User Guide Steamworks DRM

"Using Steamworks for downloadable content DRM with Ultimate DLC Toolkit"

Trivial Interactive

Version 1.2.x

This short guide will cover the necessary setup for creating, managing and fetching downloadable content intended for a Steam game targeting Windows, Mac or Linux platforms.

Plugins

Steamwork DRM requires one of the following plugins to be installed in your Unity project for communication with Steam services.

☑ Supports the popular **Steamworks.Net** plugin for Unity.
☑ Supports the **Facepunch.Steamworks** plugin for Unity.
☑ Support for other Steam API plugins can be added manually if required.

Features

- Listing remote DLC – DLC hosted on Steam can be fetched at any time, even if the base game does not know about it.
- Check DLC ownership – Use Steam user account to verify ownership of DLC before it is ever installed or loaded.
- Retrieving DLC data stream – Get the data stream of installed DLC for loading purposes, without needing to know the install location ahead of time (Assumes that the DLC will be installed under the same root directory as the base game – can be in sub folders of any depth).
- On demand installation – The app can request installation of owned DLC when convenient via the DRM API.
- On demand uninstallation – The app can request that installed DLC be removed from the user's device (not recommended for most cases).
- Track DLC Usage – The app can inform Steam services when a certain DLC is being used by the game (Loaded/unloaded) so that play time and other statistics can be measured by Steam services. This is fully automatic and does not require and extra work to support.

Project Setup

The following section will cover the necessary setup steps required for the supported Steamworks plugins. You can skip to the section that applies to you and continue reading about Steam DLC once you have completed the setup guide for your chosen plugin.

Skip to applicable plugin:

Steamworks.Net, Facepunch.Steamworks, Custom/Other Steam plugin

Or

Continue reading about Steam DLC

Steamworks.Net

This section is specific to the `Steamworks.Net` plugin for Unity, and if you are using a different plugin you can skip to the appropriate section.

Ultimate DLC Toolkit has fully featured built-in support for Steam using the `Steamworks.Net` API. This means that once setup, Ultimate DLC Toolkit will handle all the Steam API calls and requests automatically when you make a request via the Unified `DLC` API.

First you will need to setup and enable `Steamworks.Net` support in your project by following these steps:

1. **Install Steamworks.Net:** First you will need to install the plugin if you have not already by following the instructions on the github page, or via the Steamworks.Net docs.

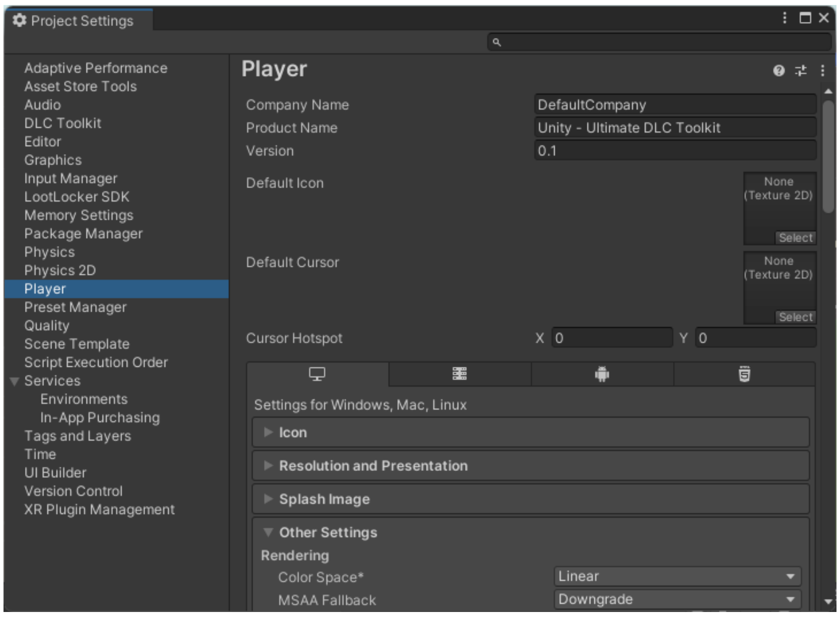
2. **Initialize Steam:** Next you should ensure that Steam is initialized before making any calls to the unified DLC API, otherwise the requests will fail with an error message that Steam is not running. To initialize Steam via the `Steamworks.Net` API is simply a case of adding the `SteamManager` script to a game object in one of your first run scenes. It is recommended that you check `SteamManager.Initialized` before making any calls via the DLC unified API.

3. **Enable Steam DRM:** Finally, you can enable the `Steamworks.Net` built-in DRM support in Ultimate DLC Toolkit by adding a define symbol in the Unity player settings:

    a. In the Unity editor, select the following menu item to open the project settings for your game:

    `Edit -> Project Settings`

    b. In the project settings window, select the `Player` section on the left navigation panel:



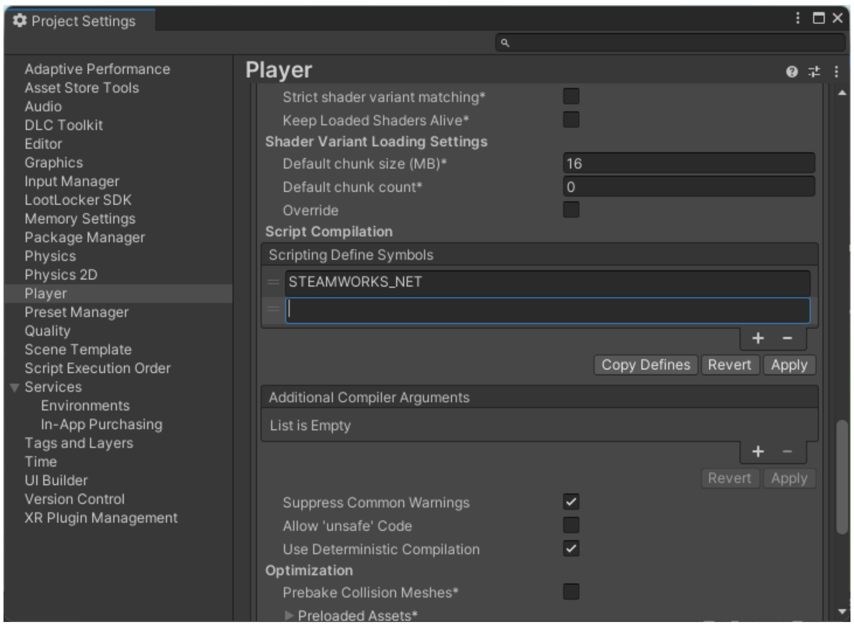
Player

Company Name: DefaultCompany
Product Name: Unity - Ultimate DLC Toolkit
Version: 0.1
Default Icon: None (Texture 2D)
Default Cursor: None (Texture 2D)
Cursor Hotspot: X 0 Y 0

Settings for Windows, Mac, Linux

Rendering
Color Space*: Linear
MSAA Fallback: Downgrade

c. Scroll down and find the list view named `Scripting Define Symbols`. Hit the `+` button to add a new entry:

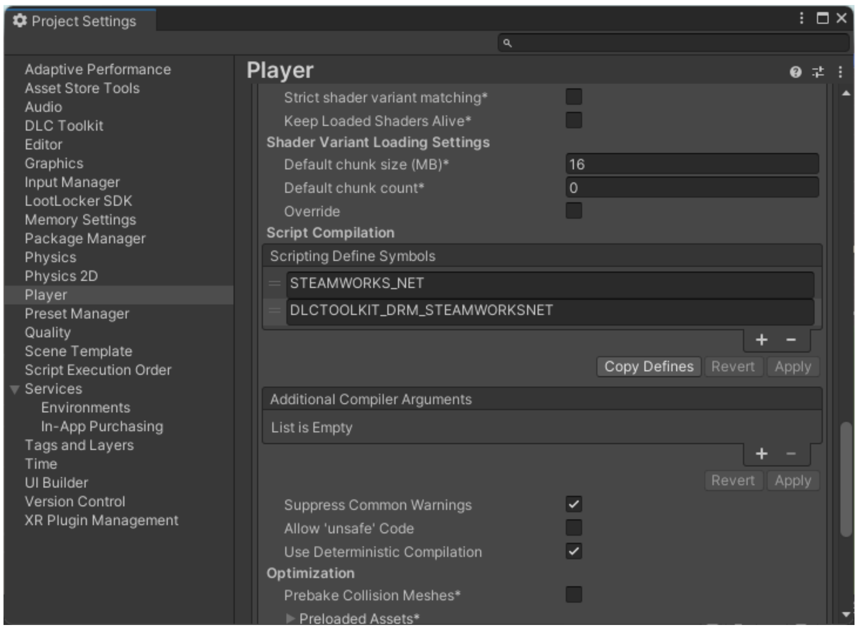


Player

Strict shader variant matching*	☐
Keep Loaded Shaders Alive*	☐
Shader Variant Loading Settings	
Default chunk size (MB)*	16
Default chunk count*	0
Override	☐
Script Compilation	
Scripting Define Symbols	STEAMWORKS_NET
Additional Compiler Arguments	List is Empty
Suppress Common Warnings	✓
Allow 'unsafe' Code	☐
Use Deterministic Compilation	✓
Optimization	
Prebake Collision Meshes*	☐

d. Input the following define symbol and click the `Apply` button once finished:

```
DLCTOOLKIT_DRM_STEAMWORKSNET
```



Player

Strict shader variant matching*	☐
Keep Loaded Shaders Alive*	☐
Shader Variant Loading Settings	
Default chunk size (MB)*	16
Default chunk count*	0
Override	☐
Script Compilation	
Scripting Define Symbols	STEAMWORKS_NET DLCTOOLKIT_DRM_STEAMWORKSNET
Additional Compiler Arguments	List is Empty
Suppress Common Warnings	✓
Allow 'unsafe' Code	☐
Use Deterministic Compilation	✓
Optimization	
Prebake Collision Meshes*	☐

e. Allow Unity to recompile scripts and refresh. `Steamworks.Net` is now setup and integrated into your Ultimate DLC Toolkit game project!

Facepunch.Steamworks

This section is specific to the `Facepunch.Steamworks` plugin for Unity, and if you are using a different plugin you can skip to the appropriate section.

Ultimate DLC Toolkit has fully featured built-in support for Steam using the ` Facepunch.Steamworks` API. This means that once setup, Ultimate DLC Toolkit will handle all the Steam API calls and requests automatically when you make a request via the Unified `DLC` API.

First you will need to setup and enable ` Facepunch.Steamworks` support in your project by following these steps:

1. **Install Facepunch.Steamworks:** First you will need to install the plugin if you have not already by following the instructions on the Installing For Unity docs page.

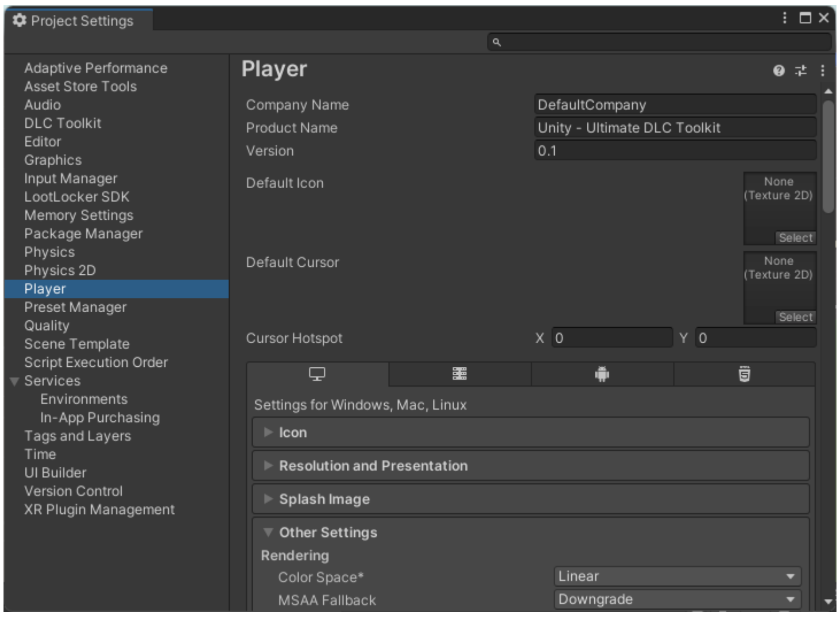
2. **Initialize Steam:** Next you should ensure that Steam is initialized before making any calls to the unified DLC API, otherwise the requests will fail with an error message that Steam is not running. To initialize Steam via the `Facepunch.Steamworks` API, you will need to call `Steamworks.SteamClient.Init` with your Steam AppId from one of your game scripts as shown in the setting up docs, and then you will also need to run Steam callbacks, also shown in the setting up docs. It is recommended that you check Steamworks.SteamClient.IsValid` before making any calls via the DLC unified API.

3. **Enable Steam DRM:** Finally, you can enable the `Facepunch.Steamworks` built-in DRM support in Ultimate DLC Toolkit by adding a define symbol in the Unity player settings:

    a. In the Unity editor, select the following menu item to open the project settings for your game:

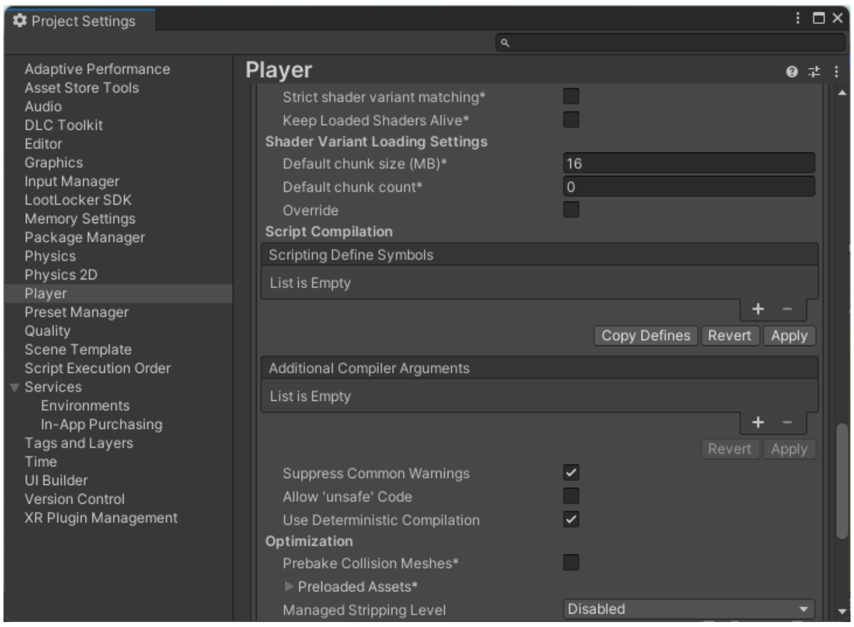
    `Edit -> Project Settings`

    b. In the project settings window, select the `Player` section on the left navigation panel:



Company Name: DefaultCompany
Product Name: Unity - Ultimate DLC Toolkit
Version: 0.1
Default Icon: None (Texture 2D)
Default Cursor: None (Texture 2D)
Cursor Hotspot: X 0 Y 0
Settings for Windows, Mac, Linux
Color Space*: Linear
MSAA Fallback: Downgrade

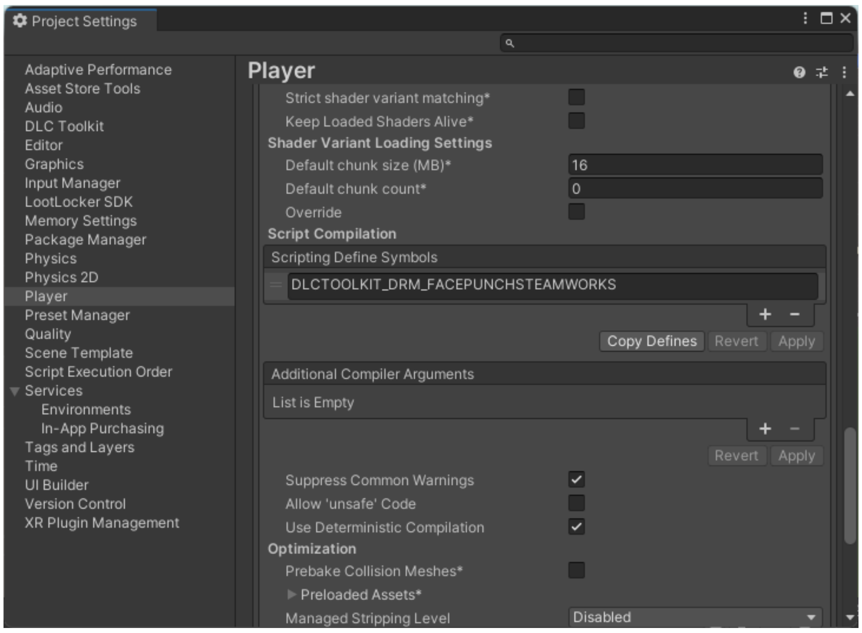
c. Scroll down and find the list view named `Scripting Define Symbols`. Hit the `+` button to add a new entry:



Strict shader variant matching*: ☐
Keep Loaded Shaders Alive*: ☐
Default chunk size (MB)*: 16
Default chunk count*: 0
Override: ☐
Scripting Define Symbols: List is Empty
Additional Compiler Arguments: List is Empty
Suppress Common Warnings: ✓
Allow 'unsafe' Code: ☐
Use Deterministic Compilation: ✓
Prebake Collision Meshes*: ☐
Managed Stripping Level: Disabled

d. Input the following define symbol and click the `Apply` button once finished:

```
DLCTOOLKIT_DRM_FACEPUNCHSTEAMWORKS
```


Project Settings

Adaptive Performance
Asset Store Tools
Audio
DLC Toolkit
Editor
Graphics
Input Manager
LootLocker SDK
Memory Settings
Package Manager
Physics
Physics 2D
Player
Preset Manager
Quality
Scene Template
Script Execution Order
Services
Environments
In-App Purchasing
Tags and Layers
Time
UI Builder
Version Control
XR Plugin Management

Player

Strict shader variant matching*	☐
Keep Loaded Shaders Alive*	☐
Shader Variant Loading Settings	
Default chunk size (MB)*	16
Default chunk count*	0
Override	☐

Script Compilation

Scripting Define Symbols
DLCTOOLKIT_DRM_FACEPUNCHSTEAMWORKS

Copy Defines | Revert | Apply

Additional Compiler Arguments
List is Empty

Revert | Apply

Suppress Common Warnings	✓
Allow 'unsafe' Code	☐
Use Deterministic Compilation	✓
Optimization	
Prebake Collision Meshes*	☐
Preloaded Assets*	
Managed Stripping Level	Disabled

e. Allow Unity to recompile scripts and refresh. `Facepunch.Steamworks` is now setup and integrated into your Ultimate DLC Toolkit game project!

Custom Steam API

This section is specific to the using a custom or unsupported Steam plugin for Unity, and if you are using a different plugin you can skip to the appropriate section.

If you are using a different or custom API for integrating with Steam services, then you will need to create a custom DRM provider so that you can integrate with Ultimate DLC Toolkit.

Creating a custom DRM provider requires implementing a C# interface, and then registering it for use by the unified DLC API. You can find a complete code sample here, and it is just a case of using your intended alternate Steam API in place of the example API:

Custom DRM Provider Code Sample

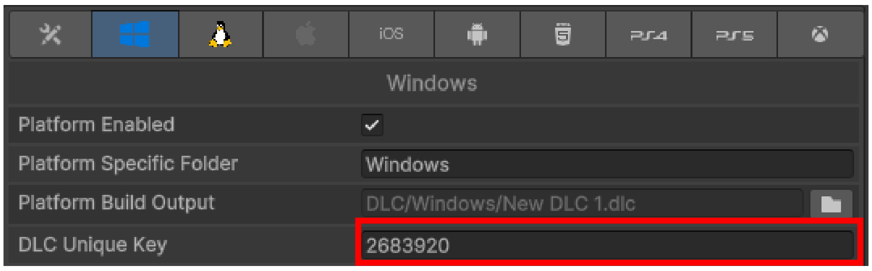
Creating Steam DLC

The information listed from this section onwards applies to Steam DLC regardless of the integration plugin used.

Creating DLC content for Steam has one minor difference from the standard workflow outlined in the main user guide. That difference being that the DLC unique key **MUST** be set to the Steam AppID for the associated DLC created via the Steam partner dashboard.

To do this you must first create a DLC app for you game in the Steam partner dashboard, which will generate a unique app id for that DLC. Creating the DLC listing via the Steam dashboard is outside the scope of this document, but you can check the Steam docs for DLC which contains a helpful video for creating new DLC for your game, and once completed you will have generated a unique numeric value (usually more than 6 digits) which is your Steam AppID for the DLC.

Once you have the DLC AppID, it is simply a case of selecting the associated DLC profile asset in your Unity project (See the main user guide for more info) and assigning the AppID as the DLC unique key for the platforms you intend to target (Windows, Mac and Linux are supported on Steam).



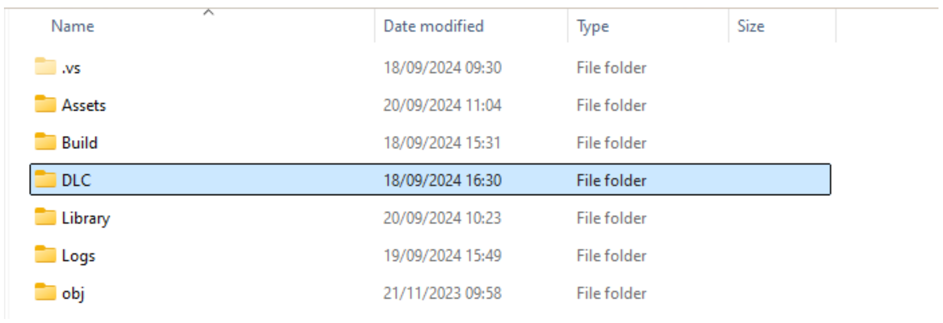
Platform Enabled ✓
Platform Specific Folder: Windows
Platform Build Output: DLC/Windows/New DLC 1.dlc
DLC Unique Key: 2683920

With that setup, you can now create your DLC assets and content using the normal workflow as outlined in the main user guide.

Building Steam DLC

Building DLC content for Steam does not require any additional steps and is simply a case of using the Ultimate DLC Toolkit build menu, or the inspector window of your selected DLC Profile to build the content.

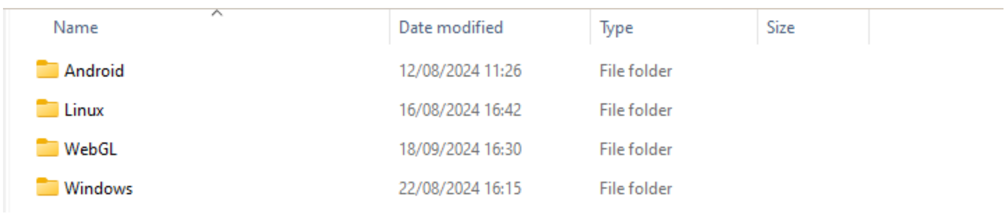
The only consideration is the build output location of the DLC in order to make it easy to setup in SteamPipe for uploading. It is recommended that any DLC with the `Ship With Game` option disabled should be built in a separate location as the game:



Name	Date modified	Type	Size
.vs	18/09/2024 09:30	File folder	
Assets	20/09/2024 11:04	File folder	
Build	18/09/2024 15:31	File folder	
DLC	18/09/2024 16:30	File folder	
Library	20/09/2024 10:23	File folder	
Logs	19/09/2024 15:49	File folder	
obj	21/11/2023 09:58	File folder	

For example, in the above image you can see that the built DLC is placed in the `DLC` folder, and the game standalone is placed in the `Build` folder, which will make it easier to manage later.

Of course, inside that folder you can also build the DLC into a platform specific folder if you are building for multiple platforms:

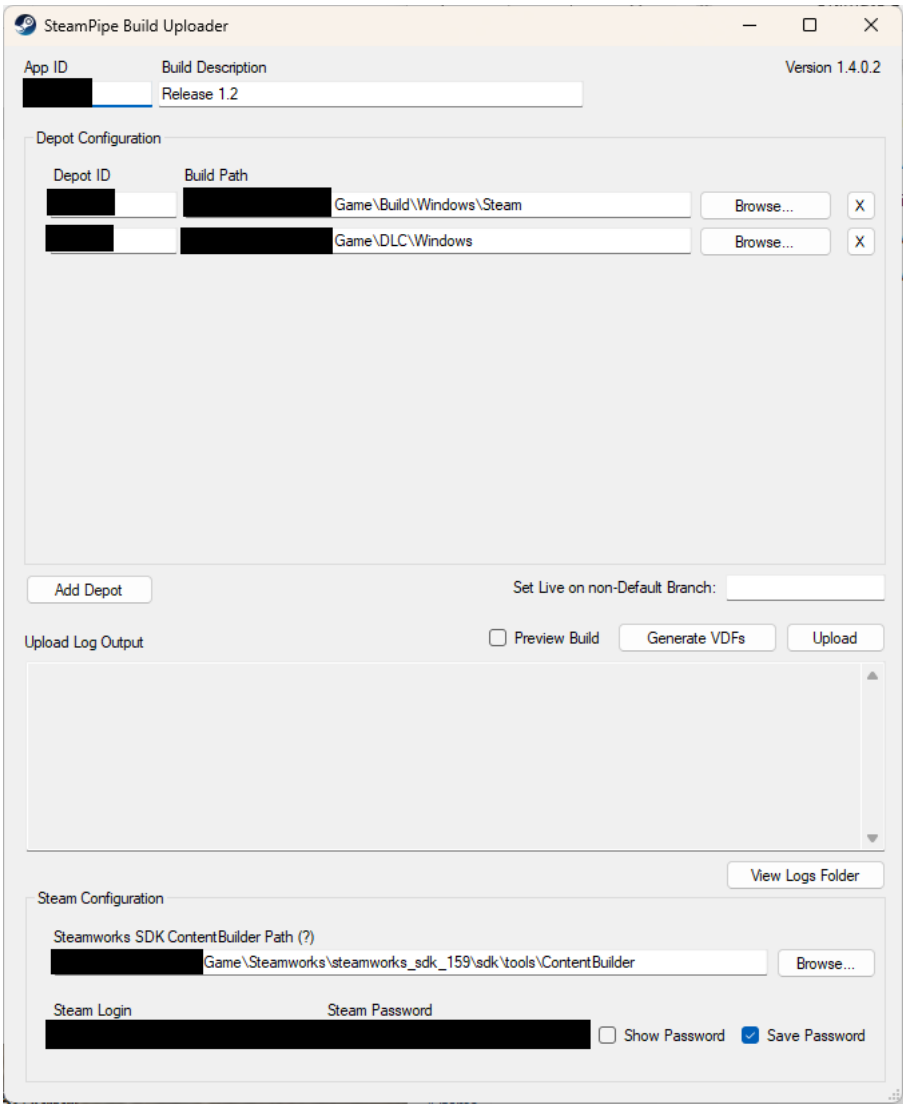


Name	Date modified	Type	Size
Android	12/08/2024 11:26	File folder	
Linux	16/08/2024 16:42	File folder	
WebGL	18/09/2024 16:30	File folder	
Windows	22/08/2024 16:15	File folder	

Uploading DLC to Steam

Once you have created your DLC and built it, you can then test it in the editor using the Ultimate DLC Toolkit test mode (More info in the main user guide), or you can upload to Steam so that you can test how the DLC would work as a standalone.

The whole process of uploading to Steam is outside the scope of this document and there is detailed documentation available form Steam, but as a brief example using the SteamPipe GUI tool, you will want to add any DLC that you have built so that it can be uploaded:


SteamPipe Build Uploader

App ID: [redacted]
Build Description: Release 1.2
Version 1.4.0.2

Depot Configuration

Depot ID	Build Path
[redacted]	[redacted]Game\Build\Windows\Steam
[redacted]	[redacted]Game\DLC\Windows

Set Live on non-Default Branch:

Upload Log Output

Preview Build

Steam Configuration

Steamworks SDK ContentBuilder Path (?): [redacted]Game\Steamworks\steamworks_sdk_159\sdk\tools\ContentBuilder

Steam Login / Steam Password: [redacted]

Show Password
Save Password ✓

Private information has been omitted, but you can see that there is a depot added for the windows build of the game, and also a windows version of the DLC, pointing to the folders mentioned in the previous section. You should repeat this for any other platforms you need to support, and then simply hit the `Upload` button.

After that and assuming there were no problems, you can then go to the `SteamPipe -> Builds` section of the Steam partner dashboard where you will be able to see the build you just uploaded and make it live.

At that point, any time you launch the game via the Steam client, the DLC content will be installed automatically (Since you will be a developer and have access by default), and you can then use the main unified DLC Api outlined in the main user guide for accessing and loading the available DLC content.