移动应用开发（四）

张凯斌

福建农林大学
资源与环境学院

2025.3.12

内容回顾

- 结束了编程基础介绍和第一次上机实验
  - 理论共识
    - 计算机系统层次结构
      - 重点介绍了计算机体系结构和操作系统
  - 编程工具共识
    - markdown+vscode及插件/deveco studio+github
  - 编写代码的一手经验
    - 页面跳转实验

内容展望

- 接下来将进入真正意义上的开发编程教学
  - 介绍实例代码
    - 书本上的部分代码片段演示
  - 讲解具体函数
    - 使用API参考文档进行功能扩展
  - 实现软件功能
    - 最终自行完成一个综合实战

内容安排

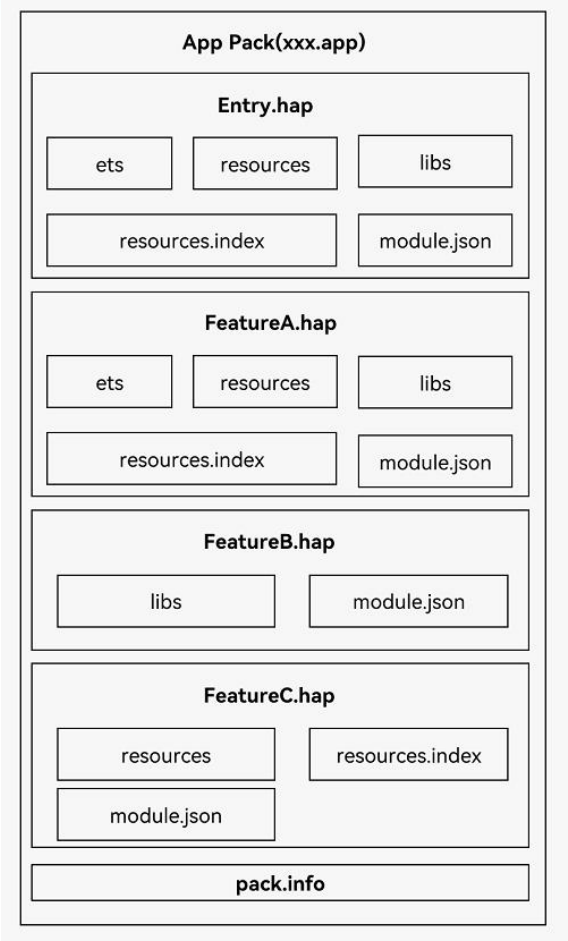
- 具体课程安排
  - 第4周（本周）周三
    - 页面信息传递
    - UI初步
  - 第4周（本周）周五
    - UI完结
  - 第5周周三
    - 课本第9章
  - 第5周周五
    - 课本第7章
  - 第6、7周周三
    - 移动GIS初步（为课程设计做准备）

本次课主要内容

- 四图
  - 以四张图整体上把握App的结构关系和开发要点
- 四函数
  - 以四个函数为例，演示**使用**API和**查询**API的方法
    - 授之以渔
- Want信息传递
- UI基础组件演示

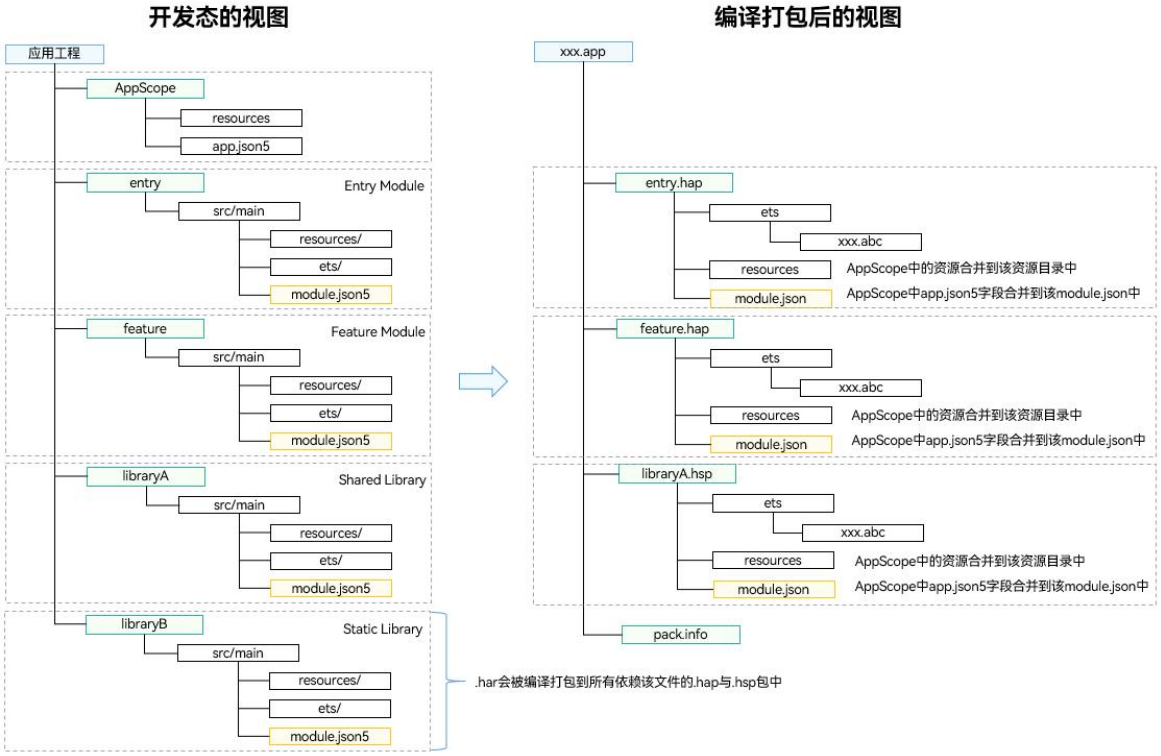
四张图（1/6）

- 应用程序包结构图
  - 一个App可以有多个HAP (Harmony Ability Package)
    - Entry是应用入口模块
    - Feature是应用扩展模块



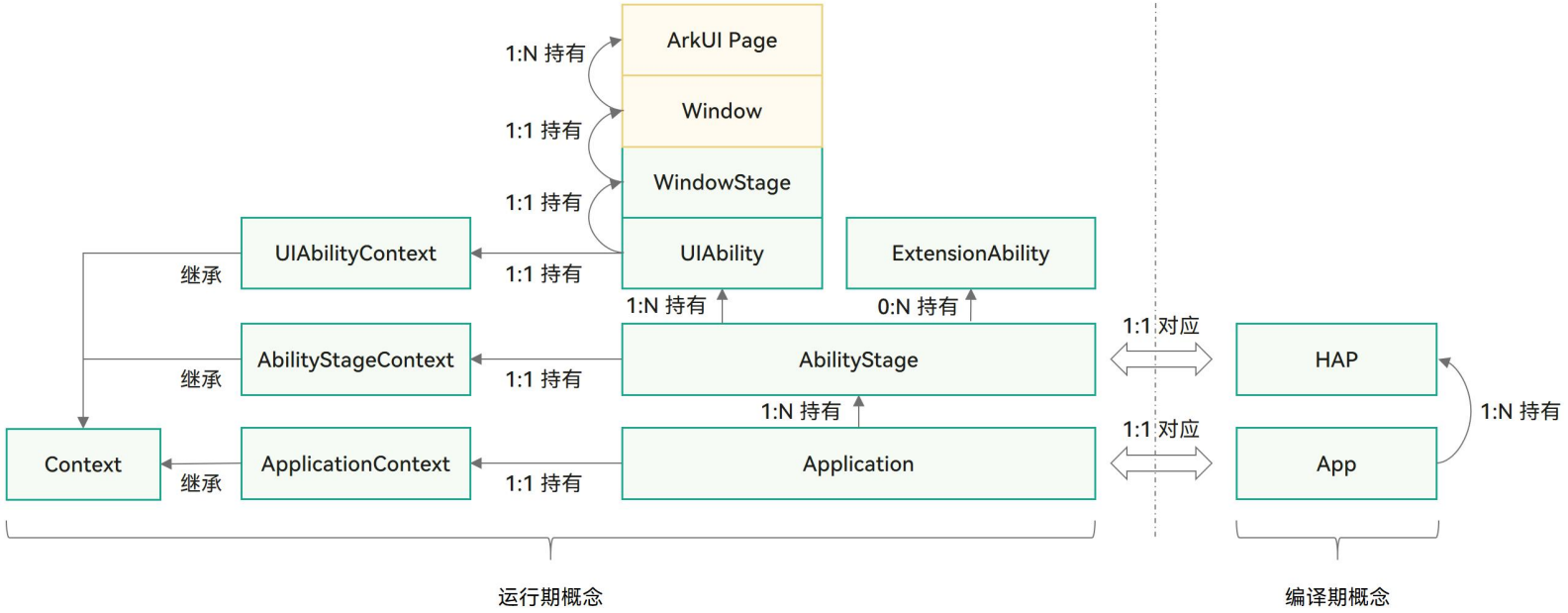
四张图（2/6）

- 应用程序目录结构图
  - HSP (Harmony Shared Package) 是动态共享包
  - HAR (Harmony Archive) 是静态共享包



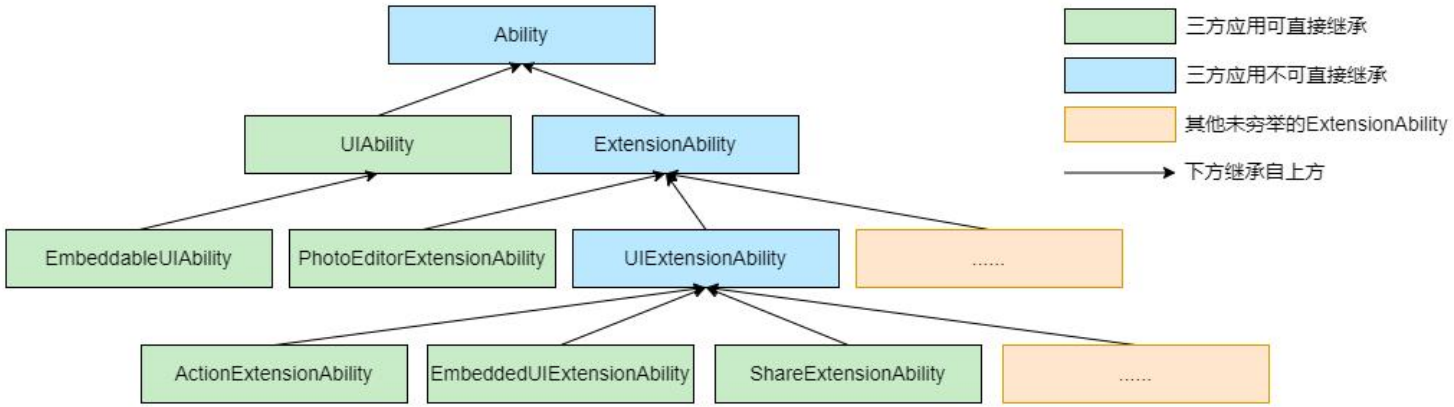
四张图（3/6）

- Stage模型概念图
  - 此图非常重要，概括了整个Stage应用模型的整体架构



四张图（4/6）

- Ability继承关系图
  - 此图非常重要，概括了Ability类的继承关系



四张图（5/6）

- Stage模型组件
  - AbilityStage组件
  - 每个Entry类型或者Feature类型的HAP在运行期都有一个AbilityStage类实例
  - UIAbility组件和ExtensionAbility组件
    - UIAbility组件是一种包含UI的应用组件，主要用于和用户交互
    - ExtensionAbility组件是一种面向特定场景的应用组件
      - 开发者并不直接从ExtensionAbility组件派生，而是需要使用ExtensionAbility组件的派生类

四张图（6/6）

- Stage模型组件（contd.）
  - WindowStage
    - 每个UIAbility实例都会与一个WindowStage类实例绑定
      - 该类起到了应用进程内窗口管理器的作用
    - WindowStage持有一个主窗口（Window）
      - 该主窗口为ArkUI提供了绘制区域（Pages）
  - Context
    - Context及其派生类向开发者提供在运行期可以调用的各种资源和能力
    - UIAbility组件和各种ExtensionAbility组件的派生类都有各自不同的Context类
      - 都继承自基类Context，根据所属组件，提供不同的能力

四个函数（1/2）

- 四个函数

```
pushUrl(options: RouterOptions): Promise<void>

pushUrl(options: RouterOptions, callback: AsyncCallback<void>): void

pushUrl(options: RouterOptions, mode: RouterMode): Promise<void>

pushUrl(options: RouterOptions, mode: RouterMode, callback: AsyncCallback<void>): void
```

  - 它们是同一个函数吗？
  - 该如何使用它们？

- 集中注意力听讲，此处**非常重要**

四个函数（2/2）

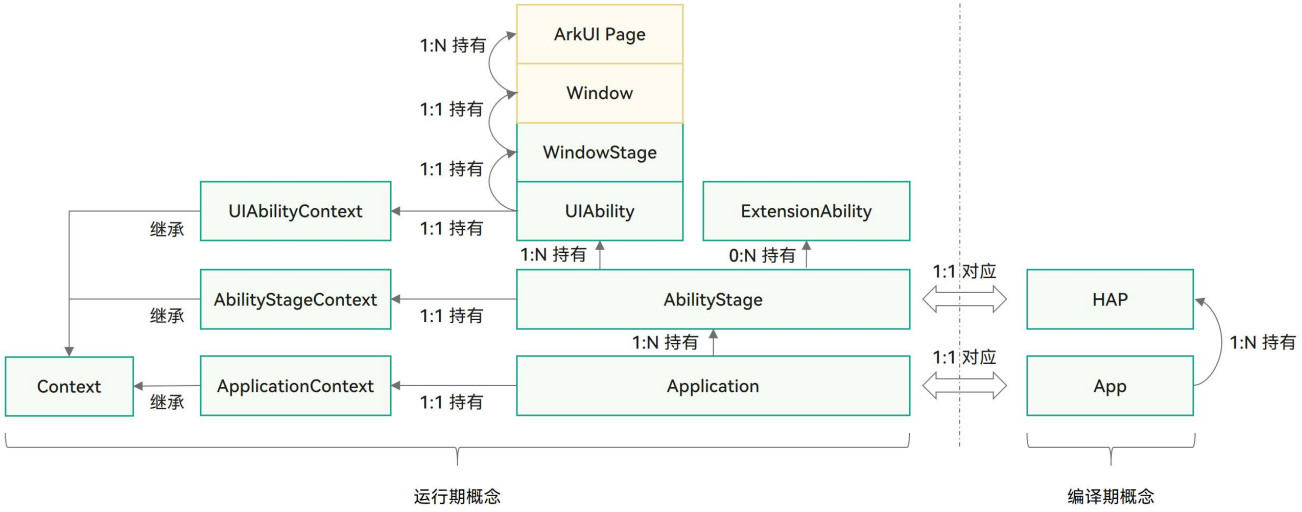
- 总结
  - 不同函数签名是不同的函数定义
    - 不要只关注函数名称而是要关注函数签名
    - 函数签名 = 函数名称 + 形参列表
  - 掌握报错机制
    - try-catch（语句支持）
    - err-callback（函数内置）
  - 认识到日志的重要性
    - console.err()
    - console.log()
  - 掌握使用API参考文档的方法
    - 当你认识到API参考文档比书籍和视频都重要时，你就可以开始职业生涯了

Want机制（1/4）

- 传递信息的另一种方式—Want
  - Want是UIAbilityContext.startAbility的一个参数

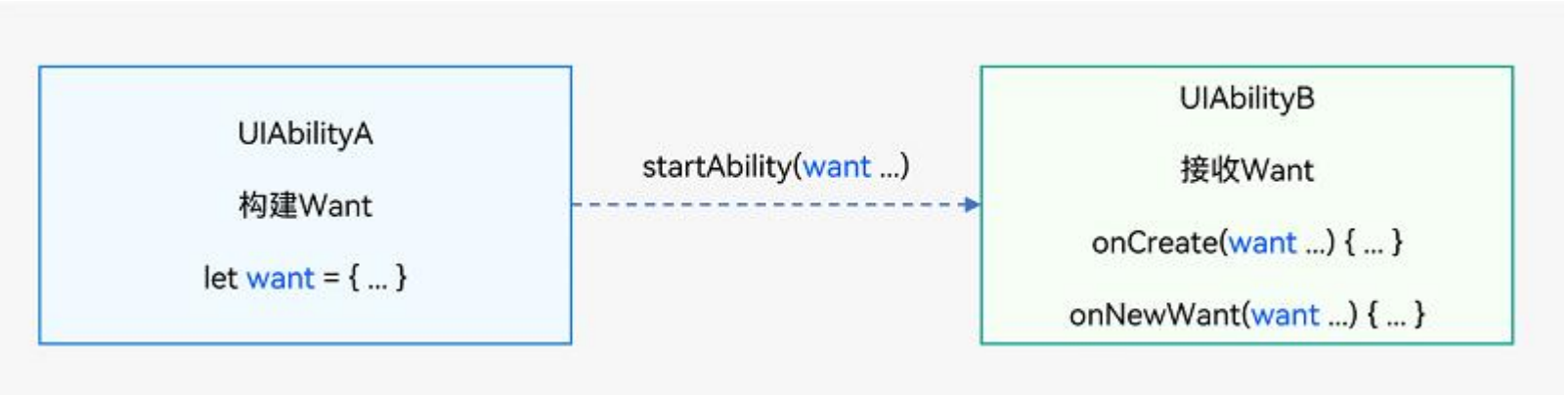
```
startAbility(want: Want, callback: AsyncCallback<void>): void
```

  - 注意区分Want方式传递信息与前述pushUrl方式的不同
    - pushUrl是pages之间的传递，此时只有一个UIAbility
    - Want是不同UIAblity下的不同pages之间的传递



Want机制（2/4）

- Want用法示意
  - 当Ability A启动Ability B并需要传入一些数据给AbilityB时，Want可以作为一个数据载体将数据传给Ability B



Want机制（3/4）

- Want的类型
  - 显式Want
    - 在启动Ability时指定了abilityName和bundleName的Want称为显式Want

```
let want = {
    deviceId: '',
    bundleName: 'com.example.myapplication',
    abilityName: 'calleeAbility',
};
```

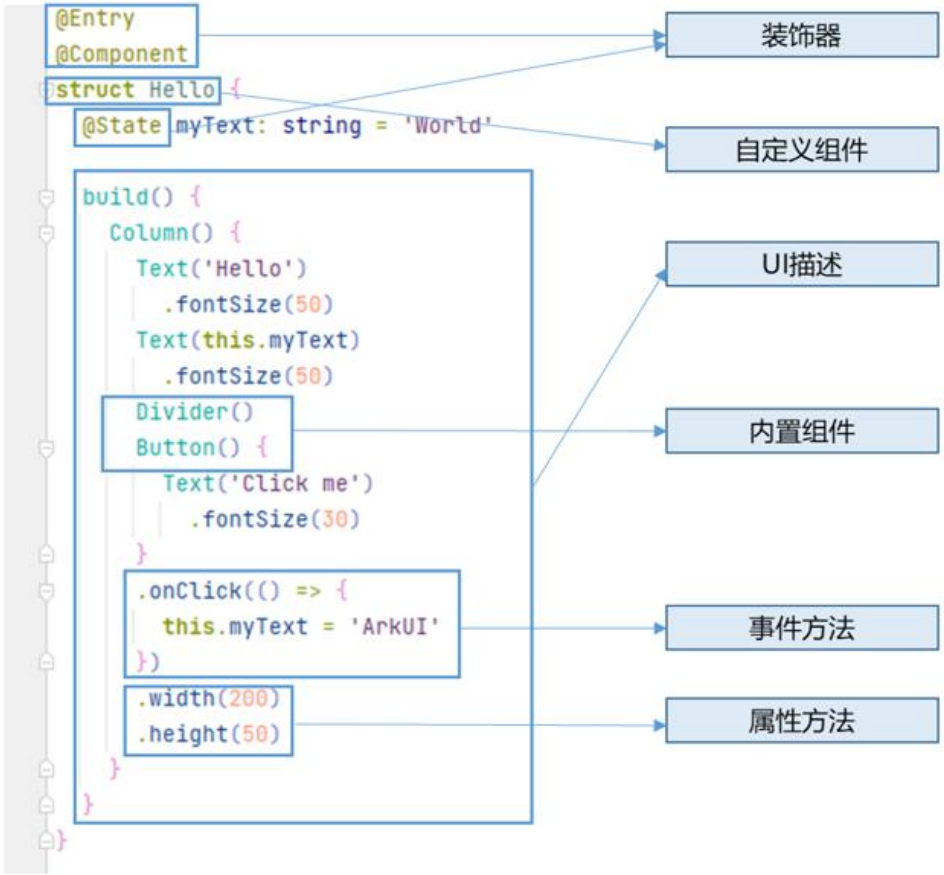
Want机制（4/4）

- Want的类型（contd.）
  - 隐式Want
    - 在启动Ability时未指定abilityName的Want称为隐式Want
    - 在当前应用中使用其他应用提供的某个能力，而不关心提供该能力的具体应用
      - 隐式Want使用skills标签来定义需要使用的能力

```
let want = {
    action: 'ohos.want.action.search',
    entities: [ 'entity.system.browsable' ],
    uri: 'https://www.test.com:8080/query/student',
    type: 'text/plain',
};
```

UI开发（1/2）

- 声明式开发范式



UI开发（2/2）

- 常用组件
  - 基础组件
  - 容器组件
  - 媒体组件
  - 绘制组件
  - 画布组件

- 重点在代码实际演示
  - 本节课演示部分代表性基础组件

Q&A

Good Luck!