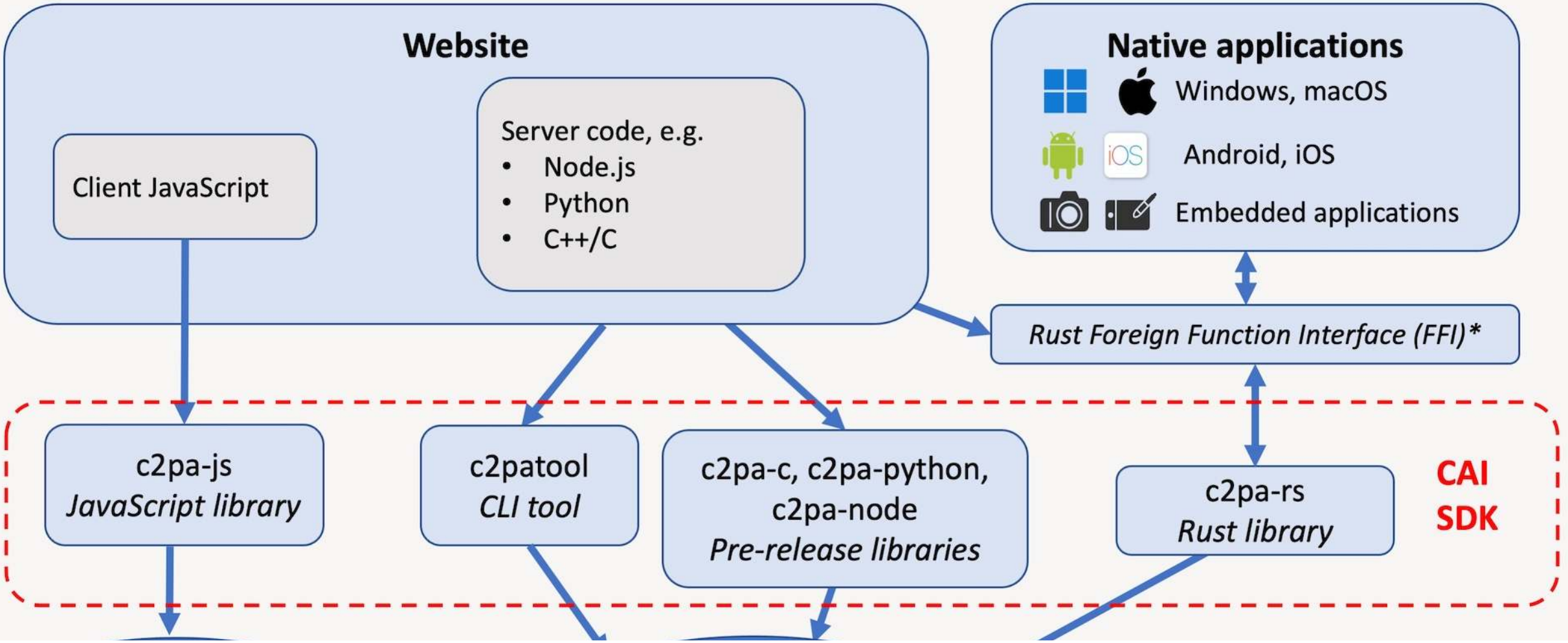


SimpleC2PA Library for Mobile Developers

An easy solution from the ProofMode team for mobile apps to add signed C2PA actions, claims, and attestations to media files

What is C2PA?

SECURE AND TRANSPARENT METADATA EMBEDDING

C2PA is a standard that enables secure and transparent embedding of metadata and provenance information into digital media files.

PROVENANCE TRACKING FOR DIGITAL CONTENT

C2PA provides a way to track the provenance and history of digital content, ensuring its authenticity and integrity.

DEVELOPED BY A CONSORTIUM OF INDUSTRY LEADERS

The C2PA standard was developed by a consortium of industry leaders, including Adobe, Microsoft, and others, to address the need for secure content attribution.

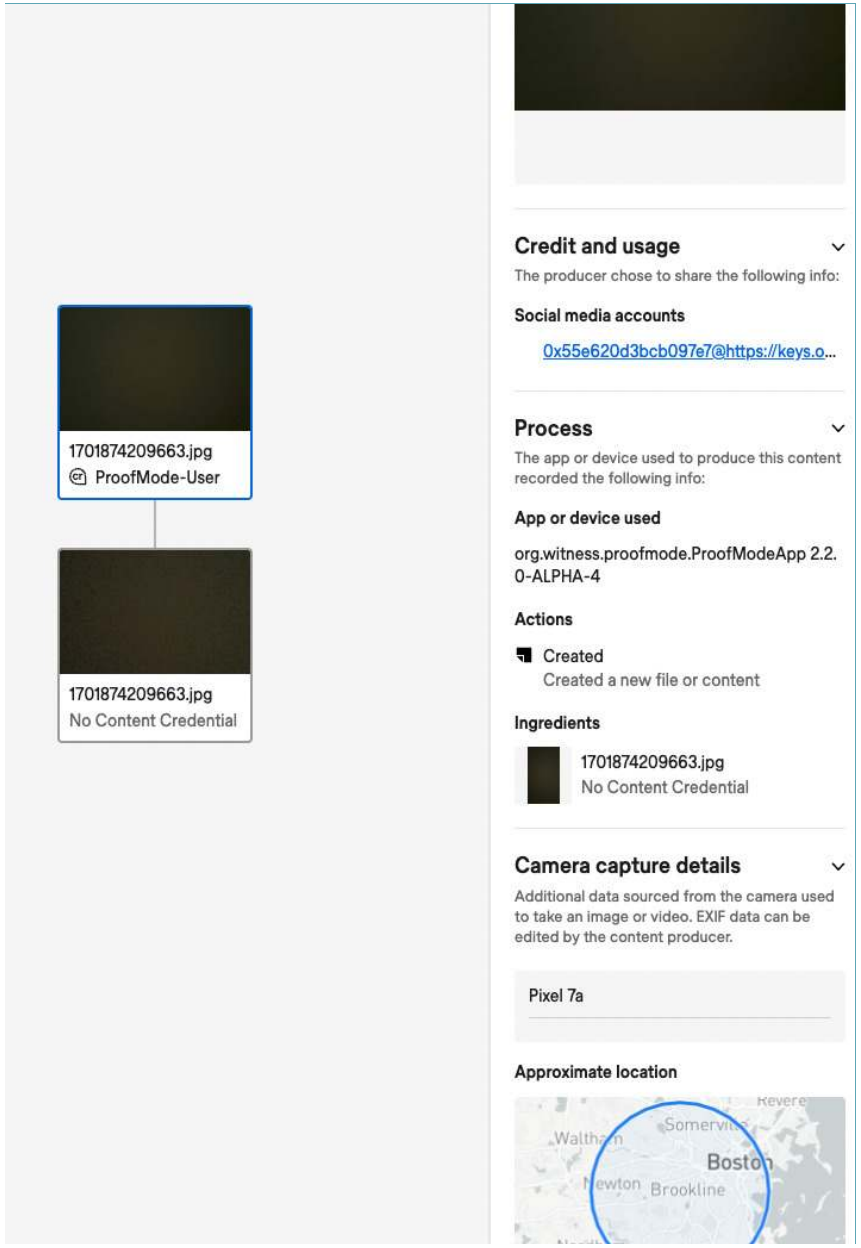
C2PA IS AN IMPORTANT STANDARD THAT ENABLES SECURE AND TRANSPARENT CONTENT PROVENANCE, EMPOWERING CREATORS, PUBLISHERS, AND CONSUMERS TO BETTER UNDERSTAND THE ORIGIN AND HISTORY OF DIGITAL MEDIA.

The Simple-C2PA Rust Library

The Simple-C2PA Rust Library is a project that builds upon the C2PA Rust library to provide an easy solution for mobile app developers to enable their apps to add signed C2PA actions, claims, and attestations to media files.

It provides pre-built libraries and developer interfaces in Swift and Kotlin.

It also includes support for generating a private key and self-signed x509 certificate entirely locally on the device.



Key Features

SIGNING C2PA ACTIONS, CLAIMS, AND ATTESTATIONS

The Simple-C2PA Rust Library allows developers to easily sign C2PA actions, claims, and attestations, ensuring the integrity and provenance of digital media files.

GENERATING A PRIVATE KEY AND SELF-SIGNED X509 CERTIFICATE ON THE DEVICE

The library can generate a private key and self-signed x509 certificate entirely on the device, without the need for external services or infrastructure.

THE SIMPLE-C2PA RUST LIBRARY PROVIDES A COMPREHENSIVE SET OF FEATURES TO SIMPLIFY THE INTEGRATION OF C2PA FUNCTIONALITY IN MOBILE APPS, HELPING DEVELOPERS BUILD MORE SECURE AND TRANSPARENT DIGITAL MEDIA EXPERIENCES.

iOS Integration

- ## ADD THE SIMPLEC2PA SWIFT PACKAGE IN XCODE

  To use the Simple-C2PA Rust Library on iOS, add the SimpleC2PA Swift package in Xcode with the repository URL https://gitlab.com/guardianproject/proofmode/simple-c2pa.

- ## THE CURRENT VERSION IS 0.0.16

  The current version of the SimpleC2PA Swift package is 0.0.16.

Sample Swift Code

```
let rootCert = try! createRootCertificate(organization: nil, validityDays: nil);

let contentCert = try! createContentCredentialsCertificate(rootCertificate:
rootCert, organization: nil, validityDays: nil)

let fileData = FileData(path: imagePath, bytes: nil, fileName: filename)

let cc = ContentCredentials(certificate: contentCert, file: fileData,
applicationInfo: nil) try!

cc.addCreatedAssertion() try!

cc.embedManifest(outputPath: outputPath)
```

Android Integration

- **ADD THE MAVEN REPOSITORY TO YOUR PROJECT**
  First, add our Maven repository to your Android project by including the following code in your project-level build.gradle file:

  allprojects { repositories { ... maven { url = uri("https://gitlab.com/api/v4/projects/51891540/packages/maven") } ... } }

- **IMPORT THE SIMPLE-C2PA LIBRARY**
  Then, import the simple-c2pa library, currently at version 0.0.16, in your app-level build.gradle file by adding the following dependency:

  implementation("info.guardianproject:simple-c2pa:0.0.16")

Sample Kotlin Code

```
val rootCert = createRootCertificate(null, null)

val contentCert = createContentCredentialsCertificate(rootCert, null, null)

val fileData = FileData(imagePath, null, fileName)

val cc = ContentCredentials(contentCert, fileData, null)

cc.addCreatedAssertion()

cc.embedManifest(outputPath)
```

ProofMode DEVELOP

SimpleC2PA Mobile and libProofMode SDKs

Provides an easy solution for mobile app developers to add decentralized provenance and authentication features

Now integrated into CuttingRoom Reporter at cuttingroom.com

Tella

Document & protect

In challenging environments, with limited or no internet connectivity or in the face of repression, Tella makes it easier and safer to document human rights violations and collect data.

GET IT ON Google Play

Download on the App Store

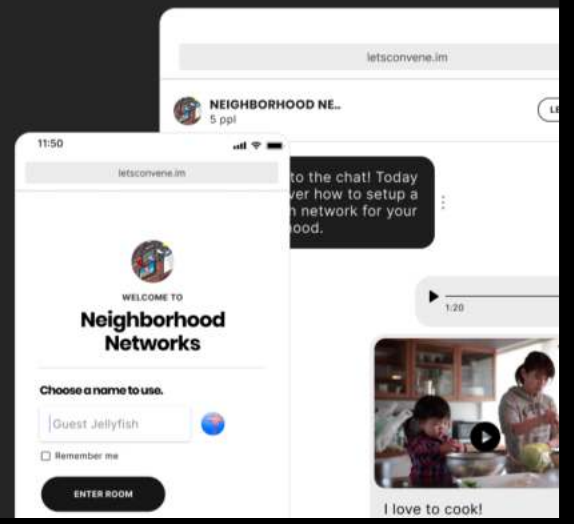
GET IT ON F-Droid

Simply Connect

Tired of looking for the messaging app everyone is using? Chat with Convene.

- ✓ No phone numbers
- ✓ No account setup
- ✓ No app

OPEN A PRIVATE ROOM



c2pa-Record-28.m4a

Issued by ProofMode-User

⚠ This Content Credential was issued by an unknown source.

No thumbnail available

OpenArchive

Our Work · About · News · Donate

Save By OpenArchive

Our App, *Save*

Save is an intuitive, privacy-first decentralized mobile archiving app that helps people preserve their media for the long term.

Download on the App Store

GET IT ON Google Play


Helge Høibraaten, Co-CEO
Cutting Room Reporter
now has support for C2PA!
C2PA-ready!
C2PA-ready!

ProofCheck
VERIFICATION BY PROOFMODE

VERIFY ANOTHER FILE

Content Credentials

CRR-2025-04-03-13-07-13--IG88-1-REC-original.MOV

0:00 / 0:07

1 video checked

Media signed

Proof data present

C2PA verified

DOWNLOAD

powered by proofmode.org

Content Credentials

CRR-2025-04-03-13-07-13--IG88-1-REC-original.MOV

1 video checked

Media signed

Content Credentials

CRR-2025-04-03-13-07-13--IG88-1-REC-original.MOV

General

Claim Generator: Simple-C2PA/0.0.15 c2pa-rs/0.39.0

Format: video/quicktime

Certificate

Signature Issuer: CuttingRoom

Certificate Number: 601141676075864286591878496549735905429590899554

Assertions

stds.exif: @context: dc: http://purl.org/dc/elements/1.1/
exif: http://ns.adobe.com/exif/1.0/
exifEX: http://cipa.jp/exif/2.32/
rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
tiff: http://ns.adobe.com/tiff/1.0/
xmp: http://ns.adobe.com/xap/1.0/
exif:GPSLatitude: 0.0
exif:GPSLongitude: 0.0
exif:GPSTimeStamp: 2025-04-03 17:07:13 +0000
exifEX:LensMake: iPhone14, iOS 18.3.2
exifEX:LensModel: Normal

COMPLETED

00:08

00:00

C2PA

Copied to Photos

RENAME VIDEO

Content Credentials

CRR-2025-04-03-13-07-13--IG88-1-REC-original.MOV

General

```
Claim Generator:  Simple-C2PA/0.0.15 c2pa-rs/0.39.0
          Format:  video/quicktime
```

Certificate

```
Signature Issuer:  CuttingRoom
Certificate Number:  601141676075864286591878496549735905429590899554
```

Assertions

```
stds.exif:  @context:  dc: http://purl.org/dc/elements/1.1/
                       exif: http://ns.adobe.com/exif/1.0/
                       exifEX: http://cipa.jp/exif/2.32/
                       rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
                       tiff: http://ns.adobe.com/tiff/1.0/
                       xmp: http://ns.adobe.com/xap/1.0/
            exif:GPSLatitude: 0.0
            exif:GPSLongitude: 0.0
            exif:GPSTimeStamp: 2025-04-03 17:07:13 +0000
            exifEX:LensMake: iPhone14, iOS 18.3.2
            exifEX:LensModel: Normal
            tiff:Make: CuttingRoom Reporter
            tiff:Model: CuttingRoom Reporter v2.42.1
```

```
val appLabel = getAppName(mContext)
val appVersion = getAppVersionName(mContext)
var appIconUri = APP_ICON_URI
var appInfo = ApplicationInfo(appLabel,appVersion,appIconUri)

var mediaFile = FileData(fileImageIn.absolutePath, null, fileImageIn.name)
var contentCreds = userCert?.let { ContentCredentials(it,mediaFile, appInfo) }

if (isDirectCapture)
     contentCreds?.addCreatedAssertion()
else
     contentCreds?.addPlacedAssertion()

if (!allowMachineLearning)
     contentCreds?.addRestrictedAiTrainingAssertions()
else
     contentCreds?.addPermissiveAiTrainingAssertions()

contentCreds?.addEmailAssertion(emailAddress, emailDisplay)

contentCreds?.addPgpAssertion(pgpFingerprint, pgpFingerprint)
contentCreds?.addWebsiteAssertion(webLink)
```

```
let signingIdentity = C2PAHelper.shared.getSigningIdentity()

let fileData = FileData(path: inputPath.path, bytes: inputData, fileName: inputPath.lastPathComponent)

let appName = Bundle.main.object(forInfoDictionaryKey: "CFBundleDisplayName") as? String ?? "ProofMode"
let appVersion = Bundle.main.object(forInfoDictionaryKey: "CFBundleVersion") as? String ?? ""
let appInfo = ApplicationInfo(name: appName, version: appVersion, iconUri: C2PAHelper.APP_ICON_URI)

let cc = ContentCredentials(certificate: userCert, file: fileData, applicationInfo: appInfo)

if isCapture { try? cc.addCreatedAssertion() }
else {  try? cc.addPlacedAssertion() }

if !Settings.shared.optionBlockAI { try? cc.addPermissiveAiTrainingAssertions() }
else { try? cc.addRestrictedAiTrainingAssertions() }

if let email = signingIdentity.email, let emailDisplay = signingIdentity.emailDisplay {
  try? cc.addEmailAssertion(email: email, displayName: emailDisplay)
}
if let pgpFingerprint = signingIdentity.pgpFingerprint {
    try? cc.addPgpAssertion(fingerprint: pgpFingerprint, displayName: pgpFingerprint)
}
if let webLink = signingIdentity.webLink { try? cc.addWebsiteAssertion(url: webLink) }
let _ = try cc.embedManifest(outputPath: outputPath.path)
```

Roadmap

- **IDENTITY AUTHORITY**

  Endorsed signing certificate through app and hardware integrity checks through centralized ProofSign service

- **IMPROVED SIDECAR SUPPORT**

  Addition of sidecar file support to enable C2PA without media modification

- **SIMPLE RECOMPRESSION AND SIGNING**

  Easy support for common operations, like social sharing of recompressed images

- **C2PA 2.1 SUPPORT WITH CAWG**

  Streaming, Identity, Endorsement and more

Building the Libraries From Source

- **ANDROID > CARGO MAKE ANDROID-BUILD**

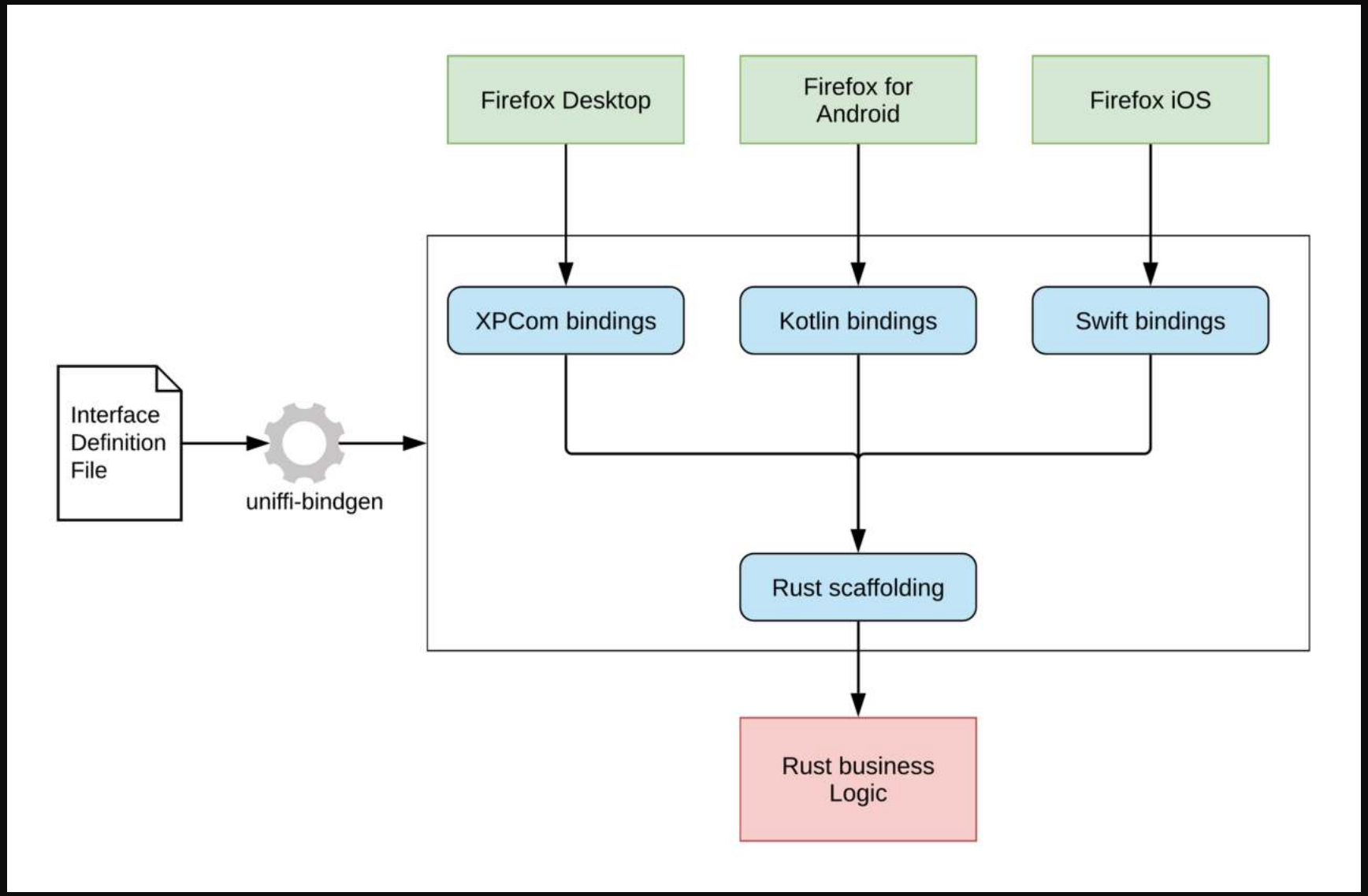
  To build the native Android library for the Simple-C2PA Rust Library, use the cargo-make command 'cargo make android-build'.

  This command will compile the Rust code into a native Android library. You will also need to have Docker installed and the latest version of cross.

- **APPLE > CARGO MAKE APPLE-BUILD**

  To build the native Apple library for the Simple-C2PA Rust Library, use the cargo-make command 'cargo make apple-build'.

  This command will compile the Rust code into a native Apple library. You will need to run this command on a Mac with Xcode installed.



UniFFI is a tool that "automatically" generates foreign-language bindings targeting Rust libraries.

ProofMode

ProofMode captures, authenticates and verifies smartphone multimedia from source to recipient. It enhances metadata, fingerprints hardware, cryptographically signs content, and uses third-party notaries for a decentralized, privacy-focused chain of custody—empowering activists, journalists, and everyday people.

Guardian Project

A global team who builds and designs easy to use secure apps, open-source software libraries, and customized solutions that can be used around the world by any person or organization looking to protect their communications and data from unjust intrusion, interception and monitoring.