Real Time Bidding (RTB) Project

OpenRTB API Specification Version 2.5

FINAL

December 2016

Introduction

The RTB Project, formerly known as the OpenRTB Consortium, assembled in November 2010 to develop a new API specification for companies interested in an open protocol for the automated trading of digital media across a broader range of platforms, devices, and advertising solutions. This document is OpenRTB version 2.5 released in November of 2016; this is the culmination of the working group efforts and can be found at: http://www.iab.com/openrtb

About the IAB Technology Lab

The IAB Technology Laboratory is a nonprofit research and development consortium charged with producing and helping companies implement global industry technical standards and solutions. The goal of the Tech Lab is to reduce friction associated with the digital advertising and marketing supply chain while contributing to the safe growth of an industry.

The IAB Tech Lab spearheads the development of technical standards, creates and maintains a code library to assist in rapid, cost-effective implementation of IAB standards, and establishes a test platform for companies to evaluate the compatibility of their technology solutions with IAB standards, which for 18 years have been the foundation for interoperability and profitable growth in the digital advertising supply chain.

The OpenRTB Work Group is a working group within the IAB Technology Lab. Further details about the IAB Technology Lab can be found at: https://iabtechlab.com/

IAB Contact Information

Jennifer Derke
Senior Manager, Product
IAB Technology Lab
openRTB@iab.com

OpenRTB Co-Chairs

Dr. Bill Simmons
CTO – DataXu
OpenRTB Founder

Dr. Jim Butler
VP Engineering, Publisher Platforms – AOL
OpenRTB Specification Author

Dr. Neal Richter
Founder, Principal – Hebbian Labs

Table of Contents

Getting Started

This specification contains a detailed explanation of an RTB (Real-Time Bidding) interface. Not all objects are required, and each object may contain a number of optional parameters. To assist a first-time reader of the specification, we have indicated which fields are essential to support a minimum viable real time bidding interface for various scenarios (banner, video, etc.).

A minimal viable interface should include the **required** and **recommended** parameters, but the scope for these parameters may be limited to specific scenarios. In these cases, the Description column may further qualify their **required** or **recommended** status. Optional parameters may be included to ensure maximum value is derived by the parties.

	Attribute	Type	Description
Examples of required attributes. Grouped at the tops of tables for convenience.	`id`	string; required	. . .
	`imp`	object array; required	. . .
Examples of recommended attributes. Grouped after required attributes.	`site`	object; recommended	. . .
	`app`	object; recommended	. . .
	`device`	object; recommended	. . .
	`user`	object; recommended	. . .
Examples of optional attributes, with and without defaults. Attributes are assumed optional unless explicitly qualified as required or recommended.	`test`	integer; default 0	. . .
	`at`	integer; default 2	. . .
	`tmax`	integer	. . .
	`wseat`	string array	. . .

Figure 1: Example of how Required, Recommended, and Optional attributes are presented.

IMPORTANT: Since **recommended** attributes are not required, they may not be available from all supply sources. It is suggested that all parties to OpenRTB transaction develop an integration checklist to identify which attributes the supply side supports in the bid request, and which attributes the demand side requires for ad decisioning.

1. Introduction

1.1 Mission / Overview

The mission of the OpenRTB project is to spur growth in Real-Time Bidding (RTB) marketplaces by providing open industry standards for communication between buyers of advertising and sellers of publisher inventory. There are several aspects to these standards including but not limited to the actual real-time bidding protocol, information taxonomies, offline configuration synchronization, and many more.

This document specifies a standard for the Real-Time Bidding Interface that has grown out of previous OpenRTB collaboration on the "block list project" and the "OpenRTB Mobile" project. These protocol standards aim to simplify the connection between suppliers of publisher inventory (i.e., exchanges, networks working with publishers, and sell-side platforms) and competitive buyers of that inventory (i.e., bidders, demand side platforms, or networks working with advertisers).

The overall goal of OpenRTB is and has been to create a *lingua franca* for communicating between buyers and sellers. The intent is not to regulate exactly how each business operates. As a project, we aim to make integration between parties easier, so that innovation can happen at a deeper-level at each of the businesses in the ecosystem.

1.2 History of OpenRTB

OpenRTB was launched as a pilot project between three demand-side platforms (DataXu, MediaMath, and Turn) and three sell-side platforms (Admeld, PubMatic, and The Rubicon Project) in November 2010. The first goal was to standardize communication between parties for exchanging block lists. Version 1.0 of the OpenRTB block list specification was released in December 2010.

After a positive response from the industry, Nexage approached the OpenRTB project with a proposal to create an API specification for OpenRTB focusing on the actual real-time bid request/response protocol and specifically to support mobile advertising. The mobile subcommittee was formed between companies representing the buy-side (DataXu, Fiksu, and [X+1]) and companies representing the sell-side (Nexage, Pubmatic, Smaato, and Jumptap). This project resulted in the OpenRTB Mobile 1.0 specification, which was released in February 2011.

Following the release of the mobile specification, a video subcommittee was formed with video ad exchanges (BrightRoll and Adap.tv) collaborating with DataXu and ContextWeb to incorporate support for video. The goal was to incorporate support for display, video, and mobile in one document. This effort resulted in OpenRTB 2.0, which was released as a unified standard in June 2011.

Due to very widespread adoption by the industry, OpenRTB was adopted as an IAB standard in January 2012 with the release of version 2.1. Governance over the technical content of the specification remains with the OpenRTB community and its governance rules.

1.3 Version History

OpenRTB Real-Time Bidding API

2.5 Support for header bidding, billing and loss notifications, Flex Ads, Payment ID, tactic ID, impression metrics, out-stream video, and many more minor enhancements.

2.4 Support for Audio ad units and the largest set of minor to moderate enhancements in v2.x history.

2.3 Support for Native ad units and multiple minor enhancements.

2.2 New enhancements for private marketplace direct deals, video, mobile, and regulatory signals.

2.1 Revisions for IQG compliance, minor enhancements, and corrections.

2.0 Combines display, mobile, and video standards into a unified specification.

1.0 Original Release of OpenRTB Mobile.

OpenRTB Display Block List Branch

1.2 Publisher Preferences API *(proposed)*.

1.1 Minor edits to include real-time exchange of creative attributes.

1.0 Original Release of OpenRTB block list specifications.

1.4 Resources

OpenRTB GitHub Repository	github.com/openrtb/OpenRTB/
Development Community Mailing List	groups.google.com/group/openrtb-dev
User Community Mailing List	groups.google.com/group/openrtb-user

1.5 Terminology

The following terms are used throughout this document specifically in the context of the OpenRTB Interface and this specification.

Term	Definition
RTB	Bidding for individual impressions in real-time (i.e., while a consumer is waiting).
Exchange	A service that conducts an auction among bidders per impression.
Bidder	An entity that competes in real-time auctions to acquire impressions.
Seat	An advertising entity (e.g., advertiser, agency) that wishes to obtain impressions and uses bidders to act on their behalf; a customer of a bidder and usually the owner of the advertising budget.
Publisher	An entity that operates one or more sites.
Site	Ad supported content including web and applications unless otherwise specified.
Deal	A pre-arranged agreement between a Publisher and a Seat to purchase impressions under certain terms.

2. OpenRTB Basics

The following figure illustrates the OpenRTB interactions between an exchange and its bidders. Ad requests originate at publisher sites or applications. For each inbound ad request, bid requests are broadcast to bidders, responses are evaluated under prevailing auction rules, and a winner is selected. The winning bidder is notified of the auction win via a win notice. Ad markup can either be included in the bid prospectively or in response to the win notice. A separate billing notice is also available to accommodate varying policies enacted by exchanges which are beyond the scope of the OpenRTB specification (e.g., billing on device delivery, viewability, etc.). The win notice informs the bidder's pricing algorithms of a success, whereas the billing notice indicates that spend should actually be applied. A loss notification is also available to inform the bidder of the reason their bid did not win.

The URLs for win, billing, and loss notices and the ad markup itself can contain any of several standard macros that enable the exchange to communicate critical data to the bidder (e.g., clearing price).

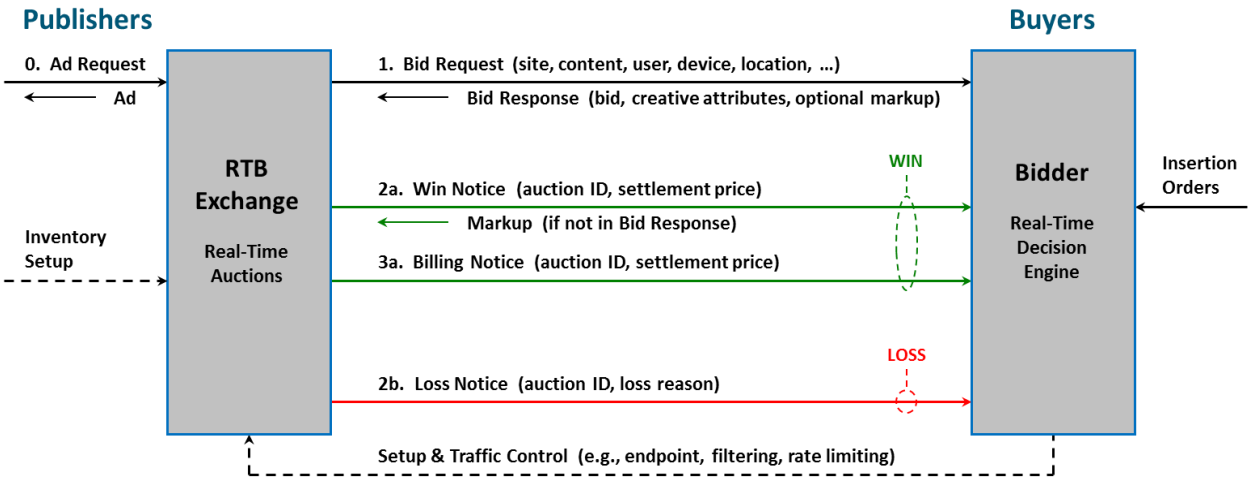

Figure 2: Reference Model - Request Sequence.

This specification focuses on the real-time interactions of bid request and response and the win, billing, and loss notices. Other interactions (e.g., block list synchronization, traffic control, creative review, etc.) are candidates for future OpenRTB initiatives or alternate projects.

2.1 Transport

The base protocol between an exchange and its bidders is HTTP. Specifically, HTTP POST is required for bid requests to accommodate greater payloads than HTTP GET and facilitate the use of binary representations. Win notices may be either POST or GET at the discretion of the exchange.

Calls returning content (e.g., any bid response, a win notice that returns markup) should return HTTP code 200. Calls returning no content in response to valid requests (e.g., an empty bid response which is one option for indicating no-bid, a win notice that does not return markup) should return HTTP 204. Invalid calls (e.g., a bid request containing a malformed or corrupt payload) should return HTTP 400 with no content.

BEST PRACTICE: One of the simplest and most effective ways of improving connection performance is to enable HTTP Persistent Connections, also known as Keep-Alive. This has a profound impact on overall performance by reducing connection management overhead as well as CPU utilization on both sides of the interface.

2.2 Security

HTTPS (i.e., secure HTTP) is not required for OpenRTB compliance. However, there is a growing trend in the industry to use HTTPS for added security of exchange/bidder communications. It is recommended, therefore, that exchanges and bidders consider supporting both HTTP and HTTPS.

2.3 Data Format

JSON (JavaScript Object Notation) is the suggested format for bid request and bid response data payloads. JSON was chosen for its combination of human readability and compactness. The data payloads are described in Section 3 and Section 4.

Optionally, an exchange may also offer binary representations (e.g., compressed JSON, ProtoBuf, Avro, etc.), which can be more efficient in terms of transmission time and bandwidth. The IAB Tech Lab may offer reference implementations for these or other formats. When available, the use of these IAB reference implementations is highly recommended to reduce exchange-specific variations.

The bid request specifies the representation as a mime type using the Content-Type HTTP header. The mime type for the standard JSON representation is "application/json" as shown. The format of the bid response must be the same as the bid request.

```
Content-Type: application/json
```

If alternative binary representations are used, the exchange or SSP should specify the Content-Type appropriately. For example: "Content-Type: avro/binary" or "Content-Type: application/x-protobuf". If the content-type is missing, the bidder should assume the type is application/json, unless a different default has been selected by an exchange.

As a convention, the absence of an attribute has a formal meaning. In most cases, this indicates that the value is unknown, unless otherwise specified.

2.4 Data Encoding

Compressing data sent between exchanges and bidders can be very beneficial. Compression greatly reduces the size of data transferred and thus saves network bandwidth for both exchanges and bidders. To realize this savings fully, compression should be enabled for both the bid request sent by the exchange and the bid response returned by the bidder.

Compression can be enabled on the bid response using standard HTTP 1.1 mechanisms. Most webservers already support gzip compression of response content and as such it is an ideal choice. For an exchange to signal they would like the response to be compressed, it should set the standard HTTP 1.1 Accept-Encoding header. The encoding value used should be "gzip".

```
Accept-Encoding: gzip
```

This header represents to bidders an indication by the exchange that it is capable of accepting gzip encoding for the response. If the bidder server supports this and is correctly configured, it will automatically respond with content that is gzip encoded. This will be indicated using the standard HTTP 1.1 Content-Encoding header.

```
Content-Encoding: gzip
```

To enable compression on the bid request, it must first be agreed upon between the exchange and the bidder that this is supported. This is similar to when a custom data format is used since the exchange has to know both format and encoding before sending the bid request. If the bidder supports it, the exchange should indicate it is sending a gzip compressed bid request by setting the HTTP 1.1 Content-Encoding header. The encoding value used should be "gzip".

```
Content-Encoding: gzip
```

If this header is not set then it is assumed that the request content isn't encoded. In HTTP 1.1, the Content-Encoding header is usually only used for response content. However by using this header for the request content as well we are able to indicate a request is compressed regardless of the data format used. This is useful since even binary data formats can benefit from being compressed.

2.5 OpenRTB Version HTTP Header

The OpenRTB Version should be passed in the header of a bid request with a custom header parameter. This will allow bidders to recognize the version of the message contained before attempting to parse the request.

Additionally, it is recommended albeit optional that bidders place an identically formatted message in the HTTP header of the response with the protocol version the bidder has implemented. The message may contain a different version number than the request header.

```
x-openrtb-version: 2.5
```

This version should be specified as <major>.<minor> (e.g., 2.5). First or second level increments on the version are changes to the protocol. In general, second-level changes should be backwards compatible, whereas first level changes need not be backwards compatible. Any third level revisions (such as 2.5.1) should not change the protocol itself; only descriptions and notes that don't affect the protocol content. Third level versions should not be included in this header since they should have no technical impact.

2.6 Privacy by Design

The OpenRTB project fully supports privacy policies as specified by buyers and sellers of advertising. In particular OpenRTB supports do-not-track (Section 3.2.18), COPPA restriction signaling (Section 7.5), and the ability to pass user preferences from sellers to buyers through the User object (Section 3.2.20).

2.7 Relationship to Inventory Quality Guidelines

OpenRTB is fully compatible with the Inventory Quality Guidelines (IQG) available here: www.tagtoday.net/iqg. In particular, many of the taxonomies and lists used in this specification are derived from either the IQG or the IAB Technology Lab.

2.8 Customization and Extensions

The OpenRTB spec allows for exchange specific customization and extensions of the specification. Any object may contain extensions. In order to keep extension fields consistent across platforms, they should consistently be named "`ext`".

3. Bid Request Specification

RTB transactions are initiated when an exchange or other supply source sends a bid request to a bidder. The bid request consists of the top-level bid request object, at least one impression object, and may optionally include additional objects providing impression context.

3.1 Object Model

Following is the object model for the bid request. The top-level object (i.e., in JSON the unnamed outer object) is denoted as `BidRequest` in the model. Of its direct subordinates, only `Imp` is technically required since it is fundamental to describing the impression being sold and its requires at least one of `Banner` (which may allow multiple formats), `Video`, `Audio`, and `Native` to define the type of impression (i.e., whichever one or more the publisher is willing to accept; although a bid will be for exactly one of those specified). An impression can optionally be subject to a private marketplace.

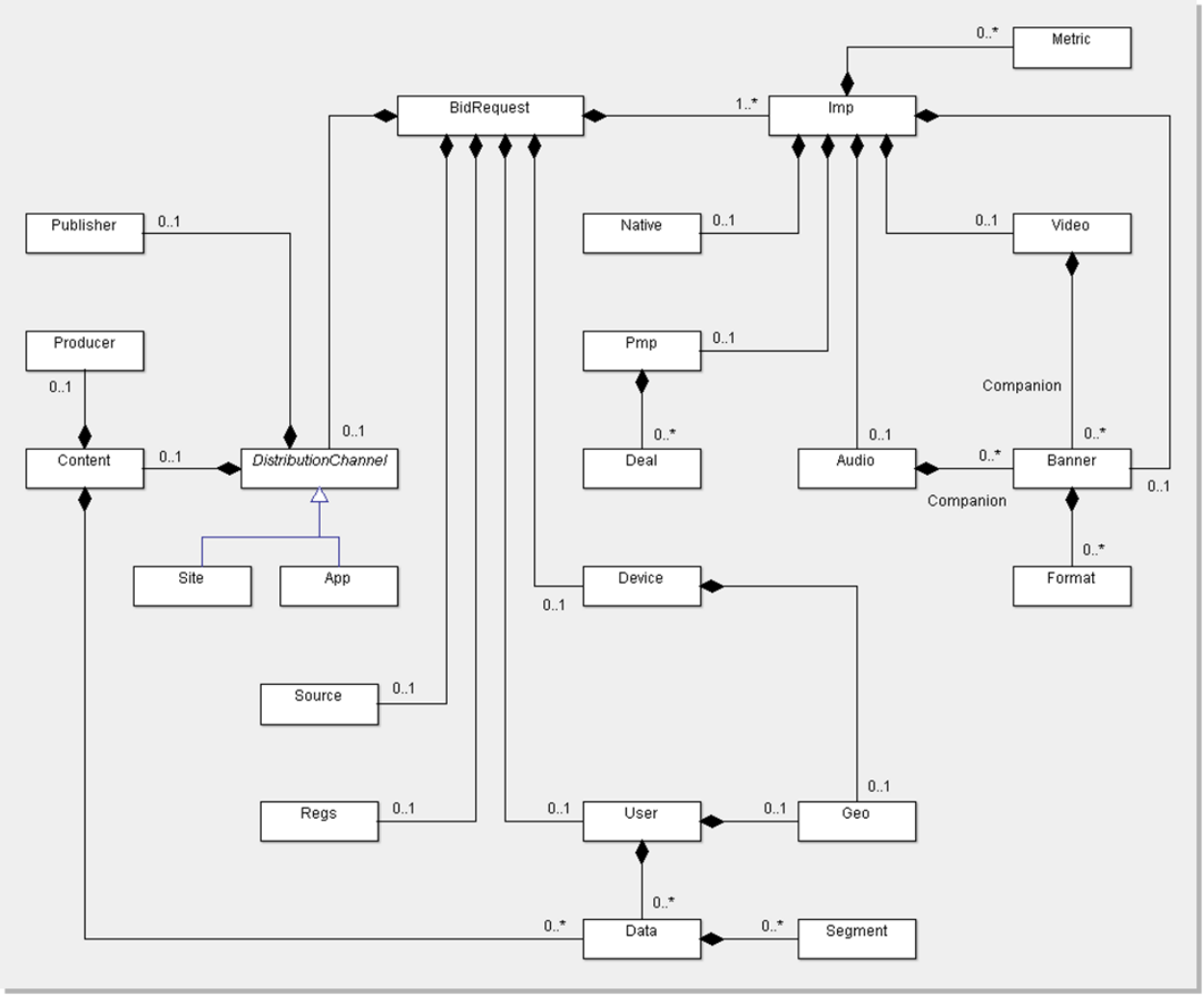

Figure 3: Bid Request object model.

Other subordinates to the `BidRequest` provide various forms of information to assist bidders in making targeting and pricing decisions. This includes details about the user, the device they're using, the location of either, regulatory constraints, and the content and media in which the impression will occur.

On the latter, there is the distinction between site (i.e., website) and application (i.e., non-browser app typically in mobile). The abstract class called `DistributionChannel` is just a modeling concept to indicate that a `BidRequest` is related to either a `Site` or an `App`, but not both (i.e., a distribution channel is an abstraction of site and app). Both sites and apps can be further described by data about their publisher, the content, and the content's producer.

Not shown in the model figure is an extensions object. This is an object of undefined structure that can be added to any other object to convey exchange-specific extensions to the standard. Exchanges using these objects are responsible for publishing their extensions to their bidders.

The following table summarizes the objects in the Bid Request model and serves as an index into the detailed definitions in the subsections that follow.

Object	Section	Description
`BidRequest`	3.2.1	Top-level object.
`Source`	3.2.2	Request source details on post-auction decisioning (e.g., header bidding).
`Regs`	3.2.3	Regulatory conditions in effect for all impressions in this bid request.
`Imp`	3.2.4	Container for the description of a specific impression; at least 1 per request.
`Metric`	3.2.5	A quantifiable often historical data point about an impression.
`Banner`	3.2.6	Details for a banner impression (incl. in-banner video) or video companion ad.
`Video`	3.2.7	Details for a video impression.
`Audio`	3.2.8	Container for an audio impression.
`Native`	3.2.9	Container for a native impression conforming to the Dynamic Native Ads API.
`Format`	3.2.10	An allowed size of a banner.
`Pmp`	3.2.11	Collection of private marketplace (PMP) deals applicable to this impression.
`Deal`	3.2.12	Deal terms pertaining to this impression between a seller and buyer.
`Site`	3.2.13	Details of the website calling for the impression.
`App`	3.2.14	Details of the application calling for the impression.
`Publisher`	3.2.15	Entity that controls the content of and distributes the site or app.
`Content`	3.2.16	Details about the published content itself, within which the ad will be shown.
`Producer`	3.2.17	Producer of the content; not necessarily the publisher (e.g., syndication).
`Device`	3.2.18	Details of the device on which the content and impressions are displayed.
`Geo`	3.2.19	Location of the device or user's home base depending on the parent object.
`User`	3.2.20	Human user of the device; audience for advertising.
`Data`	3.2.21	Collection of additional user targeting data from a specific data source.
`Segment`	3.2.22	Specific data point about a user from a specific data source.

3.2 Object Specifications

The subsections that follow define each of the objects in the bid request model. Several conventions are used throughout:

- Attributes are "required" if their omission would technically break the protocol.
- Some optional attributes are denoted "recommended" due to their elevated business importance.
- Unless a default value is explicitly specified, an omitted attribute is interpreted as "unknown".

3.2.1 Object: BidRequest

The top-level bid request object contains a globally unique bid request or auction ID. This `id` attribute is required as is at least one impression object (Section 3.2.4). Other attributes in this top-level object establish rules and restrictions that apply to all impressions being offered.

There are also several subordinate objects that provide detailed data to potential buyers. Among these are the `Site` and `App` objects, which describe the type of published media in which the impression(s) appear. These objects are highly recommended, but only one applies to a given bid request depending on whether the media is browser-based web content or a non-browser application, respectively.

Attribute	Type	Description
`id`	string; required	Unique ID of the bid request, provided by the exchange.
`imp`	object array; required	Array of `Imp` objects (Section 3.2.4) representing the impressions offered. At least 1 `Imp` object is required.
`site`	object; recommended	Details via a `Site` object (Section 3.2.13) about the publisher's website. Only applicable and recommended for websites.
`app`	object; recommended	Details via an `App` object (Section 3.2.14) about the publisher's app (i.e., non-browser applications). Only applicable and recommended for apps.
`device`	object; recommended	Details via a `Device` object (Section 3.2.18) about the user's device to which the impression will be delivered.
`user`	object; recommended	Details via a `User` object (Section 3.2.20) about the human user of the device; the advertising audience.
`test`	integer; default 0	Indicator of test mode in which auctions are not billable, where 0 = live mode, 1 = test mode.
`at`	integer; default 2	Auction type, where 1 = First Price, 2 = Second Price Plus. Exchange-specific auction types can be defined using values greater than 500.
`tmax`	integer	Maximum time in milliseconds the exchange allows for bids to be received including Internet latency to avoid timeout. This value supersedes any *a priori* guidance from the exchange.
`wseat`	string array	White list of buyer seats (e.g., advertisers, agencies) allowed to bid on this impression. IDs of seats and knowledge of the buyer's customers to which they refer must be coordinated between bidders and the exchange *a priori*. At most, only one of `wseat` and `bseat` should be used in the same request. Omission of both implies no seat restrictions.

bseat	string array	Block list of buyer seats (e.g., advertisers, agencies) restricted from bidding on this impression. IDs of seats and knowledge of the buyer's customers to which they refer must be coordinated between bidders and the exchange *a priori*. At most, only one of wseat and bseat should be used in the same request. Omission of both implies no seat restrictions.
allimps	integer; default 0	Flag to indicate if Exchange can verify that the impressions offered represent all of the impressions available in context (e.g., all on the web page, all video spots such as pre/mid/post roll) to support road-blocking. 0 = no or unknown, 1 = yes, the impressions offered represent all that are available.
cur	string array	Array of allowed currencies for bids on this bid request using ISO-4217 alpha codes. Recommended only if the exchange accepts multiple currencies.
wlang	string array	White list of languages for creatives using ISO-639-1-alpha-2. Omission implies no specific restrictions, but buyers would be advised to consider language attribute in the Device and/or Content objects if available.
bcat	string array	Blocked advertiser categories using the IAB content categories. Refer to List 5.1.
badv	string array	Block list of advertisers by their domains (e.g., "ford.com").
bapp	string array	Block list of applications by their platform-specific exchange-independent application identifiers. On Android, these should be bundle or package names (e.g., com.foo.mygame). On iOS, these are numeric IDs.
source	object	A Sorce object (Section 3.2.2) that provides data about the inventory source and which entity makes the final decision.
regs	object	A Regs object (Section 3.2.3) that specifies any industry, legal, or governmental regulations in force for this request.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.2 Object: Source

This object describes the nature and behavior of the entity that is the source of the bid request upstream from the exchange. The primary purpose of this object is to define post-auction or upstream decisioning when the exchange itself does not control the final decision. A common example of this is header bidding, but it can also apply to upstream server entities such as another RTB exchange, a mediation platform, or an ad server combines direct campaigns with 3rd party demand in decisioning.

Attribute	Type	Description
fd	Integer; recommended	Entity responsible for the final impression sale decision, where 0 = exchange, 1 = upstream source.
tid	string; recommended	Transaction ID that must be common across all participants in this bid request (e.g., potentially multiple exchanges).
pchain	string; recommended	Payment ID chain string containing embedded syntax described in the TAG Payment ID Protocol v1.0.

ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.3 Object: Regs

This object contains any legal, governmental, or industry regulations that apply to the request. The coppa flag signals whether or not the request falls under the United States Federal Trade Commission's regulations for the United States Children's Online Privacy Protection Act ("COPPA").

Attribute	Type	Description
coppa	integer	Flag indicating if this request is subject to the COPPA regulations established by the USA FTC, where 0 = no, 1 = yes. Refer to Section 7.5 for more information.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.4 Object: Imp

This object describes an ad placement or impression being auctioned. A single bid request can include multiple Imp objects, a use case for which might be an exchange that supports selling all ad positions on a given page. Each Imp object has a required ID so that bids can reference them individually.

The presence of Banner (Section 3.2.6), Video (Section 3.2.7), and/or Native (Section 3.2.9) objects subordinate to the Imp object indicates the type of impression being offered. The publisher can choose one such type which is the typical case or mix them at their discretion. However, any given bid for the impression must conform to one of the offered types.

Attribute	Type	Description
id	string; required	A unique identifier for this impression within the context of the bid request (typically, starts with 1 and increments.
metric	object array	An array of Metric object (Section 3.2.5).
banner	object	A Banner object (Section 3.2.6); required if this impression is offered as a banner ad opportunity.
video	object	A Video object (Section 3.2.7); required if this impression is offered as a video ad opportunity.
audio	object	An Audio object (Section 3.2.8); required if this impression is offered as an audio ad opportunity.
native	object	A Native object (Section 3.2.9); required if this impression is offered as a native ad opportunity.
pmp	object	A Pmp object (Section 3.2.11) containing any private marketplace deals in effect for this impression.
displaymanager	string	Name of ad mediation partner, SDK technology, or player responsible for rendering ad (typically video or mobile). Used by some ad servers to customize ad code by partner. Recommended for video and/or apps.

displaymanagerver	string	Version of ad mediation partner, SDK technology, or player responsible for rendering ad (typically video or mobile). Used by some ad servers to customize ad code by partner. Recommended for video and/or apps.
instl	integer; default 0	1 = the ad is interstitial or full screen, 0 = not interstitial.
tagid	string	Identifier for specific ad placement or ad tag that was used to initiate the auction. This can be useful for debugging of any issues, or for optimization by the buyer.
bidfloor	float; default 0	Minimum bid for this impression expressed in CPM.
bidfloorcur	string; default "USD"	Currency specified using ISO-4217 alpha codes. This may be different from bid currency returned by bidder if this is allowed by the exchange.
clickbrowser	integer	Indicates the type of browser opened upon clicking the creative in an app, where 0 = embedded, 1 = native. Note that the Safari View Controller in iOS 9.x devices is considered a native browser for purposes of this attribute.
secure	integer	Flag to indicate if the impression requires secure HTTPS URL creative assets and markup, where 0 = non-secure, 1 = secure. If omitted, the secure state is unknown, but non-secure HTTP support can be assumed.
iframebuster	string array	Array of exchange-specific names of supported iframe busters.
exp	integer	Advisory as to the number of seconds that may elapse between the auction and the actual impression.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.5 Object: Metric

This object is associated with an impression as an array of metrics. These metrics can offer insight into the impression to assist with decisioning such as average recent viewability, click-through rate, etc. Each metric is identified by its type, reports the value of the metric, and optionally identifies the source or vendor measuring the value.

Attribute	Type	Description
type	string; required	Type of metric being presented using exchange curated string names which should be published to bidders *a priori*.
value	float; required	Number representing the value of the metric. Probabilities must be in the range 0.0 – 1.0.
vendor	string; recommended	Source of the value using exchange curated string names which should be published to bidders *a priori*. If the exchange itself is the source versus a third party, "EXCHANGE" is recommended.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.6 Object: Banner

This object represents the most general type of impression. Although the term "banner" may have very specific meaning in other contexts, here it can be many things including a simple static image, an expandable ad unit, or even in-banner video (refer to the `Video` object in Section 3.2.7 for the more generalized and full featured video ad units). An array of `Banner` objects can also appear within the `Video` to describe optional companion ads defined in the VAST specification.

The presence of a `Banner` as a subordinate of the `Imp` object indicates that this impression is offered as a banner type impression. At the publisher's discretion, that same impression may also be offered as video, audio, and/or native by also including as `Imp` subordinates objects of those types. However, any given bid for the impression must conform to one of the offered types.

Attribute	Type	Description
`format`	object array; recommended	Array of format objects (Section 3.2.10) representing the banner sizes permitted. If none are specified, then use of the `h` and `w` attributes is highly recommended.
`w`	integer	Exact width in device independent pixels (DIPS); recommended if no `format` objects are specified.
`h`	integer	Exact height in device independent pixels (DIPS); recommended if no `format` objects are specified.
`wmax`	integer; DEPRECATED	*NOTE: Deprecated in favor of the `format` array.* Maximum width in device independent pixels (DIPS).
`hmax`	integer; DEPRECATED	*NOTE: Deprecated in favor of the `format` array.* Maximum height in device independent pixels (DIPS).
`wmin`	integer; DEPRECATED	*NOTE: Deprecated in favor of the `format` array.* Minimum width in device independent pixels (DIPS).
`hmin`	integer; DEPRECATED	*NOTE: Deprecated in favor of the `format` array.* Minimum height in device independent pixels (DIPS).
`btype`	integer array	Blocked banner ad types. Refer to List 5.2.
`battr`	integer array	Blocked creative attributes. Refer to List 5.3.
`pos`	integer	Ad position on screen. Refer to List 5.4.
`mimes`	string array	Content MIME types supported. Popular MIME types may include "`application/x-shockwave-flash`", "`image/jpg`", and "`image/gif`".
`topframe`	integer	Indicates if the banner is in the top frame as opposed to an iframe, where 0 = no, 1 = yes.
`expdir`	integer array	Directions in which the banner may expand. Refer to List 5.5.
`api`	integer array	List of supported API frameworks for this impression. Refer to List 5.6. If an API is not explicitly listed, it is assumed not to be supported.

id	string	Unique identifier for this banner object. Recommended when Banner objects are used with a Video object (Section 3.2.7) to represent an array of companion ads. Values usually start at 1 and increase with each object; should be unique within an impression.
vcm	integer	Relevant only for Banner objects used with a Video object (Section 3.2.7) in an array of companion ads. Indicates the companion banner rendering mode relative to the associated video, where 0 = concurrent, 1 = end-card.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.7 Object: Video

This object represents an in-stream video impression. Many of the fields are non-essential for minimally viable transactions, but are included to offer fine control when needed. Video in OpenRTB generally assumes compliance with the VAST standard. As such, the notion of companion ads is supported by optionally including an array of Banner objects (refer to the Banner object in Section 3.2.6) that define these companion ads.

The presence of a Video as a subordinate of the Imp object indicates that this impression is offered as a video type impression. At the publisher's discretion, that same impression may also be offered as banner, audio, and/or native by also including as Imp subordinates objects of those types. However, any given bid for the impression must conform to one of the offered types.

Attribute	Type	Description
mimes	string array; required	Content MIME types supported (e.g., "video/x-ms-wmv", "video/mp4").
minduration	integer; recommended	Minimum video ad duration in seconds.
maxduration	integer; recommended	Maximum video ad duration in seconds.
protocols	integer array; recommended	Array of supported video protocols. Refer to List 5.8. At least one supported protocol must be specified in either the protocol or protocols attribute.
protocol	integer; DEPRECATED	*NOTE: Deprecated in favor of protocols.* Supported video protocol. Refer to List 5.8. At least one supported protocol must be specified in either the protocol or protocols attribute.
w	integer; recommended	Width of the video player in device independent pixels (DIPS).
h	integer; recommended	Height of the video player in device independent pixels (DIPS).
startdelay	integer; recommended	Indicates the start delay in seconds for pre-roll, mid-roll, or post-roll ad placements. Refer to List 5.12 for additional generic values.
placement	integer	Placement type for the impression. Refer to List 5.9.

linearity	integer	Indicates if the impression must be linear, nonlinear, etc. If none specified, assume all are allowed. Refer to List 5.7.
skip	integer	Indicates if the player will allow the video to be skipped, where 0 = no, 1 = yes. If a bidder sends markup/creative that is itself skippable, the Bid object should include the attr array with an element of 16 indicating skippable video. Refer to List 5.3.
skipmin	integer; default 0	Videos of total duration greater than this number of seconds can be skippable; only applicable if the ad is skippable.
skipafter	integer; default 0	Number of seconds a video must play before skipping is enabled; only applicable if the ad is skippable.
sequence	integer	If multiple ad impressions are offered in the same bid request, the sequence number will allow for the coordinated delivery of multiple creatives.
battr	integer array	Blocked creative attributes. Refer to List 5.3.
maxextended	integer	Maximum extended ad duration if extension is allowed. If blank or 0, extension is not allowed. If -1, extension is allowed, and there is no time limit imposed. If greater than 0, then the value represents the number of seconds of extended play supported beyond the maxduration value.
minbitrate	integer	Minimum bit rate in Kbps.
maxbitrate	integer	Maximum bit rate in Kbps.
boxingallowed	integer; default 1	Indicates if letter-boxing of 4:3 content into a 16:9 window is allowed, where 0 = no, 1 = yes.
playbackmethod	integer array	Playback methods that may be in use. If none are specified, any method may be used. Refer to List 5.10. Only one method is typically used in practice. As a result, this array may be converted to an integer in a future version of the specification. It is strongly advised to use only the first element of this array in preparation for this change.
playbackend	integer	The event that causes playback to end. Refer to List 5.11.
delivery	integer array	Supported delivery methods (e.g., streaming, progressive). If none specified, assume all are supported. Refer to List 5.15.
pos	integer	Ad position on screen. Refer to List 5.4.
companionad	object array	Array of Banner objects (Section 3.2.6) if companion ads are available**.**
api	integer array	List of supported API frameworks for this impression. Refer to List 5.6. If an API is not explicitly listed, it is assumed not to be supported.
companiontype	integer array	Supported VAST companion ad types. Refer to List 5.14. Recommended if companion Banner objects are included via the companionad array. If one of these banners will be rendered as an end-card, this can be specified using the vcm attribute with the particular banner (Section 3.2.6).
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.8 Object: Audio

This object represents an audio type impression. Many of the fields are non-essential for minimally viable transactions, but are included to offer fine control when needed. Audio in OpenRTB generally assumes compliance with the DAAST standard. As such, the notion of companion ads is supported by optionally including an array of `Banner` objects (refer to the `Banner` object in Section 3.2.6) that define these companion ads.

The presence of a `Audio` as a subordinate of the `Imp` object indicates that this impression is offered as an audio type impression. At the publisher's discretion, that same impression may also be offered as banner, video, and/or native by also including as `Imp` subordinates objects of those types. However, any given bid for the impression must conform to one of the offered types.

Attribute	Type	Description
`mimes`	string array; required	Content MIME types supported (e.g., "`audio/mp4`").
`minduration`	integer; recommended	Minimum audio ad duration in seconds.
`maxduration`	integer; recommended	Maximum audio ad duration in seconds.
`protocols`	integer array; recommended	Array of supported audio protocols. Refer to List 5.8.
`startdelay`	integer; recommended	Indicates the start delay in seconds for pre-roll, mid-roll, or post-roll ad placements. Refer to List 5.12.
`sequence`	integer	If multiple ad impressions are offered in the same bid request, the sequence number will allow for the coordinated delivery of multiple creatives.
`battr`	integer array	Blocked creative attributes. Refer to List 5.3.
`maxextended`	integer	Maximum extended ad duration if extension is allowed. If blank or 0, extension is not allowed. If -1, extension is allowed, and there is no time limit imposed. If greater than 0, then the value represents the number of seconds of extended play supported beyond the `maxduration` value.
`minbitrate`	integer	Minimum bit rate in Kbps.
`maxbitrate`	integer	Maximum bit rate in Kbps.
`delivery`	integer array	Supported delivery methods (e.g., streaming, progressive). If none specified, assume all are supported. Refer to List 5.15.
`companionad`	object array	Array of `Banner` objects (Section 3.2.6) if companion ads are available**.**
`api`	integer array	List of supported API frameworks for this impression. Refer to List 5.6. If an API is not explicitly listed, it is assumed not to be supported.
`companiontype`	integer array	Supported DAAST companion ad types. Refer to List 5.14. Recommended if companion `Banner` objects are included via the `companionad` array.
`maxseq`	integer	The maximum number of ads that can be played in an ad pod.

feed	integer	Type of audio feed. Refer to List 5.16.
stitched	integer	Indicates if the ad is stitched with audio content or delivered independently, where 0 = no, 1 = yes.
nvol	integer	Volume normalization mode. Refer to List 5.17.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.9 Object: Native

This object represents a native type impression. Native ad units are intended to blend seamlessly into the surrounding content (e.g., a sponsored Twitter or Facebook post). As such, the response must be well-structured to afford the publisher fine-grained control over rendering.

The Native Subcommittee has developed a companion specification to OpenRTB called the Dynamic Native Ads API. It defines the request parameters and response markup structure of native ad units. This object provides the means of transporting request parameters as an opaque string so that the specific parameters can evolve separately under the auspices of the Dynamic Native Ads API. Similarly, the ad markup served will be structured according to that specification.

The presence of a `Native` as a subordinate of the `Imp` object indicates that this impression is offered as a native type impression. At the publisher's discretion, that same impression may also be offered as banner, video, and/or audio by also including as `Imp` subordinates objects of those types. However, any given bid for the impression must conform to one of the offered types.

Attribute	Type	Description
request	string; required	Request payload complying with the Native Ad Specification.
ver	string; recommended	Version of the Dynamic Native Ads API to which `request` complies; highly recommended for efficient parsing.
api	integer array	List of supported API frameworks for this impression. Refer to List 5.6. If an API is not explicitly listed, it is assumed not to be supported.
battr	integer array	Blocked creative attributes. Refer to List 5.3.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.10 Object: Format

This object represents an allowed size (i.e., height and width combination) or Flex Ad parameters for a banner impression. These are typically used in an array where multiple sizes are permitted. It is recommended that either the `w`/`h` pair or the `wratio`/`hratio`/`wmin` set (i.e., for Flex Ads) be specified.

Attribute	Type	Description
w	integer	Width in device independent pixels (DIPS).
h	integer	Height in device independent pixels (DIPS).
wratio	integer	Relative width when expressing size as a ratio.
hratio	integer	Relative height when expressing size as a ratio.

wmin	integer	The minimum width in device independent pixels (DIPS) at which the ad will be displayed the size is expressed as a ratio.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.11 Object: Pmp

This object is the private marketplace container for direct deals between buyers and sellers that may pertain to this impression. The actual deals are represented as a collection of `Deal` objects. Refer to Section 7.3 for more details.

Attribute	Type	Description
private_auction	integer; default 0	Indicator of auction eligibility to seats named in the Direct Deals object, where 0 = all bids are accepted, 1 = bids are restricted to the deals specified and the terms thereof.
deals	object array	Array of `Deal` (Section 3.2.12) objects that convey the specific deals applicable to this impression.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.12 Object: Deal

This object constitutes a specific deal that was struck *a priori* between a buyer and a seller. Its presence with the `Pmp` collection indicates that this impression is available under the terms of that deal. Refer to Section 7.3 for more details.

Attribute	Type	Description
id	string; required	A unique identifier for the direct deal.
bidfloor	float; default 0	Minimum bid for this impression expressed in CPM.
bidfloorcur	string; default "USD"	Currency specified using ISO-4217 alpha codes. This may be different from bid currency returned by bidder if this is allowed by the exchange.
at	integer	Optional override of the overall auction type of the bid request, where 1 = First Price, 2 = Second Price Plus, 3 = the value passed in `bidfloor` is the agreed upon deal price. Additional auction types can be defined by the exchange.
wseat	string array	Whitelist of buyer seats (e.g., advertisers, agencies) allowed to bid on this deal. IDs of seats and the buyer's customers to which they refer must be coordinated between bidders and the exchange *a priori*. Omission implies no seat restrictions.
wadomain	string array	Array of advertiser domains (e.g., advertiser.com) allowed to bid on this deal. Omission implies no advertiser restrictions.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.13 Object: Site

This object should be included if the ad supported content is a website as opposed to a non-browser application. A bid request must not contain both a `Site` and an `App` object. At a minimum, it is useful to provide a site ID or page URL, but this is not strictly required.

Attribute	Type	Description
`id`	string; recommended	Exchange-specific site ID.
`name`	string	Site name (may be aliased at the publisher's request).
`domain`	string	Domain of the site (e.g., "mysite.foo.com").
`cat`	string array	Array of IAB content categories of the site. Refer to List 5.1.
`sectioncat`	string array	Array of IAB content categories that describe the current section of the site. Refer to List 5.1.
`pagecat`	string array	Array of IAB content categories that describe the current page or view of the site. Refer to List 5.1.
`page`	string	URL of the page where the impression will be shown.
`ref`	string	Referrer URL that caused navigation to the current page.
`search`	string	Search string that caused navigation to the current page.
`mobile`	integer	Indicates if the site has been programmed to optimize layout when viewed on mobile devices, where 0 = no, 1 = yes.
`privacypolicy`	integer	Indicates if the site has a privacy policy, where 0 = no, 1 = yes.
`publisher`	object	Details about the Publisher (Section 3.2.15) of the site.
`content`	object	Details about the Content (Section 3.2.16) within the site.
`keywords`	string	Comma separated list of keywords about the site.
`ext`	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.14 Object: App

This object should be included if the ad supported content is a non-browser application (typically in mobile) as opposed to a website. A bid request must not contain both an `App` and a `Site` object. At a minimum, it is useful to provide an App ID or bundle, but this is not strictly required.

Attribute	Type	Description
`id`	string; recommended	Exchange-specific app ID.
`name`	string	App name (may be aliased at the publisher's request).
`bundle`	string	A platform-specific application identifier intended to be unique to the app and independent of the exchange. On Android, this should be a bundle or package name (e.g., com.foo.mygame). On iOS, it is typically a numeric ID.
`domain`	string	Domain of the app (e.g., "mygame.foo.com").
`storeurl`	string	App store URL for an installed app; for IQG 2.1 compliance.

cat	string array	Array of IAB content categories of the app. Refer to List 5.1.
sectioncat	string array	Array of IAB content categories that describe the current section of the app. Refer to List 5.1.
pagecat	string array	Array of IAB content categories that describe the current page or view of the app. Refer to List 5.1.
ver	string	Application version.
privacypolicy	integer	Indicates if the app has a privacy policy, where 0 = no, 1 = yes.
paid	integer	0 = app is free, 1 = the app is a paid version.
publisher	object	Details about the Publisher (Section 3.2.15) of the app.
content	object	Details about the Content (Section 3.2.16) within the app.
keywords	string	Comma separated list of keywords about the app.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.15 Object: Publisher

This object describes the publisher of the media in which the ad will be displayed. The publisher is typically the seller in an OpenRTB transaction.

Attribute	Type	Description
id	string	Exchange-specific publisher ID.
name	string	Publisher name (may be aliased at the publisher's request).
cat	string array	Array of IAB content categories that describe the publisher. Refer to List 5.1.
domain	string	Highest level domain of the publisher (e.g., "publisher.com").
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.16 Object: Content

This object describes the content in which the impression will appear, which may be syndicated or non-syndicated content. This object may be useful when syndicated content contains impressions and does not necessarily match the publisher's general content. The exchange might or might not have knowledge of the page where the content is running, as a result of the syndication method. For example might be a video impression embedded in an iframe on an unknown web property or device.

Attribute	Type	Description
id	string	ID uniquely identifying the content.
episode	integer	Episode number.
title	string	Content title. *Video Examples:* "Search Committee" (television), "A New Hope" (movie), or "Endgame" (made for web). *Non-Video Example:* "Why an Antarctic Glacier Is Melting So Quickly" (Time magazine article).

series	string	Content series. *Video Examples:* "The Office" (television), "Star Wars" (movie), or "Arby 'N' The Chief" (made for web). *Non-Video Example:* "Ecocentric" (Time Magazine blog).
season	string	Content season (e.g., "Season 3").
artist	string	Artist credited with the content.
genre	string	Genre that best describes the content (e.g., rock, pop, etc).
album	string	Album to which the content belongs; typically for audio.
isrc	string	International Standard Recording Code conforming to ISO-3901.
producer	object	Details about the content `Producer` (Section 3.2.17).
url	string	URL of the content, for buy-side contextualization or review.
cat	string array	Array of IAB content categories that describe the content producer. Refer to List 5.1.
prodq	integer	Production quality. Refer to List 5.13.
videoquality	integer; DEPRECATED	*Note: Deprecated in favor of `prodq`.* Video quality. Refer to List 5.13.
context	integer	Type of content (game, video, text, etc.). Refer to List 5.18.
contentrating	string	Content rating (e.g., MPAA).
userrating	string	User rating of the content (e.g., number of stars, likes, etc.).
qagmediarating	integer	Media rating per IQG guidelines. Refer to List 5.19.
keywords	string	Comma separated list of keywords describing the content.
livestream	integer	0 = not live, 1 = content is live (e.g., stream, live blog).
sourcerelationship	integer	0 = indirect, 1 = direct.
len	integer	Length of content in seconds; appropriate for video or audio.
language	string	Content language using ISO-639-1-alpha-2.
embeddable	integer	Indicator of whether or not the content is embeddable (e.g., an embeddable video player), where 0 = no, 1 = yes.
data	object array	Additional content data. Each `Data` object (Section 3.2.21) represents a different data source.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.17 Object: Producer

This object defines the producer of the content in which the ad will be shown. This is particularly useful when the content is syndicated and may be distributed through different publishers and thus when the producer and publisher are not necessarily the same entity.

Attribute	Type	Description
id	string	Content producer or originator ID. Useful if content is syndicated and may be posted on a site using embed tags.

name	string	Content producer or originator name (e.g., "Warner Bros").
cat	string array	Array of IAB content categories that describe the content producer. Refer to List 5.1.
domain	string	Highest level domain of the content producer (e.g., "producer.com").
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.18 Object: Device

This object provides information pertaining to the device through which the user is interacting. Device information includes its hardware, platform, location, and carrier data. The device can refer to a mobile handset, a desktop computer, set top box, or other digital device.

Attribute	Type	Description
ua	string; recommended	Browser user agent string.
geo	object; recommended	Location of the device assumed to be the user's current location defined by a Geo object (Section 3.2.19).
dnt	integer; recommended	Standard "Do Not Track" flag as set in the header by the browser, where 0 = tracking is unrestricted, 1 = do not track.
lmt	integer; recommended	"Limit Ad Tracking" signal commercially endorsed (e.g., iOS, Android), where 0 = tracking is unrestricted, 1 = tracking must be limited per commercial guidelines.
ip	string; recommended	IPv4 address closest to device.
ipv6	string	IP address closest to device as IPv6.
devicetype	integer	The general type of device. Refer to List 5.21.
make	string	Device make (e.g., "Apple").
model	string	Device model (e.g., "iPhone").
os	string	Device operating system (e.g., "iOS").
osv	string	Device operating system version (e.g., "3.1.2").
hwv	string	Hardware version of the device (e.g., "5S" for iPhone 5S).
h	integer	Physical height of the screen in pixels.
w	integer	Physical width of the screen in pixels.
ppi	integer	Screen size as pixels per linear inch.
pxratio	float	The ratio of physical pixels to device independent pixels.
js	integer	Support for JavaScript, where 0 = no, 1 = yes.
geofetch	integer	Indicates if the geolocation API will be available to JavaScript code running in the banner, where 0 = no, 1 = yes.
flashver	string	Version of Flash supported by the browser.
language	string	Browser language using ISO-639-1-alpha-2.

carrier	string	Carrier or ISP (e.g., "VERIZON") using exchange curated string names which should be published to bidders *a priori*.
mccmnc	string	Mobile carrier as the concatenated MCC-MNC code (e.g., "310-005" identifies Verizon Wireless CDMA in the USA). Refer to https://en.wikipedia.org/wiki/Mobile_country_code for further examples. Note that the dash between the MCC and MNC parts is required to remove parsing ambiguity.
connectiontype	integer	Network connection type. Refer to List 5.22.
ifa	string	ID sanctioned for advertiser use in the clear (i.e., not hashed).
didsha1	string	Hardware device ID (e.g., IMEI); hashed via SHA1.
didmd5	string	Hardware device ID (e.g., IMEI); hashed via MD5.
dpidsha1	string	Platform device ID (e.g., Android ID); hashed via SHA1.
dpidmd5	string	Platform device ID (e.g., Android ID); hashed via MD5.
macsha1	string	MAC address of the device; hashed via SHA1.
macmd5	string	MAC address of the device; hashed via MD5.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

BEST PRACTICE: There are currently no prominent open source lists for device makes, models, operating systems, or carriers. Exchanges typically use commercial products or other proprietary lists for these attributes. Until suitable open standards are available, exchanges are highly encouraged to publish lists of their device make, model, operating system, and carrier values to bidders.

BEST PRACTICE: Proper device IP detection in mobile is not straightforward. Typically it involves starting at the left of the `x-forwarded-for` header, skipping private carrier networks (e.g., `10.x.x.x` or `192.x.x.x`), and possibly scanning for known carrier IP ranges. Exchanges are urged to research and implement this feature carefully when presenting device IP values to bidders.

3.2.19 Object: Geo

This object encapsulates various methods for specifying a geographic location. When subordinate to a `Device` object, it indicates the location of the device which can also be interpreted as the user's current location. When subordinate to a `User` object, it indicates the location of the user's home base (i.e., not necessarily their current location).

The `lat`/`lon` attributes should only be passed if they conform to the accuracy depicted in the `type` attribute. For example, the centroid of a geographic region such as postal code should not be passed.

Attribute	Type	Description
lat	float	Latitude from -90.0 to +90.0, where negative is south.
lon	float	Longitude from -180.0 to +180.0, where negative is west.
type	integer	Source of location data; recommended when passing `lat`/`lon`. Refer to List 5.20.

accuracy	integer	Estimated location accuracy in meters; recommended when lat/lon are specified and derived from a device's location services (i.e., type = 1). Note that this is the accuracy as reported from the device. Consult OS specific documentation (e.g., Android, iOS) for exact interpretation.
lastfix	integer	Number of seconds since this geolocation fix was established. Note that devices may cache location data across multiple fetches. Ideally, this value should be from the time the actual fix was taken.
ipservice	integer	Service or provider used to determine geolocation from IP address if applicable (i.e., type = 2). Refer to List 5.23.
country	string	Country code using ISO-3166-1-alpha-3.
region	string	Region code using ISO-3166-2; 2-letter state code if USA.
regionfips104	string	Region of a country using FIPS 10-4 notation. While OpenRTB supports this attribute, it has been withdrawn by NIST in 2008.
metro	string	Google metro code; similar to but not exactly Nielsen DMAs. See Appendix A for a link to the codes.
city	string	City using United Nations Code for Trade & Transport Locations. See Appendix A for a link to the codes.
zip	string	Zip or postal code.
utcoffset	integer	Local time as the number +/- of minutes from UTC.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.20 Object: User

This object contains information known or derived about the human user of the device (i.e., the audience for advertising). The user `id` is an exchange artifact and may be subject to rotation or other privacy policies. However, this user ID must be stable long enough to serve reasonably as the basis for frequency capping and retargeting.

Attribute	Type	Description
id	string; recommended	Exchange-specific ID for the user. At least one of `id` or `buyeruid` is recommended.
buyeruid	string; recommended	Buyer-specific ID for the user as mapped by the exchange for the buyer. At least one of `buyeruid` or `id` is recommended.
yob	integer	Year of birth as a 4-digit integer.
gender	string	Gender, where "M" = male, "F" = female, "O" = known to be other (i.e., omitted is unknown).
keywords	string	Comma separated list of keywords, interests, or intent.
customdata	string	Optional feature to pass bidder data that was set in the exchange's cookie. The string must be in base85 cookie safe characters and be in any format. Proper JSON encoding must be used to include "escaped" quotation marks.

geo	object	Location of the user's home base defined by a Geo object (Section 3.2.19). This is not necessarily their current location.
data	object array	Additional user data. Each Data object (Section 3.2.21) represents a different data source.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.21 Object: Data

The data and segment objects together allow additional data about the related object (e.g., user, content) to be specified. This data may be from multiple sources whether from the exchange itself or third parties as specified by the id field. A bid request can mix data objects from multiple providers. The specific data providers in use should be published by the exchange *a priori* to its bidders.

Attribute	Type	Description
id	string	Exchange-specific ID for the data provider.
name	string	Exchange-specific name for the data provider.
segment	object array	Array of Segment (Section 3.2.22) objects that contain the actual data values.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

3.2.22 Object: Segment

Segment objects are essentially key-value pairs that convey specific units of data. The parent Data object is a collection of such values from a given data provider. The specific segment names and value options must be published by the exchange *a priori* to its bidders.

Attribute	Type	Description
id	string	ID of the data segment specific to the data provider.
name	string	Name of the data segment specific to the data provider.
value	string	String representation of the data segment value.
ext	object	Placeholder for exchange-specific extensions to OpenRTB.

4. Bid Response Specification

RTB responses contain bids that reference specific impressions within a bid request. Bids are in essence an offer to buy. The bid response consists of the top-level bid response object and optional objects that depict the specific bids. An empty HTTP response constitutes a no-bid and is in fact the most bandwidth friendly form of this signal although returning a response with a "no-bid reason" is encouraged. A malformed response or a response that contains no actual bids will also be interpreted as no-bid.

4.1 Object Model

Following is the object model for the bid response. The top-level object (i.e., in JSON the unnamed outer object) is denoted as `BidResponse` in the model. A bid response may contain bids from multiple "seats" (i.e., the buying entity upstream from the actual bidder). In fact a response may contain multiple bids from the same seat; typically but not necessarily from different campaigns. This can improve the seat's chances of winning since most exchanges enforce various block lists on behalf of their publishers.

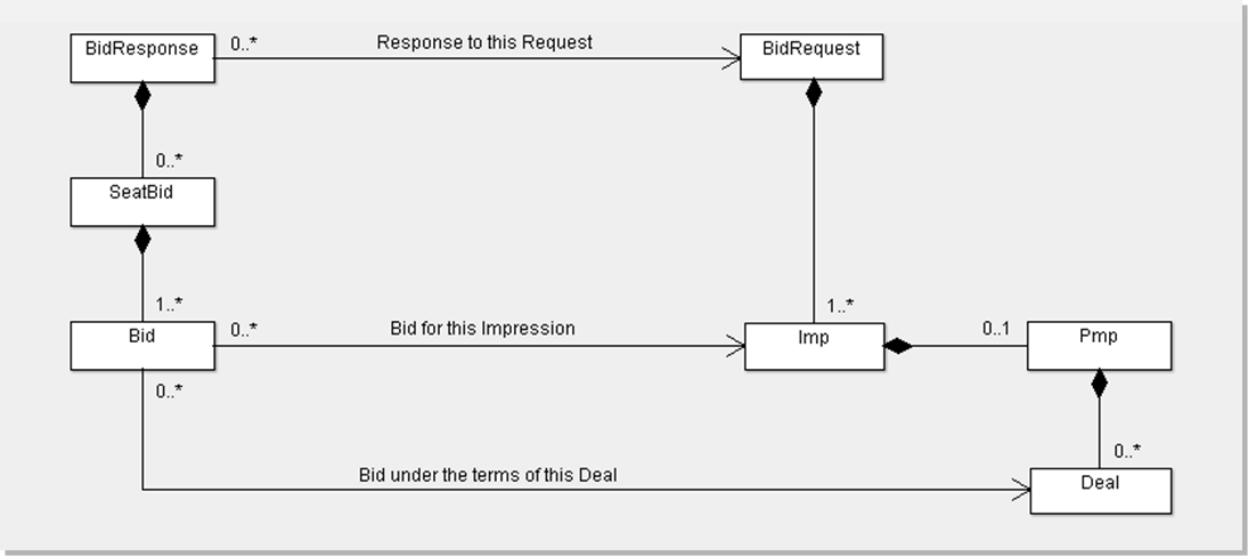

Figure 4: Bid Response object model.

Referring to the figure, the actual response objects are shown on the left, specifically the `BidResponse` top level object the seat specific `SeatBid` collections of `Bid` objects. The other objects shown are those objects from the bid request to which response objects related. Specifically, `BidResponse` includes the `BidRequest` ID for positive tracking purposes, and since a request can include multiple impressions `Bid` includes the ID of the `Imp` for which the bid is an offer to purchase. If a bid is made under the terms of a private marketplace deal, the `Bid` also includes the ID of the specific `Deal` object.

Not shown in the model figure is an extensions object. This is an object of undefined structure that can be added to any other object to convey bidder-specific extensions to the standard. Bidders using these objects are responsible for publishing their extensions to their exchanges.

The following table summarizes the objects in the Bid Response model and serves as an index into the detailed definitions in the subsections that follow.

Object	Section	Description
BidResponse	4.2.1	Top-level object.
SeatBid	4.2.2	Collection of bids made by the bidder on behalf of a specific seat.
Bid	4.2.3	An offer to buy a specific impression under certain business terms.

4.2 Object Specifications

The subsections that follow define each of the objects in the bid response model. Several conventions are used throughout:

- Attributes are "required" if their omission would technically break the protocol.
- Some optional attributes are denoted "recommended" due to their elevated business importance.
- Unless a default value is explicitly specified, an omitted attribute is interpreted as "unknown".

4.2.1 Object: BidResponse

This object is the top-level bid response object (i.e., the unnamed outer JSON object). The `id` attribute is a reflection of the bid request ID for logging purposes. Similarly, `bidid` is an optional response tracking ID for bidders. If specified, it can be included in the subsequent win notice call if the bidder wins. At least one `seatbid` object is required, which contains at least one bid for an impression. Other attributes are optional.

To express a "no-bid", the options are to return an empty response with HTTP 204. Alternately if the bidder wishes to convey to the exchange a reason for not bidding, just a `BidResponse` object is returned with a reason code in the `nbr` attribute.

Attribute	Type	Description
id	string; required	ID of the bid request to which this is a response.
seatbid	object array	Array of seatbid objects; 1+ required if a bid is to be made.
bidid	string	Bidder generated response ID to assist with logging/tracking.
cur	string; default "USD"	Bid currency using ISO-4217 alpha codes.
customdata	string	Optional feature to allow a bidder to set data in the exchange's cookie. The string must be in base85 cookie safe characters and be in any format. Proper JSON encoding must be used to include "escaped" quotation marks.
nbr	integer	Reason for not bidding. Refer to List 5.24.
ext	object	Placeholder for bidder-specific extensions to OpenRTB.

4.2.2 Object: SeatBid

A bid response can contain multiple `SeatBid` objects, each on behalf of a different bidder seat and each containing one or more individual bids. If multiple impressions are presented in the request, the `group` attribute can be used to specify if a seat is willing to accept any impressions that it can win (default) or if it is only interested in winning any if it can win them all as a group.

Attribute	Type	Description
`bid`	object array; required	Array of 1+ `Bid` objects (Section 4.2.3) each related to an impression. Multiple bids can relate to the same impression.
`seat`	string	ID of the buyer seat (e.g., advertiser, agency) on whose behalf this bid is made.
`group`	integer; default 0	0 = impressions can be won individually; 1 = impressions must be won or lost as a group.
`ext`	object	Placeholder for bidder-specific extensions to OpenRTB.

4.2.3 Object: Bid

A `SeatBid` object contains one or more Bid objects, each of which relates to a specific impression in the bid request via the `impid` attribute and constitutes an offer to buy that impression for a given `price`.

Attribute	Type	Description
`id`	string; required	Bidder generated bid ID to assist with logging/tracking.
`impid`	string; required	ID of the `Imp` object in the related bid request.
`price`	float; required	Bid price expressed as CPM although the actual transaction is for a unit impression only. Note that while the type indicates float, integer math is highly recommended when handling currencies (e.g., BigDecimal in Java).
`nurl`	string	Win notice URL called by the exchange if the bid wins (not necessarily indicative of a delivered, viewed, or billable ad); optional means of serving ad markup. Substitution macros (Section 4.4) may be included in both the URL and optionally returned markup.
`burl`	string	Billing notice URL called by the exchange when a winning bid becomes billable based on exchange-specific business policy (e.g., typically delivered, viewed, etc.). Substitution macros (Section 4.4) may be included.
`lurl`	string	Loss notice URL called by the exchange when a bid is known to have been lost. Substitution macros (Section 4.4) may be included. Exchange-specific policy may preclude support for loss notices or the disclosure of winning clearing prices resulting in ${AUCTION_PRICE} macros being removed (i.e., replaced with a zero-length string).
`adm`	string	Optional means of conveying ad markup in case the bid wins; supersedes the win notice if markup is included in both. Substitution macros (Section 4.4) may be included.

adid	string	ID of a preloaded ad to be served if the bid wins.
adomain	string array	Advertiser domain for block list checking (e.g., "ford.com"). This can be an array of for the case of rotating creatives. Exchanges can mandate that only one domain is allowed.
bundle	string	A platform-specific application identifier intended to be unique to the app and independent of the exchange. On Android, this should be a bundle or package name (e.g., com.foo.mygame). On iOS, it is a numeric ID.
iurl	string	URL without cache-busting to an image that is representative of the content of the campaign for ad quality/safety checking.
cid	string	Campaign ID to assist with ad quality checking; the collection of creatives for which `iurl` should be representative.
crid	string	Creative ID to assist with ad quality checking.
tactic	string	Tactic ID to enable buyers to label bids for reporting to the exchange the tactic through which their bid was submitted. The specific usage and meaning of the tactic ID should be communicated between buyer and exchanges *a priori*.
cat	string array	IAB content categories of the creative. Refer to List 5.1.
attr	integer array	Set of attributes describing the creative. Refer to List 5.3.
api	integer	API required by the markup if applicable. Refer to List 5.6.
protocol	integer	Video response protocol of the markup if applicable. Refer to List 5.8.
qagmediarating	integer	Creative media rating per IQG guidelines. Refer to List 5.19.
language	string	Language of the creative using ISO-639-1-alpha-2. The non-standard code "xx" may also be used if the creative has no linguistic content (e.g., a banner with just a company logo).
dealid	string	Reference to the `deal.id` from the bid request if this bid pertains to a private marketplace direct deal.
w	integer	Width of the creative in device independent pixels (DIPS).
h	integer	Height of the creative in device independent pixels (DIPS).
wratio	integer	Relative width of the creative when expressing size as a ratio. Required for Flex Ads.
hratio	integer	Relative height of the creative when expressing size as a ratio. Required for Flex Ads.
exp	integer	Advisory as to the number of seconds the bidder is willing to wait between the auction and the actual impression.
ext	object	Placeholder for bidder-specific extensions to OpenRTB.

For each bid, the `nurl` attribute contains the win notice URL. If the bidder wins the impression, the exchange calls this notice URL to inform the bidder of the win and to convey certain information using substitution macros (see Section 4.4) such as the clearing price. The win notice return or the `adm` attribute can be used to serve markup (see Section 4.3). In either case, the exchange will also apply the aforementioned substitution to any macros found in the markup.

BEST PRACTICE: The essential function of the win notice is to inform a bidder that they won an auction. It does not necessarily imply ad delivery, creative viewability, or billability. Exchanges are highly encouraged to publish to their bidders their event triggers, billing policies, and any other meaning they attach to the win notice. Also, please refer to Section 7.2 for additional guidance on expirations.

BEST PRACTICE: Firing of the billing notice should be server-side and as "close" as possible to where the exchange books revenue in order to minimize discrepancies between exchange and bidder.

BEST PRACTICE: For VAST Video, the IAB prescribes that the VAST impression event is the official signal that the impression is billable. If the `burl` attribute is specified, it too should be fired at the same time if the exchange is adhering to this policy. However, subtle technical issues may lead to additional discrepancies and bidders are cautioned to avoid this scenario.

Several other attributes are used for ad quality checks or enforcing publisher restrictions. These include the advertiser domain via `adomain`, a non-cache-busted URL to an image representative of the content of the campaign via `iurl`, an ID of the campaign and of the creative within the campaign via `cid` and `crid` respectively, an array of creative attribute via `attr`, and the dimensions via `h` and `w`. If the bid pertains to a private marketplace deal, the `dealid` attribute is used to reference that agreement from the bid request.

4.3 Ad Serving Options

The fulfilment of an RTB transaction within the scope of this OpenRTB specification lies in the delivery of markup. Depending on the impression and other ad type constraints, this markup can be XHTML, HTML5, XHTML or HTML5 with embedded JavaScript, a VAST document for video, a Native ad unit structure, and potentially other formats in the future.

The OpenRTB specification does not require any processing of the ad markup by the exchange other than macro substitution (refer to Section 4.4) and delivery to the supply-side. There are, however, multiple standard methods for transferring markup from the bidder to the exchange. The method used is at the discretion of the bidder, but an OpenRTB compliant exchange is expected to support all methods as defined in the subsections that follow.

4.3.1 Markup Served on the Win Notice

In this method, ad markup is returned to the exchange is via the win notice. In this case, the response body of the win notice call (i.e., invoking the `bid.nurl` attribute) contains the ad markup and only the ad markup; there must be no other structured data in the response body. Using this method, the `bid.adm` attribute must be omitted.

4.3.2 Markup Served in the Bid

In this method, ad markup is returned directly in the bid itself. This is accomplished via the `bid.adm` attribute. If both the `adm` attribute and win notice return data, the `adm` contents will take precedence.

4.3.3 Comparison of Ad Serving Approaches

Each of the ad serving methods has its own advantages that may be of varying importance to either the exchange or the bidder.

Ad Served on the Win Notice

- *Reduced Bandwidth Costs:* Serving ad markup only upon winning can save large amounts of bandwidth usage, the costs for which can be large at high volumes or when sending multiple bids per bid response.
- *Additional Bidder Flexibility:* Bidders may typically know the ad they will serve at the time of bid, but this provides an additional optional decision point after the clearing price has been established.

Ad Served in the Bid

- *Reduced Risk of Forfeiture:* A forfeit is the scenario in which a bidder wins, but forfeits due to failure to serve the ad markup. The risk of an additional HTTP failure (e.g., calling the win notice) is mitigated by this method.
- *Potential Concurrency:* The exchange can choose to return that ad markup and call the win notice concurrently, thereby improving user experience.

4.4 Substitution Macros

The win notice and billing notice URLs and their format are defined by the bidder. In order for the exchange to convey certain information to the bidder (e.g., the clearing price), a number of substitution macros can be inserted into these URLs. Prior to calling a win or billing notice URL, the exchange will search the specified URL for any of the defined macros and replace them with the appropriate data. Note that the substitution is simple in the sense that wherever a legal macro is found, it will be replaced without regard for syntax correctness. Furthermore, if the source value is an optional parameter that was not specified, the macro will simply be removed (i.e., replaced with a zero-length string).

These same substitution macros can also be placed in the ad markup. The exchange will perform the same data substitutions as in the aforementioned notice URLs. This occurs irrespective of whether the markup is returned on the win notice or passed in the `bid.adm` attribute of the bid response. A use case for macros in the ad markup might be when a bidder prefers to receive its win notification from the device itself. To accomplish this, the bidder would include a tracking pixel in the ad markup, the URL for which would include any of the available macros.

Macro	Description
`${AUCTION_ID}`	ID of the bid request; from `BidRequest.id` attribute.
`${AUCTION_BID_ID}`	ID of the bid; from `BidResponse.bidid` attribute.
`${AUCTION_IMP_ID}`	ID of the impression just won; from `imp.id` attribute.
`${AUCTION_SEAT_ID}`	ID of the bidder seat for whom the bid was made.
`${AUCTION_AD_ID}`	ID of the ad markup the bidder wishes to serve; from `bid.adid` attribute.
`${AUCTION_PRICE}`	Clearing price using the same currency and units as the bid.
`${AUCTION_CURRENCY}`	The currency used in the bid (explicit or implied); for confirmation only.
`${AUCTION_MBR}`	Market Bid Ratio defined as: clearance price / bid price.
`${AUCTION_LOSS}`	Loss reason codes. Refer to List 5.25.

Note that OpenRTB compliance exchanges must support all macros for which data is available and support substitution in both markup and URLs for win and billing notification.

BEST PRACTICE: When rendering markup for test or ad quality purposes, some macro values (e.g., clearing price) may not be known. In these cases, substitute "AUDIT" as the macro value.

Prior to substitution, macro data values can be encoded for security purposes using various obfuscation or encryption algorithms. This may be of particular interest for use cases such as the foregoing where price information is carried beyond the exchange, through the publisher, and into the device browser via a tracking pixel in the markup.

To specify that a particular macro is to be encoded, the suffix "`:X`" should be appended to the macro name, where X is a string that indicates the algorithm to be used. Algorithms choices are not defined by this specification and must be mutually agreed upon between exchange and bidder. As an example, suppose that the price macro is to be encoded using Base64 and that its code is "`B64`". The macro would then be written as follows:

```
${AUCTION_PRICE:B64}
```

BEST PRACTICE: Encoding of macro data should be used sparingly due to the additional processing overhead. For communications strictly between exchange and bidder (e.g., a win notice called from the exchange), encoding is generally considered unnecessary.

5. Enumerated Lists Specification

All reference lists are actively maintained by the IAB on the OpenRTB website. As such, implementers should ensure they are working from the latest lists and enumerations.

5.1 Content Categories

The following list represents the IAB's contextual taxonomy for categorization. Standard IDs have been adopted to easily support the communication of primary and secondary categories for various objects. This OpenRTB table has values derived from the IAB Tech Lab Content Taxonomy. Practitioners should keep in sync with updates as published on www.iab.com.

Value	Description
IAB1	**Arts & Entertainment**
IAB1-1	Books & Literature
IAB1-2	Celebrity Fan/Gossip
IAB1-3	Fine Art
IAB1-4	Humor
IAB1-5	Movies
IAB1-6	Music
IAB1-7	Television
IAB2	**Automotive**
IAB2-1	Auto Parts
IAB2-2	Auto Repair
IAB2-3	Buying/Selling Cars
IAB2-4	Car Culture
IAB2-5	Certified Pre-Owned
IAB2-6	Convertible
IAB2-7	Coupe
IAB2-8	Crossover
IAB2-9	Diesel
IAB2-10	Electric Vehicle
IAB2-11	Hatchback
IAB2-12	Hybrid
IAB2-13	Luxury
IAB2-14	Minivan
IAB2-15	Motorcycles
IAB2-16	Off-Road Vehicles
IAB2-17	Performance Vehicles

IAB2-18	Pickup
IAB2-19	Road-Side Assistance
IAB2-20	Sedan
IAB2-21	Trucks & Accessories
IAB2-22	Vintage Cars
IAB2-23	Wagon
IAB3	**Business**
IAB3-1	Advertising
IAB3-2	Agriculture
IAB3-3	Biotech/Biomedical
IAB3-4	Business Software
IAB3-5	Construction
IAB3-6	Forestry
IAB3-7	Government
IAB3-8	Green Solutions
IAB3-9	Human Resources
IAB3-10	Logistics
IAB3-11	Marketing
IAB3-12	Metals
IAB4	**Careers**
IAB4-1	Career Planning
IAB4-2	College
IAB4-3	Financial Aid
IAB4-4	Job Fairs
IAB4-5	Job Search
IAB4-6	Resume Writing/Advice
IAB4-7	Nursing
IAB4-8	Scholarships
IAB4-9	Telecommuting
IAB4-10	U.S. Military
IAB4-11	Career Advice
IAB5	**Education**
IAB5-1	7-12 Education
IAB5-2	Adult Education
IAB5-3	Art History
IAB5-4	College Administration
IAB5-5	College Life

IAB5-6	Distance Learning
IAB5-7	English as a 2nd Language
IAB5-8	Language Learning
IAB5-9	Graduate School
IAB5-10	Homeschooling
IAB5-11	Homework/Study Tips
IAB5-12	K-6 Educators
IAB5-13	Private School
IAB5-14	Special Education
IAB5-15	Studying Business
IAB6	**Family & Parenting**
IAB6-1	Adoption
IAB6-2	Babies & Toddlers
IAB6-3	Daycare/Pre School
IAB6-4	Family Internet
IAB6-5	Parenting - K-6 Kids
IAB6-6	Parenting teens
IAB6-7	Pregnancy
IAB6-8	Special Needs Kids
IAB6-9	Eldercare
IAB7	**Health & Fitness**
IAB7-1	Exercise
IAB7-2	ADD
IAB7-3	AIDS/HIV
IAB7-4	Allergies
IAB7-5	Alternative Medicine
IAB7-6	Arthritis
IAB7-7	Asthma
IAB7-8	Autism/PDD
IAB7-9	Bipolar Disorder
IAB7-10	Brain Tumor
IAB7-11	Cancer
IAB7-12	Cholesterol
IAB7-13	Chronic Fatigue Syndrome
IAB7-14	Chronic Pain
IAB7-15	Cold & Flu
IAB7-16	Deafness

IAB7-17	Dental Care
IAB7-18	Depression
IAB7-19	Dermatology
IAB7-20	Diabetes
IAB7-21	Epilepsy
IAB7-22	GERD/Acid Reflux
IAB7-23	Headaches/Migraines
IAB7-24	Heart Disease
IAB7-25	Herbs for Health
IAB7-26	Holistic Healing
IAB7-27	IBS/Crohn's Disease
IAB7-28	Incest/Abuse Support
IAB7-29	Incontinence
IAB7-30	Infertility
IAB7-31	Men's Health
IAB7-32	Nutrition
IAB7-33	Orthopedics
IAB7-34	Panic/Anxiety Disorders
IAB7-35	Pediatrics
IAB7-36	Physical Therapy
IAB7-37	Psychology/Psychiatry
IAB7-38	Senior Health
IAB7-39	Sexuality
IAB7-40	Sleep Disorders
IAB7-41	Smoking Cessation
IAB7-42	Substance Abuse
IAB7-43	Thyroid Disease
IAB7-44	Weight Loss
IAB7-45	Women's Health
IAB8	**Food & Drink**
IAB8-1	American Cuisine
IAB8-2	Barbecues & Grilling
IAB8-3	Cajun/Creole
IAB8-4	Chinese Cuisine
IAB8-5	Cocktails/Beer
IAB8-6	Coffee/Tea
IAB8-7	Cuisine-Specific

IAB8-8	Desserts & Baking
IAB8-9	Dining Out
IAB8-10	Food Allergies
IAB8-11	French Cuisine
IAB8-12	Health/Low-Fat Cooking
IAB8-13	Italian Cuisine
IAB8-14	Japanese Cuisine
IAB8-15	Mexican Cuisine
IAB8-16	Vegan
IAB8-17	Vegetarian
IAB8-18	Wine
IAB9	**Hobbies & Interests**
IAB9-1	Art/Technology
IAB9-2	Arts & Crafts
IAB9-3	Beadwork
IAB9-4	Bird-Watching
IAB9-5	Board Games/Puzzles
IAB9-6	Candle & Soap Making
IAB9-7	Card Games
IAB9-8	Chess
IAB9-9	Cigars
IAB9-10	Collecting
IAB9-11	Comic Books
IAB9-12	Drawing/Sketching
IAB9-13	Freelance Writing
IAB9-14	Genealogy
IAB9-15	Getting Published
IAB9-16	Guitar
IAB9-17	Home Recording
IAB9-18	Investors & Patents
IAB9-19	Jewelry Making
IAB9-20	Magic & Illusion
IAB9-21	Needlework
IAB9-22	Painting
IAB9-23	Photography
IAB9-24	Radio
IAB9-25	Roleplaying Games

IAB9-26	Sci-Fi & Fantasy
IAB9-27	Scrapbooking
IAB9-28	Screenwriting
IAB9-29	Stamps & Coins
IAB9-30	Video & Computer Games
IAB9-31	Woodworking
IAB10	**Home & Garden**
IAB10-1	Appliances
IAB10-2	Entertaining
IAB10-3	Environmental Safety
IAB10-4	Gardening
IAB10-5	Home Repair
IAB10-6	Home Theater
IAB10-7	Interior Decorating
IAB10-8	Landscaping
IAB10-9	Remodeling & Construction
IAB11	**Law, Government, & Politics**
IAB11-1	Immigration
IAB11-2	Legal Issues
IAB11-3	U.S. Government Resources
IAB11-4	Politics
IAB11-5	Commentary
IAB12	**News**
IAB12-1	International News
IAB12-2	National News
IAB12-3	Local News
IAB13	**Personal Finance**
IAB13-1	Beginning Investing
IAB13-2	Credit/Debt & Loans
IAB13-3	Financial News
IAB13-4	Financial Planning
IAB13-5	Hedge Fund
IAB13-6	Insurance
IAB13-7	Investing
IAB13-8	Mutual Funds
IAB13-9	Options
IAB13-10	Retirement Planning

IAB13-11	Stocks
IAB13-12	Tax Planning
IAB14	**Society**
IAB14-1	Dating
IAB14-2	Divorce Support
IAB14-3	Gay Life
IAB14-4	Marriage
IAB14-5	Senior Living
IAB14-6	Teens
IAB14-7	Weddings
IAB14-8	Ethnic Specific
IAB15	**Science**
IAB15-1	Astrology
IAB15-2	Biology
IAB15-3	Chemistry
IAB15-4	Geology
IAB15-5	Paranormal Phenomena
IAB15-6	Physics
IAB15-7	Space/Astronomy
IAB15-8	Geography
IAB15-9	Botany
IAB15-10	Weather
IAB16	**Pets**
IAB16-1	Aquariums
IAB16-2	Birds
IAB16-3	Cats
IAB16-4	Dogs
IAB16-5	Large Animals
IAB16-6	Reptiles
IAB16-7	Veterinary Medicine
IAB17	**Sports**
IAB17-1	Auto Racing
IAB17-2	Baseball
IAB17-3	Bicycling
IAB17-4	Bodybuilding
IAB17-5	Boxing
IAB17-6	Canoeing/Kayaking

IAB17-7	Cheerleading
IAB17-8	Climbing
IAB17-9	Cricket
IAB17-10	Figure Skating
IAB17-11	Fly Fishing
IAB17-12	Football
IAB17-13	Freshwater Fishing
IAB17-14	Game & Fish
IAB17-15	Golf
IAB17-16	Horse Racing
IAB17-17	Horses
IAB17-18	Hunting/Shooting
IAB17-19	Inline Skating
IAB17-20	Martial Arts
IAB17-21	Mountain Biking
IAB17-22	NASCAR Racing
IAB17-23	Olympics
IAB17-24	Paintball
IAB17-25	Power & Motorcycles
IAB17-26	Pro Basketball
IAB17-27	Pro Ice Hockey
IAB17-28	Rodeo
IAB17-29	Rugby
IAB17-30	Running/Jogging
IAB17-31	Sailing
IAB17-32	Saltwater Fishing
IAB17-33	Scuba Diving
IAB17-34	Skateboarding
IAB17-35	Skiing
IAB17-36	Snowboarding
IAB17-37	Surfing/Body-Boarding
IAB17-38	Swimming
IAB17-39	Table Tennis/Ping-Pong
IAB17-40	Tennis
IAB17-41	Volleyball
IAB17-42	Walking
IAB17-43	Waterski/Wakeboard

IAB17-44	World Soccer
IAB18	**Style & Fashion**
IAB18-1	Beauty
IAB18-2	Body Art
IAB18-3	Fashion
IAB18-4	Jewelry
IAB18-5	Clothing
IAB18-6	Accessories
IAB19	**Technology & Computing**
IAB19-1	3-D Graphics
IAB19-2	Animation
IAB19-3	Antivirus Software
IAB19-4	C/C++
IAB19-5	Cameras & Camcorders
IAB19-6	Cell Phones
IAB19-7	Computer Certification
IAB19-8	Computer Networking
IAB19-9	Computer Peripherals
IAB19-10	Computer Reviews
IAB19-11	Data Centers
IAB19-12	Databases
IAB19-13	Desktop Publishing
IAB19-14	Desktop Video
IAB19-15	Email
IAB19-16	Graphics Software
IAB19-17	Home Video/DVD
IAB19-18	Internet Technology
IAB19-19	Java
IAB19-20	JavaScript
IAB19-21	Mac Support
IAB19-22	MP3/MIDI
IAB19-23	Net Conferencing
IAB19-24	Net for Beginners
IAB19-25	Network Security
IAB19-26	Palmtops/PDAs
IAB19-27	PC Support
IAB19-28	Portable

IAB19-29	Entertainment
IAB19-30	Shareware/Freeware
IAB19-31	Unix
IAB19-32	Visual Basic
IAB19-33	Web Clip Art
IAB19-34	Web Design/HTML
IAB19-35	Web Search
IAB19-36	Windows
IAB20	**Travel**
IAB20-1	Adventure Travel
IAB20-2	Africa
IAB20-3	Air Travel
IAB20-4	Australia & New Zealand
IAB20-5	Bed & Breakfasts
IAB20-6	Budget Travel
IAB20-7	Business Travel
IAB20-8	By US Locale
IAB20-9	Camping
IAB20-10	Canada
IAB20-11	Caribbean
IAB20-12	Cruises
IAB20-13	Eastern Europe
IAB20-14	Europe
IAB20-15	France
IAB20-16	Greece
IAB20-17	Honeymoons/Getaways
IAB20-18	Hotels
IAB20-19	Italy
IAB20-20	Japan
IAB20-21	Mexico & Central America
IAB20-22	National Parks
IAB20-23	South America
IAB20-24	Spas
IAB20-25	Theme Parks
IAB20-26	Traveling with Kids
IAB20-27	United Kingdom
IAB21	**Real Estate**

IAB21-1	Apartments
IAB21-2	Architects
IAB21-3	Buying/Selling Homes
IAB22	**Shopping**
IAB22-1	Contests & Freebies
IAB22-2	Couponing
IAB22-3	Comparison
IAB22-4	Engines
IAB23	**Religion & Spirituality**
IAB23-1	Alternative Religions
IAB23-2	Atheism/Agnosticism
IAB23-3	Buddhism
IAB23-4	Catholicism
IAB23-5	Christianity
IAB23-6	Hinduism
IAB23-7	Islam
IAB23-8	Judaism
IAB23-9	Latter-Day Saints
IAB23-10	Pagan/Wiccan
IAB24	**Uncategorized**
IAB25	**Non-Standard Content**
IAB25-1	Unmoderated UGC
IAB25-2	Extreme Graphic/Explicit Violence
IAB25-3	Pornography
IAB25-4	Profane Content
IAB25-5	Hate Content
IAB25-6	Under Construction
IAB25-7	Incentivized
IAB26	**Illegal Content**
IAB26-1	Illegal Content
IAB26-2	Warez
IAB26-3	Spyware/Malware
IAB26-4	Copyright Infringement

5.2 Banner Ad Types

The following table indicates the types of ads that can be accepted by the exchange unless restricted by publisher site settings.

Value	Description
1	XHTML Text Ad (usually mobile)
2	XHTML Banner Ad. (usually mobile)
3	JavaScript Ad; must be valid XHTML (i.e., Script Tags Included)
4	iframe

5.3 Creative Attributes

The following table specifies a standard list of creative attributes that can describe an ad being served or serve as restrictions of thereof.

Value	Description
1	Audio Ad (Auto-Play)
2	Audio Ad (User Initiated)
3	Expandable (Automatic)
4	Expandable (User Initiated - Click)
5	Expandable (User Initiated - Rollover)
6	In-Banner Video Ad (Auto-Play)
7	In-Banner Video Ad (User Initiated)
8	Pop (e.g., Over, Under, or Upon Exit)
9	Provocative or Suggestive Imagery
10	Shaky, Flashing, Flickering, Extreme Animation, Smileys
11	Surveys
12	Text Only
13	User Interactive (e.g., Embedded Games)
14	Windows Dialog or Alert Style
15	Has Audio On/Off Button
16	Ad Provides Skip Button (e.g. VPAID-rendered skip button on pre-roll video)
17	Adobe Flash

5.4 Ad Position

The following table specifies the position of the ad as a relative measure of visibility or prominence. This OpenRTB table has values derived from the Inventory Quality Guidelines (IQG). Practitioners should

keep in sync with updates to the IQG values as published on IAB.com. Values “4” - “7” apply to apps per the mobile addendum to IQG version 2.1.

Value	Description
0	Unknown
1	Above the Fold
2	DEPRECATED - May or may not be initially visible depending on screen size/resolution.
3	Below the Fold
4	Header
5	Footer
6	Sidebar
7	Full Screen

5.5 Expandable Direction

The following table lists the directions in which an expandable ad may expand, given the positioning of the ad unit on the page and constraints imposed by the content.

Value	Description
1	Left
2	Right
3	Up
4	Down
5	Full Screen

5.6 API Frameworks

The following table is a list of API frameworks supported by the publisher.

Value	Description
1	VPAID 1.0
2	VPAID 2.0
3	MRAID-1
4	ORMMA
5	MRAID-2
6	MRAID-3

5.7 Video Linearity

The following table indicates the options for video linearity. "In-stream" or "linear" video refers to pre-roll, post-roll, or mid-roll video ads where the user is forced to watch ad in order to see the video content. "Overlay" or "non-linear" refer to ads that are shown on top of the video content.

This OpenRTB table has values derived from the Inventory Quality Guidelines (IQG). Practitioners should keep in sync with updates to the IQG values.

Value	Description
1	Linear / In-Stream
2	Non-Linear / Overlay

5.8 Protocols

The following table lists the options for the various bid response protocols that could be supported by an exchange.

Value	Description
1	VAST 1.0
2	VAST 2.0
3	VAST 3.0
4	VAST 1.0 Wrapper
5	VAST 2.0 Wrapper
6	VAST 3.0 Wrapper
7	VAST 4.0
8	VAST 4.0 Wrapper
9	DAAST 1.0
10	DAAST 1.0 Wrapper

5.9 Video Placement Types

The following table lists the various types of video placements derived largely from the IAB Digital Video Guidelines.

Value	Description
1	In-Stream Played before, during or after the streaming video content that the consumer has requested (e.g., Pre-roll, Mid-roll, Post-roll).

2	In-Banner Exists within a web banner that leverages the banner space to deliver a video experience as opposed to another static or rich media format. The format relies on the existence of display ad inventory on the page for its delivery.
3	In-Article Loads and plays dynamically between paragraphs of editorial content; existing as a standalone branded message.
4	In-Feed - Found in content, social, or product feeds.
5	Interstitial/Slider/Floating Covers the entire or a portion of screen area, but is always on screen while displayed (i.e. cannot be scrolled out of view). Note that a full-screen interstitial (e.g., in mobile) can be distinguished from a floating/slider unit by the `imp.instl` field.

5.10 Playback Methods

The following table lists the various playback methods.

Value	Description
1	Initiates on Page Load with Sound On
2	Initiates on Page Load with Sound Off by Default
3	Initiates on Click with Sound On
4	Initiates on Mouse-Over with Sound On
5	Initiates on Entering Viewport with Sound On
6	Initiates on Entering Viewport with Sound Off by Default

5.11 Playback Cessation Modes

The following table lists the various modes for when playback terminates.

Value	Description
1	On Video Completion or when Terminated by User
2	On Leaving Viewport or when Terminated by User
3	On Leaving Viewport Continues as a Floating/Slider Unit until Video Completion or when Terminated by User

5.12 Start Delay

The following table lists the various options for the video or audio start delay. If the start delay value is greater than 0, then the position is mid-roll and the value indicates the start delay.

Value	Description
> 0	Mid-Roll (value indicates start delay in second)
0	Pre-Roll
-1	Generic Mid-Roll
-2	Generic Post-Roll

5.13 Production Quality

The following table lists the options for content quality. These values are defined by the IAB; refer to www.iab.com/wp-content/uploads/2015/03/long-form-video-final.pdf for more information.

Value	Description
0	Unknown
1	Professionally Produced
2	Prosumer
3	User Generated (UGC)

5.14 Companion Types

The following table lists the options to indicate markup types allowed for companion ads that apply to video and audio ads. This table is derived from VAST 2.0+ and DAAST 1.0 specifications. Refer to www.iab.com/guidelines/digital-video-suite for more information.

Value	Description
1	Static Resource
2	HTML Resource
3	iframe Resource

5.15 Content Delivery Methods

The following table lists the various options for the delivery of video or audio content.

Value	Description
1	Streaming
2	Progressive
3	Download

5.16 Feed Types

The following table lists the types of feeds, typically for audio.

Value	Description
1	Music Service
2	FM/AM Broadcast
3	Podcast

5.17 Volume Normalization Modes

The following table lists the types of volume normalization modes, typically for audio.

Value	Description
0	None
1	Ad Volume Average Normalized to Content
2	Ad Volume Peak Normalized to Content
3	Ad Loudness Normalized to Content
4	Custom Volume Normalization

5.18 Content Context

The following table lists the various options for indicating the type of content being used or consumed by the user in which the impression will appear. This OpenRTB table has values derived from the Inventory Quality Guidelines (IQG). Practitioners should keep in sync with updates to the IQG values.

Value	Description
1	Video (i.e., video file or stream such as Internet TV broadcasts)
2	Game (i.e., an interactive software game)
3	Music (i.e., audio file or stream such as Internet radio broadcasts)
4	Application (i.e., an interactive software application)
5	Text (i.e., primarily textual document such as a web page, eBook, or news article)
6	Other (i.e., none of the other categories applies)
7	Unknown

5.19 IQG Media Ratings

The following table lists the media ratings used in describing content based on the IQG 2.1 categorization. Refer to www.iab.com/guidelines/digital-video-suite for more information.

Value	Description
1	All Audiences
2	Everyone Over 12
3	Mature Audiences

5.20 Location Type

The following table lists the options to indicate how the geographic information was determined.

Value	Description
1	GPS/Location Services
2	IP Address
3	User provided (e.g., registration data)

5.21 Device Type

The following table lists the type of device from which the impression originated.

OpenRTB version 2.2 of the specification added distinct values for Mobile and Tablet. It is recommended that any bidder adding support for 2.2 treat a value of 1 as an acceptable alias of 4 & 5.

This OpenRTB table has values derived from the Inventory Quality Guidelines (IQG). Practitioners should keep in sync with updates to the IQG values.

Value	Description	Notes
1	Mobile/Tablet	Version 2.0
2	Personal Computer	Version 2.0
3	Connected TV	Version 2.0
4	Phone	New for Version 2.2
5	Tablet	New for Version 2.2
6	Connected Device	New for Version 2.2
7	Set Top Box	New for Version 2.2

5.22 Connection Type

The following table lists the various options for the type of device connectivity.

Value	Description
0	Unknown
1	Ethernet

2	WIFI
3	Cellular Network – Unknown Generation
4	Cellular Network – 2G
5	Cellular Network – 3G
6	Cellular Network – 4G

5.23 IP Location Services

The following table lists the services and/or vendors used for resolving IP addresses to geolocations.

Value	Description
1	ip2location
2	Neustar (Quova)
3	MaxMind
4	NetAcuity (Digital Element)

5.24 No-Bid Reason Codes

The following table lists the options for a bidder to signal the exchange as to why it did not offer a bid for the impression.

Value	Description
0	Unknown Error
1	Technical Error
2	Invalid Request
3	Known Web Spider
4	Suspected Non-Human Traffic
5	Cloud, Data center, or Proxy IP
6	Unsupported Device
7	Blocked Publisher or Site
8	Unmatched User
9	Daily Reader Cap Met
10	Daily Domain Cap Met

5.25 Loss Reason Codes

The following table lists the options for an exchange to inform a bidder as to the reason why they did not win an impression.

Value	Description
0	Bid Won
1	Internal Error
2	Impression Opportunity Expired
3	Invalid Bid Response
4	Invalid Deal ID
5	Invalid Auction ID
6	Invalid (i.e., malformed) Advertiser Domain
7	Missing Markup
8	Missing Creative ID
9	Missing Bid Price
10	Missing Minimum Creative Approval Data
100	Bid was Below Auction Floor
101	Bid was Below Deal Floor
102	Lost to Higher Bid
103	Lost to a Bid for a PMP Deal
104	Buyer Seat Blocked
200	Creative Filtered - General; reason unknown.
201	Creative Filtered - Pending processing by Exchange (e.g., approval, transcoding, etc.)
202	Creative Filtered - Disapproved by Exchange
203	Creative Filtered - Size Not Allowed
204	Creative Filtered - Incorrect Creative Format
205	Creative Filtered - Advertiser Exclusions
206	Creative Filtered – App Bundle Exclusions
207	Creative Filtered - Not Secure
208	Creative Filtered - Language Exclusions
209	Creative Filtered - Category Exclusions
210	Creative Filtered - Creative Attribute Exclusions
211	Creative Filtered - Ad Type Exclusions
212	Creative Filtered - Animation Too Long
213	Creative Filtered - Not Allowed in PMP Deal
≥ 1000	Exchange specific (should be communicated to bidders *a priori*)

6. Bid Request/Response Samples

6.1 GitHub Repository

The official OpenRTB Github repository now contains a set of validated example requests. This repository should be considered the canonical examples for implementers.

github.com/openrtb/examples

6.2 Validator

An OpenRTB Validator has been developed to test compliance of bid response and bid response JSON payloads. The Validator is available for all final versions of OpenRTB specification. The code for the validator is distributed freely under a BSD-3 open source license at the below URL.

github.com/openrtb/openrtb2x/tree/2.0/openrtb-validator

6.3 Bid Requests

6.3.1 Example 1 – Simple Banner

Following is a basic example of a bid request for a banner ad. Some optional parameters are included in this example.

```
{
    "id": "80ce30c53c16e6ede735f123ef6e32361bfc7b22",
    "at": 1, "cur": [ "USD" ],
    "imp": [
        {
            "id": "1", "bidfloor": 0.03,
            "banner": {
                "h": 250, "w": 300, "pos": 0
            }
        }
    ],
    "site": {
        "id": "102855",
        "cat": [ "IAB3-1" ],
        "domain": "www.foobar.com",
        "page": "http://www.foobar.com/1234.html ",
        "publisher": {
            "id": "8953", "name": "foobar.com",
            "cat": [ "IAB3-1" ],
            "domain": "foobar.com"
        }
    },
    "device": {
```

```
        "ua": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/537.13
(KHTML, like Gecko) Version/5.1.7 Safari/534.57.2",
        "ip": "123.145.167.10"
    },
    "user": {
        "id": "55816b39711f9b5acf3b90e313ed29e51665623f"
    }
}
```

6.3.2 Example 2 – Expandable Creative

This example builds the first and adds parameters to describe support for an expandable creative, and passes data about the user from “Data Provider 1”.

```
{
    "id": "123456789316e6ede735f123ef6e32361bfc7b22",
    "at": 2, "cur": [ "USD" ],
    "imp": [
        {
            "id": "1", "bidfloor": 0.03,
            "iframebuster": [ "vendor1.com", "vendor2.com" ],
            "banner": {
                "h": 250, "w": 300, "pos": 0,
                "battr": [ 13 ],
                "expdir": [ 2, 4 ]
            }
        }
    ],
    "site": {
        "id": "102855",
        "cat": [ "IAB3-1" ],
        "domain": "www.foobar.com",
        "page": "http://www.foobar.com/1234.html",
        "publisher": {
            "id": "8953", "name": "foobar.com",
            "cat": [ "IAB3-1" ],
            "domain": "foobar.com"
        }
    },
    "device": {
        "ua": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/537.13
(KHTML, like Gecko) Version/5.1.7 Safari/534.57.2",
        "ip": "123.145.167.10"
    },
    "user": {
        "id": "55816b39711f9b5acf3b90e313ed29e51665623f",
        "buyeruid": "5456787654678765678987656789876543",
        "data": [
            {
                "id": "6", "name": "Data Provider 1",
                "segment": [
                    {
```

```
                    "id": "12341318394918", "name": "auto intenders"
                },
                {
                    "id": "1234131839491234", "name": "auto enthusiasts"
                },
                {
                    "id": "23423424", "name": "data-provider1-age",
                    "value": "30-40"
                }
            ]
        }
    ]
  }
}
```

6.3.3 Example 3 – Mobile

This example uses a device object to reflect a mobile device, and an app object to reflect a request from a mobile application.

```
{
   "id": "IxexyLDIIk",
   "at": 2,
   "bcat": [ "IAB25", "IAB7-39", "IAB8-18", "IAB8-5", "IAB9-9" ],
   "badv": [ "apple.com", "go-text.me", "heywire.com" ],
   "imp": [
      {
         "id": "1", "bidfloor": 0.5, "instl": 0,
         "tagid": "agltb3B1Yi1pbmNyDQsSBFNpdGUY7fD0FAw",
         "banner": {
            "w": 728, "h": 90, "pos": 1,
            "btype": [ 4 ],
            "battr": [ 14 ],
            "api": [ 3 ]
         }
      }
   ],
   "app": {
      "id": "agltb3B1Yi1pbmNyDAsSA0FwcBiJkfIUDA", "name": "Yahoo Weather",
      "cat": [ "IAB15", "IAB15-10" ],
      "ver": "1.0.2",
      "bundle": "12345",
      "storeurl": "https://itunes.apple.com/id628677149",
      "publisher": {
         "id": "agltb3B1Yi1pbmNyDAsSA0FwcBiJkfTUCV", "name": "yahoo",
         "domain": "www.yahoo.com"
      }
   },
   "device": {
      "dnt": 0,
      "ua": "Mozilla/5.0 (iPhone; CPU iPhone OS 6_1 like Mac OS X)
AppleWebKit/534.46 (KHTML, like Gecko) Version/5.1 Mobile/9A334 Safari/7534.48.3",
```

```
        "ip": "123.145.167.189",
        "ifa": "AA000DFE74168477C70D291f574D344790E0BB11",
        "carrier": "VERIZON",
        "language": "en",
        "make": "Apple", "model": "iPhone",
        "os": "iOS", "osv": "6.1",
        "js": 1,
        "connectiontype": 3,
        "devicetype": 1,
        "geo": {
            "lat": 35.012345, "lon": -115.12345,
            "country": "USA",
            "metro": "803",
            "region": "CA", "city": "Los Angeles", "zip": "90049"
        }
    },
    "user": {
        "id": "ffffffd5135596709273b3a1a07e466ea2bf4fff",
        "yob": 1984, "gender": "M"
    }
}
```

6.3.4 Example 4 – Video

The following example illustrates a bid request for a video impression with two companion ad slots (1 expandable). Additionally, the video content itself is described in the "content" object. A few notes about specific fields in the example:

- `protocols`: Only VAST 2.0 and 3.0 are allowed. Note that a wrapper response is not allowed in this example.
- `sequence`: It is not explicitly included so the default of “1” should be assumed.
- `battr`: User interactive and alert type ads (value “13” and “14”, respectively) are explicitly being blocked for both the video and its companions.
- `pos`: Indicates this opportunity is “above the fold”.
- `api`: Indicates that VPAID 1.0 containers are explicitly supported. As such, the mime types supported for VPAID are only “application/x-shockwave-flash” and “application/javascript”. Note that there is an implicit restriction as to which protocol is allowed in which mime type. JavaScript support was not specified until VPAID 2.0, while Flash supports both VPAID 1.0 and 2.0.
- `companiontype`: Indicates only static or HTML resources are allowed.

```
{
    "id": "1234567893",
    "at": 2, "tmax": 120,
    "imp": [
        {
            "id": "1", "bidfloor": 0.03,
            "video": {
                "w": 640, "h": 480, "pos": 1,
                "startdelay": 0, "minduration": 5, "maxduration": 30,
```

```
            "maxextended": 30,
            "minbitrate": 300, "maxbitrate": 1500,
            "api": [ 1, 2 ],
            "protocols": [ 2, 3 ],
            "mimes": [
               "video/x-flv",
               "video/mp4",
               "application/x-shockwave-flash",
               "application/javascript"
            ],
            "linearity": 1,
            "boxingallowed": 1,
            "playbackmethod": [ 1, 3 ],
            "delivery": [ 2 ],
            "battr": [ 13, 14 ],
            "companionad": [
               {
                  "id": "1234567893-1",
                  "w": 300, "h": 250, "pos": 1,
                  "battr": [ 13, 14 ],
                  "expdir": [ 2, 4 ]
               },
               {
                  "id": "1234567893-2",
                  "w": 728, "h": 90, "pos": 1,
                  "battr": [ 13, 14 ]
               }
            ],
            "companiontype": [ 1, 2 ]
         }
      }
   ],
   "site": {
      "id": "1345135123", "name": "Site ABCD",
      "domain": "siteabcd.com",
      "cat": [ "IAB2-1", "IAB2-2" ],
      "page": "http://siteabcd.com/page.htm",
      "ref": "http://referringsite.com/referringpage.htm",
      "privacypolicy": 1,
      "publisher": {
         "id": "pub12345", "name": "Publisher A"
      },
      "content": {
         "id": "1234567",
         "series": "All About Cars",
         "season": "2", "episode": 23, "title": "Car Show",
         "cat": [ "IAB2-2" ],
         "keywords": "keyword-a,keyword-b,keyword-c"
      }
   },
   "device": {
      "ip": "64.124.253.1",
      "ua": "Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10.6; en-US; rv:1.9.2.16)
Gecko/20110319 Firefox/3.6.16",
```

```
        "os": "OS X",
        "flashver": "10.1", "js": 1
    },
    "user": {
        "id": "456789876567897654678987656789",
        "buyeruid": "54567876546787656789876567898765 4",
        "data": [
            {
                "id": "6", "name": "Data Provider 1",
                "segment": [
                    {
                        "id": "12341318394918", "name": "auto intenders"
                    },
                    {
                        "id": "1234131839491234", "name": "auto enthusiasts"
                    }
                ]
            }
        ]
    }
}
```

6.3.5 Example 5 – PMP with Direct Deal

Following is a basic example of a bid request for a banner ad with a direct deal. Some optional parameters are included in this example.

```
{
    "id": "80ce30c53c16e6ede735f123ef6e32361bfc7b22",
    "at": 1, "cur": [ "USD" ],
    "imp": [
        {
            "id": "1", "bidfloor": 0.03,
            "banner": {
                "h": 250, "w": 300, "pos": 0
            },
            "pmp": {
                "private_auction": 1,
                "deals": [
                    {
                        "id":"AB-Agency1-0001",
                        "at": 1, "bidfloor": 2.5,
                        "wseat": [ "Agency1" ]
                    },
                    {
                        "id":"XY-Agency2-0001",
                        "at": 2, "bidfloor": 2,
                        "wseat": [ "Agency2" ]
                    }
                ]
            }
        }
```

```
    ],
    "site": {
        "id": "102855",
        "domain": "www.foobar.com",
        "cat": [ "IAB3-1" ],
        "page": "http://www.foobar.com/1234.html",
        "publisher": {
            "id": "8953", "name": "foobar.com",
            "cat": [ "IAB3-1" ],
            "domain": "foobar.com"
        }
    },
    "device": {
        "ua": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/537.13
(KHTML, like Gecko) Version/5.1.7 Safari/534.57.2",
        "ip": "123.145.167.10"
    },
    "user": {
        "id": "55816b39711f9b5acf3b90e313ed29e51665623f"
    }
}
```

6.3.6 Example 6 – Native Ad

Following is a basic example of a bid request for a Native ad; similar otherwise to the simple banner example in Section 6.3.1. Notice the `request` attribute in the `Native` object contains an encoded string of a native ad request that conforms to the Dynamic Native Ads API, specifically version 1.0 as indicated by the `ver` attribute.

Notice that the contents of the `request` attribute is a JSON encoded string of the Dynamic Native Ads request including its `native` top level object. This necessitates separate parsing steps for the outer OpenRTB structure and the Dynamic Native Ads request payload in order to enforce a separation of these specifications. The goal of this is to enable independent evolution of these specifications while avoiding the need to implement parsers for each pairwise combination thereof or to write custom JSON parsers.

```
{
    "id": "80ce30c53c16e6ede735f123ef6e32361bfc7b22",
    "at": 1, "cur": [ "USD" ],
    "imp": [
        {
            "id": "1", "bidfloor": 0.03,
            "native": {
                "request": "{\"native\":{\"ver\":\"1.0\",\"assets\":[ ... ]}}",
                "ver": "1.0",
                "api": [ 3 ], "battr": [ 13, 14 ]
            }
        }
    ],
    "site": {
        "id": "102855",
```

```
        "cat": [ "IAB3-1" ],
        "domain": "www.foobar.com",
        "page": "http://www.foobar.com/1234.html ",
        "publisher": {
            "id": "8953", "name": "foobar.com",
            "cat": [ "IAB3-1" ],
            "domain": "foobar.com"
        }
    },
    "device": {
        "ua": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/537.13
(KHTML, like Gecko) Version/5.1.7 Safari/534.57.2",
        "ip": "123.145.167.10"
    },
    "user": {
        "id": "55816b39711f9b5acf3b90e313ed29e51665623f"
    }
}
```

6.4 Bid Responses

6.4.1 Example 1 – Ad Served on Win Notice

Following is an example of a bid response with the ad served on win notice. The bid for this impression is a $9.43 CPM.

```
{
    "id": "1234567890", "bidid": "abc1123", "cur": "USD",
    "seatbid": [
        {
            "seat": "512",
            "bid": [
                {
                    "id": "1", "impid": "102", "price": 9.43,
                    "nurl": "http://adserver.com/winnotice?impid=102",
                    "iurl": "http://adserver.com/pathtosampleimage",
                    "adomain": [ "advertiserdomain.com" ],
                    "cid": "campaign111",
                    "crid": "creative112",
                    "attr": [ 1, 2, 3, 4, 5, 6, 7, 12 ]
                }
            ]
        }
    ]
}
```

6.4.2 Example 2 – VAST XML Document Returned Inline

Following is an example of a bid response that returns the VAST document inline to be served. A few notes about specific fields in the example:

- The bid for this impression is a $3.00 CPM.
- Note that since there both a win notice URL and an inline VAST document in the `adm` attribute, which constitutes the ad markup. The win notice is still called, but if it were to return markup it would be ignored in favor of the contents of the `adm` attribute.

```
{
   "id": "123",
   "seatbid": [
      {
         "bid": [
            {
               "id": "12345", "impid": "2", "price": 3.00,
               "nurl": "http://example.com/winnoticeurl",
               "adm": "<?xml version=\"1.0\" encoding=\"utf-8\"?>\n<VAST
version=\"2.0\">\n<Ad id=\"12345\">\n<InLine>\n<AdSystem
version=\"1.0\">SpotXchange</AdSystem>\n<AdTitle>\n <![CDATA[Sample
VAST]]>\n</AdTitle>\n<Impression>http://sample.com</Impression>\n<Description>\n<![C
DATA[A sample VAST feed]]>\n</Description>\n <Creatives>\n<Creative sequence=\"1\"
id=\"1\">\n<Linear>\n<Duration>00:00:30</Duration>\n  <TrackingEvents>
</TrackingEvents>\n<VideoClicks>\n<ClickThrough>\n<![CDATA[http://sample.com/openrtb
test]]>\n</ClickThrough>\n</VideoClicks>\n<MediaFiles>\n<MediaFile
delivery=\"progressive\" bitrate=\"256\" width=\"640\" height=\"480\"
type=\"video/mp4\">\n<![CDATA[http://sample.com/video.mp4]]>\n
</MediaFile>\n</MediaFiles>\n</Linear>\n
</Creative>\n</Creatives>\n</InLine>\n</Ad>\n</VAST>"
            }
         ]
      }
   ]
}
```

6.4.3 Example 3 – Direct Deal Ad Served on Win Notice

Following is an example of a bid response with the ad served on win notice. The bid for this impression is a $5.00 CPM against a direct deal.

```
{
   "id": "1234567890", "bidid": "abc1123", "cur": "USD",
   "seatbid": [
      {
         "seat": "512",
         "bid": [
            {
               "id": "1", "impid": "102", "price": 5.00,
               "dealid": "ABC-1234-6789",
               "nurl": "http: //adserver.com/winnotice?impid=102",
               "adomain": [ "advertiserdomain.com" ],
               "iurl": "http: //adserver.com/pathtosampleimage",
               "cid": "campaign111",
               "crid": "creative112",
```

```
                "adid": "314",
                "attr": [ 1, 2, 3, 4 ]
            }
        ]
    }
  ]
}
```

6.4.4 Example 4 – Native Markup Returned Inline

Following is an example of a bid response that returns a native ad inline to be served. The `adm` attribute contains an encoded string of a native ad request that conforms to the Dynamic Native Ads API and specifically the same version as that used for the request string. Alternatively, the `adm` attribute could have been omitted in favor of returning the native ad markup in the response to the win notice `nurl`.

```
{
   "id": "123",
   "seatbid": [
      {
         "bid": [
            {
               "id": "12345", "impid": "2", "price": 3.00,
               "nurl": "http://example.com/winnoticeurl",
               "adm": "{\"native\":{\"ver\":\"1.0\",\"link\":{ ... },
\"imptrackers\":[ ... ],\"assets\":[ ... ]}}"
            }
         ]
      }
   ]
}
```

7. Implementation Notes

The following section will provide brief notes on how certain objects and fields are to be interpreted and implemented.

7.1 No-Bid Signaling

This section covers best practices for using the optional no-bid signaling. See the List 5.24 for the enumerated list of no-bid reason codes.

Many exchanges support multiple response types as a no-bid:

- HTTP 204 "No Content" from the bidder *(most economical in terms of bandwidth)*.
- An empty JSON object:
  ```
  {}
  ```
- A well-formed no bid response:
  ```
  {"id": "1234567890", "seatbid": []}
  ```
- A well-formed no bid response with a reason code:
  ```
  {"id": "1234567890", "seatbid": [], "nbr": 2}
  ```

An important issue in RTB is when impressions are triggered by software robots mimicking web browsers. Such robots may be implicitly or explicitly driving these false transactions. The following represents a set of symmetric best practices for exchanges and bidders to help recognize and reject these events.

Responsibility of the exchange

Make best effort to classify and reject "non-human traffic" requests for ads to the exchange via the following best practices:

- (Recommended) Filter impressions from known spiders via user-agent classification.
- (Recommended) Filter impressions from suspected NHT via a "detector".

Responsibility of the bidder

- (Recommended) no-bid impressions from known spiders via user-agent classification.
- (Recommended) no-bid impressions from suspected NHT via a "detector".
- Specify a no-bid reason code in either case.

Where:

- For exchanges, filtering the impression means that the exchange should respond to the "ad call" with either a blank HTTP 204 response or an unpaid ad (PSA) and not offered to any bidders.
- For bidders, filtering the impression means that the bidder should respond with a no-bid.
- For both exchanges and bidders, the impression transaction records should be clearly marked in any logging systems and be removed from contributing to any event counts associated with planning, forecasting, and reporting systems.

7.2 Impression Expiration

Recapping the typical impression flow through RTB, an ad will be requested by a client (e.g., web browser, mobile app or an SDK therein) possibly through other server intermediaries, and ultimately to the RTB exchange. The exchange conducts an auction among buyers who bid with a proposed price, possibly markup for use if the bid wins (markup can also be delivered on the win notice itself), and other metadata about the bid. The exchange then selects a winner, issues a win notice to the winning bidder, and passes the markup back to the client.

Winning the auction, however, does not guarantee that the ad will be successfully delivered to the client or that it will meet viewability expectations. Furthermore, policies vary among exchanges as to the criteria for billing. Most consider an ad billable upon some form of delivery or rendering vs. the auction win alone. This aligns better with the buyer's obvious goal of ensuring that the impressions they pay for are actually displayed.

Some exchanges attempt to facilitate this alignment by placing the win notice in the winning ad markup so that it can serve as both a win notice and rendering notice. This is neither endorsed nor prohibited by OpenRTB except that it precludes the exchange from accepting markup on the win notice return as described in Section 4.3.1. Similarly, many buyers use their own tracking URL placed within their ad markup to signal rendering independent of the OpenRTB auction win notice. In video specifically, VAST supports an impression tracking URL that is often used for billing and is always distinct from the auction win notice.

To abstract the concept, let us refer to "*billing notice*" as the firing of some notification URL at the time when the clearing price of the impression will be booked as spend. This is irrespective of whether the actual OpenRTB win notice URL is delegated to the client for firing or some other tracking URL is used.

For buyers, this billing notice is used to book progress toward spend goals and frequency caps and drive pacing algorithms. When the billing notice is delayed significantly, these critical functions can be seriously impaired. There are legitimate reasons for some delays such as caching. A common scenario is a video interstitial impression in a mobile app. Refining the example, consider a game where the video is prefetched during game play so that it can be shown after the current game level ends. This is important for the user experience, but can delay the rendering of the ad for many minutes.

Bidders are strongly advised to track the time between the auction and the win and/or billing notices to ensure reasonable delays. If unreasonable delays are encountered frequently, bidders may elect to ignore such events and bring them to the attention of the exchange for resolution. Unfortunately, the sequence from ad request through the auction and finally to rendering and billing is fundamentally not transactional. There are simply too many parties, policies, and technologies involved and thus a good support relationship between exchange and buyer is still important.

The OpenRTB protocol does provide some real-time assistance, however. The `imp.exp` attribute (Section 3.2.4) in the bid request allows an exchange to provide guidance to bidders of the number of seconds that may elapse between the auction and the billing event. As usual, omitted means unknown. Bidders can then decide if they want to bid understanding the likely delay. Bidders are advised, however, to interpret this as guidance as opposed to a contract unless the exchange expresses otherwise since exchanges are not always in a position to make hard guarantees (e.g., the SDK within the client app may not be under the exchange's control).

Similarly, the `bid.exp` attribute (Section 4.2.3) in the bid response allows the bidder to express the maximum number of seconds they are willing to tolerate between auction and billing notice. This allows

the exchange to drop bids with expiration constraints it believes are likely to be violated. Bidders should not assume that a delayed billing notice greater than their specified bid expirations will not be billable. That is a policy and contract discussion between bidder and exchange and not imposed by OpenRTB.

The following expiration times are offered as examples of reasonable delays based on the nature of the impression. These are only provided as rules of thumb. A more data-driven method of determining these times in specific situations is highly recommended.

- Desktop and mobile web browsers: 1 Minute
- Mobile app banner ads that may be cached: 5 Minutes
- Mobile app native ads that may be cached: 10 Minutes
- Mobile and video interstitials: 30 Minutes *(or even longer)*
- Audio or video with server-side stitching: Very Long or Unknown

7.3 PMP & Direct Deals

Best Practice Bidding Logic

```
Receive request and parse;
Create empty bid list for response;

If request contains the impression[].pmp object;
   match bids against each pmp.deals[];
   enforce targeting for dealID and seatID;
   append best M matching bids to response;

If pmp.private_auction = False;
   match open auction bids against the request;
   append top N bids by price to response;

Return response list to exchange;
```

Recommendations

- M >= 1, preferably one per matching Deal ID.
- N >= 2 to assist with blocking rate issues.
- Minimum viable is “1+1” bidding.
- Ideal is “M+N” bidding.

Warning

Returning only one bid when both Deal ID and open auction bids are valid creates problems. The exchange side may be configured by a publisher to prioritize all Deal ID bids above open auction bids, or to force a price auction between them with different floors by class of bid. There are multiple common practices that depend on how the publisher prefers to sell inventory with Deal ID.

Policy Recommendations

- A Deal ID should be utilized for any situation where the auction may be awarded to a bid not on the basis of price alone. Any prioritization of bids other than by price should have a Deal ID.
- A Deal ID is recommended for all situations where a preferential floor may be assigned to a seat entity.

Anti-Patterns

The below is a set of anti-patterns that OpenRTB supporting platforms have observed in various attempts to implement Deal ID bidding logic.

Subjecting Deal ID Bids to an internal auction on price

The ideal bidding logic describes a process of being liberal about sending bids. Deal ID bids may not be subject to a classic price auction. There may be an expectation that the buyer and seller want prioritization to achieve a larger objective: complete delivery of the Deal represented by the Deal ID. Thus any bidding logic that sorts Deal ID bids by price (with or without open marketplace bids) and truncates the list too aggressively can endanger the fulfillment of the Deal.

Associating Deal ID to the wrong Object

A Deal ID should be treated as a "targeting token" associated to orders, line-items or campaigns. If the Deal ID is associated to a Seat/Buyer it may create an undesired application of the Deal ID too many active campaigns. Alternatively if it is associated to the Advertiser it may limit that entity to only a single Deal ID.

Improper Handling of the Private vs Open Market Flag

The `pmp.private_auction` flag indicates that the seller is willing or not willing to accept open market bids (i.e., "all bidders are welcome"). If this flag is not read and interpreted correctly, bid responses may be invalid. Open market bids sent to a private impression auction may be rejected and should not have been exposed to all bidders.

Improper handling of Seat IDs

If Seat IDs are treated as a filter of eligible demand partners on an open market impression, this defeats the "all bidders are welcome" intention.

Silently Applying Margin Discounts to Deal ID Bids

With Deal ID buyers are sellers are communicating directly. The Exchange and Bidder become third-party automation platforms. If there are any automatic or silent discounts of bid prices (based upon margins or fees) set by either the exchange or the bidder, then the Deal may fail to function correctly.

Use cases

Case-1: Open Trading Agreement with Buyer

- Between publisher and buying entity.
- Publisher sets an access rule defining the price floor for a specific buyer.
- Locked to the buyer.
- Broadcast price floor.
- Public/open inventory.
- No Deal ID needed (Deal ID is optional).
- No named advertiser(s).
- No prioritization of bids.
- Daily total or frequency caps optional on publisher/exchange side.
- All placements or limited to specific placements.
- Targeting is up to the buyer/bidder.

Case-2: Open Trading Agreement with Buyer with Named Advertisers

- As Case-1 with a list of named advertisers.

Case-3: Open Bidding with Deal ID as Value-added Markers

- Between publisher and buying entity.
- Publisher sets a price floor for URL masked inventory.
- Public/open inventory (i.e., all buyers welcome).
- Deal ID represents “Package Tokens”.
- Each Deal ID signals that the impression falls into various content and placement categories.
- Floor is associated to each Deal ID to signal cost for usage of that token.
- Winner is decided by bid price.
- Execution of targeting is up to the buyer/bidder.

Case-4: First Look Trading Agreement

- Between publisher and buying entity.
- Publisher sets an access rule defining the price floor for the buyer.
- Locked to the buyer.
- Known price floor.
- Deal ID needed.
- Optional named advertiser list.
- Prioritization of bids expected.
- Daily total or frequency caps optional on publisher/exchange side.
- All placements or limited to specific placements.
- Targeting is up to the buyer/bidder.

Case-5: Direct Option Deal with Advertiser via RTB

- Between Publisher and Advertiser or their representative.
- Publisher sets a rule defining a price floor and prioritization for specific advertiser(s).
- Fill rate is expected to be greater than or equal to X%.
- Locked to the buyer.
- Private/exclusive inventory.

- Limited to a set list of advertiser names (generally variants of one name).
- Known price floor.
- Deal ID needed.
- Prioritization of bids expected.
- Daily total or frequency caps will apply on bidder side; optional on Exchange side.
- Limited to specific placements.
- Targeting is mostly enforced by buyer/bidder.

Case-6: Direct Option Deal with Advertiser via RTB with Private Data

- Same as Case-4.
- Deal ID represents some combination of private first-party data from the Publisher.

Case-7: Full-Fill Direct Deal with Advertiser via RTB

- Same as Case-4.
- Fill rate is expected to be 100% or nearly so.

Case-8: Full-Fill Direct Deal with Advertiser via RTB with Private Data

- Same as Case-6.
- Deal ID represents some combination of private first-party data from the Publisher.

7.4 Skippability

This section clarifies the common use cases related to declaring skippability of video creatives.

Under most circumstances for RTB transactions, publishers and exchanges prefer to control the ability to skip the ad. OpenRTB therefore assumes by default that a standard linear video ad can be used as the response to a skippable request and the ability to skip the ad will be provided by the supplier's player automatically.

The presence of the `video.skip` attribute in the bid request with a value of "1" should be assumed to mean that the publisher will impose a skip button on the ad. The absence of the `video.skip` attribute should be assumed to mean that it is unknown whether the publisher will impose a skip button.

DSPs should confirm with publishers whether it is permissible to respond with ads that provide their own skip functionality (e.g., using VPAID to render a skip button). If bidding with such an ad and only if doing so, the bid must indicate creative attribute "16" using the `attr` array in the bid response.

Note: VAST 4.0 separates VPAID interactivity from the media file so this is deprecated and only applies to earlier versions of VAST.

Some examples of these concepts follow:

Bid Request

Case-1: Skippable after N Seconds for All Creatives

In this case, the publisher will impose skippability. All ads will be skippable, but only after 5 seconds of play. Creatives with a total duration of 5 seconds or less would not be skippable since they would never reach this threshold.

```
"video": {
   ..., "skip": 1, "skipafter": 5, ...
}
```

Case-2: Skippable after N Seconds for Minimum Duration Creatives

In this case, the publisher will impose skippability. However, only creatives with a total duration greater than 15 seconds will be skippable. For ads that satisfy this minimum total duration, skippability is enabled after 5 seconds of play. Note that although these values are integers, they will compare as precise values with actual video durations. For example, a video with duration 15.1 seconds does satisfy a `skipmin` value of 15 (i.e., think of the `skipmin` value as being 15.0).

```
"video": {
   ..., "skip": 1, "skipmin": 15, "skipafter": 5, ...
}
```

Case-3: Non-Skippable unless Requested by the Ad Markup

In this case, the publisher will not impose skippability. Ads will only be skippable if requested by the ad markup. This is supported by VPAID and VAST 3.0, for example.

```
"video": {
   ..., "skip": 0, ...
}
```

Case-4: Unknown Skippability

In this case, the `skip` attribute is omitted which indicates that exchange does not know if skippability will be imposed by the publisher. This may be the case, for example, when the exchange is not an SSP and thus may not have control or full knowledge of the publisher's intentions.

Bid Response

Consider Case-3 above, where the publisher does not impose skippability. If the ad markup itself will request skippability (e.g., via VPAID or VAST 3.0), then the bid must signal this intention. This is accomplished by including creative attribute 16 (i.e., Skippable) in the bid as shown below. If the markup is not going to request skippability, then this creative attribute should not be indicated.

When responding to Case-3 with this skippable attribute specified in the bid, the publisher should provide skippability either by instructing the VAST 3.0 player to activate skippability (refer to the VAST 3.0 "skipoffset" attribute) or by allowing the ad to render its own skip button using VPAID.

```
"bid": {
   ..., "attr": [16], ...
}
```

In Case-1 and Case-2 where the publisher may impose its own skippability, creative attribute 16 should not be specified. Furthermore, publishers are advised to filter responses containing attribute 16 since this could conflict with the skip button rendered by the publisher. When using a VAST 3.0 response, publishers may choose to implement support for VAST 3.0 "skipoffset" at their discretion and ads should be assumed to play non-skippable if the player does not support it.

7.5 COPPA Regulation Flag

The United States Federal Trade Commission has changed the compliance rules for the Children's Online Privacy Protection Act ("COPPA"), effective July 1, 2013. The proposal effects websites, and associated services), that have been identified as: (1) directed to users under 13 years of age; or (2) collecting information from users actually known to be under 13 (collectively "Children's Sites").

The FTC has written a comprehensive FAQ on the change here:

www.ftc.gov/tips-advice/business-center/guidance/complying-coppa-frequently-asked-questions

Steve Bellovin, CTO of the FTC, argued for a standardized signaling protocol in a blog posted dated January 2013:

www.ftc.gov/news-events/blogs/techftc/2013/01/coppa-signaling

Impacts

The FAQ specifically calls out these areas relevant for OpenRTB as "Personal Information" that is not to be collected:

- Geo-location information sufficient to identify street name and name of a city or town.
- Persistent identifiers when they can be used to recognize a user over time and across different Web sites or online services.

Recommendations to Implementers

OpenRTB Exchanges and Bidders should:

- Provide a facility for sites to be declared as "child directed".
- Implement the regulations object extension.
- Provide facilities within campaigns to target for and against this signal.
- Degrade the Geographic information to be less exact prior to logging or transmission.
- Suppress the assignment and synchronization of identifiers, depending on usage.

It is recommended that when `regs.coppa` = 1, the exchange should additionally manipulate the OpenRTB bid request object as follows:

Device Object

- Suppress `didmd5` and `didsha1` device ID fields.
- Truncate `ip` field - remove lowest 8 bits.
- Truncate `ipv6` field - remove lowest 32 bits.

Geo Object

- Suppress `lat` and `lon` fields.
- Suppress `metro`, `city`, and `zip` fields.

User Object

- Suppress `id`, `buyeruid`, `yob`, and `gender` fields.

Appendix A. Additional Information

- Creative Commons / Attribution License
  creativecommons.org/licenses/by/3.0
- IAB (Interactive Advertising Bureau)
  www.iab.com
- IAB Quality Assurance Guidelines (QAG):
  www.iab.com/guidelines/iab-quality-assurance-guidelines-qag-taxonomy/
- JavaScript Object Notation (JSON)
  www.json.org
- MMA (Mobile Marketing Association)
  mmaglobal.com
- OpenRTB Project on Github
  github.com/openrtb/OpenRTB/
- Apache Avro
  avro.apache.org
- Protocol Buffers (Protobuf)
  code.google.com/p/protobuf
- Google Metro Codes
  code.google.com/apis/adwords/docs/appendix/metrocodes.html
- U.N. Code for Trade and Transport Locations:
  www.unece.org/cefact/locode/service/location.htm

Appendix B. Specification Change Log

This appendix serves as an index of specification changes from v2.4 to v2.5. These changes pertain only to the substance of the specification and not routine document formatting, organization, or content without technical impact.

Section	Description
2.4	*Section: Data Encoding* New section added.
3.1	*Object Model: Bid Request* Updated to include `Source` and `Metric` objects.
3.2.1	*Object: BidRequest* Attributes `bseat`, `wlang`, and `source` have been added.
3.2.2	*Object: Source* New `Source` object has been added including the Payment ID `pchain` attribute.
3.2.4	*Object: Imp* Attribute `metric` has been added.
3.2.5	*Object: Metric* New `Metric` object has been added.
3.2.6	*Object: Banner* Attribute `vcm` has been added.
3.2.7	*Object: Video* Attributes `placement` and `playbackend` have been added. Guidance added to use only the first element of attribute **`playbackmethod`** in preparation for future conversion to an integer.
3.2.10	*Object: Format* Attributes `wratio`, `hratio`, and `wmin` have been added.
3.2.13	*Object: Device* Attribute `mccmnc` has been added. Attribute `carrier` has been clarified to eliminate a reference to using "WIFI" as a carrier.
4.2.3	*Object: Bid* Attributes `burl`, `lurl`, `tactic`, `language`, `wratio`, and `hratio` have been added.
4.4	*Substitution Macros:* Macros ${AUCTION_MBR} and ${AUCTION_LOSS} have been added. A best practice has been added to use "AUDIT" for unknown values when rendering for test or quality purposes.
5.6	*List: API Frameworks* Item 6 has been added.
5.9	*List: Video Placement Types* New list has been added.
5.10	*List: Playback Methods* Items 5-6 have been added.

Section	Description
5.11	*List: Playback Cessation Modes* New list has been added.
5.24	*List: No-Bid Reason Codes* Items 9-10 have been added.
5.25	*List: Loss Reason Codes* New list has been added.