

Плагин Signature V2 Внутренний документ

версия: 1.11.3

Предупреждение: этот документ предназначен для архитектуры ПЛАГИНА

- [Как читать этот документ?](#)
- [Параметры клиента](#)
 - [Android](#)
 - [Извлеките отпечаток сертификата SHA1](#)
 - [Извлеките отпечаток сертификата внутреннего совместного доступа к приложению](#)
 - [Извлеките отпечаток сертификата подписания приложения в Google Play](#)
 - [Извлеките ваш отпечаток сертификата подписи приложений Huawei AppGallery](#)
 - [iOS](#)
- [Гид по интеграции](#)
 - [Android](#)
 - [iOS](#)
 - [Устранение неполадок: ошибка верификации биткода «Failed to verify bitcode»](#)
 - [Устранение неполадок: подписание кода не выполнено](#)
 - [React Native](#)
 - [Cordova](#)
 - [Xamarin](#)
 - [Unity](#)
 - [Устранение неполадок: “Unknown CPU Architecture in AdjustSigSdk.a” Error](#)
- [Активация SDK Signature в Dashboard](#)
- [Включение/отключение подписи для тестирования](#)
 - [Плагин -> Руководство по обновлению библиотеки плагина](#)
- [Верификация](#)
 - [Step 1: С помощью sig_doctor](#)
 - [Использование](#)
 - [Шаг 2: Подготовка устройства](#)
 - [Шаг 3: Генерация приложения](#)
 - [Шаг 4A: Нативное Android приложение](#)
 - [Шаг 4B: Нативное iOS приложение](#)
 - [Шаг 4C: React Native Android приложение](#)
 - [Шаг 4D: React Native iOS приложение](#)
 - [Шаг 4E: Unity Android приложение](#)
 - [Шаг 4F: Unity iOS приложение](#)
 - [Шаг 4G: Xamarin Android приложение](#)
 - [Шаг 4H: Cordova Android приложение](#)
 - [Шаг 4I: Cordova iOS приложение](#)
- [Важные примечания для Play Store](#)
- [Контакт](#)

Как правильно читать этот документ?

- Если вы впервые интегрируете эту библиотеку:
 - i. Прочитайте раздел [Параметры клиента](#) section
 - ii. Отправьте свои параметры аккаунт-менеджеру
 - iii. В ответ вы получите библиотеку
 - iv. Пробежитесь по разделу [Гида по интеграции](#), соответствующему вашей платформе

v. Наконец, верифицируйте интеграцию с помощью раздела

[Верификация](#)

- Если вы хотите только интегрировать библиотеку:
 - i. Пробежитесь по разделу [Гид по интеграции](#), соответствующему вашей платформе
 - ii. После этого, верифицируйте интеграцию с помощью раздела [Верификация](#)
- Если вы хотите только верифицировать интеграцию библиотеки
 - i. Прочитайте раздел [Верификация](#)
- Если вы являетесь существующим пользователем библиотеки и
 - i. Хотели бы перейти на более новую версию библиотеки, прочитайте раздел [Плагин -> Руководство по обновлению библиотеки плагина](#)

Параметры клиента

Android

Для начала работы с бета-версией, Adjust понадобятся от вас три параметра. Кроме токена приложения, эти параметры являются общедоступными.

Нам требуется эта информация, так как библиотека создается под каждого отдельного клиента; двух одинаковых библиотек не существует. По этой причине библиотеке необходимо выполнить некоторые проверки базового пакета и сертифицировать отпечаток, чтобы убедиться, что она запущена в том приложении, для которого предназначена.

Серверы Google вместе с любым устройством, загружающим приложение, будут проверять отпечаток приложения, подписывающий сертификат.

Нам понадобятся три параметра:

- Название пакета (можно найти в ссылке на страницу вашего приложения в Google Play)
 - Он должен выглядеть как 'com.adjust.sdk'
- Токен приложения Adjust
 - Предоставляется Adjust, разработчикам будет просто его найти
 - Отпечаток(-ки), подписывающий(-е) сертификат, SHA-1 (используйте инструмент keytool для командной строки, который поставляется в Android SDK от Google)

Извлечение отпечатка сертификата SHA1

Найдите файл .jks, который используется для подписи последней версии приложения. Очень важно использовать **именно** то хранилище ключей (keystore), которое вы будете использовать для публикации **последней** версии вашего приложения в Google Play.

Выполните следующую команду:

```
$ keytool -list -v -keystore <location/of/your/key.jks> -alias YOUR_KEY_ALIAS
```

Убедитесь, что заменили то, что в скобках <>, *своими* значениями. Разработчик должен знать псевдоним хранилища ключа (keystore alias), его местоположение и пароль. Команда выше предложит вам ввести пароль хранилища ключа, поэтому убедитесь, что он под рукой.

После введения команды должно появиться следующая информация:

```
Alias name: Key0
Creation date: May 15, 2018
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=aaaa
Issuer: CN=aaaa
Serial number: 642f1b62
Valid from: Tue May 15 09:46:06 CEST 2018 until: Sat May 09 09:46:06 CEST 2043
Certificate fingerprints:
MD5: E7:88:9F:8C:9D:F4:14:C1:CF:E8:4C:97:F3:F2:3A:E3
SHA1: C4:BD:07:91:BC:09:F8:B6:15:CD:BC:A3:3F:BC:68:8B:C2:EF:4F:F5
SHA256: 55:FB:97:0F:46:0F:94:EC:07:EA:01:69:50:5A:20:3F:A0:91:60:A4:F1:33:58:EA:76:DC:54:9E:A7:6A:B9:1A
Signature algorithm name: SHA256withRSA
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3
```

Пожалуйста, отправьте нам отпечаток SHA1. В примере кода выше это: C4:BD:07:91:BC:09:F8:B6:15:CD:BC:A3:3F:BC:68:8B:C2:EF:4F:F5 .

Ваш результат **будет** другим.

Если вы используете [внутренний доступ к приложениям Google Play](#), прочитайте соответствующий раздел ниже.

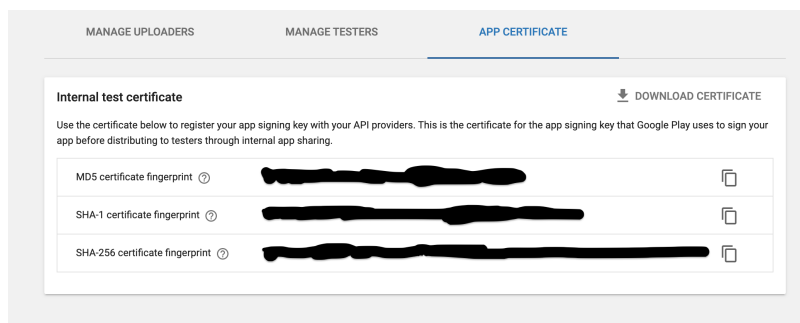
Если вы используете [функцию подписания приложений в Google Play](#), прочитайте соответствующий раздел ниже.

Извлечение отпечатка сертификата внутреннего доступа к приложениям

Если вы используете [функцию внутреннего доступа к приложению в Google Play](#), нам понадобится и **отпечаток сертификата SHA-1** вашей организации (см. раздел выше) и **отпечаток сертификата внутреннего тестирования**, чтобы убедиться, что интеграция будет корректно работать во время тестирования и в режиме продакшн.

Пожалуйста, выполните следующие шаги для извлечения отпечатков:

1. Перейдите в Google Play Console и войдите в учетную запись
2. Выберите приложение для подписи
3. Нажмите Release Management → App Releases → Manage internal appsharing → App certificates
4. Скопируйте оттуда отпечаток сертификата SHA-1, в том числе ключ из хранилища вашей организации, который вы использовали для подписи приложения



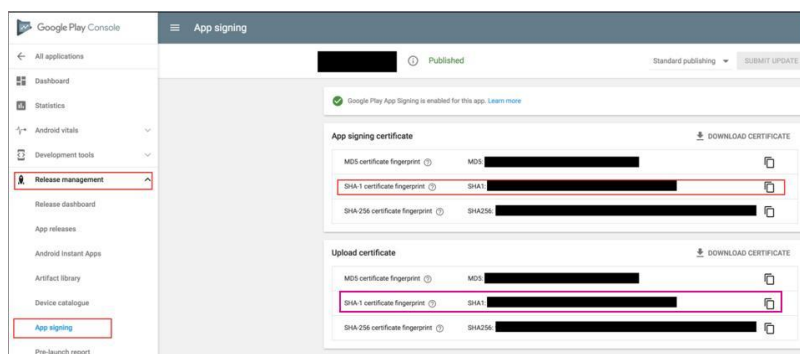
Извлечение отпечатка сертификата функции подписания приложений в Google Play

Если вы используете [функцию подписания приложений в Google Play](#), нам понадобится и **отпечаток сертификата SHA-1** вашей организации (см. соответствующий раздел выше) и **отпечаток сертификата подписания**

приложений, чтобы убедиться, что интеграция будет корректно работать во время тестирования и в режиме продакшн.

Пожалуйста, выполните следующие шаги для извлечения отпечатков:

1. Перейдите в Google Play Console и войдите в учетную запись
2. Выберите приложение для подписи
3. Нажмите Release Management → App Signing
4. Скопируйте сертификат отпечатка SHA-1 из **обоих секций**



Извлечение отпечатка сертификата функции подписания приложений в Huawei AppGallery

Если вы используете [функцию подписания приложений Huawei AppGallery](#), нам понадобится и **отпечаток сертификата SHA-1** вашей организации (см. соответствующий раздел выше) и **отпечаток сертификата подписания приложений**, чтобы убедиться, что интеграция будет корректно работать во время тестирования и в режиме продакшн.

Вы найдете отпечаток сертификата подписания приложений, если выполните шаги, описанные в «Вопрос 6» в FAQ Huawei по этой ссылке: https://developer.huawei.com/consumer/en/doc/development/AppGallery-connect-Guides/agc-app_bundle_faq

iOS

Параметры, которые нам понадобятся:

- Токен приложения Adjust
- Идентификатор приложения (Bundle ID)

Руководство по интеграции

Наше новое решение подписи SDK, Signature V2, – это подключаемый модуль, совместимый с общедоступными SDK. Signature V2 проще в интеграции и обеспечивает большую безопасность, чем предыдущая версия. Вот основные изменения:

- Содержание алгоритмов шифрования, защищающих связь с нашими серверами;
 - Мы используем укрепленный алгоритм запутывания следов, защищающий от перехватчиков данных;
 - Уникальные рандомизированные параметры шифрования, уникальные для каждого приложения.
- Мы используем шифрование с открытым ключом для предотвращения злонамеренного взлома.

Signature V2 **полностью заменяет** Signature V1 (наше решение с app secret). Поэтому мы **настоятельно рекомендуем** перейти к этому решению, если вы ранее использовали секреты приложения, **и, после того как вы убедитесь, что большая часть вашего трафика поступает от Signature V2** (через две недели после полного развертывания новой версии приложения в магазине), секреты Signature V1 можно будет удалить с панели управления.

ВАЖНО: как и в случае с Signature V1, если вы используете FPS Adjust, вы по-прежнему будете получать информацию об отклоненных установках посредством колбеков в Signature V2.

ВАЖНО: до версии 4.21.1 (для всех платформ) **библиотека Signature V2 была частью Adjust SDK**. Это значит, что команда разработчиков не могла интегрировать свежие изменения вместе с обновлением SDK и ей приходилось ждать, когда Adjust выпустит отдельное обновление Signature V2, включающее необходимые изменения.

Android

Signature V2 не является интерактивной. Это значит, что кроме интеграции библиотеки в проект, никаких других функций не нужно добавлять в кодовую базу клиента, как и убирать из нее.

Это также означает, что **в общедоступном SDK нет никаких функциональных изменений**: все события, колбеки сессий, атрибуция, прочие запросы SDK и его функциональность будут протекать ожидаемо.

Это минимальные требования для библиотеки (библиотека **не будет** работать без них).

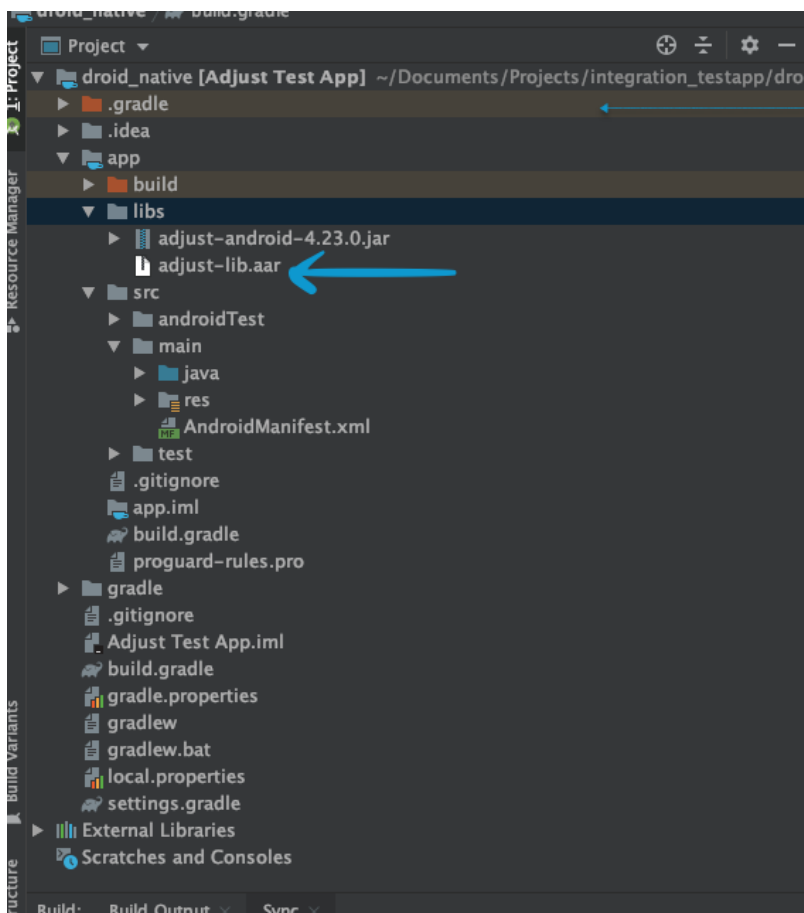
- Android API >= 18

ВАЖНО: для работы библиотеки необходима версия Android Adjust SDK >= 4.21.1.

ВАЖНО: если вы используете ProGuard, необходимо сохранить ту же самую конфигурацию Proguard для Signature V2, что используется для Adjust SDK.

Для интеграции библиотеки

1. Создайте новую директорию **libs** внутри директории модуля приложения.
2. Скопируйте предоставленную библиотеку AAR в созданный каталог **libs**.
 - **ВАЖНО:** мы будем использовать название «adjust-lib.aar» в этом документе для обозначения вашей настраиваемой библиотеки. Замените это название тем, которое вы получите от Adjust.



3. Откройте файл «build.gradle» на уровне приложения и добавьте следующую информацию в соответствующих разделах.

```
android {
    defaultConfig {
        ndk.abiFilters 'armeabi-v7a', 'arm64-v8a', 'x86', 'x86_64'
    }
}

dependencies {
    implementation files('libs/adjust-lib.aar')
}
```

4. Кликните sync project with Gradle files .

iOS

Signature V2 не является интерактивной. Это значит, что кроме интеграции библиотеки в проект, **никаких других функций не нужно добавлять в кодовую базу клиента, как и убирать** из нее.

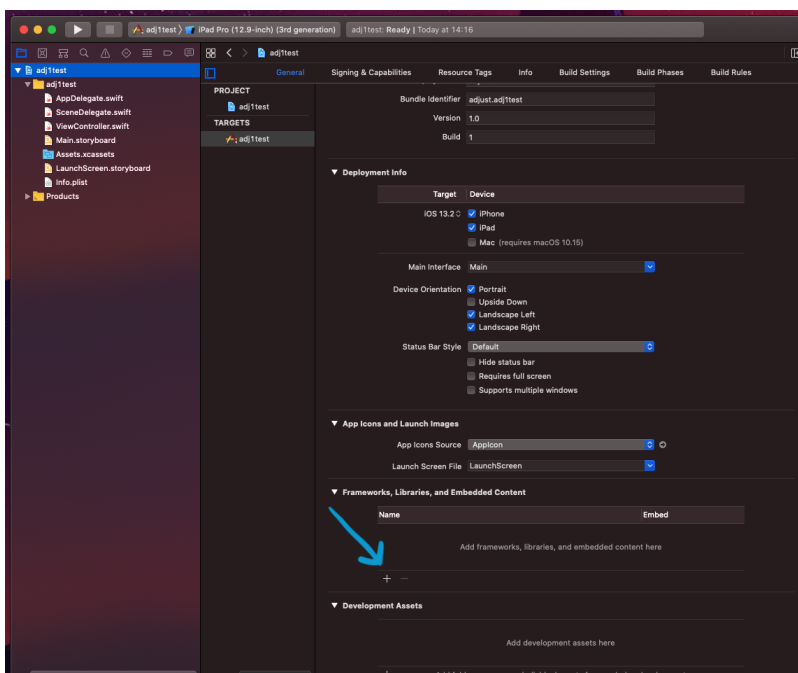
Это также означает, что **в общедоступном SDK нет никаких функциональных изменений**: все события, колбеки сессий, атрибуция, прочие запросы SDK и его функциональность будут протекать ожидаемо.

ВАЖНО: для работы библиотеки необходима версия iOS Adjust SDK >= 4.21.1.

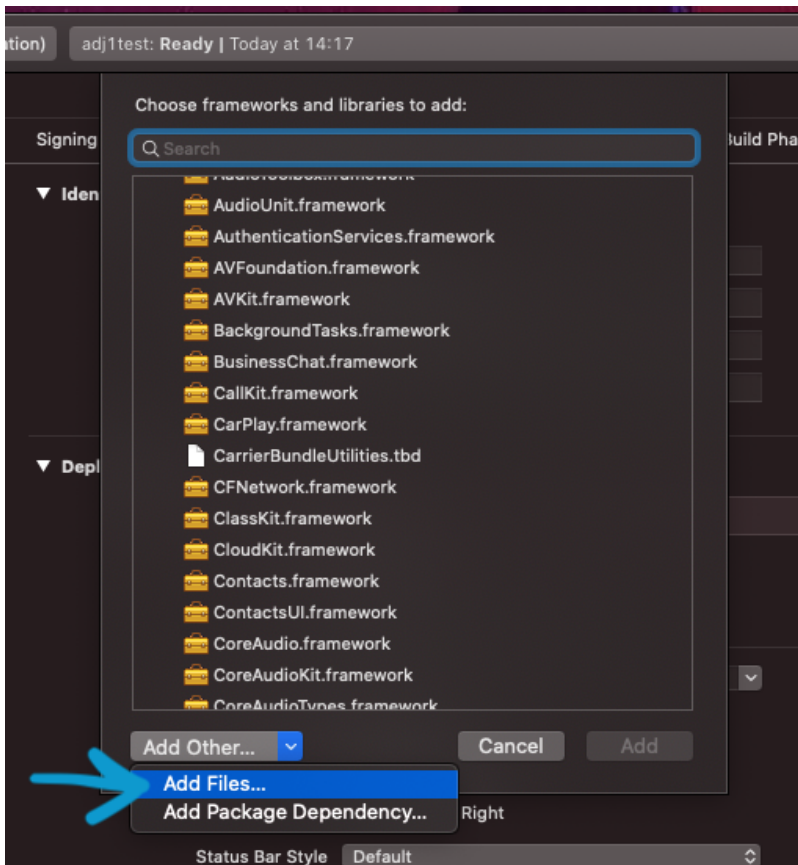
Интеграция динамического фреймворка

ВАЖНО: мы поставляем и динамический универсальный фреймворк, и динамический XCFramework. Шаги по интеграции идентичные. Чтобы не перегружать документ, я буду ссылаться ниже на использование XCFramework.

- **ВАЖНО:** мы будем использовать название «AdjustSigSdk» в этом документе для обозначения вашей настраиваемой библиотеки. Замените это название тем, что получите от Adjust.
1. Скопируйте динамический файл `AdjustSigSdk.xcframework` в каталог проекта
 2. В Xcode выберите ваш проект в Project Navigator
 3. В левой стороне основного окна выберите цель
- Для XCode >= 11:
 - i. Перейдите на вкладку `General`, разверните группы `Frameworks, Libraries` и `embedded Content`.
 - ii. Нажмите на кнопку `+` в нижней части этой секции.



3. Нажмите `Add Other > Add Files`, перейдите к строке в проекте, где вы вставили `AdjustSigSdk.xcframework`, и нажмите на нее.



4. После добавления обязательно нажмите `Embed & Sign` для `AdjustSigSdk.xcframework`.



- Для XCode < 11:

ВАЖНО: для XCode < 11 можно использовать только универсальный динамический фреймворк, а не XCFramework.

- На вкладке `Build Phases` разверните `Link Binary с группой Libraries`
- Нажмите на кнопку `+` в нижней части этой секции.
- Добавьте `AdjustSigSdk.framework` и задайте его как «optional». Нажав `Add Other...`, убедитесь, что выбрали файл `AdjustSigSdk.framework`, а не символическую ссылку Xcode в рамках всплывающего выбора фреймворка.
- Перейдите на вкладку `General`, разверните группу `Embedded Binaries`.
- Если библиотеки еще там нет, нажмите `Add Other...` и выберите `AdjustSigSdk.framework`. Нажав `Add Other...`, убедитесь, что выбрали файл `AdjustSigSdk.framework`, а не символическую ссылку Xcode в рамках всплывающего выбора фреймворка.
- Вы увидите диалоговое окно с заголовком `Choose options for adding these files`. При необходимости проверьте, выбрали ли вы действие `Copy items if needed`. Оно также добавит `AdjustSigSdk.framework` в навигатор проекта.

Интеграция статического фреймворка

Мы не рекомендуем использовать статические фреймворки, так как динамические фреймворки гораздо лучше и легче для вашего приложения. Как уже было сказано, если вы используете статические фреймворки, не забудьте включить `-force_load $(PROJECT_DIR)/$(PROJECT_NAME)/AdjustSigSdk.framework/AdjustSigSdk` в ваш XCode проект под обозначением «Other Linker Flags». Если вы не знаете, как это сделать, пожалуйста, не используйте статический фреймворк. Используйте динамический фреймворк (см. инструкции выше).

Устранение неполадок: ошибка «Не удалось проверить биткод»

Пример возможной ошибки при интеграции подписи:

```
Failed to verify bitcode in
AdjustSigSdk.framework/AdjustSigSdk:
error: Cannot extract bundle from
/var/folders/qy/yfq4b9750lg6x21scz02hq5c0000gn/T/IDEDistributionOptionThinning.Xnl/Payload/AdjustExample-iOS.
```

Вы используете динамический *non* xcframework и вам необходимо удалить двоичные файлы для архитектур `i386` и `x86_64` :

1. Выберите ваш проект в Project Navigator
2. В левой стороне основного окна выберите цель
3. Перейдите к вкладке `Build Phases` , нажмите кнопку `+` и выберите `New Run Script Phase`
4. Появится новый Run Script, назовите его «Strip Adjust Framework» и поместите его в «Build Phases» **под** вкладкой «Embed Frameworks»
5. Скопируйте фрагмент кода [по этой ссылке](#) в область ввода
6. Очистите и создайте новый билд

Или же, если позволяет ваш процесс сборки, подумайте о том, чтобы перейти на динамический фреймворк xcframework.

Устранение неполадок: подписание кода не выполнено

Это может произойти, если вы включили скрипт запуска «Strip Adjust Framework»(см. раздел выше) и поместили вкладку «Embed Frameworks» в неправильном месте в «BuildPhases».

Правильный порядок «Build Phases» должен выглядеть так (другие фазы можно помещать между этими):

```
зависимости
Run Script
Compile Sources
Link Binary with Libraries
Copy Bundle Resources
Embed Pods Frameworks
Embed Frameworks
Strip Adjust Framework
```

Если вы добавили вышеупомянутый скрипт запуска «Strip Adjust Framework» к фазамбилда, вам нужно поместить «Embed Frameworks»

React Native

Signature V2 не является интерактивной. Это значит, что кроме интеграции библиотеки в проект, **никаких других функций не нужно добавлять в кодовую базу клиента, как и убирать** из нее.

Это также означает, что **в общедоступном SDK нет никаких функциональных изменений**: все события, колбеки сессий, атрибуция, прочие запросы SDK и его функциональность будут протекать ожидаемо.

Это минимальные требования для библиотеки (библиотека **не будет** работать без них).

- Если вы используете Android
 - Android API ≥ 18
 - Android Adjust SDK $\geq 4.21.1$
- Если вы используете iOS
 - iOS Adjust SDK $\geq 4.21.1$

Чтобы интегрировать Signature V2 для React Native, сначала интегрируйте SDK Adjust согласно инструкции в документации по SDK Adjust. Signature V2 требует наличия SDK Adjust.

После того как SDK Adjust интегрирован, вы можете интегрировать Signature V2 для React Native, выполнив исходные для [Android](#) и [iOS](#) шаги интеграции, приведенные ранее в этом документе.

Уточнение: в этой библиотеке мы предоставляем только нативные библиотеки, а не полный пакет React, так как в последнем **нет необходимости**. Для этой библиотеки нет методов API, ориентированных на клиента. SDK Adjust знает, что делать с нативной библиотекой, если он находит ее в приложении, без необходимости добавления в код клиента.

Cordova

Signature V2 не является интерактивной. Это значит, что кроме интеграции библиотеки в проект, **никаких других функций не нужно добавлять в кодовую базу клиента, как и убирать** из нее.

Это также означает, что **в общедоступном SDK нет никаких функциональных изменений**: все события, колбеки сессий, атрибуция, прочие запросы SDK и его функциональность будут протекать ожидаемо.

Это минимальные требования для библиотеки (библиотека **не будет** работать без них).

- Если вы используете Android
 - Android API ≥ 18
 - Android Adjust SDK $\geq 4.21.1$
- Если вы используете iOS
 - iOS Adjust SDK $\geq 4.21.1$

ВАЖНО: если вы используете ProGuard, необходимо сохранить ту же самую конфигурацию Proguard для Signature V2, что используется для Adjust SDK.

Что касается фактических шагов интеграции:

1. Создайте папку в вашем проекте и назовите ее `ext`
2. Распакуйте библиотеку туда, в папке `ext` у вас появится еще одна с названием `cordova-adjust-sig`
3. Запустите `cordova plugin add ext/cordova-adjust-sig`. Теперь загрузите Adjust SDK посредством `cordova plugin add com.adjust.sdk` или любым другим способом из [общедоступного README](#).
4. Откройте проект iOS Xcode в `platforms/ios` с Xcode. Выберите ваш проект в Project Navigator. В левой стороне главного дисплея нажмите на ваш проект.
5. Кликните на вкладку `Build Settings` (настройки билда)
6. Кликните два раза на `Other Linker Flags`
7. Нажмите `+`
8. Вставьте: `-force_load $(PROJECT_DIR)/$(PROJECT_NAME)/Plugins/com.adjust.sdk.sig/AdjustSigSdk.framework/AdjustSigSdk`

Xamarin

Signature V2 не является интерактивной. Это значит, что кроме интеграции библиотеки в проект, **никаких других функций не нужно добавлять в кодовую базу клиента, как и убирать** из нее.

Это также означает, что **в общедоступном SDK нет никаких функциональных изменений**: все события, колбеки сессий, атрибуция, прочие запросы SDK и его функциональность будут протекать ожидаемо.

Это минимальные требования для библиотеки (библиотека **не будет** работать без них).

- Если вы используете Android
 - Android API ≥ 18
 - Android Adjust SDK $\geq 4.21.1$
- Если вы используете iOS
 - iOS Adjust SDK $\geq 4.21.1$

ВАЖНО: если вы используете ProGuard, необходимо сохранить ту же самую конфигурацию Proguard для Signature V2, что используется для Adjust SDK.

Что касается фактических шагов интеграции:

- Android:
 - i. Скопируйте полученный файл «AdjustSigSdk.Xamarin.Android.dll» в проект Android (например, «MyProject/MyProject.Android/AdjustSigSdk.Xamarin.Android.dll»).
 - ii. В Solution Explorer кликните правой кнопкой мыши на название проекта и нажмите `Add > Reference...`
 - iii. На вкладке «.Net Assembly» нажмите «Выбрать...», чтобы выбрать копируемый файл и нажмите ОК.
- iOS:
 - i. Скопируйте полученный файл «libAdjustSigSdk.iOS.a» в проект iOS (например, «MyProject/MyProject.iOS/libAdjustSigSdk.iOS.a»).
 - ii. В Solution Explorer кликните правой кнопкой мыши на название проекта и нажмите «Добавить» > «Добавить Native Reference», чтобы выбрать скопированный файл.
 - iii. В разделе «Native References» кликните правой кнопкой мыши «libAdjustSigSdk.iOS» и выберите «Свойства»
 - Поставьте галочку в Force Load
 - Добавьте в «Linker Flags» следующее: «-L\$(ProjectDir)»

Unity

Signature V2 не является интерактивной. Это значит, что кроме интеграции библиотеки в проект, **никаких других функций не нужно добавлять в кодовую базу клиента, как и убирать** из нее.

Это также означает, что **в общедоступном SDK нет никаких функциональных изменений**: все события, колбеки сессий, атрибуция, прочие запросы SDK и его функциональность будут протекать ожидаемо.

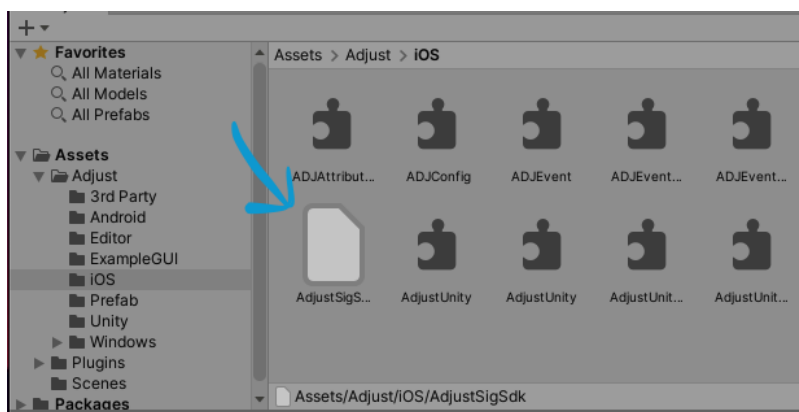
Это минимальные требования для библиотеки (библиотека **не будет** работать без них).

- Если вы используете Android
 - Android API ≥ 18
 - Android Adjust SDK $\geq 4.21.1$
- Если вы используете iOS
 - iOS Adjust SDK $\geq 4.21.1$

ВАЖНО: если вы используете ProGuard, необходимо сохранить ту же самую конфигурацию Proguard для Signature V2, что используется для Adjust SDK.

Что касается фактических шагов интеграции:

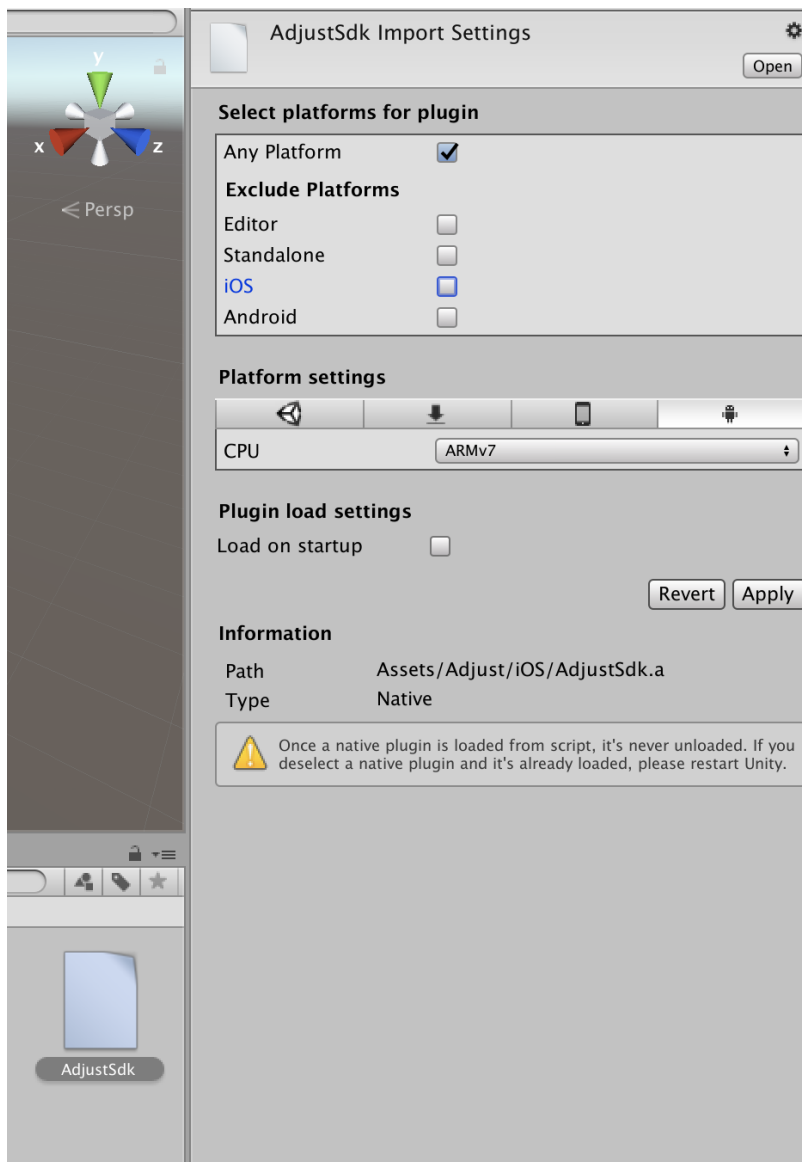
1. Переместите .aar файл в Assets/Adjust/Android
2. Переместите .a файл в Assets/Adjust/iOS
3. Если вы пользуетесь Adjust Unity SDK 4.23.1 и более поздних версий, интегрированный процесс post-build позаботится об остальном.
4. Если вы пользуетесь версией Adjust Unity SDK **более ранней**, чем 4.23.1, выполните следующие действия:
 - i. Откройте проект iOS Xcode с Xcode.
 - ii. Выберите ваш проект в навигаторе проектов (Project Navigator).
 - iii. В левой стороне главного дисплея нажмите на ваш проект.
 - Кликните на вкладку Build Settings (настройки билда)
 - Кликните два раза на Other Linker Flags
 - Нажмите на +
 - Вставьте следующее: `-force_load $(PROJECT_DIR)/Libraries/Adjust/iOS/AdjustSigSdk.a`



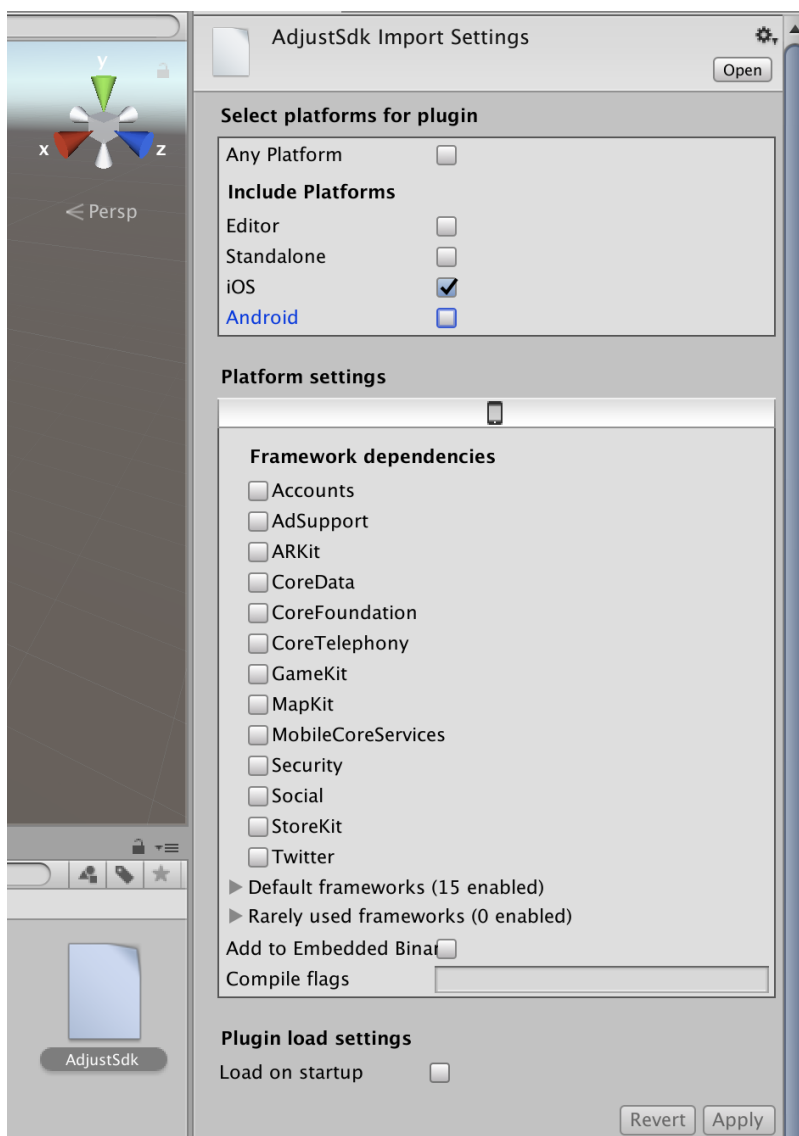
Устранение неполадок: «Unknown CPU Architecture inAdjustSigSdk.a» Error

Unity Editor версии 2018 г. и позднее иногда считывает статическую библиотеку Adjust iOS AdjustSigSdk.a как поддерживаемую «всеми архитектурами», а не только iOS.

- Выберите файл AdjustSigSdk.a Assets/Adjust/iOS как показано ниже:



- Теперь выключите `Any Platform` и выберите платформу `iOS`, как показано ниже:



При включении цифровой подписи SDK на панели управления

Посетите [эту страницу](#), пожалуйста.

ВАЖНО: если подпись не установлена как обязательный параметр в настройках панели управления, будут приниматься все установки, включая те, у которых нет подписи или подпись недействительна. После «включения» подписи серверы Adjust немедленно начинают отклонять все установки, не содержащие действительную подпись. Поэтому важно убедиться, что переключатель применения подписи установлен на значение «ВКЛ.», если большинство входящих запросов на установку происходит из версии приложения, содержащей Signature V2.

При отключении подписи для тестирования

Подпись включена по умолчанию. И всё же, иногда во время тестирования вы можете захотеть **отключить** подпись в вашем test suite. Если вы отключите подпись, ваши запросы Adjust не будут подписаны. Ниже описано, как вы можете это сделать.

Мы напоминаем, что эту функцию следует использовать **только для тестирования**, а не в продакшене.

ВНИМАНИЕ: переключатель применения подписи SDK на панели управления Adjust **влияет на Signature V2**. Если вызывается параметр «disableSigning()» и параметр «Применить подпись SDK» установлен на **ВКЛ.**, **весь трафик отклоняется**, независимо от того, выбран ли в среде «ТЕСТОВЫЙ РЕЖИМ» или какое-то другое значение.

- Android:

```
// Проверьте, находитесь ли вы в модуле SANDBOX во время тестирования
AdjustConfig config = new AdjustConfig(this, appToken, AdjustConfig.ENVIRONMENT_SANDBOX);
AdjustFactory.disableSigning()
```

- iOS:

```
// Проверьте, находитесь ли вы в модуле SANDBOX во время тестирования
ADJConfig*adjustConfig = [ADJConfig configWithAppToken:yourAppToken environment:ADJEnvironmentSandbox];
[ADJAdjustFactory disableSigning]
```

Плагин -> Руководство по обновлению библиотеки плагина

Если вы хотите обновить библиотеку Signature V2 до более новой версии (обновления рекомендуется проводить каждые 6 месяцев), выполните следующие действия.

1. Запросите новую библиотеку Signature V2 для интересующих фреймворков и платформ у вашего менеджера по работе с клиентами.
2. Получив новую библиотеку Signature V2, ознакомьтесь с прилагаемой к ней документацией.
3. Чтобы избежать проблем, *полностью* удалите предыдущую библиотеку Signature V2 из вашего приложения.
4. Следуйте руководству по интеграции и проверке, как указано в документации, прилагаемой к новой библиотеке Signature V2.

Подтверждение

Тестирование нового решения SDK немного отличается от тестирования предыдущего Adjust SDK. Пожалуйста, выполните следующие шаги, чтобы убедиться, что интеграция прошла успешно

Шаг 1. Использование «sig_doctor»

`sig_doctor` – это отдельный инструмент для верификации успешной интеграции Signature V2. Он включен в полученный вами zip-файл Signature V2.

Существует двоичный `sig_doctor` для трех основных операционных систем (Windows, MacOS и Linux). Более того, нужно также тестировать и iOS IPAs/frameworks, и Android APKs/AARs для всех поддерживаемых пересекающихся платформ (Unity, Cordova и т.д.)

Зависимости

Бинарный файл скомпилирован статически, у него нет зависимостей.

Использование

- Для Android: соберите **релизный** APK, в котором используются релизные ключи магазина.
 - Это должен быть тот же самый билд, который вы отправите в магазин приложений, перед публикацией.
 - Если вы используете *Android App Bundles*, процесс будет идентичным процессу создания APK. Тестирование нужно будет провести с вашим релизным APK, а не с AAB. Кроме того, можно использовать `bundletool` для создания APK из файла AAB, при предоставлении `--mode=universal` в качестве аргумента.
- Для iOS: соберите IPA (не имеет значения, какой именно: `development`, `ad hoc` или `enterprise`).
- Кликните два раза на бинарный файл `adjust_sig_doctor` на основе вашей платформы (Windows, OSX, Linux).
- Следуйте инструкциям в диалоговом окне.

С использованием CLI

`sig_doctor` – это также CLI инструмент. Запустите его кнопкой `-h`.

Шаг 2: Подготовка устройства

1. Используйте настоящее устройство, а не эмулятор или симулятор.
2. Полностью удалите приложение с устройства.
3. Убедитесь, что устройство задано как «forgotten» (забытое). Вы можете сделать это либо с помощью тестовой консоли, либо посредством ввода ссылки, представленной ниже в браузер и перехода по ней:
 - для Android: `http://app.adjust.com/forget_device?app_token={yourAppToken}&gps_adid={gpsAdidValue}`
 - Замените `yourAppToken` и `gps_adid` соответственно
 - для iOS: `http://app.adjust.com/forget_device?app_token={yourAppToken}&idfa={idfaValue}`
 - Замените `yourAppToken` и `idfaValue` соответственно
4. Подключите устройство к машине разработки
5. Наконец, для тестирования понадобится рекламный идентификатор устройства. Для его получения вы можете использовать приложение Adjust Insights ([Android](#) | [iOS](#)), чтобы получить этот идентификатор.

Шаг 3: Подготовка устройства

ВАЖНО: использование библиотеки в режиме отладки внутри Android Studio или Xcode запустит механизм библиотеки по проверке наличия подписи и пометит установку как «недоверенную». Тест **необходимо** проводить с использованием запакowanego приложения. Для Android это означает подпись APKс помощью релизного хранилища ключей и его установку на настоящее устройство. Для iOS это означает архивацию проекта и генерацию «Ad Hoc» IPA, который должен быть установлен на настоящее устройство.

Шаг 4A: Native Android App

- Вам нужно собрать релизную версию приложения и подписать ее тем же хранилищем ключей, что вы использовали для сертификата отпечатка SHA1, который вы отправили Adjust.
- Установите ваше приложение посредством ADB на подготовленное устройство (см. выше).
- Запустите его, чтобы установка была отправлена на серверы Adjust.
- Используйте консоль тестирования [Testing Console](#) вместе с Google Advertising ID или IDFA вашего устройства, чтобы проверить, что `SignatureVerificationResult` (результат верификации подписи) – это `Valid Signature` (действительная подпись)

ВАЖНО: если `SignatureVerificationResult` не был определен как 'Valid Signature', не публикуйте приложение. Свяжитесь с вашим аккаунт-менеджером/сейлз-инженером. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

ВАЖНО: использование библиотеки в режиме отладки внутри Android Studio или Xcode запустит механизм проверки наличия подписи и пометит установку как «недоверенную». Тест **необходимо** проводить с использованием запакowanego приложения. Для Android это означает подпись APKс помощью релизного хранилища ключей и его

установку на настоящее устройство. Для iOS это означает архивацию проекта и генерацию «Ad Hoc» IPA, который должен быть установлен на настоящее устройство.

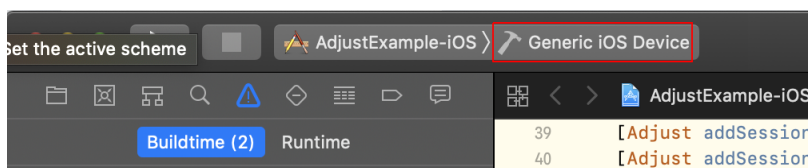
Шаг 4B: Native iOS App

Необходимые шаги.

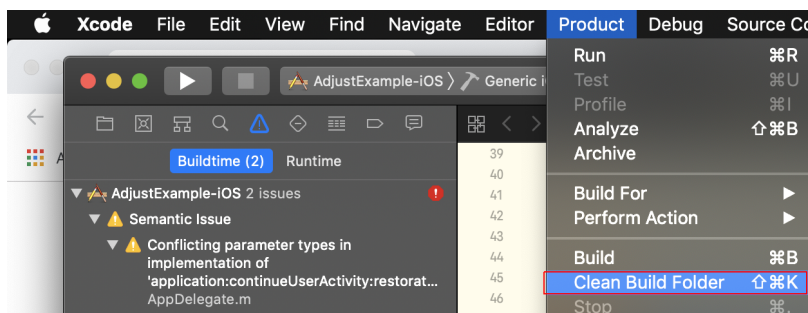
1. Заархивируйте development версию вашего приложения как IPA файл.
2. Вы только что отправили ваше приложение посредством Xcode.

Подробнее.

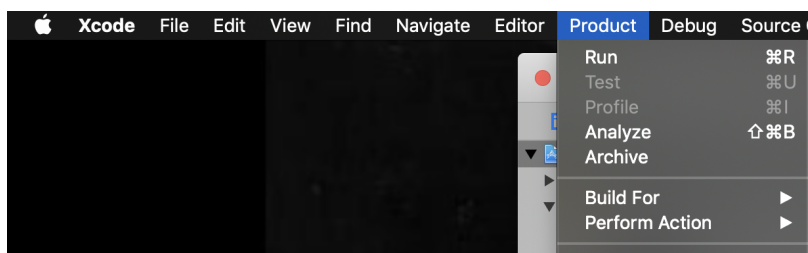
- Выберите `Generic iOS Device` из выпадающего меню со списком устройств (вы найдете его справа от меню отладки и стоп-знаков на основной панели).



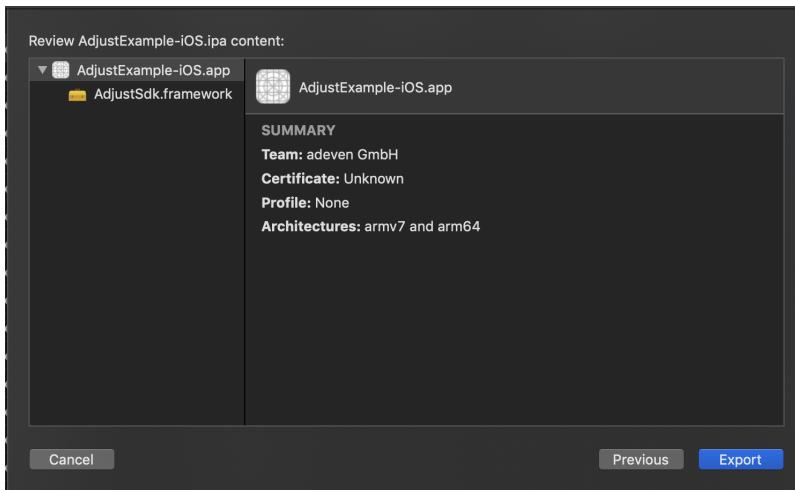
- Нажмите `Product` → `Clean Build Folder` (или `Clean`) на панели инструментов Xcode.



- Кликните `Product` → `Archive` на вашей Xcode панели.



- Выберите первую позицию
 - Если ваш Xcode < 10, кликните `Export`
 - Если ваш Xcode >= 10, кликните `Distribute App`
 - Нажмите `Development` и продолжите
ВАЖНО: если вы столкнетесь с ошибкой *Failed to verify bitcode*, пожалуйста, обратитесь к этому [подразделу](#), посвященному устранению неполадок в этом документе в разделе «iOS Integration Guide».
 - Вы увидите диалоговое окно «Development distribution options». Не меняя параметры, кликните `Next` в этом и следующем меню
 - Вы получите следующее сообщение:



- Кликните `Export` и сохраните архив
- Подключите физическое устройство iOS с помощью USB
- Нажмите `Windows` -> `Devices and Simulators` на панели инструментов Xcode
- Вы увидите ваше тестовое устройство. Кликните на значок `+` в нижней части панели «Installed Apps» и выберите `.ipa` файл из каталога, который вы архивировали
- Поздравляем, вы загрузите приложение посредством Xcode! Запустите ваше приложение с устройства iOS и проверьте совпадение `Test Console` с IDFA устройства, чтобы проверить, успешно ли прошла интеграция Signature V2.
- Вы также можете просмотреть логи устройства или в окне `Devices and Simulators` или с помощью приложения `Console` на MacOS, которое предустанавливается вместе с MacOS.

Используйте консоль тестирования [Testing Console](#) вместе с Google Advertising ID или IDFA вашего устройства, чтобы проверить, что `SignatureVerificationResult` (результат верификации подписи) – это `Valid Signature` (действительная подпись)

ВАЖНО: если `SignatureVerificationResult` не был определен как 'Valid Signature', не публикуйте приложение. Свяжитесь с вашим аккаунт-менеджером/сейлз-инженером. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

ВАЖНО: использование библиотеки в режиме отладки внутри Android Studio или Xcode запустит механизм проверки наличия библиотеки и пометит установку как «недоверенную». Тест **необходимо** проводить с использованием запакованного приложения. Для Android это означает подпись APKс помощью релизного хранилища ключей и его установку на настоящее устройство. Для iOS это означает архивацию проекта и генерацию «Ad Hoc» IPA, который должен быть установлен на настоящее устройство.

Шаг 4C: React Native Android app

Примените те же шаги, что и в [разделе 4A](#). Вам нужно будет также запустить `react-native bundle`, чтобы построить Javascript слой для офлайн-пользователей. Запустите следующую команду и замените `'-entry-file'` и `'-assets-dest'` параметрами их копиями в вашем проекте:

```
$ react-native bundle --platform android --dev false --entry-file
index.js --bundle-output
android/app/src/main/assets/index.android.bundle --assets-dest
android/app/src/main/res
```

В итоге должен получиться такой результат:

```
[1] % react-native bundle --platform android --dev false --entry-file index.js --bundle-output android/app/src/main/assets/index.android.bundle --assets-dest android/app/src/main/res
Loading dependency graph, done.
Loading dependency graph... bundle: Writing bundle output to: android/app/src/main/assets/index.android.bundle
bundle: Done writing bundle output
```

ВАЖНО: если ответ был атрибутирован как «untrusted» (ненадежный), остановите текущий процесс. Не публикуйте приложение и свяжитесь со своим аккаунт-менеджером. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

Если вы столкнетесь с такой ситуацией, предоставьте нам данные тестового устройства: GoogleAdvertising ID (Android) или IDFA (iOS). Мы используем их для отладки установки. Вы можете использовать приложение Adjust Insights ([Android](#) | [iOS](#)), чтобы извлечь Google Advertising ID.

Шаг 4D: React Native iOS App

Примените те же шаги, что и в [разделе 4B](#).

ВАЖНО: если ответ был атрибутирован как «untrusted» (ненадежный), остановите текущий процесс. Не публикуйте приложение и свяжитесь с командой мобильной безопасности Adjust. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

Если вы столкнетесь с такой ситуацией, предоставьте нам данные тестового устройства: Google Advertising ID (Android) или IDFA (iOS). Мы используем их для отладки установки. Вы можете использовать приложение Adjust Insights ([Android](#) | [iOS](#)), чтобы извлечь iOS IDFA устройства.

Шаг 4E: Unity Android App

Примените те же шаги, что и в [разделе 4A](#).

ВАЖНО: если ответ был атрибутирован как «untrusted» (ненадежный), остановите текущий процесс. Не публикуйте приложение и свяжитесь с командой мобильной безопасности Adjust. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

Если вы столкнетесь с такой ситуацией, предоставьте нам данные тестового устройства: GoogleAdvertising ID (Android) или IDFA (iOS). Мы используем их для отладки установки. Вы можете использовать приложение Adjust Insights ([Android](#) | [iOS](#)), чтобы извлечь Google Advertising ID.

Шаг 4F: Unity iOS App

Примените те же шаги, что и в [разделе 4B](#).

ВАЖНО: если ответ был атрибутирован как «untrusted» (ненадежный), остановите текущий процесс. Не публикуйте приложение и свяжитесь с командой мобильной безопасности Adjust. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

Если вы столкнетесь с такой ситуацией, предоставьте нам данные тестового устройства: GoogleAdvertising ID (Android) или IDFA (iOS). Мы используем их для отладки установки. Вы можете использовать приложение Adjust Insights ([Android](#) | [iOS](#)), чтобы извлечь iOS IDFA устройства.

Шаг 4G: Xamarin Android App

Примените те же шаги, что и в [разделе 4A](#).

ВАЖНО: если ответ был атрибутирован как «untrusted» (ненадежный), остановите текущий процесс. Не публикуйте приложение и свяжитесь с командой мобильной безопасности Adjust. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

Если вы столкнетесь с такой ситуацией, предоставьте нам данные тестового устройства: GoogleAdvertising ID (Android) или IDFA (iOS). Мы используем их для отладки установки. Вы можете использовать приложение Adjust Insights ([Android](#) | [iOS](#)), чтобы извлечь Google Advertising ID.

Шаг 4H: Cordova Android App

Примените те же шаги, что и в [разделе 4A](#).

ВАЖНО: если ответ был атрибутирован как «untrusted» (ненадежный), остановите текущий процесс. Не публикуйте приложение и свяжитесь с командой мобильной безопасности Adjust. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

Если вы столкнетесь с такой ситуацией, предоставьте нам данные тестового устройства: GoogleAdvertising ID (Android) или IDFA (iOS). Мы используем их для отладки установки. Вы можете использовать приложение Adjust Insights ([Android](#) | [iOS](#)), чтобы извлечь Google Advertising ID.

Шаг 4I: Cordova iOS App

Выполните те же шаги, что и в [разделе 4B](#), после того как вы:

- Соберите билд в Cordova
- Импортируйте билд в Xcode

ВАЖНО: если ответ был атрибутирован как «untrusted» (ненадежный), остановите текущий процесс. Не публикуйте приложение и свяжитесь с командой мобильной безопасности Adjust. В этом случае нам нужно будет более внимательно изучить детали вашей интеграции.

Если вы столкнетесь с такой ситуацией, предоставьте нам данные тестового устройства: GoogleAdvertising ID (Android) или IDFA (iOS). Мы используем их для отладки установки. Вы можете использовать приложение Adjust Insights ([Android](#) | [iOS](#)), чтобы извлечь iOS IDFA устройства.

Предупреждения в Play Store

При загрузке приложения в Play Store может появиться следующее предупреждение:

Этот комплект приложений содержит нативный код, а вы не загрузили символы отладки. Мы рекомендуем загружать файл символов, чтобы облегчить анализ и отладку сбоев и ANR.

Символов отладки нет преднамеренно, и предупреждение можно смело игнорировать.

Контакт

Если вы столкнулись с проблемой, обратитесь к вашему аккаунт-менеджеру. Он поможет вам оперативно ее решить.

Если у вас есть предложения по улучшению процесса тестирования, вы можете также направить ваши пожелания аккаунт-менеджеру. Мы будем благодарны за любые комментарии, которые сделают процесс проще для обеих сторон.

– Команда мобильной безопасности Adjust