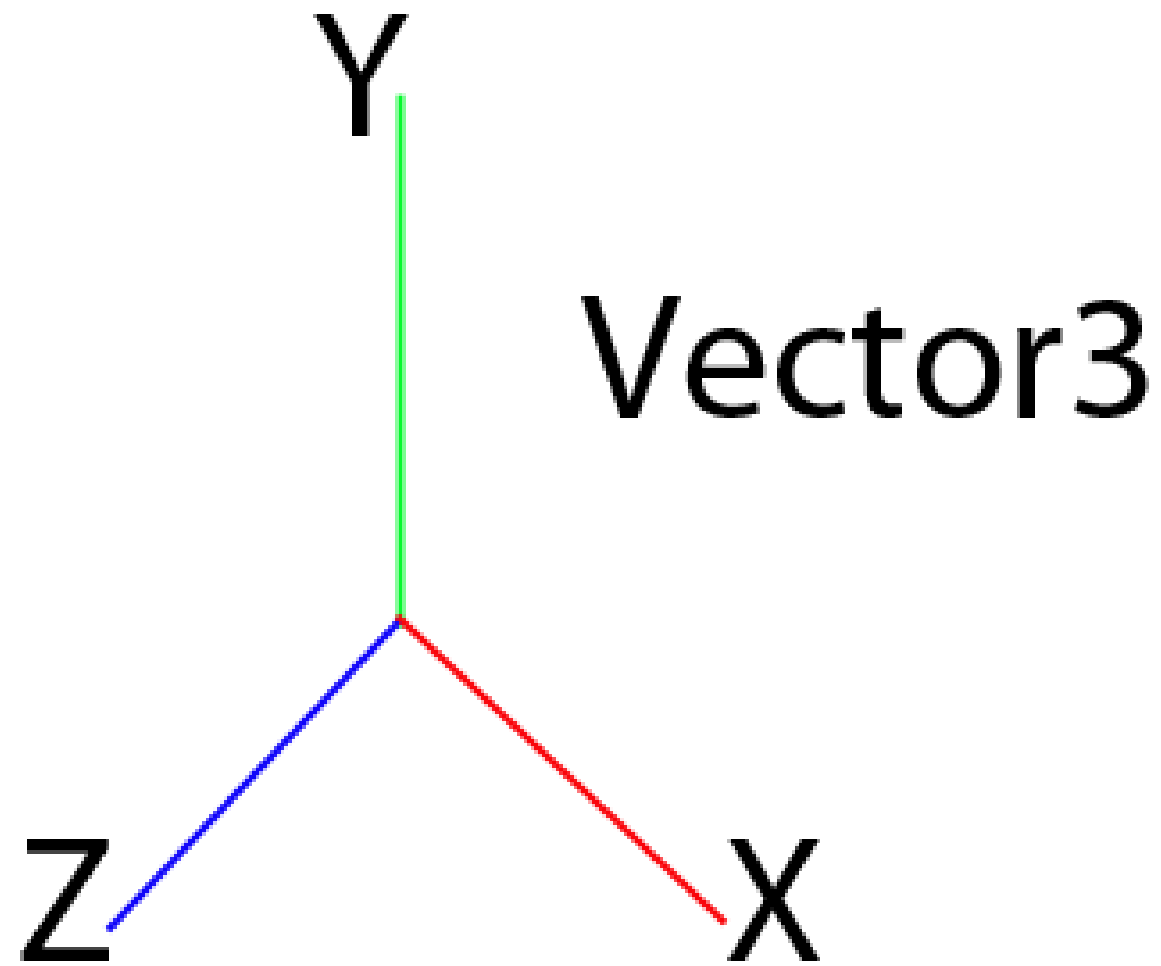




Vector3

<http://docs.unity3d.com/ScriptReference/Vector3.html>





Vector3

struct in UnityEngine

Description

Representation of 3D vectors and points.

This structure is used throughout Unity to pass 3D positions and directions around. It also contains functions for doing common vector operations.

```
Vector3 position = new Vector3(0f, 0f, 1f);
```

Transform



The image shows a screenshot of the Unity Transform component inspector. It has a grey background and a dropdown arrow on the left. The title 'Transform' is in bold. There are three sections: Position, Rotation, and Scale. Each section has three input fields for X, Y, and Z values. The Position section shows X: 12.47817, Y: 5.677895, Z: -16.50604. The Rotation section shows X: -39.401, Y: 83.193, Z: -106.283. The Scale section shows X: 1, Y: 1, Z: 1. There are help and settings icons in the top right corner.

Property	X	Y	Z
Position	12.47817	5.677895	-16.50604
Rotation	-39.401	83.193	-106.283
Scale	1	1	1

Transform.localPosition

SWITCH TO MANUAL

public [Vector3](#) localPosition;

Transform.position

SWITCH TO MANUAL

public [Vector3](#) position;

Transform.localScale

SWITCH TO MANUAL

public [Vector3](#) localScale;

Transform.eulerAngles

SWITCH TO MANUAL

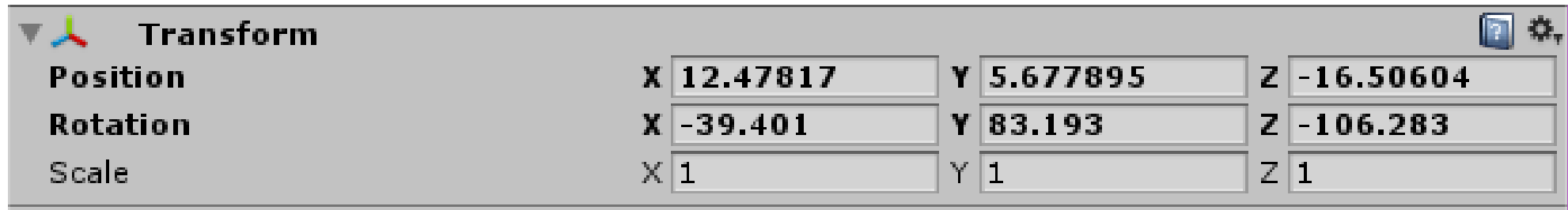
public [Vector3](#).eulerAngles;

Transform.localEulerAngles

SWITCH TO MANUAL

public [Vector3](#) localEulerAngles;

Transform



- Access "Transform" component via: "transform"
- Ref: docs.unity3d.com/ScriptReference/Transform.html

```
Vector3 myVector = new Vector3(1, -2, 10);  
Vector3 zeroVector = Vector3.zero; // (0, 0, 0)  
Vector3 oneVector = Vector3.one; // (1, 1, 1)  
Vector3 bigVector = Vector3.one * 10; // (10, 10, 10)  
Vector3 sumVector = bigVector + myVector; // (11, 8, 20)
```

<u>back</u>	Shorthand for writing Vector3(0, 0, -1).
<u>down</u>	Shorthand for writing Vector3(0, -1, 0).
<u>forward</u>	Shorthand for writing Vector3(0, 0, 1).
<u>left</u>	Shorthand for writing Vector3(-1, 0, 0).
<u>one</u>	Shorthand for writing Vector3(1, 1, 1).
<u>right</u>	Shorthand for writing Vector3(1, 0, 0).
<u>up</u>	Shorthand for writing Vector3(0, 1, 0).
<u>zero</u>	Shorthand for writing Vector3(0, 0, 0).

<https://docs.unity3d.com/ScriptReference/Vector3.html>

Random

<https://docs.unity3d.com/ScriptReference/Random.html>

Random.Range

public static float **Range**(float min, float max);

Parameters

Description

Returns a random float number between and min [inclusive] and max [inclusive] (Read Only).

Note that max is inclusive, so using Random.Range(0.0f, 1.0f) could return 1.0 as a value.

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour
{
    public GameObject prefab;

    // Instantiate the prefab somewhere between -10.0 and 10.0 on the x-z plane
    void Start()
    {
        Vector3 position = new Vector3(Random.Range(-10.0f, 10.0f), 0, Random.Range(-10.0f, 10.0f));
        Instantiate(prefab, position, Quaternion.identity);
    }
}
```

Random.rotationUniform

public static [Quaternion](#) rotationUniform;

Description

Returns a random rotation with uniform distribution (Read Only).

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    void Example() {
        transform.rotation = Random.rotationUniform;
    }
}
```



Random.ColorHSV

public static [Color](#) ColorHSV();

public static [Color](#) ColorHSV(float hueMin, float hueMax);

public static [Color](#) ColorHSV(float hueMin, float hueMax, float saturationMin, float saturationMax);

public static [Color](#) ColorHSV(float hueMin, float hueMax, float saturationMin, float saturationMax, float valueMin, float valueMax);

public static [Color](#) ColorHSV(float hueMin, float hueMax, float saturationMin, float saturationMax, float valueMin, float valueMax, float alphaMin, float alphaMax);

<http://alloyui.com/examples/color-picker/hsv/>



Distance



Vector3.Distance

public static float **Distance**([Vector3](#) a, [Vector3](#) b);

Parameters

Description

Returns the distance between a and b.

`Vector3.Distance(a,b)` is the same as `(a-b).magnitude`.

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    public Transform other;
    void Example() {
        if (other) {
            float dist = Vector3.Distance(other.position, transform.position);
            print("Distance to other: " + dist);
        }
    }
}
```



Accessing Components

Components On The Same Object

```
public class LightColorSwitcher : MonoBehaviour {  
  
    private Light LightComponent;  
  
    // Use this for initialization  
    void Start () {  
        LightComponent = GetComponent<Light>();  
    }  
  
    // Update is called once per frame  
    void Update () {  
  
    }  
}
```

Generic Method

```
LightComponent = GetComponent<Light>();
```



TYPE OF
COMPONENT

Components On Other Objects

(Inspector Method)

```
public class Script04_Distance : MonoBehaviour {  
  
    public Transform PlayerTransform;  
  
    // Use this for initialization  
    void Start () {  
  
    }  
  
    // Update is called once per frame  
    void Update () {  
  
    }  
}
```



Components On Other Objects

(Scripting Method)

```
public class Script04_Distance : MonoBehaviour {  
  
    private Transform PlayerTransform;  
  
    // Use this for initialization  
    void Start () {  
  
        GameObject player = GameObject.Find("RigidBodyFPSController");  
        PlayerTransform = player.transform;  
  
    }  
  
    // Update is called once per frame  
    void Update () {  
  
    }  
  
}
```

```
public class DistanceDemo : MonoBehaviour {  
  
    public Transform PlayerTransform;  
  
    void Update () {  
        // Find the distance  
        float distance = Vector3.Distance(PlayerTransform.position, transform.position);  
  
        // Check how this object is to the player  
        if (distance <= 3f) {  
            Debug.Log("Player is close!");  
        } else {  
            Debug.Log("Player is far!");  
        }  
    }  
}
```

Color

<http://docs.unity3d.com/ScriptReference/Color.html>



Color

struct in UnityEngine

Description

Representation of RGBA colors.

This structure is used throughout Unity to pass colors around. Each color component is a floating point value with a range from 0 to 1.

Components (r,g,b) define a color in RGB color space. Alpha component (a) defines transparency - alpha of one is completely opaque, alpha of zero is completely transparent.

Color Constructor

`public Color(float r, float g, float b, float a);`

Parameters

r	Red component.
g	Green component.
b	Blue component.
a	Alpha component.

Description

Constructs a new Color with given r,g,b,a components.

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    public Color color = new Color(0.2F, 0.3F, 0.4F, 0.5F);
}
```

Color.Lerp

public static [Color](#) Lerp([Color](#) a, [Color](#) b, float t);

Parameters

a	Color a
b	Color b
t	Float for combining a and b

Description

Linearly interpolates between colors a and b by t.

t is clamped between 0 and 1. When t is 0 returns a. When t is 1 returns b.

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    public Color lerpedColor = Color.white;
    void Update() {
        lerpedColor = Color.Lerp(Color.white, Color.black, Mathf.PingPong(Time.time, 1));
    }
}
```

Mathf

<http://docs.unity3d.com/ScriptReference/Mathf.html>

Mathf.Repeat

public static float **Repeat**(float **t**, float **length**);

Parameters

Description

Loops the value **t**, so that it is never larger than **length** and never smaller than 0.

This is similar to the modulo operator but it works with floating point numbers. For example, using 3.0 for **t** and 2.5 for **length**, the result would be 0.5. With **t** = 5 and **length** = 2.5, the result would be 0.0. Note, however, that the behaviour is not defined for negative numbers as it is for the modulo operator.

In the example below the value of time is restricted between 0.0 and just under 3.0. This is then used to keep the x position in this range.

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    void Update() {
        transform.position = new Vector3(Mathf.Repeat(Time.time, 3), transform.position.y, transform.position.z);
    }
}
```

Mathf.PingPong

public static float **PingPong**(float **t**, float **length**);

Parameters

Description

PingPongs the value t, so that it is never larger than length and never smaller than 0.

The returned value will move back and forth between 0 and length.

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    void Update() {
        transform.position = new Vector3(Mathf.PingPong(Time.time, 3), transform.position.y, transform.position.z);
    }
}
```