# ink-Unity integration

This Unity package allows you to integrate inkle's ink narrative scripting language with Unity and provides tools to **compile**, **play** and **debug** your stories.

# Overview

- **Using ink in your game**: Allows running and controlling ink files in Unity via the C# runtime API.
- **ink player**: Provides a powerful Ink Player Window for playing and debugging stories.
- **Auto compilation**: Instantly creates and updates a JSON story file when a `.ink` is updated.
- **Inspector tools**: Provides an icon for ink files, and a custom inspector that provides information about a file.

# Getting started

## :inbox_tray: Installation

There are 4 different ways to install this plugin:

### :star:As a .UnityPackage:star:

This will import the source into your Assets folder. This is a good option if you intend to edit the source for your own needs. * Download the latest .UnityPackage. * Open the downloaded file to import it into your Unity project.

### As a UPM Package

Installing via a package allows you to easily update via Unity's Package Manager window. This is best if you don't need to edit the source. * When installed via UPM, demo projects can be imported from Packages > Ink Unity Integration > Demos

#### Via Package Manager

- Add the following line to PROJECT ROOT/Packages/manifest.json: `"com.inkle.ink-unity-integration": "https://github.com/inkle/ink-unity-integration.git#upm"`

#### OpenUPM

- Navigate to OpenUPM and follow their instructions
- The project will have installed at Packages > Ink Unity Integration.

### From GitHub

- You can clone/download/fork the project on GitHub.
- The easiest way to download it is to click the green Code button and select Download ZIP
- Install by moving the folder Packages/Ink to anywhere in your Unity project's Assets folder

### Via the Asset Store

onvinience a .UnityPackage is hosted at the Unity Asset Store. **This version is updated rarely, and so is not recommended.** This will import the source into your Assets folder. This is a good option if you intend to edit the source for your own needs.

## :video_game: Demos

This project includes a demo scene, providing a simple example of how to control an ink story with C# code using Unity UI.

(If you imported this package as a UPM, then you must first import the demos from Packages > Ink Unity Integration > Demos)

To run a demo, double-click the scene file at the root of the demo folder to open it, and press the Play button at the top of the screen to start it.

## :page_facing_up: C# API

The C# API provides all you need to control ink stories in code; advancing your story, making choices, diverting to knots, saving and loading, and much more.

It is documented in the main ink repo.

For convenience, the package also creates an (**Help > Ink > API Documentation**) menu option.

## :pencil2: Writing ink

For more information on writing with **ink**, see the documentation in the main ink repo.

For convenience, the package also creates an (**Help > Ink > Writing Tutorial**) menu option.

## :question: Further Help

For assistance with writing or code, Inkle's Discord forum is full of lovely people who can help you out!

To keep up to date with the latest news about ink sign up for the mailing list.

# Features

## Compilation

Ink files must be compiled to JSON before they can be used in-game. **This package compiles all edited ink files automatically.** By default, compiled files are created next to their ink file.

### Editor Compilation

This package provides tools to automate this process when a .ink file is edited.

**Disabling auto-compilation**: You might want to have manual control over ink compilation. If this is the case, you can disable "Compile ink automatically" in the InkSettings file or delete the InkPostProcessor class.

**Manual compilation**: If you have disabled auto-compilation, you can manually compile all ink files using the **Assets > Recompile Ink** menu item, individually via the inspector of an ink file, or via code using InkCompiler.CompileInk().

**Play mode delay**: By default, ink does not compile while in play mode. This can be disabled in the InkSettings file.

### In-game Compilation

The compiler is included in builds (See WebGL best practices for information on removing it), enabling you to allow the editing of ink files as part of your game.

## Ink Player Window

The Ink Player Window (**Window > Ink Player**) allows you to play stories in an editor window, and provides functionality to edit variables on the fly, test functions, profile performance, save and load states, and divert.

To play a story, click the "play" button shown on the inspector of a compiled ink file, or drag a compiled ink story TextAsset into

the window.

**Editor Attaching**: Attaching the InkStory instance used by your game to the Ink Player window allows you to view and edit your story as it runs in game.

See BasicInkExampleEditor.cs in the Examples folder for an example of how to: * Show an attach/detach button on an inspector * Automatically attach on entering play mode

More information on using and extending Ink Player Window

## Inspector tools

This package replaces the icon for ink files to make them easier to spot, and adds a custom inspector for a selected ink file.

**The Inspector**: Selecting an ink file displays its last compile time; lists any include files; and shows any errors, warnings or todos. It also shows a Play button which runs the story in the Ink Player Window.

# Visual Scripting Support

## Bolt

There is currently no support for Bolt, Unity's official visual scripting tool. If you're interested in building one, we'd love to see it!

## PlayMaker

There's unofficial support for PlayMaker here.

We'd love to see this supported more if you'd like to assist the effort!

# Source control tips

When you edit ink files, the compiler will also update the corresponding compiled .json file. If no compiled file existed before, Unity will also create a meta file for it. It is recommended that you always commit both ink and json files at the same time to avoid the file being re-compiled by your team members.

Adding or removing ink files will also make changes to the InkLibrary file, and we could recommend authors also commit this file for the same reasons.

# WebGL best practices

WebGL builds should be as small as possible. The ink compiler is included in builds, but is typically only used in the editor. If your game doesn't require compiling ink at runtime we recommend adding a .asmdef at Ink Unity Integration > InkLibs > InkCompiler that only functions in the editor.

# FAQ

- Is the Linux Unity Editor supported?

  *Yes!*

- What versions of Unity are supported?

  We support 2020 LTS and above. Until version 1.1.1 we supported 2018 LTS, which should also work going back to at least Unity 5.

# Support us!:heart:

Ink is free, forever; but we'd really appreciate your support! If you're able to give back, generous donations at our Patreon mean the world to us.

# License