基于 ArkUI 的声明式 UI 与传统命令式 UI 开发对比研究

孙磊
阜阳职业技术学院信息与智能制造学院，安徽 阜阳 236000
DOI: 10.61369/TACS.2026040032

摘　　要： 早期的 Android 应用开发，是以命令式开发的方式实现 UI 组件的构建。随着页面功能的复杂化，复杂交互和状态同步过程中代码冗余问题也随之而来。声明式开发框架通过组合函数进行 UI 定义，实现由状态变量驱动 UI 组件更新。本文以基于 ArkUI 的声明式 UI 与传统的命令式 UI 为研究对象，对比分析两种不同开发范式的核心构建逻辑。通过一个手机新闻列表应用的开发实践，分析两者的优点。实验证明声明式开发相较于命令式开发，其 UI 结构和组件之间的状态依赖比较简单、开发效率高、代码可维护性高、状态管理更佳。

关 键 词： ArkUI；声明式 UI；命令式 UI；开发对比

A Comparative Study on Declarative UI Based on ArkUI and Traditional Imperative UI Development

Sun Lei
Fuyang Institute of Technology, Fuyang, Anhui 236000

Abstract: The UI component construction was achieved through imperative development method in the early days of Android application development. While the page functionalities are becoming more and more complex, here come the issues of code redundancy during complex interactions and state synchronization. The declarative development frameworks define the UI through composition functions, realizing the state variables to make UI components update. This thesis compares and analyzes the core constructions of two different development methods: the declarative UI based on the ArkUI and the traditional imperative UI. The thesis compares the two method and analyzes their respective advantages by the development practice of a mobile news list application. The result shows that the declarative development, compared to the imperative development, has a better UI structure and state dependencies between components, higher development efficiency, and higher code maintainability.

Keywords: ArkUI; declarative UI; imperative UI; development comparison

一、背景

在移动互联网高速发展的当下，智能手机、平板、智能穿戴设备已经成为人们日常生活中必不可少的一部分。各种各样的应用程序如雨后春笋般出现，市场竞争愈发激烈。应用服务质量和用户满意度成为应用程序发展的核心竞争力[1]。良好的用户体验（User Experience，UX）和用户界面（User Interface，UI）是确保手机 App 在激烈竞争中能脱颖而出的重要因素。UI 界面既要保证视觉的美观，还要保证与用户的交互性。在此过程中，前后端开发需要紧密地协作，以确保应用的运作和良好的用户使用体验[2]。

移动互联网时代，Android 手机占据半壁江山，涌现了众多基于 Android 开发的手机应用 App。早期 Android 开发时，基本上是以命令式开发的方式实现 UI 页面的构建。开发者通过 XML 布局文件定义 UI 页面的结构，然后在 Activity 文件中通过组件 id 获得 UI 组件对象，最后对 UI 组件添加监听事件等。这种 UI 构建方式在简单页面时尚可使用，但随着页面功能的复杂化，复杂交互和状态同步中代码冗余等问题随之而来。

针对命令式 UI 的不足，目前声明式 UI 成了业界的主流解决方案趋势。声明式框架通过组合函数进行 UI 定义，仅需声明目标界面，框架自动处理底层更新，可有效提升代码的可读性、可维护性等，消除 UI 组件状态与 UI 的不一致隐患。目前，苹果公司的 SwiftUI、谷歌公司的 Jetpack Compose 等技术都采用了声明式开发，紧跟技术发展趋势[3]。华为公司推出的 Harmony 方舟开发框架（ArkUI），是一套构建声明式 UI 的开发框架。其使用简洁的 UI 信息语法、种类丰富的 UI 组件，有效提升了应用 UI 界面的开发效率[4,5]。

目前，声明式 UI 在 iOS 的 SwiftUI、Android 的 Jetpack Compose 研究比较广泛，作为后起之秀的 ArkUI，这一新框架的深入开发体验研究相对较少。本文通过研究 ArkUI 的声明式开发框架，通过一个手机新闻列表应用的开发实践，从开发效率、代码可读性、可维护性等角度与 Android 传统命令式开发进行对比，分析两者之间的优点和适用范围。

二、UI 框架技术

（一）HarmonyOS 应用架构

HarmonyOS 是华为研发的移动操作系统。其核心设计理念是通过统一的架构和语言，实现手机、平板、智能穿戴设备等多种终端设备的协同与资源共享[6]。同时，HarmonyOS 支持“一次开发、多端部署”，旨在在万物互联时代提供一个安全流畅的操作系统[7]。

HarmonyOS 技术的核心是方舟编程语言（ArkTS）、DevEco Studio 工具链以及声明式的方舟 UI 框架（ArkUI）[8]。

ArkTS 语言

它是 HarmonyOS 的开发语言，区别于 Android 使用的 Java 和 Kotlin。在 TypeScript 的基础上，深化代码静态检查与分析过程，实现程序性能的提升 [9,10]。

DevEco Studio

它是为鸿蒙应用开发设计的一站式集成开发环境，为开发者提供了丰富的工具和功能，为开发者的开发、编译、调试工作提供了便利。同时其安装过程较 Android Studio 更加方便快捷，简化了其配置过程，对国内用户友好。

ArkUI

它是一套构建应用界面的声明式 UI 开发框架。详细介绍见 2.2 节。

（二）ArkUI 核心框架

1. 命令式 UI 与声明式 UI

命令式 UI：需要描述界面从原始状态到目标状态的每一个操作步骤，通过一系列指令构建和更新界面。

声明式 UI：需要描述界面在任意状态下应呈现的 UI 画面，运行时自动根据状态变化完成界面更新。

2. 界面构建对比

命令式 UI：使用 Java 风格的面向对象的 API 方法，通过视图引用组件并调用其属性、方法。

```
Button button = findViewById(R.id.button);
button.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
           Text.setText("Clicked");
        }
      }
);
```

声明式 UI：使用函数语句来描述 UI 界面，将 UI 结构定义为状态的函数。

```
 @State message : string = "Hello";
build(){
    Button("Click here")
    .onClick(() => {
      this.message = "Clicked"
    })
    Text(`${this.message}`)
}
```

3. 状态对比

命令式：UI 状态存储在各个视图控件中，更新时需要通过手动方式同步各组件视图。

声明式：UI 状态进行集中管理，通过框架的差异算法自动将新状态进行 UI 变更。

4. 基于 ArkTS 的声明式开发

ArkUI 共有两种不同的开发范式—基于 ArkTS 的声明式开发和兼容 JS 的类 Web 开发，本文主要讨论前者。

基于 ArkTS 的声明式开发范式的本质是通过描述式语言实现 UI 与状态的单向数据流绑定 [11]。该模型将 UI 抽象为状态的纯函数映射（UI = f(state)），开发者仅需声明状态变量与界面结构的逻辑关系，系统就可以自动高效地完成 UI 页面的更新渲染 [12]。

在 ArkTS 中，通过装饰器（如 @State）来标记状态变量。一个变量只有在被装饰器标记之后才能成为“状态变量”，否则就只是一个简单的普通变量。被装饰器标记的状态变量的变化会自动触发 UI 界面的更新渲染，肩负起更新 UI 的责任。

```
 @State stateCount: number = 0
 normalCount: number = 0
 build() {
   Text(' 状态变量：'+`${this.stateCount}`)
   Text(' 普通变量：'+`${this.normalCount}`)
   Button(' 增加 ')
   .onClick(() => {
       this.stateCount++
       this.normalCount++
       console.log(' 状态变量当前值：', this.stateCount)
       console.log(' 普通变量当前值：', this.normalCount)
    })
 }
```

如上述代码所示，有两个变量，stateCount 和 normalCount。前者 stateCount 被装饰器 @State 修饰成为状态变量，后者 normalCount 则是一个普通变量。当点击“增加”按钮时，引用状态变量 stateCount 的 Text 组件的数字会立即更新，因为触发了组件重新渲染；引用普通变量 normalCount 的 Text 组件的数字不会发生变化，即使变量的值的确增加了。查看应用的运行日志，发现两个变量的数值在增加，从而证明状态变量可以驱动 UI 刷新，普通变量不可以。

现代应用开发逐渐向声明式开发范式过渡，因此本文聚焦于 ArkTS 的声明式 UI 进行研究，对比基于 ArkTS 的声明式开发与基于早期 Android 的命令式开发，分析两者之间的优点和适用范围。

三、新闻列表开发实践与开发体验评估

本章在上述基于 Android 的命令式 UI 和基于 ArkUI 的声明式 UI 的基础上，通过实践验证的方式，分别开发一个新闻 APP 的列表功能页面，然后采用对比分析法，对比两种不同的开发方

式在开发效率、代码可读性、可维护性方面的差异与特点。

（一）应用整体设计

当前，设计并开发一个新闻 APP 的列表功能页面，可实现新闻列表的上下滑动功能。页面整体布局为垂直线性布局，布局上方是一个固定位置的文本框，用于标记新闻列表页的标题。在主页标题下方是一个可滑动的 List 结构，里面显示新闻列表，如图1所示。列表的每一个 Item 是一个水平线性布局结构，左侧为新闻图片的展示框，右侧是一个垂直线性布局结构，其上侧是新闻标题，下侧是新闻摘要，如图2所示。

新闻列表中的新闻数据来自资源文件中解析的 Json 格式文件，模拟从网络上拉取的新闻数据。本文选取了2026年3月部分新闻条目作为数据来源，数据经过整理，抽取出新闻标题、摘要、图片资源。

```
[
  {
    "title": " 总书记的两会之问 落地有回响 ",
    "summary": " 全国两会上，习近平总书记与代表委员共商国是，频频互动。",
    "imageUrl": "app.media.news1"
  }
]
```

图1 新闻列表主页

图2 新闻 Item 子项

（二）基于 ArkUI 的新闻列表开发

基于 ArkUI 开发的新闻列表主页面，其展示效果如图1所示。主要包含以下文件：

1.News.etc 文件：新闻类的数据结构，包括新闻标题、摘要、图片资源等属性。

2.news.json 文件：新闻数据的 Json 文件，包括十条数据，用于模拟从网络上下载的时事新闻。

3.NewsListPage.etc 文件：新闻列表的主页面，实现新闻列表页的 UI 绘制、Json 数据解析、新闻列表项加载。

在 NewsListPage.etc 文件中，涉及四个方法和一个属性：

1.newList 数组：自定义新闻数组变量，存储新闻数据。

2.loadNewsFromRawFile()：从 rawfile 目录异步加载 json 文件，并解析数据。

3.aboutToAppear()：ArkTS 组件中的生命周期函数，此处用于预加载阶段的数据初始化。通过调用异步数据加载方法，为组件内部的状态变量获得初始值。

4.NewsItem()：用于创建单个新闻列表项组件。

5.build()：存放 UI 描述的代码。

涉及两个装饰器的使用：

1.@State 装饰器：用于修饰自定义 newsList 数组变量，保存状态数据，存储从 news.json 文件中解析的新闻数据，数据状态变化会触发所在组件的 UI 进行重新渲染。这里的 newsList 数组会在新闻列表中进行加载填充。

2.@Builder 装饰器：用于装饰自定义的新闻列表项 NewsItem() 构建函数，以便在新闻列表中复用列表项组件。

（三）基于 Android 的新闻列表开发

基于 Android 开发新闻列表主页面的功能实现，开发的展示效果图如图3所示。主要包含以下文件：

1.activity_news_list.xml 文件：新闻页的 Xml 布局文件。

2.item_news.xml 文件：新闻列表项 Item 的 Xml 布局文件。

3.news.json 文件：新闻数据的 Json 文件。

4.News.java 文件：新闻类的数据结构。

5.GsonParser.java 文件：Json 文件解析的工具类，用于解析 news.json 文件中的新闻信息。

6.NewsAdapter.java 文件：新闻列表项 Item 填充适配，加载 item_news 文件，并对表项中的新闻标题、摘要、图片进行填充，并且需要获得 ViewHolder 对刷新的新闻表项进行不断加载填充。

7.NewsListActivity.java 文件：主 Activity 页面，加载 activity_news_list 文件，使用 NewsAdapter 对新闻列表进行数据填充。

图3 Android 新闻列表主页

（四）开发体验测评与分析

对比上述两种功能相同的新闻列表开发，前者共涉及3个主要文件，未进行单独的前端页面的设计。在主文件中，直接进行 UI

的布局，并通过 @State 保存状态数据，通过 @Build 装饰自定义组件。其 UI 结构和组件之间的状态依赖比较简单且一目了然，对 APP 初学者友好。共涉及核心代码100余行。

后者共涉及7个主要文件，并且单独定义了主页面和列表项的 Xml 文件，分别由主 Activity 文件和新闻 Adapter 文件加载，依赖复杂。NewsAdapter 需单独设计，并在列表子项 item 中填充数据，开发逻辑与依赖较为复杂，其涉及的前端布局设计对 Web 前端开发者更加友好。共涉及核心代码140行，布局文件代码70行，代码开发效率与可维护性较低。

四、结语

本文以华为 Harmony 的 ArkUI 声明式开发框架为研究对象，与早期 Android 的命令式开发框架进行开发对比研究。分别用这两种方式各开发一个新闻列表 APP，从开发效率、代码可读性、可维护性等角度，对两种不同的开发效果进行对比。实验结果表明，基于 ArkUI 的声明式开发，其 UI 结构、状态依赖更加一目了然，开发效率更高，体现了 ArkUI 声明式 UI 在提升开发体验方面的价值；传统的命令式开发对 Web 前端开发者更加友好，体现了两种不同方法的适用范围。不过本研究仍有局限，例如评测主观性较强，App 复杂度相对较低。下一步工作，可在此基础上，对基于 ArkUI 的声明式开发在内存消耗、APP 打开速率方面进行更深层次的比较。

参考文献

[1] 王玉珍，窦晓艺 . 直播电商中互动性对消费者购买行为影响研究 —— 基于用户粘性中介作用的分析 [J]. 统计与管理 ,2025,40(06):63-68.DOI:10.16722/j.issn.1674-537x.2025.06.005.

[2] 徐梦霞，徐豪，郑韬 . 基于用户体验的 UI 设计与前后端功能实现的深度融合 [J]. 现代传输 ,2025,(06):50-54.

[3] 杨政 . 跨平台与声明式交互界面技术在企业移动应用中的探索应用 [J]. 电子世界 ,2021(7):13-14.

[4] 范承宇，李竞择，欧阳迪 . 基于方舟开发框架的智能装备监控应用研究 [J]. 机电产品开发与创新 ,2024,37(2):114-117. DOI:10.3969/j.issn.1002-6673.2024.02.031.

[5] 林钟雪 . 鸿蒙创新性应用开发：基于 HarmonyOS Next 与 ArkTS API12+ 的深度实践指南 [EB/OL].(2025-02-25)[2025-09-24].https://juejin.cn/post/7474976018568527881.

[6] 黄日胜，刘中柱 . 基于鸿蒙系统的智慧教室移动控制 App 的设计与实现 [J]. 安徽电子信息职业技术学院学报 ,2025,24(04):60-66.

[7] 陈赵云 .HarmonyOS 首选项数据存储研究 [J]. 电脑知识与技术 ,2026,22(01):68-70.DOI:10.14004/j.cnki.ckt.2026.0010.

[8] 张雅文 . 原生鸿蒙操作系统应用开发的关键技术与产业化路径研究 [J]. 信息系统工程 ,2025,(10):133-136.

[9] 薛云兰，邓远飞 . 基于 ArkTS 语言的面试助手鸿蒙应用开发 [J]. 互联网周刊 ,2025,(21):34-37.

[10] 李涛，徐威，赵彦，等 . 基于 Harmony OS NEXT 的智慧桃园系统设计 [J]. 常州信息职业技术学院学报 ,2024,23(6):36-42.

[11] 张正海 . 基于 ArkTS 的 HarmonyOS 原生应用开发研究 [J]. 信息与电脑 (理论版),2024,36(19):87-89.

[12] 基于 HarmonyOS 应用开发的课程建设初探 [J]. 龙军；何畅；赵东东 . 电脑知识与技术 ,2022(01).

