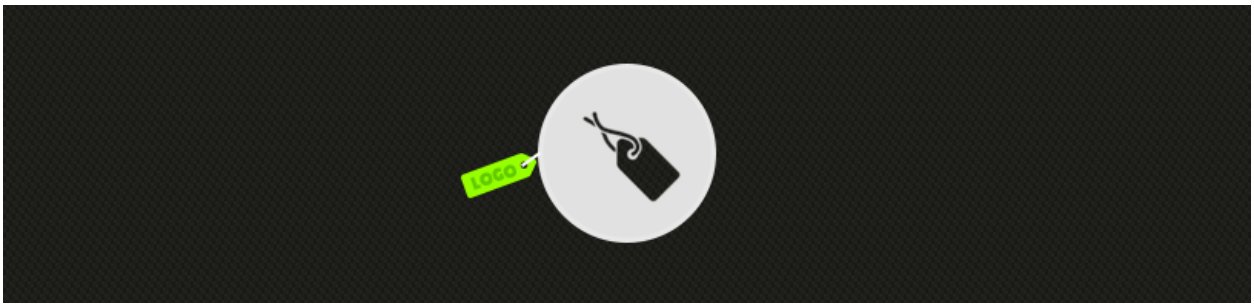
Pigeon Coop Toolkit – All The Tags!



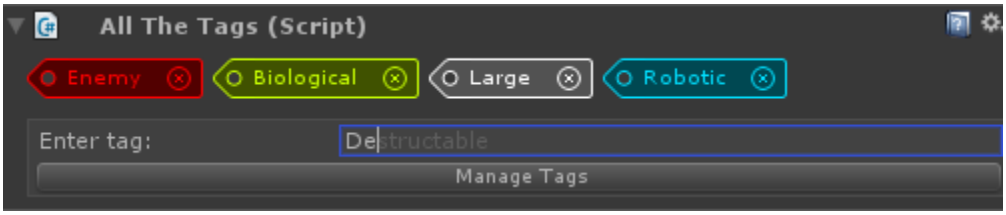
All The Tags! Replaces the built in tagging system in Unity3D. It's sole purpose is to allow you to add more than one tag to any object, however there other benefits too! It performs substantially better than doing a standard string compare, generates no garbage and allows you to customize your tags with colors to stay organized!

Contents

Components

All The Tags

Find it under Component → Pigeon Coop Toolkit → ***All The Tags!***

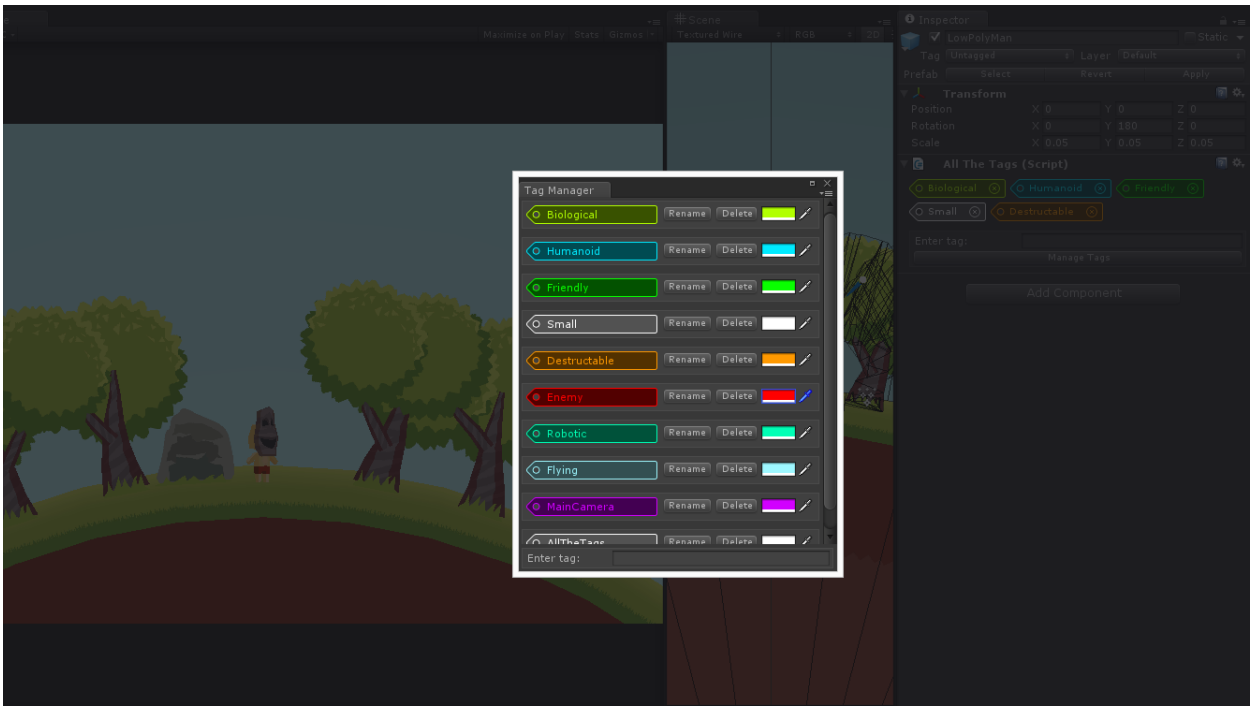


If you want to add some tags to an object using All The Tags!, attach this component to it. Through it, you can attach tags to your Game Object. Enter a tag into the text field and press enter to add the tag. To accept the 'autocomplete' option, press TAB. Remember, tags are case sensitive! ("Large" != "large").

If a tag does not exist, it is added to your Tag Manager automatically.

Preview Utility

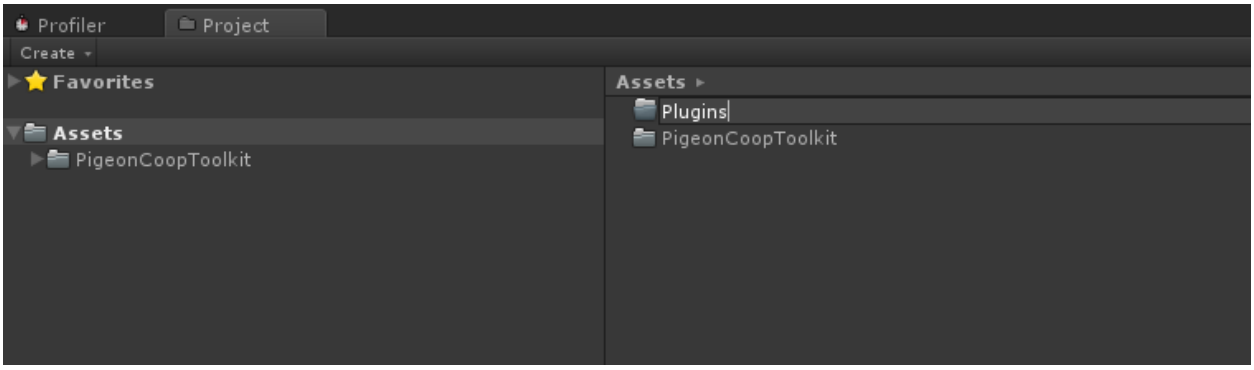
Find it under Window → Pigeon Coop Toolkit → All The Tags! → ***Tag Manager***



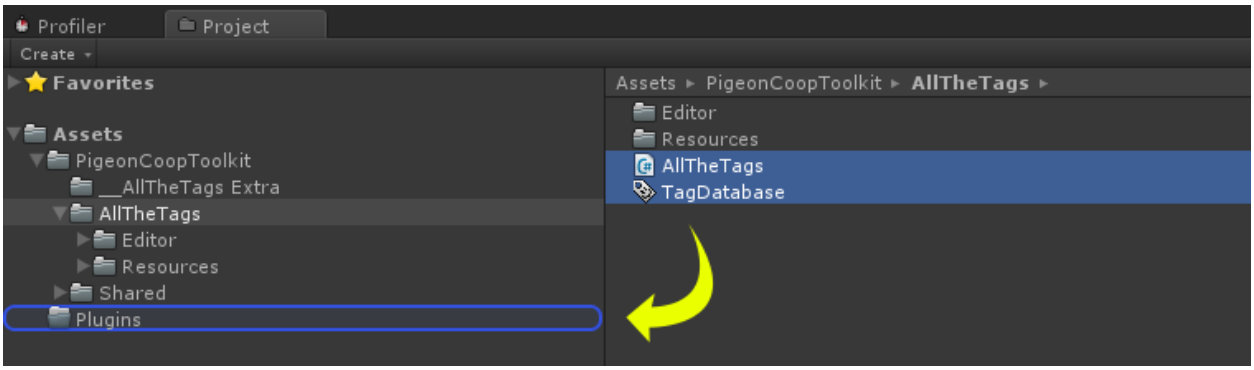
From here you can change the color of your tags and rename the tags. Renaming a tag will update the tag on every object with that tag automatically. Removing a tag is permanent, so be careful when using the 'delete' button!

Javascript Usage

You can use All The Tags with Javascript relatively easily. You will have to move some files around though, as Unity will not allow you to access C# functions from Javascript unless the C# files are located in a specific folder.



Create a ***Plugins*** folder in the ***root of your project***. The location and name is important.



Move the “AllTheTags” script and the “TagDatabase” script into your plugins folder. You can now access All The Tags from Javascript! Here is a simple example script:

```
#pragma strict
import PigeonCoopToolkit.AllTheTags;

function Start () {
    Debug.Log( Tags.HasTag(gameObject,"TestTag") );
}
```

Code Reference

```
bool Tags.HasTag(GameObject tag, string tag)
```

Checks if the GameObject has the tag attached. Tags are case sensitive.

```
void Tags.AddTag(GameObject tag, string tag)
```

Adds the tag to a GameObject. You can only add tags that have already been defined in the Tag Manager if you use this function at run time. (You can't "define" new tags on the fly). Tags are case sensitive.

```
void Tags.RemoveTag(GameObject tag, string tag)
```

Removes a tag from a GameObject. Tags are case sensitive.

```
ReadOnlyCollection<GameObject> Tags.FindGameObjectsWithTag(string tag)
```

Finds all GameObjects with tag tag in the current scene. Returns both active and inactive objects. Never returns *null*.

```
List<GameObject> Tags.FindGameObjectsWithTagAroundPoint(string tag, Vector3
worldPoint, float distance)
```

Finds all GameObjects with tag tag in the current scene within distance units away from worldPoint. Returns both active and inactive objects. Never returns *null*.

```
List<GameObject> FindChildrenWithTag(GameObject go, string tag)
```

Finds all children of target GameObject with a certain tag. Includes the target GameObject too. Returns both active and inactive objects. Never returns *null*.

```
List<GameObject> FindChildrenWithTag(GameObject go, string tag, bool
includeInactive)
```

Finds all children of target GameObject with a certain tag. Includes the target GameObject too. Never returns *null*.