SAS® 360 Match

Android SDK

Updated October 28, 2022

Contents

Android SDK

The SAS® 360 Match Android SDK provides UI elements encapsulating the request and rendering of SAS® 360 Match ads within Android mobile applications. The API provides functionality similar to that of Apple's iAd Framework and Google's AdMob/DoubleClick API.

The SDK implementation supports version 2.0 of the IAB's MRAID specification for rich media advertisements. MRAID-compliant ads can request expansion to full screen or an arbitrary size, provide for two-part creatives, respond to visibility changes, control screen orientation, and engage with other device functions.

An ad placement is represented by either an `Ad` or `InterstitialAd`, both subclasses of `AbstractAd`. The first type, `Ad`, is used in cases where the ad is mixed with other application content in the same screen ("inline"). The second type, `InterstitialAd`, is used in cases where the ad should appear full screen, blocking other application content until the user explicitly dismisses it.

Each type is capable of rendering one ad at a time, although it can be reused and requested to show different ads over time. Each ad request is represented by an `AdRequest` that encapsulates all targeting parameters for the ad request, including size. The same `AdRequest` can be used multiple times, if appropriate, such as to refresh the ad object to show a different ad.

To show any type of ad, the application first creates the `Ad` or `InterstitialAd` object and asks it to load and render an `AdRequest`. The request and rendering occur asynchronously. The application can provide the ad an `AdDelegate` that receives callbacks when the ad has been loaded successfully (or not).

For inline ads, the `Ad` provides access to its underlying view, so that the app can set its layout appropriately and include it in an Activity's view hierarchy. It is the app's responsibility to ensure that the view size is appropriate for the size of ad requested from SAS® 360 Match. A scale factor can be set on the ad to compensate between differences in the rendered ad's pixel dimensions and the dimensions of the view in which the ad appears.

The app is also responsible for how and when the view appears (or not) while loading is in progress. Using conventional UI framework methods, the app can keep the view hidden until the load is successful, can show a superview containing an activity spinner, can show a placeholder image, etc.

For interstitial ads, the app merely calls the `InterstitialAd`'s `show` method. The ad is presented modally, blocking the application's regular user interface. A black "close" icon is automatically added to the upper right corner of the ad. MRAID-compliant ads can override showing the close icon to allow a creative-specific close graphic to appear.

By default, when a user touches an ad, its "click" action triggers a full-screen in-app interstitial view of the action's web page. An `AdDelegate` can be notified when an in-app interstitial ad action has been dismissed by the user. If the app sets the ad's `actionInBrowser` property,

the action launches a page in the device's default browser instead and causes the app to be pushed to the background.

But the `AdDelegate` can also block the normal ad action and take an action for itself by implementing the `willBeginAction` method. In combination with calling the ad's `executeJavaScript` method, this gives the app powerful methods to customize a user's interaction with the ad.

Setting Up the Android SDK

The mobile SDK for Android is accessible in a privately managed maven repository.

If you previously used directly downloaded packages for the mobile SDK, remove the SASCollector.jar file from your app's lib folder.

1. To add the repository as a dependency source to your project, add the following block to the repository's definitions in the top level build.gradle file in your project source code:

```
allprojects {
  repositories {
    google()
       mavenCentral()
          maven {
              url
                'https://sdk.ci360.sas.com/android'
              }
       }
}
```

2. Add the mobile SDK as a dependency in your app/build.gradle file or other submodule where you intend to use the SAS Customer Intelligence 360 mobile SDK APIs. For example:

```
dependencies {
    implementation "com.sas.mkt:mobile-sdk-android:<version>"
}
```

3. Perform a project sync between Gradle and Android Studio to ensure that the library is accessible.

Compatibility

The SDK is compatible with Android versions 3.0 (API level 11) and higher and has been tested through version 5.0 (API level 21).

The SDK provides view containers that support rendering MRAID 2.0-compliant ads. All MRAID functions and behaviors are supported as described in the IAB specification "Mobile Rich-media Ad Interface Definitions (MRAID) v.2.0", dated April 16, 2013, with these clarifications and exceptions:

- Knowledge of the current visibility of an inline (non-interstitial) ad requires assistance from the app—specifically, the Activity hosting the ad. Unless the app provides such support, an inline ad is always regarded as “visible”. See the AbstractAd’s onVisibilityChange method below. Interstitial ads take care of this automatically.

- An inline ad in the “default” state takes no action itself when MRAID close() is called. A callback is provided so that the app can take an appropriate action (for example, hiding the ad, loading a new ad, or ignoring the close altogether). See the AdDelegate’s willClose method below.

- The “storePicture” and “createCalendarEvent” MRAID methods are not yet supported.

Building

The app’s manifest.xml must include the following:

```
<uses-permission android:name="android.permission.INTERNET" />
```

and in the `<application>` section:

```
    <activity android:name="com.sas.ia.android.sdk.InterstitialActivity"
         android:configChanges="orientation|screenSize|keyboard|keyboardHidden" />
    <activity android:name="com.sas.ia.android.sdk.InterstitialWebActivity"
         android:configChanges="orientation|screenSize|keyboard|keyboardHidden" />
```

The SDK also includes several resources that must be added to the app’s res/drawable-xhdpi directory, and one asset file that must be added to the app’s assets directory.

Changes from Version 2.x

- MRAID 2.0 ads are now supported, rather than just MRAID 1.0.

- Android API level 11 is the minimum version supported.

- Static methods in AdRequest must now be used to set the SAS® 360 Match customer id and domain used by subsequent ad requests. Prior versions relied on supplying these with each initialization of an AdRequest.

- The Ad class is now a subclass of RelativeLayout, and therefore can be referenced when building UI layouts with interactive tools.

Classes in com.sas.ia.android.sdk

AbstractAd

The following methods and properties apply to both **Ad** and **InterstitialAd** objects.

```
public void setDelegate(AdDelegate delegate)
public AdDelegate getDelegate()
```

- The delegate for the ad. Defaults to null (no delegate).
- The AbstractAd maintains a weak reference to this object. The app should maintain a strong reference while the delegate is assigned to this ad.

```
public Activity getParentActivity()
```

- The activity hosting display of this ad's view. For an `Ad`, this is the same Activity given to its constructor. For an `InterstitialAd`, this is the Activity of the interstitial view of the ad.

```
public WebView getWebView()
```

- Returns the underlying WebView that renders the ad. This view is a subview of the top-level ad view and should not be added to the app's view hierarchy directly. Access is provided here in cases where the app needs to further customize settings in the WebView.

```
public void setScale(int scale)
```

- Sets how the underlying WebView scales the ad content. This can be set only once, prior to loading an ad in the view. The scale affects how the pixel size of the ad is scaled to device pixels.
- `scale` is the scale factor, expressed in percent. A value of 0 (the default) causes the WebView to perform variable scaling depending on the device's screen density. In this case, the app should size the ad's view to be the equivalent of the ad's size in dips (device independent pixels).
- If `scale` is anything >0, that exact scaling is used. Especially, when `scale` is 100, a 1-to-1 pixel scaling is used, and the ad rendered in the view is expected to match its size exactly. If not, white space might appear or parts of the ad could be clipped.

```
public void load(AdRequest adRequest)
```

- Initiates asynchronous loading and rendering of a new ad in the ad's view.
- Upon successfully loading a non-default ad, the delegate's `onLoaded` method is called.
- Upon successfully loading a default ad, the delegate's `onDefaultLoaded` method is called.
- Upon failure, the delegate's `onLoadFailed` method is called.

```
public int fcid()
```

- The internal SAS® 360 Match id of the flight creative (ad) loaded. The FCID is known only for ads set to use beacon counting.

    - =-2 when an ad with an unknown FCID is loaded.
    - = -1 when no ad is loaded.
    - = 0 when a default is loaded.
    - >0 when an ad with a known FCID is loaded.

```
public boolean isLoaded()
```

- Set to True when a non-default ad is successfully loaded.

```
public boolean isDefaultLoaded()
```

- Set to True when a default ad is successfully loaded.

```
public void close()
```

- Requests closing presentation of this ad. The AdDelegate's `willClose` and `onClose` methods are called and have the effect as described in the AdDelegate API section below.
- This method has the same effect as an MRAID-compliant ad calling the MRAID `close()` method.
- For an `InterstitialAd`, this method is equivalent to the user touching the ad's close icon.

```
public boolean isActionInProgress()
```

- Set to True when the ad has switched to its in-app "action" mode, where another view is temporarily presented on top of the regular application user interface, in response to the user touching this ad, and for the purpose of further engagement with the ad or advertiser.
- This property is not set when an ad action is launched in the device's browser (`isActionInBrowser` is true).

```
public void setActionInBrowser(boolean actionInBrowser)
public boolean isActionInBrowser()
```

- When false (the default), an ad's action is presented within the app as a full-screen interstitial.
- When true, the action is presented in the device's default browser, and the app is sent to the background.

```
public Activity getActionActivity()
```

- The Activity hosting display of the ad's action view. This is non-null only when a user has touched an ad and caused its action view to be presented within the app.

```
public void cancelAction()
```

- Cancels any in-app action initiated from this ad. If such an action is in progress, the action's view is removed, allowing the application's user interface to become active again. If an ad action is launched in the device's browser (`adActionInBrowser` is true), this method has no effect.

```
public void onVisibilityChange(boolean visible)
```

- Informs the ad of changes to its visibility, typically determined by the Activity hosting it. This method exists solely to inform MRAID-compliant ads of visibility changes, so that they might adjust their behavior when becoming visible or not visible. This happens automatically for an `InterstitialAd`, but an `Ad` needs the assistance of its parent Activity.

```
public String executeJavaScript(String js, String jsStringExpression)
```

- Executes arbitrary JavaScript in the underlying WebView holding the most recently loaded ad. This method can be used to inspect the ad's contents, manipulate it, etc.

- The `js` string, if non-null, must consist of complete JavaScript statements, functions, etc. This JavaScript is executed immediately before evaluating the `jsStringExpression`.

- The `jsStringExpression`, if non-null, must be a JavaScript expression yielding a String. The result of evaluating this expression is returned as the method result. If null, a null string is returned.

- The JavaScript is executed synchronously. If execution takes more than 1 second, execution is canceled and a null is returned. If either of the strings provided are not valid JavaScript, a 1-second time-out also occurs and a null is returned.

Ad (extends AbstractAd)

```
public Ad(Activity activity)
public Ad(Activity activity, AttributeSet attrs)
public Ad(Activity activity, AttributeSet attrs, int defStyle)
```

- Constructors.

- The parameters are applied to the underlying View that is instantiated as a part of this ad. `activity` is the activity hosting presentation of this ad.

```
public View getView()
```

- Returns the view hosting presentation of the ad. This method is provided for compatibility with previous versions, as the Ad is itself the View hosting the ad. This view should be added to an Activity's view hierarchy, as necessary, to mix this ad's display with other app content on the same screen. Loading of an ad into this view (via `load`) can be done before or after the view has been added to the Activity's view hierarchy.

InterstitialAd (extends AbstractAd)

```
public InterstitialAd(Activity activity)
```

- Constructor.
- `activity` is the activity on top of which this interstitial ad will appear.

```
public void show()
```

- Presents this ad as a full-screen interstitial covering the regular application user interface. Loading of an ad (via `load`) can be done before or after this method is called.

AdRequest

```
public static String domain;
```

- The ad serving domain used in all subsequently created requests. If the domain doesn't start with "http://" or "https://", "http://" is assumed.
- This member must be set before instantiating any AdRequest objects.

```
public static String customerId;
```

- The customer's SAS® 360 Match id used in all subsequently created requests.
- This member must be set before instantiating any AdRequest objects.

```
public AdRequest(Map<String,String> tags)
```

- Constructor for data encapsulating an ad request.
- `tags` is the complete collection of tags and values to supply in the ad request, represented as key/value pairs. The tag names and their values should be the same would be required for conventional SAS® 360 Match ad serving.
- For tags without values (such as NOCOMPANION), supply `null` for the value.

```
public AdRequest(String url)
```

- Constructor that provides the complete URL to be used in the request, including the domain and customer id.

- The older constructor, `AdRequest(String domain, String customerId, Map<String,String> tags)`, is considered deprecated.

MRAIDWebView

Each Ad and InterstitialAd creates an MRAIDWebView object. The object is a container for holding the underlying ad content and provides the additional functionality required by MRAID-compliant rich media ads. There is no need for an app to instantiate MRAIDWebView objects directly.

But an app might set any of several static features of MRAIDWebView, as desired, described below.

```
public static boolean mraidTracing;
```

- A tracing option, used for troubleshooting MRAID ads. The default is false. When true, the MRAID behavior of loaded ads is traced in detail in the ADB console.

```
public static boolean supportsSMSText;
```

- Boolean for whether an MRAID ad should be allowed to send an SMS text message.

- Default is true.

```
public static boolean supportsTelephone;
```

- Boolean for whether an MRAID ad should be allowed to initiate a telephone call.

- Default is true.

```
public static boolean supportsPicture;
```

- Boolean for whether an MRAID ad should be allowed to store a picture on the device.

- Default is false.

- The picture storage feature is not yet supported by MRAIDWebView, so this setting is ignored.

```
public static boolean supportsCalendar;
```

- Boolean for whether an MRAID ad should be allowed to store a calendar event on the device.

- Default is NO.
- The calendar event storage feature is not yet supported by MRAIDWebView, so this setting is ignored.

AdDelegate

To receive callbacks, an app must subclass AdDelegate and override whichever methods are of interest.

```
public void onLoaded(AbstractAd ad)
```

- Called after an ad succeeds loading an advertisement (initiated by `load`), and the ad loaded is not a default.
- Typically, this should trigger adding the ad view to the user interface, or animating it into position, etc.
- `ad` is the ad just loaded.

```
public void onDefaultLoaded(AbstractAd ad)
```

- Called after an ad succeeds loading an advertisement (initiated by `load`), but the ad loaded is a "default" ad. Receipt of a default ad means that the ad server currently had no other ad to serve that could fulfill the `AdRequest` made.
- `ad` is the ad just loaded.

```
public void onLoadFailed(AbstractAd ad, int errorCode, String description,
      String failingUrl)
```

- Called after an ad fails loading an advertisement (initiated by `load`).
- `ad` is the ad that failed to load.
- `errorCode` and `description` describe the error, as returned by the underlying WebView.
- `failingUrl` is the URL whose rendering failed.

```
public boolean willClose(AbstractAd ad)
```

- Called when an ad is about to close. This occurs when a user touches an `InterstitialAd`'s close icon, or when an MRAID-compliant ad asks to be closed, or when the ad's `close` method is called.
- `ad` is the ad that is about to close.

- Returning true for an `InterstitialAd` allows it to remove itself from the screen. Returning false blocks it from closing.
- An `Ad` takes no action itself upon close, but this method allows the ad's Activity to take some action to close the ad, if it desires. Returning true only changes the ad's MRAID state to "hidden", and returning false leaves the state alone.

```
public void onClosed(AbstractAd ad)
```

- Called when an ad is closed.
- `ad` is the ad that was closed.

```
public boolean willBeginAction(AbstractAd ad, String url)
```

- Called when an ad is about to initiate an action in response to a user touching some portion of the ad, or in response to an MRAID-compliant ad calling the `open()` method.
- `ad` is the ad whose action is about to begin.
- `url` is the destination URL for the action.
- Returning false blocks the action from occurring.

```
public void onActionFinished(AbstractAd ad)
```

- Called when an ad action's interstitial view is dismissed. When the ad action was shown in the device's browser, this method is not called.
- `ad` is the ad whose action has just finished.

```
public boolean willExpand(AbstractAd ad, String url)
```

- Called when an MRAID-compliant ad is about to expand itself to cover the current screen.
- `ad` is the ad that will expand.
- `url` is null if the ad renders itself in the expanded area. If non-null, then it is the destination URL of additional content that will be displayed in the expanded area.
- Returning false prevents the expansion from occurring.

```
public void onExpandFinished(AbstractAd ad)
```

- Called when an ad's expanded display has closed.

- `ad` is the ad whose expanded display has just closed.

```
public boolean willResize(AbstractAd ad, Rect size)
```

- Called when an MRAID-compliant ad is about to resize itself and break out of its current screen.
- `ad` is the ad that will resize.
- `size` gives the position and size the ad will adopt after being resized.
- Returning false blocks the resize from occurring.

```
public void onResizeFinished(AbstractAd ad)
```

- Called when an ad's resized display has closed and the ad has returned to its former position and size.
- `ad` is the ad whose resized display has just closed.