admixer

iOS SDK User Guide

2018.02

목차

I. 개요

- 본 문서는 **AdMixer iOS SDK**를 프로젝트에 적용하기 위한 문서입니다.
- **AdMixer**는 광고 플랫폼을 통합하여 배너 광고 및 전면 광고를 표시하는 기능을 제공합니다.
- 적용 버전은 다음과 같으며, 각각의 라이브러리는 해당 사이트에서 최신 버전을 받아 라이브러리를 교체하여 사용하실 수 있습니다.
  제공되는 **AdMixer SDK**의 **light** 버전은 **Ad Network**사의 **SDK** 를 포함하고 있지 않으나,
  full버전은 **lipo**를 통해 **simulator**용과 **release**용을 합친 **Ad Network** 사의 **SDK**가 포함되어 있습니다.
  - Adfit (3.0.0)
  - ADMOB (7.28.0)
  - CAULY (3.1.0)
  - Syrub Ad (T-AD) (3.6.4.0)
  - ShallWeAd (2.4.7)
  - InMobi (7.0.4)
  - iAd
  - Facebook (4.27.2)
  - MEBA A-plus (MAN) (0.2_20170113)
- 먼저 **AdMixer** 개발자센터 **(http://admixer.co.kr)** 에 가입한 후, 개발중인 앱에 대한 정보를 추가해서 앱구분을 위한 **AxKey(**어플리케이션 키**)**를 발급 받으셔야 합니다.
- **AdMixer**는 적용할 광고 플랫폼을 **AdMixer** 사이트를 통해 손쉽게 변경하실 수 있으며 각 광고 별 노출 비율 및 세부 설정 또한 사이트를 통해 변경하실 수 있습니다.

II. SDK 구성 (1 / 3)

➢ **AdMixer iOS SDK는 다음과 같이 구성되어 있습니다.**

- Include : AdMixer iOS SDK를 control하기 위한 라이브러리및 해당 header 파일
  - libAdMixer.a (AdMixer 라이브러리, 디바이스 빌드 전용)
  - AdMixer.h
  - AdMixerAdAdapter.h
  - AdMixerInfo.h
  - AdMixerInterstitial.h
  - AdMixerInterstitialPopupOption.h
  - AdMixerView.h
  - AdMixerHalfAd.h
  - AdMixerHalfAdPopupOption.h
  - AdMixerCustomPopup.h
  - AXError.h
  - AXLog.h

II. SDK 구성 (2 / 3)

➢ **AdMixer iOS SDK는 다음과 같이 구성되어 있습니다.**

- libAdMixer_Combo.a
  - AdMixer 라이브러리로 디바이스/시뮬레이터 공용
  - libAdMixer.a 파일과 유사하나 시뮬레이터 빌드에서도 동작합니다.
  - 이 라이브러리를 사용하시려면 Include의 libAdMixer.a 파일 대신 이 파일을 프로젝트에 포함하시면 됩니다.
  - 이 라이브러리로 빌드 시에는 앱의 크기가 1~2MB 정도 커질 수 있습니다.

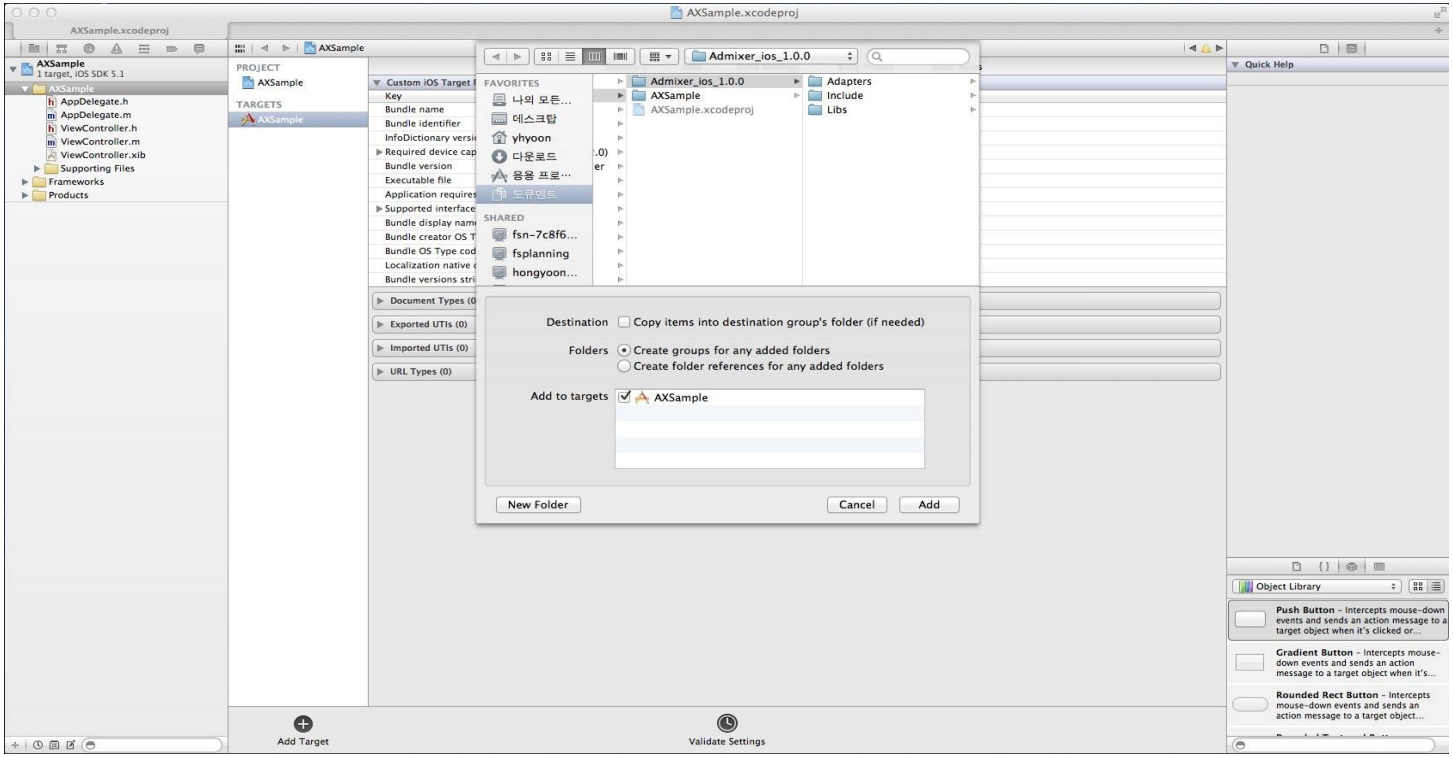
II. SDK 구성 (3 / 3)

➢ **AdMixer iOS SDK는 다음과 같이 구성되어 있습니다.**

- Adapters: 포함시키고자 하는 Ad Network의 adapter. 필요한 광고플랫폼을 선별하시어, 프로젝트에 포함시키시면 됩니다.
  - AdfitAdapter (Adfit 적용 시에는 프로젝트 설정의 other linker flags에 -ObjC 옵션을 추가하셔야 합니다. 또한, AdFitSDK.framework을 앱 프로젝트의 General > Embeded Binaries 항목으로 끌어서 놓습니다. Objective-C 기반 프로젝트의 경우, Always Embed Swift Standard Libraries항목을 Yes로 설정해주세요.)
  - AdmobAdapter (Build Options 섹션에서 Enable Bitcode를 Yes로 설정하시고, other linker flags에 -ObjC 옵션을 추가하셔야 합니다. 수동 버전관리를 위해 CocoaPods가 아닌 SDK 파일을 사용하시기 바랍니다.)
  - CaulyAdapter
  - ShallWeAd
  - TadAdapter
  - InmobiAdapter (Inmobi 적용 시에는 프로젝트 설정의 other linker flags에 –ObjC 옵션을 추가하셔야 하며, info.plist 파일에 NSLocationWhenInUseUsageDescription을 Yes로 설정하셔야 합니다.)
  - IAdAdapter
  - Facebook (Facebook 적용 시에는 프로젝트 설정의 Apple LLVM 6.0–Language–Modules안의 Enable Modules를 YES로 설정해주셔야 합니다.)
  - ManAdapter
- AdMixerSample: AdMixer iOS SDK 사용 예제

III. 프로젝트 설정

1. **Admixer_ios_x.y.z.zip**을 풉니다.
2. **Include** 폴더를 프로젝트에 추가합니다. **(AdMixer** 라이브러리 추가**)**
3. **Adapters** 폴더를 프로젝트에 추가합니다. **(Adapter** 추가**)**



III. 프로젝트 설정

4. iOS 9 업데이트에 따른 추가 설정

- 프로젝트 설정의 Build Settings에서 Enable Bitcode를 NO로 설정해야 합니다.

- **ATS(App Transport Security) 처리**

: iOS9부터 애플 보안 정책에 따라 암호화 되지 않은 HTTP 통신이 차단되어 광고에 영향을 줄 수 있습니다.

따라서, 프로젝트 내에 프로젝트명-info.plist 파일에 아래 내용을 추가해주시기 바랍니다.

1) NSAppTransportSecurity(Type: Dictionary) 항목을 생성

2) 하위에 NSAllowsArbitraryLoads(Type: Boolean) 아이템을 생성하여 YES로 설정합니다.

해당 설정을 통해 앱 내 모든 HTTP 통신을 허용할 수 있습니다. (For iOS 9 and later)

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

**** (다음 장을 반드시 확인해주세요) ****

III. 프로젝트 설정

5. 2017년 iOS 10 업데이트에 따른 ATS 추가 설정

- ATS(App Transport Security) 처리

: 2017년부터 ATS를 활성화하도록 보안이 강화되었습니다. 따라서, 기존 설정에 추가적인 옵션값이 필요합니다.

(단, Ad Network에 따라 HTTPS 지원현황에 차이가 있을 수 있으므로, 반드시 각 Ad Network 가이드 내 ATS 설정값을 확인바랍니다.)

1) 기존 설정대로 NSAllowsArbitraryLoads 아이템을 YES로 설정합니다. (For iOS 9 and later)

2) NSAllowsArbitraryLoadsForMedia(Type: Boolean) 아이템을 생성하여 YES로 설정합니다. (For iOS 10)

(현재는 Cauly 권장 설정값으로, AV Foundation 리소스에서만 HTTP를 허용하는 값이므로 Ad Network별 가이드 참고)

3) NSAllowsArbitraryLoadsInWebContent(Type: Boolean) 아이템을 생성하여 YES로 설정합니다.

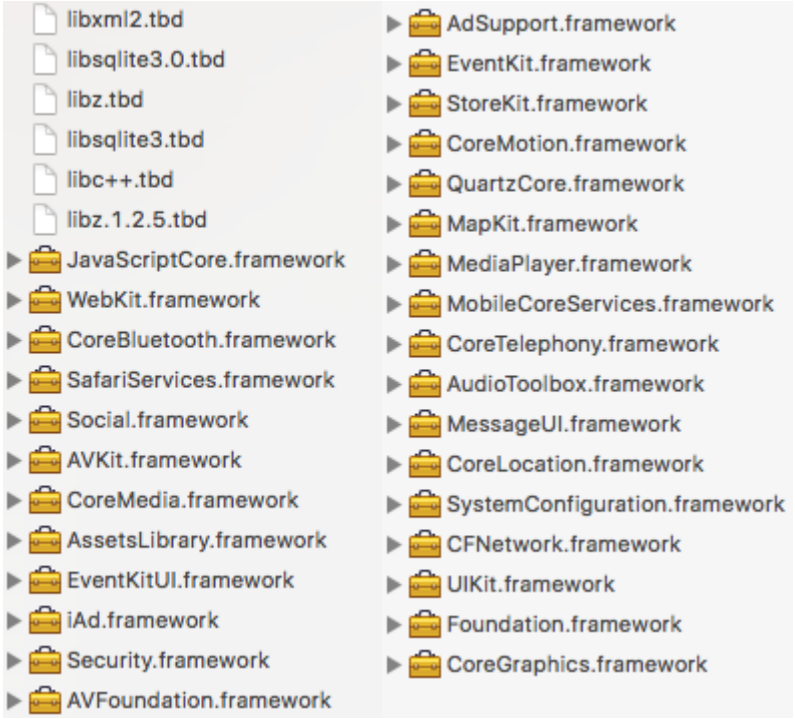
(고수익 배너 사용 시 권장하는 설정값이나, 웹뷰에서만 HTTP를 허용하는 값이므로 사용하시는 Ad Network에 따른 설정이 필요한 부분입니다.)

예) Syrub Ad 혹은 MEBA 사용 시 NSAllowsArbitraryLoads만 설정해야 합니다.

```
<key>NSAppTransportSecurity</key>
<dict>
    <key>NSAllowsArbitraryLoads</key> (해당 값만 설정 시, 앱 심사요청 과정에서 ATS 비활성 사유를 입력하셔야 합니다.)
    <true/>
    <key>NSAllowsArbitraryLoadsForMedia</key> (optional)
    <true/>
    <key>NSAllowsArbitraryLoadsInWebContent</key> (optional)
    <true/>
</dict>
```

IV. 광고플랫폼 별 라이브러리 Import

- 아래는 광고 플랫폼 별로 필요한 설정을 하나로 묶어 놓은 것입니다.
  다음의 **framework**이 포함되어 있는지 확인하시기 바랍니다.



- 자세한 내용은 해당 **Ad Network**사가 배포하는 기술문서를 참고하시기 바랍니다.

V. Banner 광고 추가 – 광고 뷰 추가

➢ **아래 코드는 Banner 광고를 추가한 예제입니다.**

- Banner광고를 위해서는 AdMixerViewDelegate 구현하셔야 합니다.
- Banner광고를 담는 AdMixerView가 필요합니다.

```
@interface ViewController : UIViewController<AdMixerViewDelegate, AdMixerInterstitialDelegate> {
    AdMixerView * _adView;
}
```

V. Banner 광고 추가 – 광고 뷰 추가

➢ **아래 코드는 Banner 광고를 추가한 예제입니다. (계속)**

- 필요한 adapter들을 등록합니다. appCode: 필드에 값을 넣으면, 웹에서 설정하신 configuration을 네트워크를 통해 받지 못하게 될 상황에서 2차적으로 사용하게 됩니다.
- 해당 필드를 nil로 넣으시면, configuration을 통해 얻은 값만 유효합니다.
- Inmobi의 경우, 애드믹서 전략 내의 앱코드에 각 배너광고의 placement id값으로 변경해주시기 바랍니다.
- Admixer의 information을 기록하는 객체(AdMixerInfo)를 생성하고, axKey필드를 발급받은 admixer키로 채웁니다.
- 배너 크기 요청 조절
  - AXBannerSize_Default – 기존 방식, 320*48 기준, Admob은 smart banner, Cauly는 iPhone/IPad 자동 구분
  - AXBannerSize_Iphone – 320*48
  - AXBannerSize_IPad_Small – 468*60 (현재는 admob, cauly, inmobi 만 지원)
  - AXBannerSize_IPad_Large – 728*90 (현재는 admob, cauly, inmobi 만 지원)
- 고수익 배너 높이 설정
  - AdMixerRTBBannerHeightRatio – 화면 높이의 10%를 배너광고의 높이로 설정 (기본 설정값)

```
- (void)viewDidLoad
{
        [super viewDidLoad];
계속...
```

V. Banner 광고 추가 – 광고 뷰 추가 (계속)

계속…

```
[AdMixer registerUserAdAdapterNameWithAppCode:AMA_ADFIT cls:[AdfitAdapter class] appCode:@"adfit_app_code"];
[AdMixer registerUserAdAdapterNameWithAppCode:AMA_ADMOB cls:[AdmobAdapter class] appCode:@"admob_app_code"];
[AdMixer registerUserAdAdapterNameWithAppCode:AMA_CAULY cls:[CaulyAdapter class] appCode:@"cauly_app_code"];
[AdMixer registerUserAdAdapterNameWithAppCode:AMA_TAD cls:[TAdAdapter class] appCode:@"tad_app_code"];
[AdMixer registerUserAdAdapterNameWithAppCode:AMA_INMOBI cls:[InmobiAdapter class] appCode:@”inmobi_app_code"];
[AdMixer registerUserAdAdapterNameWithAppCode:AMA_IAD cls:[IAdAdapter class] appCode:@”iad_app_code"];
[AdMixer registerUserAdAdapterNameWithAppCode:AMA_FACEBOOK
                                                cls:[FacebookAdapter class] appCode:@”facebook_app_code"];
[AdMixer registerUserAdAdapterNameWithAppCode:AMA_MAN cls:[ManAdapter class] appCode:@”man_app_code”];


// Inmobi  적용 시에는 Inmobi에서 발급한  account id값을 설정하셔야 합니다.
[IMSdk initWithAccountID:@”my_inmobi_account_id”];

// Admob 적용 시에는 Admob 에서 발급한 app id값을 설정하셔야 합니다.
[GADMobileAds configureWithApplicationID:@”my_admob_application_id”];


// Man 적용 시에 publisher id, media id, section id를 설정하셔야 합니다.
[ManAdapter setPublisherId:@”my_man_publisher_id”];
[ManAdapter setMediaId:@”my_man_media_id”];
[ManAdapter setBannerSectionId:@”my_man_banner_section_id”];
[ManAdapter setInterstitialSectionId:@”my_man_interstitial_section_id”];
```

V. Banner 광고 추가 – 광고 뷰 추가 (계속)

계속…

```
[AdMixer setLogLevel:AXLogLevelDebug]; // 로그 레벨 설정


AdMixerInfo * adInfo = [[[AdMixerInfo alloc] init] autorelease];
adInfo.axKey = @"my_admixer_key"; // AdMixer AxKey 값을 설정합니다.
adInfo.rtbVerticalAlign = AdMixerRTBVAlignCenter;
// 고수익 배너 상/하단 여백 처리 방식 지정(AdMixerRTBVAlignTop, AdMixerRTBVAlignCenter, AdMixerRTBVAlignBottom)
adInfo.defaultAdTime = 0; // 디폴트 광고 유지 시간(초단위로 지정한 시간 이후 광고 로딩)
adInfo.rtbBannerHeight = AdMixerRTBBannerHeightRatio; // 고수익 배너광고의 높이를 비율로 설정


_adView = [[AdMixerView alloc] initWithFrame:CGRectMake(0, 0, 320, 50)];
_adView.delegate = self;
_adView.adSize = AXBannerSize_Default; // 배너 크기 요청 조절
[self.view addSubview:_adView];


[_adView startWithAdInfo:adInfo baseViewController:self];

// 앱 백그라운드 진입 시 혹은 다른 뷰로 넘어갈 때 띠배너 객체를 해제하는 방법은 아래와 같습니다.
[_adView stop];
[_adView removeFromSuperview];
_adView = nil;

}
```

V. Banner 광고 추가 – 광고 뷰 추가 (계속)

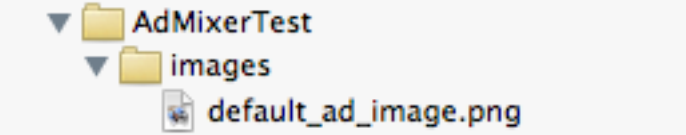
- **AdMixerViewDelegate를 통한 이벤트 수신**
  - AdMixerViewDelegate를 위한 method를 구현합니다.
  - onSucceededToReceiveAd는 광고를 잘 받아온 경우 호출됩니다.
  - onFailedToReceiveAd는 광고 받기에 실패한 경우 호출됩니다.
  - 현재 선택된 adapter 이름 확인 : [adView currentAdapterName];
  - 현재 선택된 adapter 크기 확인 : [adView currentAdapterSize];

```
#pragma mark - AdMixerViewDelegate

- (void)onSucceededToReceiveAd:(AdMixerView *)adView {
                NSLog(@"onSucceededToReceiveAd");
}

- (void)onFailedToReceiveAd:(AdMixerView *)adView error:(AXError *)error {
                NSLog(@"onFailedToReceiveAd : %@", error.errorMsg);
}
```

V. Banner 광고 추가 – Default Ad

➢ Default Ad에 대한 추가

- Default Ad 노출 상황
  - 앱의 시작시
  - 앱이 아무런 광고를 수신하지 못한 상태에서 네트워크 접속 오류가 발생하는 경우
- Custom Default Ad에 대한 지원
  1. 애드믹서 웹페이지에서 default ad를 설정 – 앱이 서버로부터 광고 노출에 대한 configuration을 받아오면서 activate
  2. 앱 프로젝트에 default_ad_image.png를 추가 – 앱이 서버로부터 광고 노출에 대한 configuration을 받지 못한 상태에서 activate
     - » 애드믹서는 애드믹서의 default_ad_image.png를 제공합니다. 해당 default_ad_image.png를 프로젝트에 추가하시면, 애드믹서의 default ad가 노출됩니다.
     - » AdMixerTest application project에 default_ad_image.png를 추가한 예제



V. Interstitial 광고 추가 – 광고 형태

➢ **전면광고에는 2가지 형태가 제공됩니다.**

- 일반 전면광고
- 팝업형 전면광고 (고수익 전면배너만 사용 가능합니다.)

일반 전면광고 예시

팝업형 전면광고 예시

V. Interstitial 광고 추가 – 광고 뷰 추가

- **아래 코드는 Interstitial광고를 추가한 예제입니다.**
  - Interstitial광고를 위해서는 AdMixerInterstitialDelegate 구현하셔야 합니다.

```
@interface ViewController : UIViewController<AdMixerViewDelegate, AdMixerInterstitialDelegate> {
    AdMixerView * _adView;
}
```

V. Interstitial 광고 추가 – 광고 뷰 추가

➢ **아래 코드는 Interstitial광고를 추가한 예제입니다. (계속)**

- Banner와 같이 adapter들을 등록합니다.
- 하우스광고 전면배너는 AdMixer iOS SDK 1.3.3 버전부터 제공됩니다.
- AdMixer의 information을 기록하는 객체(AdMixerInfo)를 생성하고, axKey필드를 발급받은 AdMixer키로 채웁니다.
- startWithAdInfo API를 호출 하시면 onSucceededToReceiveInterstitalAd 이벤트와 동시에 자동적으로 전면 광고가 표시됩니다.
- loadWithAdInfo API를 호출하시면 onSucceededToReceiveInterstitalAd 이벤트를 받아서 광고 로딩이 성공하면 원하는 시점에 displayAd API로 전면 광고를 노출하게 됩니다.
- loadWithAdInfo 지원 Ad Network : Admob, Cauly, Inmobi, T-AD, Facebook (다른 AdNetwork는 loadWithAdInfo 호출 시 로드하지 않습니다.)
- loadWithAdInfo API를 호출하고 일정시간이 지나면 광고가 노출이 되어도 유효노출로 처리되지 않는 경우가 있습니다. 이 경우는 애드네트워크별로 다르므로, 해당 애드네트워크사에 확인하여 loadWithAdInfo 호출 후 적절한 타임아웃을 걸어서 재호출 하는 방식으로 사용하시기 바랍니다. 애드네트워크별로 최소호출간격이 있는 경우도 있으므로, 적당한 타임아웃을 설정하시기 바랍니다. (ex. AdMob 광고노출인증유효시간 20분)

V. Interstitial 광고 추가 – 광고 뷰 추가

➢ 아래 코드는 **Interstitial광고를 추가한 예제입니다. (계속)**

```
- (IBAction)interstitialAdButtonAction:(id)sender {

        AdMixerInfo * adInfo = [[[AdMixerInfo alloc] init] autorelease];
        adInfo.axKey = @"my_admixer_key";
        // adInfo.interstitialTimeout = 5; // 전면 광고 요청 timeout 설정(Default값은 0으로 서버에서 주는 시간으로 처리)
        // adInfo.setBackgroundAlpha = YES; // 전면/하프광고 노출 시  광고 외 영역 반투명 처리(Default값은 NO)

        /* 팝업형 전면광고 설정코드 */
        AdMixerInterstitialPopupOption *adConfig = [[AdMixerInterstitialPopupOption alloc] init]; // 팝업형 전면광고 옵션설정
        [adConfig setButton:@"광고종료" withButtonColor:[UIColor  whiteColor]];
        // 광고를 닫는 기능의 버튼이 제공되며, 버튼문구와 버튼색상을 지정가능 (버튼문구 Default값은 '광고종료')
        [adConfig setButtonFrameColor:[UIColor blackColor]]; // 버튼프레임 색상을 지정가능
        [adInfo setInterstitialAdType:AdMixerInterstitialPopupType withInterstitialPopupOption:adConfig];
        // 팝업형 전면광고 option을 adInfo에 적용 (option nil로 설정시 Default option으로 제공됨)
        // AdMixerInterstitialBasicType : 일반(Default), AdMixerInterstitialPopupType : 팝업형

        AdMixerInterstitial * interstitial = [[AdMixerInterstitial alloc] init];
        interstitial.delegate = self;
        [interstitial startWithAdInfo:adInfo baseViewController:self];    // 광고 로딩 시 바로 표시
        // [interstitial loadWithAdInfo:adInfo baseViewController:self]; // 광고 로딩 후 원하는 시점에 displayAd를 호출해서 광고 표시
        [interstitial release];
}
```

V. Interstitial광고 추가 – 광고 뷰 추가

➢ **아래 코드는 Interstitial광고를 추가한 예제입니다. (계속)**

- AdMixerInterstitialDelegate를 위한 method를 구현합니다.
- onSucceededToReceiveInterstitialAd는 광고를 잘 받아온 경우 호출됩니다.
- onFailedToReceiveInterstitialAd는 광고 받기에 실패한 경우 호출됩니다.
- onClosedInterstitialAd는 전면 광고 창이 닫혔을 때에 호출됩니다.
- onDisplayedInterstitialAd는 전면 광고가 화면에 보여졌을 때에 호출됩니다. (선택사항)
- onClickedPopupButton은 팝업형 전면광고 버튼을 누르면 호출됩니다. (선택사항)

V. Interstitial광고 추가 – 광고 뷰 추가

➢ 아래 코드는 **Interstitial광고**를 추가한 예제입니다. **(계속)**

```
#pragma mark - AdMixerInterstitialDelegate

- (void)onSucceededToReceiveInterstitalAd:(AdMixerInterstitial *)intersitial {
        NSLog(@"onSucceededToReceiveInterstitalAd");
}

- (void)onFailedToReceiveInterstitialAd:(AdMixerInterstitial *)intersitial error:(AXError *)error {
        NSLog(@"onFailedToReceiveInterstitialAd : %@", error.errorMsg);
}

- (void)onClosedInterstitialAd:(AdMixerInterstitial *)intersitial {
        NSLog(@"onClosedInterstitialAd");
}

- (void)onDisplayedInterstitialAd:(AdMixerInterstitial *)intersitial {
        NSLog(@"onDisplayedInterstitialAd");
}

-(void)onClickedPopupButton:(AdMixerInterstitial *)interstitial {
        NSLog(@” onClickedPopupButton”);
}
```

V. Interstitial광고 닫기 (close)

➢ **아래 코드는 Interstitial광고 노출 후 원하는 시점에서 닫기 위해 close method를 구현한 예입니다.**

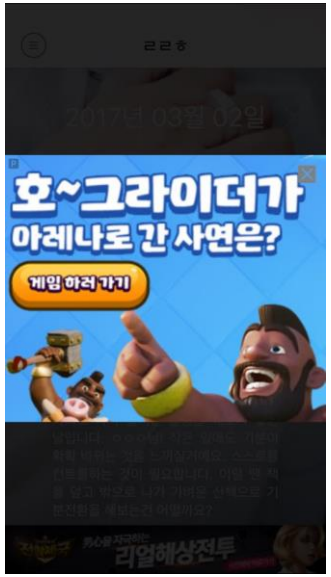
- 전면광고 close지원 Ad Network : 고수익전면배너, House Ad, T-AD
- 전면광고가 화면에 노출되기 전에 close를 호출하게 되면 동작하지 않습니다.
- 아래와 같이 타이머를 사용할 경우, 반드시 close delegate에서 타이머를 해제하여야 합니다. (타이머 작동 이전에 사용자가 전면광고를 끄게 되면, close method가 다음 전면광고에 영향을 줄 수 있습니다.)

```
AdMixerInterstitial *_interstitial ;
NSTimer *timer;
                                        ……

- (void)onDisplayedInterstitialAd:(AdMixerInterstitial *)intersitial {
                    NSLog(@"onDisplayedInterstitialAd");

                    timer = [NSTimer scheduledTimerWithTimeInterval:5 target:self
                                        selector:@selector(closeInterstiail) userInfo:nil repeats:NO];
                    // 전면광고 노출 후 5초 타이머 설정
}
-(void)onClosedInterstitialAd:(AdMixerInterstitial *)intersitital {
                     NSLog(@"onClosedInterstitialAd");
                    [timer invalidate]; // 타이머 해제
}
- (void)closeInterstitial {
                    If(_interstitial) {
                            [_interstitial close]; // 전면광고 닫기
                            _interstitial = nil;
                    }
}
```

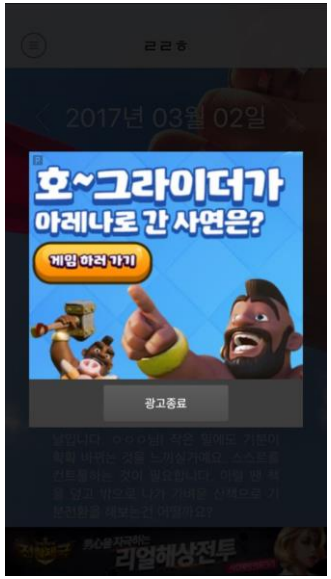
VI. Half 광고 추가 – 광고 형태

- 하프광고에는 2가지 형태가 제공됩니다.
  - 일반 하프광고
  - 팝업형 하프광고
- 하프광고는 고수익 및 고수익 NCPI만 제공합니다.

일반 하프광고 예시



팝업형 하프광고 예시



VI. Half 광고 추가 – 광고 뷰 추가

➢ **아래 코드는 Half광고를 추가한 예제입니다.**

- Half 광고를 위해서는 AdMixerHalfAdDelegate 구현하셔야 합니다.

```
@interface ViewController : UIViewController<AdMixerViewDelegate, AdMixerInterstitialDelegate, AdMixerHalfAdDelegate> {

    AdMixerView * _adView;
    AdMixerInterstitial * _interstitialAd;
    AdMixerHalfAd * _halfAd;
}
```

VI. Half 광고 추가 – 광고 뷰 추가

➢ **아래 코드는 Half광고를 추가한 예제입니다. (계속)**

- Banner와 같이 adapter들을 등록합니다.
- AdMixer의 information을 기록하는 객체(AdMixerInfo)를 생성하고, axKey필드를 발급받은 AdMixer키로 채웁니다.
- startWithAdInfo API를 호출 하시면 onSucceededToReceiveHalfAd 이벤트와 동시에 자동적으로 하프 광고가 표시됩니다.
- loadWithAdInfo API를 호출하시면 onSucceededToReceiveHalfAd 이벤트를 받아서 광고 로딩이 성공하면 원하는 시점에 displayAd API로 하프 광고를 노출하게 됩니다.

VI. Half 광고 추가 – 광고 뷰 추가

➢ 아래 코드는 **Half** 광고를 추가한 예제입니다. **(계속)**

```
- (IBAction)halfAdButtonAction:(id)sender {

        AdMixerInfo * adInfo = [[[AdMixerInfo alloc] init] autorelease];
        adInfo.axKey = @"my_admixer_key";
        // adInfo.halfTimeout = 5; // 하프 광고 요청 timeout 설정(Default값은 0으로 서버에서 주는 시간으로 처리)
        // adInfo.setBackgroundAlpha = YES; // 전면/하프광고 노출 시  광고 외 영역 반투명 처리(Default값은 NO)

        /* 팝업형 하프광고 설정코드 */
        AdMixerHalfAdPopupOption *adConfig = [[AdMixerHalfAdPopupOption alloc] init]; // 팝업형 하프광고 옵션설정
        [adConfig setButton:@"광고종료" withButtonColor:[UIColor  whiteColor]];
        // 광고를 닫는 기능의 버튼이 제공되며, 버튼문구와 버튼색상을 지정가능 (버튼문구 Default값은 '광고종료')
        [adConfig setButtonFrameColor:[UIColor blackColor]]; // 버튼프레임 색상을 지정가능
        [adInfo setHalfAdType:AdMixerHalfAdPopupType withHalfAdPopupOption:adConfig];
        // 팝업형 하프광고 option을 adInfo에 적용 (option nil로 설정시 Default option으로 제공됨)
        // AdMixerHalfAdBasicType : 일반(Default), AdMixerHalfAdPopupType : 팝업형

        AdMixerHalfAd * halfAd = [[AdMixerHalfAd alloc] init];
        halfAd.delegate = self;
        [halfAd startWithAdInfo:adInfo baseViewController:self];        // 광고 로딩 시 바로 표시
        // [halfAd loadWithAdInfo:adInfo baseViewController:self];     // 광고 로딩 후 원하는 시점에 displayAd를 호출해서 광고 표시
        [halfAd release];
}
```

VI. Half 광고 추가 – 광고 뷰 추가

➢ **아래 코드는 Half광고를 추가한 예제입니다. (계속)**

- AdMixerHalfAdDelegate를 위한 method를 구현합니다.
- onSucceededToReceiveHalfAd는 광고를 잘 받아온 경우 호출됩니다.
- onFailedToReceiveHalfAd는 광고 받기에 실패한 경우 호출됩니다.
- onClosedHalfAd는 하프 광고 창이 닫혔을 때에 호출됩니다.
- onDisplayedHalfAd는 하프 광고가 화면에 보여졌을 때에 호출됩니다. (선택사항)
- onClickedHalfAdPopupButton은 팝업형 하프광고 버튼을 누르면 호출됩니다. (선택사항)

VI. Half 광고 추가 – 광고 뷰 추가

➢ 아래 코드는 **Half**광고를 추가한 예제입니다. **(계속)**

```
#pragma mark - AdMixerHalfAdDelegate

- (void)onSucceededToReceiveHalfAd:(AdMixerHalfAd*)halfAd {
        NSLog(@"onSucceededToReceiveHalfAd");
}

- (void)onFailedToReceiveHalfAd:(AdMixerHalfAd *)halfAd error:(AXError *)error {
        NSLog(@"onFailedToReceiveHalfAd : %@", error.errorMsg);
}

- (void)onClosedHalfAd:(AdMixerHalfAd *)halfAd {
        NSLog(@"onClosedHalfAd");
}

- (void)onDisplayedHalfAd:(AdMixerHalfAd *)halfAd {
        NSLog(@"onDisplayedHalfAd");
}

-(void)onClickedHalfAdPopupButton:(AdMixerHalfAd *)halfAd {
        NSLog(@” onClickedHalfAdPopupButton”);
}
```

VI. Half 광고 닫기 (close)

➢ **아래 코드는 Half광고 노출 후 원하는 시점에서 닫기 위해 close method를 구현한 예입니다.**

- 하프광고가 화면에 노출되기 전에 close를 호출하게 되면 동작하지 않습니다.
- 아래와 같이 타이머를 사용할 경우, 반드시 close delegate에서 타이머를 해제하여야 합니다. (타이머 작동 이전에 사용자가 하프광고를 끄게 되면, close method가 다음 하프광고에 영향을 줄 수 있습니다.)

```
AdMixerHalfAd *_halfAd ;
NSTimer *timer;
                                        ……

- (void)onDisplayedHalfAd:(AdMixerHalfAd *)halfAd {
                    NSLog(@"onDisplayedHalfAd");

                    timer = [NSTimer scheduledTimerWithTimeInterval:5 target:self
                                        selector:@selector(closeHalfAd) userInfo:nil repeats:NO];
                    // 하프광고 노출 후 5초 타이머 설정
}
-(void)onClosedHalfAd:(AdMixerHalfAd *)halfAd {
                     NSLog(@"onClosedHalfAd");
                    [timer invalidate]; // 타이머 해제
}
- (void)closeHalfAd {
                    If(_halfAd) {
                            [_halfAd close]; // 하프광고 닫기
                            _halfAd= nil;
                    }
}
```

VII. Custom Popup 추가 – 시작/종료

➢ Custom Popup이란?

- Custom Popup은 IOS 앱에 공지사항, 업데이트 안내, 리뷰 유도 등의 다어얼로그를 손쉽게 추가할 수 있도록 다이얼로그 UI 및 서버 설정 로딩, 업데이트 및 리뷰 화면 연결 등의 기능을 제공합니다.

➢ Custom Popup 시작

- [[AdMixerCustomPopup sharedInstance] startCustomPopupWithAxKey:adInfo.axKey viewController:self];
- 위의 API를 이용해서 Custom Popup 모듈을 초기화합니다.
- 초기화가 완료되면 초기화 완료 이벤트가 발생하고 이 시점에 노출 지점을 “앱 시작 시”로 지정한 팝업이 노출됩니다.
- 초기화가 완료되기 전까지는 코드가 Custom Popup 노출 지점을 지나가더라도 팝업이 노출되지 않습니다.

➢ Custom Popup 종료

- [[AdMixerCustomPopup sharedInstance] stopCustomPopup];
- Custom Popup의 기능이 더 이상 필요하지 않은 경우 혹은 앱 종료 시에는 위의 API를 이용해서 Custom Popup 기능이 더 이상 동작하지 않도록 할 수 있습니다.

다음장에 계속…

VII. Custom Popup 추가 – 이벤트 수신

➢ Custom Popup 이벤트 수신

- [AdMixerCustomPopup sharedInstance].delegate = self;
- 위의 API를 이용해서 Custom Popup과 관련된 이벤트를 수신할 수 있습니다.
- listener는 AdMixerCustomPopupDelegate protocol을 구현해서 이벤트를 수신할 수 있습니다.

➢ Custom Popup 이벤트 종류

- onStartedCustomPopup : Custom Popup이 시작됨 (“앱 시작 시”로 지정된 팝업 노출)
- onWillShowCustomPopup:(NSString *)pageName : Popup이 열리기 직전
- onShowCustomPopup:(NSString *)pageName : Popup이 열린 직후
- onWillCloseCustomPopup:(NSString *)pageName : Popup이 닫히기 직전
- onCloseCustomPopup:(NSString *)pageName: Popup이 닫힌 직후
- onHasNoCustomPopup : 열릴 수 있는 팝업이 하나도 없는 경우

다음장에 계속…

VII. Custom Popup 추가

➢ Custom Popup 추가 방법

- Custom Popup을 표시 하기 위해서는 아래의 두 가지 작업이 필요합니다.
- **서버에서 팝업 추가**
  - AdMixer 사이트의 “앱/사이트 관리 > 앱/사이트 설정 > Custom 팝업”에서 Custom Popup을 추가하실 수 있습니다.
  - 각각의 Custom Popup은 클라이언트에서 지정한 노출 지점에서 지정된 조건에 따라 팝업됩니다.
  - “앱 실행 시”로 지정된 팝업은 클라이언트에서 별도의 노출 지점을 지정하지 않아도 Custom Popup이 시작되면(onStartedCustomPopup이 호출되면) 조건에 따라 팝업이 노출됩니다.
- **클라이언트에 노출 지점 추가**
  - [[AdMixerCustomPopup sharedInstance] checkCustomPopupPage:pageName viewController:self];
  - 위의 API를 이용해서 클라이언트에 Custom Popup 노출 지점을 추가합니다.
  - 노출 지점을 지정했다고 반드시 팝업이 노출되지는 않습니다. 서버에서 해당 노출 지점에 대한 팝업을 추가해야만 팝업이 노출됩니다.
  - 하나의 노출 지점에 여러개의 팝업을 추가한 경우 최대 3개까지 팝업이 노출됩니다

다음장에 계속…

VII. Custom Popup 추가

- **Custom Popup 적용 예**

```
- (void)viewDidLoad {
        [super viewDidLoad];
        [[AdMixerCustomPopup sharedInstance] startCustomPopupWithAxKey:adInfo.axKey viewController:self];
        [AdMixerCustomPopup sharedInstance].delegate = self;
}

- (void)viewDidUnload {
        [super viewDidUnload];
        [[AdMixerCustomPopup sharedInstance] stopCustomPopup];
}

-  (IBAction)customPopupButtonAction:(id)sender {
        [[AdMixerCustomPopup sharedInstance] checkCustomPopupPage:@"test" viewController:self];
}
```

VIII. 자주하는 질문 (1/2)

- **라이브러리 크기를 줄일 수 있습니까?**
  - 모든 광고를 적용하시는 편이 표시할 광고가 없는 상황을 줄일 수 있기 때문에 가급적이면 모든 광고를 포함하시는 편이 좋겠지만 프로그램 크기가 커져서 문제가 되신다면 꼭 필요한 광고 플랫폼을 결정하시고 불필요한 adapter를 삭제하시고, 필요한 광고플랫폼의 adapter만 등록하시면 됩니다.
- **전면 광고가 자주 나오지 않습니다.**
  - 전면 광고의 경우 배너 광고보다 광고 물량이 적어 설정하신 광고 플랫폼이 적은 경우 광고가 나오지 않을 확률도 그 만큼 커지게 됩니다. 가급적이면 모든 광고 플랫폼을 추가하시는 편이 좋습니다.
- **문제 확인을 위한 로그는 없나요?**
  - [AdMixer setLogLevel:AXLogLevelAll];
  - 위의 코드를 넣으시면 상세한 로그를 확인하실 수 있습니다. 문제 발생 시 해당 로그를 전달해 주시면 좀 더 정확한 원인 파악에 도움이 됩니다.
- **하나의 App에 복수개의 AdMixer_Key를 적용해도 되나요?**
  - AdMixer SDK는 한 App에 한 개의 AdMixer_Key를 적용하는 것을 고려하여 설계되었습니다. 복수개의 AdMixer_Key를 적용하실 경우 부작용이 있을 수 있습니다.

VIII. 자주하는 질문 (2/2)

- **시뮬레이터 Configuration에서는 빌드가 되지 않습니다.**
  - 기본으로 포함되어 있는 libAdMixer.a 라이브러리는 디바이스 전용 라이브러리입니다. 상위 폴더에 있는 libAdMixer_Combo.a 라이브러리로 교체하시면 됩니다.
- **전면 광고와 Custom Popup이 동시에 노출됩니다.**
  - Custom Popup의 초기화 시점을 전면 광고 노출 이후로 처리하시거나 Custom Popup 노출 이후에 전면 광고를 요청하시면 두 가지가 동시에 처리되는 것을 방지할 수 있습니다.
- **Custom Popup이 노출되지 않습니다.**
  - startCustomPopupWithAxKey:viewController:를 호출하시면 onStartedCustomPopup 이벤트가 호출된 이후부터 Custom Popup이 노출됩니다.
  - onStartedCustomPopup 이전에 checkCustomPopup을 호출하신 경우에는 Popup이 노출되지 않습니다.
  - 서버에서 설정한 버전 코드나 노출 횟수 등에 따라 Popup이 노출되지 않을 수 있습니다.
- **ARC(Automatic Reference Counting) 설정이 되어 있지 않은 것 같습니다.**
  - ARC는 빌드 시에 소스파일(.m)에 적용하는 옵션으로 라이브러리 자체에는 ARC 적용에 따른 차이는 없습니다. ARC가 적용된 프로젝트 혹은 소스에서는 AdMixer 관련 Class 사용 시 retain, release를 호출하실 필요가 없고, ARC가 적용되지 않은 소스에서는 반드시 retain, release를 적절하게 호출해 주셔야 합니다. 다만, ARC 프로젝트 내에서 각 adapter(.m)파일을 import하여 사용하실 때는 build phases에서 compile sources에 있는 adapter(.m)파일마다 -fno-objc-arc 옵션값을 설정하시기 바랍니다.

The End

admixer

http://www.admixer.co.kr