adform

PRODUCTION GUIDE

Adform – Outstream Video Inview

MAKING DISPLAY ADVERTISING SIMPLE, RELEVANT, REWARDING

Outstream Video Inview

How it helps Publishers/Sell-side?

- Solves the lack of In-Stream inventory problem.
- Highly engaging In-View format.
- Renders inside publishers' content.
- Same tag can be used on multiple placements.

Usually, video inventory has a high price and high engagement, but is difficult to sell at scale. The demand for buying In-Stream pre-roll inventory is usually much higher than the supply, so publishers seek for other means to compensate that. It is popular to sell Rich Media formats that can play video ads, but this requires producing video banners on each instance of sales and the buy side becomes more complicated.

Selling video inventory at scale requires a format that is both easy to run from the buying side and has attractive engagement metrics and viewability from the selling side.

How it helps Agencies/Buy-side?

- Set as In-Stream Ad.
- Ad does not require anything apart from a video file.
- Same Video Ad can be run across all inventory placements, both pre-roll/in-stream and out-stream, using the same RTB activity.
- In-View high viewability format.

On one hand, the workflow is greatly simplified and buying side can deliver video ads in a matter of minutes, instead of dealing with Flash/HTML5 publisher specific templates that are necessary to setup a banner with video. On the other hand, agencies can be rest assured that engagement is high and viewability is guaranteed.

Technical specifications:

- HTML5 based video player;
- In-View;
- Base banner format is IAB Pushdown;
- Connects to PMP/SSP;
- Programmatically buying side exposes it as an In-Stream inventory;
- Video Ad is delivered using VAST/VPAID.

How to get started?

- **Contact** pod.video@adform.com.
- PMP In-Stream placement is generated for a publisher.
- Custom tag is generated (several lines of JavaScript, same as standard Adform tags).
- Publisher has to integrate the tag into a website.

- Inventory can be bought through our RTB.

Test tag

Publishers can use this tag to test how this format works on their page.

```
<script language="javascript" src="http://track.adform.net/adfscript/?bn=5686686"
data-pmp-id="79992"></script>
```

Production/Live tag

Publishers can use these tags to run actual campaigns

Pushdown expandable:

```
<script language="javascript" src="http://track.adform.net/adfscript/?bn=5686686"
data-pmp-id="*****"></script>
```

Standard banner with CSS3 animation:

```
  <div id="adform-container" class="adform-outstream-container" style="height: 0;
display: block; overflow: hidden;">

    <script language="javascript"
src="http://track.adform.net/adfscript/?bn=5895600" data-pmp-id="*****"></script>

</div>
```

A tag used for Jubbi.dk :

```
  <style>

            .adform-container{display:block;margin:0;padding:0}@media(min-width:
768px){.adform-container{max-width:516px}}@media(min-width: 1000px){.adform-
container{max-width:419px}}@media(min-width: 1600px){.adform-container{max-
width:597px}}

            </style>

            <div class="adform-container">

                <script language="javascript"
src="http://track.adform.net/adfscript/?bn=5686686" data-pmp-id="87269"></script>

            </div>
```

Async tag example:

```
<script type="text/javascript">

    (function(window, document, Adform){

        window._adform = window._adform || [];

        _adform.push(['5686686.on.init', function (settings) {

            var flashvars = settings.html.flashvars;

            flashvars.pmpId = *****;

        }]);

    })(window, document, (Adform = window.Adform || {}));

</script>



<script data-adfscript="track.adform.net/adfscript/?bn=5686686">

(function(c,b,e,a,d){

    c.getElementById("adform-adf"+b)||(a=c.createElement(b),

    a.type="text/java"+b,a.async=a.defer=!0,a.id="adform-adf"+b,


a.src="http"+e+"://s1.adform.net/banners/scripts/adfscript.js?"+Math.round(new
Date/6E4),

    (d=c.getElementsByTagName(b)[0]).parentNode.insertBefore(a,d))

})(document,"script","https:"==location.protocol?"s":"");

</script>
```

***** - PMP In-Stream Placement Master Tag ID. When creating In-Stream placement in the PMP you should assign a creative that has MP4 and WEBM as required file types.

NOTE: If the banner is supposed to run on mobile placements as well **MPG** video must be assigned to in-stream banner for autoplay functionality to work on mobile devices.

Additional parameters:

- **data-pmp-id** - PMP master tag ID.
- **data-vast-id** - (OR) VAST tag BN (either PMP or VAST must be defined to show the ad).

- **data-max-width** - max width for initial banner. If not defined banner will be 100% width.
- **data-max-height** - max height for initial banner. The priority of this setting is over the data-max-width property. You should not use these two at once.
- **data-aspect-ratio** - *string width:height*, ex.: 16:9 or 4:3. Default is 16:9.
- **data-viewport-threshold** - *string percent:time*, ex.: 50:1 or 25:2, or 75:0.5.
- **data-viewport-screen** - *string true*. What to follow, viewport or banner.
- **data-sound-enabled** - *string true*. By default sound is disabled.
- **data-sound-onhover** - *string false*. Trigger sound on/off by hovering mouse on banner area.

Smart Insert

If publisher does not have a specific placement OutStream should be rendered into, one can use "smart insert" feature.

Example:

```
    <script>

        (function(window, document) {

            window.ADFSlot = function() {

                var el = document.querySelectorAll('#entr-content > p');

                var insertion = 'near-fold'; // possible methods: 'near-fold' and
'middle'

                return {

                    paragraphs: el,

                    insertion: insertion

                };

            }

        })(window, document);

    </script>

    <!-- OutStream tag goes here -->
```

- **ADFSlot** global method should return paragraphs from article and insertion method.
- **paragraphs** should use CSS selector syntax in querySelectorAll method.

- If **insertion** property is left empty then you should pass specific element to **paragraphs** before which banner will render.
- Alternatively, you can pass **'middle'** property in the **insertion** parameter and banner will be rendered in the middle of article.

Passbacks

Passbacks can be implemented by publishers to execute custom javascript code depending on PMP outcome.

```
<script>

    (function(window, document, Adform){

        window._adform = window._adform || [];

        _adform.push(['on.init', function (settings) {

            this.on('pmp_empty', onEmpty);

            this.on('pmp_success', onSuccess);


            function onEmpty () {

                // execute custom code when PMP does not return an ad

            }


            function onSuccess() {

                // execute custom code when PMP returns an ad

            }

        }]);

})(window, document, (Adform = window.Adform || {}));

</script>

<!-- OutStream tag goes here -->
```

Other

For any questions please contact pod.video@adform.com.