ArkUI

XXX
XXX

*本次活动分享内容仅代表个人观点，不代表华为官方观点

ArkUI实战

*本次活动分享内容仅代表个人观点，不代表华为官方观点

ArkUI实战

- 课程目标
- 概念介绍
- 项目实操
- ArkUI总结

*本次活动分享内容仅代表个人观点，不代表华为官方观点

课程目标

1、学习和了解方舟UI框架（ ArkUI ） 、ArkTS的声明式开发范式

2、掌握ArkUI基础组件（ Text、Button等 ）、容器组件（ Column、Row、List、Grid等 ）

3、常用组件在HarmonyOS系统中的应用

4、学完课程后，能独立开发出一个简单的页面

ArkUI实战

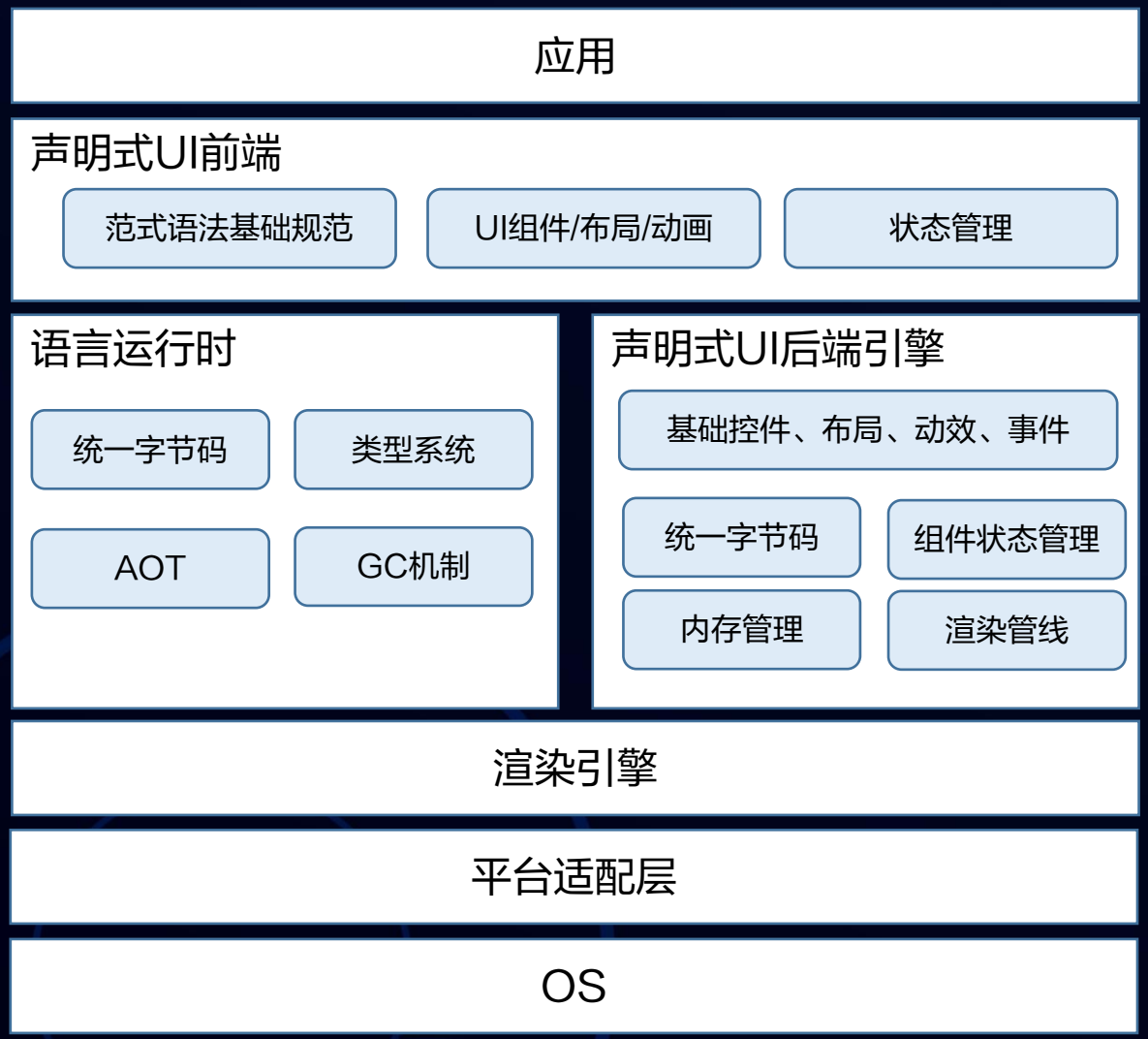
概念介绍

方舟UI框架（简称ArkUI）：为HarmonyOS应用的UI开发提供了完整的基础设施，包括简洁的UI语法、丰富的UI功能（组件、布局、动画以及交互事件），以及实时界面预览工具等，可以支持开发者进行可视化界面开发。

ArkTS：是应用开发语言，基于TypeScript（简称TS）语言扩展而来，扩展能力包含各种装饰器、自定义组件、UI描述机制。

声明式开发范式：基于ArkTS的声明式开发范式的方舟UI框架是一套开发极简、高性能、支持跨设备的UI开发框架，提供了构建HarmonyOS应用UI所必需的能力。

ArkUI框架介绍



- 声明式UI前端

提供了UI开发范式的基础语言规范，并提供内置的UI组件、布局和动画，提供了多种状态管理机制，为应用开发者提供一系列接口支持；

- 语言运行时

选用方舟语言运行时，提供了针对UI范式语法的解析能力、跨语言调用支持的能力和TS语言高性能运行环境。

- 声明式UI后端引擎

后端引擎提供了兼容不同开发范式的UI渲染管线，提供多种基础组件、布局计算、动效、交互事件，提供了状态管理和绘制能力。

- 渲染引擎

提供了高效的绘制能力，将渲染管线收集的渲染指令，绘制到屏幕的能力

- 平台适配层

提供了对系统的抽象接口，具备接入不同系统的能力，如系统渲染管线、生命周期调度等。

声明式UI语法

声明式UI关注的是UI描述的结果。命令式UI关注的是UI的过程

```
<!--mycomponent.html-->
<div class=“column”>
  <text class=“title”>
    Hello HarmonyOS
  </text>
</div>
```

Web界面定义

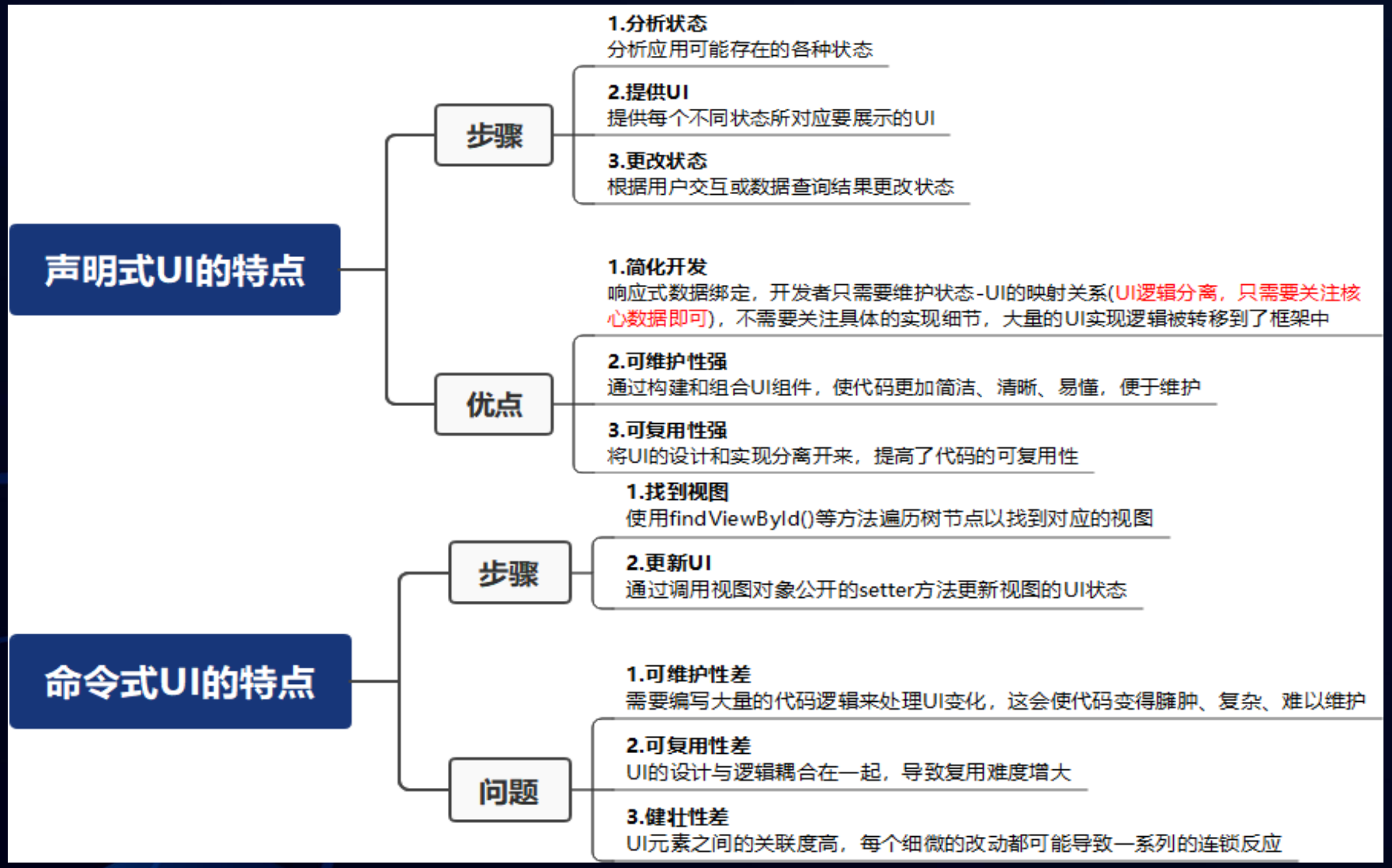
```
/* mycomponent.css */
.column {
  flex-direction: column;
  justify-content: center;
}
.title {
  font-size: 38px;
  color: "red";
}
```

命令式UI操作

```
// Imperative style
View b = findViewById(title)
b.setOnClickListener(() => {
  b.setColor(Color.RED)
});
```

数据驱动的声明式 UI

```
@State mText = ‘Hello HarmonyOS’;
...
Column() {
  Text(this.mText)
    .fontSize(this. mFontSize)
    .color(Color.red)
    .onClick(() => {
      this.mText = ‘Hello ArkUI’
    })
}
```



从命令式到声明式的转变

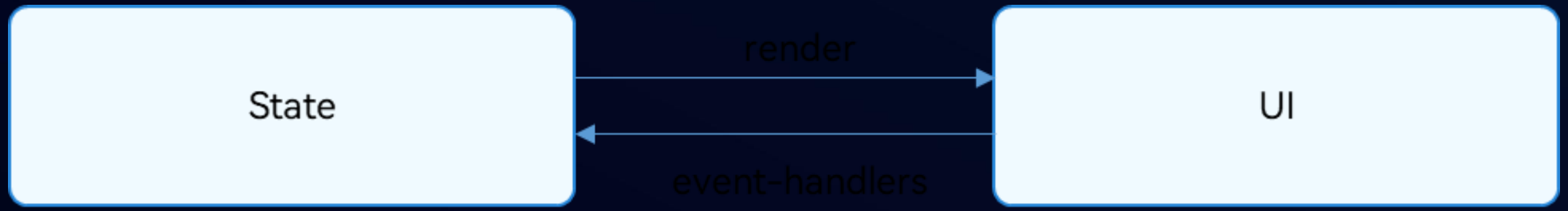
- 关注UI的抽象描述而非解决方案
- 强调数据和状态的定义
- 声明式的数据绑定
- 组件化和组合思想

```
@Entry
@Component
struct Page {
  @State title: string = 'world';

  build() {
    Row() {
      Column() {
        Text(`Hello ${this.title} `)
          .fontSize(50)
          .fontWeight(FontWeight.Bold)

        Divider()
        Button('click me')
          .onClick(() => {
            this.title = 'ArkUI'
          })
          .height(50)
          .width(100)
          .margin({ top: 20 })
      }
      .width('100%')
    }
    .height('100%')
  }}
```

数据驱动UI原理



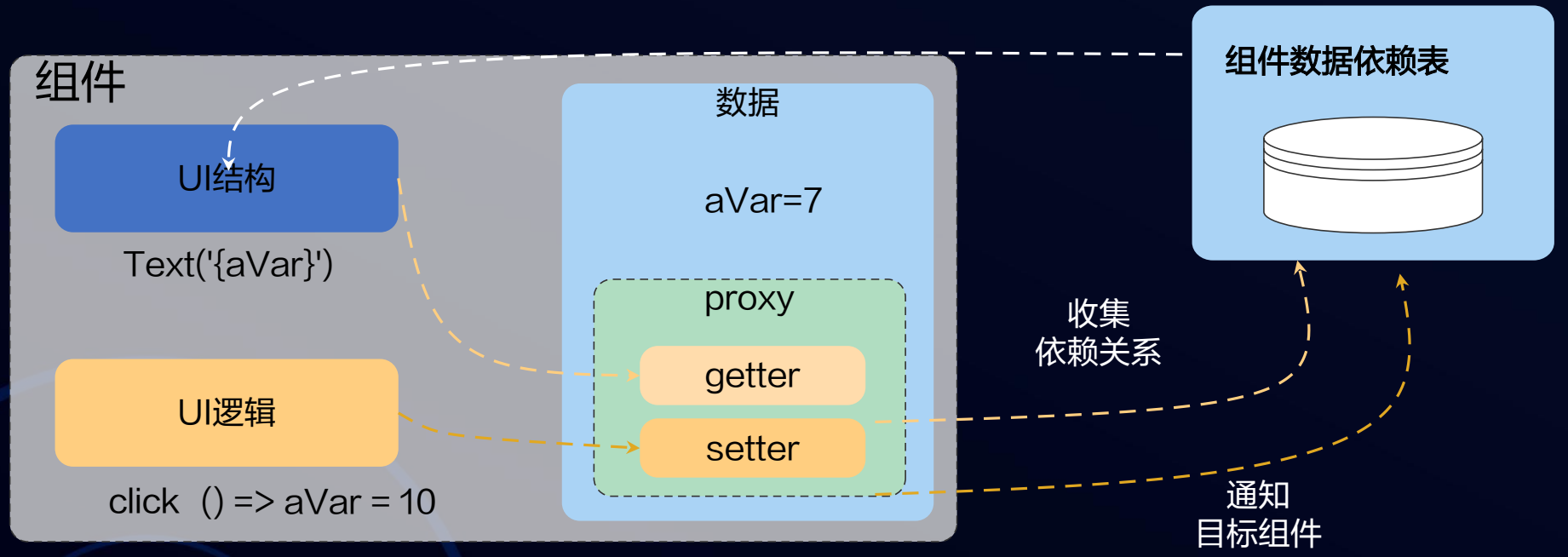
- View(UI)：UI渲染，一般指自定义组件的build方法和@Builder装饰的方法内的UI描述。
- State：状态，指驱动UI更新的数据。用户通过触发组件的事件方法，改变状态数据。状态数据的改变，引起UI的重新渲染

State

UI

数据驱动UI原理

在首次渲染的时候，执行build方法渲染系统组件，如果有自定义子组件，则创建自定义组件的实例。在执行build()函数的过程中，框架会观察每个状态变量的读取状态，拦截状态变量的get和set操作。



- **状态变量 -> UI组件**（包括ForEach和if）
- **UI组件 -> 此组件的更新函数**，即一个lambda方法，作为build()函数的子集，创建对应的UI组件并执行其属性方法

重新渲染

- 框架观察到了变化，将启动重新渲染
- 根据框架持有的两个map，框架可以知道该状态变量管理了哪些UI组件，以及这些UI组件对应的更新函数。执行这些UI组件的更新函数，实现最小化更新

ArkUI实战

HMOS世界应用展示

*本次活动分享内容仅代表个人观点，不代表华为官方观点

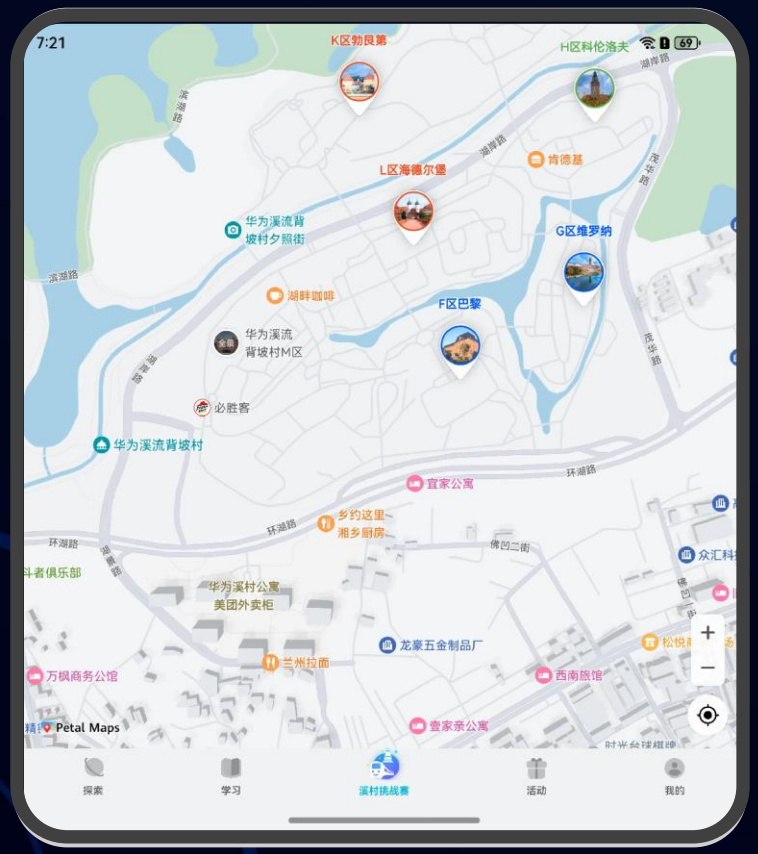
探索页签

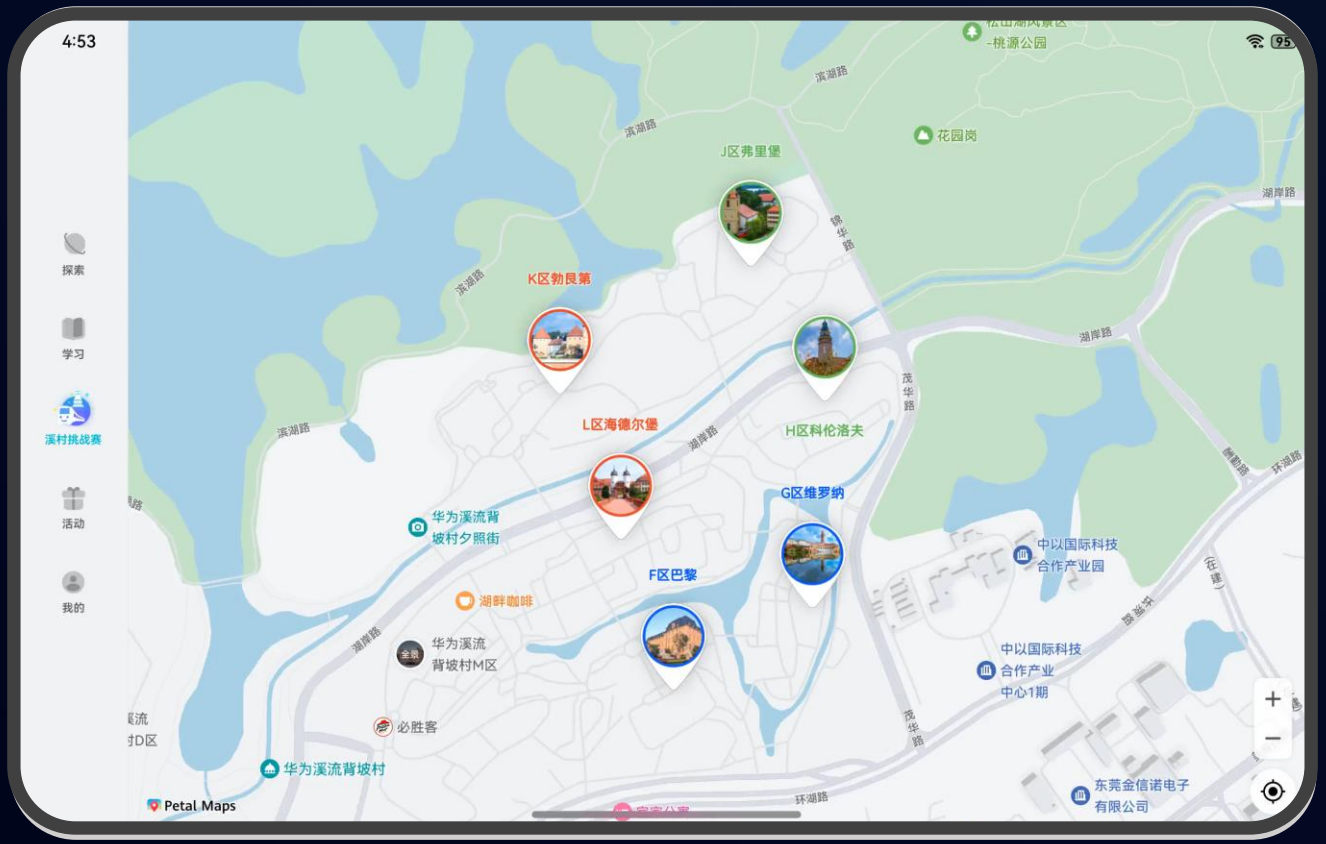
*本次活动分享内容仅代表个人观点，不代表华为官方观点

课程页签

*本次活动分享内容仅代表个人观点，不代表华为官方观点

溪村挑战赛页签–导航图



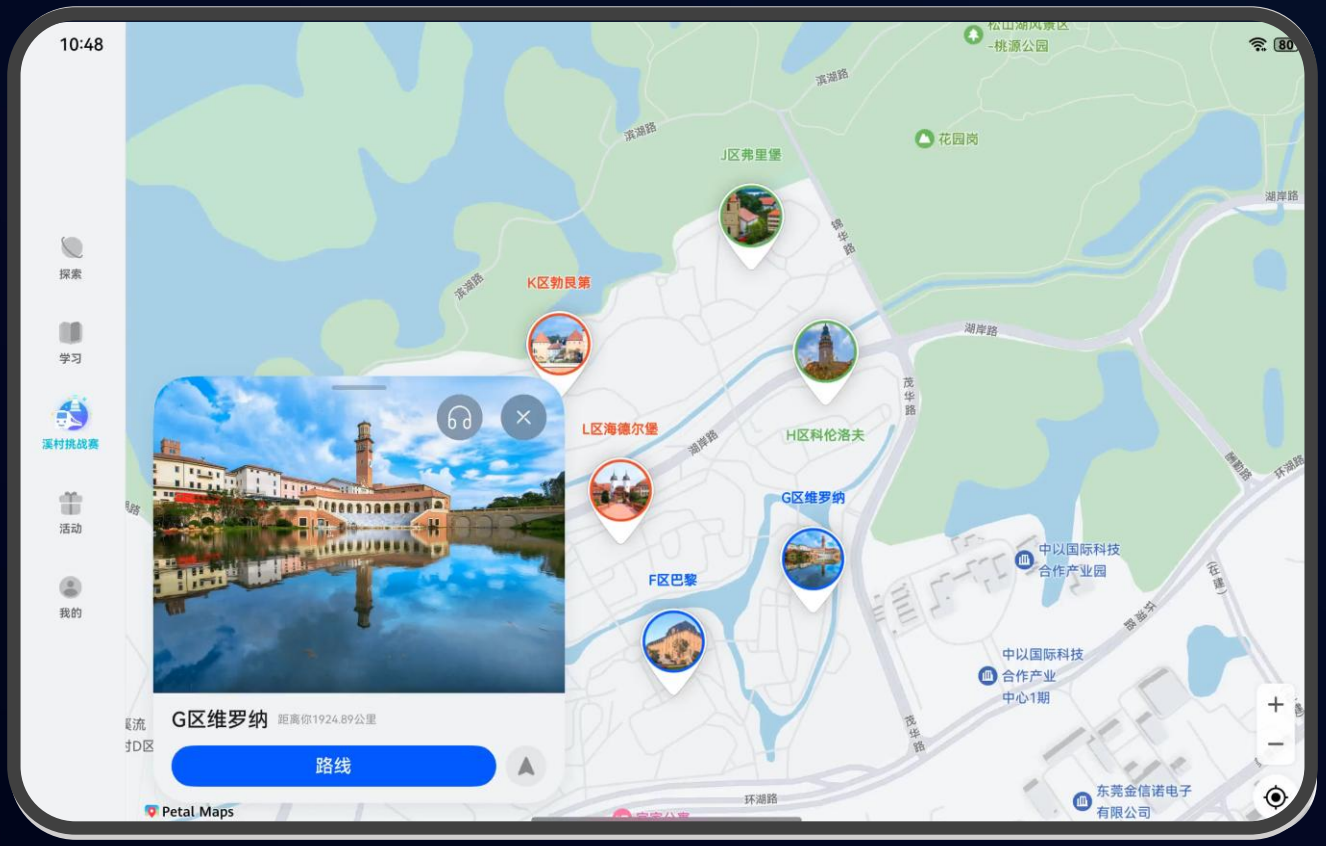


*本次活动分享内容仅代表个人观点，不代表华为官方观点

溪村挑战赛页签–AI朗读

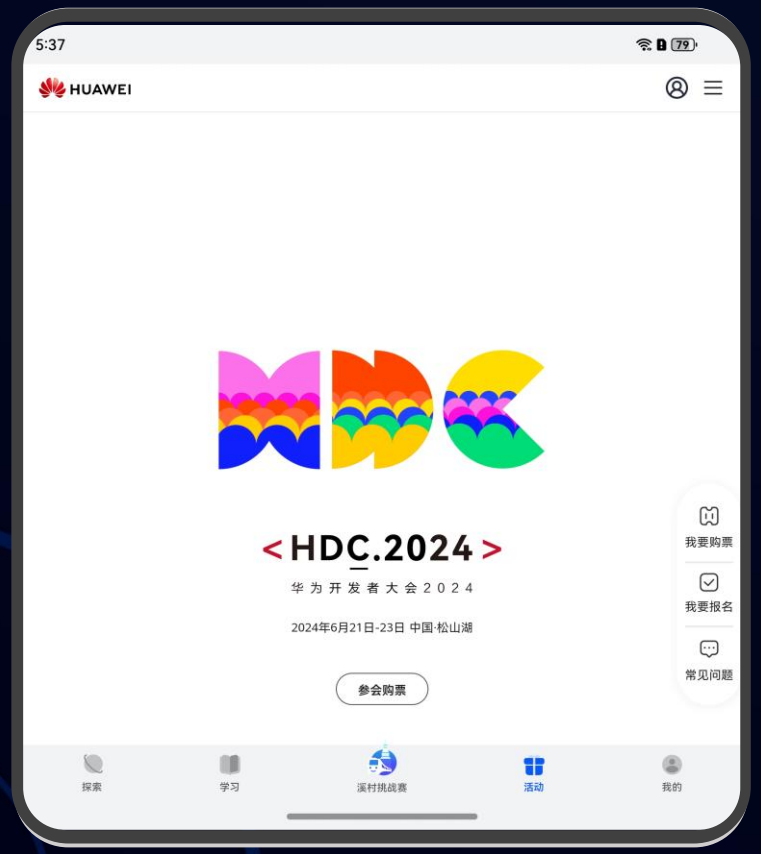
*本次活动分享内容仅代表个人观点，不代表华为官方观点

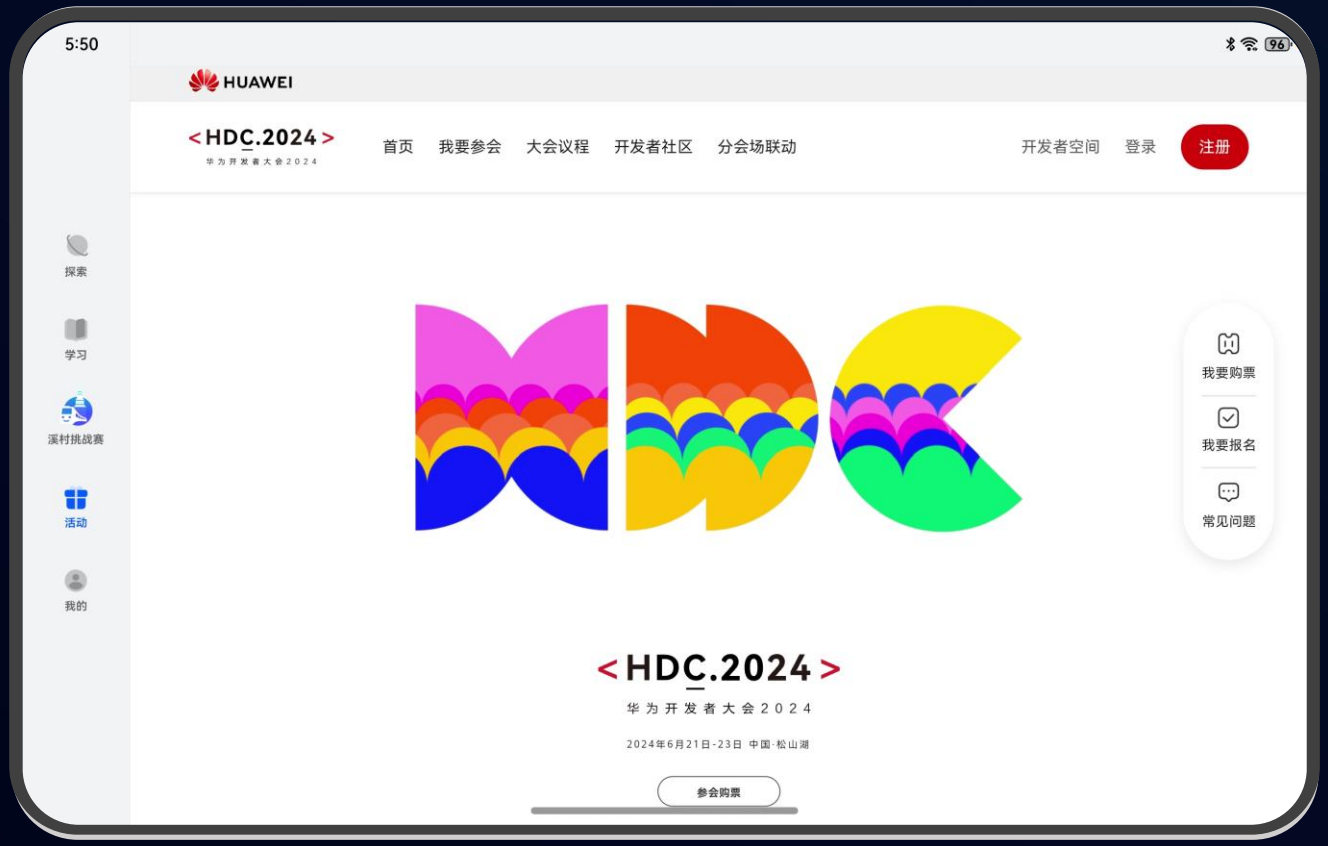
溪村挑战赛页签–主体抠图



*本次活动分享内容仅代表个人观点，不代表华为官方观点

活动页签





*本次活动分享内容仅代表个人观点，不代表华为官方观点

我的页签–深色模式

*本次活动分享内容仅代表个人观点，不代表华为官方观点

ArkUI实战开发

登录页

鸿蒙世界首页

热门资讯

*本次活动分享内容仅代表个人观点，不代表华为官方观点

目录

组件概览

基础组件	容器组件	媒体组件	绘制组件	画布组件
• Image • Text • TextInput • Button • …	• Column • Row • Flex • List • …	• Video	• Circle • Ellipse • Line • …	• Canvas

属性和事件

```
@Component
struct ListItemComponent {
  @State isChange: boolean = false;

  build() {
    Row() {
      ...
      Text(this.vote)
        .width(ItemStyle.LAYOUT_WEIGHT_RIGHT)
        .fontSize(FontSize.SMALL)
      ...
    }
    .height(ItemStyle.BAR_HEIGHT)
    .width(WEIGHT)
    .onClick(() => {
      this.isChange = !this.isChange;
    })
  }
}
```

属性方法：设置组件的属性。

事件方法：设置组件对事件的响应逻辑。

属性和事件

CommonMethod 中定义了ArkUI组件的**通用属性方法和通用事件方法**

```
declare class CommonMethod<T> {
  width(value: Length): T;
  height(value: Length): T;
  ...
  onClick(event: (event?: ClickEvent) => void): T;
  onTouch(event: (event?: TouchEvent) => void): T;
  ...
}
```

1、设置组件的样式
例如设置组件的宽、高

2、给组件添加事件的回调
例如监听点击事件、触摸事件

通用属性方法用于组件基本属性的配置，通用事件方法用于添加组件对通用事件的响应逻辑

属性和事件

```
interface TextInterface {
    (content?: string | Resource): TextAttribute;
}

declare class TextAttribute extends CommonMethod<TextAttribute> {
    fontColor(value: ResourceColor): TextAttribute;

    fontSize(value: number | string | Resource): TextAttribute;
    ...
}

declare const TextInstance: TextAttribute;
declare const Text: TextInterface;
```

1、继承了CommonMethod
支持通用属性和事件
例如点击文本事件onClick

2、支持该组件特有的属性和事件
例如文本颜色、字体大小

ArkUI组件支持该组件特有的属性和事件方法，其中大多数支持通用属性和通用事件。

目录

- 组件概览
- 从简单的页面开始
  - Image、Text、TextInput、Button
  - Column、Row
- 构建更加丰富的页面

*本次活动分享内容仅代表个人观点，不代表华为官方观点

案例介绍：常用组件与布局

案例：常用组件与布局

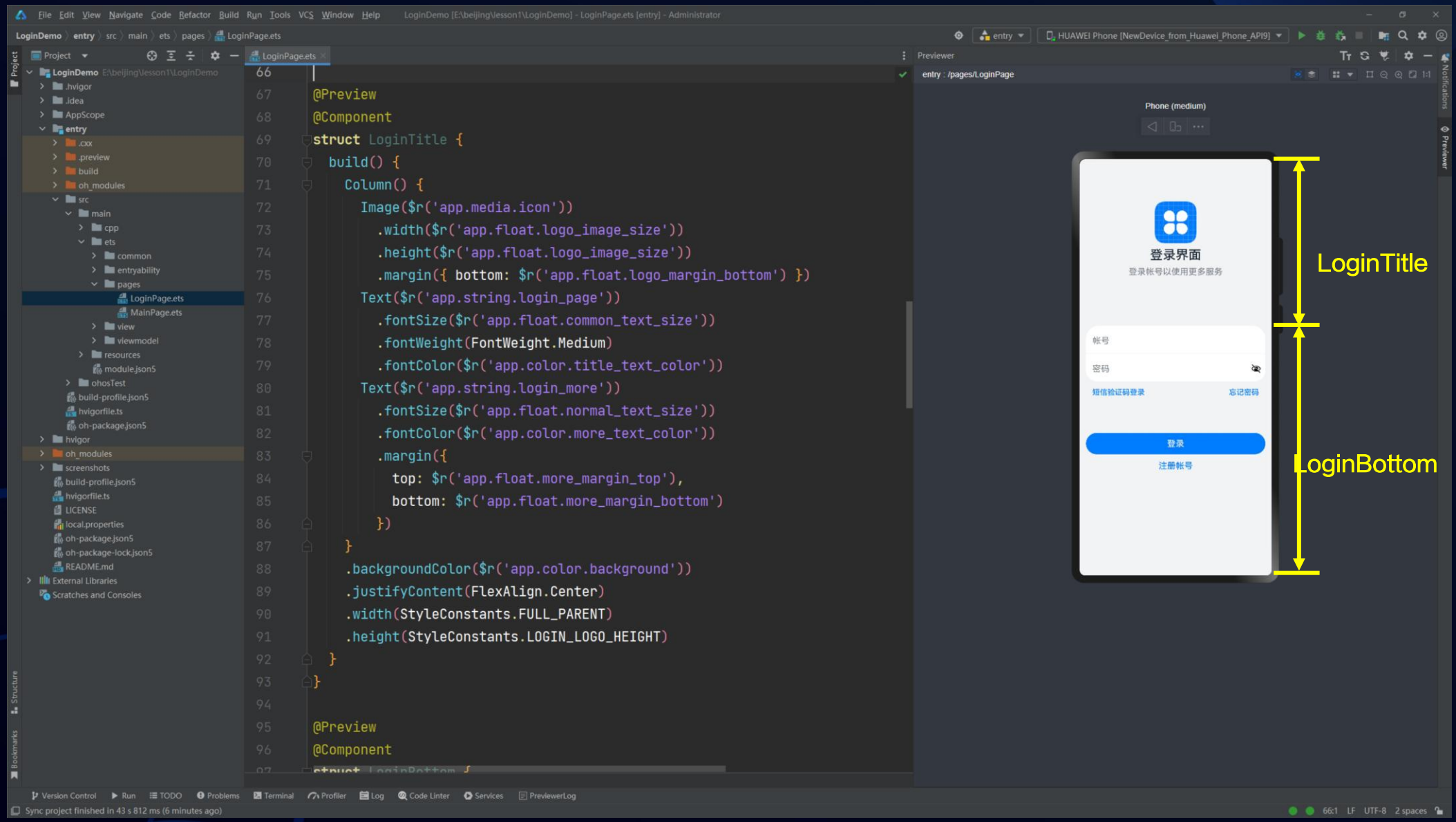
介绍 源码下载

介绍

HarmonyOS ArkUI提供了丰富多样的UI组件，您可以使用这些组件轻松地编写出更加丰富、漂亮的界面。在本篇Codelab中，您将通过一个简单的购物社交应用示例，学习如何使用常用的基础组件和容器组件。本示例主要包含：“登录”、“首页”、“我的”三个页面。

相关概念

- Text：显示一段文本的组件。
- Image：图片组件，支持本地图片和网络图片的渲染展示。
- TextInput：可以输入单行文本并支持响应输入事件的组件。
- Button：按钮组件，可快速创建不同样式的按钮。
- LoadingProgress：用于显示加载动效的组件。
- Flex：应用弹性方式布局子组件的容器组件。
- Column：沿垂直方向布局的容器。
- Row：沿水平方向布局容器。
- List：列表包含一系列相同宽度的列表项。适合连续、多行呈现同类数据，例如图片和文本。
- Swiper：滑动容器，提供切换子组件显示的能力。
- Grid：网格容器，由“行”和“列”分割的单元格所组成，通过指定“项目”所在的单元格做出各种各样的布局。



*本次活动分享内容仅代表个人观点，不代表华为官方观点

Image

渲染展示图片

Text

展示一段文本信息

TextInput

输入单行文本，响应输入事件

Button

响应用户点击操作

*本次活动分享内容仅代表个人观点，不代表华为官方观点

基础组件的使用

Image(src: string | PixelMap | Resource)

入参：src是图片数据源

用来渲染展示图片，支持加载本地和网络图片

```
export struct LoginPage {

  build() {
    Column() {
      ...
      Image($r('app.media.logo'))
      .width($r('app.float.logo_image_size'))
      .height($r('app.float.logo_image_size'))
      ...
    }
  }
}
```

在float.json文件定义图片宽高

```
{
  "float": [
    {
      "name": "logo_image_size",
      "value": "78vp"
    }
  ]
}
```

1、设置图片数据源

2、设置图片大小

基础组件的使用

```
Text($r('app.string.login_page'))
  .fontSize($r('app.float.page_title_text_size'))
  .fontWeight(FontWeight.Medium)
  .fontColor($r('app.color.title_text_color'))
```

```
TextInput({ placeholder: $r（‘app.string.account’）})
  .maxLength(CommonConstants.INPUT_ACCOUNT_LENGTH)
  .type(InputType.Number)
  ...
  .onChange((value: string) => {
    ... // 记录输入内容
  })
```

设置onChange事件

```
Button($r('app.string.login'), { type: ButtonType.Capsule })
  ...
  .onClick(() => {
    // 登录逻辑
  })
```

设置按钮点击事件

Column容器

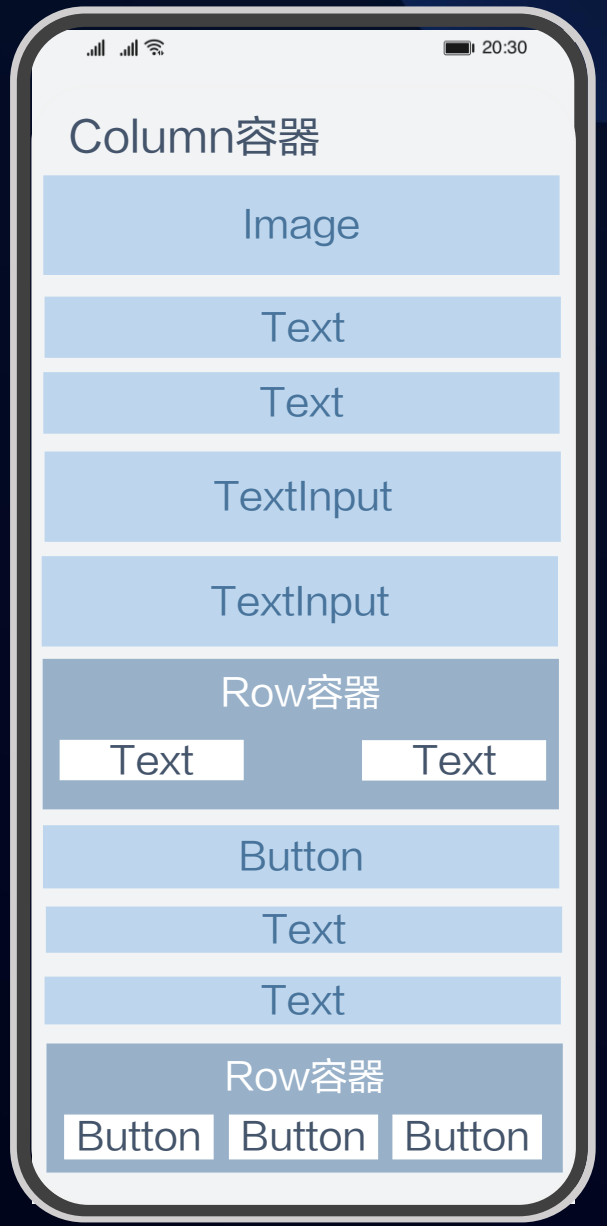
整个页面沿垂直方向布局

Row容器

文本组件沿水平方向布局

Row容器

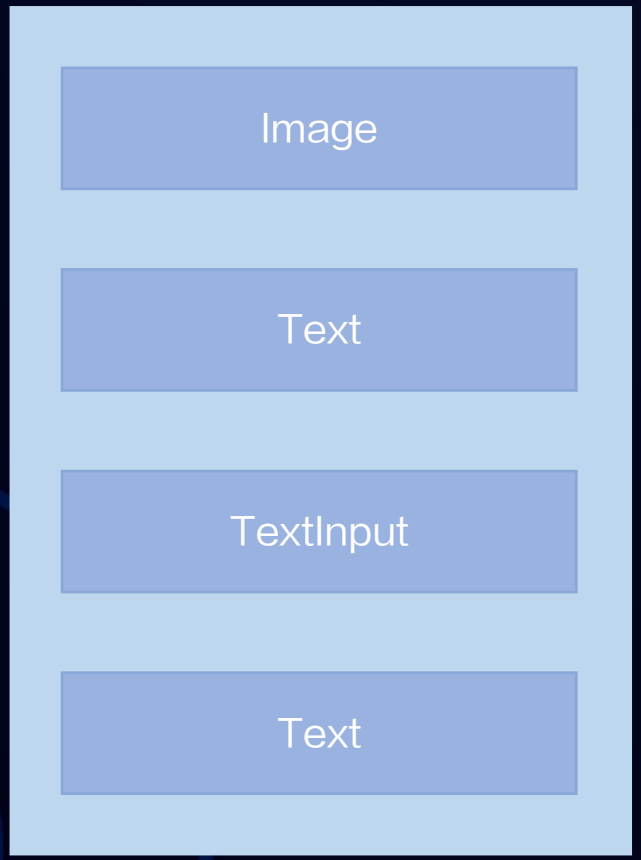
按钮组件沿水平方向布局



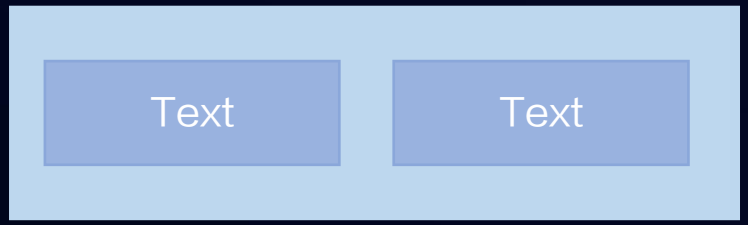
*本次活动分享内容仅代表个人观点，不代表华为官方观点

Column、Row容器

Column容器



Row容器



Column、Row的使用

```
build() {
  Column() {
    Image(...)
    Text(...)
    TextInput(...)
    Row() {
      Text(...)
      Text(...)
    }
    .justifyContent(FlexAlign.SpaceBetween)
    .width('100%')
    Button(...)
    ...
    Row({ space: CommonConstants.LOGIN_METHODS_SPACE }) {
      this.imageButton($r(app.media.login_method1))
      this.imageButton($r(app.media.login_method2))
      this.imageButton($r(app.media.login_method3))
    }
  }
}
```

*本次活动分享内容仅代表个人观点，不代表华为官方观点

- 组件概览
- 从简单的页面开始
- 构建更加丰富的页面
  - List
  - Swiper、Grid、Tabs

*本次活动分享内容仅代表个人观点，不代表华为官方观点

Scroll容器
通过滚动容器实现页面滚动视图

Swiper容器
通过滑动容器实现轮播图效果

Grid容器
使用网格容器进行布局

List容器
使用列表容器进行布局

*本次活动分享内容仅代表个人观点，不代表华为官方观点

使用List容器构建列表（1）：数据初始化

```
export default class ItemData {
  id: string;
  img ?: string;
  title ?: string;
  ...
  constructor(id: string, img ?: string, title?: string, ...)
}
```

列表项数据结构

```
export class MainViewModel {
  ...
  getArticleListData(): Array<ItemData> {
    let articleListData: ItemData[] = [];
    return articleListData;
  }
}
```

列表数据初始化

初始化具体列表内容

*本次活动分享内容仅代表个人观点，不代表华为官方观点

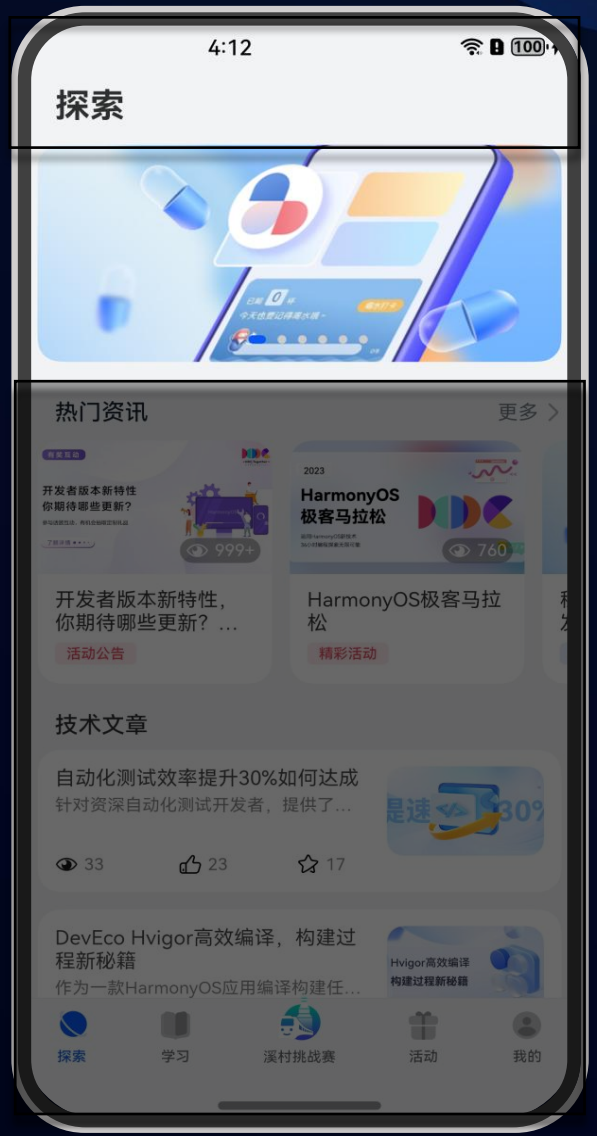
使用List容器构建列表（2）：构建列表

```
@Component
export default struct ArticleList {
  @Builder articleCardView(item: ItemData) { ... }
  ...
    List() {
      LazyForEach(mainViewModel.getArticleListData(), (item: ItemData) => {
        ListItem() {
          this.articleCardView(item)
        }
      }, item => item.toString())
    }
    .divider({ ... })
  ...
}
```

*本次活动分享内容仅代表个人观点，不代表华为官方观点

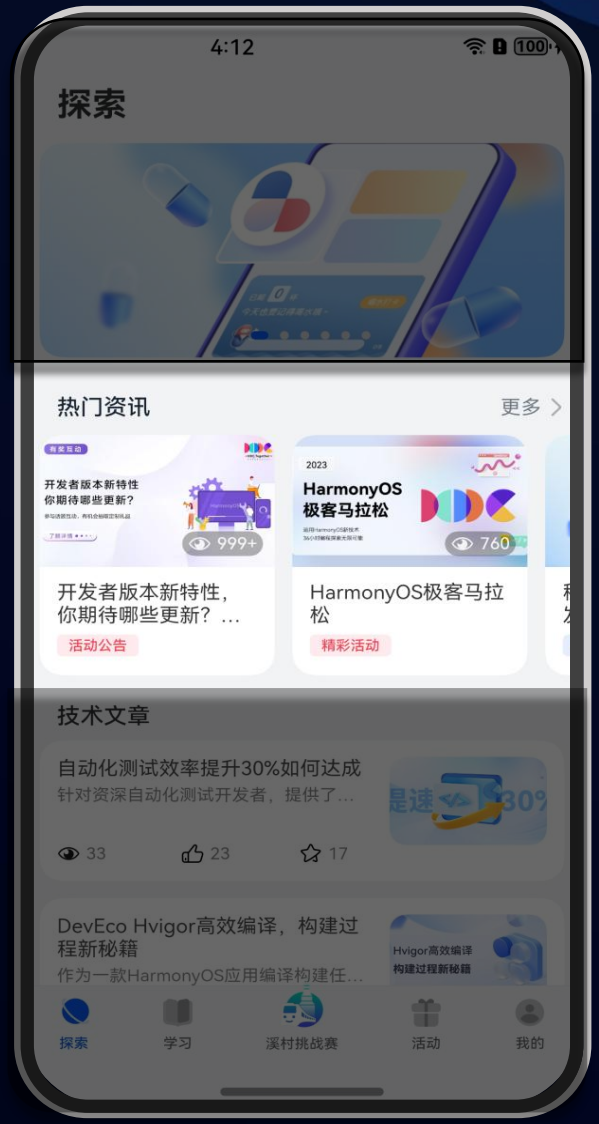
使用Swiper构建轮播图

```
@Component
export default struct Home {
  private swiperController: SwiperController = new
SwiperController();
  build() {
    Scroll() {
      Column({ space:
CommonConstants.COMMON_SPACE }) {
        Swiper(this.swiperController) {
          ForEach(this.data, (item: string) => {
            ...
          }, (item: string) => item)
        }
        .indicator(…)     // 设置圆点导航点样式
        .displayArrow({…})   // 设置导航点箭头样式
        .autoPlay(true)  // 自动播放轮播图
        ...
      }
    }
  }
}
```



使用Grid容器创建网格

```
@Component
export default struct Home {
  build() {
    Scroll() {
      Column({ space: CommonConstants.COMMON_SPACE }) {
        ...
        Grid() {
          ForEach(mainViewModel.getHotGridData(), (item: ItemData) => {
            GridItem()...
          }, item => item.toString())
        }
        .columnsTemplate('1fr 1fr 1fr 1fr 1fr 1fr’)
        .rowsTemplate(‘1fr’)
        .columnsGap($r('app.float.home_grid_columnsGap’))
        .rowsGap($r('app.float.home_grid_rowGap’))
        ...
      } } } }
```



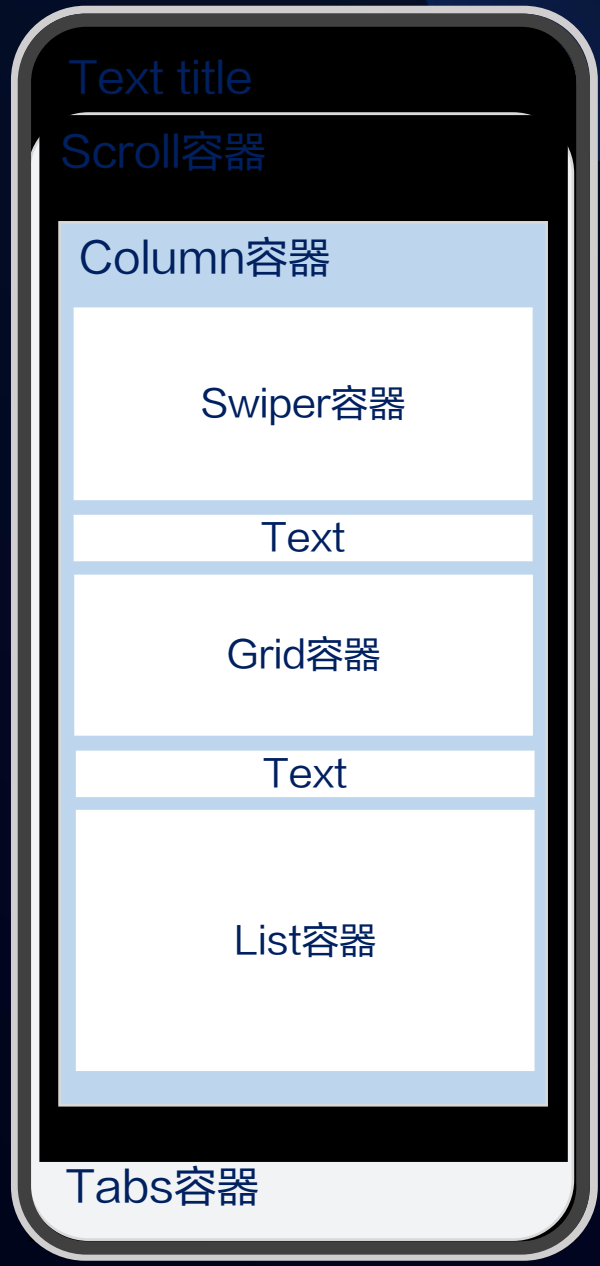
使用WaterFlow容器创建瀑布流

```
@Component
export default struct Home {
  build() {
    Scroll() {
      Column({ space: CommonConstants.COMMON_SPACE }) {
        WaterFlow({ footer: (): void => this.itemFoot() }) {
          LazyForEach(this.datasource, (item: LearningResource) => {
            FlowItem() {
              FlowItemComponent({ item: item })
            }
          }, (item: LearningResource) => JSON.stringify(item))
        }
        .layoutWeight(1)
        .layoutDirection(FlexDirection.Column)
        .columnsTemplate('1fr 1fr')
        .columnsGap('12vp')
        .rowsGap('12vp')
      } } } }
```

*本次活动分享内容仅代表个人观点，不代表华为官方观点

Tabs容器

通过页签进行内容视图切换，每个页签通过TabContent对应一个内容视图



*本次活动分享内容仅代表个人观点，不代表华为官方观点

使用Tabs构建页签导航

```
Tabs({
  barPosition: BarPosition.End,
  controller: this.tabsController,
}) {
  TabContent() {      // 探索
    this.tabTitles(...)
    Discover()
  }
  .tabBar(…)
  …
  TabContent() {      // 我的
    this.tabTitles(...)
    Mine()
  }
  .tabBar(…)
}
.barHeight(…)
.barMode(BarMode.Fixed)
.onChange((index: number) => {
  this.index = index;
})
```

*本次活动分享内容仅代表个人观点，不代表华为官方观点

Codelab: 常用组件与布局

01 Column

沿垂直方向布局的内容使用Column容器

02 Row

沿水平方向布局的内容使用Row容器

03 List

常用设置页面使用List容器布局

04 Swiper

首页通过swiper容器完成一组图片轮播效果

05 Grid

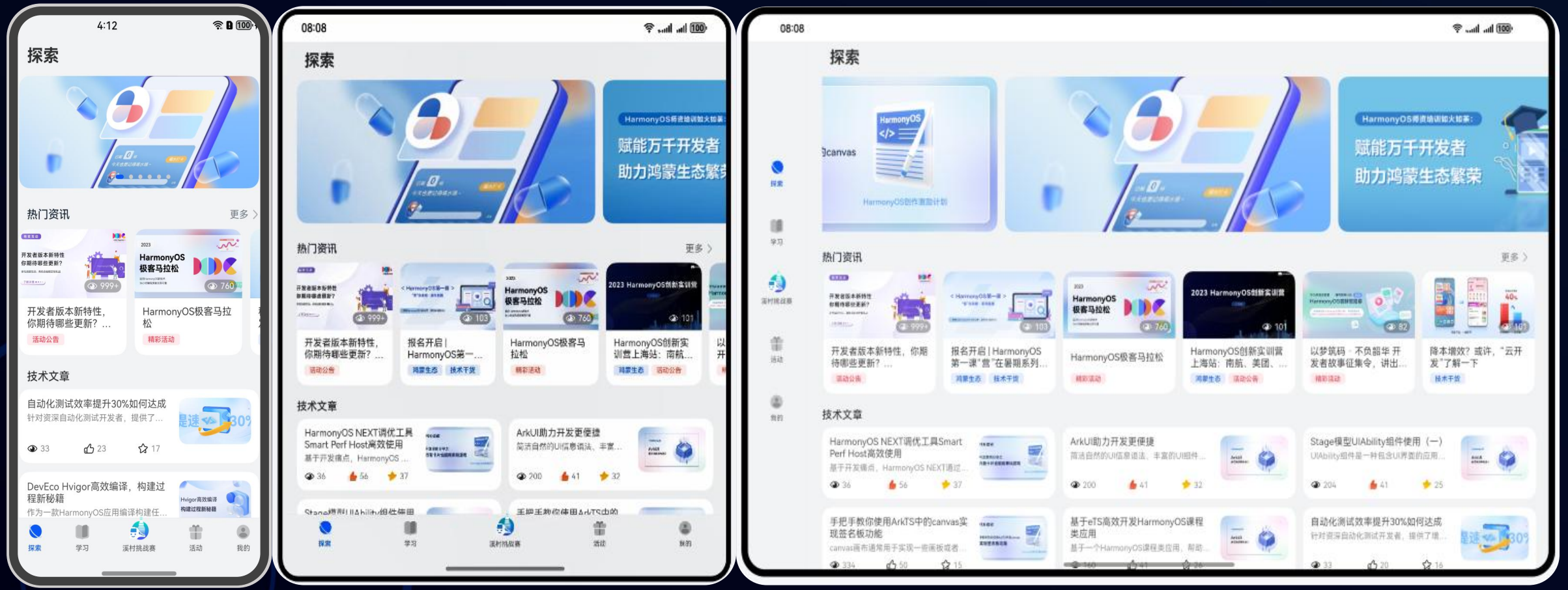
首页网格菜单使用Grid容器进行布局

06 Tabs

首页使用Tabs容器进行页面切换

*本次活动分享内容仅代表个人观点，不代表华为官方观点

一多适配示意



*本次活动分享内容仅代表个人观点，不代表华为官方观点

ArkUI实战

- 课程目标
- 概念介绍
- 项目实操
- ArkUI总结

*本次活动分享内容仅代表个人观点，不代表华为官方观点

使用容器组件构建布局

线性布局（Row/Column）

层叠布局（Stack）

弹性布局（Flex）

相对布局（RelativeContainer）

栅格布局（GridRow/GridCol）

创建列表（List）

创建网格（Grid/GridItem）

创建轮播（Swiper）

使用组件绘制UI

按钮（Button）

单选框（Radio）

切换按钮（Toggle）

进度条（Progress）

文本显示（Text/Span）

文本输入（TextInput/TextArea）

显示图片（Image）

视频播放（Video）

相关内容

设置页面路由和组件导航

显示图形

使用动画

支持交互事件

ArkUI总结

1、方舟UI框架（ArkUI）、ArkTS的声明式开发范式

2、Button、Image、TextInput、Text等基础组件的使用

3、Column、Row、List、Tabs等容器的使用

谢 谢

*本次活动分享内容仅代表个人观点，不代表华为官方观点