

# Scripting

Scripting in Unity uses C# and is very well documented. In the sections below, I'll provide examples and edge cases where possible, and link to the relative documentation for quick reference.

For a collection of classes and structs that are *required* for Unity to function, which means they will *always* be available to you when scripting in Unity, head over to [UnityEngine.CoreModule Documentation](#)

## Transform

This class controls and tracks object position, rotation, scale.

[Official Transform Class Documentation](#)

### Local Space

Local space is the transform relative to the object's parent. An example of this can be seen below where I have selected an object and the transform controls are centralized on the exact transform of that object relative to its local position.



Take notice of three things in the above screenshot. First, we have selected **Local** position in the top-left near our transform controls. Clicking this button again will toggle between Local and World space. Second, take note of the World Space axis shown at the top-right of the scene view. Third, in contrast to the World space axis, notice the GameObject's axis shown in the scene view are different in orientation. The transform axis shown on the GameObject are modifying and referring to the GameObject's transform within Local space.

### World Space

World space is the position of the GameObject rooted within the scene. An example of this is seen in selecting the exact same object in the editor and toggling world space transform view. This makes sure the transform controls are the same as the World Space axis, instead of referring directly to the transform of a local object.



Again, take notice of three things in the above screenshot. First, we have selected `Global` position in the top-left near our transform controls. Clicking this button again will toggle between Local and World space. Second, take note of the World Space axis shown at the top-right of the scene view. Third, in contrast to the World space axis, notice the GameObject's axis shown in the scene view are different in orientation. The transform axis shown on the GameObject are modifying and referring to the GameObject's transform within World space.

# Vector3

[Official Vector3 Struct Documentation](#)

## Axis

In Unity 3D you will use the `X`, `Y`, and `Z` axis frequently both in positioning within the editor and scripting. It helps to have a clear understanding of the names these axis can be referred to with, as it will greatly improve your ability to access and modify these values without over complicating things.



The `X` axis can be accessed with the `right` keyword when accessing any class which stores axis information

The `Y` axis can be accessed with the `Up` keyword when accessing any class which stores axis information

The `Z` axis can be accessed with the `forward` keyword when accessing any class which stores axis information

Similarly, when modifying a `Vector`, we can easily flip these axis by accessing the opposite of these keywords -

The `X` negative axis can be accessed with the `left` keyword when accessing any class which stores axis information

The `Y` negative axis can be accessed with the `down` keyword when accessing any class which stores axis information

The `Z` negative axis can be accessed with the `back` keyword when accessing any class which stores axis information

So, the `Vector3` equivalents to these would be

`X` axis, `Vector3(1.0f, 0.0f, 0.0f)`, is equivalent to `right`

`Y` axis, `Vector3(0.0f, 1.0f, 0.0f)`, is equivalent to `up`

Z axis, `Vector3(0.0f, 0.0f, 1.0f)`, is equivalent to `forward`

X negative axis, `Vector3(-1.0f, 0.0f, 0.0f)`, is equivalent to `left`

Y negative axis, `Vector3(0.0f, -1.0f, 0.0f)`, is equivalent to `down`

Z negative axis, `Vector3(0.0f, 0.0f, -1.0f)`, is equivalent to `back`

# Quaternion

## Official Quaternion Struct Documentation

---

Revision #5

Created 7 June 2020 16:40:51 by Shaun Reed

Updated 8 June 2020 15:34:23 by Shaun Reed