

# Recipes

User Guide

Online help: <https://github.com/h3b7/lowpoly-recipes-support/wiki>

FAQs: <https://github.com/h3b7/lowpoly-recipes-support/wiki/FAQ>

Ask a question: [\(GitHub\) lowpoly-recipes-support](#)

Request a feature: [\(GitHub\) lowpoly-recipes-support](#)

Report a bug: [\(GitHub\) lowpoly-recipes-support](#)

Support email: [support@h3b7.com](mailto:support@h3b7.com)

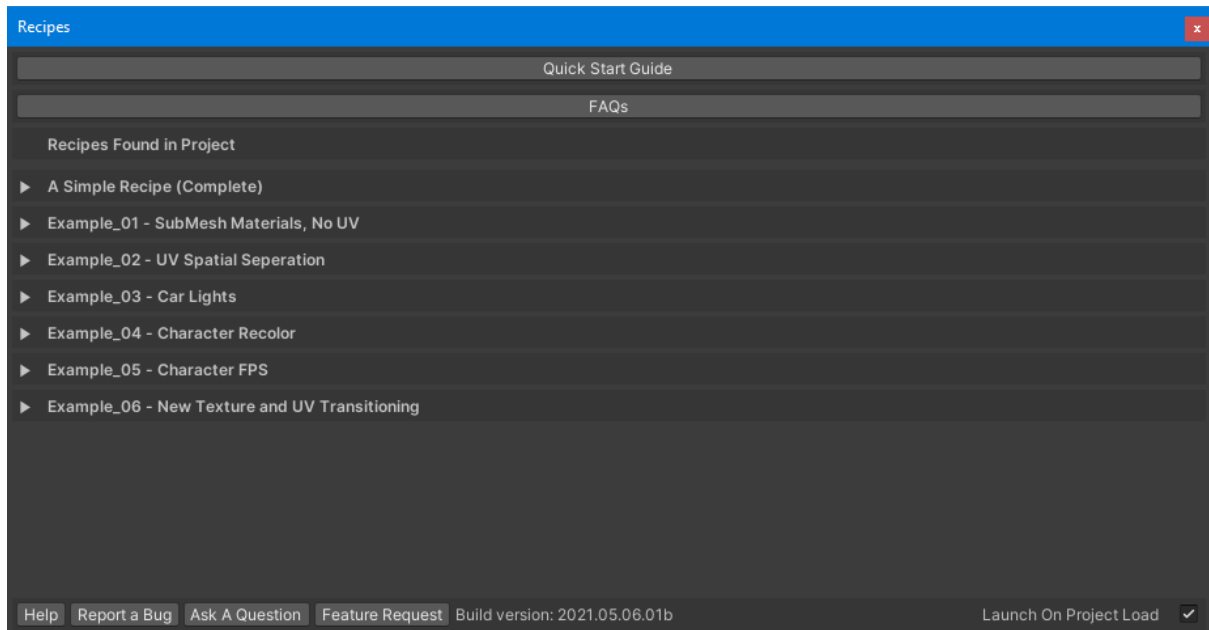
## Table of Contents

QuickStart .....	4
Package Overview .....	7
Package Structure: .....	7
H3B7/Menus .....	7
H3B7/Recipes .....	7
H3B7/Recipes/Demo.....	7
H3B7/Recipes/Icons .....	7
H3B7/Recipes/Script .....	7
H3B7/Recipes/Script/Attributes .....	7
H3B7/Recipes/Script/Common .....	7
H3B7/Recipes/Script/Core .....	8
H3B7/Recipes/Script/Editor .....	8
H3B7/Recipes/Script/GameObjects.....	8
H3B7/Recipes/Script/Instructions .....	9
H3B7/Recipes/Script/Menus .....	9
H3B7/Recipes/Script/Meshes .....	9
H3B7/Recipes/Script/Recipes .....	10
H3B7/Recipes/Script/Requirements.....	10
H3B7/Recipes/Script/Settings.....	10
H3B7/Recipes/Script/Support.....	10
H3B7/Recipes/Script/TextureBuilder.....	10
H3B7/Recipes/Script/TextureFeatures .....	10
H3B7/TextureInspector.....	10
How to Use This Package .....	11
How Does It Work? .....	11
How to Get Started? .....	11
Examples .....	12
Example 01 – Sub Mesh Materials, No UV .....	12
Example 02 - UV Spatial Separation.....	12
Example 03 - Car Lights .....	12
Example 04 - Character Recolor.....	12
Tools.....	13
Launcher.....	13
Texture Inspector .....	14



## QuickStart

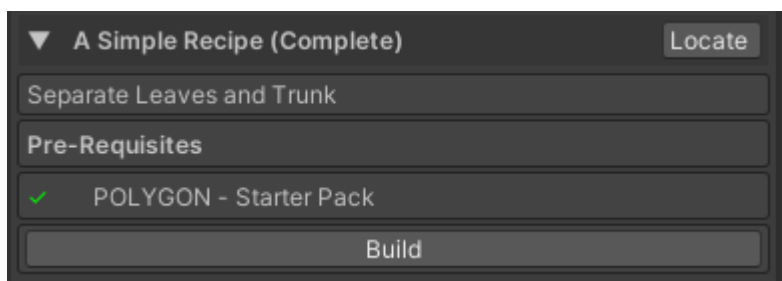
On installation you should be presented with the Launcher window



Expand A Simple Recipe entry...



We can see this Recipe requires a missing Asset Package. Click the Resolve button. This should take you to the Asset Store entry for this Third-Party Asset Package. After downloading and importing the Recipe's Pre-Requisites will be met.



Once the Pre-Requisites have been met, we can proceed to build the Recipe. Click the Build button.






The Recipe is built to the current Scene.

**Note.** At this point you may wish to add the original into the Scene for comparison:

 Assets/PolygonStarter/Prefabs/SM\_Generic\_Tree\_01.prefab

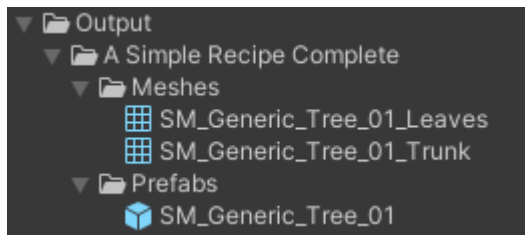
Now let us inspect what has been created.

▼  SM\_Generic\_Tree\_01  
     SM\_Generic\_Tree\_01\_Leaves  
     SM\_Generic\_Tree\_01\_Trunk

The original has been separated into two separate parts.

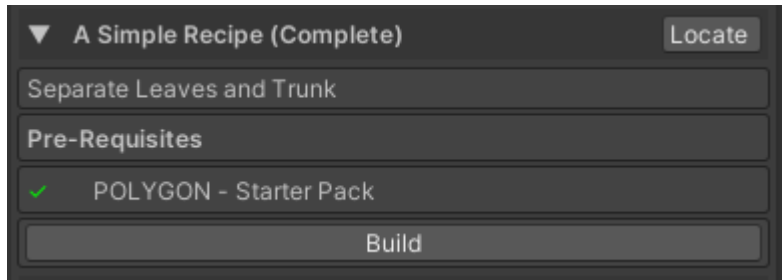


Navigate the Project window to the resulting files:

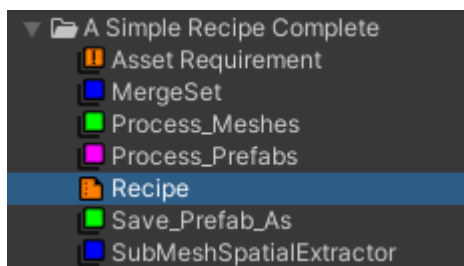


We can see a new prefab was created and two meshes with corresponding materials.

Look back at the Launcher.



Click the Locate Button to find the Recipe in the Project window.



The Recipe is composed of reusable Instructions.

- **Recipe:** The root container.
  - **Asset Requirement:** This checks for the existence of an Asset required by the Recipe.
  - **Process Prefabs:** Loads A source Prefab into the Scene and runs operation on it.
    - **Process Meshes:** Looks for Meshes referenced by the instantiated Prefab, runs Mesh Instructions on them and saves the result to the output location.
      - **SubMeshSpatialExtractor:** splits the mesh on spatial separation. In this case it separates based on vertex separation, but it can also separate on UVs.
      - **MergeSet:** This is used to rename the separated meshes.
    - **Save Prefab As:** Save the modified GameObject as a new Prefab in the output folder.

## Package Overview

### Package Structure:

#### H3B7/Menus

These are menus that provide helpful functions not necessarily related to the Recipes.

- **CopyHierarchyPath.cs:** Provides a menu on the Hierarchy context menu to “Copy Hierarchy Path”. This builds a path from the root object and adds that to the clipboard. Useful for Target Fields.
- **DeleteDeactivatedChildren.cs:** Provides a menu on the Hierarchy context menu to “Delete Deactivated Children”. This will delete any child GameObject that is currently deactivated.

#### H3B7/Recipes

This is where you will find the main package.

#### H3B7/Recipes/Demo

All demo assets are in this folder.

- **A Simple Recipe Complete:** A completed Recipe from the walkthrough.
- **Common:** Reusable Instructions.
- **Example\_XX:** See the Example section for more info on specific examples.
- **Scripts:** Contains scripts specific to the Demo Scene.
- **Getting Started:** A Demo Scene. This Provides access to the example Recipes.

#### H3B7/Recipes/Icons

Contains any Icons used in the package.

#### H3B7/Recipes/Script

The scripts sit within their own Assembly: H3B7.Recipes. This ensures they are for Editor use only. By their nature, many Instructions will not work outside the Editor and so need to be protected from accidental inclusion.

#### H3B7/Recipes/Script/Attributes

- **SectionAttribute.cs:** used by Elements (Recipes, Instructions, etc) to separate fields into collapsible groups – similar to the HeadingAttribute in Unity.
- **ToolAttribute.cs:** Adds a button or Drag-Drop target onto the Element’s Inspector window. Add to an instance method on the Element sub-class or to a static method on a public static class with the target sub-class as the first parameter. No Additional parameters will generate a button; one additional parameter will be a Drag-Drop target.

#### H3B7/Recipes/Script/Common

Contains Scripts used by all parts of asset.

- **Element.cs:** All Elements (Recipes, Instructions, etc) derive from this class. It provides base functionality and a common reference point for the Inspector GUI.
- **IAssetDatabase.cs:** An Interface to provide separation between the Instructions and Unity’s AssetDatabase.
- **IProgressBar.cs:** Provides separation between Instructions and the implementation of the progress bar.
- **PathProcessingOperations.cs:** An enumeration for how to treat name items.

- **StringMatcher.cs:** Helper class to process String matches. Supports wildcards at the start and/or end. Supports regex by prefixing with a '#'.

### H3B7/Recipes/Script/Core

This folder contains the core functionality and common base classes.

- **Blackboard.cs:** When a recipe is built a Blackboard is passed to the Instructions. This contains the data for Instructions to work on.
- **BlackboardExtensions.cs:** C# extension methods for the blackboard. The preference is wherever possible the core functionality of an instruction should be implemented in an extension, either on the Blackboard or relevant Type (GameObject, Transform etc).
- **BlackboardPrefabExtensions.cs:** C# extension methods for the blackboard related to Prefabs.
- **Element.cs:** The base class for Recipes, Instructions, etc. This derives from ScriptableObject.
- **Instruction.cs:** Base class for Instructions.
- **MeshBlackboard.cs:** A blackboard for Mesh operations.
- **MeshBlackboardExtensions.cs:** C# extension methods for the Mesh Blackboard.
- **MeshExtractor.cs:** A base class for Mesh Processors that extract Sub-Meshes.
- **MeshProcessor.cs:** Base class for Mesh Instructions.
- **RecipeBase.cs:** Base class for Recipes. This lets us add specialised Recipes.
- **RecipesSettings.cs:** ScriptableObject for holding settings. The settings are stored in Recipes/Script/Settings/.
- **Requirement.cs:** Base class for Recipe Requirements.

### H3B7/Recipes/Script/Editor

Scripts that are Editor related.

**Note:** The entire Script folder is covered by an Editor only Assembly.

- **AssetHelper.cs:** Implementation of IAssetDatabase.
- **CommonEditor.cs** The Inspector GUI common to all Elements.
- **EditorGroup.cs:** Disposable wrapper for calls requiring end methods such as EditorGUILayout.EndVertical.
- **H3B7EditorUtilities.cs:** GUI Helper methods
- **ProgressBar.cs:** Disposable wrapper for Unity's progress bar. Implements IProgressBar.
- **RecipesOnLoad.cs:** Helper to check if the Recipe Launcher should be opened.
- **RecipesWindow.cs:** The Recipe Launcher.
- **ToolDiscovery.cs:** Used by the Inspector to find ToolAttributes.

### H3B7/Recipes/Script/GameObjects

This folder contains the Instructions that require a target GameObject, usually instantiated via InstructionProcessPrefabs.

- **GameObjectBuildAvatar**
- **GameObjectDebug**
- **GameObjectPopulateMeshPlaceholderSet**
- **GameObjectPositionInLine**
- **GameObjectProcessMesh**
- **GameObjectProcessMeshes**



- **GameObjectPurgeInactive**
- **GameObjectRemoveChild**
- **GameObjectRemoveComponent**
- **GameObjectRenameTarget**
- **GameObjectSaveAsPrefab**
- **GameObjectSetMesh**
- **GameObjectSetScaleToOne**
- **GameObjectSetTarget**
- **GameObjectStartPosition**
- **GameObjectTransform**

#### [H3B7/Recipes/Script/Instructions](#)

This folder contains general Instructions.

- **InstructionCreateTexture**
- **InstructionMaterialMapping**
- **InstructionModifyMaterial**
- **InstructionProcessPrefabs**
- **InstructionProcessPrefabsAtPath**
- **InstructionSet**
- **InstructionTextureImportSetting**

#### [H3B7/Recipes/Script/Menus](#)

This Folder Contains any menu specific to Recipes.

- **CreateUVExtractorFromSelectedMenu**

#### [H3B7/Recipes/Script/Meshes](#)

This folder contains Mesh Processors. They are called by **GameObjectProcessMeshes** to process any meshes referenced by the Target **GameObject** or its children.

- **MeshAxisExtractor**
- **MeshBoneWeightExtractor**
- **MeshConditional**
- **MeshConditionalSet**
- **MeshDiagnosticProcessor**
- **MeshLeftoverExtractor**
- **MeshMergeExtractor**
- **MeshMergeSet**
- **MeshPlaceholderSet**
- **MeshRemoveDataChannels**
- **MeshRename**
- **MeshSet**
- **MeshSpatialExtractor**
- **MeshTextureFeatureExtractor**
- **MeshTransform**
- **MeshUVExtractor**
- **MeshUVRemap**

### H3B7/Recipes/Script/Recipes

This folder contains any Recipe types that can be created.

- **Recipe**

### H3B7/Recipes/Script/Requirements

This folder contains any Requirement types that can be created.

- **AssetRequirement.cs**

### H3B7/Recipes/Script/Settings

This folder contains the settings file.

- **Settings.asset**: The current setting for the Recipe Tool

### H3B7/Recipes/Script/Support

This Folder contains support classes with extension / helper methods. It is broken down into separate classes for each supported type.

### H3B7/Recipes/Script/TextureBuilder

This folder contains classes used by InstructionCreateTexture to create new Textures.

- **ColorFeature.cs**: Blocks of color. Specify Size, Colors, Orientation.
- **ColorFeatureOrientations.cs**: The orientation of Color Features: Vertical, Horizontal etc.
- **ComplexFeature.cs**: A collection of other Features pre-arranged relative to a common origin.
- **Feature.cs**: Base class for Features.
- **FeaturePacker.cs**: Helper class to pack features into the texture.
- **ImageFeature.cs**: A Feature containing a reference to a Region in an existing Texture. Used for amalgamating Textures.

### H3B7/Recipes/Script/TextureFeatures

This folder contains the TextureFeatureSet (TFS) functionality. It defines Features within a Texture for use in extracting Sub-Meshes or Mapping UVs to another Texture.

- **TextureFeature.cs**: The description of a Feature: position, size, name etc.
- **TextureFeatureSet.cs**: The ScriptableObject.
- **TextureFeatureSetTools.cs**: Additional tools for TFSs. Create from Selection of Sprites.

### H3B7/TextureInspector

- **Settings.asset**: The stored settings for this tool.
- **Texture2DExtensions.cs**: C# Extension methods for Texture2Ds
- **TextureInspectorSettings.cs**: ScriptableObject that holds the settings for the tool.
- **TextureInspectorWindow.cs**: Texture Inspector Tool. See section below.

## How to Use This Package

### How Does It Work?

Recipes revolve around a Recipe Object. When you build a Recipe, a Blackboard is created which is then passed to the Recipe's Instructions. These Instructions can Instantiate Prefabs, make modifications to the GameObjects, Meshes, Materials etc and then save as files separate from the original.

### How to Get Started?

Start by building the example recipes. Then follow the Simple Recipe instructions.

## Examples

### Example 01 – Sub Mesh Materials, No UV

Converts Some trees to use Sub Meshes with separate materials. Uses specific UV regions to separate the Meshes. Strips out the UV channel as it is no longer needed.

### Example 02 - UV Spatial Separation

Produces the same output as Example\_01 but uses UV spatial separation followed by mesh specific merging.

### Example 03 - Car Lights

Separates out the Lights from a car model.

### Example 04 - Character Recolor

1. Separates skin, eyes, lips and hair into Sub Meshes
2. Creates separate materials.
3. Creates Prefabs

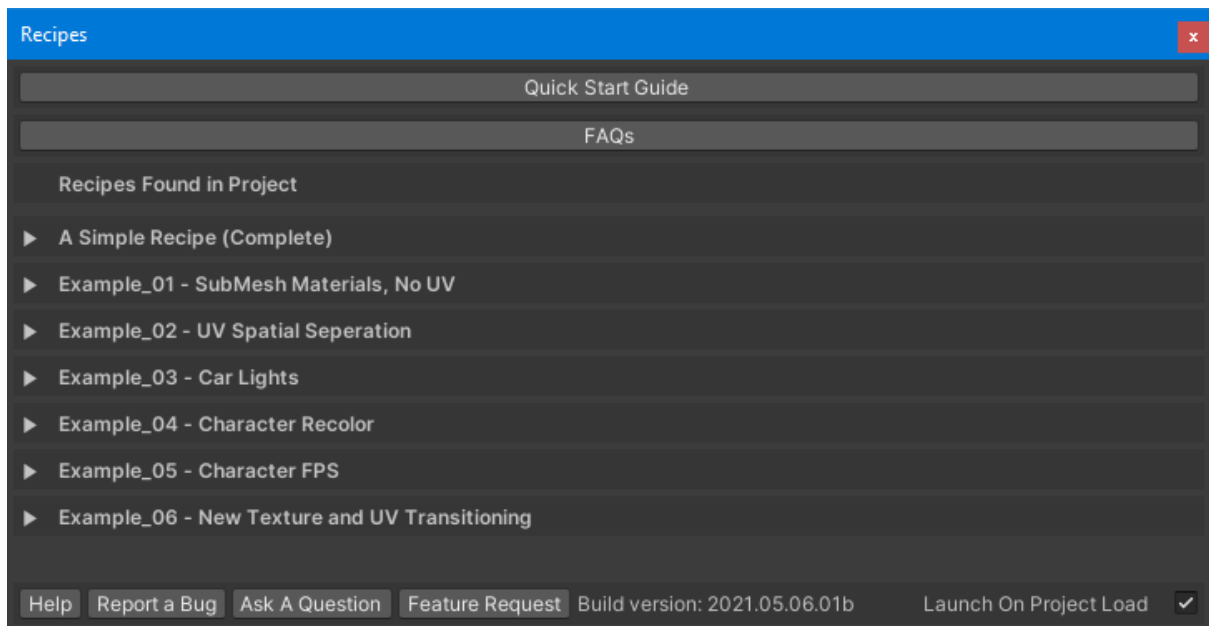
### Example\_05 - Character FPS

Separates the arms and head.  
Also re-colors the skin.

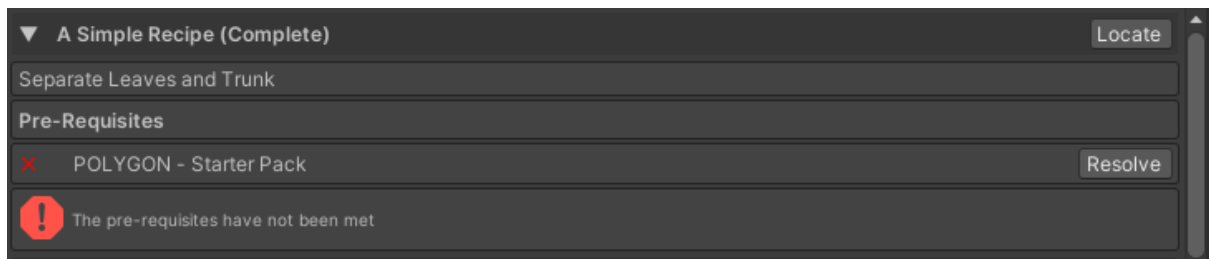
## Tools

### Launcher

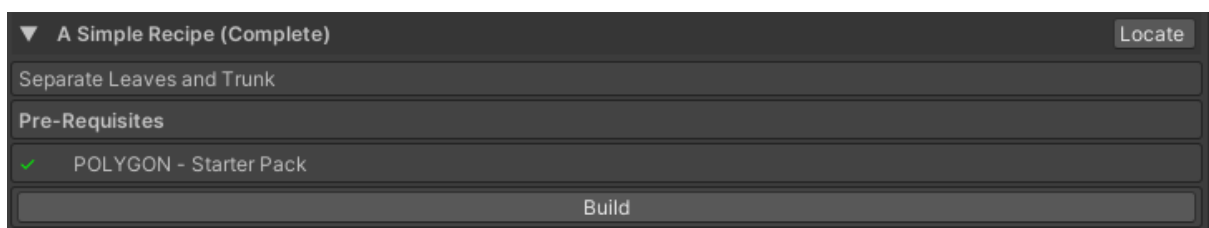
The recipe Launcher list all Recipes within the current Project.



Left click on an entry to expand its details:



If a Pre-Requisite (Requirement) of the Recipe has not been met, then the build option will not be available. Click the Resolve button to fix the issue. In this case a required Asset is not present and needs importing.



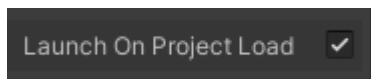
After all Pre-Requisites have been met, the Build option becomes available. This will Build the Recipe.

**Note.** You can also build the Recipe from a Tool on Recipe's Inspector GUI.

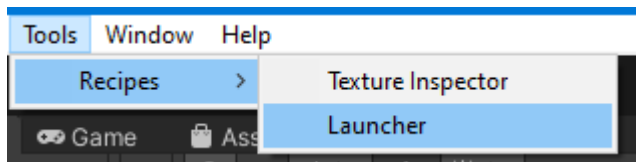
The Locate button in the top right will ping the Recipe in the Project window.



The Help buttons at the bottom link to the online help and issue reporting. When reporting an issue please add the Build version (to the right of the buttons).

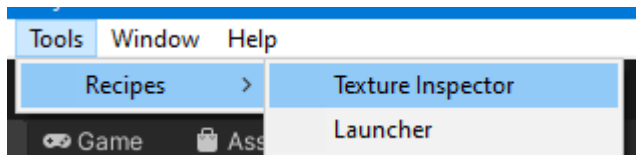


In the bottom right is an option to disable launching on load. If disabled, you can manually launch via the Tools menu:

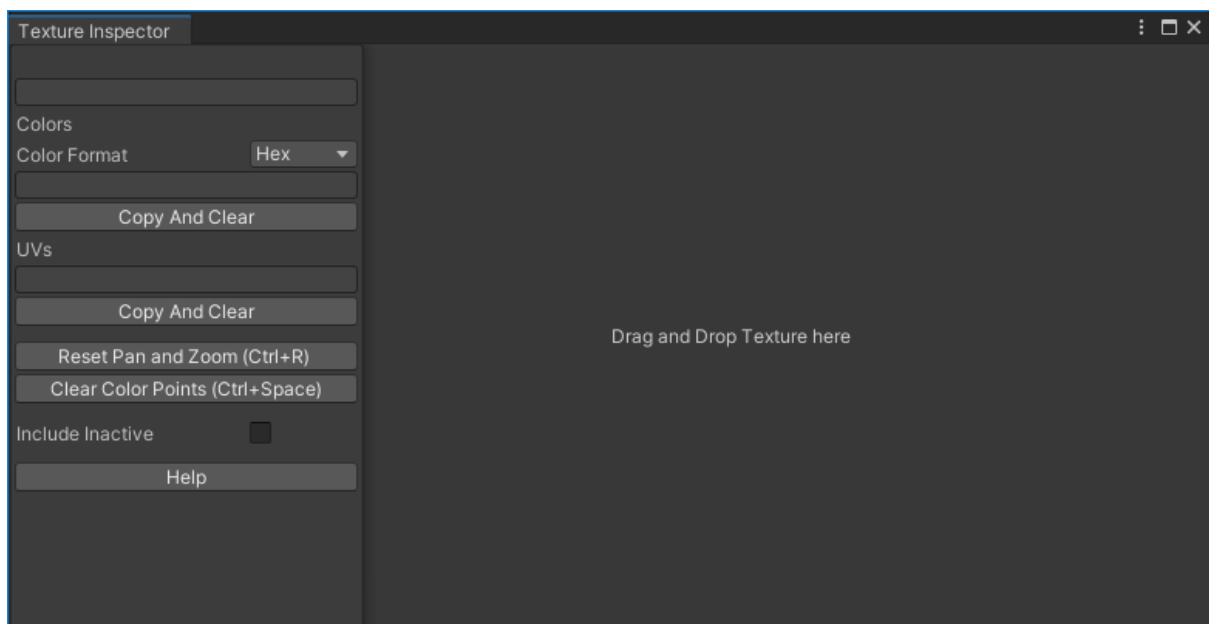


## Texture Inspector

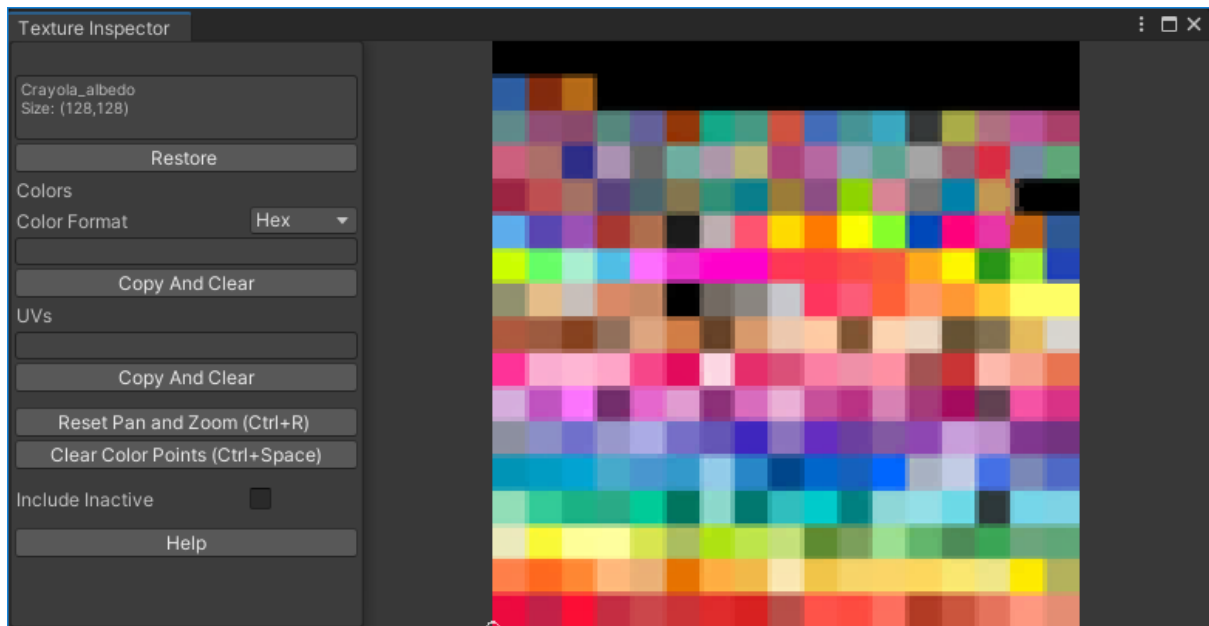
The Texture Inspector tool is used to quickly visualise a texture and see how a Model's UV channel maps onto the Texture. Also provides tools for inspecting color and UV coordinates.



Launch the Texture Inspector via the Tools menu.



To start Drag and Drop a texture onto the window.

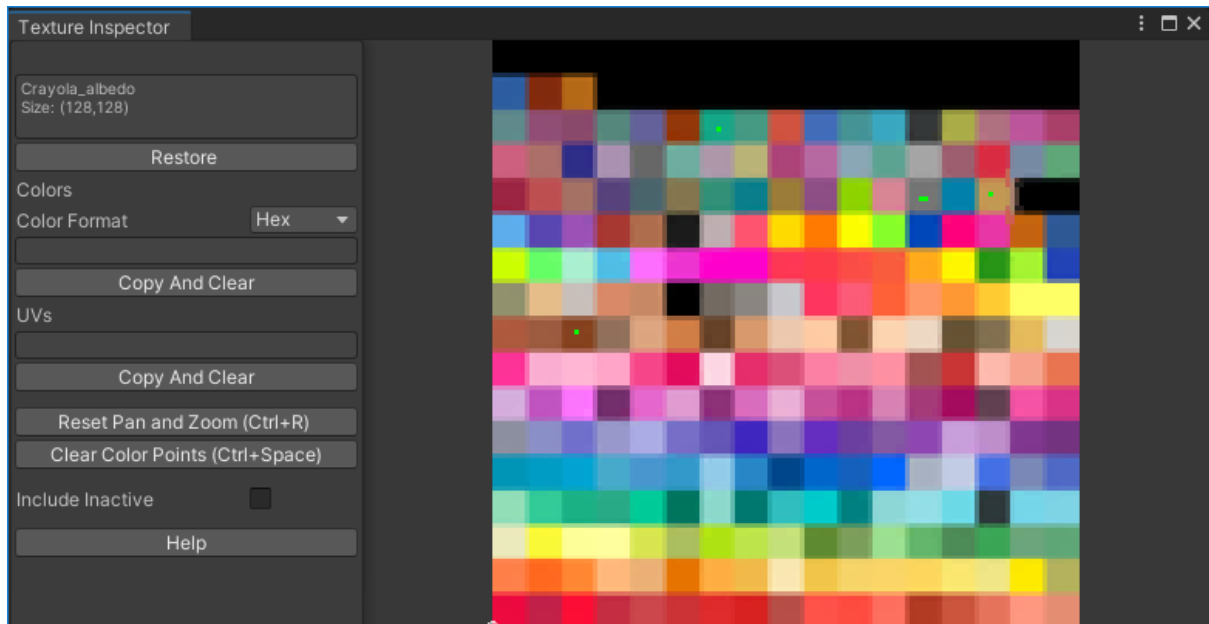


The Texture will now be displayed.

**Note.** The texture will be affected by any Import settings you may have, such as Filter Mode and Compression.

Controls: Centre Click & Drag the mouse to Pan the texture. Scroll Wheel to Zoom.

Now Drag and Drop a Mesh, Prefab or GameObject onto the Window

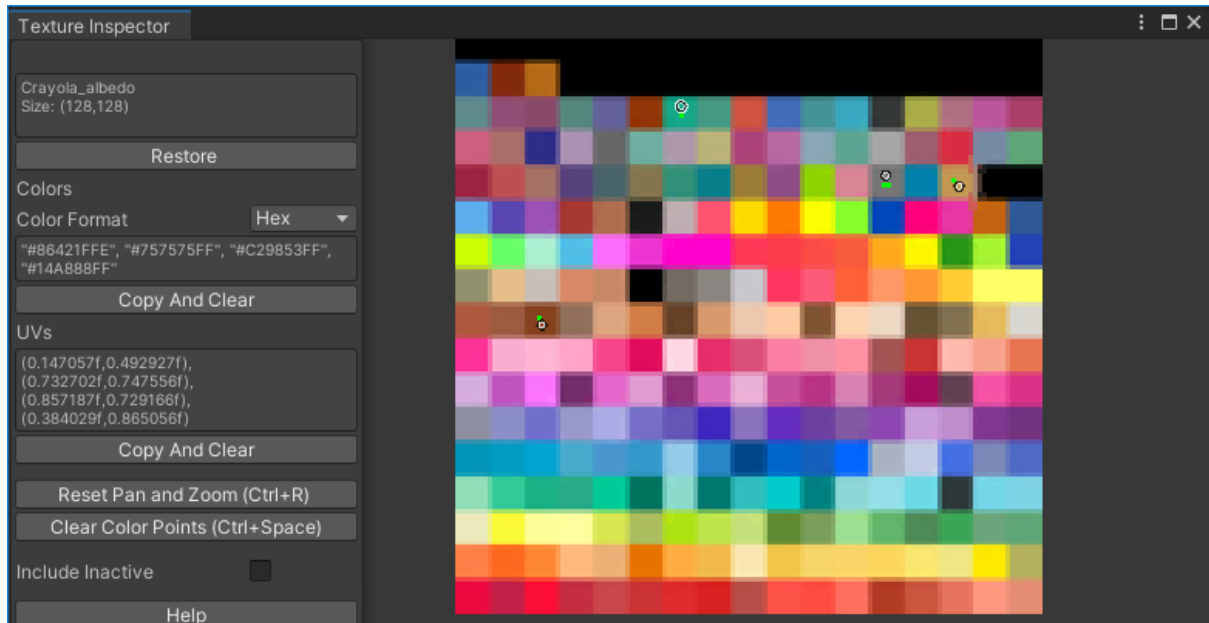


The UVs will be mapped out in green. The Restore button will reset the display, removing the green UVs.

**Note.** The colors are stored in the Settings file:  
`Assets/H3B7/TextureInspector/Settings.asset`

Left click to add Selection Points. Selection points let you inspect colors and UV coordinates.

**Note.** The last point selected will be a larger symbol and will remain after clearing. This is to allow you to keep track when copying points.



GUI Layout:

- **Texture Info:** The current texture and its size
- **Restore:** Button to restore the current texture (remove UV overlays)
- **Color Format:** Output format for the 'Copy and Clear' button. Hex: "#707C84FF" Float: (0.3529412f, 0.3764706f, 0.4f,1f)
- **Color Data:** Displays the color for the selected points
- **Copy And Clear:** Copies the content of the Color Data to the clipboard and clears the points.
- **UV Data:** Displays the UV coordinates for the selected points
- **Copy And Clear:** Copies the content of the UV Data to the clipboard and clears the points.
- **Reset Pan and Zoom:** Restores Pan and Zoom settings.
- **Clear Color Points:** clears the selected points.
- **Help:** Opens the online help page.