iOS Swift编程：苹果应用开发入门

01

了解iOS开发环境和工具

介绍macOS操作系统和Xcode集成开发环境

macOS操作系统

- 基于BSD Unix的开源操作系统
- 提供稳定且安全的环境，适用于开发iOS应用
- 支持多任务和图形化界面，便于开发者使用

Xcode集成开发环境

- 苹果官方提供的开发工具
- 集成了代码编辑、构建、调试、性能分析等功能
- 支持Swift、Objective-C等多种编程语言
- 提供强大的界面和丰富的插件，提高开发效率

选择合适的工作空间

- 根据项目需求和设备类型，选择合适的工作空间
- 创建新的工作空间或打开已有工作空间
- 管理多个项目和工作组，方便团队协作

安装Xcode和配置开发环境

- 下载Xcode安装包
  - 访问官方网站
  - 选择合适的版本
  - 下载安装包并进行安装
- 安装命令行工具
  - 打开Xcode
  - 选择Preferences
  - 选择Components
  - 下载并安装Command Line Tools
- 配置环境变量
  - 打开终端应用
  - 输入echo 'export PATH="/Applications/Xcode.app/Contents/Developer/usr/bin:$PATH"' >> ~/.bash_profile
  - 输入source ~/.bash_profile使配置生效
- 创建工作空间
  - 打开Xcode
  - 选择File>New>Workspace
  - 选择合适的工作空间位置和名称

02

学习Swift编程基础

了解Swift数据类型和变量声明

- Swift数据类型
  - 整数类型：Int,UInt,Int8,Int16,Int32,Int64
  - 浮点数类型：Double,Float
  - 布尔类型：Bool
  - 字符类型：Character,String
  - 可选类型：Optional
  - 元组类型：Tuple
  - 数组类型：Array
  - 字典类型：Dictionary
- 变量声明
  - 使用let声明不可变变量
  - 使用var声明可变变量
  - 初始化变量时可选类型必须使用强制解包
  - 使用var声明的变量必须显式初始化

掌握Swift控制结构和循环语句

- 控制流
  - 条件语句：if,else,guard
    - 使用if、else进行条件判断
    - 使用guard进行条件判断，提前结束函数
  - 循环语句：for,while
    - 使用for-in进行循环遍历数组、集合等
    - 使用while进行条件循环
    - 使用repeat-while进行逆条件循环
    - 使用for-condition-break和for-condition-continue控制循环流程

学习Swift函数和闭包

函数

- 使用func声明函数
- 函数的参数和返回值可以使用类型推断
- 函数可以嵌套调用
- 函数可以作为参数传递给其他函数

闭包

- 闭包是一种匿名函数
- 闭包可以捕获其所在的上下文中的变量和常量
- 闭包在函数参数中使用非常灵活
- 闭包可以作为参数传递给其他函数
- 闭包可以用来实现高阶函数

03

掌握iOS应用结构和组织方式

了解iOS应用的生命周期和方法

- iOS应用生命周期
  - 应用启动
  - 应用进入前台
  - 应用进入后台
  - 应用被挂起
  - 应用被终止
  - 应用恢复
- 生命周期方法
  - application(_:didFinishLaunchingWithOptions:)：应用启动时调用
  - applicationWillEnterForeground(_:)：应用进入前台时调用
  - applicationDidEnterBackground(_:)：应用进入后台时调用
  - applicationDidBecomeActive(_:)：应用被恢复时调用
  - applicationWillResignActive(_:)：应用即将进入后台时调用
  - applicationWillTerminate(_:)：应用被终止时调用
- 应用状态
  - 未激活状态：UIApplicationStateInactive
  - 活跃状态：UIApplicationStateActive
  - 背景状态：UIApplicationStateBackground

学习iOS应用的界面布局和控件

- 界面布局
  - 使用Auto Layout进行布局
  - 使用Stack View进行布局
  - 使用Size Classes进行响应式布局
- 控件
  - 标签：UILabel
  - 文本框：UITextField
  - 按钮：UIButton
  - 开关：UISwitch
  - 滑块：UISlider
  - 分页面控制器：UIPageControl
  - 表视图：UITableView
  - 集合视图：UICollectionView
  - 导航控制器：UINavigationController
  - 标签控制器：UITabBarController
  - 工具栏：UIToolbar
  - 分割视图控制器：UISplitViewController

掌握iOS应用的导航和模态视图

模态视图

- 使用UIViewController的present(_:animated:completion:)方法展示模态视图
- 使用UIViewController的dismiss(animated:completion:)方法关闭模态视图
- 使用UIPopoverController展示弹出视图
- 使用SFSafariViewController和WKWebView访问网页

应用导航

- 使用UINavigationController进行导航控制
- 使用UIViewController的navigationItem和toolbarItems进行自定义导航栏和工具栏
- 使用UINavigationController和UIViewController的pushViewController(_:animated:)和popViewController(animated:)方法进行页面跳转

04

学习Swift语言的高级特性

理解协议和扩展

01

协议

- 定义了一组方法的规范
- 可以让不同的类型实现相同的方法
- 用于实现多态

02

扩展

- 增加了新的方法、属性和下标
- 可以扩展已有类型的功能
- 扩展可以覆盖已有类型的方法

03

协议类型推断

- 使用where关键字进行协议类型推断
- 使用@typealias创建协议类型的别名