Read the Docs Template Documentation

Release 1.0

Read the Docs

Apr 16, 2021

CONTENTS

Here you will find information on how to use AdInMo's InGamePlay Brand Advertising SDK including release notes and other information that may help you.

Click here to go back to the AdInMo website.

CHAPTER

ONE

CONTENTS:

1.1 Advanced Placements

This section covers other features that are available with placements. While not a requirement for the AdInMo SDK, we highly recommend you make use of these features.

1.1.1 Debug Size Threshold

The AdInMo SDK comes with a tool that allows developers to ensure the size of their placements within Unity Editor is large enough to generate revenue.

Select the Adinmo Manager in your scene and check **Debug Size Threshold**. A border will appear around your placements. **Green** means you are able to generate revenue from your placement. **Red** means your placement is currently too small.

1.1.2 Placement Groups

Using groups allows you to have multiple placements displaying the same advertisement campaign. Currently, the AdInMo SDK supports up to 10 groups.

Once a group has been created, placements can be assigned to the group. The option to create/edit groups can be found either when creating a new placement or by editing an existing one.

1.2 Advanced

From here you can add more functionality to the AdInMo SDK within your game. This section contains a range of functions that you can add and will require editing C# scripts.

1.2.1 Extra Revenue

Between levels, call the function below to earn extra revenue. See the example scene called *ExampleDialog.unity* to see how to call it and register a callback to know when this dialog is done.

```
AdinmoManger.ShowDialog()
```

1.2.2 Callback Function

For added control, there is a callback function you can register. This is useful for hiding an object until the replacement textures have been downloaded and applied, thus avoiding a visual pop.

```
AdinmoManager.SetOnReadyCallback()
```

Your delegate function must be of the form:

```
void MyDelegate( string message );
```

1.2.3 Background Colour

If the placement doesn't fill the object its applied to, you may need to alter the background colour. We provide a call to get the background colour set by the advertisement so you can update the object as required.

```
Color AdinmoTexture.GetBorderColor()
```

1.2.4 Rotate Textures

You can rotate which textures are displayed on which placements by calling the following function, e.g. between levels.

```
void AdinmoManager.CycleTextures()
```

1.2.5 Temporarily Disable AdInMo Processes/Traffic

If you want to temporarily disable all AdInMo processing and network traffic, you can call:

```
AdinmoManager.Pause()
```

To resume normal operations, call:

```
AdinmoManager.Resume()
```

1.2.6 Dynamic Placements

AdInMo supports the concept of dynamic ads, these are ad placements that are created programmatically rather than through the Unity editor.

This allows AdInMo to support situations where a game developer programmatically creates objects in the game, our SDK supports the concept of then creating the ad unit programmatically once the object has been created.

Here's a simple example of how this works using the AdInMo Unity SDK.

1. Create a game object (image, quad or sprite) adding the 'Adinmo Texture' component with a placement key.
2. Next, create a prefab of the placement game object by dragging it into the assets window and removing the original from the hierarchy.
3. Copy and paste the code below these steps into a new C# script, and save it.
4. Create an empty game object and add the script onto it.
5. Now select and drag the placement prefab (step 2) into the the example script.

```
using UnityEngine;

public class InstantiationExample : MonoBehaviour
{
    // Reference to the Prefab. Drag a Prefab with an AdInMo Texture component␣
↪attached to it into this field in the Inspector.

    public GameObject myAdinmoPrefab;

    // This script will simply instantiate the Prefab when the game starts.

    void Start()
    {
        // Instantiate at position (0, 0, 0) and zero rotation.

        Instantiate(myPrefab, new Vector3(0, 0, 0), Quaternion.identity);
    }
}
```

1.3 Full Release Notes

Here you will find a full list of release notes for the AdInMo SDK. To download the most recent version of the AdInMo SDK, click here .

1.3.1 Version 1.5.2 (02/10/2021)

- Startup time improvements, pause reduction.
- Improve image cache performance.
- Better, more careful logging.
- Remove unused textures in more cases.
- Fixed bugs in logic when there are no image candidates available for some reason.

Full List

1.3.2 Version 1.5

Version 1.5.0 (01/12/2021)

- Improve image cache cleanup logic.
- Make downloaded textures non-readable, which further improves the memory usage.
- Other minor fixes.

1.3.3 Version 1.4

Version 1.4.11 (12/29/2020)

- Fixed a corner case upon old textures cache cleanup.
- Improved error handling in a few cases.
- Other minor stability improvements.

Version 1.4.10 (12/16/2020)

- Hotfix for image cache duplicate key issue.

Version 1.4.9 (07/04/2020)

- Performance updates.

Version 1.4.8 (11/25/2019)

- Only allow one AdinmoManager to exist in the scene. Duplicates are auto-destroyed.

Version 1.4.7 (11/11/2019)

- Properly handle AdInMoTextures getting destroyed.

Version 1.4.6 (10/9/2019)

- Fixed frame rate hitches during Sprite. Create() at a app load time or when a texture cycles for the first time.

Version 1.4.5 (10/5/2019)

- Corrected screen space computation for Canvases with Render Mode set to Screen Space – Camera and a Render Camera specified.
- Fixed order of vertices when drawing Debug Size Threshold on Quads.

Version 1.4.4 (9/26/2019)

- Fixed an internal reporting-only bug.

Version 1.4.3 (9/19/2019)

- Fixed a bug where AdInMoManager.IsFilled() could return false when it should actually have returned true.

Version 1.4.2 (9/8/2019)

- Fixed an issue introduced in 1.4 that caused the AdInMo plugin to silently fail in some IL2CPP builds.
- There is now an included file Scripts/AdInMoCompatibility.cs to keep backward compatible reflected functions from getting dead-stripped in IL2CPP.

Version 1.4.1 (9/4/2019)

- Removed checkbox from AdinmoManager that if unchecked could reduce revenue

Version 1.4 (8/22/2019)

- If the server connection fails at app startup because it was choked out by other services, a retry will be attempted 5 seconds later.
- Added AdinmoManager.Pause() and AdInMoManager.Resume() to temporary all AdInMo processing and network traffic.
- To maximize your revenue, ads now cycle automatically every several seconds rather than waiting for you to all AdInMo.CycleTextures().
- Added AdinmoManager.IsFilled() which will return false in the rare case that no ads were filled.
- Added a camera override to each AdInMoTexture in case they are drawn with a different camera from your main scene.

1.3.4 Version 1.3

Version 1.3.4 (7/8/2019)

- Removed unnecessary Asset Store folder from package.

Version 1.3.3 (7/6/2019)

- Fixed a crash in AdinmoSender.GetServerConfig() if too many other web services are running from a game and AdInMo gets starved out.
- Fixed EncodeToPNG error occurring in certain version of Unity.
- Fixed other minor backward compatibility warnings with certain Unity versions.
- Lowest supported version is Unity now 2017.1.

Version 1.3.2 (5/10/2019)

- Fixed coverage computation when UI.Image is used on canvas with Canvas.renderMode.WorldSpace.

Version 1.3.1 (4/9/2019)

- Don't show dialog with blank choices in offline mode.

Version 1.3 (3/16/2019)

- Added AdinmoManager.ShowDialog() for extra developer revenue (see documentation).
- Improved image caching to further minimize run-time download bandwidth.
- Show API out of date message in editor window.
- Some under-the-hood bug fixes.

1.3.5 Version 1.2

Version 1.23.2 (1/31/2019)

- Fixed placement size computation for images with a parent Canvas set to "Screen Space — Camera" and it's camera is not "None".

Version 1.23.1 (1/11/2019)

- Fixed an editor only bug where the Scene camera was used for "Debug Size Threshold" rather than the game's main camera when the Scene window was used.

Version 1.23 (1/2/2019)

- AdinmoManager checkbox added for "Debug Size Threshold." This puts a debug border around Placements. Green means it is big enough to pay for an add. Red means it's too small.
- AdinmoManager checkbox added for "Development Build" to avoid sending impressions during test builds that checked as "Release" through Unity.
- Compatibility confirmed with Unity 2018.3.0.

Version 1.22 (12/4/2018)

- Added checkbox to AdInMoTexture to disable rendering until the ad is ready, thus reducing the need for programming a callback.
- Strip spaces from Placement Key and Game Key to reduce cut-and-paste errors.
- Compatibility confirmed with Unity 2018.2.18.

Version 1.21 (11/16/2018)

- Unity backward compatibility support. Tested back to Unity 5.6.1f.
- Simplified AdInMoManager prefab to a single game object with a single component.

Version 1.2 (10/16/2018)

- Added public Color AdInMoTexture.GetBorderColor() to allow developers to choose matching backgrounds.
- Optimized network bandwidth.

Version 1.1 (9/11/2018)

- Added offline caching mode which allows developers to continue to earn money on impressions while the player is in airplane mode or a network connection is unavailable.
- Bug fixes and performance improvements.

Version 1.0 (5/13/2018)

- Initial Commercial Release.

1.4 SDK Walkthrough

SDK Version 1.5.2

Oldest Supported version of Unity 2018.4

Our walkthrough goes through everything step-by-step to get the AdInMo SDK working with your game project within Unity.

Table of Contents

1.4.1 Register with AdInMo

To begin using the AdInMo SDK, an AdInMo developer account is required.

1. Go to AdInMo Website
2. Click **Login/Sign Up button**, top-right of the page.
3. Sign up using your **email** or with your **Google account**.
4. Fill in the appropriate details and confirm your account using the instructions given. If signing up via email you will be asked to confirm your account.

1.4.2 Portal: Creating a Game

A **Game** is a representation of your game within AdInMo. During this process you will create a **Game Key** that is required to work with the AdInMo SDK, **please keep this information private**.

1. Login to your developer account on AdInMo.
2. Click the **+ Add New Game** button.
3. Fill in the details of your game - these details can be edited at any time - and click **Save**.
4. Each game has its own **game key**, this can be viewed on the details page of a game. **You will need this game key later on in the walkthrough**.
5. To view the details of your game, simply click the name of the game you want to view, either on the overview or game list page.

1.4.3 Portal: Creating a Placement

This part requires at least one game on the portal.

Placements are simply an object which will display an advertisement, they must contain a unity 2D texture. Placements can be grouped together to show the same advertisement across different placements. One game can support multiple placements.

During this process you will create a **Placement Key** that is required to work with the AdInMo SDK, **please keep this information private**.

1. View the game you wish to create a placement for on the portal.
2. Click the, **+ Add New Placement** button, if you already have any placements they can be viewed here.
3. Fill out the details of your placement and select an aspect ratio. Placements can be edited at any time.
4. To create a group, click **New Group**. Once a group has been created it can be edited by clicking **Edit Group**. By default, a placement is set to have no group.
5. Each placement has its own **placement key**, this can be found on the placement list. **You will need this placement key later on in the walkthrough**.

1.4.4 Download AdInMo SDK

Oldest Supported version of Unity 2018.4

1. Navigate to the SDK Download page.
2. Click Download SDK.
3. Create or open a project within Unity Editor. If downloading from the Unity Asset Store, you can either import from the package manager or through the asset store tab depending on the unity version.
4. From the top menu, click **Assets**, **Import Package**, **Custom Package**.
5. Select the unity package you downloaded in step 2.
6. Ensure all the items are selected and click **Import**.
7. You should now see a folder called **Adinmo** displayed in the Unity Assets window.

1.4.5 Unity Editor: AdinmoManager

A **game key** is required for this part.

First, the **AdinmoManager** needs to be added to the scene. Next, copy and paste the **game key** of your game into the **AdinmoManager**.

Please ensure development mode is unchecked on the AdinmoManager before going live, otherwise impressions won't count.

1. From the Assets window, select **Adinmo**, **Prefabs**.
2. Click and drag the prefab called **Adinmo Manager** into the scene hierarchy.
3. **If you are working on a development build, please check this box on the Adinmo Manager.**
4. Navigate back to the portal and to the appropriate game.
5. View the details of the game on the portal and copy the game key (starts with gk_...).
6. In Unity Editor, paste the copied **game key** in the AdinmoManager in the **game key** field.

1.4.6 Unity Editor: Adding AdinmoTexture Component

A **Placement key** is required for this part.

Next step is to add the **Adinmo Texture** component to the game object that will act as the advertising placement within your game.

1. Create a game object, either an **image**, **quad** or **sprite**.
2. Select the game object in the scene and click **Add Component** in the inspector window.
3. In the search box, type and select **Adinmo Texture**.
4. Paste the **placement key** of the placement from the portal into the **Adinmo Texture** component, Placement keys start with pk_....

Note: each placement requires its own placement key and can be grouped if needed.

1.4.7 Finished!

Run your game and advertisements will appear in your game.

For support, please check our FAQ.

Please Note!

You will not be able to see impressions using a development build, uncheck this option if you are building a release.

Unity Editor also has its own development build option under **File**, **Build Settings**. Ensure the **Development Build** option is not checked when building for release.

CHAPTER

TWO

GOT A QUESTION?

We've shared some of our most frequently asked questions to help you find solutions to any questions you may have.