
Inochi2D

Release 0.7

Luna the Foxgirl

Sep 10, 2023

INOCHI CREATOR

1	What is Inochi2D?	3
1.1	Meet Ada	4

On this page, you will find documentation for the officially supported projects under the Inochi2D banner. Use the sidebar on the left to select a project to get documentation from, and a topic you want to read about.

WHAT IS INOCHI2D?

Inochi2D is an open source specification for a 2D character animation system that allows characters to be animated in real-time for games and VTubing. Inochi2D achieves this by deforming textures in real time based on parameters the rigger sets in Inochi Creator.

This documentation page covers:

- Inochi Creator
- Inochi Session
- Inochi2D Specification for implementers
- Inochi2D SDK for developers

Inochi Creator is the official tooling to create animated rigs for use with Inochi2D compliant software and games.

Inochi Session is the official tooling to use Inochi2D models for VTubing, using various forms of body and face tracking.

Inochi2D Specification is meant for developers that want to implement Inochi2D in their preferred engine.

Inochi2D SDK is the official Software Development Kit that allows integrating Inochi2D in to your games and software.

1.1 Meet Ada



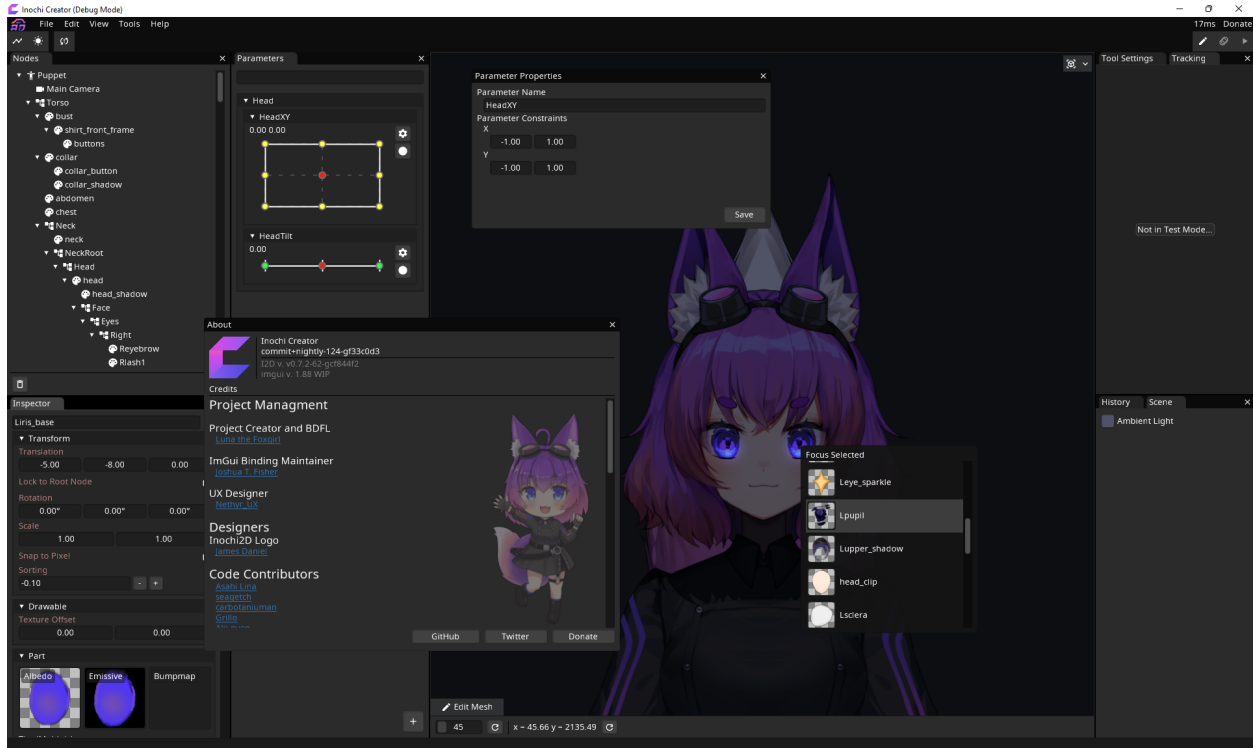
Ada is Inochi2D's mascot, an airheaded, friendly and helpful foxgirl. She will appear throughout the documentation with tips, warnings and other tidbits of information that may be useful.

Hello, there!



Nice to meet ya! I'm Ada, local vertex engineer fox! Please don't mind me shedding a bit of fur on your experience
Oh, and don't forget: If you have questions not covered here you can always reach out to us on [our Discord!](#)

1.1.1 About Inochi Creator



Inochi Creator is an open source editor for the [Inochi2D puppet format](#). This application allows you to rig models for use in games or for other real-time applications such as [VTubing](#). Animation is achieved by morphing, transforming and in other ways distorting layered 2D textures in real-time. These distortions can trick the end user in to perceiving 3D depth in the 2D art.

If you are a VTuber wanting to use Inochi2D we highly recommend checking out [Inochi Session](#) as well.

To get started check out our [Getting Started Guide](#)

1.1.2 Getting Started

System Requirements

The following system requirements have to be met before running Inochi Creator.

	Minimum	Recommended
Processor	x86_64 or ARM64 CPU ¹	Recent x86_64 or ARM64 CPU ¹
Memory	512 MB	1 GB or more
Disk Space	100 MB or more ²	500 MB or more ²
Graphics Processor	OpenGL 3.2 Support ³ , 512 MB VRAM	OpenGL 3.2 Support ³ , 1 GB VRAM or more
Display Resolution (Pixels)	960x800	1920x1080 or higher

¹ x86_64 and ARM64 refers to 64 bit CPUs by ARM, Intel and AMD.

² Disk Space requirements depend on the size and amount of models you have on your system.

³ OpenGL 3.2 is supported on the following GPUs:

- Gen 6 or later Intel HD Graphics on Linux and macOS

Inochi Creator does not support 32 bit systems currently to reduce the maintenance burden.

Downloading Inochi Creator

Inochi Creator is released in 2 streams, **Stable** and **Nightly**. Stable releases are generally recommended for having better stability, but it lacks behind in features with Nightly. Meanwhile, Nightly is built by GitHub from the latest code every night, this means it has the latest features, but also the latest bugs.

Stable is available to download on:

- [Our Website](#)
- [Itch.io](#)
- [GitHub](#)
- [Flathub](#)

Nightly is available to download on:

- [GitHub](#)

If you use *Itch.io's Launcher* or *Flatpak*, Inochi Creator will automatically stay updated. Our ZIP and DMG packages do currently not auto-update, but we plan to change this in the future.



Danger:

Nightly builds are not stable, we really don't recommend using them unless you know what you're doing. Nightly builds are likely to crash and can corrupt your files in the process!

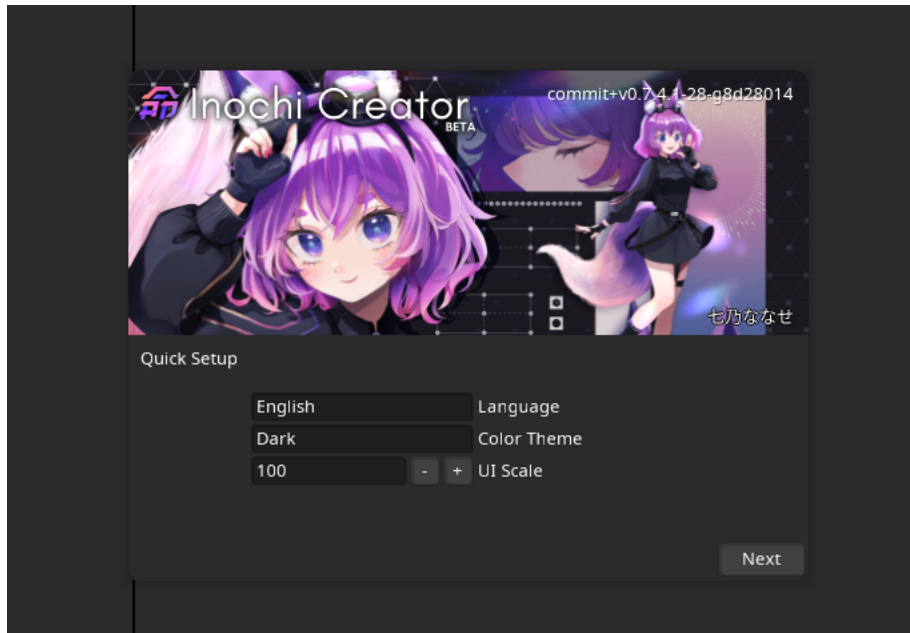
Make frequent backups!

-
- Gen 7 or later Intel HD Graphics on Windows
 - Nvidia GeForce 8 series or later
 - AMD Radeon HD 2000 series or later (marketed as TeraScale 1)
 - Apple Silicon M1 or later (OpenGL 3.1 with required extensions)

VRAM requirements depend on the size of the model and texture count.

First Time Setup

First time you launch Inochi Creator the Welcome Screen will pop up, you can set some basic settings here like Color Theme, Language and UI Scaling⁴.



Inochi Creator supports multiple languages, so if English is not your first language you may change the language, by clicking the dropdown in the “Language” section

After having set up the basic settings you are then free to start creating Inochi2D models!

You can download example models on the same page you downloaded the version of Inochi Creator you installed.

Notes

1.1.3 Your First Model

The following is a small tutorial teaching you the basics of modeling in Inochi Creator.

Starting with a minified form of the Aka PSD you will learn the basic layout of the UI, how to mesh parts, how to use Parameters, adding Physics and so forth.

First make sure you have *Inochi Creator installed and basic setup is done* before continuing to the next section.

⁴ You will not see the UI scaling option on macOS

This is intentional, UI Scaling is handled by the operating system on systems running macOS, To change the UI Scaling, open the system settings and change the UI scaling there.

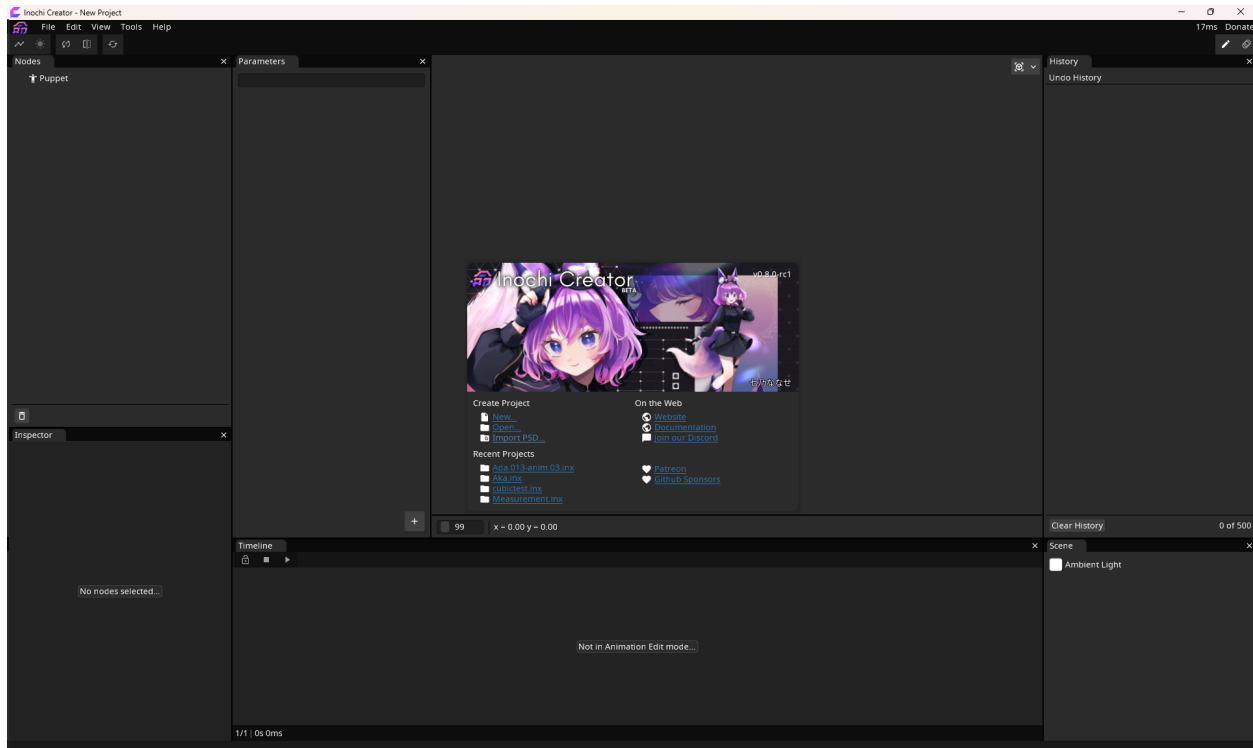
1. Setting Up

Getting the Tutorial Aka Model

Download the tutorial Aka model from [here](#).

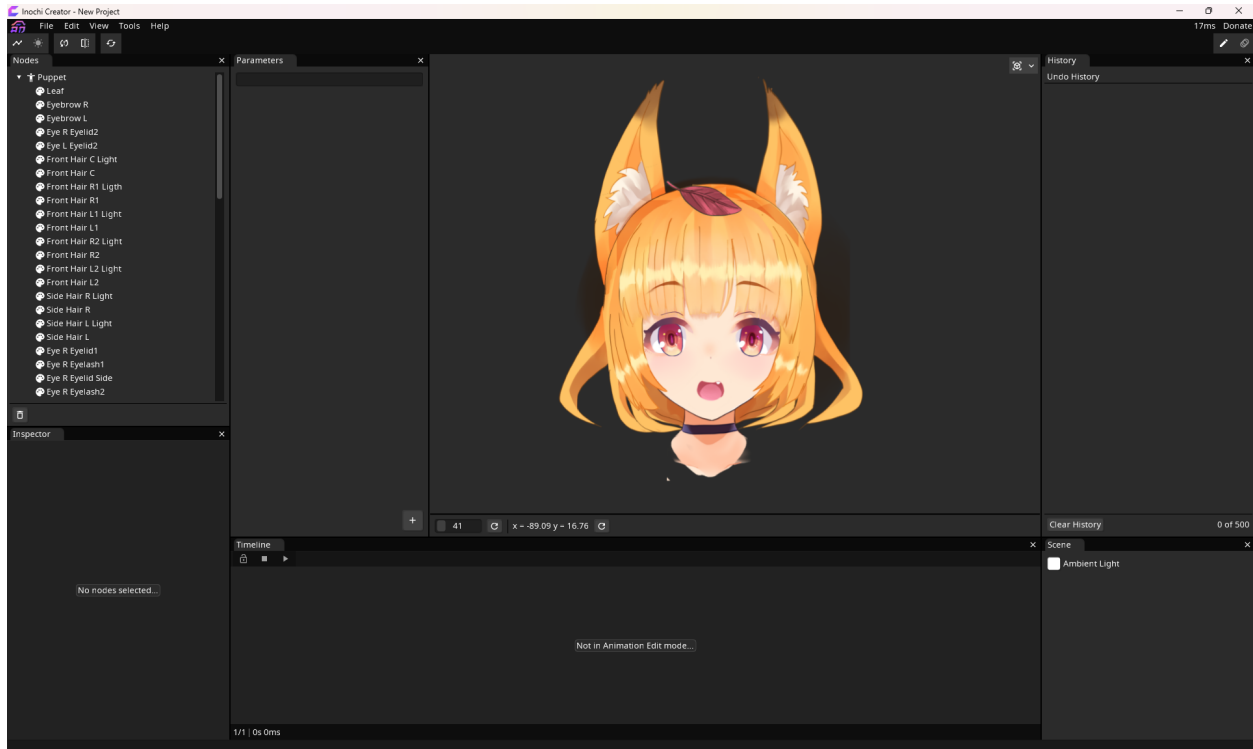
This is a version of the Aka example model by Seagetch, but with only the head, which is what we'll focus on.

Importing the model in to Creator



To import Aka, either click **Import PSD** within the startup window, or select **File->Import->Photoshop Document** from the main menu.

Once selected Inochi Creator should look like this:



Something's off...



You may wonder why the shadows are looking all garbled on her right now, right?

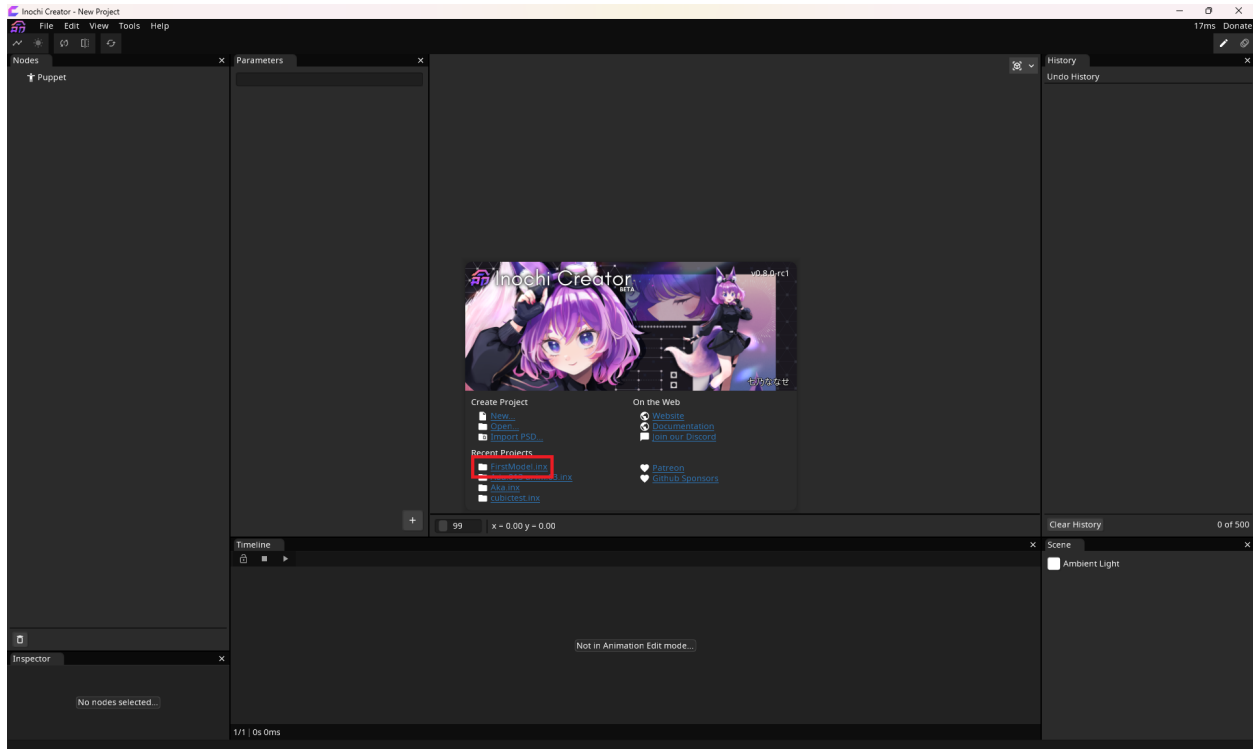
All the shadow layers in the PSD are just bundled on to each other, so shadows cast shadows on shadows. We'll fix that in the coming chapters!

Saving and Loading a Model

Now that we have the model loaded in to Inochi Creator, let's save it!

Go to the **File->Save** option in the main menu and select where to save it. For now, let's save it as **FirstModel** in a location of your choosing.

If you now close and re-open Inochi Creator your project should be visible in the Recent Projects list!



2. Layouting

Now that we have Aka's head loaded in to our project and we can save and load it, the next step is to lay out the model for rigging, this step is called Layouting.

You can think of layouting as building out the skeleton for our model.

For this chapter we'll be following the diagram below



```

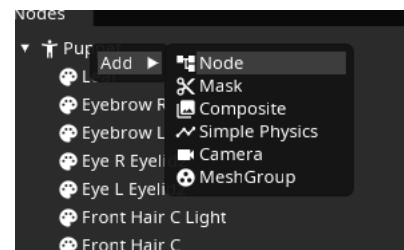
Neck Root1
└─ Head Root2
    └─ Left Eye Root3
    └─ Right Eye Root4
    └─ Mouth Root5
    └─ Front Hair6
    └─ Leaf Root7
    └─ Left Ear9
    └─ Right Ear8
    └─ Etc...

```

We'll be using the basic *Node* to build out this skeleton, which will make some rigging easier later.

Creating our skeleton

1. Adding a Node



To add new nodes to the tree right click an existing node and select the type of Node you want within the Add submenu, in this case we want the Node node type.

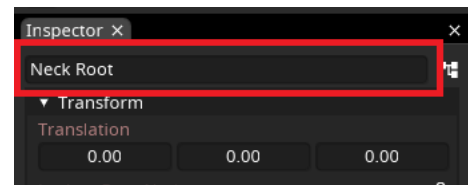


Tip:

The Add submenu will add the Node as a child to the node you right clicked!

See the [Nodes](#) page for more information about how parenting works!

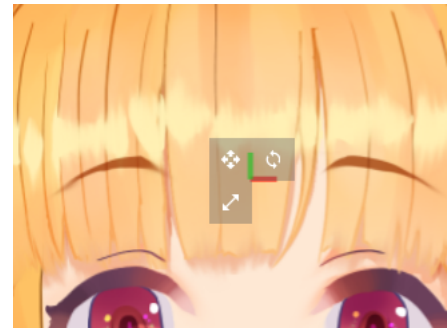
2. Renaming a Node



After adding our new Node, we'll see it appear as "Unnamed Node" within the node tree. We should probably rename it so that it's easier to know what it does at a glance.

Click the node and within the Inspector panel, rename the node to "Neck Root"

3. Transforming a Node



If you look in the viewport with our Neck Root selected you will see the following gizmo pop up.

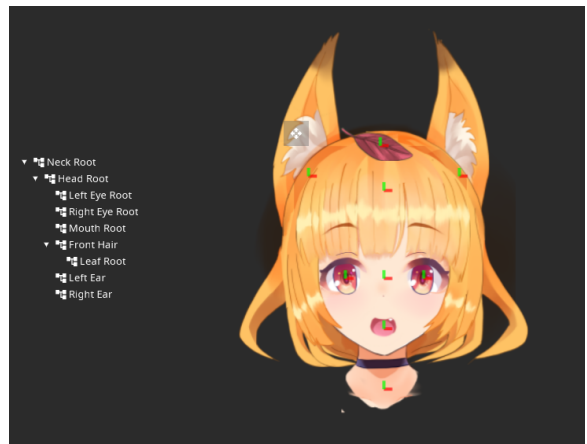
This gizmo tells you where the node is in the scene, the colors represent where the X, Y and Z axis of the node are pointing

X is red, Y is green, Z is blue. Z is currently obscured by the fact that it's pointing away from the camera.



Either by dragging the move gizmo (the button that looks like arrows pointing in 4 directions) or by editing the Translation section in the Inspector, move Neck Root to the bottom of the neck.

4. Finishing the basic Layout



With these 3 basic functions you should be able to create a skeleton, remember that Add adds nodes as children of another node and that the position of every node is *relative* to their parent node!

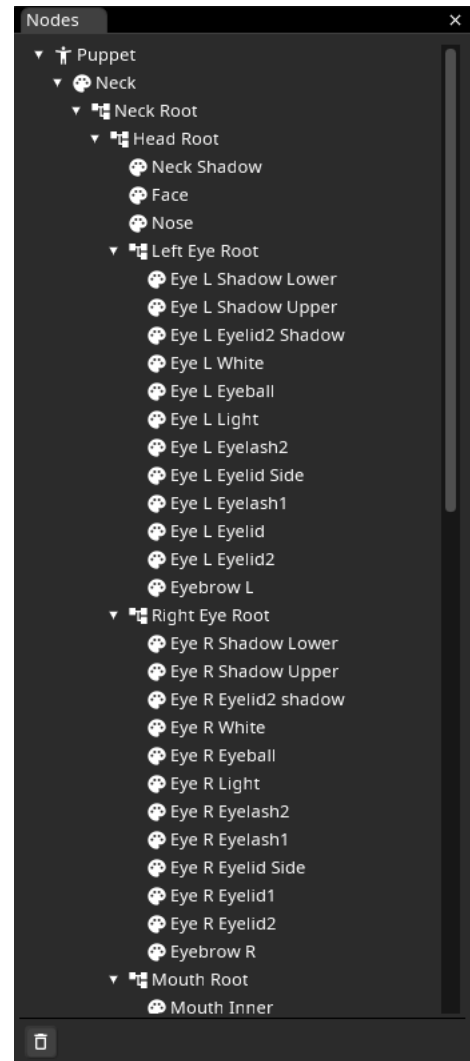
Try to re-create this layout using the techniques above.

A challenge



Now that you know the basics of creating a layout, try creating additional layout points for the nose and back hair!

Moving things around



Now that we have our skeleton, it's mostly just about moving things in to the right place, drag and drop all the Part Nodes (the ones with the color palette icon) in to their place!

Try to roughly follow this layout!

You can now try to select Neck Root and rotate the node around the Z axis (3rd box in the Rotation section or using the rotate Gizmo), notice that all the children of the Neck Root rotate with it!

Congratulations! You've finished laying out your model, for now at least.



Tip:

You can download the resulting model file [here](#), if you need a reference!

1.1.4 Nodes

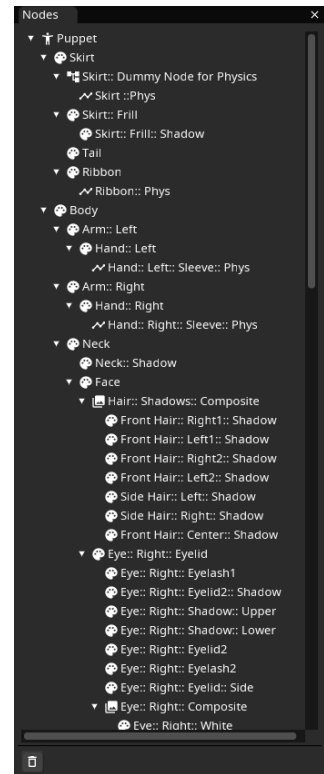
Inochi2D is based of a concept called a [Scene Graph](#).



Note:

If you've used a modern game engine such as Godot, Unity, Unreal Engine or 3D modeling software such as Blender you may already be familiar with the concept!

Think of Godot Nodes, or Unity GameObjects as the scene graph.



The scene graph is accessed via the Nodes panel, which displays them in a tree view.

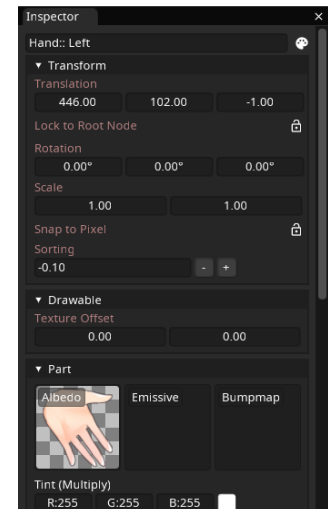
Each item in the tree is called a *Node*, each Node has a type on top of that.

Every node has the concept of a Transform in common. A transform contains information about the position, rotation, scale and sorting index of the node.

A node which is attached to another node is called a *child Node*, and the Node which the child is attached to is called the *parent Node*.

Child Nodes inherit the position, rotation, scale and sorting index of their parents, which means if a parent nodes moves around, so will its children move around relative to it.

The icon that a node has is used to denote its type to you.

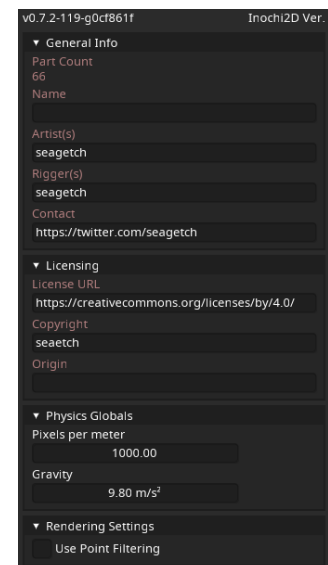


The properties of a Node are accessed via the Inspector panel, depending on the type of the Node, the items in the Inspector panel will change.

The Inspector Panel's contents are divided into sections which can be collapsed by clicking their header. Each section contains settings for a specific subtype of that Node.

For example, a *Part* Node is based on the *Drawable* Node type, which in turn is based on the *Node* type. As such you will see properties for Node (Transform), Drawable and Part as sections.

Puppet



The puppet node is a special node that you can not move. It contains properties for model information, model-wide settings such as physics system options and rendering options.

General Info & Licensing

These sections stores general information that most software reading the model will show to the user.

As for licensing, note that unlike competing software, we do not claim any form of license or ownership over the models produced with our software.

As such you are free to specify any license you want and have to enforce said license yourself.

Physics Globals

This section stores settings related to the physics handling of the model

Pixels per meter

How many pixels count as 1 meter in the physics system. You can adjust this value to match the physics to the canon height of the character, or otherwise to your liking.

Gravity

How strong the gravity is for the model, by default it uses Earth gravity of 9.8 m/s squared.

Rendering Settings

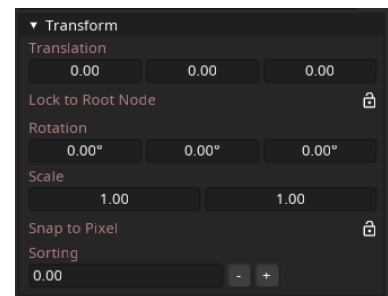
This section stores settings related to the rendering of the model.

Use Point Filtering

If you're making a pixel art model, this setting will force the rendering to preserve the hard edges of pixels. This will allow you to save texture memory by requiring a lot less scaling of your model.

Competing packages such as Live2D normally requires a ~5000% (50x) scale-up of your model to look good, Inochi2D can get away with 200% (2x) if you turn this option on.

Node



The Node is the base Node type which all other Nodes inherit from, it only stores a Transform.

The transform stored by a Node will be inherited by its children. As such, children's position, rotation, scale and sorting index is relative to their parent.

A translation of 0, 0, 0 will result in the child node being in the center of its parent node.



Tip:

If you put a Node at 0, 0, 0, then add a child Node at 0, 0, 32 and rotate the parent around the Y axis, the child node will spin around its parent!

How cool is that?

Translation

Translation is a fancy mathematical word for “position”, the translation defines where the Node is in 3D space.

Lock to Root Node

This option allows the translation, scale and rotation to be independent of the parent. In this mode the node moves only based on the special root node of the model.

This is useful if you have some static element you want to only move when the model itself moves.

Rotation

The rotation of the Node in degrees in 3D space.

Scale

The scale of the Node, as Nodes are generally 2D objects in 3D space you can only scale them on the X and Y axis.

Snap to Pixel

Snaps the Node to the nearest pixel when moving.

Sorting

The sorting index of the Node, smaller (and negative) numbers are closer to the camera, while bigger numbers are further away.

This value is used to define what Part is drawn in what order.

1.1.5 Exporting Video

The following topic explains how to set up Inochi Creator for video export and how to export video with it.



Note:

FFMPEG is needed for video export! Without it the option will be grayed out.

If you're not sure whether ffmpeg is installed on your system you can run the following in your command prompt/terminal!

```
ffmpeg --version
```

If FFMPEG is not installed continue to the *Installing FFMPEG* section.

Otherwise continue to the *The Export Video Dialog* section.

Installing FFMPEG

Installing FFMPEG on Windows

Currently you'll need to manually install ffmpeg on your system on Windows, we plan to support an automatic installer within Inochi Creator eventually.

Manual local installation

1. Go to <http://ffmpeg.org/download.html>, and find the Windows packages.
2. Follow the instructions on the page you are redirected to
3. Once you have the ZIP archive find the bin/ folder within it.
4. Put ffmpeg.exe next to inochi-creator.exe.

System-wide with Chocolatey

If you have [Chocolatey](#) installed you can install ffmpeg system wide with it.

```
choco install ffmpeg
```

Installing FFMPEG on Linux

Download FFMPEG from your system's package manager, on Ubuntu and Debian based systems the following command will install ffmpeg

```
sudo apt install ffmpeg
```

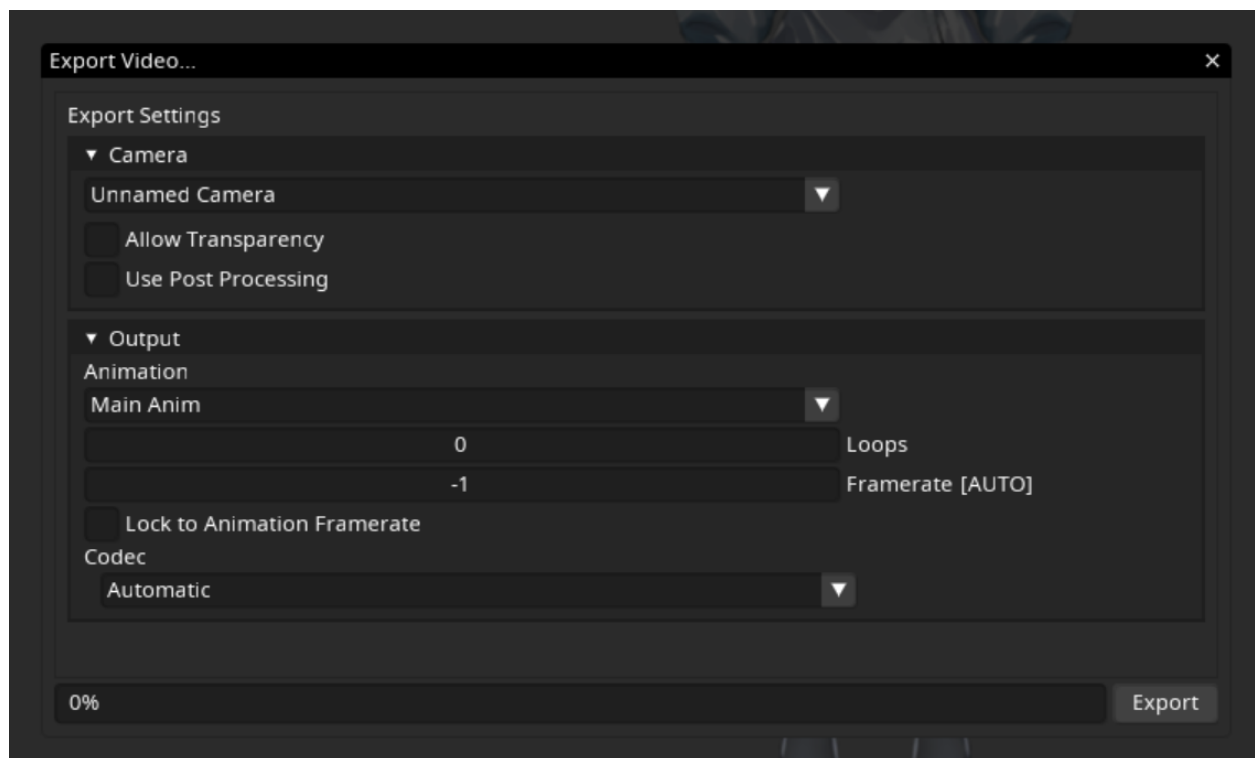
Installing FFMPEG on macOS

FFMPEG can be installed on macOS through homebrew, which you can install here <https://brew.sh/>

Once brew is setup you can run the following command to install ffmpeg to your system.

```
brew install ffmpeg
```

The Export Video Dialog



The video export dialog allows you to specify settings for the exported video.

Some settings may be discarded due to incompatibility with the output file format, such as discarding transparency for opaque video formats.

**Note:**

You will need at least 1 animation and 1 camera in your scene to export video!

Otherwise you just get a sad error box telling you to go do that.

Starting an Export

To start an export, go to **File->Export->Video** from the main menu.

You will then be presented with a dialog asking you where you wish to save the video file.

If the option is grayed out, see [Installing FFmpeg](#).

Camera

The following section allows you to adjust some settings about the camera for export.

Camera dropdown

The camera dropdown lets you select which camera to use, if the camera is affected by the Animation, then you can export video with zooms, pans, etc.

Allow Transparency

Whether Inochi Creator should try to export the background with transparency, only some formats and codecs support this, though!

Use Post Processing

Whether Inochi Creator should use the scene lighting in the exported video, this allows your model to have glowing elements, etc.

Output

Animation dropdown

Lets you select which animation to export to file.

Loops

How many times the main section of the animation should be looped, setting this value to 0 is semantically the same as setting it to 1.

Framerate

The framerate of the video file to export

Lock to Animation Framerate

Disables smoothing of mismatched framerates between video export and animation framerate.

Codec

Which video codec to use for export, by default Inochi Creator will try to automatically infer the codec.

Eg. for H.264 video you'll want to export an mp4 file, with the H.264 codec selected in the list.



Warning:
wrong!

The assumptions Inochi Creator makes with the Automatic option may be

If you want to be 100% sure that Inochi Creator uses the right codec, please set it from the list!

1.1.6 Inochi Session

Due to a planned massive restructuring of Inochi Session, documentation for it will be sparse at the moment, sorry!

1.1.7 About Inochi2D

Inochi2D is a library for realtime 2D puppet animation and the reference implementation of the Inochi2D Puppet standard.

It works by deforming 2D meshes created from layered art at runtime based on parameters, this deformation tricks the viewer in to seeing 3D depth and movement in the 2D art.

The Inochi2D Project provides official tools for manipulating and using these models for VTubing, via *Inochi Creator* and *Inochi Session*.

The Inochi2D section covers the specification for implementing your own instance of Inochi2D, if you wish to know more about the API of the Inochi2D D package, see the [dpldocs page](#).

1.1.8 API Reference

1.1.9 Frequently Asked Questions

We get asked some questions often, we've compiled some of them and listed answers to them here.

How much does Inochi2D Cost?

Nothing. Inochi2D is free, BSD 2-clause licensed software. You may download and use it at no cost, with the only limit being that you may not remove the license text from the software. You can *optionally* support the development monetarily via [Patreon](#) or [GitHub Sponsors](#), though this is completely optional.

Is Inochi2D ready for use?

Inochi2D and the accompanying tools are still heavily work-in-progress. They are usable for usecases we officially support, but it's still evolving very quickly so we can't promise that it is ready for game development. Some of the tooling is still a work in progress and will be improved with time to be easier to use.

Can I load my PSD file in to Inochi Creator?

Yes, under the File->Import menu you can import Photoshop PSD documents, do note that PSD support is still a work in progress and some metadata may load incorrectly (blending modes being applied wrong or layers being hidden that shouldn't) You may need to do some manual work setting blending modes and unhiding/hiding nodes in the node tree in that case.

Can I use my existing Live2D models with Inochi2D?

No, Inochi2D is an entirely new open format that is on a fundamental level incompatible with Live2D. As such the model will have to be re-rigged for Inochi2D. Inochi2D should for the most part be compatible with PSDs cut for Live2D use, though.

What advantages does Inochi2D have over Live2D?

Inochi2D has multiple advantages, some still in development:

1. Inochi2D is open source and has no licensing fees for integration in your project
2. Inochi2D natively supports multi-texturing for render passes, this means you can add special rendering effects to your model at low rendering cost, such as eyes that glow in the dark, make your model react to lighting in a scene and more.
3. Inochi2D is extensible, if you're a developer you can add new Node and Automation types to Inochi2D without breaking other applications that support Inochi2D.
4. Inochi2D combines all the required assets and data in to its INP file format, this format additionally allows apps to save settings to your model, allowing those settings to transfer over if you move the model to a new computer.
5. Inochi2D is quite light on PC resources, while Inochi2D has not had a proper optimization pass yet the performance is already pretty decent. We'll be improving performance further over time as we get closer to 1.0.

Does Inochi2D support pre-defined animations?

From version 0.8 of Inochi2D you can create animations and save them to your model.

Does Inochi2D do face tracking?

Inochi2D in of itself is just a format for 2D puppets/models. The official Inochi Session app and facetrack-d library provides you with the ability to use your Inochi2D puppet in conjunction with face tracking technologies.

Why does text in Inochi Creator in X language look like question marks?

Inochi Creator uses ImGui to render its UI, as of current ImGui does not support dynamically loading glyphs from fonts, and loading every text character under the sun would be bad for performance. Until dynamic character loading is in place English and Japanese are the main supported languages. Translations may specify metadata files with font and glyph range information for their respective languages, but text from those languages will only render properly if you set Inochi Creator to that language. Inochi Creator does not support languages that require advanced font shaping (eg. Arabic) as well as right-to-left languages currently due to ImGui limitations. If those limitations some day get solved we'll let you know.

I would like to use Inochi2D for blockchain/web3/etc., is this supported?

No, and we're not interested in any "projects" using that technology. We ask that you do not harass our team members and contributors about it, nor bring it up on our Discord.

1.1.10 About Inox2D

Inox2D is the official Rust implementation of Inochi2D, meant to ease the usage of Inochi2D in Rust applications as well as webassembly workloads.

1.1.11 API Reference

1.1.12 INP Specification

INP Stands for **Inochi2D Puppet**, and is a binary container format to contain Inochi2D model, texture and extra data. INP is subject to change as we get closer to the 1.0 release.

Format Layout



Note:

Inochi2D stores values in Big Endian format, please make sure you handle this correctly!
You can find out more about endianness [here](#).

Length (bytes)	Contents	Notes
8	TRNSRTS\0	Magic bytes which tag the file as an INP file. (Trans Rights!)
4	JSON Payload Length	Length of JSON Payload
<i>Payload Length</i>	JSON Data	The Inochi2D model and rigging data
8	TEX_SECT	Texture Section Header
4	Texture Count	Amount of textures in the texture section
<i>Till Texture Blob End</i>	<i>Texture Blob</i>	Contains a texture and tag denoting what type the texture is.
8	EXT_SECT	OPTIONAL: Header for extended vendor data section
4	Payload Count	IF EXT_SECT EXISTS Amount of payloads that are in this section
<i>Till EXT Section End</i>	<i>EXT Section Blob</i>	IF EXT_SECT EXISTS The section blob of this EXT section.

Texture Blob

Every texture entry in the Texture Blob have the following encoding

Length (bytes)	Contents	Notes
4	Texture Payload Length	Length of the Texture Payload
1	<i>Texture Encoding</i>	A byte defining what texture encoding is in use. See Texture Encoding section.
<i>Payload Length</i>	Texture Data	Encoding of data depends on previous type

Extended Vendor Data Blob

Length (bytes)	Contents	Notes
4	Name Length	Length of EXT Payload name
<i>Name Length</i>	Name	The name of the EXT payload
4	Payload Length	Length of the EXT payload
<i>Payload Length</i>	Payload	Contents of payload, encoding is up to the individual developer.

Texture Encoding

There's 3 currently officially supported formats in Inochi2D, which are the following:

ID	Format
0	PNG - Portable Network Graphics (Lossless)
1	TGA - Truevision TGA (Lossless)
2	BC7 - BPTC Texture Compression (Lossy)

Profiles & Extensions

NOTE: Profiles and extensions are a *to be implemented* feature of the Inochi2D specification.

Most of this documentation is preliminary context.

The Inochi2D format is meant to be portable and usable across many platforms, but for some usescases you want more powerful features.

For example, maybe you're targeting modern PCs or smartphones and your project's artists want more flexibility in how they render their art.

Inochi2D provides Profiles and Extensions to cover these needs.

Inochi2D is split up in to 3 main profiles, each profile being a superset of the other.

1. Core
2. Standard
3. Extended

Additionally Inochi2D will be supporting named extensions to the format, so that implementors can chose to add, remove or modify features of the spec for their needs.

Every vendor which makes an extension has a unique prefix. The official Inochi2D Extension prefix is I2D.

Extensions will depend on a profile and Inochi2D version to be available, the extensions will in general specify which version of Inochi2D they target and what profile.

Tag	Vendor
I2D	Inochi2D Project

Profiles

Following are the profiles specced out for Inochi2D as of current.

This list may be subject to change.

The Core Profile

Core is meant for low power devices which may not have recent features, think PCs from 2012 and newer, or smartphones from 2013.

As such, Core is quite stripped down compared to what you see in Inochi Creator by default, and is to some extent rather limited.

Due to the nature of the core profile it's not recommended for most use cases, only use this if you're targetting really old hardware.

	Minimum	Recommended
GL	2.1	2.2
DirectX	9	10
Vulkan	1.0	1.0
Metal	Latest	Latest

The Standard Profile

The standard profile is the profile most people should be targeting.

It's aimed at modern PCs, phones and games consoles, although some features may need to be approximated on some platforms, it's the best in terms of feature-to-stability ratio.

	Minimum	Recommended
GL	3.1	3.2
DirectX	10	12
Vulkan	1.0	1.3
Metal	Latest	Latest

The Extended Profile

The extended profile contains features that have a potential of making it in to Standard. These features can be very experimental in nature and is, in general, not recommended for production use. Generally things making it to Extended should start out as extensions instead.

	Minimum	Recommended
GL	3.1	4.5
DirectX	10	12
Vulkan	1.0	1.3
Metal	Latest	Latest

Extensions

Extensions recognized by the Inochi2D Project are listed here.

If more documentation is available about the extension, said documentation can be seen by clicking the name of the extension.

Name	Version	Profile
I2D_BLEND_ACCURACY ¹	0.8	Standard
I2D_EXTENDED_BLEND_MODES ¹	0.8	Standard

1.1.13 Payload Specification

This section contains information about the JSON based payload of the Inochi2D format.

¹ This extension was introduced before extensions were added to the spec.
As such, older Inochi2D implementations will more or less just assume it's there.

Blending

Inochi2D allows for porter-duff blending modes to be applied to Part nodes and Composite nodes, these blending nodes affect how colors are mixed on compositing with the screen.

Additionally the special blending modes “Clip to Lower” and “Slice from Lower” are supported to provide a nicer looking but less flexible form of masking.

Rendering equations

Mode	Equation	Src	Dst	Src (Alpha)	Dst (Alpha)
Normal	Add	1	1-SrcAlpha		
Multiply	Add	DstColor	1-SrcAlpha		
Screen	Add	1	1-SrcColor		
Lighten	Max	DstColor	1		
Color Dodge	Add	DstColor	1		
Linear Dodge	Add	1	1-SrcColor	1	1-SrcAlpha
Add (Glow)	Add	1	1	1	1-SrcAlpha
Subtract	ReverseSubtract	1-DstColor	1		
Exclusion	Add	1-DstColor	1-SrcColor	1	1
Inverse	Add	1-DstColor	1-SrcAlpha		
Destination In	Add	0	SrcAlpha		
Clip to Lower	Add	DstAlpha	1-SrcAlpha		
Slice from Lower	Add	0	1-SrcAlpha		

Blend Accuracy Extension

Inochi2D provides the I2D_BLEND_ACCURACY *extension*, this extension is entirely implementation side.

If the Inochi2D implementation indicates support for this extension, it means that instead of emulating porter and duff blending modes, it will use GPU extensions to implement them properly.

This extension affects the rendering of the following blending modes.

- Normal
- Multiply
- Screen
- Lighten
- Color Dodge
- Subtract

- Inverse

This extension additionally affects the rendering of the following blending modes if `I2D_EXTENDED_BLEND_MODES` is supported.

- Overlay
- Darken
- Color Burn
- Hard Light
- Soft Light
- Difference
- Exclusion

These blending modes use legacy blending if you have more than a Albedo output on some platforms.

Implementation Notes

- This extension can be implemented in OpenGL via the `GL_KHR_blend_equation_advanced` and `GL_KHR_blend_equation_advanced_coherent` extensions.
- This extension can be implemented in Vulkan via the `VK_EXT_blend_operation_advanced` extension.
- This extension can be implemented on tiler GPUs, such as PowerVR and Apple Silicon chips, via framebuffer fetch.

This extension was introduced in Inochi2D 0.8 Standard.

Extended Blend Modes

Inochi2D provides the `I2D_EXTENDED_BLEND_MODES` *extension*.

This extension adds support for the following blending modes.

- Overlay
- Darken
- Color Burn
- Hard Light
- Soft Light
- Difference
- Exclusion

These blending modes are disabled if you have more than Albedo output.

Implementation Notes

- This extension can be emulated by shaders using Texture Barriers.
- This extension can be emulated by shaders using a ping-pong framebuffer setup, this is unoptimal.

This extension was introduced in Inochi2D 0.8 Standard.

This extension is influenced by I2D_BLEND_ACCURACY.