

Swarm Unity Integration Guide

Welcome to the **Swarm Unity Integration Guide**!

This guide provides step-by-step instructions for quickly integrating Swarm into a Unity Android game or app.

(Note: The current version of the Swarm Unity Plugin is only for Android and it will not work on other platforms. Also, since the Swarm Unity Plugin relies on deeper parts of Android, it won't work in the Unity Editor and must be run on an Android device.)

In just minutes, you'll be able to **Setup**, add **Leaderboards**, **Achievements**, **Cloud Data**, and a **Store**. Swarm is a very flexible and scalable platform. Each feature can be used independently of the others, and each feature automatically scales to match your needs - whether your game has tens or millions of players. You choose what feature(s) to use and when to use them. The Swarm Unity Plugin also includes pre-built screens and dialogs that save you valuable time.

To use the Swarm Unity Plugin, first follow the **Setup** section of this guide. After completing the **Setup**, you can go to any section in the guide to implement any feature(s) you wish.

Table of Contents:

I.	Setup	2
II.	Leaderboards	5
III.	Achievements	7
IV.	Cloud Data	9
V.	Store	10
VI.	Publishing	15
VII.	FAQs & Pro Tips	15
VIII.	Help	15

I. Setup

Estimated Integration Time: 15 minutes

In this section, we show you how to get started, setup Swarm, and highlight some of the features available to you. After completing these steps, you'll be ready to add **Leaderboards**, **Achievements**, **Cloud Data**, a **Store**, and more!

Prerequisites:

- Unity3d Software + Android (<https://store.unity3d.com>)
- Android SDK (<http://developer.android.com/sdk/index.html>)
- Swarm Unity Plugin (<http://swarmconnect.com/admin/docs/sdk>)

Setup Your App in the Swarm Admin Panel:

If you do not already have a Swarm account, follow these instructions...

1. Create an account on the Swarm website (<http://swarmconnect.com/register>)
2. Follow the instructions on the website to create your first app.
3. After naming your app, click on the Documentation link.
4. From the Documentation screen, click on "Your Apps" at the top of the screen.
5. Click on the name of your app.
6. Click on the App Details button at the top of the Swarm Admin Panel.
7. Write down the App ID and App Key (you'll need them soon).

If you do already have a Swarm account, then follow these instructions instead...

1. Go to the Admin Panel (<http://swarmconnect.com/admin>) and click on "Create a New App".
2. Name the App and click on Submit.
3. Click on the App Details button at the top of the Swarm Admin Panel.
4. Write down the App ID and App Key (you'll need them soon).

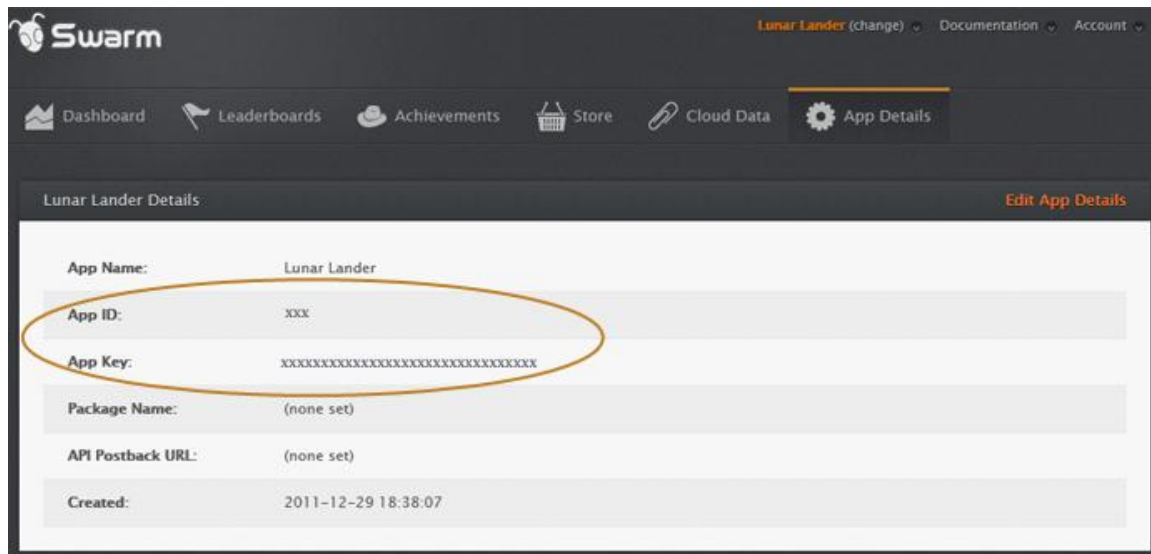


Figure 1: Swarm Admin Panel - App Details shows the App ID and App Key

Import Swarm Unity Plugin:

In Unity, import the SwarmUnityPlugin.unitypackage file by clicking on Assets > Import Package > Custom Package... and then choosing the SwarmUnityPlugin.unitypackage file. On the “Importing package” popup, be sure to import all items by clicking on “All” and then on “Import”.

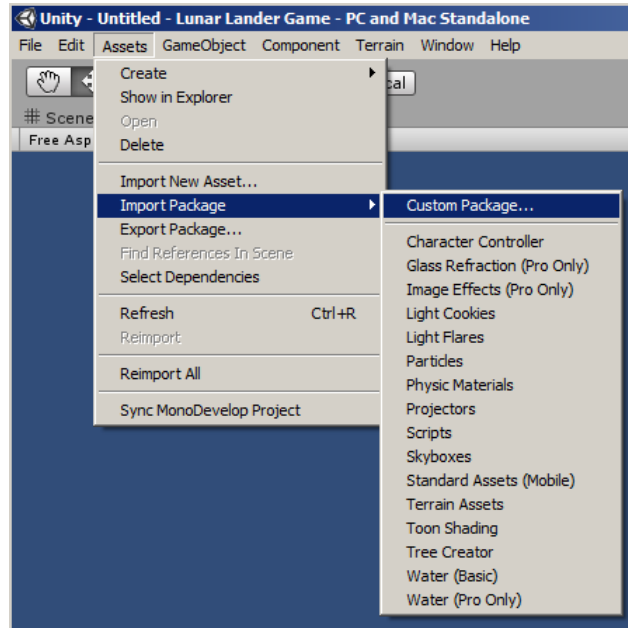


Figure 2: Importing the SwarmUnityPlugin.unitypackage

Initialize Swarm in Unity:

Initializing Swarm will prompt a new player to create an account or login as a guest. After the player has logged in for the first time, they will not be required to login again. This makes initialization seamless for your players.

You can choose when to initialize Swarm, but keep in mind that features will not work unless Swarm has been initialized first. For example, this line of code could be attached to a button or called inside of the Start() function.

To initialize Swarm, add the following line of code to your C# script.

```
Swarm.init(AppId, AppKey);
```

Replace AppId with the integer shown beside “App ID” in your Swarm Admin Panel. Replace AppKey with the string shown beside “App Key” in your Swarm Admin Panel.

Add a Login Listener:

Adding a login listener enables your code to know when a player has successfully logged in, started the login process, cancelled the login, or logged out. This code should be called prior to calling init.

To add a login listener, add the following line of code to your C# script.

```
SwarmLoginManager.addListener(delegate(int status) {
```

```

        if (status == SwarmLoginManager.USER_LOGGED_IN) {
            // The player has successfully logged in
            // For example, you may wish to call Application.LoadLevel("MyLevel");

        } else if (status == SwarmLoginManager.LOGIN_STARTED) {
            // The player has started logging in

        } else if (status == SwarmLoginManager.LOGIN_CANCELED) {
            // The player has cancelled the login

        } else if (status == SwarmLoginManager.USER_LOGGED_OUT) {
            // The player has logged out

        }
    });

```

Update the AndroidManifest.xml

The Swarm Unity Plugin comes with an AndroidManifest.xml file that will need to be merged into your own AndroidManifest.xml.

1. Replace “com.company.product” with your package name
2. Replace “@drawable/icon” with the path to your icon (if it is different)
3. Replace “Swarm Unity Demo” with your game’s label
4. Make other modifications if needed and only where specified by the comments

Congratulations!

Your game now has the basic infrastructure required to start adding social features. You can add [Leaderboards](#), [Achievements](#), [Cloud Data](#), and a [Store](#)! All features are completely modular and can be added in any order. Feel free to jump to any feature in this guide.

II. Leaderboards

Estimated Integration Time: 10 minutes

Leaderboards are a great way to get players to compete with their friends and the rest of your community, and they have been shown to increase the number of player sessions while giving players a reason to keep getting better.

Leaderboards track scores as floats and can be sorted as Descending (high to low) or Ascending (low to high).

- Descending Examples: Points, Survival Time
- Ascending Examples: Golf Scores, Racing Times

In this section we show you how to add a leaderboard. However, you may add as many leaderboards as you wish.

Prerequisite: Swarm [Setup](#)

Step 1: Define Leaderboards in the Swarm Admin Panel

1. Go to your Swarm Admin Panel (<http://swarmconnect.com/admin>)
2. Click on your game's name in the list.
3. Create the leaderboard and click on Submit.

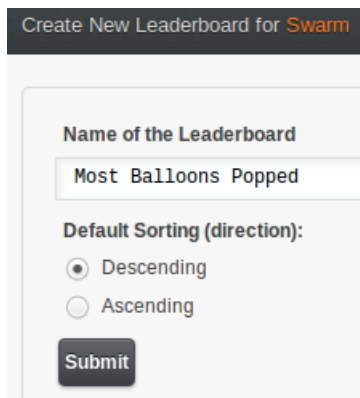
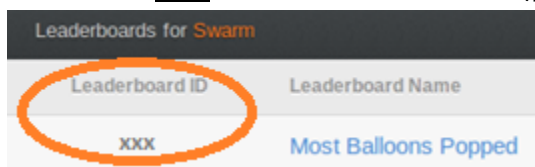


Figure 3: Creating a Leaderboard

4. Write down your leaderboard's ID number (you'll need it soon).



Leaderboard ID	Leaderboard Name
XXX	Most Balloons Popped

Figure 4: The leaderboard ID is shown in the orange oval.

Step 2: Setup Leaderboards in Your Game

To submit a score to a leaderboard, use the following line of code (in your C# script) where `leaderboardID` is your leaderboard ID number and `score` is the float value of the score you wish to submit.

```
SwarmLeaderboard.submitScore(leaderboardId, score);
```

To display a single leaderboard, use the following line of code (in your C# script) where leaderboardID is your leaderboard ID number.

```
SwarmLeaderboard.showLeaderboard(leaderboardId);
```

To display the list of all leaderboards, use the following line of code (in your C# script).

```
Swarm.showLeaderboards();
```

To submit a score to a leaderboard and then immediately show that leaderboard, you must use the following line of code (in your C# script).

```
SwarmLeaderboard.submitScoreAndShowLeaderboard(leaderboardId, score);
```

Congratulations!

You now know how to add leaderboards to your games. It's pretty easy, isn't it? Have you added [Achievements](#), [Cloud Data](#), or a [Store](#) yet? They all offer more excellent ways to improve your games!

III.Achievements

Estimated Integration Time: 10 minutes

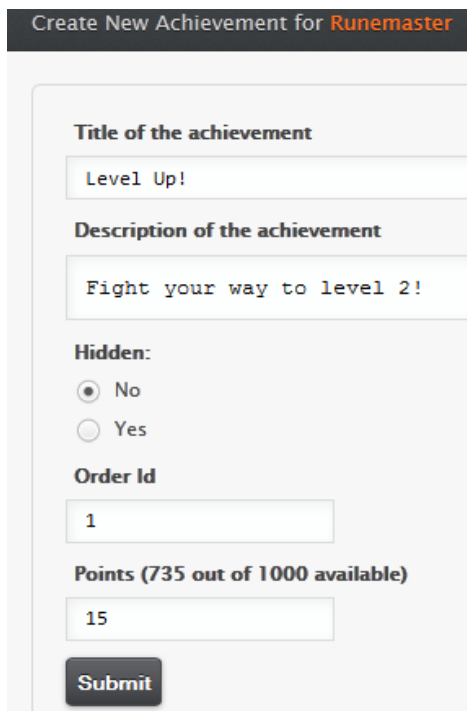
Achievements are an easy way to entice players to reach for interesting goals. Do you want to encourage your players to get to level 3? Add an achievement! Do you want your players to try out every level? Add an achievement! You get the idea.

In this section we show you how to add an achievement. However, you may add as many achievements as you wish.

Prerequisite: Swarm [Setup](#)

Step 1: Define Achievements in the Swarm Admin Panel

1. Go to your Swarm Admin Panel (<http://swarmconnect.com/admin>)
2. Click on your game's name in the list.
3. Create the achievement and click on Submit.



Create New Achievement for **Runemaster**

Title of the achievement
Level Up!

Description of the achievement
Fight your way to level 2!

Hidden:
☒ No
☐ Yes

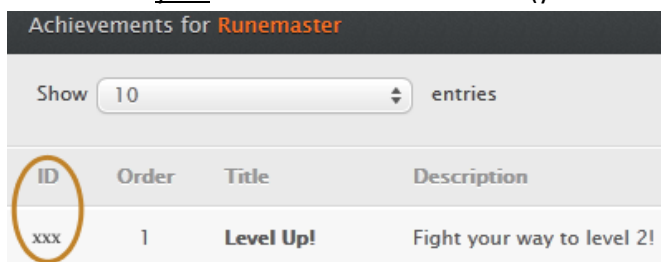
Order Id
1

Points (735 out of 1000 available)
15

Submit

Figure 5: Creating an Achievement

4. Write down your achievement's ID number (you'll need it soon).



Achievements for Runemaster			
Show 10 entries			
ID	Order	Title	Description
xxx	1	Level Up!	Fight your way to level 2!

Figure 6: The achievement ID is shown in the orange oval.

Step 2: Setup Achievements in Your Game

To unlock an achievement, use the following line of code (in your C# script) where achievementID is your achievement ID number.

```
SwarmAchievement.unlockAchievement(achievementID);
```

To display the list of all achievements, use the following line of code (in your C# script).

```
Swarm.showAchievements();
```

To unlock an achievement and then immediately show the list of achievements, you must use the following line of code (in your C# script).

```
SwarmAchievement.unlockAchievementAndShowAchievements(achievementID);
```

Congratulations!

You now know how to add achievements to your games. It's not too tough, right? Have you added [Leaderboards](#), [Cloud Data](#), or a [Store](#) yet? They all offer more great ways to improve your games!

IV. Cloud Data

Estimated Integration Time: 5 minutes

Cloud Data provides an easy way to save and load data across multiple devices and installs. Think of Cloud Data as an online version of Android's SharedPreferences, where you can set and get data, and have it saved to the cloud in a seamless way. Cloud Data can be used to store level progress, configuration options, favorites, bookmarks, or anything else you might need.

In this section we show you how save data to the cloud and retrieve it back again. Cloud data is stored and retrieved using key, value pairs.

Prerequisite: Swarm [Setup](#)

Step 1: Store Cloud Data

To save data to the cloud, use the following line of code (in your C# script) where key is a String used to identify the data, and data is a String representation of the actual data you wish to save to the cloud.

```
SwarmActiveUser.saveUserData(key, data);
```

Since data is stored on a per-player basis, the player must be logged in to save data to the cloud.

Step 2: Retrieve Cloud Data

To get data from the cloud, use the following line of code (in your C# script) where key is the String used to identify the data.

```
SwarmActiveUser.getUserData (key, delegate(string responseData) {  
  
    // The data will be asynchronously returned in the responseData String.  
    if (responseData != null) {  
        // use responseData  
    }  
});
```

Swarm automatically caches the cloud data efficiently, because a player may not always have a reliable connection to the internet. If the player is offline, cached versions of the data will be returned during get requests, and updated on both the server (and in the cache) on save requests. However, it is possible for a request to fail. If the player is offline and the data has never been requested when the player was online, cloud data get requests can return null.

Congratulations!

That's all there is to making use of cloud data storage! Super simple stuff, right? Have you added [Leaderboards](#), [Achievements](#), or a [Store](#) yet? They all offer more great ways to improve your games!

V. Store

Estimated Integration Time: 20 minutes

The store provides an easy way to monetize your app through selling virtual goods to your users. Virtual goods can be anything you want, from removing an advertisement banner in a free app to the sale of a new sword that can be used to fight off enemies.

Purchasing items can be done through the store screens or directly in your app via a popup dialog. All purchases that a player makes are automatically managed in the player's inventory. Items sold can be consumable (something that is destroyed on use, like a health potion) or non-consumable (like a sword, hat, or even ad-banner removal).

When players purchase items from your store, you (as the developer) can earn real money based on how many Swarm Coins they spend. For more details, please read the Terms of Service (TOS) at <http://swarmconnect.com/terms> regarding Swarm Coins.

Prerequisite: Swarm [Setup](#)

Terminology:

- **Item** - An Item is something that the player can own. Items are sold to a user by using Store Item Listings (described below), and added to their inventory. An Item could be a sword, ad-removal, fish, or just about anything you can imagine.

An Item can be consumable or non-consumable. A consumable item is removed from the user's Inventory when it is used, but a non-consumable item remains in the player's Inventory when it is used. Examples of consumable items may be things like potions and food whereas examples of non-consumable items may be things like swords, hats, and ad-banner removal.

An Item can contain a payload. A payload is a developer-defined String of data to be delivered to the app when a user purchases an Item. Example: an array of parameters for a Sword: "damage=10, speed=5".

- **Store Item Listing** – A Store Item Listing is a way for you to list Store Items in Store Categories. For example, a Store Item Listing may contain the Item "Sword" in category "Weapons" for a price of 10 Swarm Coins.

Store Item Listings have a quantity to allow you to sell multiple of an Item in a single purchase. For example, you could choose to sell 1 Health Potion for 10 Swarm Coins or 5 Health Potions for 40 Swarm Coins.

- **Store Category** - A Store Category is a logical block of Store Item Listings. A Store Category is a grouping of things to be sold together. For example, a Store Category could be "Weapons", "Armor", or "Hats".

Step 1: Configure Your Store

What you sell in your store is completely up to you. For this example, we will be selling "Health Potions", which will allow players to heal themselves by restoring 200 health points.

1. Go to your Swarm Admin Panel (<http://swarmconnect.com/admin>)

2. Click on your game's name in the list.
3. Create click on the Store button at the top.
4. Click on "Create a Category"

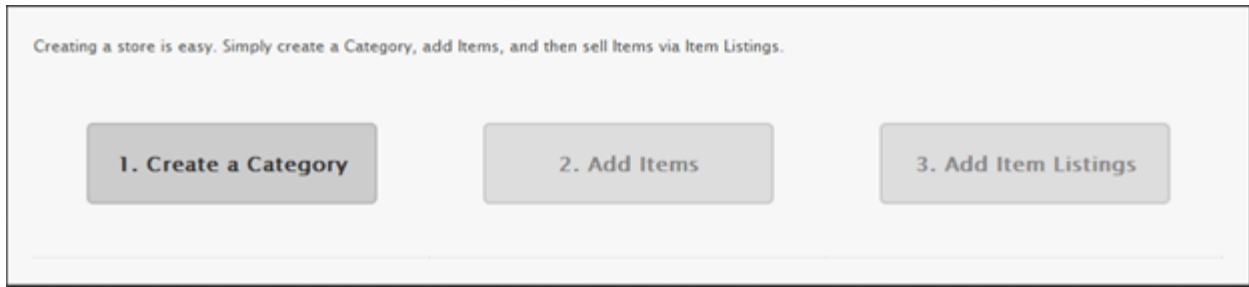


Figure 7: Setting up a Store

5. Name the Store Category and click on Submit.

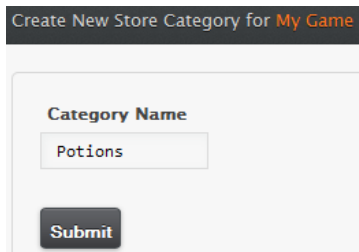


Figure 8: Creating a Store Category

6. Click on the Store button at the top and then on Add Items.
7. Create an Item.

Create New Store Item for My Game

Item Name

Health Potion

Consumable

If an Item is consumable, then 1 of the Item will be removed from the user's inventory when the Item is used.

☒ Yes

☐ No

Item Description

Restores 200 health points!

Data Payload (hidden from users)

A payload that can be used for storing extra information.

200

Submit

Figure 9: Creating a New Store Item

8. After you create the new Item, write down the Item's ID number (you'll need it soon).

9. Click on the Store button at the top and then on Add Item Listings.

10. Create a Store Item Listing to enable players to buy a single health potion and click on Submit.

New Store Item Listing for **My Game**

Listing Title

Category

Item

Icon
The icon must be a .png image with width 80px and height 80px.

Price (in Swarm Coins)
When a user buys this Item Listing you will earn \$0.01 per Swarm Coin.

Quantity

Order Id
Index of the Item Listing in its Category (1 = first).

Status
Active Item Listings will be visible to users.
☒ Active
☐ Inactive

Show to Versions:
From version (inclusive), through version (inclusive)

Figure 10: Creating a new Store Item Listing

11. To enable players to buy 5 potions at a time, you can create another Store Item Listing and replace the Price with 40 (to give players a discount for buying in bulk) and the Quantity with 5. In your game, you can choose your own prices and quantities.

12. Write down the Store Item Listing Id numbers (you'll need them soon)



Store item listings for Swarm				
Id	Thumb	Title	Quantity	Price
XXXX		Health Potion	1	\$ 10
XXXX		Health Potions 5 Pack	5	\$ 40

Figure 11: Store Item Listing ID numbers are circled above in orange

Step 2: Sell the Store Item Listings

- Option 1 – Sell through the Store front – To display your entire Store to the player, call the following line of code (in your C# script):

```
Swarm.showStore();
```

- Option 2 – Sell a Store Item Listing directly – To sell a specific Store Item Listing to a player, call the following line of code (in your C# script):

```
SwarmStore.purchaseItemListing(idnum, delegate(int responseData) {  
    // responseData = 1 on purchase success, 0 on purchase failure  
});
```

Step 3: Check if a Player has an Item

To check if a player has a particular Item in his/her inventory, call the following line of code (in your C# script) where itemId is the integer value of the Item ID number shown in your Swarm Admin Panel. Note that the Item ID number is different than the Store Item Listing ID number.

```
SwarmUserInventory.getItemQuantity(itemId, delegate(int responseData) {  
    // The quantity of the item in the player's inventory will be  
    // returned asynchronously as an int in responseData  
});
```

Step 4: Use Consumable Purchased Items

When a consumable item is consumed, it reduces the quantity of the item in the player's inventory by 1. To consume an item, call the following line of code (in your C# script) where itemId is the integer value of the Item ID number shown in your Swarm Admin Panel. Note that the Item ID number is different than the Store Item Listing ID number.

```
SwarmUserInventory.consumeItem(itemId);
```

Congratulations!

That's all there is to setting up your Store! Now your players can start purchasing virtual goods, and you can start monetizing your app in a whole new way. Have you added **Leaderboards**, **Achievements**, or **Cloud Data** yet? They all offer more great ways to improve your games!

VI. Publishing

(Recommended) Enhance Your Game's Icon

Add a Swarm badge to your app icon so that users know at first glance that your game has the social features they want! Get badges at <http://swarmconnect.com/admin/marketing>.

(Recommended) Celebrate Your Game's Launch

Let us know when you're ready to launch your game, and we might help give you a little extra promotional boost to celebrate the launch. You can reach us by email at support@swarmconnect.com.

VII. FAQs & Pro Tips

(Optional) Controlling Swarm's Notifications & Toasts

By default, Swarm automatically displays messages to tell users when they've reached a new high score on a leaderboard, unlocked an achievement, or successfully purchased an item from your store. However, there are times when you may wish to control the display of these messages.

You can disable Swarm's notifications & toasts by calling the following line of code:

```
Swarm.disableNotificationPopups();
```

You can enable Swarm's notifications & toasts by calling the following line of code:

```
Swarm.enableNotificationPopups();
```

(Optional) Getting the username of the currently logged in user

To get the username of the currently logged in user, use the following line of code:

```
SwarmActiveUser.getUsername();
```

(Optional) Showing the Swarm Dashboard

The Swarm dashboard offers players with quick access to all of the Swarm features. Use the following line of code to display the Swarm dashboard:

```
Swarm.showDashboard();
```

VIII. Help

Don't worry if you get stuck! Our friendly, helpful support team is available to answer your questions. Simply send an email to support@swarmconnect.com and someone will be with you shortly.

