# 1. UNITY ADS

1. **SETUP FOR UNITY EDITOR**

* Switch to Android/iOS platform
* Configure the Player Settings: product name, package name, keystore, icon, ...
* Go to Services > Create Unity Project ID, turn on Ads

1. **SETUP FOR UNITY DASHBOARD**

<https://operate.dashboard.unity3d.com>

* Monetization > Projects > Choose the project
* Monetization > Platform >
  1. Copy the *GAME ID*
  2. Choose Google Play Store or Apple App Store
* It will show AD Placements > Video, Rewarded video > Set to the default you want.

1. SCRIPTING FOR UNITY

# BannerAdsUnity.cs

public class BannerAdsUnity : MonoBehaviour  
 {  
 string bannerPlacement = "bannerBottom";  
 public static BannerAdsUnity Instance;  
 public bool TestMode;  
#if UNITY\_IOS  
 public const string gameID = "2979944";  
#elif UNITY\_ANDROID  
 public const string gameID = "2979945";  
#elif UNITY\_EDITOR  
 public const string gameID = "1111111";  
#endif  
 private void Awake()  
 {  
 Instance = this;  
 DontDestroyOnLoad(gameObject);  
 }  
 void Start()  
 {  
 if (GameSettings.settings.boughtRemoveAds)  
 {  
 }  
 else  
 Advertisement.Initialize(gameID, TestMode);  
 }  
 public bool ShowBannerWhenReady()  
 {  
 if (!Advertisement.IsReady(bannerPlacement))  
 {  
 }  
 else  
 {  
 Advertisement.Banner.Show(bannerPlacement);  
 AppsFlyerManager.Instance.TrackBannerUnityView();  
 return true;  
 }  
 return false;  
 }  
 }

# InterstitialAdsUnity.cs

public class IntertitialAds : MonoBehaviour  
 {  
 public static IntertitialAds Instance;  
 string placementId = "interstitial";  
 public bool testMode = false;  
#if UNITY\_IOS  
 public const string gameID = "2979944";  
#elif UNITY\_ANDROID  
 public const string gameID = "2979945";  
#elif UNITY\_EDITOR  
 public const string gameID = "1111111";  
#endif  
  
 private void Awake()  
 {  
 if (Instance == null)  
 {  
 DontDestroyOnLoad(gameObject);  
 Instance = this;  
 }  
 else if (Instance != this)  
 Destroy(gameObject);  
 }  
  
 private void Start()  
 {  
 if (GameSettings.settings.boughtRemoveAds)  
 {  
 }  
 else  
 Monetization.Initialize(gameID, testMode);  
 }  
  
 public bool ShowAdWhenReady()  
 {  
 if (!Monetization.IsReady(placementId))  
 return false;  
 ShowAdPlacementContent ad = null;  
 ad = Monetization.GetPlacementContent(placementId) as ShowAdPlacementContent;  
 if (ad != null)  
 {  
 ad.Show();  
 ad.Show(AdFinished);  
 AppsFlyerManager.Instance.TrackInterstitialUnityView();  
 }  
  
 return true;  
 }  
  
 void AdFinished(ShowResult result)  
 {  
 if (result == ShowResult.Finished)  
 {  
// GamePlay.Instance.OverType = GamePlay.GameOverType.None;  
 Monetization.Initialize(gameID, testMode);  
 }  
 }  
 }

# RewardAdsUnity.cs

public class RewardAdsUnity : MonoBehaviour  
{  
 public static RewardAdsUnity Instance;  
 private string placementId = "rewardedVideo";  
 public bool testMode = false;  
#if UNITY\_IOS  
 public const string gameID = "2979944";  
#elif UNITY\_ANDROID  
 public const string gameID = "2979945";  
#elif UNITY\_EDITOR  
 public const string gameID = "1111111";  
#endif  
 private void Awake()  
 {  
 if (Instance == null)  
 {  
 DontDestroyOnLoad(gameObject);  
 Instance = this;  
 }  
 else if (Instance != this)  
 Destroy(gameObject);  
 }  
 private void Start()  
 {  
 Monetization.Initialize(gameID, testMode);  
 }  
 public bool ShowVideoAds()  
 {  
 if (!UnityAdsReady())  
 return false;  
 ShowAdPlacementContent ad = null;  
 ad = Monetization.GetPlacementContent(placementId) as ShowAdPlacementContent;  
 if (ad != null)  
 {  
 AppsFlyerManager.Instance.TrackVideoRewardUnityViewed();  
 GameManager.Instance.soundManager.PauseMusic();  
 ad.Show(AdFinished);  
 }  
 return true;  
 }  
 public bool UnityAdsReady()  
 {  
 return Monetization.IsReady(placementId);  
 }  
 void AdFinished(ShowResult result)  
 {  
 if (result == ShowResult.Finished) // if user finished a video, so reward   
 {  
 AppsFlyerManager.Instance.TrackVideoRewardUnityFinished();  
 GameManager.Instance.soundManager.PlayMusic();  
 Monetization.Initialize(gameID, testMode);  
 GamePlay.Instance.AfterVideoAds();  
 }  
 }  
}

# 2. ADD FACEBOOK ADS TO GAME

# Import facebook sdk to unity game

# Facebook > Edit Settings > add key

# Go to facebook > app game > add platform (ios, android)

# Go to [developers.facebook.com](https://developers.facebook.com) > Settings > Basic

# add privacy, term, add platform

# <http://www.mofiin.com/privacy>

# <http://www.mofiin.com/terms>

# turn on Single Sign On > Deep Linking, add add name, package, class name, key hash

# Add to Game scene a game object with script Facebook.cs (code below)

# AndroidManifest add

# @string/facebook\_app\_id

# @string/fb\_login\_protocol\_scheme

# <https://developers.facebook.com/tools/app-ads-helper/> to check

# To test ads in your phone: use test ad type

# IMG\_16\_9\_APP\_INSTALL

# Example:

# IMG\_16\_9\_APP\_INSTALL#457854978072123\_469445200246434

# AndroidManifest.xml

<**activity  
 android:name="com.facebook.ads.AudienceNetworkActivity"  
 android:configChanges="keyboardHidden|orientation|screenSize"** />

# FBBanner.cs

public class FBBannerAds : MonoBehaviour  
 {  
 private bool \_bannerLoaded;  
 private AdView \_adView;  
 public bool Testing;  
 public static FBBannerAds Instance;  
 private void Awake()  
 {  
 if (Instance == null)  
 {  
 DontDestroyOnLoad(gameObject);  
 Instance = this;  
 }  
 else if (Instance != this)  
 Destroy(gameObject);  
 }  
 private void Start()  
 {  
 if (GameSettings.settings.boughtRemoveAds)  
 {  
 HideAds();  
 }  
 else  
 LoadBanner();  
 }  
 private void LoadBanner()  
 {  
#if UNITY\_ANDROID  
 string adUnitId = GameConfig.Instance.FbAds.androidBanner.Trim();  
#elif UNITY\_IPHONE  
 string adUnitID = GameConfig.Instance.FbAds.iOSBanner.Trim();  
#endif  
 if (\_adView)  
 \_adView.Dispose();  
  
 \_adView = Testing  
 ? new AdView("IMG\_16\_9\_APP\_INSTALL#457854978072123\_469445200246434", AdSize.BANNER\_HEIGHT\_50)  
 : new AdView(adUnitId, AdSize.BANNER\_HEIGHT\_50);  
  
 \_adView.Register(gameObject);  
 \_adView.AdViewDidFailWithError = (delegate(string error)  
 {  
 \_bannerLoaded = false;  
// Debug.Log("Banner failed to load with error: " + error);  
 });  
 \_adView.AdViewWillLogImpression = (delegate() { });  
 \_adView.AdViewDidClick = (delegate() { });  
  
 // Initiate a request to load an ad.  
 \_adView.LoadAd();  
 // Set delegates to get notified on changes or when the user interacts with the ad.  
 \_adView.AdViewDidLoad = (delegate() // Banner loaded.  
 {  
 \_bannerLoaded = true;  
 \_adView.Show(AdPosition.BOTTOM);  
 AppsFlyerManager.Instance.TrackBannerFacebookView();  
 });  
 \_adView.AdViewWillLogImpression = (delegate() // Banner logged impression  
 {  
 });  
 \_adView.AdViewDidClick = (delegate() // Banner clicked  
 {  
 });  
 }  
 public bool BannerLoaded()  
 {  
 return \_bannerLoaded;  
 }  
 private void HideAds()  
 {  
 gameObject.SetActive(false);  
 }  
 }

# FBInterstitial.cs

public class FBInterstitial : MonoBehaviour  
 {  
 private InterstitialAd interstitialAd;  
 private bool isLoaded;  
 private bool didClose;  
 public bool Testing;  
 public static FBInterstitial Instance;  
 private void Awake()  
 {  
 if (Instance == null)  
 {  
 DontDestroyOnLoad(gameObject);  
 Instance = this;  
 }  
 else if (Instance != this)  
 Destroy(gameObject);  
 }  
  
 private void Start()  
 {  
 if (GameSettings.settings.boughtRemoveAds)  
 HideAds();  
 else  
 LoadInterstitial();  
 }  
 private void HideAds()  
 {  
 gameObject.SetActive(false);  
 }  
 public void LoadInterstitial()  
 {  
#if UNITY\_ANDROID  
 string adUnitId = GameConfig.Instance.FbAds.androidInterstitial.Trim();  
#elif UNITY\_IPHONE  
 string adUnitID = GameConfig.Instance.FbAds.iOSInterstitial.Trim();  
#endif  
 interstitialAd = Testing  
 ? new InterstitialAd("IMG\_16\_9\_APP\_INSTALL#457854978072123\_469446143579673")  
 : new InterstitialAd(adUnitId);  
 this.interstitialAd.Register(this.gameObject);  
  
 // Set delegates to get notified on changes or when the user interacts with the ad.  
 this.interstitialAd.InterstitialAdDidLoad = (delegate()  
 {  
// Debug.Log("Interstitial ad loaded.");  
 this.isLoaded = true;  
 });  
 interstitialAd.InterstitialAdDidFailWithError = (delegate(string error)  
 {  
// Debug.Log("Interstitial ad failed to load with error: " + error);  
 });  
 interstitialAd.InterstitialAdWillLogImpression = (delegate()  
 {  
// Debug.Log("Interstitial ad logged impression.");  
 });  
 interstitialAd.InterstitialAdDidClick = (delegate()  
 {  
// Debug.Log("Interstitial ad clicked.");  
 });  
  
 this.interstitialAd.interstitialAdDidClose = (delegate()  
 {  
// Debug.Log("Interstitial ad did close.");  
 didClose = true;  
 if (this.interstitialAd != null)  
 {  
 this.interstitialAd.Dispose();  
 LoadInterstitial();  
 }  
 });  
#if UNITY\_ANDROID  
 this.interstitialAd.interstitialAdActivityDestroyed = (delegate()  
 {  
 if (!didClose)  
 {  
 }  
 else  
 {  
 }  
 });  
#endif  
  
 this.interstitialAd.LoadAd();  
 }  
 public bool ShowInterstitial()  
 {  
 if (!this.isLoaded)  
 {   
 }  
 else  
 {  
// Debug.Log("Interstitial Ad not loaded!");  
 this.interstitialAd.Show();  
 AppsFlyerManager.Instance.TrackInterstitialFacebookView();  
 this.isLoaded = false;  
 return true;  
 }  
 LoadInterstitial();  
 return false;  
 }  
 }

# FBRewardedVideo.cs

public class FBRewardedVideo : MonoBehaviour  
 {  
 private RewardedVideoAd rewardedVideoAd;  
 private bool isLoaded;  
 private bool didClose;  
 public bool Testing;  
 public static FBRewardedVideo Instance;  
 private void Awake()  
 {  
 if (Instance == null)  
 {  
 DontDestroyOnLoad(gameObject);  
 Instance = this;  
 }  
 else if (Instance != this)  
 Destroy(gameObject);  
 }  
 private void Start()  
 {  
 LoadRewardedVideo();  
 }  
 private void LoadRewardedVideo()  
 {  
#if UNITY\_ANDROID  
 var videoId = GameConfig.Instance.FbAds.androidRewardedVideo.Trim();  
#elif UNITY\_IPHONE  
 var videoId = GameConfig.Instance.FbAds.iOSRewardedVideo.Trim();  
#endif  
 RewardData rewardData = new RewardData();  
 rewardData.UserId = "USER\_ID";  
 rewardData.Currency = "REWARD\_ID";  
 rewardedVideoAd =  
 Testing  
 ? new RewardedVideoAd("VID\_HD\_16\_9\_15S\_APP\_INSTALL#2025198837536206\_2115449855177770", rewardData)  
 : new RewardedVideoAd(videoId, rewardData);  
 rewardedVideoAd.Register(gameObject);  
 // Set delegates to get notified on changes or when the user interacts with the ad.  
 rewardedVideoAd.RewardedVideoAdDidLoad = (delegate()  
 {  
// Debug.Log("RewardedVideo ad loaded.");  
 isLoaded = true;  
 });  
 rewardedVideoAd.RewardedVideoAdDidFailWithError = (delegate(string error)  
 {  
// Debug.Log("RewardedVideo ad failed to load with error: " + error);  
 });  
 rewardedVideoAd.RewardedVideoAdWillLogImpression = (delegate()  
 {  
// Debug.Log("RewardedVideo ad logged impression.");  
 });  
 rewardedVideoAd.RewardedVideoAdDidClick = (delegate()  
 {  
// Debug.Log("RewardedVideo ad clicked.");  
 });  
 rewardedVideoAd.RewardedVideoAdDidClose = (delegate()  
 {  
 didClose = true;  
 if (rewardedVideoAd != null)  
 {  
 rewardedVideoAd.Dispose();  
 }  
 });  
 rewardedVideoAd.RewardedVideoAdDidSucceed = (delegate() // for finished video  
 {  
 GamePlay.Instance.AfterVideoAds();  
 LoadRewardedVideo();  
 });  
 rewardedVideoAd.RewardedVideoAdDidFail = (delegate() // for fail finished  
 {  
 GameOver.Instance.OnCancel();  
 });  
#if UNITY\_ANDROID  
 rewardedVideoAd.rewardedVideoAdActivityDestroyed = (delegate()  
 {  
 if (!didClose)  
 {  
 GameOver.Instance.OnCancel();  
 }  
 });  
#endif  
 rewardedVideoAd.LoadAd();  
 }  
 public bool VideoIsReady()  
 {  
 return isLoaded;  
 }  
 public bool ShowRewardedVideo()  
 {  
 if (!isLoaded)  
 LoadRewardedVideo();  
 else  
 {  
 rewardedVideoAd.Show();  
 AppsFlyerManager.Instance.TrackRewardVideoFacebookView();  
 isLoaded = false;  
 return true;  
 }  
 return false;  
 }  
 }

# Complete video:

this.rewardedVideoAd.RewardedVideoAdDidSucceed = (delegate() // for finished video  
{  
 GamePlay.Instance.AfterVideoAds();  
});  
rewardedVideoAd.RewardedVideoAdDidFail = (delegate() // for fail finished  
{  
 GameOver.Instance.OnCancel();  
});

# 3. APPLOVIN ADS IMPLEMENTATION

# - Import Applovin SDK for unity

# 1. Android and iOS

# Add following script to the game object with name:ApplovinListener

# 2. iOS

**Link the Following Frameworks in Your Project**

- AdSupport

- AVFoundation

- CoreGraphics

- CoreMedia

- CoreTelephony

- SafariServices (NEW)

- StoreKit

- SystemConfiguration

- UIKit

- WebKit (NEW)

- libz.tbd (NEW)

## Enable Xcode Flags

The **-ObjC** flag **must** be added for static libraries to run correctly.

To enable the -ObjC flag, click on your project settings, go to *Build Settings*, search for *Other Linker Flags* and add **-ObjC**

# Add this code to gameobject

# AppLovinManager.cs

public class AppLovinManager : MonoBehaviour  
 {  
 private const string SDK\_KEY = "your\_key";  
 private bool \_isPreloadingRewardedVideo = false;  
 public static AppLovinManager Instance;  
 private void Awake()  
 {  
 if (Instance == null)  
 {  
 DontDestroyOnLoad(gameObject);  
 Instance = this;  
 }  
 else if (Instance != this)  
 Destroy(gameObject);  
 }  
 void Start()  
 {  
 AppLovin.SetSdkKey(SDK\_KEY);  
 AppLovin.InitializeSdk();  
// AppLovin.SetTestAdsEnabled("true");  
 AppLovin.SetUnityAdListener(gameObject.name); // gameobject with name ApplovinListener  
 Invoke("LoadVideoDelay", 12f);  
 }  
  
 private void LoadVideoDelay()  
 {  
 AppLovin.LoadRewardedInterstitial();  
 if (GameSettings.settings.boughtRemoveAds)  
 AppLovin.HideAd();  
 }  
 public bool ShowInterstitial()  
 {  
 if (!AppLovin.HasPreloadedInterstitial())  
 {  
 }  
 else  
 {  
 AppLovin.ShowInterstitial();  
 return true;  
 }  
 AppLovin.PreloadInterstitial();  
 return false;  
 }  
 public bool InterstitialIsReady() // if readey inter  
 {  
 var result = AppLovin.HasPreloadedInterstitial();  
 if (result)  
 {  
 }  
 else  
 AppLovin.PreloadInterstitial();  
 return result;  
 }  
 public bool ShowRewardedInterstitial() // show video  
 {  
 if (!AppLovin.IsIncentInterstitialReady())  
 {  
 \_isPreloadingRewardedVideo = true;  
 AppLovin.LoadRewardedInterstitial();  
unity }  
 else  
 {  
 \_isPreloadingRewardedVideo = false;  
 AppLovin.ShowRewardedInterstitial();  
 AppsFlyerManager.Instance.TrackVideoAppLovinView();  
 return true;  
 }  
 return false;  
 }  
 public void ShowBanner()  
 {  
 if (GameSettings.settings.boughtRemoveAds)  
 {  
 return;  
 }  
 AppLovin.ShowAd(AppLovin.AD\_POSITION\_CENTER, AppLovin.AD\_POSITION\_BOTTOM);  
 AppsFlyerManager.Instance.TrackBannerAppLovinView();  
 }  
 void onAppLovinEventReceived(string ev)  
 {  
 if (ev.Equals("LOADBANNERFAILED")) // fail to load banner  
 YuanAds.Instance.ShowBanner();  
 if (ev.Equals("REWARDAPPROVEDINFO"))// if user complete video and call reward function here

{   
 // Process an event like REWARDAPPROVEDINFO:100:Credits  
 char[] delimiter = {'|'};  
 string[] split = ev.Split(delimiter);  
 // Pull out the amount of virtual currency.  
 double amount = double.Parse(split[1]);  
 // Pull out the name of the virtual currency  
 string currencyName = split[2];  
 // Do something with this info - for example, grant coins to the user  
 // myFunctionToUpdateBalance(currencyName, amount);  
 Log("Rewarded " + amount + " " + currencyName);  
 AppLovin.LoadRewardedInterstitial();  
 }  
// }  
 // Check if this is a Rewarded Video preloading event  
 else if (\_isPreloadingRewardedVideo && (ev.Equals("LOADED") || ev.Equals("LOADFAILED")))  
 {  
 \_isPreloadingRewardedVideo = false;  
 if (ev.Equals("LOADED"))  
 {  
// RewardedVideoButtonTitle.text = REWARDED\_VIDEO\_BUTTON\_TITLE\_SHOW;  
 }  
 else  
 {  
// RewardedVideoButtonTitle.text = REWARDED\_VIDEO\_BUTTON\_TITLE\_PRELOAD;  
 }  
 }  
 }  
 }

**4. ADMOB ADS**

<https://developers.google.com/admob/unity/start>

* Step by step
  1. import plugin
  2. get the singleton instance
     + create AdsManager.cs
  3. load an add
     + banner:
       - Request banner() will show banner also
       - Show banner in scene you want:
     + interstitial:
       - request() > show() > destroy
     + reward video
       - request() > show() > destroy
  4. show it
  5. use events to get reward

be careful with appId and adUnitId

code

<https://github.com/googleads/googleads-mobile-unity/blob/master/samples/HelloWorld/Assets/Scripts/GoogleMobileAdsDemoScript.cs>