Quick Start Guide

Online Documentation

This PDF document is provided just in case it's impossible to reach the online resources; it only covers the basics of using the engine and may be out of date. For the extensive and up-to-date materials, please visit naninovel.com.

Introduction

Naninovel is an extension for Unity game engine. It consists of a C# framework and editor utilities to support development of visual novel games.

Prerequisites

Naninovel is an extension for Unity game engine, so it's strongly recommended to at least learn the basics of using the engine before starting with Naninovel. The following manual chapters should be considered essential:

- Installing Unity
- Creating Projects
- Editor Interface
- Using the Asset Store
- Publishing Builds

In case you're not going to build any custom gameplay outside of Naninovel, feel free to ignore the scene-related information altogether, as Naninovel will take care of that.

Create New Unity Project

Consult the Unity manual on how to create a new project.

When creating a project you'd probably like to use 2D Template to set the editor in 2D behaviour mode, so that images will be imported as sprite assets by default and you won't have to manually change the import settings. You can change the editor behaviour mode later using the project settings.

Install Naninovel

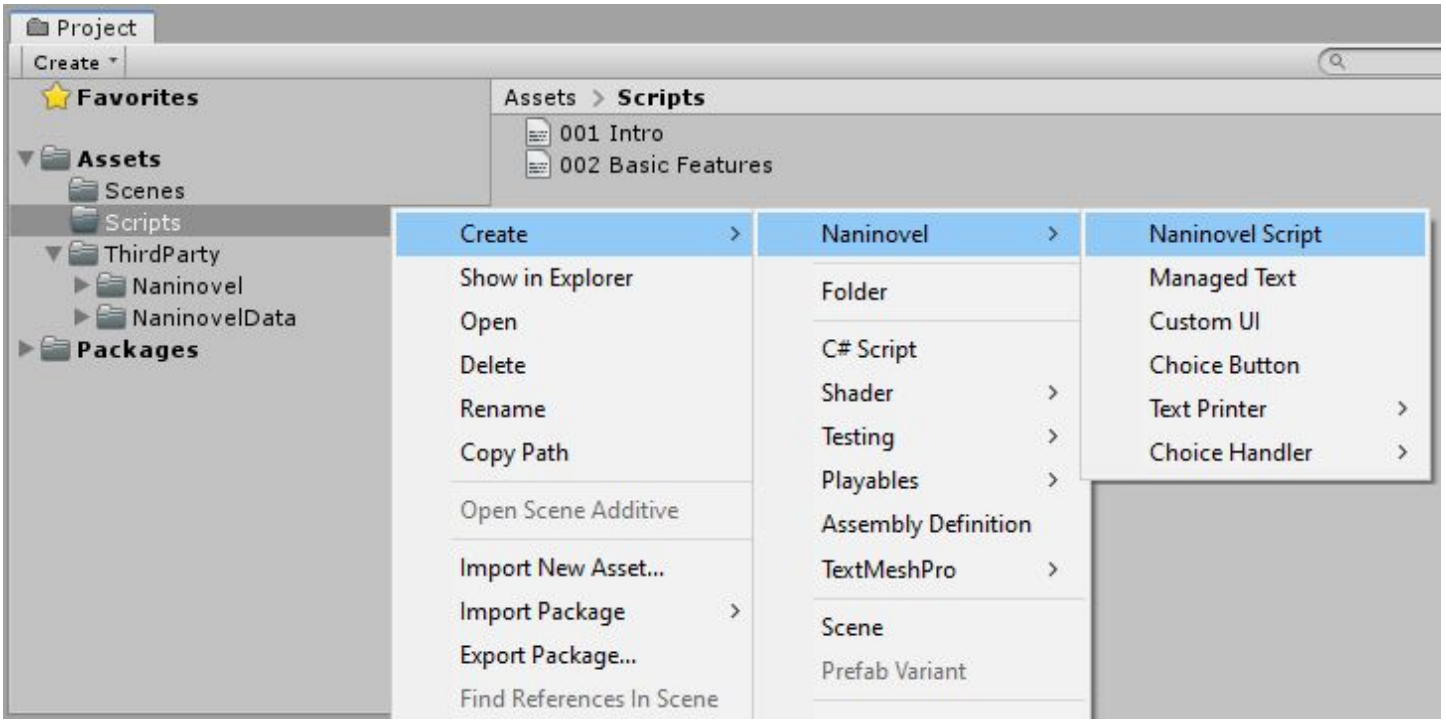
Import Naninovel package using the Asset Store window and wait for the initial scripts compilation and assets import process.

You are free to move Naninovel package folder anywhere inside your project assets directory, if you wish. Over the course of using Naninovel a number of assets (configuration, settings, saves, etc) will be automatically generated inside Assets/NaninovelData folder.

In contrast to the package folder, you shouldn't manually move the data folder (it'll be automatically regenerated). If you wish to change the location of the data folder, edit Generated Data Path property in the engine configuration menu.

Add Naninovel Script

Use `Create → Naninovel → Naninovel Script` assets context menu to create a naninovel script asset.



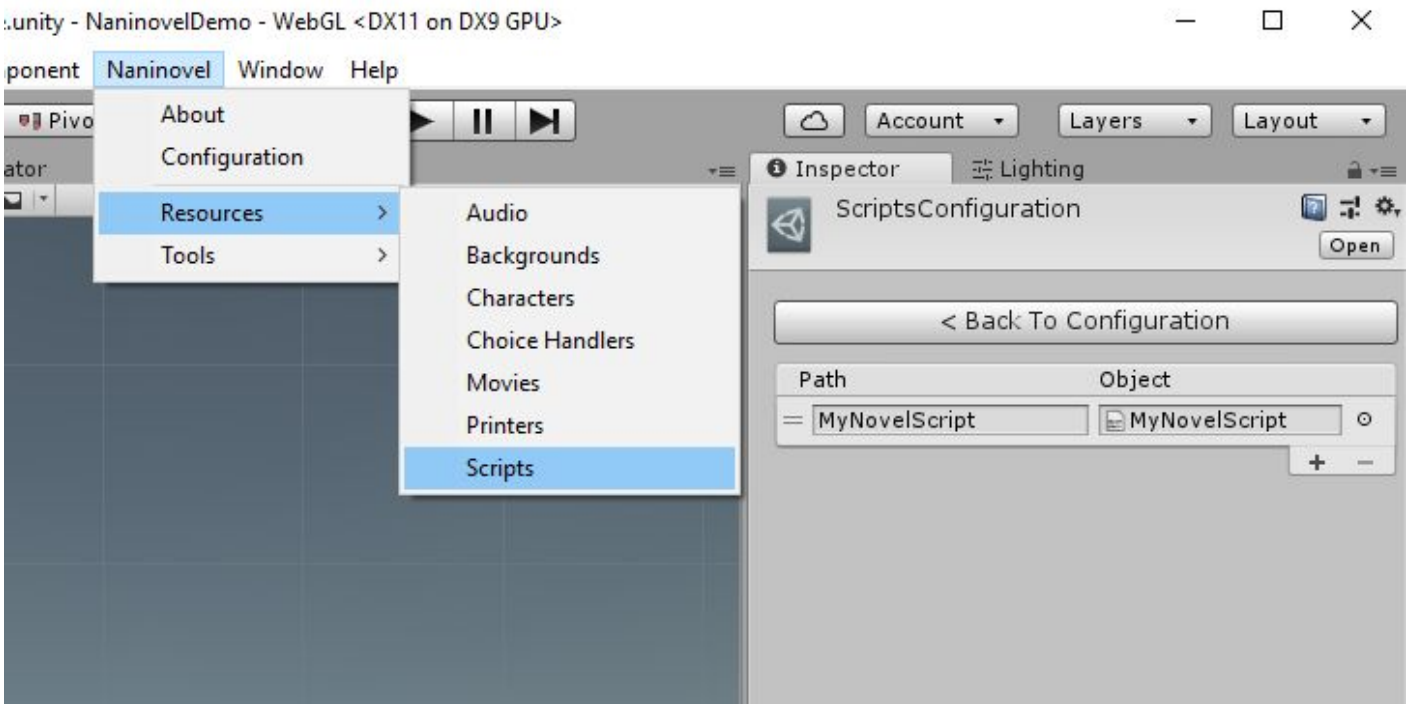
Notice: you can create and store naninovel scripts (as well as all the other Naninovel resources) under any project folder and organize them in any way you like; the naming is also up to you. The above illustration is just an example.

Naninovel scripts are text documents (.nani extension) where you control what happens on scenes. You can open and edit them with a text editor of your choice, like Notepad, TextEdit or Atom.

Script001.nani | Script002.nani

```
>n Nani-kun
>b Bubble
>k Kohaku

; Default Nanikun position
>pos pos:50,45,1

# Start

@bgm Wind volume:0.5
@back Desert.SlideIn
@char {n}.Happy pos:50,-10

n: Imagine you've imported Naninovel package to your Unity project.[i]
[char {n} pos:80 wait:false] For starters, create a <b>novel script</b>.
Novel scripts are text documents where you control what happens on
scenes.[i] Open them with a text editor of your choice, like Notepad, Word
or Sublime.
```

In order for a Naninovel-related asset (like our created script) to become "visible" for the engine, it should be assigned as a project resource. When creating the scripts via the create assets menu, they're assigned automatically. To assign (or edit/remove) a script resource manually use script resources window accessible with `Naninovel → Resources → Scripts` editor context menu. To add a script, press + (plus sign) button in the list to add a new record and drag-drop script asset to the list. It's also possible to drag-drop multiple assets or even whole folders to the list to add them in batch.



Open the created script in a text editor and add the following text:

```
Hello World!
@stop
```

The first line will print the text "Hello World!" when the game is run and the second is required to gracefully stop script execution.

Enter play mode and start a new game to see the result.

It's also possible to store the naninovel scripts at a Resources/Scripts folder, which will make them all "visible" to the engine by default. This is not recommended though, as some of the editor features (like script selection dropdowns) won't work.

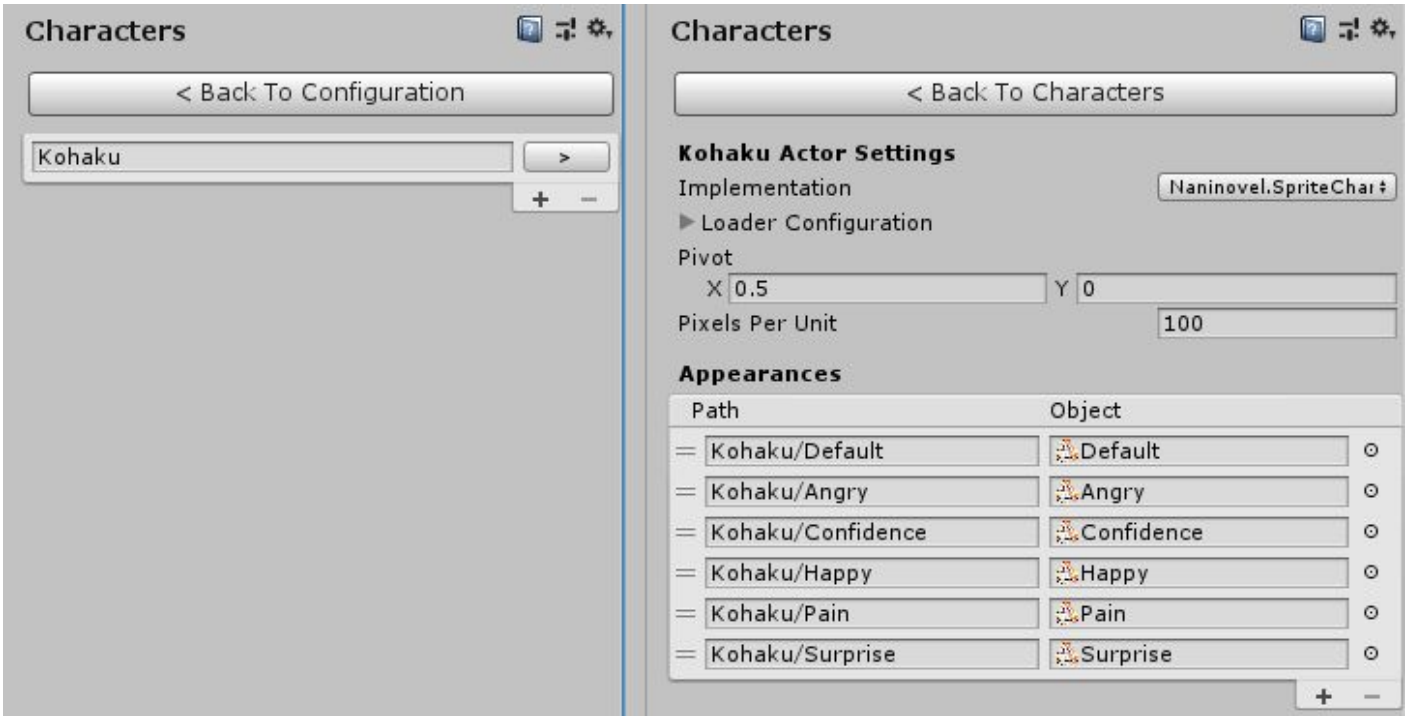
In case "NEW GAME" button of the title menu is not active, make sure "Start Game Script" property in the script configuration (Naninovel → Configuration → Scripts) is equal to the name of the created script. The property is populated automatically when creating the first script via create asset menu, but this may not work if you copy some existing script to the project.

Add Character

Characters in Naninovel can be based on regular and diced sprites, animated Live2D models and 3D meshes; you can add your own implementations as well. For the purpose of this tutorial, we'll use a sprite implementation.

Each character is represented by ID and a set of appearances. To add a sprite character, you can either use the character manager GUI (recommended) or place the character's appearance sprites in a `Resources/Characters/CharacterName` folder, where `CharacterName` is the name of the character.

If you choose to use the manager GUI, access it with `Naninovel → Resources → Characters` menu, add new character record specifying its ID, then double click the ID record (or press button at the and of the record) and add all the appearance sprites to the Resources list. Just like with naninovel scripts, you can drag-drop multiple assets and folders to the list.



Let's assume the added character ID is "Kohaku". Edit naninovel script to show the added character:

```
@char Kohaku
Hello World!
@stop
```

Run the game and you'll see one of the character appearance sprites at the center of the screen. When you don't specify an appearance, either the one named equal to character's ID or "Default" will be chosen by default. To select a specific appearance, add its name after the character ID separated by a dot like this:

```
@char Kohaku.Happy
Hello World!
@stop
```

Given there is an appearance with the name "Happy" added for the character "Kohaku", the corresponding sprite will now be shown instead of the default one.

You can now also associate the printed text with the character by adding its ID followed by a colon before the text:

```
@char Kohaku.Happy
Kohaku: Hello World!
@stop
```

To hide (remove from scene) a character (or any other actor, like background, text printer, etc), use @hide command followed by actor ID:

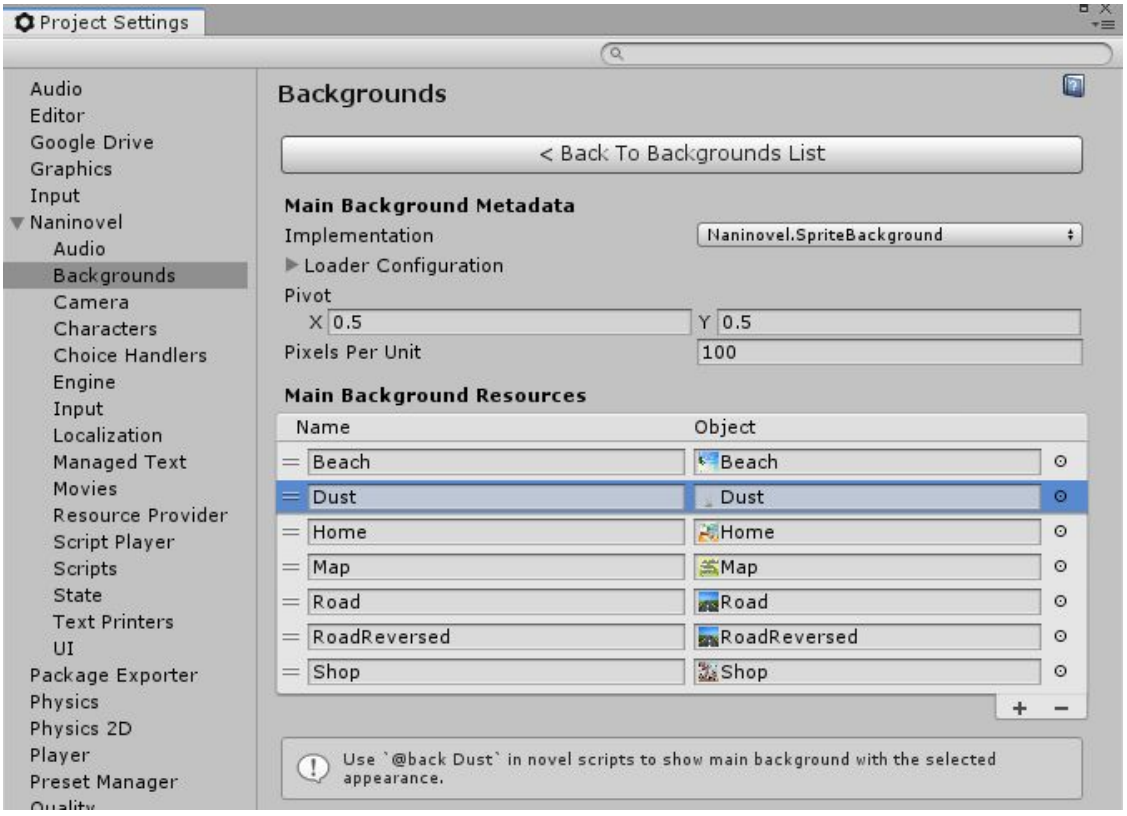
```
@char Kohaku.Happy
Kohaku: Hello World!
@hide Kohaku
@stop
```

Add Background

Similar to characters, a background can be represented in multiple ways in Naninovel: sprite, generic object, video and scene; custom user implementations are also possible.

While you can create multiple independent background actors, in a typical VN game you'll usually use just one and transition it to different appearances. To simplify the routine, a MainBackground actor is added to the background actors list by default and you don't have to specify the ID every time to change its appearance in naninovel scripts.

Add sprite background in the same way you've added a character: using editor GUI `Naninovel → Resources → Backgrounds` (MainBackground actor will be chosen automatically) or place the appearance sprites under a `Resources/Backgrounds/MainBackground` folder.



Let's assume the added background appearance sprite is named "City". To show a background, use a @back command followed by the background appearance name:

```
@back City
```

When switching between backgrounds a cross-fade transition effect will be used by default. To change the effect, specify transition type after the appearance name:

```
@back City
@back School.RadialBlur
```

This will transition "City" to "School" using "RadialBlur" transition effect.

Add Music and Sound Effects

To add a BGM (background music) or SFX (sound effect) asset, either use Naninovel → Resources → Audio editor menu (recommended) or place the audio assets inside Resources/Audio folder. You can use any audio formats supported by Unity.

Let's assume the added BGM file name is "ThePromenade". To play this track as a background music use @bgm command followed by the name of the track:

```
@bgm ThePromenade
```

A cross-fade effect will be automatically applied when switching the music tracks. The music will loop by default, though you can change this, as well as volume and fade duration using command parameters.

On the contrary, sound effects won't loop by default. Assuming you've added an "Explosion" SFX, use an @sfx command to play it back:

```
@sfx Explosion
```

Support

If you have issues with the engine, make sure you're using the latest available version. You can update the main package inside the Unity editor using Asset Store window. In case you're using any of the engine extensions, download the latest packages using the following links:

- NaninovelLive2D
- NaninovelPlayMaker

If updating didn't help, try to re-install the package by deleting Naninovel folder from the project and re-importing it from the Asset Store.

Always **remember to backup your project** (or use a VCS) before updating or deleting a package.

Issue Tracker

In case the above steps didn't help to resolve the issue, check the issue tracker — chances are the problem is already being worked on; and if it's not, let us know by creating a bug report.

Developer Support

To receive direct developer support join Naninovel Discord server (discord.gg/avhRzP3) and register your copy of the asset using the following registration form: naninovel.com/register.

After the registration you'll automatically get a "Verified User" role on the server and access to the #support channel. You can also message @Elringus#6359 to discuss the issue in private.