《嵌入式应用开发》

青蛙影院项目—影院列表实验指导手册

版本：V 1.0

目录

（一）实验目的

- 掌握青蛙影院基于 ArkTS 语言的开发。
- 掌握图片轮播功能的开发。
- 掌握列表组件的使用。
- 掌握如何在 ArkTS 语言中使用 MVVM 开发模式进行开发。

（二）实验涉及知识点

- 基础组件和布局。
- 装饰器的使用。
- 图片轮播 Swiper 组件的使用。
- List 组件的使用。
- MVVM 开发模式。

（三）实验准备

参考开发环境：

操作系统：Window 10

开发工具：DevEco Studio 3.1.1

HarmonyOS SDK 版本：API version 9 及以上版本

开发语言：ArkTS

内存：8G 及以上

（四）详细实验过程

影院页面的实现的效果图如下图所示。

整体页面分析，大概可以分为 2 个部分：顶部的广告轮播图，以及下面的影院列表。

1 影院广告的实现

我们通过 ViewModel 统一提供数据，在 MainViewModel.ets 中继续添加如下代码。

```
  //广告轮播图
  getSwiperAdvImages(): Array<Resource> {
    let advImages: Resource[] = [
    $r('app.media.adv1'),
    $r('app.media.adv2'),
    $r('app.media.adv3'),
    $r('app.media.adv4')
    ];

    return advImages;
  }
```

接下来，我们在影院页面自定义组件来实现广告的轮播图效果。打开 Cinema.ets 文件编写如下代码。

```
import viewmodel from '../../viewmodel/MainViewModel'

@Preview
@Component
export struct Cinema {
  build() {
    Scroll() {
      Column() {
        //顶部的轮播广告
        MySwiper()

      }
    }
  }
}

@Component
struct MySwiper {
  build() {
    Swiper() {
```

```
      ForEach(viewmodel.getSwiperAdvImages(), (item: Resource, index?: number)
=> {
        Image(item)
      }, item => item.toString())
    }.autoPlay(true)
  }
}
```

2 影院列表的实现

接下来我们对影院列表的数据进行分析，得出每一个影院大致所包含的信息如下：

- 影片名
- 分店名
- 地址
- 特色标签，【退，改签，小吃，杜比全景声，巨幕，4k 厅...]
- 最低价

我们将其封装成一个 bean，在 bean 包下新建一个 CinemaBean.ets 文件。核心代码如下。

```
//封装影院数据
export default class CinemaBean {
  id:number   //影院 id
  name:string //影片名
  branchName:string //分店名
  address:string  //地址
  tags:Array<string>  //特色标签，【退，改签，小吃，杜比全景声，巨幕，4k 厅...]
  minPrice:number //最低价

  constructor(id: number, name: string, branchName: string, address: string,
              tags:Array<string>,minPrice:number) {
```

```
    this.id = id;
    this.name = name
    this.branchName = branchName
    this.address = address
    this.tags = tags
    this.minPrice = minPrice
  }
}
```

然后，通过 ViewModel 统一提供数据（同样需要在 MainViewModel.ets 中导入 CinemaBean.ets），MainViewModel.ets 中定义的影院列表的数据方法如下。

```
//获取影院数据
getAllCinemas(): Array<CinemaBean> {
  let cinemas: CinemaBean[] = [
    new CinemaBean(1, '万象影城', '大朗店', '大朗镇复康路 229 号', ['退', '4k 影厅'], 29.90),
    new CinemaBean(2, '万达影城', '寮步店', '寮步镇寮步路 29 号', ['改签', '4k 影厅'], 9.90),
    new CinemaBean(3, '青蛙影城', '松山湖店', '松山湖新城路 29 号', ['改签', '巨幕厅'], 19.90),
    new CinemaBean(4, '飞鹤影城', '人民店', '人民公社北京路 229 号', ['退', '小吃', '4k 影厅'], 39.90),
    new CinemaBean(5, '万家喜影城', '东城店', '东城区东方路 129 号', ['小吃', '券包'], 29.90),
    new CinemaBean(6, '熊猫影城', '大学城店', '大学城宏光路 2 号', ['小吃', '4k 影厅'], 19.90),
    new CinemaBean(7, '万象影城', '万象汇店', '大朗镇万汇路 22 号', ['退', '爆米花', '8D'], 29.90),
    new CinemaBean(8, '万象影城', '万象城店', '常平镇万象城 29 号', ['可乐续杯', '4D'], 9.90),
    new CinemaBean(9, '大雄影城', '常平店', '常平镇复康路 22 号', ['改签', '4k 影厅'], 59.90),
    new CinemaBean(10, '飞鹤影城', '东坑店', '东坑镇复康路 29 号', ['退',
```

```
'3DMax'], 29.90),
    new CinemaBean(11, '飞鹤影城', '南城店', '南城复康路 9 号', ['退', '改签', '4k
影厅'], 29.90),
    new CinemaBean(12, '万象影城', '大朗店', '大朗镇复康路 129 号', ['改签', '影城
卡', '巨幕'], 19.90),
    new CinemaBean(13, '万达影城', '大朗店', '大朗镇复康路 129 号', ['退', '小吃
'], 9.90),
    new CinemaBean(14, 'HelloKit 影城', '人民店', '大朗镇人民路 29 号', ['小吃', '
巨幕'], 39.90),
    new CinemaBean(15, '红太阳影城', '松山湖店', '松山湖镇复康路 22 号', ['影城卡',
'改签', '小吃'], 29.90),
    new CinemaBean(16, '绿巨人影城', '大朗店', '大朗镇复康路 9 号', ['退', '4k 影厅
'], 19.90)

  ]

  return cinemas;
}
```

接下来，我们来实现影院列表，在 Cinema.ets 中继续编写如下代码。

```
@Component
struct MyCinemaList {
  build() {
    List() {
      ForEach(viewmodel.getAllCinemas(), (item: CinemaBean, index?: number) =>
 {
        ListItem() {

          Column() {
            Row() {
              //电影院名称
              Row() {
                Text(item.name)
                  .fontSize(20)
```

```
                Text(`(${item.branchName})`).fontSize(20)
              }
              //价格
              Text(`${item.minPrice}元起`).fontColor(Color.Red)
            }.width("100%").justifyContent(FlexAlign.SpaceBetween)
            // 影院地址
            Text(item.address).width("100%").fontColor(Color.Blue)
            // 影院标签
            Row({ space: 5 }) {
              ForEach(item.tags, (item2: string, index?: number) => {
                Text(`${item2}`).borderWidth(1).fontColor(Color.Gray)
              }, item2 => item2.toString)
            }.width("100%")
          }.padding(5)


        }
      }, item => item.toString())
    }.listDirection(Axis.Vertical)
    .divider({
      strokeWidth: 1,
      color: Color.Gray,
      startMargin: 5,
      endMargin: 5
    })
    .width("100%")
    .height("100%")
  }
}
```

最后，我们将我们的自定义组件添加到影院页面组件中去。

```
@Preview
@Component
export struct Cinema {
  build() {
```

```
    Scroll() {
      Column() {
        //顶部的轮播广告
        MySwiper()
        //电影院列表
        MyCinemaList()
      }
    }
  }
}
```

3 影院页面的完整代码

影院页面的完整代码如下。

```
import CinemaBean from '../../bean/CinemaBean'
import viewmodel from '../../viewmodel/MainViewModel'
@Preview
@Component
export struct Cinema {
  build() {
    Scroll() {
      Column() {
        //顶部的轮播广告
        MySwiper()
        //电影院列表
        MyCinemaList()
      }
    }
  }
}

@Component
struct MySwiper {
  build() {
```

```
    Swiper() {
      ForEach(viewmodel.getSwiperAdvImages(), (item: Resource, index?: number)
 => {
        Image(item)
      }, item => item.toString())
    }.autoPlay(true)
  }
}



@Component
struct MyCinemaList {
  build() {
    List() {
      ForEach(viewmodel.getAllCinemas(), (item: CinemaBean, index?: number) =>
 {
        ListItem() {

          Column() {
            Row() {
              //电影院名称
              Row() {
                Text(item.name)
                  .fontSize(20)
                Text(`(${item.branchName})`).fontSize(20)
              }
              //价格
              Text(`${item.minPrice}元起`).fontColor(Color.Red)
            }.width("100%").justifyContent(FlexAlign.SpaceBetween)
            // 影院地址
            Text(item.address).width("100%").fontColor(Color.Blue)
            // 影院标签
            Row({ space: 5 }) {
              ForEach(item.tags, (item2: string, index?: number) => {
                Text(`${item2}`).borderWidth(1).fontColor(Color.Gray)
```

```
              }, item2 => item2.toString)
            }.width("100%")
          }.padding(5)

        }
      }, item => item.toString())
    }.listDirection(Axis.Vertical)
    .divider({
      strokeWidth: 1,
      color: Color.Gray,
      startMargin: 5,
      endMargin: 5
    })
    .width("100%")
    .height("100%")
  }
}
```