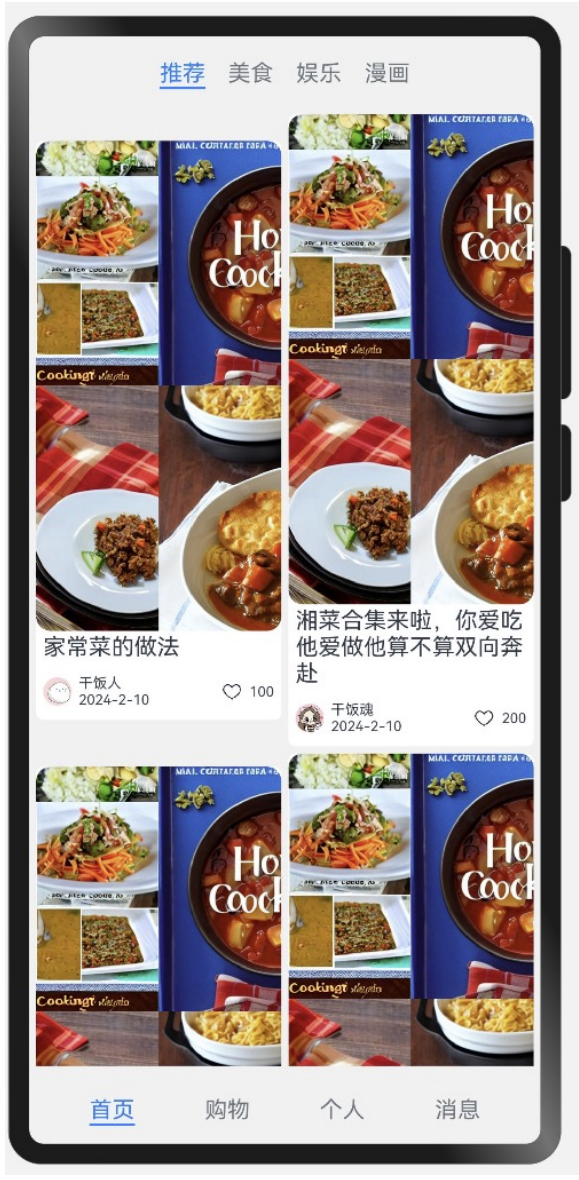
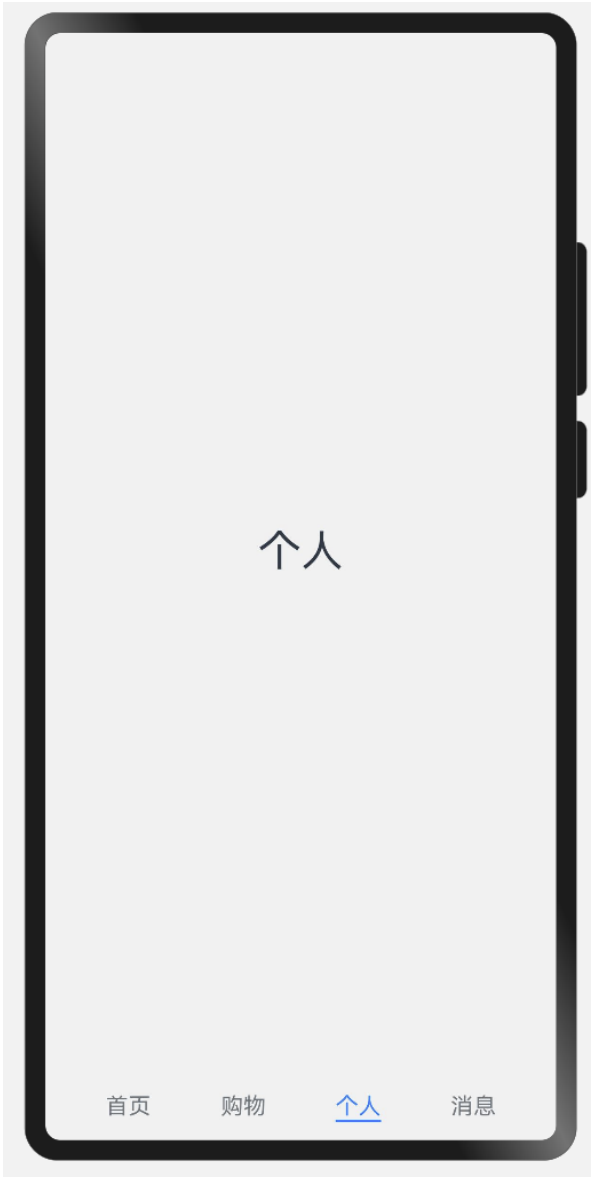
主页面开发

主页面 UI 分析


推荐
美食
娱乐
漫画
家常菜的做法
干饭人
2024-2-10
100
湘菜合集来啦，你爱吃他爱做他算不算双向奔赴
干饭魂
2024-2-10
200
首页
购物
个人
消息

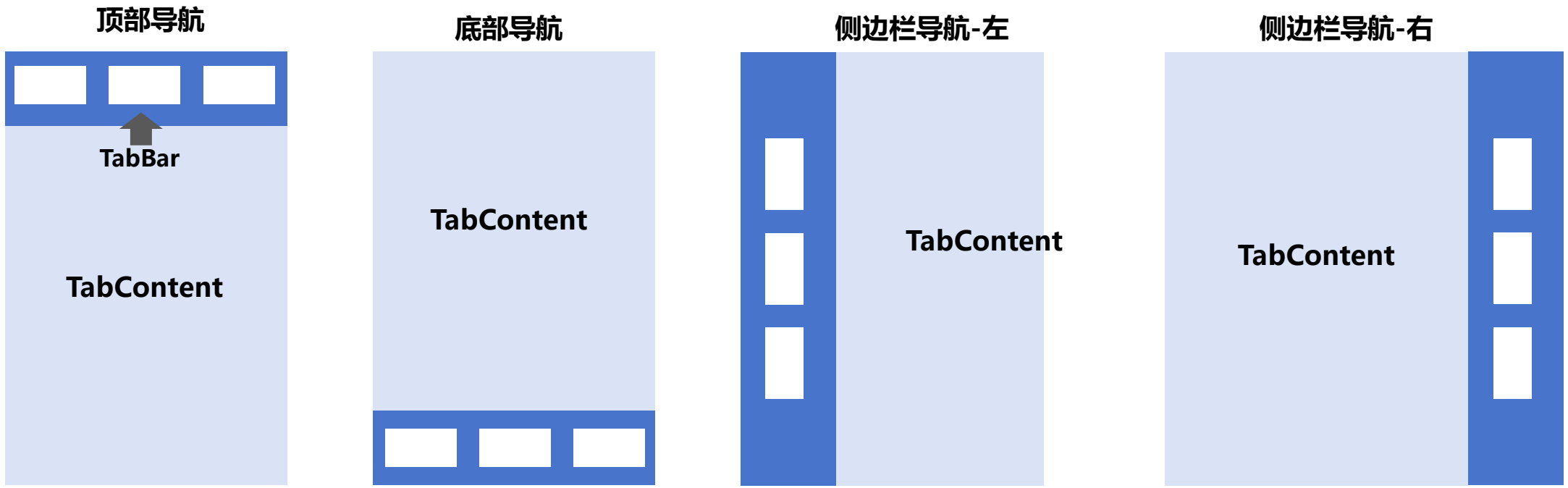

购物
首页
购物
个人
消息


个人
首页
购物
个人
消息


消息
首页
购物
个人
消息

Tabs 搭建主页面

Tabs 组件介绍


顶部导航
TabBar
TabContent
底部导航
TabContent
侧边栏导航-左
TabContent
侧边栏导航-右
TabContent

Tabs 组件介绍

Tabs组件

- 导航栏位置使用Tabs组件的参数**barPosition**进行设置，其值有2个：默认值**Start**，导航栏位于顶部。**End**，导航栏位于底部。
- 实现侧边导航栏需要设置Tabs的属性**vertical为true**。在顶部导航栏中设置vertical为true则导航栏在左侧，在底部导航栏中设置vertical为true则导航栏在右侧，默认在左侧。侧边导航栏多用于平板横屏界面

TabContent组件

在Tabs组件中使用花括号包裹TabContent，每一个TabContent对应的内容需要有一个**页签，通过**TabContent的**tabBar属性进行配置**。TabContent组件不支持设置宽高属性，其宽度默认撑满Tabs父组件，高度由Tabs父组件高度与TabBar组件高度决定。

Tabs 组件-接口定义

```
interface TabsInterface {
    /**
     * Called when the view is switched.
     * @since 7
     */
    (value?: {
        barPosition?: BarPosition;
        index?: number;
        controller?: TabsController;
    }): TabsAttribute;
}
```

参数说明：

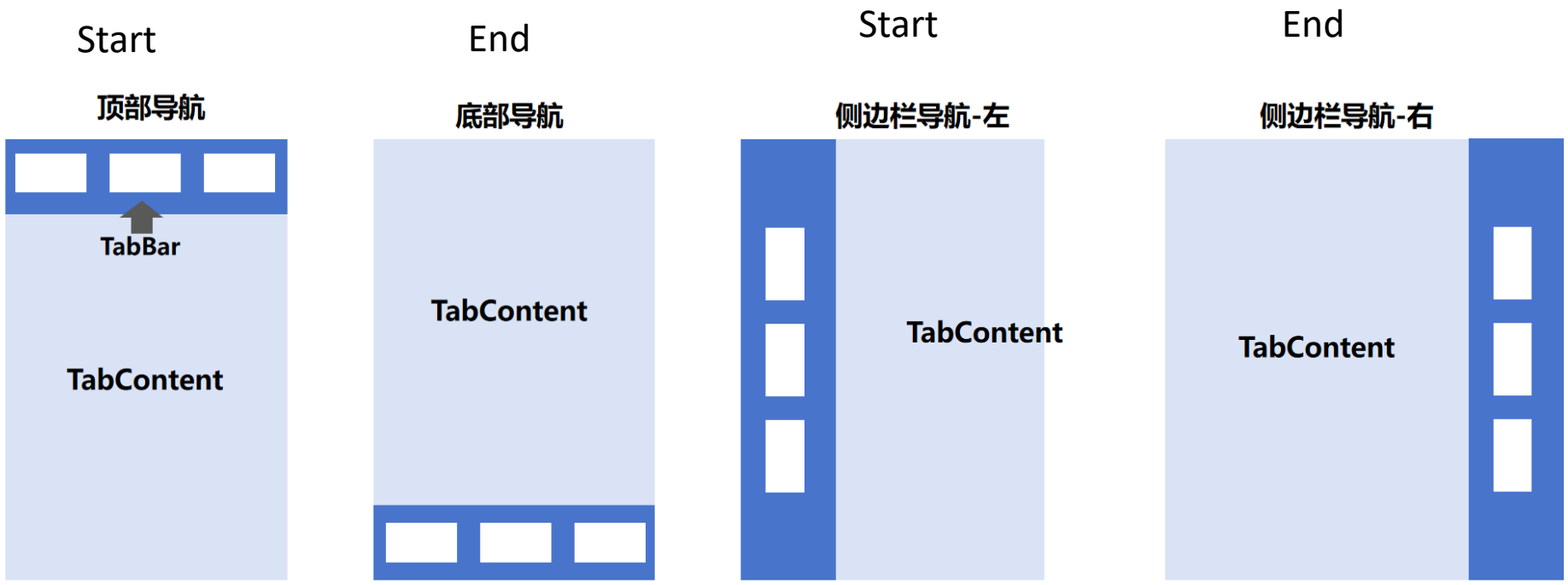
- barPosition：位置
- index：默认显示的索引
- controller：控制器

TabsController()的方法

```
changeIndex(value: number): void;
```

Tabs 组件-属性方法

vertical(value: boolean): TabsAttribute;
是否垂直显示
barPosition(value: BarPosition): TabsAttribute;
设置位置



Tabs 组件-属性方法

scrollable(value: boolean): TabsAttribute;
设置是否可以滚动
barMode(value: BarMode): TabsAttribute;
设置Bar的模式

BarMode 枚举
- Scrollable
- Fixed

barWidth(value: Length): TabsAttribute;
设置Bar的宽度
barHeight(value: Length): TabsAttribute;
设置Bar的高度
animationDuration(value: number): TabsAttribute;
设置Bar切换的动画时长

Tabs 组件-事件方法

onChange(event: (index: number) => void): TabsAttribute;
当发生切换时触发的回调方法，返回切换的索引。

Grid组件

Grid 组件-接口定义

```
interface GridInterface {
    /**
     * Grid is returned when the parameter is transferred.
     * @since 7
     */
    (scroller?: Scroller): GridAttribute;
}
```

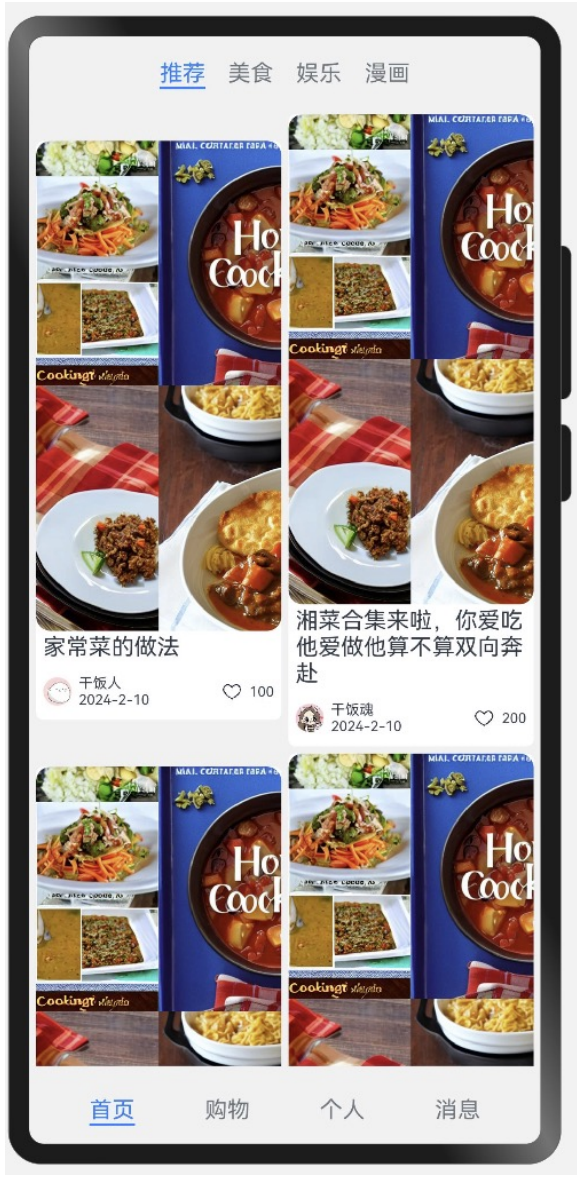
接收一个参数 Scroller

GridItem 子组件

Grid 组件-属性方法

columnsTemplate(value: string): GridAttribute;
列模板
rowsTemplate(value: string): GridAttribute;
行模板
columnsGap(value: Length): GridAttribute;
列间距
rowsGap(value: Length): GridAttribute;
行间距
scrollBarWidth(value: number | string): GridAttribute;
滚动条的宽度
scrollBarColor(value: Color | number | string): GridAttribute;
滚动条的颜色
scrollBar(value: BarState): GridAttribute;
滚动条的状态 //On/Off/Auto

主页面具体实现


推荐
美食
娱乐
漫画
家常菜的做法
干饭人
2024-2-10
100
湘菜合集来啦，你爱吃他爱做他算不算双向奔赴
干饭魂
2024-2-10
200
首页
购物
个人
消息


购物
首页
购物
个人
消息


个人
首页
购物
个人
消息


消息
首页
购物
个人
消息

侧边栏抽屉效果

SideBarContainer-组件-接口定义

SideBarContainer(type?: SideBarContainerType)

名称	描述
Embed	侧边栏嵌入到组件内，和内容区并列显示。
Overlay	侧边栏浮在内容区上面。

正常状态

Overlay

Embed

SideBarContainer-组件-属性方法

showSideBar(value: boolean): SideBarContainerAttribute;
是否显示侧边栏，默认true
controlButton(value: ButtonStyle): SideBarContainerAttribute;
设置侧边栏控制按钮的属性
showControlButton(value: boolean): SideBarContainerAttribute;
是否显示控制按钮，默认true
sideBarWidth(value: number): SideBarContainerAttribute;
侧边栏的宽度
autoHide(value: boolean): SideBarContainerAttribute;
当侧边栏拖拽到小于最小宽度后，是否自动隐藏，默认true
sideBarPosition(value: SideBarPosition): SideBarContainerAttribute;
设置侧边栏的位置，默认Start，也就是左侧

SideBarContainer-组件-事件方法

onChange(callback: (value: boolean) => void): SideBarContainerAttribute;
当侧边栏的状态在显示和隐藏之间切换时触发回调。true表示显示，false表示隐藏。

数据持久化之首选项

首选项存储-简介

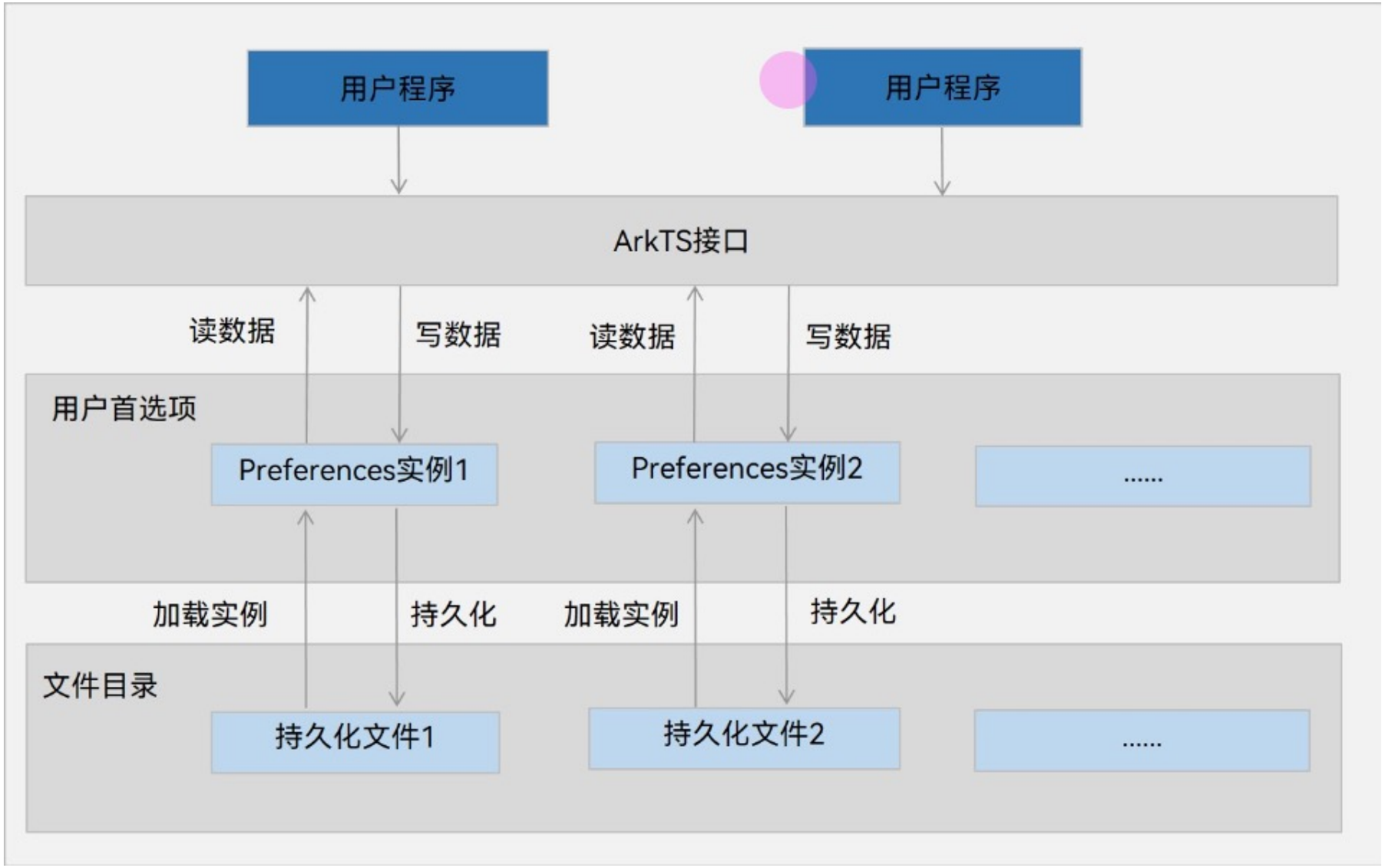
简介

用户首选项为应用提供Key-Value键值型的数据处理能力，支持应用持久化轻量级数据，并对其修改和查询。当用户希望有一个全局唯一存储的地方，可以采用用户首选项来进行存储。

Preferences会将该数据缓存在内存中，当需要持久化时可以使用flush接口将内存中的数据写入持久化文件中。

Preferences会随着存放的数据量越多而导致应用占用的内存越大，因此，Preferences不适合存放过多的数据。

首选项存储-运作机制图


用户程序
用户程序
ArkTS接口
读数据
写数据
读数据
写数据
用户首选项
Preferences实例1
Preferences实例2
.....
加载实例
持久化
加载实例
持久化
文件目录
持久化文件1
持久化文件2
.....

首选项存储-使用说明

1、导包

```
import dataPreferences from '@ohos.data.preferences';
```

2、获取首选项存储实例

3、调用首选项存储实例提供的put、get、delete方法进行存、取、删除等操作。

使用注意事项：

- Key键为string类型，要求非空且长度不超过80个字节。
- 如果Value值为string类型，可以为空，不为空时长度不超过8192个字节。
- 内存会随着存储数据量的增大而增大，所以存储的数据量应该是轻量级的，建议存储的数据不超过一万条，否则会在内存方面产生较大的开销。

首选项存储-接口说明

getPreferences(context: Context, name: string, callback: AsyncCallback<Preferences>): void

获取 Preferences 实例

put(key: string, value: ValueType, callback: AsyncCallback<void>): void

写入，可通过flush将Preferences实例持久化。

flush(callback: AsyncCallback<void>): void

将数据存储到文件中。

get(key: string, defValue: ValueType, callback: AsyncCallback<ValueType>): void

读取，根据key取值

delete(key: string, callback: AsyncCallback<void>): void

删除，根据key 删除记录

deletePreferences(context: Context, name: string, callback: AsyncCallback<void>): void

删除Preferences 实例