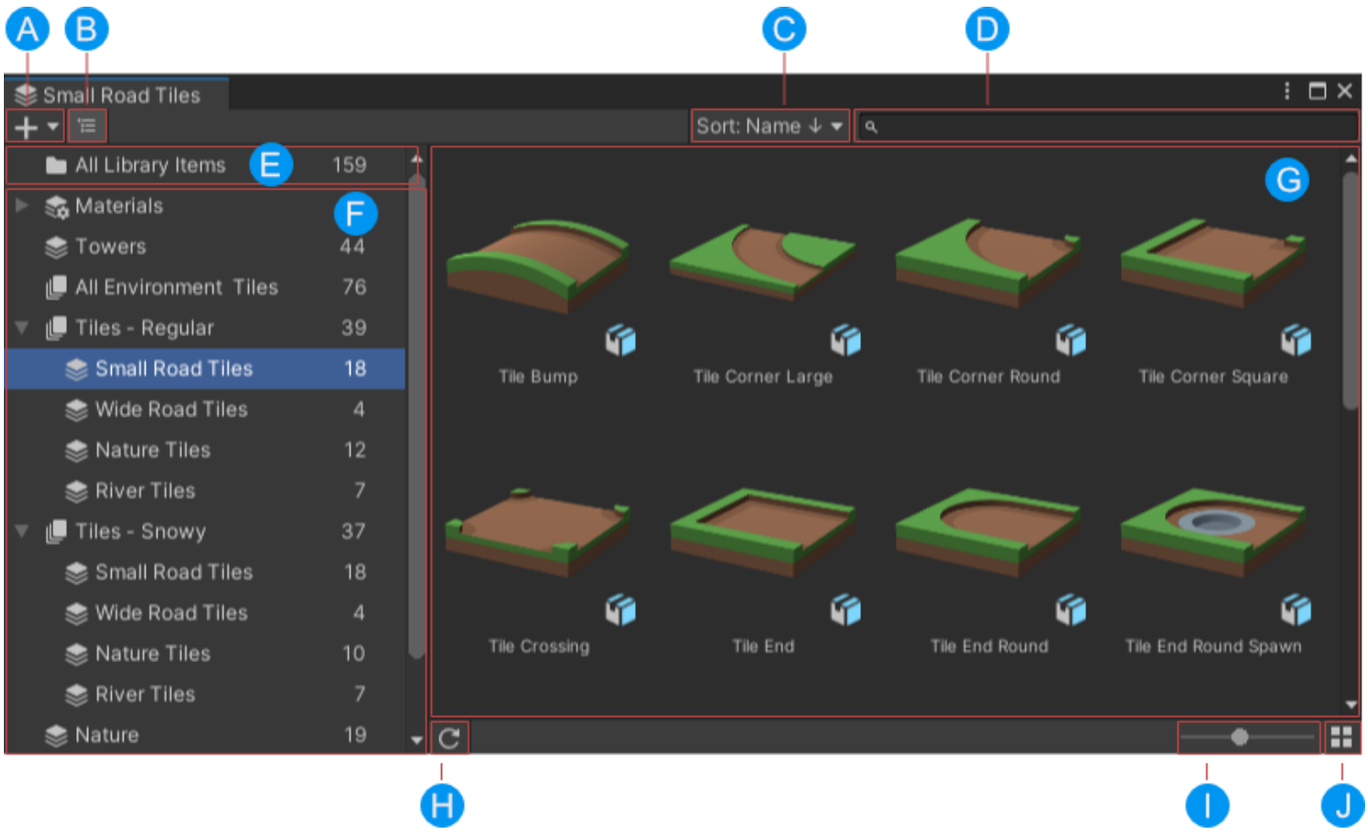
Getting Started

If using Unity 2020.3 or older, then the package QuickSearch 3.0.0-preview must be added. Preview packages need to be enabled to find it in the package manager. It requires the 3.0 preview package, not 2.0.

Window Overview

To open the Smart Library window use the menu (**Window > Smart Library**) or the shortcut “Shift + L”.



A	Add Button	Allows you to add collections to the library by selecting a type from a dropdown menu.
B	Collections View Visibility	Allows you to hide the collections tree view.
C	Sort dropdown menu	Allows you to sort the assets in the item area by ascending or descending order of either their name or type.
D	Search Field	Allows you to search in the selected collection for assets by their name.
E	All Items	A permanent entry in the tree view that contains every assets that is in at least one collection in the library.
F	Collections Tree View	Displays all the collections in the library.
G	Item Area	Displays the assets in the selected collection.
H	Update Items Button	Allows you to manually run a check on the selected collection to ensure that all assets in it match the collection’s rules, removing any that don’t.
I	Item Size	Allows you to set the size of the items in the item area.
J	Item View Style	Allows you to toggle the item area between a grid and list style view.

Collections

Collections are the main feature of Smart Library. They hold references to assets in the project.

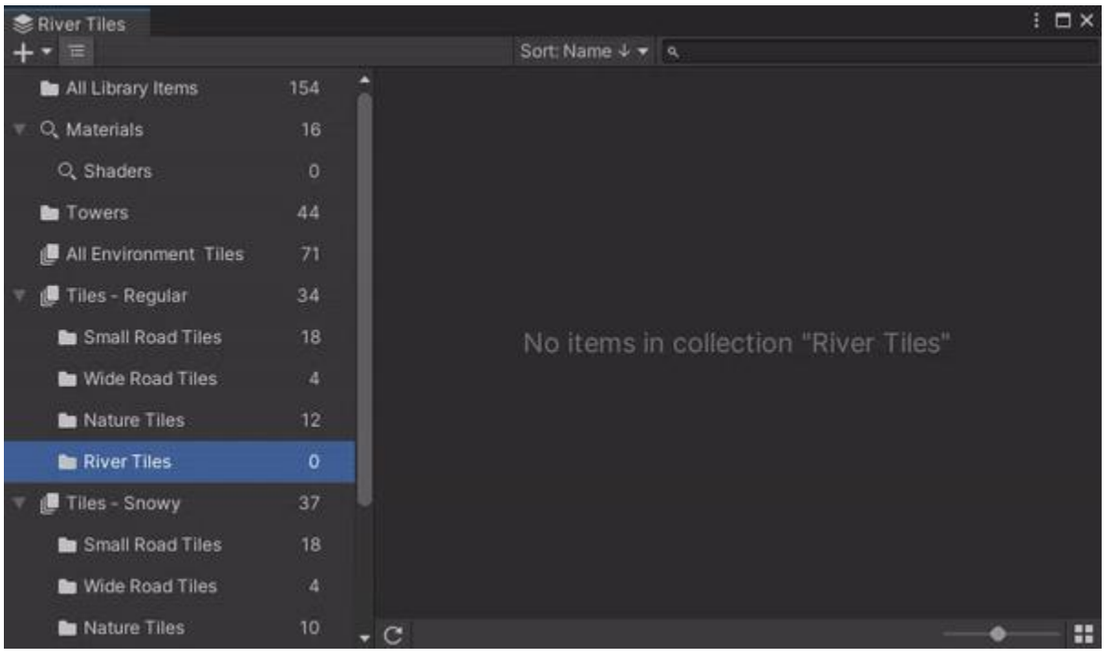
They have a hierarchical structure and can be parented to one another for better organization.

Configuring Collections

Every collection has settings that can be changed to modify how the collection functions. These can be accessed either through the collection context menu, or by click the gear icon next to a collection in the Collections Panel.

Standard Collection

The standard collection type, assets can be added to it by dragging and dropping them on to either the item panel, or on to the collection in the collections panel.
Assets can be dragged from other collections too. Holding down Ctrl while dropping an asset will copy the asset from its current collection to the new collection if both support manually adding and removing assets. If Ctrl is not held, then the asset will be just be moved instead.
Adding a folder adds all the assets in it to the collection instead of the folder itself.



Assets can be removed by either pressing the delete key while they are selected in the Library window, or through the context menu.

Smart Collection

Smart Collections find assets in the project that match its rules and folders, adding them to the collection. Also automatically removes assets from the Smart Collection if they change or are moved and no longer match the rules.

At least one folder or rule must be defined for the Smart Collection to work.

Settings

Folders

The list of folders to match an asset against.

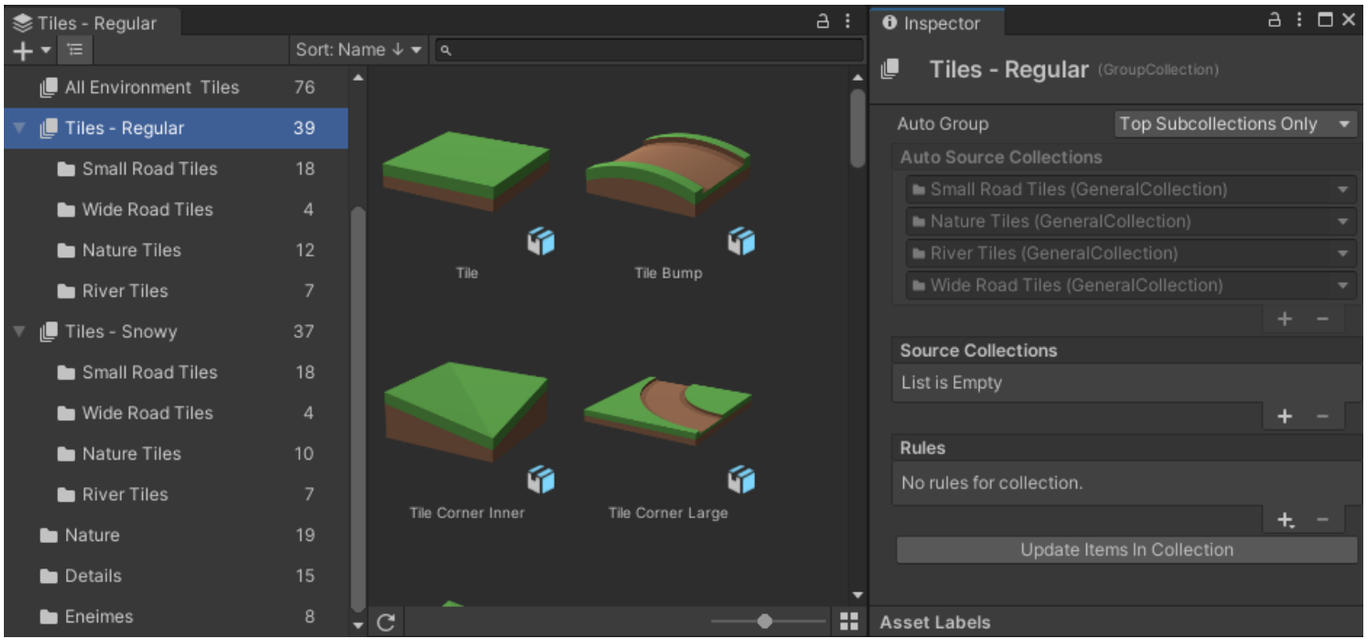
Options

- Include: Whether the asset needs to be in the folder or can't be in the folder.
- Any Depth: Matches the asset as long as the folder is in it's path at some point.
- Top Only: Matches the asset only if it is directly inside of the folder.
- Field: The target folder. Supports drag and drop.

Compound Collection

Compound Collections are used to show the assets from multiple collections at once. Assets cannot be added by drag and dropping or removed by deleting.

If an asset does not match the rules of the Compound Collection, then it will not be added. If no rules are defined, then all assets can be added.



Settings

Auto Source

An option for the Compound Collection to automatically get the assets from its subcollection without having to specify them.

- None: Will not automatically get assets from its subcollections.
- Top Subcollections Only: Automatically gets the assets from subcollections who are direct children of it.
- All Subcollections: Automatically gets the assets from all subcollections.

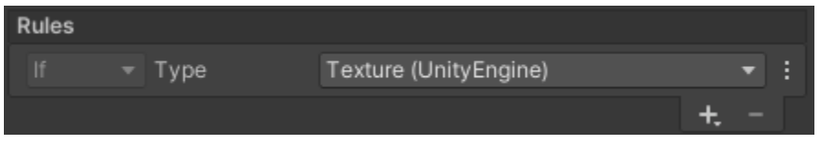
Source Collections

The list of collections to get assets from.

Rules

Every Collection contains a stack of rules that determined what assets can and cannot be added to the Collection. Before an asset is added to a collection it is compared to the rules to determine if it can be added. The rules are evaluated from the top down, comparing the asset to each rule.

There are several different types of rules, each evaluates something different about an asset. For example, the Type Rule compares the type of the asset (Texture, Material, Prefab, etc.) to type that the user specifies, the asset must be the same time to successfully pass the rule.
If there are no rules for a Collection it defaults to allowing all assets.



(Only assets that are Textures can be added to the collection)

Rules located in a Collection's settings, accessed either through the gear icon on the collection in the collections panel, or through the "Properties…" item in the context menu of the collections panel.

Operand

Each rule has an operand which determines how the result (true, or false) of the rule's evaluation is handled in relation to the rules above it.
The operands consist of "AND", "AND NOT", "OR" and "OR NOT". These correspond to the logical operand "&&", "&& !", "||", and "|| !".

The operand of the first rule is replaced with "If" and grayed out because it has no rules above it that evaluate the result of its evaluation.

Scopes

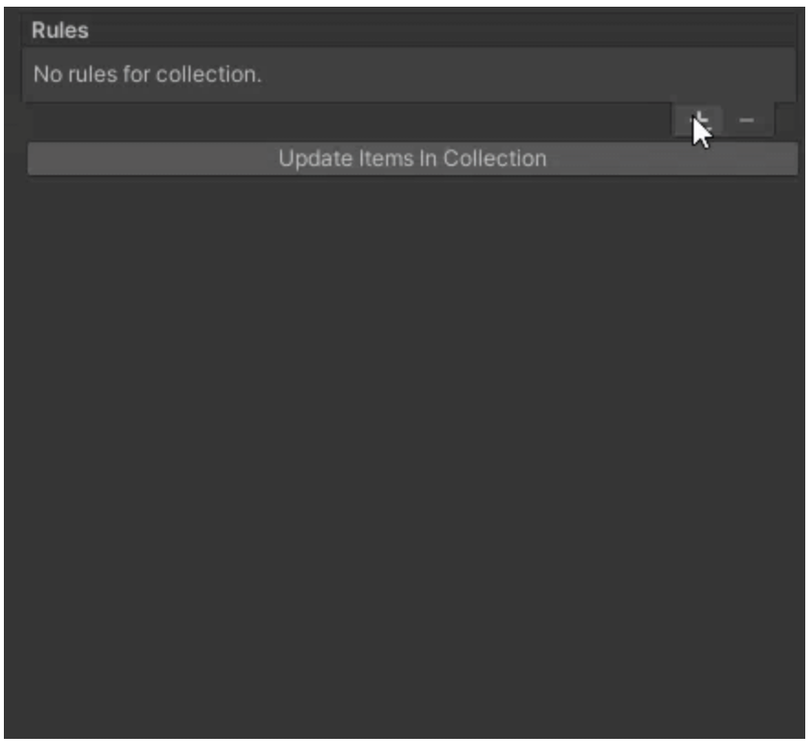
The menu on the right of each rule can be used to create and edit scopes. The rules inside of a scope are evaluated separately from the rest of the rules stack as their own self-contained stack. The result of a scope is evaluated with the rest of the main stack as if it were a rule.

The operand of the first rule in a scope is used to determine how the result of the scope is handled by the rules outside of the scope.

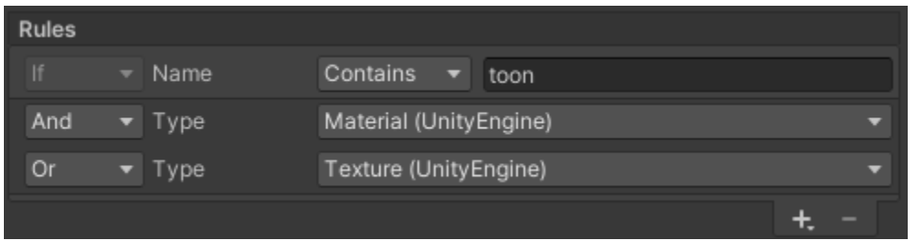
Examples

The following example demonstrates how to only allow assets whose name contain "toon" and are either a texture or a material.

The two rules for the Type are inside of a scope so that they are evaluated separately from the Name rule. Otherwise, it would be checking if an asset contained "toon" in it's name, and was a texture, and if either of those are false, it would then check if it was a material. Meaning that all materials could be added, even if they don't contain "toon" in their name.



Result:



Name Rule

Compares the file name of an asset with a string field using one of several matching options to determine if the asset's name matches the rule.

Type name	Description
Starts With	Matches if the start of the asset's name is the same as the full specified text.
Contains	Matches if the asset's name contains the full specified text within it.
Regex	Allows use of Regex to match against the asset's name. Matching if the asset's name matches the regex pattern.

Remarks
At least 2 characters must be provided in the rule's text field for it to work.

Type Rule

Compares the Object type of an asset with the specified type to determine if the asset matches the rule.

Component Rule

Checks that the Object is a prefab and checks if the top level GameObject has a component of the specified type.

Prefab Rule

Checks that the Object is a prefab of the specified type.

- Any: Matches if the object is a prefab of any type. Same as using a Type Rule set to GameObject
- Regular: Matches if the object is a prefab of a GameObject, but not a variant.
- Model: Matches if the object is a prefab of an imported model like a .fbx, or .obj.
- Variant: Matches if the object is a variant of another prefab.

Extension Rule

Compares the file extension of an asset with the specified string field to determine if the asset's file name matches the rule.

Label Rule

Compares the asset labels of an asset with the specified string field to determine if the asset contains a label matching the text in the field.