Ad types

- DOOH: Digital Out Of Home
- Display ads

DOOH: Digital Out Of Home

Architecture Overview

Adhese supports DOOH through dedicated endpoints, which allow:

- Media asset distribution
- Playlist generation for screens/players
- Proof-of-play tracking
- Player-level targeting and reporting

A typical DOOH setup contains the following components:

- Screens
- Digital Signage CMS / Player
- Adhese Ad Server
- Server-side data mapping
- Reporting and Proof-of-Play Tracking

Implementation prerequisites

Before starting, ensure that the following are available:

Adhese Requirements

- Active Adhese account
- DOOH feature enabled by Adhese support
- Configured Inventory & templates
- Configured player ID target

Player / CMS Requirements

- Ability to perform HTTP GET requests
- JSON parsing support
- Local asset caching support
- Video/image playback support
- Ability to fire tracking URLs

Adhese DOOH endpoints

Assets Download Endpoint (Heads-up)

> Ensure that the *Use in headsup* setting is activated for all your DOOH formats. You can perform this action in the admin UI.
>
> 
>

This is an optional endpoint used to retrieve and cache all active creative assets for a DOOH position. The response structure is fixed and cannot be customised.

Request example

```
https://headsup-[customer].adhese.org/api/headsup/download-list/sl[positioncode]
```

Response example

```
{
  "media": [
    {
      "ad": {
        "id": "https://pool-demo.adhese.com/pool/lib/ad1.mp4",
        "mime": "video/mp4",
        "curl": "https://pool-demo.adhese.com/pool/lib/ad1.mp4",
        "checksum": "abc123"
      }
    }
  ]
}
```

> Detailed information on the headsup endpoint can be found here

Playlist Endpoint

Adhese provides two stack endpoints for retrieving a playlist.

1. The /**m/stack/** endpoint, which retrieves a list with a limit on the number of ads in the response

2. The **/e/stack/** endpoint, which doesn't have a predefined limit

Request example

```
/m/stack/: https://ads-[customer].adhese.com/m/stack/sl[positioncode]/pi[player
ID]?max_ads=[amount]
/e/stack/: https://ads-[customer].adhese.com/e/stack/sl[positioncode]/pi[player ID]
```

Example response


```
{
    "ads":[
        {
            "id":"https://pool-demo.adhese.com/pool/lib/562_2nd_1.mp4",
            "dur":28.07,
            "proofOfPlay":"https://ads-
demo.adhese.com/track/3474/sl357/tlnone/piplayer_id/A2?1746011926824",
            "error":"https://ads-demo.adhese.com/track/3474-
PLAY_ERROR_[ERRORCODE]/sl357/tlnone/piplayer_id/A2/?1746011926824"
        },
        {
            "id":"https://pool-demo.adhese.com/pool/lib/561_2nd_1.mp4",
            "dur":11.45,
            "proofOfPlay":"https://ads-
demo.adhese.com/track/3444/sl357/tlnone/piplayer_id/A2?1746011926824",
            "error":"https://ads-demo.adhese.com/track/3444-
PLAY_ERROR_[ERRORCODE]/sl357/tlnone/piplayer_id/A2/?1746011926824"
        },
        {
            "id":"https://pool-demo.adhese.com/pool/lib/560_2nd_1.mp4",
            "dur":18.85,
            "proofOfPlay":"https://ads-
demo.adhese.com/track/3455/sl357/tlnone/piplayer_id/A2?1746011926824",
            "error":"https://ads-demo.adhese.com/track/3455-
PLAY_ERROR_[ERRORCODE]/sl357/tlnone/piplayer_id/A2/?1746011926824"
        }
    ]
}
```


Common Fields

Field	Description
id	Creative URL

dur	Duration in milliseconds
proofOfPlay	Tracking URL after playback
error	Tracking URL for playback errors

> The response structure consists of an array of *ads*, each containing several objects. The object structure is linked to the Advar template that was used to create the campaigns. This template is maintained either in Adhese itself or in the configured GitHub repository for your account.

> Detailed information on the stack endpoint can be found here

Player ID target

You can provide extra target information through the request by adding extra target sections to the URL: **/[prefix][value]/.** One example of such a target is the player ID, which has the prefix 'pi', as shown in the above example. This ID can be used for:

- Player/screen level targeting and reporting
- Location-based targeting and reporting

Providing extra target / reporting data

Player / store location and other custom metadata can be provided server-side by uploading a **player => metadata mapping** in Adhese.

Each metadata property will be linked to its own target prefix. This data will be mapped automatically to incoming requests based on the player ID and will become available for targeting and reporting purposes.

> Adhese requires the player ID prefix - or any other target prefixes - to be configured by support.

Operational Recommendations

Asset Caching

Always cache assets locally to:

- Reduce bandwidth
- Avoid playback interruptions
- Improve offline resilience

Offline Mode

Players should continue playback using:

- Last valid playlist
- Cached creatives

Troubleshooting

Issue	**Possible Cause**
Empty media array	No active campaigns or targeting-related issues
Endpoint redirects	DOOH endpoints not enabled
Headsup endpoint empty	No active campaigns or *Use in heads-up* format config disabled
Playback errors	Unsupported codec or corrupt file
No reporting	Proof-of-play URL not triggered

Display ads

Introduction

Display advertising with Adhese can be delivered through several integration routes, ranging from a drop-in web SDK to a fully server-side API. All routes ultimately talk to the same adserver engine, so the choice of integration method is about *how your platform requests and renders ads*, not about what campaigns can run.

What do we mean by display ads?

Display advertising covers visual ad formats rendered in a reserved container. Adhese delivers all common display creative types through every integration method described in this document:

- **Image banners** - a hosted image with a click-through link; the simplest and most common display format.
- **HTML5 banners** - rich, animated or interactive creatives built as HTML5 bundles and delivered as an HTML fragment.
- **Third-party tags** - ad markup hosted by an external ad server or demand partner. Adhese returns the tag; the creative itself is loaded from the third party at render time.

The creative type is a trafficking choice, not an integration choice: whichever type a campaign uses, your integration receives the same kind of response and renders it into the same slot container. Building the integration once makes all three creative types available.

Where can display ads run?

Anywhere the platform can request ads and render the returned markup or image:

- **Websites** - desktop and mobile web, whether ads are requested in the browser (SDK, tags, Prebid) or server-side by your CMS or backend before the page is delivered.
- **Apps** - native Android and iOS applications, using the mobile SDKs or the Request API directly, with creatives rendered in a webview or native component.

Integration methods

Method	Best for

TypeScript SDK	Modern websites and web apps
Prebid.js / Prebid Server	Publishers running header bidding
Request API	CMS, apps, custom stacks
Native mobile SDKs	Android / iOS apps