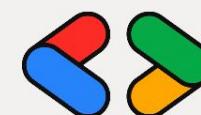
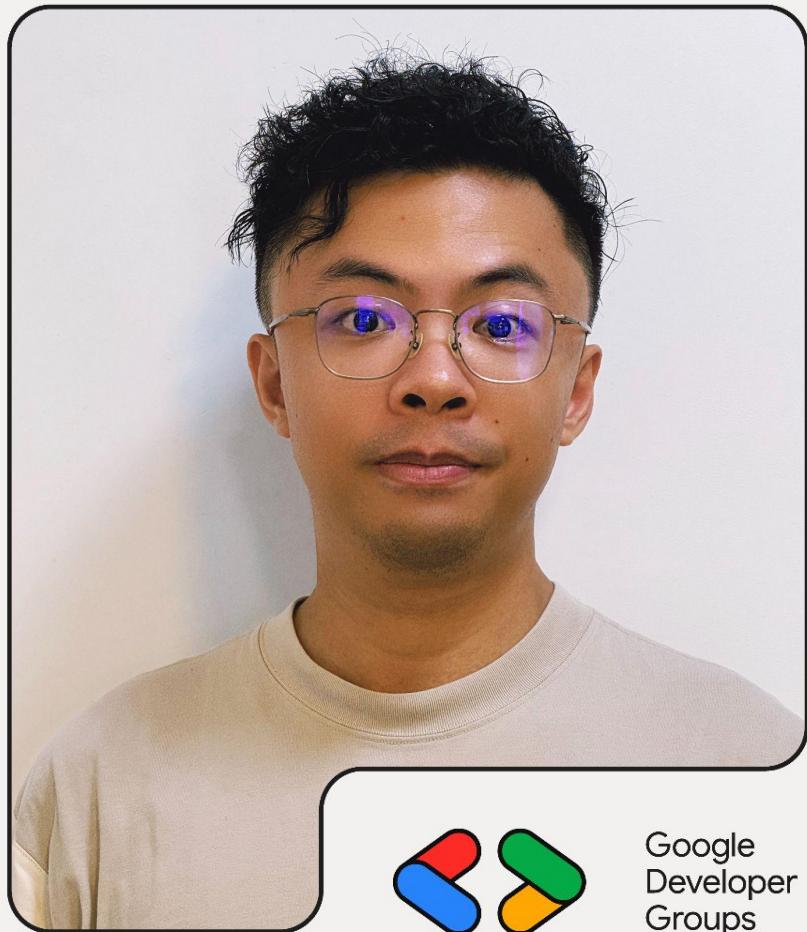




Adapting the MediaPipe Demos for Kotlin Multiplatform

EI

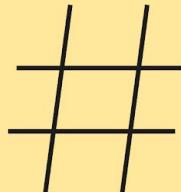


Google
Developer
Groups

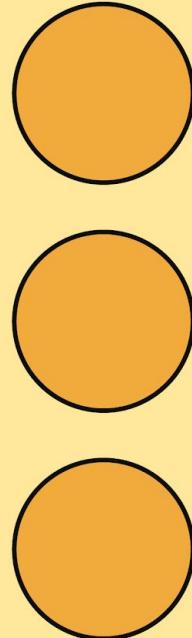
- 
1. On Device Model + KMP
 2. 將 Android 專案遷移到 KMP
 3. 換個角度看 KMP
- 

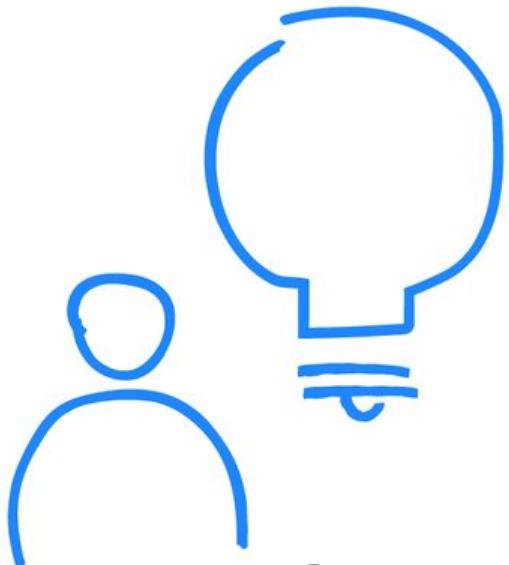
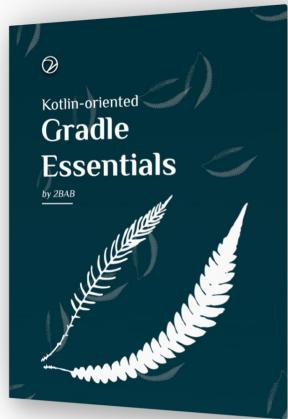
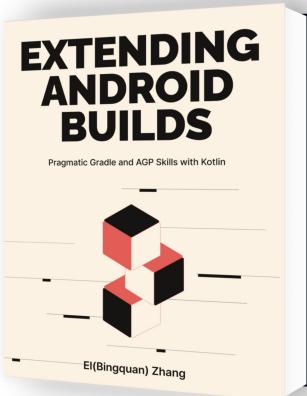


Google
Developer
Groups

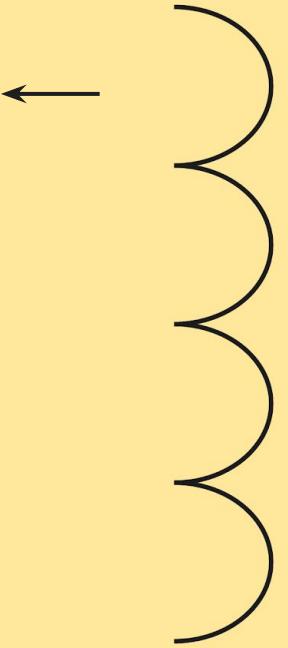


Mobile
@DevFest



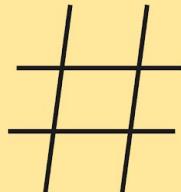


DevFest
Taipei

- 
1. On Device Model + KMP
 2. 將 Android 專案移植到 KMP
 3. 換個角度看 KMP
- 



Google
Developer
Groups



Mobile
@DevFest

Google x On-Device Model

(只考慮行動設備)

MediaPipe Solutions provides a suite of libraries to **quickly** apply **AI and machine learning (ML)** techniques in your applications.

<https://mediapipe-studio.webapps.google.com/home>

Solution	Android	Web	Python	iOS	Customize model
LLM Inference API	●	●	●	●	●
Object detection	●	●	●	●	●
Image classification	●	●	●	●	●
Image segmentation	●	●	●		
Interactive segmentation	●	●	●		
Hand landmark detection	●	●	●	●	
Gesture recognition	●	●	●	●	●
Image embedding	●	●	●		
Face detection	●	●	●	●	
Face landmark detection	●	●	●		
Face stylization	●	●	●		●
Pose landmark detection	●	●	●		
Image generation	●				●
Text classification	●	●	●	●	●
Text embedding	●	●	●		
Language detector	●	●	●		
Audio classification	●	●	●		



Google Developer Groups

Google x On-Device Model

(只考慮行動設備)

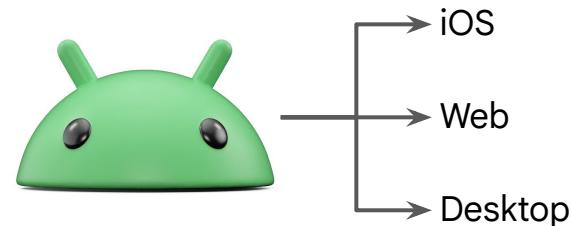
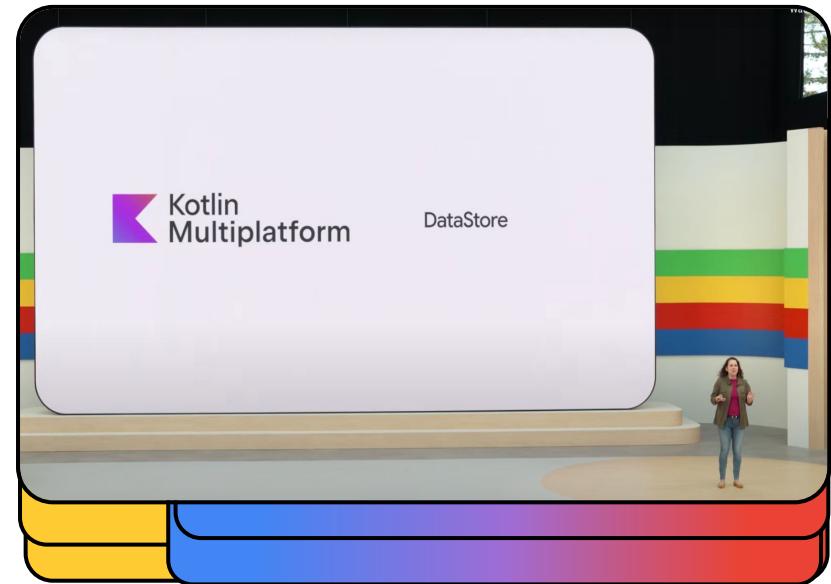
Model	支援框架	硬體要求	開放原始碼	實作穩定性	
Gemma (1&2)	MediaPipe	較簡單 (但不是低)	Y	推出時間久，記憶體 足夠時還算穩定	
Gemini Nano	AICore	較複雜	N	才出 0.0.1-exp01, 一個特殊符號可能 就會導致無法輸出	

Kotlin Multiplatform on Android

First-class tooling and **library support** for Kotlin Multiplatform on Android.

Now expanded to your favourite libraries like **Room / DataStore**.

Leveraging KMP for business logic across Android, iOS and web. Strongly **enhancing** the **ecosystem** of KMP.



KMP Case Studies



MediaPipe Demo

(以 LLM 為例)

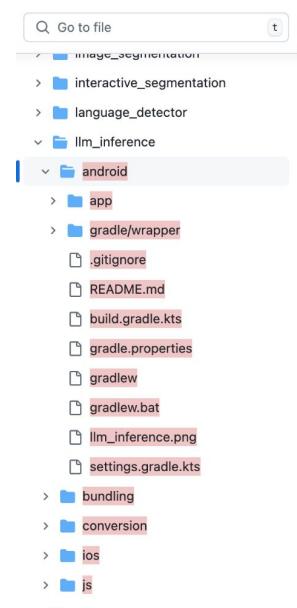
1. 純 Kotlin
 2. UI 完全採用 Jetpack Compose
 3. 所依賴的 LLM 任務 SDK 高度
封裝，僅暴露三個核心方法



這不已經達到 KMP 的標準？



Google Developer Groups



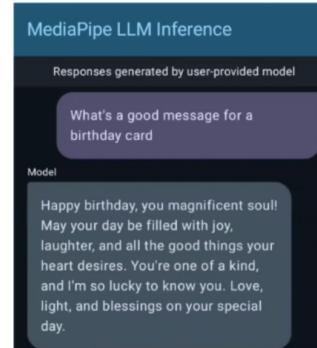
README.m

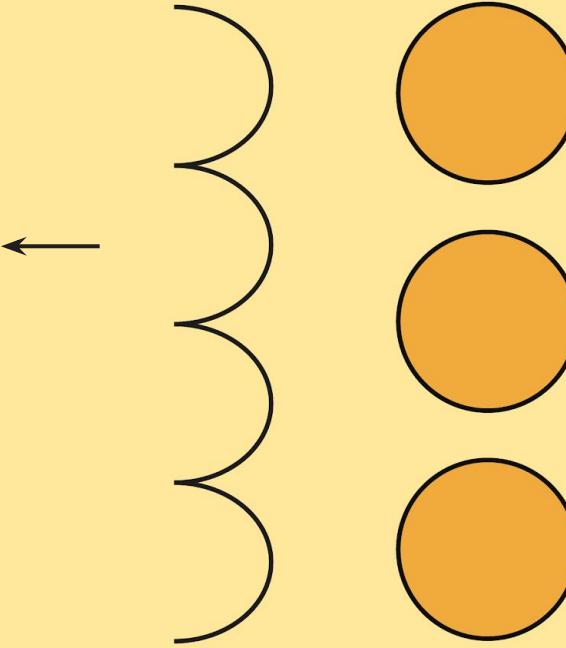
MediaPipe LLM Inference Android Demo

Overview

This is a sample app that demonstrates how to use the LLM Inference API to run common tasks such as text retrieval, email drafting, and document summarization.

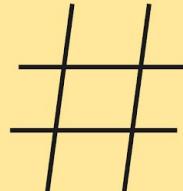
This application must be run on a physical Android device to take advantage of the device



- 
1. On Device Model + KMP
 2. 將 Android 專案移植到 KMP
 3. 換個角度看 KMP
- 



Google
Developer
Groups

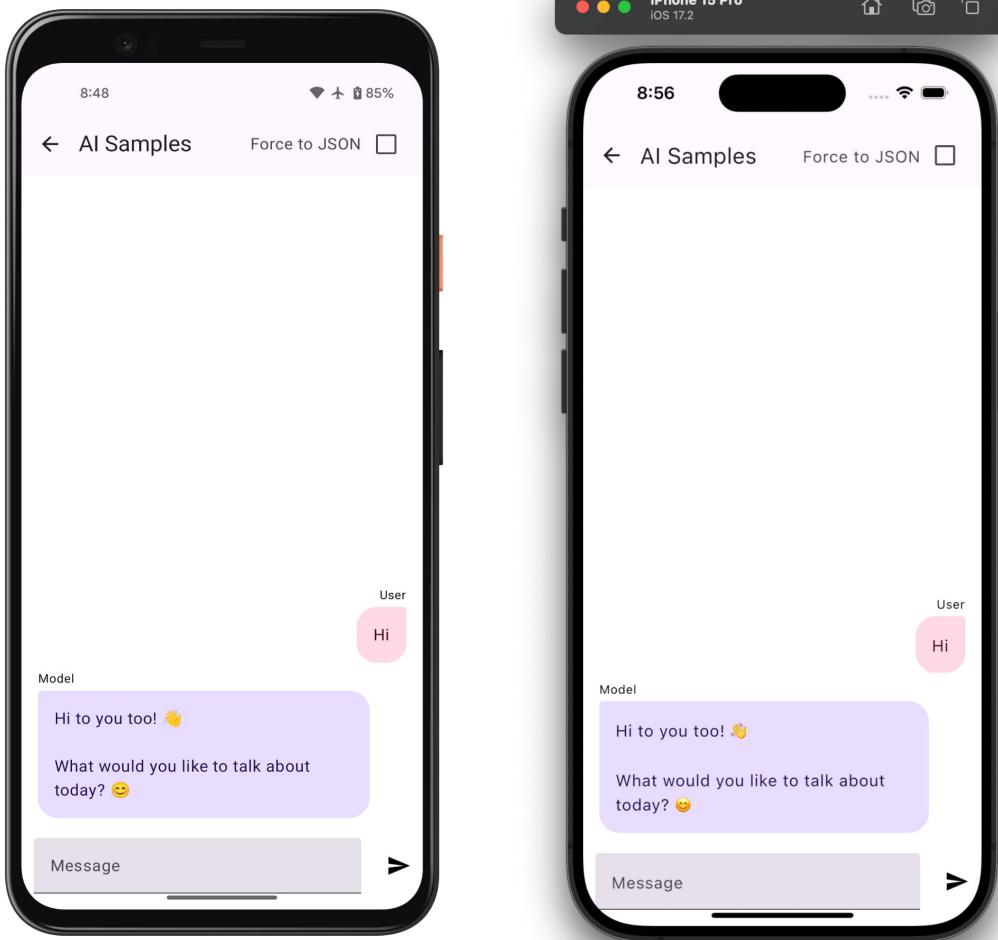


Mobile
@DevFest

LLM Inference

Showing up a conversational chat interface with gemma2-2b models.
The UI design and original codebase come from Google.

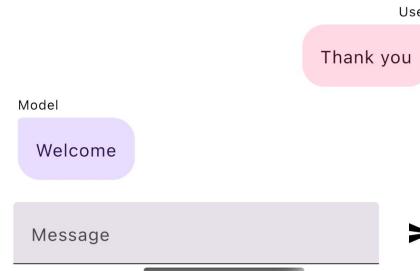
https://github.com/google-ai-edge/mediapipe-samples/tree/main/examples/llm_inference/android



UI Migration Live Demo

遷移技巧第壹式

1. 大膽嘗試
2. 替換 Android 組件的 API 為 Voyage(當然如果用了新的ViewModel 和 Navigation, 可能更簡單)
3. 替換 R 為 Res、以及 Kotlin 的 Log Message 等等



Logic Migration

```
1 class LLMInferenceAndroidImpl(private val ctx: Context): LLMOperator {  
2     ...  
3     override suspend fun initModel(): String? {  
4         ...  
5         return try {  
6             ...  
7             loadModel(modelPath)  
8         } catch (e: Exception) {}  
9     }  
10    private fun loadModel(modelPath: String) {  
11        val options = LlmInference.LlmInferenceOptions.builder()  
12        ...  
13        .setResultListener { partialResult, done →  
14            partialResultsFlow.tryEmit(partialResult to done)  
15        }  
16        .build()  
17  
18        llmInference = LlmInference.createFromOptions(ctx, options)  
19    }  
20    override fun sizeInTokens(text: String): Int = llmInference.sizeInTokens(text)  
21    override suspend fun generateResponse(inputText: String): String {  
22        ...  
23        return llmInference.generateResponse(inputText)  
24    }  
25    override suspend fun generateResponseAsync(inputText: String): Flow<Pair<String, Boolean>> {  
26        ...  
27        llmInference.generateResponseAsync(inputText)  
28        return partialResultsFlow.asSharedFlow()  
29    }  
30 }
```

LLM Task Engine for Android

```
1 plugins {
2     ...
3     alias(libs.plugins.cocoapods)
4 }
5 cocoapods {
6     ...
7     ios.deploymentTarget = "15"
8
9     pod("MediaPipeTasksGenAIC") {
10         version = "0.10.14"
11         extraOpts += listOf("-compiler-option", "-fmodules")
12     }
13     pod("MediaPipeTasksGenAI") {
14         version = "0.10.14"
15         extraOpts += listOf("-compiler-option", "-fmodules")
16     }
17 }
18 }
```

LLM Task Engine for iOS (Option 1)

```
1 // Note that below imports are starting with cocoapods
2 import cocoapods.MediaPipeTasksGenAI.MPPLLMInference
3 import cocoapods.MediaPipeTasksGenAI.MPPLLMInferenceOptions
4 import platform.FoundationNSBundle
5 ...
6 class LLMOperatorIOSImpl: LLMOperator {
7
8     private val inference: MPPLLMInference
9
10    init {
11        val modelPath = NSBundle mainBundle.pathForResource(..., "bin")
12
13        val options = MPPLLMInferenceOptions(modelPath!!)
14        options.setModelPath(modelPath!!)
15        options.setMaxTokens(2048)
16
17        ...
18        // NPE was thrown here right after it printed the success initialization message internally.
19        inference = MPPLLMInference(options, null)
20    }
21
22    override fun generateResponse(inputText: String): String {...}
23    override fun generateResponseAsync(inputText: String, ...): ... {
24        ...
25    }
26 }
```

LLM Task Engine for iOS (Option 1)

可能是因為 Interop 目前的支援仍有限，對於 Swift 程式碼中帶有 throws 關鍵字的 init 方法轉換有些問題 (KT 1.9/2.0)。

```
1 // Add one more interface for Swift
2 // app/src/iosMain/.../llm/LLMOperator.kt
3 interface LLMOperatorSwift {
4     suspend fun loadModel(modelName: String)
5     fun sizeInTokens(text: String): Int
6     suspend fun generateResponse(inputText: String): String
7     suspend fun generateResponseAsync(
8         inputText: String,
9         progress: (partialResponse: String) → Unit,
10        completion: (completeResponse: String) → Unit
11    )
12 }
```

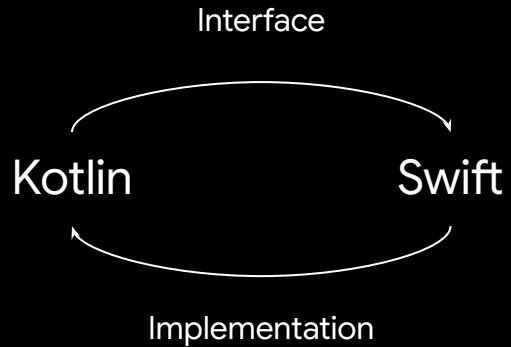
LLM Task Engine for iOS (Option 2)

```
1 // iosApp/iosApp/LLMIferenceDelegate.swift
2 class LLMOperatorSwiftImpl: LLMOperatorSwift {
3 ...
4     var llmInference: LlmInference?
5     func loadModel(modelName: String) async throws {
6         let path = Bundle.main.path(forResource: modelName, ofType: "bin")!
7         let llmOptions = LlmInference.Options(modelPath: path)
8         ...// options parameters setup
9         llmInference = try LlmInference(options: llmOptions)
10    }
11    func generateResponse(inputText: String) async throws -> String {
12        return try llmInference!.generateResponse(inputText: inputText)
13    }
14    func generateResponseAsync(inputText: String, progress: @escaping (String) -> Void) async throws {
15        try llmInference!.generateResponseAsync(inputText: inputText) { partialResponse, error in
16            // progress
17            if let e = error {
18                completion(e.localizedDescription)
19                return
20            }
21            if let partial = partialResponse {
22                progress(partial)
23            }
24        } completion: {completion("")}
25    }
26 }
```

LLM Task Engine for iOS (Option 2)

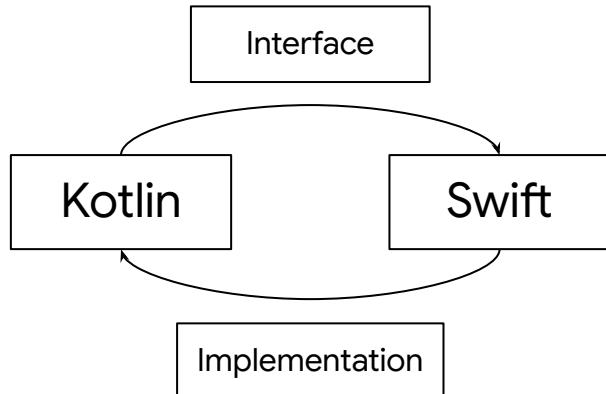
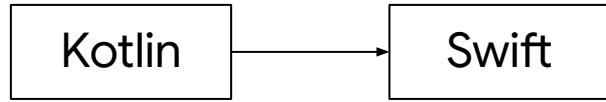
```
1 // iosApp/iosApp/iosApp.swift
2 class AppDelegate: UIResponder, UIApplicationDelegate {
3     ...
4     func application() {
5         ...
6         let delegate = try LLMOperatorSwiftImpl()
7         MainKt.onStartUp(llmInferenceDelegate: delegate)
8     }
9 }
10
11 // LLMOperatorIOSImpl.kt
12 class LLMOperatorIOSImpl(
13     private val delegate: LLMOperatorSwift) : LLMOperator {
14     ...
15 }
```

LLM Task Engine for iOS (Option 2)



遷移技巧第貳式

1. Kotlin 可以直接呼叫透過 CocoaPods 引入的第三方函式庫。
2. Kotlin 也能透過 iOS 專案呼叫第三方函式庫。
3. ObjC Export 有升級空間, Swift Export 更是值得期待。



遷移技巧第貳式

1. Kotlin 可以直接呼叫透過 CocoaPods 引入的第三方函式庫。
2. Kotlin 也能透過 iOS 專案呼叫第三方函式庫。
3. ObjC Export 有升級空間, Swift Export 更是值得期待。

小彩蛋：

Mediapipe 已支援 Gemma2-2B, 你可以讓它寫封郵件, 改個簡報備忘, 或者分析一份履歷。

12:03

AI Samples (gemma2-2b-it-gpu-int8)

Improvements:

* **Focus on impact:** Instead of just listing tasks and responsibilities, the summary could emphasize the impact Ty's work has had on Uber, developers, and the industry.

* **Quantify achievements:** Including specific metrics like team size, projects delivered, and code contributions would strengthen the impact of his qualifications.

* **Tailor to target audience:** The resume could be tailored to specific job applications, highlighting skills and experiences most relevant to each role.

* **More visual appeal:** Use strong action verbs and bullet points for better readability.

Overall, this resume effectively showcases Ty Smith's impressive technical skills and leadership experience. With minor adjustments, it can further amplify his achievements and make him a compelling candidate.



Google Developer Groups

Message



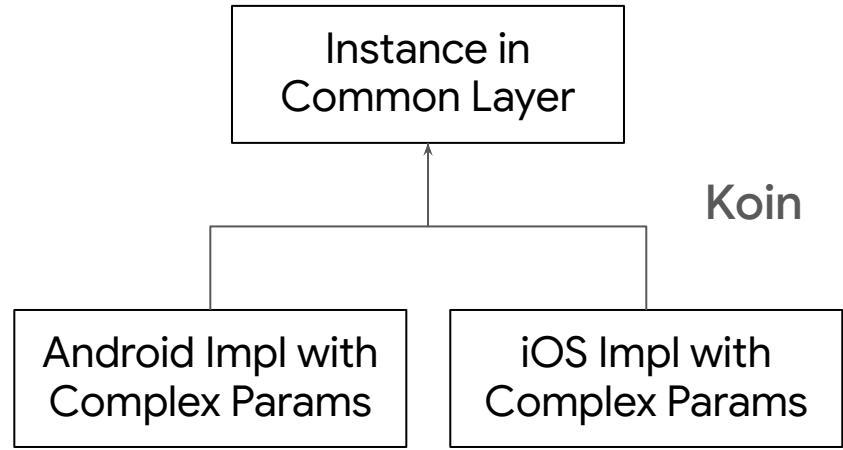
**What about the
passing procedure?**

Multiplatform DI With Koin

```
1 // Common
2 expect class LLMOperatorFactory {
3     fun create(): LLMOperator
4 }
5 val sharedModule = module {
6     // Creating LLMOperator for Common Layer from different LLMOperatorFactory
7     single<LLMOperator> { get<LLMOperatorFactory>().create() }
8 }
9
10 // Android
11 actual class LLMOperatorFactory(private val context: Context){
12     actual fun create(): LLMOperator = LLMInferenceAndroidImpl(context)
13 }
14 val androidModule = module {
15     // Inject Application Context for Android
16     single { LLMOperatorFactory(androidContext()) }
17 }
18
19 // iOS
20 actual class LLMOperatorFactory(private val llmInferenceDelegate: LLMOperatorSwift) {
21     actual fun create(): LLMOperator = LLMOperatorIOSImpl(llmInferenceDelegate)
22 }
23 module {
24     // Inject the delegate from `onStartup` method for iOS
25     single { LLMOperatorFactory(llmInferenceDelegate) }
26 }
```

遷移技巧第叁式

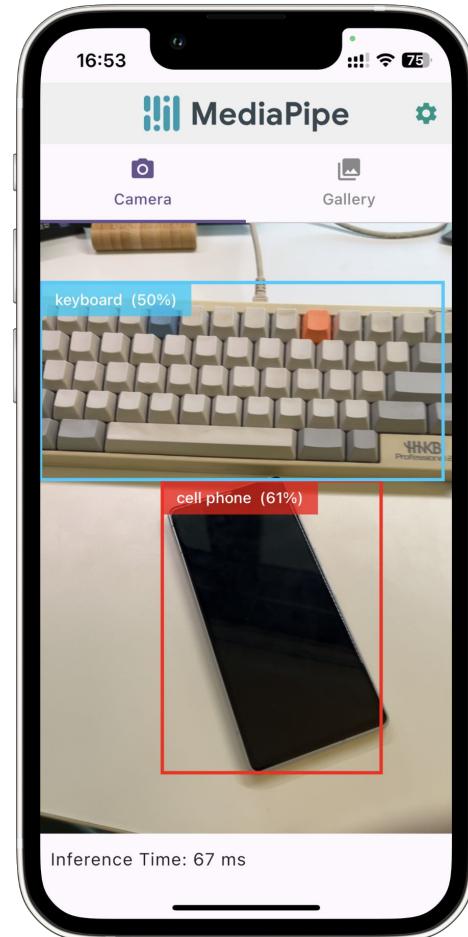
1. 當兩端的 Impl 實例需要不同的建構子參數時，通常會使用 KMP 的 expect 與 actual 關鍵字來解決此類需求。
2. 利用 `expect` 類別不需要聲明建構子參數的特性，增加了一層封裝。
3. 透過 Koin，為各平台注入所需的參數(很多時候還要增加額外的計算邏輯)，並統一將建立好的介面實作注入到Common 層的使用場景中。

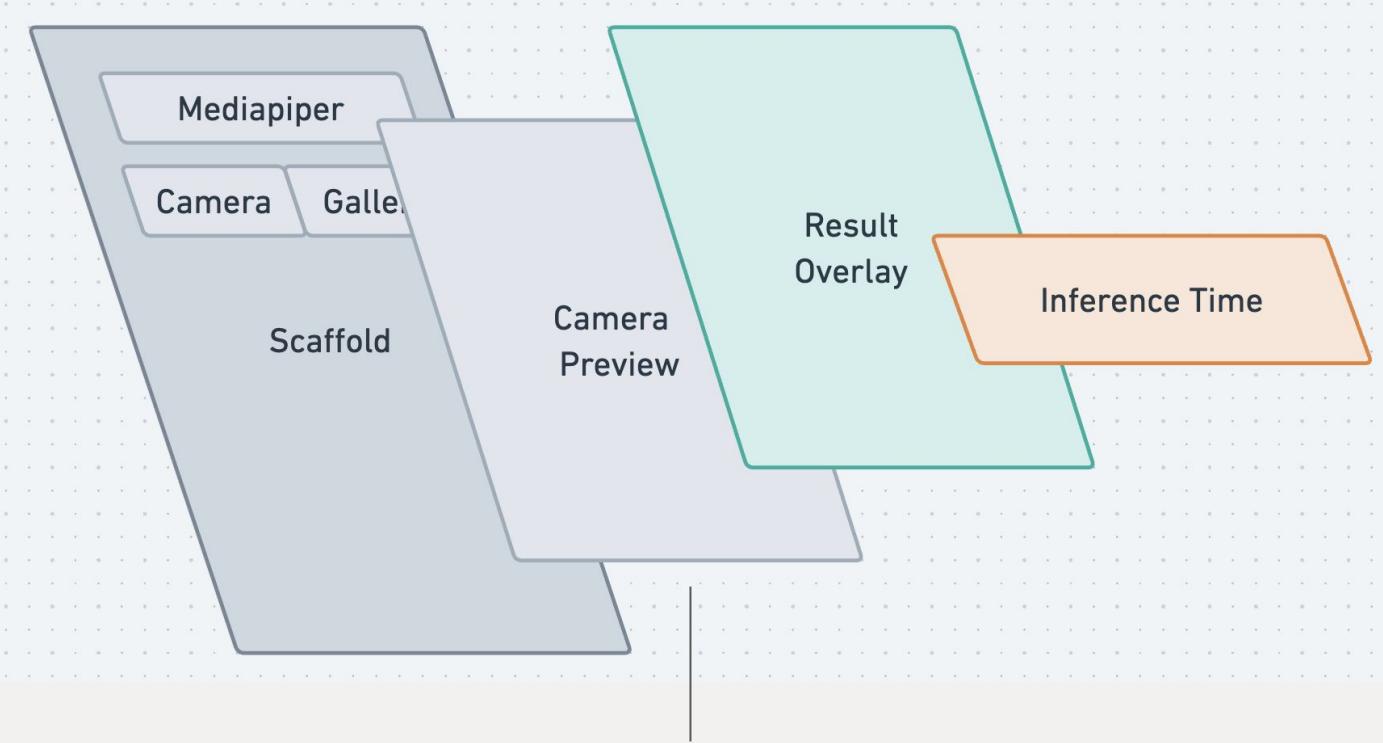


Object Detection

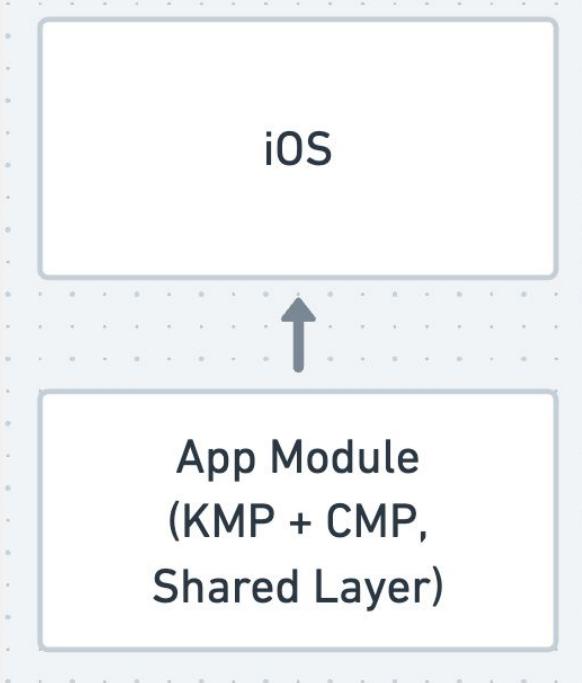
(物件偵測) Showing up a live object detection with camera preview. The UI design and original codebase come from Google.

https://github.com/google-ai-edge/mediapipe-samples/tree/main/examples/object_detection/android-jetpack-compose





Google Developer Groups



2. Kotlin 如何呼叫上層的 iOS 專案？

Compose Multiplatform + Native View

```
1 @Composable
2 fun CameraView(
3     threshold: Float,
4     maxResults: Int,
5     delegate: Int,
6     mlModel: Int,
7     setInferenceTime: (newInferenceTime: Int) → Unit,
8 ) {
9     CameraPermissionControl {
10         CameraPreview(
11             threshold,
12             maxResults,
13             delegate,
14             mlModel,
15             setInferenceTime,
16             onDetectionResultUpdate = { detectionResults →
17                 ...
18             })
19     }
20 }
21
22 @Composable
23 expect fun CameraPermissionControl(PermissionGrantedContent: @Composable @UiComposable () → Unit)
24
25 @Composable
26 expect fun CameraPreview(...)
```

```
2 import platform.AVFoundation.*  
3  
4 @Composable  
5 actual fun CameraPermissionControl(PermissionGrantedContent: @Composable @UiComposable () -> Unit) {  
6     var hasCameraPermission by remember { mutableStateOf<Boolean?>(null) }  
7     LaunchedEffect(Unit) {  
8         hasCameraPermission = requestCameraAccess()  
9     }  
10    when (hasCameraPermission) {  
11        true -> { PermissionGrantedContent() }  
12        false -> { Text("...") }  
13        null -> { Text("Requesting camera permission...") }  
14    }  
15 }  
16  
17 private suspend fun requestCameraAccess(): Boolean = suspendCoroutine { continuation ->  
18     val authorizationStatus = AVCaptureDevice.authorizationStatusForMediaType(AVMediaTypeVideo)  
19     when (authorizationStatus) {  
20         AVAuthorizationStatusNotDetermined -> {  
21             AVCaptureDevice.requestAccessForMediaType(AVMediaTypeVideo) { granted ->  
22                 continuation.resume(granted)  
23             }  
24         }  
25         AVAuthorizationStatusRestricted, AVAuthorizationStatusDenied -> {  
26             continuation.resume(false)  
27         }  
28         AVAuthorizationStatusAuthorized -> {  
29             continuation.resume(true)  
30         }  
31         else -> {  
32             continuation.resume(false)  
33         }  
34     }  
35 }
```

iOS Permission Control in Kotlin

Pass Back

```
1 typealias IOSCameraPreviewCreator = (
2     threshold: Float,
3     maxResults: Int,
4     delegate: Int,
5     mlModel: Int,
6     setInferenceTime: (newInferenceTime: Int) → Unit,
7     callback: IOSCameraPreviewCallback
8 ) → UIView
9
10 typealias IOSCameraPreviewCallback = (result: ObjectDetectionResult) → Unit
11
12 fun onStartup(iosCameraPreviewCreator: IOSCameraPreviewCreator) {
13     Startup.run { koinApp →
14         koinApp.apply {
15             modules(module {
16                 single { LLMOperatorFactory() }
17                 single<IOSCameraPreviewCreator> { iosCameraPreviewCreator }
18             })
19         }
20     }
21 }
```

Pass Back

```
1 import androidx.compose.ui.viewinterop.UIKitView
2 import platformUIKit.UIView
3
4 @Composable
5 actual fun CameraPreview(
6     ...
7 ) {
8     val iOSCameraPreviewCreator = koinInject<IOSCameraPreviewCreator>()
9     UIKitView(
10         factory = {
11             val iosCameraPreview: UIView = iOSCameraPreviewCreator(
12                 threshold,
13                 maxResults,
14                 delegate,
15                 mLModel,
16                 setInferenceTime,
17                 onDetectionResultUpdate)
18             iosCameraPreview
19         },
20         modifier = Modifier.fillMaxSize(),
21         update = { _ → }
22     )
23 }
```

遷移技巧第肆式

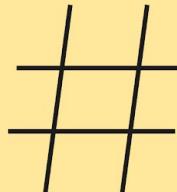
1. 使用 Compose Multiplatform 嵌入 UIKit View, 可以解決許多高效能需求, 包括需要直接使用**複雜原生 API 和硬體資源**的情況。
2. 實際上, 彩色方框(Overlay)若是在 CMP 編寫, 可以擴展出更多可能。例如視訊通話時, 結合人臉識別 Model, 提供 Emoji 選擇與渲染。由於**邏輯的高度複雜性, 重用同一個 StickerOverlay 的價值變得非常顯著**。



- 
1. On Device Model + KMP
 2. 將 Android 專案移植到 KMP
 3. 換個角度看 KMP



Google
Developer
Groups



Mobile
@DevFest

換個角度看 KMP

1. 可“遷移”是個值得利用的條件。
2. 企業：重視成本。
3. Kotlin/Native 被視為 Native，但 CMP 還未成熟。



新創



成熟企業

Android	iOS
Native	Half Native

99% KMP+CMP

80% KMP+CMP
20% Swift + SwiftUI

Android	iOS
Native	75% Native

Android iOS

Native 75% Native

「新專案」

-> 重用舊專案的模組化的 Android 原始碼

「舊專案是跨平台方案，想轉 Native」

-> 把 Logic 部分重寫，UI 盡量 Native（核心功能）



Google Developer Groups

我只是在寫 Android 專案時稍加
改動，結果，iOS 也寫好了。

Why KMP?

Fluent Interop

以 iOS 為例, ObjC-Interop 賦予 Kotlin 直接呼叫 iOS 相依項目的能力; 反過來, Kotlin 編譯生成的 .framework 也能讓 Swift/ObjC 無縫地呼叫 Kotlin 所撰寫的功能。



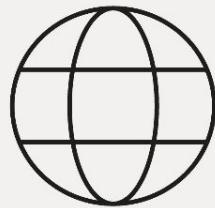
Native-level Perf

以 iOS 為例, 非 UI 的常見場景都能達到接近 Swift 的效能水準。

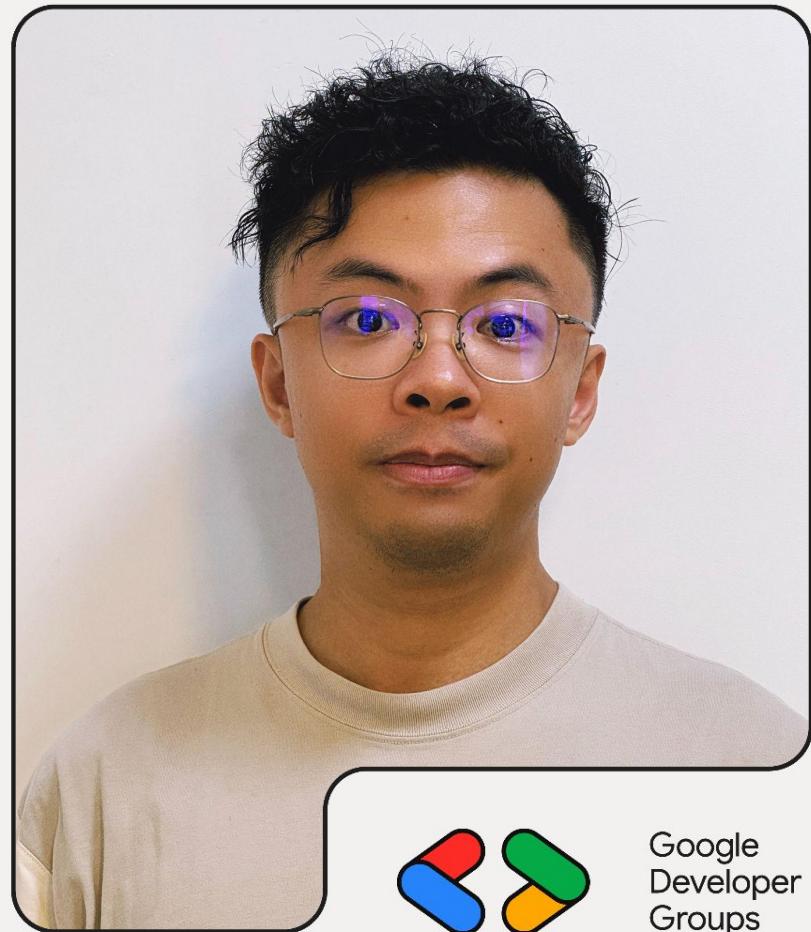
Progressive Adoption

你永遠可以選擇只使用它來實現部分功能, 並充分運用現有的 Android 程式碼。以 iOS 為例, 基於 Kotlin 的 Android 應用程式只需簡單修改, 即可輕鬆遷移到 iOS, 無需重寫整個應用。

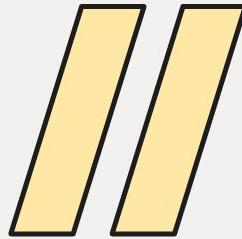
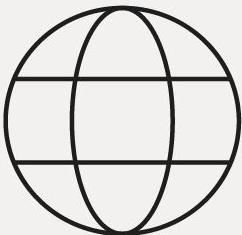
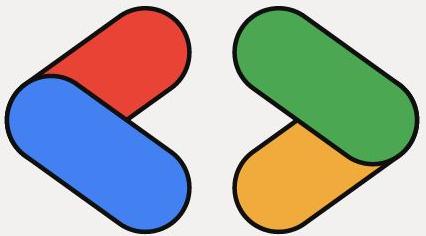




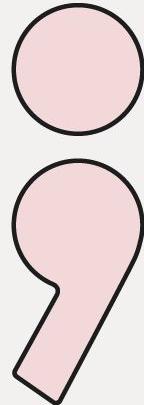
Thank You



Google
Developer
Groups



This is an example of a section title slide.



Google
Developer
Groups



Google Blue 500
#4285f4



Google Green 500
#34a853



Google Yellow 500
#fbfc04



Google Red 500
#ea4335



Halftone Blue
#57caff



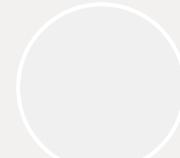
Halftone Green
#5cdb6d



Halftone Yellow
#ffd427



Halftone Red
#ff7daf



Off-White
#f0f0f0



Pastel Blue
##c3ecf6



Pastel Green
#ccf6c5



Pastel Yellow
#ffe7a5



Pastel Red
#f8d8d8



Black 02
#1e1e1e

Headlines

Google Sans Bold

Subheads

Roboto Mono Light

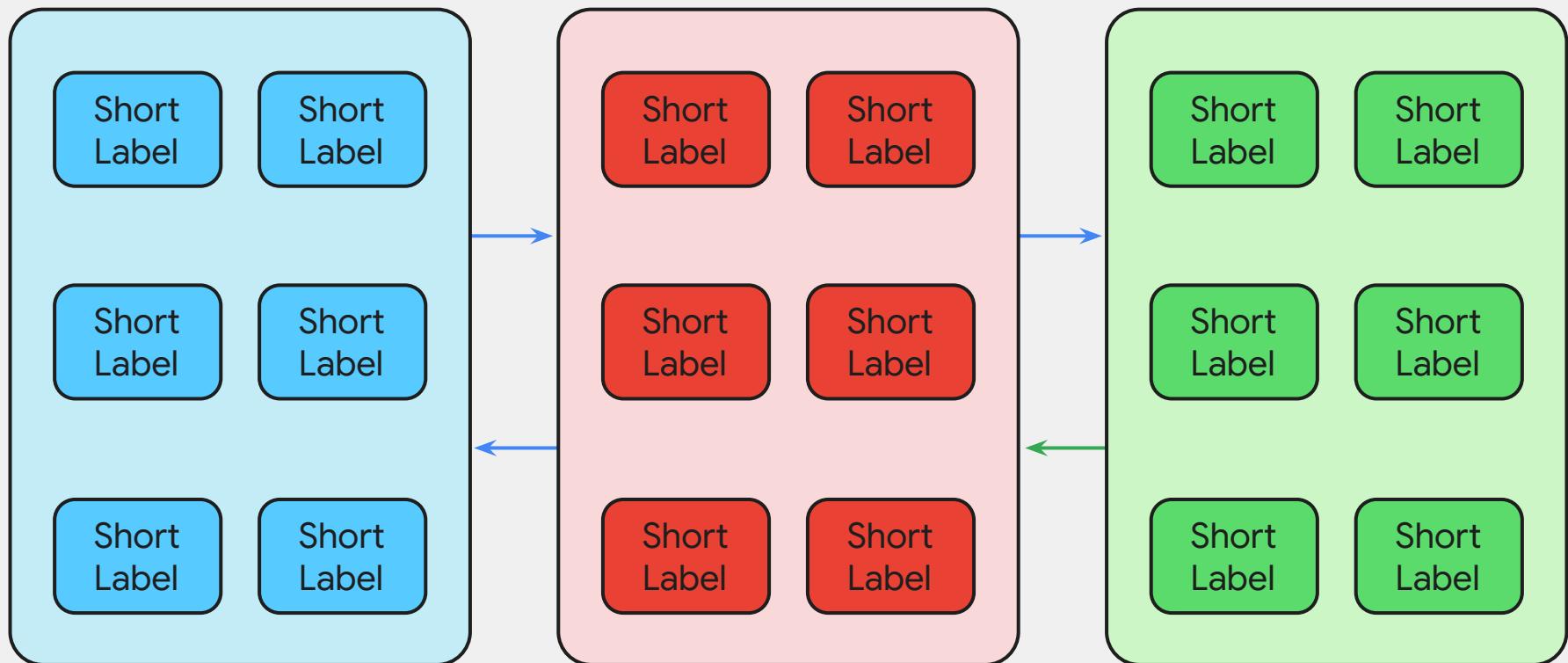
Body

Google Sans Normal

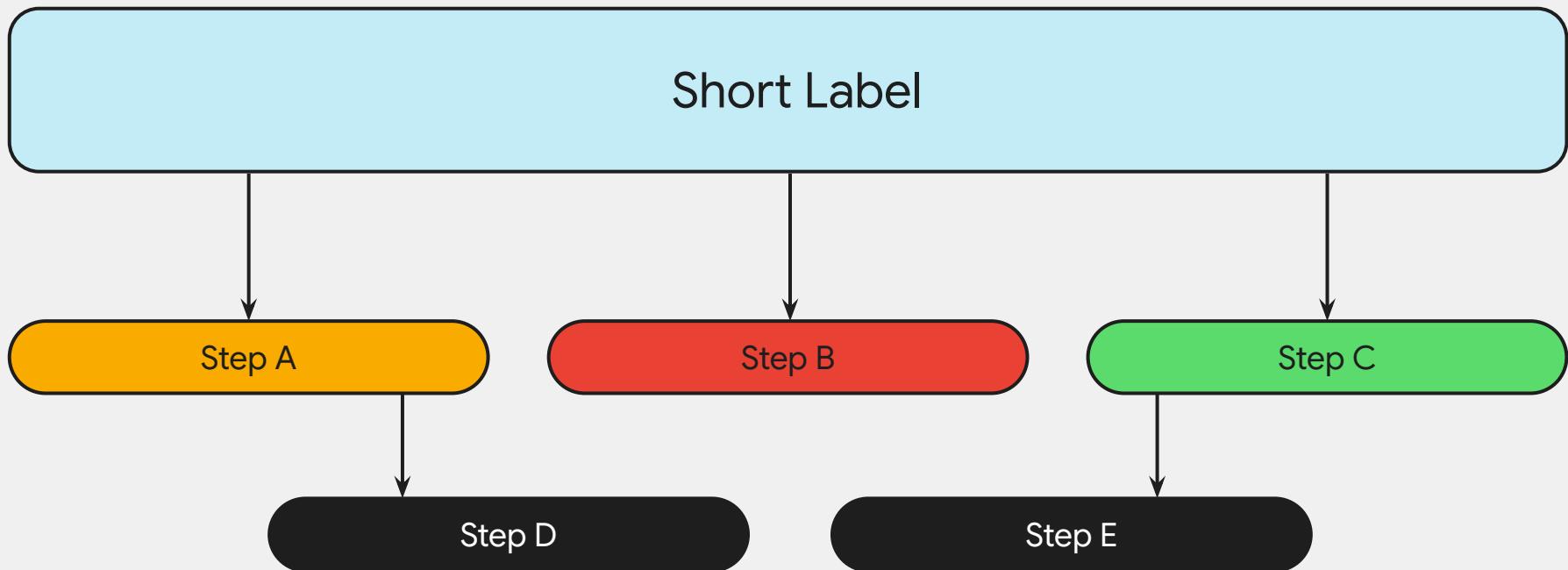


Google Developer Groups

Simple Chart



All-Purpose Chart



Big Portfolio Chart

Label One

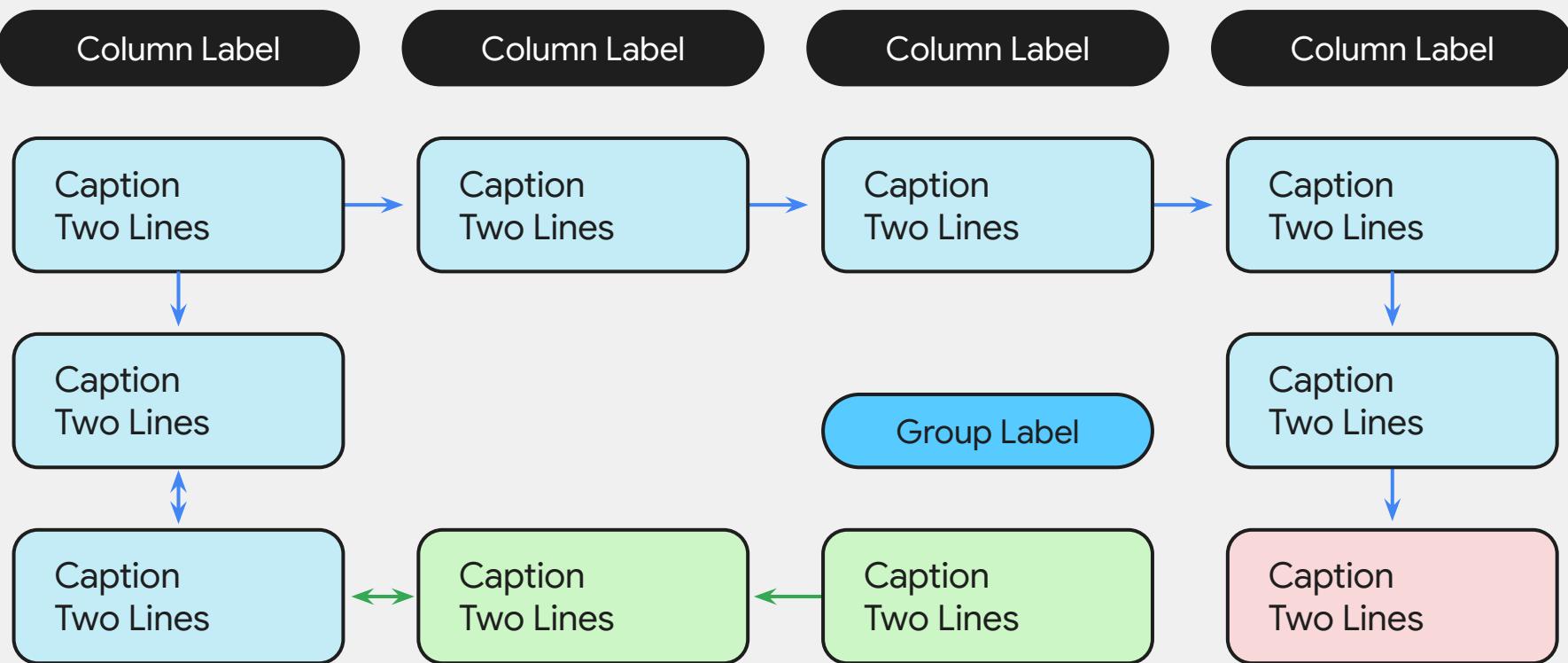
Label Two

Label Three

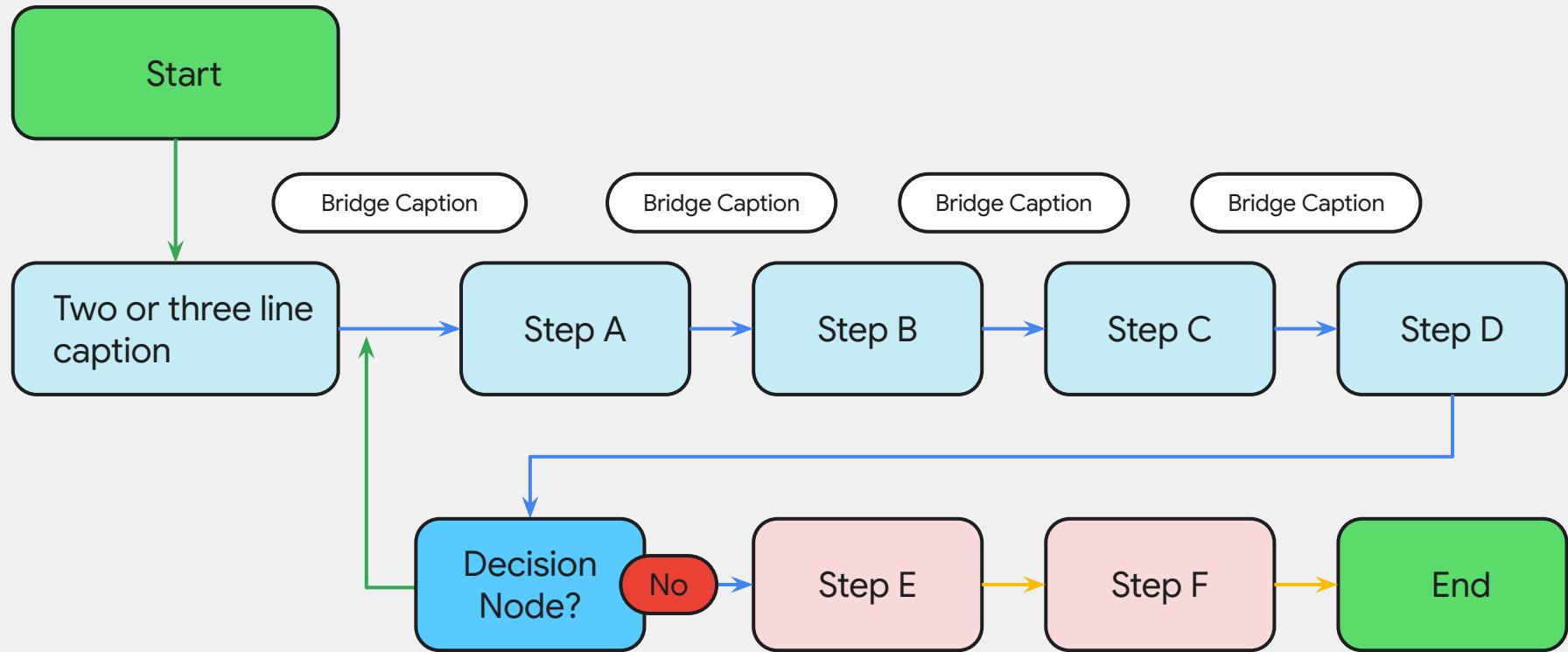
Label Four

Short Label

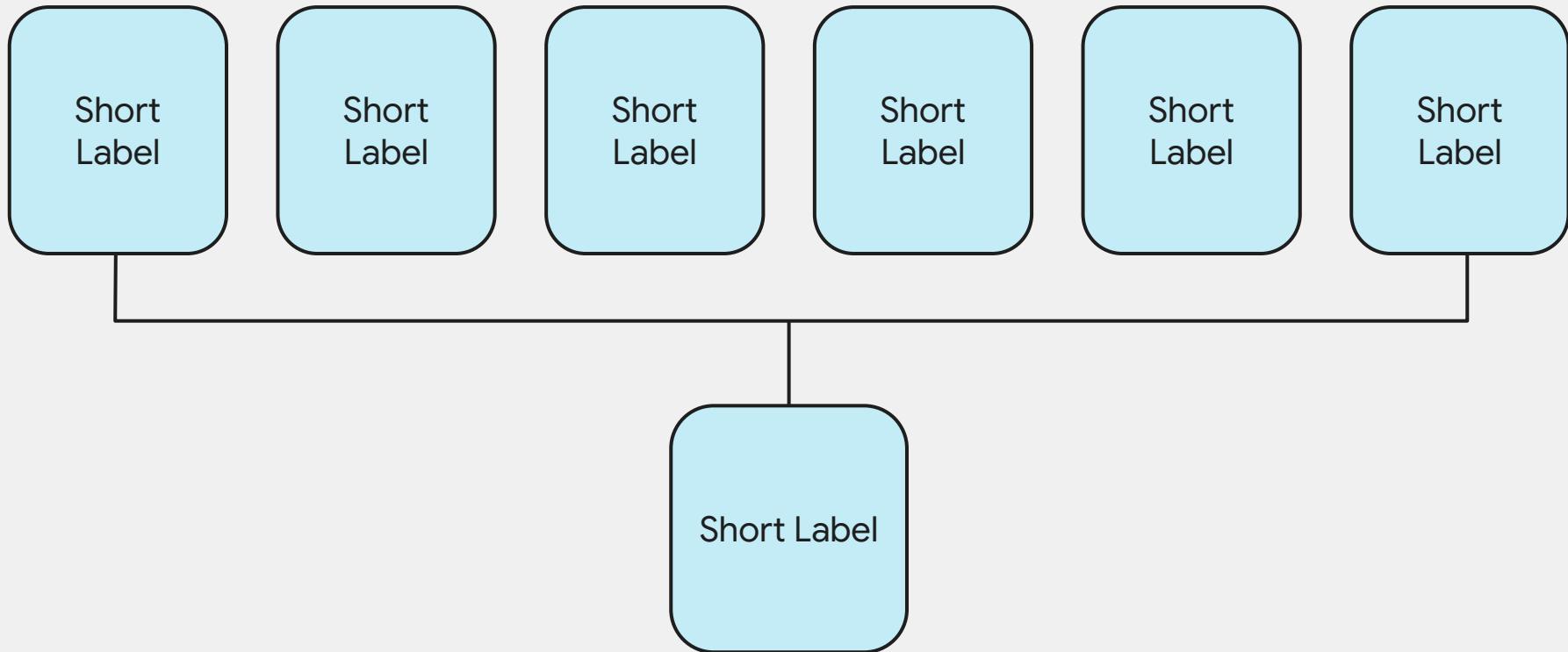
Process Chart



Flow-Style Chart



Group Chart



Process Chart

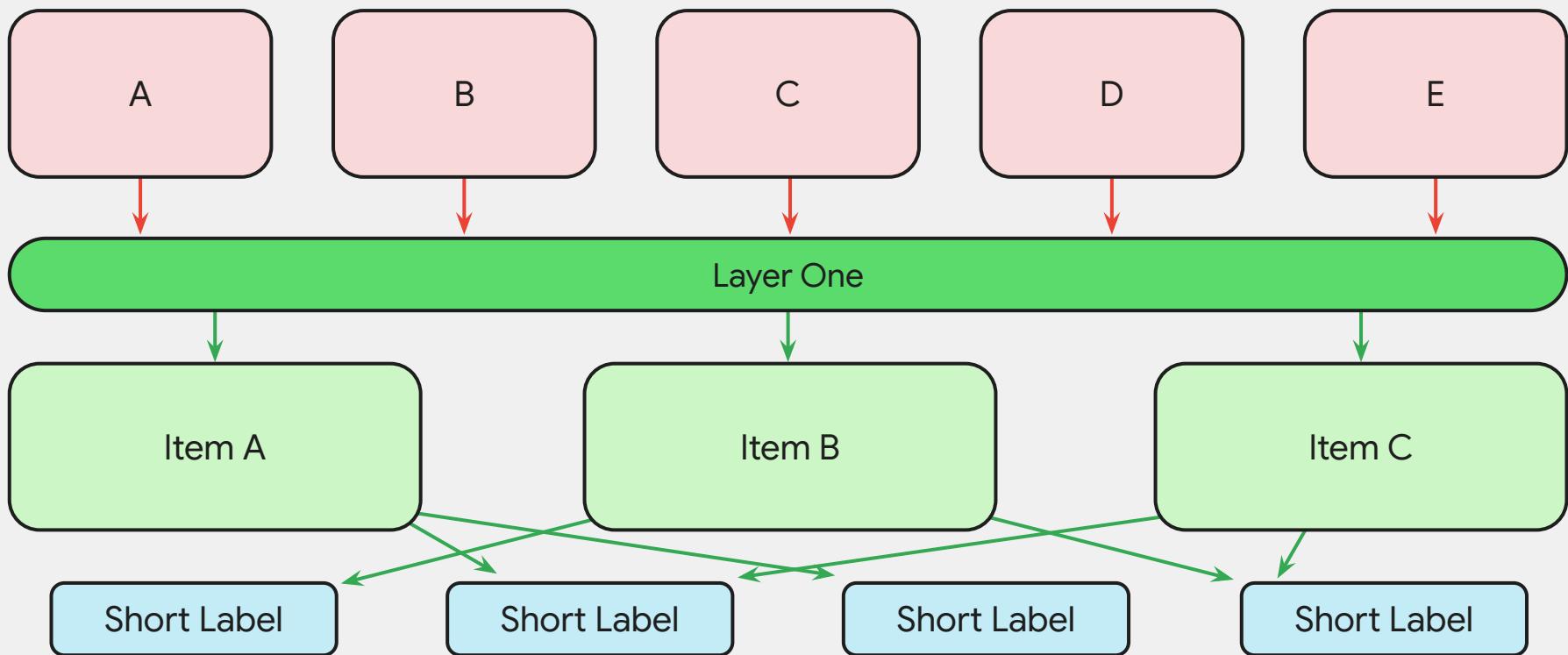
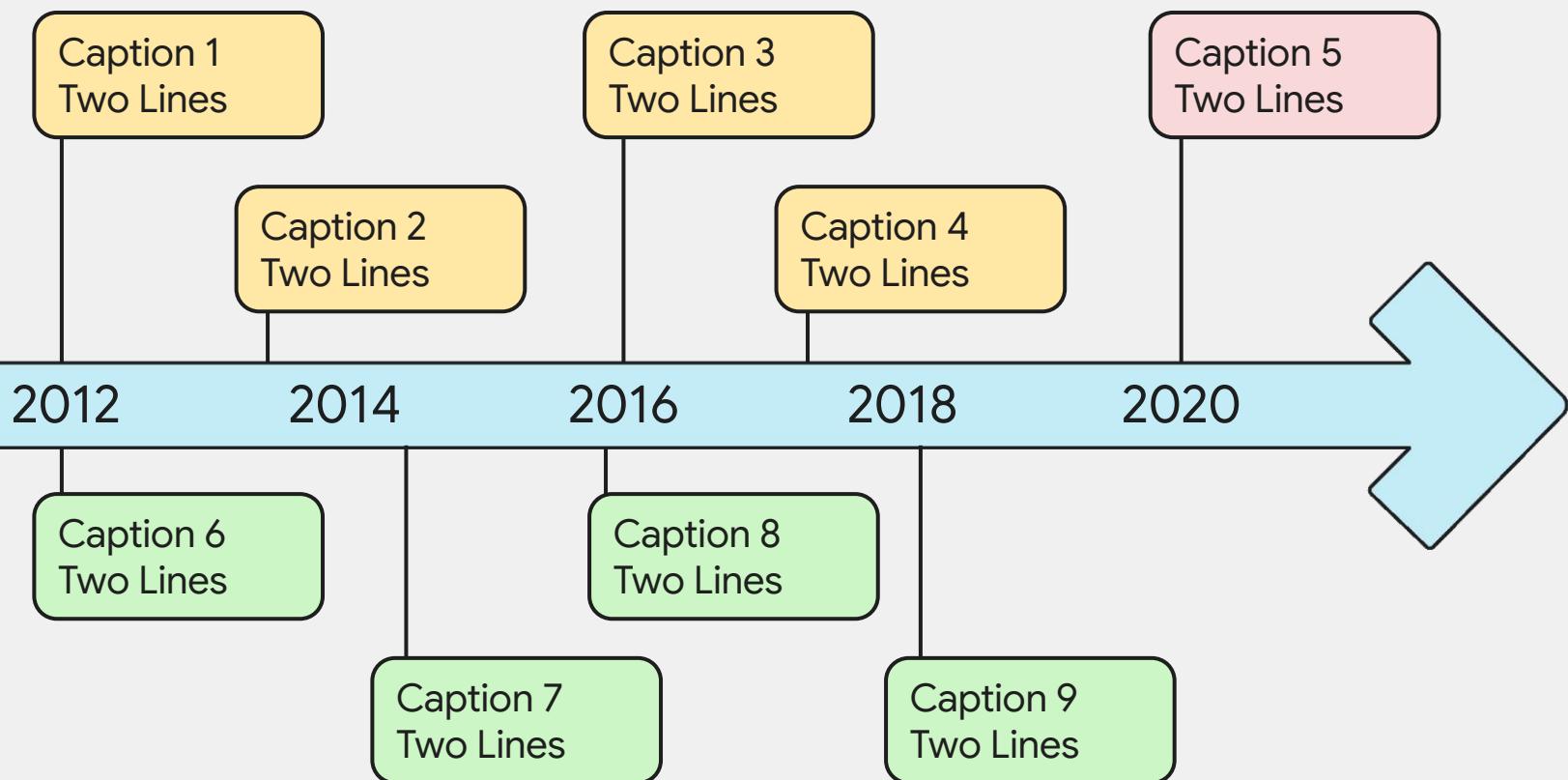


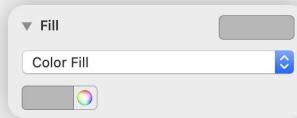
Chart Title



Icons

Icons

All icons are vector objects and can be recolored using the fill menu.



Accessibility



Expand



Late



Credit card



Extension



Thumb Up



Remove



Verified



Q&A



Finance



Android



Turn in



Trash



Actions



Download



History



Store



List



Wallet



Announcement



Backup



Document



Favorite 1



Open



Home



Print



Swap



Account



Ratio



Tag



Server



Favorite 2



Grade/rate



Lock



Language



Receipt



Add shopping



Chart



Bug



Event



Find Page



Page view



Basket



Time



Work

Icons

All icons are vector objects and can be recolored using the fill menu.



Alarm



Assessment



Sync



Exit App



Movie



Visibility



Trolley



Open



Location



Settings



Assignment



Check



Explore



Thumb Down



Today



Perm Media



People



search



Airplane



Signal



Photo



Play 1



Block



Send



Smartphone



Style



Walk



Bluetooth



WiFi



Upload



Play 2



Email



Laptop



iPhone



Controls



Bike



Pie Chart



Money



Attachment



Video



Business



Chromebook



Security



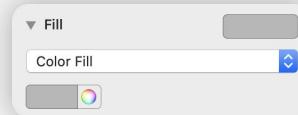
Notification



Bus

Icons

All icons are vector objects and can be recolored using the fill menu.



Developer



Write



Cloud



Audio



Key



Desktop Mac



Watch



Person



Car



Devices



Quote



Folder



Web Page



Archive



Desktop PC



Flag



World



Boat



Software



Emotion



Mic



Call



Cut



headphones



Camera



Education



Train



Weather



Link



Movie



Chart



Paste



Keyboard



TV



MMS



Subway



Hotel



Laundry



Location History



Layers



Offer



Map



Bar



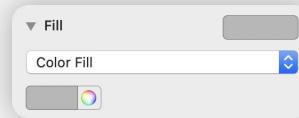
Pizza



Web

Icons

All icons are vector objects and can be recolored using the fill menu.



Cafe



Theatre



Gaming



Florist



Restaurant



Gas



Delivery



Hospital



Taxi



Print



Radio



Stream



Flags

Americas



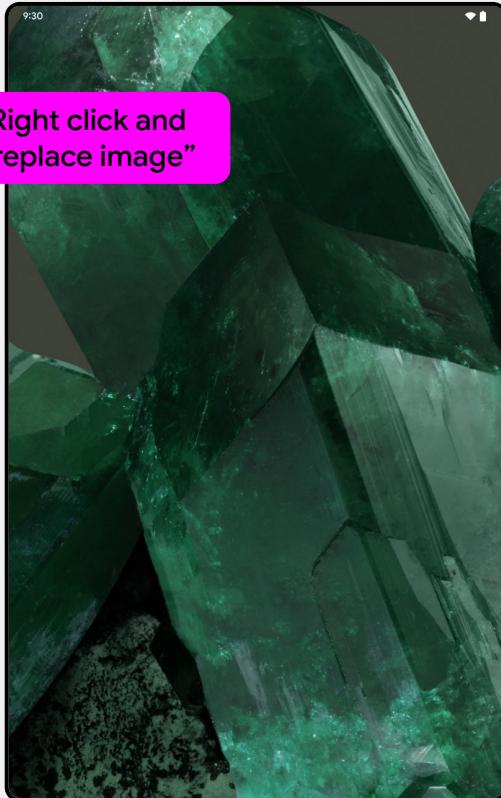
APAC



EMEA









DEVICE FRAME - PIXEL 8

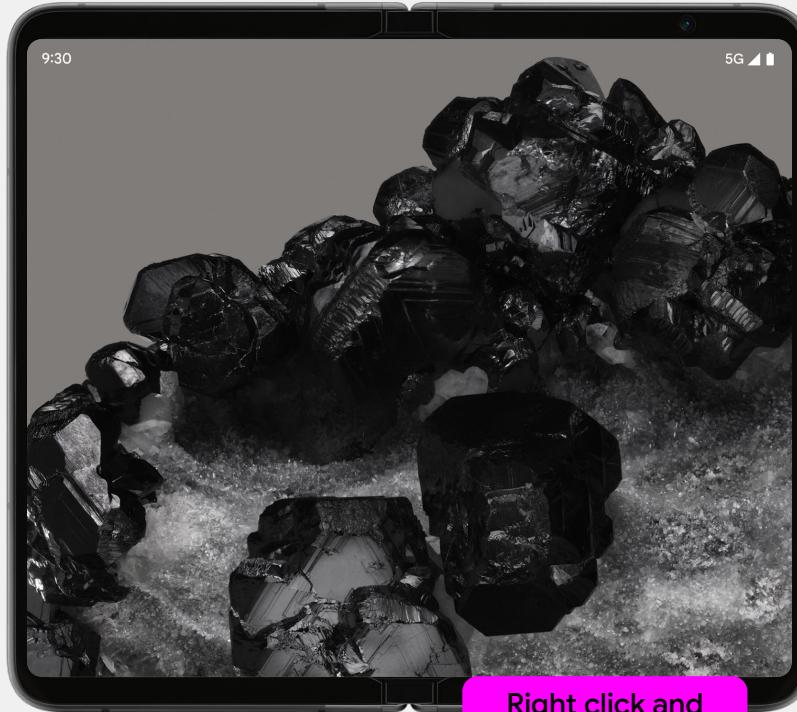
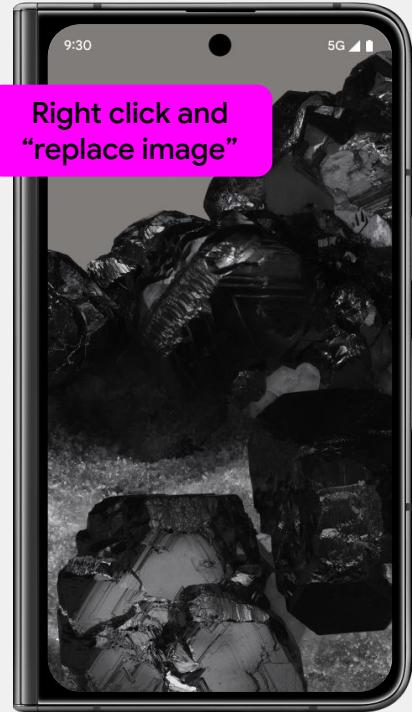


Right click and
“replace image”

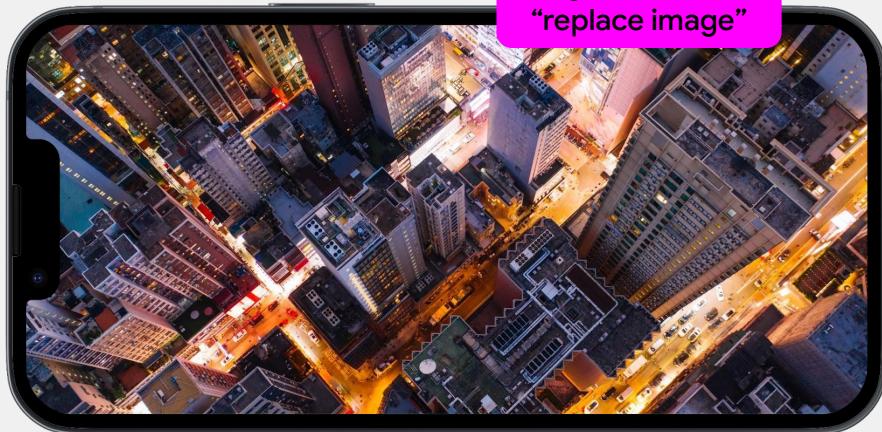


Right click and
“replace image”

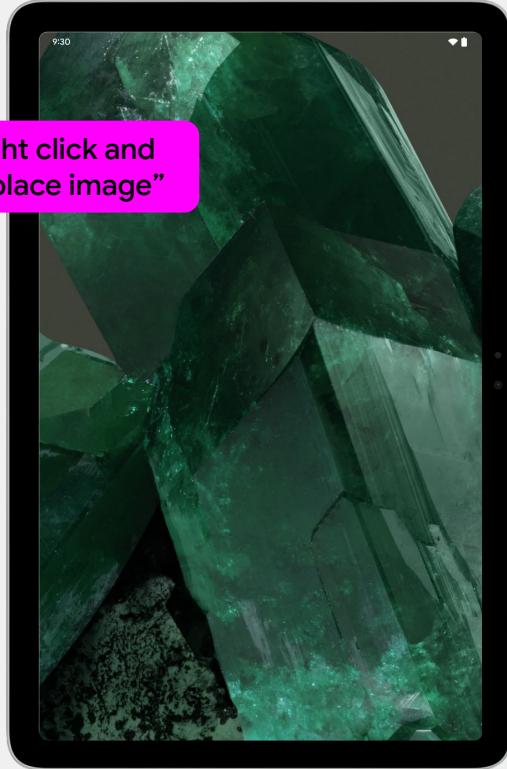
DEVICE FRAME - PIXEL FOLD



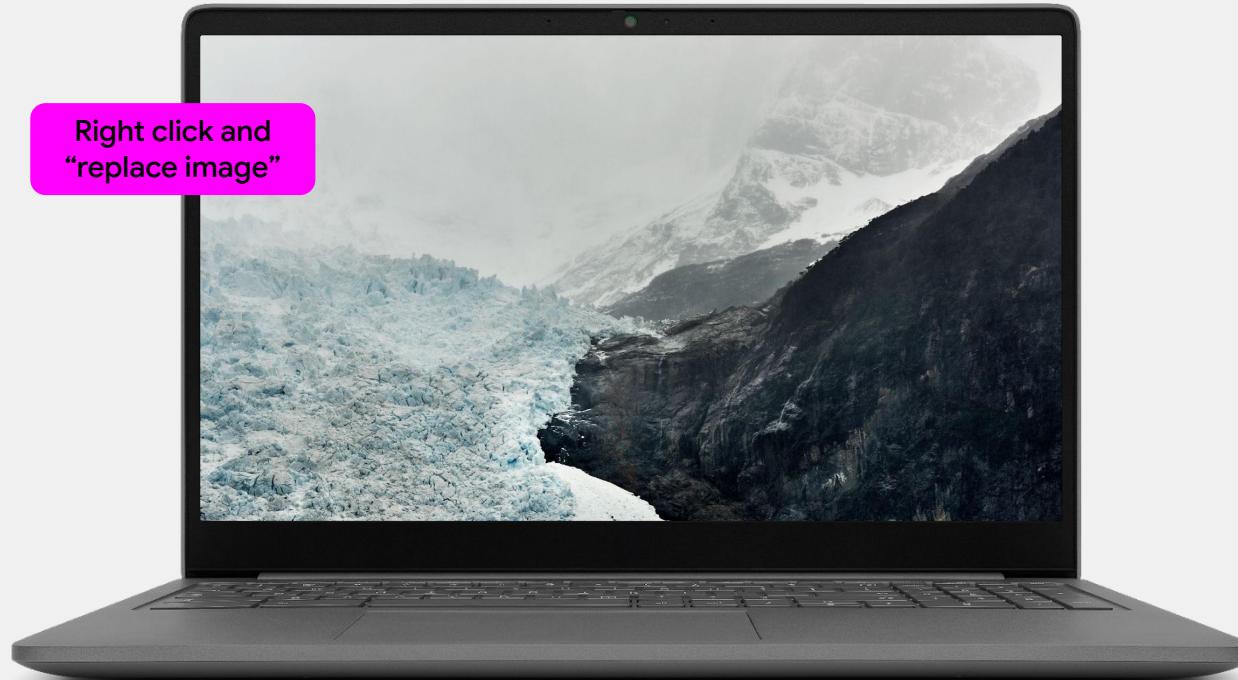
DEVICE FRAME - IPHONE 14



DEVICE FRAME - TABLET









Logo Library

Logos can be scaled to any size



Source: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non erat sem