

November 20, 2017

Visual guide for Unity3D Newgrounds API

Hello there game developer!

My name is Albert (GeckoMLA19097 on Newgrounds) and I created this visual guide with the intention to help you with the first time setup of the Newgrounds Unity3D API.

This visual guide is more of an explanation of the steps that already exist for the Unity3D API but from a beginner's point of view and it is intended (more specifically) to help you with the medal and scoreboards systems only.

However (and as a little disclosure) as I'm a beginner maybe this guide is lacking depth, in general I feel I got really lucky to find great examples and references to implement the system in my own project. So if you have real trouble implementing it I believe it is best contacting the API developer Josh Tuttle a.k.a PsychoGoldfish <https://psychogoldfish.newgrounds.com/> or search the forums for help. These are good places for a start:

Official guide made by Josh: <https://bitbucket.org/newgrounds/newgrounds.io-for-unity-c/>

Forum posts with code examples: <https://www.newgrounds.com/bbs/topic/1425182>

By the way if you are curious about the result I got, here is my game:

<https://www.newgrounds.com/portal/view/701764>

This guide is divided in four parts:

A. Creating your game project on Newgrounds.

(Same instructions as <http://www.newgrounds.io/get-started/>)

B. Adding the API to your Unity3D game.

(I reference some of the instructions in <https://bitbucket.org/newgrounds/newgrounds.io-for-unity-c/> and add some extra explanations.)

C. Adding medals / scoreboards on Newgrounds and initial code in Unity3D.

(Sample code is taken from <https://bitbucket.org/newgrounds/newgrounds.io-for-unity-c/> and from <https://www.newgrounds.com/bbs/topic/1425182>)

D. Questions I had while implementing the API medals and scoreboards.

**Please note that you still will need some amount of experience with the Unity3D environment and coding in C# otherwise it can get confusing.*

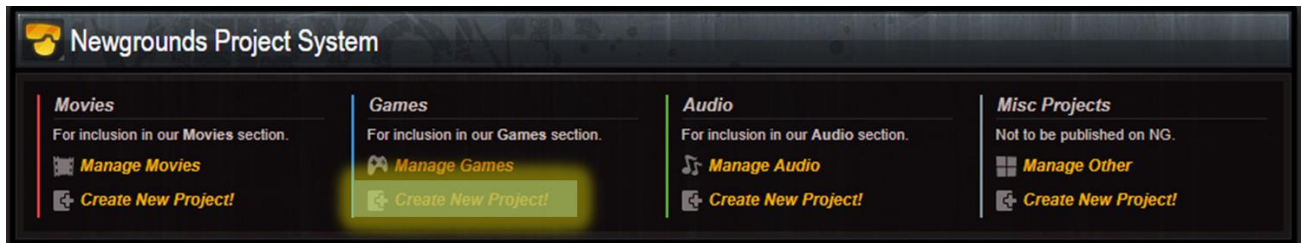
November 20, 2017

A. Creating your game project on Newgrounds:

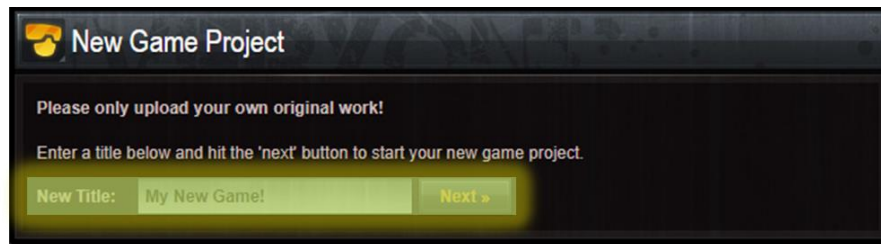
- Navigate to <http://www.newgrounds.com/projects/games>



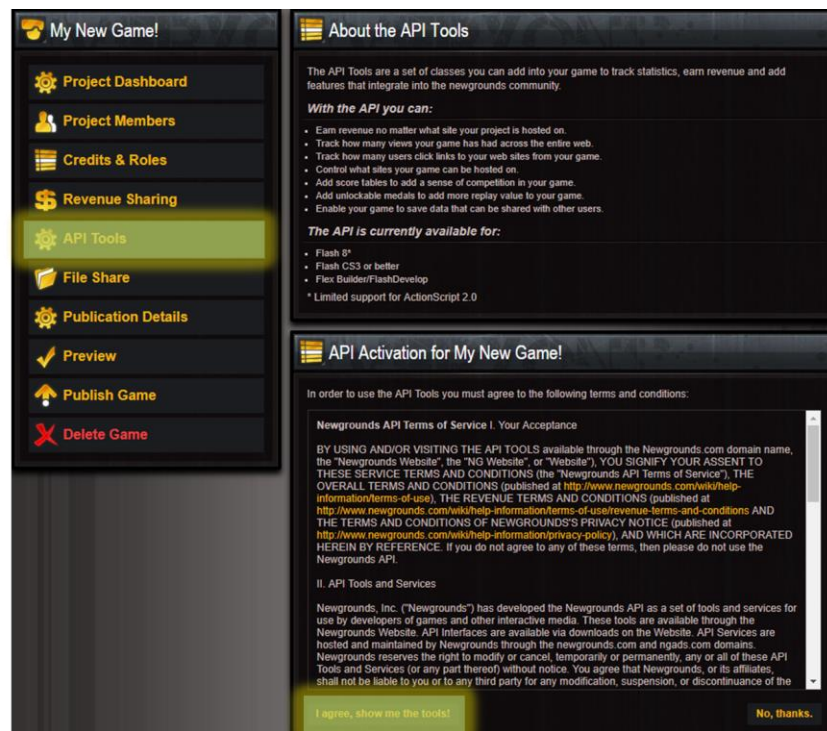
Then after clicking the 'P' button:



- Find the 'New Game Project' panel and enter a title for your project. Then click the 'Next' button.

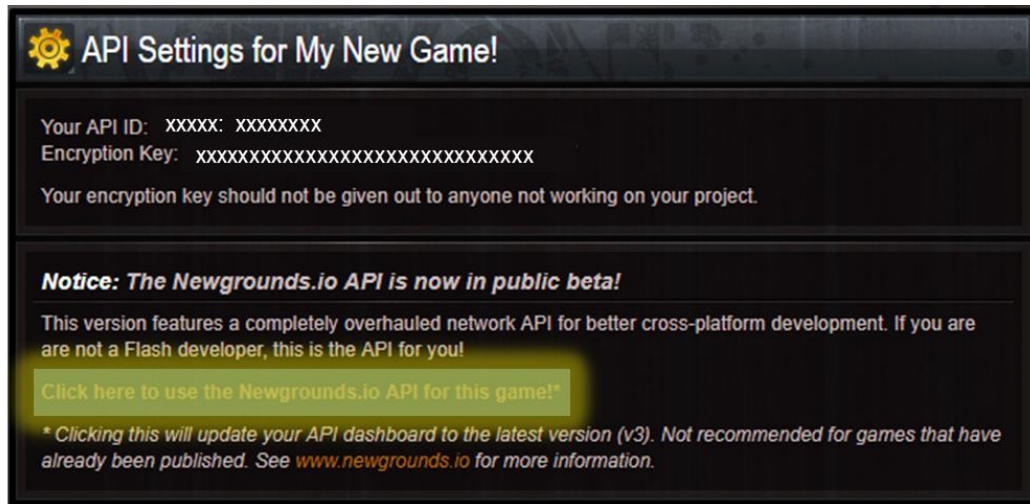


- From your new project page, click the 'API Tools' button to open your API Dashboard and click 'I agree'



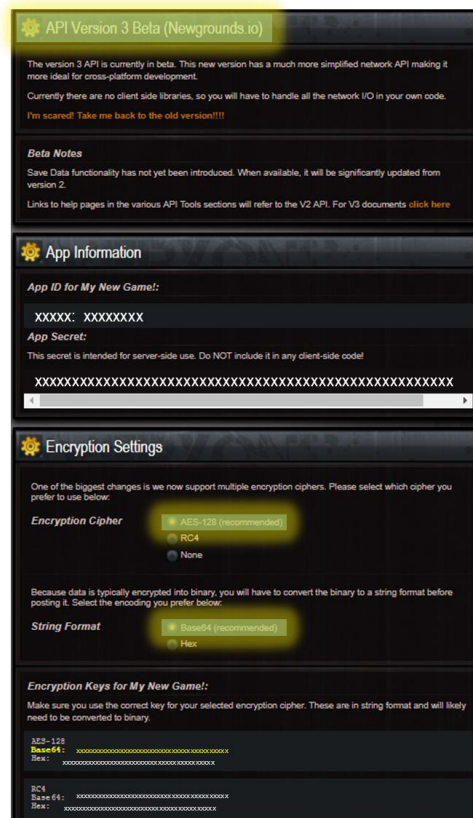
November 20, 2017

- The default dashboard will show settings for the old Newgrounds Flash API. You need to switch to the Newgrounds.io API by clicking 'Click here to use the Newgrounds.io API for this game!' link near the top of the page.



- You should now see the 'API Version 3 Beta (Newgrounds.io)' dashboard. At the top of the page is your basic 'App Information'. Make a note of your App ID, you will need it to use Newgrounds.io. At the end of the page is a form for configuring your Encryption Settings. You might need to change these depending on what encryption methods are available in your platform. If your settings are correct (and you are using encryption), make a note of the highlighted encryption key.

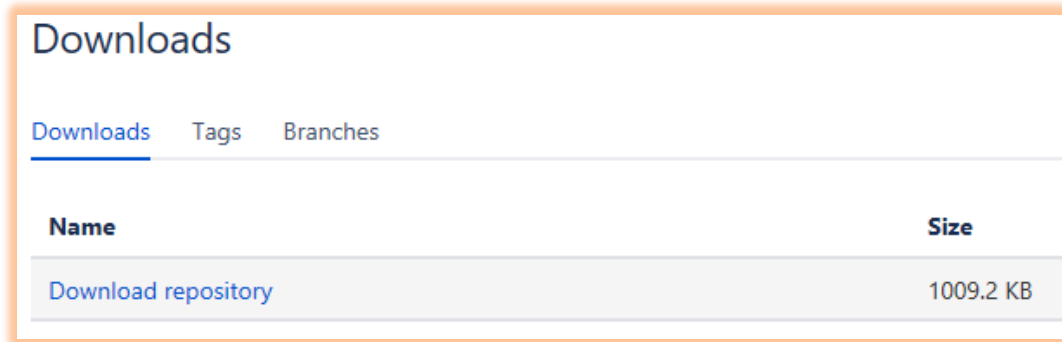
**Guide's Note. The encryption cypher for Unity3D is AES-128 string format Base64.*




November 20, 2017

B. Adding the API to your Unity3D game.

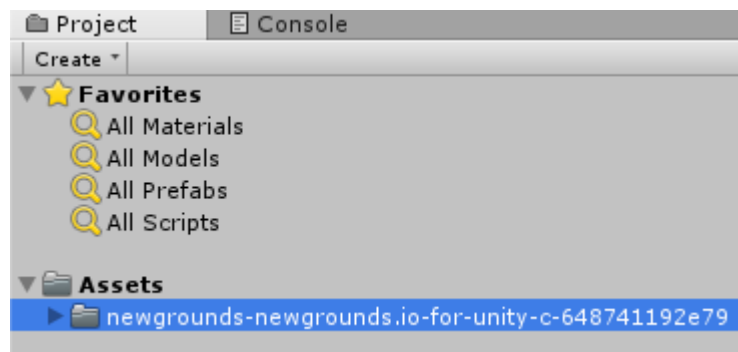
- Get the repository (files) from <https://bitbucket.org/newgrounds/newgrounds.io-for-unity-c/downloads> and unzip it to your project directory.



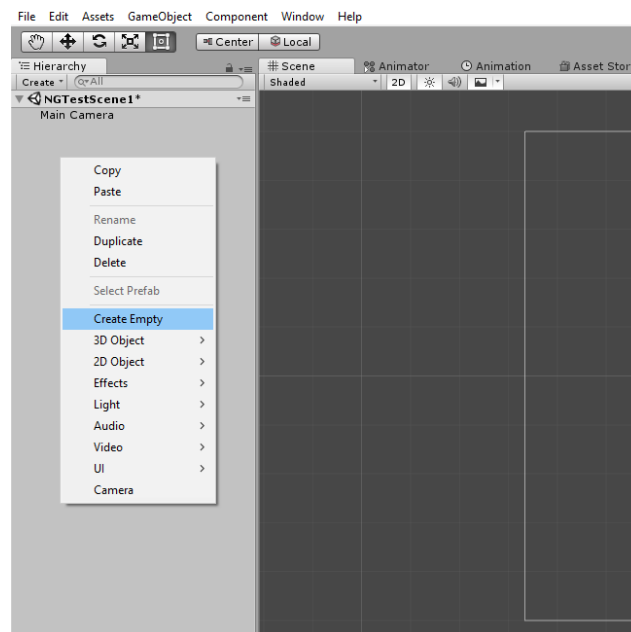
**Download the repository to your computer*

 newgrounds-newgrounds.io-for-unity-c-648741192e79

**Copy the above folder into your Assets folder in Unity*

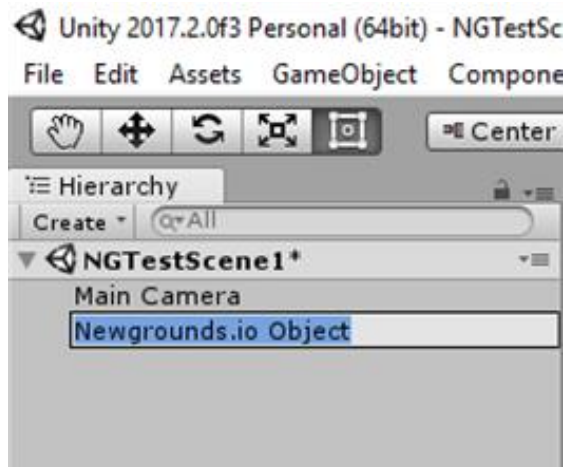


- Within your Unity Project create an Empty GameObject (Ctrl+Shift+N)

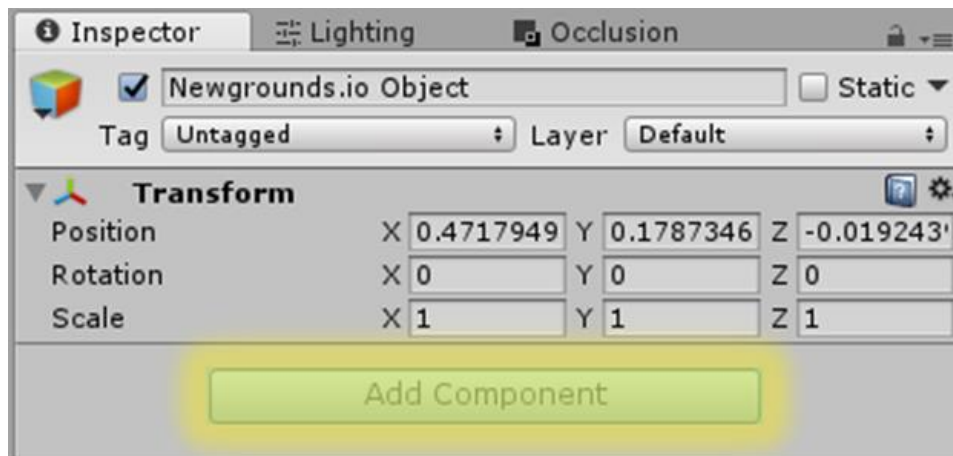


November 20, 2017

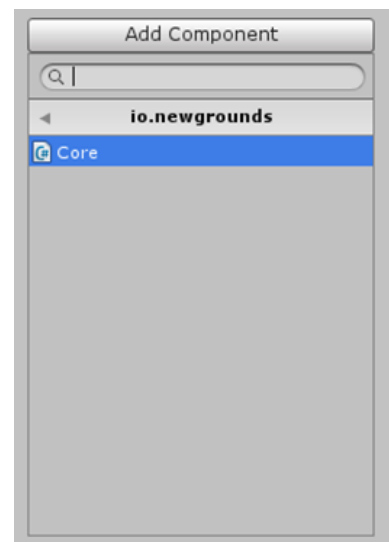
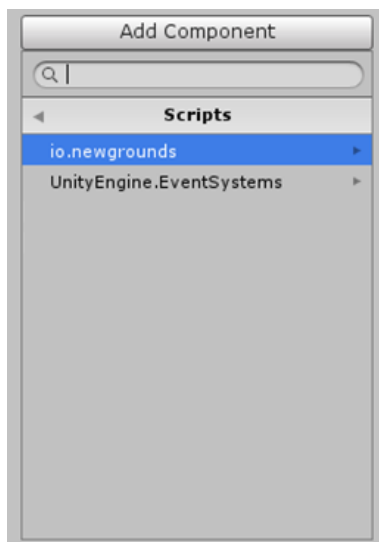
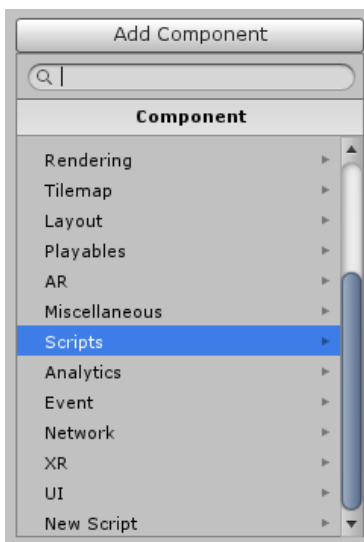
- Select the new GameObject and rename it to "Newgrounds.io Object" from the Inspector panel.



- Select the object and on the Inspector panel, click the 'Add Component' button.

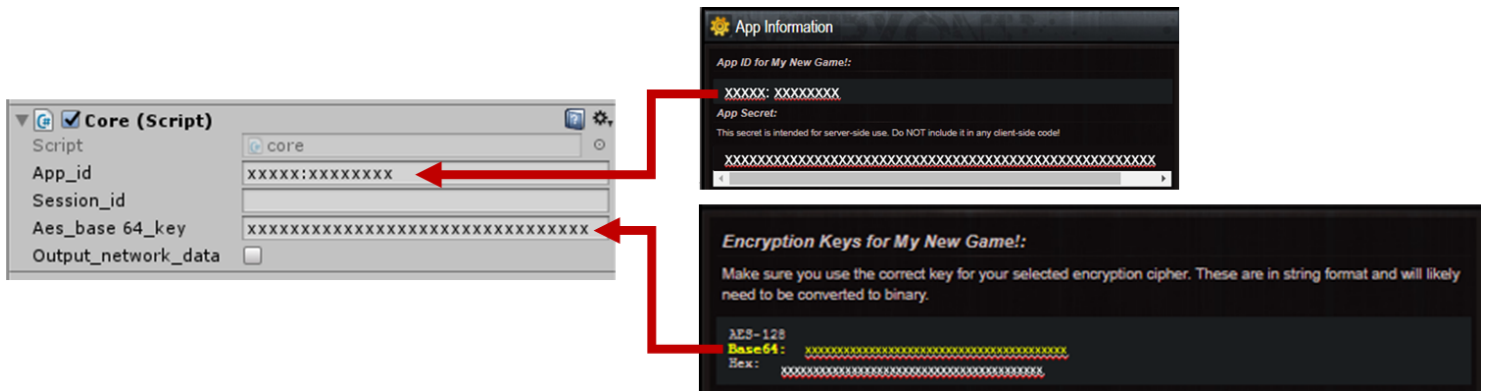


- Select Scripts -> io.newgorunds -> Core



November 20, 2017

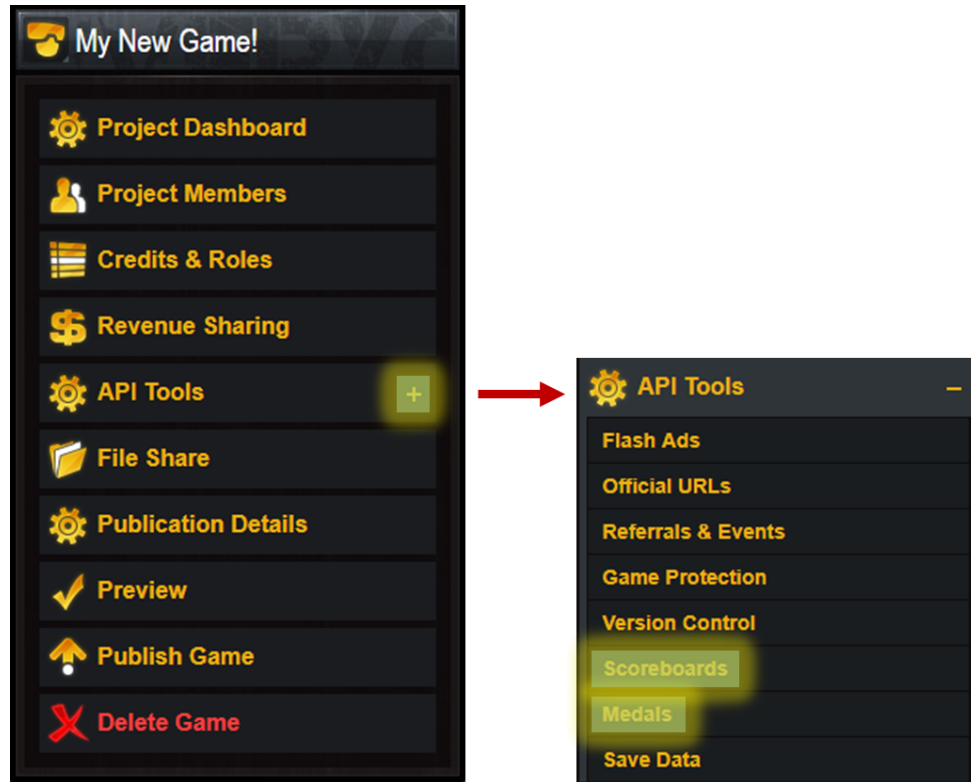
- Fill in the App_id and Aes_base64_key properties with the values found in the 'API Tools' section of your Newgrounds.com project.



**Leave the Session_id in blank. This will be filled out by the API itself automatically if you are playing the game on Newgrounds. Later on we will add some code to be able to test the game within the Unity Editor.*

C. Adding medals / scoreboards on Newgrounds and initial code in Unity3D.

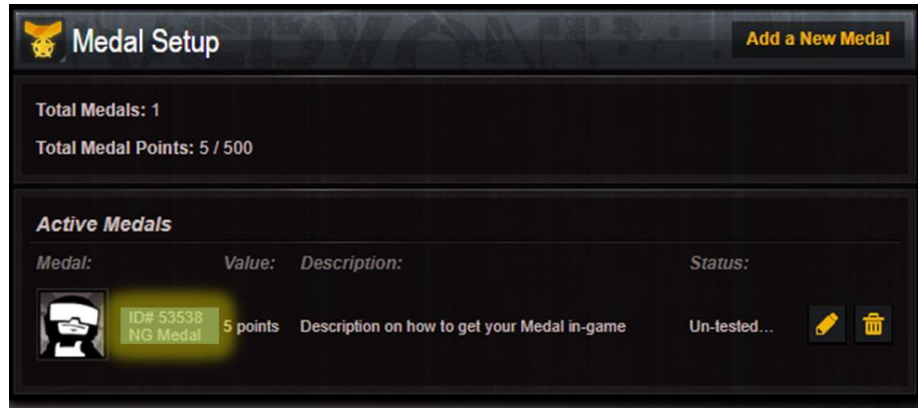
- To add a medal or scoreboard go to your game project page on Newgrounds and click '+' button on the API Tools option. And then select the option you would like to implement.



- For medals there will be a dashboard with the option to 'Add a New Medal'. Fill out the information and upload an icon for the medal, then click 'Submit'.

The image shows the 'Add a New Medal' form on Newgrounds. The form has a title 'Add a New Medal' with a medal icon. Below the title, it says 'You currently have 500 medal points available for new medals.' and a note: 'Note: once a medal has been unlocked over 5000 times, the difficulty value can not be changed.' The form contains several input fields: 'Medal Name' (with 'NG Medal' entered), 'Medal Description' (with 'Description on how to get your Medal in-game' entered), 'Medal Difficulty' (with 'Easy (5 points)' selected from a dropdown), 'Secret Medal?' (with 'No' selected from a dropdown), and 'New Icon' (with a file upload area). The 'New Icon' section shows a preview of the uploaded icon 'icon.png' (3.3 KB, 50 px wide by 50 px tall) and a 'Delete' button. To the right of the 'New Icon' section, there is a note: 'Must be a 50x50 .gif or .jpg < 28k. No adult imagery or animation.' At the bottom of the form, there is a 'Submit' button and a label 'Add Medal?'.

- You now will see your newly created medal with the status 'Un-tested'. The most important thing here is the ID number so write it down.

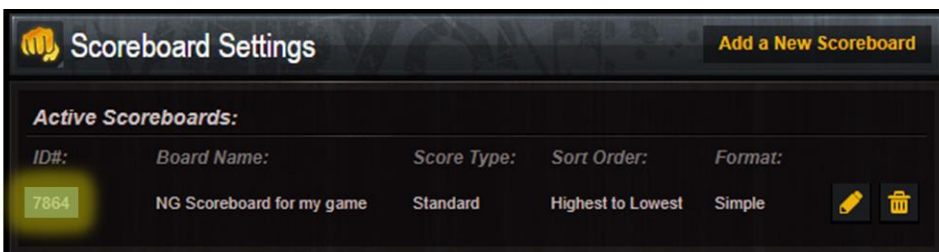


**On further steps we will get into coding so the status then reads 'Approved'.*

- For scoreboards there will be a dashboard with the option to 'Add a new Scoreboard'. Fill out the information so it meets your needs, then click 'Submit'

The screenshot shows the 'Add a new Scoreboard' form. It has a header with a fist icon and the text 'Add a new Scoreboard'. The form contains several fields: 'Scoreboard Name' with the value 'NG Scoreboard for my game' and a note 'May only contain numbers, letters, dashes and spaces.'; 'Score Type' with two radio buttons: 'Standard: Scores will be posted individually' (selected) and 'Incremental: New scores are added to user's existing score value'; 'Sort Scores' with two radio buttons: 'From highest to lowest' (selected) and 'From lowest to highest'; 'Score Format' with a dropdown menu set to 'Simple' and a note 'All score formats are saved as integers which may require some conversion on your part. See the examples below for additional help.'; and a 'Submit' button.

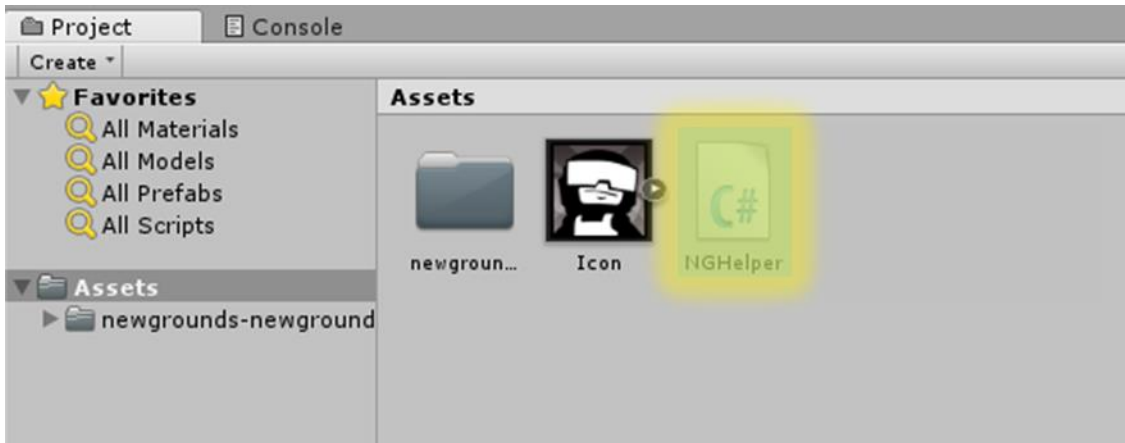
- You will now see your newly created scoreboard. The most important thing here is the ID number so write it down.



- Now, prepare yourself as this part is the most confusing of them all.

The only object that can talk to the Newgrounds server is the 'Newgrounds.io Object', every instruction to unlock medals or post to a scoreboard is done by this object so we need to trigger the functions within that object's script to successfully implement our medals.

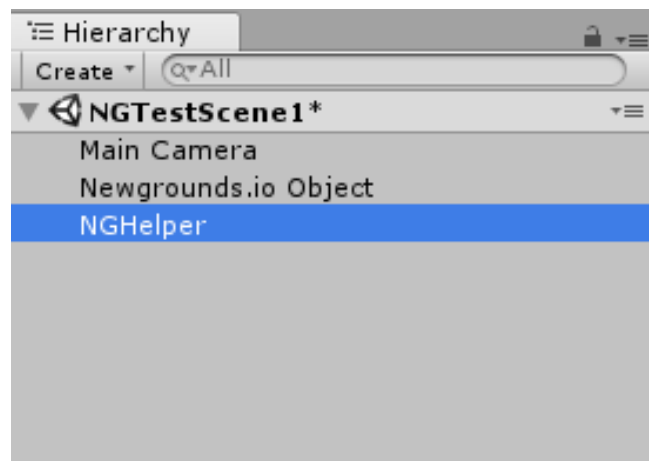
To do that 'triggering' I prefer to create a new script, I will call it 'NGHelper.cs'



- What we are about to do is:
 1. Create a new object in your project so the 'NGHelper.cs' script is available in-game.
 2. Link the 'NGHelper.cs' script to the 'Newgrounds.io Object' script.
 3. With C# code, check if you are connected to the Newgrounds server via a Session_ID, if not connected you will create a new Session_ID so you can do tests within the Unity Editor.
 4. With C# code, create medalUnlock() and submitScore() functions to trigger the 'Newgorunds.io Object' calls to the server.
 5. Implement further coding into other GameObjects so they unlock the medals or post scores.

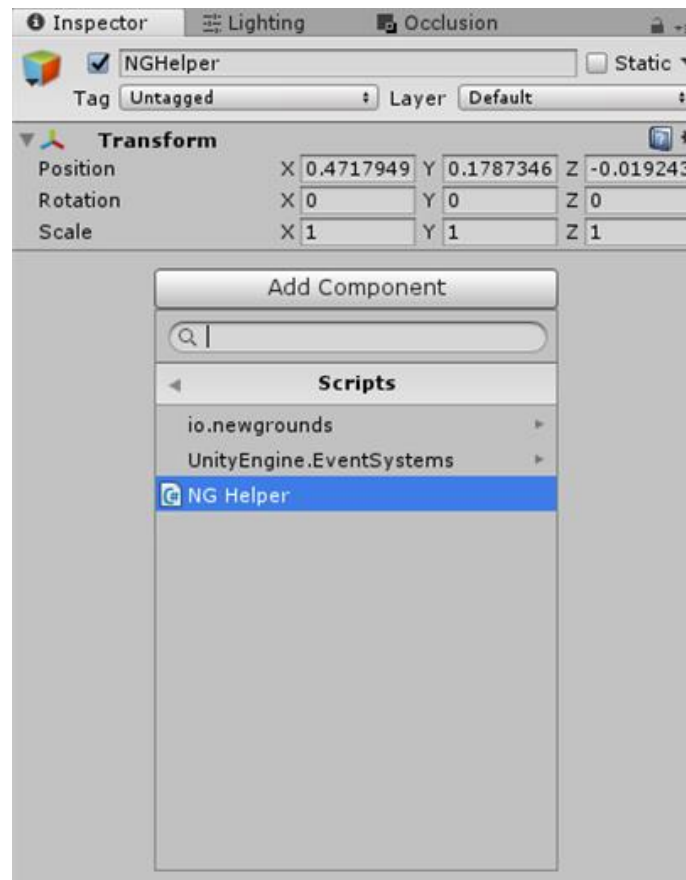
1. Create a new object in your project so the NGHelper.cs script is available in-game.

- Create an Empty GameObject. I will rename it to 'NGHelper'

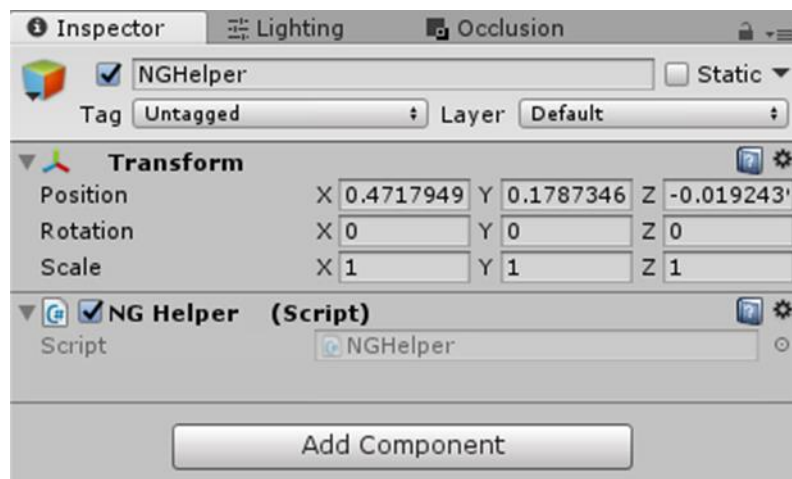


November 20, 2017

- Select it and on the Inspector panel click the 'Add Component' button and search the 'NGHelper.cs' script (or just drag the script to the Objects Inspector panel)



- This is how it should look after adding the script component.



2. Link the 'NGHelper.cs' script to the 'Newgrounds.io Object' script.

- Open the 'NGHelper.cs' script with MonoBehaviour editor (or Visual Studio) and at the top just below the class declaration write the following code:

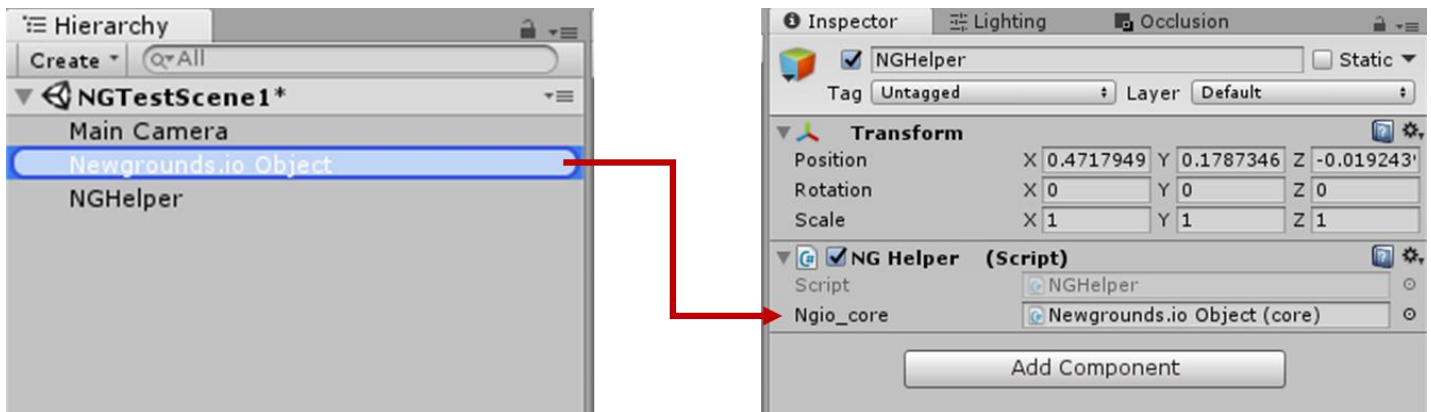
```
public io.newgrounds.core ngio_core;
```

- It should look something like this:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class NGHelper : MonoBehaviour {
    public io.newgrounds.core ngio_core;
```

- Then on the Unity Editor select the 'NGHelper' GameObject and on the Inspector panel find the script component 'Ngio_core'. Then link the 'Newgrounds.io Object' to the script by dragging it over the blank space.



3. With C# code, check if you are connected to the Newgrounds server via a Session_ID, if not connected you will create a new Session_ID so you can do tests within the Unity Editor.

- Now back into MonoBehaviour (or Visual Studio) and into the 'NGHelper.cs' add the following code within the Start() function:

```
void Start()
{
    ngio_core.onReady(() => {

        // Call the server to check login status
        ngio_core.checkLogin((bool logged_in) => {

            if (logged_in)
            {
                onLoggedIn();
            }
            else
            {
                //Opens up Newgrounds Passport if they are not logged in
                requestLogin();
            }
        });
    });
}
```

This will trigger the 'Newgrounds.io Object' to ask the server if you have an active Session_ID, you can see on the example above that if the logged_in boolean is true it will search the function called onLoggedIn() otherwise it will search the function requestLogin().

Those functions do not exist yet on our script so we have to create them.

- First let's do the onLoggedIn() function. Below is only an example code, in here you can do whatever you want to do after the Log In was confirmed. I mainly use it to load the Main menu Scene of my game with a SceneManager.LoadScene("MainTitle"); function.

```
// Gets called when the player is signed in.
void onLoggedIn()
{
    // Do something. You can access the player's info with:
    io.newgrounds.objects.user player = ngio_core.current_user;
}
```

- Now let's do the requestLogin() function:

```
// When the user clicks your log-in button
void requestLogin()
{
    // This opens passport and tells the core what to do when a definitive result comes back.
    ngio_core.requestLogin(onLoggedIn, onLoginFailed, onLoginCancelled);
}
```

This code has three parameter which consist on function calls:

- The first parameter calls whatever function you want when Newgrounds Passport successfully created a new Session_ID. In the example it is calling the 'onLoggedIn()' function which we have already created.
- The second parameter calls whatever function you want when Newgrounds Passport has failed to create a new Session_ID.
- The third parameter calls whatever function you want when Newgrounds Passport has been prompted but the player cancels the Login.

**I believe any of these parameters are optional but to be sure always create these three functions.*

We already have created the function of the first parameter 'onLoggedIn()' so we only need to create two more functions onLoginFailed() and onLoginCancelled():

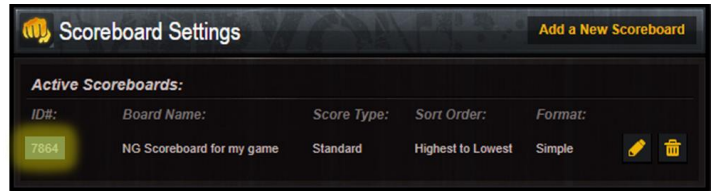
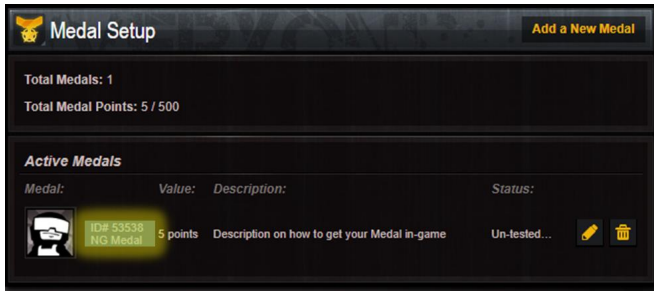
```
// Gets called if there was a problem with the login (expired sessions, server problems, etc).
void onLoginFailed()
{
    // Do something. You can access the login error with:
    io.newgrounds.objects.error error = ngio_core.login_error;
}

// Gets called if the user cancels a login attempt.
void onLoginCancelled()
{
    // Do something...
}
```

And that is basically it, now you have successfully checked if the player is logged in into Newgrounds with an active Session_ID or prompted a Login window to create a new Session_ID.

4. With C# code, create `medalUnlock()` and `submitScore()` functions to trigger the 'Newgorunds.io Object' calls to the server.

- We now may proceed to add our functions to unlock a medal or submit to the scoreboard, it is here where you need those Number IDs from the Newgrounds game project dashboard page:



- Still in the 'NGHelper.cs' script create down below your current code these two other functions:

```
public void unlockMedal(int medal_id)
{
    // create the component
    io.newgrounds.components.Medal.unlock medal_unlock = new io.newgrounds.components.Medal.unlock();

    // set required parameters
    medal_unlock.id = medal_id;

    // call the component on the server, and tell it to fire onMedalUnlocked() when it's done.
    medal_unlock.callWith(ngio_core);
    Debug.Log("Sent a message to the server to unlock a medal");
}

public void NGSubmitScore(int score_id, int score)
{
    io.newgrounds.components.ScoreBoard.postScore submit_score = new io.newgrounds.components.ScoreBoard.postScore();
    submit_score.id = score_id;
    submit_score.value = score; // *Or wherever the name of the variable you want to submit is **
    submit_score.callWith(ngio_core);
    Debug.Log("Sent a message to the server to submit to the Scoreboard");
}
```

These are the triggers I talked about earlier, with these two functions you can call the server. Please beware that it needs to be in an active Session_ID to call these functions successfully and you also need to be connected to the Internet to call the server.

Also note the parameters for each function. They are requesting the corresponding number ID, and in the case of scores it is also requesting a value parameter which is the actual score to be submitted.

At this point it is encouraged you create test game projects and play with these function so you get familiarized on the workflow, create new medals/scoreboards and directly input the values to see how the server reacts.

**An example line of code inserted at the end of `onLoggedIn()` function:*

```
// Gets called when the player is signed in.
void onLoggedIn()
{
    unlockMedal(53538);
    NGSubmitScore(7864, 1000);
}
```


November 20, 2017

Here it is shown how your complete 'NGHelper.cs' script may look after finishing the setup:

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.SceneManagement;
5
6 public class NGHelper : MonoBehaviour
7 {
8     public io.newgrounds.core ngio_core;
9
10    // Gets called when the player is signed in.
11    void onLoggedIn()
12    {
13        // Do something. You can access the player's info with:
14        io.newgrounds.objects.user player = ngio_core.current_user;
15        SceneManager.LoadScene("MainMenu");
16    }
17
18    // Gets called if there was a problem with the login (expired sessions, server problems, etc).
19    void onLoginFailed()
20    {
21        // Do something. You can access the login error with:
22        io.newgrounds.objects.error error = ngio_core.login_error;
23    }
24
25    // Gets called if the user cancels a login attempt.
26    void onLoginCancelled()
27    {
28        // Do something...
29    }
30
31    // When the user clicks your log-in button
32    void requestLogin()
33    {
34        // This opens passport and tells the core what to do when a definitive result comes back.
35        ngio_core.requestLogin(onLoggedIn, onLoginFailed, onLoginCancelled);
36    }
37
38    // Check if the user has a saved login when your game starts
39    void Start()
40    {
41        // Do this after the core has been initialized
42        ngio_core.onReady(() =>
43        {
44            // Call the server to check login status
45            ngio_core.checkLogin((bool logged_in) =>
46            {
47                if (logged_in)
48                {
49                    onLoggedIn();
50                }
51                else
52                {
53                    requestLogin();
54                }
55            });
56        });
57    }
58
59    public void unlockMedal(int medal_id)
60    {
61        // create the component
62        io.newgrounds.components.Medal.unlock medal_unlock = new io.newgrounds.components.Medal.unlock();
63
64        // set required parameters
65        medal_unlock.id = medal_id;
66
67        // call the component on the server, and tell it to fire onMedalUnlocked() when it's done.
68        medal_unlock.callWith(ngio_core);
69        Debug.Log("Sent a message to the server to unlock a medal");
70    }
71
72    public void NGSubmitScore(int score_id, int score)
73    {
74        io.newgrounds.components.ScoreBoard.postScore submit_score = new io.newgrounds.components.ScoreBoard.postScore();
75        submit_score.id = score_id;
76        submit_score.value = score; // *Or wherever the name of the variable you want to submit is **
77        submit_score.callWith(ngio_core);
78        Debug.Log("Sent a message to the server to submit to the Scoreboard");
79    }
80 }
81
82
83
```

5. Implement further coding into other GameObjects so they unlock the medals or post scores.

Now for the final step, you will need to call the trigger functions `unlockMedal()`; and `submitScore()`; with any other GameObject's script within Unity.

Just remember that the only GameObject that can actually call the server is the 'Newgrounds.io Object' and that we have created the NGHelper GameObject & script to trigger those calls.

So in a new script you can use a `GameObject.Find("")` function to reference the NGHelper public functions `unlockMedal()`; and `submitScore()`;

You can use something like this to unlock a medal:

```
GameObject.Find("NGHelper").GetComponent<NGHelper>().unlockMedal(53538);
```

Or something like this to submit to the scoreboards:

```
int scoreValue = 1000;  
GameObject.Find("NGHelper").GetComponent<NGHelper>().NGSubmitScore(7864, scoreValue);
```

**Here I'm forcing the scoreValue to be 1000 but you can reference any variable so it submits the correct score you want to send to the server.*

Lastly, these examples are only to get the medals and scoreboards working, there are a ton other functions you can use, like doing something when the server responds to your calls, managing callbacks and results or check if the user is logged in before attempting to unlock a medal so you assure the server is reachable, etc. as I don't really understand any other functions of the 'Newgrounds.io Object' I don't feel confident enough to include them in this guide.

If you feel adventurous enough you can check the full documentation here:

<https://bitbucket.org/newgrounds/newgrounds.io-for-unity-c/src/648741192e790f52a087214ffc10b411bdac9b85/Help/?at=default>

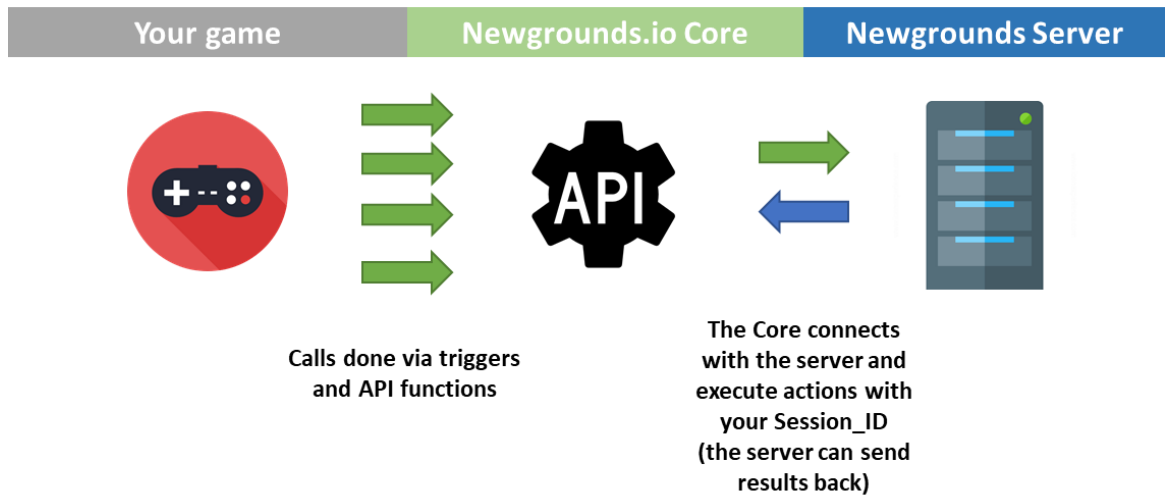
And the full official explanation of how the API works here:

<https://www.newgrounds.com/wiki/creator-resources/newgrounds-apis/newgrounds-io/tutorials/using-the-network-api>

D. Questions I had while implementing the API medals and scoreboards.

1. How the API works?

I'm taking a HUUUUUGE risk by trying to answer this since I don't really know how the API works nor am I an expert in server connections, but as far as my understanding goes I can summarize it in this little diagram:



The Newgrounds core is the only GameObject that can communicate with the server so you should reference it in any part of your game scripts when you attempt something related to Newgrounds.

If you want to know how the API really works then you should read this:

<https://www.newgrounds.com/wiki/creator-resources/newgrounds-apis/newgrounds-io/tutorials/using-the-network-api>

2. What is the Session ID & do I need to write something in that blank space in the Inspector panel?

This question first came to me when I tried to set up the API and because I couldn't find any mention of it in the documentation I didn't know how to fill that part in.

The Session ID just refers to the currently logged in user and you actually don't have to fill that, the API does it automatically if you are playing your game directly on Newgrounds, however if you want to test it within the Unity Editor you will need some coding to start a new session from within the Unity Editor with the help of Newgrounds Passport feature. *(See part 3 of this guide for coding examples).*

3. What is Newgrounds Passport?

Newgrounds Passport is a remote login feature used to login into your Newgrounds user from wherever you are. Most of the time it won't be needed because this connection is automatic if you are inside Newgrounds network but there are times when it is a lifesaver for example when you want to test your game within Unity3D.

4. I have added a new medal but it reads “Un-tested”, what does that mean?

It means no one has unlocked the medal yet, more technically, the server has never received the instruction to give the medal to any user.

To change the status to “Approved” you will need to unlock it by playing your own game (if you are a responsible developer that is, and why would you release a game if you haven’t tested it? Go back to work!).

5. Can players get the medals if the status is “Un-tested”?

Yes, taking into account your coding is right and the game is published, people will be able to get the medal. But if you ask me it is not really advisable to publish a game if you haven’t tested the medals yourself.

6. Why the cool Newgrounds medal unlocked animation is not showing up in-game?

What the Unity3D API does is send/receive JSON messages to/from the server so all UI has to be done by you (at least in Unity, I don’t know for Flash games maybe that API does have the medal unlock animation). It was really sad knowing this because the Newgrounds unlock animation is pretty cool.

However it also means you have complete creative freedom to let the player know when a medal was unlocked. It can get as simple as showing a “Medal unlocked” message or as complex as using a Particle System with multiple fanfare audio and NPCs clapping, the only limit is your imagination really.

I hope this visual guide has been of help to you.

If you have been able to implement these features I would love to see your games, please stop by at <https://geckomla19097.newgrounds.com/> and say hi!