Interactive Advertising Bureau

MRAID Video Addendum

Abstract

This document describes a standardized implementation of video in MRAID using VPAID.

Version 1.0 Released March 26, 2015

Table of Contents

Acknowledgment

This document was prepared by the IAB's MRAID Working Group. The IAB gratefully acknowledges members from the following companies, who made substantial contributions of time, expertise, and writing to this project.

- ASE Publishing
- BrightRoll
- Crisp Media
- Innovid
- Opera MediaWorks
- Rhythm

For more information about the IAB MRAID standard, please see www.iab.net/mraid. For more information on VPAID, please see www.iab.net/vpaid. Please address questions about this document to mobile@iab.net.

1. Executive Summary

This document standardizes support for video within MRAID ads with a minimal need for changes to existing standards. This addendum extends the capabilities of MRAID with relevant aspects of VPAID according to the following principles:

- Avoid use of VPAID for any functionality that overlaps both specs - in these cases the existing MRAID functionality is used.
- Make minimal additions to MRAID to cover initialization of video ads and other required new functionality.
- Maintain backwards compatibility with MRAID v2.

MRAID-compliant containers/mobile rich media SDKs (which this guideline will refer to as "containers" going forward) that wish to support mobile video ads served within an MRAID container should adopt the requirements and capabilities outlined in this addendum.

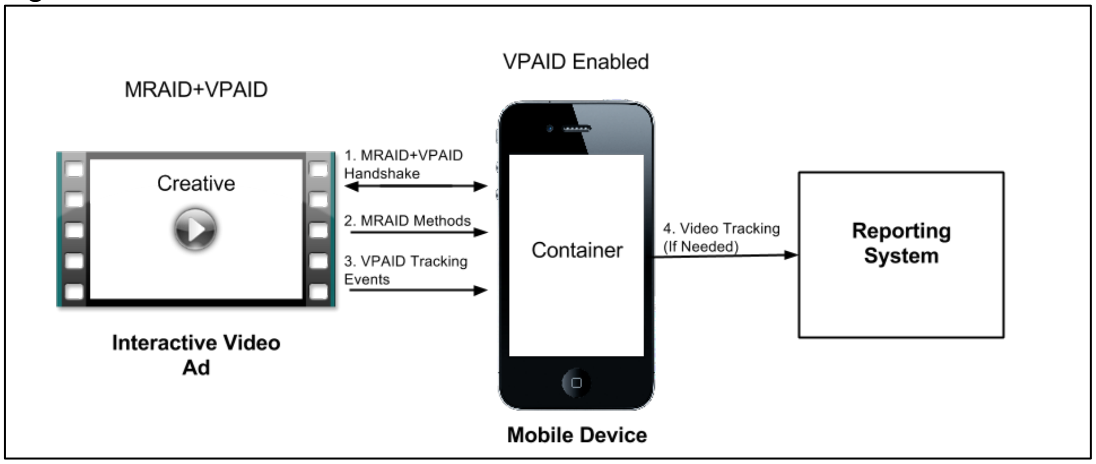
VPAID is an in-stream standard, and the intent of this spec is to support in-stream-like video experiences in an in-app context. To align with the expectations of advertisers buying video impressions, ads using this spec should be interstitial placements as defined by MRAID's getPlacementType() method.

Reporting of video ad impressions must follow definitions established in the IAB Digital Video Ad Impression Measurement Guidelines. (See: http://www.iab.net/media/file/dig_vid_imp_meas_guidelines_final.pdf) While firms may choose to use VPAID metrics capabilities for video that runs in-banner, containers compliant with this document must only count ads running as MRAID interstitials as "video ad impressions." See Section 5.1.

This specification provides the following major features and benefits to ad designers:

- Visibility into in-app video behavior and tracking events metrics via existing VPAID events, methods and properties.
- Clearer MRAID handling of click through events.
- Ability to manage the visibility of the base ad on click (mraid.open) events

Figure 1



2. The Challenge of Interactive Mobile Video

MRAID standardizes communication between an ad creative and the mobile environment (the "container") into which it is served. Publishers and intermediaries currently have visibility into any of the events that require the ad to interact with external controls. However video-specific events, such as video starts, quartiles, and completion metrics that occur within the control of the creative itself are not standardized in MRAID and therefore cannot be tracked consistently across publishers.

VPAID is specifically designed to allow standardized communication between a video creative and a player including extensive communication regarding the video-specific metrics described above.

Neither MRAID nor VPAID alone currently supports an interactive mobile video unit that can take advantage of the full extent of mobile rich media elements while still allowing publishers and intermediaries to have insight into inline video behavior.

By listening to VPAID events, an MRAID container can report on video events in a standardized way that is consistent with the reporting sent by the video ad itself directly to the video ad server. While not strictly mandatory, this is the most straightforward way to ensure alignment in the reporting of video impressions and other metrics between the container provider and the video ad server.

If the ad seller/container provider is billing on video impressions (as opposed to rich media impressions), it is important that the container report video impressions in accordance with the IAB Digital Video Ad Impression Measurement Guidelines. In practice that means counting an impression on the start of the video not when the ad code is loaded.

3. VPAID Support in MRAID Creatives

This section outlines the specifics of how VPAID and MRAID work together to accomplish the goals of this spec.

3.1 VPAID and MRAID Initialization and Ad and Container Recognition

Ads that use MRAID in tandem with VPAID must indicate support to the container via specific steps that happen as the ad starts to run. This initialization process will ensure that both the ad and the container recognize one another and are synchronized. The steps are illustrated in Figure 2 below and outlined in sections 3.1.1 through 3.1.4.

Compatible creatives must go through an additional handshake process after the standard MRAID handshake. This second handshake enables the container to subscribe to VPAID events and should be initiated as soon as possible by the creative, so containers have the needed time to subscribe to video events before the actions occur in the creative.

3.1.1 MRAID Initialization

MRAID ads use the standard mraid.js script tag approach to alert the container that they require MRAID's libraries and other supported features. Ads that comply with the requirements of this addendum must continue to follow that protocol, and wait for either the container to return the MRAID "ready" event or to set the container's state to "default" before proceeding.

3.1.2 Creative Verifies VPAID Support Available

MRAID provides a supports() method that is extended in this specification. A creative must call mraid.supports("vpaid") to query the container. A container using this addendum will respond with a Boolean of true. A container that uses the base MRAID v2 standard will return an error, an undefined result, or a Boolean of false. If the container returns true, the creative may invoke VPAID initialization given in section 3.1.3.

3.1.3 Creative Initializes VPAID

Once the creative has successfully queried the container that mraid.supports("vpaid") is true, the creative must invoke mraid.initVpaid(vpaidObject) to pass the VPAID object to the container. This step must occur before the creative fires any VPAID style events or the container will not be able to subscribe in time.

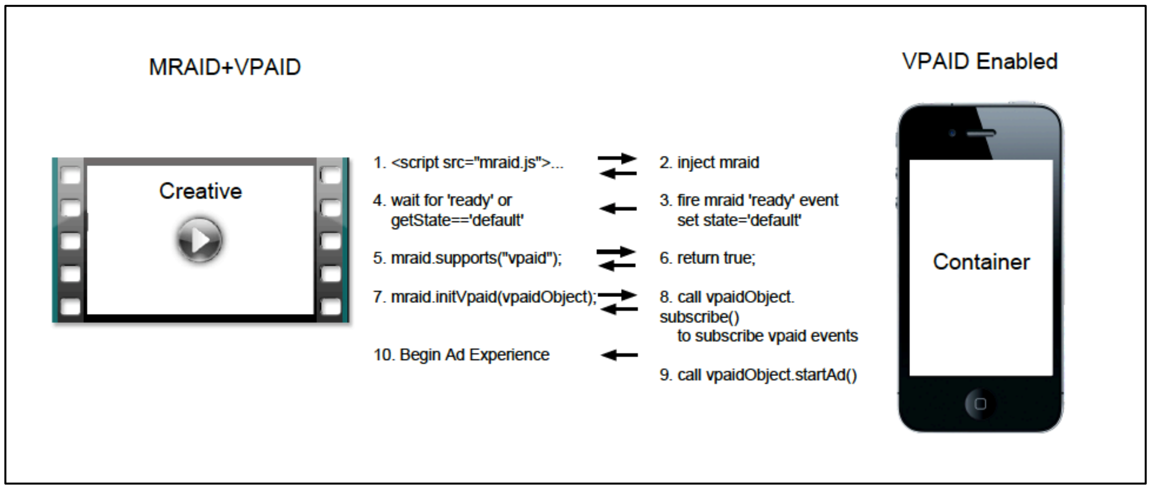
3.1.4 Container Subscribes to Relevant VPAID Events

Upon receiving the initVpaid(vpaidObject) function call, the container will subscribe to any needed VPAID events.

3.1.5 Container Calls vpaidObject.startAd()

Once the container has subscribed to the VPAID events, the container will invoke vpaidObject.startAd() method to allow the creative to start the video creative. As the creative begins to play, it will start to generate VPAID events.

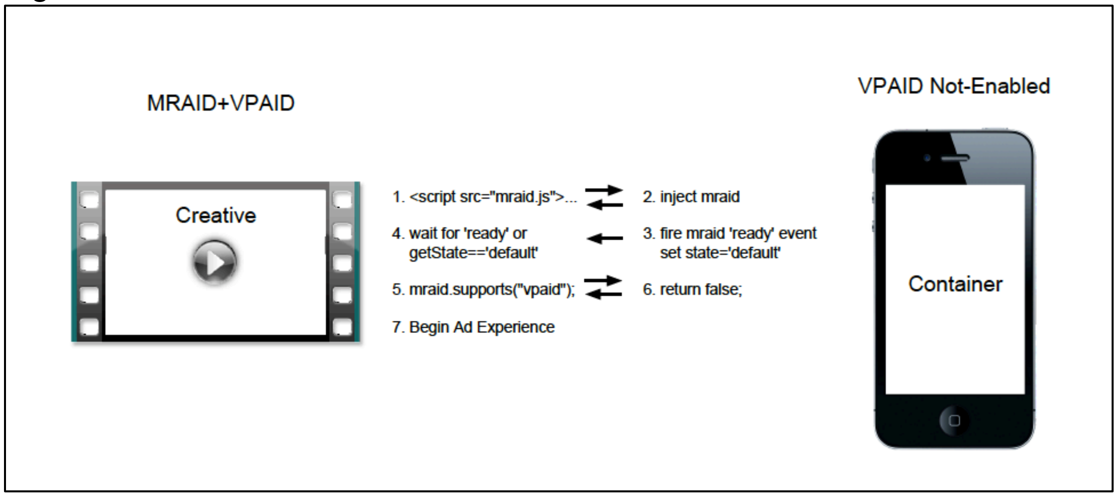
Figure 2


MRAID+VPAID
VPAID Enabled
Creative
Container
1. <script src="mraid.js">...
2. inject mraid
4. wait for 'ready' or getState=='default'
3. fire mraid 'ready' event set state='default'
5. mraid.supports("vpaid");
6. return true;
7. mraid.initVpaid(vpaidObject);
8. call vpaidObject. subscribe() to subscribe vpaid events
10. Begin Ad Experience
9. call vpaidObject.startAd()

3.1.6. Other Scenarios

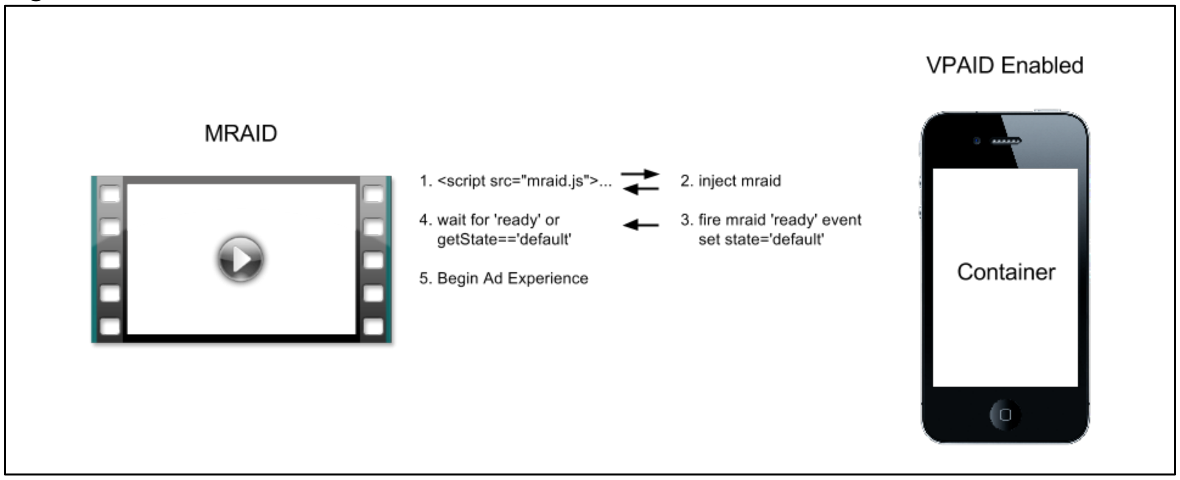
If an MRAID v2 container that does not use this addendum receives a creative that attempts to use VPAID style event, the mraid.supports("vpaid") results in an error, an undefined result, or returns false. The video creative may still play, but the container will not attempt to subscribe to video tracking events and will not call vpaidObject.startAd(). So the creative should begin the ad experience once it receives the false return from mraid.supports("vpaid").

Figure 3


MRAID+VPAID
Creative
1. <script src="mraid.js">...
2. inject mraid
3. fire mraid 'ready' event set state='default'
4. wait for 'ready' or getState=='default'
5. mraid.supports("vpaid");
6. return false;
7. Begin Ad Experience
VPAID Not-Enabled
Container

This addendum was written to enable backwards compatibility and so any standard MRAID v2 creative will work in a container that complies with this addendum via the standard MRAID v2 initialization sequence. (See Figure 4, below.)

Figure 4


MRAID
1. <script src="mraid.js">...
2. inject mraid
3. fire mraid 'ready' event set state='default'
4. wait for 'ready' or getState=='default'
5. Begin Ad Experience
VPAID Enabled
Container

3.2 VPAID Event and Method Support

Once both standards are initialized, the ad and the container can communicate through either the VPAID or the MRAID API depending on the specific need. To prevent overlap between the standards, the ad must use MRAID to control ad behavior, and use VPAID only for measurement functionality that MRAID doesn't support. Stated another way, the container must not do anything to the webview running the ad in response to VPAID events, only in response to MRAID methods.

For example, at the end of a video ad play, an ad may fire a VPAID AdVideoComplete event for tracking purposes, but the container will only dismiss the webview with the video when the ad invokes the mraid.close() method.

3.2.1 Events

MRAID events are dispatched from the container to the ad unit to provide information about the ad space to the ad unit. The ad creative can continue to subscribe to any MRAID events by using the MRAID addEventListener() method. Events will behave as they do with non-video MRAID ads, with one exception: this addendum provides some additional requirements for an mraid.viewableChange event, as described in section 4.4.)

VPAID events are dispatched from the ad unit to the container to provide information about the video element behavior. Containers that are compliant with this addendum should subscribe to VPAID events by using the VPAID subscribe() method.

The following relevant VPAID events will be available for containers to subscribe to. For details on these see Section 3.3 of the IAB VPAID standard (http://www.iab.net/media/file/VPAID_2.0_Final_04-10-2012.pdf).

- AdImpression
- AdStarted
- AdVideoStart
- AdVideoFirstQuartile
- AdVideoMidpoint
- AdVideoThirdQuartile
- AdVideoComplete
- AdClickThru
- AdInteraction
- AdDurationChanged
- AdUserAcceptInvitation
- AdUserMinimize
- AdUserClose
- AdPaused
- AdPlaying
- AdLog
- AdError

3.2.2 Methods

MRAID methods are typically dispatched from the ad unit to the container when the ad needs to interact with the container. All MRAID events must be supported as normal by containers that comply with this specification. Note, however, that the MRAID playVideo method would **not** be used for the video element of the MRAID+VPAID creatives as this interaction requires the video to play inline. One new MRAID method is added here, mraid.initVpaid(vpaidObject). The ad uses this to inform the container that the container needs to prepare VPAID.

To prevent conflicting instructions from the container based on VPAID methods, the supported VPAID methods are restricted to those required to initiate the VPAID API, and VPAID's startAd() method, used to signal the creative that it can begin video playback. Other VPAID methods, such as resizeAd(), skipAd(), and stopAd(), are not supported as the container should not provide instructions to the ad creative for how it should play. This allows the ad unit to remain in control of the overall ad experience. The five VPAID methods that containers compliant with this specification can use are:

- subscribe()
- unsubscribe()
- getAdDuration()
- getAdRemainingTime()

- startAd()

4. Additions to the MRAID Spec

In addition to the use of the VPAID API to support additional video tracking, containers compliant with the MRAID Video Addendum must support some additional capabilities beyond those in the MRAID v2 standard.

New and Extended MRAID Events and Method values

- `supports method()`
    - `vpaid` value
- `viewableChange()` used to support click throughs
- `initVpaid(vpaidObject)`

4.1 Identifying VPAID-Compliant MRAID Containers

supports method: vpaid value

In addition to the arguments allowed for mraid.supports() in MRAID v2, containers compliant with this addendum must also handle mraid.supports(vpaid). This should return "true" for containers that have implemented all the requirements contained in this addendum, in addition to the standard MRAID v2 requirements.

Containers should only return true for mraid.supports(vpaid) if the environment in which the ad is running allows both autoplay video and inline video (see section 4.2).

Table 1: Additional value for mraid.supports()

Value	**Description**
vpaid	The device/container supports this addendum to the MRAID standard and the attendant VPAID capabilities

4.2 Invoking vpaidObject

Containers that comply with this addendum must support a new method, which ads can use to pass a vpaid object to the container. The vpaid object is used to give the container the ability to subscribe to events as well as start the ad once container is ready.

```
mraid.initVpaid(vpaidObject)
```

parameters:

- vpaidObject – a reference to the JavaScript VPAID object present in the ad

return values:

- none

Once the ad has checked for support for VPAID in the container that's running it, it needs to tell the container to start using vpaidObject methods like subscribe. It does this via a new method included in this addendum, mraid.initVpaid(vpaidObject), which provides the container with the vpaidObject reference that it needs to call VPAID methods. After using the vpaidObject to subscribe to VPAID events as needed using vpaidObject.subscribe(), the container should call vpaidObject.startAd() so that the ad will know it is allowed to start the video ad experience.

4.3 Support for Autostart Video

To support the optimal interactive video experience, the container webview must support inline video and autostart video. The video element in the ad creative should be able to start without user interaction and play inline.

Reference to the autostart support parameter for iOS is already included in the MRAID spec under the additional information provided for inline video. Containers supporting this addendum must add autostart support to a webview. Quoting from MRAID v2:

In order to enable inline video playback and autostart of video, MRAID compliant SDKs should consistently insert the any necessary enabling tags into the webview depending on [the] operating system of the device.

For iOS devices, the following tags must be used:

- **`webView.mediaPlaybackRequiresUserAction = NO;`**
- `webView.allowsInlineMediaPlayback = YES;`

Autostart is not currently referenced under the MRAID description for Android inline support, but containers compliant with this addendum must enable autostart on supported Android versions (Jelly Bean and above).

For Android (4.2 and above) devices, the SDK must use webSettings to enable video to play without user interaction:

- `webView = new WebView(this); //this points to the activity`
- `WebSettings webSettings = webView.getSettings();`
- `webSettings.setMediaPlaybackRequiresUserGesture(false);`

Prior to Android 4.2, no changes are required to webSettings to allow autostart support, and autostart can always be assumed.

For devices earlier than Android 2.x, there is no video element so MRAID units including video would not be supported.

If autostart is not supported on a given device, operating system or app, then the container must return mraid.supports(vpaid) as "false", so that the ad creative can either play a non-interactive version of the video content, allow the user to initiate video playback manually, or take some other action.

Note, however, that video ads requiring manual playback should be an edge case - advertisers should make sure that publisher/network ad servers target around OS versions that do not support autostart to make sure this is the case.

4.4 Support for Click-through Behavior

When the container opens an external webview in response to the MRAID open() method, there is no way for the container to indicate to the creative the shift from the current webview as within the users focus to the second webview. Likewise, when a user closes the second webview and returns to the first webview, there is currently no method to indicate that the user has resumed focus on the main webview. This prevents the creative from knowing when to pause any animation, such as a video element, when the user moves to a second webview and when to resume it when the user returns to the main webview.

In the interest of minimizing development by publishers/SDK owners, it is advisable to use currently supported MRAID functionality to facilitate this communication between the container and the ad unit. The viewableChange event, per the existing MRAID v2 spec, "fires when the ad moves from on-screen to off-screen and vice versa". Since the container is managing a webview for the click through which entirely covers the ad unit, this is consistent with the existing spec.

Ad developers should be aware that using viewableChange for viewability measurement purposes may not accurately reflect whether the user has left the ad creative.

1. On requesting the URL to display (this can either be from the main ad unit, or from within an interactive engagement slate), the ad unit calls mraid.open()
2. The container opens and maintains a new webview over the existing ad unit and fires viewableChange(false). This webview must contain a close button, and may also contain other controls as desired (back button that applies to the webview content, for example).
3. Upon calling mraid.open(), the creative must also fire VPAID AdClickThru, in order for the integration layer to track the click via VPAID if desired. Note that the AdClickThru event has three parameters (url, id, and playerHandles). Containers implementing this addendum should refer to VPAID v2.0 Section 3.3.14 for more information on working with the AdClickThru event.
4. If the ad elects to pause the video element when a second webview is opened on top of the primary webview, the ad must fire the VPAID event AdPaused in order to indicate the video was paused.
5. When the second webview is closed, the container must fire viewableChange(true) in order to inform the creative that the ad is now in view again. The creative can then resume from where it left off. Whenever video playback resumes, the VPAID AdPlaying event must be fired.

5. Additional Requirements for Interactive Mobile Video

Containers and ads that use this specification's combination of MRAID and VPAID capabilities must follow three additional constraints to ensure consistency across implementations and compliance with other relevant IAB guidelines.

5.1 Video Ad Impressions Must Be Counted Appropriately

MRAID and VPAID are creative specifications, not measurement guidelines. However, ads that comply with this addendum must follow IAB digital video ad definitions and the IAB Video Impression Measurement Guidelines and fire AdImpression events appropriately, when the video actually starts playing:

> *A valid digital video ad impression may only be counted when an ad counter (logging server) receives and responds to an HTTP request for a tracking asset from a client. The count must happen after the initiation of the stream, post-buffering, as opposed to the linked digital video content itself.*
>
> *Specifically, measurement should not occur when the buffer is initiated, rather measurement should occur when the ad itself begins to appear on the user's browser, closest to the opportunity to see.*
>
> *Source: http://www.iab.net/media/file/dig_vid_imp_meas_guidelines_final.pdf*

Additionally, containers following this addendum should only count video ad impressions on the VPAID AdImpression event. If the container is treating the ad as a rich media ad, the container can count the impression on adload as usual, it does not wait for the AdImpression event.

Moreover, only ads that interrupt the flow of content and take up substantially all of the window can be counted as "video ads." MRAID ads that the container identifies as "interstitial" ads for the purposes of the mraid.getPlacementType() method comply with this definition. Containers compliant with this addendum should track and report VPAID events for such ads, and only for such ads. By contrast, any ads running as "inline" ads by MRAID's definition may contain video, but must never be counted as video ads, and containers should not report their VPAID events for measurement purposes.

The following table lists six common scenarios when video is found in mobile ads, highlighting when a compliant container may report them as "video ads" for counting and reporting purposes.

Table 1: Mobile Video Formats and VPAID Applicability

Format	Description or Example	MRAID placementType	Report VPAID events?
Video interstitial, non-dismissible	E.g. a pre-app ad or a video interstitial between two levels of a game	interstitial	Yes
YouTube-like video pre-roll, skippable after 5 seconds	E.g. a 640x480 placement on an iPad that takes over 3/4 of a screen and plays before the video content	interstitial	Yes
Post-roll with an end-card	E.g., a 15 second post-roll video with an interactive end-card that lingers.	interstitial	Yes
Simple banner that plays a video on click	E.g., a 300x250 banner in a newspaper app	inline	No
A banner that on click expands and plays a video	E.g. a 320x50 static banner that on tap calls mraid.expand() and plays a video in a <video> element. After the video finishes, the ad collapses. For the user, experience is almost the same as using mraid.playVideo() from the banner... a basic tap-to-video.	inline	No
A banner that on click expands and offers a grid of videos	Similar to the previous, but when expanding, instead of a video playing immediately, the creative contains a grid of poster images for different videos. The user may choose and play multiple before closing.	inline	No

5.2 Additional Container Elements Must Not Overlap Video Creative Area

To allow the ad creative studio to fully design the experience of the ad unit, the container should not overlay additional elements, such as countdown timers, 'advertisement' notifications, etc., on top of the area allotted to the creative. These additional elements can obscure the creative and block interactivity.

If a container has elements it needs to include during the ad, they should be restricted to an area outside of the boundaries exposed to the ad creative for display. In response to the getCurrentPosition() method and getMaxSize() or any resize() methods, the container should return values that exclude the area required for its additional elements.

This requirement is covered in the current MRAID spec but frequently overlooked - the getScreenSize() section includes the note that this method "will return the TOTAL size of the device screen, including area (if any) reserved by the OS for status/system bars or other functions." This is to establish it in contrast to the getMaxSize which should exclude any area required for the additional container elements.

In addition to providing a clear area for the creative to control without overlaying elements, the container should never include additional video controls as these may conflict with the creative controls.

5.3 Styling the Close Indicator to Delay Video Skippability

MRAID requires that expandable and interstitial ads always be closable by a user, via a control located in the upper-right corner of the ad webview. However, to provide flexibility to creative designers, MRAID allows an ad to suppress the default, container-supplied close indicator (e.g., a box with an x) in favor of a close indicator supplied by the creative itself. Ad designers unfamiliar with this aspect of MRAID should review the section of MRAID v2 titled "Closing Expandable and Interstitial Ads."

This addendum does not change the requirement for a container-supplied close control. This creates a challenge for ad designers seeking to run video ads that are skippable only after a set duration—say 5 or 7 seconds. We recommend that ad designers seeking to create such ads use MRAID's useCustomClose() method, and style the close indicator as a transparent image for whatever duration the ad wants to delay skippability, replacing it with a visible indicator after that.

This is not an ideal solution—a viewer will still be able to tap the top-right corner to dismiss the video, even before the close indicator becomes visible. However, this approach has the virtue of making it as easy as possible for MRAID v2-compliant containers to support this addendum.