FPT Education
FPT UNIVERSITY

Unity

Basic Game Object Manipulation and Transformations

Objectives

- Understand the concept of game objects and their role in Unity.
- Learn basic game object manipulation techniques.
- Grasp the fundamentals of transforming game objects in Unity.

What are Game Objects?

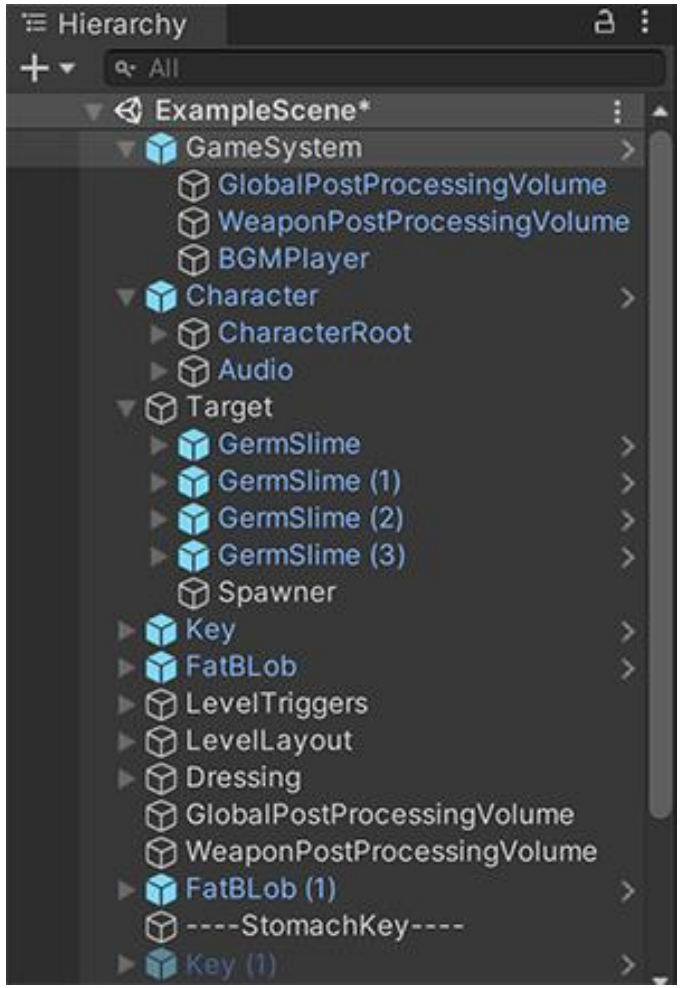
GameObjects are the foundational components of a Unity game project.

"*GameObjects are the fundamental objects in Unity that represent characters, props and scenery. They do not accomplish much in themselves but they act as containers for Components, which implement the functionality.*"[8]

Every object in your game is a GameObject, from characters and collectible items to lights, cameras and special effects.

2: Buttfield-Addison Paris, Manning Jon, Nugent Tim, Unity game development cookbook: essentials for every game, 2019
8: Unity Technologies, Unity Manual, 2023

Hierarchy Panel - GameObjects - Nested



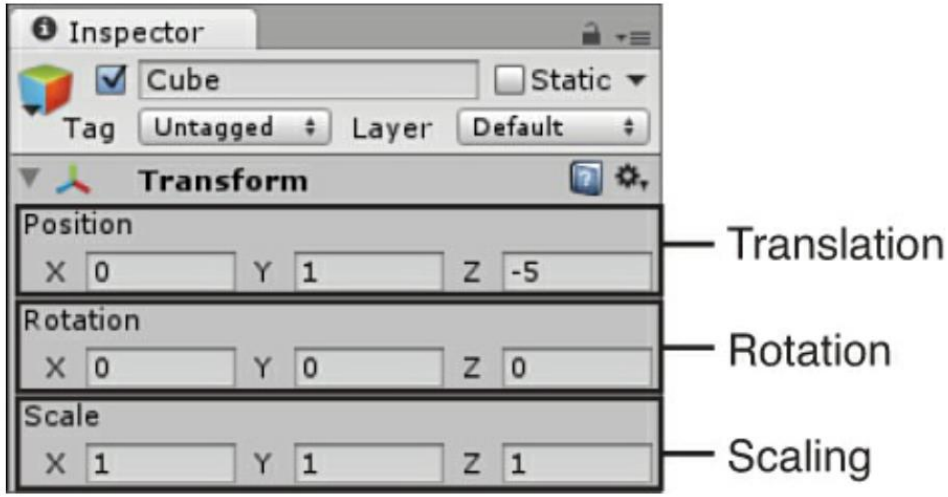
A parent GameObject has other GameObjects connected to it that take on its Transform properties.
A child GameObject is connected to another GameObject, and takes on that GameObject's Transform properties.

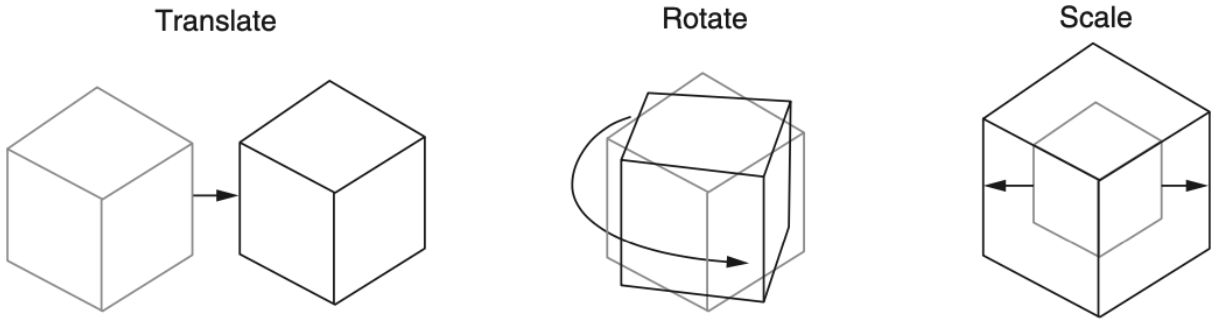
A child GameObject moves, rotates, and scales exactly as its parent does. Child GameObjects can also have child GameObjects of their own. A GameObject can have multiple child GameObjects, but only one parent GameObject.

Transforms

"*The Transform stores a GameObject's Position, Rotation, Scale and parenting state. A GameObject always has a Transform component attached: you can't remove a Transform or create a GameObject without a Transform component.*"[8]

Property:	Function:
Position	Position of the Transform in the x, y, and z coordinates.
Rotation	Rotation of the Transform around the x-axis, y-axis, and z-axis, measured in degrees.
Scale	Scale of the Transform along the x-axis, y-axis, and z-axis. The value "1" is the original size (the size at which you imported the GameObject).





1: Hocking, Joseph; Schell, Jesse, Unity in action: multiplatform game development in C#, 2022
3: Geig, Mike, Sams teach yourself Unity Game development in 24 hours, 2014

Transform Tool

You can use the Transform tools on any GameObject in a scene.
When you select a GameObject, the tool Gizmo appears within it. The appearance of the Gizmo depends on which tool you select.

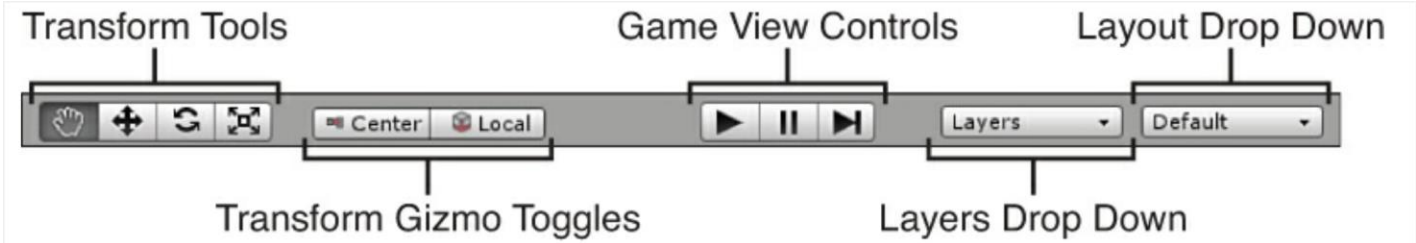
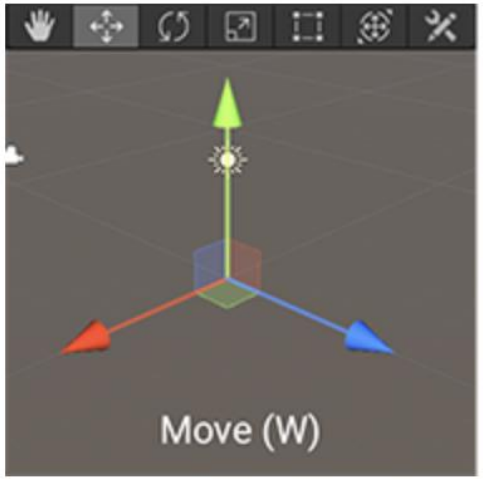
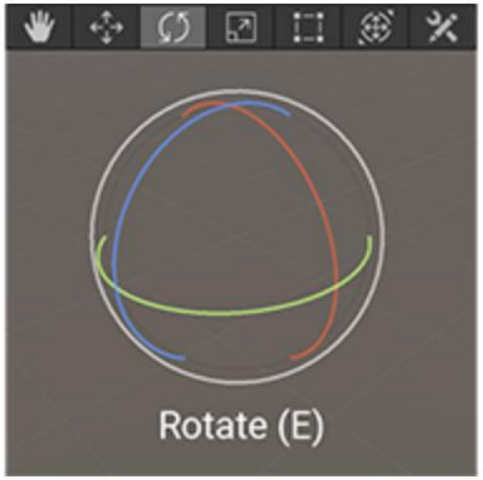
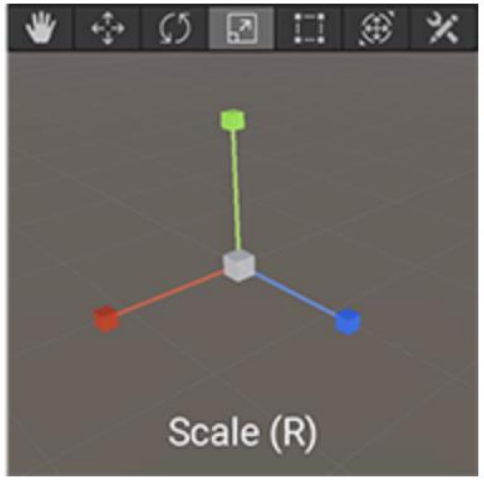


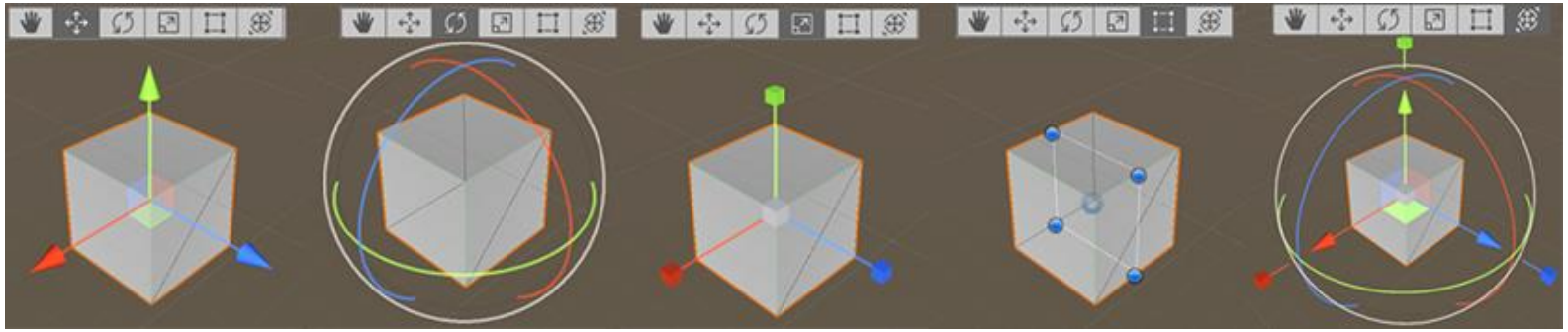
FIGURE 1.15 The toolbar.







Scene View – Position GameObjects



RectTransform (T)

The RectTransform is commonly used for positioning 2D elements such as Sprites or UI elements, but it can also be useful for manipulating 3D GameObjects. It combines moving, scaling and rotation into a single Gizmo.

Transform (Y)

The Transform tool combines the Move, Rotate and Scale tools. Its Gizmo provides handles for movement and rotation. When the Tool Handle Rotation is set to Local (see below), the Transform tool also provides handles for scaling the selected GameObject.

Transforms - Understanding 3D coordinate space

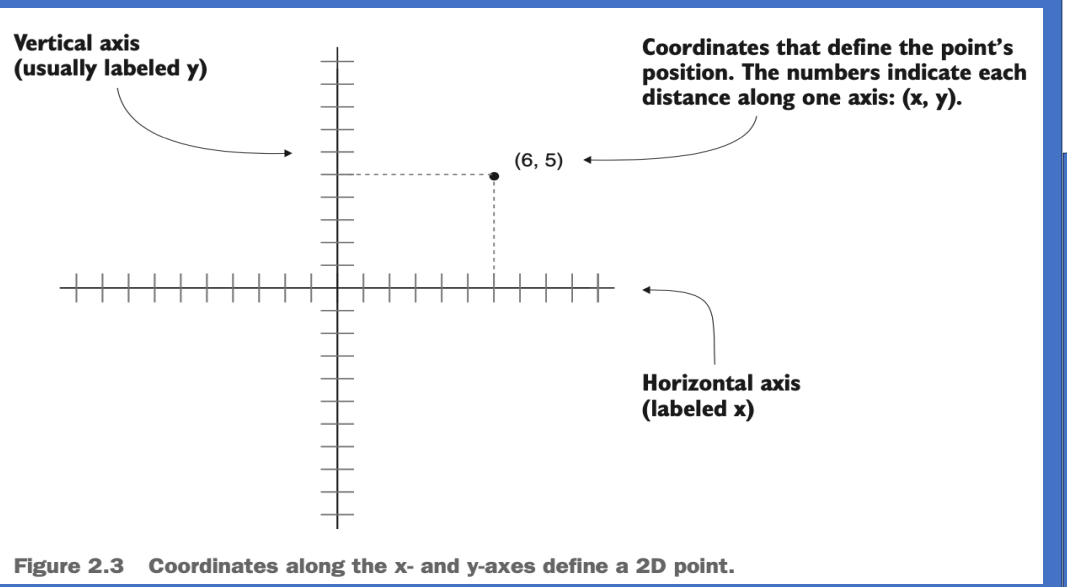


Figure 2.3 Coordinates along the x- and y-axes define a 2D point.

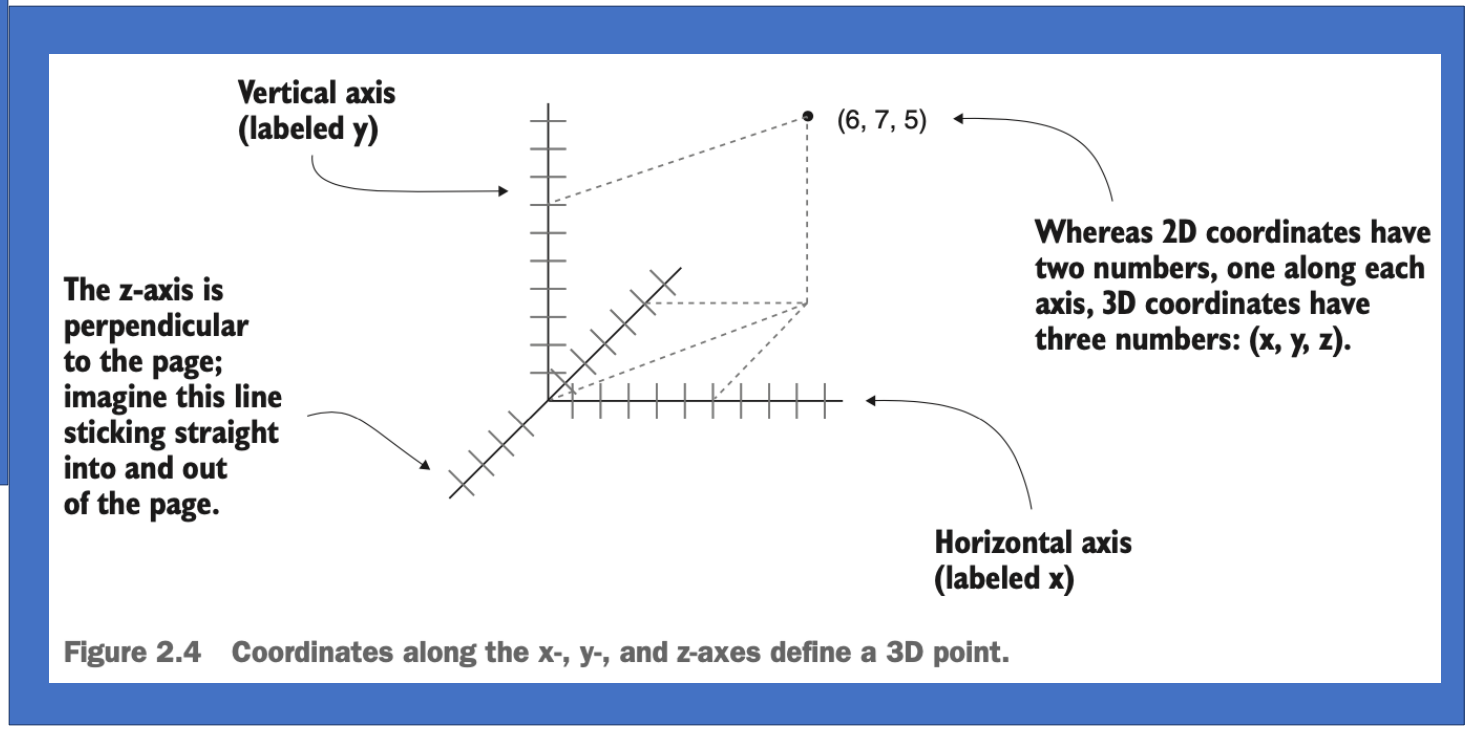
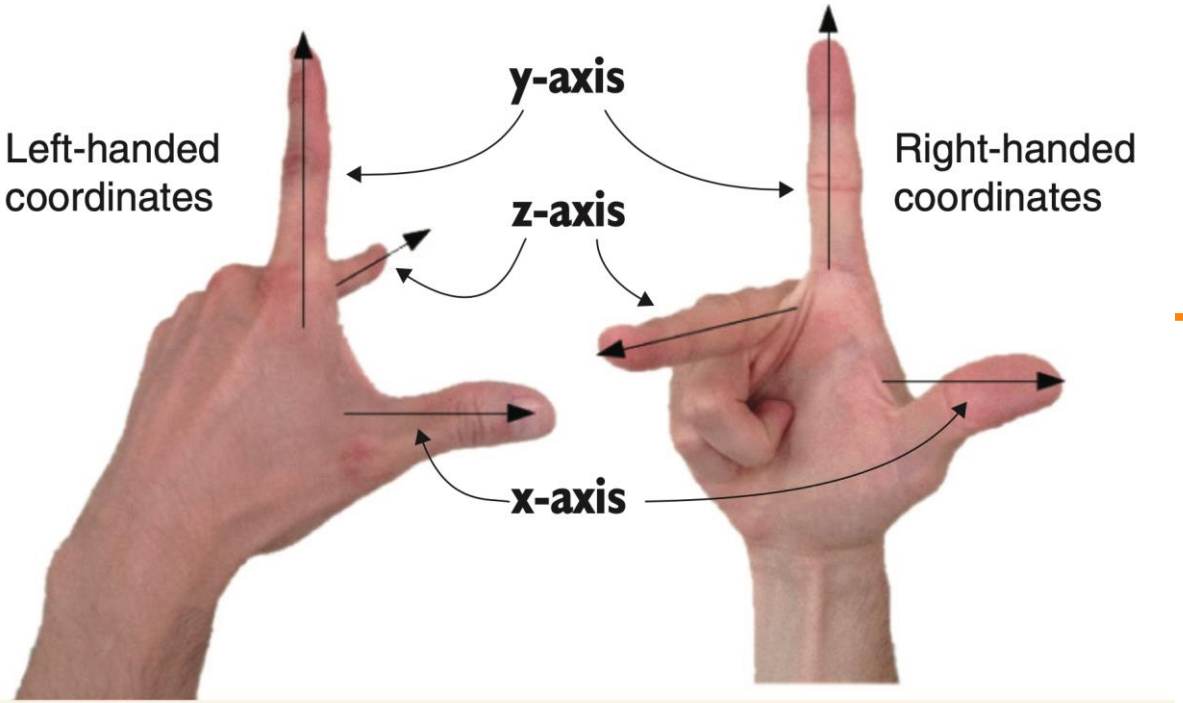


Figure 2.4 Coordinates along the x-, y-, and z-axes define a 3D point.

Transforms - Understanding 3D coordinate space



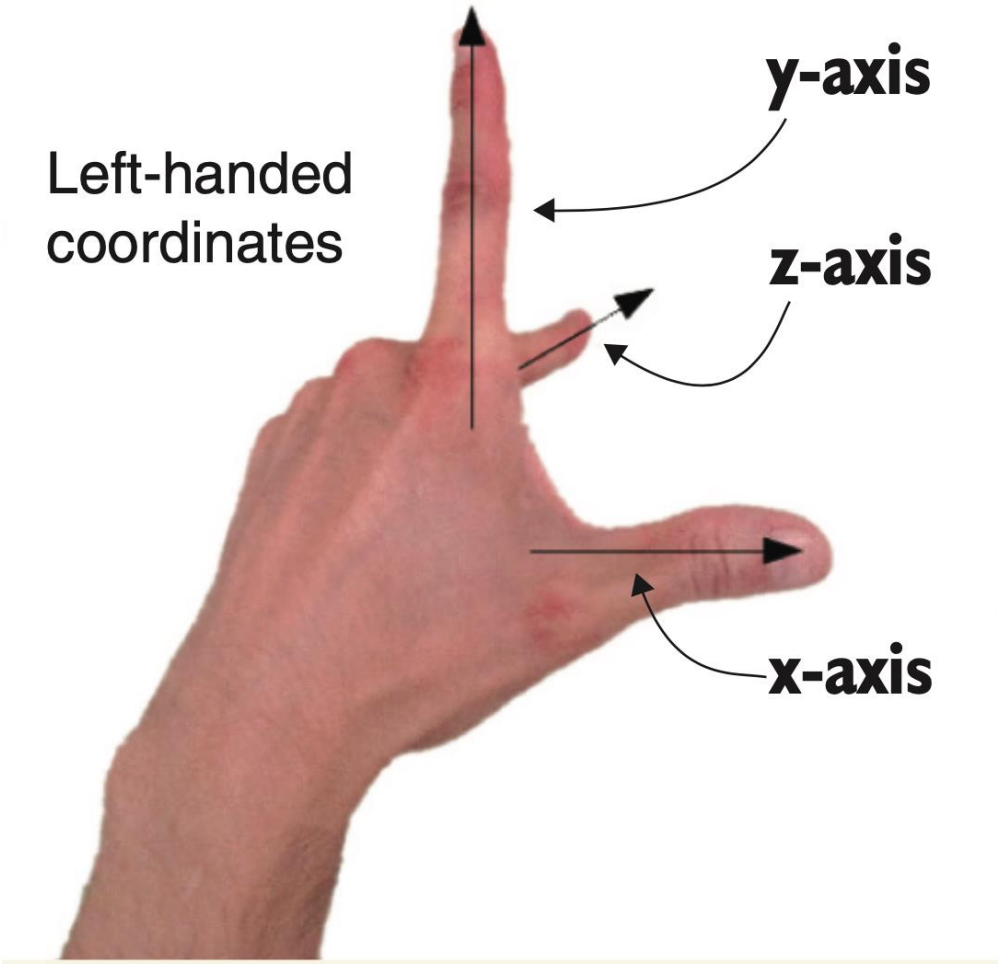
The z-axis points in a different direction on the left hand versus the right hand.

Left-handed vs. right-handed coordinates

The positive and negative direction of each axis is arbitrary, and the coordinates still work no matter in which direction the axes point. You simply need to maintain consistency within a given 3D graphics tool (animation tool, game development tool, and so forth).

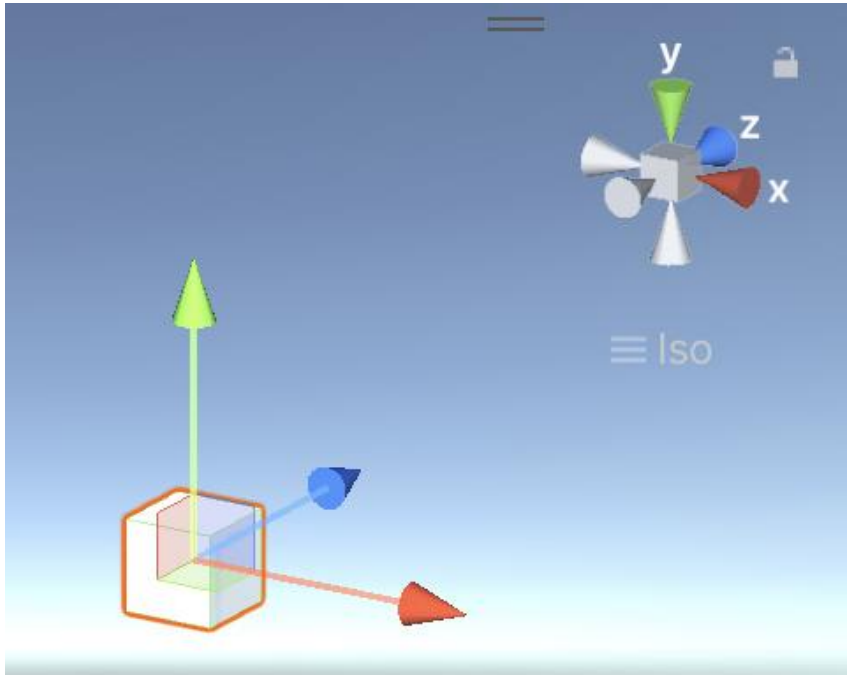
But in almost all cases, x goes to the right and y goes up; what differs between different tools is whether z goes into or comes out of the page. These two directions are referred to as left-handed or right-handed; as this figure shows, if you point your thumb along the x-axis and your index finger along the y-axis, then your middle finger points along the z-axis.

Transforms - Understanding 3D coordinate space



Unity uses a left-handed coordinate system, as do many 3D art applications.

Many other tools use right-handed coordinate systems (ex: OpenGL), so don't get confused if you ever see different coordinate directions.

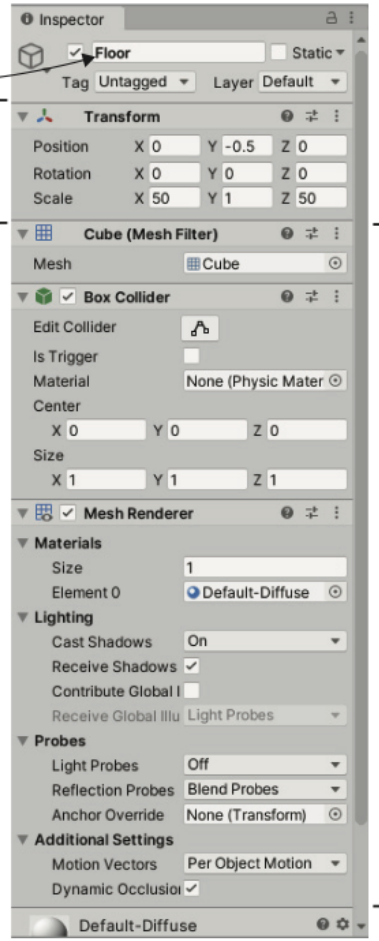


Transform Component in Inspector

At the top, you can type in a name for the object. For example, call the floor object `Floor`.

Position and scale the cube to create a floor for the room. It won't look like a cube anymore after being stretched out with differing scale values on different axes.

Meanwhile, the position is lowered slightly to compensate for the height; we set the Y scale to 1, and the object is positioned around its center.

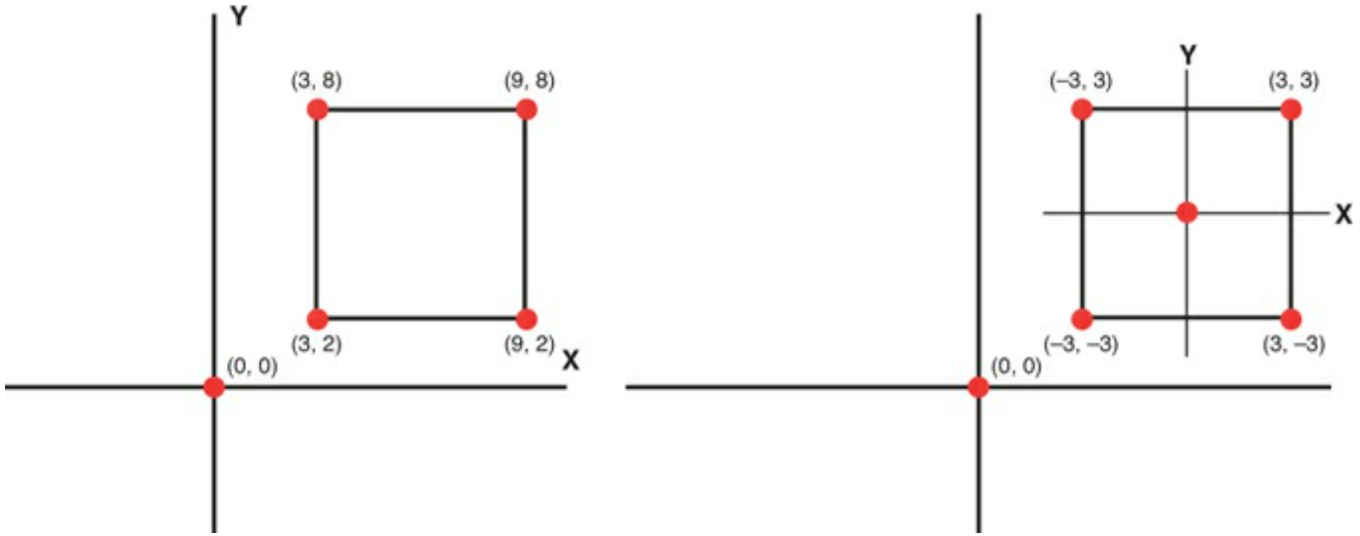


The remaining components filling the view come with a new Cube object but don't need to be adjusted right now. These components include a Mesh Filter (to define the geometry of the object), a Mesh Renderer (to define the material on the object), and a Box Collider (so that the object can be collided with during movement).

1: Hocking, Joseph; Schell, Jesse, Unity in action: multiplatform game development in C#, 2022

Understanding local vs. world coordinate space

- Every game object in Unity has a position, but a position needs a reference frame. In particular, Unity offers two ways to see (and set) coordinates:
- **World coordinates**: These are absolute coordinates of where the game object is located (by absolute, I mean with respect to the world frame, which is considered to be absolute in the game)
- **Local coordinates**: These are the coordinates of where the game object is with respect to its parent



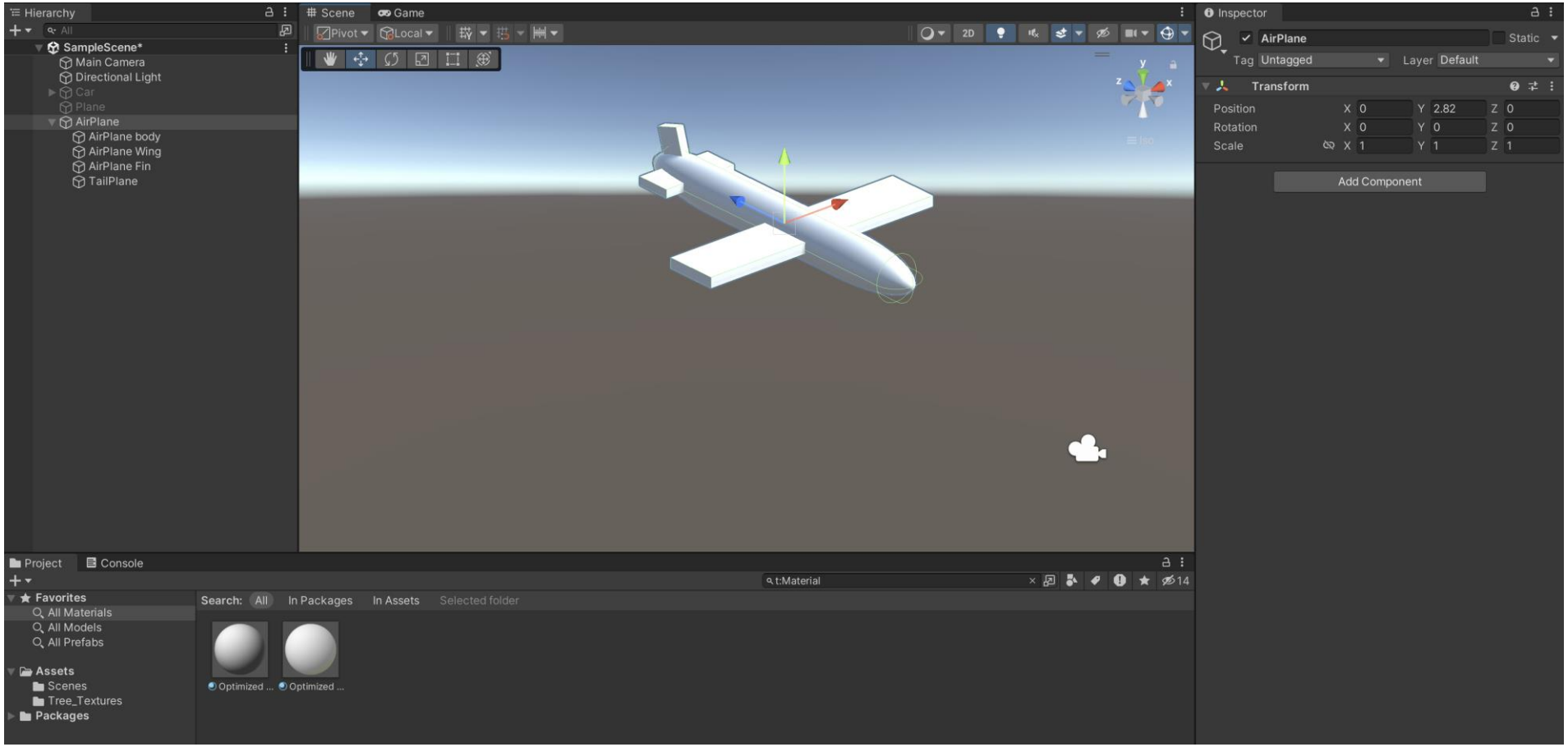
Parenting and Child Relationships

"*Game objects can be the child of other game objects. When a game object moves, rotates, or scales, its children are affected as well. This means that you can create hierarchies of game objects that work together as a system; for example, a car could have each of its four wheels as child objects, which means that they automatically stay in the correct position as the car moves around.*"[2]

TIPs: Before linking any child objects to it, make sure to reset the Trans- form options (Position and Rotation to 0, 0, 0 and Scale to 1, 1, 1) of the empty root object to avoid any oddities in the position of child objects.

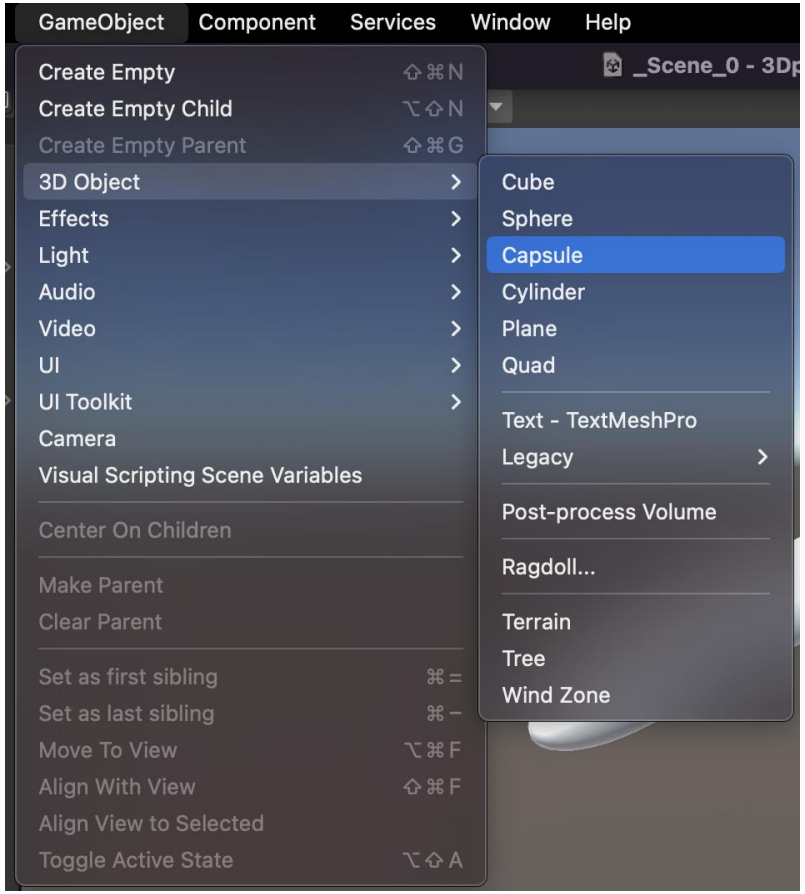
Pratice

In order to understand the moving of GameObject's transform, use Unity 3D for this practice.

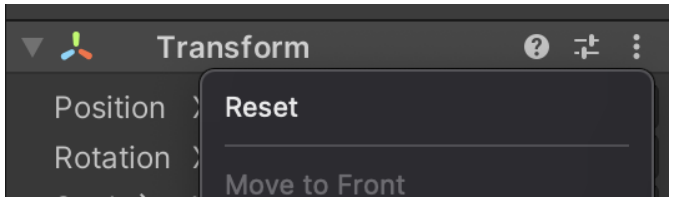


Pratice 1 – steps to create a 3D airplane

1. Add Capcule 3D GameObject. Rotate, scale it to make plane body.



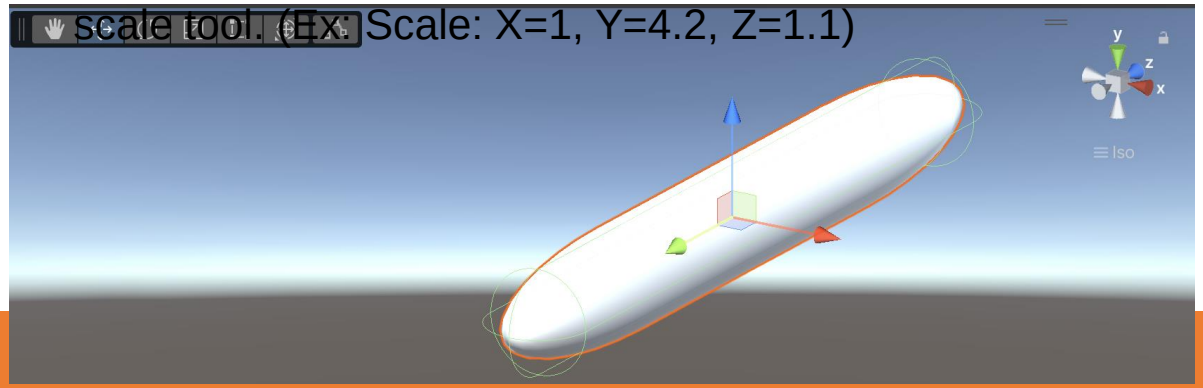
1. To add a capsule, select the menu GameObject > 3D Object > Capsule. Alternatively, right-click in the Hierarchy panel > 3D Object > Capsule. Rename it to 'AirplaneBody'.



1. *Ensure the GameObject is positioned at 0*
2. by right-click on the Transform component
3. in the Inspector and select Reset.

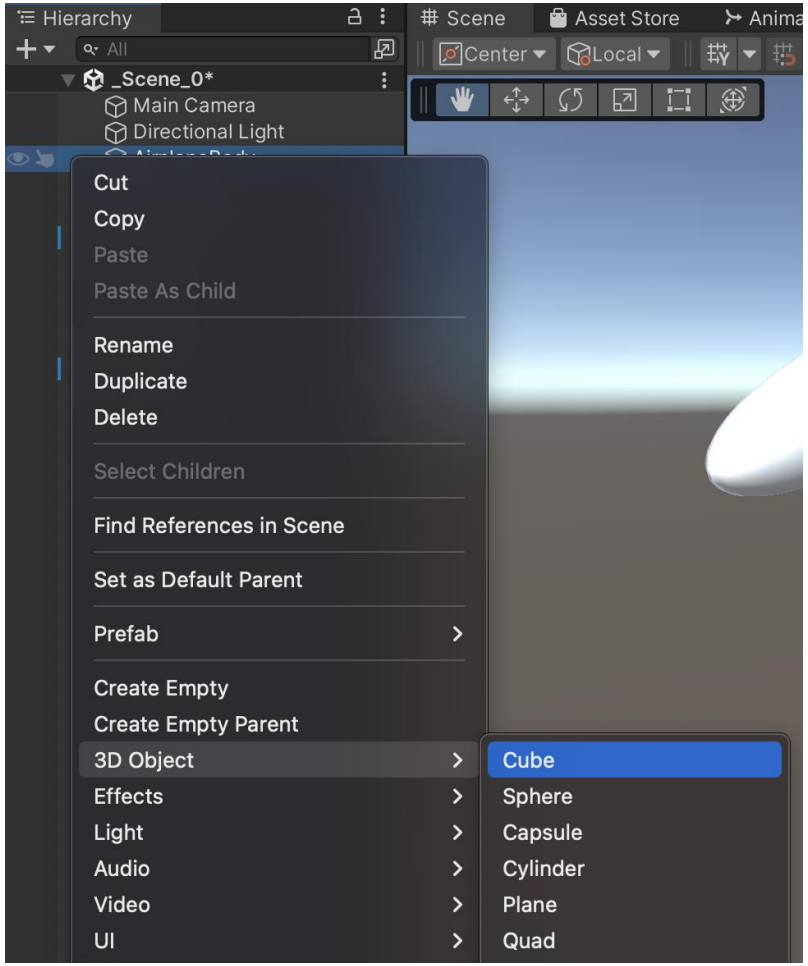
1. Switch to the Rotate tool by pressing E. Rotate the Capsule 90º degrees around the X-axis. The goal is to orient the airplane forward along the positive Z-axis. (Ex: Rotation X = -90)

1. Switch to the Scale tool by pressing R. Adjust the size of the airplane body using the scale tool. (Ex: Scale: X=1, Y=4.2, Z=1.1)



Pratice 1 – steps to create a 3D airplane

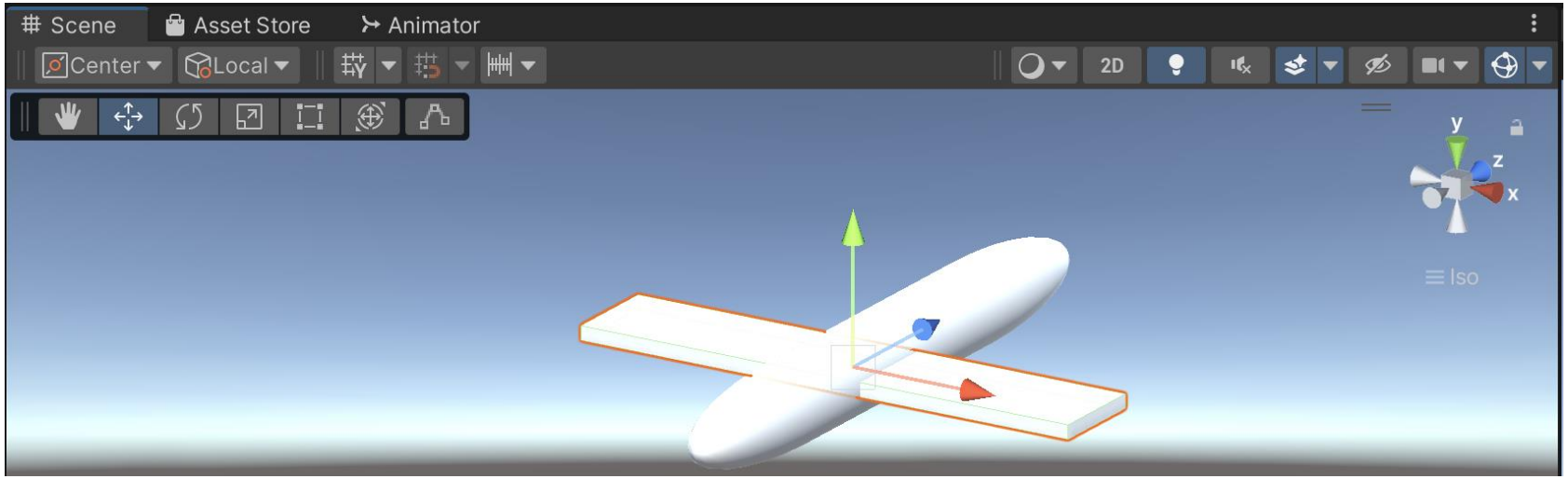
2. Add Cube. Use React, scale, rotate and move tool to make airplane's wing



1. To add a Cube, right-click in the Hierarchy panel > 3D Object > Cube. Rename it to 'AirplaneWing'. Reset Transform values.

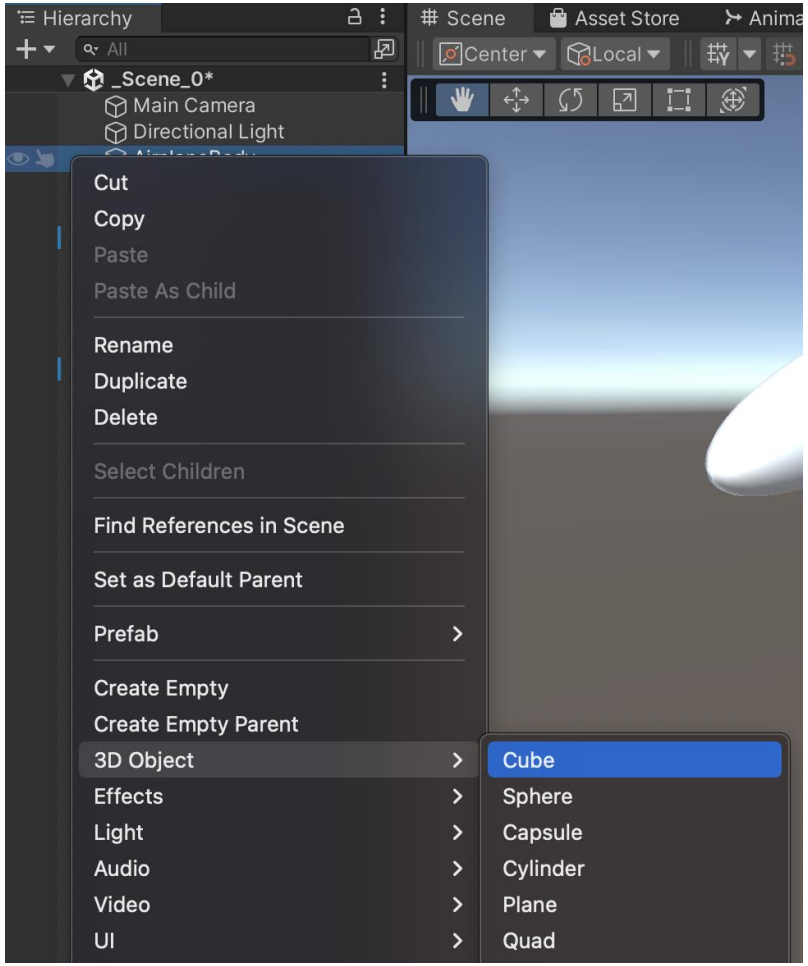
1. Switch to the Scale tool by pressing R. Adjust the size of the airplane body using the scale tool. (Ex: Scale: X=6.8, Y=0.18, Z=1.3)

1. Swith to Move tool (W), move the wing a little bit to head of airplane (Z = 0.58)



Pratice 1 – steps to create a 3D airplane

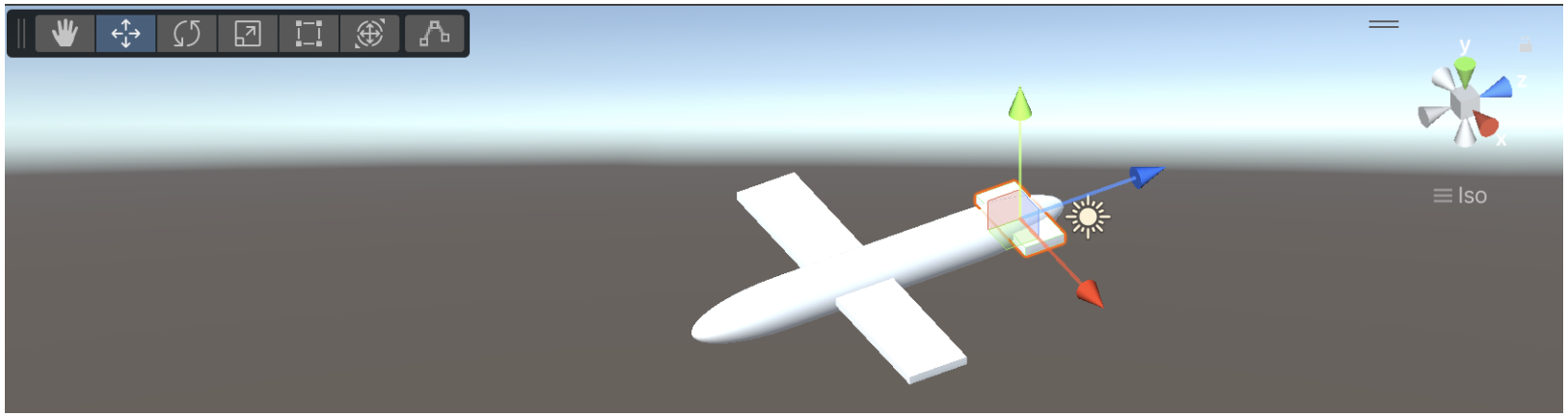
3. Add a Cube. Use React, scale, rotate and move tool to make tailplane



1. To add a Cube, right-click in the Hierarchy panel > 3D Object > Cube. Rename it to 'TailPlane'. Reset Transform values.

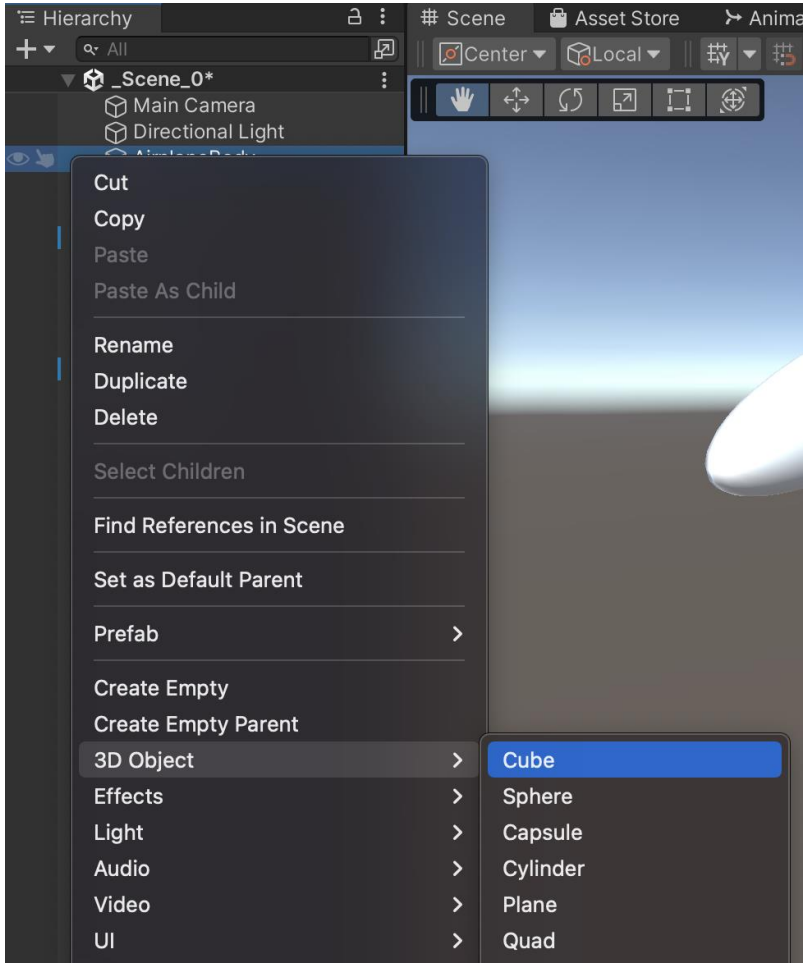
1. Switch to the Scale tool (R). Adjust the size of the airplane body using the scale tool. (Ex: Scale: X=1.93, Y=0.2, Z=0.87)

1. Swith to Move tool (W), move the tailplane to the rear of airplane (Z = 3.23)



Pratice 1 – steps to create a 3D airplane

4. Add a Cube. Use React, scale, rotate and move tool to make AirplaneFin

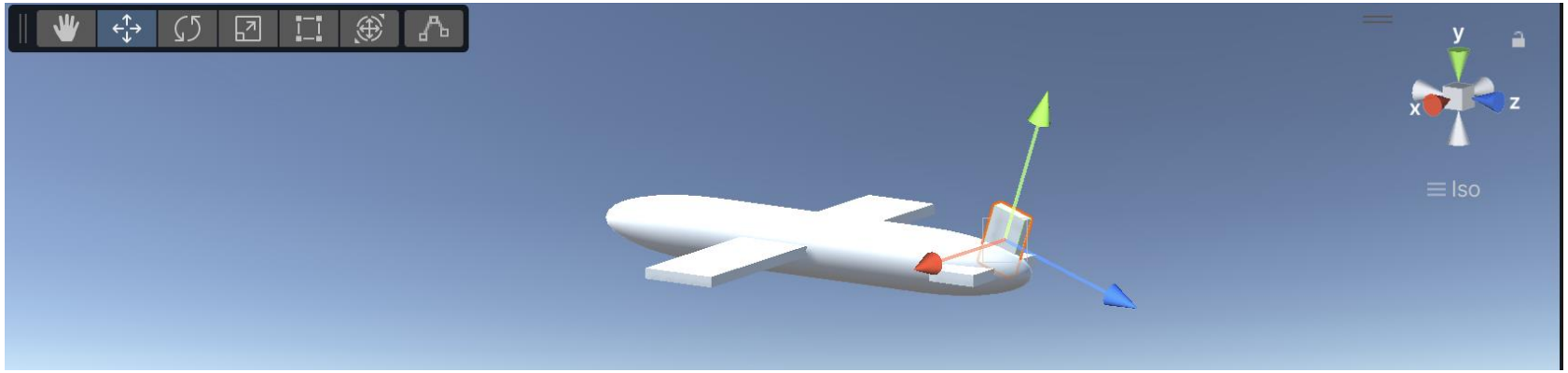


1. To add a Cube, right-click in the Hierarchy panel > 3D Object > Cube. Rename it to 'AirplaneFin'. Reset Transform values.

1. Switch to the Scale tool (R). Adjust the size of the airplane body using the scale tool. (Ex: Scale: X=1.93, Y=0.2, Z=0.87)

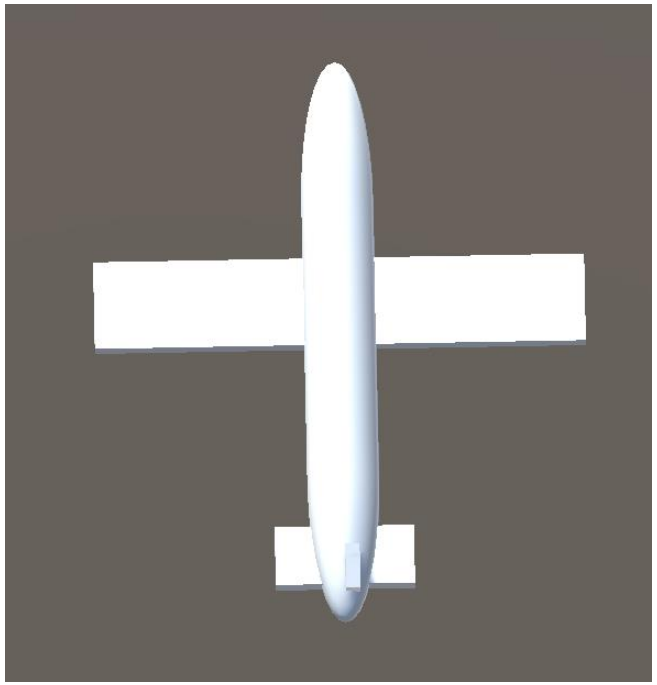
1. Swith to Move tool (W), move the AirplaneFine to the rear of airplane (X=-0.1, Y=0.5, Z = 3.6)

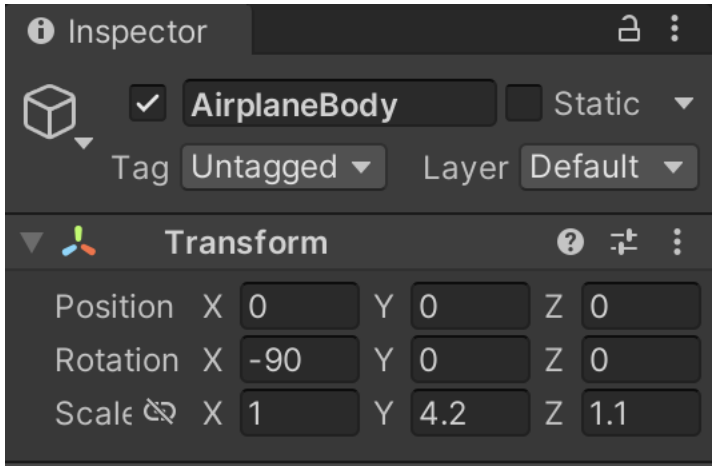
1. Use Rotate tool, rotate AirplaneFin 18º along the X axis. (X=18)

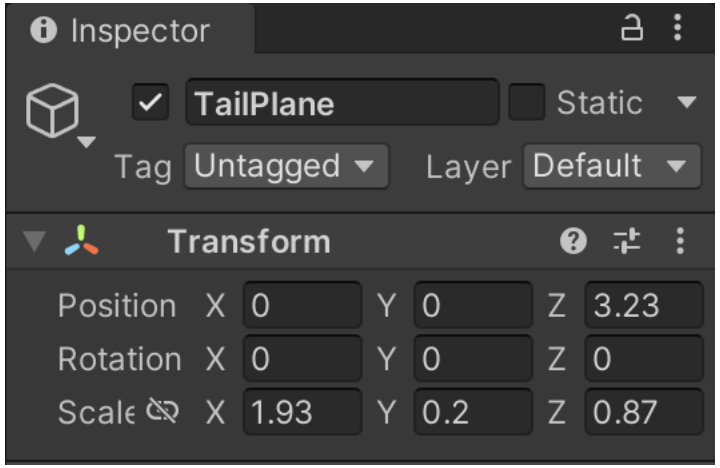


Pratice 1 – Use transform tool to create 3D Objects

Finally, we have a 3D airplane object with the following parts:

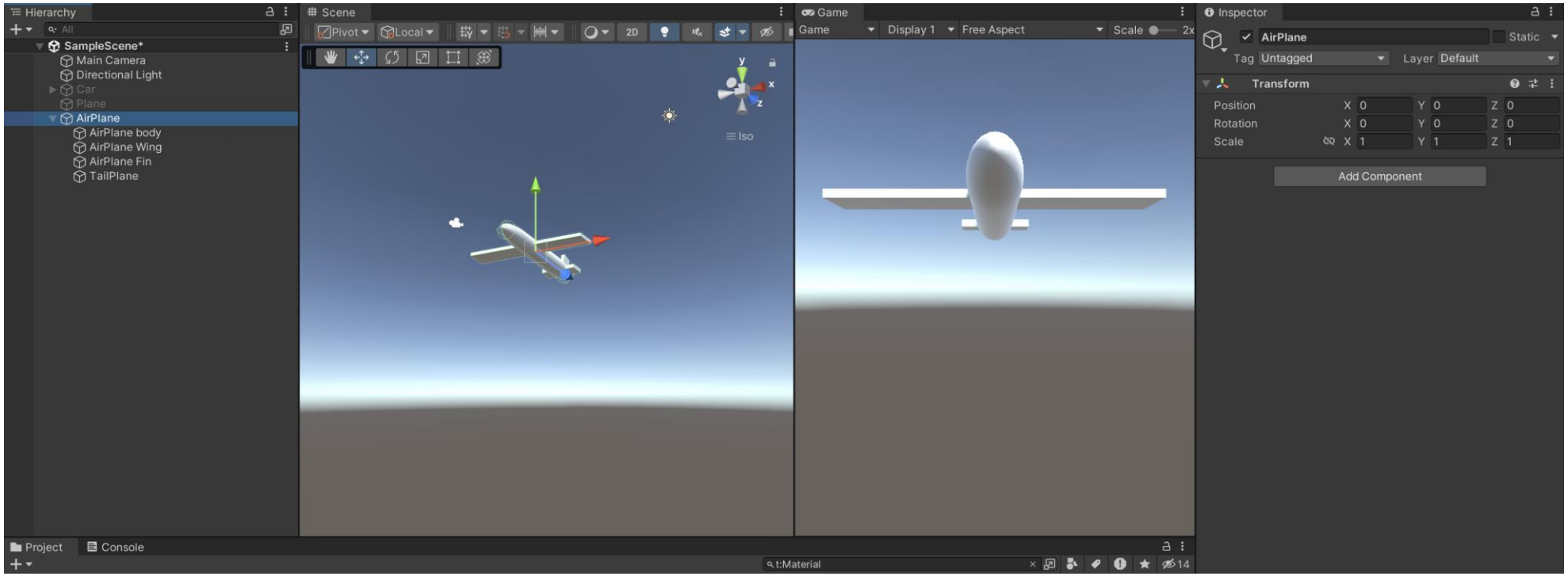






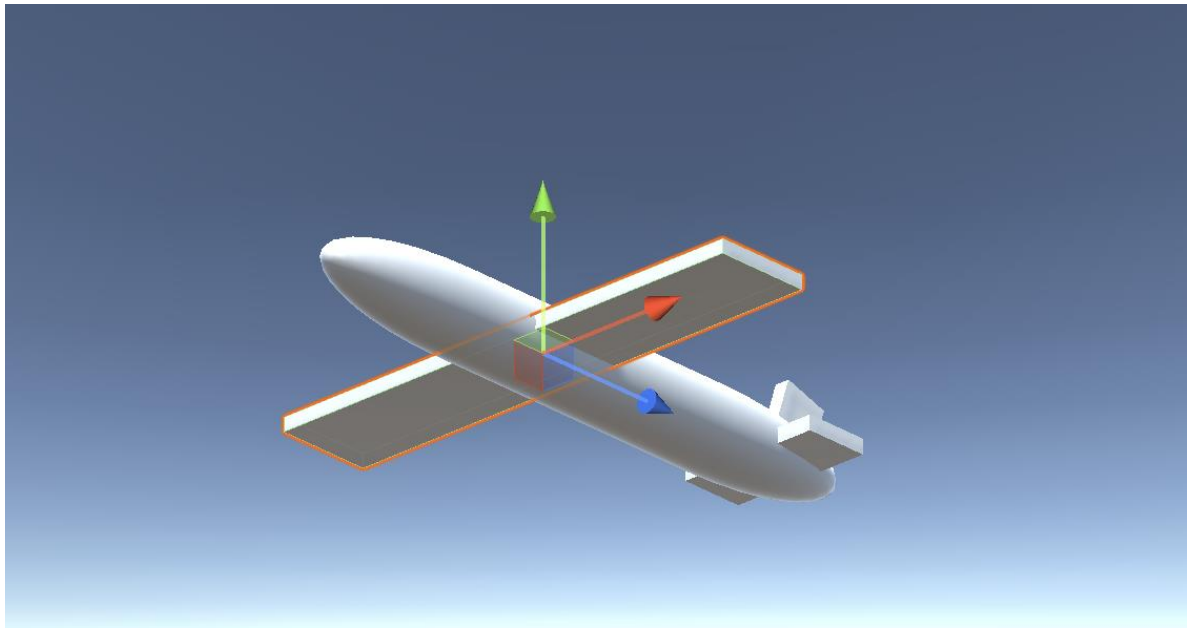
Pratice 2 – Create nested object

1. Create empty 3D GameObject. **Reset Transform values**. Rename this empty object to AirPlane
2. Drop all airplane parts to AirPlane.
3. Move, rotate and scale AirPlane object to test the transform of other airplane's part



Challange

Change the Z-axis direction of the Airplane

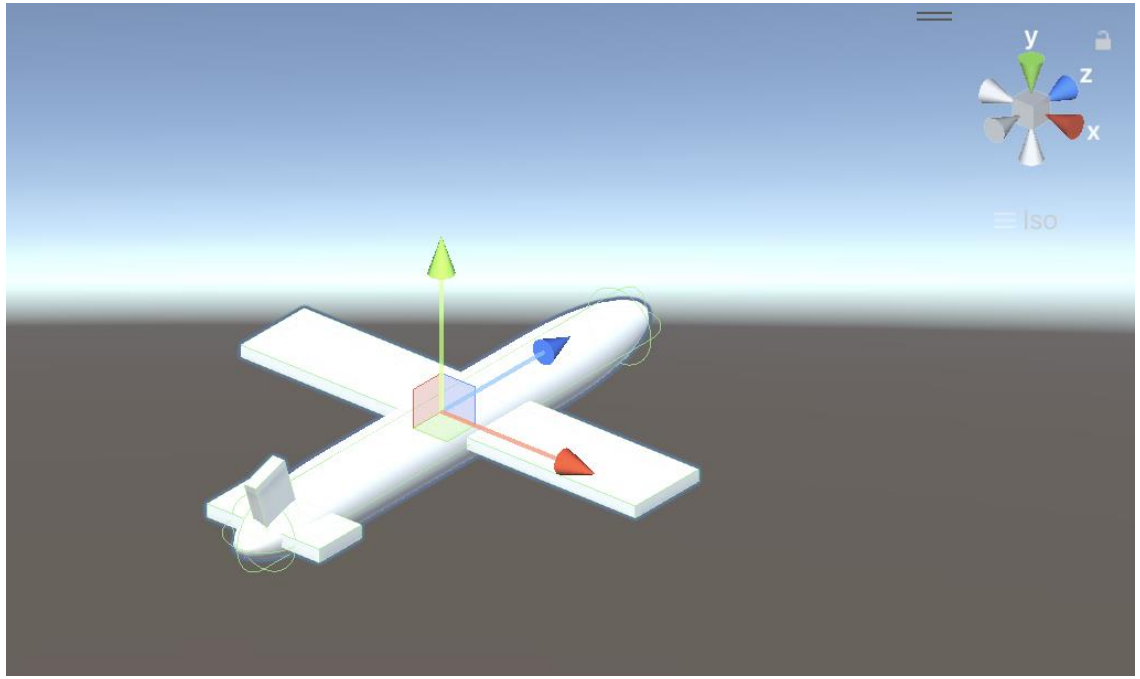


Looking at the airplane's coordinates, you'll notice that the Z-axis is opposite to the direction of the airplane's flight. This means that as the airplane moves forward, the Z-axis values decrease.

In typical gameplay, players often move along the positive Z-axis, where moving forward increases the Z-axis values.

Question: How can you invert the Z-axis direction of the airplane while ensuring that when *resetting the Transform values*, the airplane's orientation isn't reversed as before?

Answer



1. Rotate the airplane 180 degrees along the Y-axis.
2. Create a new Empty Object. Reset the Transform of this new empty object.
3. Move (drag and drop) all components from the old empty object to this new empty object

Conclusions

- Explored game object fundamentals and their significance in Unity scenes.
- Learned manipulation techniques—selecting, moving, rotating, and scaling objects.
- Mastered Transform component usage and understood coordinate systems' impact.
- Grasped parent-child relationships' influence on object transformations.

References

1: Hocking, Joseph; Schell, Jesse, Unity in action: multiplatform game development in C#, 2022

2: Buttfield-Addison Paris, Manning Jon, Nugent Tim, Unity game development cookbook: essentials for every game, 2019

3: Geig, Mike, Sams teach yourself Unity Game development in 24 hours, 2014

8: Unity Technologies, Unity Manual, 2023