# Extra Unity Topics

By Peter Mitchell

Contents

[Extra Unity Topics 1](#_Toc79171645)

[1 Using Debugger with Unity 1](#_Toc79171646)

[2 GitHub Unity Project Setup 3](#_Toc79171647)

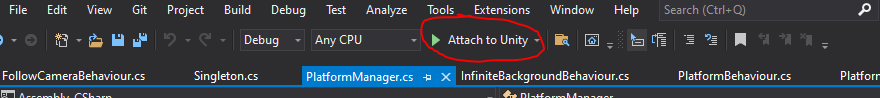
[2.1 Note about GitHub and Large Files 5](#_Toc79171648)

# Using Debugger with Unity

You can view additional information at:

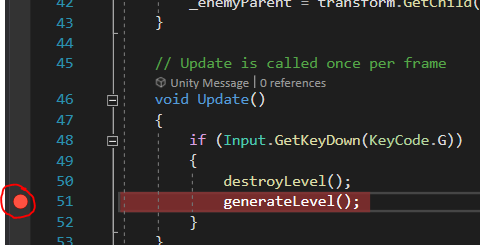
<https://docs.unity3d.com/Manual/ManagedCodeDebugging.html>

1. Click Attach to Unity in Visual Studio.

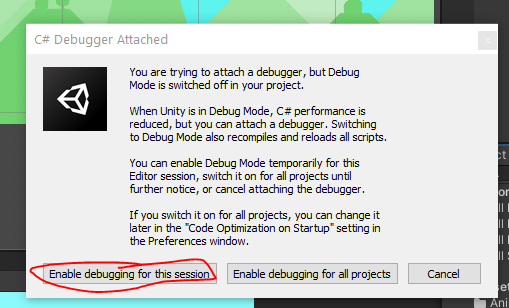


1. Create a Breakpoint.

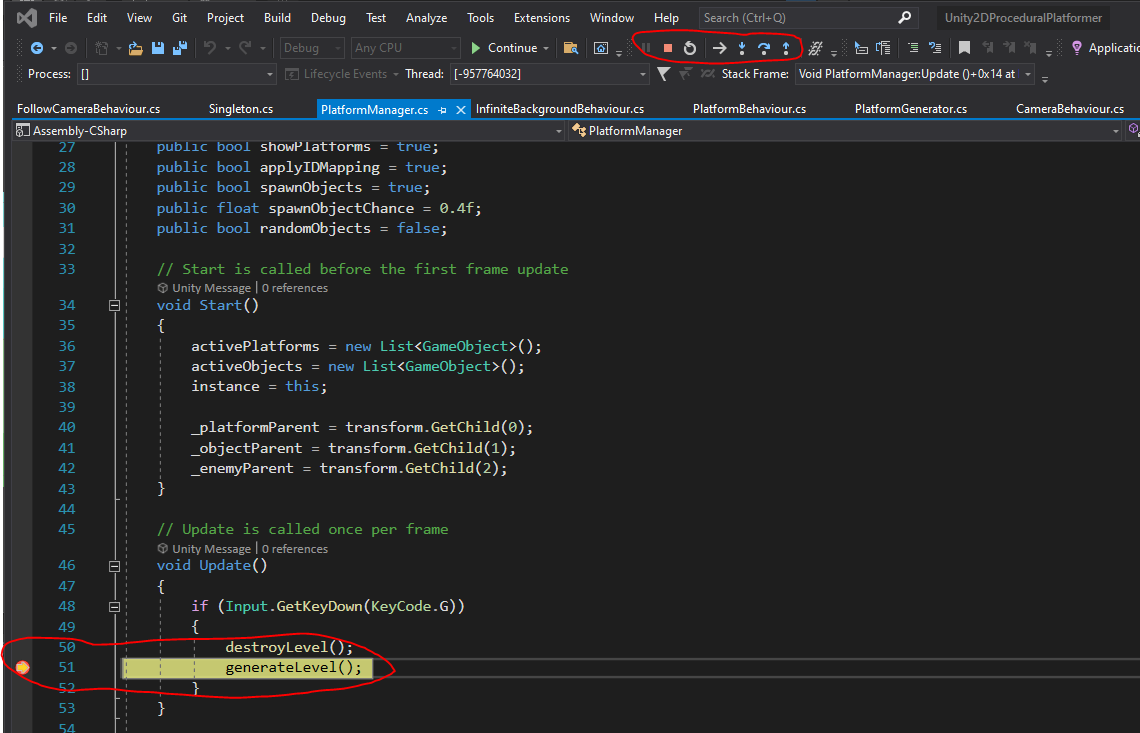
You can either click in that far left differently coloured column or you can select some code, Right click it, then Breakpoint->Insert Breakpoint.



1. When you return to Unity you will find a popup. Click Enable debugging for this session.



1. Run your game. And do something that triggers the Breakpoint you set. Note that it will likely have to recompile your scripts to allow debugging.
2. You can see below that the program has paused at the Breakpoint. In the top is a circled set of controls. And you can see the line is highlighted to show where the program is paused currently.



1. The controls are shown larger here:



In order:

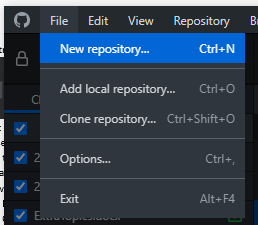
* Pause (to pause at a current point in execution).
* Stop to Terminate.
* Restart to Restart the Application.
* Show Next Statement.
* Step Into.
* Step Over.
* Step Out.

And if you just want to continue execution as normal you can click “Continue” over where the Attach button was. You will mostly probably use Step Into to move inside functions, and Step Over to move to the next step of a function.

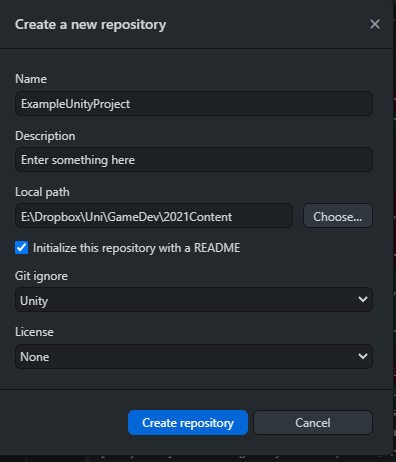
You will find down the bottom associated variable states and the Call Stack. In combination these allow you to track down the cause of bugs by watching your variables change as you move through the program line by line.

# GitHub Unity Project Setup

1. Create a GitHub account and download the GitHub Desktop application. You can use the command line to control everything with GitHub if you wish instead, but if you are doing that you probably don’t need this guide and should be able to find the resources yourself.
2. Create a new Repository.



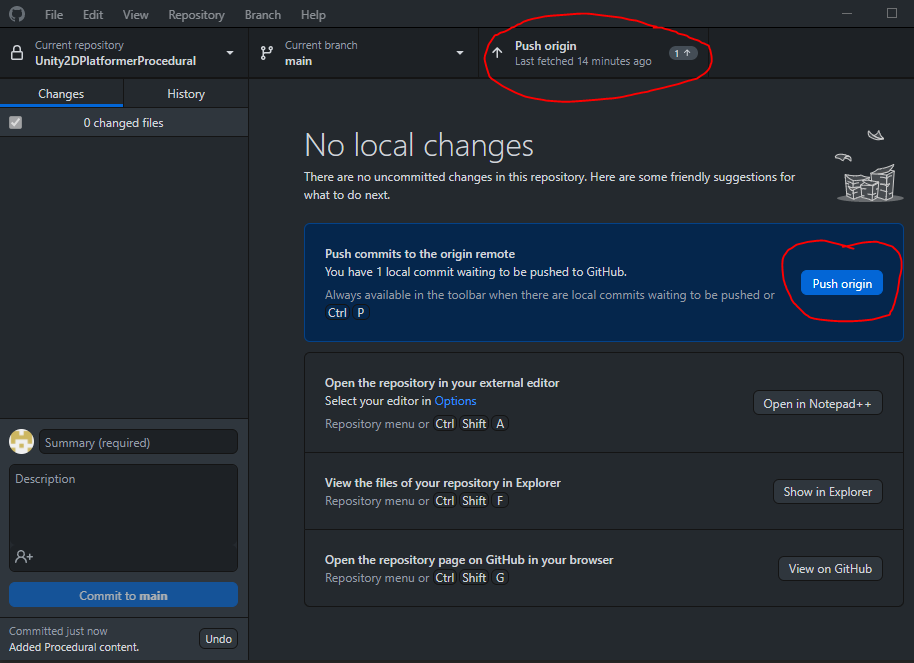
1. You will see a dialog as seen below.



Make sure to set a project name, description, a path where you want the project to be placed on your system, (optionally) tick the Initialise with README, and VERY importantly select “Unity” from the Git ignore dropdown. This will exclude files that are generated by Unity and are not necessary to the project being used on multiple systems.

Click Create repository when ready.

1. Either create your Unity project inside the folder you chose (I would suggest a nested folder). Or copy your Unity project folder into it.
2. You will see a list of all the added/changed files as shown to the right.
3. Write a summary that describes the content you are adding and if necessary, a description to further elaborate. Then click the Commit to main button.
4. When you are ready to make the changes appear on GitHub you can use either of the “Push origin” buttons as seen over page.



## Note about GitHub and Large Files

If you are dealing with files near 100MB or more, you will have to use Git LFS as part of your setup. You can find information about this below.

<https://docs.github.com/en/github/managing-large-files/versioning-large-files/configuring-git-large-file-storage>