EC Ad iOS SDK Documentation

Online Guide: http://demo.engageclick.com/ecsdk

Introduction

The EngageClick iOS AdSDK provides an Objective C API that allows you to add a rich advertising experience to your iOS apps for iPhone, iPad and iPod touch devices.
The API is provided as a library file and a header file that goes with it.

Setting up your Xcode project

You need to use a minimum of iOS 5 to build your application with the EC AdSDK, so please set the Base SDK accordingly.

- The SDK depends on the following iOS frameworks and libraries, these need to be added to your Xcode project as references
    - SystemConfiguration.framework
    - CoreLocation.framework
    - CoreTelephone.framework
    - UIKit.framework
    - Foundation.framework
    - CoreGraphics.framework
    - AdSupport.framework
    - Storekit.framework
    - AudioToolBox.framework
    - QuartzCore.framework
    - MediaPlayer.framework
    - MapKit.framework
    - Social.framework
    - Accounts.framework
    - MessageUI.framework
    - EventKit.framework

    - CoreGraphics.framework

- Include the location of the ECAdManager.h in the **User Header Search Path** in the **Search Paths** section of **Build Settings** for the target of your project.
- In the **Build Phases** section for your target add libECAdLib.a to the **Link Binaries With Libraries** section
- In the **Build Phases** section for your target add the file ECAdLibResources.bundle to the **Copy Bundle Resources** section
- In **Build Settings** and **Linking** section add -all_load to the **Other Linker Flags** line

To Integrate Modal Ads:

Do the following to show a Modal Ad in your application

- Include the header file ECAdManager.h
- Place the following code in Appdelegate's - (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions

```
[[ECAdManager sharedManager] startSession:YOUR_PUB_KEY];
```

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    // Override point for customization after application launch.
    [[ECAdManager sharedManager]
        startSession:@"f3dd0f66ee055083e46978cdceac44c4fe55fbc52b558a62d66aa1de404099468f72d283cca5d14d4e2a44306e51784ea77918b07359737aa99b6fa77335f634"];

    return YES;
}
```

- Initialize the parameters that need to be passed into the EC AdSDK.
  Create a NSDictionary and set NSString values for the following keys
    - kECAdAppPublisherKey - You should have gotten this when you setup your EngageClick account, if not please contact your EngageClick representative
    - kECAdAppZoneIDKey - You should have gotten this when you setup your EngageClick account, if not please contact your EngageClick representative
    - kECAdAppIDKey - The AppID of the app that is showing the Ad
    - kECAdReferrerKey - A String that will indicate which action or screen shows the Ad
    - kECKeywordKey - A set of key words that make sense for the location that the Ad is being shown
    - kECCategoryKey - The category of your app e.g. game, lifestyle etc.

  Here is how the code for this step might look like

```
// initialize the params that need to be passed to the EC Ad SDK
```

```
self.adParams = [[NSMutableDictionary alloc] init];
[self.adParams setObject:@"<YOUR_PUB_KEY>" forKey:kECAdAppPublisherKey];
[self.adParams setObject:@"<YOUR_APPID>" forKey: kECAdAppIDKey];
[self.adParams setObject:@"LevelChangeViewController" forKey:kECAdReferrerKey];
[self.adParams setObject:@"level, change, powerup" forKey:kECKeywordKey];
[self.adParams setObject:@"games" forKey:kECCategoryKey];
```

```
self.adParams = [NSMutableDictionary dictionary];

/*
 Ad Sizes:
 EC_AD_320x50,
 EC_AD_300x50,
 EC_AD_300x250,
 EC_AD_320x480,
 EC_AD_480x320,
 EC_AD_468x60,
 EC_AD_120x600,
 EC_AD_728x90,
 EC_AD_728x1024,
 EC_AD_1024x728,
 EC_AD_768x1024,
 EC_AD_1024x768
 */

[self.adParams setObject:@"20b84a54-8fbe-11e3-9f9e-22000a1c450e" forKey:kECAdAppIDKey]; // Put in your App Id Here
[self.adParams setObject:[NSNumber numberWithInt:EC_AD_320x50] forKey:kECAdSize]; //  Put in your Ad Size here

// Provide user paramters below if any.
[self.adParams setObject:[NSMutableDictionary dictionaryWithObjectsAndKeys:@"male",@"Gender",@"26",@"Age", nil]
   forKey:kECAdUserParams];
```

- Make the call to show the Ad, get an instance of the ECAdManager and make the call to show the Ad pass in the ViewController that is making the call and the params that you have set above.
  Here is how the code for this call looks like

```
if (NO == [ECAdManager sharedManager] showECModalAdWithParameters:self.adParams
withViewController:self refresh:0.0])
{
   NSLog(@"Failed to show EngageClick Ad");
}
```

- Listen to the result of showing the Ad, register to listen to the notfication kECAdManagerDidShowAdNotification. When you get this notification it will tell you the result of showing the Ad. The user info for the notification will contain the key kECAdStatusKey, the values associated with this key can be one of
  - kECAdUserCompletedInteraction
  - kECAdUserClosedAd
  - kECAdAdTimedOut

based on this result you can decide what needs to happen next in your app.

To Integrate Banner Ads:

1) Create a Parameter dictionary as shown in the snippet below:

```
self.adParams = [NSMutableDictionary dictionary];

/*
 Ad Sizes:
 EC_AD_320x50,
 EC_AD_300x50,
 EC_AD_300x250,
 EC_AD_320x480,
 EC_AD_480x320,
 EC_AD_468x60,
 EC_AD_120x600,
 EC_AD_728x90,
 EC_AD_728x1024,
 EC_AD_1024x728,
 EC_AD_768x1024,
 EC_AD_1024x768
 */

[self.adParams setObject:@"20b84a54-8fbe-11e3-9f9e-22000a1c450e" forKey:kECAdAppIDKey]; // Put in your App Id Here
[self.adParams setObject:[NSNumber numberWithInt:EC_AD_320x50] forKey:kECAdSize]; //  Put in your Ad Size here

// Provide user paramters below if any.
[self.adParams setObject:[NSMutableDictionary dictionaryWithObjectsAndKeys:@"male",@"Gender",@"26",@"Age", nil]
    forKey:kECAdUserParams];
```

2) Call the following method to invoke the Banner Ad Container view. Once the view is received it can be placed at the bottom of the view initially or could be hidden until the Ad is loaded completely.

```
- (IBAction)showBannerAd:(id)sender {
   if (bannerView) {
      [bannerView removeFromSuperview];
      bannerView = nil;
   }
   /* Set th refresh Parameter to number of seconds the ad needs to refresh every time. Default is set to 0
   You can also use [[ECAdManager sharedManager] refreshBannerAd] to refresh the ad manually
   */
   ECBannerAdView *view =     [[ECAdManager sharedManager] showECBannerAdWithParameters:self.adParams withViewController:self.view Custom:NO refresh:0.0 delegate:self];

//   Uncomment this line to manually set the delegate to override "modalAdPresentingViewcontroller" method
//   [[ECAdManager sharedManager] setBannerAdDelegate:self];

   if (view) {
      if (iPad) {
         if (UIInterfaceOrientationIsLandscape(self.interfaceOrientation))
            [view layoutFrame:CGRectMake(0,self.view.frame.size.width,self.view.frame.size.height,90)];
         else
            [view layoutFrame:CGRectMake(0,self.view.frame.size.height,self.view.frame.size.width,90)];

      }
      else {
         if (UIInterfaceOrientationIsLandscape(self.interfaceOrientation))
            [view layoutFrame:CGRectMake(0,self.view.frame.size.width,self.view.frame.size.height,50)];
         else
            [view layoutFrame:CGRectMake(0,self.view.frame.size.height,self.view.frame.size.width,50)];

      }
      [view refreshAd];
      bannerView = view;
   }
}
```

3) Once the Ad is loaded the view could be animated as below. Use bannerAdViewDidLoad delegate to show up the ad on the Ad success.

```
- (void)bannerAdViewDidLoad:(ECBannerAdView *)bannerAdView {
   [UIView animateWithDuration:1.0 animations:^{
      if (iPad) {
         if (UIInterfaceOrientationIsLandscape(self.interfaceOrientation))
            [bannerAdView layoutFrame:CGRectMake(0,self.view.frame.size.width-90,self.view.frame.size.height,90)];
         else
            [bannerAdView layoutFrame:CGRectMake(0,self.view.frame.size.height-90,self.view.frame.size.width,90)];

      }
      else {
         if (UIInterfaceOrientationIsLandscape(self.interfaceOrientation))
            [bannerAdView layoutFrame:CGRectMake(0,self.view.frame.size.width-50,self.view.frame.size.height,50)];
         else
            [bannerAdView layoutFrame:CGRectMake(0,self.view.frame.size.height-50,self.view.frame.size.width,50)];

      }


      [bannerAdView setCloseButtonHidden:YES];
   } completion:^(BOOL finished) {
   }];
}
```

4) The following delegates are used in BannerAdView

```
- (void)bannerAdDidFinishWithResult:(kECBannerAdResult)result withServerResponse:(NSDictionary *)response;
- (void)bannerAdViewDidLoad:(ECBannerAdView *)bannerAdView;
- (void)bannerAdView:(ECBannerAdView *)bannerAdView didFailWithError:(NSError *)error;
- (void)bannerAdView:(ECBannerAdView *)bannerAdView didClickLink:(NSString *)urlStr;
- (void)bannerAdView:(ECBannerAdView *)bannerAdView willExpand:(NSString *)urlStr;
- (void)bannerAdViewDidClose:(ECBannerAdView *)bannerAdView;
- (void)bannerAdViewDidRestore:(ECBannerAdView *)bannerAdView;
- (UIViewController *)modalAdPresentingViewcontroller;
```

5) For Orientation change, the Ad need to be refreshed. Please see the following snippet.

```
- (void)willAnimateRotationToInterfaceOrientation:(UIInterfaceOrientation)toInterfaceOrientation duration:(NSTimeInterval)duration {
   // This is to notify the MRAID about the orientation Change.
   [bannerView rotateToOrientation:toInterfaceOrientation] ;

   // The below methods can be used to refresh the Ad on Orientation Change
   //    [bannerView layoutFrame:CGRectMake(0,self.view.frame.size.height - 125, self.view.frame.size.width, 120)];
   //    [bannerView  performSelector:@selector(refreshAd)];
}
```

To Integrate Video Ads:

1) Create a Parameter dictionary as done previously with the necessary Pub key and Zone id.
2) The video ad will be displayed as a modal view. So to trigger the video ad just call this method.
   `[[ECAdManager sharedManager] showVideoAd:self.adParams];`