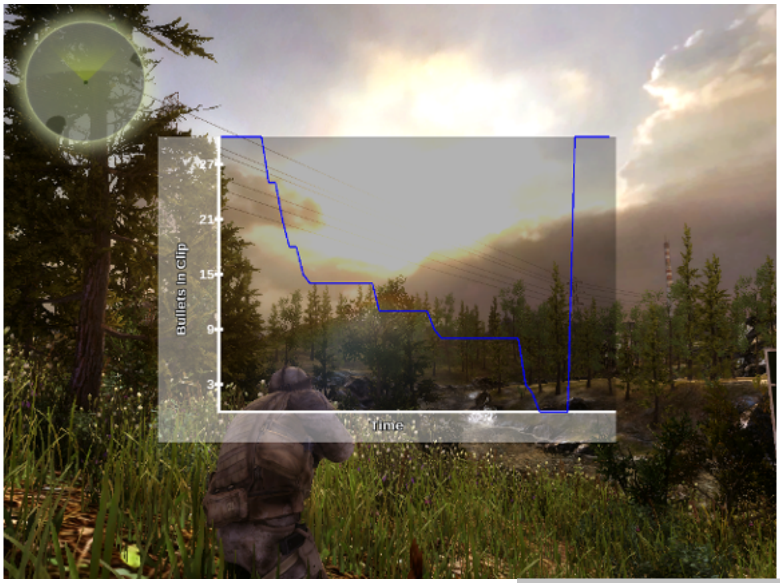
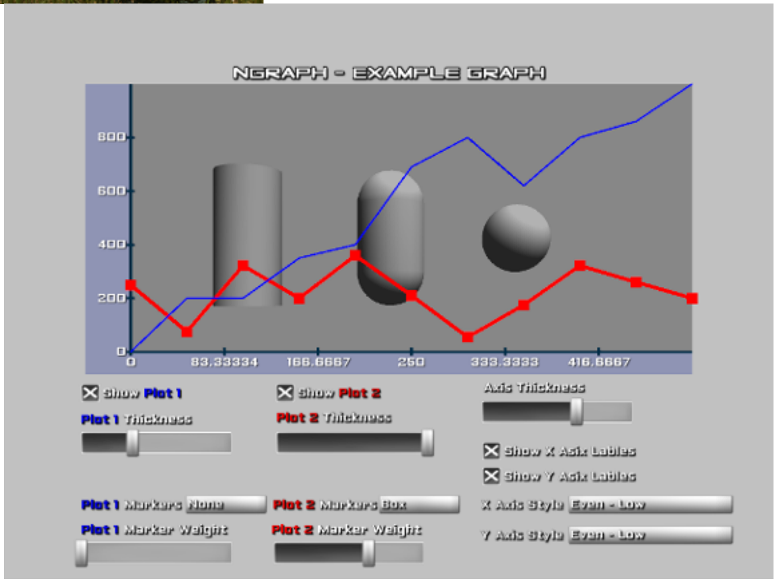
HOME UNITY ASSET STORE

N-GRAPH

DOCUMENTATION





☒ Show Plot 1

☒ Show Plot 2

☒ Show X Axis Lables

☒ Show Y Axis Lables

Plot 1 Markers None

Plot 2 Markers Box

X Axis Style Even - Low

Y Axis Style Even - Low

EXAMPLES

Interactive

Live Update

Equations

SUPPORT

Support is available on our forums for those who have purchased N-Graph though the Unity Asset Store.

The example at the botom of this page is the best place to start. If you need something more raw, then the API documentation can be found here.

INTRODUCTION

N-Graph is a Unity run-time graphing tool available in the Unity Asset Store. Its purpose is to allow Unity developers to display dynamic graphs to their players. Graphs can be a great way to show the player information both during and after a game. With N-Graph you can now easily display information to the player with only a few lines of code.

Currently, N-Graph works with Daikon Forge or NGUI 3.0.4 (lower if you delete the N-Graph/Editor folder). This means that you must have Daikon Forge (available in the Unity Asset Store) or NGUI (available in the Unity Asset Store) imported into your project before you import N-Graph. There are plans to also support the new Untiy 4 native GUI system when it is released.

GETTING STARTED

Step 1: (Only for 3rd part GUIs) Prerequisites
If you are using a 3rd party GYU system, it must be installed. Install Daikon Forge, 2DTK, or NGUI from the Unity Asset Store, then install N-Graph from the Unity Asset Store. Once these packages are imported, you can create your first graph.

Step 2: (Only for 3rd part GUIs) Extract The Correct GUI System
If you are using a 3rd party GYU system extract the package named with the GUI system you own in the NGrpah asset folder. If your project is using more than one of them, you can extract each one. **Do not extract packages that have not had thier corresponding assets imported.**

Step 3 (Native uGUI): Create a GUI
Only use this step if you are using Unity's native GUI.
This step may be skipped if you already have a GUI with a canvas in place.
Create a standard UI using the UI menu to get a Canvas in place. (Menu: GameObject → UI → Canvas)

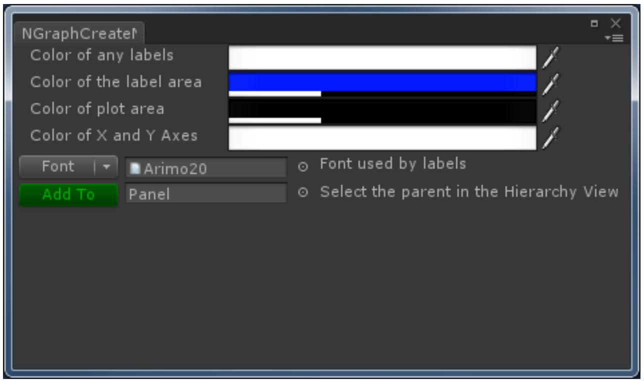
Step 3 (NGUI): Create a GUI
This step may be skipped if you already have a GUI in place.
Create a standard NGUI UI using the NGUI UI Wizard. (Menu: NGUI → Open → UI Wizard)

Step 3 (Daikon Forge): Create a GUI
This step may be skipped if you already have a GUI in place.
Create a standard Daikon Forge UI using the Daikon Forge UI Wizard. (Menu: Game Object → Daikon Forge → UI Wizard)

Step 4: Create A Graph

- Open the N-Graph "Create New Graph" Wizard. (Menu: Window → N-Graph → New Graph)

- Fill out the form and click the green "Add" button. (The "Add" button will not be green until all the required items in the form are filled in - including the font.)



NGraphCreatel

Color of any labels

Color of the label area

Color of plot area

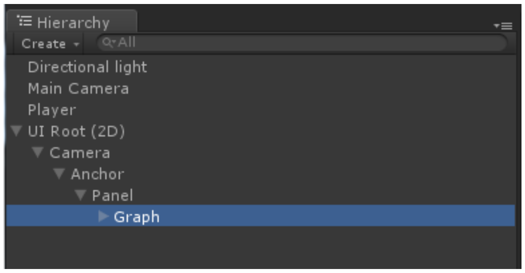
Color of X and Y Axes

Font | Arimo20 | Font used by labels

Add To | Panel | Select the parent in the Hierarchy View

Step 5: Add Some Data

Create a custom script and add it to the newly created "Graph" game object. An example of a script that plots a data series can be found below:



SimplePlotExample.cs

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

// This MonoBehavior expects to be attached to the same game object
// that a UINgraph is attached to.  This object is called "graph" if
// you created it with the NG-raph Graph Creation Wizard.
public class InteractiveExampleGraphScript : MonoBehaviour
{
  UINgraph mGraph;
  NGraphDataSeriesXy mSeries1;

  void Awake ()
  {
    // Capture the graph
    mGraph = gameObject.GetComponent<UINgraph>();
    // Setup the graph
    mGraph.setRanges(0, 500, 0, 1000);

    // Create the data we want to plot
    List<Vector2> data = new List<Vector2>();
    data.Add(new Vector2(0, 0));
    data.Add(new Vector2(50, 200));
    data.Add(new Vector2(100, 200));
    data.Add(new Vector2(150, 350));
    data.Add(new Vector2(200, 400));
    data.Add(new Vector2(250, 690));
    data.Add(new Vector2(300, 800));
    data.Add(new Vector2(350, 620));
    data.Add(new Vector2(400, 800));
    data.Add(new Vector2(450, 860));
    data.Add(new Vector2(500, 1000));

    // Add a X/Y Plot the the graph and capture the plot and color it blue
    mSeries1 = mGraph.addDataSeries<NGraphDataSeriesXy>("1", Color.blue);
    // Apply our data to the plot.
    mSeries1.Data = data;
  }
}
```