NWH ::coding

Grapher

DOCUMENTATION FOR VERSION 1.0

CONTENTS

IMPORTANT!

Due to the Unity Asset Store folder structure policy for the assets, folder "Grapher" needs to be manually moved to "Editor" folder so the structure is as following:

Assets > Editor > Grapher

This is needed for Grapher to work properly.
If "Editor" folder does not exist in your project create a new one.

CONTACT & MEDIA

- For any questions, problems or suggestions contact me at arescec@yandex.com.
- YouTube Channel
- Unity Asset Store Publisher Profile
- Grapher Forum

1. QUICK START

PLAY

1) Import the main asset forlder „Grapher“ to the **root** of your assets. Main folder should now be located at „Assets > Grapher“. This is so icons can be properly loaded.

2) Open the script containing the variable you want to graph and add this line to the code:

```
Grapher.Log(myVar, "myName");
```

   Supported: int, float, long, double, Vector2, Vector3, bool, enum, List, Queue, Array, ArrayList

3) Open Grapher by pressing *Ctrl + G* (Windows) or *Command + G* (Mac OS), or go to Window > Grapher in the Unity3D. You could always dock this window so you do not have to reopen it all the time.

4) Enter play mode in Unity Editor.

REPLAY - INSTANT

1) If you want to replay the values you have just graphed, after exiting the play mode in Unity Editor press „Play“ button in the bottom bar of the Grapher window. This kind of replay is supported even if logging to file is disabled.

REPLAY - FROM FILE

1) If you want to replay the values you have previously logged to a file, while out of the Unity Editor play mode press „Open“ button and either:

   a. select an individual file (.csv) which will load a single variable
   b. select a session file (.ses) which will load all the .csv files from that play session.

   You can use „Open“ button multiple times to add additional files.

2) Press „Play“ button in the bottom bar of the Grapher Window to start or pause the replay.

2. UI Elements

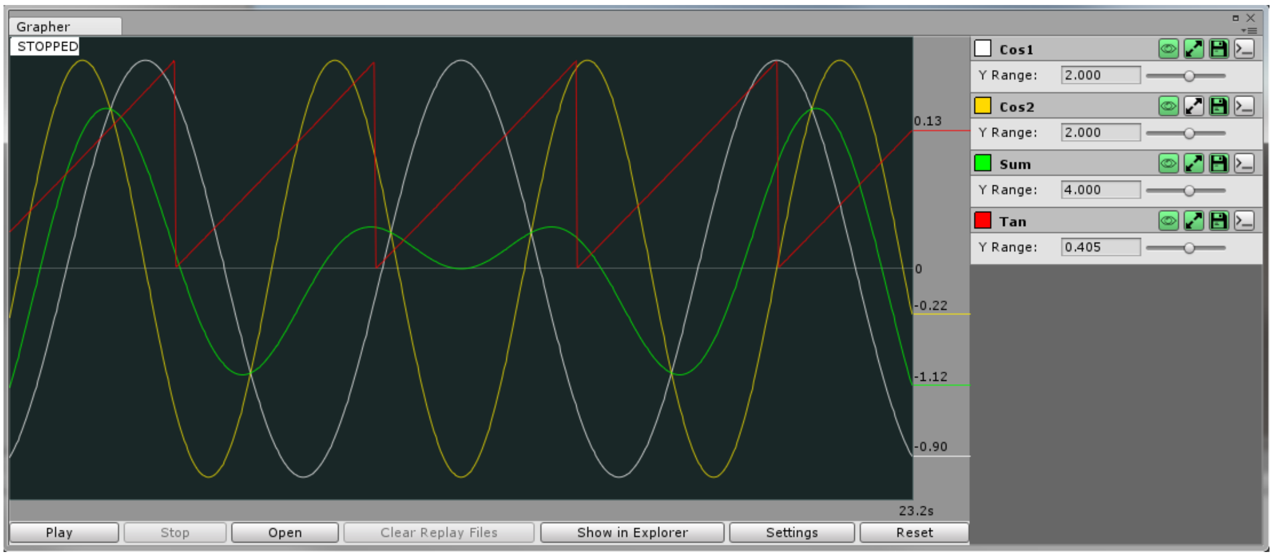


IMAGE 1: EXAMPLE GRAPHER EDITOR WINDOW.

PLAY (BUTTON)

- Only available ouside of Unity Editor play mode. Used to start the replay. Also doubles as a pause button.

PLAY (BUTTON)

- Only available ouside of Unity Editor play mode. Freezes the replay. Pressing it again will toggle Grapher back into play and continue with playback.

STOP (BUTTON)

- Only available outside of Unity Editor play mode. Freezes the replay at the given moment. Pressing Play afterwards will restart the playback from the start.

CLEAR REPLAY FILES (BUTTON)

- Removes all files from the replay queue and clears all the replay chanels.

SHOW IN EXPLORER (BUTTON)

- Opens the folder Grapher is logging to in file browser.

SETTINGS (BUTTON)

- Opens Grapher settings window.

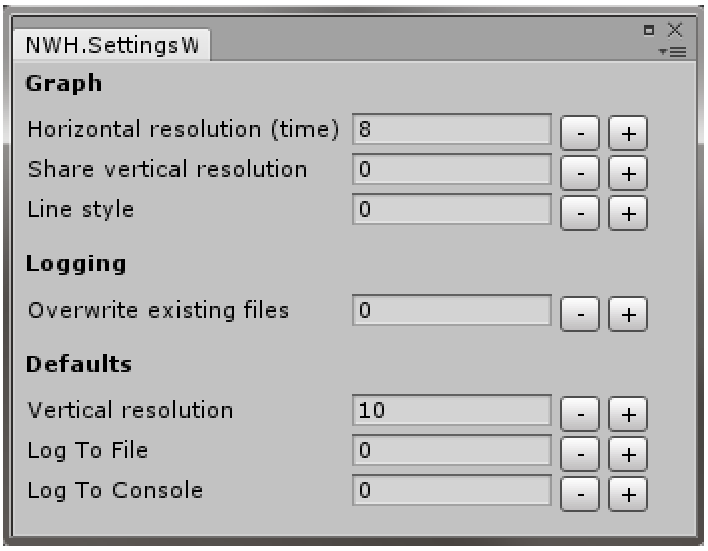
RESET (BUTTON)

- Resets Grapher. Chanel variables such as color and state of the toggle buttons will stay in memory if same chanel is logged again. If used during logging to file in Unity Editor play mode, all the values prior to reset will not be logged or available for instant replay.

TOGGLE ICONS

- **View** – shows or hides the graph for the corresponding variable.
- **Auto scale** – changes the range of the Y axis to fit the graph inside the viewport.
- **Log to file** – saves variable's graph to the file. Save happens when exiting Unity Editor play mode so it does not matter when you activate (or deactivate) it as only the state it was in when exiting play mode matters.
- **Log to console** – if enabled Debug.Log() is called for the variable.

3. Settings


NWH.SettingsW

Graph

Horizontal resolution (time)	8
Share vertical resolution	0
Line style	0

Logging

Overwrite existing files	0

Defaults

Vertical resolution	10
Log To File	0
Log To Console	0

IMAGE 2: SETTINGS WINDOW EXAMPLE.

HORIZONTAL RESOLUTION

- Determines how long samples are visible, e.g. 2 second time window would mean that the oldest part of graph (furthest to the left) is 2 seconds old.

SHARE VERTICAL RESOLUTION

- 0 = disabled, 1 = enabled. If enabled all the channels that have auto scale icon enabled will share the same vertical resolution. E.g. the highest value logged is 100 and auto scale for that chanel will set vertical resolution to 240 (-100 to +100 and 20% extra overhead for visibility). Since this was the higest resolution and share is enabled all the chanels will have vertical resolution onf 240.

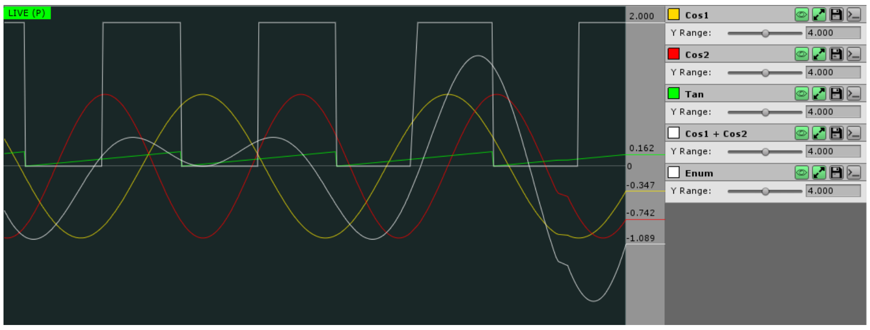

🟨 Cos1	Y Range: 4.000
🟥 Cos2	Y Range: 4.000
🟩 Tan	Y Range: 4.000
⬜ Cos1 + Cos2	Y Range: 4.000
⬜ Enum	Y Range: 4.000

IMAGE 3: SHARE VERTICAL RESOLUTION ENABLED.

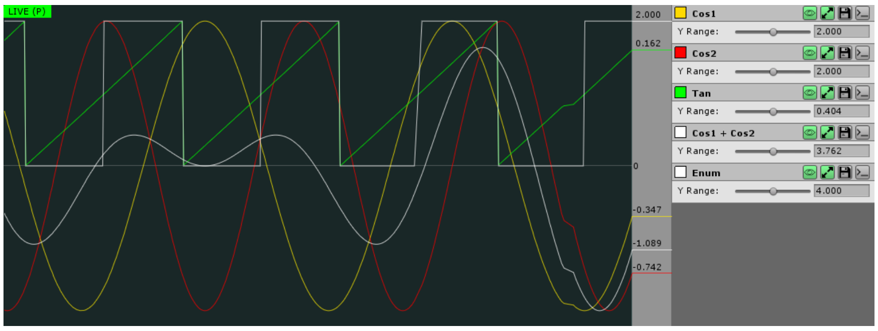

🟨 Cos1	Y Range: 2.000
🟥 Cos2	Y Range: 2.000
🟩 Tan	Y Range: 0.404
⬜ Cos1 + Cos2	Y Range: 3.762
⬜ Enum	Y Range: 4.000

IMAGE 4: SHARE VERTICAL RESOLUTION DISABLED.

LINE STYLE

- 0 = line, 1 = dots. If used while playing or replaying dots will slow down Grapher significantly since they are individually drawn.

OVERWRITE EXISTING FILES

- 0 = false, 1 = true. If enabled, files will not be uniquely named and therefore overwritten. If disabled, each file will have timestamp and session number.

VERTICAL RESOLUTION (DEFAULT)

- Default vertical resolution for each new chanel added that does not have autoscale enabled or was not previously graphed.

LOG TO FILE (DEFAULT)

- 0 = false, 1 = true. Default state of the „Log To File“ icon when a variable is logged for the first time. If the variable has already been graphed previously, saved value will be used instead of default.

LOG TO CONSOLE (DEFAULT)

- 0 = false, 1 = true. Default state of the „Log To Console“ icon when a variable is logged for the first time. If the variable has already been graphed previously, saved value will be used instead of default.

4. Logging

To log a variable you can use one of the following overloads:

- `Grapher.Log(<T> value, string name);`
- `Grapher.Log(<T> value, int id) ;`
- `Grapher.Log(<T> value, string name, Color color) ;`
- `Grapher.Log(<T> value, string name, float time);`
- `Grapher.Log(<T> value, string name, Color color, float time);`

Where supported types are:

- **int, float, long, double** – displayed and logged as float
- **Vector2,** Vector3 – displayed and logged as two or three different chanels (variables). Each dimesnion is represented with one float and appendix x, y or z is added to the provided name.
- **bool, enum** – also displayed and logged as float. For bool 0 = false and 1 = true. For enum first element equals 1, second 2 and so on.
- **List, Queue, Array, ArrayList, (IEnumerable)...** – for each element a separate chanel is created, graphed and logged. Graphing a list with 3 elements will give the same result as typing Grapher.Log() for each. Appendix with index number is added to the name.

CHANGE LOG

v1.0

- Initial