**GAMEDEVREJECTS:**

**C# PROG IN UNITY : THE TRANSFORM COMPONENT**

**Orlando Unity3d Development Group**

**Date: 24-Oct-21**

**All Copyright GAMEDEVREJECTS©2021**

**Contents**

[**3.8.0** **The Transform Component** 3](#_Toc85963143)

[**3.8.1** **Create a new Project** 3](#_Toc85963144)

[**3.8.2** **Transform.position** 5](#_Toc85963145)

[**3.8.3** **Transform.translate** 5](#_Toc85963146)

[**3.8.4** **Transform.rotate** 7](#_Toc85963147)

[**3.8.5** **Transform.localScale** 8](#_Toc85963148)

[**Glossary** 10](#_Toc85963149)

[**Resources** 11](#_Toc85963150)

# **3.8.0 The Transform Component**

Here we will learn about the default **Transform** component attached to every game object in the scene window.

|  |
| --- |
| **View/Navigate to the Transform Component** |
| (Navigate to)[HIERARCHY tab]> (Select)[ CAPSULE](The gameObject in our scene) > VIEW the CAPSULE TRANSFORM component (in the INSPECTOR tab) |

The Transform component is a container, which holds the **1)Position 2)Rotation & 3)Scale** for the “selected” gameObject in the **Scene Window**.

The **transform.position** => defines the gameObjects location (i.e its x,y and z co-ordinate values) in 2D or 3D game space.

The **transform.rotation.eulerAngles** => defines how much the object has rotated in degrees about the x, y, z axis.

The **transform.localScale** => estimate of how big or small the object is.

We access the Transform component (i.e. the position, rotation, and scale) of any game object.

Let’s create a new Project as follows

# **3.8.1 Create a new Project**

Navigate to Unity Project; and let’s create a new project, as follows

(NavigateTo).UNITY HUB -- (IN).Unity Hub – Projects – New – 3D – Settings -- / Project Name – Rename. <TEST001 > /

(NavigateTo).UNITY HUB -- (IN).Unity Hub – Projects – New – 3D – Settings -- / Location.< (Select).Filepath> /

(NavigateTo).UNITY HUB -- (IN).Unity Hub – Projects – New – 3D – Settings -- / (Select.Click).Create /

Once you’ve created the project and launched it. Create the following Asset folders in the Project Window as follows:

* **\_TEXTURES, \_MATERIALS, \_SCRIPTS, \_SCENES, \_PREFABS**

(IN).PROJECT -- (Select).ASSETS -- (Select.Right Click).Create – Folder -- / (F2/Rename).< \_TEXTURES > /

(IN).PROJECT -- (Select).ASSETS -- (Select.Right Click).Create – Folder -- / (F2/Rename).< \_MATERIALS > /

(IN).PROJECT -- (Select).ASSETS -- (Select.Right Click).Create – Folder -- / (F2/Rename).<\_SCRIPTS >/

(IN).PROJECT -- (Select).ASSETS -- (Select.Right Click).Create – Folder -- / (F2/Rename).<\_SCENES >/

(IN).PROJECT -- (Select).ASSETS -- (Select.Right Click).Create – Folder -- / (F2/Rename).<\_PREFABS >/

Create a new Scene

(IN).PROJECT -- ASSETS -- / (RightClick).<\_SCENES >.Create – Scene.(Rename). < SCENE001 >/

Navigate to the Scene Window; and complete the following:

Plane x 1

(NavigateTo).SCENE -- (IN).HIERARCHY -- (Right Click).3D Object -- / (Select).Plane /

(NavigateTo).INSPECTOR – (IN).TRANSFORM -- / (Select).”3DOTS” – (Select).Reset /

(IN).TRANSFORM – SCALE -- / (Type).< X= 2, Y= 2, Z= 2 > /

Capsule x 1

(NavigateTo).SCENE -- (IN).HIERARCHY -- (Right Click).3D Object -- / (Select).Capsule /

(Navigate to INSPECTOR] – (IN).TRANSFORM -- / (Select).3DOTS – (Select).Reset /

(IN).TRANSFORM – POSITION -- / (Input).< X = 0, Y= 1, Z= 0 > /

(IN).TRANSFORM – ROTATION -- / < X= 0, Y= 0, Z= 0 >/

(IN).TRANSFORM – SCALE -- / (Input).< X= 1, Y= 1, Z= 1 > /

Materials x 2: RED, GREEN

Red

[ Navigate to PROJECT ] -- ASSETS -- \_MATERIALS – (Right Click).Create – Materials -- / (F2/Rename).< RED > /

(IN).INSPECTOR – Main Maps – Color Picker -- / Color. < R = 255, G = 0, B = 0, A = 255 > /

(IN).PROJECT – ASSETS -\_MATERIALS -- / (Select).(Drag).RED – (Onto).(IN).HIERARCHY.(Drop).CAPSULE

Green

[ Navigate to PROJECT ] -- ASSETS -- \_MATERIALS – (Right Click).Create – Materials -- / (F2/Rename).< GREEN > /

(IN).INSPECTOR – Main Maps – Color Picker -- / Color. < R = 0, G = 255, B = 0, A = 255 > /

(IN).PROJECT – ASSETS -\_MATERIALS -- / (Select).(Drag).GREEN – (Onto).(IN).HIERARCHY.(Drop).PLAN

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class VectorEx1 : MonoBehaviour

{

    Vector3 pos1; //Create a new object from the Vector3 class type

    // Use this for initialization

    void Start ()

    {

        pos1 = new Vector3(8f, 0, 4f);

        transform.Translate(pos1);

        Vector3 pos = transform.position;

        Vector3 rot = transform.rotation.eulerAngles;

        Vector3 scale = transform.localScale;

        Debug.Log($"Position = {pos}");

        Debug.Log($"Rotation = {rot}");

        Debug.Log($"Scale    = {scale}");

    }

    // Update is called once per frame

    void Update ()

    {

        //

    }

}

# **3.8.2 Transform.position**

Remember every game object has a **transform** component > and each transform component has a **x,y, & z** co-ordinate position. So, to get the x-position of the gameObject in our scene. We access the x-co-ordinate by writing **transform.position.x**. Now to move the object in our scene. Let's say 5f units every frame (from Left to Right). We can do that "by updating the position" every frame.

1. Firstly, we create(& declare) the object in our script from the Vector Class => **Vector3 tempPos**;
2. Secondly, we store the initial gameObject position (x, y, z co-ordinate value position ) inside of a temporary variable **tempPos** => **transform.position** = tempPos;
3. Thirdly now, because we want to update the position of our object by 5f units (in the x-direction) every frame. We use the void Update() function repeatedly, to 1)re-calculate the gameObject's new position 2)Update the gameObject Transform component.

Please note whenever updating the transform position you cannot do it directly (the new position must be calculated first). It can hold only 1 (x, y, z co-ordinate position) at a time. i.e. a gameobject cannot be in two locations at the same time.

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class VectorEx1 : MonoBehaviour

{

    Vector3 tempPos; //Declare tempPos outside, so both the Start() and Update() functions have access

    // Use this for initialization

    void Start ()

    {

        //we reference the initial position of our object then set its value to a temporary variable

        tempPos = transform.position;

    }

   // Update is called once per frame

   void Update ()

    {

        tempPos.x += 0.05f; //here we modify x-value of the position transform component

        Debug.Log($"x-value =  {tempPos.x}"); //outputting to screen

        transform.position = tempPos; //we update the transform.position component.

    }

}

# **3.8.3 Transform.translate**

To physically move a game object in our scene from one point to another point we use the **transform.translate** method.

Since we want to move the game object, and repeatedly update the frame, we need to use the **void update()** function. The first thing to do is we need to define what speed we need to move our game object. We do this by declaring a public variable for the speed as follows:

public float speed = 0.01f;

Next, we need to write transform component which gives us access to the position information attached to our game object then use a special function called translate using dot notation, as follows:

transform.Translate()

If I press the down/up arrow key, you will see many different variations of this function. For now, we just want to keep it simple, and move our game object along horizontally in the x-axis, we apply a speed in the x-axis direction to our object, but keep, y = z = 0

transform.Translate(speed,0,0)

Comment out //transform.Translate(speed,0,0). Now to get our gameobject moving vertically in the y direction. We simply keep x = 0, y = Speed, z = 0

transform.Translate(0,speed,0)

[CTRL+SHIFT+SAVE] then HIT play in Unity

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class VectorEx1 : MonoBehaviour

{

    public float speed = 0.5f;

    // Use this for initialization

    void Start()

    {

    }

    // Update is called once per frame

    void Update()

    {

        if (Input.GetKeyDown("a"))

        {

            //speed = speed - 0.1f;

            speed-=0.1f;

        }

        else if (Input.GetKeyDown("s"))

        {

            //speed = speed + 0.1f;

            speed += 0.1f;

        }

        transform.Translate(speed\*Time.deltaTime, 0, 0);

        Debug.Log("x speed = {speed}");

    }

}

# **3.8.4 Transform.rotate**

In this section we are going to learn how we can rotate our game object.

|  |
| --- |
| **CREATE 1 x NEW SCRIPT** |
| (Navigate to)[PROJECT tab] > (Select) [ASSETS] [\_SCRIPTS] > (RIGHT CLICK)(Create)[C# Script] > (F2 RENAME or NAME) < NewScriptX > |
| > (Select)[\_NewScriptX](Drag) > (Navigate to)[HIERARCHY tab][CAPSULE](Drop)[\_NewScriptX] Or (Select)[\_NewScriptX](Drag) > (Navigate to)[INSPECTOR tab][CAPSULE](Drop)[\_NewScriptX] Add the script as a component in the inspector |
| > (Select)[\_NewScriptX](Drag) > (Navigate to)[HIERARCHY tab][CAPSULE](Drop)[\_NewScriptX] Or Add the script as a component in the inspector |

Like how we used transform.translate to move or change the position of a game object. We can access the rotation component, by calling **transform.Rotate** = > then by inputting an x angle, y angle and z angle. Which is **the amount of angle we want to rotate the object per frame with respect to the x axis y axis and z axis.**

transform.Rotate

First, we need to declare and define the Rotation speed

public float speed = 0.05f;

Next, let's say we want to Rotate the object with respect to the x-axis. We write:

transform.Rotate(speed,0,0)

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class RotateEx1 : MonoBehaviour

{

    public float speed = 0.5f;

    // Use this for initialization

    void Start ()

    {

    }

   // Update is called once per frame

   void Update ()

    {

        transform.Rotate(speed, 0, 0);

    }

}

Similarly, if we want to create the rotation about the y-axis, we simply add speed to the y-parameter, and keep x=z=0

|  |
| --- |
| void Update () |
| { |
| transform.Rotate(0,speed,0); |
| } |

|  |
| --- |
| **CREATE 1 x NEW SCRIPT** |
| (Navigate to)[PROJECT tab] > (Select) [ASSETS] [\_SCRIPTS] > (RIGHT CLICK)(Create)[C# Script] > (F2 RENAME or NAME) < NewScriptX > |
| > (Select)[\_NewScriptX](Drag) > (Navigate to)[HIERARCHY tab][CAPSULE](Drop)[\_NewScriptX] Or (Select)[\_NewScriptX](Drag) > (Navigate to)[INSPECTOR tab][CAPSULE](Drop)[\_NewScriptX] Add the script as a component in the inspector |
| > (Select)[\_NewScriptX](Drag) > (Navigate to)[HIERARCHY tab][CAPSULE](Drop)[\_NewScriptX] Or Add the script as a component in the inspector |

# **3.8.5 Transform.localScale**

We access the scale value by using the transform.localScale function. Unfortunately, Unity does not give us permission to change the scale directly, so we create a temp variable. i.e. store localScale inside of a temp variable => temp =transform.localScale

Vector3 temp;

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class RotateEx1 : MonoBehaviour

{

    Vector3 temp;

    // Use this for initialization

    void Start ()

    {

    }

   // Update is called once per frame

    void Update ()

    {

        temp = transform.localScale; //temp we are storing the localScale

        temp.x += 0.01f; //increasing or change the scale by 1 each & every frame

        transform.localScale = temp; //assigning temp to the localScale

    }

}

...As you can see. It moves very fast. So, we can use Time.deltatime.

# **Glossary**

# **Resources**

/End