Kotlin/Wasm 的探索之路

范圣佑 (Shengyou Fan)
JetBrains Developer Advocate
QCon Beijing 2024 (4.11)

InfoQ 极客传媒

已结束 北京站 2000 人

QCon 全球软件开发大会 暨智能软件开发生态展

架构演进 | AIGC
研发效能 | 向量数据库

6 月 14–15 日 深圳站 1000 人

ArchSummit 全球架构师峰会 暨智能软件开发生态展

业务架构 | 高可用
AI 架构 | 架构师成长

10 月 18–19 日 上海站 1500 人

QCon 全球软件开发大会 暨智能软件开发生态展

AIGC | AI 原生应用
软件工程 | 金融科技

4月 5月 6月 8月 10月 12月

5 月 17–18 日 北京站 2000 人

AiCon 全球人工智能开发与应用大会 暨大模型应用生态展

通用大模型 | AI Agent
LLMOps | AI Infra

8 月 16–17 日 上海站 1000 人

FCon 全球金融科技大会 暨智能软件开发生态展

金融大模型 | 核心系统
数字化转型 | 数据安全

12 月 20–21 日 北京站 1000 人

ArchSummit 全球架构师峰会 暨智能软件开发生态展

未来架构 | 高可用高并发
AI 应用 | 架构师思维

极客邦科技 2024 年会议规划

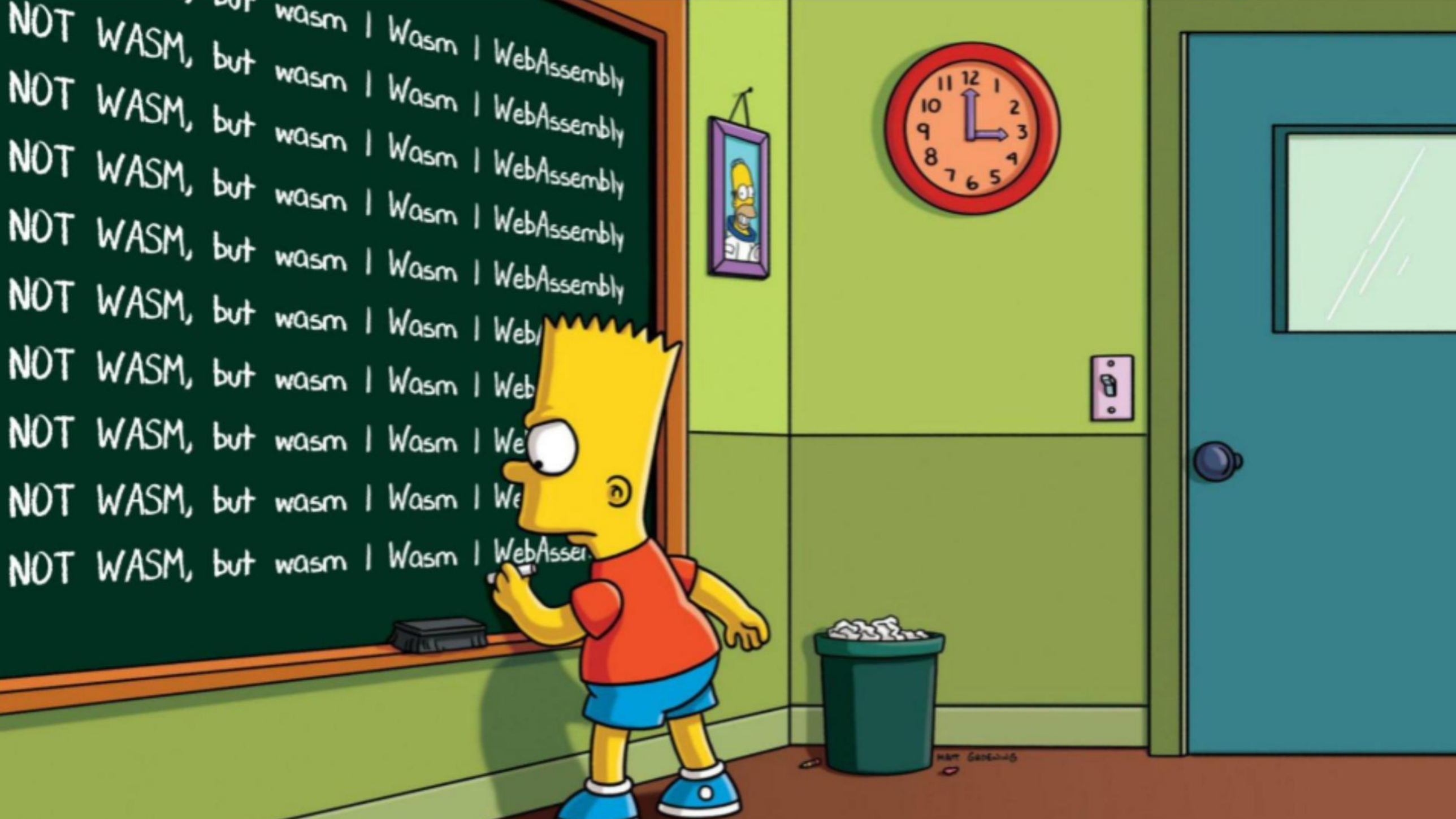
促进软件开发及相关领域知识与创新的传播

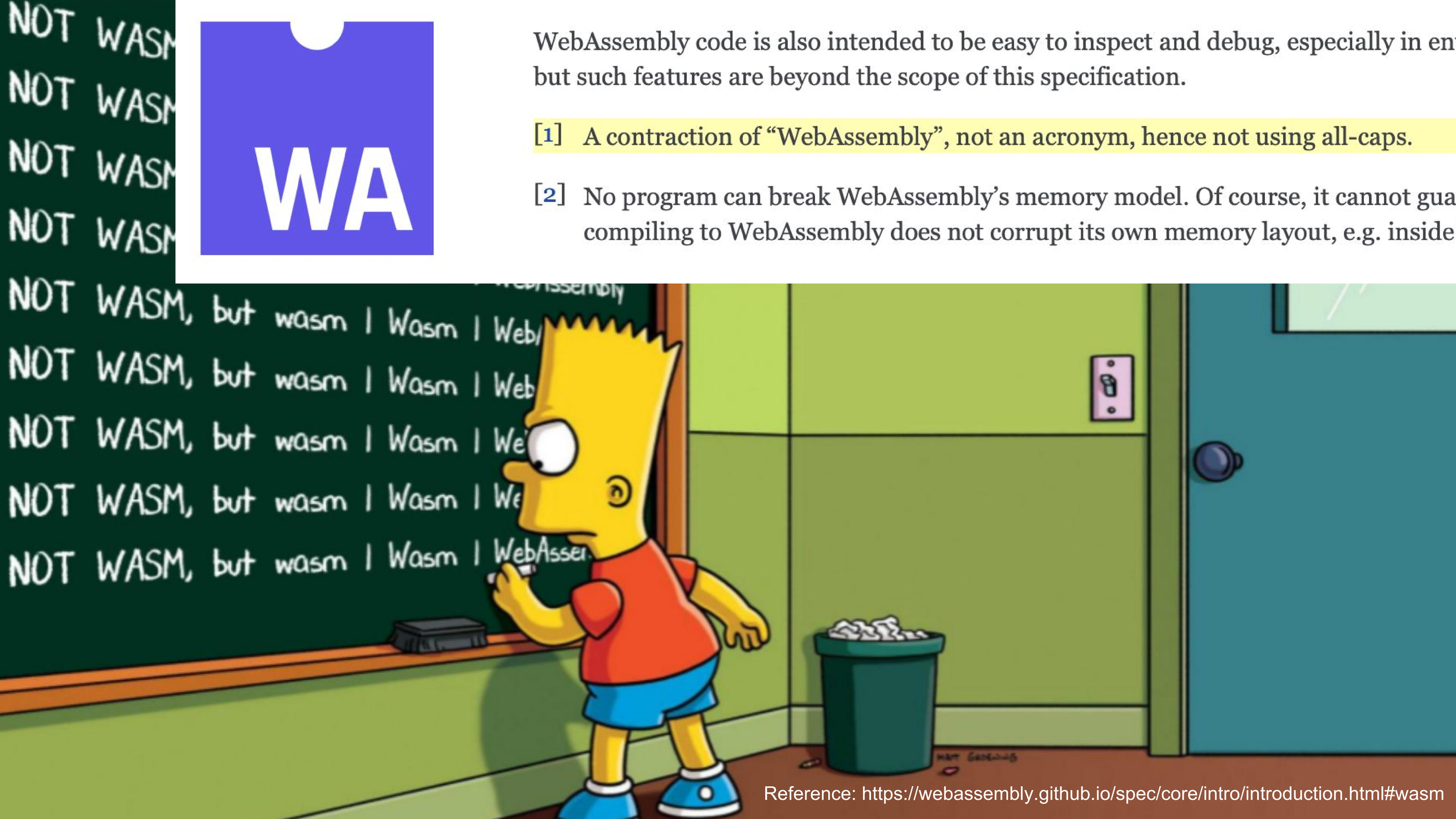
访问大会官网

参会咨询

本日话题

- Kotlin/Wasm 技术背景
- Kotlin/Wasm 技术演示
- 发展现状及成果
- 展望未来
- 学习材料


NOT WASM, but wasm | Wasm | WebAssembly
NOT WASM, but wasm | Wasm | WebAssembly
NOT WASM, but wasm | Wasm | WebAssembly
NOT WASM, but wasm | Wasm | WebAssembly
NOT WASM, but wasm | Wasm | WebAssembly
NOT WASM, but wasm | Wasm | Web
NOT WASM, but wasm | Wasm | Web
NOT WASM, but wasm | Wasm | We
NOT WASM, but wasm | Wasm | We
NOT WASM, but wasm | Wasm | WebAsser


WA
WebAssembly code is also intended to be easy to inspect and debug, especially in en
but such features are beyond the scope of this specification.
[1] A contraction of “WebAssembly”, not an acronym, hence not using all-caps.
[2] No program can break WebAssembly’s memory model. Of course, it cannot gua
compiling to WebAssembly does not corrupt its own memory layout, e.g. inside
NOT WASM, but wasm | Wasm | Web
NOT WASM, but wasm | Wasm | Web
NOT WASM, but wasm | Wasm | We
NOT WASM, but wasm | Wasm | We
NOT WASM, but wasm | Wasm | WebAsse
Reference: https://webassembly.github.io/spec/core/intro/introduction.html#wasm

技术背景

- 简介 Kotlin 语言
- WebAssembly 的潜力
- 简介 Kotlin/Wasm 项目

Kotlin

- 空安全
- 静态类型
- 语法简洁且富有表现力
- 支持多平台开发
- 务实且实用
- “乐趣”无穷的开发体验

（使用 **fun** 关键字声明函式）

由 JET BRAINS 研发的编程语言

发展时间线

2011

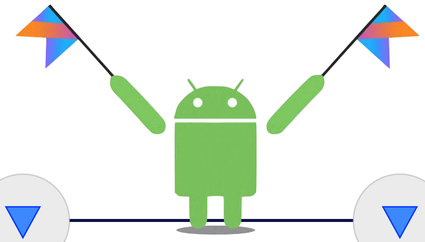
JetBrains 正式公布
Kotlin 编程语言

2016

Kotlin 1.0
正式发布

2017

Google I/O 宣布
支持以 Kotlin 开发
Android App



2019

Google I/O 宣布
Android 开发
优先支持 Kotlin

2022

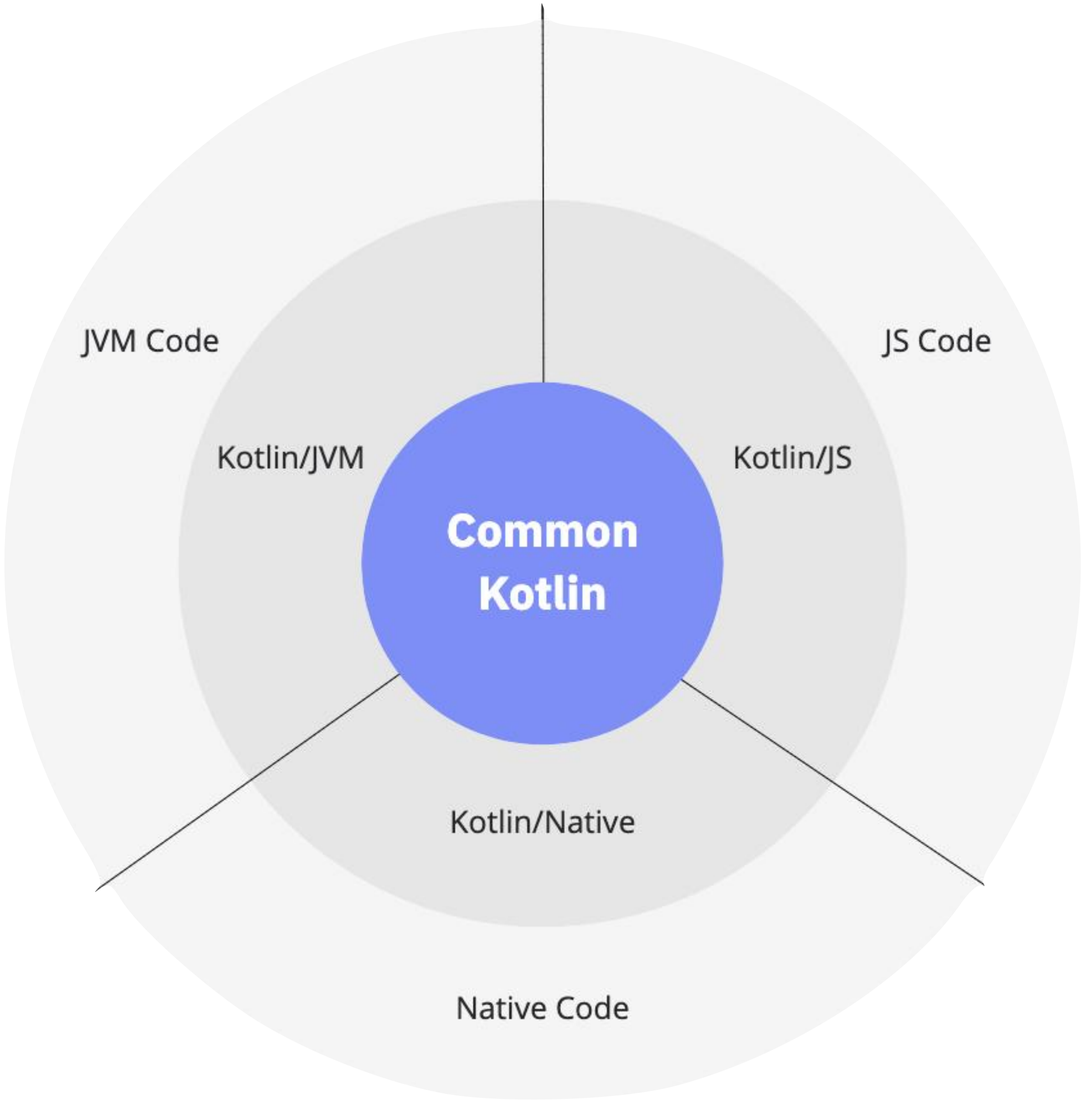
发布 Kotlin 1.7 - 1.8

2023

发布 Kotlin 1.9

架构与工具链

- 编译器（预计于 2024 年发布新版 K2 编译器）
- 多平台发布
- 支持编译器插件
- JVM 生态工具链



多平台开发

Server

Kotlin/JVM

Web

Kotlin/JS

Kotlin/Wasm

Desktop

Kotlin/JVM

Android

Kotlin/JVM

iOS

Kotlin/Native

广泛采用

移动端

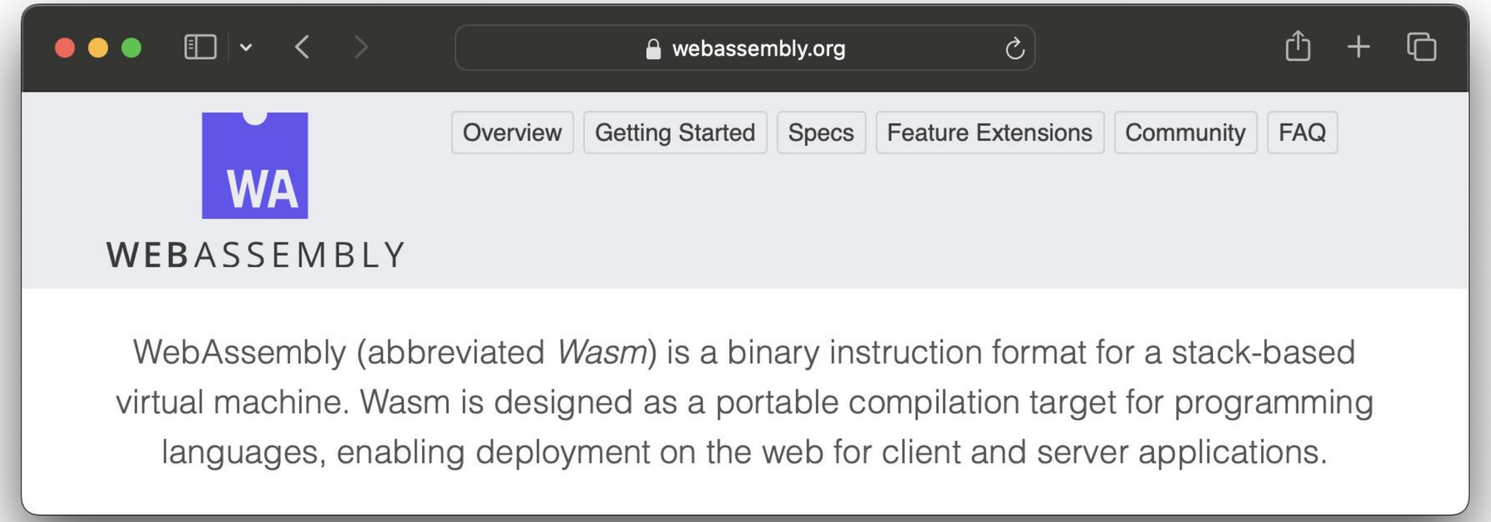
后端

构建工具

WebAssembly

重点特色：

- 可移植性
- 性能佳
- 安全
- 高效

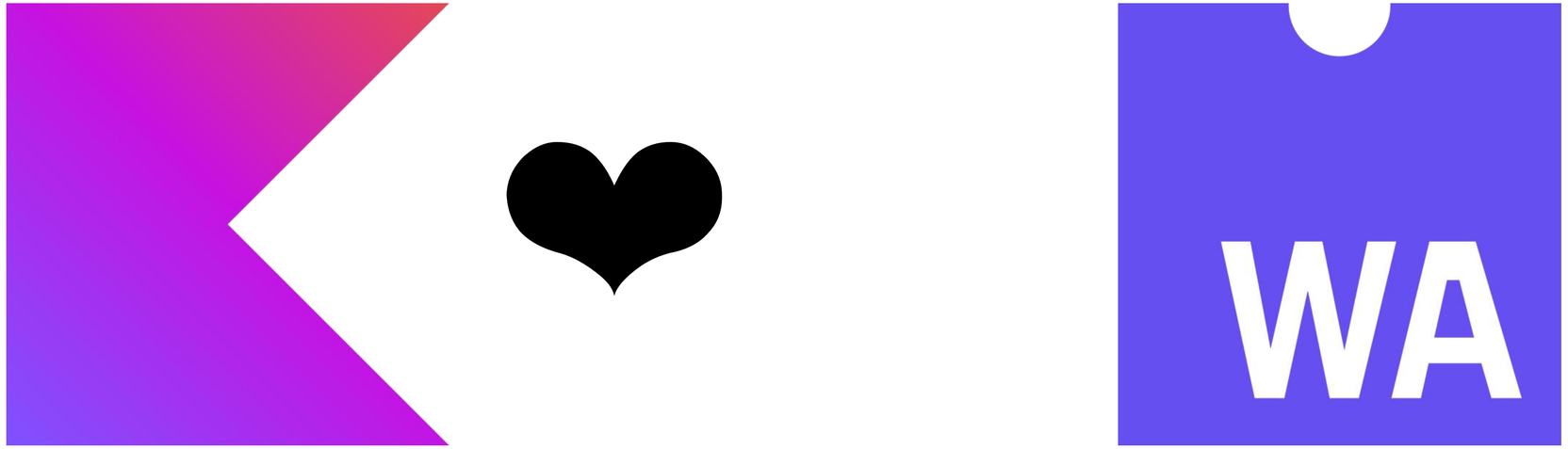


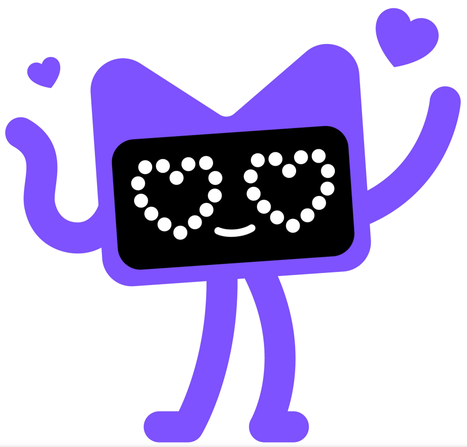
> WebAssembly（缩写为 Wasm）是一种基于堆栈的虚拟机的二进制指令格式。 Wasm 被设计为编程语言的可移植编译目标，支持在网络上部署客户端和服务器应用程序。

WebAssembly 的潜力

随处运行的潜力：

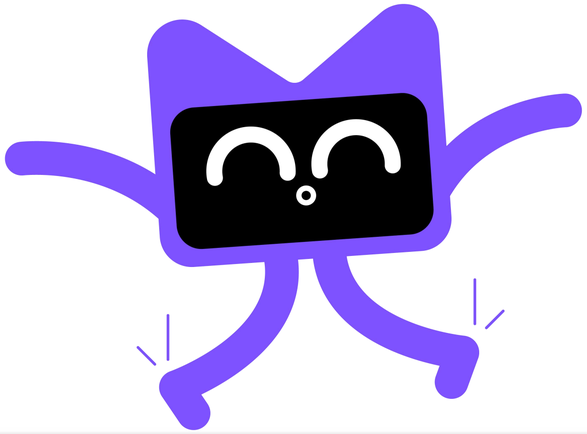
- 浏览器
- 云
- 边缘计算
- 移动端
- 嵌入式系统
- 区块链
- 桌面
- 更多…



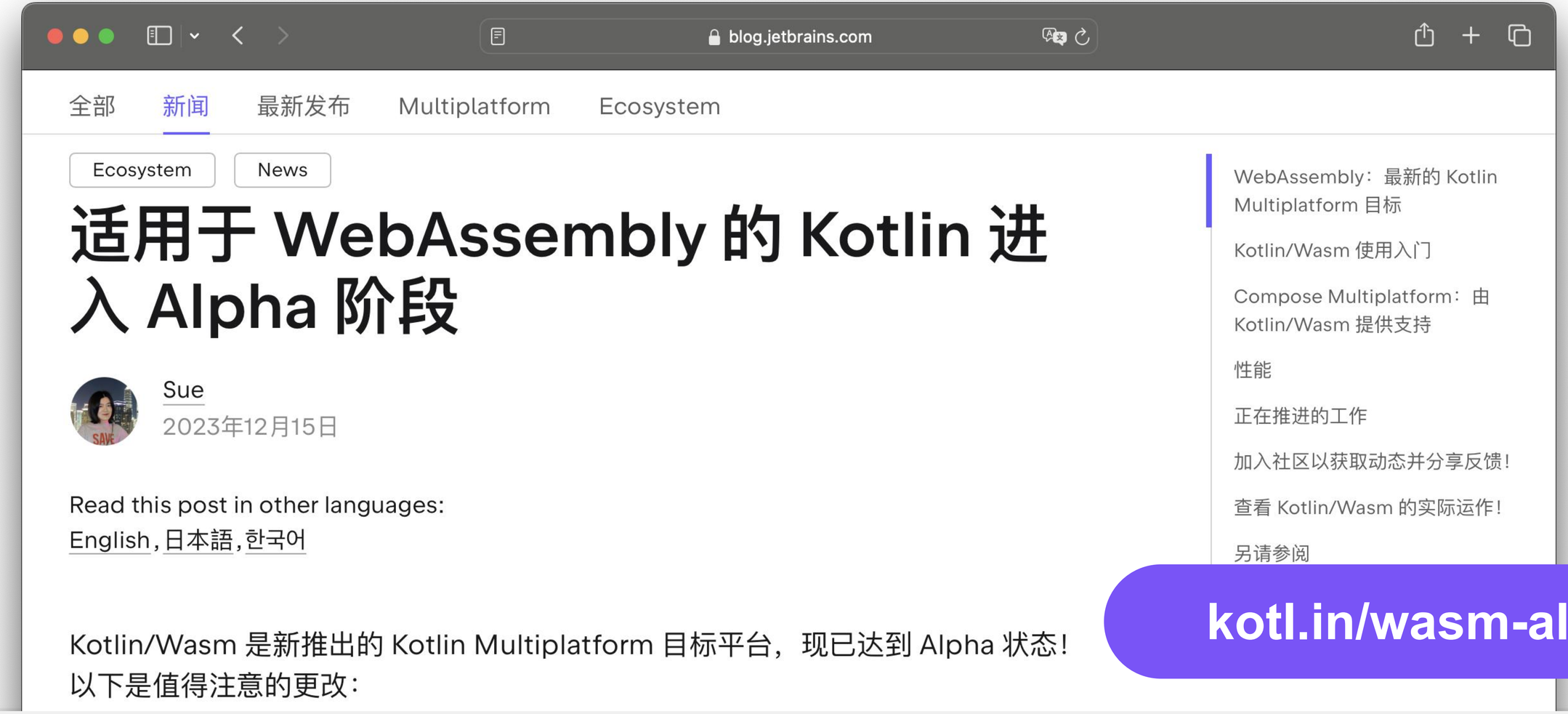


Kotlin/Wasm 项目让 WebAssembly 成为新的 Kotlin 多平台编译目标，并为开发者提供相应的工具链，扩展 Kotlin 在 WebAssembly 方面的潜力！

🚀 Kotlin/Wasm 现已发布 Alpha 版！



Alpha 版发布博文



kotl.in/wasm-alpha

技术演示

- 创建 Kotlin/Wasm 项目
- 操作 HTML/CSS/事件
- 运行 Kotlin/Wasm 程序
- 集成 Compose Multiplatform
- 除错工具
- 发布 Kotlin/Wasm 项目


WA
创建 Kotlin/Wasm 项目

在现有项目里增加 Kotlin/Wasm

```
plugins {
    kotlin("multiplatform") version "..."
}
kotlin {
    wasmJs {
        binaries.executable()
        browser {
            // ...
        }
    }
    sourceSets {
        // ...
        val wasmJsMain by getting {
            dependencies {
                // ...
            }
        }
        val wasmJsTest by getting
    }
}
```

修改 HTML DOM 及 CSS (1/3)

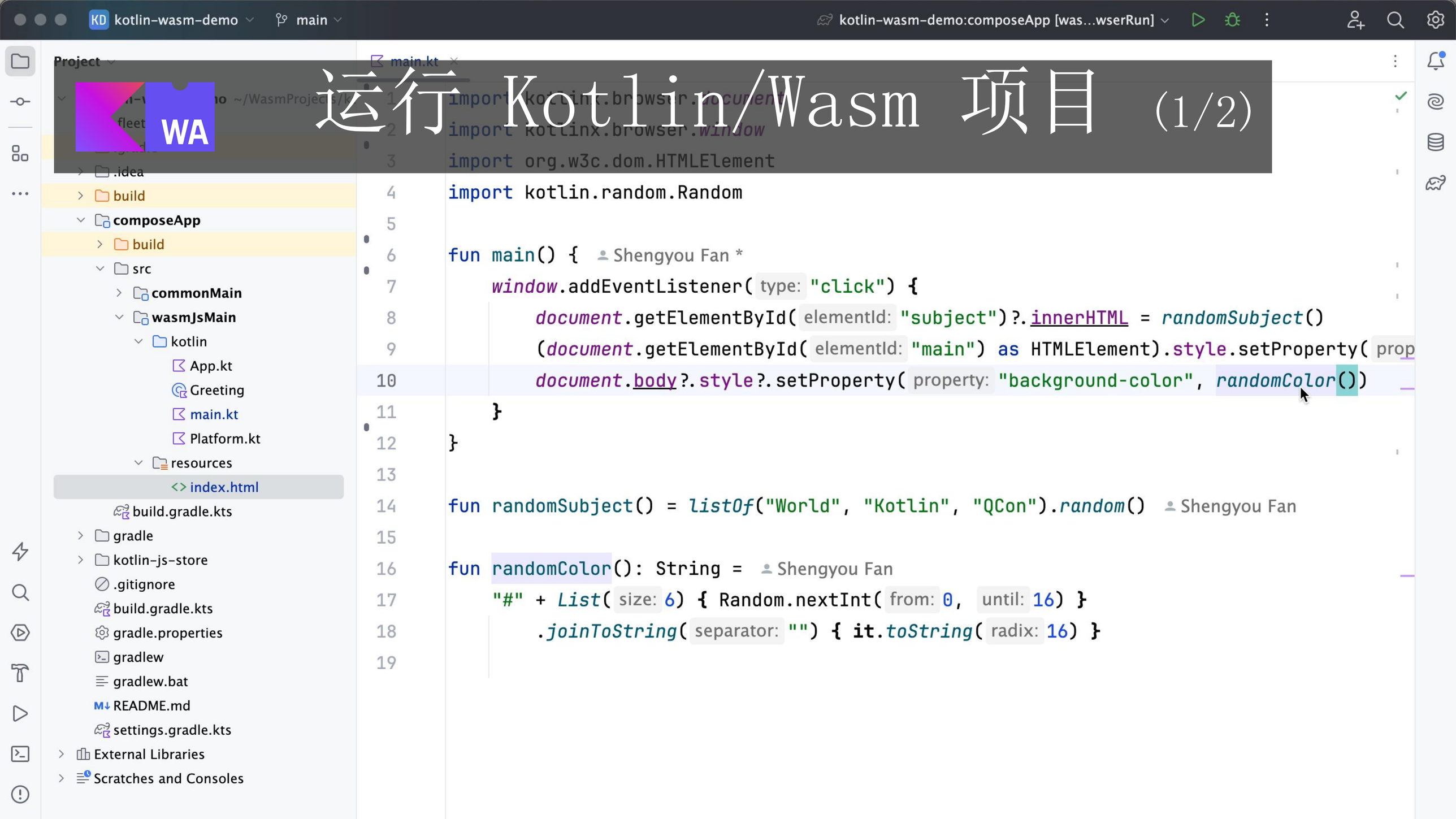
```
// ...
<head>
    // ...
    <script type="application/javascript" src="composeApp.js"></script>
    <style>
        // ...
    </style>
</head>
<body>
<div id="main">
    <p>Hello, <span id="subject">World</span></p>
</div>
</body>
// ...
```

修改 HTML DOM 及 CSS (2/3)

```
import kotlinx.browser.document
import org.w3c.dom.HTMLElement

fun main() {
    document.getElementById("subject")?.innerHTML = /* ... */

    (document.getElementById("main") as HTMLElement)
        .style.setProperty("color", /* ... */)

    document.body?.style?
        .setProperty("background-color", /* ... */)
}
```

绑定事件 (3/3)

```
import kotlinx.browser.document
import org.w3c.dom.HTMLElement

fun main() {
    window.addEventListener("click") {
        document.getElementById("subject")?.innerHTML = /* ... */

        (document.getElementById("main") as HTMLElement)
            .style.setProperty("color", /* ... */)

        document.body?.style?
            .setProperty("background-color", /* ... */)
    }
}
```


kotlin-wasm-demo
main
kotlin-wasm-demo:composeApp [was...wserRun]
Project
main.kt
运行 Kotlin/Wasm 项目 (1/2)
WA
.idea
build
composeApp
build
src
commonMain
wasmJsMain
kotlin
App.kt
Greeting
main.kt
Platform.kt
resources
index.html
build.gradle.kts
gradle
kotlin-js-store
.gitignore
build.gradle.kts
gradle.properties
gradlew
gradlew.bat
README.md
settings.gradle.kts
External Libraries
Scratches and Consoles
import kotlinx.browser.window
import org.w3c.dom.HTMLElement
import kotlin.random.Random
fun main() { Shengyou Fan *
window.addEventListener(type: "click") {
document.getElementById(elementId: "subject")?.innerHTML = randomSubject()
(document.getElementById(elementId: "main") as HTMLElement).style.setProperty(prop
document.body?.style?.setProperty(property: "background-color", randomColor())
}
}
fun randomSubject() = listOf("World", "Kotlin", "QCon").random() Shengyou Fan
fun randomColor(): String = Shengyou Fan
"#" + List(size: 6) { Random.nextInt(from: 0, until: 16) }
.joinToString(separator: "") { it.toString(radix: 16) }

运行 Kotlin/Wasm 项目 (2/2)

Hello, World

Inspector | Console | Debugger | Network | Style Editor | Performance | Memory | Storage | Accessibility | Application

Sources | Outline | Search

composeApp.wasm

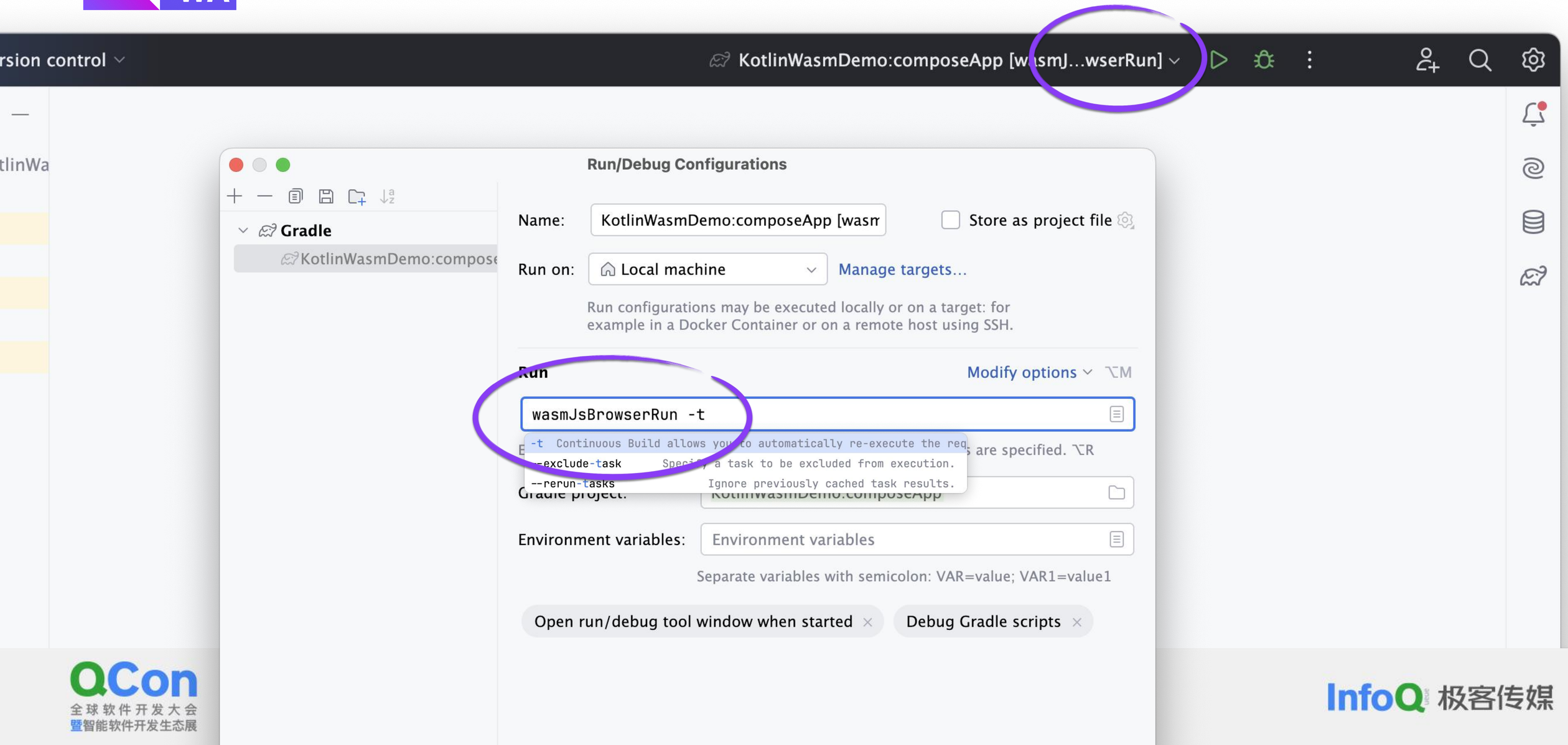
- Main Thread
  - localhost:8080
    - generated/compose/resourceGenerator/kotl
    - Library/Application Support/kotlin/daemon
    - mnt/agent/work
    - src/wasmJsMain/kotlin
    - 3ffe35b7143ad475da5a.wasm
    - composeApp.js
    - composeApp.wasm
    - kotlin_skiko_mjs.js
    - skiko.js
    - skiko.wasm
  - webpack-internal://
  - Webpack

```
00000000 (module
0006569C   (func $unknown0 (import "js_code" "kotlin.captureStackTrace") (result externref))
000656C1   (func $unknown1 (import "js_code" "kotlin.wasm.internal.throwJsError") (param externref externref externref) (resul
000656ED   (func $unknown2 (import "js_code" "kotlin.wasm.internal.stringLength") (param externref) (result i32))
00065719   (func $unknown3 (import "js_code" "kotlin.wasm.internal.jsExportStringToWasm") (param externref i32 i32 i32))
0006574D   (func $unknown4 (import "js_code" "kotlin.wasm.internal.importStringFromWasm") (param i32 i32 externref) (result ex
00065781   (func $unknown5 (import "js_code" "kotlin.wasm.internal.getJsEmptyString") (result externref))
000657B1   (func $unknown6 (import "js_code" "kotlin.wasm.internal.externrefToString") (param externref) (result externref))
000657E2   (func $unknown7 (import "js_code" "kotlin.wasm.internal.externrefEquals") (param externref externref) (result i32))
00065811   (func $unknown8 (import "js_code" "kotlin.wasm.internal.externrefHashCode") (param externref) (result i32))
00065842   (func $unknown9 (import "js_code" "kotlin.wasm.internal.isNullish") (param externref) (result i32))
0006586B   (func $unknown10 (import "js_code" "kotlin.wasm.internal.externrefToInt") (param externref) (result i32))
00065899   (func $unknown11 (import "js_code" "kotlin.wasm.internal.externrefToBoolean") (param externref) (result i32))
000658CB   (func $unknown12 (import "js_code" "kotlin.wasm.internal.externrefToLong") (param externref) (result i64))
000658FA   (func $unknown13 (import "js_code" "kotlin.wasm.internal.externrefToFloat") (param externref) (result f32))
0006592A   (func $unknown14 (import "js_code" "kotlin.wasm.internal.externrefToDouble") (param externref) (result f64))
0006595B   (func $unknown15 (import "js_code" "kotlin.wasm.internal.intToExternref") (param i32) (result externref))
00065989   (func $unknown16 (import "js_code" "kotlin.wasm.internal.getJsTrue") (result externref))
000659B2   (func $unknown17 (import "js_code" "kotlin.wasm.internal.getJsFalse") (result externref))
000659DC   (func $unknown18 (import "js_code" "kotlin.wasm.internal.longToExternref") (param i64) (result externref))
00065A0B   (func $unknown19 (import "js_code" "kotlin.wasm.internal.floatToExternref") (param f32) (result externref))
00065A3B   (func $unknown20 (import "js_code" "kotlin.wasm.internal.doubleToExternref") (param f64) (result externref))
00065A6C   (func $unknown21 (import "js_code" "kotlin.wasm.internal.newJsArray") (result externref))
00065A96   (func $unknown22 (import "js_code" "kotlin.wasm.internal.jsArrayPush") (param externref externref))
```

{ } (1, 1)

- Watch expressions
- Breakpoints
  - [x] Pause on debugger statement
  - [ ] Pause on exceptions
- XHR Breakpoints
- Event Listener Breakpoints Log
- DOM Mutation Breakpoints

持续构建



集成 Compose Multiplatform

```
<body>
    <canvas id="ComposeTarget">
    </canvas>
</body>
```

```
import androidx.compose.ui.ExperimentalComposeUiApi
import androidx.compose.ui.window.CanvasBasedWindow

@OptIn(ExperimentalComposeUiApi::class)
fun main() {
    CanvasBasedWindow(canvasElementId = "ComposeTarget") {
        /* ... */
    }
}
```

Kotlin 多平台开发 (Android)

compose-jetsnack | main | Pixel 3a API 31 | android

Project

JetsnackApp.kt

Running Devices | Pixel 3a API 31

```
17  package com.example.jetsnack.ui
18
19  import ...
28
    Oleksandr Karpovich
29  @Composable
30  fun JetsnackApp() {
31      JetsnackTheme {
32          val appState = rememberMppJetsnackApp
33          JetsnackScaffold(
34              bottomBar = {
35                  if (appState.shouldShowBottom
36                      JetsnackBottomBar(
37                          tabs = appState.botto
38                          currentRoute = appSta
39                          navigateToRoute = app
40                      )
41                  }
42              },
43              snackbarHost = { it: SnackbarHostState
44                  SnackbarHost(
```

Kotlin 多平台开发

(Desktop)

compose-jetsnack | main | compose-jetsnack:desktop [run]

Project

- desktop
  - build
  - src
    - jvmMain
      - kotlin
        - Main.kt
  - build.gradle.kts
- gradle
- kotlin-js-store
- screenshots
- web
- .gitignore
- build.gradle.kts
- gradle.properties
- gradlew
- gradlew.bat
- local.properties
- README.md
- settings.gradle.kts
- External Libraries
- Scratches and Consoles

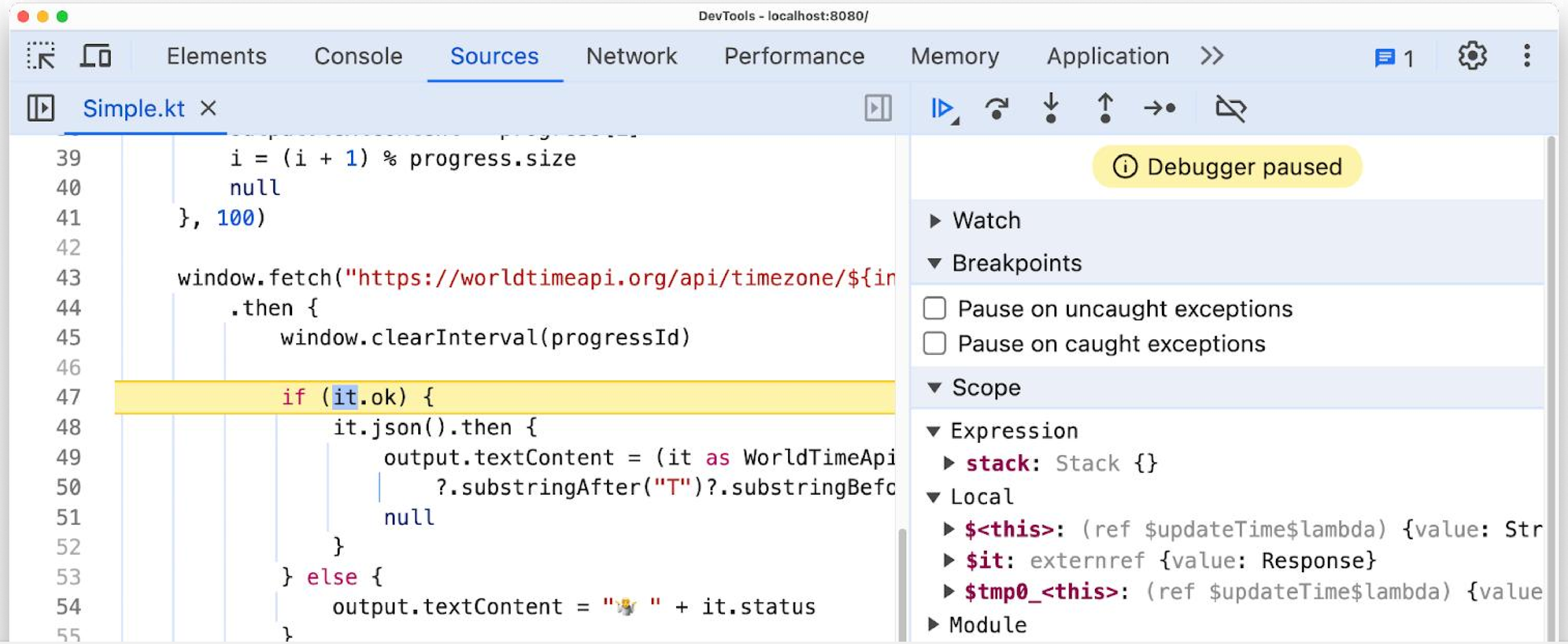
Main.kt

```
1  import androidx.compose.ui.unit.dp
2  import androidx.compose.ui.window.Window
3  import androidx.compose.ui.window.application
4  import androidx.compose.ui.window.rememberWindowState
5  import com.example.jetsnack.JetSnackAppEntryPoint
6
7
8  fun main() = application {  Oleksandr Karpovich
9      Window(
10         onCloseRequest = ::exitApplication,
11         state = rememberWindowState(width = 1200.dp, height = 960.dp)
12     ) {
13         JetSnackAppEntryPoint()
14     }
15 }
16
```

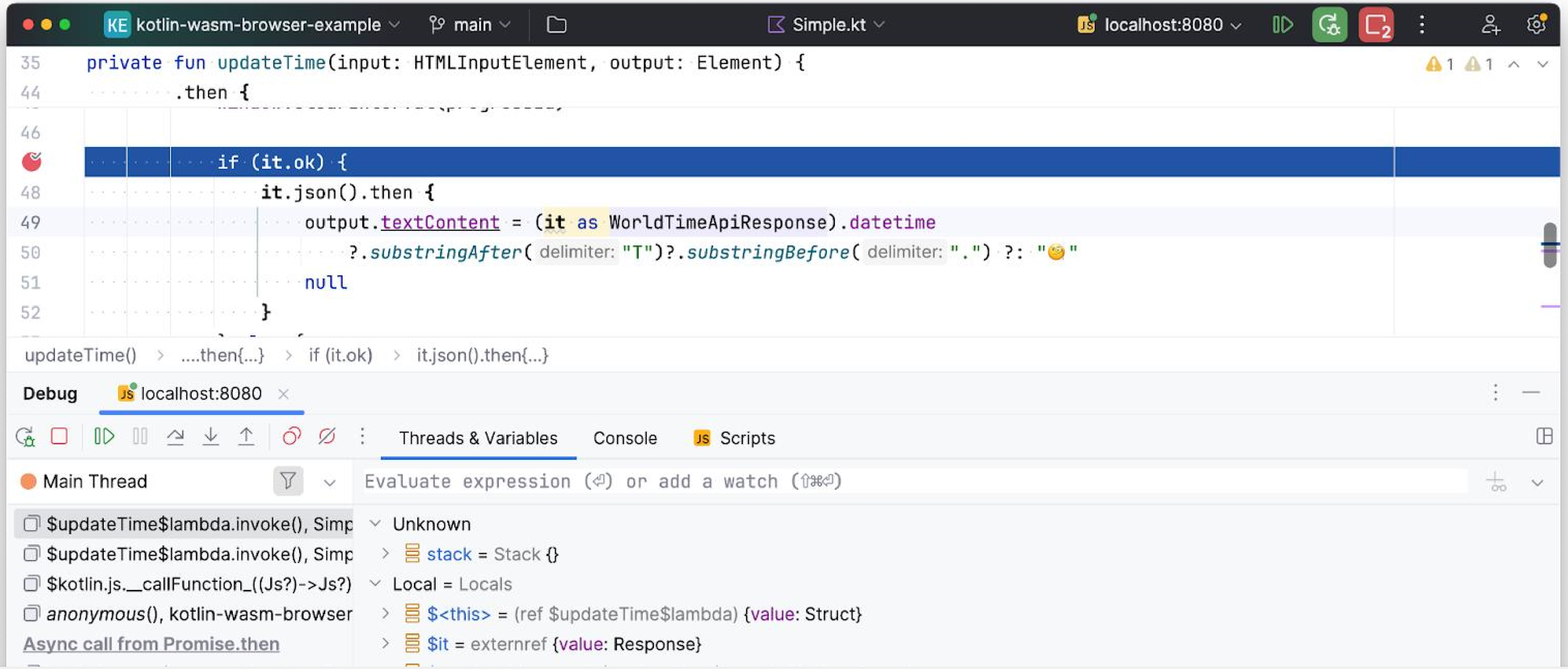
Kotlin 多平台开发（Wasm）

```
@OptIn(ExperimentalComposeLibApi::class, ExperimentalResourceApi::class)
fun main() {
    configureWebResources {
        // same as default - this is not necessary to add here. It's here to show this
        resourcePathMapping { path → "./$path" }
    }
    CanvasBasedWindow( title: "JetSnack", canvasElementId = "jetsnackCanvas") {
        var loading: Boolean by remember { mutableStateOf( value: true) }

        if (loading) {
            LinearProgressIndicator(modifier = Modifier.fillMaxWidth())
        } else {
            JetSnackAppEntryPoint()
        }

        LaunchedEffect(Unit) {
            loadMontserratFont()
            loadKarlaFont()
            prepareImagesCache()
            loading = false
        }
    }
}
```

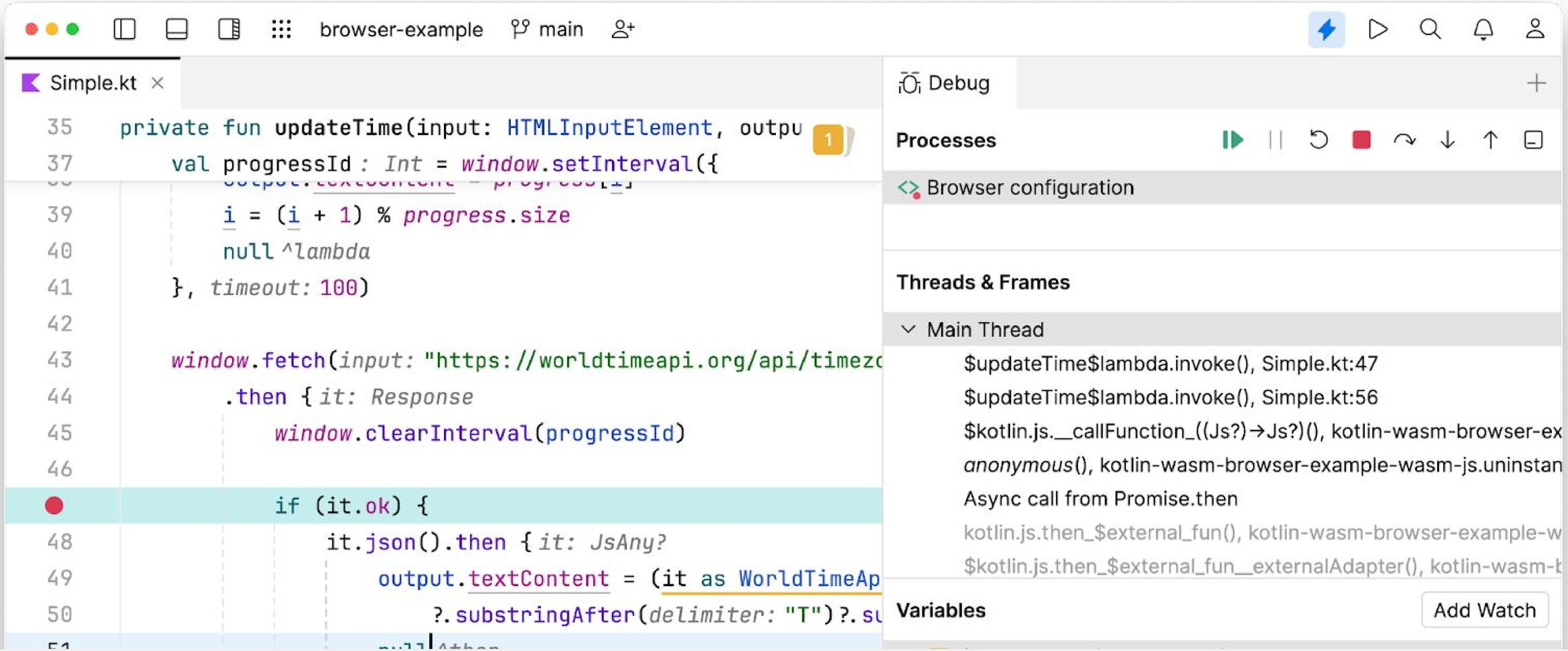
除错工具 (1/3)

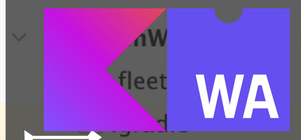

DevTools - localhost:8080/
Elements
Console
Sources
Network
Performance
Memory
Application
Simple.kt
39 i = (i + 1) % progress.size
40 null
41 }, 100)
42
43 window.fetch("https://worldtimeapi.org/api/timezone/${in
44 .then {
45 window.clearInterval(progressId)
46
47 if (it.ok) {
48 it.json().then {
49 output.textContent = (it as WorldTimeApi
50 ?.substringAfter("T")?.substringBefo
51 null
52 }
53 } else {
54 output.textContent = "" + it.status
Debugger paused
Watch
Breakpoints
Pause on uncaught exceptions
Pause on caught exceptions
Scope
Expression
stack: Stack {}
Local
$<this>: (ref $updateTime$lambda) {value: Str
$it: externref {value: Response}
$tmp0_<this>: (ref $updateTime$lambda) {value
Module

除错工具 (2/3)


kotlin-wasm-browser-example
main
Simple.kt
localhost:8080
35 private fun updateTime(input: HTMLInputElement, output: Element) {
44 .then {
46
if (it.ok) {
48 it.json().then {
49 output.textContent = (it as WorldTimeApiResponse).datetime
50 ?.substringAfter(delimiter: "T")?.substringBefore(delimiter: ".") ?: "🧐"
51 null
52 }
updateTime() › ....then{...} › if (it.ok) › it.json().then{...}
Debug
localhost:8080
Threads & Variables
Console
Scripts
Main Thread
Evaluate expression (⏎) or add a watch (⇧⌘⏎)
$updateTime$lambda.invoke(), Simp
$updateTime$lambda.invoke(), Simp
$kotlin.js.__callFunction_((Js?)->Js?)
anonymous(), kotlin-wasm-browser
Async call from Promise.then
Unknown
stack = Stack {}
Local = Locals
$<this> = (ref $updateTime$lambda) {value: Struct}
$it = externref {value: Response}

除错工具 (3/3)


browser-example
main
Simple.kt
35 private fun updateTime(input: HTMLInputElement, outpu
37 val progressId : Int = window.setInterval({
39 i = (i + 1) % progress.size
40 null ^lambda
41 }, timeout: 100)
42
43 window.fetch(input: "https://worldtimeapi.org/api/timezo
44 .then { it: Response
45 window.clearInterval(progressId)
46
48 if (it.ok) {
49 it.json().then { it: JsAny?
50 output.textContent = (it as WorldTimeAp
?.substringAfter(delimiter: "T")?.su
Debug
Processes
Browser configuration
Threads & Frames
Main Thread
$updateTime$lambda.invoke(), Simple.kt:47
$updateTime$lambda.invoke(), Simple.kt:56
$kotlin.js.__callFunction_((Js?)→Js?)(), kotlin-wasm-browser-ex
anonymous(), kotlin-wasm-browser-example-wasm-js.uninstan
Async call from Promise.then
kotlin.js.then_$external_fun(), kotlin-wasm-browser-example-w
$kotlin.js.then_$external_fun__externalAdapter(), kotlin-wasm-b
Variables
Add Watch



发布 Kotlin/Wasm 项目

KotlinWasmDemo Version control Current File

Project

~/Downloads/KotlinWasmDemo

- .idea
- composeApp
  - src
    - commonMain
    - wasmJsMain
      - kotlin
        - App.kt
        - Greeting
        - main.kt
        - Platform.kt
      - resources
    - build.gradle.kts
  - gradle
  - kotlin-js-store
  - .gitignore
  - build.gradle.kts
  - gradle.properties
  - gradlew
  - gradlew.bat
  - README.md
  - settings.gradle.kts
- External Libraries
- Scratches and Consoles

main.kt

```
1  import androidx.compose.ui.ExperimentalComposeUiApi
2  import androidx.compose.ui.window.CanvasBasedWindow
3
4  @OptIn(ExperimentalComposeUiApi::class)
5  fun main() {
6      CanvasBasedWindow(canvasElementId = "ComposeTarget") { App() }
7  }
```

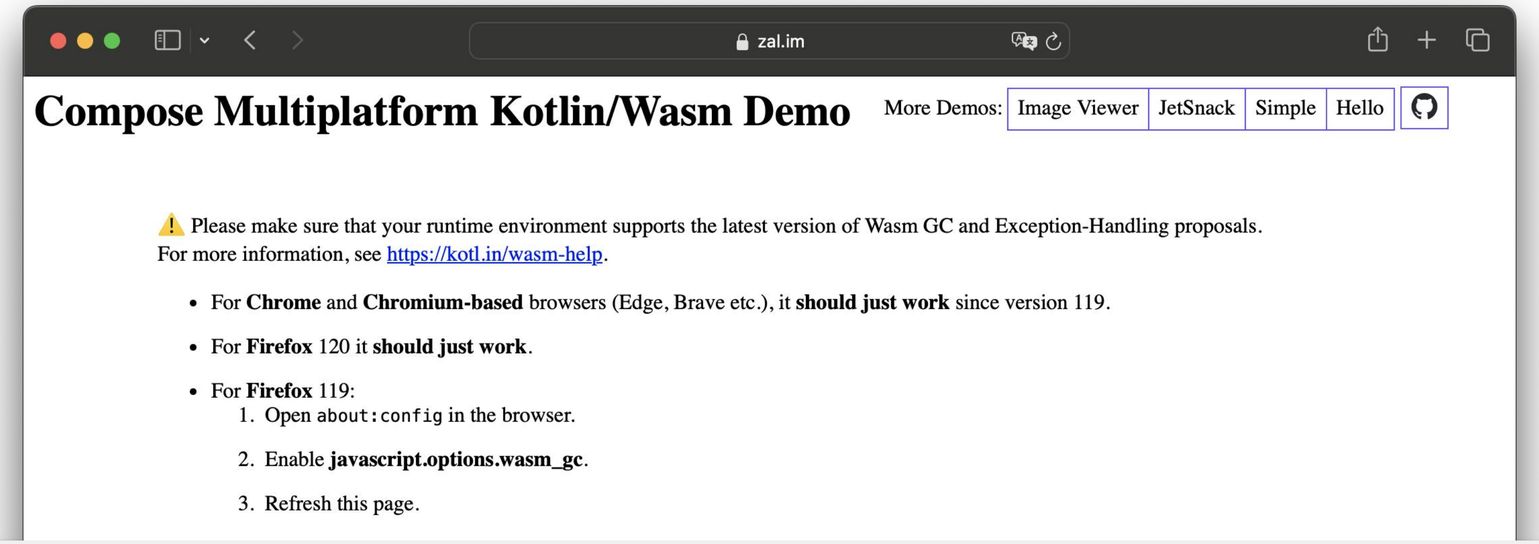
发展现状及成果

- 浏览器支持
- 体积
- 性能
- 工作成果概览

浏览器支持

大多数主流浏览器已默认支持：

✅ ✅ 🏗️



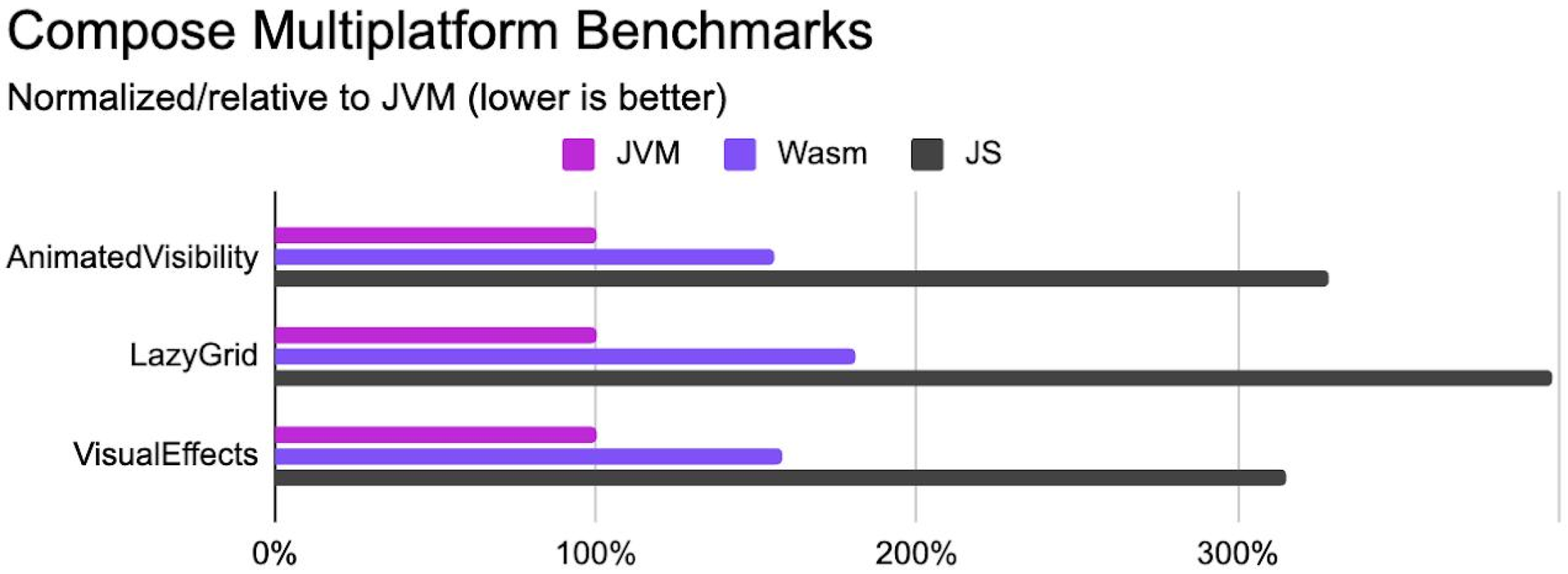
体积

```
import kotlinx.browser.document
import kotlinx.dom.appendText

fun main() {
    val p = document.createElement("p")
    p.innerHTML = "Hello World!"
    document.body?.appendChild(p)
}
```

3K

使用 Kotlin/Wasm 1.9 编写的
“Hello World”程序的 Wasm 体积

性能



kotl.in/wasm-benchmarks

2023 年工作成果

- 支持最新的 Wasm GC
- 支持 WASI API
- 提升库的支持
- 与 VM Vendor 合作
- 优化开发体验

展望未来

- 浏览器之外
- 优势
- 适用场景
- 当前限制

浏览器之外的 Kotlin/Wasm (1/2)

支持 Wasm 的 JS 虚拟机：

- ☑ Deno 1.38+
- SOON Node.js 22


kotlin-wasm-nodejs-example
main
nodejs-example [wasmJsNodeRun]
Project
build.gradle.kts (kotlin-wasm-nodejs-example)
Main.kt
Gradle
Kotlin/Wasm 支持 JS VMs
WA
build
gradle
kotlin-js-store
screenshots
src
wasmJsMain
kotlin
Main.kt
.gitignore
build.gradle.kts
gradle.properties
gradlew
gradlew.bat
README.md
settings.gradle.kts
External Libraries
Scratches and Consoles
mavenCentral()
}
kotlin {
wasmJs {
binaries.executable()
nodejs()
}
}
rootProject.the<NodeJsRootExtension>().apply {
nodeVersion = "22.0.0-nightly202404032241e8c5b3"
nodeDownloadBaseUrl = "https://nodejs.org/download/night
}
tasks.withType<org.jetbrains.kotlin.gradle.targets.js.npm.ta
args.add("--ignore-engines")
}
kotlin-wasm-nodejs-example
Tasks
build
build setup
help
ide
kotlin node
wasmJsNodeDevelopmentRun
wasmJsNodeProductionRun
wasmJsNodeRun
nodejs
other
verification
Run Configurations

浏览器之外的 Kotlin/Wasm (2/2)

可独立运行的 Wasm 虚拟机：

- SOON WasmEdge 0.14.0-alpha.3+*
- Wasmtime
- WAMR

WASI

WASI = WebAssembly System Interface

- ☑ WASI Preview 1 (Kotlin 1.9.20 开箱即用)
-  Component Model
-  WASI 0.2 (aka Preview 2)


kotlin-wasm-wasi-example
main
wasi-example [wasmWasiDenoRun]
Project
MonotonicTime.kt
Kotlin/Wasm 支持 WASI
WA
build
gradle
screenshots
src
wasmWasiMain
kotlin
MonotonicTime.kt
wasmWasiTest
kotlin
Test.kt
.gitignore
build.gradle.kts
gradle.properties
gradlew
gradlew.bat
README.md
settings.gradle.kts
External Libraries
Scratches and Consoles
import kotlin.wasm.unsafe.Pointer
import kotlin.wasm.unsafe.UnsafeWasmMemoryApi
import kotlin.wasm.unsafe.withScopedMemoryAllocator
fun main() { Zalim Bashorov
println("Hello from Kotlin via WASI")
println("Current 'realtime' timestamp is: ${wasiRealTime()}")
println("Current 'monotonic' timestamp is: ${wasiMonotonicTime()}")
}
@WasmImport(module: "wasi_snapshot_preview1", name: "clock_time_get") Igor Yakovlev
private external fun wasiRawClockTimeGet(clockId: Int, precision: Long, resultPtr: Int): Int
private const val REALTIME = 0
private const val MONOTONIC = 1
@OptIn(UnsafeWasmMemoryApi::class) Igor Yakovlev +1
fun wasiGetTime(clockId: Int): Long = withScopedMemoryAllocator { allocator →
val rp0 = allocator.allocate(size: 8)
val ret = wasiRawClockTimeGet(
clockId = clockId,
precision = 1,

Kotlin/Wasm 的优势

- 简洁的语法
- 安全
- 性能佳
- 强大的开发工具链
  - IDE
  - Debugger
  - Build tools

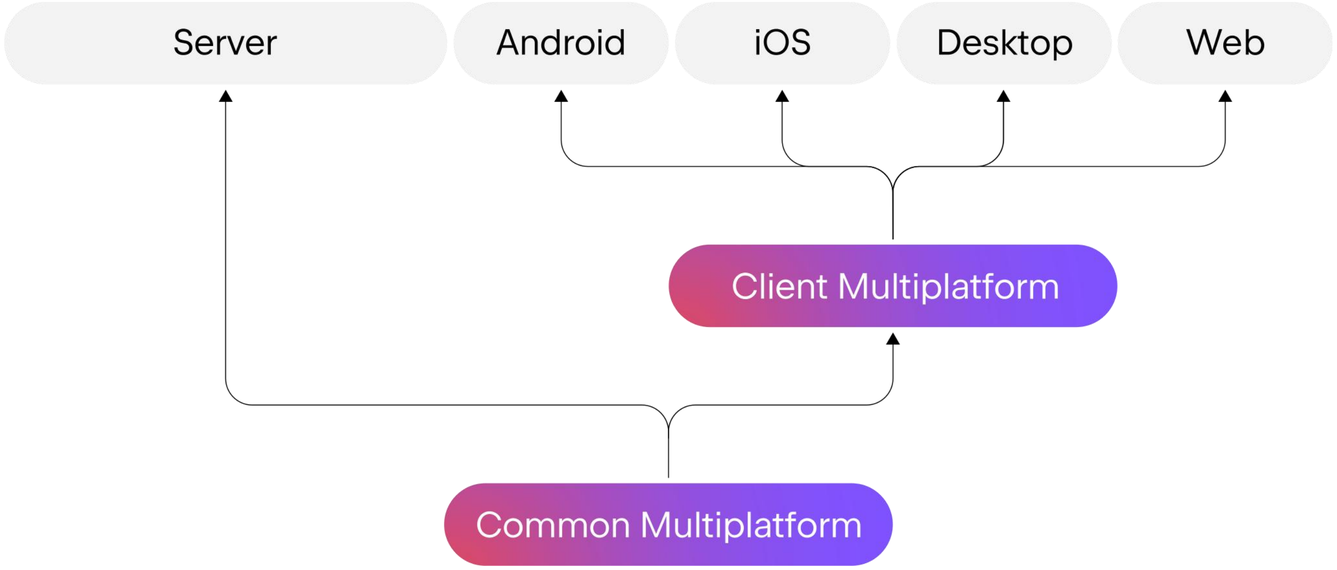
- Kotlin 多平台开发及生态
  - Compose Multiplatform
  - Ktor
  - kotlinx-{coroutine, serialization, io, datetime, …}

Kotlin/Wasm 适用场景

- 使用 Compose Multiplatform 开发 Web 应用
- Serverless (Cloud Functions、Lambda、Worker 等)
- 边缘计算

当前限制

- 相关技术仍在早期发展阶段*
- 需要多平台库作者的支持



*Stability levels explained: https://kotlinlang.org/docs/components-stability.html#stability-levels-explained

持续成长

- ~1K 每周活跃用户 (IDE 使用量)
- 1100+ 话题信息量 (官方 Kotlin Slack 频道)
- Kotlin Slack (**#webassembly** 及 **#compose-web** 频道) 及社交媒体上的活跃动态
- ~600 库 (Maven Central)

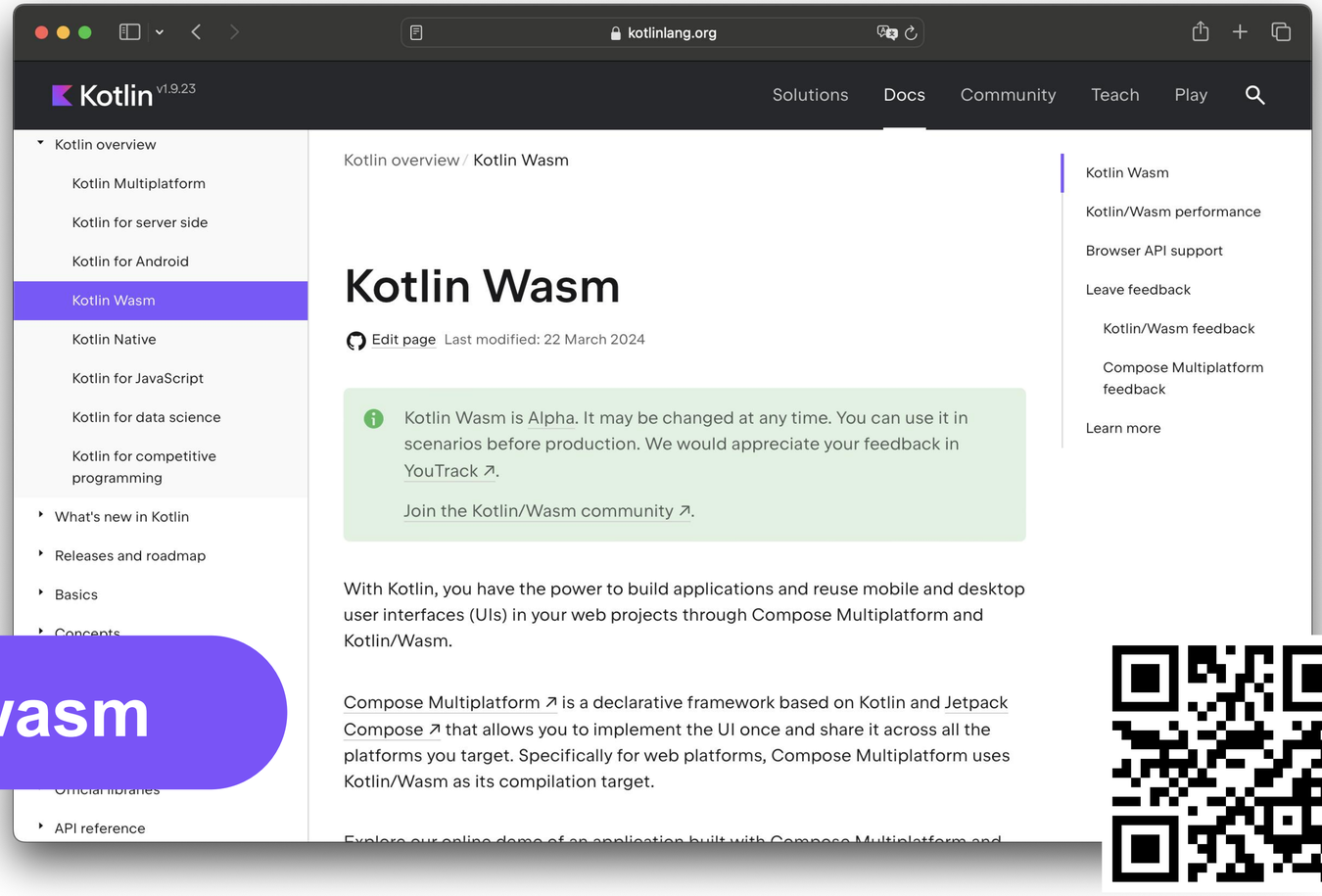
学习材料

- 文档
- 视频
- 示例
- 社区
- 官方帐号

Kotlin/Wasm 学习材料

(1/4)

- 官网文档

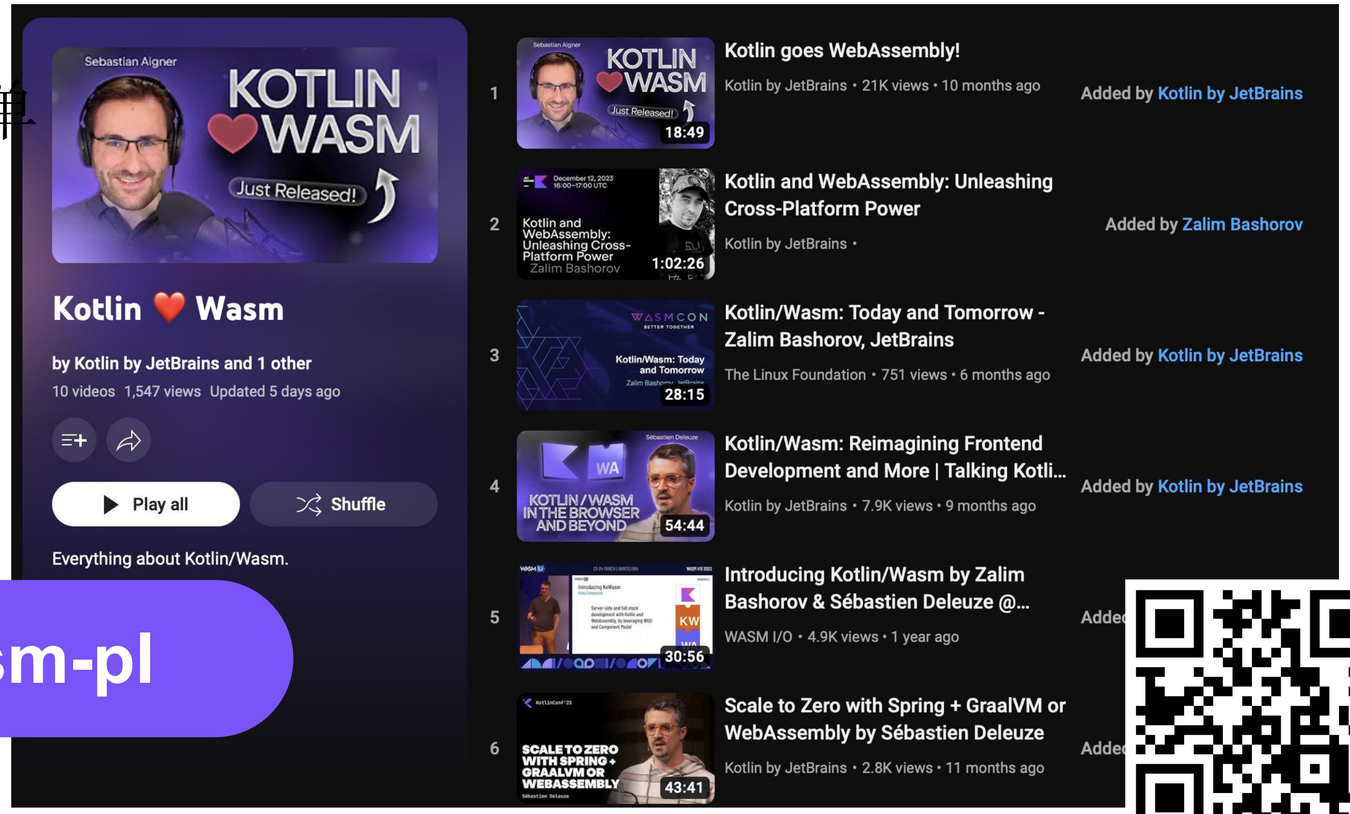


kotl.in/wasm

Kotlin/Wasm 学习材料

(2/4)

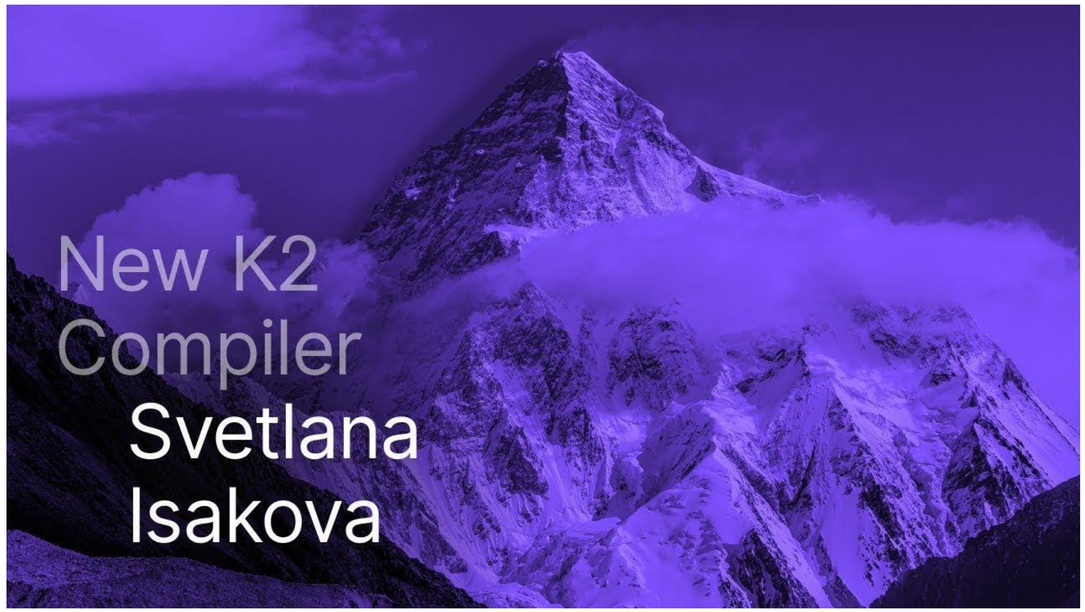
- 官网文档
- Kotlin ❤ Wasm 视频播放清单



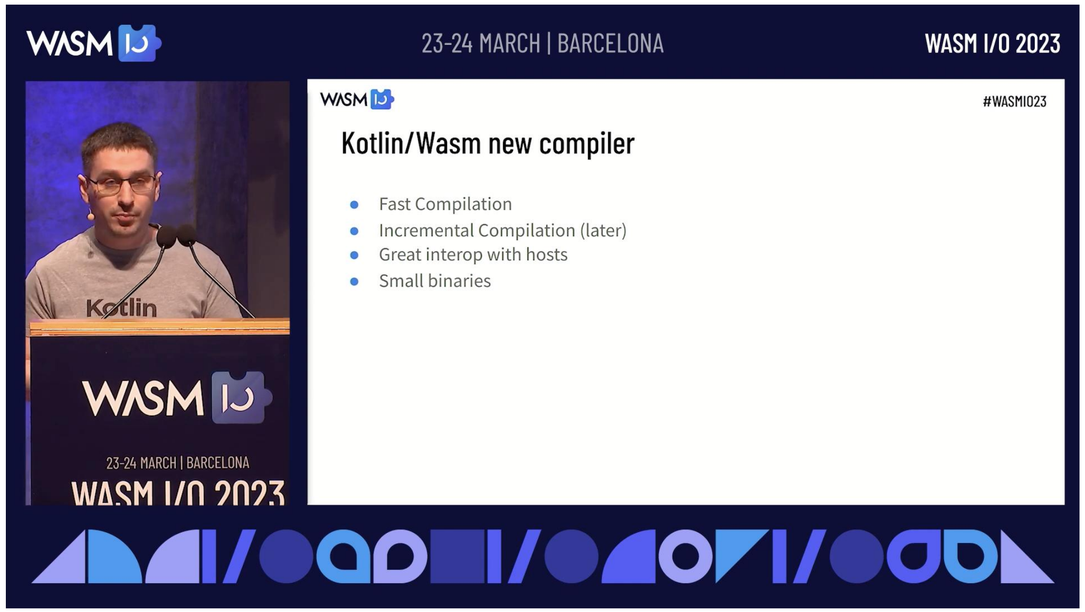
kotl.in/wasm-pl

Kotlin 编译器

讲解



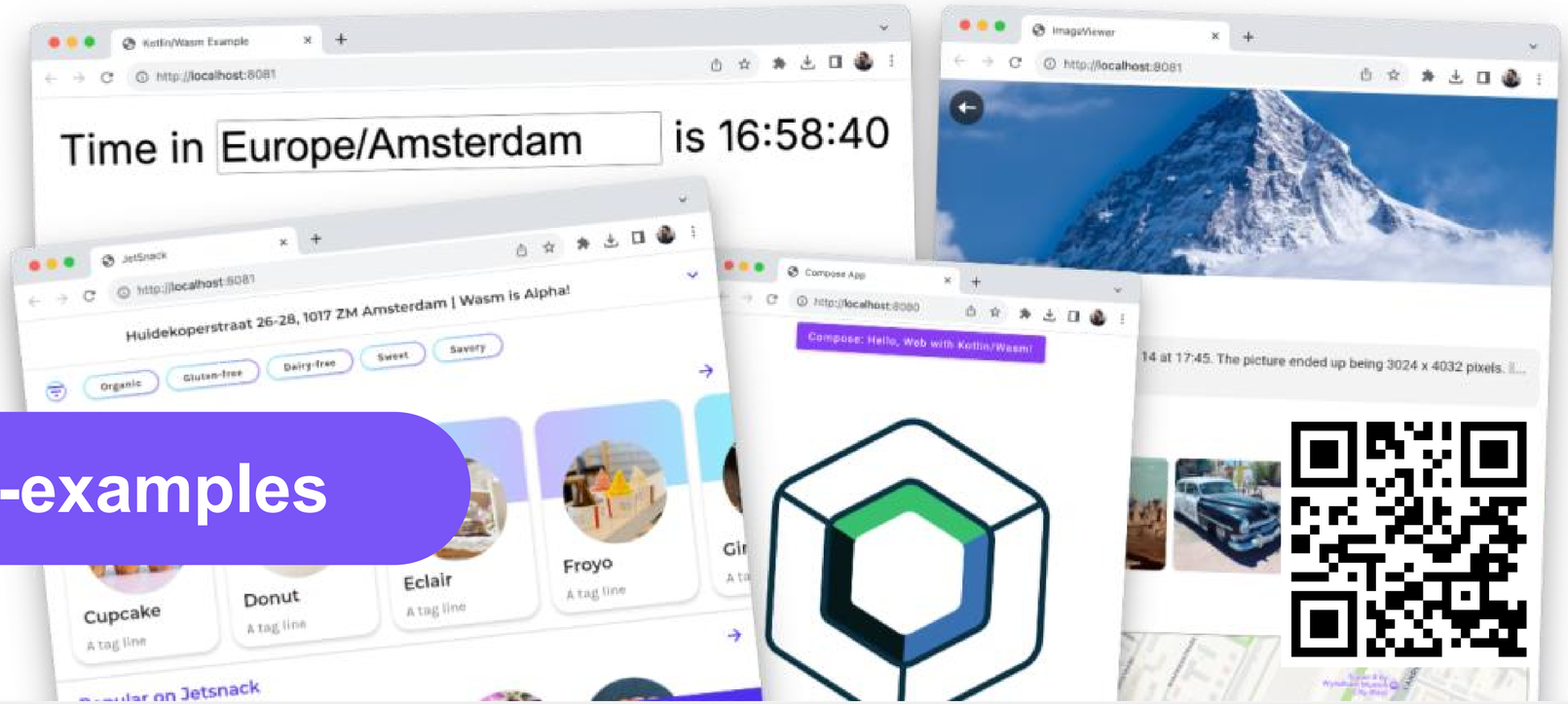
K2 编译器之路



团队在 WASM I/O 2023 的发表视频

Kotlin/Wasm 学习材料 (3/4)

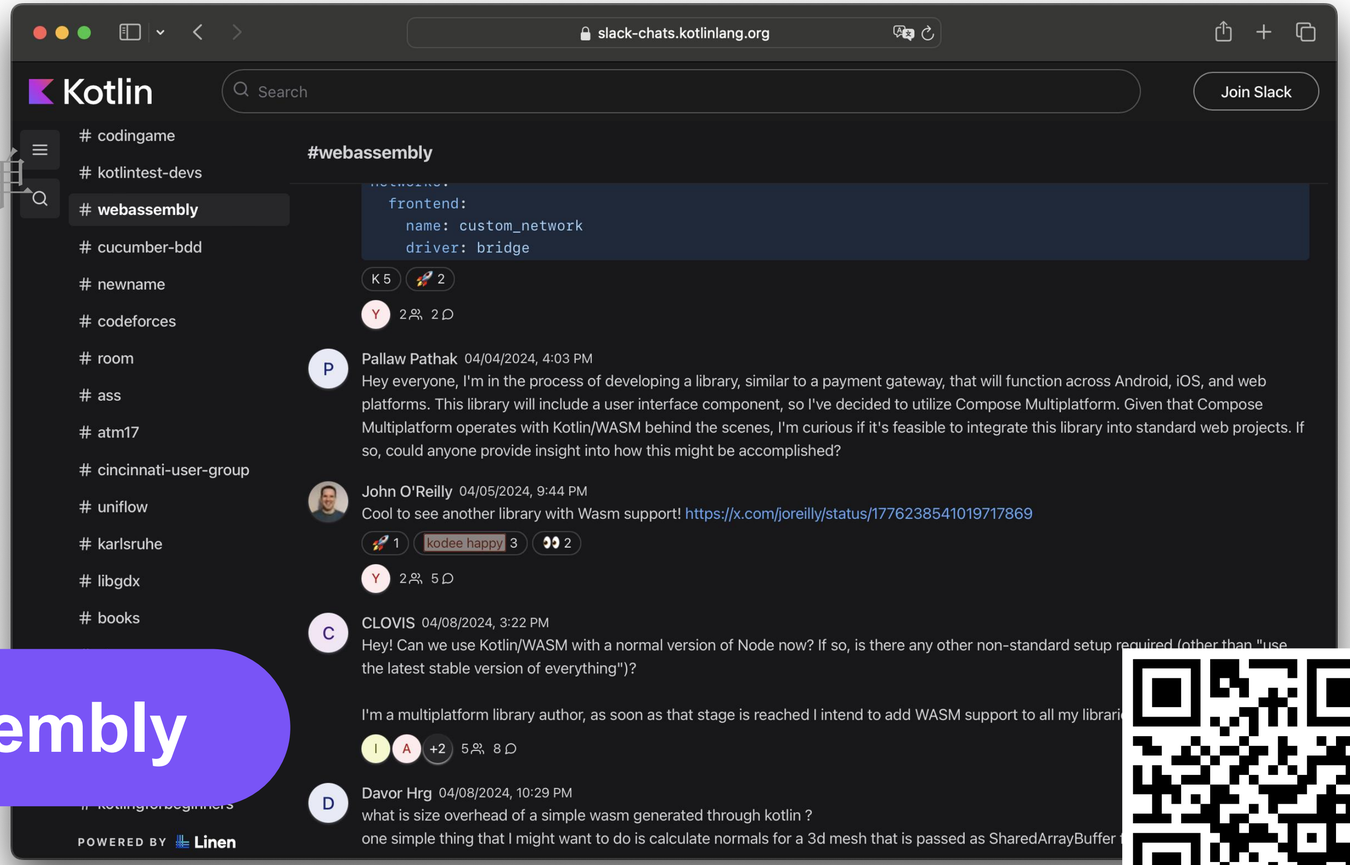
- 官网文档
- Kotlin ❤ Wasm 视频播放清单
- Kotlin/Wasm 官方示例



kotl.in/wasm-examples

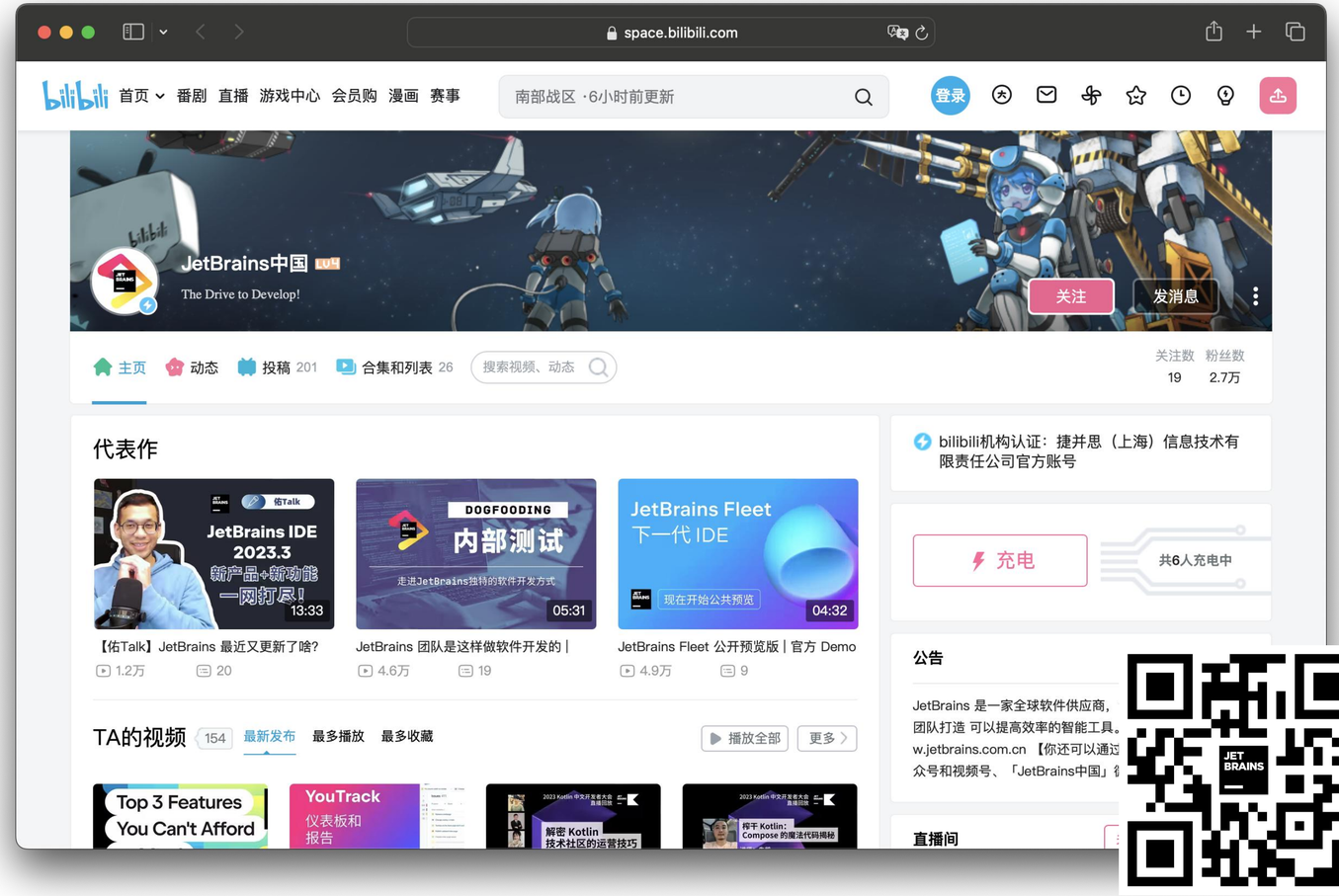
Kotlin/Wasm 学习材料 (4/4)

- 官网文档
- Kotlin ❤ Wasm 视频播放清单
- Kotlin/Wasm 官方示例
- Kotlin Slack 频道



官方帐号 (1/2)

- JetBrains B 站帐号



官方帐号 (2/2)

- JetBrains B 站帐号
- JetBrains 微信公众号
- Kotlin 开发者微信公众号

JetBrains

微信扫描二维码，关注我的公众号

Kotlin开发者

微信扫描二维码，关注我的公众号

Kotlin 炉边漫谈播客

#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

✍ 欢迎投稿你的 Kotlin 用例

InfoQ 极客传媒

已结束 北京站 2000 人

QCon 全球软件开发大会 暨智能软件开发生态展

架构演进 | AIGC
研发效能 | 向量数据库

6 月 14–15 日 深圳站 1000 人

ArchSummit 全球架构师峰会 暨智能软件开发生态展

业务架构 | 高可用
AI 架构 | 架构师成长

10 月 18–19 日 上海站 1500 人

QCon 全球软件开发大会 暨智能软件开发生态展

AIGC | AI 原生应用
软件工程 | 金融科技

4月 5月 6月 8月 10月 12月

5 月 17–18 日 北京站 2000 人

AiCon 全球人工智能开发与应用大会 暨大模型应用生态展

通用大模型 | AI Agent
LLMOps | AI Infra

8 月 16–17 日 上海站 1000 人

FCon 全球金融科技大会 暨智能软件开发生态展

金融大模型 | 核心系统
数字化转型 | 数据安全

12 月 20–21 日 北京站 1000 人

ArchSummit 全球架构师峰会 暨智能软件开发生态展

未来架构 | 高可用高并发
AI 应用 | 架构师思维

极客邦科技 2024 年会议规划

促进软件开发及相关领域知识与创新的传播

访问大会官网

参会咨询

THANKS

范圣佑 (Shengyou Fan)
JetBrains Developer Advocate
shengyou.fan@jetbrains.com

大模型正在重新定义软件

Large Language Model Is Redefining The Software