column({ space: rvp(5) }) {  
 Text('地图找房').fontSize(rvp(16))  
 Text('在你想住的地方找房').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 }.alignItems(HorizontalAlign.Start)  
 }  
 }  
 .width('100%')  
 .margin({ top: rvp(20) })  
 .padding(rvp(24))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .borderRadius(BORDER\_RADIUS\_S)  
 .shadow({  
 radius: SHADOW\_RADIUS,  
 color: $r('app.color.shadow'),  
 offsetX: 0,  
 offsetY: 0  
 })  
 }.margin({ top: rvp(49) }).alignItems(HorizontalAlign.Start).width('100%')  
 }  
}

1. **See** 使用 **FindRoom** 组件

import ScrollContainer from '../components//ScrollContainer'  
import NavBar from '../views/See/NavBar'  
import FindRoom from '../views/See/FindRoom'  
  
import { PADDING } from '../constants/size'  
import { rvp } from '../utils/responsive'  
  
@Component  
export default struct See {  
 @Builder  
 navBuilder() {  
 NavBar()  
 }  
  
 @Builder  
 contentBuilder() {  
 Column() {  
 FindRoom()  
 }.width('100%').height('100%').padding({ left: rvp(PADDING), right: rvp(PADDING) }).linearGradient({  
 angle: 180,  
 direction: GradientDirection.Top,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ],  
 repeating: false,  
 })  
 }  
  
 build() {  
 ScrollContainer({  
 navBuilder: this.navBuilder,  
 contentBuilder: this.contentBuilder  
 })  
  
 }  
}

# 8. 服务页面功能

## 8.1. 布局分析



## 8.2. 发送请求，获取数据

1. 定义接口函数&类型

* model > ServiceData

export type IVipServiceList = IVipServiceItem[]  
  
export interface IVipServiceItem {  
 "name": string  
 "content": number  
}  
  
export interface IHomeServiceItem {  
 "icon": string  
 "title": string  
}  
  
export type IHomeServiceList = IHomeServiceItem[]  
  
export interface IServiceActivity {  
 "name": string  
 "img": string  
 "title": string  
 "subTitle": string  
}  
  
export interface ICleaningServiceInfoItem {  
 "img": string  
 "title": string  
 "subTitle": string  
 "originalPrice": string  
 "priceAfterDiscounts": string  
}  
  
export type ICleaningServiceInfoList = ICleaningServiceInfoItem[]  
  
  
export interface INewcomerZone {  
 "img": string  
}  
  
export interface IServiceData {  
 "vipServiceList": IVipServiceList  
 "homeServiceList": IHomeServiceList  
 "serviceActivity": IServiceActivity  
 "newcomerZone": INewcomerZone  
 "cleaningServiceInfo": ICleaningServiceInfoList  
}

* api > service

import { http } from '../utils/http';  
import type { IServiceData } from '../model/ServiceData'  
  
// 获取租房列表数据  
export const getServiceDataApi = (): Promise<IServiceData> => {  
 return http.get<IServiceData>('/service/info')  
}

1. 组件发送请求，更新数据

import { getServiceDataApi } from '../api/service';  
  
@Component  
export default struct Service {  
  
 aboutToAppear(): void {  
 this.getServiceData()  
 }  
  
 async getServiceData() {  
 const serviceData = await getServiceDataApi();  
 }  
  
 build() {  
 Column() {  
 Text("Service");  
 }  
 }  
}

## 8.3. Bg 组件功能

1. 引入图片资源

[📎service\_bg.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512743175-ce6a8b4a-084f-478b-818e-9a8d141ee538.png)

[📎arrow\_down\_2.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512743310-a4fb6af0-c0c6-4632-aa09-2a9b83f516e5.svg)

1. 搭建 **views > Service > Bg** 组件

import { getWindowTopHeight, rvp } from '../../utils/responsive';  
  
@Component  
export default struct Bg {  
 build() {  
 Stack() {  
 Image($r('app.media.service\_bg'))  
 .width('100%')  
 .height(rvp(185) + getWindowTopHeight())  
 .objectFit(ImageFit.Fill)  
 Image($r('app.media.arrow\_down\_2'))  
 .width(rvp(10))  
 .height(rvp(3))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(166) + getWindowTopHeight() })  
 }  
 }  
}

1. **Service** 使用 **Bg** 组件

import { getServiceDataApi } from '../api/service';  
  
import Bg from '../views/Service/Bg';  
  
@Component  
export default struct Service {  
 aboutToAppear(): void {  
 this.getServiceData()  
 }  
  
 async getServiceData() {  
 const serviceData = await getServiceDataApi();  
 }  
  
 build() {  
 Scroll() {  
 Stack() {  
 Bg()  
 }.alignContent(Alignment.TopStart)  
 }  
 .height('100%')  
 .scrollBar(BarState.Off)  
 .align(Alignment.TopStart)  
 .backgroundColor('#F9F9F9')  
 }  
}

## 8.4. NavBar 组件功能

1. 引入图片资源

[📎arrow\_down.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512743488-88c45dd3-155d-41df-bb30-e6170521c9e7.svg)

[📎customer.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512743650-29b5b0ee-03ee-41db-b476-3da7f2d497a5.svg)

1. 搭建 **views > Service > NavBar** 组件

import { PADDING } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct NavBar {  
 @Prop fontColor: string  
 @Prop bgColor: string  
  
 build() {  
 Row() {  
 Row({ space: rvp(5) }) {  
 Text('请选择地址').fontSize(rvp(16)).fontColor(this.fontColor)  
 Image($r('app.media.arrow\_down')).width(rvp(6)).height(rvp(6)).fillColor(this.fontColor)  
 }  
  
 Row({ space: rvp(28) }) {  
 Column({ space: rvp(1) }) {  
 Image($r('app.media.message')).width(rvp(20)).height(rvp(20)).fillColor(this.fontColor)  
 Text('消息').fontSize(rvp(12)).fontColor(this.fontColor)  
 }  
  
 Column({ space: rvp(1) }) {  
 Image($r('app.media.customer')).width(rvp(20)).height(rvp(20)).fillColor(this.fontColor)  
 Text('行程').fontSize(rvp(12)).fontColor(this.fontColor)  
 }  
 }  
 }  
 .width('100%')  
 .height(rvp(44))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 .backgroundColor(this.bgColor)  
 }  
}

1. **Service** 使用 **NavBar** 组件

import { getServiceDataApi } from '../api/service';  
import { IColor } from '../model/HomeData';  
  
import Bg from '../views/Service/Bg';  
import ScrollContainer from '../components/ScrollContainer';  
import NavBar from '../views/Service/NavBar';  
  
@Component  
export default struct Service {  
 aboutToAppear(): void {  
 this.getServiceData()  
 }  
  
 async getServiceData() {  
 const serviceData = await getServiceDataApi();  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Bg()  
 }  
  
 build() {  
 ScrollContainer({  
 navBuilder: this.navBuilder,  
 contentBuilder: this.contentBuilder  
 })  
 }  
}

1. 解决颜色显示问题 - TODO

此时如果我们在 **Home** 组件和 **Service** 组件之间反复滑动，**NavBar** 组件的颜色或背景色显示会出现问题。

原因：每个页面读取到的颜色是相同的，所以上个页面改动的颜色会影响下一个页面。

解决方案：每个页面的颜色都单独存储，每次切换页面恢复上一次的颜色

export interface IColor {  
 navBarBgColor: string;  
 navBarFontColor: string;  
}  
  
export type IColors = IColor[]

import Home from './Home';  
import See from './See';  
import Service from './Service';  
import Discover from './Discover';  
import My from './My';  
  
import rvp from '../utils/responsive/responsiveVP';  
import { xs } from '../utils/responsive/responsiveFontSize';  
import type { IColors } from '../model/ColorsData'  
  
@Entry  
@Component  
struct Index {  
 @StorageLink('navBarFontColor') navBarFontColor: string = ''  
 @StorageLink('navBarBgColor') navBarBgColor: string = ''  
 // 当前 tab 高亮的下标  
 @State currentIndex: number = 0;  
 colors: IColors = []  
  
 // 生成 tab 的函数  
 @Builder TabBuilder(index: number, text: string, icon: Resource, activeIcon: Resource) {  
 Column() {  
 // 项目中常用的颜色、大小等会定义成常量，将来方便复用和统一调整。而不常用的大小颜色就不需要定义了，直接写死即可。  
 Image(this.currentIndex === index ? activeIcon : icon).width(rvp(28))  
 Text(text)  
 .fontSize(xs())  
 .fontColor(this.currentIndex === index ? $r('app.color.black') : '#a0a0a0')  
 .height(rvp(15))  
 }.width('100%').height('100%').justifyContent(FlexAlign.Center)  
 }  
  
 build() {  
 Column() {  
 Tabs({ barPosition: BarPosition.End, index: this.currentIndex }) {  
 TabContent() {  
 Home()  
 }.tabBar(this.TabBuilder(0, '首页', $r('app.media.tabbar\_home'), $r('app.media.tabbar\_home\_active')))  
  
 TabContent() {  
 See()  
 }.tabBar(this.TabBuilder(1, '想看', $r('app.media.tabbar\_see'), $r('app.media.tabbar\_see\_active')))  
  
 TabContent() {  
 Service()  
 }.tabBar(this.TabBuilder(2, '服务', $r('app.media.tabbar\_service'), $r('app.media.tabbar\_service\_active')))  
  
 TabContent() {  
 Discover()  
 }.tabBar(this.TabBuilder(3, '发现', $r('app.media.tabbar\_discover'), $r('app.media.tabbar\_discover\_active')))  
  
 TabContent() {  
 My()  
 }.tabBar(this.TabBuilder(4, '我的', $r('app.media.tabbar\_my'), $r('app.media.tabbar\_my\_active')))  
 }  
 .vertical(false)  
 .barMode(BarMode.Fixed)  
 .onChange((index: number) => {  
 // 将上个页面color存储起来  
 this.colors[this.currentIndex] = {  
 navBarBgColor: this.navBarBgColor,  
 navBarFontColor: this.navBarFontColor  
 }  
 // 将当前页面的color赋值上去  
 this.navBarFontColor = this.colors[index]?.navBarFontColor || 'rgb(255, 255, 255)'  
 this.navBarBgColor = this.colors[index]?.navBarBgColor || 'rgba(255, 255, 255, 0)'  
  
 this.currentIndex = index  
 })  
 .barHeight(rvp(50)) // 设置 tab 导航栏的高度，默认 56  
 .scrollable(false)  
 .width('100%')  
 .height('100%')  
 .backgroundColor($r('app.color.white'))  
 }.height('100%')  
 }  
}

## 8.5. VIP 组件功能

1. 引入图片资源

[📎vip.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512743819-069f83b4-c2f4-4cd9-a597-3ed7917cbc03.png)[📎vip\_bg.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512743980-6b54d79b-84a4-4413-b5b5-0f72c5394c4f.png)[📎crown.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512744197-f34a5794-31a4-4469-865b-9e1ead12eecf.svg)

1. 父子组件通信

* **Service** 父组件传递数据给 **VIP** 子组件

import { getServiceDataApi } from '../api/service';  
import { IColor } from '../model/HomeData';  
import { IVipServiceList } from '../model/ServiceData';  
import { getWindowTopHeight, rvp } from '../utils/responsive';  
import { PADDING } from '../constants/size';  
  
import Bg from '../views/Service/Bg';  
import ScrollContainer from '../components/ScrollContainer';  
import NavBar from '../views/Service/NavBar';  
import VIP from '../views/Service/VIP';  
  
@Component  
export default struct Service {  
 @State vipServiceList: IVipServiceList = [];  
 @State homeServiceList: IHomeServiceList = [];  
  
 aboutToAppear() {  
 this.getServiceData()  
 }  
  
 async getServiceData() {  
 const serviceData = await getServiceDataApi();  
 this.vipServiceList = serviceData.vipServiceList;  
 this.homeServiceList = serviceData.homeServiceList;  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 VIP({ vipServiceList: this.vipServiceList })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: getWindowTopHeight() })  
 }.alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 navBuilder: this.navBuilder,  
 contentBuilder: () => {  
 this.contentBuilder()  
 }  
 })  
 }  
}

* **VIP** 子组件声明接受

import type { IVipServiceList } from '../../model/ServiceData'  
  
@Component  
export default struct VIP {  
 @Prop vipServiceList: IVipServiceList  
  
 build() {  
 Column() {  
 Text("VIP");  
 }  
 }  
}

1. 搭建 **VIP** 组件布局样式

import { IVipServiceItem, IVipServiceList } from '../../model/ServiceData';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct VIP {  
 @Prop vipServiceList: IVipServiceList;  
  
 getUnit(name: string): string {  
 console.log(name)  
 switch (name) {  
 case '余额':  
 return ' 元';  
 case '优惠券':  
 return ' 张';  
 case '会员特权':  
 return ' 项';  
 case '会员福利':  
 return ' 个';  
 default:  
 return '';  
 }  
 }  
  
 build() {  
 Stack() {  
 Image($r('app.media.vip\_bg')).width('100%').height('100%').objectFit(ImageFit.Fill)  
 Column({ space: rvp(16) }) {  
 Row({ space: rvp(5) }) {  
 Image($r('app.media.crown')).width(rvp(18)).height(rvp(18))  
 Text('开通谷粒卡享特权').fontSize(rvp(18)).fontColor($r('app.color.white'))  
 Blank()  
 Image($r('app.media.vip')).width(rvp(43)).height(rvp(19)).objectFit(ImageFit.Fill)  
 }.width('100%').padding({ left: rvp(39), right: rvp(41) })  
  
 Row() {  
 ForEach(this.vipServiceList, (vip: IVipServiceItem) => {  
 Column({ space: rvp(5) }) {  
 Text(vip.name).fontColor($r('app.color.white')).fontSize(rvp(14))  
 Text() {  
 Span(vip.content + '').fontSize(rvp(18)).fontWeight(700).fontColor($r('app.color.white'))  
 Span(this.getUnit(vip.name)).fontSize(rvp(12)).fontColor($r('app.color.white'))  
 }  
 }  
 }, (vip: IVipServiceItem) => vip.name)  
 }.padding({ left: rvp(35), right: rvp(41) }).justifyContent(FlexAlign.SpaceBetween).width('100%')  
 }.padding({ top: rvp(12) }).height('100%')  
 }.margin({ top: rvp(5 + 44) }).height(rvp(99))  
 }  
}

## 8.6. DomesticService 组件功能

1. 父子组件通信

import { getServiceDataApi } from '../api/service';  
import { IColor } from '../model/HomeData';  
import { IHomeServiceList, IVipServiceList } from '../model/ServiceData';  
import { getWindowTopHeight, rvp } from '../utils/responsive';  
import { PADDING } from '../constants/size';  
  
import Bg from '../views/Service/Bg';  
import ScrollContainer from '../components/ScrollContainer';  
import NavBar from '../views/Service/NavBar';  
import VIP from '../views/Service/VIP';  
import DomesticService from '../views/Service/DomesticService';  
  
@Component  
export default struct Service {  
 @State vipServiceList: IVipServiceList = [];  
 @State homeServiceList: IHomeServiceList = [];  
  
 aboutToAppear() {  
 this.getServiceData()  
 }  
  
 async getServiceData() {  
 const serviceData = await getServiceDataApi();  
 this.vipServiceList = serviceData.vipServiceList;  
 this.homeServiceList = serviceData.homeServiceList;  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 VIP({ vipServiceList: this.vipServiceList })  
 DomesticService({ homeServiceList: this.homeServiceList })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: getWindowTopHeight() })  
 }.alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 navBuilder: this.navBuilder,  
 contentBuilder: () => {  
 this.contentBuilder()  
 }  
 })  
 }  
}

import { IHomeServiceList } from '../../model/ServiceData';  
  
@Component  
export default struct DomesticService {  
 @Prop homeServiceList: IHomeServiceList;  
  
 build() {  
 Column() {  
 Text("DomesticService");  
 }  
 }  
}

1. 封装 **components > Card** 公共组件

import { BORDER\_RADIUS } from '../constants/size';  
import { rvp } from '../utils/responsive';  
  
@Component  
export default struct Card {  
 @Prop cardPadding: number;  
 @BuilderParam render: () => void  
  
 build() {  
 Column() {  
 this.render()  
 }  
 .padding(this.cardPadding)  
 .backgroundColor($r('app.color.white'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .width('100%')  
 .alignItems(HorizontalAlign.Start)  
 }  
}

1. 搭建 **views > Services > DomesticService** 组件

import Card from '../../components/Card';  
  
import type { IHomeServiceItem, IHomeServiceList } from '../../model/ServiceData';  
  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct DomesticService {  
 @Prop homeServiceList: IHomeServiceList;  
  
 @Builder  
 renderBuilder() {  
 Row() {  
 Text('家庭服务').fontSize(rvp(16)).fontWeight(700)  
 Row({ space: rvp(10) }) {  
 Text('全部服务').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_right')).width(rvp(4))  
 }  
 }.justifyContent(FlexAlign.SpaceBetween).width('100%')  
  
 Grid() {  
 ForEach(this.homeServiceList, (nav: IHomeServiceItem) => {  
 GridItem() {  
 Column({ space: rvp(10) }) {  
 Image(nav.icon).width(rvp(24)).height(rvp(24)).objectFit(ImageFit.Fill)  
 Text(nav.title).fontSize(rvp(12))  
 }  
 }.width(rvp(60))  
 }, (nav: IHomeServiceItem) => nav.title)  
 }  
 .columnsTemplate('1fr 1fr 1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr 1fr')  
 .rowsGap(rvp(22))  
 .columnsGap(rvp(18))  
 .margin({ top: rvp(20), bottom: rvp(9) })  
 .width('100%')  
 .height(rvp(188))  
 }  
  
 build() {  
 Column() {  
 Card({  
 cardPadding: rvp(8), renderBuilder: () => {  
 this.renderBuilder()  
 }  
 })  
 }.margin({ top: rvp(43) })  
 }  
}

## 8.7. ServiceCalendar 组件功能

1. 父子组件通信

import { getServiceDataApi } from '../api/service';  
import { IColor } from '../model/HomeData';  
import { IHomeServiceList, IVipServiceList, IServiceActivity } from '../model/ServiceData';  
import { getWindowTopHeight, rvp } from '../utils/responsive';  
import { PADDING } from '../constants/size';  
  
import Bg from '../views/Service/Bg';  
import ScrollContainer from '../components/ScrollContainer';  
import NavBar from '../views/Service/NavBar';  
import VIP from '../views/Service/VIP';  
import DomesticService from '../views/Service/DomesticService';  
import ServiceCalendar from '../views/Service/ServiceCalendar';  
  
@Component  
export default struct Service {  
 @State vipServiceList: IVipServiceList = [];  
 @State homeServiceList: IHomeServiceList = [];  
 @State serviceActivity: IServiceActivity = [];  
  
 aboutToAppear() {  
 this.getServiceData()  
 }  
  
 async getServiceData() {  
 const serviceData = await getServiceDataApi();  
 this.vipServiceList = serviceData.vipServiceList;  
 this.homeServiceList = serviceData.homeServiceList;  
 this.serviceActivity = serviceData.serviceActivity;  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 VIP({ vipServiceList: this.vipServiceList })  
 DomesticService({ homeServiceList: this.homeServiceList })  
 ServiceCalendar({ serviceActivity: this.serviceActivity })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: getWindowTopHeight() })  
 }.alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 navBuilder: this.navBuilder,  
 contentBuilder: () => {  
 this.contentBuilder()  
 }  
 })  
 }  
}

import type { IServiceActivity } from '../../model/ServiceData';  
  
@Component  
export default struct ServiceCalendar {  
 @Prop serviceActivity: IServiceActivity  
  
 build() {  
 Column() {  
 Text("ServiceCalendar");  
 }  
 }  
}

1. 定义生成日历月份的类 **utils > calendar**

export default class Calendar {  
 currentMonth: number;  
  
 constructor() {  
 this.currentMonth = new Date().getMonth() + 1  
 }  
  
 addZero(val: number) {  
 return val < 10 ? '0' + val : `${val}`;  
 }  
  
 getLastNthMonth(nth: number) {  
 const currentMonth = this.currentMonth;  
 const diffVal = currentMonth - nth; // 1 - 2 === -1  
 if (diffVal < 1) {  
 return 12 + diffVal  
 } else {  
 return diffVal;  
 }  
 }  
  
 getNextNthMonth(nth: number) {  
 const currentMonth = this.currentMonth;  
 const sumVal = currentMonth + nth; // 12 + 3 === 15  
 if (sumVal > 12) {  
 return sumVal - 12  
 }  
 return sumVal;  
 }  
  
 generatorCalendarList() {  
 return [  
 this.addZero(this.getLastNthMonth(2)),  
 this.addZero(this.getLastNthMonth(1)),  
 this.addZero(this.currentMonth),  
 this.addZero(this.getNextNthMonth(1)),  
 this.addZero(this.getNextNthMonth(2)),  
 this.addZero(this.getNextNthMonth(3)),  
 ]  
 }  
}

1. 搭建 **views > Services > ServiceCalendar** 组件

import type { IServiceActivity } from '../../model/ServiceData';  
import Calendar from '../../utils/calendar';  
import { rvp } from '../../utils/responsive';  
import Card from '../../components/Card';  
import { BORDER\_RADIUS, BORDER\_RADIUS\_S } from '../../constants/size';  
  
@Component  
export default struct ServiceCalendar {  
 @Prop serviceActivity: IServiceActivity  
 @State calendarList: string[] = []  
  
 aboutToAppear() {  
 const calendar = new Calendar();  
 this.calendarList = calendar.generatorCalendarList();  
 }  
  
 @Builder  
 renderBuilder() {  
 // 导航  
 Row() {  
 Text('服务日历').fontSize(rvp(16)).fontWeight(700)  
 Row({ space: rvp(10) }) {  
 Text('全部服务').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_right')).width(rvp(4))  
 }  
 }.justifyContent(FlexAlign.SpaceBetween).width('100%').padding({ top: rvp(8), left: rvp(8), right: rvp(8) })  
  
 // 日历  
 Row() {  
 ForEach(this.calendarList, (item: string, index) => {  
 Row() {  
 Text(item).fontSize(rvp(16)).fontWeight(700).width(rvp(20))  
 Text('月').fontSize(rvp(12)).fontColor($r('app.color.gray\_second')).lineHeight(18)  
 }  
 .backgroundColor(index === 2 ? '#CDFFD9' : '#FFF')  
 .padding(index === 2 ? rvp(6) : 0)  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 }, (item: string) => item)  
 }  
 .justifyContent(FlexAlign.SpaceBetween)  
 .width('100%')  
 .margin({ top: rvp(9) })  
 .padding({ left: rvp(8), right: rvp(8) })  
  
 // 活动礼包  
 Column({ space: rvp(7) }) {  
 Text(this.serviceActivity.name).fontSize(rvp(14)).fontWeight(700)  
 Row({ space: rvp(10) }) {  
 Image(this.serviceActivity.img)  
 .width(rvp(100))  
 .height(rvp(100))  
 .objectFit(ImageFit.Fill)  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 Column() {  
 Text(this.serviceActivity.title).fontSize(rvp(14)).margin({ top: rvp(7) })  
 Text(this.serviceActivity.subTitle).fontSize(rvp(12)).margin({ top: rvp(7) })  
 Row() {  
 Text('立即参与')  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(rvp(12))  
 .fontColor('#1F632F')  
 .backgroundColor('#CDFFD9')  
 .width(rvp(70))  
 .height(rvp(24))  
 .textAlign(TextAlign.Center)  
 }.margin({ top: rvp(22) }).padding({ right: rvp(8) }).width('100%').justifyContent(FlexAlign.End)  
 }.height(rvp(100)).alignItems(HorizontalAlign.Start).layoutWeight(1)  
 }.width('100%').height(rvp(100))  
 }  
 .width('100%')  
 .height(rvp(140))  
 .padding(rvp(8))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .margin({ top: rvp(12) })  
 .linearGradient({  
 angle: 210,  
 colors: [['#D0FFDB', 0], ['rgba(255, 255, 255, 1)', 0.29], ['rgba(255, 255, 255, 0)', 1]],  
 repeating: false  
 })  
 .alignItems(HorizontalAlign.Start)  
 }  
  
 build() {  
 Column() {  
 Card({  
 cardPadding: 0, renderBuilder: () => {  
 this.renderBuilder()  
 }  
 })  
 }.margin({ top: rvp(8) })  
 }  
}

## 8.8. ServiceTabs 组件功能

1. 父子组件通信

import { getServiceDataApi } from '../api/service';  
import { IColor } from '../model/HomeData';  
import { ICleaningServiceInfoList, IHomeServiceList, IServiceActivity, IVipServiceList } from '../model/ServiceData';  
import { getWindowTopHeight, rvp } from '../utils/responsive';  
import { PADDING } from '../constants/size';  
  
import Bg from '../views/Service/Bg';  
import ScrollContainer from '../components/ScrollContainer';  
import NavBar from '../views/Service/NavBar';  
import VIP from '../views/Service/VIP';  
import DomesticService from '../views/Service/DomesticService';  
import ServiceCalendar from '../views/Service/ServiceCalendar';  
import ServiceTabs from '../views/Service/ServiceTabs';  
  
@Component  
export default struct Service {  
 @State vipServiceList: IVipServiceList = [];  
 @State homeServiceList: IHomeServiceList = [];  
 @State serviceActivity: IServiceActivity = {  
 "name": '',  
 "img": '',  
 "title": '',  
 "subTitle": '',  
 };  
 @State serviceTabImage: string = ''  
 @State cleaningServiceInfo: ICleaningServiceInfoList = []  
  
 aboutToAppear() {  
 this.getServiceData()  
 }  
  
 async getServiceData() {  
 const serviceData = await getServiceDataApi();  
 this.vipServiceList = serviceData.vipServiceList;  
 this.homeServiceList = serviceData.homeServiceList;  
 this.serviceActivity = serviceData.serviceActivity;  
 this.serviceTabImage = serviceData.newcomerZone.img;  
 this.cleaningServiceInfo = serviceData.cleaningServiceInfo;  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 VIP({ vipServiceList: this.vipServiceList })  
 DomesticService({ homeServiceList: this.homeServiceList })  
 ServiceCalendar({ serviceActivity: this.serviceActivity })  
 ServiceTabs({ serviceTabImage: this.serviceTabImage, cleaningServiceInfo: this.cleaningServiceInfo })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: getWindowTopHeight() })  
 }.alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 navBuilder: this.navBuilder,  
 contentBuilder: () => {  
 this.contentBuilder()  
 }  
 })  
 }  
}

import type { ICleaningServiceInfoList } from '../../model/ServiceData';  
  
@Component  
export default struct ServiceTabs {  
 @Prop serviceTabImage: string  
 @Prop cleaningServiceInfo: ICleaningServiceInfoList  
  
 build() {  
 Column() {  
 Text("ServiceTabs");  
 }  
 }  
}

1. 搭建 **views > Services > ServiceTabs** 组件

import Card from '../../components/Card';  
import { ICleaningServiceInfoItem, ICleaningServiceInfoList } from '../../model/ServiceData';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct ServiceTabs {  
 @Prop serviceTabImage: string  
 @Link cleaningServiceInfo: ICleaningServiceInfoList  
 @State currentIndex: number = 0  
  
 // tab导航栏  
 @Builder  
 TabBuilder(index: number, name: string, subName: string) {  
 Column({ space: rvp(2) }) {  
 Text(name).fontSize(this.currentIndex === index ? rvp(18) : rvp(14)).fontWeight(700)  
 Text(subName).fontSize(rvp(12)).fontColor($r('app.color.gray\_second'))  
 if (this.currentIndex === index) {  
 Divider()  
 .strokeWidth(rvp(4))  
 .width(rvp(20))  
 .color('#51BB9E')  
 }  
 }.width('100%')  
 }  
  
 // tab内容区  
 @Builder  
 Content() {  
 Column({ space: rvp(8) }) {  
 this.NewArea()  
 this.CleanRoom()  
 }  
 }  
  
 // 新人专区  
 @Builder  
 NewArea() {  
 Card({ cardPadding: rvp(8) }) {  
 Text() {  
 Span('新人').fontSize(rvp(16)).fontColor($r('app.color.primary'))  
 Span('专区').fontSize(rvp(16))  
 }  
  
 Image(this.serviceTabImage).width('100%').height(rvp(90)).objectFit(ImageFit.Fill).margin({ top: rvp(8) })  
 }  
 }  
  
 // 洁净家  
 @Builder  
 CleanRoom() {  
 Card({ cardPadding: rvp(8) }) {  
 Text('洁净家').fontSize(rvp(16))  
 Row({ space: rvp(8) }) {  
 ForEach(this.cleaningServiceInfo, (item: ICleaningServiceInfoItem) => {  
 Column() {  
 Image(item.img).width('100%').height(rvp(115)).objectFit(ImageFit.Fill)  
 Text(item.title).fontSize(rvp(14)).fontWeight(700).margin({ top: rvp(8) })  
 Text(item.subTitle).fontSize(rvp(10)).fontColor($r('app.color.gray\_second')).margin({ top: rvp(7) })  
 Row() {  
 Text('到手价').fontSize(rvp(10)).fontColor($r('app.color.gray\_second'))  
 Text('￥' + item.priceAfterDiscounts).fontSize(rvp(16)).fontColor('#D9351A').margin({ left: rvp(6) })  
 Text('￥' + item.originalPrice)  
 .fontSize(rvp(10))  
 .fontColor($r('app.color.gray\_second'))  
 .decoration({ type: TextDecorationType.LineThrough, color: $r('app.color.gray\_second') })  
 .margin({ left: rvp(4) })  
 }.margin({ top: rvp(8) })  
 }.layoutWeight(1).alignItems(HorizontalAlign.Start)  
 }, (item: ICleaningServiceInfoItem) => item.title)  
 }.margin({ top: rvp(8) })  
 }  
 }  
  
 build() {  
 Tabs({ barPosition: BarPosition.Start }) {  
 TabContent() {  
 this.Content()  
 }.tabBar(this.TabBuilder(0, '会员精选', '专业服务'))  
  
 TabContent() {  
 this.Content()  
 }.tabBar(this.TabBuilder(1, '居家保洁', '解放双手'))  
  
 TabContent() {  
 this.Content()  
 }.tabBar(this.TabBuilder(2, '深层清洁', '高温蒸洗'))  
  
 TabContent() {  
 this.Content()  
 }.tabBar(this.TabBuilder(3, '新居乔迁', '一站搬家'))  
 }  
 .vertical(false)  
 .barMode(BarMode.Fixed)  
 .barWidth('100%')  
 .barHeight(rvp(53))  
 .onChange((index: number) => {  
 this.currentIndex = index  
 })  
 .width('100%')  
 .height(rvp(423))  
 .margin({ top: rvp(18) })  
 }  
}

# 9. 发现页面功能

## 9.1. 布局分析



## 9.2. 发送请求，获取数据

1. 定义接口和类型

* **model > DiscoverData**

export interface IDanMuItem {  
 "avatar": string  
 "content": string  
}  
  
export type IDanMuList = IDanMuItem[]  
  
export interface IMainPost {  
 "followerNum": number  
 "title": string  
 "userAvatar": string  
 "userName": string  
 "postContent": string  
}  
  
export interface IContentBlockItem {  
 img: string  
}  
  
export type IContentBlockList = IContentBlockItem[]  
  
export interface IPostZone {  
 "mainPost": IMainPost  
 "contentBlockList": IContentBlockList  
}  
  
export interface IBenefitsItem {  
 "title": string  
 "img": string  
}  
  
export type IBenefits = IBenefitsItem[]  
  
export interface IDiscoverData {  
 "danMuList1": IDanMuList,  
 "danMuList2": IDanMuList,  
 "postZone": IPostZone,  
 "benefits": IBenefits  
}

* **api > discover**

import { http } from '../utils/http';  
import type { IDiscoverData } from '../model/DiscoverData';  
  
// 获取发现列表数据  
export const getDiscoverDataApi = (): Promise<IDiscoverData> => {  
 return http.get<IDiscoverData>('/discover/info')  
}

1. 组件发送请求，更新数据

import { getDiscoverDataApi } from '../api/discover'  
  
@Component  
export default struct Discover {  
 aboutToAppear(): void {  
 this.getDiscoverData()  
 }  
  
 async getDiscoverData() {  
 const discoverData = await getDiscoverDataApi()  
 }  
  
 build() {  
 Column() {  
 Text("Discover");  
 }  
 }  
}

## 9.3. 使用 ScrollContainer 组件

import { getDiscoverDataApi } from '../api/discover'  
import ScrollContainer from '../components/ScrollContainer'  
  
@Component  
export default struct Discover {  
 aboutToAppear(): void {  
 this.getDiscoverData()  
 }  
  
 async getDiscoverData() {  
 const discoverData = await getDiscoverDataApi()  
 }  
  
 @Builder  
 navBuilder() {  
 Text('navBar')  
 }  
  
 @Builder  
 contentBuilder() {  
 Text('content')  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: this.contentBuilder,  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 9.4. NavBar 组件功能

1. 搭建 **views > Discover > NavBar** 组件

import { PADDING } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct NavBar {  
 @Prop bgColor: string = ''  
  
 build() {  
 Row() {  
 Text('家庭服务').fontSize(rvp(18)).fontWeight(700).fontColor($r('app.color.black'))  
 Image($r('app.media.message')).width(rvp(20)).height(rvp(20)).fillColor($r('app.color.black'))  
 }  
 .width('100%')  
 .height(rvp(44))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 .backgroundColor(this.bgColor)  
 }  
}

1. **Discover** 使用 **NavBar** 组件

import { getDiscoverDataApi } from '../api/discover'  
import { IColor } from '../model/HomeData'  
  
import ScrollContainer from '../components/ScrollContainer'  
import NavBar from '../views/Discover/NavBar'  
  
@Component  
export default struct Discover {  
 aboutToAppear(): void {  
 this.getDiscoverData()  
 }  
  
 async getDiscoverData() {  
 const discoverData = await getDiscoverDataApi()  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar({ bgColor: $$.bgColor })  
 }  
  
 @Builder  
 contentBuilder() {  
 Text('content')  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: this.contentBuilder,  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 9.5. Bg 组件功能

1. 引入图片资源

[📎discover\_bg.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512744394-7cbe0980-a9f7-402a-81ea-e5782323ec24.png)

1. 搭建 **views > Discover > Bg** 组件

import { getWindowTopHeight, rvp } from '../../utils/responsive'  
  
@Component  
export default struct Bg {  
 build() {  
 Image($r('app.media.discover\_bg'))  
 .width('100%')  
 .height(rvp(322 - 36) + getWindowTopHeight())  
 .objectFit(ImageFit.Fill)  
 }  
}

1. **Discover** 使用 **Bg** 组件

import { getDiscoverDataApi } from '../api/discover'  
import { IColor } from '../model/HomeData'  
  
import ScrollContainer from '../components/ScrollContainer'  
import NavBar from '../views/Discover/NavBar'  
import Bg from '../views/Discover/Bg'  
  
@Component  
export default struct Discover {  
 aboutToAppear(): void {  
 this.getDiscoverData()  
 }  
  
 async getDiscoverData() {  
 const discoverData = await getDiscoverDataApi()  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar({ bgColor: $$.bgColor })  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 }  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: this.contentBuilder,  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 9.6. 文字组件功能

import { getDiscoverDataApi } from '../api/discover'  
import { IColor } from '../model/HomeData'  
  
import ScrollContainer from '../components/ScrollContainer'  
import NavBar from '../views/Discover/NavBar'  
import Bg from '../views/Discover/Bg'  
import { getWindowTopHeight, rvp } from '../utils/responsive'  
  
@Component  
export default struct Discover {  
 aboutToAppear(): void {  
 this.getDiscoverData()  
 }  
  
 async getDiscoverData() {  
 const discoverData = await getDiscoverDataApi()  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar({ bgColor: $$.bgColor })  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 Text('与100万用户发现生活').margin({ top: rvp(44), left: rvp(16) }).fontSize(rvp(14))  
 }.alignItems(HorizontalAlign.Start).padding({ top: getWindowTopHeight() })  
 }.alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: this.contentBuilder,  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 9.7. BannerScroll 组件功能

1. 搭建 **views > Discover > BannerScroll** 组件

import { rvp } from '../../utils/responsive';  
  
import type { IDanMuItem, IDanMuList } from '../../model/DiscoverData';  
  
@Component  
export default struct BannerScroll {  
 @Prop bannerList: IDanMuList;  
  
 build() {  
 List() {  
 ForEach(this.bannerList, (banner: IDanMuItem) => {  
 ListItem() {  
 Row({ space: rvp(6) }) {  
 Image(banner.avatar).width(rvp(24)).height(rvp(24)).objectFit(ImageFit.Fill).borderRadius(rvp(24))  
 Text(banner.content).fontSize(rvp(14))  
 }  
 .padding({  
 top: rvp(6),  
 left: rvp(6),  
 bottom: rvp(6),  
 right: rvp(10)  
 })  
 .borderRadius(rvp(42))  
 .backgroundColor('#CDFFD9')  
 .margin({ left: rvp(10) })  
 }  
 }, (banner: IDanMuItem, index) => JSON.stringify(banner) + index)  
 }  
 .listDirection(Axis.Horizontal) // 排列方向  
 .edgeEffect(EdgeEffect.None) // 滑动到边缘无效果  
 .width('100%')  
 .scrollBar(BarState.Off)  
 }  
}

1. **Discover** 使用 **BannerScroll** 组件

import { getDiscoverDataApi } from '../api/discover'  
import { IColor } from '../model/HomeData'  
import { getWindowTopHeight, rvp } from '../utils/responsive'  
import { IDanMuList } from '../model/DiscoverData'  
  
import ScrollContainer from '../components/ScrollContainer'  
import NavBar from '../views/Discover/NavBar'  
import Bg from '../views/Discover/Bg'  
import BannerScroll from '../views/Discover/BannerScroll'  
  
@Component  
export default struct Discover {  
 @State danMuList1: IDanMuList = []  
 @State danMuList2: IDanMuList = []  
  
 aboutToAppear(): void {  
 this.getDiscoverData()  
 }  
  
 async getDiscoverData() {  
 const discoverData = await getDiscoverDataApi()  
 this.danMuList1 = discoverData.danMuList1  
 this.danMuList2 = discoverData.danMuList2  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar({ bgColor: $$.bgColor })  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 Text('与100万用户发现生活').margin({ top: rvp(44), left: rvp(16) }).fontSize(rvp(14))  
 Column({ space: rvp(16) }) {  
 BannerScroll({ bannerList: this.danMuList1 })  
 BannerScroll({ bannerList: this.danMuList2 })  
 }.width('100%').margin({ top: rvp(20) })  
 }.alignItems(HorizontalAlign.Start).padding({ top: getWindowTopHeight() })  
 }.alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 9.8. Ad 组件功能

1. **Discover** 组件将数据通信给 **Ad** 组件

import { getDiscoverDataApi } from '../api/discover'  
import { IColor } from '../model/HomeData'  
import { getWindowTopHeight, rvp } from '../utils/responsive'  
import { IDanMuList, IPostZone } from '../model/DiscoverData'  
  
import ScrollContainer from '../components/ScrollContainer'  
import NavBar from '../views/Discover/NavBar'  
import Bg from '../views/Discover/Bg'  
import BannerScroll from '../views/Discover/BannerScroll'  
import Ad from '../views/Discover/Ad'  
import { PADDING } from '../constants/size'  
  
@Component  
export default struct Discover {  
 @State danMuList1: IDanMuList = []  
 @State danMuList2: IDanMuList = []  
 @State postZone: IPostZone = {  
 "mainPost": {  
 "followerNum": 0,  
 "title": '',  
 "userAvatar": '',  
 "userName": '',  
 "postContent": '',  
 },  
 "contentBlockList": []  
 }  
  
 aboutToAppear(): void {  
 this.getDiscoverData()  
 }  
  
 async getDiscoverData() {  
 const discoverData = await getDiscoverDataApi()  
 this.danMuList1 = discoverData.danMuList1  
 this.danMuList2 = discoverData.danMuList2  
 this.postZone = discoverData.postZone  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar({ bgColor: $$.bgColor })  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 Text('与100万用户发现生活').margin({ top: rvp(44), left: rvp(16) }).fontSize(rvp(14))  
 Column({ space: rvp(16) }) {  
 BannerScroll({ bannerList: this.danMuList1 })  
 BannerScroll({ bannerList: this.danMuList2 })  
 }.width('100%').margin({ top: rvp(20) })  
  
 Ad({ postZone: this.postZone })  
 }  
 .alignItems(HorizontalAlign.Start)  
 .padding({ top: getWindowTopHeight(), left: rvp(PADDING), right: rvp(PADDING) })  
 .width('100%')  
 }.alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

1. **Ad** 组件声明接受数据

import type { IPostZone } from '../../model/DiscoverData';  
  
@Component  
export default struct Ad {  
 @Prop postZone: IPostZone  
  
 build() {  
 Column() {  
 Text("Ad");  
 }  
 }  
}

1. 搭建 **Ad** 组件结构样式

import { BORDER\_RADIUS } from '../../constants/size';  
import type { IContentBlockItem, IPostZone } from '../../model/DiscoverData';  
import { rvp } from '../../utils/responsive';  
import Card from '../../components/Card';  
  
@Component  
export default struct Ad {  
 @Prop postZone: IPostZone  
  
 build() {  
 Row({ space: rvp(14) }) {  
 Column() {  
 Card({ cardPadding: rvp(12) }) {  
 Text(`${this.postZone.mainPost.followerNum}位小谷粒`)  
 .fontSize(rvp(10))  
 .fontColor($r('app.color.gray\_second'))  
 .margin({ top: rvp(4), left: rvp(4) })  
 Text(this.postZone.mainPost.title).fontSize(rvp(26)).fontWeight(700).margin({ top: rvp(20), left: rvp(4) })  
 Column({ space: rvp(10) }) {  
 Row({ space: rvp(8) }) {  
 Image(this.postZone.mainPost.userAvatar).width(rvp(28)).height(rvp(28)).objectFit(ImageFit.Fill)  
 Text(this.postZone.mainPost.userName).fontSize(rvp(12)).fontColor('#636363')  
 }  
  
 Text(this.postZone.mainPost.postContent)  
 .fontSize(rvp(12))  
 .fontColor('#636363')  
 .maxLines(4)  
 .textOverflow({  
 overflow: TextOverflow.Ellipsis  
 })  
 }  
 .width('100%')  
 .height(rvp(120))  
 .padding(rvp(12))  
 .backgroundColor('#F9F9F9')  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .alignItems(HorizontalAlign.Start)  
 .margin({ top: rvp(7) })  
 }  
 }.width(rvp(192)).height('100%')  
  
 Column({ space: rvp(9) }) {  
 ForEach(this.postZone.contentBlockList, (item: IContentBlockItem) => {  
 Image(item.img).width('100%').height(rvp(67)).objectFit(ImageFit.Fill)  
 })  
 }.width(rvp(122)).height('100%')  
 }.width('100%').margin({ top: rvp(19) }).height(rvp(218))  
 }  
}

## 9.9. Welfare 布局实现

1. **Discover** 组件将数据通信给 **Welfare** 组件

import { getDiscoverDataApi } from '../api/discover'  
import { IColor } from '../model/HomeData'  
import { getWindowTopHeight, rvp } from '../utils/responsive'  
import { IBenefits, IDanMuList, IPostZone } from '../model/DiscoverData'  
import { PADDING } from '../constants/size'  
  
import ScrollContainer from '../components/ScrollContainer'  
import NavBar from '../views/Discover/NavBar'  
import Bg from '../views/Discover/Bg'  
import BannerScroll from '../views/Discover/BannerScroll'  
import Ad from '../views/Discover/Ad'  
import Welfare from '../views/Discover/Welfare'  
  
@Component  
export default struct Discover {  
 @State danMuList1: IDanMuList = []  
 @State danMuList2: IDanMuList = []  
 @State postZone: IPostZone = {  
 "mainPost": {  
 "followerNum": 0,  
 "title": '',  
 "userAvatar": '',  
 "userName": '',  
 "postContent": '',  
 },  
 "contentBlockList": []  
 }  
 @State benefits: IBenefits = []  
  
 aboutToAppear(): void {  
 this.getDiscoverData()  
 }  
  
 async getDiscoverData() {  
 const discoverData = await getDiscoverDataApi()  
 this.danMuList1 = discoverData.danMuList1  
 this.danMuList2 = discoverData.danMuList2  
 this.postZone = discoverData.postZone  
 this.benefits = discoverData.benefits  
 }  
  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar({ bgColor: $$.bgColor })  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 Text('与100万用户发现生活').margin({ top: rvp(44), left: rvp(16) }).fontSize(rvp(14))  
 Column({ space: rvp(16) }) {  
 BannerScroll({ bannerList: this.danMuList1 })  
 BannerScroll({ bannerList: this.danMuList2 })  
 }.width('100%').margin({ top: rvp(20) })  
  
 Column() {  
 Ad({ postZone: this.postZone })  
 Welfare({ benefits: this.benefits })  
 }.padding({ top: getWindowTopHeight(), left: rvp(PADDING), right: rvp(PADDING) })  
 }  
 .alignItems(HorizontalAlign.Start)  
 .padding({ top: getWindowTopHeight() })  
 .width('100%')  
 }.alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

1. **Welfare** 组件声明接受数据

import type { IBenefits } from '../../model/DiscoverData';  
  
@Component  
export default struct Welfare {  
 @Prop benefits: IBenefits  
  
 build() {  
 Column() {  
 Text("Welfare");  
 }  
 }  
}

1. 搭建 **Welfare** 组件结构样式

import Card from '../../components/Card'  
import type { IBenefits, IBenefitsItem } from '../../model/DiscoverData'  
  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct Welfare {  
 @Prop benefits: IBenefits  
  
 build() {  
 Column({ space: rvp(10) }) {  
 ForEach(this.benefits, (item: IBenefitsItem) => {  
 Card({ cardPadding: rvp(12) }) {  
 Text("小谷粒福利")  
 .width(rvp(80))  
 .height(rvp(24))  
 .borderRadius(rvp(23))  
 .backgroundColor('#CDFFD9')  
 .fontSize(rvp(12))  
 .fontColor('#1F632F')  
 .textAlign(TextAlign.Center)  
 Text(item.title)  
 .fontSize(26)  
 .fontColor('#FF000000')  
 .fontWeight(700)  
 .margin({ top: rvp(18), left: rvp(4) })  
 Image(item.img)  
 .width('100%')  
 .height(rvp(143))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(9), left: rvp(4), bottom: rvp(6) })  
 }  
 })  
 }.margin({ top: rvp(10) })  
 }  
}

# 10. 我的页面功能

## 10.1. 布局分析



## 10.2. Bg 组件功能

1. 搭建 **views > My > Bg** 组件

import { getWindowTopHeight, rvp } from '../../utils/responsive'  
  
@Component  
export default struct Bg {  
 build() {  
 Column() {  
 }.width('100%').height(rvp(211 - 36) + getWindowTopHeight()).linearGradient({  
 angle: 180,  
 colors: [  
 ['#6DC782', 0],  
 ['#8FE1A2', 0.43],  
 ['#8FE1A2', 1],  
 ]  
 })  
 }  
}

1. **My** 组件使用 **Bg** 组件

import ScrollContainer from '../components/ScrollContainer'  
import Bg from '../views/My/Bg'  
  
@Component  
export default struct My {  
 @Builder  
 navBuilder() {  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 }.width('100%')  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 10.3. NavBar 组件功能

1. 引入图片资源

[📎setting.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512744614-15eaa93a-b135-4a37-8062-4189f2ae0541.svg)[📎switch.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512744758-e4918d55-493c-4cb2-ab92-2ea66b781453.svg)

1. 搭建 **views > My > NavBar** 组件

import { PADDING } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct NavBar {  
 @Prop bgColor: string  
 @Prop fontColor: string  
  
 build() {  
 Row() {  
 Row() {  
 Image($r('app.media.switch')).width(rvp(16))  
 Text('业主版').fontSize(rvp(12))  
 }  
 .borderRadius(rvp(31))  
 .backgroundColor('rgba(255, 255, 255, 0.65)')  
 .width(rvp(79))  
 .height(rvp(26))  
 .justifyContent(FlexAlign.Center)  
  
 Row({ space: 16 }) {  
 Image($r('app.media.customer')).width(rvp(20)).height(rvp(20)).fillColor(this.fontColor)  
 Image($r('app.media.message')).width(rvp(20)).height(rvp(20)).fillColor(this.fontColor)  
 Image($r('app.media.setting')).width(rvp(20)).height(rvp(20)).fillColor(this.fontColor)  
 }  
 }  
 .width('100%')  
 .height(rvp(44))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 .backgroundColor(this.bgColor)  
 }  
}

1. **My** 组件使用 **NavBar** 组件

import { IColor } from '../model/HomeData'  
  
import ScrollContainer from '../components/ScrollContainer'  
import Bg from '../views/My/Bg'  
import NavBar from '../views/My/NavBar'  
  
@Component  
export default struct My {  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 }.width('100%')  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 10.4. UserInfo 组件功能

1. 引入图片资源

[📎avatar\_unlogin.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512744957-3ca85666-a2a4-4e00-8124-078e2c2cb5fa.png)

1. 封装 **components > Avatar** 公共组件

@Component  
export default struct Avatar {  
 @Prop src: string  
 @Prop avatarSize: number  
  
 build() {  
 Image(this.src || $r('app.media.avatar\_unlogin')).width(this.avatarSize).height(this.avatarSize).objectFit(ImageFit.Fill)  
 }  
}

1. 搭建 **views > My > UserInfo** 组件

import Avatar from '../../components/Avatar'  
  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct UserInfo {  
 @State avatarSrc: string = ''  
  
 build() {  
 Row({ space: rvp(10) }) {  
 Avatar({ src: this.avatarSrc, avatarSize: rvp(60) })  
 Column({ space: rvp(4) }) {  
 Text('Hi，小谷粒').fontSize(rvp(16)).fontColor($r('app.color.white')).fontWeight(700)  
 Text('谷粒点15').fontSize(rvp(12)).fontColor($r('app.color.white'))  
 }.alignItems(HorizontalAlign.Start)  
 }.margin({ top: rvp(11) }).width('100%')  
 }  
}

1. **My** 使用 **UserInfo** 组件

import { IColor } from '../model/HomeData'  
import { getWindowTopHeight, rvp } from '../utils/responsive'  
import { PADDING } from '../constants/size'  
  
import ScrollContainer from '../components/ScrollContainer'  
import Bg from '../views/My/Bg'  
import NavBar from '../views/My/NavBar'  
import UserInfo from '../views/My/UserInfo'  
  
@Component  
export default struct My {  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 UserInfo()  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(44) + getWindowTopHeight() })  
 }.width('100%').align(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 10.5. TimePlan 组件功能

1. 搭建 **views > My > TimePlan** 组件

import { BORDER\_RADIUS } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct TimePlan {  
 build() {  
 Column({ space: rvp(11) }) {  
 Row() {  
 Text('小谷粒时光计划').fontColor('#2C5B37').fontSize(rvp(16)).fontWeight(700)  
 Image($r('app.media.arrow\_right')).width(6).height(12).fillColor('#FF2C5B37').objectFit(ImageFit.Fill)  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text('签约后可解锁会员权益').fontColor('#2C5B37').fontSize(rvp(12))  
 }  
 .width('100%')  
 .height(rvp(80))  
 .margin({ top: rvp(15) })  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .padding({  
 left: rvp(22),  
 right: rvp(22),  
 top: rvp(18),  
 bottom: rvp(18)  
 })  
 .alignItems(HorizontalAlign.Start)  
 .linearGradient({  
 angle: 116,  
 colors: [  
 ['#FFF1CD', 0],  
 ['#FFEBB5', 1],  
 ]  
 })  
 }  
}

1. **My** 使用 **TimePlan** 组件

import { IColor } from '../model/HomeData'  
import { getWindowTopHeight, rvp } from '../utils/responsive'  
import { PADDING } from '../constants/size'  
  
import ScrollContainer from '../components/ScrollContainer'  
import Bg from '../views/My/Bg'  
import NavBar from '../views/My/NavBar'  
import UserInfo from '../views/My/UserInfo'  
import TimePlan from '../views/My/TimePlan'  
  
@Component  
export default struct My {  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 UserInfo()  
 TimePlan()  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(44) + getWindowTopHeight() })  
 }.width('100%').align(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 10.6. Order 组件功能

1. 引入图片资源

[📎my\_icon\_1.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512745117-509c231e-dc42-4f60-a377-2c3bb8eccae1.svg)[📎my\_icon\_2.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512745241-0b6bc1a8-d2a0-4a82-883d-1cecb5f6f800.svg)[📎my\_icon\_3.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512745438-a5e3134b-df9c-48b8-bafd-08b0532bd27c.svg)[📎my\_icon\_4.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512745576-1a5a08b9-4d57-447f-bffe-843b8e8b5b9f.svg)[📎my\_icon\_5.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512745752-96ab658c-0c3f-4ae9-a27e-d877235c23fb.svg)[📎my\_icon\_6.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512745858-b79931a6-df6a-48ff-9ce4-edea558913e3.svg)[📎my\_icon\_7.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512745974-1181ad2c-9507-4eb1-b1bf-b9eec3cd052d.svg)[📎my\_icon\_8.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512746116-24421e02-0894-4f25-a78c-ce64a43e870d.svg)[📎my\_icon\_9.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512746223-6175aa28-e070-4250-84a1-2ddc6b0d991e.svg)

1. 定义类型文件 **model > NavList**

export interface INavItem {  
 id: number  
 text: string  
 icon: Resource  
}  
  
export type INavList = INavItem[]

1. 搭建 **views > My > Order** 组件

import Card from '../../components/Card'  
import { PADDING } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
import type { INavItem, INavList } from '../../model/NavList'  
  
@Component  
export default struct Order {  
 @State navList1: INavList = [  
 {  
 id: 1,  
 text: '想看',  
 icon: $r('app.media.my\_icon\_1')  
 },  
 {  
 id: 2,  
 text: '足迹',  
 icon: $r('app.media.my\_icon\_2')  
 },  
 {  
 id: 3,  
 text: '约看',  
 icon: $r('app.media.my\_icon\_3')  
 },  
 {  
 id: 4,  
 text: '拼租',  
 icon: $r('app.media.my\_icon\_4')  
 }  
 ]  
 @State navList2: INavList = [  
 {  
 id: 1,  
 text: '合同',  
 icon: $r('app.media.my\_icon\_5')  
 },  
 {  
 id: 2,  
 text: '账单',  
 icon: $r('app.media.my\_icon\_6')  
 },  
 {  
 id: 3,  
 text: '订单',  
 icon: $r('app.media.my\_icon\_7')  
 },  
 {  
 id: 4,  
 text: '评价',  
 icon: $r('app.media.my\_icon\_8')  
 },  
 {  
 id: 5,  
 text: '钱包',  
 icon: $r('app.media.my\_icon\_9')  
 }  
 ]  
  
 build() {  
 Column() {  
 Card({ cardPadding: rvp(PADDING) }) {  
 Row({ space: rvp(38) }) {  
 ForEach(this.navList1, (nav: INavItem) => {  
 Row({ space: rvp(4) }) {  
 Image(nav.icon).width(rvp(16)).height(rvp(16))  
 Text(nav.text).fontSize(rvp(12)).fontColor($r('app.color.black'))  
 }  
 }, (nav: INavItem) => nav.id + '')  
 }.width('100%').justifyContent(FlexAlign.Center)  
  
 Divider().margin({ top: rvp(16) })  
 Row({ space: rvp(39) }) {  
 ForEach(this.navList2, (nav: INavItem) => {  
 Column({ space: rvp(6) }) {  
 Image(nav.icon).width(rvp(16)).height(rvp(16))  
 Text('合同').fontSize(rvp(12)).fontColor($r('app.color.black'))  
 }  
 }, (nav: INavItem) => nav.id + '')  
 }.margin({ top: rvp(12) }).justifyContent(FlexAlign.Center).width('100%')  
 }  
 }.margin({ top: rvp(10) })  
 }  
}

1. **My** 使用 **Order** 组件

import { IColor } from '../model/HomeData'  
import { getWindowTopHeight, rvp } from '../utils/responsive'  
import { PADDING } from '../constants/size'  
  
import ScrollContainer from '../components/ScrollContainer'  
import Bg from '../views/My/Bg'  
import NavBar from '../views/My/NavBar'  
import UserInfo from '../views/My/UserInfo'  
import TimePlan from '../views/My/TimePlan'  
import Order from '../views/My/Order'  
  
@Component  
export default struct My {  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 UserInfo()  
 TimePlan()  
 Order()  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(44) + getWindowTopHeight() })  
 }.width('100%').align(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

## 10.7. PlanList 组件功能

1. 引入图片资源

[📎my\_people.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512746362-97d9a530-73e8-49ca-a8b6-5f2fe6367fe6.png)[📎my\_shenfen.png](https://www.yuque.com/attachments/yuque/0/2024/png/45022700/1723512746498-f6b78756-80fa-46ce-8041-e4740177e444.png)

1. 搭建 **views > My > PlanList** 组件

import Card from '../../components/Card'  
  
import { BORDER\_RADIUS, BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct PlanList {  
 build() {  
 Column() {  
 Card({ cardPadding: rvp(20) }) {  
 Stack() {  
 Column() {  
 Row() {  
 Text('我的家').fontSize(rvp(16)).fontWeight(700)  
 Row() {  
 }  
 .width(rvp(1))  
 .height(rvp(12))  
 .backgroundColor($r('app.color.gray'))  
 .margin({ left: rvp(14), right: rvp(12) })  
  
 Text('感谢你选择谷粒').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 }  
  
 Text('选择谷粒，我们能为你提供').fontSize(rvp(14)).margin({ top: rvp(30) })  
 Row() {  
 Row({ space: rvp(16) }) {  
 Image($r('app.media.my\_shenfen')).width(rvp(38))  
 Column({ space: rvp(2) }) {  
 Text('代办').fontSize(rvp(14))  
 Text('你还未进行实名认证').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 }.alignItems(HorizontalAlign.Start)  
 }  
  
 Image($r('app.media.arrow\_right')).width(rvp(8)).fillColor('#AEAEAE')  
 }  
 .padding(rvp(18))  
 .margin({ top: rvp(10) })  
 .backgroundColor($r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .width('100%')  
 .justifyContent(FlexAlign.SpaceBetween)  
  
 Column() {  
 Row({ space: rvp(10) }) {  
 Text('专属谷粒计划').fontSize(rvp(14))  
 Text('你在谷粒的每一天都很珍贵').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }  
  
 Column({ space: rvp(8) }) {  
 Text('服务折扣').fontSize(rvp(12))  
 Text('每住100天，折扣多一点').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.margin({ top: 30 }).alignItems(HorizontalAlign.Start)  
  
 Column({ space: rvp(8) }) {  
 Text('免费换租').fontSize(rvp(12))  
 Text('城市那么大，你不必讲究').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.margin({ top: rvp(16) }).alignItems(HorizontalAlign.Start)  
 }  
 .width('100%')  
 .margin({ top: rvp(10) })  
 .alignItems(HorizontalAlign.Start)  
 .backgroundColor('#FFF8E4')  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .padding(rvp(PADDING))  
  
 Column() {  
 Row({ space: rvp(10) }) {  
 Text('专属谷粒计划').fontSize(rvp(14))  
 Text('你在谷粒的每一天都很珍贵').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }  
  
 Column({ space: rvp(8) }) {  
 Text('服务折扣').fontSize(rvp(12))  
 Text('每住100天，折扣多一点').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.margin({ top: 30 }).alignItems(HorizontalAlign.Start)  
  
 Column({ space: rvp(8) }) {  
 Text('免费换租').fontSize(rvp(12))  
 Text('城市那么大，你不必讲究').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.margin({ top: rvp(16) }).alignItems(HorizontalAlign.Start)  
 }  
 .width('100%')  
 .margin({ top: rvp(10) })  
 .alignItems(HorizontalAlign.Start)  
 .backgroundColor('#DBFFF0')  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .padding(rvp(PADDING))  
  
 }.alignItems(HorizontalAlign.Start)  
  
 Image($r('app.media.my\_people'))  
 .width(rvp(45))  
 .height(rvp(82))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(8), right: rvp(18) })  
 }.alignContent(Alignment.TopEnd)  
 }  
 }.margin({ top: rvp(10) })  
 }  
}

1. **My** 使用 **PlanList** 组件

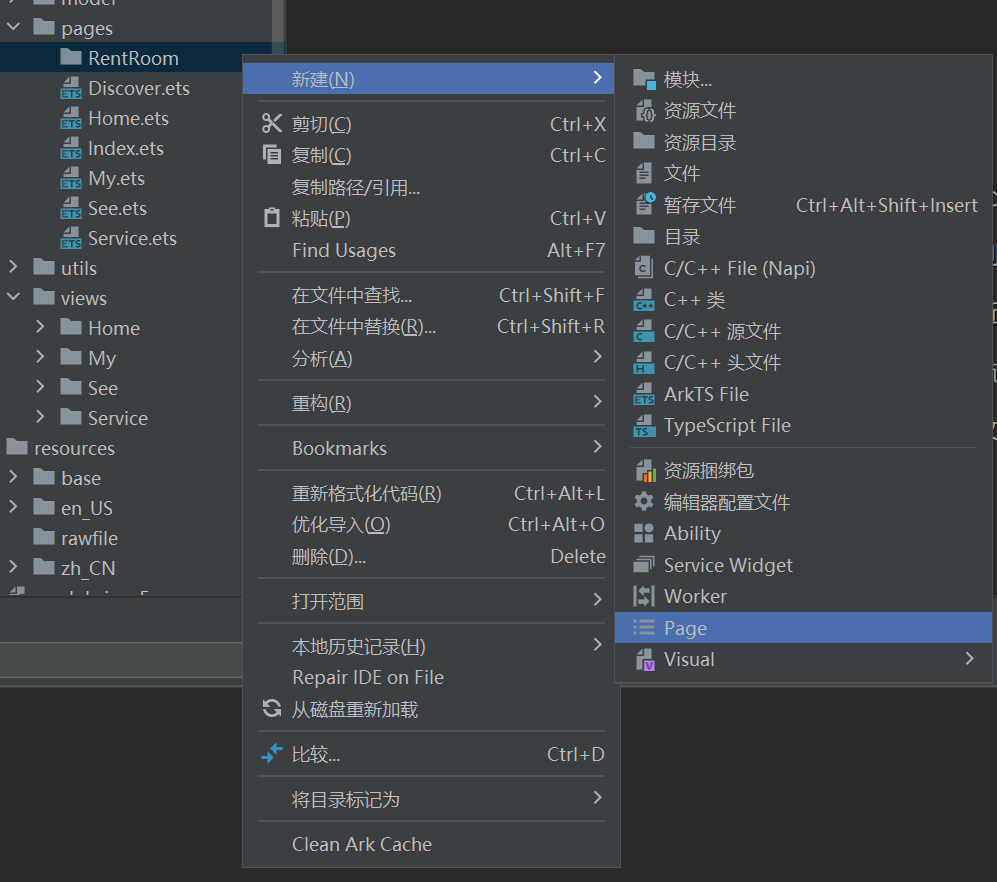
import { IColor } from '../model/HomeData'  
import { getWindowTopHeight, rvp } from '../utils/responsive'  
import { PADDING } from '../constants/size'  
  
import ScrollContainer from '../components/ScrollContainer'  
import Bg from '../views/My/Bg'  
import NavBar from '../views/My/NavBar'  
import UserInfo from '../views/My/UserInfo'  
import TimePlan from '../views/My/TimePlan'  
import Order from '../views/My/Order'  
import PlanList from '../views/My/PlanList'  
  
@Component  
export default struct My {  
 @Builder  
 navBuilder($$: IColor) {  
 NavBar($$)  
 }  
  
 @Builder  
 contentBuilder() {  
 Stack() {  
 Bg()  
 Column() {  
 UserInfo()  
 TimePlan()  
 Order()  
 PlanList()  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(44) + getWindowTopHeight() })  
 }.width('100%').align(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 contentBuilder: () => {  
 this.contentBuilder()  
 },  
 navBuilder: this.navBuilder,  
 })  
 }  
}

# 11. 租房列表页面功能

## 11.1. 点击跳转租房列表

1. 通过新建 Page 方式，新建租房列表基本组件 **pages > RentRoom > RentRoomList**

新建 Page 方式会自动配置路由（在 **resource > base > profile > main\_pages.json**）



@Entry  
@Component  
struct RentRoomList {  
 @State message: string = 'Hello World';  
  
 build() {  
 RelativeContainer() {  
 Text(this.message)  
 .id('RentRoomListHelloWorld')  
 .fontSize(50)  
 .fontWeight(FontWeight.Bold)  
 .alignRules({  
 center: { anchor: '\_\_container\_\_', align: VerticalAlign.Center },  
 middle: { anchor: '\_\_container\_\_', align: HorizontalAlign.Center }  
 })  
 }  
 .height('100%')  
 .width('100%')  
 }  
}

1. 点击 RoomRecommend 的更多推荐，跳转到租房列表页面

import { IRoomRecommendItem, ITagItem, RoomRecommendDataSource } from '../../model/RoomRecommendDataSource'  
import { getRoomRecommendListApi } from '../../api/home'  
  
import { BORDER\_RADIUS, PADDING } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
import router from '@ohos.router'  
  
@Component  
export default struct RoomRecommend {  
 // 获取推荐房源列表  
 @State roomRecommendDataSource: RoomRecommendDataSource = new RoomRecommendDataSource([])  
  
 aboutToAppear() {  
 this.getRoomRecommendList();  
 }  
  
 async getRoomRecommendList() {  
 const roomRecommendList = await getRoomRecommendListApi();  
 this.roomRecommendDataSource = new RoomRecommendDataSource(roomRecommendList)  
 }  
  
 goRentRoomList = () => {  
 router.pushUrl({ url: 'pages/RentRoom/RentRoomList' })  
 }  
  
 // 头部导航  
 @Builder  
 RoomNav() {  
 Row() {  
 Text() {  
 Span('周边房源')  
 Span('推荐').fontColor($r('app.color.primary'))  
 }.fontSize(rvp(18))  
  
 Row({ space: rvp(6) }) {  
 Text('更多推荐').fontColor($r('app.color.gray\_second')).fontSize(rvp(12))  
 Image($r('app.media.arrow\_right')).width(rvp(4)).height(rvp(8))  
 }.onClick(this.goRentRoomList)  
 }  
 .width('100%')  
 .height(rvp(67))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(23) })  
 }  
  
 // 房源列表  
 @Builder  
 RoomList() {  
 if (this.roomRecommendDataSource.roomRecommendList.length) {  
 Grid() {  
 LazyForEach(this.roomRecommendDataSource, (room: IRoomRecommendItem) => {  
 GridItem() {  
 Stack() {  
 Column() {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: BORDER\_RADIUS,  
 topRight: BORDER\_RADIUS  
 })  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(rvp(16)).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(rvp(14)).fontColor('#E03810')  
 }.height(rvp(22))  
  
 Text(room.rentArea + '㎡').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text(room.houseTitle)  
 .fontSize(rvp(14))  
 .margin({ top: rvp(1) })  
 .height(rvp(19))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag: ITagItem, index) => {  
 Text(tag.name)  
 .padding({  
 top: rvp(1),  
 bottom: rvp(1),  
 left: rvp(6),  
 right: rvp(6)  
 })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(rvp(12))  
 }, (tag: ITagItem) => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(91))  
 }.width('100%')  
  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(12))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(rvp(12)).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }.alignContent(Alignment.TopStart).backgroundColor($r('app.color.white'))  
 }.height(rvp(210))  
 }, (room: IRoomRecommendItem) => room.id)  
 }  
 .columnsTemplate('1fr 1fr')  
 .columnsGap(rvp(8))  
 .rowsGap(rvp(8))  
 .width('100%')  
 .padding({  
 left: rvp(PADDING),  
 right: rvp(PADDING),  
 top: rvp(8),  
 bottom: rvp(8)  
 })  
 .backgroundColor($r('app.color.bg\_gray'))  
 .borderRadius(BORDER\_RADIUS)  
 }  
 }  
  
 build() {  
 Column() {  
 this.RoomNav()  
 this.RoomList()  
 }.width('100%')  
 }  
}

## 11.2. 布局分析



## 11.3. 头部导航

1. 引入图片资源

[📎back.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512746628-5f62262b-5357-4e32-bc29-b5a9faac2112.svg)

1. 搭建 **components > NavBar** 公共组件

import router from '@ohos.router';  
import { PADDING } from '../constants/size';  
import { rvp } from '../utils/responsive';  
  
@Component  
export default struct NavBar {  
 @Prop title: string;  
 back = () => {  
 router.back()  
 }  
  
 build() {  
 Stack() {  
 Text(this.title).fontSize(rvp(16)).width('100%').textAlign(TextAlign.Center)  
 Image($r('app.media.back')).width(rvp(24)).height(rvp(24)).onClick(this.back)  
 }.width('100%').height(rvp(44)).padding({ left: rvp(PADDING), right: rvp(PADDING) }).alignContent(Alignment.Start)  
 }  
}

1. **RentRoomList** 组件使用 **NavBar** 组件

import NavBar from '../../components/NavBar'  
import { getWindowTopHeight } from '../../utils/responsive'  
  
@Entry  
@Component  
struct RentRoomList {  
 build() {  
 Stack() {  
 Column() {  
 NavBar({ title: '租房列表' })  
 }.height('100%').padding({ top: getWindowTopHeight() })  
 }.height('100%').alignContent(Alignment.TopStart)  
 }  
}

## 11.4. 搭建头部搜索区 SearchFilter

1. 引入图片资源

[📎arrow\_down\_3.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512746731-a613825b-34a8-4d0a-9642-cf87962fcd69.svg)[📎arrow\_down\_3\_active.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512746903-32551b22-31f2-4bf1-b7dd-528835a216c6.svg)

1. 搭建 **views > RentRoomList > SearchFilter** 组件

import { PADDING } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct SearchFilter {  
 build() {  
 Row() {  
 Row({ space: rvp(6) }) {  
 Text('位置').fontSize(rvp(14))  
 Image($r('app.media.arrow\_down\_3')).width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
 }  
  
 Row({ space: rvp(6) }) {  
 Text('户型').fontSize(rvp(14))  
 Image($r('app.media.arrow\_down\_3')).width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
 }  
  
 Row({ space: rvp(6) }) {  
 Text('租金').fontSize(rvp(14))  
 Image($r('app.media.arrow\_down\_3')).width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
 }  
  
 Row({ space: rvp(6) }) {  
 Text('排序').fontSize(rvp(14))  
 Image($r('app.media.arrow\_down\_3')).width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
 }  
 }  
 .width('100%')  
 .height(rvp(44))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }  
}

1. **RentRoomList** 使用 **SearchFilter** 组件

import { getWindowTopHeight } from '../../utils/responsive'  
  
import NavBar from '../../components/NavBar'  
import SearchFilter from '../../views/RentRoomList/SearchFilter'  
  
@Entry  
@Component  
struct RentRoomList {  
 build() {  
 Stack() {  
 Column() {  
 NavBar({ title: '租房列表' })  
 SearchFilter()  
 }.height('100%').padding({ top: getWindowTopHeight() })  
 }.height('100%').alignContent(Alignment.TopStart)  
 }  
}

## 11.5. 搭建底部租房瀑布流列表组件 WaterfallFlow

1. 定义类型文件

* **model > RentRoomData**

import type { IRoomRecommendItem } from '../model/RoomRecommendDataSource'  
  
export type IPaymentType = "月付" | "季付" | "半年付" | "年付"  
  
export type IOrderBy = "面积" | "价格"  
  
export type IOrderType = "asc" | "desc"  
  
export interface IGetRentRoomListParams {  
 "page": number  
 "limit": number  
 "provinceId"?: string  
 "cityId"?: string  
 "districtId"?: string  
 "minRent"?: string  
 "maxRent"?: string  
 "paymentType"?: IPaymentType  
 "orderBy"?: IOrderBy,  
 "orderType"?: IOrderType  
}  
  
export interface IGetRentRoomListResponse {  
 records: IRentRoomList  
 total: number  
}  
  
export interface IActivity {  
 "title": string  
 "textColor": string  
 "textBackGroundColor": string  
 "#text": string  
 "icon": string  
}  
  
export interface IRentRoomItem extends IRoomRecommendItem {  
 "activity"?: IActivity  
}  
  
export type IRentRoomList = IRentRoomItem[]  
  
  
export class RentRoomListDataSource implements IDataSource {  
 private listeners: DataChangeListener[] = [];  
 data: IRentRoomList = [];  
  
 constructor(list: IRentRoomList) {  
 this.data = list;  
 }  
  
 public totalCount(): number {  
 return this.data.length;  
 }  
  
 public getData(index: number) {  
 return this.data[index];  
 }  
  
 // 增加数据  
 public pushData(item: IRentRoomItem) {  
 this.data.push(item);  
 this.notifyDataAdd(this.data.length - 1)  
 }  
  
 // 增加数据列表  
 public pushDataList(list: IRentRoomList) {  
 list.forEach(item => {  
 this.pushData(item);  
 })  
 }  
  
 // 重新加载数据  
 public reloadDataList(list: IRentRoomList): void {  
 this.data = list  
 this.notifyDataReload()  
 }  
  
 // 该方法为框架侧调用，为LazyForEach组件向其数据源处添加listener监听  
 registerDataChangeListener(listener: DataChangeListener): void {  
 if (this.listeners.indexOf(listener) < 0) {  
 this.listeners.push(listener);  
 }  
 }  
  
 // 该方法为框架侧调用，为对应的LazyForEach组件在数据源处去除listener监听  
 unregisterDataChangeListener(listener: DataChangeListener): void {  
 const pos = this.listeners.indexOf(listener);  
 if (pos >= 0) {  
 this.listeners.splice(pos, 1);  
 }  
 }  
  
 // 通知LazyForEach组件需要重载所有子组件  
 notifyDataReload(): void {  
 this.listeners.forEach(listener => {  
 listener.onDataReloaded();  
 })  
 }  
  
 // 通知LazyForEach组件需要在index对应索引处添加子组件  
 notifyDataAdd(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataAdd(index);  
 })  
 }  
  
 // 通知LazyForEach组件在index对应索引处数据有变化，需要重建该子组件  
 notifyDataChange(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataChange(index);  
 })  
 }  
  
 // 通知LazyForEach组件需要在index对应索引处删除该子组件  
 notifyDataDelete(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataDelete(index);  
 })  
 }  
}

1. 定义接口函数 **api > rentRoom**

import { http } from '../utils/http';  
import type { IGetRentRoomListParams, IGetRentRoomListResponse } from '../model/RentRoomData';  
  
// 获取租房列表数据  
export const getRentRoomListApi = (params: IGetRentRoomListParams): Promise<IGetRentRoomListResponse> => {  
 return http.get<IGetRentRoomListResponse>('/house/search', params)  
}

1. 定义基本组件 **views > RentRoomList > WaterfallFlow**

@Component  
export default struct WaterfallFlow {  
 build() {  
 Column() {  
 Text("WaterfallFlow");  
 }  
 }  
}

import { getWindowTopHeight, rvp } from '../../utils/responsive'  
  
import NavBar from '../../components/NavBar'  
import SearchFilter from '../../views/RentRoomList/SearchFilter'  
import WaterfallFlow from '../../views/RentRoomList/WaterfallFlow'  
  
@Entry  
@Component  
struct RentRoomList {  
 build() {  
 Stack() {  
 Column() {  
 WaterfallFlow()  
 }.padding({ top: getWindowTopHeight() + rvp(44 + 44) })  
  
 Column() {  
 NavBar({ title: '租房列表' })  
 SearchFilter()  
 }.height('100%').padding({ top: getWindowTopHeight() })  
  
 }.height('100%').alignContent(Alignment.TopStart)  
 }  
}

1. **WaterfallFlow** 组件发送请求，更新数据

这里只做最简单的请求需求，后续完成复杂搜索功能

import { getRentRoomListApi } from '../../api/rentRoom';  
import { RentRoomListDataSource } from '../../model/RentRoomData';  
  
@Component  
export default struct WaterfallFlow {  
 @State rootRoomListDataSource: RentRoomListDataSource = new RentRoomListDataSource()  
 @State total: number = 0  
  
 aboutToAppear() {  
 this.getRentRoomList();  
 }  
  
 async getRentRoomList() {  
 const res = await getRentRoomListApi({  
 page: 1,  
 limit: 10  
 })  
 this.rootRoomListDataSource.pushDataList(res.records)  
 this.total = res.total  
 }  
  
 build() {  
 Column() {  
 Text("WaterfallFlow");  
 }  
 }  
}

1. 完成 **WaterfallFlow** 布局样式

import { getRentRoomListApi } from '../../api/rentRoom';  
import { IRentRoomItem, RentRoomListDataSource } from '../../model/RentRoomData';  
import { ITagItem } from '../../model/RoomRecommendDataSource';  
  
import { BORDER\_RADIUS } from '../../constants/size';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct WaterfallFlow {  
 @State roomRoomListDataSource: RentRoomListDataSource = new RentRoomListDataSource([])  
 @State total: number = 0  
  
 aboutToAppear() {  
 this.getRentRoomList();  
 }  
  
 async getRentRoomList() {  
 const res = await getRentRoomListApi({  
 page: 1,  
 limit: 10  
 })  
 this.roomRoomListDataSource = new RentRoomListDataSource(res.records)  
 this.total = res.total  
 }  
  
 build() {  
 if (this.roomRoomListDataSource.totalCount()) {  
 WaterFlow() {  
 LazyForEach(this.roomRoomListDataSource, (room: IRentRoomItem) => {  
 FlowItem() {  
 Stack() {  
 Column() {  
 if (room.activity) {  
 Column() {  
 Row() {  
 Image(room.activity.icon).width(rvp(26))  
 Text(room.activity.title).fontSize(rvp(14)).fontWeight(700)  
 }  
  
 Text(room.activity.title)  
 .fontColor(room.activity.textColor)  
 .backgroundColor(room.activity.textBackGroundColor)  
 .padding({  
 top: rvp(4),  
 bottom: rvp(4),  
 left: rvp(8),  
 right: rvp(8)  
 })  
 .borderRadius(rvp(21))  
 .fontSize(rvp(10))  
 .margin({ top: rvp(3) })  
 Image(room.housePicture)  
 .width('100%')  
 .height(rvp(170))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(5) })  
 }.padding({ left: rvp(10), right: rvp(10), top: rvp(10) }).alignItems(HorizontalAlign.Start)  
 } else {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: rvp(BORDER\_RADIUS),  
 topRight: rvp(BORDER\_RADIUS)  
 })  
 }  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(rvp(14)).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(rvp(12)).fontColor('#E03810')  
 }.height(rvp(22))  
  
 Text(room.rentArea + '㎡').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text(room.houseTitle)  
 .fontSize(rvp(12))  
 .margin({ top: rvp(1) })  
 .height(rvp(19))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag: ITagItem, index) => {  
 Text(tag.name)  
 .padding({  
 top: rvp(1),  
 bottom: rvp(1),  
 left: rvp(6),  
 right: rvp(6),  
 })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(rvp(10))  
 }, (tag: ITagItem) => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(90))  
 }.width('100%').alignItems(HorizontalAlign.Start)  
  
 if (!room.activity) {  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(120))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(rvp(10)).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }  
 }  
 .alignContent(Alignment.TopStart)  
 .backgroundColor($r('app.color.white'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .height('100%')  
 .width(rvp(165))  
 }.height(room.activity ? rvp(320) : rvp(210))  
 }, (room: IRentRoomItem) => room.id)  
 }  
 .columnsTemplate("1fr 1fr")  
 .columnsGap(rvp(10))  
 .rowsGap(rvp(10))  
 .backgroundColor($r('app.color.bg\_gray'))  
 .width('100%')  
 .layoutWeight(1)  
 .layoutDirection(FlexDirection.Column)  
 .padding({ left: rvp(10), right: rvp(10), top: rvp(10) })  
 }  
 }  
}

## 11.6. 上拉加载更多数据

1. 绑定滚动到底部事件

import { getRentRoomListApi } from '../../api/rentRoom';  
import { IGetRentRoomListParams, IRentRoomItem, RentRoomListDataSource } from '../../model/RentRoomData';  
import { ITagItem } from '../../model/RoomRecommendDataSource';  
  
import { BORDER\_RADIUS } from '../../constants/size';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct WaterfallFlow {  
 @State roomRoomListDataSource: RentRoomListDataSource = new RentRoomListDataSource([])  
 @State total: number = 0  
  
 aboutToAppear() {  
 this.getRentRoomList();  
 }  
  
 async getRentRoomList() {  
 const res = await getRentRoomListApi({ page: 1, limit: 10 })  
 this.roomRoomListDataSource = new RentRoomListDataSource(res.records)  
 this.total = res.total  
 }  
  
 // 2. 定义回调函数  
 loadMore = () => {  
 }  
  
 build() {  
 if (this.roomRoomListDataSource.totalCount()) {  
 WaterFlow() {  
 LazyForEach(this.roomRoomListDataSource, (room: IRentRoomItem) => {  
 FlowItem() {  
 Stack() {  
 Column() {  
 if (room.activity) {  
 Column() {  
 Row() {  
 Image(room.activity.icon).width(rvp(26))  
 Text(room.activity.title).fontSize(rvp(14)).fontWeight(700)  
 }  
  
 Text(room.activity.title)  
 .fontColor(room.activity.textColor)  
 .backgroundColor(room.activity.textBackGroundColor)  
 .padding({  
 top: rvp(4),  
 bottom: rvp(4),  
 left: rvp(8),  
 right: rvp(8)  
 })  
 .borderRadius(rvp(21))  
 .fontSize(rvp(10))  
 .margin({ top: rvp(3) })  
 Image(room.housePicture)  
 .width('100%')  
 .height(rvp(170))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(5) })  
 }.padding({ left: rvp(10), right: rvp(10), top: rvp(10) }).alignItems(HorizontalAlign.Start)  
 } else {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: rvp(BORDER\_RADIUS),  
 topRight: rvp(BORDER\_RADIUS)  
 })  
 }  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(rvp(14)).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(rvp(12)).fontColor('#E03810')  
 }.height(rvp(22))  
  
 Text(room.rentArea + '㎡').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text(room.houseTitle)  
 .fontSize(rvp(12))  
 .margin({ top: rvp(1) })  
 .height(rvp(19))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag: ITagItem, index) => {  
 Text(tag.name)  
 .padding({  
 top: rvp(1),  
 bottom: rvp(1),  
 left: rvp(6),  
 right: rvp(6),  
 })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(rvp(10))  
 }, (tag: ITagItem) => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(90))  
 }.width('100%').alignItems(HorizontalAlign.Start)  
  
 if (!room.activity) {  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(120))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(rvp(10)).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }  
 }  
 .alignContent(Alignment.TopStart)  
 .backgroundColor($r('app.color.white'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .height('100%')  
 .width(rvp(165))  
 }.height(room.activity ? rvp(320) : rvp(210))  
 }, (room: IRentRoomItem) => room.id)  
 }  
 .columnsTemplate("1fr 1fr")  
 .columnsGap(rvp(10))  
 .rowsGap(rvp(10))  
 .backgroundColor($r('app.color.bg\_gray'))  
 .width('100%')  
 .layoutWeight(1)  
 .layoutDirection(FlexDirection.Column)  
 .padding({ left: rvp(10), right: rvp(10), top: rvp(10) })  
 .onReachEnd(this.loadMore) // 1. 绑定触底事件  
 }  
 }  
}

1. 加载更多数据

import { getRentRoomListApi } from '../../api/rentRoom';  
import { IGetRentRoomListParams, IRentRoomItem, RentRoomListDataSource } from '../../model/RentRoomData';  
import { ITagItem } from '../../model/RoomRecommendDataSource';  
  
import { BORDER\_RADIUS } from '../../constants/size';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct WaterfallFlow {  
 @State roomRoomListDataSource: RentRoomListDataSource = new RentRoomListDataSource([])  
 @State total: number = 0  
 // 1. 定义搜索条件  
 @State searchOptions: IGetRentRoomListParams = {  
 "page": 1,  
 "limit": 10,  
 "provinceCode": "",  
 "cityCode": "",  
 "districtCode": "",  
 "minRent": "",  
 "maxRent": "",  
 // "paymentType": "",  
 // "orderBy": "",  
 // "orderType": ""  
 }  
  
 aboutToAppear() {  
 this.getRentRoomList();  
 }  
  
 async getRentRoomList() {  
 const res = await getRentRoomListApi(this.searchOptions)  
 this.roomRoomListDataSource = new RentRoomListDataSource(res.records)  
 this.total = res.total  
 }  
 // 2. 加载更多  
 loadMore = () => {  
 this.searchOptions.page++;  
 this.getRentRoomList()  
 }  
  
 build() {  
 if (this.roomRoomListDataSource.totalCount()) {  
 WaterFlow() {  
 LazyForEach(this.roomRoomListDataSource, (room: IRentRoomItem) => {  
 FlowItem() {  
 Stack() {  
 Column() {  
 if (room.activity) {  
 Column() {  
 Row() {  
 Image(room.activity.icon).width(rvp(26))  
 Text(room.activity.title).fontSize(rvp(14)).fontWeight(700)  
 }  
  
 Text(room.activity.title)  
 .fontColor(room.activity.textColor)  
 .backgroundColor(room.activity.textBackGroundColor)  
 .padding({  
 top: rvp(4),  
 bottom: rvp(4),  
 left: rvp(8),  
 right: rvp(8)  
 })  
 .borderRadius(rvp(21))  
 .fontSize(rvp(10))  
 .margin({ top: rvp(3) })  
 Image(room.housePicture)  
 .width('100%')  
 .height(rvp(170))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(5) })  
 }.padding({ left: rvp(10), right: rvp(10), top: rvp(10) }).alignItems(HorizontalAlign.Start)  
 } else {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: rvp(BORDER\_RADIUS),  
 topRight: rvp(BORDER\_RADIUS)  
 })  
 }  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(rvp(14)).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(rvp(12)).fontColor('#E03810')  
 }.height(rvp(22))  
  
 Text(room.rentArea + '㎡').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text(room.houseTitle)  
 .fontSize(rvp(12))  
 .margin({ top: rvp(1) })  
 .height(rvp(19))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag: ITagItem, index) => {  
 Text(tag.name)  
 .padding({  
 top: rvp(1),  
 bottom: rvp(1),  
 left: rvp(6),  
 right: rvp(6),  
 })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(rvp(10))  
 }, (tag: ITagItem) => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(90))  
 }.width('100%').alignItems(HorizontalAlign.Start)  
  
 if (!room.activity) {  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(120))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(rvp(10)).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }  
 }  
 .alignContent(Alignment.TopStart)  
 .backgroundColor($r('app.color.white'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .height('100%')  
 .width(rvp(165))  
 }.height(room.activity ? rvp(320) : rvp(210))  
 }, (room: IRentRoomItem) => room.id)  
 }  
 .columnsTemplate("1fr 1fr")  
 .columnsGap(rvp(10))  
 .rowsGap(rvp(10))  
 .backgroundColor($r('app.color.bg\_gray'))  
 .width('100%')  
 .layoutWeight(1)  
 .layoutDirection(FlexDirection.Column)  
 .padding({ left: rvp(10), right: rvp(10), top: rvp(10) })  
 .onReachEnd(this.loadMore)  
 }  
 }  
}

1. 添加 loading 状态

import { getRentRoomListApi } from '../../api/rentRoom';  
import { IGetRentRoomListParams, IRentRoomItem, RentRoomListDataSource } from '../../model/RentRoomData';  
import { ITagItem } from '../../model/RoomRecommendDataSource';  
  
import { BORDER\_RADIUS } from '../../constants/size';  
import { rvp } from '../../utils/responsive';  
  
@Component  
export default struct WaterfallFlow {  
 @State roomRoomListDataSource: RentRoomListDataSource = new RentRoomListDataSource([])  
 @State total: number = 0  
 @State searchOptions: IGetRentRoomListParams = {  
 "page": 1,  
 "limit": 10,  
 "provinceCode": "",  
 "cityCode": "",  
 "districtCode": "",  
 "minRent": "",  
 "maxRent": "",  
 // "paymentType": "",  
 // "orderBy": "",  
 // "orderType": ""  
 }  
  
 aboutToAppear() {  
 this.getRentRoomList();  
 }  
  
 async getRentRoomList() {  
 const res = await getRentRoomListApi(this.searchOptions)  
 this.roomRoomListDataSource.pushDataList(res.records)  
 this.total = res.total  
 }  
  
 loadMore = () => {  
 console.log(1111 + '')  
 this.searchOptions.page++;  
 this.getRentRoomList()  
 }  
  
 @Builder  
 footer() {  
 Row({ space: rvp(5) }) {  
 LoadingProgress().width(rvp(20)).height(rvp(20)).color($r('app.color.gray'))  
 Text('加载数据中...').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 }.width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
  
 build() {  
 WaterFlow({  
 footer: () => {  
 this.footer()  
 }  
 }) {  
 LazyForEach(this.roomRoomListDataSource, (room: IRentRoomItem) => {  
 FlowItem() {  
 Stack() {  
 Column() {  
 if (room.activity) {  
 Column() {  
 Row() {  
 Image(room.activity.icon).width(rvp(26))  
 Text(room.activity.title).fontSize(rvp(14)).fontWeight(700)  
 }  
  
 Text(room.activity.title)  
 .fontColor(room.activity.textColor)  
 .backgroundColor(room.activity.textBackGroundColor)  
 .padding({  
 top: rvp(4),  
 bottom: rvp(4),  
 left: rvp(8),  
 right: rvp(8)  
 })  
 .borderRadius(rvp(21))  
 .fontSize(rvp(10))  
 .margin({ top: rvp(3) })  
 Image(room.housePicture)  
 .width('100%')  
 .height(rvp(170))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(5) })  
 }.padding({ left: rvp(10), right: rvp(10), top: rvp(10) }).alignItems(HorizontalAlign.Start)  
 } else {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: rvp(BORDER\_RADIUS),  
 topRight: rvp(BORDER\_RADIUS)  
 })  
 }  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(rvp(14)).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(rvp(12)).fontColor('#E03810')  
 }.height(rvp(22))  
  
 Text(room.rentArea + '㎡').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text(room.houseTitle)  
 .fontSize(rvp(12))  
 .margin({ top: rvp(1) })  
 .height(rvp(19))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag: ITagItem, index) => {  
 Text(tag.name)  
 .padding({  
 top: rvp(1),  
 bottom: rvp(1),  
 left: rvp(6),  
 right: rvp(6),  
 })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(rvp(10))  
 }, (tag: ITagItem) => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(90))  
 }.width('100%').alignItems(HorizontalAlign.Start)  
  
 if (!room.activity) {  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(120))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(rvp(10)).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }  
 }  
 .alignContent(Alignment.TopStart)  
 .backgroundColor($r('app.color.white'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .height('100%')  
 .width(rvp(165))  
 }.height(room.activity ? rvp(320) : rvp(210))  
 }, (room: IRentRoomItem) => room.id)  
 }  
 .columnsTemplate("1fr 1fr")  
 .columnsGap(rvp(10))  
 .rowsGap(rvp(10))  
 .backgroundColor($r('app.color.bg\_gray'))  
 .width('100%')  
 .layoutWeight(1)  
 .layoutDirection(FlexDirection.Column)  
 .padding({ left: rvp(10), right: rvp(10), top: rvp(10) })  
 .onReachEnd(this.loadMore)  
 }  
}

## 11.7. 搜索基本切换显示功能

import { BORDER\_RADIUS\_S } from '../../constants/size'  
import { rvp } from '../../utils/responsive'  
  
@Component  
export default struct SearchFilter {  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
 @State currentSearchType: string = ''  
  
 chooseSearchType(searchType: string) {  
 this.currentSearchType = searchType  
 }  
  
 @Builder  
 LocationRender() {  
 Text('LocationRender')  
 }  
  
 @Builder  
 RentPriceRender() {  
 Text('RentPriceRender')  
 }  
  
 @Builder  
 PaymentTypeRender() {  
 Text('PaymentTypeRender')  
 }  
  
 @Builder  
 SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item: string) => {  
 Row({ space: rvp(6) }) {  
 Text(item)  
 .fontSize(rvp(12))  
 .fontColor($r('app.color.black'))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
 Image($r('app.media.arrow\_down\_3')).imageStyles()  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(() => {  
 this.chooseSearchType(item)  
 })  
 }, (item: string) => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
@Extend(Image)  
function imageStyles() {  
 .width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
}  
  
@Extend(Button)  
function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(rvp(12))  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}  
  
@Extend(Text)  
function textStyles(active: boolean) {  
 .fontSize(rvp(12))  
 .fontColor(active ? $r('app.color.white') : $r('app.color.gray\_second'))  
 .width(rvp(94))  
 .height(rvp(26))  
 .backgroundColor(active ? '#67C0A8' : $r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .textAlign(TextAlign.Center)  
}

## 11.8. 省市区搜索功能

1. 定义数据类型

import type { IRoomRecommendItem } from '../model/RoomRecommendDataSource'  
  
export type IPaymentType = "月付" | "季付" | "半年付" | "年付"  
  
export type IOrderBy = "面积" | "价格"  
  
export type IOrderType = "asc" | "desc"  
  
export interface IGetRentRoomListParams {  
 "page": number  
 "limit": number  
 "provinceCode"?: string  
 "cityCode"?: string  
 "districtCode"?: string  
 "minRent"?: string  
 "maxRent"?: string  
 "paymentType"?: IPaymentType  
 "orderBy"?: IOrderBy,  
 "orderType"?: IOrderType  
}  
  
export interface IGetRentRoomListResponse {  
 records: IRentRoomList  
 total: number  
}  
  
export interface IActivity {  
 "title": string  
 "textColor": string  
 "textBackGroundColor": string  
 "#text": string  
 "icon": string  
}  
  
export interface IRentRoomItem extends IRoomRecommendItem {  
 "activity"?: IActivity  
}  
  
export type IRentRoomList = IRentRoomItem[]  
  
  
export class RentRoomListDataSource implements IDataSource {  
 data: IRentRoomList = [];  
 private listeners: DataChangeListener[] = [];  
  
 constructor(list: IRentRoomList) {  
 this.pushDataList(list);  
 }  
  
 public totalCount(): number {  
 return this.data.length;  
 }  
  
 public getData(index: number) {  
 return this.data[index];  
 }  
  
 // 增加数据  
 public pushData(item: IRentRoomItem) {  
 this.data.push(item);  
 this.notifyDataAdd(this.data.length - 1);  
 }  
  
 // 增加数据列表  
 public pushDataList(list: IRentRoomList) {  
 list.forEach(item => {  
 this.pushData(item);  
 })  
 }  
  
 // 重新加载数据  
 public reloadDataList(list: IRentRoomList): void {  
 this.data = list  
 this.notifyDataReload()  
 }  
  
 // 该方法为框架侧调用，为LazyForEach组件向其数据源处添加listener监听  
 registerDataChangeListener(listener: DataChangeListener): void {  
 if (this.listeners.indexOf(listener) < 0) {  
 this.listeners.push(listener);  
 }  
 }  
  
 // 该方法为框架侧调用，为对应的LazyForEach组件在数据源处去除listener监听  
 unregisterDataChangeListener(listener: DataChangeListener): void {  
 const pos = this.listeners.indexOf(listener);  
 if (pos >= 0) {  
 this.listeners.splice(pos, 1);  
 }  
 }  
  
 // 通知LazyForEach组件需要重载所有子组件  
 notifyDataReload(): void {  
 this.listeners.forEach(listener => {  
 listener.onDataReloaded();  
 })  
 }  
  
 // 通知LazyForEach组件需要在index对应索引处添加子组件  
 notifyDataAdd(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataAdd(index);  
 })  
 }  
  
 // 通知LazyForEach组件在index对应索引处数据有变化，需要重建该子组件  
 notifyDataChange(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataChange(index);  
 })  
 }  
  
 // 通知LazyForEach组件需要在index对应索引处删除该子组件  
 notifyDataDelete(index: number): void {  
 this.listeners.forEach(listener => {  
 listener.onDataDelete(index);  
 })  
 }  
}  
  
export interface IProvinceItem {  
 code: string;  
 name: string;  
}  
  
export type IProvinceList = IProvinceItem[]  
  
export interface ICityItem {  
 "code": string  
 "name": string  
 "provinceCode": string  
}  
  
export type ICityList = ICityItem[]  
  
export interface IDistrictItem {  
 "code": string  
 "name": string  
 "cityCode": string  
 "provinceCode": string  
}  
  
export type IDistrictList = IDistrictItem[]  
  
export interface GetCityListParams {  
 provinceCode: string  
}  
  
export interface GetDistrictListParams {  
 cityCode: string  
}

1. 定义获取省市区数据的函数

* 接口函数 api > rentRoom

import { http } from '../utils/http';  
import type {  
 GetCityListParams,  
 GetDistrictListParams,  
 ICityList,  
 IDistrictList,  
 IGetRentRoomListParams,  
 IGetRentRoomListResponse,  
 IProvinceList  
} from '../model/RentRoomData';  
  
// 获取租房列表数据  
export const getRentRoomListApi = (params: IGetRentRoomListParams): Promise<IGetRentRoomListResponse> => {  
 return http.get<IGetRentRoomListResponse>('/house/search', params)  
}  
  
// 获取省份数据  
export const getProvinceListApi = (): Promise<IProvinceList> => {  
 return http.get<IProvinceList>('/province')  
}  
  
// 获取城市数据  
export const getCityListApi = (provinceCode: string): Promise<ICityList> => {  
 const params: GetCityListParams = { provinceCode }  
 return http.get<ICityList>('/city', params)  
}  
  
// 获取区县数据  
export const getDistrictListApi = (cityCode: string): Promise<IDistrictList> => {  
 const params: GetDistrictListParams = { cityCode }  
 return http.get<IDistrictList>('/area', params)  
}

* 组件定义发送请求函数

import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
import { BORDER\_RADIUS\_S } from '../../constants/size'  
import { ICityList, IDistrictList, IProvinceList } from '../../model/RentRoomData'  
import { rvp } from '../../utils/responsive'  
  
  
@Component  
export default struct SearchFilter {  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
 @State currentSearchType: string = ''  
 @State provinceList: IProvinceList = [] // 省  
 @State cityList: ICityList = [] // 市  
 @State districtList: IDistrictList = [] // 区  
 @State provinceCode: string = '' // 省code  
 @State cityCode: string = '' // 市code  
 @State districtCode: string = '' // 区code  
  
 chooseSearchType(searchType: string) {  
 this.currentSearchType = searchType  
 }  
  
 // 获取省市区  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.provinceList = provinceList  
 this.provinceCode = provinceList[0].code;  
 this.getCityList()  
 }  
  
 async getCityList() {  
 const cityList = await getCityListApi(this.provinceCode)  
 this.cityList = cityList;  
 this.cityCode = cityList[0].code;  
 this.getDistrictList()  
 }  
  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.cityCode)  
 this.districtList = districtList  
 this.districtCode = districtList[0].code;  
 }  
  
 @Builder  
 LocationRender() {  
 Text('LocationRender')  
 }  
  
 @Builder  
 RentPriceRender() {  
 Text('RentPriceRender')  
 }  
  
 @Builder  
 PaymentTypeRender() {  
 Text('PaymentTypeRender')  
 }  
  
 @Builder  
 SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item: string) => {  
 Row({ space: rvp(6) }) {  
 Text(item)  
 .fontSize(rvp(12))  
 .fontColor($r('app.color.black'))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
 Image($r('app.media.arrow\_down\_3')).imageStyles()  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(() => {  
 this.chooseSearchType(item)  
 })  
 }, (item: string) => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
@Extend(Image)  
function imageStyles() {  
 .width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
}  
  
@Extend(Button)  
function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(rvp(12))  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}  
  
@Extend(Text)  
function textStyles(active: boolean) {  
 .fontSize(rvp(12))  
 .fontColor(active ? $r('app.color.white') : $r('app.color.gray\_second'))  
 .width(rvp(94))  
 .height(rvp(26))  
 .backgroundColor(active ? '#67C0A8' : $r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .textAlign(TextAlign.Center)  
}

1. 完成初始化数据展示

import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
import { BORDER\_RADIUS\_S } from '../../constants/size'  
import {  
 ICityItem,  
 ICityList,  
 IDistrictItem,  
 IDistrictList,  
 IProvinceItem,  
 IProvinceList  
} from '../../model/RentRoomData'  
import { rvp } from '../../utils/responsive'  
  
  
@Component  
export default struct SearchFilter {  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
 @State currentSearchType: string = ''  
 @State provinceList: IProvinceList = [] // 省  
 @State cityList: ICityList = [] // 市  
 @State districtList: IDistrictList = [] // 区  
 @State provinceCode: string = '' // 省code  
 @State cityCode: string = '' // 市code  
 @State districtCode: string = '' // 区code  
  
 chooseSearchType(searchType: string) {  
 console.log(searchType)  
 this.currentSearchType = searchType  
 if (searchType === '地区') {  
 // 发送请求，获取数据展示  
 this.getProvinceList()  
 }  
 }  
  
 // 获取省市区  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.provinceList = provinceList  
 this.provinceCode = provinceList[0].code;  
 this.getCityList()  
 }  
  
 async getCityList() {  
 const cityList = await getCityListApi(this.provinceCode)  
 this.cityList = cityList;  
 this.cityCode = cityList[0].code;  
 this.getDistrictList()  
 }  
  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.cityCode)  
 this.districtList = districtList  
 this.districtCode = districtList[0].code;  
 }  
  
 @Builder  
 LocationRender() {  
 Row() {  
 List() {  
 ForEach(this.provinceList, (item: IProvinceItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.provinceCode ? $r('app.color.primary') :  
 $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }  
 }, (item: IProvinceItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor($r('app.color.bg\_gray')).scrollBar(BarState.Off)  
  
 List() {  
 ForEach(this.cityList, (item: ICityItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.cityCode ? $r('app.color.primary') : $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }  
 }, (item: ICityItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#EBEBEB').scrollBar(BarState.Off)  
  
 List() {  
 ForEach(this.districtList, (item: IDistrictItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.districtCode ? $r('app.color.primary') :  
 $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }  
 }, (item: IDistrictItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#E6E6E6').scrollBar(BarState.Off)  
 }.height(rvp(240)).width('100%').backgroundColor($r('app.color.white'))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second'))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)')  
 }  
  
 @Builder  
 RentPriceRender() {  
 Text('RentPriceRender')  
 }  
  
 @Builder  
 PaymentTypeRender() {  
 Text('PaymentTypeRender')  
 }  
  
 @Builder  
 SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item: string) => {  
 Row({ space: rvp(6) }) {  
 Text(item)  
 .fontSize(rvp(12))  
 .fontColor($r('app.color.black'))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
 Image($r('app.media.arrow\_down\_3')).imageStyles()  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(() => {  
 this.chooseSearchType(item)  
 })  
 }, (item: string) => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
@Extend(Image)  
function imageStyles() {  
 .width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
}  
  
@Extend(Button)  
function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(rvp(12))  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}

1. 省市区三级列表数据动态请求展示

import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
import { BORDER\_RADIUS\_S } from '../../constants/size'  
import {  
 ICityItem,  
 ICityList,  
 IDistrictItem,  
 IDistrictList,  
 IProvinceItem,  
 IProvinceList  
} from '../../model/RentRoomData'  
import { rvp } from '../../utils/responsive'  
  
  
@Component  
export default struct SearchFilter {  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
 @State currentSearchType: string = ''  
 @State provinceList: IProvinceList = [] // 省  
 @State cityList: ICityList = [] // 市  
 @State districtList: IDistrictList = [] // 区  
 @State provinceCode: string = '' // 省code  
 @State cityCode: string = '' // 市code  
 @State districtCode: string = '' // 区code  
  
 chooseSearchType(searchType: string) {  
 console.log(searchType)  
 this.currentSearchType = searchType  
 if (searchType === '地区') {  
 // 发送请求，获取数据展示  
 this.getProvinceList()  
 }  
 }  
  
 // 获取省市区  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.provinceList = provinceList  
 this.provinceCode = provinceList[0].code;  
 this.getCityList()  
 }  
  
 async getCityList() {  
 const cityList = await getCityListApi(this.provinceCode)  
 this.cityList = cityList;  
 this.cityCode = cityList[0].code;  
 this.getDistrictList()  
 }  
  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.cityCode)  
 this.districtList = districtList  
 this.districtCode = districtList[0].code;  
 }  
  
 @Builder  
 LocationRender() {  
 Row() {  
 List() {  
 ForEach(this.provinceList, (item: IProvinceItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.provinceCode ? $r('app.color.primary') :  
 $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }.onClick(() => {  
 this.provinceCode = item.code;  
 this.getCityList()  
 })  
 }, (item: IProvinceItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor($r('app.color.bg\_gray')).scrollBar(BarState.Off)  
  
 List() {  
 ForEach(this.cityList, (item: ICityItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.cityCode ? $r('app.color.primary') : $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }.onClick(() => {  
 this.cityCode = item.code;  
 this.getDistrictList()  
 })  
 }, (item: ICityItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#EBEBEB').scrollBar(BarState.Off)  
  
 List() {  
 ForEach(this.districtList, (item: IDistrictItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.districtCode ? $r('app.color.primary') :  
 $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }.onClick(() => {  
 this.districtCode = item.code;  
 })  
 }, (item: IDistrictItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#E6E6E6').scrollBar(BarState.Off)  
 }.height(rvp(240)).width('100%').backgroundColor($r('app.color.white'))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second'))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)')  
 }  
  
 @Builder  
 RentPriceRender() {  
 Text('RentPriceRender')  
 }  
  
 @Builder  
 PaymentTypeRender() {  
 Text('PaymentTypeRender')  
 }  
  
 @Builder  
 SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item: string) => {  
 Row({ space: rvp(6) }) {  
 Text(item)  
 .fontSize(rvp(12))  
 .fontColor(this.currentSearchType === item ? '#67C0A8' : $r('app.color.black'))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
  
 if (this.currentSearchType === item) {  
 Image($r('app.media.arrow\_down\_3\_active'))  
 .width(rvp(12)).height(rvp(12)).objectFit(ImageFit.Fill)  
 } else {  
 Image($r('app.media.arrow\_down\_3'))  
 .width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
 }  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(() => {  
 this.chooseSearchType(item)  
 })  
 }, (item: string) => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
  
@Extend(Button)  
function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(rvp(12))  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}

1. 完成省市区其他功能

import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
import { BORDER\_RADIUS\_S } from '../../constants/size'  
import {  
 ICityItem,  
 ICityList,  
 IDistrictItem,  
 IDistrictList,  
 IProvinceItem,  
 IProvinceList  
} from '../../model/RentRoomData'  
import { rvp } from '../../utils/responsive'  
  
  
@Component  
export default struct SearchFilter {  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
 @State currentSearchType: string = ''  
 @State provinceList: IProvinceList = [] // 省  
 @State cityList: ICityList = [] // 市  
 @State districtList: IDistrictList = [] // 区  
 @State provinceCode: string = '' // 省code  
 @State cityCode: string = '' // 市code  
 @State districtCode: string = '' // 区code  
  
 chooseSearchType(searchType: string) {  
 console.log(searchType)  
 this.currentSearchType = searchType  
 if (searchType === '地区') {  
 // 发送请求，获取数据展示  
 this.getProvinceList()  
 }  
 }  
  
 // 获取省市区  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.provinceList = provinceList  
 this.provinceCode = this.provinceCode || provinceList[0].code;  
 this.getCityList()  
 }  
  
 async getCityList() {  
 const cityList = await getCityListApi(this.provinceCode)  
 this.cityList = cityList;  
 this.cityCode = this.cityCode || cityList[0].code;  
 this.getDistrictList()  
 }  
  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.cityCode)  
 this.districtList = districtList  
 this.districtCode = this.districtCode || districtList[0].code;  
 }  
  
 // 重置地区  
 resetLocation = () => {  
 this.provinceCode = this.provinceList[0].code;  
 this.cityCode = '';  
 this.districtCode = '';  
 this.getCityList()  
 }  
 hidden = () => {  
 this.currentSearchType = '';  
 }  
  
 @Builder  
 LocationRender() {  
 Row() {  
 List() {  
 ForEach(this.provinceList, (item: IProvinceItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.provinceCode ? $r('app.color.primary') :  
 $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }.onClick(() => {  
 this.provinceCode = item.code;  
 this.getCityList()  
 })  
 }, (item: IProvinceItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor($r('app.color.bg\_gray')).scrollBar(BarState.Off)  
  
 List() {  
 ForEach(this.cityList, (item: ICityItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.cityCode ? $r('app.color.primary') : $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }.onClick(() => {  
 this.cityCode = item.code;  
 this.getDistrictList()  
 })  
 }, (item: ICityItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#EBEBEB').scrollBar(BarState.Off)  
  
 List() {  
 ForEach(this.districtList, (item: IDistrictItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(rvp(12))  
 .fontColor(item.code === this.districtCode ? $r('app.color.primary') :  
 $r('app.color.black'))  
 }  
 .width('100%')  
 .padding({  
 left: rvp(16),  
 right: rvp(16),  
 top: rvp(6),  
 bottom: rvp(6)  
 })  
 }.onClick(() => {  
 this.districtCode = item.code;  
 })  
 }, (item: IDistrictItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#E6E6E6').scrollBar(BarState.Off)  
 }.height(rvp(240)).width('100%').backgroundColor($r('app.color.white'))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second')).onClick(this.resetLocation)  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.hidden)  
 }  
  
 @Builder  
 RentPriceRender() {  
 Text('RentPriceRender')  
 }  
  
 @Builder  
 PaymentTypeRender() {  
 Text('PaymentTypeRender')  
 }  
  
 @Builder  
 SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item: string) => {  
 Row({ space: rvp(6) }) {  
 Text(item)  
 .fontSize(rvp(12))  
 .fontColor(this.currentSearchType === item ? '#67C0A8' : $r('app.color.black'))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
  
 if (this.currentSearchType === item) {  
 Image($r('app.media.arrow\_down\_3\_active'))  
 .width(rvp(12)).height(rvp(12)).objectFit(ImageFit.Fill)  
 } else {  
 Image($r('app.media.arrow\_down\_3'))  
 .width(rvp(6)).height(rvp(3)).objectFit(ImageFit.Fill)  
 }  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(() => {  
 this.chooseSearchType(item)  
 })  
 }, (item: string) => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
  
@Extend(Button)  
function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(rvp(12))  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}

1. 将选中的数据传递给 **WaterFallFlow** 组件，**WaterFallFlow** 组件发送请求更新数据

**SearchFilter** 组件和 **WaterFallFlow** 组件是兄弟关系，不能使用父子组件通信的方案进行通信。

这里我们自己封装自定义事件来进行通信。

* 定义自定义事件函数 **utils > emitter.ts**

ETS 不支持 any 类型，所以我们定义成 TS 文件

type ICallback = (...args: any[]) => void  
  
class Emitter {  
 private callbacks = {}  
  
 on(eventName: string, cb: ICallback) {  
 if (this.callbacks[eventName]) {  
 this.callbacks[eventName].push(cb)  
 } else {  
 this.callbacks[eventName] = [cb]  
 }  
 }  
  
 once(eventName: string, cb: ICallback) {  
 const callback = (...args) => {  
 cb(...args)  
 this.off(eventName, callback);  
 }  
 this.on(eventName, callback);  
 }  
  
 emit(eventName: string, ...args: any[]) {  
 if (!this.callbacks[eventName]) {  
 // 事件不存在  
 throw new Error(`${eventName}自定义事件不存在~`)  
 return  
 }  
  
 this.callbacks[eventName].forEach(cb => cb(...args))  
 }  
  
 off(eventName?: string, cb?: ICallback) {  
 if (!eventName) {  
 // 解绑所有事件  
 this.callbacks = {};  
 return;  
 }  
  
 if (!this.callbacks[eventName]) {  
 // 事件不存在  
 throw new Error(`${eventName}自定义事件不存在~`)  
 return  
 }  
  
 if (!cb) {  
 // 解绑单个事件的所有回调函数  
 delete this.callbacks[eventName]  
 return;  
 }  
  
 // 解绑单个事件的单个回调  
 this.callbacks[eventName] = this.callbacks[eventName].filter(callback => cb !== callback)  
 }  
}  
  
export default Emitter;  
  
export const globalEmitter = new Emitter();

* **WaterFallFlow** 组件绑定自定义事件，接受 **SearchFilter** 组件传递过来的数据并更新 searchOptions, 然后发送请求更新数据

import { getRentRoomListApi } from '../../api/rentRoom';  
import { IGetRentRoomListParams, IRentRoomItem, RentRoomListDataSource } from '../../model/RentRoomData';  
import { ITagItem } from '../../model/RoomRecommendDataSource';  
  
import { BORDER\_RADIUS } from '../../constants/size';  
import { rvp } from '../../utils/responsive';  
import { globalEmitter } from '../../utils/emitter';  
  
@Component  
export default struct WaterfallFlow {  
 @State roomRoomListDataSource: RentRoomListDataSource = new RentRoomListDataSource([])  
 @State total: number = 0  
 @State searchOptions: IGetRentRoomListParams = {  
 "page": 1,  
 "limit": 10,  
 "provinceCode": "",  
 "cityCode": "",  
 "districtCode": "",  
 "minRent": "",  
 "maxRent": "",  
 // "paymentType": "",  
 // "orderBy": "",  
 // "orderType": ""  
 }  
  
 aboutToAppear() {  
 this.getRentRoomList();  
 this.registerGlobalEmitter();  
 }  
  
 registerGlobalEmitter() {  
 globalEmitter.on('searchOptionsChange', async (searchOptions: Partial<IGetRentRoomListParams>) => {  
 this.searchOptions.page = 1;  
 this.searchOptions.limit = 10;  
 if (searchOptions.provinceCode) {  
 this.searchOptions.provinceCode = searchOptions.provinceCode;  
 }  
 if (searchOptions.cityCode) {  
 this.searchOptions.cityCode = searchOptions.cityCode;  
 }  
 if (searchOptions.districtCode) {  
 this.searchOptions.districtCode = searchOptions.districtCode;  
 }  
 this.getRentRoomList()  
 })  
 }  
  
 async getRentRoomList() {  
 const res = await getRentRoomListApi(this.searchOptions)  
 this.roomRoomListDataSource.pushDataList(res.records)  
 this.total = res.total  
 }  
  
 loadMore = () => {  
 console.log(1111 + '')  
 this.searchOptions.page++;  
 this.getRentRoomList()  
 }  
  
 @Builder  
 footer() {  
 Row({ space: rvp(5) }) {  
 LoadingProgress().width(rvp(20)).height(rvp(20)).color($r('app.color.gray'))  
 Text('加载数据中...').fontSize(rvp(12)).fontColor($r('app.color.gray'))  
 }.width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
  
 build() {  
 WaterFlow({  
 footer: () => {  
 this.footer()  
 }  
 }) {  
 LazyForEach(this.roomRoomListDataSource, (room: IRentRoomItem) => {  
 FlowItem() {  
 Stack() {  
 Column() {  
 if (room.activity) {  
 Column() {  
 Row() {  
 Image(room.activity.icon).width(rvp(26))  
 Text(room.activity.title).fontSize(rvp(14)).fontWeight(700)  
 }  
  
 Text(room.activity.title)  
 .fontColor(room.activity.textColor)  
 .backgroundColor(room.activity.textBackGroundColor)  
 .padding({  
 top: rvp(4),  
 bottom: rvp(4),  
 left: rvp(8),  
 right: rvp(8)  
 })  
 .borderRadius(rvp(21))  
 .fontSize(rvp(10))  
 .margin({ top: rvp(3) })  
 Image(room.housePicture)  
 .width('100%')  
 .height(rvp(170))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(5) })  
 }.padding({ left: rvp(10), right: rvp(10), top: rvp(10) }).alignItems(HorizontalAlign.Start)  
 } else {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: rvp(BORDER\_RADIUS),  
 topRight: rvp(BORDER\_RADIUS)  
 })  
 }  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(rvp(14)).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(rvp(12)).fontColor('#E03810')  
 }.height(rvp(22))  
  
 Text(room.rentArea + '㎡').fontSize(rvp(10)).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Text(room.houseTitle)  
 .fontSize(rvp(12))  
 .margin({ top: rvp(1) })  
 .height(rvp(19))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag: ITagItem, index) => {  
 Text(tag.name)  
 .padding({  
 top: rvp(1),  
 bottom: rvp(1),  
 left: rvp(6),  
 right: rvp(6),  
 })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(rvp(10))  
 }, (tag: ITagItem) => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(90))  
 }.width('100%').alignItems(HorizontalAlign.Start)  
  
 if (!room.activity) {  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(120))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(rvp(10)).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }  
 }  
 .alignContent(Alignment.TopStart)  
 .backgroundColor($r('app.color.white'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .height('100%')  
 .width(rvp(165))  
 }.height(room.activity ? rvp(320) : rvp(210))  
 }, (room: IRentRoomItem) => room.id)  
 }  
 .columnsTemplate("1fr 1fr")  
 .columnsGap(rvp(10))  
 .rowsGap(rvp(10))  
 .backgroundColor($r('app.color.bg\_gray'))  
 .width('100%')  
 .layoutWeight(1)  
 .layoutDirection(FlexDirection.Column)  
 .padding({ left: rvp(10), right: rvp(10), top: rvp(10) })  
 .onReachEnd(this.loadMore)  
 }  
}

* **SearchFilter** 组件点击查看房源，触发自定义事件

import { BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import rvp from '../../utils/responsive/responsiveVP'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
import type {  
 ICityItem,  
 ICityList,  
 IDistrictItem,  
 IDistrictList,  
 IPaymentTypeItem,  
 IPaymentTypeList,  
 IProvinceItem,  
 IProvinceList,  
 IRentPriceItem,  
 IRentPriceList,  
 ISortItem,  
 ISortList  
} from '../../model/RentRoomData'  
  
import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
  
import { globalEmitter } from '../../utils/emitter'  
  
@Component  
export default struct SearchFilter {  
  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
  
 @State currentSearchType: string = ''  
  
 // 3. 地区数据  
 @State locationData: any = {  
 provinceList: [],  
 cityList: [],  
 districtList: [],  
 provinceCode: '',  
 cityCode: '',  
 districtCode: '',  
 isSearch: false,  
 searchProvinceCode: '',  
 searchCityCode: '',  
 searchDistrictCode: '',  
 searchDistrictName: ''  
 }  
  
 getSearchTypeName(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName || item  
 }  
 return item  
 }  
 getSearchTypeNameColor(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName ? $r('app.color.primary') : $r('app.color.black')  
 }  
 return item  
 }  
  
 chooseSearchType(searchType: string) {  
 if (this.currentSearchType === searchType) {  
 this.currentSearchType = ''  
 return;  
 }  
   
 // 每个类型需要不同的处理  
 if (searchType === '地区') {  
 this.locationData.isSearch = false  
 this.locationClose()  
 this.getProvinceList()  
 }  
  
 // 更新  
 this.currentSearchType = searchType;  
 }  
 // 获取省市区数据方法  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.locationData.provinceList = provinceList  
 this.locationData.provinceCode = this.locationData.provinceCode || provinceList[0].code;  
 this.getCityList()  
 }  
 async getCityList() {  
 const cityList = await getCityListApi(this.locationData.provinceCode)  
 this.locationData.cityList = cityList;  
 this.locationData.cityCode = this.locationData.cityCode || cityList[0].code;  
 this.getDistrictList()  
 }  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.locationData.cityCode)  
 this.locationData.districtList = districtList  
 this.locationData.districtCode = this.locationData.districtCode || districtList[0].code;  
 }  
  
 handleProvinceChange(provinceCode: string) {  
 this.locationData.provinceCode = provinceCode;  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getCityList()  
 }  
 handleCityChange(cityCode: string) {  
 this.locationData.cityCode = cityCode  
 this.locationData.districtCode = ''  
 this.getDistrictList()  
 }  
 handleDistrictChange(districtCode: string) {  
 this.locationData.districtCode = districtCode;  
 }  
  
 locationSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 provinceCode: this.locationData.provinceCode,  
 cityCode: this.locationData.cityCode,  
 districtCode: this.locationData.districtCode,  
 })  
 this.locationData.isSearch = true;  
 this.locationClose()  
 }  
  
 locationClose() {  
 this.currentSearchType = ''  
 if (this.locationData.isSearch) {  
 this.locationData.searchProvinceCode = this.locationData.provinceCode  
 this.locationData.searchCityCode = this.locationData.cityCode  
 this.locationData.searchDistrictCode = this.locationData.districtCode  
 this.locationData.searchDistrictName = this.locationData.districtList.find(item => item.code === this.locationData.districtCode).name  
 } else {  
 this.locationData.provinceCode = this.locationData.searchProvinceCode  
 this.locationData.cityCode = this.locationData.searchCityCode  
 this.locationData.districtCode = this.locationData.searchDistrictCode  
 }  
 }  
  
 locationReset() {  
 this.locationData.provinceCode = ''  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getProvinceList()  
 }  
  
 @Builder LocationRender() {  
 Row() {  
 List() {  
 ForEach(this.locationData.provinceList, (item: IProvinceItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.provinceCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleProvinceChange.bind(this, item.code))  
 }, (item: IProvinceItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor($r('app.color.bg\_gray'))  
  
 List() {  
 ForEach(this.locationData.cityList, (item: ICityItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.cityCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleCityChange.bind(this, item.code))  
 }, (item: ICityItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#EBEBEB')  
  
 List() {  
 ForEach(this.locationData.districtList, (item: IDistrictItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.districtCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleDistrictChange.bind(this, item.code))  
 }, (item: IDistrictItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#E6E6E6')  
 }.height(rvp(240)).width('100%').backgroundColor($r('app.color.white'))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second')).onClick(this.locationReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.locationSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.locationClose.bind(this))  
 }  
  
 @Builder RentPriceRender() {  
 Text('RentPriceRender')  
 }  
 @Builder PaymentTypeRender() {  
 Text('PaymentTypeRender')  
 }  
 @Builder SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item) => {  
 Row({ space: rvp(6) }) {  
 Text(this.getSearchTypeName(item))  
 .fontSize(sm())  
 .fontColor(this.getSearchTypeNameColor(item))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
 Image($r('app.media.arrow\_down\_3')).imageStyles()  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(this.chooseSearchType.bind(this, item))  
 }, item => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
@Extend(Image) function imageStyles() {  
 .width(rvp(12)).height(rvp(12)).objectFit(ImageFit.Fill)  
}  
  
@Extend(Button) function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(sm())  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}  
  
@Extend(Text) function textStyles(active: boolean) {  
 .fontSize(sm())  
 .fontColor(active ? $r('app.color.white') : $r('app.color.gray\_second'))  
 .width(rvp(94))  
 .height(rvp(26))  
 .backgroundColor(active ? '#67C0A8' : $r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .textAlign(TextAlign.Center)  
}

## 11.9. 租金搜索功能

1. 定义租金列表数据类型

// ...  
export interface IRentPriceItem {  
 id: number;  
 minRent: string;  
 maxRent: string;  
 text: string;  
}  
  
export type IRentPriceList = IRentPriceItem[]

1. 完成静态结构样式

import { BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import rvp from '../../utils/responsive/responsiveVP'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
import type {  
 ICityItem,  
 ICityList,  
 IDistrictItem,  
 IDistrictList,  
 IPaymentTypeItem,  
 IPaymentTypeList,  
 IProvinceItem,  
 IProvinceList,  
 IRentPriceItem,  
 IRentPriceList,  
 ISortItem,  
 ISortList  
} from '../../model/RentRoomData'  
  
import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
  
import { globalEmitter } from '../../utils/emitter'  
  
@Component  
export default struct SearchFilter {  
  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
  
 @State currentSearchType: string = ''  
  
 getSearchTypeName(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName || item  
 }  
 return item  
 }  
 getSearchTypeNameColor(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName ? $r('app.color.primary') : $r('app.color.black')  
 }  
 return item  
 }  
 chooseSearchType(searchType: string) {  
 if (this.currentSearchType === searchType) {  
 this.currentSearchType = ''  
 return;  
 }  
 // 更新  
 this.currentSearchType = searchType;  
 // 每个类型需要不同的处理  
 if (searchType === '地区') {  
 this.locationData.isSearch = false  
 // 发送请求，获取数据展示  
 this.getProvinceList()  
 }  
 }  
  
 // 地区  
 @State locationData: any = {  
 provinceList: [],  
 cityList: [],  
 districtList: [],  
 provinceCode: '',  
 cityCode: '',  
 districtCode: '',  
 isSearch: false,  
 searchProvinceCode: '',  
 searchCityCode: '',  
 searchDistrictCode: '',  
 searchDistrictName: ''  
 }  
  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.locationData.provinceList = provinceList  
 this.locationData.provinceCode = this.locationData.provinceCode || provinceList[0].code;  
 this.getCityList()  
 }  
 async getCityList() {  
 const cityList = await getCityListApi(this.locationData.provinceCode)  
 this.locationData.cityList = cityList;  
 this.locationData.cityCode = this.locationData.cityCode || cityList[0].code;  
 this.getDistrictList()  
 }  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.locationData.cityCode)  
 this.locationData.districtList = districtList  
 this.locationData.districtCode = this.locationData.districtCode || districtList[0].code;  
 }  
 handleProvinceChange(provinceCode: string) {  
 this.locationData.provinceCode = provinceCode;  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getCityList()  
 }  
 handleCityChange(cityCode: string) {  
 this.locationData.cityCode = cityCode  
 this.locationData.districtCode = ''  
 this.getDistrictList()  
 }  
 handleDistrictChange(districtCode: string) {  
 this.locationData.districtCode = districtCode;  
 }  
 locationSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 provinceCode: this.locationData.provinceCode,  
 cityCode: this.locationData.cityCode,  
 districtCode: this.locationData.districtCode,  
 })  
 this.locationData.isSearch = true;  
 this.locationClose()  
 }  
 locationClose() {  
 this.currentSearchType = ''  
 if (this.locationData.isSearch) {  
 this.locationData.searchProvinceCode = this.locationData.provinceCode  
 this.locationData.searchCityCode = this.locationData.cityCode  
 this.locationData.searchDistrictCode = this.locationData.districtCode  
 this.locationData.searchDistrictName = this.locationData.districtList.find(item => item.code === this.locationData.districtCode).name  
 } else {  
 this.locationData.provinceCode = this.locationData.searchProvinceCode  
 this.locationData.cityCode = this.locationData.searchCityCode  
 this.locationData.districtCode = this.locationData.searchDistrictCode  
 }  
 }  
 locationReset() {  
 this.locationData.provinceCode = ''  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getProvinceList()  
 }  
  
 @Builder LocationRender() {  
 Row() {  
 List() {  
 ForEach(this.locationData.provinceList, (item: IProvinceItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.provinceCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleProvinceChange.bind(this, item.code))  
 }, (item: IProvinceItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor($r('app.color.bg\_gray'))  
  
 List() {  
 ForEach(this.locationData.cityList, (item: ICityItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.cityCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleCityChange.bind(this, item.code))  
 }, (item: ICityItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#EBEBEB')  
  
 List() {  
 ForEach(this.locationData.districtList, (item: IDistrictItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.districtCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleDistrictChange.bind(this, item.code))  
 }, (item: IDistrictItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#E6E6E6')  
 }.height(rvp(240)).width('100%').backgroundColor($r('app.color.white'))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second')).onClick(this.locationReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.locationSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.locationClose.bind(this))  
 }  
  
  
 @State rentPriceData: any = {  
 rentPriceList: [  
 {  
 id: 1,  
 minRent: '0',  
 maxRent: '0',  
 text: '不限'  
 },  
 {  
 id: 2,  
 minRent: '0',  
 maxRent: '1500',  
 text: '1500元以下'  
 },  
 {  
 id: 3,  
 minRent: '1500',  
 maxRent: '2500',  
 text: '1500-2500元'  
 },  
 {  
 id: 4,  
 minRent: '2500',  
 maxRent: '3500',  
 text: '2500-3500元'  
 },  
 {  
 id: 5,  
 minRent: '3500',  
 maxRent: '4500',  
 text: '3500-4500元'  
 },  
 {  
 id: 6,  
 minRent: '4500',  
 maxRent: '6000',  
 text: '4500-6000元'  
 },  
 {  
 id: 7,  
 minRent: '6000',  
 maxRent: '8000',  
 text: '6000-8000元'  
 },  
 {  
 id: 8,  
 minRent: '8000',  
 maxRent: '10000',  
 text: '8000-10000元'  
 },  
 {  
 id: 9,  
 minRent: '10000',  
 maxRent: '0',  
 text: '10000元以上'  
 }  
 ]  
 }  
  
 @Builder RentPriceRender() {  
 Column() {  
 Grid() {  
 ForEach(this.rentPriceData.rentPriceList, (rentPrice: IRentPriceItem) => {  
 GridItem() {  
 Text(rentPrice.text)  
 .textStyles(false)  
 }  
 }, rentPrice => rentPrice.id)  
 }  
 .columnsTemplate('1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr 1fr')  
 .height(rvp(194 - 58))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second'))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
 .height(rvp(202))  
 .width('100%')  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)')  
 }  
 @Builder PaymentTypeRender() {  
 Text('PaymentTypeRender')  
 }  
 @Builder SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item) => {  
 Row({ space: rvp(6) }) {  
 Text(this.getSearchTypeName(item))  
 .fontSize(sm())  
 .fontColor(this.getSearchTypeNameColor(item))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
 Image($r('app.media.arrow\_down\_3')).imageStyles()  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(this.chooseSearchType.bind(this, item))  
 }, item => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
@Extend(Image) function imageStyles() {  
 .width(rvp(12)).height(rvp(12)).objectFit(ImageFit.Fill)  
}  
  
@Extend(Button) function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(sm())  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}  
  
@Extend(Text) function textStyles(active: boolean) {  
 .fontSize(sm())  
 .fontColor(active ? $r('app.color.white') : $r('app.color.gray\_second'))  
 .width(rvp(94))  
 .height(rvp(26))  
 .backgroundColor(active ? '#67C0A8' : $r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .textAlign(TextAlign.Center)  
}

1. 完成租金功能

import { BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import rvp from '../../utils/responsive/responsiveVP'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
import type {  
 ICityItem,  
 ICityList,  
 IDistrictItem,  
 IDistrictList,  
 IPaymentTypeItem,  
 IPaymentTypeList,  
 IProvinceItem,  
 IProvinceList,  
 IRentPriceItem,  
 IRentPriceList,  
 ISortItem,  
 ISortList  
} from '../../model/RentRoomData'  
  
import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
  
import { globalEmitter } from '../../utils/emitter'  
  
@Component  
export default struct SearchFilter {  
  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
  
 @State currentSearchType: string = ''  
  
 getSearchTypeName(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName || item  
 } else if (item === '租金') {  
 return this.rentPriceData.searchRentPrice.text || item  
 }  
 return item  
 }  
 getSearchTypeNameColor(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName ? $r('app.color.primary') : $r('app.color.black')  
 } else if (item === '租金') {  
 return this.rentPriceData.searchRentPrice.text ? $r('app.color.primary') : $r('app.color.black')  
 }  
 return item  
 }  
 chooseSearchType(searchType: string) {  
 if (this.currentSearchType === searchType) {  
 this.currentSearchType = ''  
 return;  
 }  
  
 // 每个类型需要不同的处理  
 if (searchType === '地区') {  
 this.locationData.isSearch = false  
 this.locationClose()  
 this.getProvinceList()  
 } else if (searchType === '租金') {  
 this.rentPriceData.isSearch = false  
 this.rentPriceClose()  
 }  
  
 // 更新  
 this.currentSearchType = searchType;  
 }  
  
 // 地区  
 @State locationData: any = {  
 provinceList: [],  
 cityList: [],  
 districtList: [],  
 provinceCode: '',  
 cityCode: '',  
 districtCode: '',  
 isSearch: false,  
 searchProvinceCode: '',  
 searchCityCode: '',  
 searchDistrictCode: '',  
 searchDistrictName: ''  
 }  
  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.locationData.provinceList = provinceList  
 this.locationData.provinceCode = this.locationData.provinceCode || provinceList[0].code;  
 this.getCityList()  
 }  
 async getCityList() {  
 const cityList = await getCityListApi(this.locationData.provinceCode)  
 this.locationData.cityList = cityList;  
 this.locationData.cityCode = this.locationData.cityCode || cityList[0].code;  
 this.getDistrictList()  
 }  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.locationData.cityCode)  
 this.locationData.districtList = districtList  
 this.locationData.districtCode = this.locationData.districtCode || districtList[0].code;  
 }  
 handleProvinceChange(provinceCode: string) {  
 this.locationData.provinceCode = provinceCode;  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getCityList()  
 }  
 handleCityChange(cityCode: string) {  
 this.locationData.cityCode = cityCode  
 this.locationData.districtCode = ''  
 this.getDistrictList()  
 }  
 handleDistrictChange(districtCode: string) {  
 this.locationData.districtCode = districtCode;  
 }  
 locationSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 provinceCode: this.locationData.provinceCode,  
 cityCode: this.locationData.cityCode,  
 districtCode: this.locationData.districtCode,  
 })  
 this.locationData.isSearch = true;  
 this.locationClose()  
 }  
 locationClose() {  
 this.currentSearchType = ''  
 if (this.locationData.isSearch) {  
 this.locationData.searchProvinceCode = this.locationData.provinceCode  
 this.locationData.searchCityCode = this.locationData.cityCode  
 this.locationData.searchDistrictCode = this.locationData.districtCode  
 this.locationData.searchDistrictName = this.locationData.districtList.find(item => item.code === this.locationData.districtCode).name  
 } else {  
 this.locationData.provinceCode = this.locationData.searchProvinceCode  
 this.locationData.cityCode = this.locationData.searchCityCode  
 this.locationData.districtCode = this.locationData.searchDistrictCode  
 }  
 }  
 locationReset() {  
 this.locationData.provinceCode = ''  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getProvinceList()  
 }  
  
 @Builder LocationRender() {  
 Row() {  
 List() {  
 ForEach(this.locationData.provinceList, (item: IProvinceItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.provinceCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleProvinceChange.bind(this, item.code))  
 }, (item: IProvinceItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor($r('app.color.bg\_gray'))  
  
 List() {  
 ForEach(this.locationData.cityList, (item: ICityItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.cityCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleCityChange.bind(this, item.code))  
 }, (item: ICityItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#EBEBEB')  
  
 List() {  
 ForEach(this.locationData.districtList, (item: IDistrictItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.districtCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleDistrictChange.bind(this, item.code))  
 }, (item: IDistrictItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#E6E6E6')  
 }.height(rvp(240)).width('100%').backgroundColor($r('app.color.white'))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second')).onClick(this.locationReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.locationSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.locationClose.bind(this))  
 }  
  
  
 @State rentPriceData: any = {  
 rentPriceList: [  
 {  
 id: 1,  
 minRent: '0',  
 maxRent: '0',  
 text: '不限'  
 },  
 {  
 id: 2,  
 minRent: '0',  
 maxRent: '1500',  
 text: '1500元以下'  
 },  
 {  
 id: 3,  
 minRent: '1500',  
 maxRent: '2500',  
 text: '1500-2500元'  
 },  
 {  
 id: 4,  
 minRent: '2500',  
 maxRent: '3500',  
 text: '2500-3500元'  
 },  
 {  
 id: 5,  
 minRent: '3500',  
 maxRent: '4500',  
 text: '3500-4500元'  
 },  
 {  
 id: 6,  
 minRent: '4500',  
 maxRent: '6000',  
 text: '4500-6000元'  
 },  
 {  
 id: 7,  
 minRent: '6000',  
 maxRent: '8000',  
 text: '6000-8000元'  
 },  
 {  
 id: 8,  
 minRent: '8000',  
 maxRent: '10000',  
 text: '8000-10000元'  
 },  
 {  
 id: 9,  
 minRent: '10000',  
 maxRent: '0',  
 text: '10000元以上'  
 }  
 ],  
 rentPrice: {},  
 isSearch: false,  
 searchRentPrice: {}  
 }  
  
 handleRentPriceChange(rentPrice: IRentPriceItem) {  
 this.rentPriceData.rentPrice = rentPrice  
 }  
  
 rentPriceSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 minRent: this.rentPriceData.rentPrice.minRent,  
 maxRent: this.rentPriceData.rentPrice.maxRent,  
 })  
 this.rentPriceData.isSearch = true;  
 this.rentPriceClose()  
 }  
 rentPriceClose() {  
 this.currentSearchType = ''  
 if (this.rentPriceData.isSearch) {  
 this.rentPriceData.searchRentPrice = this.rentPriceData.rentPrice  
 } else {  
 this.rentPriceData.rentPrice = this.rentPriceData.searchRentPrice  
 }  
 }  
 rentPriceReset() {  
 this.rentPriceData.rentPrice = {}  
 }  
  
 @Builder RentPriceRender() {  
 Column() {  
 Grid() {  
 ForEach(this.rentPriceData.rentPriceList, (rentPrice: IRentPriceItem) => {  
 GridItem() {  
 Text(rentPrice.text)  
 .textStyles(this.rentPriceData.rentPrice.id === rentPrice.id)  
 .onClick(this.handleRentPriceChange.bind(this, rentPrice))  
 }  
 }, rentPrice => rentPrice.id)  
 }  
 .columnsTemplate('1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr 1fr')  
 .height(rvp(194 - 58))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second'))  
 .onClick(this.rentPriceReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.rentPriceSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
 .height(rvp(202))  
 .width('100%')  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.rentPriceClose.bind(this))  
 }  
 @Builder PaymentTypeRender() {  
 Text('PaymentTypeRender')  
 }  
 @Builder SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item) => {  
 Row({ space: rvp(6) }) {  
 Text(this.getSearchTypeName(item))  
 .fontSize(sm())  
 .fontColor(this.getSearchTypeNameColor(item))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
 Image($r('app.media.arrow\_down\_3')).imageStyles()  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(this.chooseSearchType.bind(this, item))  
 }, item => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
@Extend(Image) function imageStyles() {  
 .width(rvp(12)).height(rvp(12)).objectFit(ImageFit.Fill)  
}  
  
@Extend(Button) function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(sm())  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}  
  
@Extend(Text) function textStyles(active: boolean) {  
 .fontSize(sm())  
 .fontColor(active ? $r('app.color.white') : $r('app.color.gray\_second'))  
 .width(rvp(94))  
 .height(rvp(26))  
 .backgroundColor(active ? '#67C0A8' : $r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .textAlign(TextAlign.Center)  
}

## 11.10. 付款方式搜索功能

import { BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import rvp from '../../utils/responsive/responsiveVP'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
import type {  
 ICityItem,  
 ICityList,  
 IDistrictItem,  
 IDistrictList,  
 IPaymentTypeItem,  
 IPaymentTypeList,  
 IProvinceItem,  
 IProvinceList,  
 IRentPriceItem,  
 IRentPriceList,  
 ISortItem,  
 ISortList  
} from '../../model/RentRoomData'  
  
import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
  
import { globalEmitter } from '../../utils/emitter'  
  
@Component  
export default struct SearchFilter {  
  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
  
 @State currentSearchType: string = ''  
  
 getSearchTypeName(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName || item  
 } else if (item === '租金') {  
 return this.rentPriceData.searchRentPrice.text || item  
 }  
 return item  
 }  
 getSearchTypeNameColor(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName ? $r('app.color.primary') : $r('app.color.black')  
 } else if (item === '租金') {  
 return this.rentPriceData.searchRentPrice.text ? $r('app.color.primary') : $r('app.color.black')  
 }  
 return item  
 }  
 chooseSearchType(searchType: string) {  
 if (this.currentSearchType === searchType) {  
 this.currentSearchType = ''  
 return;  
 }  
  
 // 每个类型需要不同的处理  
 if (searchType === '地区') {  
 this.locationData.isSearch = false  
 this.locationClose()  
 this.getProvinceList()  
 } else if (searchType === '租金') {  
 this.rentPriceData.isSearch = false  
 this.rentPriceClose()  
 } else if (searchType === '付款方式') {  
 this.paymentTypeData.isSearch = false  
 this.paymentTypeClose()  
 }  
  
 // 更新  
 this.currentSearchType = searchType;  
 }  
  
 // 地区  
 @State locationData: any = {  
 provinceList: [],  
 cityList: [],  
 districtList: [],  
 provinceCode: '',  
 cityCode: '',  
 districtCode: '',  
 isSearch: false,  
 searchProvinceCode: '',  
 searchCityCode: '',  
 searchDistrictCode: '',  
 searchDistrictName: ''  
 }  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.locationData.provinceList = provinceList  
 this.locationData.provinceCode = this.locationData.provinceCode || provinceList[0].code;  
 this.getCityList()  
 }  
 async getCityList() {  
 const cityList = await getCityListApi(this.locationData.provinceCode)  
 this.locationData.cityList = cityList;  
 this.locationData.cityCode = this.locationData.cityCode || cityList[0].code;  
 this.getDistrictList()  
 }  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.locationData.cityCode)  
 this.locationData.districtList = districtList  
 this.locationData.districtCode = this.locationData.districtCode || districtList[0].code;  
 }  
 handleProvinceChange(provinceCode: string) {  
 this.locationData.provinceCode = provinceCode;  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getCityList()  
 }  
 handleCityChange(cityCode: string) {  
 this.locationData.cityCode = cityCode  
 this.locationData.districtCode = ''  
 this.getDistrictList()  
 }  
 handleDistrictChange(districtCode: string) {  
 this.locationData.districtCode = districtCode;  
 }  
 locationSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 provinceCode: this.locationData.provinceCode,  
 cityCode: this.locationData.cityCode,  
 districtCode: this.locationData.districtCode,  
 })  
 this.locationData.isSearch = true;  
 this.locationClose()  
 }  
 locationClose() {  
 this.currentSearchType = ''  
 if (this.locationData.isSearch) {  
 this.locationData.searchProvinceCode = this.locationData.provinceCode  
 this.locationData.searchCityCode = this.locationData.cityCode  
 this.locationData.searchDistrictCode = this.locationData.districtCode  
 this.locationData.searchDistrictName = this.locationData.districtList.find(item => item.code === this.locationData.districtCode).name  
 } else {  
 this.locationData.provinceCode = this.locationData.searchProvinceCode  
 this.locationData.cityCode = this.locationData.searchCityCode  
 this.locationData.districtCode = this.locationData.searchDistrictCode  
 }  
 }  
 locationReset() {  
 this.locationData.provinceCode = ''  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getProvinceList()  
 }  
 @Builder LocationRender() {  
 Row() {  
 List() {  
 ForEach(this.locationData.provinceList, (item: IProvinceItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.provinceCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleProvinceChange.bind(this, item.code))  
 }, (item: IProvinceItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor($r('app.color.bg\_gray'))  
  
 List() {  
 ForEach(this.locationData.cityList, (item: ICityItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.cityCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleCityChange.bind(this, item.code))  
 }, (item: ICityItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#EBEBEB')  
  
 List() {  
 ForEach(this.locationData.districtList, (item: IDistrictItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.districtCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleDistrictChange.bind(this, item.code))  
 }, (item: IDistrictItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#E6E6E6')  
 }.height(rvp(240)).width('100%').backgroundColor($r('app.color.white'))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second')).onClick(this.locationReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.locationSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.locationClose.bind(this))  
 }  
  
 // 租金  
 @State rentPriceData: any = {  
 rentPriceList: [  
 {  
 id: 1,  
 minRent: '0',  
 maxRent: '0',  
 text: '不限'  
 },  
 {  
 id: 2,  
 minRent: '0',  
 maxRent: '1500',  
 text: '1500元以下'  
 },  
 {  
 id: 3,  
 minRent: '1500',  
 maxRent: '2500',  
 text: '1500-2500元'  
 },  
 {  
 id: 4,  
 minRent: '2500',  
 maxRent: '3500',  
 text: '2500-3500元'  
 },  
 {  
 id: 5,  
 minRent: '3500',  
 maxRent: '4500',  
 text: '3500-4500元'  
 },  
 {  
 id: 6,  
 minRent: '4500',  
 maxRent: '6000',  
 text: '4500-6000元'  
 },  
 {  
 id: 7,  
 minRent: '6000',  
 maxRent: '8000',  
 text: '6000-8000元'  
 },  
 {  
 id: 8,  
 minRent: '8000',  
 maxRent: '10000',  
 text: '8000-10000元'  
 },  
 {  
 id: 9,  
 minRent: '10000',  
 maxRent: '0',  
 text: '10000元以上'  
 }  
 ],  
 rentPrice: {},  
 isSearch: false,  
 searchRentPrice: {}  
 }  
 handleRentPriceChange(rentPrice: IRentPriceItem) {  
 this.rentPriceData.rentPrice = rentPrice  
 }  
 rentPriceSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 minRent: this.rentPriceData.rentPrice.minRent,  
 maxRent: this.rentPriceData.rentPrice.maxRent,  
 })  
 this.rentPriceData.isSearch = true;  
 this.rentPriceClose()  
 }  
 rentPriceClose() {  
 this.currentSearchType = ''  
 if (this.rentPriceData.isSearch) {  
 this.rentPriceData.searchRentPrice = this.rentPriceData.rentPrice  
 } else {  
 this.rentPriceData.rentPrice = this.rentPriceData.searchRentPrice  
 }  
 }  
 rentPriceReset() {  
 this.rentPriceData.rentPrice = {}  
 }  
 @Builder RentPriceRender() {  
 Column() {  
 Grid() {  
 ForEach(this.rentPriceData.rentPriceList, (rentPrice: IRentPriceItem) => {  
 GridItem() {  
 Text(rentPrice.text)  
 .textStyles(this.rentPriceData.rentPrice.id === rentPrice.id)  
 .onClick(this.handleRentPriceChange.bind(this, rentPrice))  
 }  
 }, rentPrice => rentPrice.id)  
 }  
 .columnsTemplate('1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr 1fr')  
 .height(rvp(194 - 58))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second'))  
 .onClick(this.rentPriceReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.rentPriceSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
 .height(rvp(202))  
 .width('100%')  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.rentPriceClose.bind(this))  
 }  
  
 // 付款方式  
 @State paymentTypeData: any = {  
 paymentTypeList: ['月付', '季付', '半年付', '年付'],  
 isSearch: false,  
 paymentType: '',  
 searchPaymentType: ''  
 }  
 handlePaymentTypeChange(paymentType: string) {  
 this.paymentTypeData.paymentType = paymentType  
 }  
 paymentTypeSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 paymentType: this.paymentTypeData.searchPaymentType  
 })  
 this.paymentTypeData.isSearch = true;  
 this.paymentTypeClose()  
 }  
 paymentTypeClose() {  
 this.currentSearchType = ''  
 if (this.paymentTypeData.isSearch) {  
 this.paymentTypeData.searchPaymentType = this.paymentTypeData.paymentType  
 } else {  
 this.paymentTypeData.paymentType = this.paymentTypeData.searchPaymentType  
 }  
 }  
 paymentTypeReset() {  
 this.paymentTypeData.paymentType = ''  
 }  
 @Builder PaymentTypeRender() {  
 Column() {  
 Grid() {  
 ForEach(this.paymentTypeData.paymentTypeList, (paymentType) => {  
 GridItem() {  
 Text(paymentType)  
 .textStyles(this.paymentTypeData.paymentType === paymentType)  
 .onClick(this.handlePaymentTypeChange.bind(this, paymentType))  
 }  
 }, paymentType => paymentType)  
 }  
 .columnsTemplate('1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr 1fr')  
 .height(rvp(100))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second'))  
 .onClick(this.paymentTypeReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.paymentTypeSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
 .height(rvp(158))  
 .width('100%')  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.paymentTypeClose.bind(this))  
 }  
 @Builder SortRender() {  
 Text('SortRender')  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item) => {  
 Row({ space: rvp(6) }) {  
 Text(this.getSearchTypeName(item))  
 .fontSize(sm())  
 .fontColor(this.getSearchTypeNameColor(item))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
 Image($r('app.media.arrow\_down\_3')).imageStyles()  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(this.chooseSearchType.bind(this, item))  
 }, item => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
@Extend(Image) function imageStyles() {  
 .width(rvp(12)).height(rvp(12)).objectFit(ImageFit.Fill)  
}  
  
@Extend(Button) function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(sm())  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}  
  
@Extend(Text) function textStyles(active: boolean) {  
 .fontSize(sm())  
 .fontColor(active ? $r('app.color.white') : $r('app.color.gray\_second'))  
 .width(rvp(94))  
 .height(rvp(26))  
 .backgroundColor(active ? '#67C0A8' : $r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .textAlign(TextAlign.Center)  
}

## 11.11. 排序搜索功能

1. 引入图片资源

[📎area.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512747114-983d522e-5e4c-4a0f-b463-759dfb87dde3.svg)[📎rent.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512747409-11963ecb-fe1d-43bd-9e1d-9b2fb2e4249a.svg)[📎sort.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512747628-ad776cfa-2471-48d3-b1b8-756c5e7f4f57.svg)

1. 定义排序列表数据类型

// ...  
export interface ISortItem {  
 id: number;  
 icon: Resource;  
 text: string;  
 orderBy: IOrderBy;  
 orderType: IOrderType;  
}  
  
export type ISortList = ISortItem[]

1. 完成功能

import { BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import rvp from '../../utils/responsive/responsiveVP'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
import type {  
 ICityItem,  
 ICityList,  
 IDistrictItem,  
 IDistrictList,  
 IPaymentTypeItem,  
 IPaymentTypeList,  
 IProvinceItem,  
 IProvinceList,  
 IRentPriceItem,  
 IRentPriceList,  
 ISortItem,  
 ISortList  
} from '../../model/RentRoomData'  
  
import { getCityListApi, getDistrictListApi, getProvinceListApi } from '../../api/rentRoom'  
  
import { globalEmitter } from '../../utils/emitter'  
  
@Component  
export default struct SearchFilter {  
  
 @State searchTypeList: string[] = ['地区', '租金', '付款方式', '排序']  
  
 @State currentSearchType: string = ''  
  
 getSearchTypeName(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName || item  
 } else if (item === '租金') {  
 return this.rentPriceData.searchRentPrice.text || item  
 } else if (item === '付款方式') {  
 return this.paymentTypeData.searchPaymentType || item  
 } else if (item === '排序') {  
 return this.sortData.sort.text || item  
 }  
 }  
 getSearchTypeNameColor(item: string) {  
 if (item === '地区') {  
 return this.locationData.searchDistrictName ? $r('app.color.primary') : $r('app.color.black')  
 } else if (item === '租金') {  
 return this.rentPriceData.searchRentPrice.text ? $r('app.color.primary') : $r('app.color.black')  
 } else if (item === '付款方式') {  
 return this.paymentTypeData.searchPaymentType ? $r('app.color.primary') : $r('app.color.black')  
 } else if (item === '排序') {  
 return this.sortData.sort.text && this.sortData.sort.text !== '综合排序' ? $r('app.color.primary') : $r('app.color.black')  
 }  
 }  
 chooseSearchType(searchType: string) {  
 if (this.currentSearchType === searchType) {  
 this.currentSearchType = ''  
 return;  
 }  
  
 // 每个类型需要不同的处理  
 if (searchType === '地区') {  
 this.locationData.isSearch = false  
 this.locationClose()  
 this.getProvinceList()  
 } else if (searchType === '租金') {  
 this.rentPriceData.isSearch = false  
 this.rentPriceClose()  
 } else if (searchType === '付款方式') {  
 this.paymentTypeData.isSearch = false  
 this.paymentTypeClose()  
 } else if (searchType === '排序') {  
 this.sortClose()  
 }  
  
 // 更新  
 this.currentSearchType = searchType;  
 }  
  
 // 地区  
 @State locationData: any = {  
 provinceList: [],  
 cityList: [],  
 districtList: [],  
 provinceCode: '',  
 cityCode: '',  
 districtCode: '',  
 isSearch: false,  
 searchProvinceCode: '',  
 searchCityCode: '',  
 searchDistrictCode: '',  
 searchDistrictName: ''  
 }  
 async getProvinceList() {  
 const provinceList = await getProvinceListApi()  
 this.locationData.provinceList = provinceList  
 this.locationData.provinceCode = this.locationData.provinceCode || provinceList[0].code;  
 this.getCityList()  
 }  
 async getCityList() {  
 const cityList = await getCityListApi(this.locationData.provinceCode)  
 this.locationData.cityList = cityList;  
 this.locationData.cityCode = this.locationData.cityCode || cityList[0].code;  
 this.getDistrictList()  
 }  
 async getDistrictList() {  
 const districtList = await getDistrictListApi(this.locationData.cityCode)  
 this.locationData.districtList = districtList  
 this.locationData.districtCode = this.locationData.districtCode || districtList[0].code;  
 }  
 handleProvinceChange(provinceCode: string) {  
 this.locationData.provinceCode = provinceCode;  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getCityList()  
 }  
 handleCityChange(cityCode: string) {  
 this.locationData.cityCode = cityCode  
 this.locationData.districtCode = ''  
 this.getDistrictList()  
 }  
 handleDistrictChange(districtCode: string) {  
 this.locationData.districtCode = districtCode;  
 }  
 locationSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 provinceCode: this.locationData.provinceCode,  
 cityCode: this.locationData.cityCode,  
 districtCode: this.locationData.districtCode,  
 })  
 this.locationData.isSearch = true;  
 this.locationClose()  
 }  
 locationClose() {  
 this.currentSearchType = ''  
 if (this.locationData.isSearch) {  
 this.locationData.searchProvinceCode = this.locationData.provinceCode  
 this.locationData.searchCityCode = this.locationData.cityCode  
 this.locationData.searchDistrictCode = this.locationData.districtCode  
 this.locationData.searchDistrictName = this.locationData.districtList.find(item => item.code === this.locationData.districtCode).name  
 } else {  
 this.locationData.provinceCode = this.locationData.searchProvinceCode  
 this.locationData.cityCode = this.locationData.searchCityCode  
 this.locationData.districtCode = this.locationData.searchDistrictCode  
 }  
 }  
 locationReset() {  
 this.locationData.provinceCode = ''  
 this.locationData.cityCode = ''  
 this.locationData.districtCode = ''  
 this.getProvinceList()  
 }  
 @Builder LocationRender() {  
 Row() {  
 List() {  
 ForEach(this.locationData.provinceList, (item: IProvinceItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.provinceCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleProvinceChange.bind(this, item.code))  
 }, (item: IProvinceItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor($r('app.color.bg\_gray'))  
  
 List() {  
 ForEach(this.locationData.cityList, (item: ICityItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.cityCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleCityChange.bind(this, item.code))  
 }, (item: ICityItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#EBEBEB')  
  
 List() {  
 ForEach(this.locationData.districtList, (item: IDistrictItem) => {  
 ListItem() {  
 Row() {  
 Text(item.name)  
 .fontSize(sm())  
 .fontColor(item.code === this.locationData.districtCode ? $r('app.color.primary') : $r('app.color.black'))  
 }.width('100%').padding({ left: rvp(16), right: rvp(16), top: rvp(6), bottom: rvp(6) })  
 }.onClick(this.handleDistrictChange.bind(this, item.code))  
 }, (item: IDistrictItem) => item.code)  
 }.width(rvp(120)).height('100%').backgroundColor('#E6E6E6')  
 }.height(rvp(240)).width('100%').backgroundColor($r('app.color.white'))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second')).onClick(this.locationReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.locationSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.locationClose.bind(this))  
 }  
  
 // 租金  
 @State rentPriceData: any = {  
 rentPriceList: [  
 {  
 id: 1,  
 minRent: '0',  
 maxRent: '0',  
 text: '不限'  
 },  
 {  
 id: 2,  
 minRent: '0',  
 maxRent: '1500',  
 text: '1500元以下'  
 },  
 {  
 id: 3,  
 minRent: '1500',  
 maxRent: '2500',  
 text: '1500-2500元'  
 },  
 {  
 id: 4,  
 minRent: '2500',  
 maxRent: '3500',  
 text: '2500-3500元'  
 },  
 {  
 id: 5,  
 minRent: '3500',  
 maxRent: '4500',  
 text: '3500-4500元'  
 },  
 {  
 id: 6,  
 minRent: '4500',  
 maxRent: '6000',  
 text: '4500-6000元'  
 },  
 {  
 id: 7,  
 minRent: '6000',  
 maxRent: '8000',  
 text: '6000-8000元'  
 },  
 {  
 id: 8,  
 minRent: '8000',  
 maxRent: '10000',  
 text: '8000-10000元'  
 },  
 {  
 id: 9,  
 minRent: '10000',  
 maxRent: '0',  
 text: '10000元以上'  
 }  
 ],  
 rentPrice: {},  
 isSearch: false,  
 searchRentPrice: {}  
 }  
 handleRentPriceChange(rentPrice: IRentPriceItem) {  
 this.rentPriceData.rentPrice = rentPrice  
 }  
 rentPriceSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 minRent: this.rentPriceData.rentPrice.minRent,  
 maxRent: this.rentPriceData.rentPrice.maxRent,  
 })  
 this.rentPriceData.isSearch = true;  
 this.rentPriceClose()  
 }  
 rentPriceClose() {  
 this.currentSearchType = ''  
 if (this.rentPriceData.isSearch) {  
 this.rentPriceData.searchRentPrice = this.rentPriceData.rentPrice  
 } else {  
 this.rentPriceData.rentPrice = this.rentPriceData.searchRentPrice  
 }  
 }  
 rentPriceReset() {  
 this.rentPriceData.rentPrice = {}  
 }  
 @Builder RentPriceRender() {  
 Column() {  
 Grid() {  
 ForEach(this.rentPriceData.rentPriceList, (rentPrice: IRentPriceItem) => {  
 GridItem() {  
 Text(rentPrice.text)  
 .textStyles(this.rentPriceData.rentPrice.id === rentPrice.id)  
 .onClick(this.handleRentPriceChange.bind(this, rentPrice))  
 }  
 }, rentPrice => rentPrice.id)  
 }  
 .columnsTemplate('1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr 1fr')  
 .height(rvp(194 - 58))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second'))  
 .onClick(this.rentPriceReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.rentPriceSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
 .height(rvp(202))  
 .width('100%')  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.rentPriceClose.bind(this))  
 }  
  
 // 付款方式  
 @State paymentTypeData: any = {  
 paymentTypeList: ['月付', '季付', '半年付', '年付'],  
 isSearch: false,  
 paymentType: '',  
 searchPaymentType: ''  
 }  
 handlePaymentTypeChange(paymentType: string) {  
 this.paymentTypeData.paymentType = paymentType  
 }  
 paymentTypeSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 paymentType: this.paymentTypeData.searchPaymentType  
 })  
 this.paymentTypeData.isSearch = true;  
 this.paymentTypeClose()  
 }  
 paymentTypeClose() {  
 this.currentSearchType = ''  
 if (this.paymentTypeData.isSearch) {  
 this.paymentTypeData.searchPaymentType = this.paymentTypeData.paymentType  
 } else {  
 this.paymentTypeData.paymentType = this.paymentTypeData.searchPaymentType  
 }  
 }  
 paymentTypeReset() {  
 this.paymentTypeData.paymentType = ''  
 }  
 @Builder PaymentTypeRender() {  
 Column() {  
 Grid() {  
 ForEach(this.paymentTypeData.paymentTypeList, (paymentType) => {  
 GridItem() {  
 Text(paymentType)  
 .textStyles(this.paymentTypeData.paymentType === paymentType)  
 .onClick(this.handlePaymentTypeChange.bind(this, paymentType))  
 }  
 }, paymentType => paymentType)  
 }  
 .columnsTemplate('1fr 1fr 1fr')  
 .rowsTemplate('1fr 1fr 1fr')  
 .height(rvp(100))  
  
 Row({ space: rvp(22) }) {  
 Button('重置')  
 .buttonStyles(rvp(95), '#F0F0F0', $r('app.color.gray\_second'))  
 .onClick(this.paymentTypeReset.bind(this))  
 Button('查看房源')  
 .buttonStyles(rvp(211), '#67C0A8', $r('app.color.white'))  
 .onClick(this.paymentTypeSearch.bind(this))  
 }.height(rvp(58)).width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
 .height(rvp(158))  
 .width('100%')  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.paymentTypeClose.bind(this))  
 }  
  
 // 排序  
 @State sortData: any = {  
 sortList: [  
 {  
 id: 1,  
 icon: $r('app.media.sort'),  
 text: '综合排序',  
 orderBy: '',  
 orderType: ''  
 },  
 {  
 id: 2,  
 icon: $r('app.media.rent'),  
 text: '价格从低到高',  
 orderBy: 'price',  
 orderType: 'asc'  
 },  
 {  
 id: 3,  
 icon: $r('app.media.rent'),  
 text: '价格从高到低',  
 orderBy: 'price',  
 orderType: 'desc'  
 },  
 {  
 id: 4,  
 icon: $r('app.media.area'),  
 text: '面积从大到小',  
 orderBy: 'area',  
 orderType: 'desc'  
 },  
 {  
 id: 5,  
 icon: $r('app.media.area'),  
 text: '面积从小到大',  
 orderBy: 'area',  
 orderType: 'asc'  
 },  
 ],  
 sort: {  
 id: 1,  
 icon: $r('app.media.sort'),  
 text: '综合排序',  
 orderBy: '',  
 orderType: ''  
 },  
 }  
 handleSortChange(sort: ISortItem) {  
 this.sortData.sort = sort  
 this.sortSearch()  
 }  
 sortSearch() {  
 globalEmitter.emit('searchOptionsChange', {  
 orderBy: this.sortData.sort.orderBy,  
 orderType: this.sortData.sort.orderType,  
 })  
 this.sortClose()  
 }  
 sortClose() {  
 this.currentSearchType = ''  
 }  
 sortReset() {  
 this.sortData.sort = {}  
 }  
 @Builder SortRender() {  
 Column() {  
 Column() {  
 ForEach(this.sortData.sortList, (sort: ISortItem) => {  
 Row({ space: rvp(8) }) {  
 Image(sort.icon).width(rvp(16))  
 Text(sort.text)  
 .fontSize(sm())  
 .fontColor(this.sortData.sort.id === sort.id ? '#67C0A8' : $r('app.color.gray\_second'))  
 }.height(rvp(34)).onClick(this.handleSortChange.bind(this, sort))  
 }, sort => sort.id)  
 }.padding({ left: rvp(20), top: rvp(8), bottom: rvp(16) }).alignItems(HorizontalAlign.Start)  
 }  
 .width('100%')  
 .backgroundColor($r('app.color.white'))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 .alignItems(HorizontalAlign.Start)  
  
  
 Row() {  
 }.width('100%').layoutWeight(1).backgroundColor('rgba(0, 0, 0, 0.7)').onClick(this.sortClose.bind(this))  
 }  
  
 build() {  
 Column() {  
 Row() {  
 ForEach(this.searchTypeList, (item) => {  
 Row({ space: rvp(6) }) {  
 Text(this.getSearchTypeName(item))  
 .fontSize(sm())  
 .fontColor(this.getSearchTypeNameColor(item))  
 .maxLines(1)  
 .textOverflow({ overflow: TextOverflow.Ellipsis })  
 .constraintSize({  
 maxWidth: rvp(60),  
 })  
 Image($r('app.media.arrow\_down\_3')).imageStyles()  
 }.width('25%').height('100%').justifyContent(FlexAlign.Center).onClick(this.chooseSearchType.bind(this, item))  
 }, item => item)  
 }.width('100%').height(rvp(44)).justifyContent(FlexAlign.SpaceBetween)  
  
 if (this.currentSearchType === '地区') {  
 this.LocationRender()  
 }  
 if (this.currentSearchType === '租金') {  
 this.RentPriceRender()  
 }  
 if (this.currentSearchType === '付款方式') {  
 this.PaymentTypeRender()  
 }  
 if (this.currentSearchType === '排序') {  
 this.SortRender()  
 }  
 }  
 }  
}  
  
@Extend(Image) function imageStyles() {  
 .width(rvp(12)).height(rvp(12)).objectFit(ImageFit.Fill)  
}  
  
@Extend(Button) function buttonStyles(width: number, bgColor: string | Resource, fontColor: string | Resource) {  
 .type(ButtonType.Normal)  
 .width(width)  
 .height(rvp(26))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .fontSize(sm())  
 .backgroundColor(bgColor)  
 .fontColor(fontColor)  
}  
  
@Extend(Text) function textStyles(active: boolean) {  
 .fontSize(sm())  
 .fontColor(active ? $r('app.color.white') : $r('app.color.gray\_second'))  
 .width(rvp(94))  
 .height(rvp(26))  
 .backgroundColor(active ? '#67C0A8' : $r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .textAlign(TextAlign.Center)  
}

## 11.12. 空状态

1. 定义公共组件 **components > Empty**

import { sm } from '../utils/responsive/responsiveFontSize';  
import rvp from '../utils/responsive/responsiveVP';  
  
@Component  
export default struct Empty {  
 @Prop title: string;  
  
 build() {  
 Column({ space: rvp(19) }) {  
 Image($r('app.media.blank')).width(rvp(150)).height(rvp(140)).objectFit(ImageFit.Fill)  
 Text(this.title).fontSize(sm())  
 }.width('100%')  
 }  
}

1. **WaterfallFlow** 组件使用 **Empty** 组件

import Empty from '../../components/Empty';  
  
if (this.total > 0) {  
 WaterFlow() {  
 LazyForEach(this.roomRoomListDataSource, (room: IRentRoomItem) => {  
 FlowItem() {  
 Stack() {  
 Column() {  
 if (room.activity) {  
 Column() {  
 Row() {  
 Image(room.activity.icon).width(rvp(26))  
 Text(room.activity.title).fontSize(md()).fontWeight(700)  
 }  
  
 Text(room.activity.title)  
 .fontColor(room.activity.textColor)  
 .backgroundColor(room.activity.textBackGroundColor)  
 .padding({ top: rvp(4), bottom: rvp(4), left: rvp(8), right: rvp(8) })  
 .borderRadius(rvp(21))  
 .fontSize(xs())  
 .margin({ top: rvp(3) })  
 Image(room.housePicture)  
 .width('100%')  
 .height(rvp(170))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(5) })  
 }.padding({ left: rvp(10), right: rvp(10), top: rvp(10)}).alignItems(HorizontalAlign.Start)  
 } else {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: rvp(BORDER\_RADIUS),  
 topRight: rvp(BORDER\_RADIUS)  
 })  
 }  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(md()).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(sm()).fontColor('#E03810')  
 }.height(rvp(22))  
 Text(room.rentArea + '㎡').fontSize(xs()).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
 Text(room.houseTitle).fontSize(sm()).margin({ top: rvp(1) }).height(rvp(19)).maxLines(1).textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag, index) => {  
 Text(tag.name)  
 .padding({ top: rvp(1), bottom: rvp(1), left: rvp(6), right: rvp(6), })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(xs())  
 }, tag => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(90))  
 }.width('100%').alignItems(HorizontalAlign.Start)  
 if (!room.activity) {  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(120))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(xs()).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }  
 }  
 .alignContent(Alignment.TopStart)  
 .backgroundColor($r('app.color.white'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 }.height(room.activity ? rvp(320) : rvp(210))  
 }, room => room.id)  
 }  
 .columnsTemplate("1fr 1fr")  
 .columnsGap(rvp(12))  
 .rowsGap(rvp(12))  
 .backgroundColor($r('app.color.bg\_gray'))  
 .width('100%')  
 .height('100%')  
 .layoutDirection(FlexDirection.Column)  
 .padding({ left: rvp(13), right: rvp(13), top: rvp(13) })  
 .onReachEnd(this.loadMore.bind(this))  
} else {  
 Column() {  
 Empty({ title: '抱歉，没有搜索到你想要的房子。' }).margin({ top: -rvp(138) })  
 }.height('100%').justifyContent(FlexAlign.Center)  
}

# 12. 租房详情页面功能

## 12.1. 点击跳转租房详情

1. 通过新建 Page 方式，新建租房列表基本组件 **pages > RentRoom > RentRoomDetail**

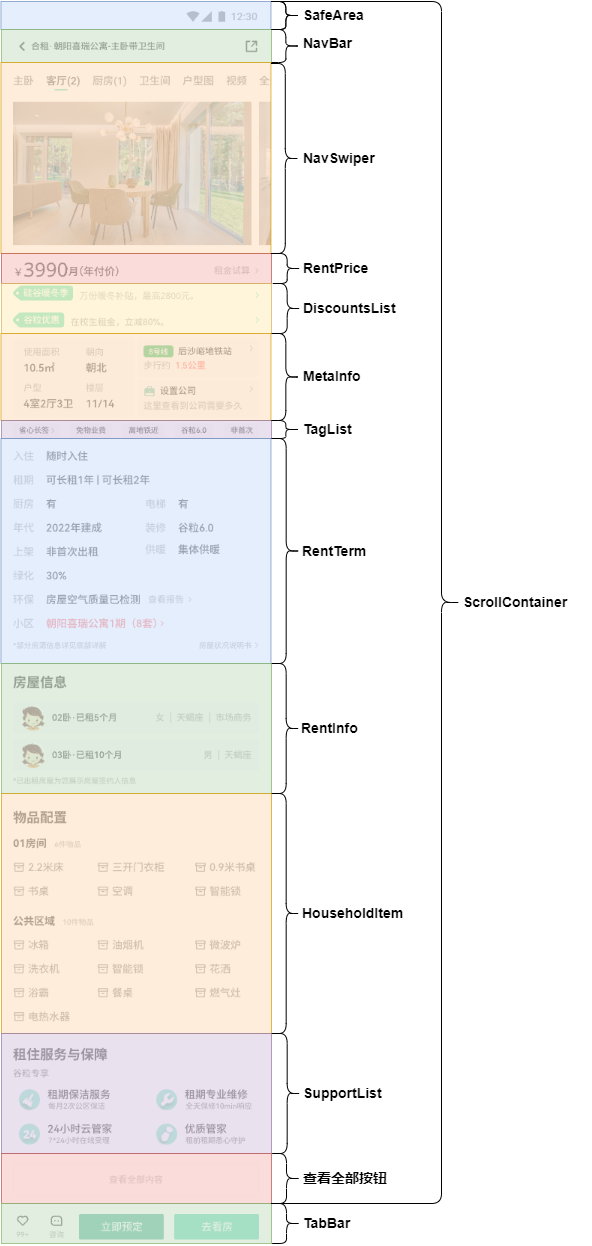
1. 首页 **RoomRecommend** 组件点击房源列表跳转租房详情

// 跳转租房详情  
goRentRoomDetail(id: string) {  
 router.pushUrl({  
 url: 'pages/RentRoom/RentRoomDetail',  
 params: {  
 id  
 }  
 })  
}  
  
// 房源列表  
@Builder RoomList() {  
 if (this.roomRecommendDataSource.roomRecommendList.length) {  
 Grid() {  
 LazyForEach(this.roomRecommendDataSource, (room: IRoomRecommendItem) => {  
 GridItem() {  
 Stack() {  
 Column() {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: rvp(BORDER\_RADIUS),  
 topRight: rvp(BORDER\_RADIUS)  
 })  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(md()).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(sm()).fontColor('#E03810')  
 }.height(rvp(22))  
 Text(room.rentArea + '㎡').fontSize(xs()).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
 Text(room.houseTitle).fontSize(sm()).margin({ top: rvp(1) }).height(rvp(19)).maxLines(1).textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag, index) => {  
 Text(tag.name)  
 .padding({ top: rvp(1), bottom: rvp(1), left: rvp(6), right: rvp(6) })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(xs())  
 }, tag => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(91))  
 }.width('100%')  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(120))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(xs()).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }.alignContent(Alignment.TopStart).backgroundColor($r('app.color.white'))  
 }.height(rvp(210)).onClick(this.goRentRoomDetail.bind(this, room.id))  
 }, room => room.id)  
 }  
 .columnsTemplate('1fr 1fr')  
 .columnsGap(rvp(8))  
 .rowsGap(rvp(8))  
 .width('100%')  
 .height(rvp(this.getHeight()))  
 .padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(8), bottom: rvp(8) }).backgroundColor($r('app.color.bg\_gray'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .hitTestBehavior(HitTestMode.None)  
 }  
}

1. 租房列表 **RentRoomList** 点击房源列表跳转租房详情

goRentRoomDetail(id: string) {  
 router.pushUrl({  
 url: 'pages/RentRoom/RentRoomDetail',  
 params: {  
 id  
 }  
 })  
}  
  
build() {  
 Stack() {  
 if (this.total > 0) {  
 WaterFlow() {  
 LazyForEach(this.roomRoomListDataSource, (room: IRentRoomItem) => {  
 FlowItem() {  
 Stack() {  
 Column() {  
 if (room.activity) {  
 Column() {  
 Row() {  
 Image(room.activity.icon).width(rvp(26))  
 Text(room.activity.title).fontSize(md()).fontWeight(700)  
 }  
  
 Text(room.activity.title)  
 .fontColor(room.activity.textColor)  
 .backgroundColor(room.activity.textBackGroundColor)  
 .padding({ top: rvp(4), bottom: rvp(4), left: rvp(8), right: rvp(8) })  
 .borderRadius(rvp(21))  
 .fontSize(xs())  
 .margin({ top: rvp(3) })  
 Image(room.housePicture)  
 .width('100%')  
 .height(rvp(170))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .objectFit(ImageFit.Fill)  
 .margin({ top: rvp(5) })  
 }.padding({ left: rvp(10), right: rvp(10), top: rvp(10)}).alignItems(HorizontalAlign.Start)  
 } else {  
 Image(room.housePicture).width('100%').height(rvp(120)).objectFit(ImageFit.Fill).borderRadius({  
 topLeft: rvp(BORDER\_RADIUS),  
 topRight: rvp(BORDER\_RADIUS)  
 })  
 }  
 Column() {  
 Row() {  
 Text() {  
 Span(room.rentPriceListing).fontSize(md()).fontColor('#E03810')  
 Span(room.rentPriceUnit).fontSize(sm()).fontColor('#E03810')  
 }.height(rvp(22))  
 Text(room.rentArea + '㎡').fontSize(xs()).fontColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
 Text(room.houseTitle).fontSize(sm()).margin({ top: rvp(1) }).height(rvp(19)).maxLines(1).textOverflow({ overflow: TextOverflow.Ellipsis })  
 Row({ space: rvp(6) }) {  
 ForEach(room.tags.slice(0, 2), (tag, index) => {  
 Text(tag.name)  
 .padding({ top: rvp(1), bottom: rvp(1), left: rvp(6), right: rvp(6), })  
 .fontColor(index === 0 ? $r('app.color.white') : '#B3B3B3')  
 .backgroundColor(index === 0 ? $r('app.color.primary') : $r('app.color.bg\_gray'))  
 .fontSize(xs())  
 }, tag => tag.name)  
 }.width('100%').margin({ top: rvp(9) })  
 }.padding(rvp(12)).alignItems(HorizontalAlign.Start).height(rvp(90))  
 }.width('100%').alignItems(HorizontalAlign.Start)  
 if (!room.activity) {  
 Image($r('app.media.room\_text\_bg')).width('100%').height(rvp(120))  
 Row({ space: rvp(6) }) {  
 Image($r('app.media.location')).width(rvp(12)).height(rvp(14))  
 Text(room.address).fontSize(xs()).fontColor($r('app.color.white'))  
 }.margin({ left: rvp(8), top: rvp(99) })  
 }  
 }  
 .alignContent(Alignment.TopStart)  
 .backgroundColor($r('app.color.white'))  
 .borderRadius(rvp(BORDER\_RADIUS))  
 }.height(room.activity ? rvp(320) : rvp(210)).onClick(this.goRentRoomDetail.bind(this, room.id))  
 }, room => room.id)  
 }  
 .columnsTemplate("1fr 1fr")  
 .columnsGap(rvp(12))  
 .rowsGap(rvp(12))  
 .backgroundColor($r('app.color.bg\_gray'))  
 .width('100%')  
 .height('100%')  
 .layoutDirection(FlexDirection.Column)  
 .padding({ left: rvp(13), right: rvp(13), top: rvp(13) })  
 .onReachEnd(this.loadMore.bind(this))  
 } else {  
 Column() {  
 Empty({ title: '抱歉，没有搜索到你想要的房子。' }).margin({ top: -rvp(138) })  
 }.height('100%').justifyContent(FlexAlign.Center)  
 }  
  
 if (this.loading) {  
 Row({ space: rvp(5) }) {  
 LoadingProgress().width(rvp(20)).height(rvp(20)).color($r('app.color.gray'))  
 Text('加载数据中...').fontSize(sm()).fontColor($r('app.color.gray'))  
 }.width('100%').backgroundColor($r('app.color.white')).justifyContent(FlexAlign.Center)  
 }  
 }.layoutWeight(1).alignContent(Alignment.Bottom)  
}

## 12.2. 布局分析



## 12.3. NavBar 组件功能

1. 引入图片资源

[📎share.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512747781-fe65815e-06f0-489a-a097-d20179dbf364.svg)

1. 搭建 **views > RentRoomDetail > NavBar** 组件

import { PADDING } from '../../constants/size'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct NavBar {  
 @StorageProp('navBarBgColor') navBarBgColor: string = ''  
 @StorageProp('navBarFontColor') navBarFontColor: string = ''  
  
 build() {  
 Row() {  
 Row({ space: rvp(8) }) {  
 Image($r('app.media.back')).width(rvp(24))  
 Text('合租· 朝阳喜瑞公寓-主卧带卫生间').fontSize(sm())  
 }  
  
 Image($r('app.media.share')).width(rvp(20)).height(rvp(20)).fillColor($r('app.color.black'))  
 }  
 .width('100%')  
 .height(rvp(44))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 .backgroundColor($r('app.color.white'))  
 }  
}

1. **RentRoomDetail** 使用 **NavBar** 组件

import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {}  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 点击回退按钮回退

import router from '@ohos.router'  
import { PADDING } from '../../constants/size'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct NavBar {  
 @StorageProp('navBarBgColor') navBarBgColor: string = ''  
 @StorageProp('navBarFontColor') navBarFontColor: string = ''  
  
 build() {  
 Row() {  
 Row({ space: rvp(8) }) {  
 Image($r('app.media.back')).width(rvp(24)).onClick(() => {  
 router.back()  
 })  
 Text('合租· 朝阳喜瑞公寓-主卧带卫生间').fontSize(sm())  
 }  
  
 Image($r('app.media.share')).width(rvp(20)).height(rvp(20)).fillColor($r('app.color.black'))  
 }  
 .width('100%')  
 .height(rvp(44))  
 .justifyContent(FlexAlign.SpaceBetween)  
 .padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 .backgroundColor($r('app.color.white'))  
 }  
}

## 12.4. 发送请求更新数据

1. 定义接口返回值类型

// ...  
export interface IHousePictureItem {  
 "spaceName": string  
 "picList": string[]  
}  
  
export type IHousePictureList = IHousePictureItem[]  
  
export interface IDistanceInfo {  
 "distance": string  
 "line": string  
 "name": string  
}  
  
export interface IRentTerm {  
 "checkInTime": string  
 "term": string  
 "kitchen": string  
 "lift": string  
 "seeTime": string  
 "year": string  
 "renovation": string  
 "grounding": string  
 "heating": string  
 "green": string  
 "protection": string  
 "house": string  
}  
  
export interface IMetaInfoItem {  
 "name": string  
 "desc": string  
}  
  
export type IMetaInfoList = IMetaInfoItem[]  
  
export interface IDiscountsItem {  
 "tagTxtBgAroundColor": string  
 "image": string  
 "txt": string  
 "tagTxt": string  
 "tagTxtColor": string  
}  
  
export type IDiscountsList = IDiscountsItem[]  
  
export interface IHouseholdItem {  
 "name": string  
 "itemList": {  
 "icon": string  
 "value":string  
 }[]  
}  
  
export type IHouseholdList = IHouseholdItem[]  
  
export interface ISupportItem {  
 "image": string  
 "title": string  
 "subTitle": string  
}  
  
export type ISupportList = ISupportItem[]  
  
export interface IRentInfoItem {  
 "image": string  
 "title": string  
 "userTag": string  
}  
  
export type IRentInfoList = IRentInfoItem[]  
  
export interface IRentRoomDetail extends Omit<IRentRoomItem, 'housePicture'> {  
 "housePicture": IHousePictureList  
 "distance\_info": IDistanceInfo  
 "rentTerm": IRentTerm  
 "metaInfo": IMetaInfoList  
 "discounts": IDiscountsList  
 "householdItem": IHouseholdList  
 "support": ISupportList  
 "rentInfo": IRentInfoList  
}

1. 定义接口函数

import request from '../utils/request'  
import type {  
 ICityList,  
 IDistrictList,  
 IRentRoomDetail,  
 IGetRentRoomListParams,  
 IGetRentRoomListResponse,  
 IProvinceList  
} from '../model/RentRoomData'  
  
// ...  
// 获取租房详情数据  
export const getRentRoomDetailApi = (id: string) => {  
 return request.get<any, IRentRoomDetail>('/house/detail', {  
 params: { id }  
 })  
}

1. 组件发送请求，更新数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type { IRentRoomDetail } from '../../model/RentRoomData'  
  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
  
 @State room: Partial<IRentRoomDetail> = {}  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 this.room = await getRentRoomDetailApi(id)  
 console.log(JSON.stringify(this.room))  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {}  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

## 12.5. NavSwiper 组件功能

### 12.5.1. 基本布局

import rvp from '../../utils/responsive/responsiveVP'  
import { md } from '../../utils/responsive/responsiveFontSize'  
  
import BasicDataSource from '../../model/BasicDataSource'  
import type { IHousePictureList, IHousePictureItem } from '../../model/RentRoomData'  
  
  
@Component  
export default struct NavSwiper {  
 @Link @Watch('handleHousePictureListChange') housePictureList: IHousePictureList  
 @State housePictureListDataSource: BasicDataSource<{ }> = new BasicDataSource()  
  
 handleHousePictureListChange() {  
 const list = this.housePictureList.reduce((p, c) => {  
 return [...p, ...c.picList]  
 }, [])  
 this.housePictureListDataSource.pushDataList(list);  
 }  
  
 @Builder NavBar() {  
 List({ space: rvp(16) }) {  
 ListItem() {}  
 ForEach(this.housePictureList, (pic: IHousePictureItem, index: number) => {  
 ListItem() {  
 Column({ space: rvp(4) }) {  
 Text(pic.spaceName + `(${pic.picList.length})`).fontSize(md()).fontColor($r('app.color.gray\_second'))  
 Divider().width(rvp(16)).height(rvp(2)).color($r('app.color.primary'))  
 }  
 }  
 }, pic => pic.spaceName)  
 ListItem() {}  
 }.listDirection(Axis.Horizontal).width('100%')  
 }  
  
 @Builder PicList() {  
 List({ space: rvp(10) }) {  
 ListItem() {}.width(rvp(6))  
 LazyForEach(this.housePictureListDataSource, rentRoomPic => {  
 ListItem() {  
 Image(rentRoomPic).width(rvp(318)).height(rvp(192))  
 }  
 }, rentRoomPic => rentRoomPic.picUrl)  
 ListItem() {}.width(rvp(6))  
 }.width('100%').listDirection(Axis.Horizontal)  
 }  
  
 build() {  
 Column({ space: rvp(16) }) {  
 this.NavBar()  
 this.PicList()  
 }.margin({ top: rvp(44 + 16) }).width('100%')  
 }  
}

### 12.5.2. 点击头部导航更新导航高亮

import rvp from '../../utils/responsive/responsiveVP'  
import { md } from '../../utils/responsive/responsiveFontSize'  
  
import BasicDataSource from '../../model/BasicDataSource'  
import type { IHousePictureItem, IHousePictureList } from '../../model/RentRoomData'  
  
  
@Component  
export default struct NavSwiper {  
 @Link @Watch('handleHousePictureListChange') housePictureList: IHousePictureList  
 @State housePictureListDataSource: BasicDataSource<string> = new BasicDataSource()  
 @State currentTabIndex: number = 0  
  
 handleHousePictureListChange() {  
 const list = this.housePictureList.reduce((p, c) => {  
 return [...p, ...c.picList]  
 }, [])  
 this.housePictureListDataSource.pushDataList(list);  
 }  
  
 setCurrentTabIndex(index: number) {  
 this.currentTabIndex = index;  
 }  
  
 @Builder NavBar() {  
 List({ space: rvp(16) }) {  
 ListItem() {  
 }  
  
 ForEach(this.housePictureList, (pic: IHousePictureItem, index: number) => {  
 ListItem() {  
 Column({ space: rvp(4) }) {  
 Text(pic.spaceName + `(${pic.picList.length})`)  
 .fontSize(md())  
 .fontColor(this.currentTabIndex === index ? $r('app.color.black') : $r('app.color.gray\_second'))  
 if (this.currentTabIndex === index) {  
 Divider().width(rvp(16)).height(rvp(2)).color($r('app.color.primary'))  
 }  
 }  
 }.onClick(this.setCurrentTabIndex.bind(this, index))  
 }, pic => pic.spaceName)  
 ListItem() {  
 }  
 }.listDirection(Axis.Horizontal).width('100%')  
 }  
  
 @Builder PicList() {  
 List({ space: rvp(10) }) {  
 ListItem() {  
 }.width(rvp(6))  
  
 LazyForEach(this.housePictureListDataSource, rentRoomPic => {  
 ListItem() {  
 Image(rentRoomPic).width(rvp(318)).height(rvp(192))  
 }  
 }, rentRoomPic => rentRoomPic.picUrl)  
 ListItem() {  
 }.width(rvp(6))  
 }.width('100%').listDirection(Axis.Horizontal)  
 }  
  
 build() {  
 Column({ space: rvp(16) }) {  
 this.NavBar()  
 this.PicList()  
 }.margin({ top: rvp(44 + 16) }).width('100%')  
 }  
}

### 12.5.3. 点击头部导航移动到指定图片位置

import rvp from '../../utils/responsive/responsiveVP'  
import { md } from '../../utils/responsive/responsiveFontSize'  
  
import BasicDataSource from '../../model/BasicDataSource'  
import type { IHousePictureItem, IHousePictureList } from '../../model/RentRoomData'  
  
  
@Component  
export default struct NavSwiper {  
 @Link @Watch('handleHousePictureListChange') housePictureList: IHousePictureList  
 @State housePictureListDataSource: BasicDataSource<{  
 url: string,  
 currentTabIndex: number  
 }> = new BasicDataSource()  
 @State currentTabIndex: number = 0  
 scroller: Scroller = new Scroller()  
  
 handleHousePictureListChange() {  
 const list = this.housePictureList.reduce((p, c, index) => {  
 return [...p, ...c.picList.map(pic => {  
 return {  
 url: pic,  
 currentTabIndex: index  
 }  
 })]  
 }, [])  
 this.housePictureListDataSource.pushDataList(list);  
 }  
  
 setCurrentTabIndex(index: number) {  
 this.currentTabIndex = index;  
 this.goCurrentTabIndexImage()  
 }  
  
 goCurrentTabIndexImage() {  
 const index = this.housePictureListDataSource.data.findIndex(pic => {  
 return pic.currentTabIndex === this.currentTabIndex  
 })  
 this.scroller.scrollTo({  
 xOffset: rvp(318) \* index + rvp(10) \* (index - 1),  
 yOffset: 0,  
 animation: {  
 duration: 500,  
 curve: Curve.EaseIn  
 }  
 })  
 }  
  
 @Builder NavBar() {  
 List({ space: rvp(16) }) {  
 ListItem() {  
 }  
  
 ForEach(this.housePictureList, (pic: IHousePictureItem, index: number) => {  
 ListItem() {  
 Column({ space: rvp(4) }) {  
 Text(pic.spaceName + `(${pic.picList.length})`)  
 .fontSize(md())  
 .fontColor(this.currentTabIndex === index ? $r('app.color.black') : $r('app.color.gray\_second'))  
 if (this.currentTabIndex === index) {  
 Divider().width(rvp(16)).height(rvp(2)).color($r('app.color.primary'))  
 }  
 }  
 }.onClick(this.setCurrentTabIndex.bind(this, index))  
 }, pic => pic.spaceName)  
 ListItem() {  
 }  
 }.listDirection(Axis.Horizontal).width('100%')  
 }  
  
 @Builder PicList() {  
 List({ space: rvp(10), scroller: this.scroller }) {  
 ListItem() {  
 }.width(rvp(6))  
  
 LazyForEach(this.housePictureListDataSource, pic => {  
 ListItem() {  
 Image(pic.url).width(rvp(318)).height(rvp(192)).objectFit(ImageFit.Fill)  
 }  
 }, pic => pic.url)  
  
 ListItem() {  
 }.width(rvp(6))  
 }.width('100%').listDirection(Axis.Horizontal)  
 }  
  
 build() {  
 Column({ space: rvp(16) }) {  
 this.NavBar()  
 this.PicList()  
 }.margin({ top: rvp(44 + 16) }).width('100%')  
 }  
}

### 12.5.4. 移动图片头部导航也发生变化

import rvp from '../../utils/responsive/responsiveVP'  
import { md } from '../../utils/responsive/responsiveFontSize'  
  
import BasicDataSource from '../../model/BasicDataSource'  
import type { IHousePictureItem, IHousePictureList } from '../../model/RentRoomData'  
  
  
@Component  
export default struct NavSwiper {  
 @Link @Watch('handleHousePictureListChange') housePictureList: IHousePictureList  
 @State housePictureListDataSource: BasicDataSource<{  
 url: string,  
 currentTabIndex: number  
 }> = new BasicDataSource()  
 @State currentTabIndex: number = 0  
 scroller: Scroller = new Scroller()  
 @State isGoCurrentTabIndexImage: boolean = false  
  
 handleHousePictureListChange() {  
 const list = this.housePictureList.reduce((p, c, index) => {  
 return [...p, ...c.picList.map(pic => {  
 return {  
 url: pic,  
 currentTabIndex: index  
 }  
 })]  
 }, [])  
 this.housePictureListDataSource.pushDataList(list);  
 }  
  
 setCurrentTabIndex(index: number) {  
 this.currentTabIndex = index;  
 this.goCurrentTabIndexImage()  
 }  
  
 goCurrentTabIndexImage() {  
 const index = this.housePictureListDataSource.data.findIndex(pic => {  
 return pic.currentTabIndex === this.currentTabIndex  
 })  
 this.scroller.scrollTo({  
 xOffset: rvp(318) \* index + rvp(10) \* (index - 1),  
 yOffset: 0,  
 animation: {  
 duration: 500,  
 curve: Curve.EaseIn  
 }  
 })  
 this.isGoCurrentTabIndexImage = true;  
 setTimeout(() => {  
 this.isGoCurrentTabIndexImage = false;  
 }, 500)  
 }  
  
 handleListScrollIndex(start: number) {  
 if (this.isGoCurrentTabIndexImage) return;  
 const pic = this.housePictureListDataSource.getData(start);  
 if (pic) {  
 this.currentTabIndex = pic.currentTabIndex;  
 }  
 }  
  
 @Builder NavBar() {  
 List({ space: rvp(16) }) {  
 ListItem() {  
 }  
  
 ForEach(this.housePictureList, (pic: IHousePictureItem, index: number) => {  
 ListItem() {  
 Column({ space: rvp(4) }) {  
 Text(pic.spaceName + `(${pic.picList.length})`)  
 .fontSize(md())  
 .fontColor(this.currentTabIndex === index ? $r('app.color.black') : $r('app.color.gray\_second'))  
 if (this.currentTabIndex === index) {  
 Divider().width(rvp(16)).height(rvp(2)).color($r('app.color.primary'))  
 }  
 }  
 }.onClick(this.setCurrentTabIndex.bind(this, index))  
 }, pic => pic.spaceName)  
 ListItem() {  
 }  
 }.listDirection(Axis.Horizontal).width('100%')  
 }  
  
 @Builder PicList() {  
 List({ space: rvp(10), scroller: this.scroller }) {  
 ListItem() {  
 }.width(rvp(6))  
  
 LazyForEach(this.housePictureListDataSource, pic => {  
 ListItem() {  
 Image(pic.url).width(rvp(318)).height(rvp(192)).objectFit(ImageFit.Fill)  
 }  
 }, pic => pic.url)  
  
 ListItem() {  
 }.width(rvp(6))  
 }.width('100%').listDirection(Axis.Horizontal).onScrollIndex(this.handleListScrollIndex.bind(this))  
 }  
  
 build() {  
 Column({ space: rvp(16) }) {  
 this.NavBar()  
 this.PicList()  
 }.margin({ top: rvp(44 + 16) }).width('100%')  
 }  
}

## 12.6. RentPrice 组件功能

### 12.6.1. 父子组件通信

1. 父组件传递数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type { IRentRoomDetail, IHousePictureList } from '../../model/RentRoomData'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { PADDING } from '../../constants/size'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 this.room = await getRentRoomDetailApi(id)  
 this.housePictureList = this.room.housePicture  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 子组件声明接受

@Component  
export default struct RentPrice {  
 @Prop rentPriceListing: string  
 @Prop rentPriceUnit: string  
}

### 12.6.2. 搭建组件布局

import { md, sm } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct RentPrice {  
 @Prop rentPriceListing: string  
 @Prop rentPriceUnit: string  
  
 build() {  
 Row() {  
 Row({ space: rvp(2) }) {  
 Text('￥').fontSize(md()).fontWeight(700)  
 Text(this.rentPriceListing).fontSize(rvp(26)).fontWeight(700).height(rvp(26))  
 Text(this.rentPriceUnit).fontSize(md())  
 }.alignItems(VerticalAlign.Bottom)  
  
 Row({ space: rvp(8) }) {  
 Text('租金试算').fontSize(sm()).fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_right')).width(rvp(4)).fillColor($r('app.color.gray'))  
 }  
 }.justifyContent(FlexAlign.SpaceBetween).width('100%').margin({ top: rvp(15) })  
 }  
}

## 12.7. DiscountsList 组件功能

### 12.7.1. 父子组件通信

1. 父组件传递数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type { IRentRoomDetail, IHousePictureList, IDiscountsList } from '../../model/RentRoomData'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { PADDING } from '../../constants/size'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 子组件声明接受

@Component  
export default struct DiscountsList {  
 @Link discountsList: IDiscountsList  
}

### 12.7.2. 搭建组件布局

import { sm } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
import type { IDiscountsList } from '../../model/RentRoomData'  
  
@Component  
export default struct DiscountsList {  
 @Link discountsList: IDiscountsList  
  
 build() {  
 Column({ space: rvp(16) }) {  
 ForEach(this.discountsList, item => {  
 Row({ space: 8 }) {  
 Image(item.image).height(rvp(20))  
 Text(item.txt).fontSize(sm()).fontColor($r('app.color.gray'))  
 Blank()  
 Image($r('app.media.arrow\_right')).width(rvp(4)).fillColor($r('app.color.primary'))  
 }.width('100%')  
 }, item => item.txt)  
 }.margin({ top: rvp(13) })  
 }  
}

## 12.8. MetaInfo 组件功能

### 12.8.1. 父子组件通信

1. 父组件传递数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type { IRentRoomDetail, IHousePictureList, IDiscountsList, IMetaInfoList, IDistanceInfo } from '../../model/RentRoomData'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { PADDING } from '../../constants/size'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 console.log(JSON.stringify(room.distance\_info))  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white'))  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 子组件声明接受

import type { IMetaInfoList } from '../../model/RentRoomData';  
  
@Component  
export default struct MetaInfo {  
 @Link metaInfoList: IMetaInfoList  
 @Link distanceInfo: IDistanceInfo  
   
 build() {  
 Column() {  
 Text("MetaInfo");  
 }  
 }  
}

### 12.8.2. 搭建组件布局

import type { IDistanceInfo, IMetaInfoList } from '../../model/RentRoomData';  
  
import rvp from '../../utils/responsive/responsiveVP';  
import { md, sm, xs } from '../../utils/responsive/responsiveFontSize';  
  
@Component  
export default struct MetaInfo {  
 @Link metaInfoList: IMetaInfoList  
 @Link distanceInfo: IDistanceInfo  
  
 build() {  
 Row({ space: rvp(4) }) {  
 Row() {  
 Grid() {  
 ForEach(this.metaInfoList, item => {  
 GridItem() {  
 Column({ space: rvp(8) }) {  
 Text(item.name).fontSize(sm()).fontColor($r('app.color.gray'))  
 Text(item.desc).fontSize(md())  
 }.alignItems(HorizontalAlign.Start).width('100%')  
 }  
 }, item => item.name)  
 }  
 .height('100%')  
 .rowsTemplate('1fr 1fr')  
 .columnsTemplate('1fr 1fr')  
 .columnsGap(rvp(10))  
 .rowsGap(rvp(8))  
 }.height('100%').padding(rvp(10)).width(rvp(162)).backgroundColor('#F8F9FC')  
  
 Column({ space: rvp(4) }) {  
 Column({ space: rvp(7) }) {  
 if (this.distanceInfo && this.distanceInfo.line) {  
 Row({ space: rvp(4) }) {  
 Text(this.distanceInfo.line)  
 .fontSize(xs())  
 .fontColor($r('app.color.white'))  
 .backgroundColor('#01ab39')  
 .padding({ top: rvp(2), bottom: rvp(2), left: rvp(7), right: rvp(7) })  
 .borderRadius(rvp(2))  
 Text(this.distanceInfo.name).fontSize(sm())  
 Blank()  
 Image($r('app.media.arrow\_right')).width(rvp(4)).fillColor($r('app.color.gray'))  
 }.width('100%')  
 Row({ space: rvp(4) }) {  
 Text('步行约').fontColor($r('app.color.gray')).fontSize(sm())  
 Text(this.distanceInfo.distance).fontSize(12).fontColor('#FF4141')  
 }.width('100%')  
 } else {  
 Text('该房源附近没有地铁').fontSize(sm())  
 }  
 }.height(rvp(50)).backgroundColor('#F8F9FC').width('100%').padding(rvp(10)).alignItems(HorizontalAlign.Start)  
  
 Column({ space: rvp(7) }) {  
 Row({ space: rvp(4) }) {  
 Image($r('app.media.bag')).width(rvp(16))  
 Text('设置公司').fontSize(sm())  
 Blank()  
 Image($r('app.media.arrow\_right')).width(rvp(4)).fillColor($r('app.color.gray'))  
 }.width('100%')  
  
 Text('这里查看到公司要多久').fontColor($r('app.color.gray')).fontSize(sm()).width('100%')  
 }.height(rvp(50)).backgroundColor('#F8F9FC').width('100%').padding(rvp(8))  
 }.width(rvp(162)).height('100%')  
 }  
 .width('100%')  
 .height(rvp(104))  
 .margin({ top: rvp(16) })  
 }  
}

## 12.9. TagList 组件功能

### 12.9.1. 父子组件通信

1. 父组件传递数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
import TagList from '../../views/RentRoomDetail/TagList'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type { IRentRoomDetail, IHousePictureList, IDiscountsList, IMetaInfoList, IDistanceInfo } from '../../model/RentRoomData'  
import type { ITagList } from '../../model/RoomRecommendDataSource'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { PADDING } from '../../constants/size'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 @State tagList: ITagList = []  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 this.tagList = room.tags  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 TagList({ tagList: $tagList })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white'))  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 子组件声明接受

import { ITagList } from '../../model/RoomRecommendDataSource'  
  
@Component  
export default struct TagList {  
 @Link tagList: ITagList  
  
 build() {  
 Column() {  
 Text("TagList");  
 }  
 }  
}

### 12.9.2. 搭建组件布局

import { ITagList } from '../../model/RoomRecommendDataSource'  
  
import { xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct TagList {  
 @Link tagList: ITagList  
  
 build() {  
 Row({ space: rvp(14) }) {  
 ForEach(this.tagList.slice(0, 4), item => {  
 Text(item.name).fontSize(xs()).backgroundColor('#F8F9FC').padding({ top: rvp(4), bottom: rvp(4), left: rvp(8), right: rvp(8) })  
 }, item => item.name)  
 }.width('100%').margin({ top: rvp(8) })  
 }  
}

## 12.10. RentTerm 组件功能

### 12.10.1. 父子组件通信

1. 父组件传递数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
import TagList from '../../views/RentRoomDetail/TagList'  
import RentTerm from '../../views/RentRoomDetail/RentTerm'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type { IRentRoomDetail, IHousePictureList, IDiscountsList, IMetaInfoList, IDistanceInfo, IRentTerm } from '../../model/RentRoomData'  
import type { ITagList } from '../../model/RoomRecommendDataSource'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { PADDING } from '../../constants/size'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 @State tagList: ITagList = []  
 @State rentTerm: Partial<IRentTerm> = {}  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 this.tagList = room.tags  
 this.rentTerm = room.rentTerm  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 TagList({ tagList: $tagList })  
 RentTerm({ rentTerm: $rentTerm })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white'))  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 子组件声明接受

import type { IRentTerm } from '../../model/RentRoomData';  
  
@Component  
export default struct RentTerm {  
 @Link rentTerm: IRentTerm  
  
 build() {  
 Column() {  
 Text("RentTerm");  
 }  
 }  
}

### 12.10.2. 搭建组件布局

import type { IRentTerm } from '../../model/RentRoomData';  
import { md } from '../../utils/responsive/responsiveFontSize';  
import rvp from '../../utils/responsive/responsiveVP';  
  
@Component  
export default struct RentTerm {  
 @Link rentTerm: IRentTerm  
  
  
 build() {  
 Column({ space: rvp(16) }) {  
 Row({ space: rvp(16) }) {  
 Text('入住').grayText(14)  
 Text(this.rentTerm.term)  
 }  
 Row({ space: rvp(16) }) {  
 Text('租期').grayText(14)  
 Text(this.rentTerm.checkInTime)  
 }  
 Row() {  
 Row({ space: rvp(16) }) {  
 Text('厨房').grayText(14)  
 Text(this.rentTerm.kitchen)  
 }.width('50%')  
 Row({ space: rvp(16) }) {  
 Text('电梯').grayText(14)  
 Text(this.rentTerm.lift)  
 }.width('50%')  
 }.width('100%')  
 Row() {  
 Row({ space: rvp(16) }) {  
 Text('年代').grayText(14)  
 Text(this.rentTerm.year).fontSize(14)  
 }.width('50%')  
 Row({ space: rvp(16) }) {  
 Text('装修').grayText(14)  
 Text(this.rentTerm.renovation).fontSize(14)  
 }.width('50%')  
 }.width('100%')  
 Row() {  
 Row({ space: rvp(16) }) {  
 Text('上架').grayText(14)  
 Text(this.rentTerm.grounding).fontSize(md())  
 }.width('50%')  
 Row({ space: rvp(16) }) {  
 Text('供暖').grayText(14)  
 Text(this.rentTerm.heating).fontSize(md())  
 }.width('50%')  
 }.width('100%')  
 Row({ space: rvp(16) }) {  
 Text('绿化').grayText(14)  
 Text(this.rentTerm.green).fontSize(md())  
 }  
 Row({ space: rvp(16) }) {  
 Text('环保').grayText(14)  
 Text(this.rentTerm.protection).fontSize(md())  
 Row({ space: rvp(5) }) {  
 Text('查看报告').grayText(12)  
 Image($r('app.media.arrow\_right')).width(rvp(4)).fillColor($r('app.color.gray'))  
 }  
 }  
 Row({ space: 16 }) {  
 Text('小区').grayText(14)  
 Row() {  
 Text(this.rentTerm.house).fontSize(md()).fontColor('#FF4141')  
 Image($r('app.media.arrow\_right')).width(rvp(4)).fillColor('#FF4141').margin({ left: rvp(5) })  
 }  
 }  
 Row() {  
 Text('\*部分房源信息详见底部详解').grayText(10)  
 Blank()  
 Text('房屋状况说明书').grayText(10)  
 Image($r('app.media.arrow\_right')).width(rvp(4)).fillColor($r('app.color.gray')).margin({ left: rvp(5) })  
 }.width('100%')  
 }.alignItems(HorizontalAlign.Start).width('100%').margin({ top: rvp(16) })  
 }  
}  
  
  
@Extend(Text) function grayText(size: number) {  
 .fontColor($r('app.color.gray')).fontSize(rvp(size))  
}

## 12.11. RentInfo 组件功能

### 12.11.1. 父子组件通信

1. 父组件传递数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
import TagList from '../../views/RentRoomDetail/TagList'  
import RentTerm from '../../views/RentRoomDetail/RentTerm'  
import RentInfo from '../../views/RentRoomDetail/RentInfo'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type { IRentRoomDetail, IHousePictureList, IDiscountsList, IMetaInfoList, IDistanceInfo, IRentTerm, IRentInfoList } from '../../model/RentRoomData'  
import type { ITagList } from '../../model/RoomRecommendDataSource'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { PADDING } from '../../constants/size'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 @State tagList: ITagList = []  
 @State rentTerm: Partial<IRentTerm> = {}  
 @State rentInfoList: IRentInfoList = []  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 this.tagList = room.tags  
 this.rentTerm = room.rentTerm  
 this.rentInfoList = room.rentInfo  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 TagList({ tagList: $tagList })  
 RentTerm({ rentTerm: $rentTerm })  
 RentInfo({ rentInfoList: $rentInfoList })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white'))  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 子组件声明接受

import type { IRentInfoList } from '../../model/RentRoomData';  
  
@Component  
export default struct RentInfo {  
 @Link rentInfoList: IRentInfoList  
  
 build() {  
 Column() {  
 Text("RentInfo");  
 }  
 }  
}

### 12.11.2. 搭建组件布局

import type { IRentInfoList } from '../../model/RentRoomData';  
  
import { xs, sm } from '../../utils/responsive/responsiveFontSize';  
import rvp from '../../utils/responsive/responsiveVP';  
  
@Component  
export default struct RentInfo {  
 @Link rentInfoList: IRentInfoList  
  
 build() {  
 Column({ space: rvp(8) }) {  
 Text('租客信息').fontSize(18).fontWeight(700)  
 ForEach(this.rentInfoList, rentInfo => {  
 Row({ space: rvp(8) }) {  
 Image(rentInfo.image).width(rvp(34))  
 Text(rentInfo.title).fontSize(sm())  
 Blank()  
 Text(rentInfo.userTag).fontColor($r('app.color.gray')).fontSize(sm())  
 }.backgroundColor('#FAF8FB').padding({ top: rvp(4), bottom: rvp(4), left: rvp(8), right: rvp(8) }).width('100%').height(rvp(42)).margin({ top: rvp(8) })  
 }, rentInfo => rentInfo.userTag)  
 Text('\*已出租房屋为您展示房屋签约人信息').fontColor($r('app.color.gray')).fontSize(xs())  
 }.alignItems(HorizontalAlign.Start).margin({ top: rvp(20) })  
 }  
}

## 12.12. HouseholdItem 组件功能

### 12.12.1. 父子组件通信

1. 父组件传递数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
import TagList from '../../views/RentRoomDetail/TagList'  
import RentTerm from '../../views/RentRoomDetail/RentTerm'  
import RentInfo from '../../views/RentRoomDetail/RentInfo'  
import HouseholdItem from '../../views/RentRoomDetail/HouseholdItem'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type { IRentRoomDetail, IHousePictureList, IDiscountsList, IMetaInfoList, IDistanceInfo, IRentTerm, IRentInfoList, IHouseholdList } from '../../model/RentRoomData'  
import type { ITagList } from '../../model/RoomRecommendDataSource'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { PADDING } from '../../constants/size'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 @State tagList: ITagList = []  
 @State rentTerm: Partial<IRentTerm> = {}  
 @State rentInfoList: IRentInfoList = []  
 @State householdList: IHouseholdList = []  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 this.tagList = room.tags  
 this.rentTerm = room.rentTerm  
 this.rentInfoList = room.rentInfo  
 this.householdList = room.householdItem  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 TagList({ tagList: $tagList })  
 RentTerm({ rentTerm: $rentTerm })  
 RentInfo({ rentInfoList: $rentInfoList })  
 HouseholdItem({ householdList: $householdList })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white'))  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 子组件声明接受

import type { IHouseholdList } from '../../model/RentRoomData';  
  
@Component  
export default struct HouseholdItem {  
 @Link householdList: IHouseholdList  
  
 build() {  
 Column() {  
 Text("HouseholdItem");  
 }  
 }  
}

### 12.12.2. 搭建组件布局

import type { IHouseholdList } from '../../model/RentRoomData';  
  
import { md, xl } from '../../utils/responsive/responsiveFontSize';  
import rvp from '../../utils/responsive/responsiveVP';  
  
@Component  
export default struct HouseholdItem {  
 @Link householdList: IHouseholdList  
  
 build() {  
 Column({ space: rvp(16) }) {  
 Text('物品配置').fontSize(xl()).fontWeight(700)  
 ForEach(this.householdList, household => {  
 Column({ space: rvp(16) }) {  
 Row({ space: rvp(10) }) {  
 Text(household.name).fontSize(md()).fontWeight(700)  
 Text(`${household.itemList.length}件物品`).grayText(10)  
 }  
 Grid() {  
 ForEach(household.itemList, item => {  
 GridItem() {  
 Row({ space: rvp(4) }) {  
 Image(item.icon).width(rvp(16))  
 Text(item.value).grayText(14)  
 }.justifyContent(FlexAlign.Start).width('100%')  
 }  
 }, item => item.value)  
 }  
 .columnsTemplate('1fr 1fr 1fr')  
 .rowsGap(rvp(16))  
 .width('100%')  
 .height(rvp(16 \* 3))  
 }.alignItems(HorizontalAlign.Start)  
 }, household => household.name)  
 }.alignItems(HorizontalAlign.Start).margin({ top: rvp(20) })  
 }  
}  
  
@Extend(Text) function grayText(size: number) {  
 .fontColor($r('app.color.gray')).fontSize(rvp(size))  
}

## 12.13. SupportList 组件功能

### 12.13.1. 父子组件通信

1. 父组件传递数据

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
import TagList from '../../views/RentRoomDetail/TagList'  
import RentTerm from '../../views/RentRoomDetail/RentTerm'  
import RentInfo from '../../views/RentRoomDetail/RentInfo'  
import HouseholdItem from '../../views/RentRoomDetail/HouseholdItem'  
import SupportList from '../../views/RentRoomDetail/SupportList'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type {  
 IDiscountsList,  
 IDistanceInfo,  
 IHouseholdList,  
 IHousePictureList,  
 IMetaInfoList,  
 IRentInfoList,  
 IRentRoomDetail,  
 IRentTerm,  
 ISupportList  
} from '../../model/RentRoomData'  
import type { ITagList } from '../../model/RoomRecommendDataSource'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { PADDING } from '../../constants/size'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 @State tagList: ITagList = []  
 @State rentTerm: Partial<IRentTerm> = {}  
 @State rentInfoList: IRentInfoList = []  
 @State householdList: IHouseholdList = []  
 @State supportList: ISupportList = []  
  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 this.tagList = room.tags  
 this.rentTerm = room.rentTerm  
 this.rentInfoList = room.rentInfo  
 this.householdList = room.householdItem  
 this.supportList = room.support  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 TagList({ tagList: $tagList })  
 RentTerm({ rentTerm: $rentTerm })  
 RentInfo({ rentInfoList: $rentInfoList })  
 HouseholdItem({ householdList: $householdList })  
 SupportList({ supportList: $supportList })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white'))  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 子组件声明接受

import type { ISupportList } from '../../model/RentRoomData';  
  
@Component  
export default struct SupportList {  
 @Link supportList: ISupportList  
  
 build() {  
 Column() {  
 Text("SupportList");  
 }  
 }  
}

### 12.13.2. 搭建组件布局

import type { ISupportList } from '../../model/RentRoomData';  
  
import { md, xl } from '../../utils/responsive/responsiveFontSize';  
import rvp from '../../utils/responsive/responsiveVP';  
  
@Component  
export default struct SupportList {  
 @Link supportList: ISupportList  
  
 build() {  
 Column({ space: rvp(8) }) {  
 Text('租住服务与保障').fontSize(xl()).fontWeight(700)  
 Text('谷粒专享').grayText(12)  
 Grid() {  
 ForEach(this.supportList, support => {  
 GridItem() {  
 Row({ space: rvp(10) }) {  
 Image(support.image).width(rvp(28))  
 Column() {  
 Text(support.title).fontSize(md())  
 Text(support.subTitle).grayText(10)  
 }.alignItems(HorizontalAlign.Start)  
 }.width('100%')  
 }  
 }, support => support.title)  
 }  
 .columnsTemplate('1fr 1fr')  
 .rowsTemplate('1fr 1fr')  
 .rowsGap(rvp(16))  
 .height(rvp(76))  
 .margin({ top: rvp(5) })  
 }.alignItems(HorizontalAlign.Start).margin({ top: rvp(20) })  
 }  
}  
  
  
@Extend(Text) function grayText(size: number) {  
 .fontColor($r('app.color.gray')).fontSize(rvp(size))  
}

## 12.14. 查看全部内容按钮布局

import router from '@ohos.router'  
  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
import TagList from '../../views/RentRoomDetail/TagList'  
import RentTerm from '../../views/RentRoomDetail/RentTerm'  
import RentInfo from '../../views/RentRoomDetail/RentInfo'  
import HouseholdItem from '../../views/RentRoomDetail/HouseholdItem'  
import SupportList from '../../views/RentRoomDetail/SupportList'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type {  
 IDiscountsList,  
 IDistanceInfo,  
 IHouseholdList,  
 IHousePictureList,  
 IMetaInfoList,  
 IRentInfoList,  
 IRentRoomDetail,  
 IRentTerm,  
 ISupportList  
} from '../../model/RentRoomData'  
import type { ITagList } from '../../model/RoomRecommendDataSource'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
@Entry  
@Component  
export default struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 @State tagList: ITagList = []  
 @State rentTerm: Partial<IRentTerm> = {}  
 @State rentInfoList: IRentInfoList = []  
 @State householdList: IHouseholdList = []  
 @State supportList: ISupportList = []  
  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 this.tagList = room.tags  
 this.rentTerm = room.rentTerm  
 this.rentInfoList = room.rentInfo  
 this.householdList = room.householdItem  
 this.supportList = room.support  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 TagList({ tagList: $tagList })  
 RentTerm({ rentTerm: $rentTerm })  
 RentInfo({ rentInfoList: $rentInfoList })  
 HouseholdItem({ householdList: $householdList })  
 SupportList({ supportList: $supportList })  
 Button('查看全部内容')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(44))  
 .fontSize(sm())  
 .fontColor('#ABA9AC')  
 .margin({ top: rvp(25) })  
 .backgroundColor($r('app.color.white'))  
 .border({ width: 1, color: '#F5F5F5', radius: rvp(BORDER\_RADIUS\_S) })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white'))  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

## 12.15. 预约租房按钮布局

import { xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct Book {  
 build() {  
 Row() {  
 Column({ space: rvp(4) }) {  
 Image($r('app.media.my\_icon\_1')).width(rvp(18))  
 Text('99+').fontSize(xs()).fontColor($r('app.color.gray'))  
 }  
 Column({ space: rvp(4) }) {  
 Image($r('app.media.message')).width(rvp(18)).fillColor($r('app.color.black'))  
 Text('咨询').fontSize(xs()).fontColor($r('app.color.gray'))  
 }  
 Button('立即预定', { type: ButtonType.Normal }).borderRadius(rvp(2)).backgroundColor('#24A17B').padding({ left: rvp(30), right: rvp(30) }).height(rvp(34))  
 Button('去看房', { type: ButtonType.Normal }).borderRadius(rvp(2)).backgroundColor('#6BE3BE').padding({ left: rvp(30), right: rvp(30) }).height(rvp(34))  
 }.backgroundColor($r('app.color.white')).height(rvp(50)).justifyContent(FlexAlign.SpaceEvenly).width('100%')  
 }  
}

import router from '@ohos.router'  
  
import SafeArea from '../../components/SafeArea'  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
import TagList from '../../views/RentRoomDetail/TagList'  
import RentTerm from '../../views/RentRoomDetail/RentTerm'  
import RentInfo from '../../views/RentRoomDetail/RentInfo'  
import HouseholdItem from '../../views/RentRoomDetail/HouseholdItem'  
import SupportList from '../../views/RentRoomDetail/SupportList'  
import Book from '../../views/RentRoomDetail/Book'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type {  
 IDiscountsList,  
 IDistanceInfo,  
 IHouseholdList,  
 IHousePictureList,  
 IMetaInfoList,  
 IRentInfoList,  
 IRentRoomDetail,  
 IRentTerm,  
 ISupportList  
} from '../../model/RentRoomData'  
import type { ITagList } from '../../model/RoomRecommendDataSource'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
@Entry  
@Component  
struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 @State tagList: ITagList = []  
 @State rentTerm: Partial<IRentTerm> = {}  
 @State rentInfoList: IRentInfoList = []  
 @State householdList: IHouseholdList = []  
 @State supportList: ISupportList = []  
  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
  
 async getRentRoomDetail() {  
 const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(id)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 this.tagList = room.tags  
 this.rentTerm = room.rentTerm  
 this.rentInfoList = room.rentInfo  
 this.householdList = room.householdItem  
 this.supportList = room.support  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Row() {  
 SafeArea()  
 }.backgroundColor($r('app.color.white'))  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 TagList({ tagList: $tagList })  
 RentTerm({ rentTerm: $rentTerm })  
 RentInfo({ rentInfoList: $rentInfoList })  
 HouseholdItem({ householdList: $householdList })  
 SupportList({ supportList: $supportList })  
 Button('查看全部内容')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(44))  
 .fontSize(sm())  
 .fontColor('#ABA9AC')  
 .margin({ top: rvp(25) })  
 .backgroundColor($r('app.color.white'))  
 .border({ width: 1, color: '#F5F5F5', radius: rvp(BORDER\_RADIUS\_S) })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white')).padding({ bottom: rvp(50) })  
 }  
  
 build() {  
 Stack() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
  
 Book()  
 }.height('100%').alignContent(Alignment.Bottom)  
 }  
}

# 13. 登录页面功能

## 13.1. 点击路由跳转

1. 新建登录页面

* pages > Login > LoginPhone

* pages > Login > LoginCode

注意：要以新建 pages 方式新建，这样会自动注册路由。如果没有通过这种方式的话，需要去 **src > main > resources > base > profile > main\_pages.json** 中注册路由

1. 在我的页面点击头像，跳转到 **LoginPhone**

import router from '@ohos.router'  
import Avatar from '../../components/Avatar'  
import { lg, sm } from '../../utils/responsive/responsiveFontSize'  
  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct UserInfo {  
 @State avatarSrc: string = ''  
  
 goLoginPhone() {  
 router.pushUrl({ url: 'pages/Login/LoginPhone' })  
 }  
  
 build() {  
 Row({ space: rvp(10) }) {  
 Avatar({ src: this.avatarSrc, avatarSize: rvp(60) }).onClick(this.goLoginPhone.bind(this))  
 Column({ space: rvp(4) }) {  
 Text('登录/注册').fontSize(lg()).fontColor($r('app.color.white')).fontWeight(700)  
 Text('点击登录注册').fontSize(sm()).fontColor($r('app.color.white'))  
 }.alignItems(HorizontalAlign.Start).onClick(this.goLoginPhone.bind(this))  
 }.margin({ top: rvp(11) }).width('100%')  
 }  
}

## 13.2. LoginPhone

### 13.2.1. NavBar 组件功能

import SafeArea from '../../components/SafeArea'  
import NavBar from '../../components/NavBar'  
  
@Entry  
@Component  
struct LoginPhone {  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 }  
 .width('100%')  
 }  
}

### 13.2.2. 搭建静态页面

1. 引入图片资源

[📎radio.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512747896-11646dd2-d9d4-4052-ac14-ff620771f277.svg)[📎radio\_active.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512748014-37eb8cd4-13c3-42a4-80a6-dd193d33263c.svg)[📎success.svg](https://www.yuque.com/attachments/yuque/0/2024/svg/45022700/1723512748152-d73bcf68-0a7b-4c47-8557-821cb0a794b0.svg)

1. 完成

import NavBar from '../../components/NavBar'  
import SafeArea from '../../components/SafeArea'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, sm } from '../../utils/responsive/responsiveFontSize'  
  
@Entry  
@Component  
struct LoginPhone {  
  
 @State phone: string = ''  
  
 @Builder LoginPhoneForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入手机号' }).inputStyles()  
 Image($r('app.media.success')).width(rvp(12)).height(rvp(8)).margin({ right: rvp(5) })  
 }.alignContent(Alignment.End)  
 Divider().color('#C4EDCE')  
 Row() {  
 Image($r('app.media.radio')).width(rvp(14)).height(rvp(14))  
 Text() {  
 Span('同意').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《用户协议》').fontSize(sm()).fontColor($r('app.color.primary'))  
 Span('和').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《隐私政策》').fontSize(sm()).fontColor($r('app.color.primary'))  
 }.margin({ left: rvp(6) })  
 }.margin({ top: rvp(16) }).width('100%')  
 Button('获取验证码')  
 .margin({ top: rvp(40) }).backgroundColor($r('app.color.primary')).fontSize(sm())  
 .padding({ top: rvp(12), bottom: rvp(12), left: rvp(15), right: rvp(15) })  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginPhoneForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

### 13.2.3. 收集&验证表单

1. 需求分析

* 需求一：

* 1. 手机号需要进行校验

* 1. 手机号符合要求显示 success 图标

* 1. 手机号不符合要求不显示 success 图标

* 需求二：

* 1. 用户协议可以选择同意，显示高亮图标

* 1. 用户协议可以选择不同意，显示默认图标

* 需求三：

* 1. 当手机号输入了且用户协议同意，获取验证码按钮即高亮

* 1. 没有输入手机号或用户协议不同意，获取验证码按钮不高亮

1. 定义正则表达式 **constants > regs.ts**

export const PHONE\_REG = /^1[3-9][0-9]{9}$/;  
  
export const CODE\_REG = /^[0-9]{6}$/;

1. 收集&验证手机号

import NavBar from '../../components/NavBar'  
import SafeArea from '../../components/SafeArea'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, sm } from '../../utils/responsive/responsiveFontSize'  
import { PHONE\_REG } from '../../constants/regs'  
  
@Entry  
@Component  
struct LoginPhone {  
  
 // 手机号功能  
 @State phone: string = ''  
 @State isPhoneValid: boolean = false  
 handleInputChange(val) {  
 this.phone = val;  
 this.isPhoneValid = PHONE\_REG.test(val);  
 }  
  
 @Builder LoginPhoneForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入手机号' }).inputStyles().onChange(this.handleInputChange.bind(this))  
 if (this.isPhoneValid) {  
 Image($r('app.media.success')).width(rvp(12)).height(rvp(8)).margin({ right: rvp(5) })  
 }  
 }.alignContent(Alignment.End)  
 Divider().color('#C4EDCE')  
 Row() {  
 Image($r('app.media.radio')).width(rvp(14)).height(rvp(14))  
 Text() {  
 Span('同意').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《用户协议》').fontSize(sm()).fontColor($r('app.color.primary'))  
 Span('和').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《隐私政策》').fontSize(sm()).fontColor($r('app.color.primary'))  
 }.margin({ left: rvp(6) })  
 }.margin({ top: rvp(16) }).width('100%')  
 Button('获取验证码')  
 .margin({ top: rvp(40) }).backgroundColor($r('app.color.primary')).fontSize(sm())  
 .padding({ top: rvp(12), bottom: rvp(12), left: rvp(15), right: rvp(15) })  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginPhoneForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

1. 同意用户协议高亮效果

import NavBar from '../../components/NavBar'  
import SafeArea from '../../components/SafeArea'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, sm } from '../../utils/responsive/responsiveFontSize'  
import { PHONE\_REG } from '../../constants/regs'  
  
@Entry  
@Component  
struct LoginPhone {  
  
 // 手机号功能  
 @State phone: string = ''  
 @State isPhoneValid: boolean = false  
 // 用户协议功能  
 @State isAgree: boolean = false  
  
 handleInputChange(val) {  
 this.phone = val;  
 this.isPhoneValid = PHONE\_REG.test(val);  
 }  
  
 handleAgree() {  
 this.isAgree = !this.isAgree;  
 }  
  
 @Builder LoginPhoneForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入手机号' }).inputStyles().onChange(this.handleInputChange.bind(this))  
 if (this.isPhoneValid) {  
 Image($r('app.media.success')).width(rvp(12)).height(rvp(8)).margin({ right: rvp(5) })  
 }  
 }.alignContent(Alignment.End)  
  
 Divider().color('#C4EDCE')  
 Row() {  
 Image(this.isAgree ? $r('app.media.radio\_active') : $r('app.media.radio'))  
 .width(rvp(14))  
 .height(rvp(14))  
 .onClick(this.handleAgree.bind(this))  
 Text() {  
 Span('同意').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《用户协议》').fontSize(sm()).fontColor($r('app.color.primary'))  
 Span('和').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《隐私政策》').fontSize(sm()).fontColor($r('app.color.primary'))  
 }.margin({ left: rvp(6) })  
 }.margin({ top: rvp(16) }).width('100%')  
  
 Button('获取验证码')  
 .margin({ top: rvp(40) }).backgroundColor($r('app.color.primary')).fontSize(sm())  
 .padding({ top: rvp(12), bottom: rvp(12), left: rvp(15), right: rvp(15) })  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginPhoneForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

1. 获取验证码按钮高亮效果

import NavBar from '../../components/NavBar'  
import SafeArea from '../../components/SafeArea'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, sm } from '../../utils/responsive/responsiveFontSize'  
import { PHONE\_REG } from '../../constants/regs'  
  
@Entry  
@Component  
struct LoginPhone {  
  
 // 手机号功能  
 @State phone: string = ''  
 @State isPhoneValid: boolean = false  
 handleInputChange(val) {  
 this.phone = val;  
 this.isPhoneValid = PHONE\_REG.test(val);  
 this.handleBtnActive()  
 }  
  
 // 用户协议功能  
 @State isAgree: boolean = false  
 handleAgree() {  
 this.isAgree = !this.isAgree;  
 this.handleBtnActive()  
 }  
  
 // 获取验证码按钮功能  
 @State isBtnActive: boolean = false;  
 handleBtnActive() {  
 if (this.isAgree && this.phone) {  
 this.isBtnActive = true  
 } else {  
 this.isBtnActive = false  
 }  
 }  
  
 @Builder LoginPhoneForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入手机号' }).inputStyles().onChange(this.handleInputChange.bind(this))  
 if (this.isPhoneValid) {  
 Image($r('app.media.success')).width(rvp(12)).height(rvp(8)).margin({ right: rvp(5) })  
 }  
 }.alignContent(Alignment.End)  
  
 Divider().color('#C4EDCE')  
 Row() {  
 Image(this.isAgree ? $r('app.media.radio\_active') : $r('app.media.radio'))  
 .width(rvp(14))  
 .height(rvp(14))  
 .onClick(this.handleAgree.bind(this))  
 Text() {  
 Span('同意').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《用户协议》').fontSize(sm()).fontColor($r('app.color.primary'))  
 Span('和').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《隐私政策》').fontSize(sm()).fontColor($r('app.color.primary'))  
 }.margin({ left: rvp(6) })  
 }.margin({ top: rvp(16) }).width('100%')  
  
 Button('获取验证码')  
 .margin({ top: rvp(40) }).backgroundColor(this.isBtnActive ? $r('app.color.primary') : '#F0F0F0').fontSize(sm())  
 .fontColor(this.isBtnActive ? $r('app.color.white') : '#C1C1C1')  
 .padding({ top: rvp(12), bottom: rvp(12), left: rvp(15), right: rvp(15) })  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginPhoneForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

### 13.2.4. 点击获取验证码按钮功能

1. 封装提示框方法 **utils > showToast**

import promptAction from '@ohos.promptAction';  
  
function getHalfHeight() {  
 const windowHeight = AppStorage.Get('windowHeight') as number || 2340 // 2340 是预览器默认高度，单位px  
 const toastHeight = 40  
 return (px2vp(windowHeight) + toastHeight) / 2  
}  
  
function showToast(message: string, duration: number = 1500) {  
 promptAction.showToast({  
 message,  
 duration,  
 bottom: getHalfHeight()  
 })  
}  
  
export default showToast

1. 点击按钮跳转

import router from '@ohos.router'  
  
import NavBar from '../../components/NavBar'  
import SafeArea from '../../components/SafeArea'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, sm } from '../../utils/responsive/responsiveFontSize'  
import { PHONE\_REG } from '../../constants/regs'  
import showToast from '../../utils/showToast'  
  
@Entry  
@Component  
struct LoginPhone {  
  
 // 手机号功能  
 @State phone: string = ''  
 @State isPhoneValid: boolean = false  
 handleInputChange(val) {  
 this.phone = val;  
 this.isPhoneValid = PHONE\_REG.test(val);  
 this.handleBtnActive()  
 }  
  
 // 用户协议功能  
 @State isAgree: boolean = false  
 handleAgree() {  
 this.isAgree = !this.isAgree;  
 this.handleBtnActive()  
 }  
  
 // 获取验证码按钮功能  
 @State isBtnActive: boolean = false;  
 handleBtnActive() {  
 if (this.isAgree && this.phone) {  
 this.isBtnActive = true  
 } else {  
 this.isBtnActive = false  
 }  
 }  
  
 // 点击跳转LoginCode页面  
 goLoginCode() {  
 if (!this.isBtnActive) return;  
 if (!this.isPhoneValid) {  
 showToast('手机号不符合格式')  
 return;  
 }  
 // 验证码在下一个页面发送  
 router.pushUrl({ url: 'pages/Login/LoginCode', params: { phone: this.phone } })  
 }  
  
  
 @Builder LoginPhoneForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入手机号' }).inputStyles().onChange(this.handleInputChange.bind(this))  
 if (this.isPhoneValid) {  
 Image($r('app.media.success')).width(rvp(12)).height(rvp(8)).margin({ right: rvp(5) })  
 }  
 }.alignContent(Alignment.End)  
  
 Divider().color('#C4EDCE')  
 Row() {  
 Image(this.isAgree ? $r('app.media.radio\_active') : $r('app.media.radio'))  
 .width(rvp(14))  
 .height(rvp(14))  
 .onClick(this.handleAgree.bind(this))  
 Text() {  
 Span('同意').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《用户协议》').fontSize(sm()).fontColor($r('app.color.primary'))  
 Span('和').fontSize(sm()).fontColor('#C1C1C1')  
 Span('《隐私政策》').fontSize(sm()).fontColor($r('app.color.primary'))  
 }.margin({ left: rvp(6) })  
 }.margin({ top: rvp(16) }).width('100%')  
  
 Button('获取验证码')  
 .margin({ top: rvp(40) }).backgroundColor(this.isBtnActive ? $r('app.color.primary') : '#F0F0F0').fontSize(sm())  
 .fontColor(this.isBtnActive ? $r('app.color.white') : '#C1C1C1')  
 .padding({ top: rvp(12), bottom: rvp(12), left: rvp(15), right: rvp(15) })  
 .onClick(this.goLoginCode.bind(this))  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginPhoneForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

## 13.3. LoginCode

### 13.3.1. 搭建静态页面

import SafeArea from '../../components/SafeArea'  
import NavBar from '../../components/NavBar'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, md } from '../../utils/responsive/responsiveFontSize'  
  
@Entry  
@Component  
struct LoginCode {  
 @State isBtnActive: boolean = false  
  
 @Builder LoginCodeForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入验证码' }).inputStyles()  
 Text(`30s`).fontSize(lg()).fontColor($r('app.color.primary'))  
 }.alignContent(Alignment.End)  
  
 Divider().color('#C4EDCE')  
 Button('确认')  
 .margin({ top: rvp(40) })  
 .backgroundColor(this.isBtnActive ? $r('app.color.primary') : '#F0F0F0')  
 .fontSize(md())  
 .width(rvp(100))  
 .fontColor(this.isBtnActive ? $r('app.color.white') : '#C1C1C1')  
 .padding({ top: rvp(12), bottom: rvp(12) })  
 .enabled(this.isBtnActive)  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginCodeForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

### 13.3.2. 发送验证码

1. 定义接口函数 **api > user**

import request from '../utils/request'  
  
// 发送验证码  
export const sendSMSCodeApi = (phone: string) => {  
 return request.post<any, string>('/sendSMSCode', { phone })  
}

1. 组件默认发送请求

import router from '@ohos.router'  
  
import SafeArea from '../../components/SafeArea'  
import NavBar from '../../components/NavBar'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, md } from '../../utils/responsive/responsiveFontSize'  
import { sendSMSCodeApi } from '../../api/user'  
import showToast from '../../utils/showToast'  
  
@Entry  
@Component  
struct LoginCode {  
 @State isBtnActive: boolean = false  
  
 @State phone: string = ''  
  
 aboutToAppear() {  
 const { phone } = router.getParams() as { phone: string }  
 this.phone = phone;  
 this.sendSMSCode()  
 }  
 // 发送验证码  
 async sendSMSCode() {  
 const code = await sendSMSCodeApi(this.phone)  
 showToast(code)  
 }  
  
 @Builder LoginCodeForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入验证码' }).inputStyles()  
 Text(`30s`).fontSize(lg()).fontColor($r('app.color.primary'))  
 }.alignContent(Alignment.End)  
  
 Divider().color('#C4EDCE')  
 Button('确认')  
 .margin({ top: rvp(40) })  
 .backgroundColor(this.isBtnActive ? $r('app.color.primary') : '#F0F0F0')  
 .fontSize(md())  
 .width(rvp(100))  
 .fontColor(this.isBtnActive ? $r('app.color.white') : '#C1C1C1')  
 .padding({ top: rvp(12), bottom: rvp(12) })  
 .enabled(this.isBtnActive)  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginCodeForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

### 13.3.3. 倒计时效果

import router from '@ohos.router'  
  
import SafeArea from '../../components/SafeArea'  
import NavBar from '../../components/NavBar'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, md } from '../../utils/responsive/responsiveFontSize'  
import { sendSMSCodeApi } from '../../api/user'  
import showToast from '../../utils/showToast'  
  
@Entry  
@Component  
struct LoginCode {  
 @State isBtnActive: boolean = false  
 // 发送验证码  
 @State phone: string = ''  
 aboutToAppear() {  
 const { phone } = router.getParams() as { phone: string }  
 this.phone = phone;  
 this.sendSMSCode()  
 }  
 async sendSMSCode() {  
 this.startCountDown()  
 this.isSendCode = true  
 const code = await sendSMSCodeApi(this.phone)  
 showToast(code)  
 }  
  
 // 倒计时  
 @State isSendCode: boolean = false  
 @State time: number = 61;  
 timeId: number  
 startCountDown() {  
 this.time--  
 this.timeId = setInterval(() => {  
 this.time--  
 if (this.time <= 0) {  
 this.time = 61  
 clearInterval(this.timeId)  
 }  
 }, 1000)  
 }  
  
 aboutToDisappear() {  
 clearInterval(this.timeId)  
 }  
  
 @Builder LoginCodeForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入验证码' }).inputStyles()  
 if (this.time <= 60) {  
 Text(`${this.time}s`).fontSize(lg()).fontColor($r('app.color.primary'))  
 } else if (this.isSendCode) {  
 Text(`重新发送`).fontSize(md()).fontColor($r('app.color.primary')).onClick(this.sendSMSCode.bind(this))  
 }  
 }.alignContent(Alignment.End)  
  
 Divider().color('#C4EDCE')  
 Button('确认')  
 .margin({ top: rvp(40) })  
 .backgroundColor(this.isBtnActive ? $r('app.color.primary') : '#F0F0F0')  
 .fontSize(md())  
 .width(rvp(100))  
 .fontColor(this.isBtnActive ? $r('app.color.white') : '#C1C1C1')  
 .padding({ top: rvp(12), bottom: rvp(12) })  
 .enabled(this.isBtnActive)  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginCodeForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

### 13.3.4. 确认按钮高亮效果

import router from '@ohos.router'  
  
import SafeArea from '../../components/SafeArea'  
import NavBar from '../../components/NavBar'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, md } from '../../utils/responsive/responsiveFontSize'  
import { sendSMSCodeApi } from '../../api/user'  
import showToast from '../../utils/showToast'  
import { CODE\_REG } from '../../constants/regs'  
  
@Entry  
@Component  
struct LoginCode {  
  
 // 发送验证码  
 @State phone: string = ''  
 aboutToAppear() {  
 const { phone } = router.getParams() as { phone: string }  
 this.phone = phone;  
 this.sendSMSCode()  
 }  
 async sendSMSCode() {  
 this.startCountDown()  
 this.isSendCode = true  
 const code = await sendSMSCodeApi(this.phone)  
 showToast(code)  
 }  
  
 // 倒计时  
 @State isSendCode: boolean = false  
 @State time: number = 61;  
 timeId: number  
 startCountDown() {  
 this.time--  
 this.timeId = setInterval(() => {  
 this.time--  
 if (this.time <= 0) {  
 this.time = 61  
 clearInterval(this.timeId)  
 }  
 }, 1000)  
 }  
  
 aboutToDisappear() {  
 clearInterval(this.timeId)  
 }  
  
 // 高亮  
 @State code: string = ''  
 @State isBtnActive: boolean = false  
 handleInputChange(val) {  
 this.code = val;  
 this.isBtnActive = CODE\_REG.test(val);  
 }  
  
 @Builder LoginCodeForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入验证码' }).inputStyles().onChange(this.handleInputChange.bind(this))  
 if (this.time <= 60) {  
 Text(`${this.time}s`).fontSize(lg()).fontColor($r('app.color.primary'))  
 } else if (this.isSendCode) {  
 Text(`重新发送`).fontSize(md()).fontColor($r('app.color.primary')).onClick(this.sendSMSCode.bind(this))  
 }  
 }.alignContent(Alignment.End)  
  
 Divider().color('#C4EDCE')  
 Button('确认')  
 .margin({ top: rvp(40) })  
 .backgroundColor(this.isBtnActive ? $r('app.color.primary') : '#F0F0F0')  
 .fontSize(md())  
 .width(rvp(100))  
 .fontColor(this.isBtnActive ? $r('app.color.white') : '#C1C1C1')  
 .padding({ top: rvp(12), bottom: rvp(12) })  
 .enabled(this.isBtnActive)  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginCodeForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

### 13.3.5. 完成登录功能

1. 定义接口函数

import request from '../utils/request'  
  
// 发送验证码  
export const sendSMSCodeApi = (phone: string) => {  
 return request.post<any, string>('/sendSMSCode', { phone })  
}  
  
// 登录  
export const loginApi = (phone: string, code: string) => {  
 return request.post<any, { token: string }>('/login', { phone, code })  
}

1. 点击确认按钮，开始登录

import router from '@ohos.router'  
  
import SafeArea from '../../components/SafeArea'  
import NavBar from '../../components/NavBar'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { lg, md } from '../../utils/responsive/responsiveFontSize'  
import { sendSMSCodeApi, loginApi } from '../../api/user'  
import showToast from '../../utils/showToast'  
import { CODE\_REG } from '../../constants/regs'  
  
@Entry  
@Component  
struct LoginCode {  
 @StorageLink('token') token: string = ''  
  
 // 发送验证码  
 @State phone: string = ''  
 aboutToAppear() {  
 const { phone } = router.getParams() as { phone: string }  
 this.phone = phone;  
 this.sendSMSCode()  
 }  
 async sendSMSCode() {  
 this.startCountDown()  
 this.isSendCode = true  
 const code = await sendSMSCodeApi(this.phone)  
 showToast(code)  
 }  
  
 // 倒计时  
 @State isSendCode: boolean = false  
 @State time: number = 61;  
 timeId: number  
 startCountDown() {  
 this.time--  
 this.timeId = setInterval(() => {  
 this.time--  
 if (this.time <= 0) {  
 this.time = 61  
 clearInterval(this.timeId)  
 }  
 }, 1000)  
 }  
  
 aboutToDisappear() {  
 clearInterval(this.timeId)  
 }  
  
 // 高亮  
 @State code: string = ''  
 @State isBtnActive: boolean = false  
 handleInputChange(val) {  
 this.code = val;  
 this.isBtnActive = CODE\_REG.test(val);  
 }  
  
 // 登录  
 async login() {  
 try {  
 const { token } = await loginApi(this.phone, this.code);  
 // 持久化存储 token  
 this.token = token  
 // 跳转回去  
 router.pushUrl({ url: 'pages/Index', params: { currentIndex: 4 } })  
 } catch {  
 showToast('验证码不正确')  
 }  
 }  
  
 @Builder LoginCodeForm() {  
 Column() {  
 Stack() {  
 TextInput({ placeholder: '请输入验证码' }).inputStyles().onChange(this.handleInputChange.bind(this))  
 if (this.time <= 60) {  
 Text(`${this.time}s`).fontSize(lg()).fontColor($r('app.color.primary'))  
 } else if (this.isSendCode) {  
 Text(`重新发送`).fontSize(md()).fontColor($r('app.color.primary')).onClick(this.sendSMSCode.bind(this))  
 }  
 }.alignContent(Alignment.End)  
  
 Divider().color('#C4EDCE')  
 Button('确认')  
 .margin({ top: rvp(40) })  
 .backgroundColor(this.isBtnActive ? $r('app.color.primary') : '#F0F0F0')  
 .fontSize(md())  
 .width(rvp(100))  
 .fontColor(this.isBtnActive ? $r('app.color.white') : '#C1C1C1')  
 .padding({ top: rvp(12), bottom: rvp(12) })  
 .enabled(this.isBtnActive)  
 .onClick(this.login.bind(this))  
 }.margin({ top: rvp(47) }).padding({ left: rvp(37), right: rvp(37) }).alignItems(HorizontalAlign.End)  
 }  
  
 build() {  
 Column() {  
 SafeArea()  
 NavBar({ title: '验证码登录' })  
 this.LoginCodeForm()  
 }  
 .width('100%').height('100%').linearGradient({  
 angle: 180,  
 colors: [  
 ['#DEFBE5', 0],  
 ['#ffffff', 0.3],  
 ['#ffffff', 1],  
 ]  
 })  
 }  
}  
  
@Styles function pressedStyles() {  
 // 无法实现点击时，完全透明  
 .backgroundColor('rgba(255, 255, 255, 0)')  
 .borderRadius(0)  
}  
  
@Extend(TextInput) function inputStyles() {  
 .padding({ left: 0, right: 0, top: 0, bottom: rvp(8) })  
 .backgroundColor(Color.Transparent)  
 .placeholderColor('#C1C1C1')  
 .placeholderFont({ size: lg() })  
 .caretColor('#C1C1C1')  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .stateStyles({  
 pressed: pressedStyles,  
 clicked: pressedStyles  
 })  
}

1. **Index** 页面判断 currentIndex 参数，从而切换页面

import Home from './Home';  
import See from './See';  
import Service from './Service';  
import Discover from './Discover';  
import My from './My';  
  
import rvp from '../utils/responsive/responsiveVP';  
import { xs } from '../utils/responsive/responsiveFontSize';  
import router from '@ohos.router';  
import type { IColors } from '../model/ColorsData'  
// 必须放在入口页面  
PersistentStorage.PersistProp('token', '')  
  
@Entry  
@Component  
struct Index {  
 @StorageLink('navBarFontColor') navBarFontColor: string = ''  
 @StorageLink('navBarBgColor') navBarBgColor: string = ''  
 // 当前 tab 高亮的下标  
 @State currentIndex: number = 0;  
 colors: IColors = []  
  
 aboutToAppear() {  
 const params = router.getParams() as { currentIndex: number }  
 if (params) {  
 this.currentIndex = params.currentIndex  
 }  
 }  
  
 // 生成 tab 的函数  
 @Builder TabBuilder(index: number, text: string, icon: Resource, activeIcon: Resource) {  
 Column() {  
 // 项目中常用的颜色、大小等会定义成常量，将来方便复用和统一调整。而不常用的大小颜色就不需要定义了，直接写死即可。  
 Image(this.currentIndex === index ? activeIcon : icon).width(rvp(28))  
 Text(text)  
 .fontSize(xs())  
 .fontColor(this.currentIndex === index ? $r('app.color.black') : '#a0a0a0')  
 .height(rvp(15))  
 }.width('100%').height('100%').justifyContent(FlexAlign.Center)  
 }  
  
 build() {  
 Column() {  
 Tabs({ barPosition: BarPosition.End, index: this.currentIndex }) {  
 TabContent() {  
 Home()  
 }.tabBar(this.TabBuilder(0, '首页', $r('app.media.tabbar\_home'), $r('app.media.tabbar\_home\_active')))  
  
 TabContent() {  
 See()  
 }.tabBar(this.TabBuilder(1, '想看', $r('app.media.tabbar\_see'), $r('app.media.tabbar\_see\_active')))  
  
 TabContent() {  
 Service({ isActiveTab: this.currentIndex === 2 })  
 }.tabBar(this.TabBuilder(2, '服务', $r('app.media.tabbar\_service'), $r('app.media.tabbar\_service\_active')))  
  
 TabContent() {  
 Discover({ isActiveTab: this.currentIndex === 3 })  
 }.tabBar(this.TabBuilder(3, '发现', $r('app.media.tabbar\_discover'), $r('app.media.tabbar\_discover\_active')))  
  
 TabContent() {  
 My()  
 }.tabBar(this.TabBuilder(4, '我的', $r('app.media.tabbar\_my'), $r('app.media.tabbar\_my\_active')))  
 }  
 .vertical(false)  
 .barMode(BarMode.Fixed)  
 .onChange((index: number) => {  
 // 将上个页面color存储起来  
 this.colors[this.currentIndex] = {  
 navBarBgColor: this.navBarBgColor,  
 navBarFontColor: this.navBarFontColor  
 }  
 // 将当前页面的color赋值上去  
 this.navBarFontColor = this.colors[index]?.navBarFontColor || 'rgb(255, 255, 255)'  
 this.navBarBgColor = this.colors[index]?.navBarBgColor || 'rgba(255, 255, 255, 0)'  
  
 this.currentIndex = index  
 })  
 .barHeight(rvp(50)) // 设置 tab 导航栏的高度，默认 56  
 .scrollable(false)  
 .width('100%')  
 .height('100%')  
 .backgroundColor($r('app.color.white'))  
 }.height('100%')  
 }  
}

### 13.3.6. 获取用户数据进行展示

1. 定义接口函数&类型

* **model > UserData**

export interface IUserInfo {  
 "id": number  
 "nickname": string  
 "avatar": string  
}

* **api > user**

import request from '../utils/request'  
import type { IUserInfo } from '../model/UserData'  
// 发送验证码  
export const sendSMSCodeApi = (phone: string) => {  
 return request.post<any, string>('/sendSMSCode', { phone })  
}  
  
// 登录  
export const loginApi = (phone: string, code: string) => {  
 return request.post<any, { token: string }>('/login', { phone, code })  
}  
  
// 获取用户数据  
export const getUserInfoApi = () => {  
 return request.get<any, IUserInfo>('/auth/user/userInfo')  
}

1. **Index** 组件传递数据给 **My** 组件

import Home from './Home';  
import See from './See';  
import Service from './Service';  
import Discover from './Discover';  
import My from './My';  
  
import rvp from '../utils/responsive/responsiveVP';  
import { xs } from '../utils/responsive/responsiveFontSize';  
import router from '@ohos.router';  
import type { IColors } from '../model/ColorsData'  
// 必须放在入口页面  
PersistentStorage.PersistProp('token', '')  
  
@Entry  
@Component  
struct Index {  
 @StorageLink('navBarFontColor') navBarFontColor: string = ''  
 @StorageLink('navBarBgColor') navBarBgColor: string = ''  
 // 当前 tab 高亮的下标  
 @State currentIndex: number = 0;  
 colors: IColors = []  
  
 aboutToAppear() {  
 const params = router.getParams() as { currentIndex: number }  
 if (params) {  
 this.currentIndex = params.currentIndex  
 }  
 }  
  
 // 生成 tab 的函数  
 @Builder TabBuilder(index: number, text: string, icon: Resource, activeIcon: Resource) {  
 Column() {  
 // 项目中常用的颜色、大小等会定义成常量，将来方便复用和统一调整。而不常用的大小颜色就不需要定义了，直接写死即可。  
 Image(this.currentIndex === index ? activeIcon : icon).width(rvp(28))  
 Text(text)  
 .fontSize(xs())  
 .fontColor(this.currentIndex === index ? $r('app.color.black') : '#a0a0a0')  
 .height(rvp(15))  
 }.width('100%').height('100%').justifyContent(FlexAlign.Center)  
 }  
  
 build() {  
 Column() {  
 Tabs({ barPosition: BarPosition.End, index: this.currentIndex }) {  
 TabContent() {  
 Home()  
 }.tabBar(this.TabBuilder(0, '首页', $r('app.media.tabbar\_home'), $r('app.media.tabbar\_home\_active')))  
  
 TabContent() {  
 See()  
 }.tabBar(this.TabBuilder(1, '想看', $r('app.media.tabbar\_see'), $r('app.media.tabbar\_see\_active')))  
  
 TabContent() {  
 Service({ isActiveTab: this.currentIndex === 2 })  
 }.tabBar(this.TabBuilder(2, '服务', $r('app.media.tabbar\_service'), $r('app.media.tabbar\_service\_active')))  
  
 TabContent() {  
 Discover({ isActiveTab: this.currentIndex === 3 })  
 }.tabBar(this.TabBuilder(3, '发现', $r('app.media.tabbar\_discover'), $r('app.media.tabbar\_discover\_active')))  
  
 TabContent() {  
 My({ isActiveTab: this.currentIndex === 4 })  
 }.tabBar(this.TabBuilder(4, '我的', $r('app.media.tabbar\_my'), $r('app.media.tabbar\_my\_active')))  
 }  
 .vertical(false)  
 .barMode(BarMode.Fixed)  
 .onChange((index: number) => {  
 // 将上个页面color存储起来  
 this.colors[this.currentIndex] = {  
 navBarBgColor: this.navBarBgColor,  
 navBarFontColor: this.navBarFontColor  
 }  
 // 将当前页面的color赋值上去  
 this.navBarFontColor = this.colors[index]?.navBarFontColor || 'rgb(255, 255, 255)'  
 this.navBarBgColor = this.colors[index]?.navBarBgColor || 'rgba(255, 255, 255, 0)'  
  
 this.currentIndex = index  
 })  
 .barHeight(rvp(50)) // 设置 tab 导航栏的高度，默认 56  
 .scrollable(false)  
 .width('100%')  
 .height('100%')  
 .backgroundColor($r('app.color.white'))  
 }.height('100%')  
 }  
}

1. 给发送请求函数添加 token 参数

import axios from '@ohos/axios';  
import Prompt from '@system.prompt';  
  
export interface ResponseType<T> {  
 code: number;  
 message: string;  
 data: T;  
 ok: boolean;  
}  
  
const request = axios.create({  
 baseURL: 'http://zufang.practice.atguigu.cn',  
 timeout: 20000,  
})  
  
request.interceptors.request.use(  
 (config) => {  
 const token = AppStorage.Get('token') as string;  
 if (token) {  
 config.headers.token = token;  
 }  
 return config;  
 }  
)  
  
request.interceptors.response.use(  
 (response) => {  
 const data = response.data;  
 if (data.code === 200) {  
 return data.data;  
 } else {  
 const message = data.message;  
 Prompt.showToast({ message })  
 return Promise.reject(message);  
 }  
 },  
 (error) => {  
 const message = error.message;  
 Prompt.showToast({ message })  
 return Promise.reject(message);  
 }  
)  
  
export default request;

1. **My** 组件接受数据，发送请求更新用户数据

import ScrollContainer from '../components/ScrollContainer'  
import SafeArea from '../components/SafeArea'  
import Bg from '../views/My/Bg'  
import NavBar from '../views/My/NavBar'  
import UserInfo from '../views/My/UserInfo'  
import TimePlan from '../views/My/TimePlan'  
import Order from '../views/My/Order'  
import PlanList from '../views/My/PlanList'  
  
import rvp from '../utils/responsive/responsiveVP'  
import { PADDING } from '../constants/size'  
  
import { getUserInfoApi } from '../api/user'  
import type { IUserInfo } from '../model/UserData'  
  
@Component  
export default struct My {  
 @Prop @Watch('handleIsActiveTabChange') isActiveTab: boolean  
 @State userInfo: IUserInfo = {  
 id: 0,  
 avatar: '',  
 nickname: ''  
 }  
  
 aboutToAppear() {  
 if (this.isActiveTab) {  
 this.getUserInfo()  
 }  
 }  
  
 async handleIsActiveTabChange() {  
 if (this.isActiveTab) {  
 const token = AppStorage.Get('token') as string;  
 if (token) {  
 this.getUserInfo()  
 }  
 }  
 }  
  
 async getUserInfo() {  
 this.userInfo = await getUserInfoApi()  
 }  
  
 @Builder NavBarRender () {  
 NavBar()  
 }  
  
 @Builder ContentRender () {  
 Stack() {  
 Bg()  
 Column() {  
 SafeArea()  
 UserInfo()  
 TimePlan()  
 Order()  
 PlanList()  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(44) })  
 }.width('100%').alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 ContentRender: this.ContentRender.bind(this),  
 NavBarRender: this.NavBarRender.bind(this),  
 })  
 }  
}

1. **My** 组件将用户数据传递给 **UserInfo** 组件进行展示

import ScrollContainer from '../components/ScrollContainer'  
import SafeArea from '../components/SafeArea'  
import Bg from '../views/My/Bg'  
import NavBar from '../views/My/NavBar'  
import UserInfo from '../views/My/UserInfo'  
import TimePlan from '../views/My/TimePlan'  
import Order from '../views/My/Order'  
import PlanList from '../views/My/PlanList'  
  
import rvp from '../utils/responsive/responsiveVP'  
import { PADDING } from '../constants/size'  
  
import { getUserInfoApi } from '../api/user'  
import type { IUserInfo } from '../model/UserData'  
  
@Component  
export default struct My {  
 @Prop @Watch('handleIsActiveTabChange') isActiveTab: boolean  
 @StorageProp('token') token: string = '';  
  
 @State userInfo: IUserInfo = {  
 id: 0,  
 avatar: '',  
 nickname: ''  
 }  
  
 aboutToAppear() {  
 if (this.isActiveTab) {  
 this.getUserInfo()  
 }  
 }  
  
 async handleIsActiveTabChange() {  
 if (this.isActiveTab && this.token) {  
 this.getUserInfo()  
 }  
 }  
  
 async getUserInfo() {  
 this.userInfo = await getUserInfoApi()  
 }  
  
 @Builder NavBarRender () {  
 NavBar()  
 }  
  
 @Builder ContentRender () {  
 Stack() {  
 Bg()  
 Column() {  
 SafeArea()  
 UserInfo({ userInfo: $userInfo })  
 TimePlan()  
 Order()  
 PlanList()  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING), top: rvp(44) })  
 }.width('100%').alignContent(Alignment.TopStart)  
 }  
  
 build() {  
 ScrollContainer({  
 ContentRender: this.ContentRender.bind(this),  
 NavBarRender: this.NavBarRender.bind(this),  
 })  
 }  
}

import router from '@ohos.router'  
import Avatar from '../../components/Avatar'  
import { IUserInfo } from '../../model/UserData'  
import { lg, sm } from '../../utils/responsive/responsiveFontSize'  
  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct UserInfo {  
 @Link userInfo: IUserInfo  
  
 @State avatarSrc: string = ''  
  
 goLoginPhone() {  
 router.pushUrl({ url: 'pages/Login/LoginPhone' })  
 }  
  
 build() {  
 Row({ space: rvp(10) }) {  
 Avatar({ src: this.userInfo.avatar || this.avatarSrc, avatarSize: rvp(60) }).onClick(this.goLoginPhone.bind(this))  
 Column({ space: rvp(4) }) {  
 Text(this.userInfo.nickname || '登录/注册').fontSize(lg()).fontColor($r('app.color.white')).fontWeight(700)  
 if (!this.userInfo.nickname) {  
 Text('点击登录注册').fontSize(sm()).fontColor($r('app.color.white'))  
 }  
 }.alignItems(HorizontalAlign.Start).onClick(this.goLoginPhone.bind(this))  
 }.margin({ top: rvp(11) }).width('100%')  
 }  
}

# 14. 预约租房页面功能

## 14.1. 预约信息功能

### 14.1.1. 基本组件

1. 搭建公共组件 **components > Drawer**

import rvp from '../utils/responsive/responsiveVP';  
  
@Component  
export default struct Drawer {  
 @Link show: boolean;  
 @BuilderParam render: () => void  
  
 build() {  
 if (this.show) {  
 Stack() {  
 Column() {  
 }.width('100%').height('100%').backgroundColor('rgba(0, 0, 0, 0.7)').onClick(() => {  
 this.show = false;  
 })  
  
 Column() {  
 this.render()  
 }.backgroundColor('#F6F6F6').width('100%').padding(rvp(15)).borderRadius({  
 topLeft: rvp(10),  
 topRight: rvp(10)  
 })  
 }  
 .width('100%')  
 .height('100%')  
 .position({ x: 0, y: 0 })  
 .zIndex(999)  
 .alignContent(Alignment.BottomStart)  
 }  
 }  
}

1. 搭建基本组件 **views > BookInfo**

import Drawer from '../../components/Drawer'  
  
@Component  
export default struct BookInfo {  
 @Link show: boolean  
  
 build() {  
 Drawer({ show: $show }) {  
 Text('Drawer')  
 }  
 }  
}

1. **RentRoomDetail** 组件切换显示 **BookInfo**

import SafeArea from '../../components/SafeArea'  
import NavBar from '../../views/RentRoomDetail/NavBar'  
import ScrollContainer from '../../components/ScrollContainer'  
import NavSwiper from '../../views/RentRoomDetail/NavSwiper'  
import RentPrice from '../../views/RentRoomDetail/RentPrice'  
import DiscountsList from '../../views/RentRoomDetail/DiscountsList'  
import MetaInfo from '../../views/RentRoomDetail/MetaInfo'  
import TagList from '../../views/RentRoomDetail/TagList'  
import RentTerm from '../../views/RentRoomDetail/RentTerm'  
import RentInfo from '../../views/RentRoomDetail/RentInfo'  
import HouseholdItem from '../../views/RentRoomDetail/HouseholdItem'  
import SupportList from '../../views/RentRoomDetail/SupportList'  
import Book from '../../views/RentRoomDetail/Book'  
import BookRentRoom from '../../views/RentRoomDetail/BookRentRoom'  
  
import { getRentRoomDetailApi } from '../../api/rentRoom'  
import type {  
 IDiscountsList,  
 IDistanceInfo,  
 IHouseholdList,  
 IHousePictureList,  
 IMetaInfoList,  
 IRentInfoList,  
 IRentRoomDetail,  
 IRentTerm,  
 ISupportList  
} from '../../model/RentRoomData'  
import type { ITagList } from '../../model/RoomRecommendDataSource'  
  
import rvp from '../../utils/responsive/responsiveVP'  
import { BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
@Entry  
@Component  
struct RentRoomDetail {  
 @State housePictureList: IHousePictureList = []  
 @State room: Partial<IRentRoomDetail> = {  
 rentPriceListing: '',  
 rentPriceUnit: ''  
 }  
 @State discountsList: IDiscountsList = []  
 @State metaInfoList: IMetaInfoList = []  
 @State distanceInfo: Partial<IDistanceInfo> = {}  
 @State tagList: ITagList = []  
 @State rentTerm: Partial<IRentTerm> = {}  
 @State rentInfoList: IRentInfoList = []  
 @State householdList: IHouseholdList = []  
 @State supportList: ISupportList = []  
 @State show: boolean = false  
  
 aboutToAppear() {  
 this.getRentRoomDetail()  
 }  
  
 async getRentRoomDetail() {  
 // const { id } = router.getParams() as { id: string }  
 const room = await getRentRoomDetailApi(`BJ1827549448671068160`)  
 this.room = room;  
 this.housePictureList = room.housePicture  
 this.discountsList = room.discounts  
 this.metaInfoList = room.metaInfo  
 this.distanceInfo = room.distance\_info  
 this.tagList = room.tags  
 this.rentTerm = room.rentTerm  
 this.rentInfoList = room.rentInfo  
 this.householdList = room.householdItem  
 this.supportList = room.support  
 }  
  
 @Builder NavBarRender() {  
 NavBar()  
 }  
  
 @Builder ContentRender() {  
 Row() {  
 SafeArea()  
 }.backgroundColor($r('app.color.white'))  
  
 Column() {  
 NavSwiper({ housePictureList: $housePictureList })  
 Column() {  
 RentPrice({ rentPriceListing: this.room.rentPriceUnitListing, rentPriceUnit: this.room.rentPriceUnit })  
 DiscountsList({ discountsList: $discountsList })  
 MetaInfo({ metaInfoList: $metaInfoList, distanceInfo: $distanceInfo })  
 TagList({ tagList: $tagList })  
 RentTerm({ rentTerm: $rentTerm })  
 RentInfo({ rentInfoList: $rentInfoList })  
 HouseholdItem({ householdList: $householdList })  
 SupportList({ supportList: $supportList })  
 Button('查看全部内容')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(44))  
 .fontSize(sm())  
 .fontColor('#ABA9AC')  
 .margin({ top: rvp(25) })  
 .backgroundColor($r('app.color.white'))  
 .border({ width: 1, color: '#F5F5F5', radius: rvp(BORDER\_RADIUS\_S) })  
 }.padding({ left: rvp(PADDING), right: rvp(PADDING) })  
 }.backgroundColor($r('app.color.white')).padding({ bottom: rvp(50) })  
 }  
  
 build() {  
 Stack() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
  
 Reservation({ show: $show })  
  
 ReservationRentRoom({ show: $show })  
 }.height('100%').alignContent(Alignment.Bottom)  
 }  
}

import { xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct Book {  
 @Link show: boolean  
  
 build() {  
 Row() {  
 Column({ space: rvp(4) }) {  
 Image($r('app.media.my\_icon\_1')).width(rvp(18))  
 Text('99+').fontSize(xs()).fontColor($r('app.color.gray'))  
 }  
  
 Column({ space: rvp(4) }) {  
 Image($r('app.media.message')).width(rvp(18)).fillColor($r('app.color.black'))  
 Text('咨询').fontSize(xs()).fontColor($r('app.color.gray'))  
 }  
  
 Button('立即预定', { type: ButtonType.Normal })  
 .borderRadius(rvp(2))  
 .backgroundColor('#24A17B')  
 .padding({ left: rvp(30), right: rvp(30) })  
 .height(rvp(34))  
 .onClick(() => {  
 this.show = true  
 })  
 Button('去看房', { type: ButtonType.Normal })  
 .borderRadius(rvp(2))  
 .backgroundColor('#6BE3BE')  
 .padding({ left: rvp(30), right: rvp(30) })  
 .height(rvp(34))  
 }.backgroundColor($r('app.color.white')).height(rvp(50)).justifyContent(FlexAlign.SpaceEvenly).width('100%')  
 }  
}

1. 搭建预约信息组件结构样式

import Drawer from '../../components/Drawer'  
import { BORDER\_RADIUS, BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { md, xl } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct BookInfo {  
 @Link show: boolean  
  
 build() {  
 Drawer({ show: $show }) {  
 Column({ space: rvp(16) }) {  
 Text('预约信息').fontSize(xl()).fontWeight(700)  
 Column({ space: rvp(20) }) {  
 Row({ space: rvp(18) }) {  
 Text('姓名').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入姓名' }).textInputStyles()  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('手机号').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入手机号' }).type(InputType.PhoneNumber).maxLength(11).textInputStyles()  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('预约日期').width(rvp(56)).fontSize(md())  
 Row() {  
 Text('2024-01-16（今天）').fontSize(md()).fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_down\_2')).width(rvp(10)).fillColor($r('app.color.gray'))  
 }.layoutWeight(1).justifyContent(FlexAlign.SpaceBetween)  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('备注信息').width(rvp(56)).fontSize(md())  
 TextArea({ placeholder: '请输入备注信息' }).textAreaStyles()  
 }.width('100%').alignItems(VerticalAlign.Top)  
 }  
 .alignItems(HorizontalAlign.Start)  
 .backgroundColor($r('app.color.white'))  
 .padding(rvp(PADDING))  
  
 Button('预约看房')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(42))  
 .padding(rvp(13))  
 .fontColor($r('app.color.white'))  
 .fontSize(md())  
 .backgroundColor('#24A17B')  
 .borderRadius(rvp(2))  
 }.alignItems(HorizontalAlign.Start).padding({ bottom: rvp(PADDING) })  
 }  
 }  
}  
  
@Styles function pressedStyles() {  
 .backgroundColor('#F6F6F6')  
}  
  
@Extend(TextInput) function textInputStyles() {  
 .layoutWeight(1)  
 .backgroundColor(Color.Transparent)  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(0)  
}  
  
@Extend(TextArea) function textAreaStyles() {  
 .layoutWeight(1)  
 .backgroundColor('#F6F6F6')  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(rvp(10))  
 .height(rvp(60))  
}

### 14.1.2. 预约日期功能

1. 定义创建日期的函数 **utils > createDateList**

const DAY\_TIME = 1000 \* 3600 \* 24;  
  
export interface IDateItem {  
 date: string  
 week: string  
}  
export type IDateList = IDateItem[]  
  
  
function formatDate(time: number): IDateItem {  
 const date = new Date(time);  
 const year = date.getFullYear();  
 const month = addZero(date.getMonth() + 1);  
 const day = addZero(date.getDate());  
 const week = date.getDay();  
 return {  
 date: `${year}-${month}-${day}`,  
 week: getWeek(week)  
 }  
}  
  
function getWeek(week: number): string {  
 switch (week) {  
 case 0:  
 return '周日'  
 case 1:  
 return '周一'  
 case 2:  
 return '周二'  
 case 3:  
 return '周三'  
 case 4:  
 return '周四'  
 case 5:  
 return '周五'  
 case 6:  
 return '周六'  
 }  
}  
  
function addZero(val: number) {  
 return val < 10 ? '0' + val : `${val}`;  
}  
  
export default function createDateList(): IDateList {  
 const nowTime = Date.now();  
 return Array.from({ length: 7 }).map((item, index) => {  
 const result = formatDate(nowTime + index \* DAY\_TIME)  
 if (index === 0) { result.week = '今天' }  
 return result  
 })  
}

1. 预约日期展示功能

import Drawer from '../../components/Drawer'  
  
import { BORDER\_RADIUS, BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { md, xl, xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
import createDateList, { IDateItem, IDateList } from '../../utils/createDateList'  
  
  
@Component  
export default struct BookInfo {  
 @Link show: boolean  
 @State dateList: IDateList = createDateList()  
 @State currentDateIndex: number = 0  
  
 build() {  
 Drawer({ show: $show }) {  
 Column({ space: rvp(16) }) {  
 Text('预约信息').fontSize(xl()).fontWeight(700)  
 Column({ space: rvp(20) }) {  
 Row({ space: rvp(18) }) {  
 Text('姓名').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入姓名' }).textInputStyles()  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('手机号').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入手机号' })  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .textInputStyles()  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('预约日期').width(rvp(56)).fontSize(md())  
 Column({ space: rvp(20) }) {  
 Row() {  
 Text('2024-01-16（今天）').fontSize(md()).fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_down\_2')).width(rvp(10)).fillColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Grid() {  
 ForEach(this.dateList, (item, index) => {  
 GridItem() {  
 Column() {  
 Text(`(${item.week})`).fontSize(xs()).fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 Text(item.date).fontSize(xs()).fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 }.width(rvp(70)).height(rvp(32)).border({  
 width: 1,  
 radius: rvp(BORDER\_RADIUS\_S),  
 color: this.currentDateIndex === index ? '#24A17B' : '#E4E4E4'  
 }).justifyContent(FlexAlign.Center)  
 }  
 }, item => item.date)  
 }.columnsTemplate('1fr 1fr 1fr')  
 .columnsGap(rvp(6))  
 .rowsGap(rvp(10))  
 .height(rvp(3 \* 32 + 20))  
 }.layoutWeight(1)  
 }.width('100%').alignItems(VerticalAlign.Top)  
  
 Row({ space: rvp(18) }) {  
 Text('备注信息').width(rvp(56)).fontSize(md())  
 TextArea({ placeholder: '请输入备注信息' }).textAreaStyles()  
 }.width('100%').alignItems(VerticalAlign.Top)  
 }  
 .alignItems(HorizontalAlign.Start)  
 .backgroundColor($r('app.color.white'))  
 .padding(rvp(PADDING))  
  
 Button('预约看房')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(42))  
 .padding(rvp(13))  
 .fontColor($r('app.color.white'))  
 .fontSize(md())  
 .backgroundColor('#24A17B')  
 .borderRadius(rvp(2))  
 }.alignItems(HorizontalAlign.Start).padding({ bottom: rvp(PADDING) })  
 }  
 }  
}  
  
@Styles function pressedStyles() {  
 .backgroundColor('#F6F6F6')  
}  
  
@Extend(TextInput) function textInputStyles() {  
 .layoutWeight(1)  
 .backgroundColor(Color.Transparent)  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(0)  
}  
  
@Extend(TextArea) function textAreaStyles() {  
 .layoutWeight(1)  
 .backgroundColor('#F6F6F6')  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(rvp(10))  
 .height(rvp(60))  
}

1. 选择预约日期功能

import Drawer from '../../components/Drawer'  
  
import { BORDER\_RADIUS, BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { md, xl, xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
import createDateList, { IDateItem, IDateList } from '../../utils/createDateList'  
  
  
@Component  
export default struct BookInfo {  
 @Link show: boolean  
 @State dateList: IDateList = createDateList()  
 @State currentDateIndex: number = 0  
 setCurrentDateIndex(index: number) {  
 this.currentDateIndex = index  
 }  
  
 build() {  
 Drawer({ show: $show }) {  
 Column({ space: rvp(16) }) {  
 Text('预约信息').fontSize(xl()).fontWeight(700)  
 Column({ space: rvp(20) }) {  
 Row({ space: rvp(18) }) {  
 Text('姓名').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入姓名' }).textInputStyles()  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('手机号').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入手机号' })  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .textInputStyles()  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('预约日期').width(rvp(56)).fontSize(md())  
 Column({ space: rvp(20) }) {  
 Row() {  
 Text(`${this.dateList[this.currentDateIndex].date}（${this.dateList[this.currentDateIndex].week}）`).fontSize(md()).fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_down\_2')).width(rvp(10)).fillColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween)  
  
 Grid() {  
 ForEach(this.dateList, (item, index) => {  
 GridItem() {  
 Column() {  
 Text(`(${item.week})`).fontSize(xs()).fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 Text(item.date).fontSize(xs()).fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 }.width(rvp(70)).height(rvp(32)).border({  
 width: 1,  
 radius: rvp(BORDER\_RADIUS\_S),  
 color: this.currentDateIndex === index ? '#24A17B' : '#E4E4E4'  
 }).justifyContent(FlexAlign.Center)  
 }.onClick(this.setCurrentDateIndex.bind(this, index))  
 }, item => item.date)  
 }.columnsTemplate('1fr 1fr 1fr')  
 .columnsGap(rvp(6))  
 .rowsGap(rvp(10))  
 .height(rvp(3 \* 32 + 20))  
 }.layoutWeight(1)  
 }.width('100%').alignItems(VerticalAlign.Top)  
  
 Row({ space: rvp(18) }) {  
 Text('备注信息').width(rvp(56)).fontSize(md())  
 TextArea({ placeholder: '请输入备注信息' }).textAreaStyles()  
 }.width('100%').alignItems(VerticalAlign.Top)  
 }  
 .alignItems(HorizontalAlign.Start)  
 .backgroundColor($r('app.color.white'))  
 .padding(rvp(PADDING))  
  
 Button('预约看房')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(42))  
 .padding(rvp(13))  
 .fontColor($r('app.color.white'))  
 .fontSize(md())  
 .backgroundColor('#24A17B')  
 .borderRadius(rvp(2))  
 }.alignItems(HorizontalAlign.Start).padding({ bottom: rvp(PADDING) })  
 }  
 }  
}  
  
@Styles function pressedStyles() {  
 .backgroundColor('#F6F6F6')  
}  
  
@Extend(TextInput) function textInputStyles() {  
 .layoutWeight(1)  
 .backgroundColor(Color.Transparent)  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(0)  
}  
  
@Extend(TextArea) function textAreaStyles() {  
 .layoutWeight(1)  
 .backgroundColor('#F6F6F6')  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(rvp(10))  
 .height(rvp(60))  
}

1. 展开&关闭预约日期功能

import Drawer from '../../components/Drawer'  
  
import { BORDER\_RADIUS, BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { md, xl, xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
import createDateList, { IDateItem, IDateList } from '../../utils/createDateList'  
  
  
@Component  
export default struct BookInfo {  
 @Link show: boolean  
 @State dateList: IDateList = createDateList()  
 @State currentDateIndex: number = 0  
 @State isCollapseDateList: boolean = true  
 setCurrentDateIndex(index: number) {  
 this.currentDateIndex = index  
 }  
 expandDateList() {  
 this.isCollapseDateList = !this.isCollapseDateList  
 }  
  
 build() {  
 Drawer({ show: $show }) {  
 Column({ space: rvp(16) }) {  
 Text('预约信息').fontSize(xl()).fontWeight(700)  
 Column({ space: rvp(20) }) {  
 Row({ space: rvp(18) }) {  
 Text('姓名').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入姓名' }).textInputStyles()  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('手机号').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入手机号' })  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .textInputStyles()  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('预约日期').width(rvp(56)).fontSize(md())  
 Column({ space: rvp(20) }) {  
 Row() {  
 Text(`${this.dateList[this.currentDateIndex].date}（${this.dateList[this.currentDateIndex].week}）`).fontSize(md()).fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_down\_2')).width(rvp(10)).fillColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween).onClick(this.expandDateList.bind(this))  
 if (!this.isCollapseDateList) {  
 Grid() {  
 ForEach(this.dateList, (item, index) => {  
 GridItem() {  
 Column() {  
 Text(`(${item.week})`).fontSize(xs()).fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 Text(item.date).fontSize(xs()).fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 }.width(rvp(70)).height(rvp(32)).border({  
 width: 1,  
 radius: rvp(BORDER\_RADIUS\_S),  
 color: this.currentDateIndex === index ? '#24A17B' : '#E4E4E4'  
 }).justifyContent(FlexAlign.Center)  
 }.onClick(this.setCurrentDateIndex.bind(this, index))  
 }, item => item.date)  
 }.columnsTemplate('1fr 1fr 1fr')  
 .columnsGap(rvp(6))  
 .rowsGap(rvp(10))  
 .height(rvp(3 \* 32 + 20))  
 }  
 }.layoutWeight(1)  
 }.width('100%').alignItems(VerticalAlign.Top)  
  
 Row({ space: rvp(18) }) {  
 Text('备注信息').width(rvp(56)).fontSize(md())  
 TextArea({ placeholder: '请输入备注信息' }).textAreaStyles()  
 }.width('100%').alignItems(VerticalAlign.Top)  
 }  
 .alignItems(HorizontalAlign.Start)  
 .backgroundColor($r('app.color.white'))  
 .padding(rvp(PADDING))  
  
 Button('预约看房')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(42))  
 .padding(rvp(13))  
 .fontColor($r('app.color.white'))  
 .fontSize(md())  
 .backgroundColor('#24A17B')  
 .borderRadius(rvp(2))  
 }.alignItems(HorizontalAlign.Start).padding({ bottom: rvp(PADDING) })  
 }  
 }  
}  
  
@Styles function pressedStyles() {  
 .backgroundColor('#F6F6F6')  
}  
  
@Extend(TextInput) function textInputStyles() {  
 .layoutWeight(1)  
 .backgroundColor(Color.Transparent)  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(0)  
}  
  
@Extend(TextArea) function textAreaStyles() {  
 .layoutWeight(1)  
 .backgroundColor('#F6F6F6')  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(rvp(10))  
 .height(rvp(60))  
}

### 14.1.3. 收集数据

1. 定义参数类型 **model > BookRentRoomList**

export interface IBookRentRoomItem {  
 "id": number  
 "userId": number  
 "houseId": string  
 "date": string  
 "name": string  
 "comment": string  
}  
  
export type IBookRentRoomList = IBookRentRoomItem[]  
  
export interface IBookInfo {  
 "name": string  
 "date": string  
 "houseId": string  
 "phone": string  
 "comment": string  
}

1. 收集姓名、手机号、备注数据

import Drawer from '../../components/Drawer'  
  
import { BORDER\_RADIUS, BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { md, xl, xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
import createDateList, { IDateItem, IDateList } from '../../utils/createDateList'  
import type { IBookInfo } from '../../model/BookRentRoomList'  
  
@Component  
export default struct BookInfo {  
 @Link show: boolean  
 @State dateList: IDateList = createDateList()  
 @State currentDateIndex: number = 0  
 @State isCollapseDateList: boolean = true  
 bookInfo: Partial<IBookInfo> = {}  
  
 handleNameChange(name) {  
 this.bookInfo.name = name;  
 }  
  
 handlePhoneChange(phone) {  
 this.bookInfo.phone = phone;  
 }  
  
 handleCommentChange(comment) {  
 this.bookInfo.comment = comment;  
 }  
  
 setCurrentDateIndex(index: number) {  
 this.currentDateIndex = index  
 }  
  
 expandDateList() {  
 this.isCollapseDateList = !this.isCollapseDateList  
 }  
  
 build() {  
 Drawer({ show: $show }) {  
 Column({ space: rvp(16) }) {  
 Text('预约信息').fontSize(xl()).fontWeight(700)  
 Column({ space: rvp(20) }) {  
 Row({ space: rvp(18) }) {  
 Text('姓名').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入姓名' }).textInputStyles().onChange(this.handleNameChange.bind(this))  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('手机号').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入手机号' })  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .textInputStyles()  
 .onChange(this.handlePhoneChange.bind(this))  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('预约日期').width(rvp(56)).fontSize(md())  
 Column({ space: rvp(20) }) {  
 Row() {  
 Text(`${this.dateList[this.currentDateIndex].date}（${this.dateList[this.currentDateIndex].week}）`)  
 .fontSize(md())  
 .fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_down\_2')).width(rvp(10)).fillColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween).onClick(this.expandDateList.bind(this))  
  
 if (!this.isCollapseDateList) {  
 Grid() {  
 ForEach(this.dateList, (item, index) => {  
 GridItem() {  
 Column() {  
 Text(`(${item.week})`)  
 .fontSize(xs())  
 .fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 Text(item.date)  
 .fontSize(xs())  
 .fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 }.width(rvp(70)).height(rvp(32)).border({  
 width: 1,  
 radius: rvp(BORDER\_RADIUS\_S),  
 color: this.currentDateIndex === index ? '#24A17B' : '#E4E4E4'  
 }).justifyContent(FlexAlign.Center)  
 }.onClick(this.setCurrentDateIndex.bind(this, index))  
 }, item => item.date)  
 }.columnsTemplate('1fr 1fr 1fr')  
 .columnsGap(rvp(6))  
 .rowsGap(rvp(10))  
 .height(rvp(3 \* 32 + 20))  
 }  
 }.layoutWeight(1)  
 }.width('100%').alignItems(VerticalAlign.Top)  
  
 Row({ space: rvp(18) }) {  
 Text('备注信息').width(rvp(56)).fontSize(md())  
 TextArea({ placeholder: '请输入备注信息' }).textAreaStyles().onChange(this.handleCommentChange.bind(this))  
 }.width('100%').alignItems(VerticalAlign.Top)  
 }  
 .alignItems(HorizontalAlign.Start)  
 .backgroundColor($r('app.color.white'))  
 .padding(rvp(PADDING))  
  
 Button('预约看房')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(42))  
 .padding(rvp(13))  
 .fontColor($r('app.color.white'))  
 .fontSize(md())  
 .backgroundColor('#24A17B')  
 .borderRadius(rvp(2))  
 }.alignItems(HorizontalAlign.Start).padding({ bottom: rvp(PADDING) })  
 }  
 }  
}  
  
@Styles function pressedStyles() {  
 .backgroundColor('#F6F6F6')  
}  
  
@Extend(TextInput) function textInputStyles() {  
 .layoutWeight(1)  
 .backgroundColor(Color.Transparent)  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(0)  
}  
  
@Extend(TextArea) function textAreaStyles() {  
 .layoutWeight(1)  
 .backgroundColor('#F6F6F6')  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(rvp(10))  
 .height(rvp(60))  
}

### 14.1.4. 完成预约看房功能

1. 定义接口函数 **api > book**

import request from '../utils/request'  
import type { IBookInfo } from '../model/BookRentRoomList'  
  
// 预约看房  
export const bookRentRoomApi = (bookInfo: IBookInfo) => {  
 return request.post<any, any>('/auth/house/reservation', bookInfo)  
}

1. 组件点击按钮，完成功能

import Drawer from '../../components/Drawer'  
  
import { BORDER\_RADIUS, BORDER\_RADIUS\_S, PADDING } from '../../constants/size'  
import { md, xl, xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
import createDateList, { IDateItem, IDateList } from '../../utils/createDateList'  
import type { IBookInfo } from '../../model/BookRentRoomList'  
import { bookRentRoomApi } from '../../api/book'  
import showToast from '../../utils/showToast'  
import { PHONE\_REG } from '../../constants/regs'  
  
@Component  
export default struct BookInfo {  
 @Prop roomId: string;  
 @Link show: boolean  
 @State dateList: IDateList = createDateList()  
 @State currentDateIndex: number = 0  
 @State isCollapseDateList: boolean = true  
 bookInfo: Partial<IBookInfo> = {}  
  
 handleNameChange(name) {  
 this.bookInfo.name = name;  
 }  
  
 handlePhoneChange(phone) {  
 this.bookInfo.phone = phone;  
 }  
  
 handleCommentChange(comment) {  
 this.bookInfo.comment = comment;  
 }  
  
 setCurrentDateIndex(index: number) {  
 this.currentDateIndex = index  
 }  
  
 expandDateList() {  
 this.isCollapseDateList = !this.isCollapseDateList  
 }  
  
 // 预约看房  
 async book() {  
 const { name, phone, comment } = this.bookInfo;  
 if (!name) {  
 showToast('请输入姓名')  
 return;  
 }  
 if (!phone) {  
 showToast('请输入手机号')  
 return;  
 }  
 if (!PHONE\_REG.test(phone)) {  
 showToast('请输入合法的手机号')  
 return;  
 }  
 try {  
 await bookRentRoomApi({  
 name,  
 date: this.dateList[this.currentDateIndex].date,  
 houseId: this.roomId,  
 phone,  
 comment  
 })  
 showToast('预约成功')  
 setTimeout(() => {  
 this.show = false  
 }, 1000)  
 } catch {  
 showToast('预约失败，时间冲突')  
 }  
 }  
  
 build() {  
 Drawer({ show: $show }) {  
 Column({ space: rvp(16) }) {  
 Text('预约信息').fontSize(xl()).fontWeight(700)  
 Column({ space: rvp(20) }) {  
 Row({ space: rvp(18) }) {  
 Text('姓名').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入姓名' }).textInputStyles().onChange(this.handleNameChange.bind(this))  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('手机号').width(rvp(56)).fontSize(md())  
 TextInput({ placeholder: '请输入手机号' })  
 .type(InputType.PhoneNumber)  
 .maxLength(11)  
 .textInputStyles()  
 .onChange(this.handlePhoneChange.bind(this))  
 }.width('100%')  
  
 Row({ space: rvp(18) }) {  
 Text('预约日期').width(rvp(56)).fontSize(md())  
 Column({ space: rvp(20) }) {  
 Row() {  
 Text(`${this.dateList[this.currentDateIndex].date}（${this.dateList[this.currentDateIndex].week}）`)  
 .fontSize(md())  
 .fontColor($r('app.color.gray'))  
 Image($r('app.media.arrow\_down\_2')).width(rvp(10)).fillColor($r('app.color.gray'))  
 }.width('100%').justifyContent(FlexAlign.SpaceBetween).onClick(this.expandDateList.bind(this))  
  
 if (!this.isCollapseDateList) {  
 Grid() {  
 ForEach(this.dateList, (item, index) => {  
 GridItem() {  
 Column() {  
 Text(`(${item.week})`)  
 .fontSize(xs())  
 .fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 Text(item.date)  
 .fontSize(xs())  
 .fontColor(this.currentDateIndex === index ? '#1C916D' : $r('app.color.gray'))  
 }.width(rvp(70)).height(rvp(32)).border({  
 width: 1,  
 radius: rvp(BORDER\_RADIUS\_S),  
 color: this.currentDateIndex === index ? '#24A17B' : '#E4E4E4'  
 }).justifyContent(FlexAlign.Center)  
 }.onClick(this.setCurrentDateIndex.bind(this, index))  
 }, item => item.date)  
 }.columnsTemplate('1fr 1fr 1fr')  
 .columnsGap(rvp(6))  
 .rowsGap(rvp(10))  
 .height(rvp(3 \* 32 + 20))  
 }  
 }.layoutWeight(1)  
 }.width('100%').alignItems(VerticalAlign.Top)  
  
 Row({ space: rvp(18) }) {  
 Text('备注信息').width(rvp(56)).fontSize(md())  
 TextArea({ placeholder: '请输入备注信息' }).textAreaStyles().onChange(this.handleCommentChange.bind(this))  
 }.width('100%').alignItems(VerticalAlign.Top)  
 }  
 .alignItems(HorizontalAlign.Start)  
 .backgroundColor($r('app.color.white'))  
 .padding(rvp(PADDING))  
  
 Button('预约看房')  
 .type(ButtonType.Normal)  
 .width('100%')  
 .height(rvp(42))  
 .padding(rvp(13))  
 .fontColor($r('app.color.white'))  
 .fontSize(md())  
 .backgroundColor('#24A17B')  
 .borderRadius(rvp(2))  
 .onClick(this.book.bind(this))  
 }.alignItems(HorizontalAlign.Start).padding({ bottom: rvp(PADDING) })  
 }  
 }  
}  
  
@Styles function pressedStyles() {  
 .backgroundColor('#F6F6F6')  
}  
  
@Extend(TextInput) function textInputStyles() {  
 .layoutWeight(1)  
 .backgroundColor(Color.Transparent)  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS\_S))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(0)  
}  
  
@Extend(TextArea) function textAreaStyles() {  
 .layoutWeight(1)  
 .backgroundColor('#F6F6F6')  
 .placeholderColor($r('app.color.gray'))  
 .placeholderFont({ size: md() })  
 .caretColor($r('app.color.gray'))  
 .fontSize(md())  
 .borderRadius(rvp(BORDER\_RADIUS))  
 .stateStyles({  
 pressed: pressedStyles,  
 })  
 .padding(rvp(10))  
 .height(rvp(60))  
}

1. 预约看房的按钮权限控制（没有登录，先去登录）

import router from '@ohos.router';  
import { xs } from '../../utils/responsive/responsiveFontSize'  
import rvp from '../../utils/responsive/responsiveVP'  
  
@Component  
export default struct Book {  
 @Link show: boolean  
  
 @StorageProp('token') token: string = '';  
  
 build() {  
 Row() {  
 Column({ space: rvp(4) }) {  
 Image($r('app.media.my\_icon\_1')).width(rvp(18))  
 Text('99+').fontSize(xs()).fontColor($r('app.color.gray'))  
 }  
  
 Column({ space: rvp(4) }) {  
 Image($r('app.media.message')).width(rvp(18)).fillColor($r('app.color.black'))  
 Text('咨询').fontSize(xs()).fontColor($r('app.color.gray'))  
 }  
  
 Button('立即预定', { type: ButtonType.Normal })  
 .borderRadius(rvp(2))  
 .backgroundColor('#24A17B')  
 .padding({ left: rvp(30), right: rvp(30) })  
 .height(rvp(34))  
 .onClick(() => {  
 if (!this.token) {  
 router.pushUrl({ url: 'pages/Login/LoginPhone' })  
 return  
 }  
 this.show = true  
 })  
 Button('去看房', { type: ButtonType.Normal })  
 .borderRadius(rvp(2))  
 .backgroundColor('#6BE3BE')  
 .padding({ left: rvp(30), right: rvp(30) })  
 .height(rvp(34))  
 }.backgroundColor($r('app.color.white')).height(rvp(50)).justifyContent(FlexAlign.SpaceEvenly).width('100%')  
 }  
}

## 14.2. 预约租房列表功能

### 14.2.1. 发送请求，更新数据

1. 新建路由组件 **pages > BookRentRoomList**

import ScrollContainer from '../components/ScrollContainer'  
  
@Entry  
@Component  
struct BookRentRoomList {  
 @Builder NavBarRender() {}  
  
 @Builder ContentRender() {}  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

1. 我的页面点击按钮跳转到预约租房列表页面

import Card from '../../components/Card'  
import { PADDING } from '../../constants/size'  
import rvp from '../../utils/responsive/responsiveVP'  
import { sm } from '../../utils/responsive/responsiveFontSize'  
  
import type { INavList, INavItem } from '../../model/NavList'  
import router from '@ohos.router'  
  
@Component  
export default struct Order {  
  
 @State navList1: INavList = [  
 {  
 id: 1,  
 text: '想看',  
 icon: $r('app.media.my\_icon\_1')  
 },  
 {  
 id: 2,  
 text: '足迹',  
 icon: $r('app.media.my\_icon\_2')  
 },  
 {  
 id: 3,  
 text: '约看',  
 icon: $r('app.media.my\_icon\_3')  
 },  
 {  
 id: 4,  
 text: '拼租',  
 icon: $r('app.media.my\_icon\_4')  
 }  
 ]  
  
 @State navList2: INavList = [  
 {  
 id: 1,  
 text: '合同',  
 icon: $r('app.media.my\_icon\_5')  
 },  
 {  
 id: 2,  
 text: '账单',  
 icon: $r('app.media.my\_icon\_6')  
 },  
 {  
 id: 3,  
 text: '订单',  
 icon: $r('app.media.my\_icon\_7')  
 },  
 {  
 id: 4,  
 text: '评价',  
 icon: $r('app.media.my\_icon\_8')  
 },  
 {  
 id: 5,  
 text: '钱包',  
 icon: $r('app.media.my\_icon\_9')  
 }  
 ]  
  
 build() {  
 Column() {  
 Card({ cardPadding: rvp(PADDING) }) {  
 Row({ space: rvp(38) }) {  
 ForEach(this.navList1, (nav: INavItem) => {  
 Row({ space: rvp(4) }) {  
 Image(nav.icon).width(rvp(16)).height(rvp(16))  
 Text(nav.text).fontSize(sm()).fontColor($r('app.color.black'))  
 }.onClick(() => {  
 if (nav.text === '约看') {  
 router.pushUrl({ url: 'pages/BookRentRoomList' })  
 }  
 })  
 }, nav => nav.id)  
 }.width('100%').justifyContent(FlexAlign.Center)  
 Divider().margin({ top: rvp(16) })  
 Row({ space: rvp(39) }) {  
 ForEach(this.navList2, (nav: INavItem) => {  
 Column({ space: rvp(6) }) {  
 Image(nav.icon).width(rvp(16)).height(rvp(16))  
 Text('合同').fontSize(sm()).fontColor($r('app.color.black'))  
 }  
 }, nav => nav.id)  
 }.margin({ top: rvp(12) }).justifyContent(FlexAlign.Center).width('100%')  
 }  
 }.margin({ top: rvp(10) })  
 }  
}

1. 定义接口函数和类型

* **model > BookRentRoomList**

export interface IBookRentRoomItem {  
 "id": string  
 "date": string  
 "houseId": string  
 "img": string  
 "title": string  
 "subTitle": string  
 "distanceInfo": null | string,  
 "rentPrice": string,  
 "comment": string  
}  
  
export type IBookRentRoomList = IBookRentRoomItem[]  
  
export interface IBookInfo {  
 "name": string  
 "date": string  
 "houseId": string  
 "phone": string  
 "comment": string  
}

* **api > book**

import request from '../utils/request'  
import type { IBookInfo, IBookRentRoomList } from '../model/BookRentRoomList'  
  
// 预约看房  
export const bookRentRoomApi = (bookInfo: IBookInfo) => {  
 return request.post<any, any>('/auth/house/reservation', bookInfo)  
}  
  
// 获取预约租房列表  
export const getBookRentRoomListApi = () => {  
 return request.get<any, IBookRentRoomList>('/auth/house/history')  
}

1. 组件发送请求，获取数据

import ScrollContainer from '../components/ScrollContainer'  
  
import { getBookRentRoomListApi } from '../api/book'  
import type { IBookRentRoomList } from '../model/BookRentRoomList'  
  
@Entry  
@Component  
struct BookRentRoomList {  
  
 @State bookList: IBookRentRoomList = []  
 aboutToAppear() {  
 this.getBookRentRoomList()  
 }  
 async getBookRentRoomList() {  
 this.bookList = await getBookRentRoomListApi()  
 }  
  
 @Builder NavBarRender() {}  
  
 @Builder ContentRender() {}  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

### 14.2.2. NavBar 组件

import ScrollContainer from '../components/ScrollContainer'  
import NavBar from '../components/NavBar'  
  
import { getBookRentRoomListApi } from '../api/book'  
import type { IBookRentRoomList } from '../model/BookRentRoomList'  
  
@Entry  
@Component  
struct BookRentRoomList {  
  
 @State bookList: IBookRentRoomList = []  
 aboutToAppear() {  
 this.getBookRentRoomList()  
 }  
 async getBookRentRoomList() {  
 this.bookList = await getBookRentRoomListApi()  
 }  
  
 @Builder NavBarRender() {  
 NavBar({ title: '预约看房' })  
 }  
  
 @Builder ContentRender() {  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}

### 14.2.3. 完成功能

import ScrollContainer from '../components/ScrollContainer'  
import NavBar from '../components/NavBar'  
import SafeArea from '../components/SafeArea'  
  
import { getBookRentRoomListApi } from '../api/book'  
import type { IBookRentRoomItem, IBookRentRoomList } from '../model/BookRentRoomList'  
  
import { sm, xs } from '../utils/responsive/responsiveFontSize'  
import rvp from '../utils/responsive/responsiveVP'  
import { PADDING } from '../constants/size'  
  
@Entry  
@Component  
struct BookRentRoomList {  
 @State bookList: IBookRentRoomList = []  
  
 aboutToAppear() {  
 this.getBookRentRoomList()  
 }  
  
 async getBookRentRoomList() {  
 this.bookList = await getBookRentRoomListApi()  
 }  
  
 @Builder NavBarRender() {  
 NavBar({ title: '预约看房' })  
 }  
  
 @Builder ContentRender() {  
 SafeArea()  
 Row() {  
 }.height(rvp(44))  
  
 List({ space: rvp(16) }) {  
 ForEach(this.bookList, (book: IBookRentRoomItem) => {  
 ListItem() {  
 Column({ space: rvp(4) }) {  
 Row() {  
 Text(book.date + '全天 随时可看').fontSize(xs()).fontColor($r('app.color.gray'))  
 }  
 Row({ space: rvp(8) }) {  
 Image(book.img).width(rvp(120)).height(rvp(120)).objectFit(ImageFit.Fill).borderRadius(rvp(12))  
 Column() {  
 Column({ space: rvp(8) }) {  
 Text(book.title).maxLines(2).textOverflow({ overflow: TextOverflow.Ellipsis })  
 Text(book.subTitle).fontSize(xs()).fontColor($r('app.color.gray'))  
 Text(book.distanceInfo).fontSize(xs()).fontColor($r('app.color.gray'))  
 }.alignItems(HorizontalAlign.Start).width('100%')  
 Row() {  
 Text(book.rentPrice).fontColor('#E03810')  
 Text('去咨询').fontColor($r('app.color.primary')).fontSize(sm())  
 }.justifyContent(FlexAlign.SpaceBetween).width('100%')  
 }.width(rvp(200)).height(rvp(120)).padding({ top: rvp(10), bottom: rvp(10) }).justifyContent(FlexAlign.SpaceBetween).alignItems(HorizontalAlign.Start)  
 }  
 }.alignItems(HorizontalAlign.Start).width('100%')  
 }.width('100%')  
 }, book => book.id)  
 }.padding(rvp(PADDING))  
 }  
  
 build() {  
 Column() {  
 ScrollContainer({  
 NavBarRender: this.NavBarRender.bind(this),  
 ContentRender: this.ContentRender.bind(this),  
 })  
 }.height('100%')  
 }  
}