Show Native Ads on iOS

Show Native Ads on iOS

Native ads give you the ability to create ads that are customized to match the look and feel of the rest of your application. This page describes our Native Ads API at a high level, and includes a usage example. For a complete reference, see the inline documentation in the code.

Native networks supported through **mediation**:

- inMobi
- Facebook
- Yahoo Flurry
- AdMob and DFP

> **Maintain references to request and response objects!**
> Maintain references to request and response objects!
>
> You must maintain a reference to the `ANNativeAdRequest` and `ANNativeAdResponse` objects you create. Otherwise they will go out of scope and cause unexpected behavior.

In order to serve native ads, you will send a "native ad request" and receive a "native ad response". In the example request and response below, we:

- Set up a request object, and set some of its properties such as the placement ID and whether to pre-load the ad's icon image.
- Optionally, set the renderer_id for this NativeAdRequest. (For more on renderer_id see Native Layout Service.) The renderer_id needs to be specified for vastxml, likes, downloads, saleprice, phone, address, display URL to be returned in the NativeAdResponse.
- Assuming the request is successful, we load the native ad assets from the response into the view and register it so that we can track user interactions such as clicks

On This Page

Request

First, we set up the request object and set some of its properties such as the placement ID and whether to pre-load the ad's icon image:

```
ANNativeAdRequest *request = [[ANNativeAdRequest alloc] init];
request.placementId = @"123456";
request.rendererId = 123;
request.gender = ANGenderMale;
request.shouldLoadIconImage = YES;
request.shouldLoadMainImage = YES;
request.delegate = self;
[request loadAd];
```

Response

Assuming the request is successful, we load the native ad assets from the response into the view and register it so that we can track user interactions such as clicks:

```
- (void)adRequest:(ANNativeAdRequest *)request didReceiveResponse:(ANNativeAdResponse
*)response {
      // (code which loads the view)
      MyDummyView *view;
      view.title.text = response.title;
      view.text.text = response.body;
      view.iconImageView.image = response.iconImage;
      view.mainImageView.image = response.mainImage;
      [view.callToActionButton setTitle:response.callToAction
forState:UIControlStateNormal];

      response.delegate = self;

      [response registerViewForTracking:view
                withRootViewController:self
                        clickableViews:@[view.callToActionButton, view.mainImageView]
                                 error:nil];

   [response unregisterViewFromTracking];
 }
```

In this example response, we use several elements of a native ad:

- A title
- An icon image
- The main ad image
- Bodytext
- A "call to action" button that the user can click to convert

List of Fields Supported in Native

As if version 5.0 of the Mobile SDK, support for native assets is aligned with how native creatives are set up on Console.

If you are still using Legacy Native in Console, you will need to move to "New" Native for your creatives.

The following is a comprehensive list of native assets supported in the SDKs.

Asset	Supported Pre 5.0?	Supported Post 5.0?	v5.0+ API-Usage Example

Image, Width, Height	Yes, Yes, Yes	Yes, Yes, Yes	`response.mainImage;`
			`response.mainImageSize;`
			`response.mainImageURL;`

Icon+Width+Height	Yes, No, No	Yes, Yes, Yes	`response.iconImage;`
			`response.iconImageURL;`
			`response.iconImageSize;`

Title	Yes	Yes	`response.title;`
Sponsored by	Yes	Yes	`response.sponsoredBy;`
Body text	Yes	Yes	`response.body;`
Desc2	Yes	Yes	`response.additionalDescription;`
Call-to-action	Yes	Yes	`response.callToAction;`
Rating, Scale	Yes, Yes	Yes, No	`response.rating;`

Likes	No	Yes (json only)	`NSDictionary *customElements = response.customElements[@"ELEMENT"];`
			
			`if(customElements){`
			`  NSString *likes =  customElements[@"likes"]`
			`NSString *downloads = customElements[@"downloads"]`
			`NSString *price = customElements[@"price"]`
			`NSString *saleprice = customElements[@"saleprice"]`
			`NSString *phone = customElements[@"phone"]`
			`NSString *address = customElements[@"address"];`
			`NSString *displayurl =  customElements[@"displayurl"]`
			`}`

Downloads	No	Yes (json only)	
Price	No	Yes (json only)	
Sale Price	No	Yes (json only)	
Phone	No	Yes (json only)	
Address	No	Yes (json only)	
Display URL	No	Yes (json only)	
Privacy URL	No	Yes	`response.privacyLink;`
Video	No	Yes	`response.vastXML;`
Custom	Yes	No	
Context	Yes	No	
Full text	Yes	No	

Related Topics

- Integrate with iOS
- Mediate with iOS