Unity3D 集成指南

1. 概述

本文档是介绍 SDK unity3d版本的集成方法，从申请账号、appid、广告位id及集成SDK进行描述，确保开发者能够顺利集成 SDK进行变现。目前TopOn SDK的Unity 3D平台支持的广告形式如下：

广告形式	说明
Native	原生广告，无UI
Video	视频广告，带有UI
Interstitial	插屏广告，带有UI
Banner	横幅广告，带有UI

开发者可以根据应用的形态选择合适的广告形式，具体集成方法见下面具体介绍。

2. 账号及相关ID准备

你可以可通过TopOn账号及相关ID准备进行账号注册及登录的操作指引

3. SDK初始化

3.1 将TopOnSDK添加至您的项目

AnyThinkSDK unity 版本 你可从TopOn商务或者运营处获取，TopOnSDK包含的文件说明：

文件名	描述	是否必须
anythinkunity3d.unitypackage	**AnyThinkSDK** Unity3d 插件包,你可以导入此包到unity3d项目中,进行项目集成	是

注: 请使用Unity3D 2018或以上版本

3.1.1 Android导入说明（针对路径：/Assets/Plugins/Android 的文件说明）

1.libs插件的说明

路径	描述	是否必须
./libs/aars_anythink	TopOn的基础SDK包	是
./libs/aars_china_network	中国区聚合的第三方SDK文件夹	否
./libs/aars/aar_international_network	非中国区绝壑的第三方SDK文件夹	否
./libs/aars/aar_v4	Android Support v4+v7的SDK包（如果源项目已经有引入，则可将该文件剔除，不加入打包）	是

其中中国区的SDK文件夹和非中国区的SDK文件夹只能选择其中一个进行打包（根据开发者应用的流量来选择接入）

（1）./libs/aars_china_network文件夹说明：

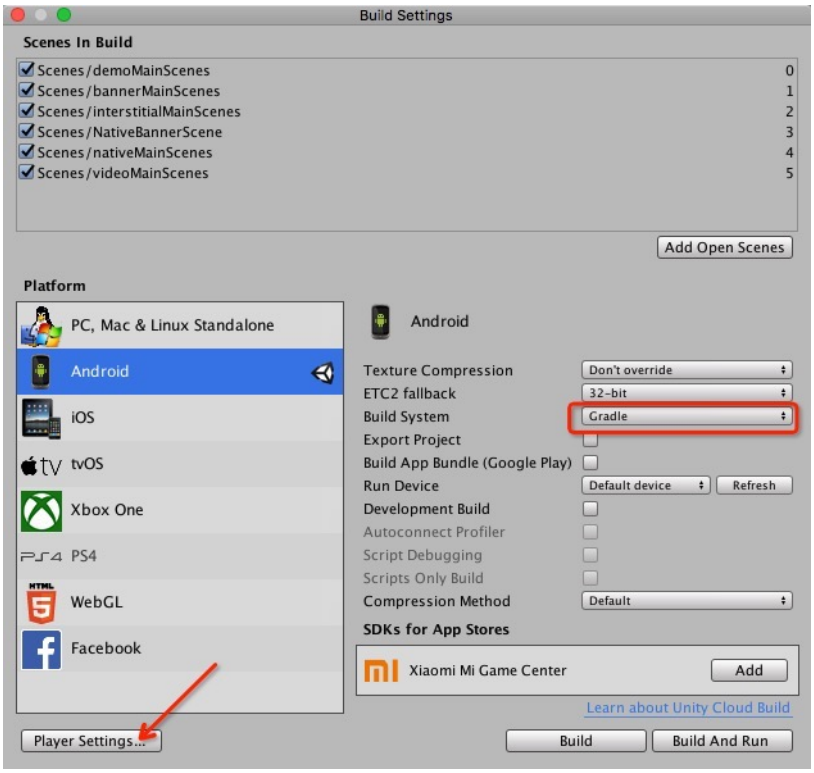
路径	描述	是否必须
./aars_china_network/aars_plugin	中国区聚合第三方SDK必须的插件包（如果源项目已经有引入，则可将该文件剔除，不加入打包）	是
./aars_china_network/aar_toutiao	**穿山甲**SDK文件夹	否
./aars_china_network/aar_baidu	**百度**SDK文件夹	否
./aars_china_network/aar_gdt	**广点通**SDK文件夹	否
./aars_china_network/aar_ks	**快手**SDK文件夹	否
./aars_china_network/aar_ksyun	**金山云**SDK文件夹	否
./aars_china_network/aar_luomi	**洛米**SDK文件夹	否
./aars_china_network/aar_mintegral_china	**Mintegral（中国区）**SDK文件夹	否
./aars_china_network/aar_oneway	**Oneway**SDK文件夹	否
./aars_china_network/aar_sigmob	**Sigmob**SDK文件夹	否
./aars_china_network/aar_uniplay	**玩转互联**SDK文件夹	否

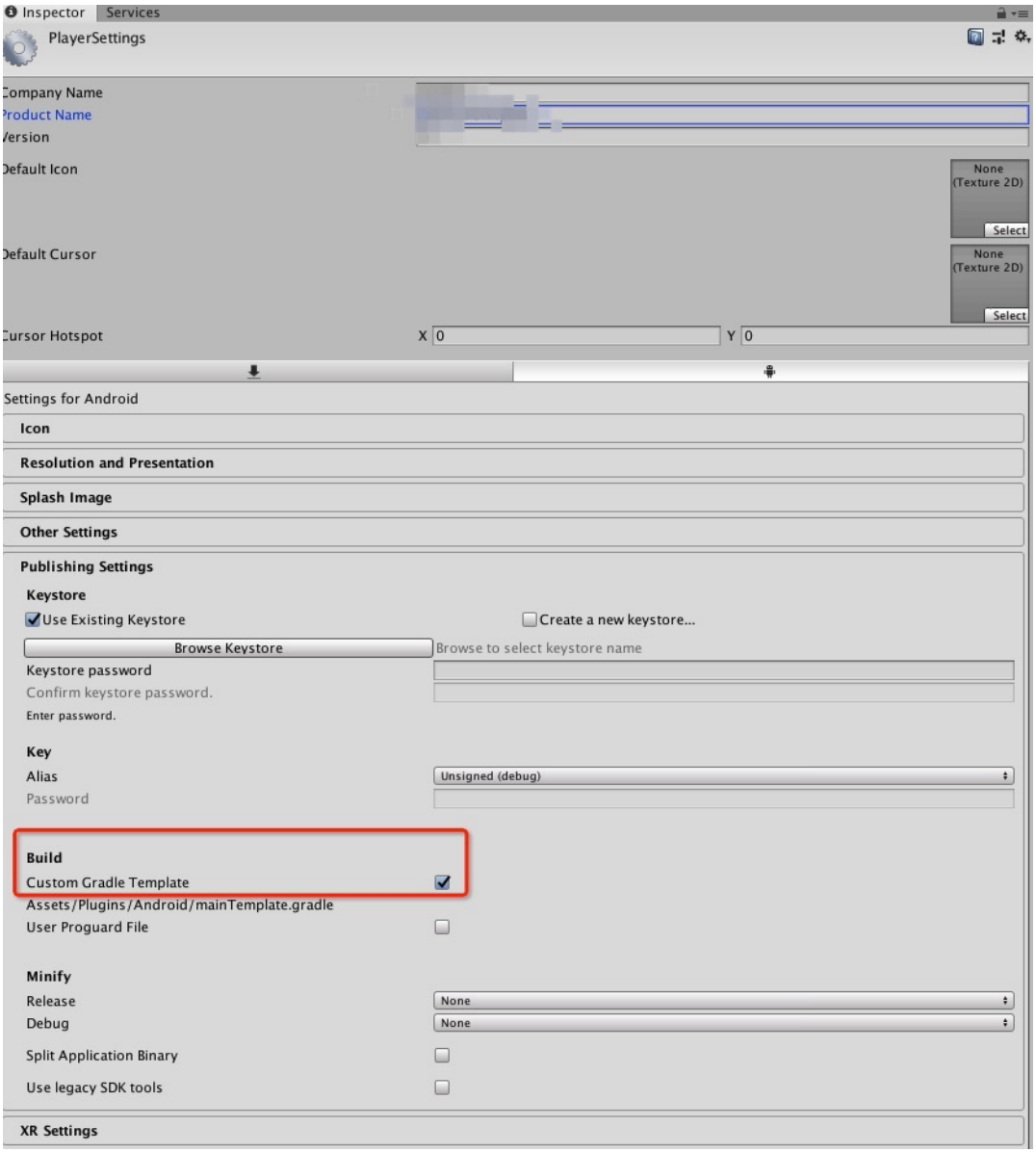
（2）./libs/aars_international_network文件夹说明：

路径	描述	是否必须
./aars_international_network/aars_plugin	非中国区聚合第三方SDK必须的插件包（如果源项目已经有引入，则可将该文件剔除，不加入打包）	是
./aars_international_network/aars_gms_service	Google Service的SDK文件夹（如果源项目已经有引入，则可将该文件剔除，不加入打包）	是
./aars_international_network/aar_admob	**Admob**SDK文件夹	否
./aars_international_network/aar_facebook	**Facebook**SDK文件夹	否
./aars_international_network/aar_adcolony	**Adcolony**SDK文件夹	否
./aars_international_network/aar_applovin	**Applovin**SDK文件夹	否
./aars_international_network/aar_appnext	**Appnext**SDK文件夹	否
./aars_international_network/aar_chartboost	**Chartboost**SDK文件夹	否
./aars_international_network/aar_flurry	**Flurry**SDK文件夹	否
./aars_international_network/aar_inmobi	**Inmobi**SDK文件夹	否
./aars_international_network/aar_ironsource	**Ironsource**SDK文件夹	否
./aars_international_network/aar_maio	**Maio**SDK文件夹	否
./aars_international_network/aar_mintegral_international	**Mintegral（非中国区）**SDK文件夹	否
./aars_international_network/aar_mopub	**Mopub**SDK文件夹	否
./aars_international_network/aar_nend	**Nend**SDK文件夹	否
./aars_international_network/aar_startapp	**StartApp**SDK文件夹	否
./aars_international_network/aar_superawesome	**SuperAwesome**SDK文件夹	否
./aars_international_network/aar_tapjoy	**Tapjoy**SDK文件夹	否
./aars_international_network/aar_unityads	**UnityAds**SDK文件夹	否
./aars_international_network/aar_vungle	**Vungle**SDK文件夹	否

2.mainTemplate.gradle引入和说明

要先使用Unity3d来打包必须的mainTemplate.gradle文件，生成路径如下图所示：


Build Settings
Scenes In Build
Scenes/demoMainScenes 0
Scenes/bannerMainScenes 1
Scenes/interstitialMainScenes 2
Scenes/NativeBannerScene 3
Scenes/nativeMainScenes 4
Scenes/videoMainScenes 5
Add Open Scenes
Platform
PC, Mac & Linux Standalone
Android
iOS
tvOS
Xbox One
PS4
WebGL
Facebook
Android
Texture Compression
Don't override
ETC2 fallback
32-bit
Build System
Gradle
Export Project
Build App Bundle (Google Play)
Run Device
Default device
Refresh
Development Build
Autoconnect Profiler
Script Debugging
Scripts Only Build
Compression Method
Default
SDKs for App Stores
Xiaomi Mi Game Center
Add
Learn about Unity Cloud Build
Player Settings...
Build
Build And Run



针对生成后的mainTemplate.gradle部分说明：
(SDK中有提供生成后的示例文件，由于不同版本的Unity3d工具生成的gradle文件会不一样，需要开发者删除后重新生成自身Unity3d工具下的gradle文件，SDK中的仅供参考)

```
buildscript {
    repositories {
        google()
        jcenter()
    }

    dependencies {
          //不同的Unity3d工具生成的版本号有可能会不一样
        classpath 'com.android.tools.build:gradle:3.2.0'
**BUILD_SCRIPT_DEPS**}
}

...

android {
    compileSdkVersion **APIVERSION**
    buildToolsVersion '**BUILDTOOLS**'

    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }

    defaultConfig {
        minSdkVersion **MINSDKVERSION**
        targetSdkVersion **TARGETSDKVERSION**
```

```
        applicationId '**APPLICATIONID**'
        ndk {
            abiFilters **ABIFILTERS**
        }
        versionCode **VERSIONCODE**
        versionName '**VERSIONNAME**'
        multiDexEnabled true //需要额外补充设置，是为了当代码行数超过64k的时候设置的
    }
    .....

}
```

3.AndroidManifest.xml说明：

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.superapp.filemanager"
    android:versionCode="2"
    android:versionName="1.1" >

<uses-sdk
    android:minSdkVersion="16"
    android:targetSdkVersion="28"
    android:usesCleartextTraffic="true" />

<!--其中usesCleartextTraffic的配置必须设置，主要作用是让游戏里可以使用http请求（必须使用）-->
<application
    android:usesCleartextTraffic="true"
>


<activity android:name="com.unity3d.player.UnityPlayerActivity" android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    <meta-data android:name="unityplayer.UnityActivity" android:value="true" />
</activity>

<!--这个设置主要是为了适配9.0以上的机器（必须使用）-->
<uses-library android:name="org.apache.http.legacy" android:required="false"/>
</application>



<!--必须要有的权限-->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />


</manifest>
```

4.与其他第三方Android SDK合并的说明

（1）必须把第三方的jar包和aar包放在目录：/Assets/Plugins/Android/libs下

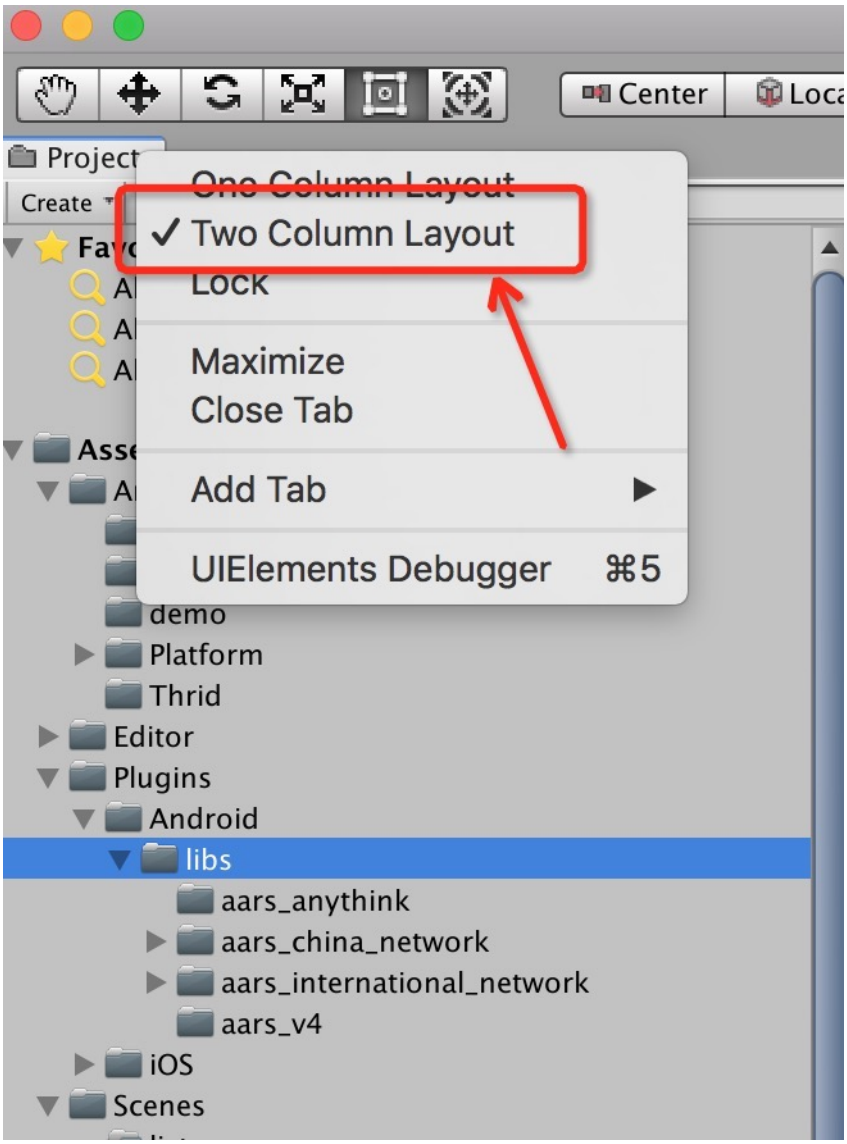
（2）如果第三方sdk有资源，则把资源的文件夹放在目录：/Assets/Plugins/Android/下

（3）如果第三方存在AndroidManifest文件，则需要和/Assets/Plugins/Android/AndroidManifest.xml文件内容合并，TopOn主要将上面说明必要的保留即可

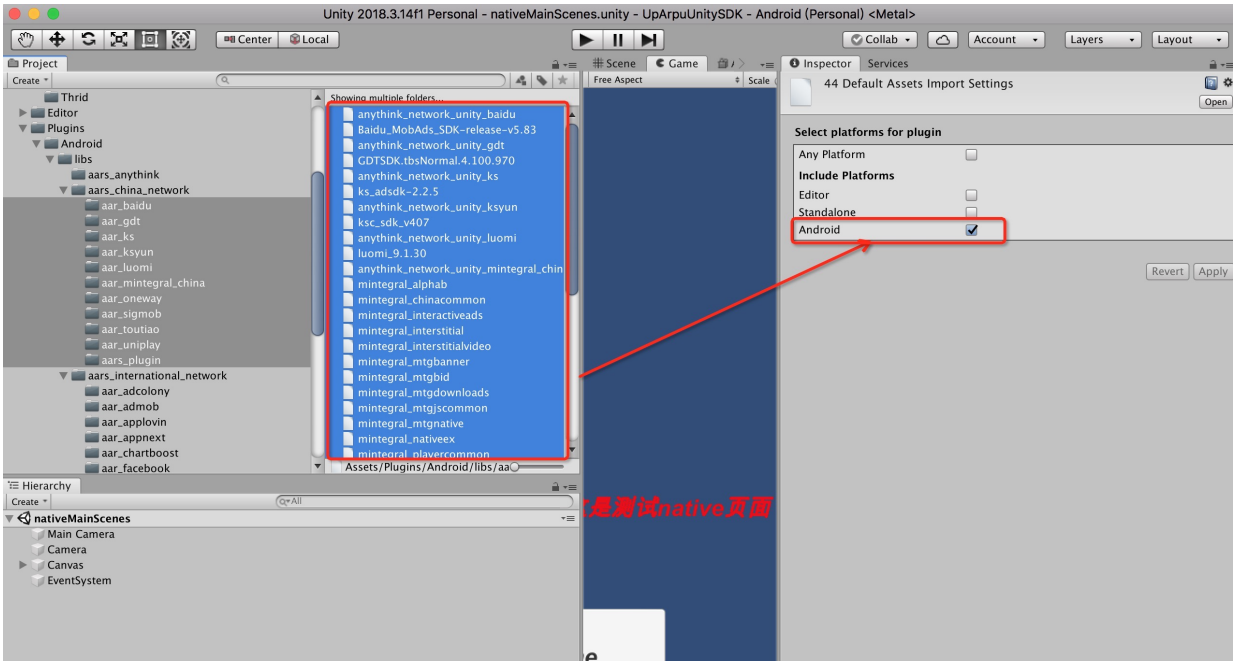
5.如何选择Android的部分AAR或者Jar不进行打包

以不引入中国区的SDK包为例子：

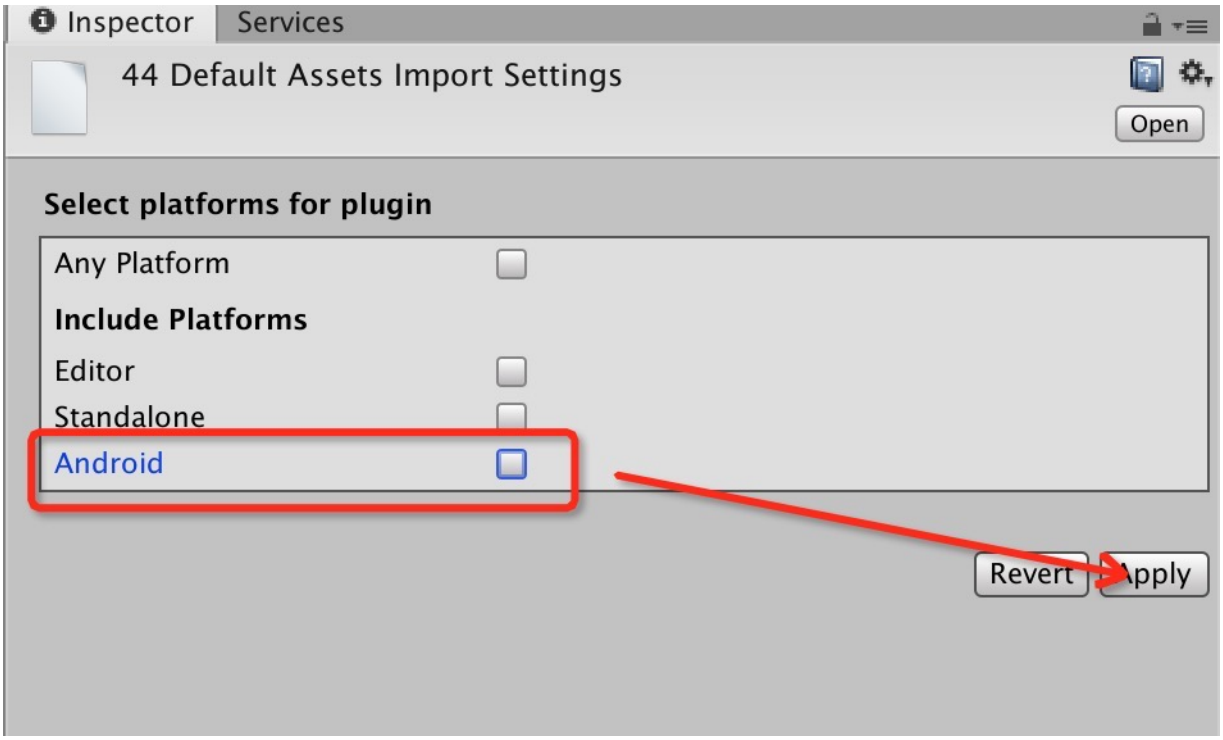
第一步：选择可展示Unity3d中的展示文件的目录



第二步：全选中国区内的SDK文件夹，在隔壁一列全选所有的aar包和jar包，然后在最右侧会出现平台的选择



第三步：把Android平台的打包选择去掉，然后选择Apply，就完成剔除打包的操作



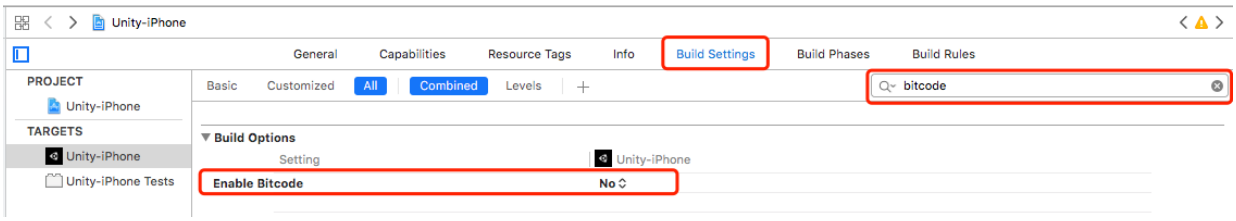
3.1.2 iOS平台导入说明

利用Unity编译出Xcode工程后，打开Xcode工程，按各第三方平台指引引入其需要的SDK并链接其依赖的系统framework及lib等，也可以看TopOn各平台接入帮助TopOn各平台接入帮助

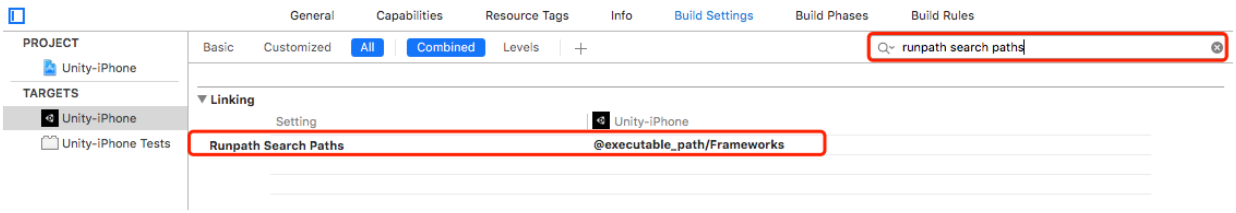
在Unity的sdk包里已经包含所有的第三方Framework包，可根据需要删除不需要的sdk包，详细哪些平台需要哪些包的引入请查看上面的帮助文档

根据以上罗列的信息引入各第三方网络所需SDK并根据各SDK要求引入系统framework和lib之后需要在Build Settings进行以下配置：

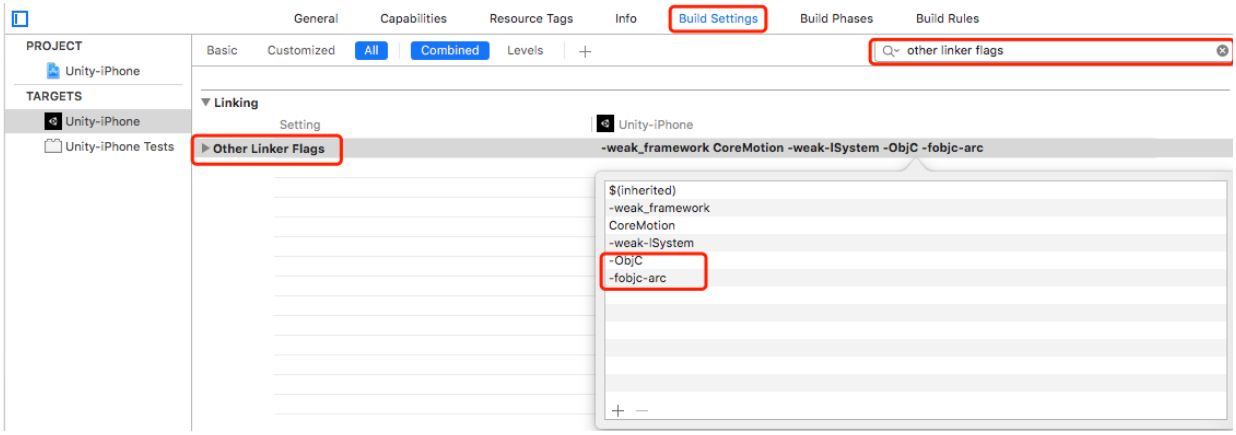
1 在Xcode 工程的**Build Settings**中，搜索**bitcode**，并将其值改为**NO**（当前版本Unity（2018.02）编译出来的Xcode工程中，此项设置默认为Yes），如图：



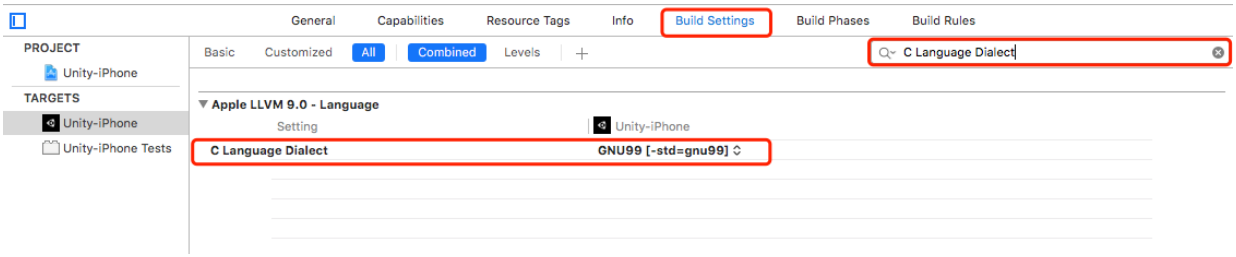
2 在Xcode 工程的**Build Settings**中，搜索**runpath search paths**，并将其值改为**@executable_path/Frameworks** 如图：



3 在Xcode 工程的**Build Settings**中，搜索**other linker flags**，在默认值基础上增加**-ObjC, -fobjc-arc** 如图：



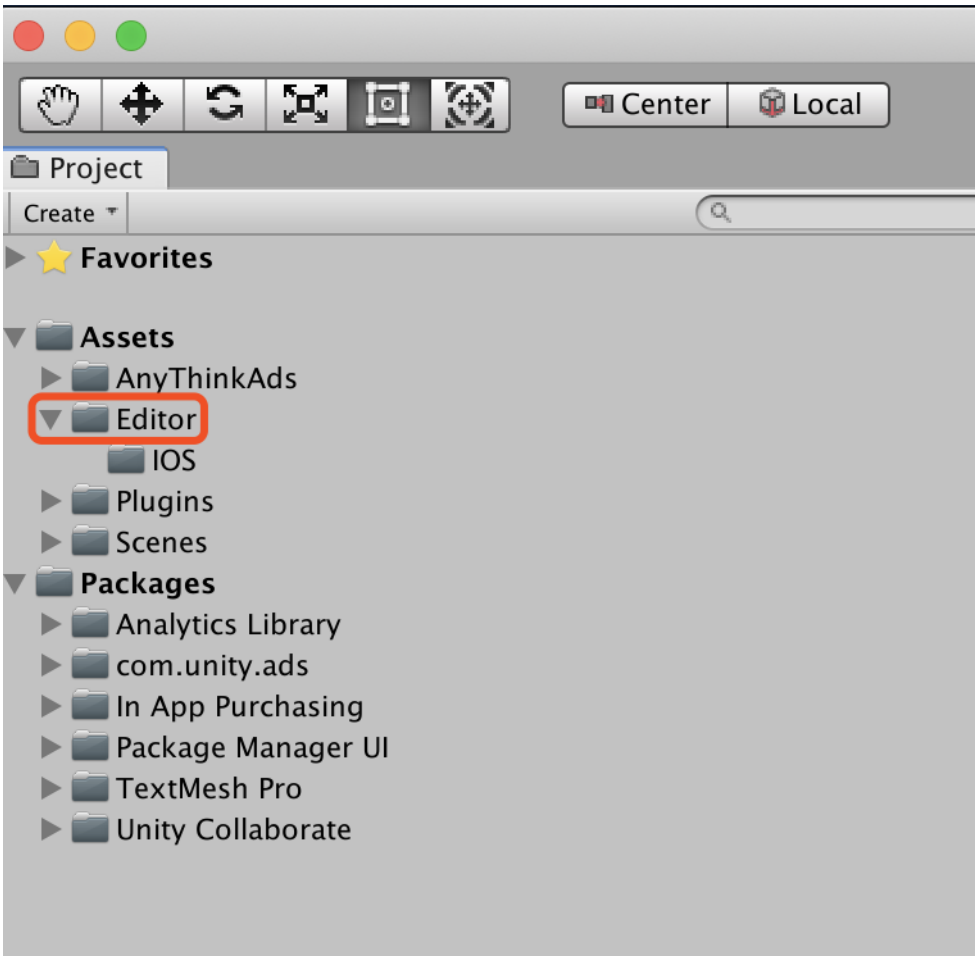
4 在Xcode 工程的**Build Settings**中，搜索**C Language Dialect**，将其值改为**GNU99[-std=gnu99]** 如图：



可以使用以下方法以编程方式完成上述所有配置 C# **Editor Script**:

```
#if (UNITY_5 && UNITY_IOS) || UNITY_IPHONE
using UnityEditor.iOS.Xcode;
#endif

public static class MyBuildPostprocess
{
    [PostProcessBuild(999)]
    public static void OnPostProcessBuild(BuildTarget buildTarget, string path)
    {
        #if (UNITY_5 && UNITY_IOS) || UNITY_IPHONE
            if (buildTarget == BuildTarget.iOS)
            {
                string projectPath = path + "/Unity-iPhone.xcodeproj/project.pbxproj";

                PBXProject pbxProject = new PBXProject();
                pbxProject.ReadFromFile(projectPath);

                string target = pbxProject.TargetGuidByName("Unity-iPhone");
                pbxProject.SetBuildProperty(target, "ENABLE_BITCODE", "NO");
                pbxProject.SetBuildProperty(target, "GCC_ENABLE_OBJC_EXCEPTIONS", "YES");
                pbxProject.SetBuildProperty(target, "GCC_C_LANGUAGE_STANDARD", "gnu99");

                pbxProject.AddBuildProperty(target, "OTHER_LDFLAGS", "-ObjC");
                pbxProject.AddBuildProperty(target, "OTHER_LDFLAGS", "-fobjc-arc");
                pbxProject.AddFileToBuild(target, pbxProject.AddFile("usr/lib/libxml2.tbd", "Libraries/libxml2.tbd", PBXSourceTree.Sdk))
                pbxProject.AddFileToBuild(target, pbxProject.AddFile("usr/lib/libresolv.9.tbd", "Libraries/libresolv.9.tbd", PBXSourceTr
                pbxProject.WriteToFile (projectPath);

                var plistPath = Path.Combine(path, "Info.plist");
                PlistDocument plist = new PlistDocument();
                plist.ReadFromFile(plistPath);
                plist.root.SetString("GADApplicationIdentifier", "ca-app-pub-9488501426181082/7319780494");
                plist.root.SetBoolean("GADIsAdManagerApp", true);
                plist.WriteToFile(plistPath);
        }
        #endif
    }
}
```

注: 您需要将此代码写到C#文件并放到**Unity3D** IDE的**Editor** 目录下:



如果一切顺利，则应编译您的项目。

4 SDK初始化

您可以通过以下代码初始化 **AnyThinkSDK** ，详细参考demo project:

```
ATSDKAPI.setChannel("unity3d_test_channel");
ATSDKAPI.initCustomMap(new Dictionary<string, string> { { "unity3d_data", "test_data" } });
ATSDKAPI.setLogDebug(true);
ATSDKAPI.initSDK("a5c4ad280995c9", "7b4e37f819dbee652ef79c4506e14288");//Use your own app_id & app_key here
```

注: ATSDKAPI类中还有一个init方法，它接受一个侦听对象，您可以使用此侦听器获得SDK初始化完成事件（成功/失败）的通知：

```
namespace AnyThinkAds.Api
{
    public class ATSDKAPI
    {
        private static readonly IATSDKAPIClient client = GetATSDKAPIClient();

        public static void initSDK(string appId, string appKey)
        {
            client.initSDK(appId, appKey);
        }

        public static void initSDK(string appId, string appKey, ATSDKInitListener listener)
        {
            client.initSDK(appId, appKey, listener);
        }
```

由于后续操作（加载，显示等）不依赖于此事件，因此您可以使用上面显示的代码并在初始化SDK后立即调用加载API。

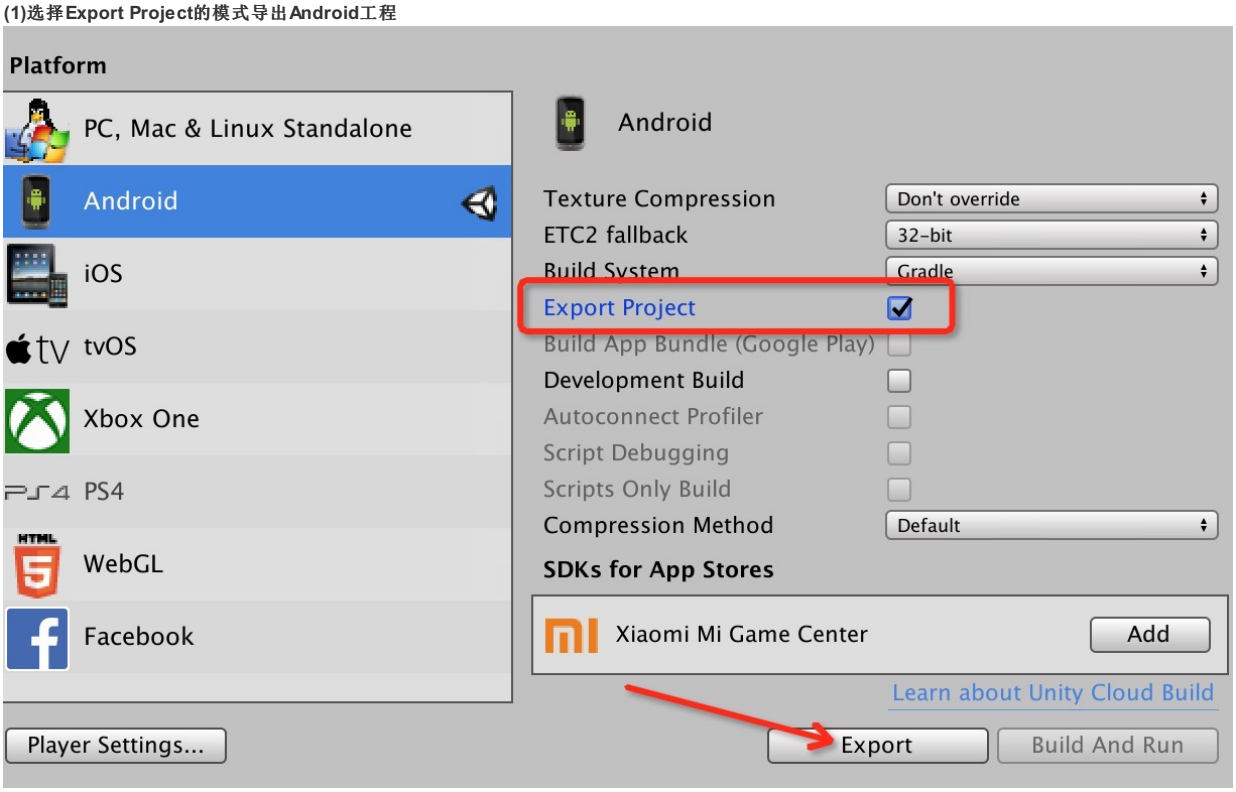
5. 原生广告

在继续之前，请确保已如上所述导入并初始化了SDK。

Android上需要注意的：

目前使用Unity直接打出Android的APK是不能使用原生视频广告，因为Unity打包APK默认会把游戏的Activity硬件加速关闭，所以无法展示视频。如果需要展示原生广告视频的话，必须使用export project的方式处理，处理方式如下：

(1)选择Export Project的模式导出Android工程



(2)修改AndroidManifest里的属性，将图中的hardwareAccelerated的属性设置成true

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" xmlns:tools="http://schemas.android.com/tools" package="com.uparpu.demo" android:
versionCode="2" android:versionName="1.1" android:installLocation="preferExternal">
  <uses-sdk android:minSdkVersion="16" android:targetSdkVersion="28" android:usesCleartextTraffic="true" />
  <application android:usesCleartextTraffic="true" android:isGame="true" android:banner="@drawable/app_banner" android:icon="@mipmap/app_icon">
    <activity android:label="@string/app_name" android:screenOrientation="fullSensor" android:launchMode="singleTask" android:configChanges="mcc|mnc|loc
    ale|touchscreen|keyboard|keyboardHidden|navigation|orientation|screenLayout|uiMode|screenSize|smallestScreenSize|fontScale|layoutDirection|density"
    android:hardwareAccelerated="false" android:name="com.uparpu.demo.UnityPlayerActivity">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
        <category android:name="android.intent.category.LEANBACK_LAUNCHER" />
      </intent-filter>
      <meta-data android:name="unityplayer.UnityActivity" android:value="true" />
    </activity>
```

完成以上步骤之后，使用当前的Android工程进行打包即可。

5.1 加载原生广告

您可以使用以下代码加载原生广告：

```
public void loadNative()
{
        Debug.Log ("Developer load native, unit id = " + mPlacementId_native_all);

        if(callbackListener == null) {
            callbackListener = new ATNativeCallbackListener();
            ATNativeAd.Instance.setListener(callbackListener);
        }

        Dictionary<string,string> gdtlocal = new Dictionary<string,string>();
        gdtlocal.Add ("gdtadtype","3");
        gdtlocal.Add ("gdtad_width","-1");
        gdtlocal.Add ("gdtad_height","-1");
        ATNativeAd.Instance.setLocalExtra(mPlacementId_native_all,gdtlocal);

        Dictionary<string,string> jsonmap = new Dictionary<string,string>();
        jsonmap.Add("age", "22");
        jsonmap.Add("sex", "lady");
        jsonmap.Add("native", "0");

       ATNativeAd.Instance.loadNativeAd(mPlacementId_native_all, jsonmap);
}
```

注: 请继续阅读以了解如何在加载成功/失败事件时得到通知。

5.2 展示原生广告

您可以使用以下代码显示原生广告：

```
public void showNative()
{
        Debug.Log ("Developer show native....");
        ATNativeConfig conifg = new ATNativeConfig ();

        string bgcolor = "#ffffff";
        string textcolor = "#000000";
        int rootbasex = 100, rootbasey = 100;

        int x = rootbasex,y = rootbasey,width = 300*3,height = 200*3,textsize = 17;
        conifg.parentProperty = new ATNativeItemProperty(x,y,width,height,bgcolor,textcolor,textsize, true);

        //adlogo
        x = 0*3;y = 0*3;width = 30*3;height = 20*3;textsize = 17;
        conifg.adLogoProperty  = new ATNativeItemProperty(x,y,width,height,bgcolor,textcolor,textsize, true);

        //adicon
        x = 0*3;y = 50*3-50;width = 60*3;height = 50*3;textsize = 17;
        conifg.appIconProperty  = new ATNativeItemProperty(x,y,width,height,bgcolor,textcolor,textsize, true);

        //ad cta
        x = 0*3;y = 150*3;width = 300*3;height = 50*3;textsize = 17;
        conifg.ctaButtonProperty  = new ATNativeItemProperty(x,y,width,height,"#ff21bcab","#ffffff",textsize, true);

        //ad desc
        x = 60*3;y = 100*3;width = 240*3-20;height = 50*3-10;textsize = 10;
        conifg.descProperty  = new ATNativeItemProperty(x,y,width,height,bgcolor,"#777777",textsize, true);

        //ad image
        x = 60*3;y = 0*3+20;width = 240*3-20;height = 100*3-10;textsize = 17;
        conifg.mainImageProperty  = new ATNativeItemProperty(x,y,width,height,bgcolor,textcolor,textsize, true);

        //ad title
        x = 0*3;y = 100*3;width = 60*3;height = 50*3;textsize = 12;
        conifg.titleProperty  = new ATNativeItemProperty(x,y,width,height,bgcolor,textcolor,textsize, true);

        ATNativeAdView anyThinkNativeAdView = new ATNativeAdView(conifg);
        AnyThinkAds.Demo.ATManager.anyThinkNativeAdView = anyThinkNativeAdView;
        Debug.Log("Developer renderAdToScene--->");
        ATNativeAd.Instance.renderAdToScene(mPlacementId_native_all, anyThinkNativeAdView);
}
```

传递给ATNativeItemProperty类的构造函数的尾部参数表示是否使用像素。 例如，在iPhone 6上，如果分别为x，y，宽度和高度分别传递30、120、300、450，则在iPhone 7上传递给Objective-C代码的实际值将为15、60、150、225 这些值将是10、40、100、150； 也就是说，最终值决定于目标设备的屏幕比例。

正如您在上面看到的，我们为您定义了一个ATNativeConfig类，用于配置本机资产的各种属性（bgColor，textColor，textSize，position等），例如CTA按钮，应用程序图标，标题文本，说明文本，封面图片 等等。 请随时修改config对象中的属性，并查看根据您的修改会发生什么。

如果要从屏幕上删除原生广告，请使用以下代码：

```
public void cleanView()
{
    Debug.Log ("Developer cleanView native....");
    ATNativeAd.Instance.cleanAdView(mPlacementId_native_all,AnyThinkAds.Demo.ATManager.anyThinkNativeAdView);
}
```

5.3 实现原生广告的Listener

要获得有关各种原生广告事件（加载成功/失败，展示和点击等）的通知，您可以定义一个 **ATNativeAdListener** 的实现类，下面是一个示例：

```
class ATNativeCallbackListener : ATNativeAdListener
{
    public void onAdLoaded(string unitId)
    {
        Debug.Log("Developer onAdLoaded------:" + unitId);
    }
    public void onAdLoadFail(string unitId, string code, string message)
    {
        Debug.Log("Developer onAdLoadFail------:" + unitId + "--code:" + code + "--msg:" + message);
    }
    public void onAdImpressed(string unitId)
```

```
    {
        Debug.Log("Developer onAdImpressed------:" + unitId);
    }
    public void onAdClicked(string unitId)
    {
        Debug.Log("Developer onAdClicked------:" + unitId);
    }
    public void onAdVideoStart(string unitId)
    {
        Debug.Log("Developer onAdVideoStart------:" + unitId);
    }
    public void onAdVideoEnd(string unitId)
    {
        Debug.Log("Developer onAdVideoEnd------:" + unitId);
    }
    public void onAdVideoProgress(string unitId, int progress)
    {
        Debug.Log("Developer onAdVideoProgress------:" + unitId);
    }
}
```

注:您在本节中看到的代码段来源于我们Demo的**nativeScene.cs** demo project.

6. 激励视频广告

6.1 加载激励视频

使用以下代码加载激励视频广告:

```
public void loadVideo()
{
    if(callbackListener == null) {
        callbackListener = new ATCallbackListener();
        Debug.Log("Developer init video....unitid:" + mPlacementId_rewardvideo_all);
        ATRewardedVideo.Instance.setListener(callbackListener);
        ATRewardedVideo.Instance.addsetting(mPlacementId_rewardvideo_all, addsetting());
    }

    Dictionary<string,string> jsonmap = new Dictionary<string,string>();
    jsonmap.Add("age", "22");
    jsonmap.Add("sex", "lady");
    jsonmap.Add("rv", "1");

    ATRewardedVideo.Instance.loadVideoAd(mPlacementId_rewardvideo_all,jsonmap);
}
```

注: 请参阅下文，了解如何获得有关激励视频广告事件的通知（加载成功/失败，展示，点击，视频开始/结束和激励）。

6.2 展示激励视频

与展示原生广告相比，展示激励视频要简单得多，只要调用展示api并传递展示广告位ID作为参数：

```
public void showVideo()
{
    Debug.Log ("Developer show video....");
    ATRewardedVideo.Instance.showAd(mPlacementId_rewardvideo_all);
}
```

6.3 实现激励视频的监听器

您可以实现**ATRewardedVideo Listener**接口的类，来获得有关激励视频广告事件的通知：

```
class ATCallbackListener : ATRewardedVideoListener {
    public void onRewardedVideoAdLoaded(string unitId){
        Debug.Log("Developer onRewardedVideoAdLoaded------");
    }

    public void onRewardedVideoAdLoadFail(string unitId, string code, string message){
        Debug.Log("Developer onRewardedVideoAdLoadFail------:code" + code + "--message:" + message);
    }

    public void onRewardedVideoAdPlayStart(string unitId){
        Debug.Log("Developer onRewardedVideoAdPlayStart------");
    }

    public void onRewardedVideoAdPlayEnd(string unitId){
        Debug.Log("Developer onRewardedVideoAdPlayEnd------");
```

```
    }

    public void onRewardedVideoAdPlayFail(string unitId, string code, string message){
        Debug.Log("Developer onRewardedVideoAdPlayFail------code:" + code + "---message:" + message);
    }

    public void onRewardedVideoAdPlayClosed(string unitId, bool isReward){
        Debug.Log("Developer onRewardedVideoAdPlayClosed------isReward:" + isReward);
    }

    public void onRewardedVideoAdPlayClicked(string unitId){
        Debug.Log("Developer onRewardVideoAdPlayClicked------");
    }

    public void onReward(string unitId){
        Debug.Log("Developer onReward------");
    }
    }
```

创建此类的一个实例，并将其传递给加载api中的listener参数，如图所示here.

注意： 本节中的代码段摘自我们Demo的**videoScenes.cs** demo project.

7. 插屏广告

7.1 加载插屏广告

使用以下代码加载插屏广告

```
public void loadInterstitialAd()
{
    if(callback == null) {
        callback = new InterstitalCallback();
        ATInterstitialAd.Instance.setListener(callback);
    }

    Dictionary<string,string> jsonmap = new Dictionary<string,string>();
    jsonmap.Add("age", "22");
    jsonmap.Add("sex", "lady");
    jsonmap.Add("interstitial", "3");

    ATInterstitialAd.Instance.loadInterstitialAd(mPlacementId_interstitial_all, jsonmap);
}
```

注: 请参阅下文，了解如何获得有关插屏广告事件的通知（加载成功/失败，展示，点击，视频开始/结束）。

7.2 展示插屏广告

与激励视频相同，插屏广告只要调用展示api并传递展示广告位ID作为参数：

```
public void showInterstitialAd()
{
    ATInterstitialAd.Instance.showInterstitialAd(mPlacementId_interstitial_all);
}
```

7.3 实现插屏的监听器

您可以实现**ATInterstitialAdListener**接口的类，来获得有关插屏广告事件的通知：

```
class InterstitalCallback : ATInterstitialAdListener
{
    public void onInterstitialAdClick(string unitId)
    {
        Debug.Log("Developer callback onInterstitialAdClick :" + unitId);
    }

    public void onInterstitialAdClose(string unitId)
    {
        Debug.Log("Developer callback onInterstitialAdClose :" + unitId);
    }

    public void onInterstitialAdEndPlayingVideo(string unitId)
    {
        Debug.Log("Developer callback onInterstitialAdEndPlayingVideo :" + unitId);
    }

    public void onInterstitialAdFailedToPlayVideo(string unitId, string code, string message)
```

```
    {
        Debug.Log("Developer callback onInterstitialAdFailedToPlayVideo :" + unitId + "--code:" + code + "--msg:" + message);
    }

    public void onInterstitialAdLoad(string unitId)
    {
        Debug.Log("Developer callback onInterstitialAdLoad :" + unitId);
    }

    public void onInterstitialAdLoadFail(string unitId, string code, string message)
    {
        Debug.Log("Developer callback onInterstitialAdLoadFail :" + unitId + "--code:" + code + "--msg:" + message);
    }

    public void onInterstitialAdShow(string unitId)
    {
        Debug.Log("Developer callback onInterstitialAdShow :" + unitId);
    }

    public void onInterstitialAdStartPlayingVideo(string unitId)
    {
        Debug.Log("Developer callback onInterstitialAdStartPlayingVideo :" + unitId);
    }

    public void onInterstitialAdFailedToShow(string unitId)
    {
        Debug.Log("Developer callback onInterstitialAdFailedToShow :" + unitId);
    }
}
```

注: 您在本节中看到的代码段摘自我们Demo的**interstitialScenes.cs** demo project.

8. Banner广告

8.1 加载Banner广告

使用以下代码加载Banner广告

```
public void loadBannerAd()
{
    if(bannerCallback == null) {
        bannerCallback = new BannerCallback();
        ATBannerAd.Instance.setListener(bannerCallback);
    }

    Dictionary<string, object> jsonmap = new Dictionary<string,object>();
    jsonmap.Add("age", "22");
    jsonmap.Add("sex", "lady");
    jsonmap.Add("banner", "2");
    ATSize bannerSize = new ATSize(this.screenWidth, 100, true);
    jsonmap.Add(ATBannerAdLoadingExtra.kATBannerAdLoadingExtraBannerAdSizeStruct, bannerSize);
    ATBannerAd.Instance.loadBannerAd(mPlacementId_native_all, jsonmap);
}
```

请继续阅读以了解如何获得有关Banner广告事件的通知，例如加载成功/失败，展示和点击。

8.2 展示Banner广告

您可以使用以下代码显示Banner广告：

```
public void showBannerAd()
{
    ATRect arpuRect = new ATRect(0,70, screenWidth, 100, true);
    ATBannerAd.Instance.showBannerAd(mPlacementId_native_all, arpuRect);
}
```

传递给ATRect类的构造函数的末尾参数表示是否使用像素。 例如，在iPhone 6上，如果分别为x，y，宽度和高度分别传递30、120、300、450，则在iPhone 7上传递给Objective-C代码的实际值将为15、60、150、225 这些值将是10、40、100、150； 也就是说，最终值将决定于目标设备的屏幕比例。

如果需要，请使用以下代码从屏幕上**移除**Banner：

```
public void removeBannerAd()
{
    ATBannerAd.Instance.cleanBannerAd(mPlacementId_native_all);
}
```

如果您只想**暂时隐藏** Banner（而不是**从屏幕上移除**），请在此处使用代码：

```
public void hideBannerAd()
{
    ATBannerAd.Instance.hideBannerAd(mPlacementId_native_all);
}
```

隐藏Banner后，可以使用以下代码重新显示它：

```
public void reshowBannerAd()
{
    ATBannerAd.Instance.showBannerAd(mPlacementId_native_all);
}
```

注意: 请注意，此处的showBannerAd方法不接受rect参数，这与您首次显示Banner广告时不同。

移除Banner广告和隐藏Banner广告的区别在于，从屏幕上移除Banner广告时也会破坏它（这意味着在再次显示之前，必须先加载Banner广告），而隐藏Banner只需调用showBannerAd方法即可重新显示以前隐藏的Banner广告**不传递ATRect参数**

8.3 实现Banner的监听器

要获得有关各种Banner广告事件（加载成功/失败，展示和点击）的通知，只需定义一个 **ATBannerAdListener**接口的实现类：

```
class BannerCallback : ATBannerAdListener
{
    public void onAdAutoRefresh(string unitId)
    {
        Debug.Log("Developer callback onAdAutoRefresh :" +  unitId);
    }

    public void onAdAutoRefreshFail(string unitId, string code, string message)
    {
        Debug.Log("Developer callback onAdAutoRefreshFail : "+ unitId + "--code:" + code + "--msg:" + message);
    }

    public void onAdClick(string unitId)
    {
        Debug.Log("Developer callback onAdClick :" + unitId);
    }

    public void onAdClose(string unitId)
    {
        Debug.Log("Developer callback onAdClose :" + unitId);
    }

    public void onAdImpress(string unitId)
    {
        Debug.Log("Developer callback onAdImpress :" + unitId);
    }

    public void onAdLoad(string unitId)
    {
        Debug.Log("Developer callback onAdLoad :" + unitId);
    }

    public void onAdLoadFail(string unitId, string code, string message)
    {
        Debug.Log("Developer callback onAdLoadFail : : " + unitId + "--code:" + code + "--msg:" + message);
    }
}
```

注: 您在本节中看到的代码段摘自我们Demo的**bannerScene.cs** demo project.

9. 开屏广告

强烈建议使用Xcode/Android Studio中的原生API（Objective-C / Java）而不是Unity3D中的C＃脚本来直接。 这样做的原因是，应用程序启动完成后可以立即加载并显示开屏广告。 如果您使用C＃将开屏广告集成到Unity3D中，则在启动Unity游戏引擎之前，它无法加载并显示开屏广告(这种情况发生在应用程序启动完成之后，并且需要花费更多时间)。 有关如何使用Objective-C / Java集成开屏广告的更多信息，请参考iOS集成说明 & Android集成说明.