iOS VMG SDK integration

This guide will assist you during the integration of our SDK. The repository includes an example app with most common configuration examples.

Compatibility

The VMG SDK is compatible with iOS 8+.

Integration

1. Prerequisites

Before starting you should have available to you a test `app_id` and at least one test `placement_id`, you can acquire these through your account manager at VMG. It's important to keep in mind that before submitting your app to the App Store you will need to remove your test `website_id` and `placement_id`. The `app_id` is used for basic configuration of the SDK, a `placement_id` is used for an ad view in your app. Your app can have multiple unique placement id's for different sections in your app, e.g. for the news section and sport section, this is useful for category targeting.

Get the SDK

Our SDK is available as a static library or, preferably, trough Cocoapods.

Cocoapods

Add `pod 'VMGSDK'` to your Podfile and run `pod install` to install our SDK through Cocoapods.

Static Library

Alternatively add our SDK as a static library. Download the latest version of the VMG SDK from our VMG GitHub repository. In your project right click on the frameworks group and select "Add files to..." and choose the `VMGSDK.framework`. Now go to the build phases of your project and check if 'Link binaries with libraries' contains `VMGSDK.framework`.

Apple App Transport Security

Despite our SDK being fully SSL ready some of the third party ad servers do not use SSL yet. Because of this we need to disable ATS to allow our SDK to function properly.
In your project `Info.plist`, add the following parameters:

```
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSAllowsArbitraryLoads</key>
    <true/>
</dict>
```

When submitting an app with ATS disabled you have to justify it, telling them it's needed for advertising needs is sufficient.

2. Initialise SDK with config

In your AppDelegate import our framework, `import VMGSDK`.
In the `application:didFinishLaunchingWithOptions` function, initialise our SDK: `_ = VMGSDK(withAppId:<Your App ID>)`.

3. Add an ad view

The ad view has a few options for implementation, it can work as a static view which auto plays and expands when an ad is loaded, and it can act as an InPageVideo format in a scrollview, pausing and playing the view as it gets scrolled in and out of view. Both of these view types are available with and without auto layout. If you find yourself wanting more options, let us know.

The view is flexible and can be integrated in the following components:

- UIView
- UIScrollview
- A view that inherits from UIScrollview, like UITableView, UICollectionView, ...
- Others

3.1 Set up

Import our framework, `import VMGSDK`.
Declare that your class implements the `VMGAdViewDelegate` delegate;

```
class ViewController: UIViewController, VMGAdViewDelegate {
```

At the point of initialisation of your class, for example `viewDidLoad` in a view controller, initialise the adview.
Depending on your needs this can be done in a few ways:

- `VMGAdView(withPlacementId placementId: Int, delegate: VMGAdViewDelegate)`
- `VMGAdView(withFrame frame: CGRect, placementId: Int, delegate: VMGAdViewDelegate)`
- `VMGAdView(withPlacementId placementId: Int, heightConstraint: NSLayoutConstraint, delegate: VMGAdViewDelegate)`
- `VMGAdView(withPlacementId placementId: Int, placeHolderView: UIView, heightConstraint: NSLayoutConstraint, delegate: VMGAdViewDelegate)`

Using auto layout

With auto layout the ad view requires a height constraint that is attached to the ad view, or its placeholder view. This height constraint is then updated when an ad is loaded and the player expands. Using a placeholder view is optional, this is useful when using interface builder, when used the ad view attaches itself to the view and matches it in size.
After you've created your constraint and/or the placeholder view, add them as a parameter when initialising the view.

Use one of these two functions to initialise your SDK:

- `adView = VMGAdView(withPlacementId placementId: Int, heightConstraint: NSLayoutConstraint, delegate: VMGAdViewDelegate)`
- `adView = VMGAdView(withPlacementId placementId: Int, placeHolderView: UIView, heightConstraint: NSLayoutConstraint, delegate: VMGAdViewDelegate)`

As a stand alone view

Without auto layout the ad view updates its own frame with an animation to expand when an ad starts playing.

Using one of these two functions to initialise your SDK:

- `adView = VMGAdView(withPlacementId placementId: Int, delegate: VMGAdViewDelegate)`
- `adView = VMGAdView(withFrame frame: CGRect, placementId: Int, delegate: VMGAdViewDelegate)`

If you want to react to the ad view expanding, the `VMGAdViewDelegate` has you covered. It has 6 optional functions that will let you know if the ad changes in size. These functions are:

- `adViewWillOpen(adView:VMGAdView)`
- `adViewDidOpen(adView:VMGAdView)`
- `adViewWillClose(ad:VMGAdView)`
- `adViewDidClose(adView:VMGAdView)`
- `adViewWillResize(adView:VMGAdView, fromSize:CGSize, toSize:CGSize, animated:Bool)`
- `adViewDidResize(adView:VMGAdView, toSize:CGSize, animated:Bool)`

If you want to match your animation duration to our expand animation, you can use the static property `VMGAdView.animationDuration` which is a `TimeInterval` object.

3.2 Inside a scroll view

When you want to use the ad view as an InPageVideo format, you will first need to disable autoplay; `adView.autoPlay = false`. By doing so, the SDK will control the ad view expansion and playback state.
Make your viewController implement the `UIScrollViewDelegate` and implement the `scrollViewDidScroll:` delegate function, in your implementation call the `containerDidScroll:` function on the ad view

```
class ViewController:UIViewController, UIScrollViewDelegate, VMGAdViewDelegate {
    <...>
    func scrollViewDidScroll(_ scrollView: UIScrollView) {
        adView.containerDidScroll(withScrollview: scrollView)
    }
```

3.3 Informing the ad view about view controller events

Let the ad view know when the user is entering and leaving the view:

```
    override func viewDidAppear(_ animated: Bool) {
        super.viewDidAppear(animated)
        adView.viewDidAppear()
    }

    override func viewDidDisappear(_ animated: Bool) {
        super.viewDidDisappear(animated)
        adView.viewDidDisappear()
    }
```

With this functionality the SDK can pause and refresh the ad view when a user leaves or enters a view.

4. Build and run!

If everything went right, and your placement has (debug) fill, you should now be able to view your first advertisements!

5. Optional

Optionally you can implement the `VMGAdViewDelegate` function `openBrowserWithUrl(_ url: URL)` if you want a custom implementation of the player clicks. By default this function has an implementation that opens the click URL in a `SFSafariViewController` instance or a `JSQWebViewController`, for iOS 8.