Kotlin Multiplatform Mobile (KMM)

from an iOS Developer’s Perspective in Practice

Christian Menschel / Gerd Bernhardt

About us

We are
Gerd & Christian

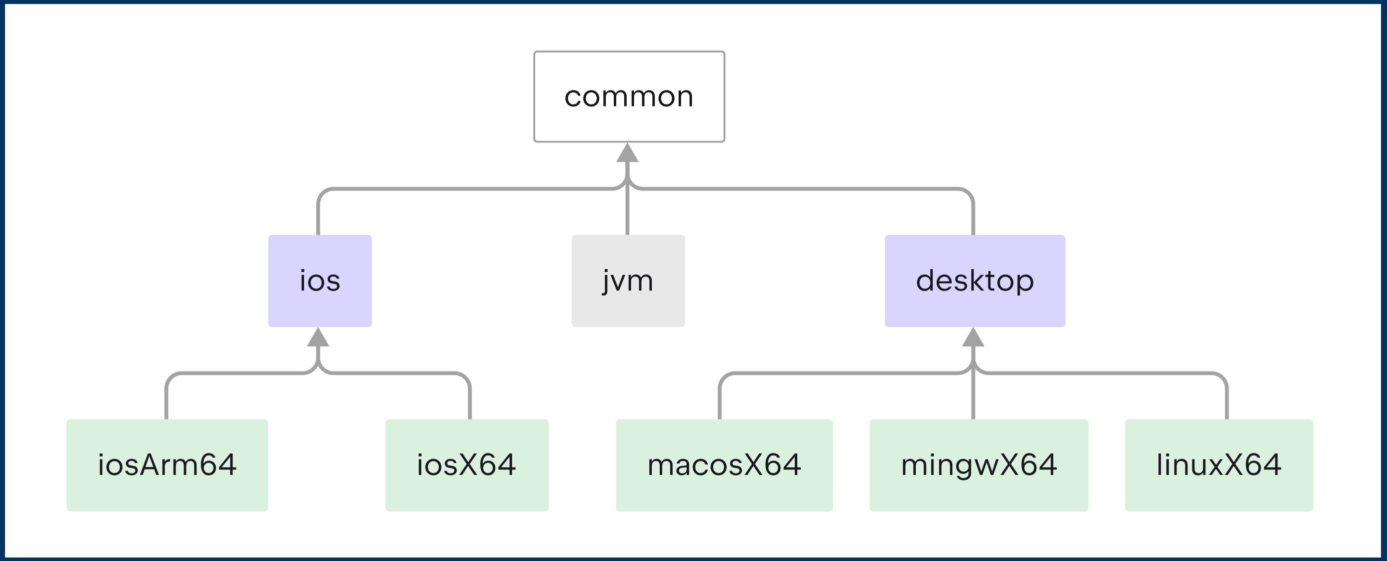
Developing iOS for many years and got in touch with

Kotlin Multiplatform Mobile (KMM) in our last / current project.

What is Kotlin Multiplatform?

What is Kotlin Multiplatform?

- Kotlin based framework
- Build multi-platform apps
- Share common code among all or some platforms
  Business logic / network-layer / models, etc.
- Access platform specific code natively
- No UI provided by KMP

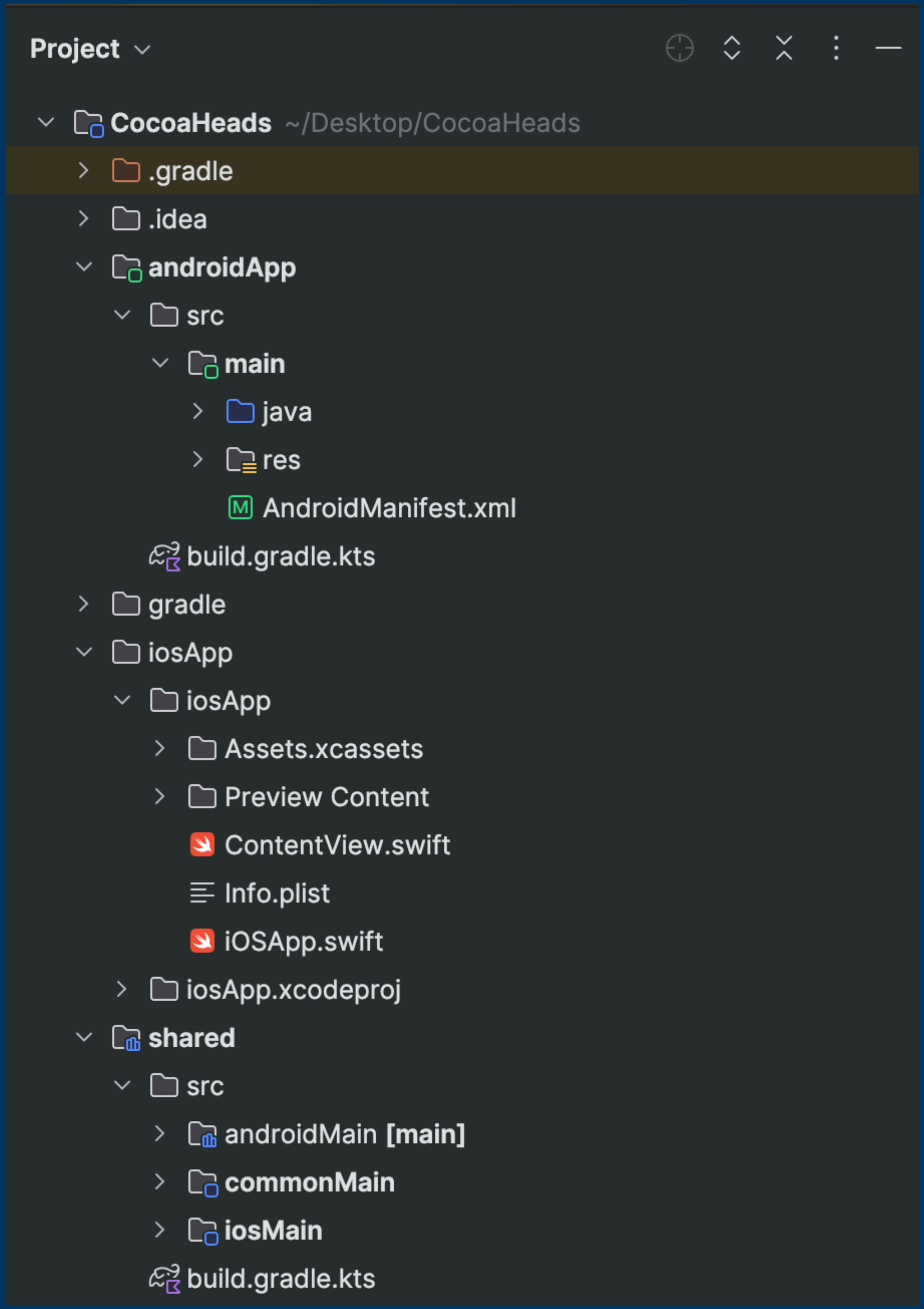


https://kotlinlang.org/docs/multiplatform.html#code-sharing-between-platforms

What is Kotlin Multiplatform?

Supported Platforms

- Android
- iOS
- Web
- Desktop



```
Project
CocoaHeads ~/Desktop/CocoaHeads
  .gradle
  .idea
  androidApp
    src
      main
        java
        res
        AndroidManifest.xml
    build.gradle.kts
  gradle
  iosApp
    iosApp
      Assets.xcassets
      Preview Content
      ContentView.swift
      Info.plist
      iOSApp.swift
    iosApp.xcodeproj
  shared
    src
      androidMain [main]
      commonMain
      iosMain
    build.gradle.kts
```

What is Kotlin Multiplatform Mobile?

What is Kotlin Multiplatform Mobile?

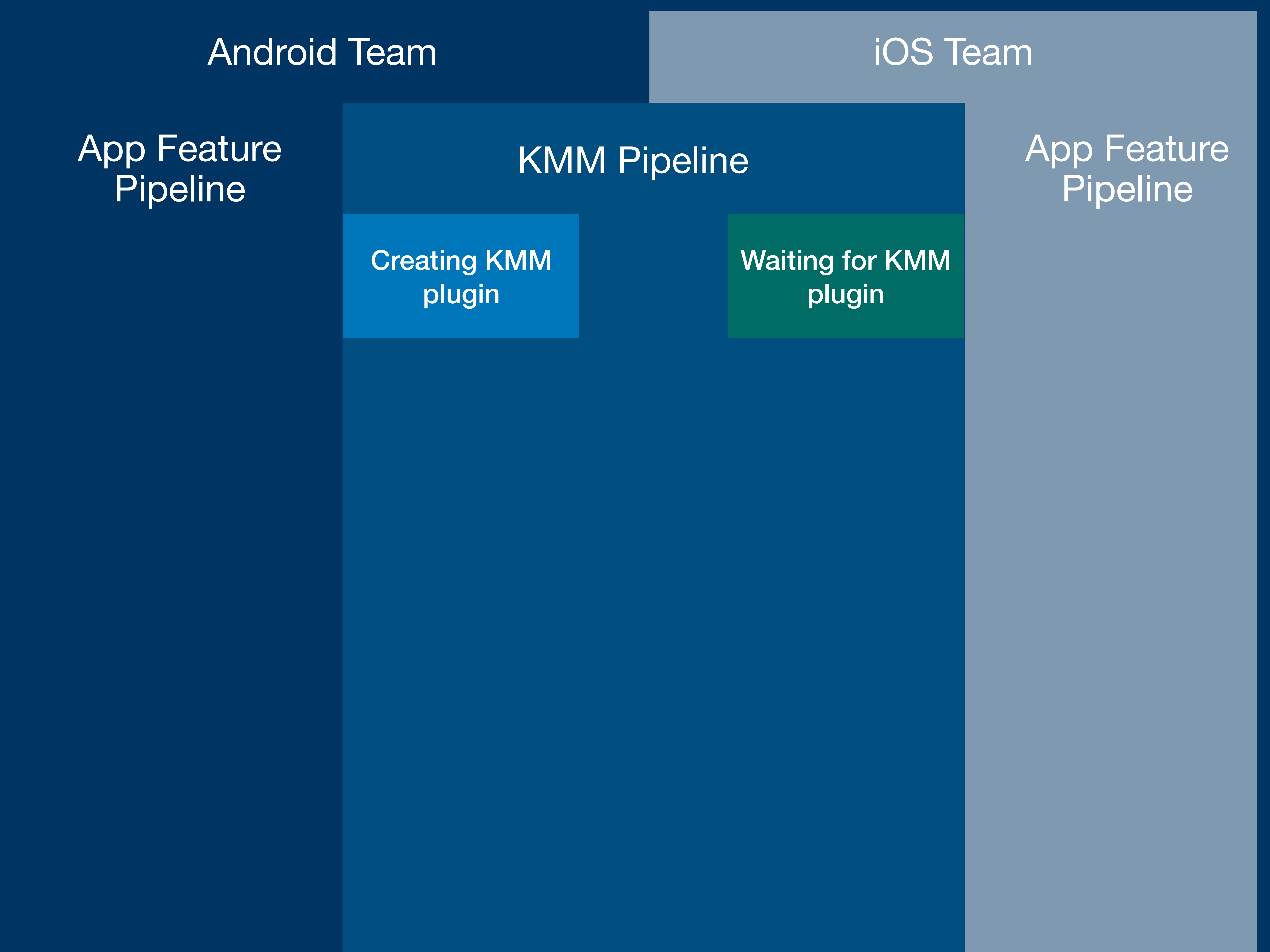
- Subset of KMP
- Targets only mobile platforms: Android and iOS
- Android Studio KMM plugin - run and debug iOS devices

What is Kotlin Multiplatform Mobile?

- Input: Write code in Kotlin in Android Studio
- Output: Generated binary XCFramework with Objective-C Code
- Integrate: Using SwiftPM or CocoaPods
- Flexible: Share the code you like (e.g. API client) or platform specific code (e.g. UI or CoreLocation)

But is this really true?

KMM in the release train



Android Team

iOS Team

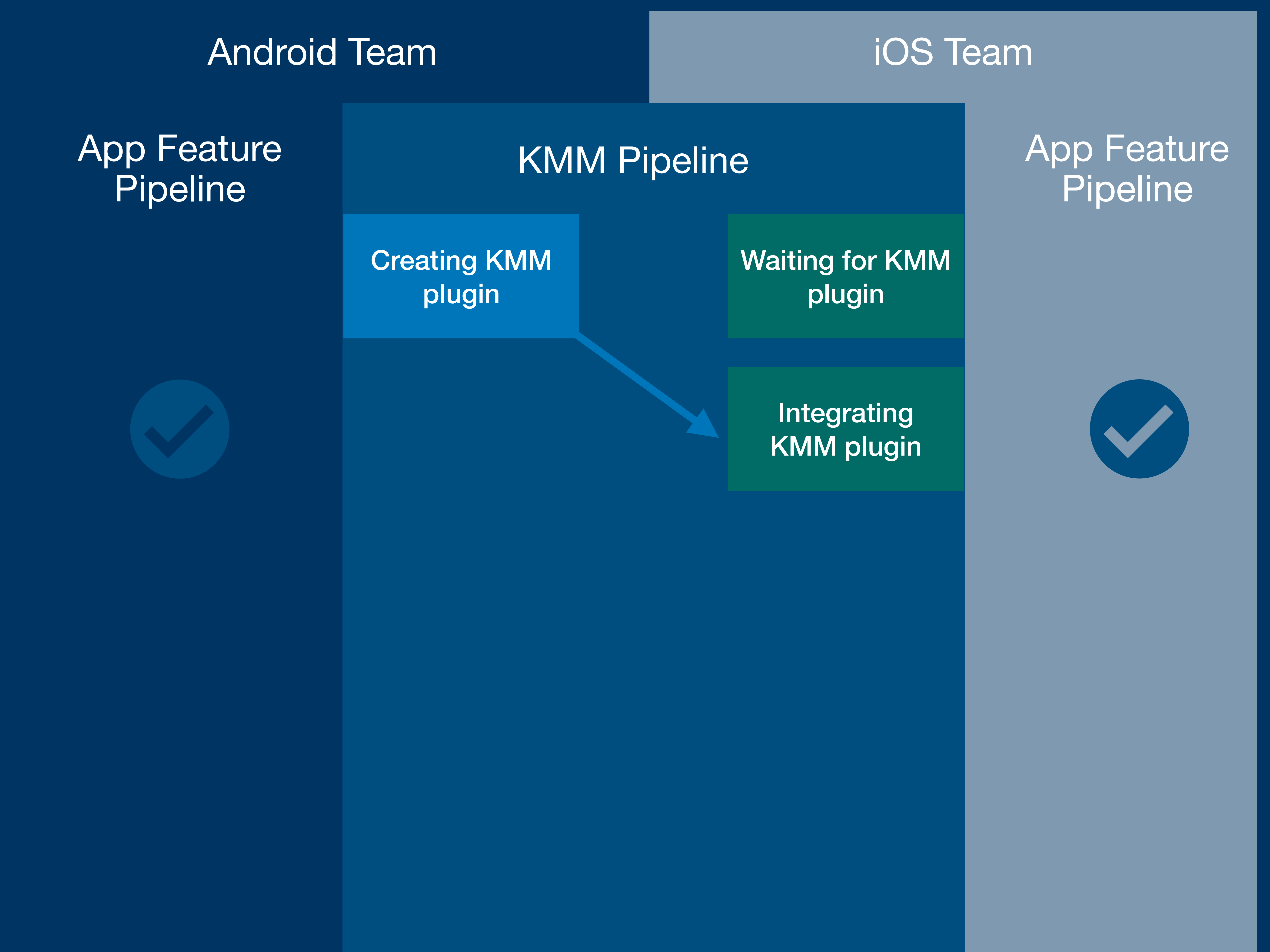
App Feature Pipeline

KMM Pipeline

App Feature Pipeline

Creating KMM plugin

Waiting for KMM plugin



Android Team

iOS Team

App Feature Pipeline

✓

KMM Pipeline

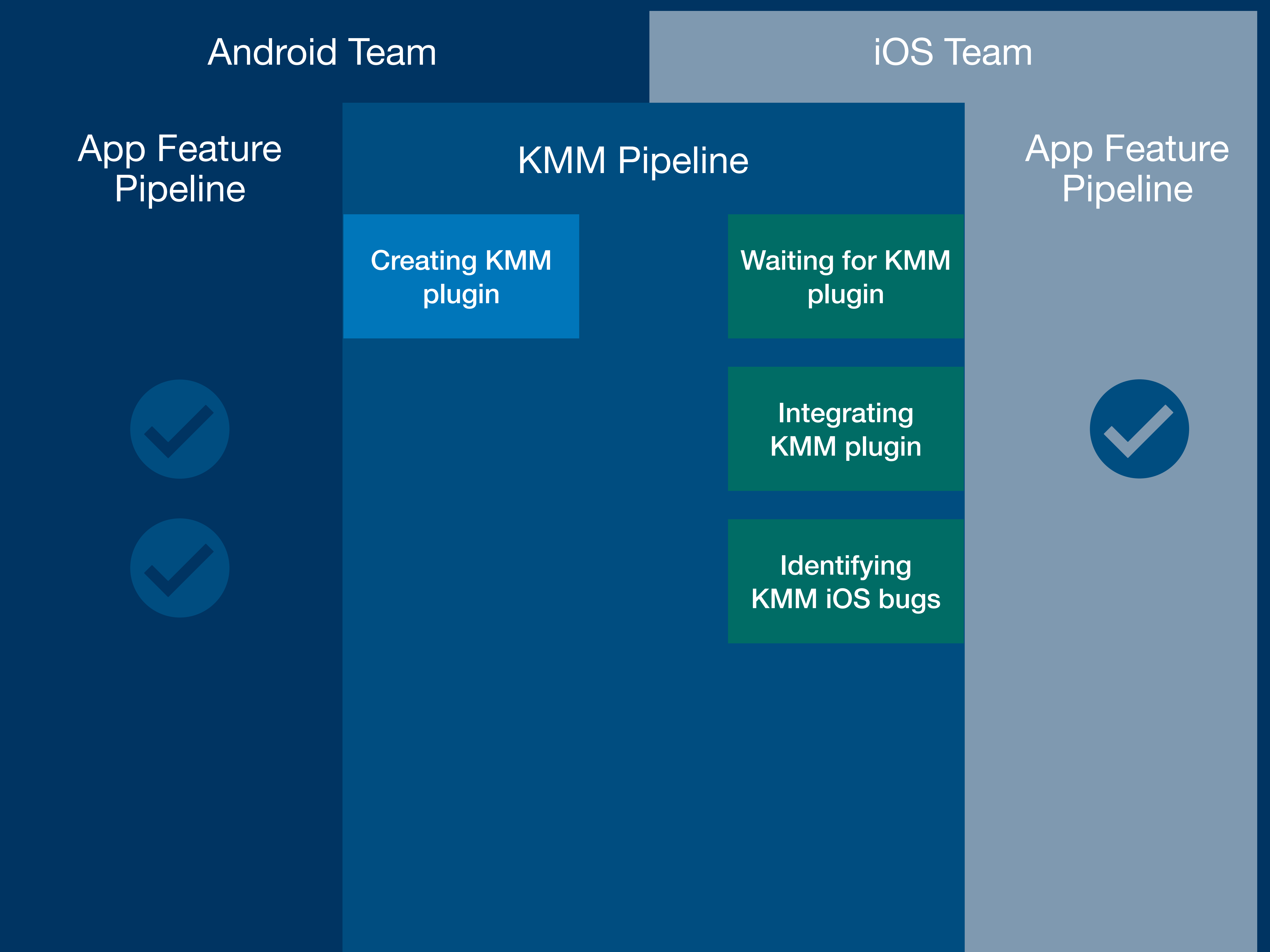
Creating KMM plugin

Waiting for KMM plugin

Integrating KMM plugin

App Feature Pipeline

✓



Android Team

iOS Team

App Feature Pipeline

✓

✓

KMM Pipeline

Creating KMM plugin

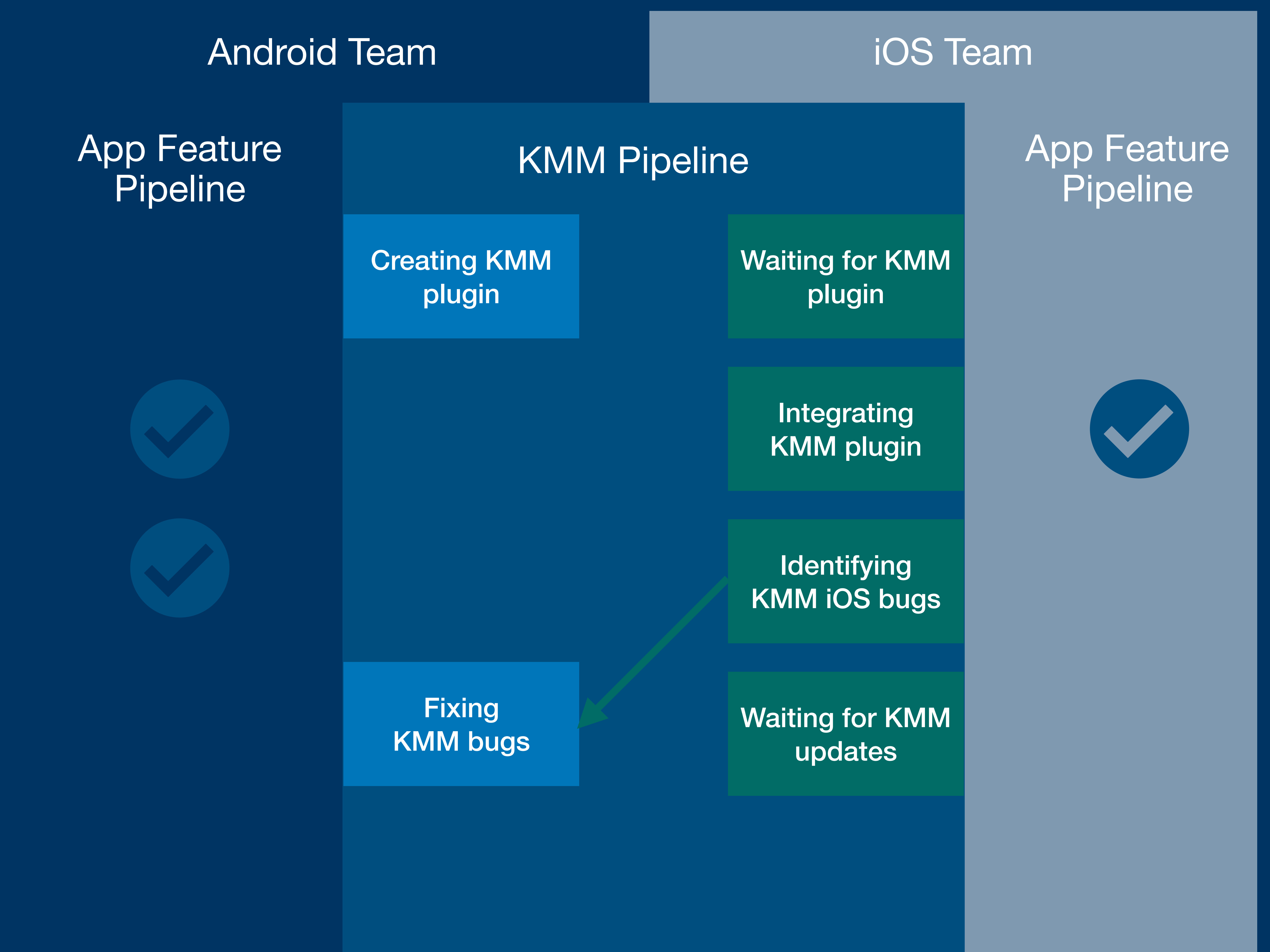
Waiting for KMM plugin

Integrating KMM plugin

Identifying KMM iOS bugs

App Feature Pipeline

✓



Android Team

iOS Team

App Feature Pipeline

✓

✓

KMM Pipeline

Creating KMM plugin

Waiting for KMM plugin

Integrating KMM plugin

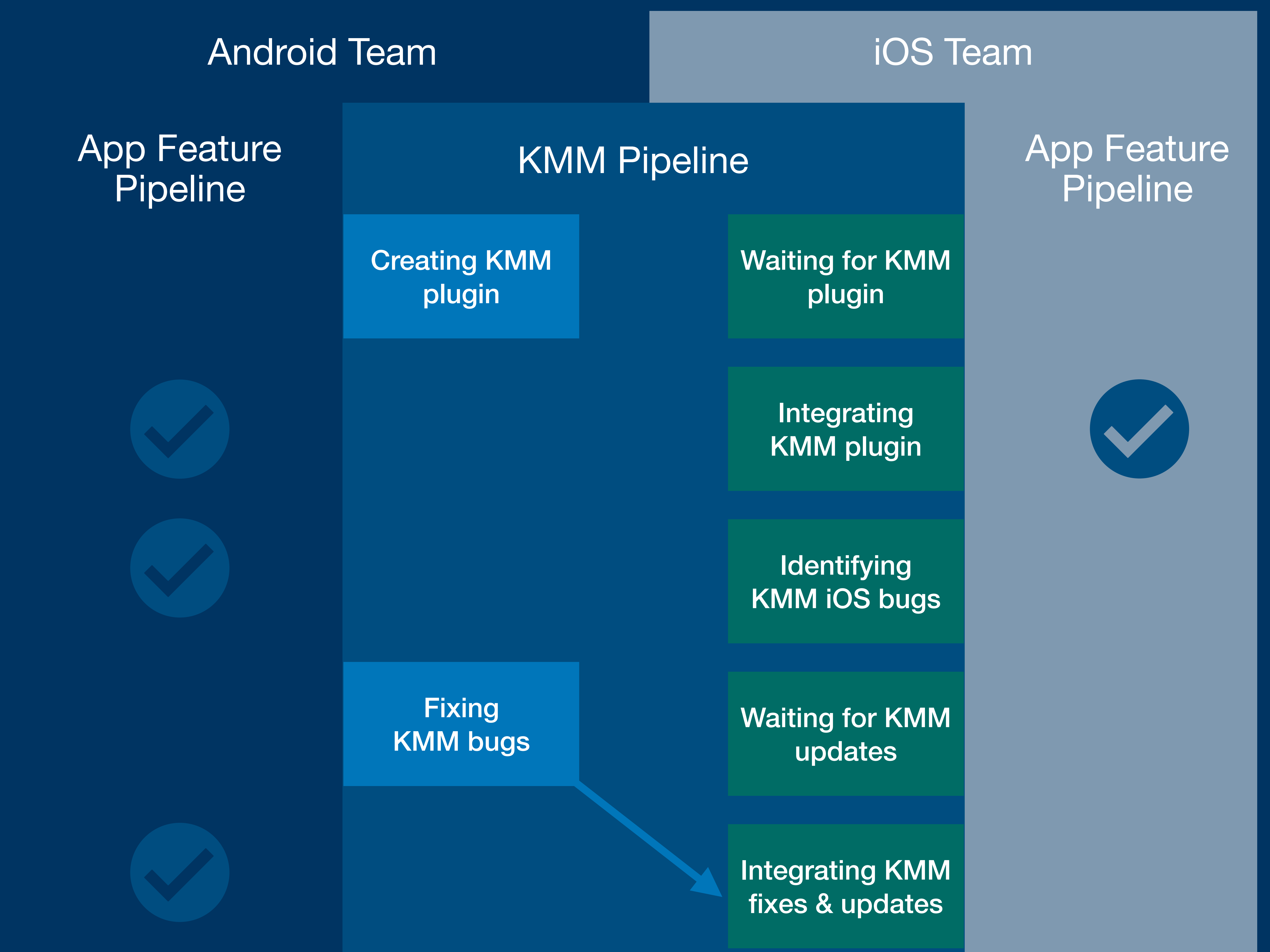
Identifying KMM iOS bugs

Fixing KMM bugs

Waiting for KMM updates

App Feature Pipeline

✓



Android Team

iOS Team

App Feature Pipeline

KMM Pipeline

App Feature Pipeline

Creating KMM plugin

Waiting for KMM plugin

✓

Integrating KMM plugin

✓

✓

Identifying KMM iOS bugs

Fixing KMM bugs

Waiting for KMM updates

✓

Integrating KMM fixes & updates

iOS is waiting & idling a lot and iOS features are behind

Demo time

Demo time

1. Install Android Studio & tools
   - In Android Studio, go to Preferences > Plugins.
   - Search for “Kotlin Multiplatform” and install the plugin.
2. Open Android Studio:
   - Click on “New Project”.
   - Select Kotlin Multiplatform App from the templates
   - commonMain: The shared module where the business logic or data models are written.
   - androidMain and iosMain: Platform-specific code (e.g., accessing platform APIs).
3. Adjust build.gradle.kts for XCFramework (see: https://kotlinlang.org/docs/multiplatform-build-native-binaries.html#build-xcframeworks)
4. Run in terminal to build the iOS framework: ./gradlew :<ProjectName>:assembleXCFramework

Pros and cons

Pros

- Write & test code only once / Code Reusability
- Can save time / Don’t solve problems twice
- One source of truth
- Native Performance: Optimizations for each platform.
- Flexibility: You can choose which parts of your application to share across platforms and which to write separately.
- Easy integration in Xcode - no environment setup hassle
- Generates real native code
- Kotlin similar to Swift - easy for iOS devs
- Team-building / cross functional team

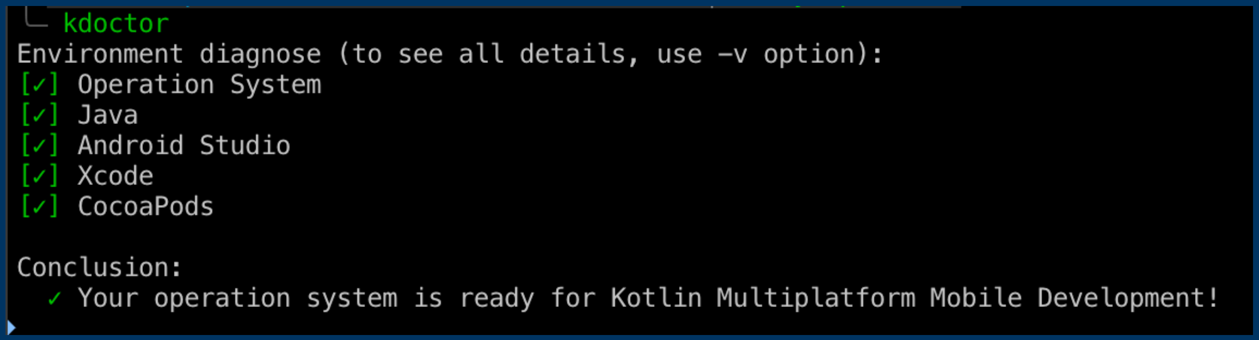
Cons

- Communication overhead - interface agreements
- Mostly Android developers write Kotlin code
- iOS development is mostly behind
- Another dependency for iOS
- Hard to debug - binary framework
- iOS only features hard to understand and debug for Android developers
- **Can introduce breaking changes and slow down iOS**
- No Swift features
- Version updates for Android studio are time consuming
- Android Studio unfamiliar for iOS developers

Tools

Tools

- Android Studio
  - Kotlin Multiplatform plugin
  - kdoctor
  - Java
- Xcode
  - Xcode-kotlin plugin (https://github.com/touchlab/xcode-kotlin)
    - Enables breakpoints and more debugging



```
└ kdoctor
Environment diagnose (to see all details, use -v option):
[✓] Operation System
[✓] Java
[✓] Android Studio
[✓] Xcode
[✓] CocoaPods

Conclusion:
  ✓ Your operation system is ready for Kotlin Multiplatform Mobile Development!
```

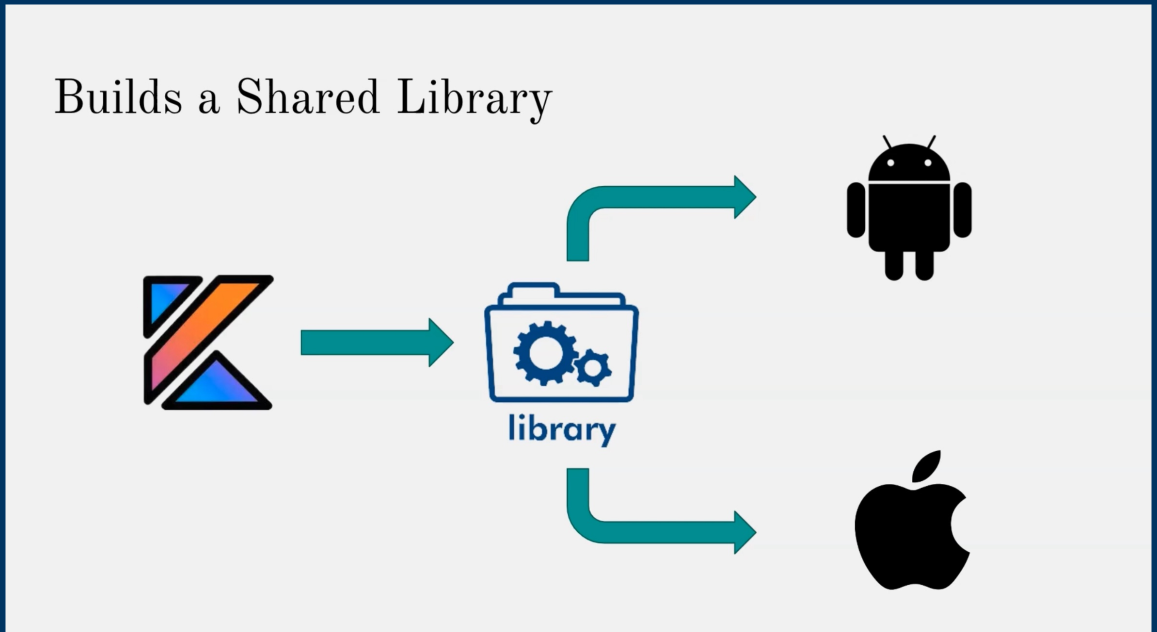
Advices

Advices

- iOS and Android team should work closely together
- Make clear interfaces / contracts
- Quicker and shorter iterations to avoid heavy breaking changes
- Consider the KMM framework as 3rd party SDK
- Focus on shared model, business logic, API Client
- Avoid deep UI integration

Advices

At the end it’s a layer between app and backend.



Q&A

Quellen

https://www.jetbrains.com/kotlin-multiplatform/

https://www.netguru.com/blog/kotlin-multiplatform-pros-and-cons

https://www.netguru.com/blog/kotlin-multiplatform-library

https://kotlinlang.org/docs/multiplatform-discover-project.html#compilation-to-a-specific-target

https://www.kodeco.com/books/kotlin-multiplatform-by-tutorials/v2.0/chapters/1-introduction

https://medium.com/@adrianwitaszak/creating-and-setting-up-kotlin-multiplatform-projects-for-android-and-ios-9cd288315111

https://touchlab.co/xcodekotlin