Authorized Sellers for Apps (app-ads.txt) Specification

DRAFT for Public Comment, BETA 1.0
November 30, 2018

About ads.txt and app-ads.txt

The ads.txt specification “Authorized Digital Sellers for Mobile Apps (app-ads.txt) Specification” was developed in the spring of 2017, covering desktop and mobile web inventory. This document describes an extension of the original ads.txt standard to meet the requirements for software applications distributed through mobile app stores, connected television app stores, and other distribution channels of this nature. Authorized Sellers for Apps (app-ads.txt) is a peer-reviewed standard developed with the support of the OpenRTB working group. This document and the complementary ads.txt specification are available at https://iabtechlab.com/ads-txt.

This specification addresses the need for ads.txt functionality for mobile apps. This specification also has the potential to allow apps to implement ads.cert or other web-based resources in the future.

Public Comment

Public comments on this draft are welcome from November 30, 2018 until February 4, 2019. Comments can be sent to openmedia@iabtechlab.com and will be reviewed by the working group. There is no set date for when this app-ads.txt specification will be out of beta.

About the IAB Technology Lab

The IAB Technology Laboratory (Tech Lab) is a non-profit research and development consortium that produces and provides standards, software, and services to drive growth of an

effective and sustainable global digital media ecosystem. Comprised of digital publishers and ad technology firms, as well as marketers, agencies, and other companies with interests in the interactive marketing arena, IAB Tech Lab aims to enable brand and media growth via a transparent, safe, effective supply chain, simpler and more consistent measurement, and better advertising experiences for consumers, with a focus on mobile and TV/digital video channel enablement. The IAB Tech Lab portfolio includes the DigiTrust real-time standardized identity service designed to improve the digital experience for consumers, publishers, advertisers, and third-party platforms. Board members include AppNexus, ExtremeReach, Google, GroupM, Hearst Digital Media, Integral Ad Science, Index Exchange, LinkedIn, MediaMath, Microsoft, Moat, Pandora, PubMatic, Rakuten, Quantcast, Telaria, The Trade Desk, and Yahoo! Japan. Established in 2014, the IAB Tech Lab is headquartered in New York City with an office in San Francisco and representation in Seattle and London.

Learn more about IAB Tech Lab at www.iabtechlab.com.

Authors:
Curtis Light, Staff Software Engineer, Google
Curt Larson, Chief Product Officer, Sharethrough

Other Significant Contributions Include:
Duke Dukellis, Director, Product Management, Google; Ian Trider, Director, RTB Platform Operations, Centro; Jan Winkler, Executive Director, AdSpirit; Jim Butler, Chief Technology Officer, Global Supply Platforms, Verizon Media Group / Oath; Jud Spencer, Principal Lead Software Engineer, The Trade Desk; Madeleine Gordon, Technical Writer, Google; Neal Richter, CTO, Rakuten Marketing, and IAB Tech Lab OpenRTB Co-Chair; Per Bjorke, Senior Product Manager, Google; Sam Tingleff, Chief Technology Officer, IAB Tech Lab; Sergio Serra, Senior Product Manager, InMobi

IAB Tech Lab Contact:
Jennifer Derke, Director of Product, Programmatic & Data, IAB Tech Lab
openRTB@iabtechlab.com

Contributors and Technical Governance

OpenRTB Working Group members provide contributions to ads.txt initiative. Participants in the OpenRTB Working group must be members of IAB Tech Lab. Technical Governance and code commits for the project are provided by the IAB Tech Lab OpenRTB Commit Group.

License

TABLE OF CONTENTS

Abstract

Authorized Sellers for Apps (app-ads.txt) is an extension to the Authorized Digital Sellers (ads.txt) standard, originally designed for protecting web ad inventory. It extends compatibility to support apps distributed through online app stores, linking app store listings to app developer websites.

To accomplish this, we outline a standard protocol for obtaining the app developer’s website URL from an app listing page in an app store, designed to have minimal implementation burden for stores. App developers publishing authorizations in an app-ads.txt file on the developer’s website centralize this configuration into an online resource that the developer independently controls. Using a developer domain creates a universal namespace, which may help identify and block instances of unauthorized developer impersonation.

Outside of providing this app-to-developer domain link, participants should view the handling of the domain-to-authorized seller enforcement as nearly identical to that used in ads.txt for web inventory (with exception involving subdomain handling).

Definitions

Authorized seller verifier: An entity that checks the authorized seller status of some ad inventory, such as a buyer or seller ad platform.

App metadata: The information about the app that is available in the app store. This can include app icon, name, description, screenshots, and developer information including website, etc. Typically, the app store will provide a page within the store's website which provides this information.

bundle_id: A platform-specific application identifier intended to be unique to the app and independent of the app store where it was distributed or the exchange the inventory is transacted through. On Android, this should be a bundle or package name (e.g., com.foo.mygame). On iOS, it follows a similar pattern (e.g., com.apple.mobilenotes). Note that bundle_id is named "bundle" in the app object in OpenRTB 2.5 and AdCOM 1.0.

store_id: An app store specific identifier representing the stock keeping unit (SKU) or another identifier used to locate the app within the specific store. For example, this may be the Amazon Stock Identification Number (ASIN) such as B00BN3YZM2, or the iTunes numeric store ID such as 1110145109. (Note that store_id may also be passed in the "bundle" field of the App object in OpenRTB 2.5 and below. In AdCOM 1.0, it is found in the "storeid" field of the App object.)

storeurl: An app store URL for an installed app that is provided on the App object in OpenRTB bid requests, required for IQG 2.1 compliance. For example, a bid request might list a storeurl of `https://play.google.com/store/apps/details?id=com.google.android.deskclock` for an Android app listed in Google Play, `https://itunes.apple.com/us/app/id1110145109` for an iOS app listed in the Apple iTunes store, or `https://channelstore.roku.com/details/151908/the-roku-channel` for a connected TV app listed in the Roku Channel Store.

Note regarding relationship between bundle_id, store_id, and storeurl: Although for Amazon and Apple iTunes app store the storeurl contains the store_id (example: `https://www.amazon.com/Amazon-App-Tester/dp/B00BN3YZM2`), this is not the case for all app stores. For example, Google Play app store's storeurl does not include the store_id but rather the bundle_id. Implementers should treat store URLs as opaque values, and it is not recommended to interpret structure from the URL parameter content for the purposes of using this specification.

Introduction

Before reading this document, we recommend that readers become familiar with the main ads.txt specification found at https://iabtechlab.com/ads-txt/, as this document relies heavily on the details outlined in the ads.txt specification.

The Authorized Sellers for Apps (app-ads.txt) specification details the following process:

- App developers provide a website URL in their app's store metadata and publish an app-ads.txt file on that website that lists authorized sellers of their app's ad inventory.
- Ad networks/sell-side platforms provide store listing URL on bid requests to facilitate enforcement of authorized seller status.
- App stores publish app metadata in a standard HTML `<meta>` tag on the app store's listing page for the app so that it can be crawled and parsed as structured data.
- Authorized seller verifiers crawl app stores to find developer website information, crawl developer websites to obtain and interpret app-ads.txt files, and enforce authorization status on inventory

The Resource Description Framework in attributes (RDFa) W3C Recommendation and the Open Graph protocol inspired this specification.

Solution specification

The following specifies requirements for app developers, ad networks/sell-side platforms, buy-side platforms, and app stores.

App developers

App developers must follow these steps to adopt app-ads.txt for their apps.

Provide developer website URL in app store listings

This specification relies on the presence of a developer website URL within the app's store listing metadata in all app stores distributing the app. Many app stores currently collect the developer's website URL to display as developer contact information within the store. Publishing a website and providing its URL is required for the app's ad inventory to participate in the authorized seller scheme.

Using this website URL, interested crawlers will derive a path and attempt to crawl an app-ads.txt file on the corresponding domain. Please see the "translate developer URL to an app-ads.txt path" section for detailed description of how verifiers will derive the location of an "/app-ads.txt" path from the published developer URL. Also see Appendix B for examples of how developer website URLs will be translated to app-ads.txt URLs and the order in which subdomains will be searched for the file.

Publish an app-ads.txt file

Refer to the main ads.txt standard located at https://iabtechlab.com/ads-txt/ for complete details about the proper format and contents of an ads.txt file for the web. Use the same guidelines as

the ads.txt specification, with the exception that the “subdomain” directive is unused in app-ads.txt files and should be ignored if encountered.

The file name is “app-ads.txt” as opposed to “ads.txt” so that app and web configurations are managed separately and do not create conflicts between each other. Due to the nature of apps having a different deployment model than web, we anticipate that this flexibility will ease adoption and maintenance compared to combining entries in one consolidated file.

Changing the developer website URL for an app

To reduce the burden on app stores, this specification asks crawlers to limit crawling frequency of the store website. Developers should anticipate that changes to the developer URL domain may take some time to be recognized by interested authorized seller verifiers. When changing domains, where possible, we recommend hosting the app-ads.txt file containing relevant entries on both the old and new domain for an extended time period before removing the file from the old domain or removing required entries from the old location.

Ad networks/sell-side platforms (aka “bid request issuers”)

Ad networks and sell-side platforms (SSPs) that issue the RTB bid requests, for the inventory to be treated as authorized per this spec, must indicate the app’s distribution channel as follows.

Include the storeurl parameter within bid requests

TAG Inventory Quality Guidelines require that ad networks, ad exchanges, and sell-side platforms provide the store URL indicating the individual store listing corresponding to the app offering the impression. Any supply-side platform or exchange not currently providing this value must do so in order to comply with the app-ads.txt requirements. Any ad networks that perform authorized sellers checks on proprietary inventory must utilize the app store URL within the verification process.

App stores

App stores are asked to support the following capabilities to facilitate the app-ads.txt standard.

Publish structured app information

To facilitate authorized seller verifiers, we request that app stores publish three HTML `<meta>` tags with the store listing page for each individual app:

- The app developer’s website URL (often currently provided as a user-clickable link on the store listing page)
- The app’s bundle_id
- The app’s store_id

The purpose of getting the bundle_id and/or store_id from the app's metadata in the app store is to cross-check that it matches the bundle_id and/or store_id in the bid request. App stores are requested to always expose the bundle_id and store_id meta tag for all app listings regardless of whether a developer_url was provided for that listing. This lets the app store assert that the store listing URL is authoritative for the given bundle ID/store ID and assists authorized seller verifiers in confirming that a given app is non-participating in app-ads.txt.

App stores must format the HTML meta tags as follows, inserting these tags into the <head> HTML tag at the beginning of the HTML doc and including the appropriate value in the content attribute:

```
<meta name="appstore:developer_url" content="https://www.path.to/page" />
<meta name="appstore:bundle_id" content="com.example.myapp" />
<meta name="appstore:store_id" content="SKU12345" />
```

This solution is similar to the Open Graph protocol or Twitter markup tags, where an "appstore:" prefix is used to illustrate the purpose of the fields. These names are registered in the WHATWG Wiki MetaExtensions page. Similar to the Twitter specification, we do not require that a formal compact namespace (CURIE) be defined for the prefix. App stores are assumed to produce valid HTML according to the W3C standards for the <meta> tag and surrounding document.

Authorized seller verifiers

Any entity that wants to verify authorized seller status for informational, reporting, or enforcement reasons should follow these steps. These steps apply to all parties in the ecosystem - ranging from sell-side platforms via exchanges to buy-side platforms.

Identify app store listing URLs for apps offering inventory

Authorized seller verifiers should determine the app store listing URLs as appropriate for the inventory they wish to verify. For OpenRTB, read the storeurl field from previously seen bid requests. For non-OpenRTB inventory, use a method appropriate for the proprietary solution.

Crawl app listing pages in app stores

Crawl the HTML pages as specified by the storeurl values of interest. Obtain the developer_url, store_id, and bundle_id properties from the HTML <meta> tags described in the App Stores section above, using an appropriate HTML parsing solution to extract the values.

Limit crawling so that unique app store URLs get crawled no more frequently than weekly and honor constraints in the store's robots.txt file. Verifiers should only crawl listings for apps where the verifier is actively receiving ad impression opportunities, rather than crawling the entire store inventory. Outside of the initial app-ads.txt adoption period, we do not anticipate that

developers will change app developer URLs frequently, and many URLs will rarely change if at all.

For resource efficiency and developer convenience, app stores or third-party aggregation services may offer APIs/file formats that provide a bulk transfer solution for obtaining these URLs. These alternative, proprietary solutions are welcome, but any first- or third-party solutions providing this service must surface the same developer URL, bundle ID, and store ID data as would appear in the standard app store URL location.

Translate developer URL to an app-ads.txt path

Follow these steps to transform the developer URL into a path to crawl for locating an app-ads.txt file.

1. Extract the host name portion of the URL.
2. Remove any “www.” or “m.” prefix present in the host name.
3. Remove all but the first (and, if present, second) name from the host name which precedes the standard public suffix. For example:
    a. example.com simply remains example.com
    b. subdomain.example.com remains subdomain.example.com
    c. another.subdomain.example.com becomes subdomain.example.com
    d. another.subdomain.example.co.uk becomes subdomain.example.co.uk
4. Append /app-ads.txt to that path.
5. Crawlers should attempt to fetch the HTTPS version of the URL first, falling back to the HTTP version if SSL is unavailable.

Ensure proper handling of standard public suffixes when canonicalizing domains. The publicsuffix.org website contains links to software libraries for various languages that you may consider using to assist with domain parsing, although implementers should be sure that the suffix list used by the library remains current. Refer to the text cases in the appendix, which cover proper parsing scenario examples. Verifiers must use the app-ads.txt file found on the subdomain according to the domain canonicalization rules described above, only defaulting to the app-ads.txt file found on the subdomain’s parent if no file is found on that subdomain.

Crawl and interpret app-ads.txt file

The structure and content of the app-ads.txt file is the same as ads.txt for web files, and should follow the established ads.txt standard. Platforms that have already implemented ads.txt for web enforcement solutions should be able to reuse this infrastructure for app-ads.txt with minor changes.

Requirements for implementing advertising systems

Follow the guidelines in the original ads.txt specification regarding utilization of the OpenRTB `publisher.id` field for locating the account identifier used to check authorization status. Then

follow the steps above to find, crawl, and interpret the app-ads.txt files. Seller and Buyer systems verifying authorized seller status will in general follow the same approach as for web-based inventory; with the exception of the methods to find the app-ads.txt file as described above.

Authoritative and canonical representation in app stores

App stores can have multiple valid storeurl values that point to the same canonical software bundle. For example, the Angry Birds app within the Apple App Store may contain country code parameters in the URL:

```
https://itunes.apple.com/fi/app/ab-classic/id343200656?mt=8
https://itunes.apple.com/us/app/ab-classic/id343200656?mt=8
https://itunes.apple.com/nl/app/ab-classic/id343200656?mt=8
```

Furthermore, there may be multiple valid query string parameter variants contained within the URL that can represent language codes, analytics tracking, and other variants.

Authorized seller verifiers should use the app store URLs for discovering the canonical bundle ID/store ID values. Authorized seller verifiers should use the canonical bundle/store ID value returned within the `appstore:bundle_id` or `appstore:store_id` parameter described below as the identifier indexed for ad serving and verification. Because of the potentially unlimited variants of store URLs that could identify the same app, we strongly recommend against attempting to index apps and developer URLs by the full store URL as the lookup key. We expect most implementers will create an index that maps from the store domain + bundle ID or store ID tuple to developer domain. For example:

```
itunes.apple.com:343200656 => rovio.com
```

Reporting

For transparency, authorized seller verifiers should report to their clients the domain used for locating the app-ads.txt file and the app store domain used to locate the corresponding developer website.

Implementer notes

Supply-side platforms should provide help center resources instructing developers on the proper way to add the appropriate entries to the correct app-ads.txt files.

Implementing web crawlers of any type can require a time-consuming engineering investment. Authorized seller verifiers can consider subscribing to a software-as-a-service solution which pre-aggregates the relevant app store listings and/or app-ads.txt file data, evaluating the build-versus-buy tradeoff. We anticipate that many participants already subscribe to app analytics data sources/APIs that currently publish developer URL metadata suitable for use for app-

ads.txt verification. We encourage these services to evaluate their implementation techniques and self-certify to their customers that the developer URL they surface matches the URL found in the standardized HTML `<meta>` tags described in this specification.

Because the developer's website listed for an app shouldn't change frequently, we do not anticipate that crawlers will need to frequently crawl app stores in trying to identify changes. Any developer URL changes that alter the domain hosting an app-ads.txt file for an app should allow for ample delay between the store listing update and old app-ads.txt file removal. Consider leaving the obsolete app-ads.txt file in place for an extended period (suggested 30 days, if possible) to avoid risk of non-coverage (if the file is removed) or unavailable inventory (if entries are removed from a remaining file).

Limitations and constraints

The app-ads.txt and ads.txt standards help prevent unauthorized platform clients from improperly using supply-side platforms and ad exchanges to offer ad inventory that they're not permitted to sell. These tools cannot stop an unscrupulous ad platform from misrepresenting an unauthorized publisher ID as an authorized one by changing the publisher ID prior to offering the impression on an ad exchange. Participants should be aware of this limitation. Developers should only introduce platforms and publisher IDs to their app-ads.txt file that they trust. Buyers should be aware that authorized inventory must still be screened and monitored for invalid traffic.

App developers who want to partition the authorized sellers between individual apps must use separate developer domains/subdomains, as the standard provides no provisions for indicating that an individual seller is only authorized for a subset of apps within an individual app-ads.txt file on that linked domain. We generally advise against attempting to exert fine-grained control using app-ads.txt.

Appendix A: Developer URL canonicalization test cases

These example inputs and expected outputs illustrate the desired developer URL to canonicalized domain transformation.

Test handling of a typical .com domain

https://www.example.com/test
example.com

https://m.example.com/test
example.com

https://example.com/test
example.com

https://subdomain.example.com/test
subdomain.example.com

https://another.subdomain.example.com/test
subdomain.example.com

Test handing of two-level public suffix

https://www.example.co.uk/test
example.co.uk

https://m.example.co.uk/test
example.co.uk

https://example.co.uk/test
example.co.uk

https://subdomain.example.co.uk/test
subdomain.example.co.uk

https://another.subdomain.example.co.uk/test
subdomain.example.co.uk

Test handling of a newer country public suffix registerable namespace

https://www.example.uk/test
example.uk

https://m.example.uk/test
example.uk

https://example.uk/test
example.uk

https://subdomain.example.uk/test
subdomain.example.uk

https://another.subdomain.example.uk/test
subdomain.example.uk

Appendix B: Developer URL to app-ads.txt file URL test cases

These example inputs and expected outputs illustrate the proper translation from developer URL to app-ads.txt path.

Test baseline developer URL

This test illustrates the baseline URL without any subdomain.

- Developer URL: https://example.com/test
- Verifier will crawl: https://example.com/app-ads.txt

Test developer URL with ignored www. subdomain

This test illustrates the normalization of the common “www” subdomain.

- Developer URL: https://www.example.com/test
- Verifier will crawl: https://example.com/app-ads.txt
- Confirm that crawler DOES NOT crawl: https://www.example.com/app-ads.txt

Test “m.” developer URL with ignored m. subdomain

This test illustrates the normalization of the common “m” subdomain.

- Developer URL: https://m.example.com/test
- Verifier will crawl: https://example.com/app-ads.txt
- Confirm that crawler DOES NOT crawl: https://m.example.com/app-ads.txt

Test developer URL with subdomain

This test illustrates how the subdomain will be used for locating an app-ads.txt file.

- Developer URL: https://subdomain.example.com/test
- Verifier will first crawl: https://subdomain.example.com/app-ads.txt
- If no file found, verifier will then crawl: https://example.com/app-ads.txt

Test developer URL with multiple subdomains

This test illustrates how only the first subdomain will be used for locating an app-ads.txt file.

- Developer URL: https://another.subdomain.example.com/test
- Verifier will first crawl: https://subdomain.example.com/app-ads.txt
- If no file found, verifier will then crawl: https://example.com/app-ads.txt

Test developer URL with subdomain on a multipart public suffix

This test illustrates how only the first subdomain will be used for locating an app-ads.txt file for a URL with a multipart public suffix.

- Developer URL: https://another.subdomain.example.co.uk/test
- Verifier will first crawl: https://subdomain.example.co.uk/app-ads.txt
- If no file found, verifier will then crawl: https://example.co.uk/app-ads.txt