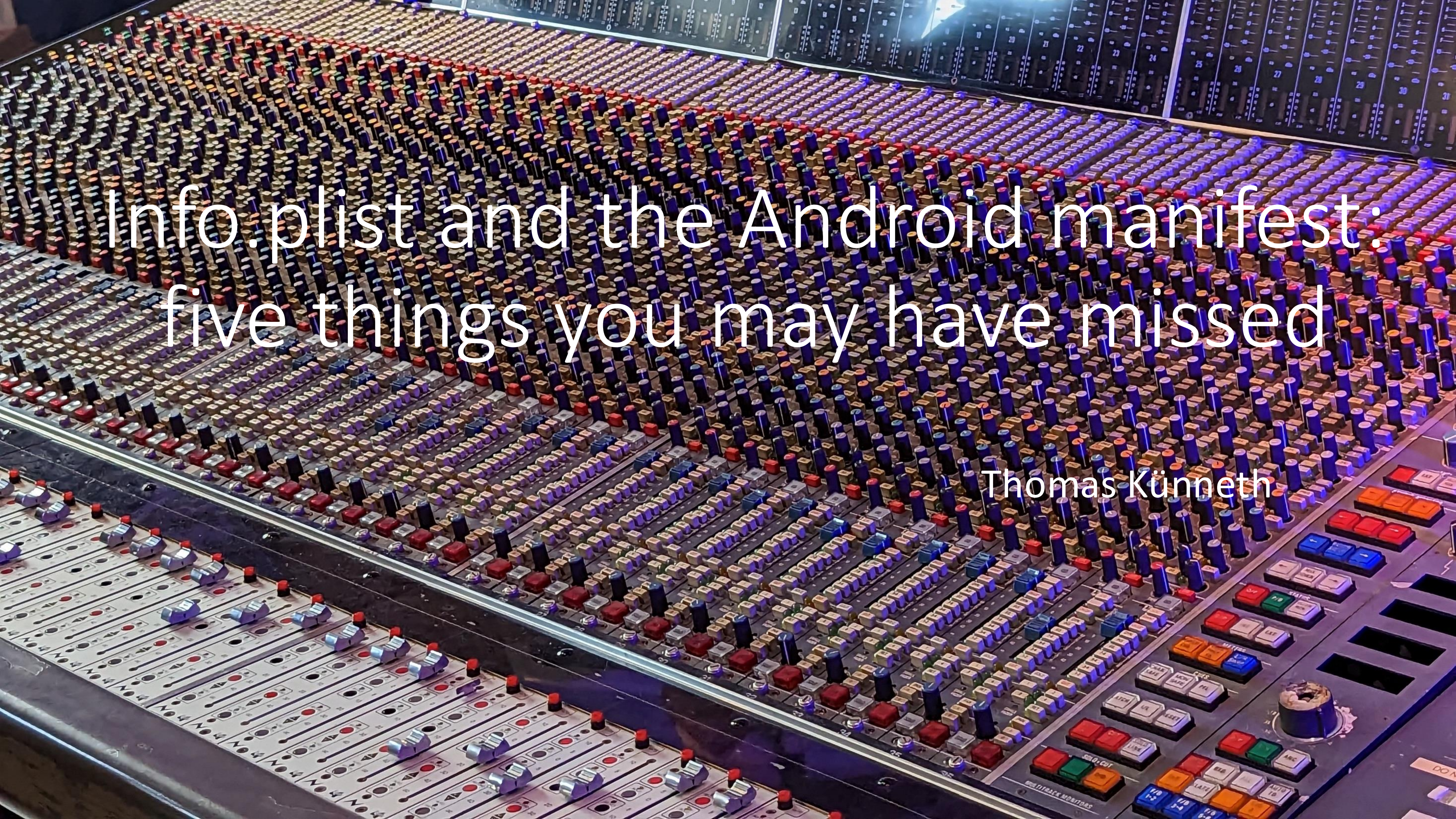
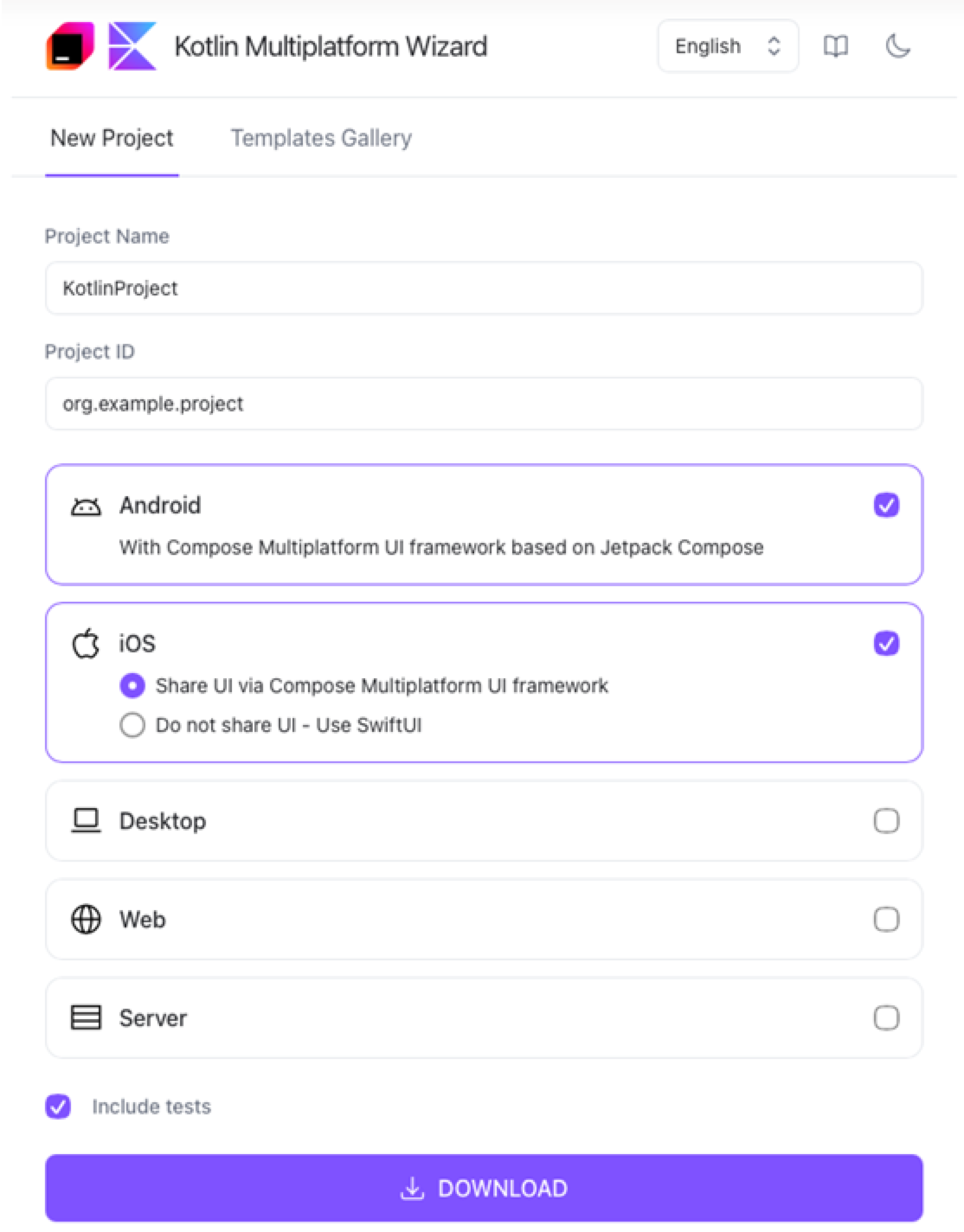


Info.plist and the Android manifest: five things you may have missed

Thomas Künneth

2 / 50

Experts



Kotlin Multiplatform Wizard

English

New Project | Templates Gallery

Project Name: KotlinProject

Project ID: org.example.project

[✓] Android
With Compose Multiplatform UI framework based on Jetpack Compose

[✓] iOS
● Share UI via Compose Multiplatform UI framework
○ Do not share UI - Use SwiftUI

[ ] Desktop

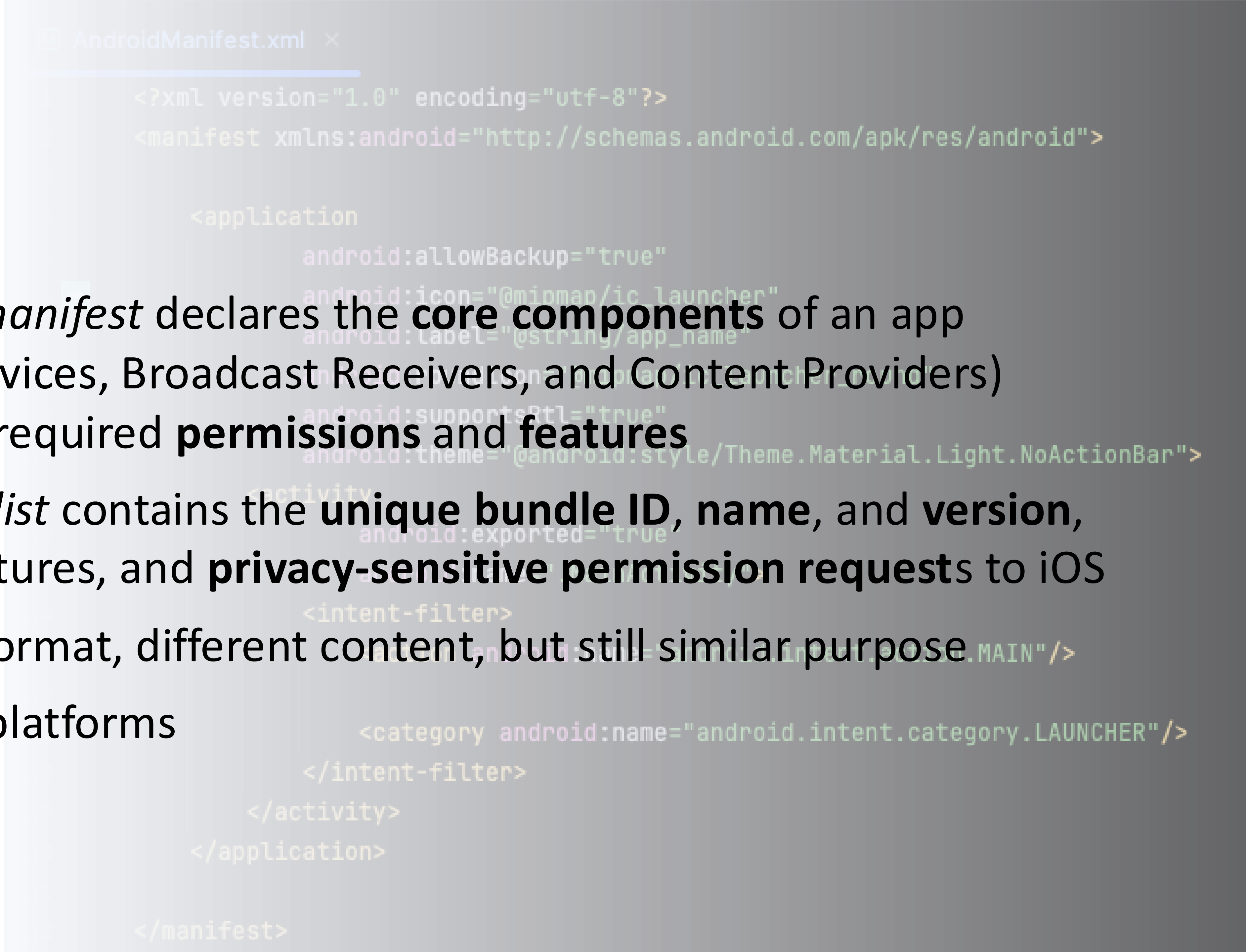
[ ] Web

[ ] Server

[✓] Include tests

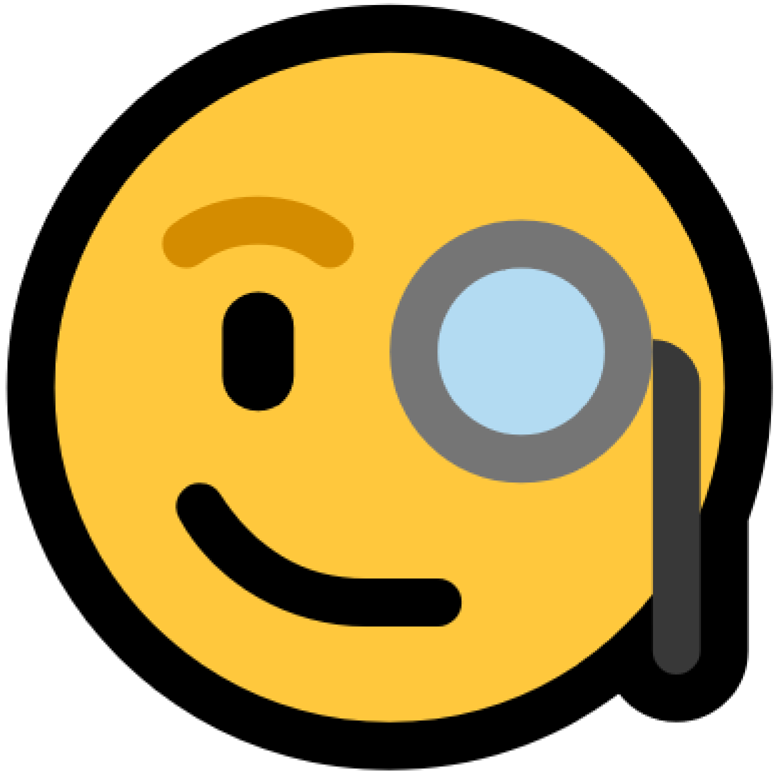
DOWNLOAD

- The *Android manifest* declares the **core components** of an app (Activities, Services, Broadcast Receivers, and Content Providers) along with all required **permissions** and **features**
- The *iOS Info.plist* contains the **unique bundle ID**, **name**, and **version**, supported features, and **privacy-sensitive permission requests** to iOS
- Different file format, different content, but still similar purpose
- Vital on both platforms


AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@android:style/Theme.Material.Light.NoActionBar">
        <activity
            android:exported="true"
            android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>

                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
    </application>

</manifest>
```

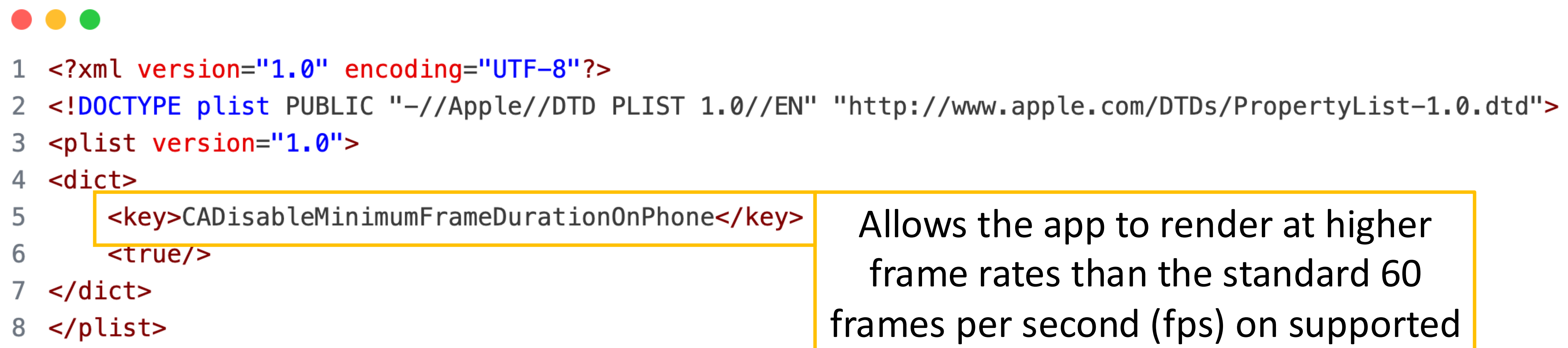
Experts



Who has made significant changes to any of the two files in their CMP/KMP projects?

Who believes their files not only meet the requirements but also contain everything that should be in?

Enhances visual performance, but may also lead to increased power consumption and higher CPU/GPU usage, potentially affecting battery life and device thermals



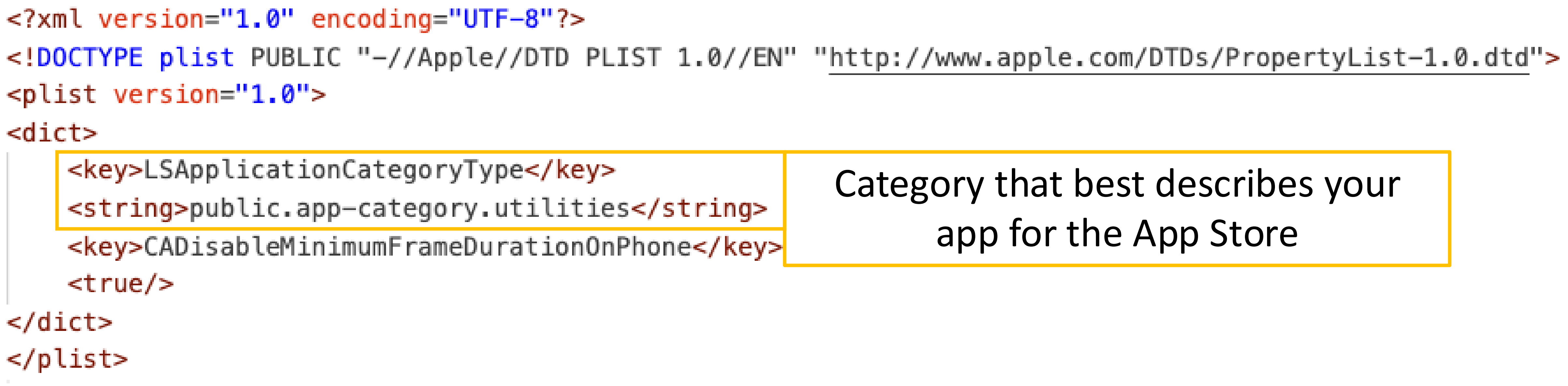
```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>CADisableMinimumFrameDurationOnPhone</key>
    <true/>
</dict>
</plist>
```

Allows the app to render at higher frame rates than the standard 60 frames per second (fps) on supported devices

https://github.com/JetBrains/compose-multiplatform/issues/3634

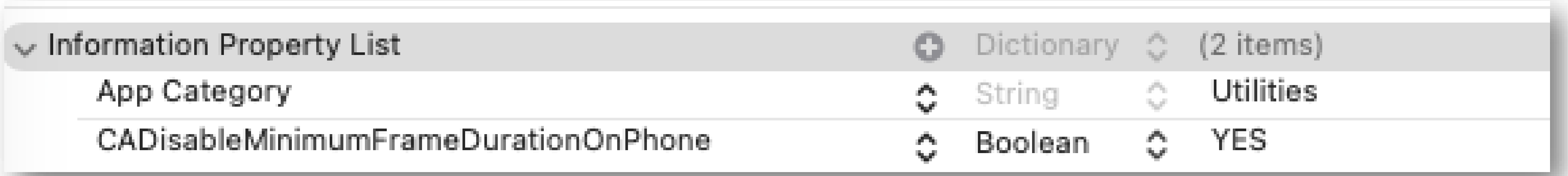
Experts

SnappMobile



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>LSApplicationCategoryType</key>
    <string>public.app-category.utilities</string>
    <key>CADisableMinimumFrameDurationOnPhone</key>
    <true/>
</dict>
</plist>
```

Category that best describes your app for the App Store



Information Property List	Dictionary	(2 items)
App Category	String	Utilities
CADisableMinimumFrameDurationOnPhone	Boolean	YES

`android:appCategory`

Categories are used to cluster multiple apps together into meaningful groups, such as when summarizing battery, network, or disk usage

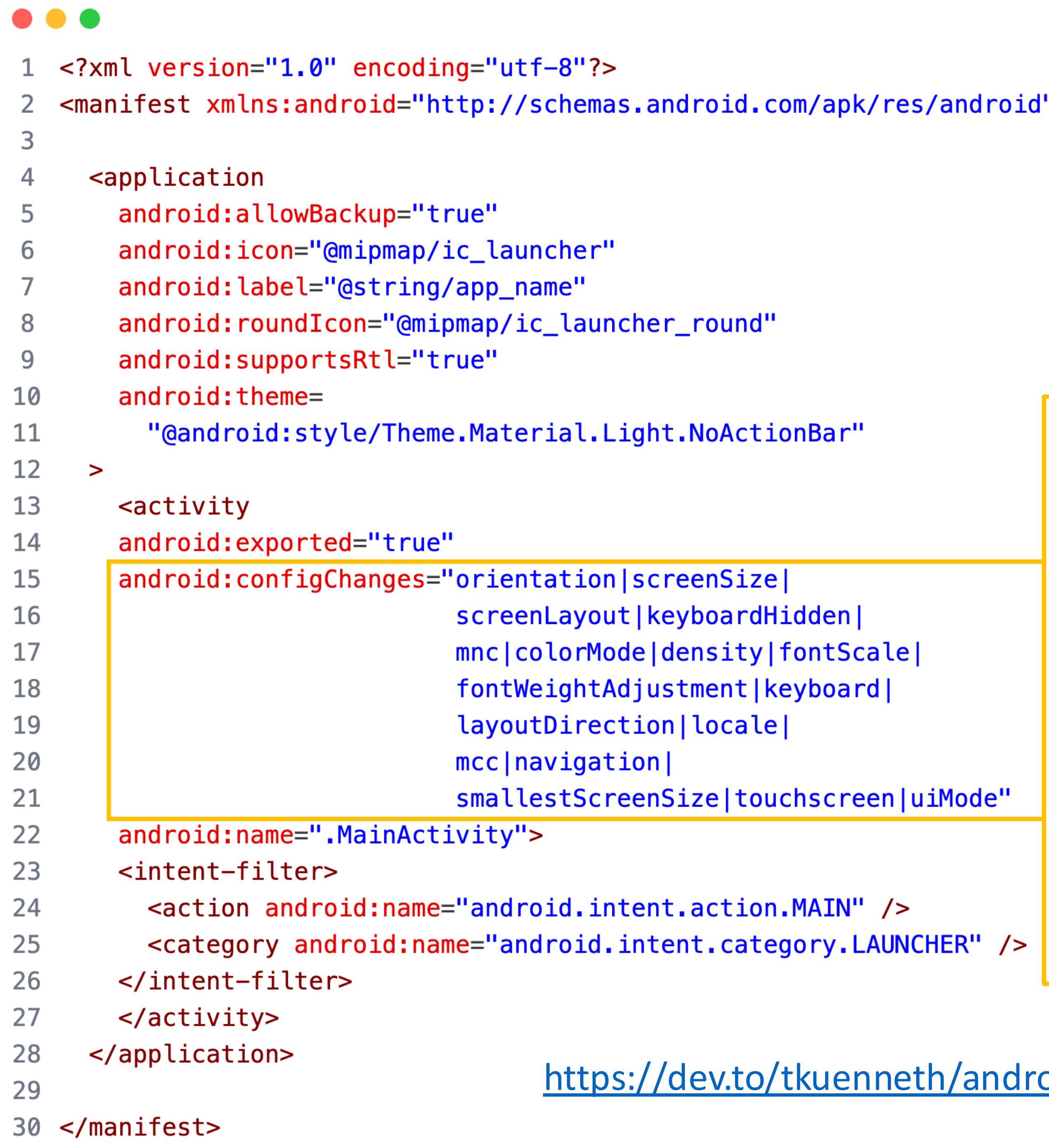
Only define this value for apps that fit well into one of the specific categories

1

Make informed decisions by understanding what a value means and why it is set

Decide if your app benefits from it being present

Not being present initially does not mean it’s not important



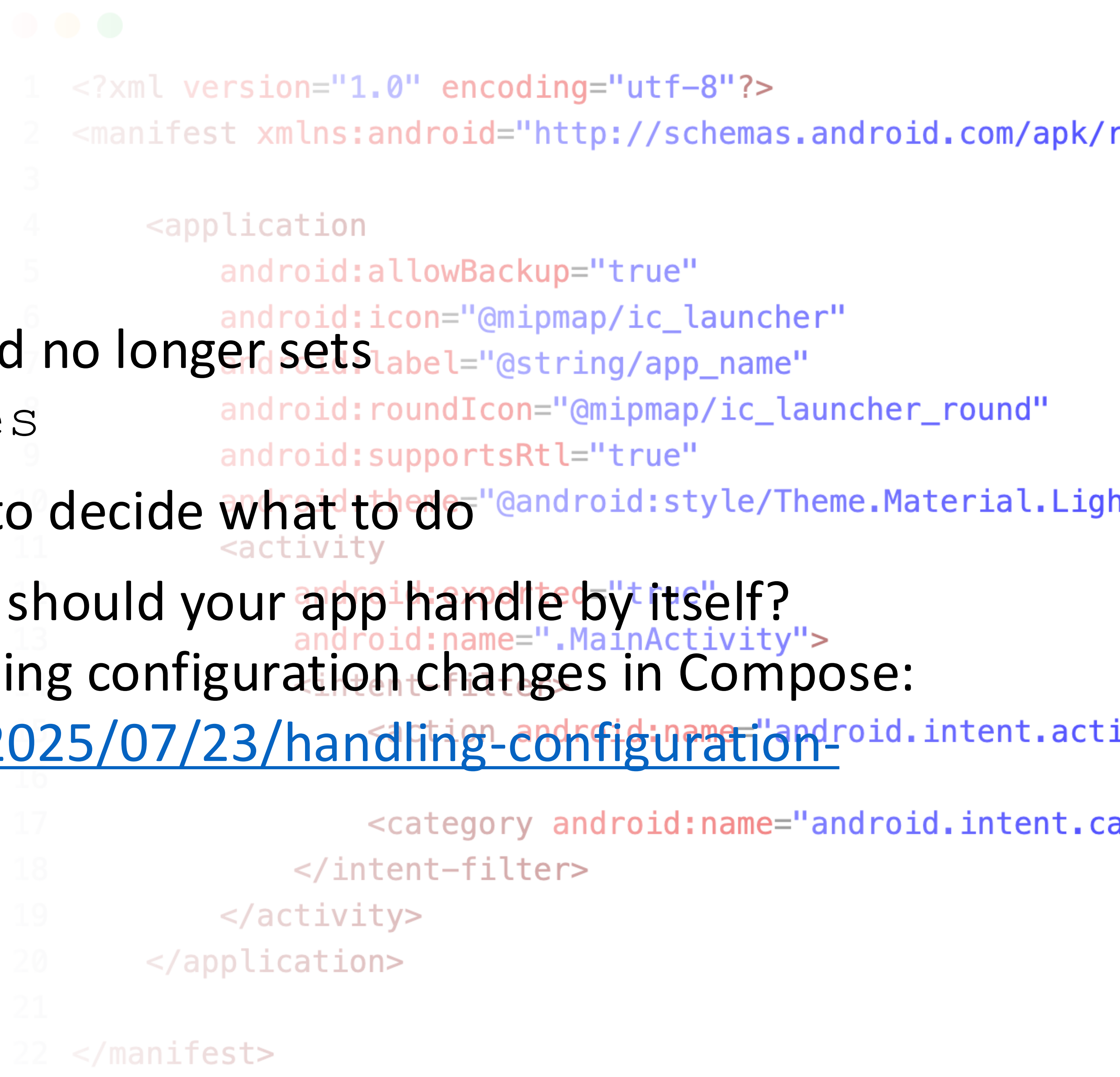
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">

  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme=
      "@android:style/Theme.Material.Light.NoActionBar"
  >
    <activity
    android:exported="true"
    android:configChanges="orientation|screenSize|
                           screenLayout|keyboardHidden|
                           mnc|colorMode|density|fontScale|
                           fontWeightAdjustment|keyboard|
                           layoutDirection|locale|
                           mcc|navigation|
                           smallestScreenSize|touchscreen|uiMode"
    android:name=".MainActivity">
    <intent-filter>
      <action android:name="android.intent.action.MAIN" />
      <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    </activity>
  </application>

</manifest>
```

Lists configuration changes that the activity handles itself. When a configuration change occurs at runtime, the activity shuts down and restarts by default, but declaring a configuration with this attribute prevents the activity from restarting. Instead, the activity remains running and its `onConfigurationChanged()` method is called.

https://dev.to/tkuenneth/android-manifest-file-checklist-4ne2

SnappMobile

- Current version of KMP wizard no longer sets `android:configChanges`
- Responsibility of the project to decide what to do
- Which configuration changes should your app handle by itself?
  See talk by Alex Vanyo: Handling configuration changes in Compose:
  https://www.droidcon.com/2025/07/23/handling-configuration-changes-in-compose/



```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/r

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@android:style/Theme.Material.Ligh
        <activity
            android:exported="true"
            android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.acti

                <category android:name="android.intent.ca
            </intent-filter>
        </activity>
    </application>

</manifest>
```

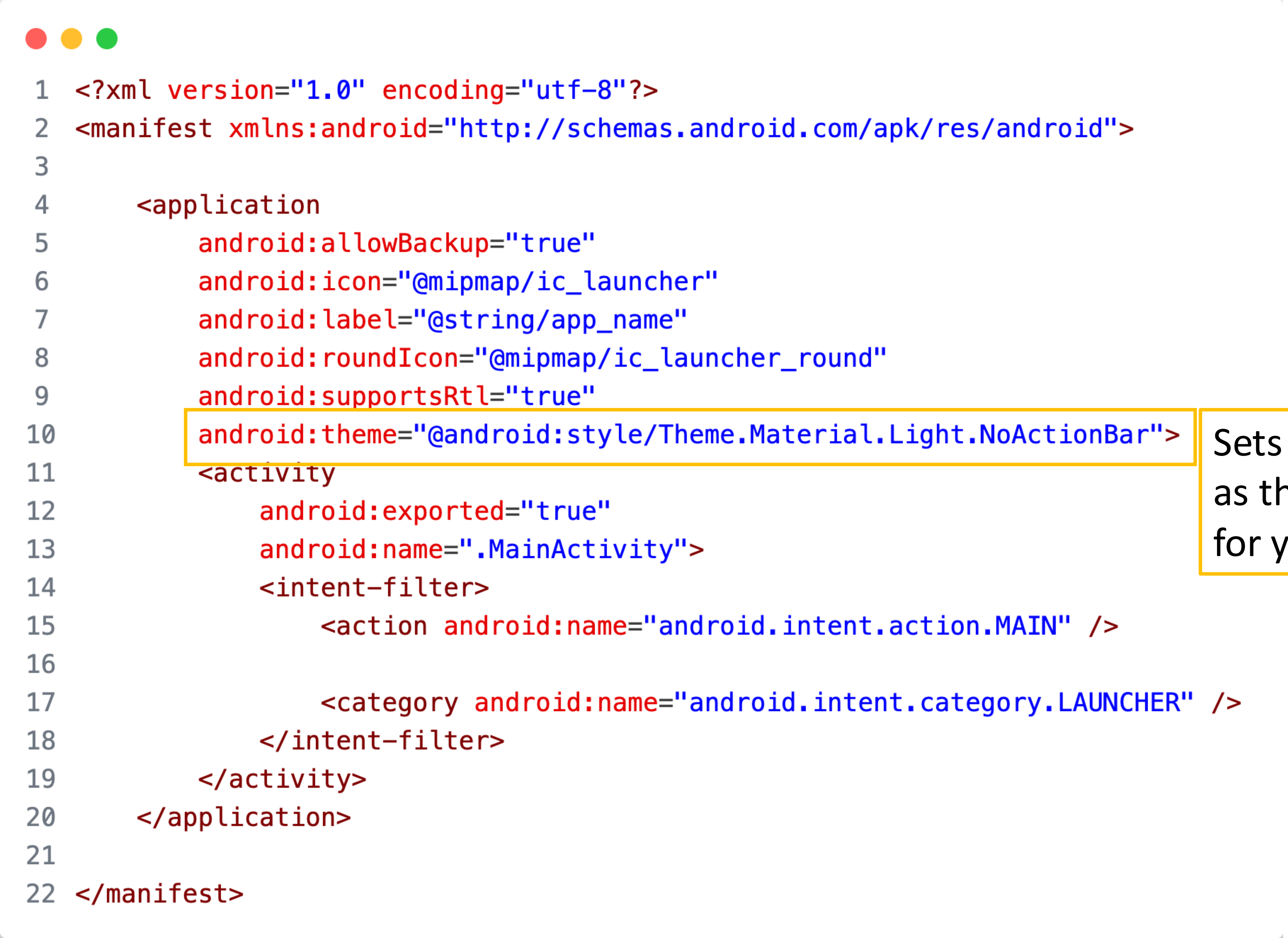
Experts

2

Handling configuration changes on Android require careful consideration

Only you know what your app needs

It’s not up to the KMP wizard to provide a complete or bullet-prof manifest / Info.plist – just a start



```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@android:style/Theme.Material.Light.NoActionBar">
        <activity
            android:exported="true"
            android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Sets a light theme as the initial theme for your app

SnappMobile

- Jetpack Compose is great at handling themes including light and dark mode
- Resouce-based themes are still relevant for
    - system-provided splash screens
    - embedded `AndroidViews`
- Without a change, the splashscreen background will appear in light mode
- Many solutions, a simple one is including Jetpack Appcompat and setting `android:theme="@style/Theme.AppCompat.DayNight.NoActionBar`

- On iOS, you can set the app's default light or dark mode behavior using `UIUserInterfaceStyle` (Appearance in Xcode)
- Possible values are `Light`, `Dark`, `Automatic` (default when not set)

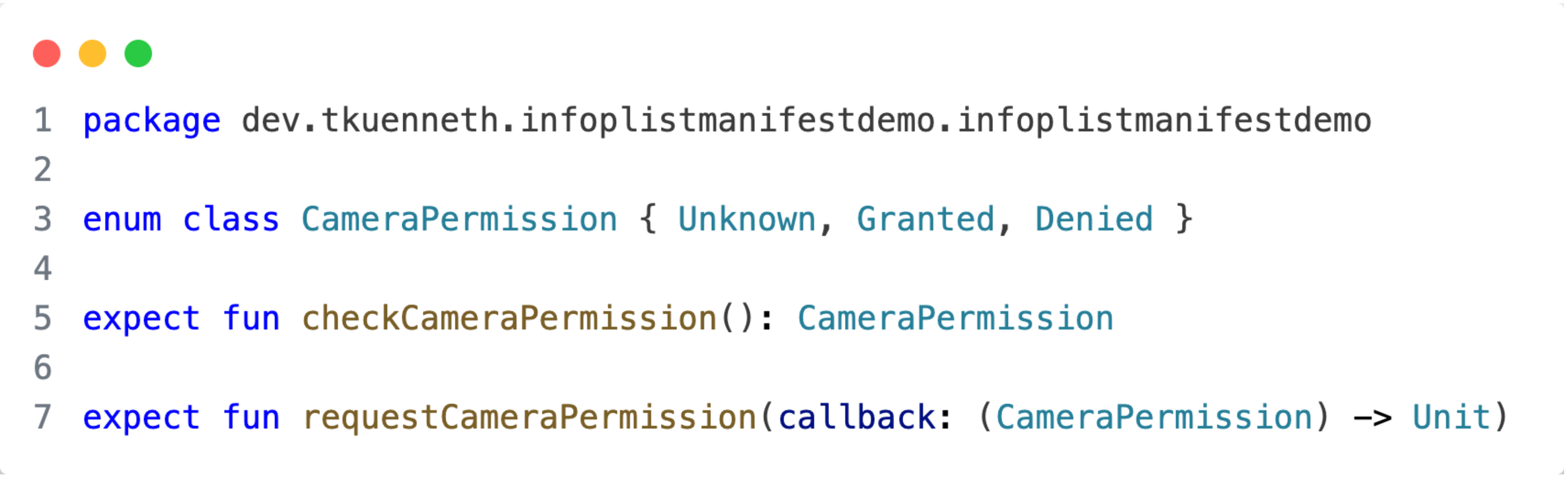
3

Jetpack Compose makes a lot of things way easier (configuration changes, themes / styles)

Still, your app may have not immediately obvious ties to the old world (splash screens, embedded `AndroidViews`)

Make sure to reflect them properly in your manifest file

- Similar to Android, certain actions like accessing files, contacts, calendars, or the camera, can be granted or denied by the user on iOS
- Like on Android, requesting a permission is done during runtime
- A description describing why the permission is requested, must be present in Info.plist

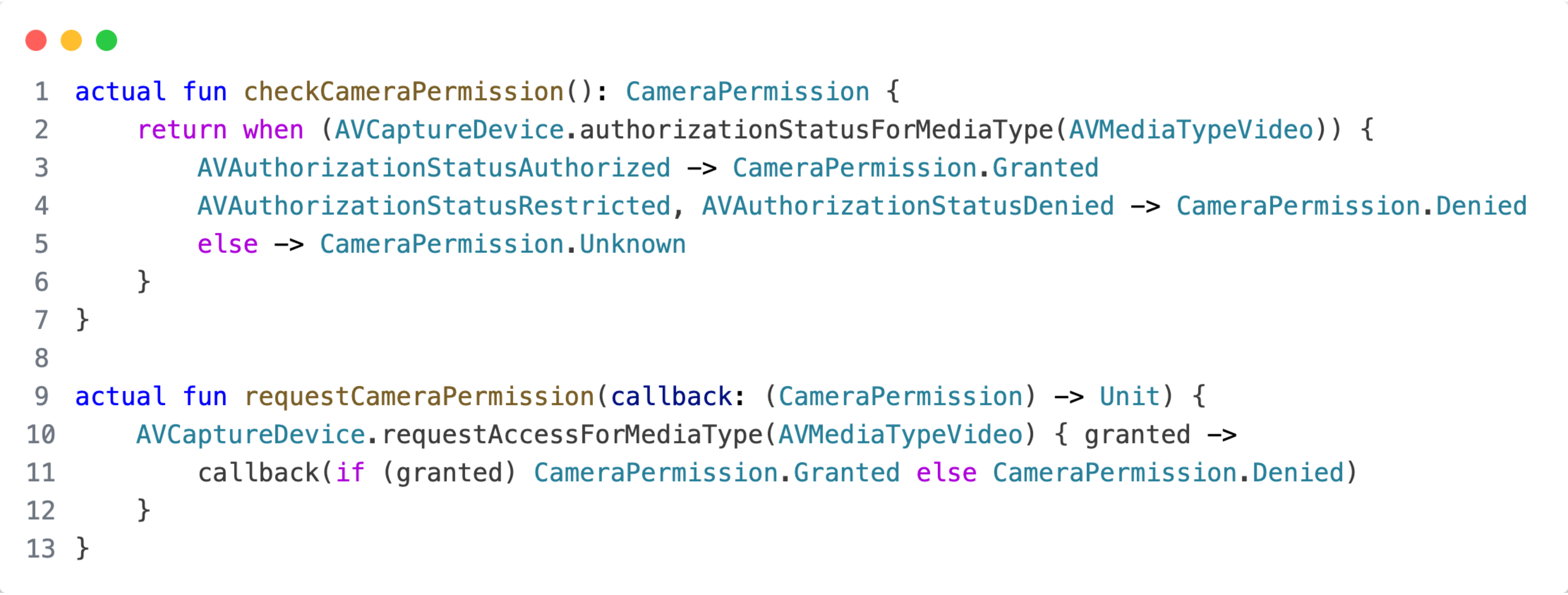


```kotlin
package dev.tkuenneth.infoplistmanifestdemo.infoplistmanifestdemo

enum class CameraPermission { Unknown, Granted, Denied }

expect fun checkCameraPermission(): CameraPermission

expect fun requestCameraPermission(callback: (CameraPermission) -> Unit)
```

https://github.com/tkuenneth/InfoPlistManifestDemo

Experts

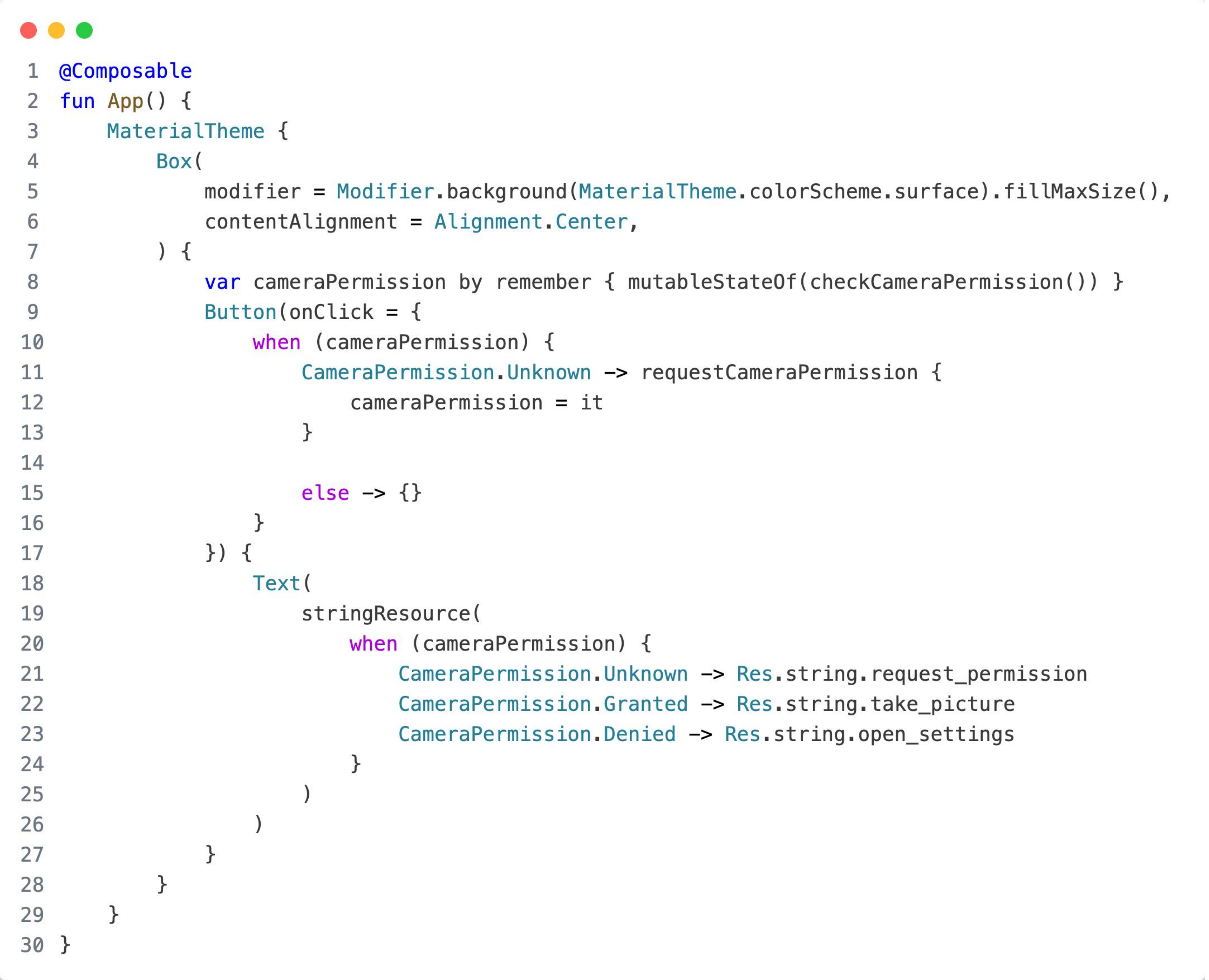
SnappMobile



```kotlin
actual fun checkCameraPermission(): CameraPermission {
    return when (AVCaptureDevice.authorizationStatusForMediaType(AVMediaTypeVideo)) {
        AVAuthorizationStatusAuthorized -> CameraPermission.Granted
        AVAuthorizationStatusRestricted, AVAuthorizationStatusDenied -> CameraPermission.Denied
        else -> CameraPermission.Unknown
    }
}

actual fun requestCameraPermission(callback: (CameraPermission) -> Unit) {
    AVCaptureDevice.requestAccessForMediaType(AVMediaTypeVideo) { granted ->
        callback(if (granted) CameraPermission.Granted else CameraPermission.Denied)
    }
}
```

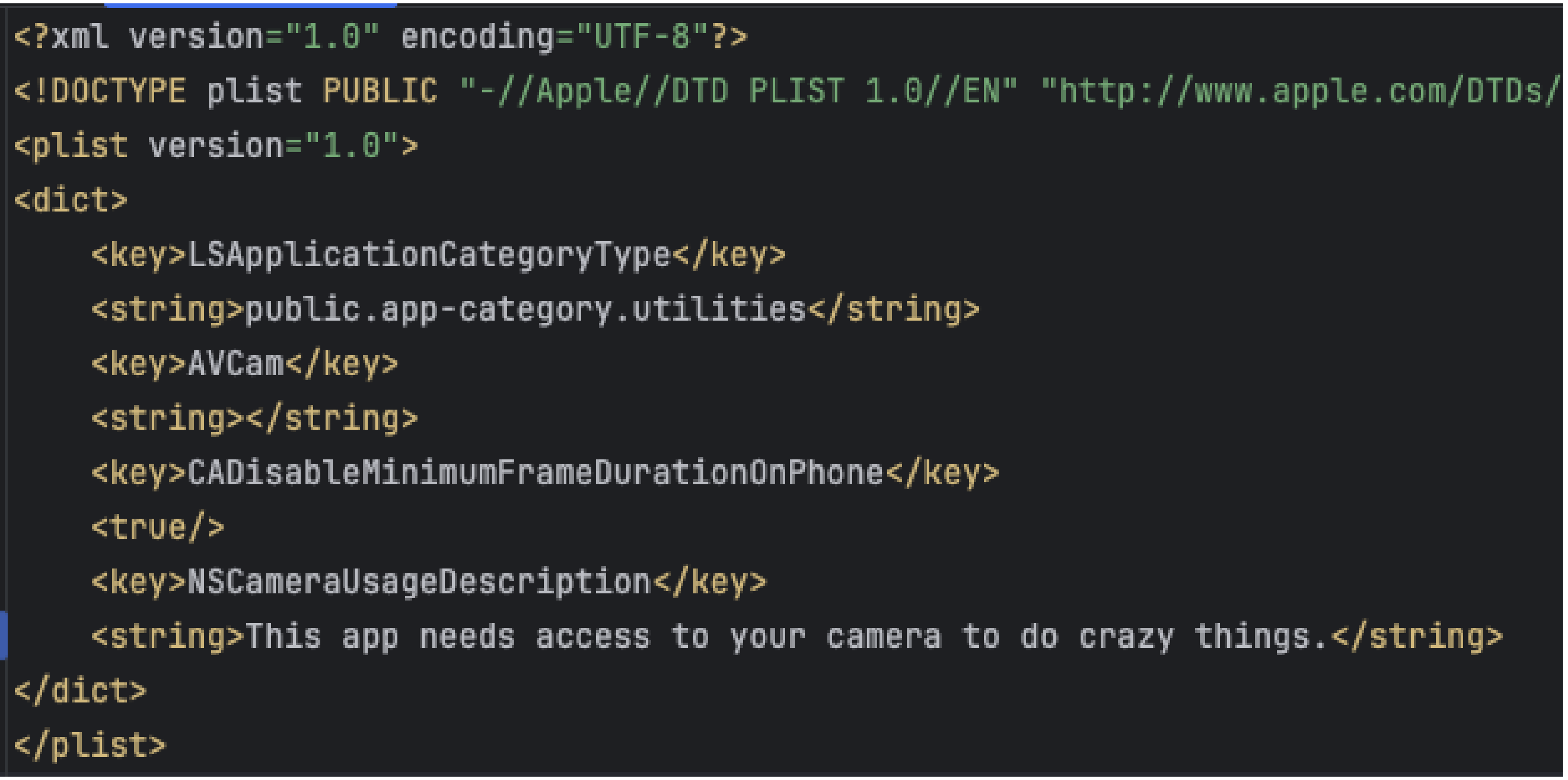
Experts

SnappMobile

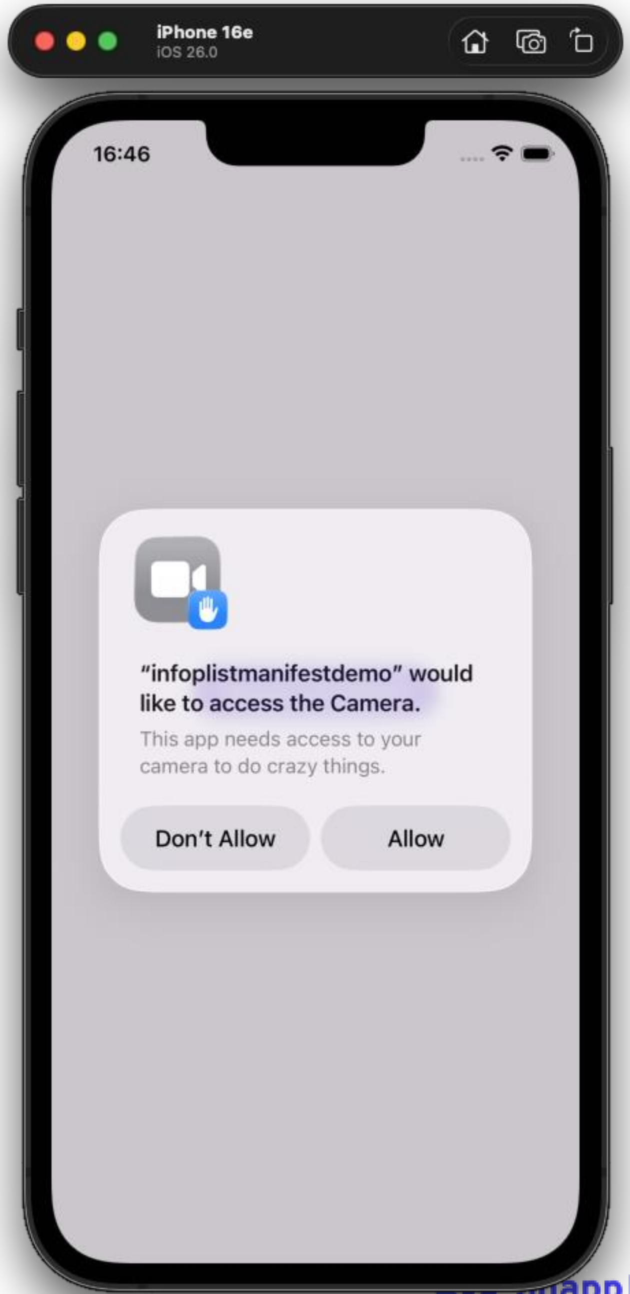


```kotlin
@Composable
fun App() {
    MaterialTheme {
        Box(
            modifier = Modifier.background(MaterialTheme.colorScheme.surface).fillMaxSize(),
            contentAlignment = Alignment.Center,
        ) {
            var cameraPermission by remember { mutableStateOf(checkCameraPermission()) }
            Button(onClick = {
                when (cameraPermission) {
                    CameraPermission.Unknown -> requestCameraPermission {
                        cameraPermission = it
                    }

                    else -> {}
                }
            }) {
                Text(
                    stringResource(
                        when (cameraPermission) {
                            CameraPermission.Unknown -> Res.string.request_permission
                            CameraPermission.Granted -> Res.string.take_picture
                            CameraPermission.Denied -> Res.string.open_settings
                        }
                    )
                )
            }
        }
    }
}
```

SnappMobile



```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
<plist version="1.0">
<dict>
    <key>LSApplicationCategoryType</key>
    <string>public.app-category.utilities</string>
    <key>AVCam</key>
    <string></string>
    <key>CADisableMinimumFrameDurationOnPhone</key>
    <true/>
    <key>NSCameraUsageDescription</key>
    <string>This app needs access to your camera to do crazy things.</string>
</dict>
</plist>
```



"infoplistmanifestdemo" would like to access the Camera.

This app needs access to your camera to do crazy things.

Experts

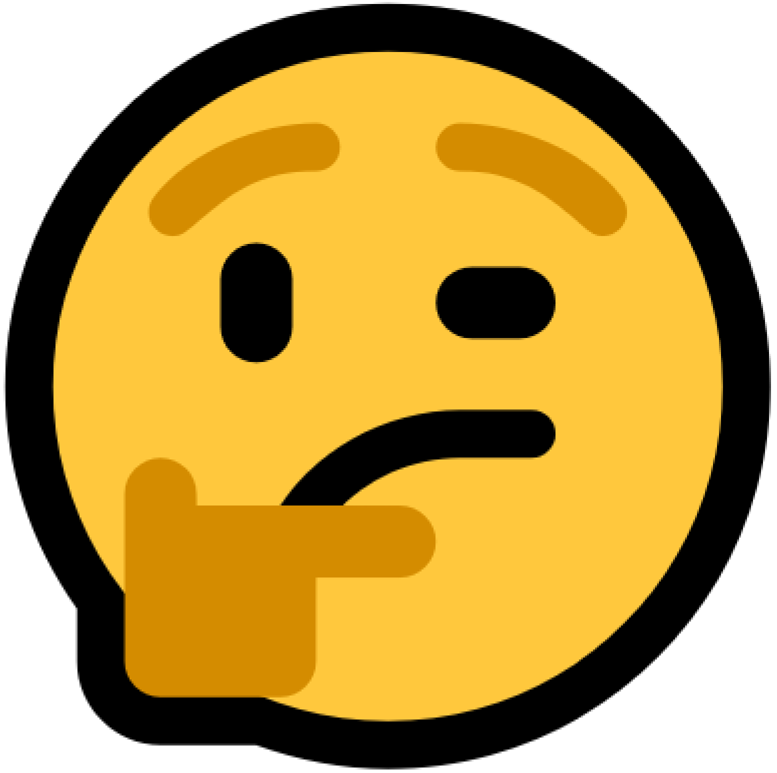
SnappMobile

4

Info.plist contains messages for permission dialogs

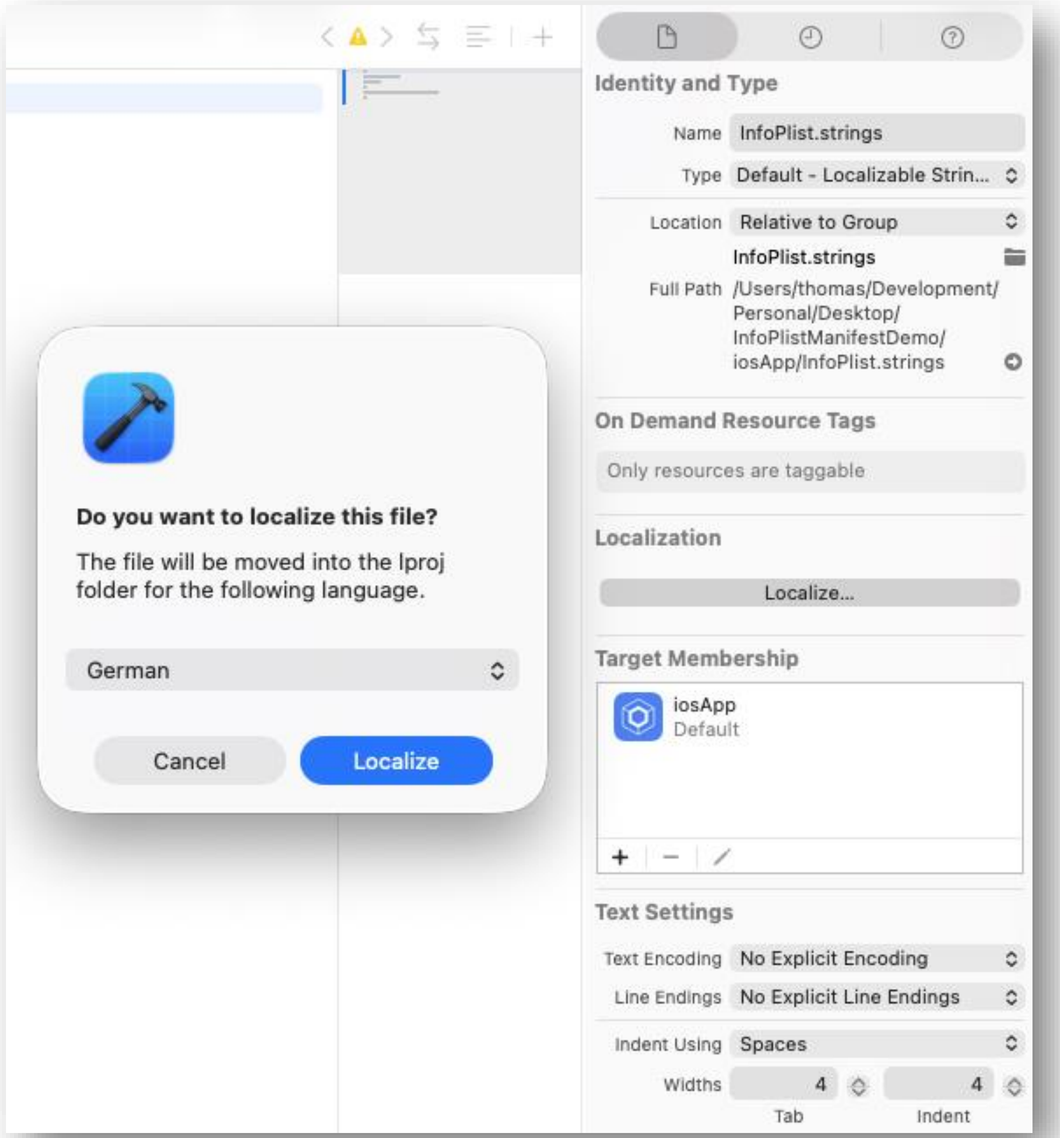
They must be present; otherwise the app will crash

Like on Android, clearly and honestly state the reason for requesting a permission



What about internationalization?

- Step 1: Add languages to your project
- Step 2: Create an InfoPlist.strings file
- Step 3: Localize this file
- Step 4: Add your translated messages



Identity and Type

Name InfoPlist.strings

Type Default - Localizable Strin...

Location Relative to Group

InfoPlist.strings

Full Path /Users/thomas/Development/Personal/Desktop/InfoPlistManifestDemo/iosApp/InfoPlist.strings

On Demand Resource Tags

Only resources are taggable

Localization

Localize...

Target Membership

iosApp Default

Text Settings

Text Encoding No Explicit Encoding

Line Endings No Explicit Line Endings

Indent Using Spaces

Widths 4 4

Tab Indent

Do you want to localize this file?

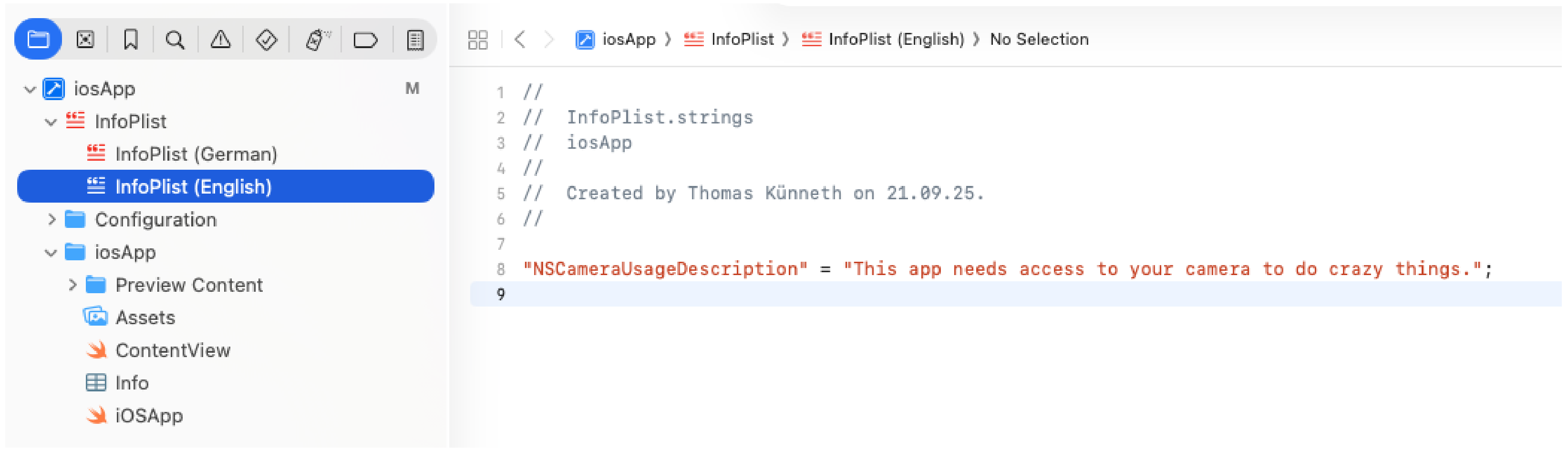
The file will be moved into the lproj folder for the following language.

German

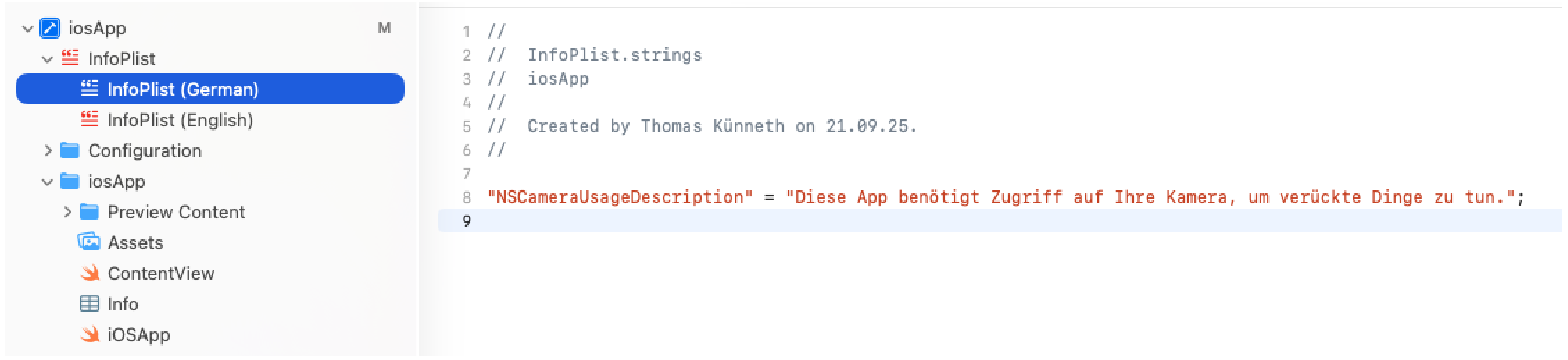
Cancel Localize

Experts

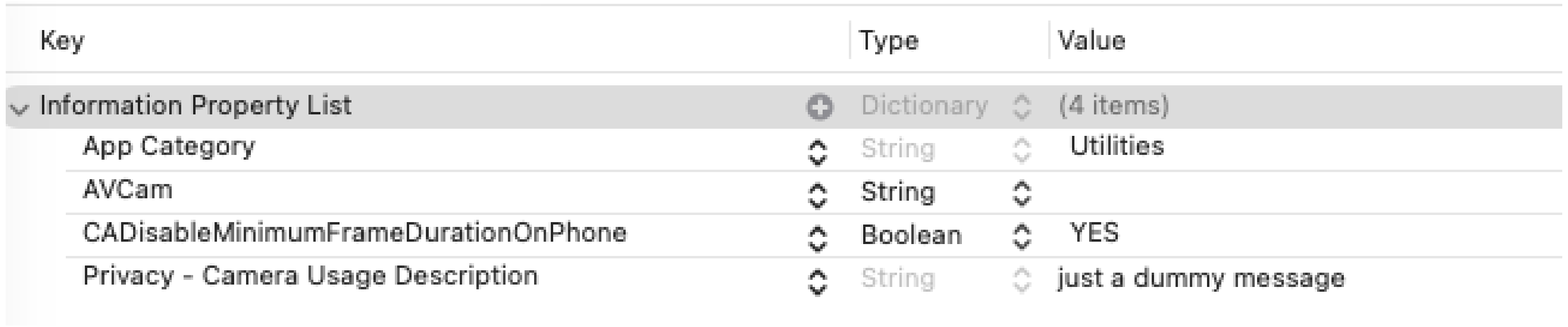
SnappMobile



```
//
//  InfoPlist.strings
//  iosApp
//
//  Created by Thomas Künneth on 21.09.25.
//

"NSCameraUsageDescription" = "This app needs access to your camera to do crazy things.";

```



```
//
//  InfoPlist.strings
//  iosApp
//
//  Created by Thomas Künneth on 21.09.25.
//

"NSCameraUsageDescription" = "Diese App benötigt Zugriff auf Ihre Kamera, um verückte Dinge zu tun.";

```



Key	Type	Value
Information Property List	Dictionary	(4 items)
App Category	String	Utilities
AVCam	String	
CADisableMinimumFrameDurationOnPhone	Boolean	YES
Privacy - Camera Usage Description	String	just a dummy message

Experts

SnappMobile

5

Texts in Info.plist can be internationalized

Thank you!

@tkuenneth

@tkuenneth

@tkuenneth.dev

https://github.com/tkuenneth/InfoPlistManifestDemo

Experts

SnappMobile