

ASSET INVENTORY

GUIDE - VERSION 1.7.0



ASSET INVENTORY

INTRODUCTION

The Asset Inventory is your ultimate asset companion: a lightning-fast search for assets outside your current project. Find content in assets you purchased or downloaded without importing and bring single files in with just a click.

Eliminate the time-consuming task of finding a sound file, a texture or a model you know you purchased but which is hidden inside one of your many Asset Store purchases. The Asset Inventory provides a **complete list of all assets you own**, including their content.



CONTENTS

Introduction	1
Features	3
Installation.....	5
Getting Started	6
Search Tab	7
Packages Tab.....	11
Reporting Tab	15
Settings Tab	17
Asset Previews	22
Advanced Features	23
Configuration	25
Technical Details	26
Limitations.....	26
FAQ.....	27
Third Party.....	29
Recommendations.....	30
Support Contact.....	31



FEATURES

The tool aims to provide **what you always wanted the Asset Store & Package Manager to be**. Find assets effectively. Search inside your purchases. Include custom packages you downloaded from Humble Bundle or other places. And packages from a registry. Manage everything through tags. Perform bulk operations. Import multiple assets and packages in one go. Identify used assets inside your projects. Search using asset-specific criteria. And much more.

Powerful Search

Browse & find anything inside your purchased Asset Store packages without importing. Quickly preview audio files. Narrow your search using asset type, tags, image dimensions, audio length, colors and more. Exclude items you don't want to see. **Save and recall** searches.

Easy Setup

Works immediately out of the box. Lots of optional view & configuration options. Hassle-free indexing. Start, stop & resume at any time. Works with Unity 2019.4 and higher. Windows, Mac & Linux. Lightning-fast indexing and search.

Intelligent Import

Import only what you need instead of a whole package. Automatically determines asset dependencies to import complex prefabs and materials. Save space & reduce clutter in your project. **Bulk import** multiple packages at once. Automatically store imported assets in a specific sub-folder and keep the Assets root clean.

Many Sources

Your complete asset library: Automatically indexes Asset Store purchases. Triggers download of missing assets. Handles packages from registries. Supports custom folders to search through Unity packages downloaded from other locations. Indexes folders and Zip archives containing **arbitrary media files** like 3D models, audio libraries, textures and more. Automatically generates previews.



Organize

Automatically imports labels from the Asset Store. Use additional **tags** to group assets effectively. Assign tags to either packages or individual files inside packages. Assign colors and group by tags. Exclude unwanted items. Perform bulk operations. Import multiple assets and packages in one go. Builds on Package2Folder to allow importing packages into a custom sub-folder.

Reverse Lookup

Quickly identify used assets and packages in your project.

Constant Updates & Support

Receive regular updates, optimizations, and new features. Super-fast support. Discuss ideas in a great Discord community.

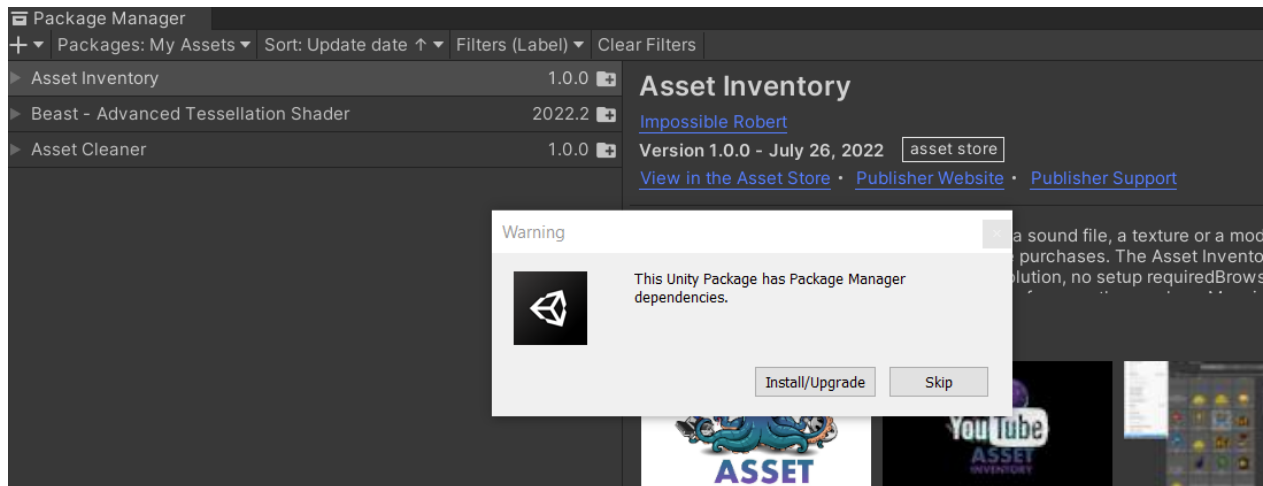


INSTALLATION

It is strongly recommended to remove any previous version of the Asset Inventory before installing a new version. The best way to do so is to delete the full Asset Inventory folder before starting Unity since sometimes libraries cannot be removed if Unity is active.

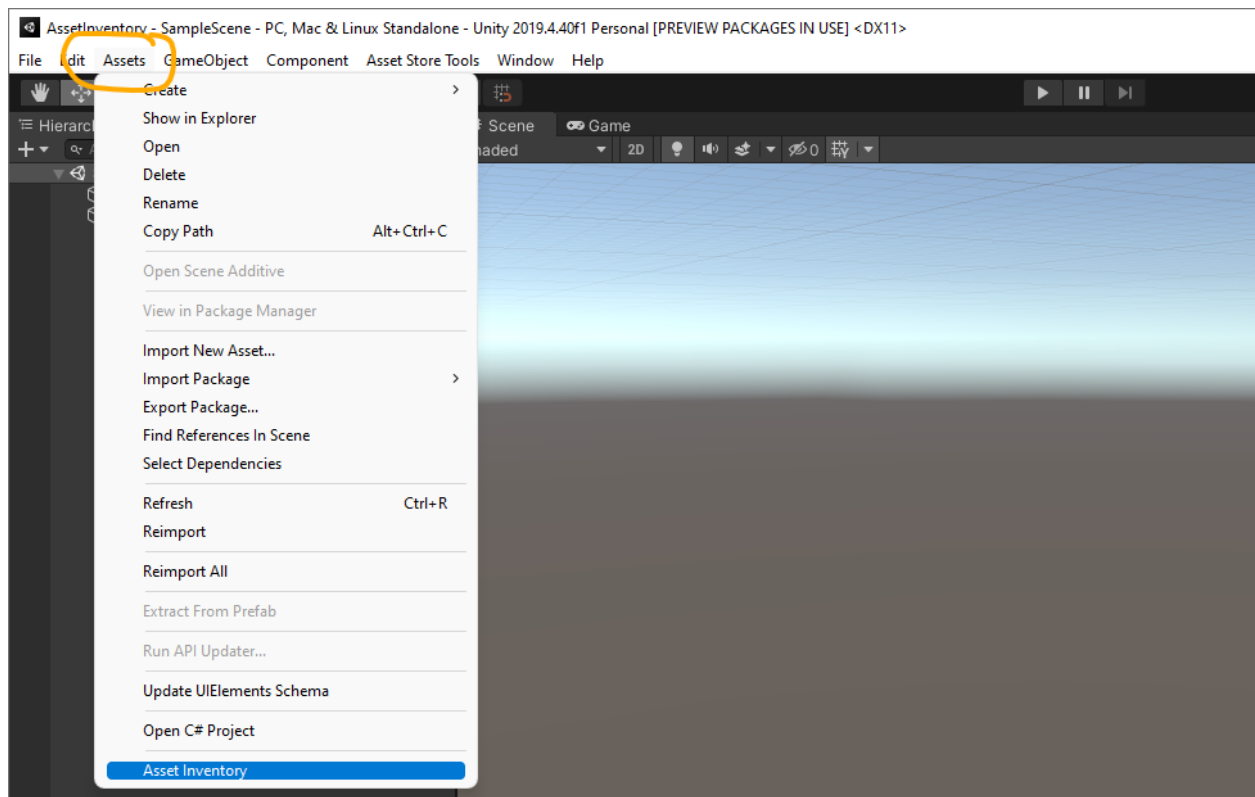
TIP: Run the Asset Inventory in a **new empty Unity project** for the initial indexing and use the newest available Alpha/Beta. The indexing results will be stored in a central location for reuse in any other project. This will give you the highest indexing performance since no other assets/scripts need to be refreshed and new versions of Unity typically bring big performance improvements for object and sound importers (in some cases 10x).

Install the package through the Unity Package Manager. When asked if to install additional dependencies, select **Install/Upgrade**, otherwise the asset will not work.



GETTING STARTED

Open the Asset Inventory through the **Assets menu**.



There is **no manual setup** needed. The only requirement is a fair amount of disk space for storing preview images and temporary files during extraction and browsing.



SEARCH TAB

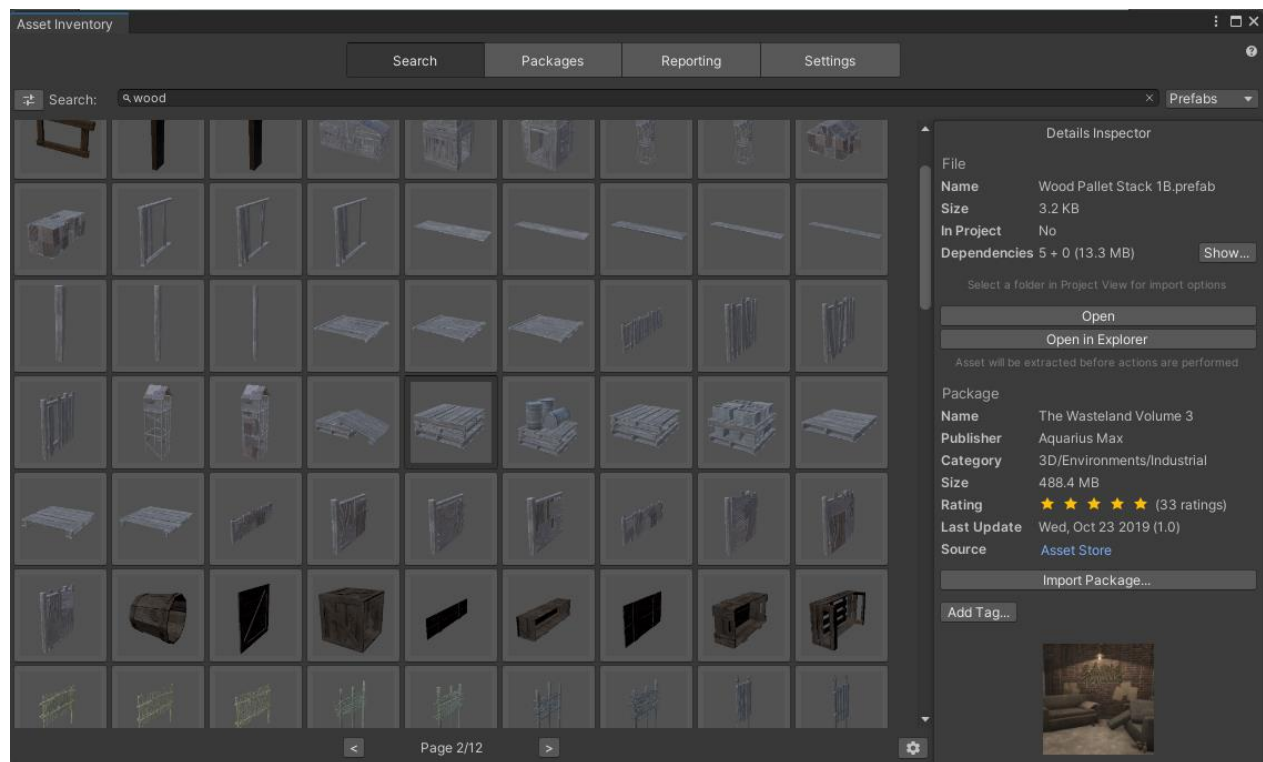
Once an asset is indexed, it can be found in the search. Searching will automatically start when typing. By default, every word entered needs to match the search result but not necessarily in the same order. If the exact phrase should be matched, prefix the search with ~.

Examples:

- "car interior" will return results like "CarBlueInterior.fbx" and "InteriorCarDes.png"
- "~car interior" will not return the above but only results like "Car Interior.fbx"
- "car +interior -fbx" will return results that match "car" and "interior" but not "fbx"

There is also an expert search available (starting searches with "=") which is described later.

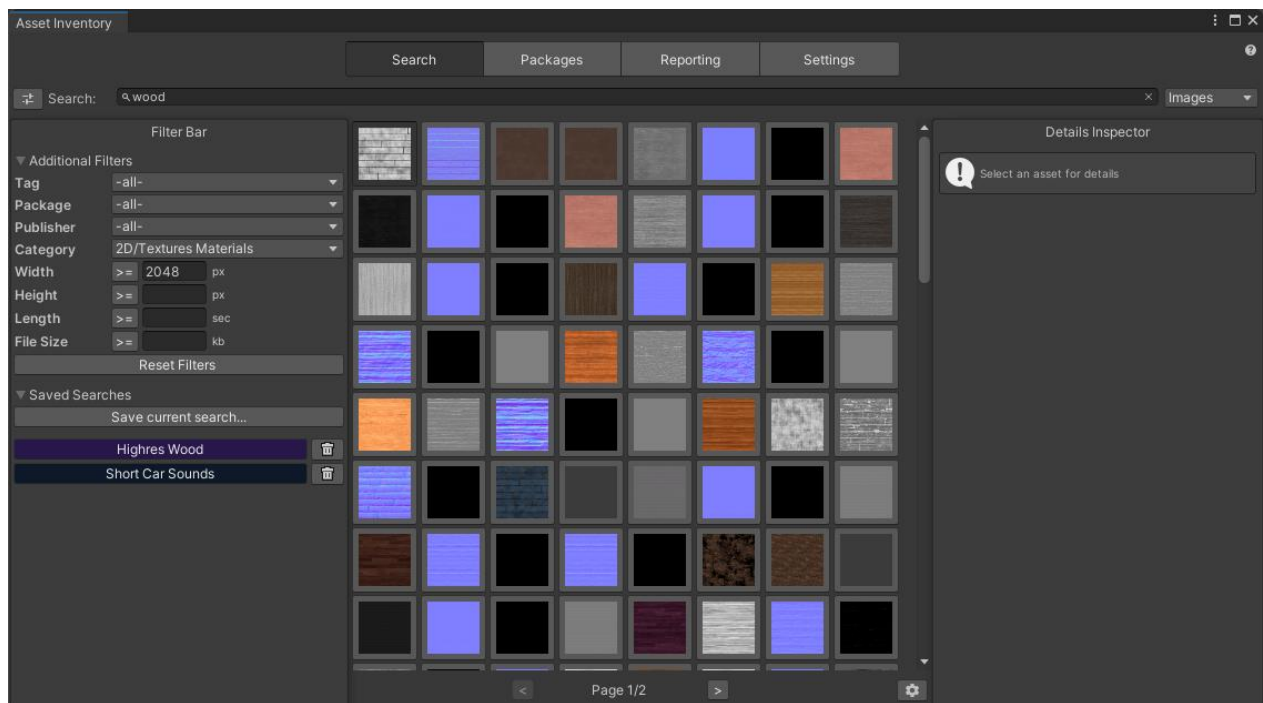
Additional filters can be selected to narrow down the results further. Once an asset is selected, details are shown on the right-hand side about the file and the package the file is contained in.



Selecting an audio file will **automatically play** it. This way it is easy to quickly preview audio files.

Clicking the **Filter icon** in front of the search will bring up the **filter bar**. It contains additional filters and media-specific properties to filter for specific image dimensions or audio length. Using the buttons in-front of each value toggles if results match that are bigger or smaller than entered. The filter dropdowns will only show values if respective assets are available.

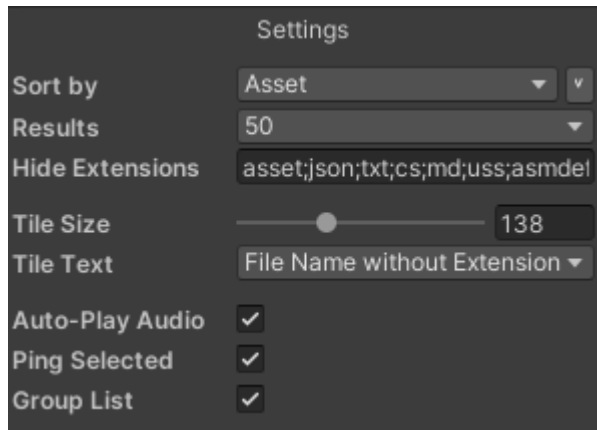
In the case of registry packages only the latest version of the package will be indexed.



The filter bar also contains the section for **Saved Searches**. This allows to persist the current search filters and later recall these easily. The results are not persisted but instead live from the database.

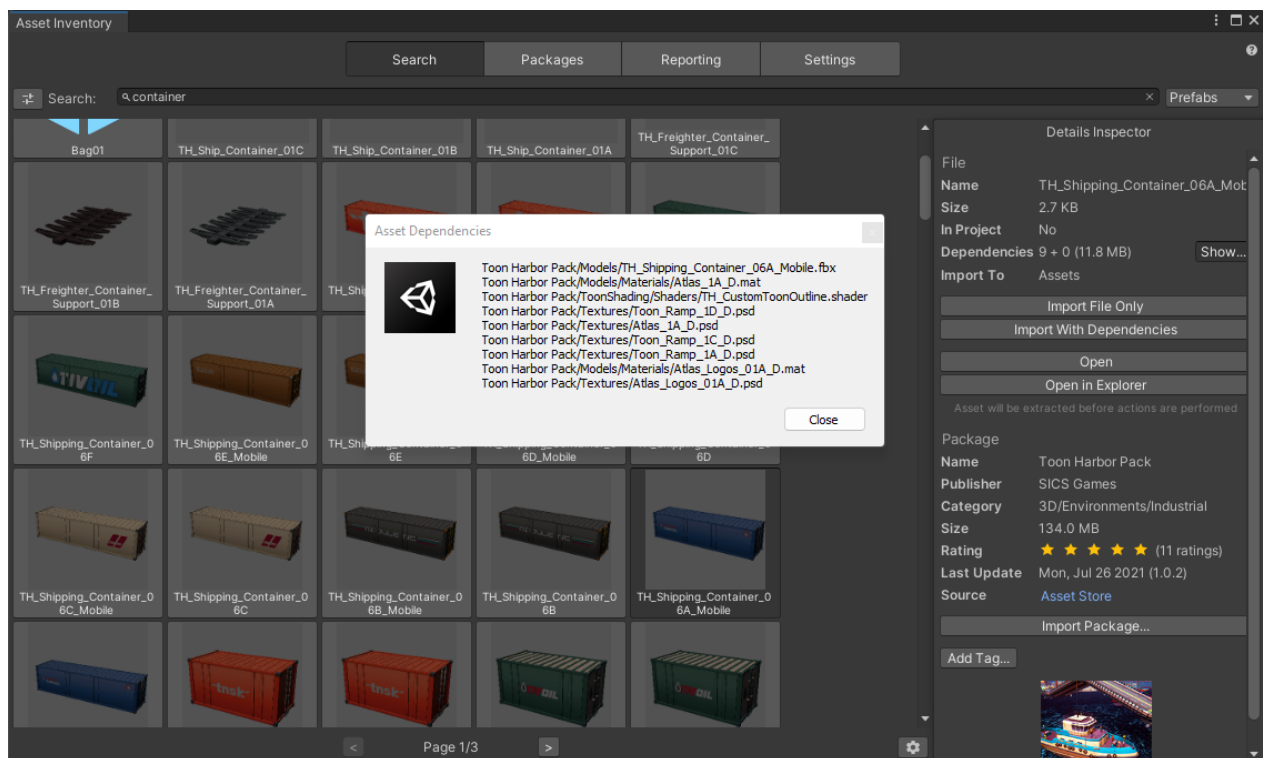


Using the **settings icon** in the lower right corner will open additional view settings. Specify the tile size, which text to display on the tiles and if assets should be pinged automatically upon selection.



IMPORTING ASSETS

Selecting a folder in the Unity **Project View** will bring up **import** options when selecting an asset in the search. Assets can be imported individually or with all detected dependencies. The latter will automatically create a sub-folder with the asset name and store all files in there.



In case there are **script dependencies** these can also be imported. Due to unforeseen dependencies in scripts, that will typically only work for scripts that are self-contained. Otherwise, there might be compilation errors.

Clicking the *Show* button behind the dependency information will bring up the dependency information details, listing all files, their size and if they are already in the project or not.

Import can also be triggered by **double-clicking** on an item or by **dragging** it into the Project Window.

EXPERT SEARCH

It is possible to use nearly the full feature set of [SQLite 3](#) to search. This mode is activated when starting the search with “=”. Afterwards the database fields and conditions can be stated.

Example

```
=Asset.PackageSize > 0 and AssetFile.Type="wav"
```

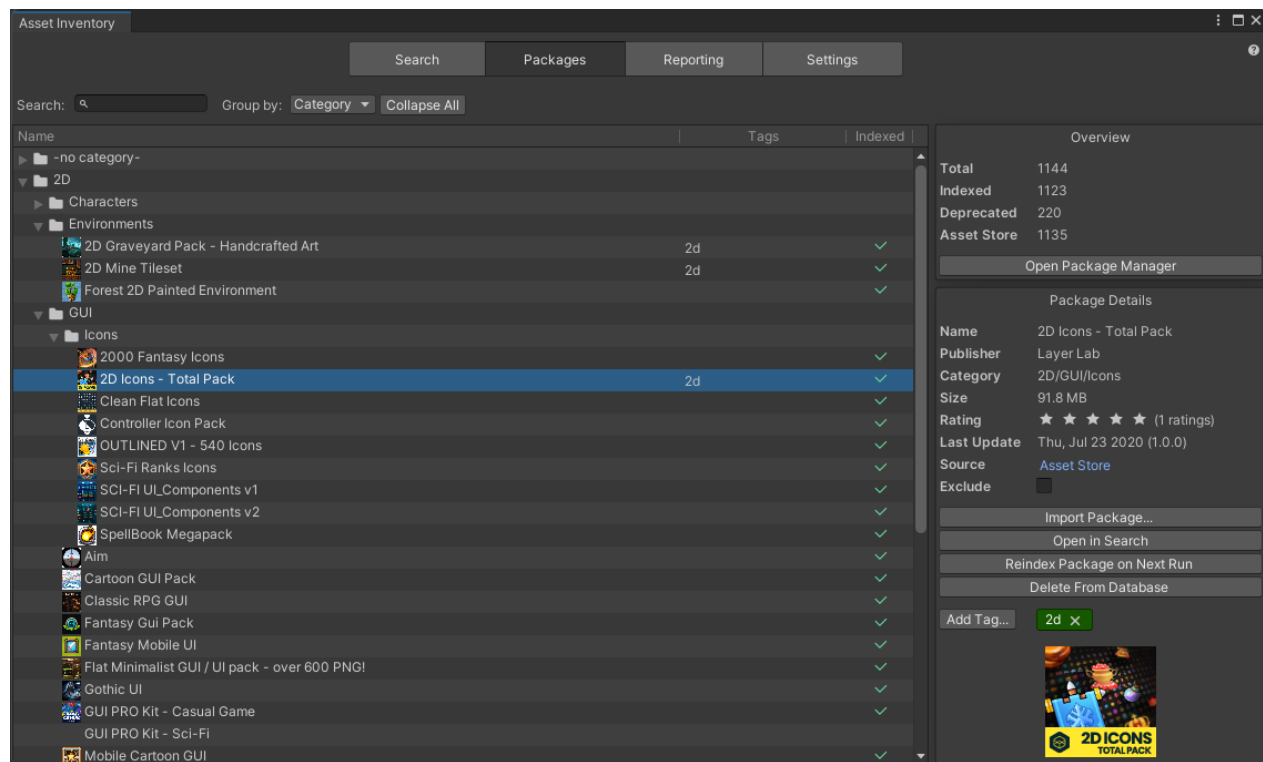


PACKAGES TAB

The packages overview will show a list of all found packages. These can come from multiple sources:

- The list of Asset Store purchases is retrieved automatically
- The local Unity asset cache is scanned
- The local Unity package cache is scanned
- If specified, additional folders will also be scanned

The list will indicate which of these were already indexed. To index more packages, download them into the cache. Display is possible as a plain list or grouped by categories, publishers, state or tags.

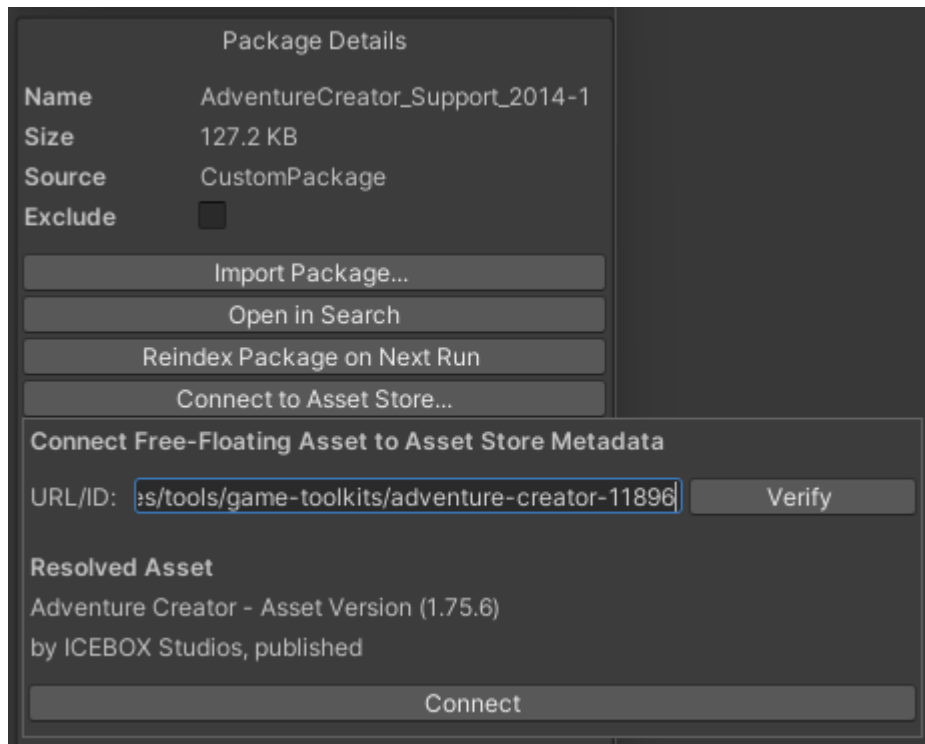


After selecting a package, multiple options will appear. It is possible to import it. Alternatively, it can also be removed from the index to trigger a reindexing on the next run. This can be useful for incorrectly indexed files. Packages can also be completely removed from the database which can be useful after cleaning up outdated assets from the local cache. **Double-clicking** any package will show the contents in the search.

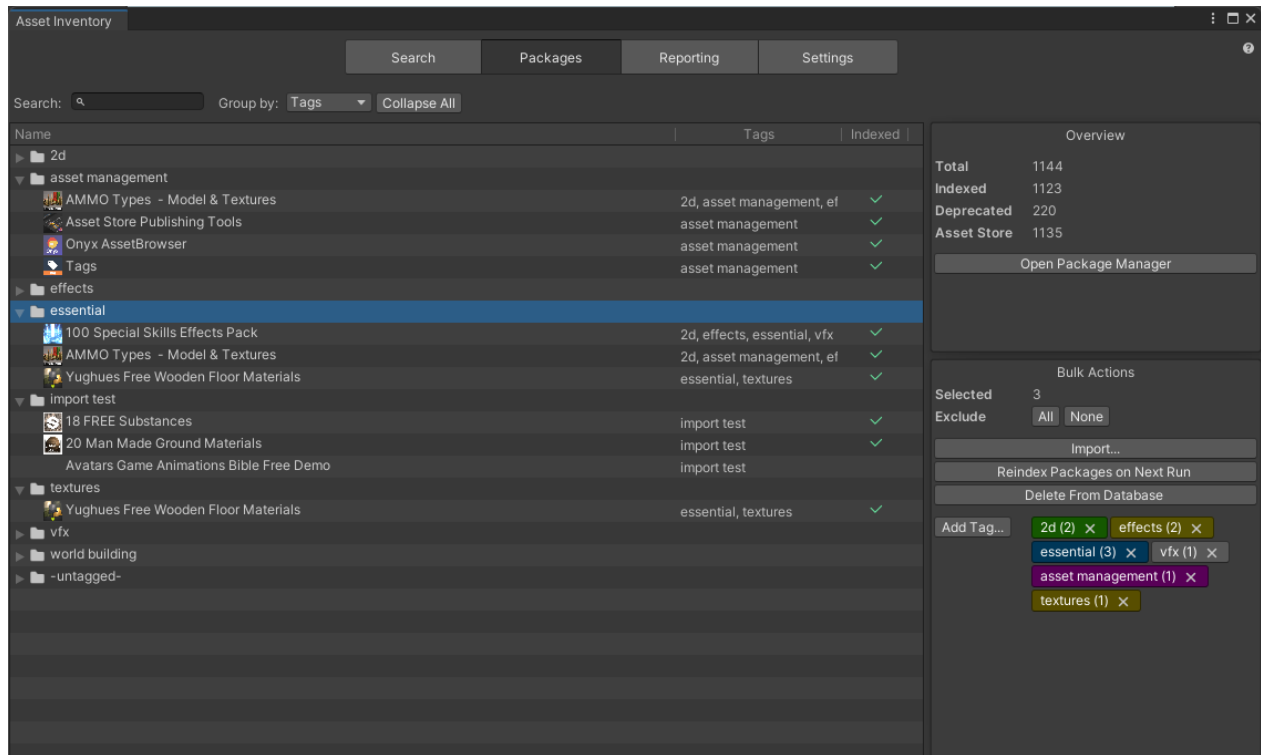


Depending on the package type additional options become visible:

- Registry Packages
 - Support for setting a preferred version which will be used as the default when importing.
- Custom Packages
 - Support for connecting to metadata from the Asset Store. This will load the name, description, rating etc. and is especially useful when having downloaded packages from alternative sources like the Humble Bundle to still have all metadata available.
 - Support for disconnecting from the Asset Store again



Tags will be imported from the Asset Store in case any are set there. In addition, local tags can be added and removed here as well (they are not synchronized back to the Asset Store yet). Bulk editing of tags is possible when selecting multiple items in the tree.



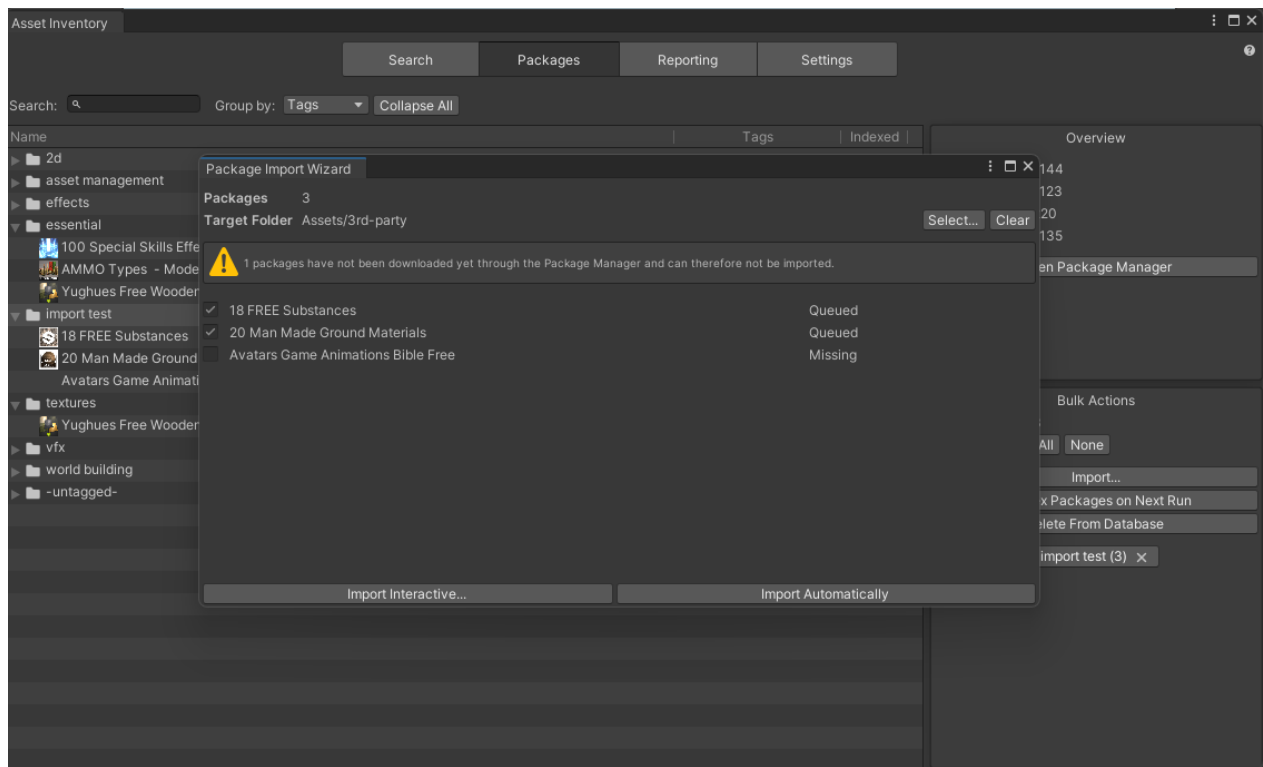
When adding tags, the **Tag Management** window can be opened through the small cog wheel. There, tags can be created, colored, renamed and deleted.

In case packages should not appear in the search results and not be indexed, the **Exclude** toggle can be used in the package details.

Bulk selection will also show the total size of the compressed assets before they are downloaded so that you can estimate if temporarily downloading all for indexing would still fit on the hard drive.

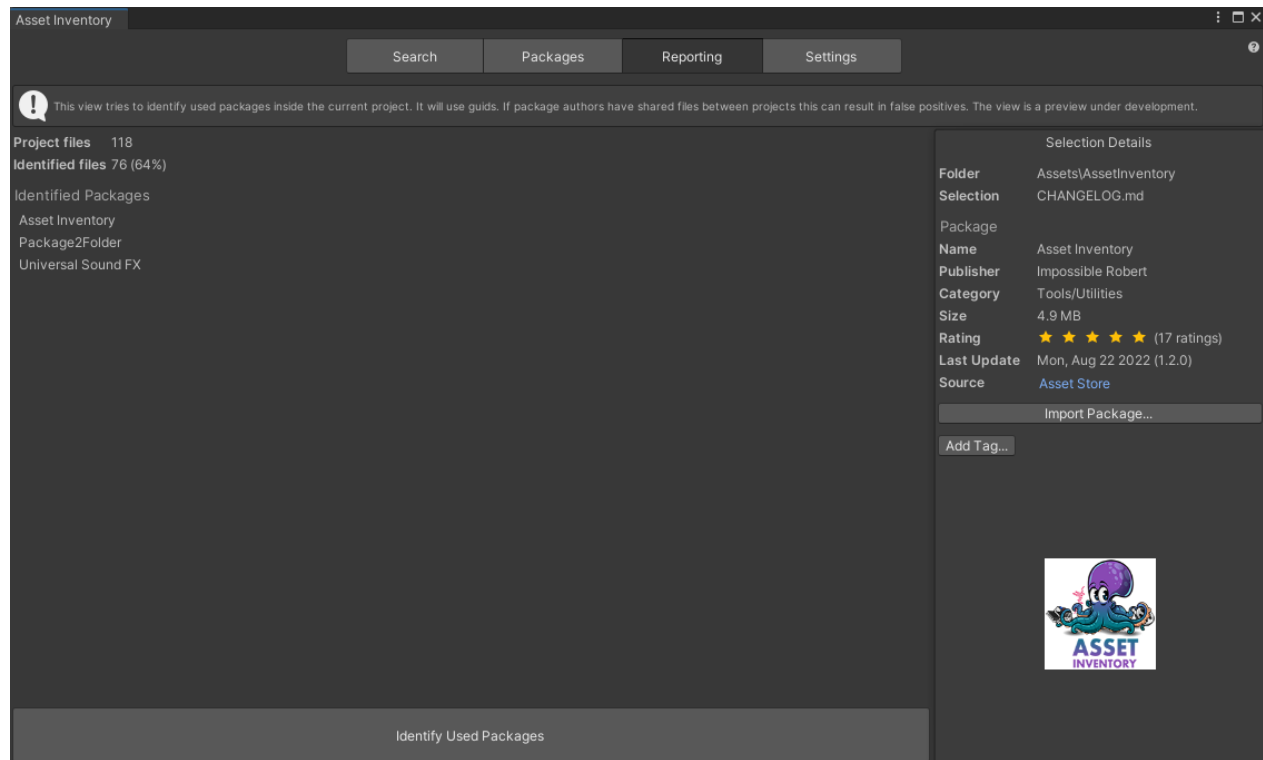


Importing packages can happen one-by-one and in bulk while multiple packages are selected. Both interactive and automatic mode is available. In addition, a **custom root folder** under which assets should be imported can be selected. This is especially helpful if the `/Assets` root should be kept clean and organized. Packages from registries will be added to the project manifest. If registry packages need a custom scoped registry this will also be added automatically.



REPORTING TAB

This module will scan the complete project and try to identify packages that were being used. It also supports exporting your data. Currently in **preview** this will evolve into a compliance reporting to check if all licenses in the project are accounted for.



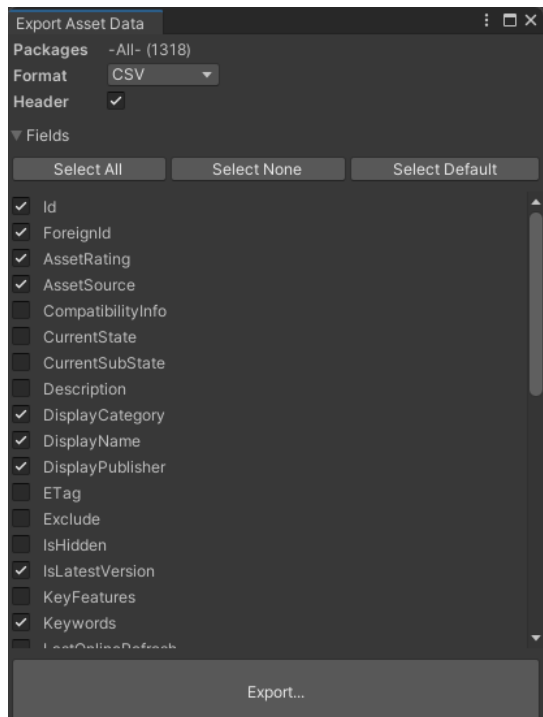
When selecting an asset in the Unity Project View the reporting tab will try to identify the associated package.

DATA EXPORT

In case you want to use your own analysis or presentation style it is possible to export the database contents into other formats. Currently supported is:

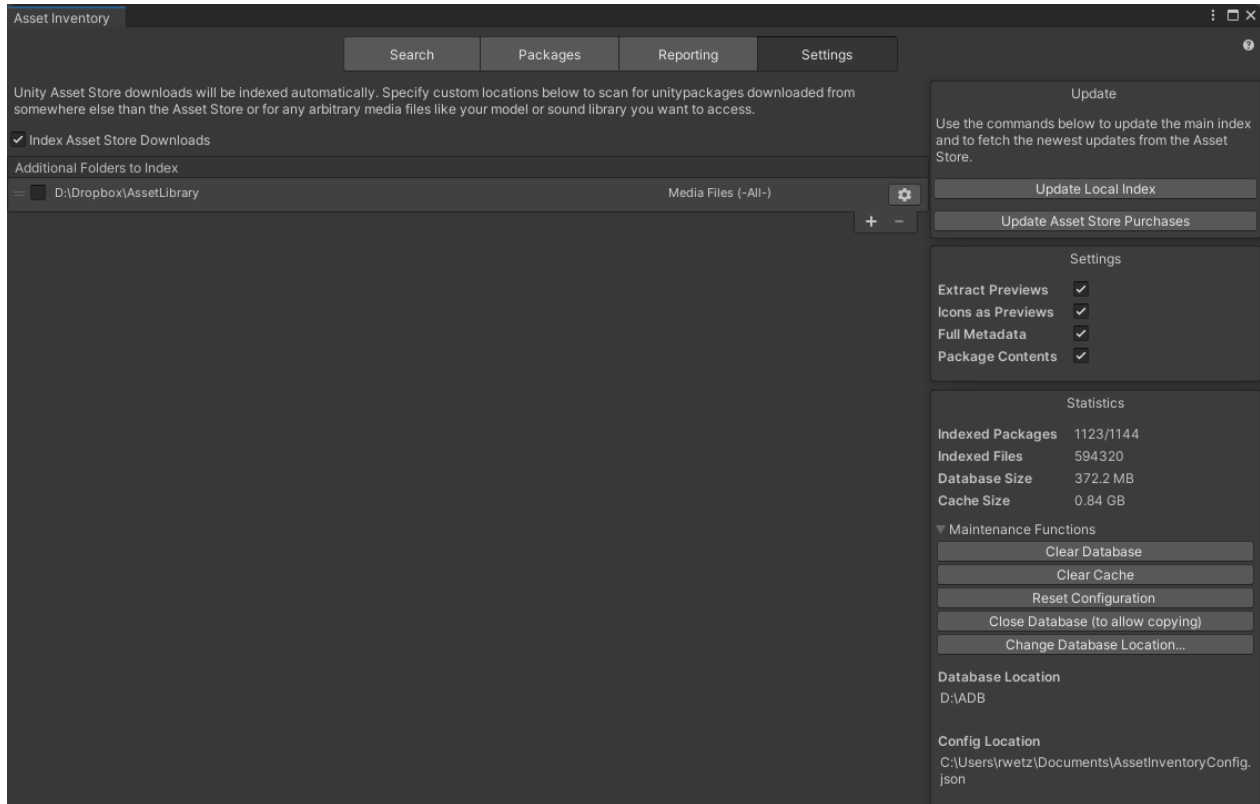
- CSV: can easily be handled in Excel and other compatible software





SETTINGS TAB

Use this section to configure what should be indexed. Indexing can be started and stopped at any time and will pick off again where it last stopped.



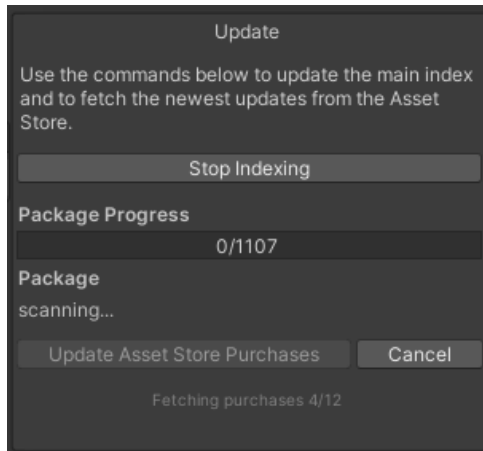
Asset Store downloads and Packages are activated by default. That means you only need to click the **Download** button in the **Package Manager** window or right in the Asset Inventory for each asset you own (without importing them) to make them available for the index.



During a normal update cycle the following actions are done in this order:

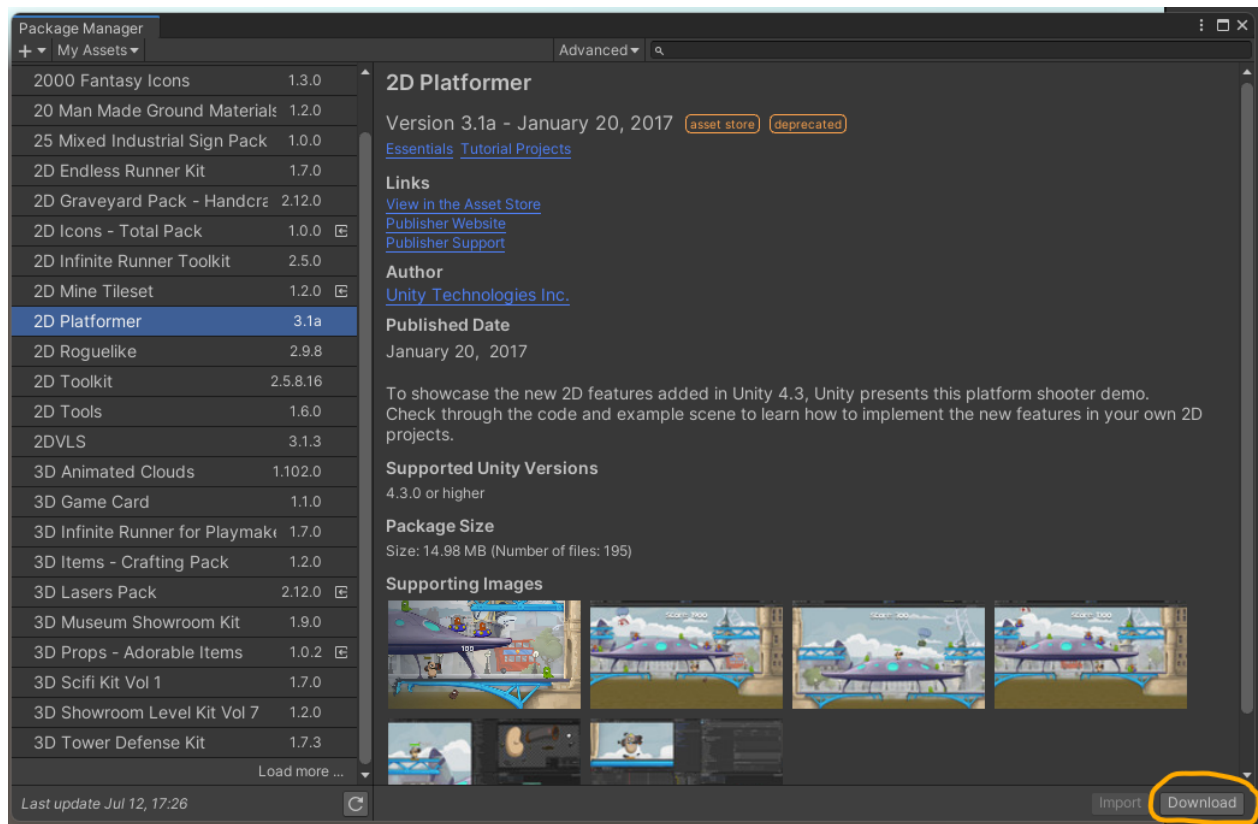
- Fetch all Asset Store purchases
- Fetch details for all purchases and for assets that received updates
- In parallel scan for new local Asset Store packages and update metadata
- Do the same for registry packages
- Index the contents of new or changed packages
- Index any additional folders if specified
- Extract colors of new assets for color search
- Perform backup activities

All activities are incremental, meaning you can interrupt and restart these at any time and they will mostly continue where they left off.



Tip: Starting from Unity version 2022.1 the Package Manager supports bulk downloading assets. Download all needed assets that way in one go. Bulk downloading is also available right in Asset Inventory from Unity 2020 and above.





If you want to index Unity packages downloaded from **other sources**, simply add the folders to the additional folders list. In case a folder ends with “*Asset Store-5.x*” it will be assumed to be a Unity compatible structure containing the *publisher/category/asset* layout. This is the best workaround when defining a **custom asset cache directory** in Unity 2022.1 and higher.

When selecting an entry from the list of additional folders, more options can be specified on the right-hand side. This way it is possible to select what types of files should be searched for:

- Unity Packages: finds any *.unitypackage* files in the folder and indexes the content
- Media Files: allows to index all kinds of other, **free-floating** assets, like images, models, audio libraries etc.
- Zip Archives: will extract archives on the fly and index all contents

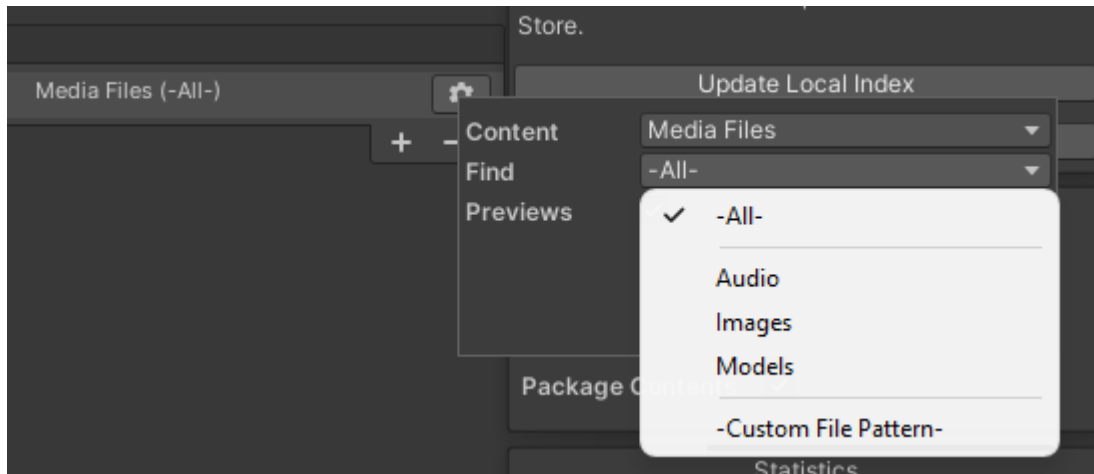
Using additional media folders will let the Asset Inventory act as the go-to place for asset management and quickly finding any available asset on your drives from a single, consistent UI. Media files will have no connected asset but otherwise behave the same. If they have a *.meta* file next to them their guids will also be stored. The order in which



additional folders are processed can be changed by dragging them up or down in the list. Deactivating an additional folder will not remove it from the index.

Once an asset is indexed it does not need to remain on your hard drive if you just want to search for it. Only when importing it needs to be available.

Preview file generation will work by temporarily copying each file into the current Unity project, letting Unity create a preview and removing it again. This process will take a bit of time but will allow Asset Inventory to show preview images and also additional meta data for more image files than just *png* and *jpg* (e.g. *gif*).



BACKUP

The tool supports copying packages to a dedicated place for permanent backup. This way you are not affected by package deprecation by Unity where it might be impossible to download older (but for you working or compatible) versions of assets anymore.

The backup tool can keep multiple versions of packages and will automatically handle the life-cycle, version comparison and more.



MAINTENANCE

Maintenance functions are also located on the Settings tab:

- ⇒ Optimize Database: compacts and reorganizes the database for more performance and less required space, should be done after bigger indexing activities
- ⇒ Clear Database: delete the complete database to start over
- ⇒ Clear Cache: delete the temporary files to save space, can be done without any side-effects
- ⇒ Reset Configuration: set everything to like it was initially
- ⇒ Close Database: allow backing up or copying the file
- ⇒ Change Database Location: set a different location for the database, move the current, use another existing one or create a new one



ASSET PREVIEWS

The search result will show previews of the assets. These previews can come from three sources:

- Included in the package and created at the time the asset was built
 - This is done by the asset store tooling and also the reason why e.g. previews for sound files can look different depending on which version of Unity was used.
- Created by Asset Inventory during import
 - This happens automatically for atomic assets without dependencies, e.g. sound files, textures or models.
- Created by Asset Inventory on-demand
 - This mode can also create previews for prefabs and materials but will take **much** longer since all dependencies need to be materialized first.

It is possible to recreate a single preview from inside the search view, all previews for a package or perform recreation for missing previews in the complete database.

Previews are created by the Unity Editor. This works well but is limited to 128 x 128 pixels. Also, VFX and 2D prefabs cannot be previewed this way. A custom mechanism is under way for future releases.

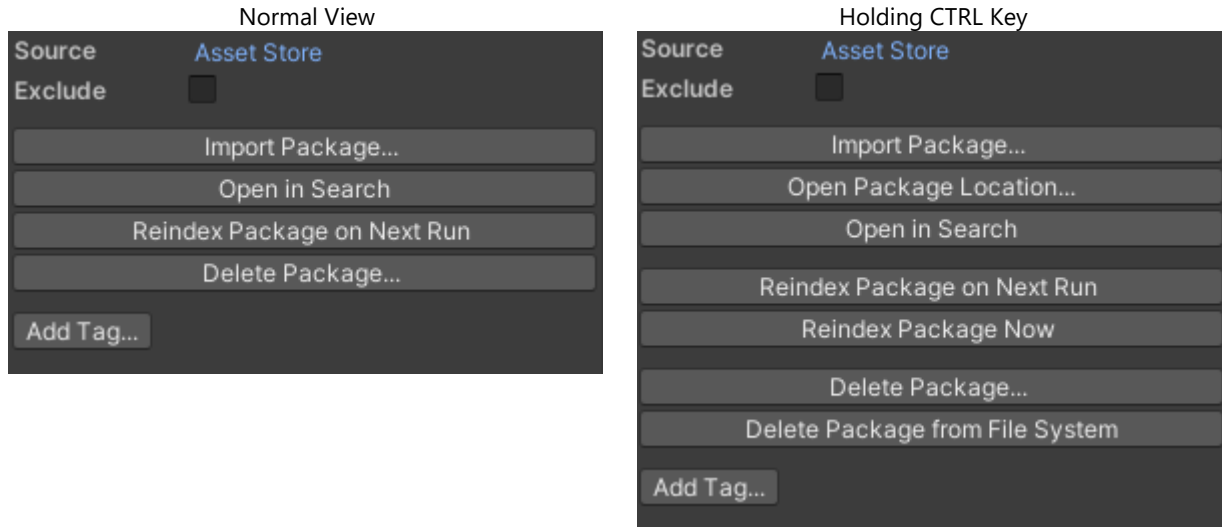
While creating previews it might happen that a new folder called *_TerrainAutoUpgrade* will appear under *Assets* as a side-effect.



ADVANCED FEATURES



















EXPERT FUNCTIONS

Many more seldom used and advanced features are initially hidden to not clutter the UI. They can be made visible by holding down the **CTRL** key.


















ASSET INVENTORY – USER GUIDE – 1.7.0

Asset

 Id	integer
 AssetSource	integer
 Location	varchar
 ForeignId	integer
 PackageSize	integer
 PreviewImage	varchar
 SafeName	varchar
 DisplayName	varchar
 SafePublisher	varchar
 DisplayPublisher	varchar
 SafeCategory	varchar
 DisplayCategory	varchar
 Slug	varchar
 Revision	integer
 Description	varchar
 KeyFeatures	varchar
 CompatibilityInfo	varchar
 SupportedUnityVersions	varchar
 Keywords	varchar
 Version	varchar
 LastRelease	bigint
 AssetRating	varchar
 RatingCount	integer
 MainImage	varchar
 Requirements	varchar
 ReleaseNotes	varchar
 ETag	varchar
 CurrentState	integer
 OfficialState	varchar
 IsHidden	integer
 Exclude	integer
 LastOnlineRefresh	bigint
 License	varchar
 LicenseLocation	varchar
 IsLatestVersion	integer
 PreferredVersion	varchar
 Registry	varchar
 Repository	varchar
 PackageSource	integer
 MainImageSmall	varchar
 MainImageIcon	varchar
 OriginalLocation	varchar
 OriginalLocationKey	varchar
 LatestVersion	varchar
 CurrentSubState	integer



AssetFile

 Id	integer
 AssetId	integer
 Guid	varchar
 Path	varchar
 SourcePath	varchar
 Type	varchar
 PreviewFile	varchar
 Size	integer
 Width	integer
 Height	integer
 Length	float
 FileName	varchar
 PreviewState	integer
 DominantColor	varchar
 DominantColorGroup	varchar

Tag

 Id	integer
 Name	varchar
 Color	varchar
 FromAssetStore	integer

TagAssignment

 Id	integer
 TagId	integer
 TagTarget	integer
 TargetId	integer



CONFIGURATION

Settings will be saved automatically in a file called *AssetInventoryConfig.json*. This happened by default in *Documents* folder of the currently logged in user. This way the tool can work independently of the Unity project and the asset search is available in an identical way everywhere.

It is possible to force the tool to use a project-specific configuration. To do so, copy the *AssetInventoryConfig.json* file anywhere into the project and restart Unity. The detected location will be shown on the **Settings tab** under **Maintenance Functions**.

Some advanced settings can only be set in the *.json* file and are not otherwise accessible through the UI. An example for this is the delay after which the search will start searching. Feel free to adjust these values as well.

ADVANCED CONFIGURATION

In order not to clutter the UI some seldom used settings are to be set directly in the configuration file. It is in formatted JSON format and can be opened in any text editor. The following properties do only appear there:

- **autoHideSettings** (*bool*): flag if to close view settings automatically or only when clicking settings button again
- **hueRange** (*float*): degrees to widen color search
- **maxResultsLimit** (*int*): hard limit of maximum results to be shown even if -all- is selected
- **searchDelay** (*float*): seconds after typing a character to wait until triggering search



TECHNICAL DETAILS

The index will be stored in an SQLite database. This database will be automatically created upon first usage. It is in your *Documents/AssetInventory* directory. The size should be rather small but increases with the number of indexed assets. As a rule of thumb assume 20Mb database + 200Mb preview images per 30k files.

Inside the *AssetInventory* directory three additional folders will be created:

- **Previews:** containing all images for the asset catalog
- **Extracted:** containing temporarily extracted files. You can safely delete the *Extracted* folder at any time if needed.
- **Backup:** acting as the default location for package backups

Some operations require Unity to perform work, e.g. creating previews. For such tasks the folder *_AssetInventoryPreviewsTemp* will appear under Assets for a short time and automatically be removed again.

LIMITATIONS

- Dependencies can only be calculated in packages that use *text/yaml* serialization. A few legacy packages still use *binary* serialization. These packages also work in most cases except if Unity cannot resolve them, e.g. because of missing scripts on prefabs.
- Importing scripts is experimental and can produce console errors if the script requires other scripts or assembly definition files to run.
- Running the game view maximized on play and having the Asset Inventory window docked will cause it to reinitialize, resetting all search settings.



FAQ

SOME PREFABS SHOW NO DEPENDENCIES

This is due to the serialization format. Only assets serialized as text can properly be parsed right now.

SOME AUDIO FILES USE DIFFERENT COLORS IN THE PREVIEW THAN OTHERS

Preview colors depend on which Unity version an asset author used to upload his asset. Unity seems to change colors over time between Unity versions. There could be a feature someday to recreate preview images with the current version of Unity to make them all uniform depending on interest.

SOME PREVIEWS ARE GREY AND DON'T SHOW ANYTHING

This can happen if the Asset Store tooling did not create a correct preview image while uploading an asset. An example is the Danger Zone asset from Unity released for 2021+. Functionality to recreate preview images is under development.

CONSOLE SHOWS ERRORS DURING INDEXING

There are a couple of errors Unity creates that cannot be intercepted but can safely be ignored. These are:

"Cannot create FMOD::Sound instance for clip "" (FMOD error: Unsupported file or audio format.)"

- It typically means that an audio clip was saved with the wrong extension, e.g., a wave as an ogg by an asset publisher. Use the **Open** button to start the native file player which will typically play the file correctly then.

"TEXTURE HAS OUT OF RANGE WIDTH / HEIGHT"

- In that case Unity cannot read the texture dimensions and you will not be able to see the dimensions of the texture or filter for it. Otherwise, it does not have any effect.

"ArgumentException: Getting control 4's position in a group with only 4 controls when doing repaint"



- This can happen if a lot of UI changes are going on while indexing is happening. Can safely be ignored and is without any effect.

"Curl error 28: Operation timed out after 30000 milliseconds with 0 bytes received"

- This can happen if Unity servers did not respond in time and the details for an asset could not be retrieved. This is solved by starting the Asset Store update process again.

"Invalid or expired API Token when contacting..."

- Happens if Unity was not in use for a longer period. The user login happens in the Unity hub and in that case the token for online access has expired. Solved by restarting Unity.

"Could not create asset from Assets/_AssetInventoryPreviewsTemp/1016771.png: File could not be read"

- Happens if Unity cannot create a preview for specific assets or if this is a malformed png. Can be ignored.

"TLS Allocator ALLOC_TEMP_THREAD, underlying allocator ALLOC_TEMP_THREAD has unfreed allocations, size 1261"

- Can occur after triggering a download. No negative side-effect seen so far. Can most likely be ignored and seems to be a Unity issue.

"You are trying to replace or create a Prefab from the instance '...' that references a missing script. This is not allowed. Please change the script or remove it from the GameObject."

- Can occur during creating preview images when a prefab has script dependencies. Can be ignored since Asset Inventory fixes this situation automatically but the error cannot be suppressed.

"NormalMap settings dialog comes up"

- Can occur during creating preview images when a texture is misconfigured in a prefab. Can be ignored since this is typically invisible in previews.



THIRD PARTY

Asset Inventory uses multiple third-party libraries that really make it shine.

- [Package2Folder](#) by Code Stage uses an ingenious idea to intercept the Unity package manager and adjust the target folder where to install assets to.
- [SQLite](#) is a great way to store data in a single file while providing easy and fast database operations.
- [SQLite-Net](#) is an amazing extension to SQLite offering convenient high-level functions when executing SQL queries.
- [Editor Audio Utils](#) does a great job to allow playing audio files also while not in play mode.
- [Redcode's AwaitExtensions](#) are extremely helpful to move over *IEnumerator* based coroutines to the more modern async paradigm of C# and use both in tandem.




RECOMMENDATIONS


I want to use this space to highlight selected assets that I think are very valuable and have friendly relations with where we support us.

THE GREAT ASSETS BY [CODE STAGE](#)


A collection of high-quality tools to boost your game and Unity productivity.




CODE STAGE
Anti-Cheat Toolkit 2021
★★★★★ (363) | ♥ (585)
€71.47 [Add to Cart](#)




CODE STAGE
Maintainer
★★★★★ (153) | ♥ (1608)
€26.80 [Add to Cart](#)



CODE STAGE
Advanced FPS Counter
★★★★★ (328) | ♥ (1165)
€8.93 [Add to Cart](#)



CODE STAGE
Code Stage Tools Bundle 2021
(not enough ratings) | ♥ (21)
€85.76 [Add to Cart](#)



CODE STAGE
Package2Folder
★★★★★ (25) | ♥ (140)
FREE [Add to My Assets](#)



SUPPORT CONTACT

Web: <https://www.wetzold.com/tools>

Discord: <https://discord.gg/uzeHzEMM4B>

Roadmap: <https://trello.com/b/SOSDzX3s/asset-inventory>

