DIGIPEN HOUDINI RESOURCE

Houdini Digital Assets (HDAs)

DigiPen Institute of Technology

West Foulks (WestFoulks@Gmail.com)

CONTENTS

PURPOSE OF THIS DOCUMENT

Welcome to this guide, designed to provide an introduction to Houdini Digital Assets (HDAs) for students who are interested in learning more about them. This document will cover the basics of creating and using HDAs and will serve as a jumping-off point for further research.

In addition to discussing HDAs within Houdini, we will also explore their use in other software through Houdini Engine, a plug-in or API that enables users to use HDAs in programs such as Unreal Engine, Unity, and Maya. Throughout this document, we will use examples from Unreal Engine to illustrate key concepts.

By the end of this guide, you should have a solid understanding of HDAs and their capabilities within Houdini and other software environments. Whether you are a student exploring Houdini for the first time or an experienced user looking to expand your knowledge, we hope that this document will be a valuable resource.

WHAT ARE HDAS?

At its core, an HDA is a node in Houdini. Many nodes that already exist in Houdini are HDAs So, you're probably using one without realizing it!

One of the benefits of HDAs is that any user can create them. This allows you to package a network of nodes into a reusable tool. For example, if you find yourself reusing a group of nodes frequently in your workflow, it's a great candidate for turning into an HDA.

By creating an HDA, you can simplify your workflow and save time by encapsulating a group of nodes into a single node that can be easily shared with others. This is particularly useful for creating custom tools and workflows that can be reused across multiple projects.

WHAT IS HOUDINI ENGINE?

Houdini Engine is an API that enables communication between other software and an installation of Houdini. This allows for a variety of powerful features, including the ability to use HDAs in other programs. In this guide, we will focus specifically on how Houdini Engine enables the use of HDAs in Unreal Engine.

Common software that has Houdini Engine implementation.
Unreal Engine, Unity, Autodesk Maya

Down Sides

Houdini Engine may be slower than using Houdini directly in some cases.

Houdini Engine's stability can vary between different software and versions.

DEMO – PROCEDURAL BRIDGE

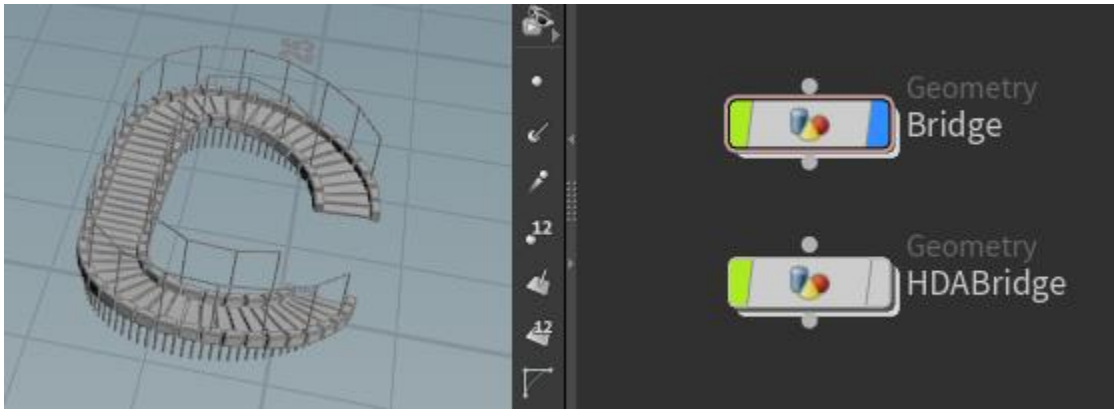
BRIEF

This demo provides a basic introduction to connecting inputs to your HDA and exporting it to Houdini. To keep things simple, we have included a file containing a network to create a procedural bridge.

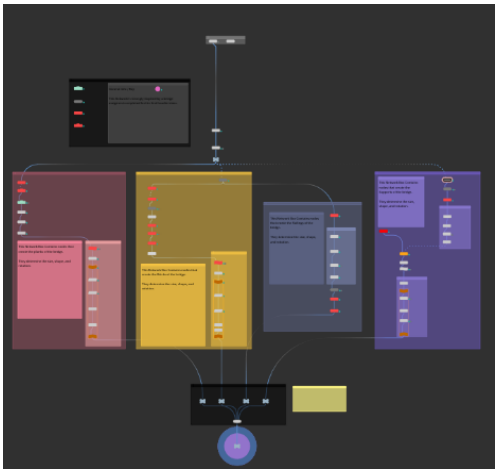
While we won't be covering how to create the bridge from scratch in this demo, we will show you how to convert it into an HDA and bring it into Unreal Engine. By doing so, you'll gain a better understanding of how to create and use HDAs in your own projects.

INSIDE HOUDINI

To get started, open the file containing the bridge. Inside, you will find two nodes: one with the basic network, and one that contains an HDA that was created from the network.



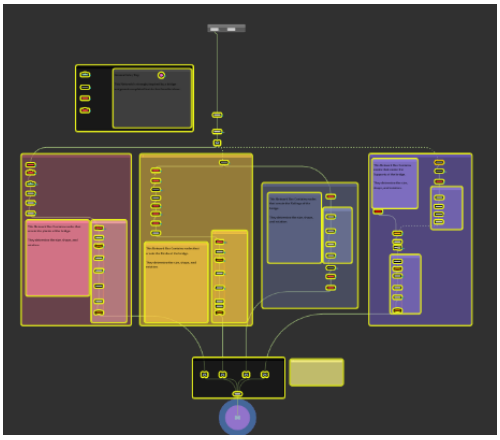
NETWORK TO HDA



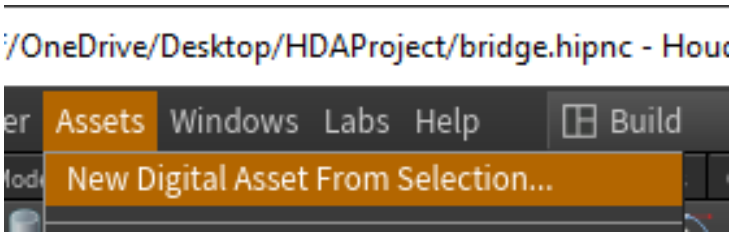
To begin, we will convert the basic network into an HDA. Take a moment to examine the network and read the comments to gain a better understanding of how it operates.

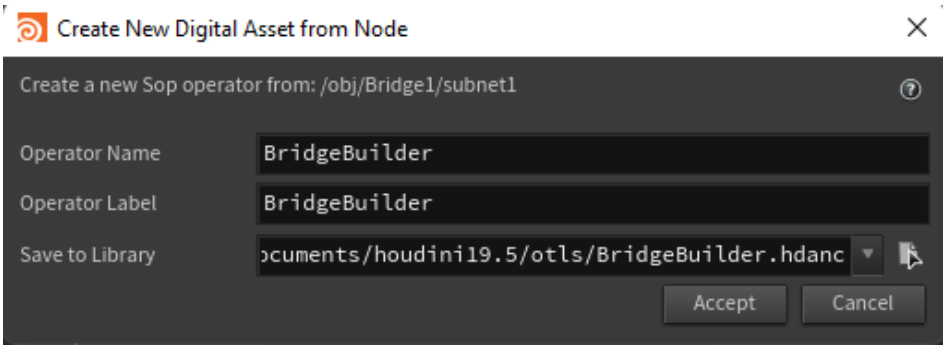
Pay special attention to the curves at the top, which represent the input of our HDA, and the "OUT_Finished" node, which will serve as the output.

Also note the "Control" node, which contains a Parameter Interface that allows you to adjust the look of the bridge.

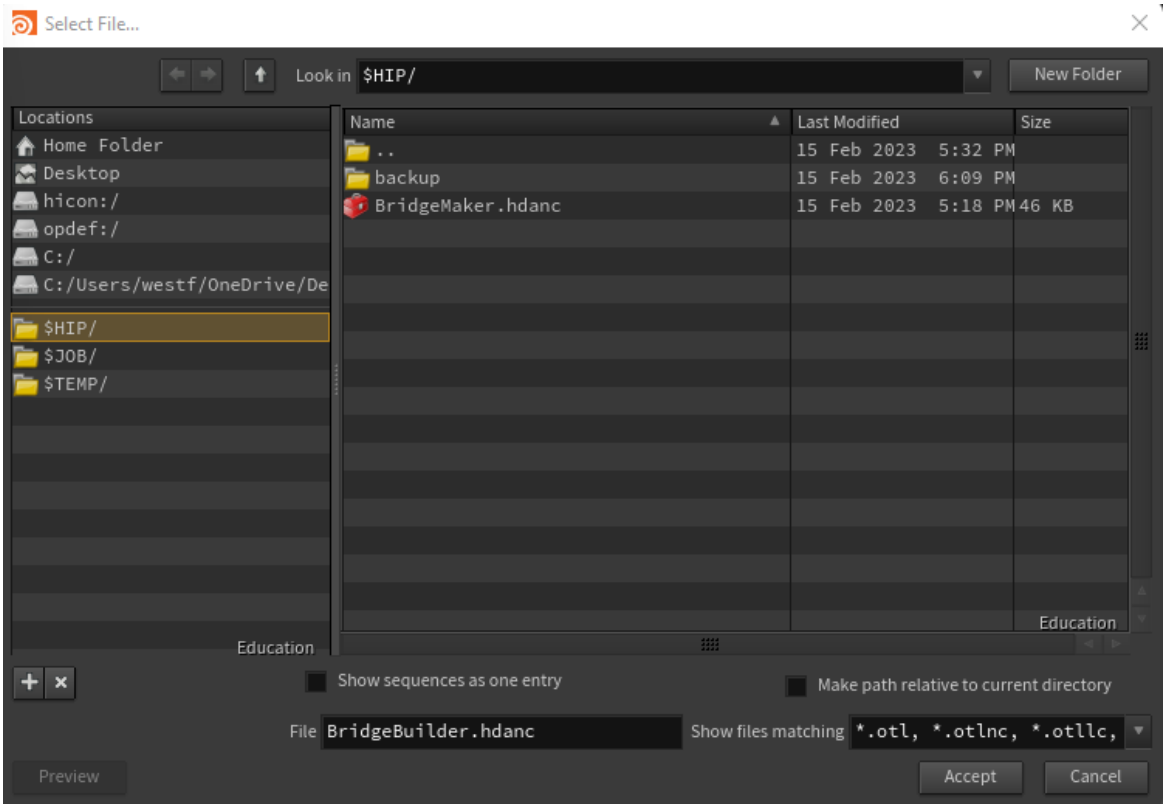


To create the HDA, select all the nodes except for the inputs and final output. Then, open the Assets menu at the top and click "New Digital Asset from Selection." While there are multiple methods for creating an HDA, we will be using this method.

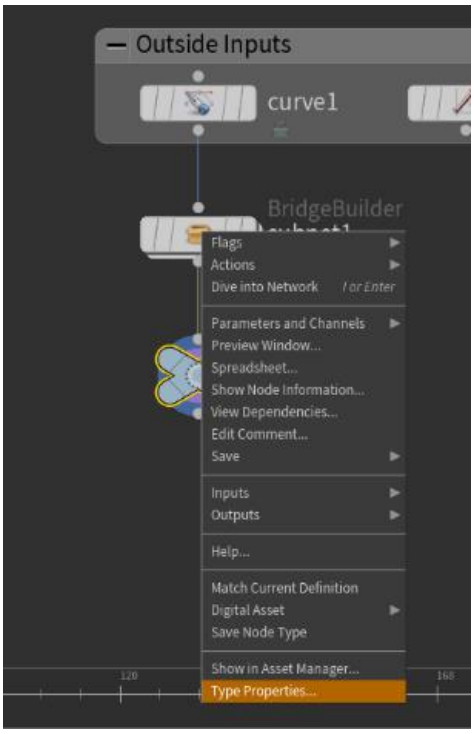




In the Dialog box that appears, give it an appropriate name and label. This is how the HDA will appear in your network and node search box.



I also suggest changing the default save location. I chose to save it in the same folder as the current hip file.



After accepting, the “Type Properties” menu will appear.

This menu is where the bulk of our work in an HDA happens.

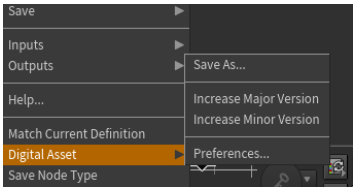
Several things to note:

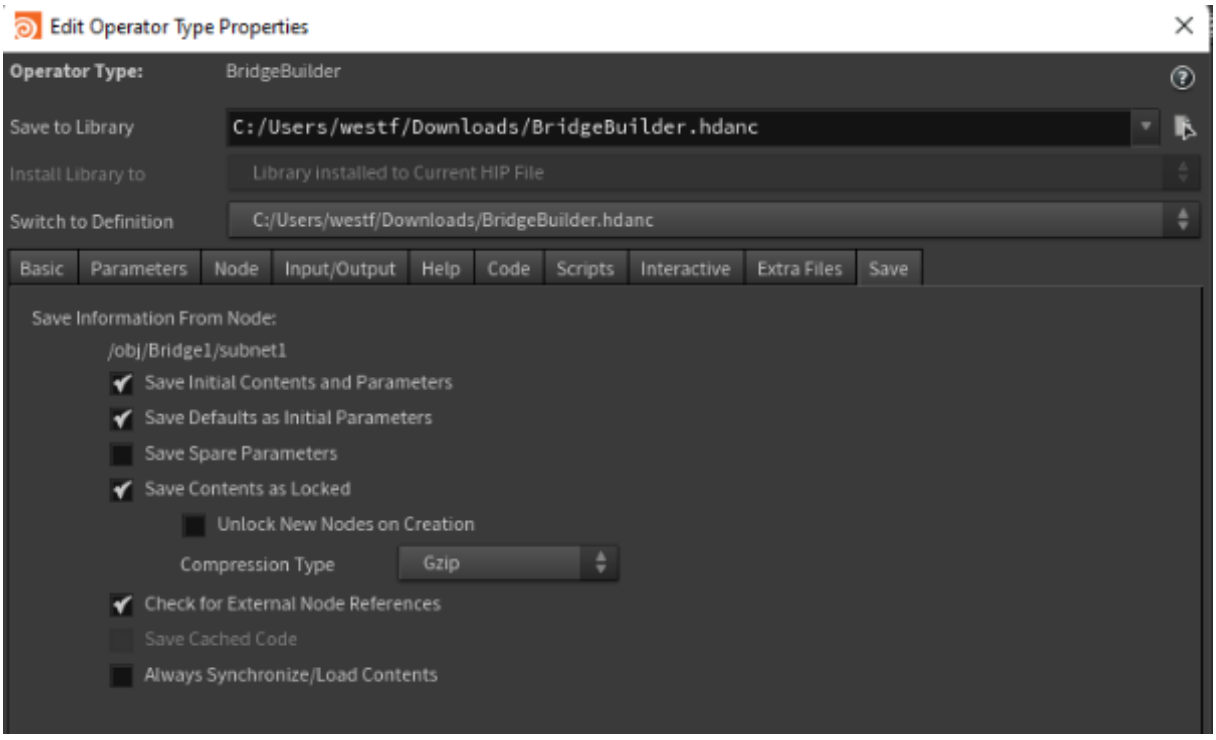
If the Type Properties menu is closed, you can find it by right clicking on a version of the HDA and clicking on *Type Properties*.

This menu, although very similar to the one used to edit a parameter interface on normal nodes, is not the same.

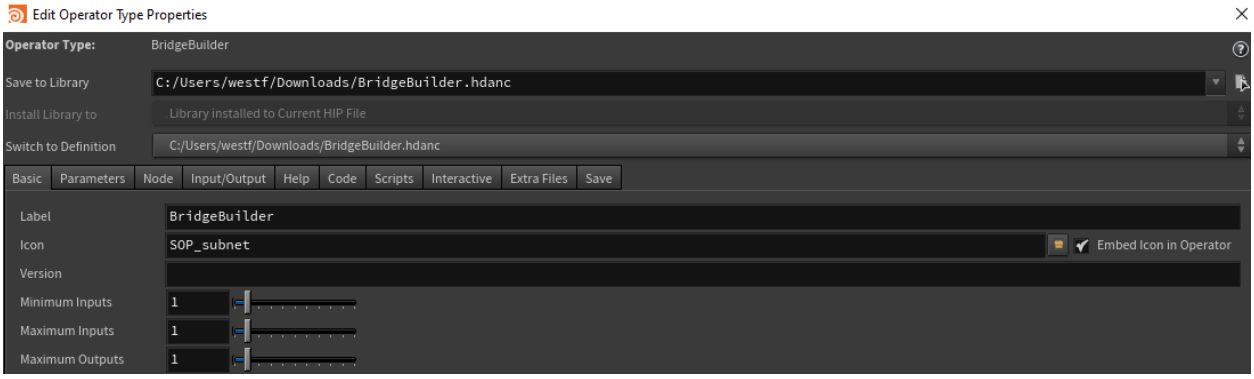
Any changes made in this menu must be saved. This can be done through *Digital Asset > Save As* in the right click menu.

Match Current Definition reverts changes to whatever was previously saved.





BASIC TAB



On the “Basic Tab” there are several parameters to take note of: Label, Minimum Inputs, Maximum Inputs, Maximum Outputs.

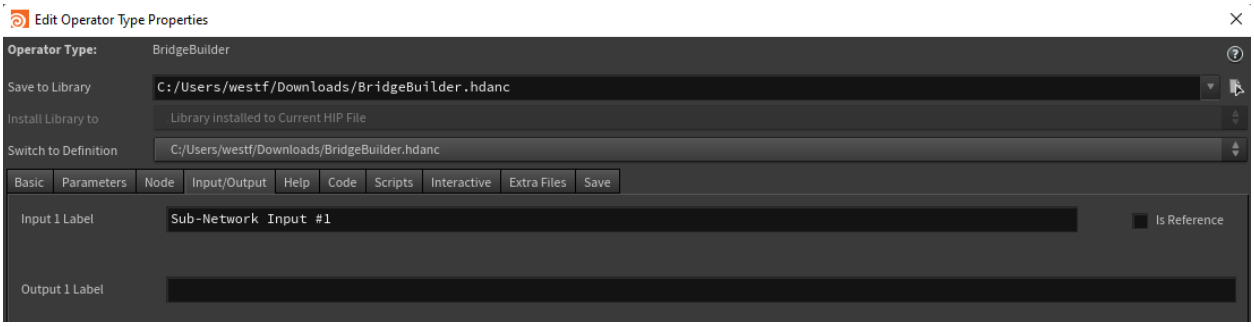
Label changes the name of the node. This is how you find a node via searching for it in the TAB Menu (the menu which lets you add nodes to a network).

Minimum Inputs changes how many inputs are required for a Node or HDA. Houdini will throw errors if there are not enough inputs. Future Reference this affects certain behaviors in Unreal as well.

Maximum Inputs the total amount of inputs allowed. These inputs should be made to be optional in your network if your network does not support optional inputs. This value should be the same as the minimum inputs. Future Reference this affects certain behaviors in Unreal as well. The Network inside a HDA can add inputs through creating a “Inputs” node and plugging a wire into it.

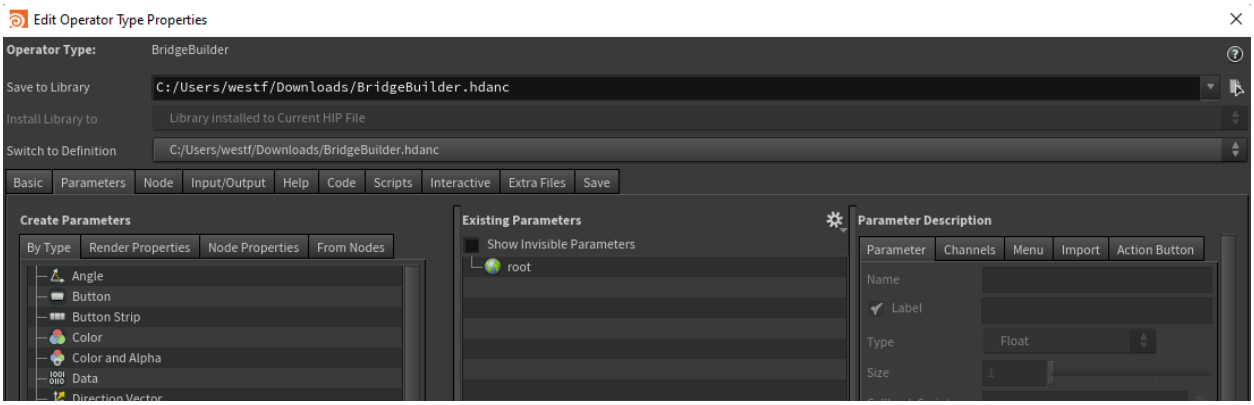
Maximum Outputs this is the changed the maximum number of outputs allowed in your node. Any Outputs past this maximum are ignored. The Network inside an HDA can add outputs through creating an “Output” node and pulling a wire out of it.

INPUT/OUTPUT TAB



The *“Input/Output”* Menu allows you to label the Input/Output Pins and Nodes of the HDA. This can be useful if you intend for other people to use it.

PARAMETER TAB

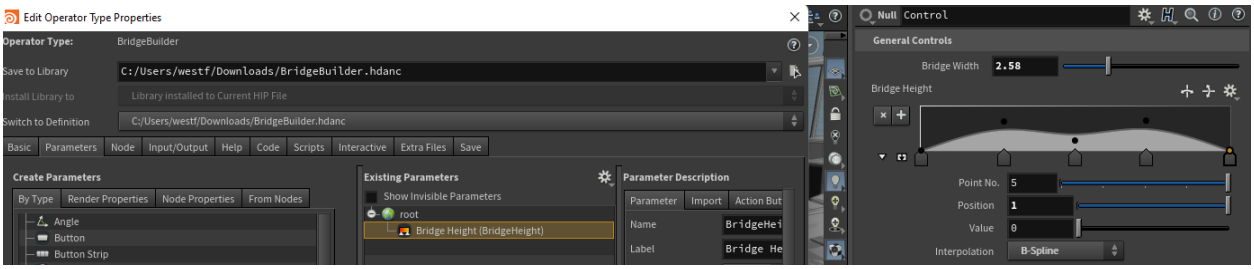


The Parameter Tab operates the same as the Parameter Interface that we can edit on other nodes, with some additional features.

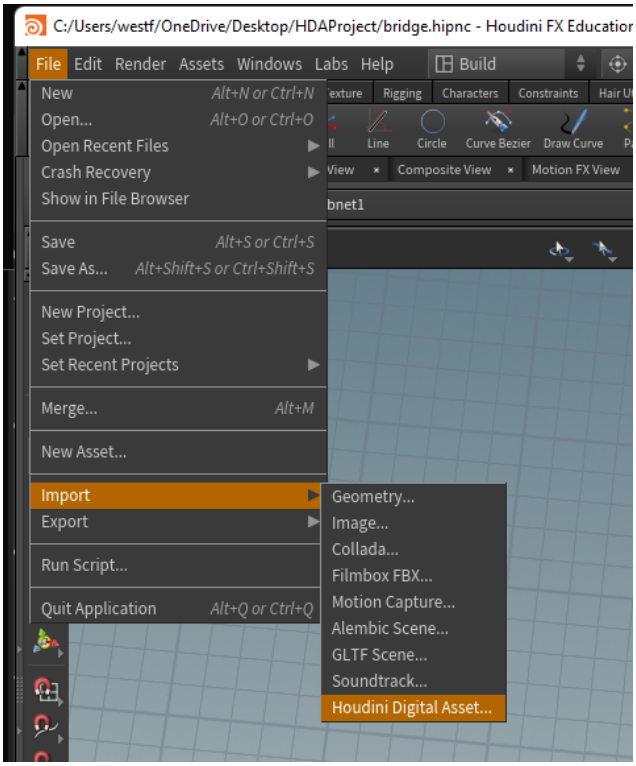
For this tutorial, we need to add the parameters from our control that are within the HDA to this Parameter menu.

1. Open the HDA's Type Properties.
   Do not close the menu.
2. Navigate to the Control inside the HDA. Click on it.
3. In the properties menu you can click and hold on a property name to drag it into the HDA's properties tab.

   This process should be repeated for all parameters you wish to control through the HDA.



IMPORTING TO OTHER HIP FILES



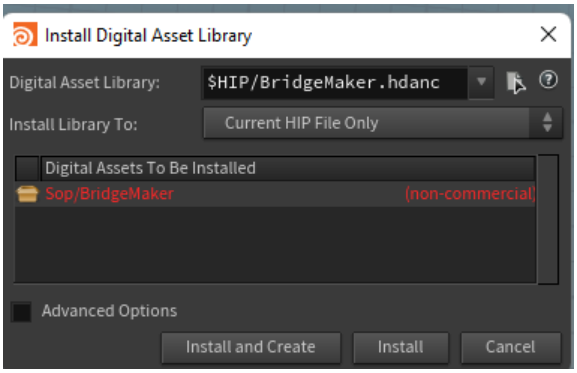
Installing an HDA into another .hip File is straightforward.

Navigate with the mouse to File > Import > Houdini Digital Asset in the top left corner.

I suggest installing your current .hip file.

“Install and create” will add the node to your network as well as the .hip file.

Installed nodes are searchable in the tab menu.



TO UNREAL

Now that we've created the HDA, we can import it into any .hip file and reuse it. Additionally, we can use this HDA with Houdini Engine. To demonstrate Houdini Engine we will use Unreal Engine 5.1.

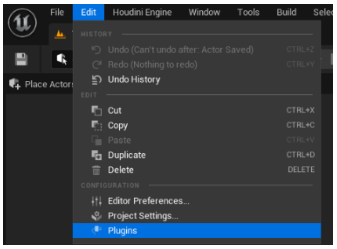
Before we jump into Unreal, there are a few things we need to do. First, we need to install Unreal Engine 5.1. Then, we need to install Houdini Engine. I will link several resources below.

Houdini Engine for Unreal - Documentation

Installing Unreal Engine - Documentation

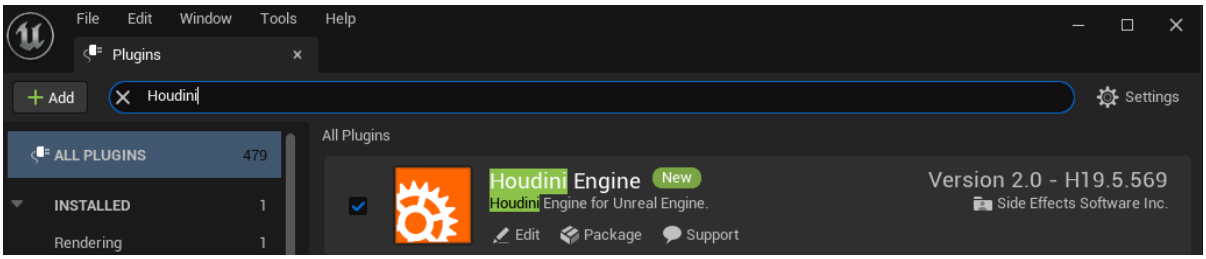
With Unreal Engine and Houdini Engine now installed, we'll want to make a new project. Let’s just make a 3rd-Person project and open it.

Third Person Template in Unreal Engine - Documentation



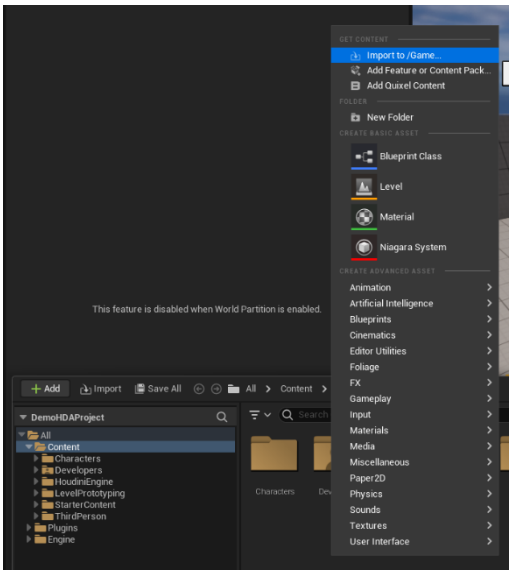
Ensure the plug-in is enabled.

1. Edit -> plugins
2. Search for Houdini
3. make sure it is checked.



INSTALLING THE HDA TO UNREAL ENGINE

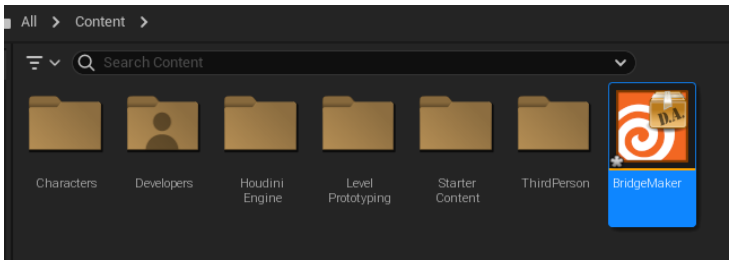
With the plug-in enabled, we can proceed to import our Bridge HDA. There are several ways to import it into your project.



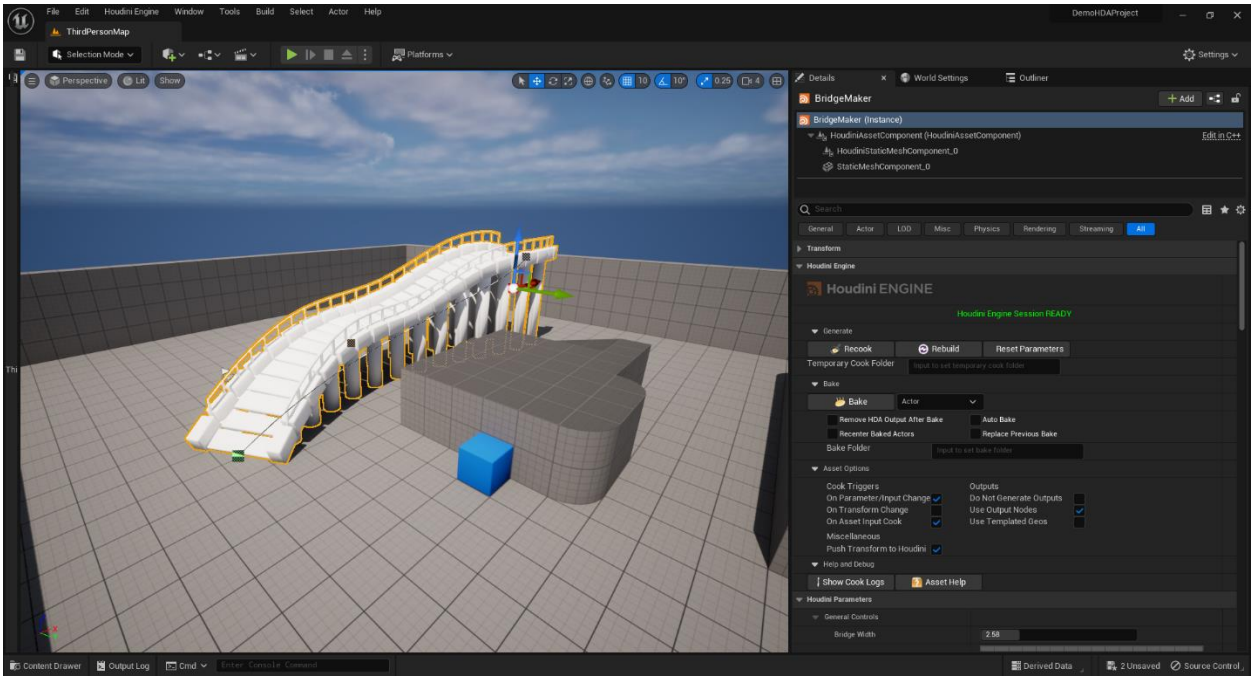
You can open a Windows browser, drag, and drop it into the content browser, or open the content browser, right-click inside, and select "Import to /Game."

Navigate to the HDA and import.

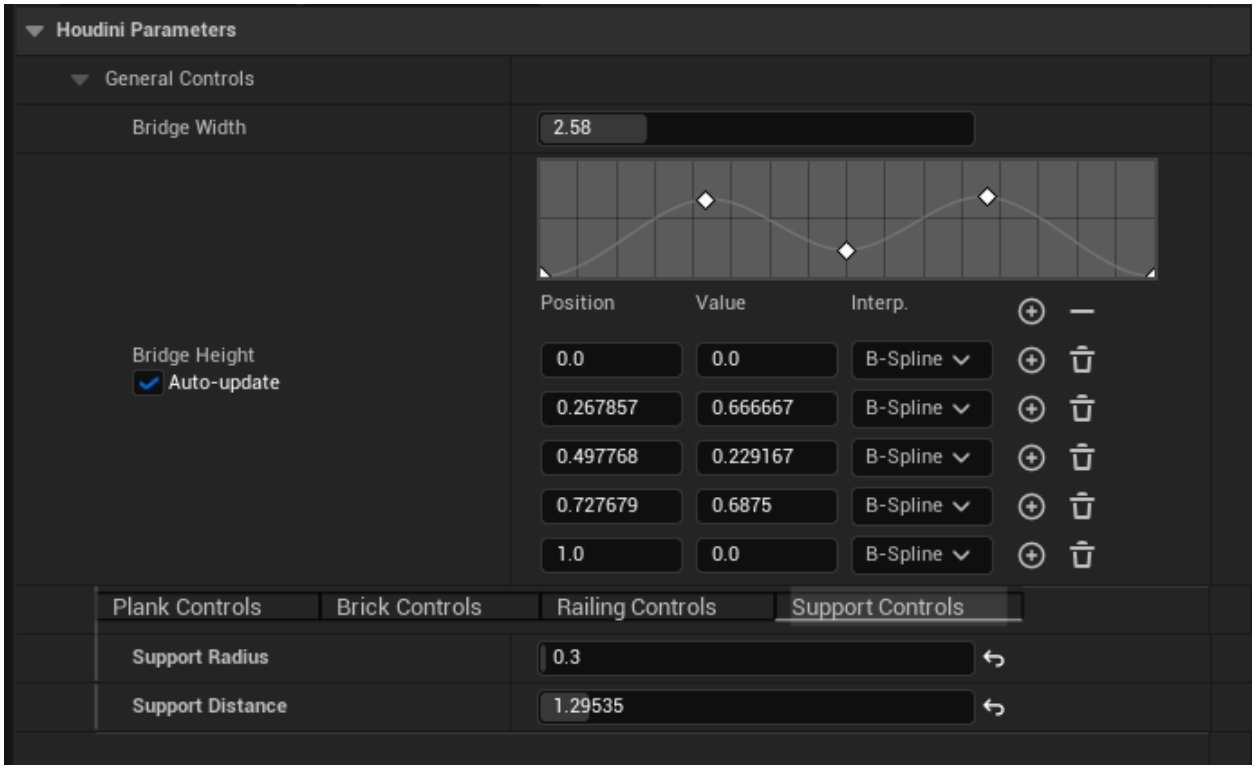
Once imported, the HDA asset in the content browser will look like the image below.

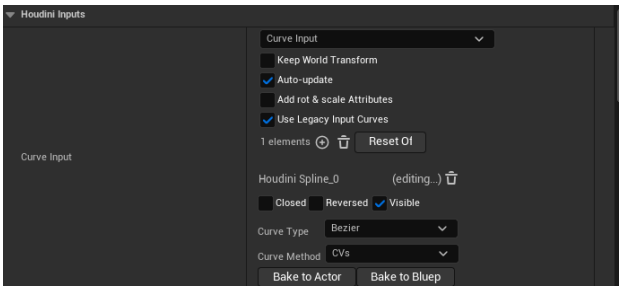


USING THE HDA



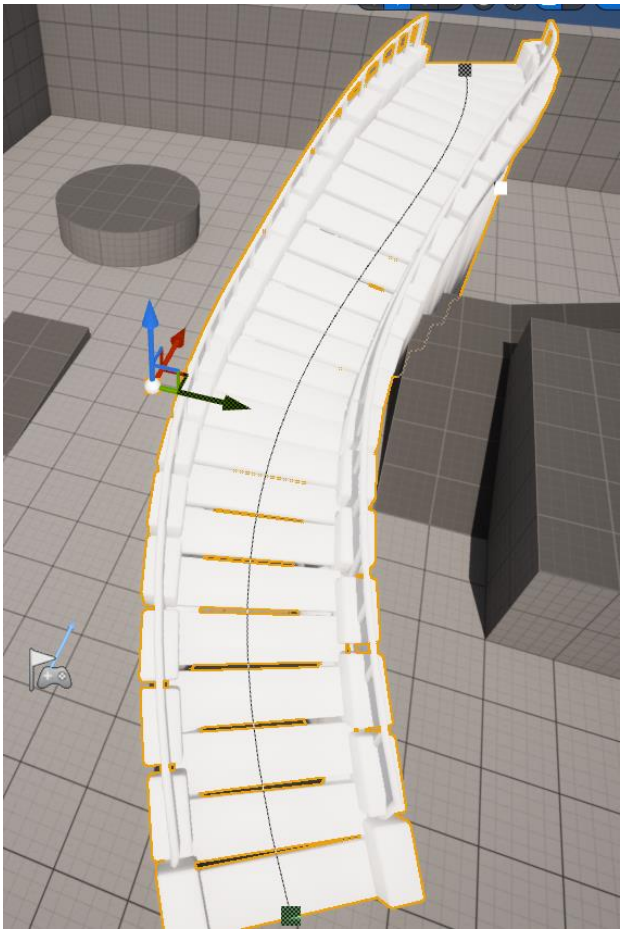
Using the HDA is as simple as dragging and dropping it into the scene you're currently working on. You'll notice in the details pane that there are a bunch of settings we can play with, in addition to the parameters we created in Houdini. These parameters work exactly as they did in Houdini.





Additionally, in the details pane, you'll see that it may have automatically set up your inputs. If it's incorrect, you can click the drop-down and select the correct input; these changes will affect how you can control the inputs in the scene.

For this project you'll remember were using a curve input in Houdini.

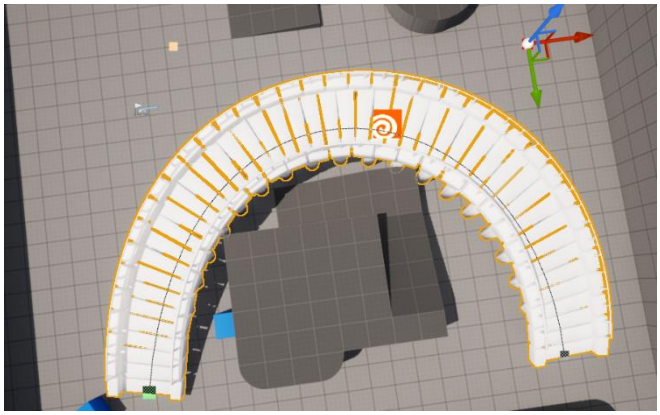


Curve type controls the general shape of the curve.

There should also be a curve that exists within the bridge.

Clicking one of the anchors and using the widget moves the curve.

Alt + dragging the widget adds a new anchor, allowing you to manipulate this HDA's shape.



There are several additional types of inputs, and I'll discuss just a couple. It's important to remember that you can have more than one input specified in the HDA, allowing different types of inputs to work together.

Input Type - Geometry: This allows you to input a mesh that exists in your scene. You can use it for boundaries of your tool, or you could edit the input mesh using an HDA.

Input Type - Curve: A great example is this project, but there are many other uses. You can create a silhouette and revolve the curve around a point. You can sweep one curve along another to create a mesh.

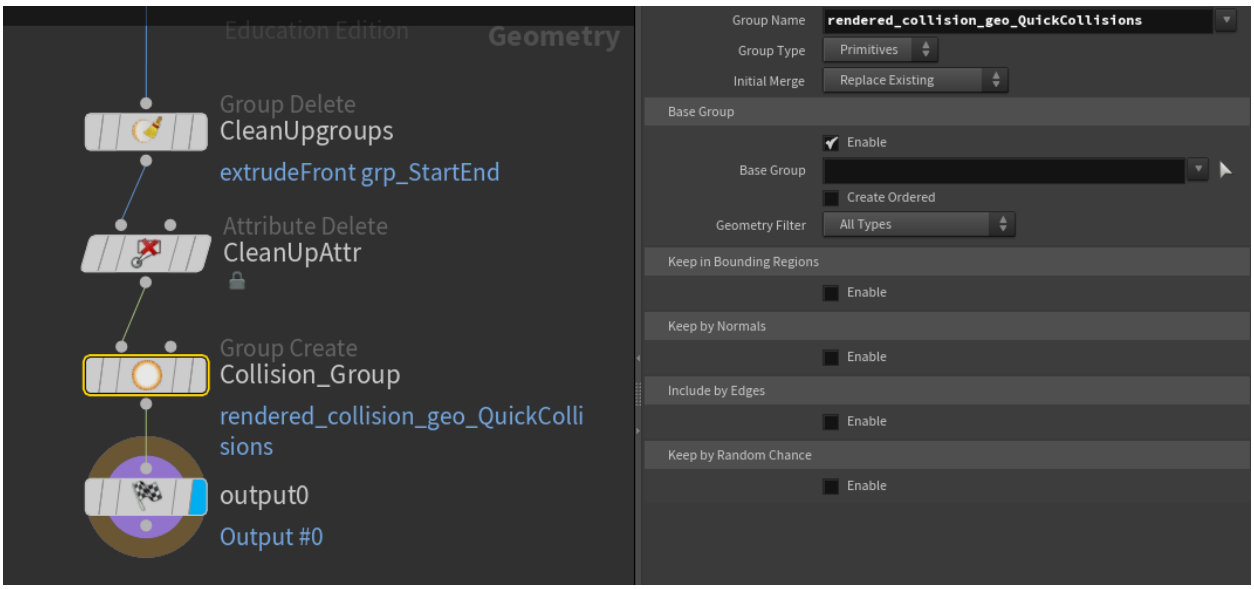
Input Type - World Outliner: This allows you to select multiple objects from Unreal's world outliner and input them into Houdini.

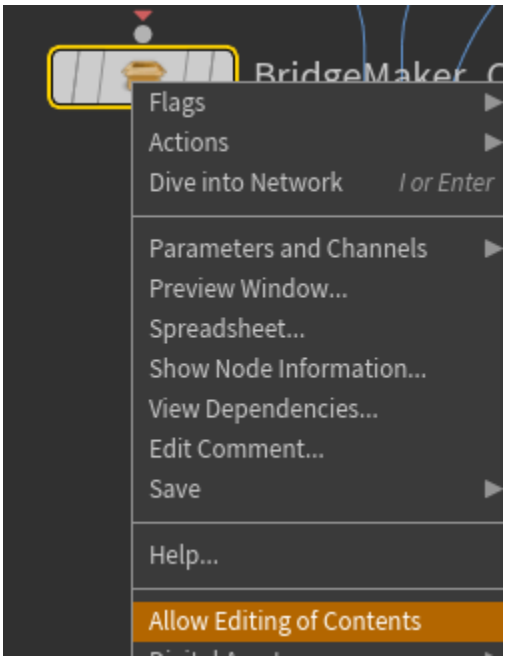
Input Type - Landscape: This allows you to import Unreal's Landscape as a height field into Houdini. It enables you to scatter new objects or edit the height field in Houdini. There is a very robust workflow for height fields in Houdini.

UNREAL SPECIFIC EXTRAS

It's important to note that Houdini Engine is powerful for speeding up your art pipelines, and there are many group types, attributes, and parameters that can be shared in specific instances through Houdini Engine.

A very popular use case is having your HDA also export geometry that will represent collisions. Then, the collisions can be finely tuned by Houdini to work exactly how you want, as opposed to letting Unreal auto-generate the collisions.





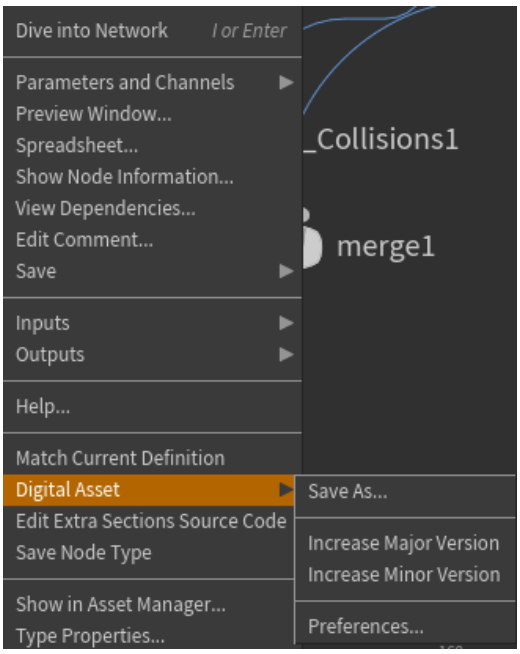
Since our bridge does not currently have collisions, let's address this.

Open Houdini, import a definition of the HDA you wish to work on.

Add the node to a network, right-click, and allow editing of contents.

Dive into the node, and right before the output, you can add a group create and name it exactly as specified above.

rendered_collision_geo_QuickCollisions



Move out of the node, right-click on it, and hit "Save As." Reimport the HDA to the Unreal project, and now, when dragged into the scene, the bridge should have collisions.

This is a quick, easy method to add collisions; however, most applications of this would be more fine-tuned within the HDA. The collisions do not normally need to be as detailed as the visual geometry.

Special Attributes and Groups - Documentation by SideFX explains the different attributes and groups that will be shared between Houdini and Unreal. This is an extremely useful reference when creating or designing a tool.

Examples of Information an HDA can create and give to unreal.
Materials, Enable Nanite, Instances, Foliage, collisions.

RESOURCES

Installation of Houdini Engine – Guide by SideFX to help with the installation of Houdini Engine.

Introduction to Digital Assets – Documentation made by SideFX.

Houdini Digital Assets | 1. Introduction to HDA – Video Tutorial made by SideFX.

Project Titan – A project by SideFX in Unreal Engine 5 using Houdini Engine focusing on Tools.

Installing and managing HDAs – Houdini Documentation for Installing HDAs and HDA Libraries
A user or studio can create entire libraries of HDAs filled with tools the commonly use. This is very similar to how Houdini LABs Tools works.

Houdini Engine for Unreal - Demo of Houdini Engine

TIPS

REFERENCE OTHER NODES

You can look inside most nodes in Houdini. Use this feature to look inside and see how they work. This can help you when researching how to create your own specific HDAs.

Simply double clicking on the node to travel inside.

If you would like to adjust the insides of a node.

- *Right click on the node and choose "Allow Editing of Contents."*
- *Then dive inside the node.*

LABS TOOLS

LABS is a tool set that can be Installed with Houdini that adds several nodes that are more experimental. This toolset is a great example of a set of HDAs that can be created for a project. The nodes in this plugin can be great resources.

GENERATING LOTS OF VARIATION WITH HDAS

While you can generate several assets manually or using an HDA there are projects where you need a substantial number of variations. Houdini's Procedural Dependency Graph (PDG) or Task Operators (TOPs) can be powerful tools for accomplishing this.

However, this topic is too large for this document and can be complicated.

FINAL THOUGHTS

WestFoulks@gmail.com
This document was created mostly with my existing knowledge, so things might become outdated. If anything is confusing, anyone is free to email me at the above address for a more detailed explanation.