AdManager の使い方

広告表示（UnityAds 用）

AdManager の実装の方法

確認＆準備

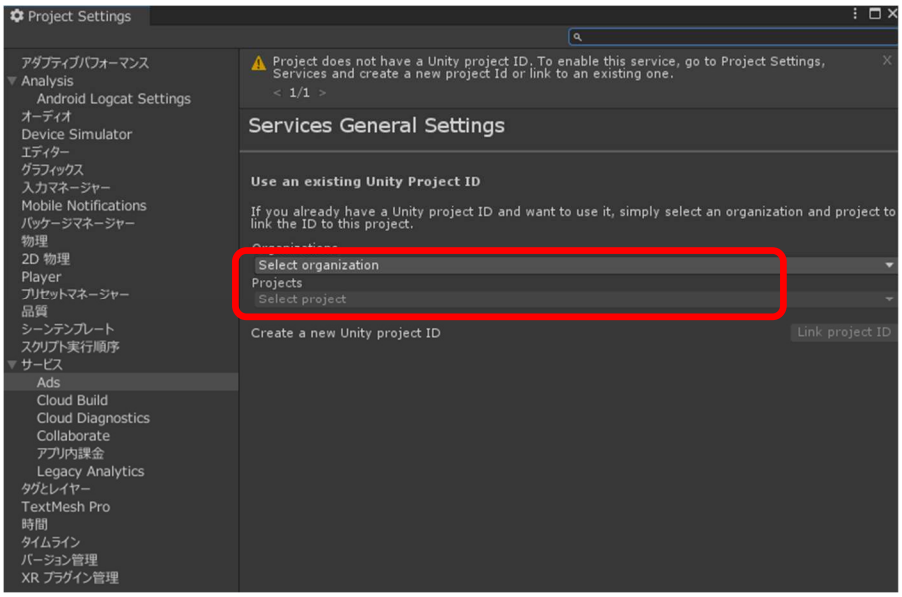
・Unity プロジェクトを開き、サービスタブを開く

雲マークをクリックするとサービスタブを開く

・プロジェクトの UnityAds 登録

Ａｄｓをクリックし ProjectSettings を開く

・ProjectSettings の[サービス]→[Ads]で UnityAds の設定を行う



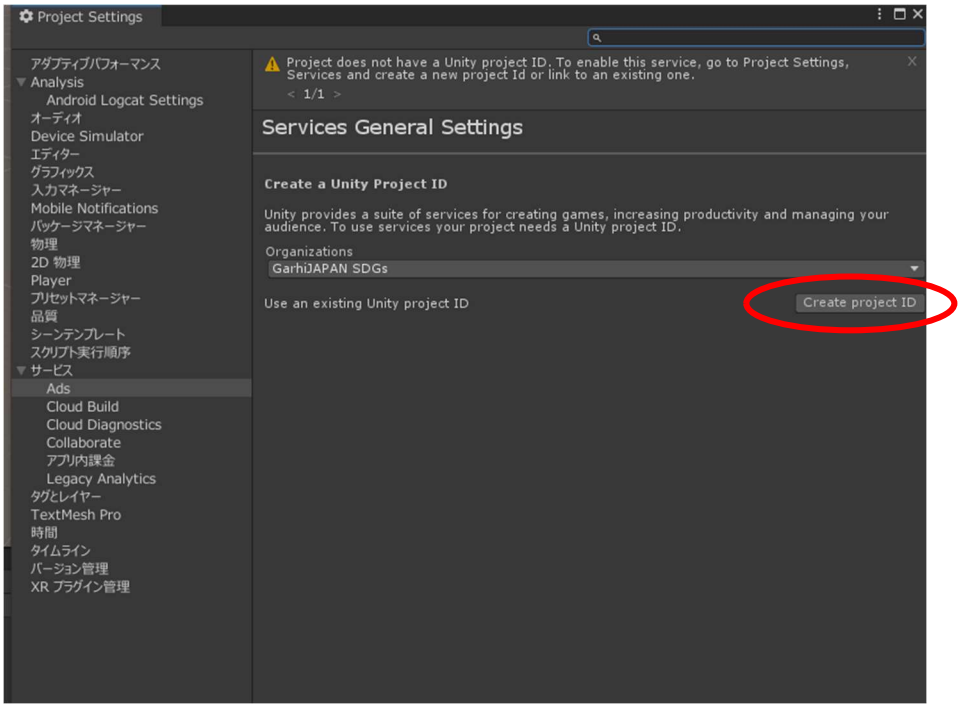
既に、UnityID を登録している場合、Organizations で GarhiJAPAN SDGs を選択し

Projects から該当するプロジェクトを選択する

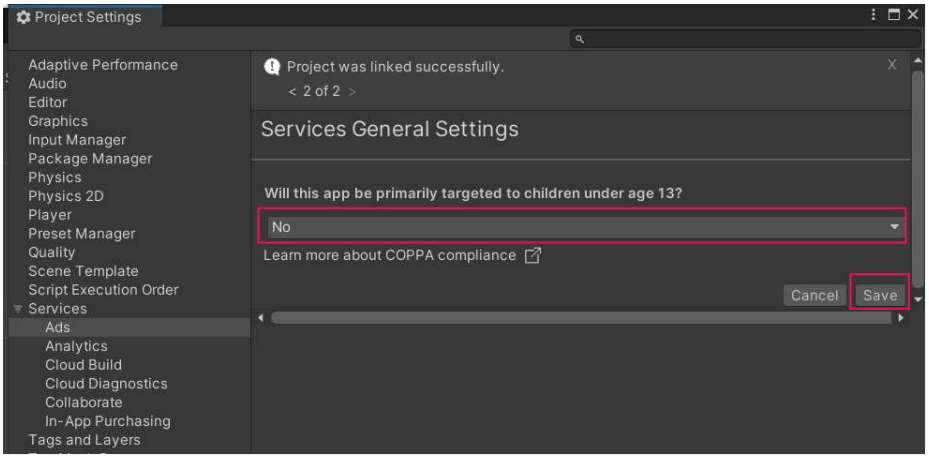
基本はプロジェクト作成済みのため、作成しなくていい

プロジェクトを作成しないとビルドできないはず

Create a Unity Project ID 画面で Organizations で GarhiJAPAN SDGs を選択し [Create project ID]をクリックする



しばらくすると、アプリのターゲット年齢を設定する。13 歳以下向けのコンテンツであれば、「Yes」を選択する。そうでなければ「No」を選択して「Save」をクリックする

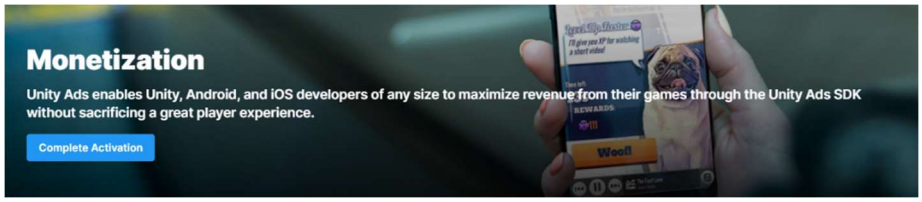


ターゲット年齢を設定したら、Project Setting ウィンドウの Ads の「オフ」をクリックしオンにする

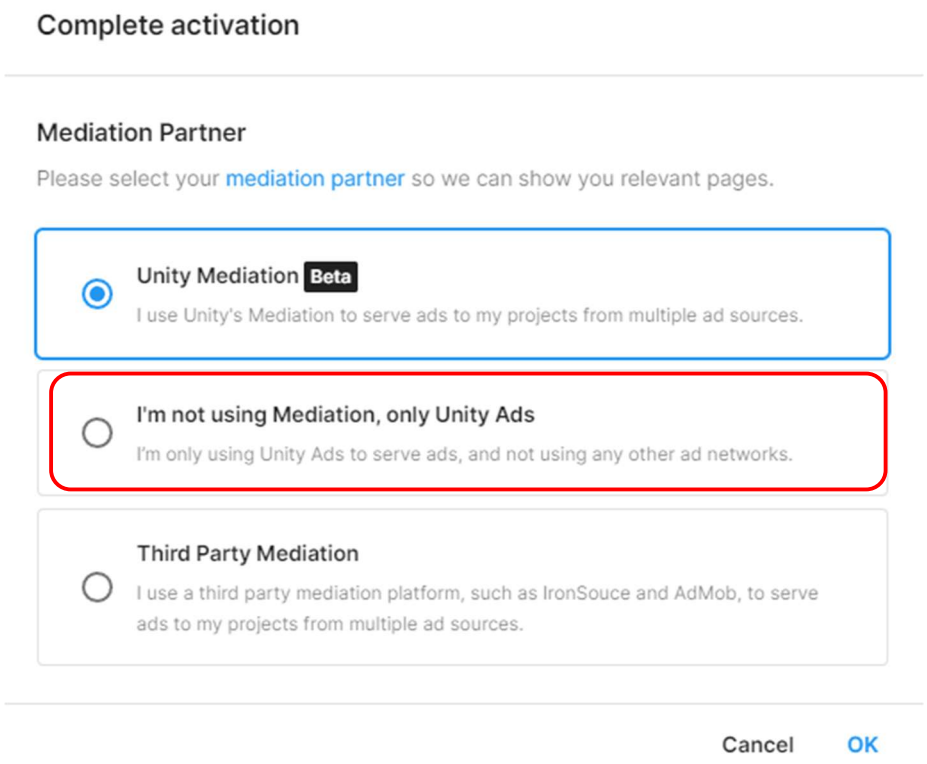
元気玉プロジェクトは基本１３才以下を対象とするので「Yes」を選択する

Unity ダッシュボード

ここで、Unity プロジェクトのダッシュボードを開き、Ad Unit を選択する

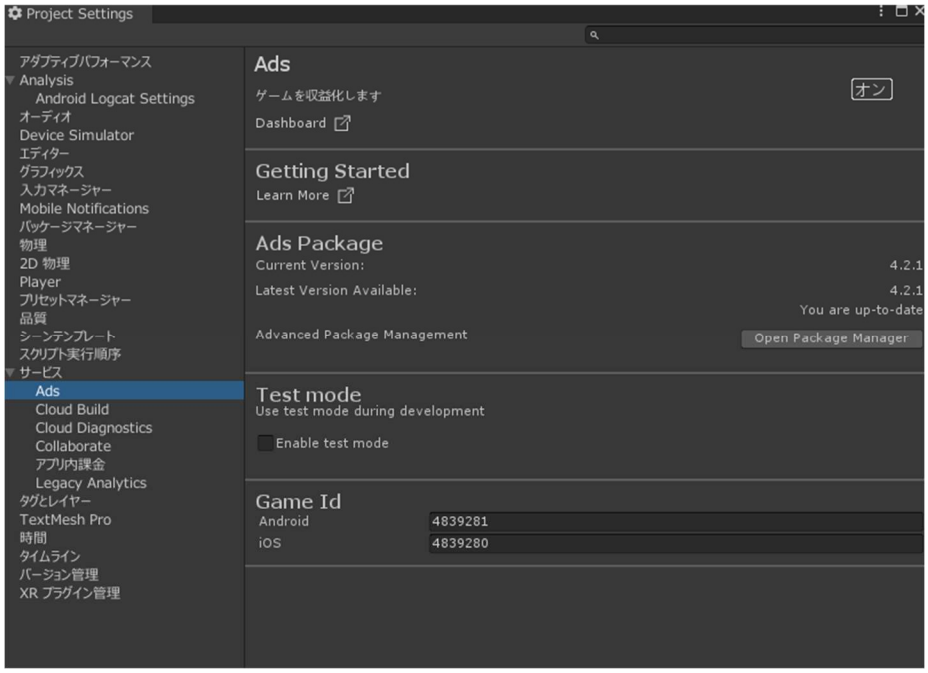


画面中ほどにある「Get Start」をクリックする



ここで、I’m not using Mediation. Only Unity Ads を選択し、「OK」をクリックすると Ad Unit が作成される

ここからは、Unity 側で設定を行う



Current Version が 4.2.1 である事を確認する

なっていない場合、Update する

ここでしばらくすると GameID 自動で設定される

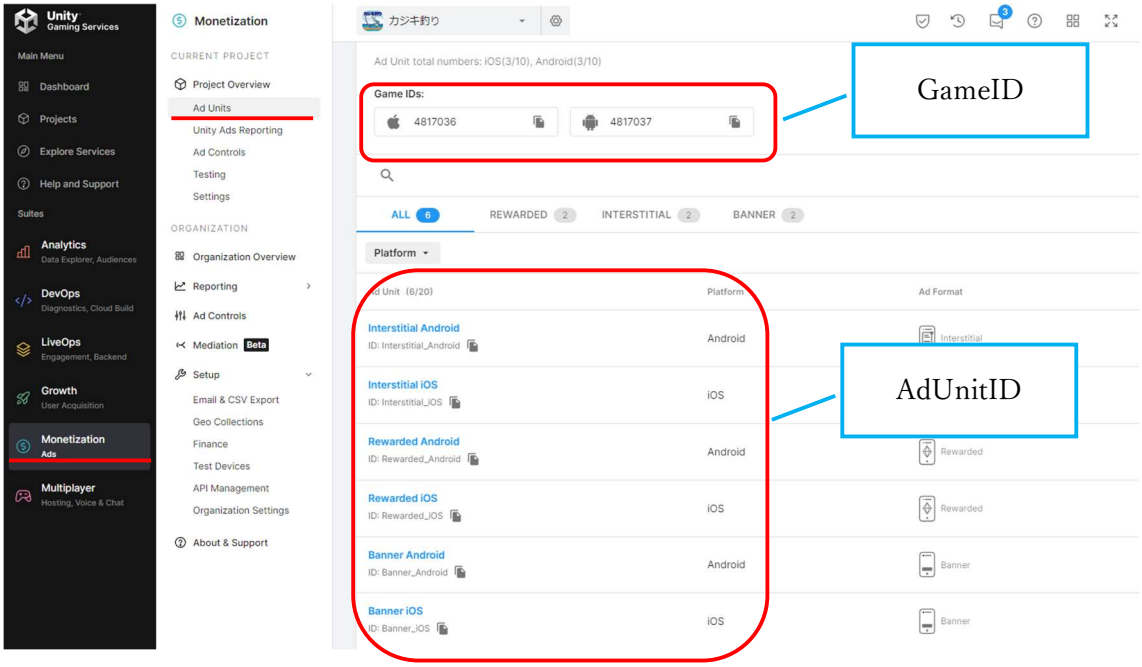
元気玉プロジェクトでは、iOS と Android の Interstitial と Banner を使用する

Interstitial:動画広告

Banner　　:バナー広告

今回使わないが Rewarded というものもある（報酬付き広告）

「Game ID」「Ad Unit ID」はプロジェクトの Unity Dashboard より確認できる

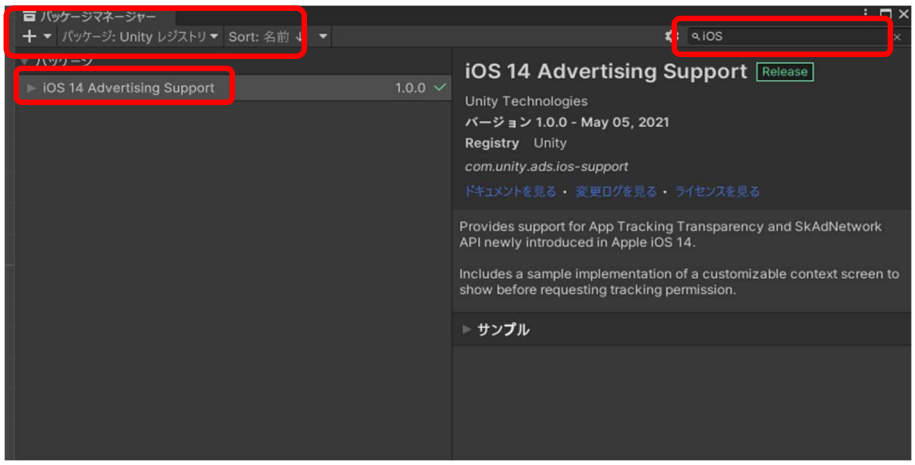


準備が終わったら

１．iOS 用の設定

１．Window メニューの PackageManager を開き、Unity レジストリを選択し、iOS を検索する

表示された「iOS 14 Advertising Support」選択し、インストールする



２．Template より以下のファイルをコピーする

Assets/Editor/XcodeBuild.cs

Assets/Plugins/iOS/IDFADialog.mm

Assets/Plugins/iOS/AdSupport.framework フォルダ毎コピー

Assets/Plugins/iOS/AppTrackingTransparency.framework フォルダ毎コピー

３．プロジェクト内の最初のシーンのスクリプトで以下を追記する

（プロジェクト内で１回定義すればいい）

・最初の using 部分

```
using System.Runtime.InteropServices;
```

・クラス内の変数定義部分

```
#if UNITY_IOS
    [DllImport("__Internal")]
    private static extern void _requestIDFA();
#endif
```

・クラスの Start()部分

```
#if UNITY_IOS && !UNITY_EDITOR
        _requestIDFA();
#endif
```

２．Template より以下ファイルを追加する

Editor/ConstantsClassCreator.cs

Editor/CreateUnityAdsConstants.cs

Editor/UnityConnectSettingsReference.cs

UnityAds/Scripts/AdManager.cs

UnityAds/Scripts/UnityAdsGameIDs.cs

UnityAds/Prefabs/UnityAdsManager.prefab

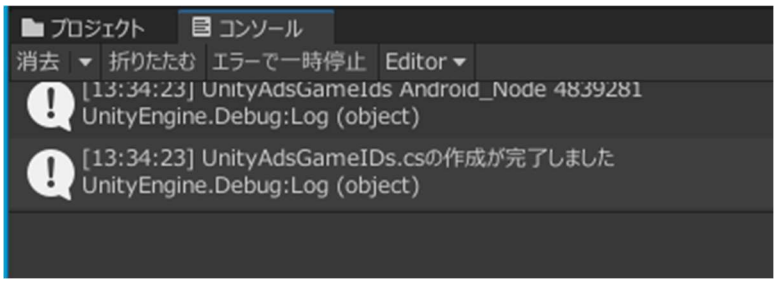
Editor と UnityAds のファルダ名をそのままの Assets は以下にコピーする

YamlDotNet ファルダをそのまま Assets 配下にコピーする

（Ads GameID の設定を簡素化する）

３．ファイル等を追加後、Unity を再起動する

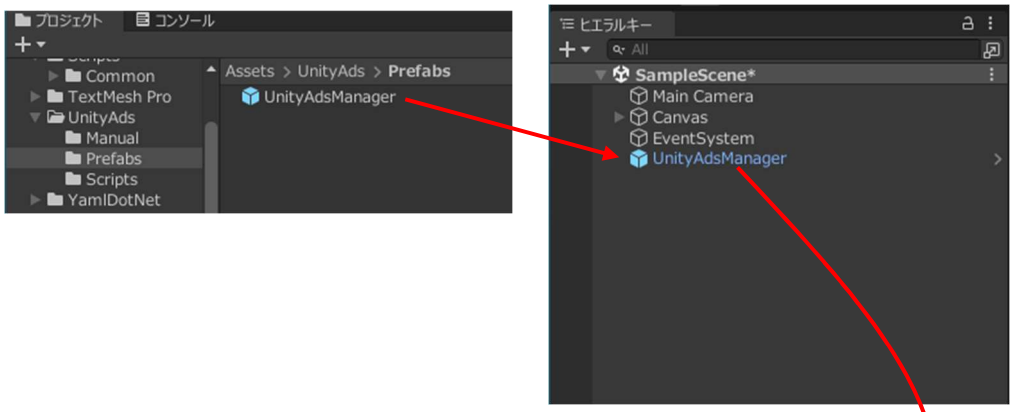
４．再起動後メニューのツール→UnityAdsSetting を押し、コンソールに「UnityAdsGameIDs.cs の作成が完了しました」と表示されることを確認する



５．UnityAds/Scripts/UnityAdsGameIDs.cs が書き換えてあることを確認する

さらに、中身の GameID があっている事を確認する

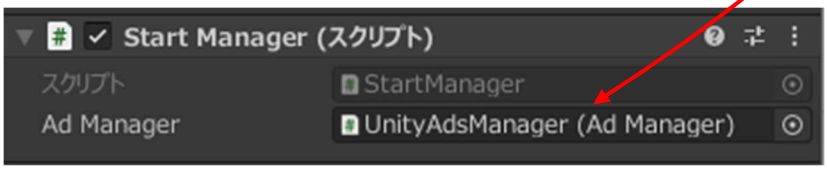
６．広告を表示したいシーンのヒエラルキーに UnityAdsManager.prefab を配置する



広告を表示したいシーに追加する

７．各シーンのスクリプトにて広告表示するコード書き加える

1. クラスの最初で「[SerializeField] AdManager _adManager;」を追記
   AdManager を使用するため、スクリプト保存後インスペクターにて
   ヒエラルキー上の UnityAdsManager をアタッチする



スクリプト名：StartManager.cs の場合、このようになる

2. バナーや動画を表示したかフラグを定義

```
private bool _bannerDisplayed = false;
private bool _interstitialDisplayed = false;
```

3. 実際に広告を表示

3.1. バナー広告

〇Start()に実装する場合

```
if (!_bannerDisplayed)
{
    StartCoroutine(_adManager.ShowBannerAdsAsync());
                                                    コールチンでバナー広告表示
    _bannerDisplayed = true;                        バナー広告表示済にする
}
```

〇Update()に実装する場合

```
                                                IsLoadedBanner でバナー広告の
                                                準備が出来ているか確認
if (!_bannerDisplayed && _adManager.IsLoadedBanner)
{
    StartCoroutine(_adManager.ShowBannerAds());  バナー広告表示
    _bannerDisplayed = true;                      バナー広告表示済にする
}
```

3.2. 動画広告

動画広告は表示させたい箇所で表示させる

ボタンを押したタイミングなど…

```
                                                IsLoadedInterstitial で広告表示の
                                                準備が出来ているか確認
if (!_interstitialDisplayed && _adManager.IsLoadedInterstitial)
{
    _adManager.ShowInterstitialAds();  動画広告表示
    _interstitialDisplayed = true;     動画広告表示済にする
}
```

動画広告もコールチン用と通常があり使い分ける

コールチン：StartCoroutine(_adManager. ShowInterstitialAdsAsync())

通常：_adManager.ShowInterstitialAds();

※注意

動画の場合、コールチンで表示すると表示のタイミングが動画準備出来た段階で表示されるので、ゲーム中にいきなり広告が出たりする可能性がある

動画広告が開始したかどうか判定を「_adManager.AdStarted」で行う

True:表示した

False:表示していない

動画広告を閉じたかどうかの判定を「_adManager.AdClosed」で行う

True:閉じた

False:閉じていない

※通常用とコールチン用で関数名を間違わないよう注意する

エラーになる

※基本的にTemplateよりファイルをコピーする際は、同一フォルダにコピーするようにしてください。

Editorなどビルド時は不要なものもあるので、可能な限り、同一フォルダにコピーする

同一フォルダがない場合、同一名のフォルダ作成する