# PubMatic Integrations

Use this guide to understand the open-source integrations supported by PubMatic.

## Why use OpenWrap?

Publishers typically set lower priorities for programmatic demand in their ad servers, thus limiting it to remnant impressions via trafficked tags. The programmatic marketplace can only propose bids for that limited volume of inventory it's exposed to. As a result, publishers lose visibility and monetization of high-value programmatic demand on a large share of their traffic.

OpenWrap provides a transparent enterprise wrapper solution for Prebid and lets publishers evaluate programmatic bids from multiple sources concurrently for every impression. OpenWrap also lets ad servers begin utilizing real-time bids returned instead of the static bids that are used in case of yield tags.

### Key features

- **Broad access to demand:** SPO deals with top demand partners and integrations with hundreds of DSPs
- **Multiple ad formats supported:** Accept demand in the desired and emerging formats including video, native, and CTV
- **Prebid open source:** Leverage a community of developers and maintain transparency and trust
- **Product maturity:** Proven success with 85+ global clients
- **Innovation:** 3+ years of development to date, we're building for the future

### How does OpenWrap work?

OpenWrap provides the pricing from programmatic open marketplaces from multiple SSP partners, which can be consumed by passing key-value pairs to ad servers in real-time.

An easy-to-use UI provides the option of controlling partners and inventory mapping allowing for quick application and testing.

#### Ad call flow:

1. PubMatic's wrapper tag JavaScript code is placed on the publisher's site, which makes an ad call to all the configured partners before calling the ad server.
2. OpenWrap runs an auction across all the bids from all partners.
3. The winning bid is sent to the publisher's ad server by dynamically adding this as a key value to the ad server call. PubMatic Wrapper line items are created in the publisher's ad server at various priorities targeting the bid signal in the ad server call. The relevant line item is activated.
4. Publisher's ad server makes the final decision on which campaign/line item to serve considering Wrapper bid as an additional parameter, and the ad is served.

**Step-by-step integration process:** OpenWrap Integration Guide

## OpenWrap SDK

### Why use OpenWrap SDK?

The OpenWrap SDK lets you leverage PubMatic's robust platform of OpenWrap capabilities within your mobile app. Publishers who want to increase monetization and access the broadest set of exchanges and unique demand, use OpenWrap's transparent enterprise wrapper solution for Prebid. OpenWrap simplifies partner management with an intuitive UI. Enterprise-grade analytics and a dedicated support team help you optimize your inventory's monetization. OpenWrap SDK takes the power of OpenWrap mobile.

### Key features

- **Scaled demand in one SDK:** integrate with 200+ demand partners, our direct relationships with buyers give you access to premium brand spend rather than CPI campaigns
- **Increased revenue through parallel auctions:** unified auctions with real-time prices offers efficiency , transparency and increased revenue
- **Simplified demand partner management:** leverage enterprise-grade analytics to optimize yield, and manage demand partners in a cloud-based UI that requires no changes to your SDK, app, or app store approval
- **Lightweight:** at under 500KB , our SDK is smaller than other solutions and you only need one for multiple partners, parallel auctions in sub-200 m s reducing latency over traditional waterfall and SDK mediation solutions
- **Flexible deployment model:** supports mainstream ad server, custom ad servers, and no ad server

**Step-by-step integration process:** OpenWrap Mobile In-App Support

# Prebid.js

## Why use Prebid.js

Prebid.js is a feature-rich header bidding platform for the web, including more than 200 demand sources and 15 analytics adapters. It supports currency conversion, GDPR, TCF2 (from version 3.12), common ID systems, and multiple ad servers.

## How does Prebid.js work?

At a high level, header bidding with Prebid.js involves just a few steps:

1. The ad server's tag on page is paused, bound by a timer, while the Prebid.js library fetches bids and creatives from various SSPs & exchanges you want to work with.
2. Prebid.js passes information about those bids (including price) to the ad server's tag on page, which passes it to the ad server as query string parameters.
3. The ad server has line items targeting those bid parameters.
4. If the ad server decides Prebid wins, the ad server returns a signal to Prebid.js telling the library to write the winning creative to the page. All finished!

The Prebid.js library is composed of these components:

- the core wrapper code
- the adapters a publisher wants to work with
- optional modules the publisher wants to utilize

## Prebid.js adapters

The Prebid.js Adapters plug into Prebid.js Core and are meant to be interchangeable depending on who the publisher wants to work with. There are two types of adapters: bidder and analytics.

Bidder Adapters represent the SSPs & Exchanges you want to work with. There are currently over 200 bidder adapters. This set of working header bidding integrations is part of what makes Prebid.js so special. Each company maintains its own Prebid.js adapter to provide the freshest code for publishers, rather than a proprietary wrapper solution trying to reverse engineer another company's adapter. It's a win-win for everyone.

PubMatic has a Prebid.js adaptor.

**Step-by-step adaptor addition process:** How to Add a New Bidder Adapter

# Prebid Mobile

## Why use Prebid Mobile?

Prebid Mobile is an open-source library that provides an end-to-end header bidding solution for mobile app publishers. Use this library with your ad server's Mobile SDK to communicate with Prebid Server to request and receive bids over RTB. These bids can then compete directly with bids from your primary ad server.

## How does Prebid Mobile work?

1. Prebid Mobile sends a request to Prebid Server. This request consists of the Prebid Server account ID and config ID for each tag included in the request.
2. Prebid Server constructs an OpenRTB bid request and passes it to the demand partners.
3. Each demand partner returns a bid response to Prebid Server. The bid response includes the bid price and the creative content.
4. Prebid Server sends the bid responses to Prebid Mobile.
5. Prebid Mobile sets key/value targeting for each ad slot through the primary ad server mobile SDK. This targeting will activate one or more of Prebid line items that were previously configured in the primary ad server.
6. If the line item associated with the Prebid Mobile bid wins, the primary ad server returns the Prebid Mobile creative JavaScript to the ad server's SDK.
7. The Prebid Mobile creative JavaScript will fetch and render the corresponding creative content.

Step-by-step Prebid Mobile setup process: Getting Started with PreBid Mobile

## Tag integration

Step-by-step tag implementation process: Video-specific parameters

## Server-to-Server (S2S) integration

Step-by-step S2S implementation process: OpenRTB 2.3