



回顾 2023 及

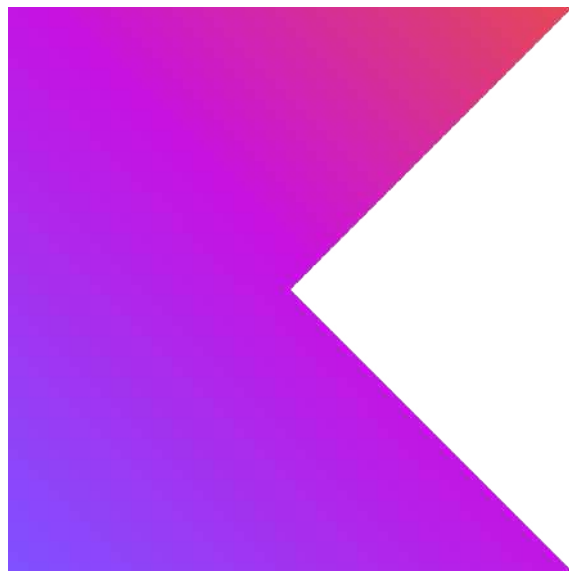
Kotlin Multiplatform 最新信息

范圣佑 & Pamela Hill

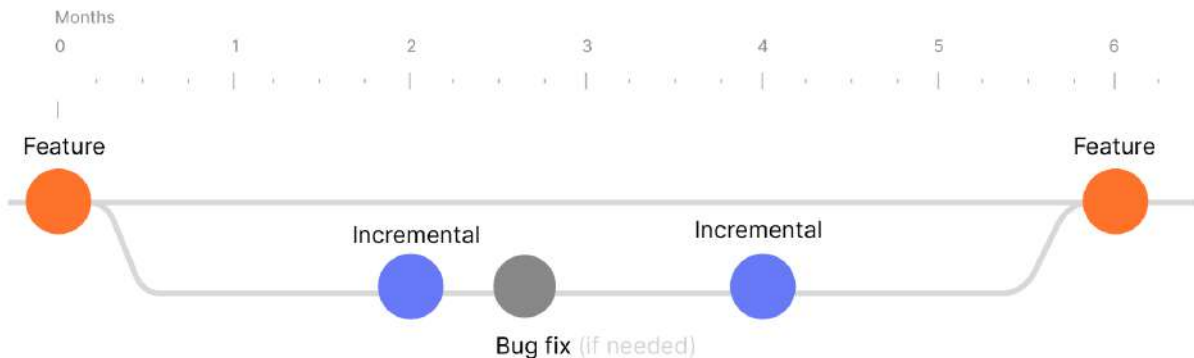


Kotlin 编程语言

- 2011 年由 JetBrains 团队发布
- 通用型、多用途语言
- 静态类型
- 面向对象 (OOP) + 函数式 (FP)
- 以 Apache 2.0 开源



Kotlin 发布周期



Feature release
Language changes
(1.4, 1.5)



Every 6 months
Not ready? Postpone
the feature, not the release.

Incremental release
Tooling improvements
(1.3.70, 1.4.20)



Every 2-3 months
in between feature releases

Bug fix release
Bug fixes
(1.3.72)



1-2 weeks
After an incremental release
if fixes are needed

v1.9.21



- 适用于所有目标的 K2 现已进入测试版阶段
- 稳定的 Kotlin Multiplatform
- 用于设置多平台项目的新默认层次结构模板
- Kotlin Multiplatform 中全面支持 Gradle 配置缓存
- Kotlin/Native 中默认启用自定义内存分配器
- Kotlin/Native 中垃圾回收器的性能改进
- Kotlin/Wasm 中的新目标和重命名目标，支持最新的 Wasm GC
- Kotlin/Wasm 的标准库中支持 WASI API

K2 编译器 (Beta)



```
// build.gradle.kts
kotlin {
    sourceSets.all {
        languageSettings {
            languageVersion = "2.0"
        }
    }
}
```

随着 1.9.20 版本的发布，新 K2 编译器已面向所有平台 (包括 JVM、Native、JS 和 Wasm) 进入测试版阶段，现在可以在任何 Kotlin 项目中试用 K2。

Kotlin Multiplatform 已可投入生产环境



博文链接

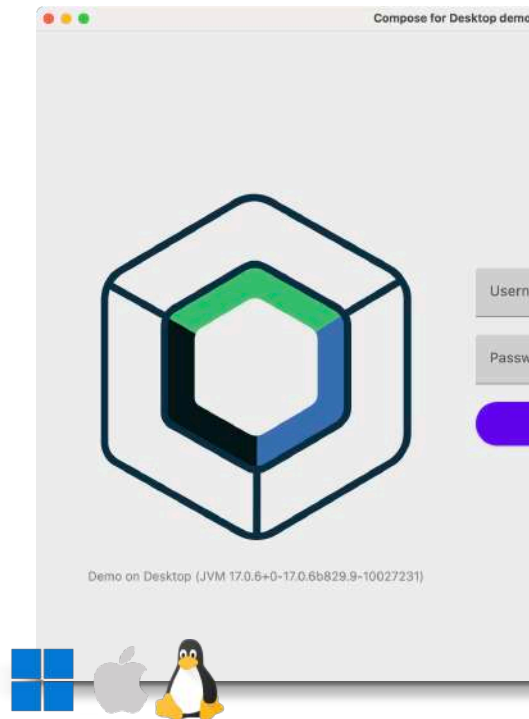
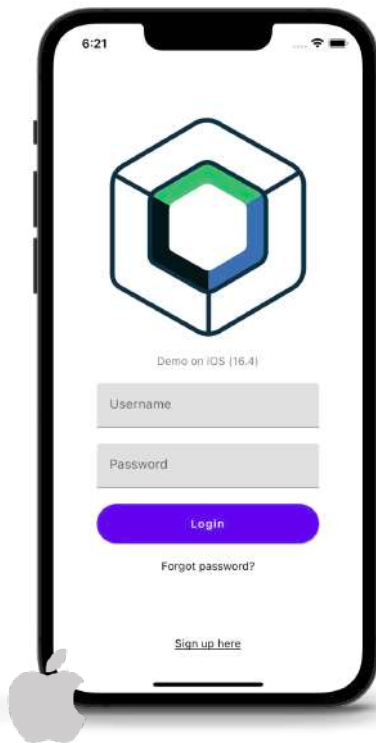
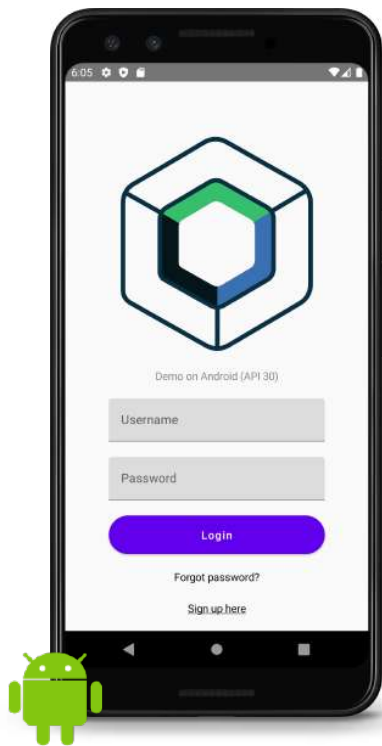
A screenshot of a web browser displaying a blog post on the JetBrains website. The browser's address bar shows 'blog.jetbrains.com'. The page has a navigation menu with '全部', '新闻', '最新发布', 'Multiplatform', and 'Ecosystem'. The main heading of the article is 'Kotlin Multiplatform 已经稳定并且可以投入生产环境'. The author is identified as 'Sue' with a profile picture and the date '2023年11月7日'. Below the author information, there is a section for 'Read this post in other languages:' with links for 'English, 日本語, 한국어, Čeština'. The main text of the article states that Kotlin Multiplatform is an open-source technology built by JetBrains, allowing developers to share code across platforms while retaining native programming advantages. It mentions that the technology has reached a stable state and is ready for production. At the bottom of the article, there is a blue button labeled '开始' (Start). On the right side of the page, there is a sidebar with a blue vertical bar and several links: '信心满满地按照您的方式共享代码', '利用不断发展的 Kotlin Multiplatform 生态系统的力量', '通过 Compose Multiplatform 获得更多代码共享自由', '探索 Kotlin Multiplatform 的未来', and 'Kotlin Multiplatform 使用入门'.

Kotlin Multiplatform 全版图



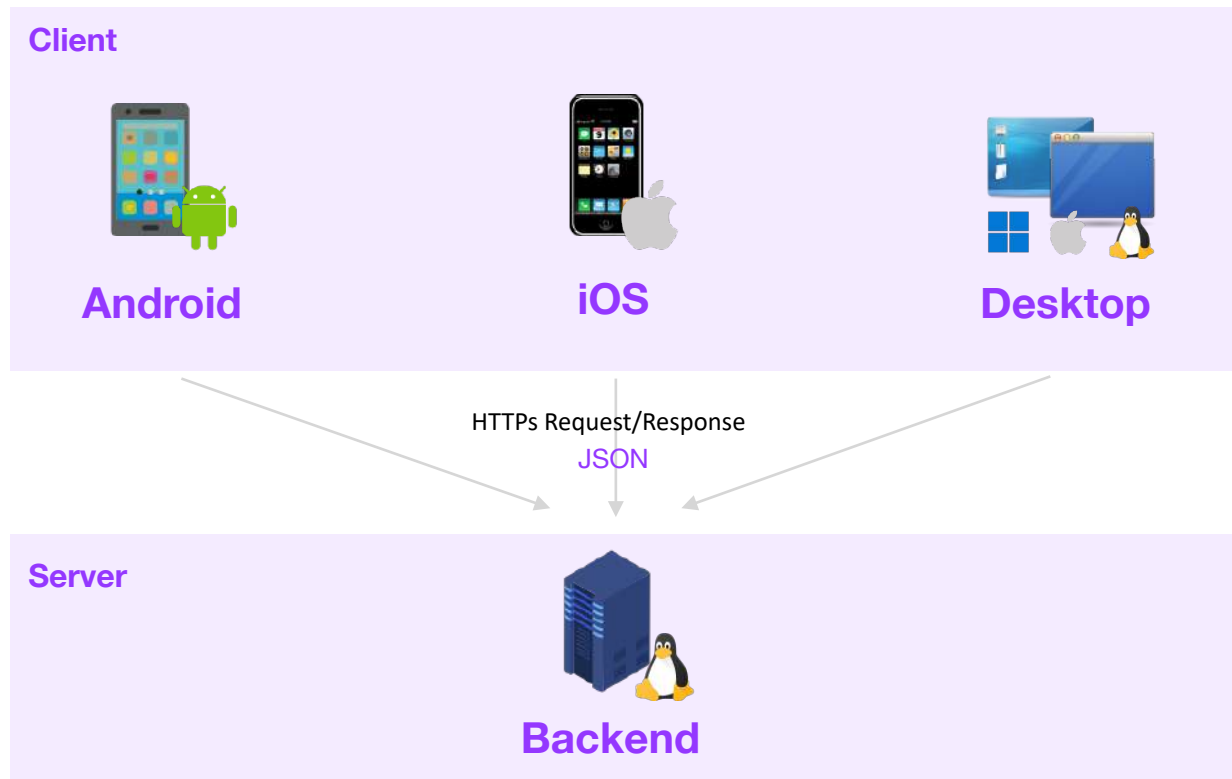
Kotlin Multiplatform 示例

- 集成多平台库
- 多平台共享 UI
- 平台专用 API
- 共用业务逻辑
- 后端 API 服务



4 个平台实现

- Android
- iOS
- Desktop
- Backend API

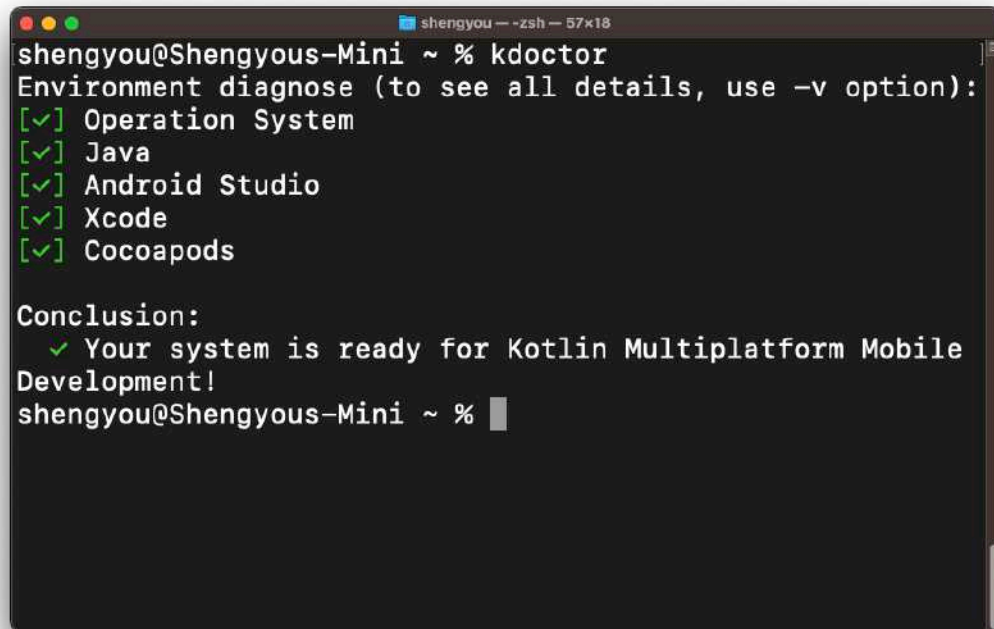


搭建开发环境

- Mac with macOS
- JDK
- Android Studio
- Xcode (+ SDK)
- Cocoapods

kdoctor 命令工具

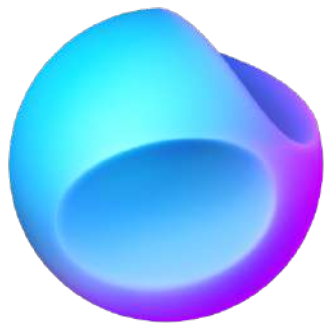
通过 Homebrew 安装 kdoctor，可检查环境是否符合开发 KMP 的需求？

A terminal window with a dark background and light text. The title bar shows 'shengyou -- zsh -- 57x18'. The prompt is 'shengyou@Shengyous-Mini ~ %'. The command 'kdoctor' has been executed, resulting in the following output:

```
shengyou@Shengyous-Mini ~ % kdoctor
Environment diagnose (to see all details, use -v option):
[✓] Operation System
[✓] Java
[✓] Android Studio
[✓] Xcode
[✓] Cocoapods

Conclusion:
  ✓ Your system is ready for Kotlin Multiplatform Mobile
  Development!
shengyou@Shengyous-Mini ~ %
```

开发工具 - JetBrains Fleet



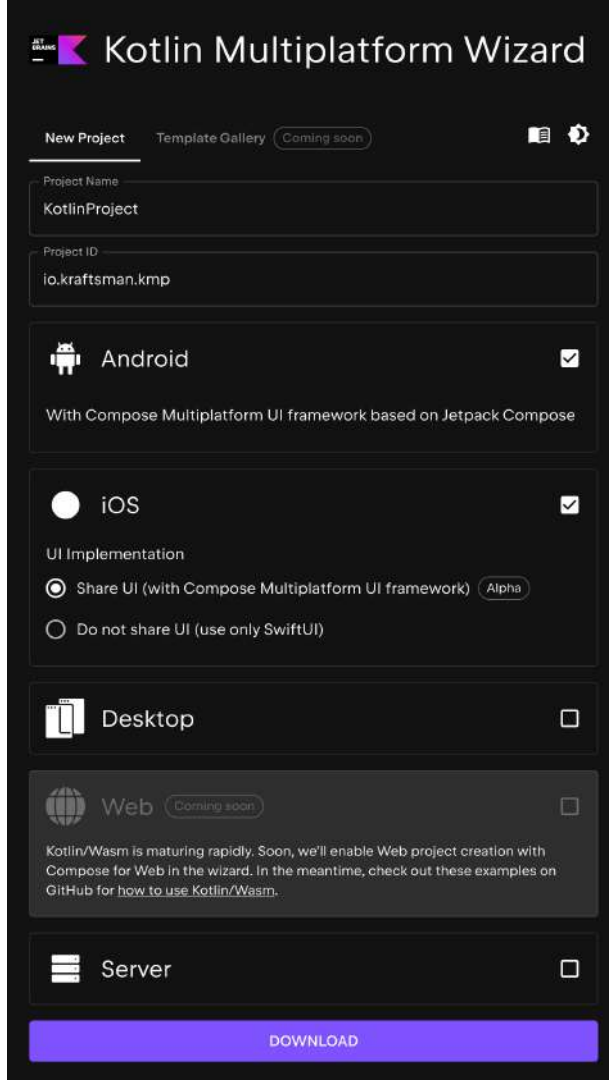
文档链接

The screenshot shows a web browser window displaying a blog post on the JetBrains website. The browser's address bar shows 'blog.jetbrains.com'. The page has a navigation menu with '全部', '新闻', '最新发布', 'Multiplatform', and 'Ecosystem'. The main content area features a 'Multiplatform' tag and the article title '欢迎使用带有 Kotlin Multiplatform Tooling 的 Fleet'. The author is 'Sue' and the date is '2023年11月15日'. Below the title, there is a link to read the post in other languages, currently set to 'English'. The article text discusses Kotlin Multiplatform (KMP) as an open-source technology for sharing code across multiple platforms like iOS, Android, desktop, and web. It mentions that the Fleet tool now supports a preview version of Kotlin Multiplatform, simplifying application development. A '开始' (Start) button is visible at the bottom of the article. On the right side, there is a sidebar with a list of navigation links: '简洁', '多语言编程', '代码导航', '重构', '调试', '单一 IDE', '定价模型', '我们对 IntelliJ IDEA 和 IntelliJ 平台的承诺', '如何开始', and '我们'. At the bottom of the sidebar is a QR code and the text '博文链接'.

博文链接

创建项目

- 开启 Kotlin Multiplatform Wizard
<https://kmp.jetbrains.com>
- 选择目标平台
- 下载 Zip 文件
- 解开 Zip 文件
- 以 JetBrains Fleet 开启项目



工程结构

- composeApp → Compose 多平台代码
 - commonMain → 多平台共用代码 
 - androidMain → Android 平台专用代码
 - iosMain → iOS 平台专用代码
 - desktopMain → JVM/Desktop 平台专用代码
- iosApp → iOS 主程序进入点
- server → 后端 API 代码

集成 Kotlin Multiplatform 库

- [Ktor](#) - HTTP Client
- [kotlinx.serialization](#) - JSON serialization/deserialization
- [kotlinx.coroutines](#) - Coroutine
- [Voyager](#) - Navigation 、 ViewModel
- [Kamel](#) - Asynchronous Media Loading

多平台生态

The screenshot shows the GitHub repository page for 'kmp-awesome'. At the top, it indicates the repository is public and has 49 watches, 107 forks, and 2k stars. The main content area features a 'README.MD' section with the title 'Awesome Kotlin Multiplatform' and a navigation menu with categories: UI (Compose Multiplatform), Presentation, Business / Domain, and Data / Core. Below the menu, there are badges for 'welcome', 'awesome', 'stars 2k', and 'maven-central v1.9.20'. The text describes Kotlin Multiplatform technology and its benefits. A 'Contents' table lists various categories like Tooling, Log, Network, Storage, Device, Dependency Injection, etc.

Awesome Kotlin Multiplatform

- UI (Compose Multiplatform)
- Presentation
- Business / Domain
- Data / Core

PRs welcome awesome stars 2k maven-central v1.9.20

Kotlin Multiplatform technology simplifies the development of cross-platform projects. It reduces time spent writing and maintaining the same code for different platforms while retaining the flexibility and benefits of native programming.

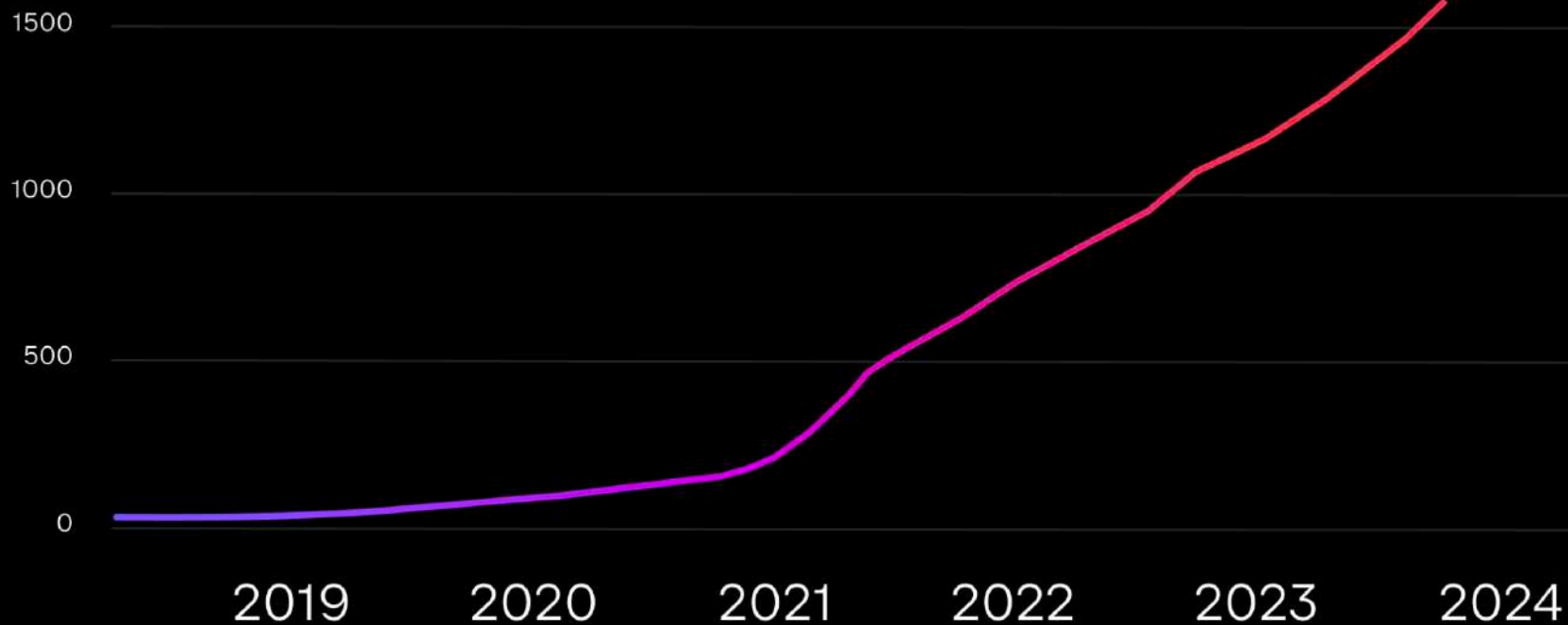
This list contains libraries which support iOS and Android targets in first place.

Contents

Tooling	Log	Network
Storage	Device	Dependency Injection
Architecture	Analytics	Test
Crypto	File	Language extensions
Serializer	Date-Time	Asynchronous
Compose UI	Graphics	Service SDK
Arithmetic	Resources	Utils

github.com/terrakok/kmp-awesome

逐年成长的多平台库



Ktor 后端实现

- 语法简单易学
- 轻量 Web 框架
- 支持 Async



文档链接

A screenshot of a web browser displaying the Ktor 2.3.6 documentation. The browser's address bar shows 'ktor.io'. The page title is 'Ktor Server / Getting started with a Ktor Server / Creating HTTP APIs'. The main heading is 'Creating HTTP APIs', with a sub-heading 'Code example: tutorial-http-api' and 'Used plugins: Routing, ContentNegotiation, kotlinx.serialization'. The page includes a sidebar with a navigation menu, a main content area with introductory text, and a right-hand sidebar with a table of contents for the tutorial steps.

Welcome

- ▼ Ktor Server
 - ▼ Getting started with a Ktor Server
 - Creating a new Ktor project
 - Creating HTTP APIs**
 - ▶ Creating a website
 - ▶ Creating a WebSocket chat
 - ▶ Developing applications
 - ▶ Running and debugging
 - Testing
 - ▶ Database integration
 - ▶ Deployment
 - ▶ Extending Ktor
- ▶ Ktor Client
- ▶ Releases
- ▶ Migrating from other frameworks
- FAQ

Ktor Server / Getting started with a Ktor Server / Creating HTTP APIs

Creating HTTP APIs

Edit page Last modified: 13 November 2023

Code example: tutorial-http-api ↗

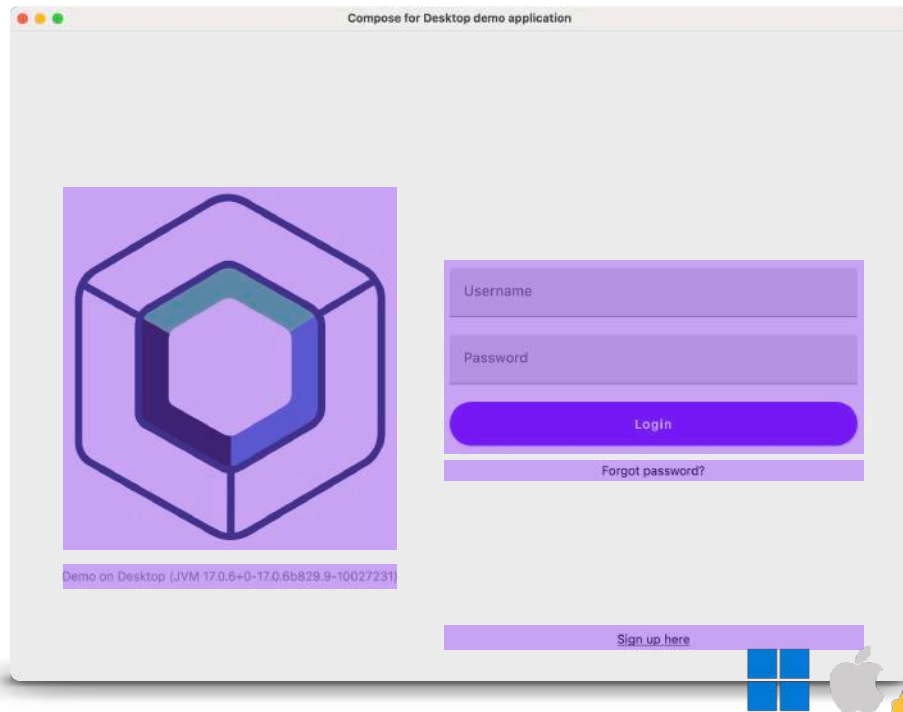
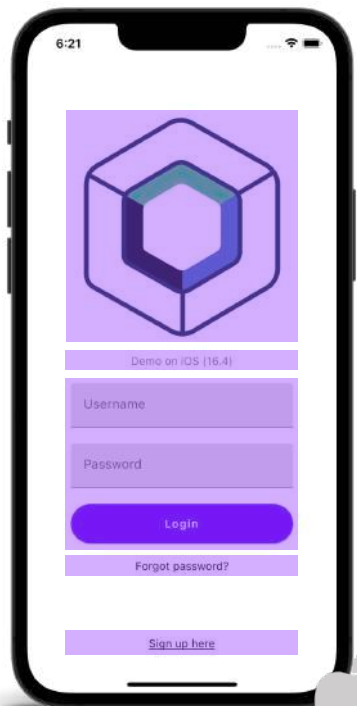
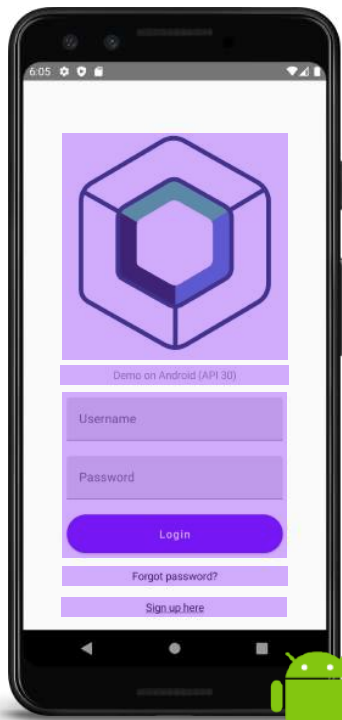
Used plugins: Routing, ContentNegotiation, kotlinx.serialization

In this tutorial, we're going to create an HTTP API that can serve as a backend for any application, be it mobile, web, desktop, or even a B2B service. We will see how routes are defined and structured, how serialization plugins help simplify tedious tasks, and how we can test parts of our application both manually and automated.

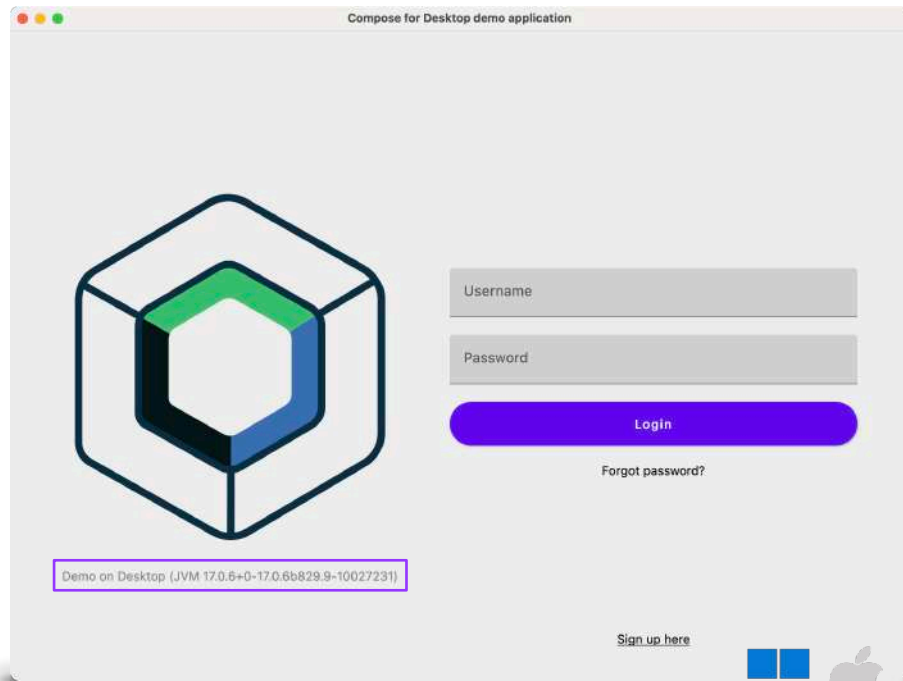
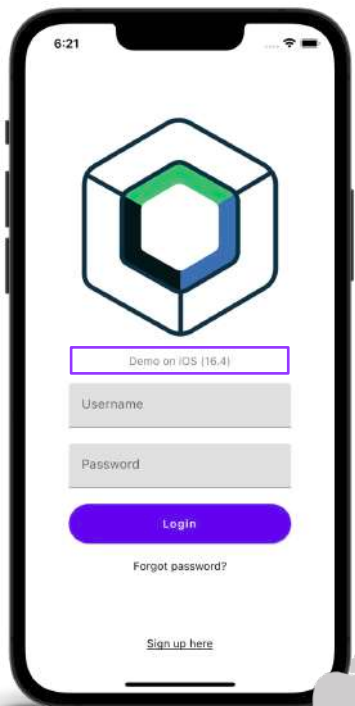
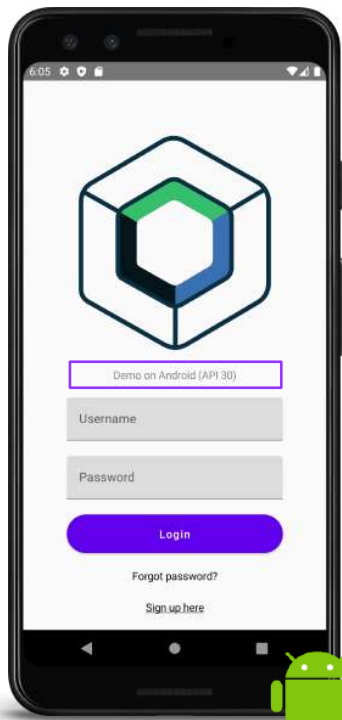
Throughout the tutorial, we'll build a simple JSON API that allows us to query information about the customers of our fictitious business, as well as the orders we currently want to fulfill. We will build a convenient way of listing all customers and orders in our system, get information for individual customers and orders, and provide the functionality to add new entries and remove old entries.

- Creating HTTP APIs
 - Prerequisites
 - Create a new Ktor project
 - Examine the project
 - Dependencies
 - Configurations: application.conf and logback.xml
 - Source code
 - Customer routes
 - Create the Customer model
 - Create the Customer storage
 - Define the routing for customers
 - Register the routes
 - Order routes
 - Create the Order model
 - Define the routing for orders

多平台间共用 UI



平台专用 API



共用业务逻辑



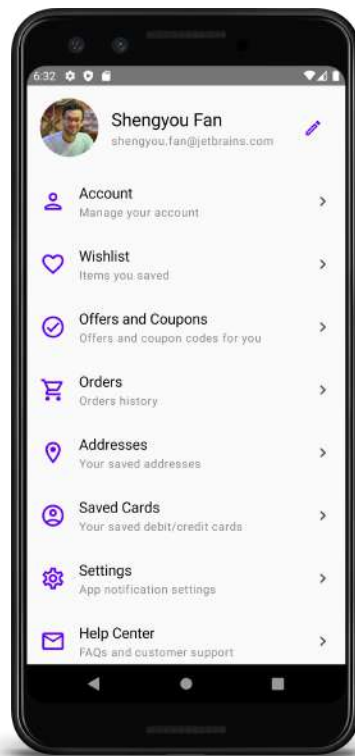
发送 HTTP Request



接收 HTTP Response

UI 换页

载入网络图片





CROSS-PLATFORM

OR

NATIVE

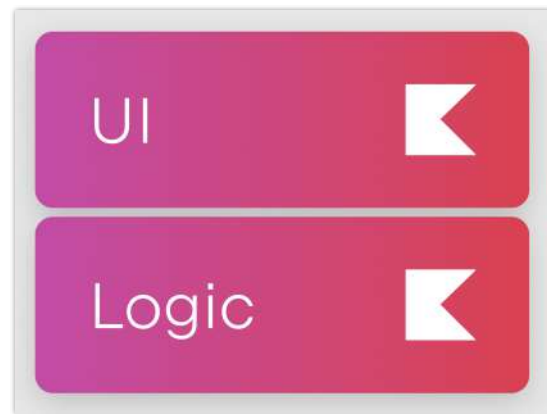
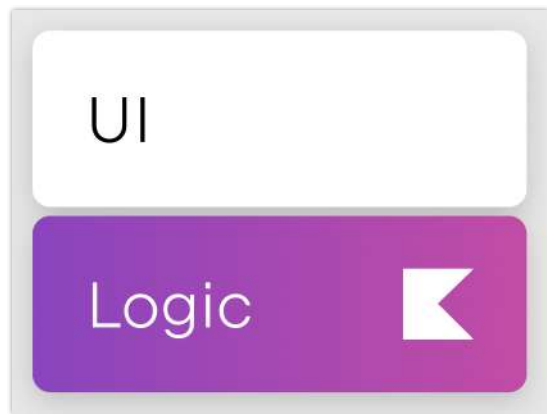
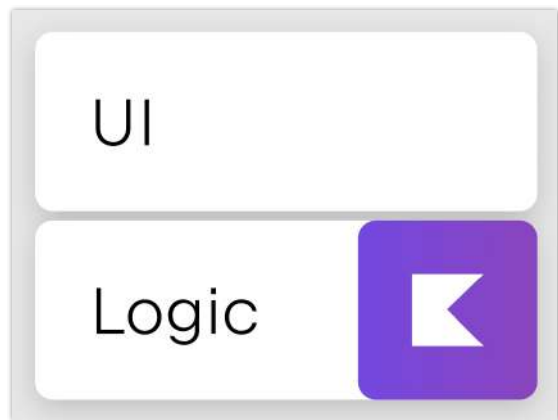


CROSS-PLATFORM

AND

NATIVE

依需求決定共享代碼的比例



采用 Kotlin Multiplatform 的企业

 **Alibaba**

Bai **百度**

 **Meituan**

 **携程旅行**

Google

NETFLIX

VISA

twitter 

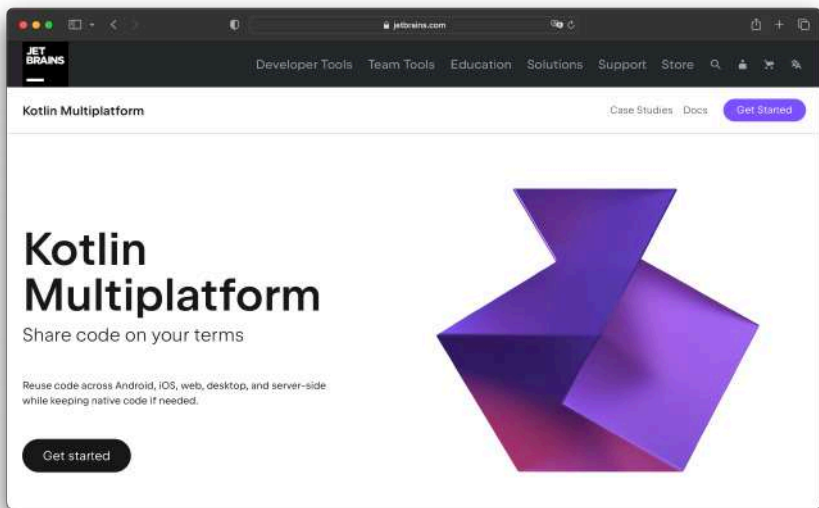
 **快手**



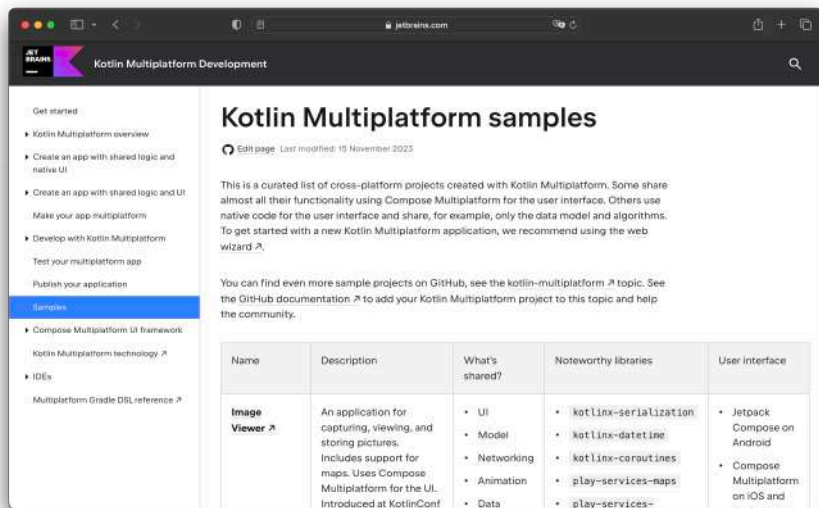
 **reddit**

kotlin.in/kmp-case-studies

Kotlin Multiplatform 学习材料

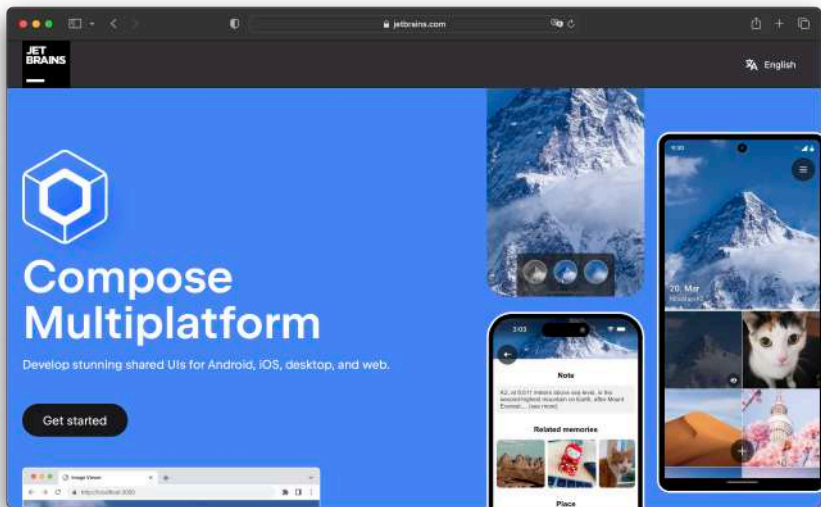


Kotlin Multiplatform 官方页面

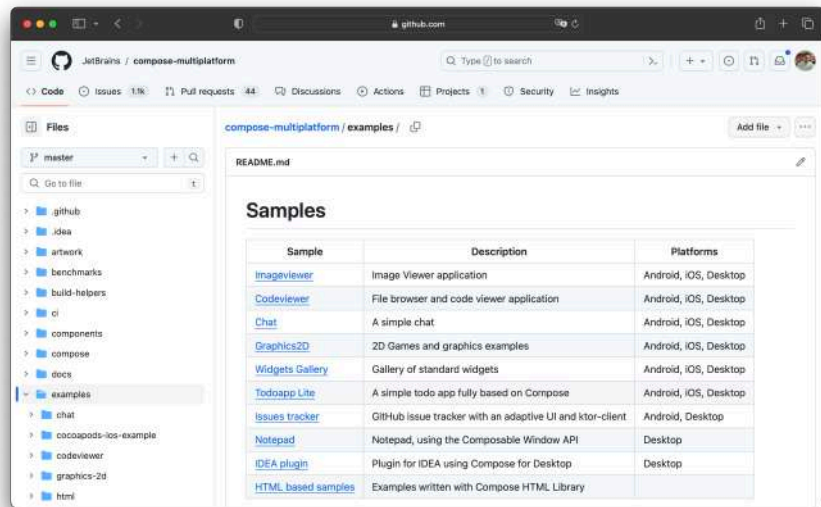


Kotlin Multiplatform 演示项目集

Compose Multiplatform 学习材料



Compose Multiplatform 官方页面



Compose Multiplatform 官方示例

Kotlin 炉边漫谈播客 - 收听 KMP 企业实例



#5 手机开发编年史 (携程)



#8 来自阿里巴巴及美团的
Kotlin Multiplatform 应用实例

👉 投稿你的 KMP 案例





KOTLINCONF

2024



COPENHAGEN

kotl.in/cfp-2024

给 Swifties/iOS 开发者的 KMP 指南



JetBrains 布道师 Pamela Hill
跟大家介绍 Kotlin 与 Swift 间的互操作技巧



Have a nice Kotlin
谢谢参与

