UI Graph

Summary

Create node graphs using UI elements inside a scene, at runtime and in the editor. Any RectTransform can be a node, and connections between nodes are easy to create and manage. Use the full power of the Unity UI in both 2D and 3D!

This tool is great for visualizing complex 3D level layouts or creating graphs of object connections. Use it to keep track of your crafting system, map out an open-world environment, or plan your code structure. Connections can be configured in a variety of ways, and are automatically managed.

This package is only intended as a visual tool. Node and connection data cannot be exported without additional tools.

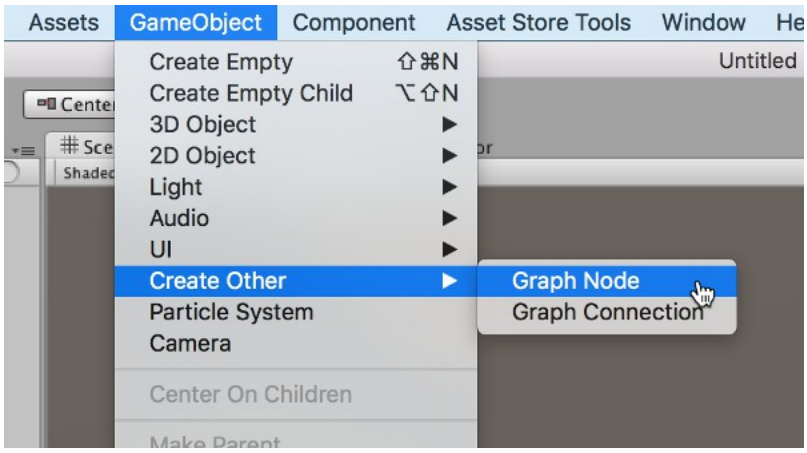
What is a Node Graph?

A node graph is a way of representing relational data. It consists of nodes, which hold some data, and connections, which show how the nodes are related. The meaning of any nodes and connections will change depending on the type of graph.

For instance, when making a graph of a crafting system, each node might show a crafting component, and the connections show which components are combined to create a product. The color of a connection might be changed to show which crafting discipline is required to make a given recipe, or how important the component is when making the product.

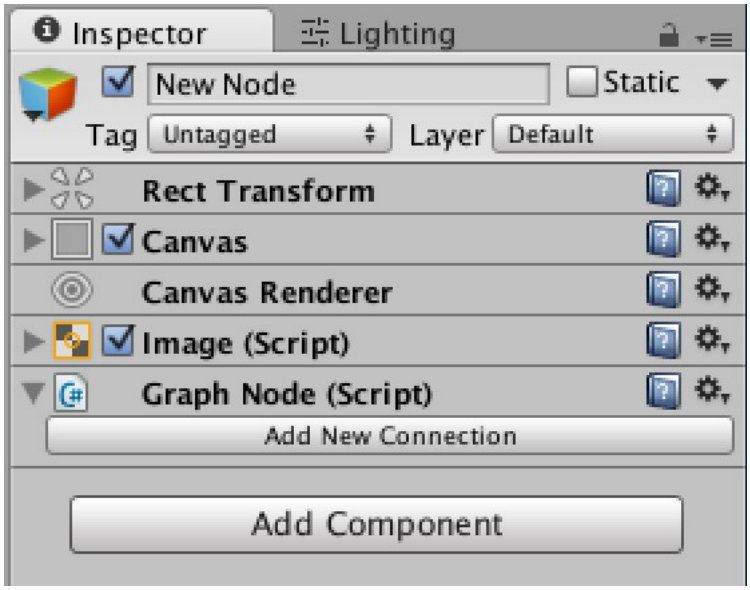
Creating a Node

To create a node, create any Unity UI Element that includes a RectTransform. This can be a canvas, a panel, or anything else. Once this UI Element is created, add a *GraphNode* component to it. This will allow you to manage any connections to this GameObject. A shortcut to create a pre-configured graph node can be found in the create menu under "Create Other -> Graph Node".



Creating a pre-configured graph node

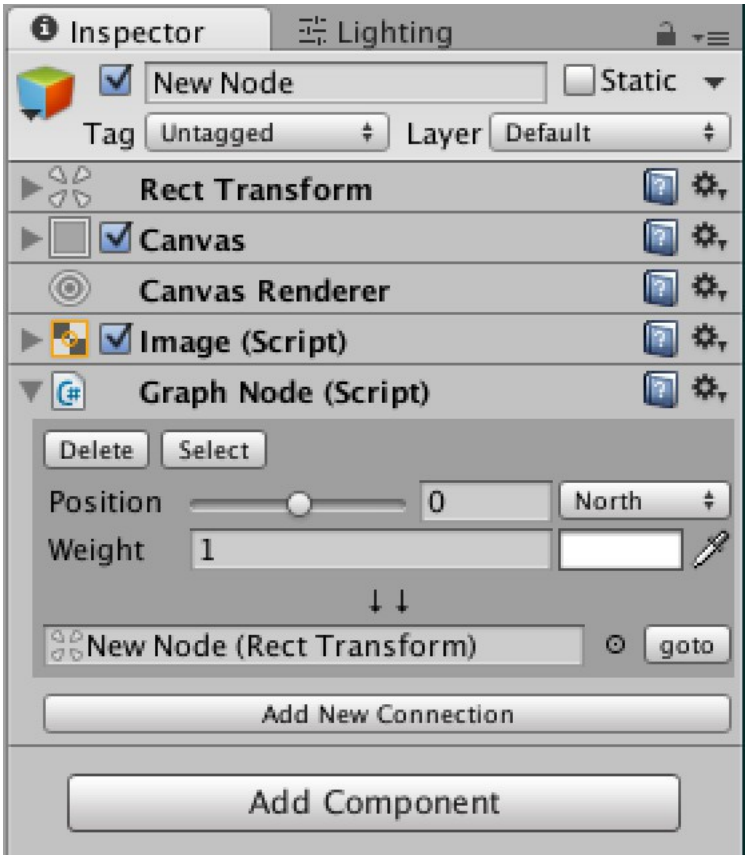
Once you add the *GraphNode* component, or add a pre-configured graph node, you should notice a new GameObject in the hierarchy called "_ConnectionManager". This GameObject has a *ConnectionManager* component, and is automatically created to keep track of any connections.



The GraphNode inspector with no connections

Adding Connections

The quickest way to add a new connection is to press the “Add New Connection” button in the *GraphNode* inspector. This will create a new connection in the *ConnectionManager* component. The new connection will be configured with one endpoint on the chosen node, and will show up immediately in the *GraphNode* inspector. You can add as many connections to a node as you want, and remove them by pressing the “Delete” button next to each connection.



The GraphNode inspector with a single connection

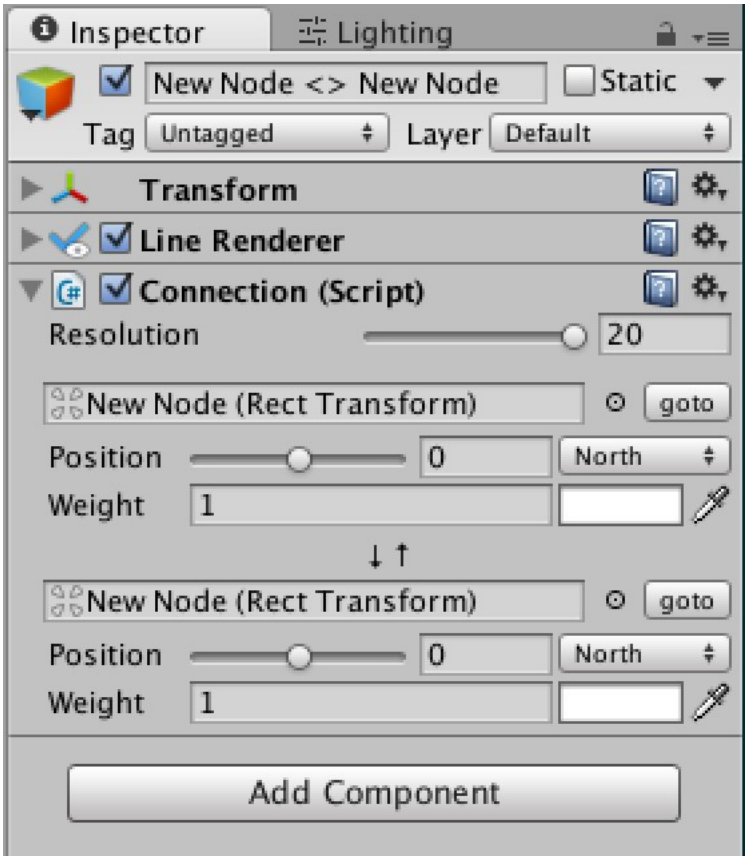
There are a number of other ways to create connections. You can also manually add a *Connection* component to a GameObject, or press the “+” button in the *ConnectionManager* inspector.

The connections to a node can be configured with several parameters. These parameters control how the connection is shown on the object itself, and will be updated in the scene whenever the values change.

Parameter	Description
Position	Controls where the line will connect to the node. The dropdown selects which edge to use, and the slider is the normalized distance along that edge, where 0 is at the center. If the dropdown is set to Radial, the slider will be a normalized angle instead.

Weight	Controls the look of the line where it connects. The number is the distance that the line will push out from the edge, and the color value is the color of the line at the connecting point.
Target	The RectTransform that this line will connect to. Pressing “goto” will select the chosen object.

Pressing the “Select” button will select the connection GameObject itself, which can also be configured. In the *Connection* inspector, as opposed to the *GraphNode* inspector, both connection points are shown and can be adjusted. Additionally, the resolution of the line can be changed. The resolution is the number of line segments that make up the curve.

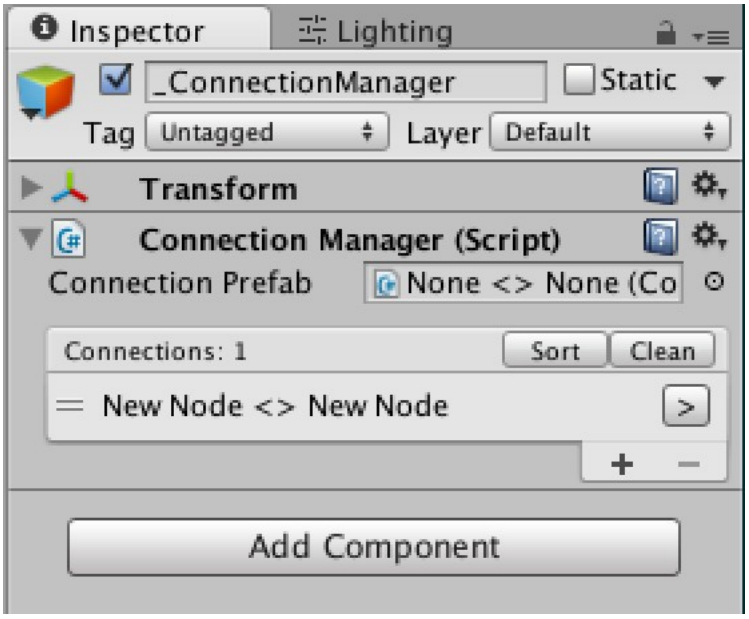


The Connection inspector

The name of a GameObject containing a *Connection* component will be automatically set whenever the connection changes. This behavior cannot be changed, and is important for managing whichever connections are in the scene. The automatically generated name will include the name of both targets in alphabetical order with a “<>” symbol between them. The targets will also be displayed in alphabetical order in the *Connection* inspector. A missing target will be given the name “None”.

Managing Connections

The connection manager can be used to easily manage all the connections in the scene. A *ConnectionManager* component is automatically created if the scene contains *Connection* or *GraphNode* components, but it can also be created manually. Only one *ConnectionManager* instance will be used, so multiple instances may cause issues.



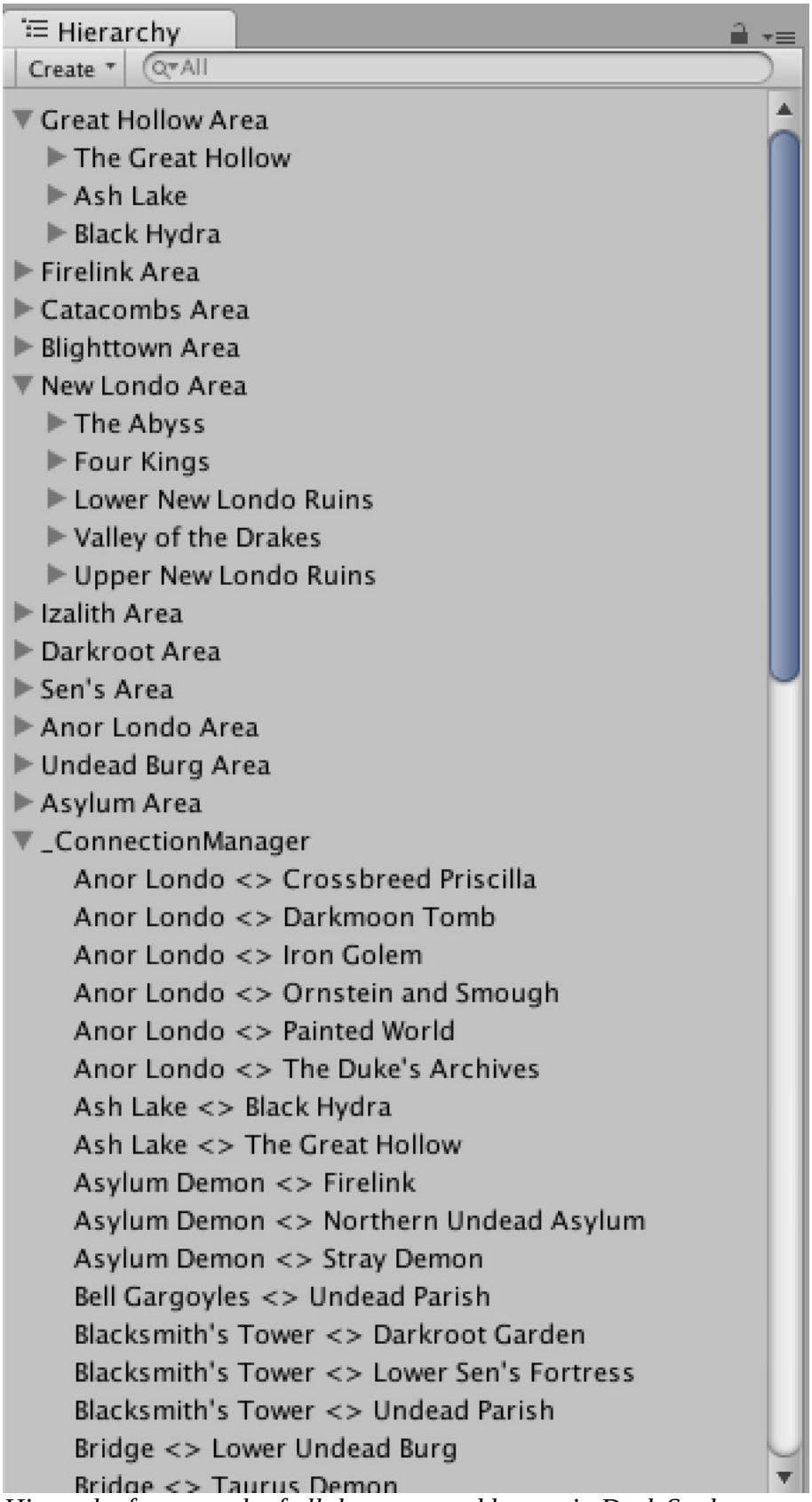
The ConnectionManager inspector

The ConnectionManager inspector shows a list of all connection objects in the scene. Selecting an element in this list will highlight the connection object in the hierarchy. The “>” button can also be pressed to select the connection object. The list can also be rearranged by dragging the leftmost part of each element.

Above the list of connections, there is a field for a connection prefab. This prefab is used whenever a connection is automatically generated in the *GraphNode* or *ConnectionManager* inspectors. This will always be configured to use the pre-supplied connection prefab, but you may create your own for the scene.

The connection list has two buttons at the top: “Sort” and “Clean”. Pressing “Sort” will rearrange all the connection objects in alphabetical order in the list and in the hierarchy. Pressing “Clean” will destroy any connection objects which have one or both targets missing.

By using the connection manager regularly to sort and clean connections, large numbers of connections can be managed easily.



Hierarchy for a graph of all the areas and bosses in Dark Souls (FromSoftware 2011)