UnityPlugin for iOS
User Guide 1.2

개요

애드립 Unity 플러그인 적용을 위한 문서입니다.

애드립 Unity 플러그인을 사용하기 위해서 다음이 필요합니다.

- **애드립 API 키**
- **애드립 최신 SDK**
- 각 **광고 플랫폼** 사이트에서 발급받은 **APP - ID** 및 **최신 SDK**
- 애드립에서 지원하는 각 **광고 플랫폼 SubAdView** 파일들

이 애드립 Unity 플러그인은 Unity 4.3.3 에서 작성 되었으며, 이전 버전에 대해서는 테스트가 이루어 지지 않았습니다.

오류 보고 또는 추가 문의사항은
http://adlibr.com 의 개발자 지원 메뉴 또는 adlib@mocoplex.com
에 남겨주시기 바랍니다.

변경이력

◆ V 1.2 (애드립 SDK 업데이트)
- Adlib.framework, 모든 SubAdView 파일 업데이트 필요

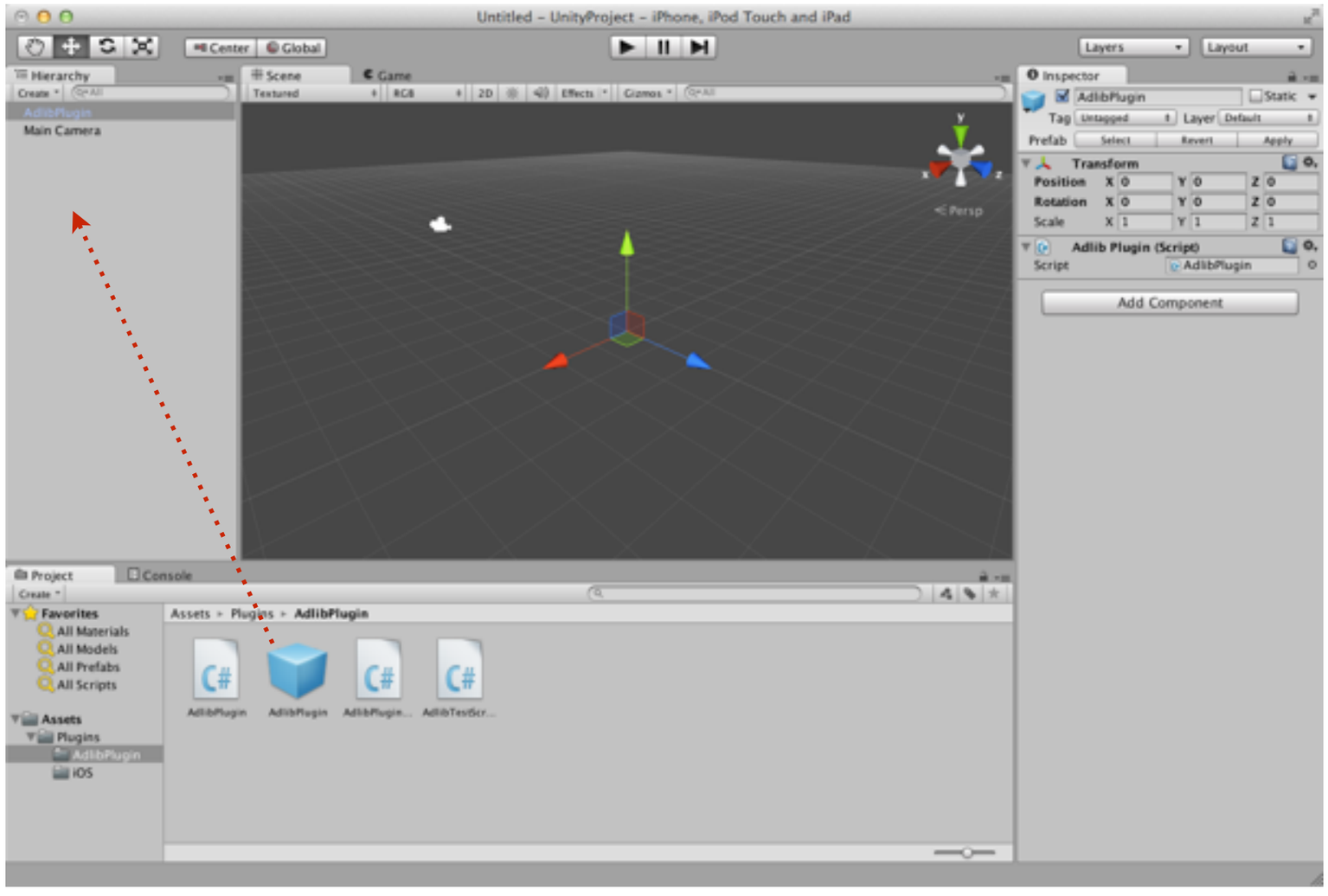
◆ V 1.1 (mediba ad 추가)
- AdlibPlugin.mm 파일 수정

플러그인 import

1. Assets > Import Package > Custom Package 메뉴를 클릭합니다.

2. AdlibUnityPlugin.unitypackage 파일을 선택하고, 모든 항목을 Import 합니다.

3. Assets/Plugins/AdlibPlugin/ 폴더에 있는 AdlibPlugin prefab 을 Unity scene으로 드래그 합니다.



플러그인 적용 #1

Assets/Plugins/iOS/AdlibPlugin.mm 을 열어 사용하지 않을 광고 플랫폼에 대한 import 선언과 setPlatform 호출을 제거합니다.

```
#import "SubAdlibAdViewAdam.h"
#import "SubAdlibAdViewAdmob.h"
#import "SubAdlibAdViewCauly.h"
#import "SubAdlibAdViewTAD.h"
#import "SubAdlibAdViewiAd.h"
#import "SubAdlibAdViewNaverAdPost.h"
#import "SubAdlibAdViewShallWeAd.h"
#import "SubAdlibAdViewInmobi.h"
#import "SubAdlibAdViewDomob.h"
#import "SubAdlibAdViewAdHub.h"
#import “SubAdlibAdViewUPlusAD.h”
#import "SubAdlibAdViewMedibaAd.h"
#import "SubAdlibAdViewMMedia.h"
#import <MillennialMedia/MMSDK.h>


- (void)initializeWithAds:(NSString*) adlibKey {

    self.viewController = UnityGetGLViewController();

    [[AdlibManager sharedSingletonClass] initAdlib:adlibKey];

    // 쓰지 않을 광고 플랫폼은 삭제해주세요.
    [[AdlibManager sharedSingletonClass] setPlatform:@"ADAM" withClass:[SubAdlibAdViewAdam class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"ADMOB" withClass:[SubAdlibAdViewAdmob class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"CAULY" withClass:[SubAdlibAdViewCauly class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"TAD" withClass:[SubAdlibAdViewTAD class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"IAD" withClass:[SubAdlibAdViewiAd class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"NAVER" withClass:[SubAdlibAdViewNaverAdPost class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"SHALLWEAD" withClass:[SubAdlibAdViewShallWeAd class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"INMOBI" withClass:[SubAdlibAdViewInmobi class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"DOMOB" withClass:[SubAdlibAdViewDomob class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"ADHUB" withClass:[SubAdlibAdViewAdHub class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"UPLUSAD" withClass:[SubAdlibAdViewUPlusAD class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"MEDIBAAD" withClass:[SubAdlibAdViewMedibaAd class]];
    [[AdlibManager sharedSingletonClass] setPlatform:@"MMEDIA" withClass:[SubAdlibAdViewMMedia class]];
    [MMSDK initialize];  // MillennialMedia v5.2.0 이상을 사용하시려면 반드시 초기화를 호출해 주세요.
}
```

플러그인 적용 #2

애드립 Unity 플러그인에는 애드립 호출을 위한 샘플 스크립트가 포함되어 있습니다.

Assets/Plugins/AdlibPlugin/AdlibTestScript.cs 에 ADLIB_API_KEY 를 입력합니다.

```
AdlibPlugin.InitializeAdlib("ADLIB_API_KEY");
```

AdlibTestScript.cs 를 Main Camera 에 컴포넌트로 추가를 하시면 애드립이 동작하는 것을 확인 할 수 있습니다.

iOS 프로젝트 빌드

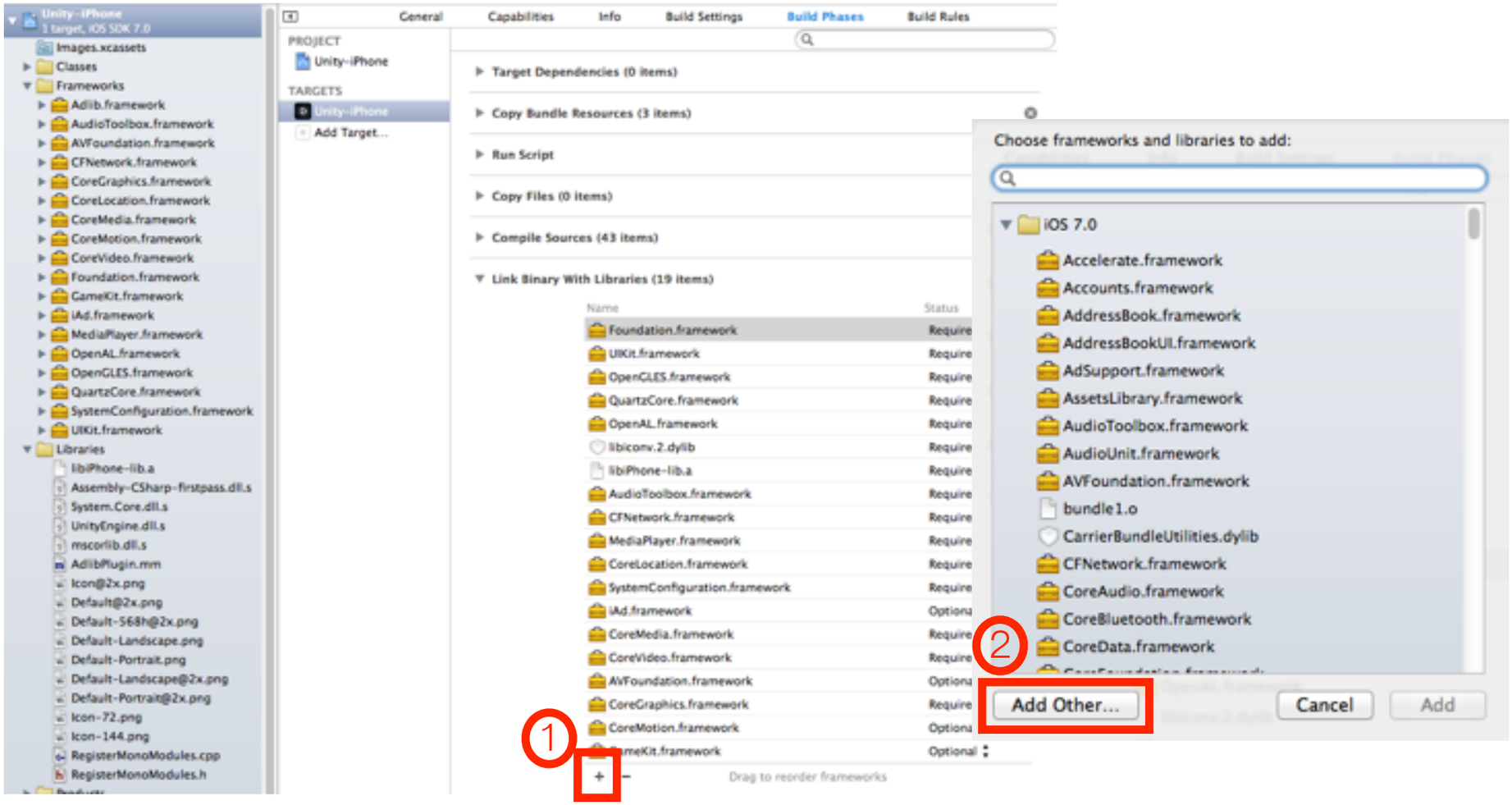
1. File > Build Settings 메뉴에서 iOS 플랫폼을 선택하여 “Switch Platform” 을 클릭 합니다.

2. “Player Settings” 를 선택하여 “Target iOS Version”을 4.3 이상으로 설정합니다.

3. “Build”를 클릭하면, Xcode 프로젝트가 생성됩니다.

iOS 프로젝트 설정 #1

ADLIB 폴더를 프로젝트 폴더에 복사한 후 아래와 같이 Adlib.framework를 추가합니다.



iOS 프로젝트 설정 #2

애드립 사용을 위한 필수 framework들을 추가합니다.

- UIKit.framework
- Foundation.framework
- CoreGraphics.framework
- CoreLocation.framework
- CoreTelephony.framework
- QuartzCore.framework
- AdSupport.framework
- libz.1.2.5.dylib

iOS 프로젝트 설정 #3

- ADLIB/SubAdView 폴더에서 사용할 광고 플랫폼의 SubAdView 파일들을 프로젝트에 추가합니다.
- 각각의 SubAdView 파일에 각 광고플랫폼 사이트에서 발급받은 앱키를 입력 합니다.
- 사용할 광고 플랫폼들의 SDK 및 필요한 framework를 추가하고, linker flags 설정을 수정합니다.
- 광고 플랫폼별 설정에 대한 자세한 사항은 각 플랫폼의 SDK 문서를 참조해 주세요.

애드립 Unity 플러그인 API

InitializeAdlib(string adlibKey)

애드립 홈페이지에서 발급받은 ADLIB API KEY를 setting하는 메소드로, 최초 한번 호출합니다.

메소드 호출 예)

```
AdlibPlugin.InitializeAdlib("ADLIB_API_KEY");
```

ShowBanner(bool positionAtTop, bool useHouseBanner)

띠배너를 화면에 노출시킵니다.

positionAtTop	true 이면 화면 상단, false이면 화면 하단에 위치.
useHouseBanner	true 이면 초기 광고 로딩까지의 공백에 하우스배너 노출, false 이면 하우스배너 노출하지 않음.

메소드 호출 예)

```
AdlibPlugin.ShowBanner(false, true);
```

HideBanner()

화면에 노출된 띠배너를 숨깁니다.

메소드 호출 예)

```
AdlibPlugin.HideBanner();
```

애드립 Unity 플러그인 API

ShowPopBanner(string frameColor, PopButton btnColor, bool useInAnim, bool useOutAnim, PopAlign btnAlign, int padding)

팝배너를 화면에 노출시킵니다.

frameColor	popBanner의 프레임 컬러 설정. “AARRGGBB”.
btnColor	popBanner의 Close 버튼 배경색 설정. PopButton.White (흰색 배경의 버튼), PopButton.Black (검정색 배경의 버튼).
useInAnim	true 이면 배너가 나타날 때 슬라이드 애니메이션 적용, false이면 애니메이션 없이 배너가 나타남.
useOutAnim	true 이면 배너가 사라질 때 슬라이드 애니메이션 적용, false이면 애니메이션 없이 배너가 사라짐.
btnAlign	팝배너 노출 및 애니메이션 적용 기준 position 설정. PopAlign.Left (좌측 기준 정렬), PopAlign.Top (상단 기준 정렬), PopAlign.Right (우측 기준 정렬), PopAlign.Bottom (하단 기준 정렬).
padding	정렬 기준으로 부터 padding(px) 만큼 떨어진 곳에 위치.

메소드 호출 예)

```
AdlibPlugin.ShowPopBanner(“ff444444”,
                          AdlibPlugin.PopButton.White,
                          true,
                          false,
                          AdlibPlugin.PopAlign.Bottom,
                          60);
```

애드립 Unity 플러그인 API

HidePopBanner()

화면에 노출된 팝배너를 닫습니다.

메소드 호출 예)

```
AdlibPlugin.HidePopBanner();
```

LoadInterstitialAd()

전면배너를 호출합니다.

위 메소드는 1차적으로 애드립의 전면배너를 호출합니다.
애드립의 전면배너 수신이 실패했을 경우, 대쉬보드의 전면배너 스케줄 설정된 순서대로 전면배너 수신이 성공할 때 까지,
타 플랫폼의 전면배너를 호출합니다.

메소드 호출 예)

```
AdlibPlugin.LoadInterstitialAd();
```

애드립 Unity 플러그인 API

ShowRewardLink(string linkId, int xPos, int yPos)

리워드링크 아이콘을 화면에 노출시킵니다.
여러번 호출하여 여러개의 아이콘을 노출할 수 있습니다.

linkId	애드립 홈페이지에서 등록하고 발급받은 리워드링크 ID.
xPos	아이콘 위치 x point.
yPos	아이콘 위치 y point.

메소드 호출 예)

```
AdlibPlugin.ShowRewardLink("리워드링크ID",
                           50,
                           60);
```

HideRewardLink()

화면에 노출된 모든 리워드링크 아이콘을 숨깁니다.

메소드 호출 예)

```
AdlibPlugin.HideRewardLink();
```

애드립 Unity 플러그인 API

애드립 Unity 플러그인은 다음의 리스너 이벤트를 제공합니다.

```
public static event Action<string> ReceivedInterstitial;  // 전면배너 수신 성공 – string은 수신된 플랫폼 이름
public static event Action FailedToReceiveInterstitial;   // 전면배너 수신 실패
public static event Action DismissedInterstitial;         // 전면배너 닫힘
public static event Action ReceivedPopBanner;             // 팝배너 수신 성공
public static event Action FailedToReceivePopBanner;      // 팝배너 수신 실패
public static event Action DismissedPopBanner;            // 팝배너 닫힘
```

+= 연산자를 이용해서 아래와 같이 이벤트를 등록할 수 있습니다.

```
// 리스너 응답을 받을 메소드명을 이벤트에 등록
AdlibPlugin.ReceivedInterstitial += HandleReceivedInterstitial;

// 리스너 응답후 호출부 구현
public void HandleReceivedInterstitial(string platform) {
          print("Received Interstitial Ad : ");
          print(platform);
     }
```