



# UnityPlugin for iOS

## User Guide 1.3

# 개요

애드립 Unity 플러그인 적용을 위한 문서입니다.

애드립 Unity 플러그인을 사용하기 위해서 다음이 필요합니다.

- 애드립 API 키
- 애드립 최신 SDK v.4.1.1.0 이상
- 각 광고 플랫폼 사이트에서 발급받은 APP - ID 및 최신 SDK
- 애드립에서 지원하는 각 광고 플랫폼 SubAdView 파일들

이 애드립 Unity 플러그인은 Unity 5.1 에서 작성 되었으며, 이전 버전에 대해서는 테스트가 이루어 지지 않았습니다.

오류 보고 또는 추가 문의사항은

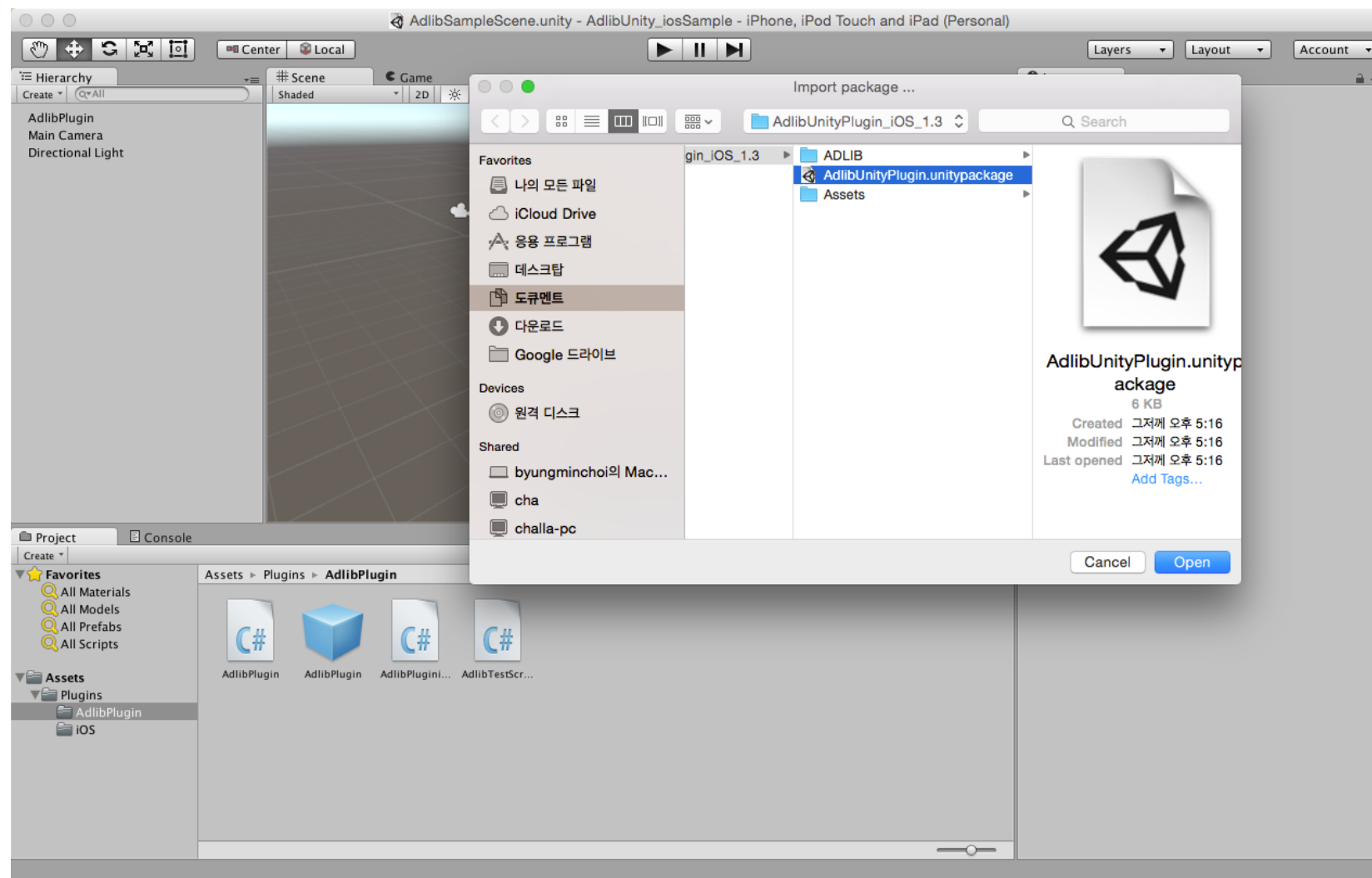
<http://adlibr.com> 의 개발자 지원 메뉴 또는 [adlib@mocoplex.com](mailto:adlib@mocoplex.com)에 남겨주시기 바랍니다.

# 변경이력

- ◆ V 1.3 (애드립 SDK 업데이트)
  - Adlib.framework v.4.1.1.0 업데이트 필요
  - 모든 SubAdView 파일 업데이트 필요
- ◆ 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.



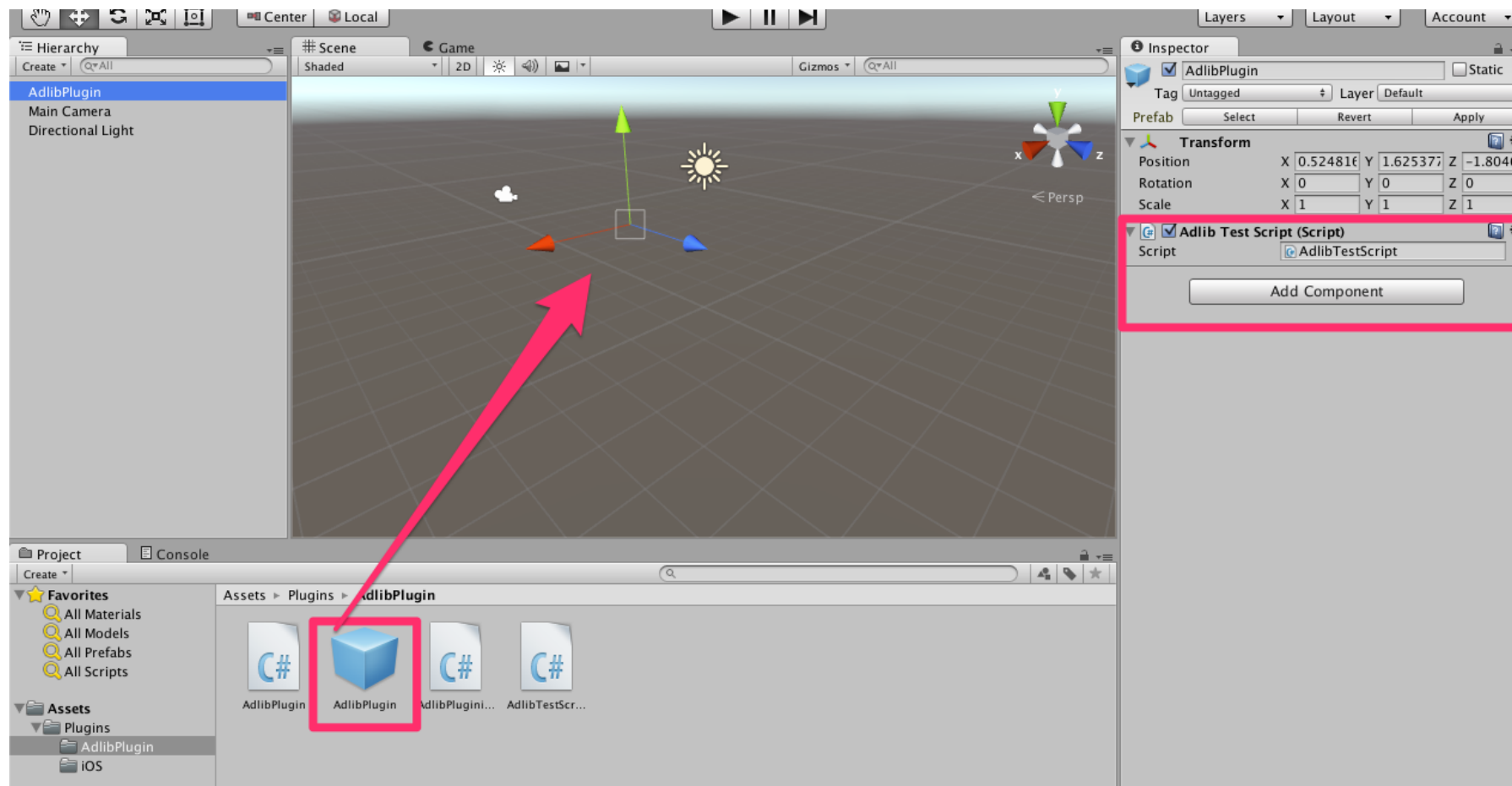
# 플러그인 적용 #1

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.



# 플러그인 적용 #2

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 "SubAdlibAdViewShallWeAd.h"
#import "SubAdlibAdViewInmobi.h"
#import "SubAdlibAdViewDomob.h"
#import "SubAdlibAdViewAdHub.h"
#import "SubAdlibAdViewMedibaAd.h"
#import "SubAdlibAdViewMMedia.h"
#import <MillennialMedia/MMSDK.h>

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

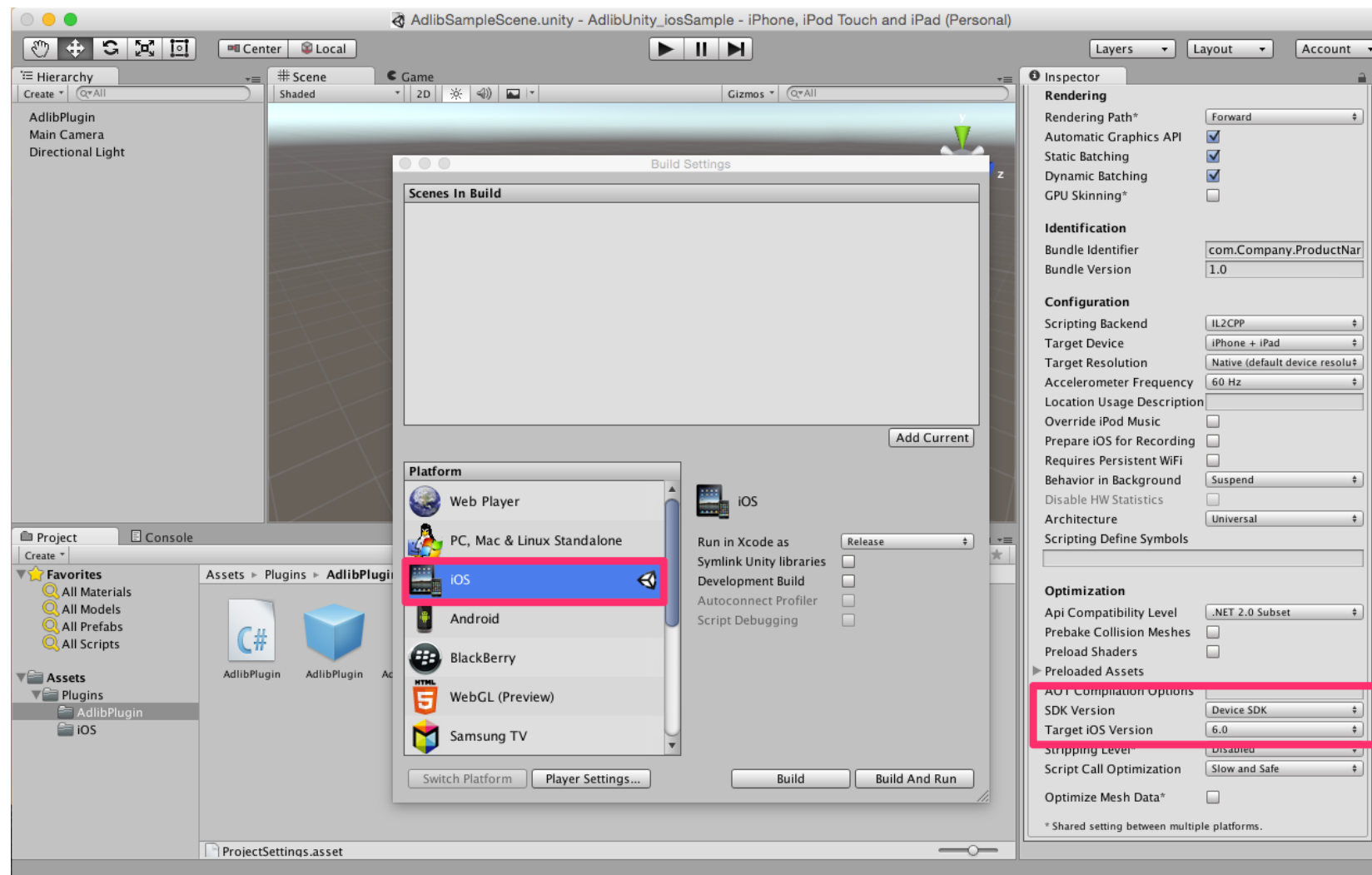
    self.viewController = UnityGetGLViewController();

    BOOL isTestMode = YES;
    AdlibMannager *manager = [AdlibManager sharedInstance];
    if (isTestMode) {
        [manager testModeLinkWithAdlibKey:adlibKey];
    } else {
        [manager linkWithAdlibKey:adlibKey];
    }

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

# iOS 프로젝트 빌드

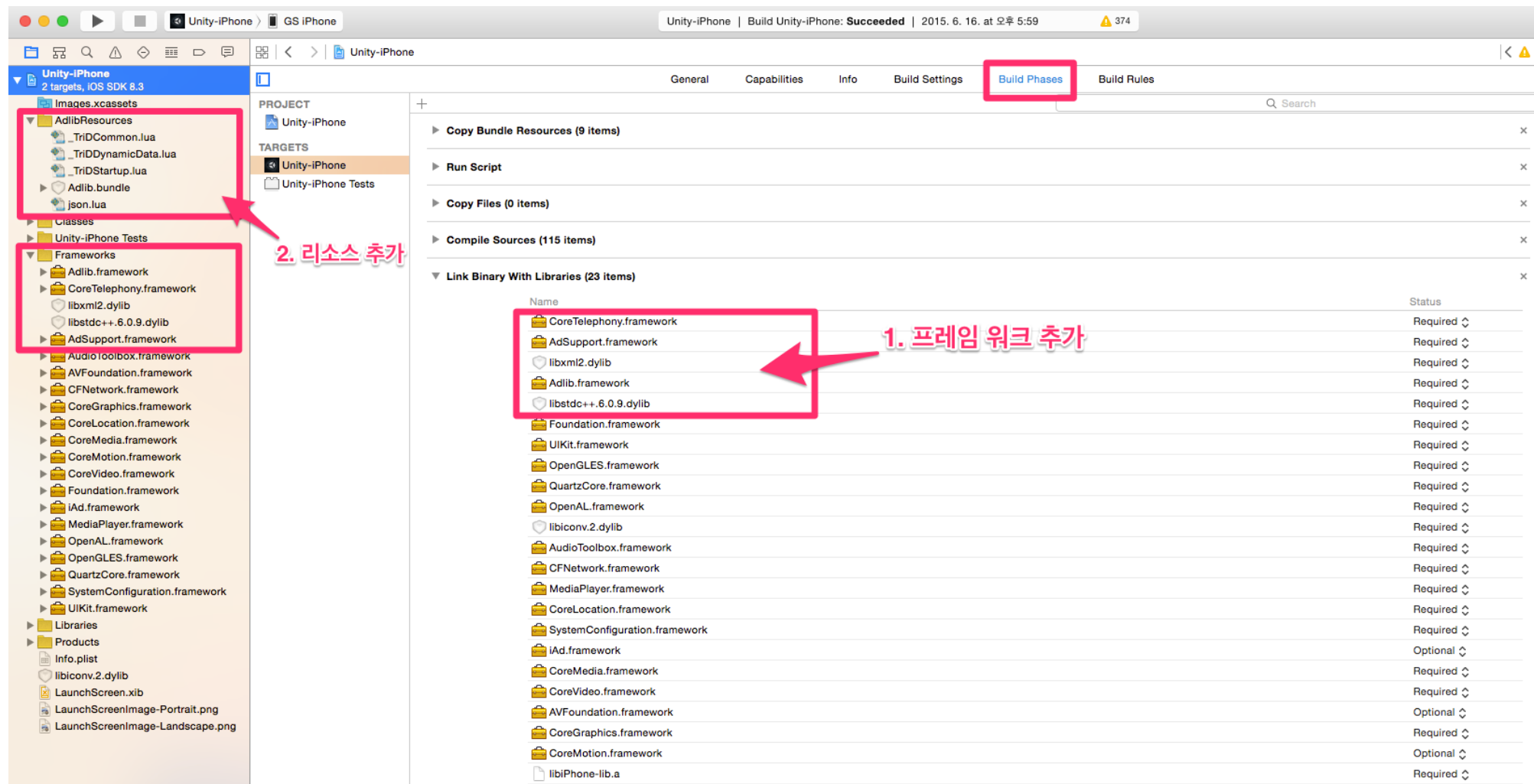
1. File > Build Settings 메뉴에서 iOS 플랫폼을 선택하여 “Switch Platform” 을 클릭 합니다.
2. “Player Settings” 를 선택하여 “Target iOS Version”을 6.0 이상으로 설정합니다.
3. “Build”를 클릭하면, Xcode 프로젝트가 생성됩니다.



# iOS 프로젝트 설정 #1

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

1. 최신 SDK에 필요한 추가 프레임워크 및 라이브러리 항목을 아래 항목과 같이 추가합니다.
2. 애드립 광고에 필요한 리소스 파일들을 포함하는 AdlibResources 폴더를 프로젝트에 포함합니다.





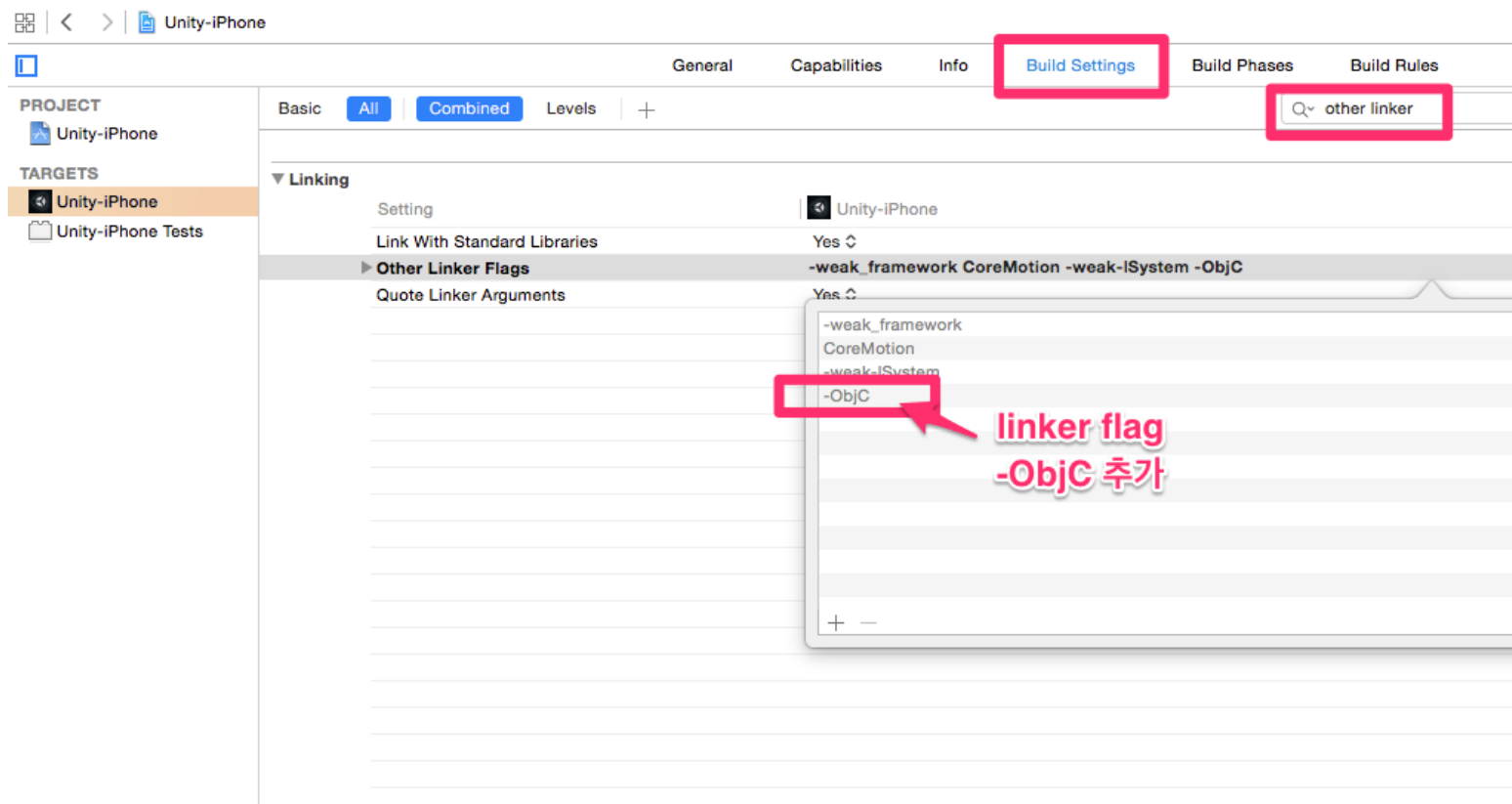
# iOS 프로젝트 설정 #2

애드립 사용을 위한 필수 framework들을 추가합니다.

- CoreTelephony.framework
- AdSupport.framework
- libxml2.dylib
- libstdc++.dylib
- OpenGLES.framework
- CoreMedia.framework

# iOS 프로젝트 설정 #3

- ADLIB/SubAdView 폴더에서 사용할 광고 플랫폼의 SubAdView 파일들을 프로젝트에 추가합니다.
- 각각의 SubAdView 파일에 각 광고플랫폼 사이트에서 발급받은 앱을 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)

띠배너를 화면에 노출시킵니다.

positionAtTop                      true 이면 화면 상단, false이면 화면 하단에 위치.

메소드 호출 예)

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

## **HideBanner**()

화면에 노출된 띠배너를 숨깁니다.

메소드 호출 예)

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

# 애드립 Unity 플러그인 API

## 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

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
```

`+=` 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);
}
```