27/01/2021

ASSIGNMENT

**Time and Framerate Management:**

Time.time:returns the amount of time since your project started playing.

- returns the time passed in seconds, since the game started

Note: is affected by Time.timeScale;

i.e. if game is paused then Time.time will also be paused and scales or descales according to Time.timeScale value

Application:

- to display a timer in the game

- to keep track of total game play time - the game pause time etc.

Time.deltaTime:returns the amount of time that elapsed since the last frame completed

- Time.deltaTime is most commonly used inside Update method

Example Code:

void Update (){

Debug.Log(Time.deltaTime);

}

- when used within Update method; it returns time passed in seconds; since the last Update method call i.e. how much time has been took to execute the last Update method call

- As game's FPS fluctuates, sometimes execution of last Update method call takes more time and sometimes less time; hence Time.deltaTime within Update method varies.

- when used within Fixed Update method; it returns time passed in seconds; since the last FixedUpdate method call i.e. how much time has been took to execute the last FixedUpdate method call

-As FixedUpdate is called after every 0.02s;Time.deltaTime within FixedUpdate method does not vary.It returns 0.02s.

Note:Time.deltaTime is used to update state of a game object per second instead of per frame.

* Time.timeScale:represents the rate at which time elapses. You can read this value, or set it to control how fast time passes, allowing you to create slow-motion effects.

- is used to get or set the scale at which time is passing

- It is used to speed up, slow down or pause; all time based updates

Note: It is recommended that; if you modify Time.timeScale then you must modify the Time.fixedDeltaTime by the same amount

Example Code:

void Start () {

Time.timeScale = 2f;

Time.fixedDeltaTime = Time.fixedDeltaTime \*Time.timeScale;

}

void Update () {

this.transform.position += new Vector3 (5f, of, of) \* Time.deltaTime;}

## **Framerate Management**

The update function allows you to monitor inputs and other events regularly from a script and take appropriate action. For example, you might move a character when the “forward” key is pressed. An important thing to remember when handling time-based actions like this is that the game’s framerate is not constant and neither is the length of time between Update function calls.

Update and FixedUpdate:

Update runs once per frame. FixedUpdate can run once, zero, or several times per frame, depending on how many physics frames per second are set in the time settings, and how fast/slow the framerate is.

Physics System:

W.K.T. in real world; objects get affected by collisions, gravity, friction, joints and other forces. Similarly in a game; game objects also must be affected by collisions, gravity, friction, joints and other forces.

To implement such real world physics behaviors; Unity uses: 2 different set of components: To handle 3D physics it has Physics components like:- Rigidbody, Box Collider, Capsule Collider, Fixed Joint, Spring Joint etc.

Colliders:

Colliders enable Unity to register when GameObjects strike or intersect each other. GameObjects must have a RigidBody component attached to them for collisions to occur.

Triggers:

Triggers are enabled with the “Is Trigger” checkbox selected. This functions the same as a Collider, but disables Physics on the component, enabling objects to pass through it via a zone. Events can be called when objects enter or exit the Trigger.

RigidBody:

allows GameObjects to be affected by Physics properties, such as Gravity. It also includes properties for Mass, Velocity, and Drag (air resistance). Objects of larger mass are less affected by objects with lower mass and vice versa.

Drag affects the dampening of velocity over time. Angular Drag affeThe Is Kinematic checkbox allows the Rigidbody to affect other objects via the Unity Physics Engine, but will not be affected themselves. For example, a Hand Avatar in a VR game can interact with objects via Physics, but we don’t want Physics to act on the hand.

The Is Kinematic checkbox also affects objects controlled by the Animation Engine. If the Is Kinematic checkbox is selected (on), the Animation Engine affects objects. If deselected (off), the Physics Engine retains control.

The Interpolate setting detects how collisions are checked. From the dropdown, select either **Interpolate** or **Extrapolate**.

* **Interpolate**: Smooth movements of objects are based on information from the previous frame in an animation’s timeline.
* **Extrapolate**: Smooth movements of objects are based on a guess of the next frame.

The **Collision Detection** dropdown menu sets the rate at which collisions are checked.

* **Discrete**: Default.
* **Continuous**: Fast objects that interact with static objects.
* **Continuous Dynamic**: Fast objects that interact with other fast objects.
* **Continuous Speculative**: Predictive collision checking.

Scripting:

The functions used for Colliders when interacting with other GameObjects include:

1)OnCollisionEnter:Called when a collision is registered.

2)OncollisionStay:Called during a collision

3)OnCollisionExit:Called when a collision has stopped

In order to enable or disable gravity, you would use the useGravity function.

**rigidbody.addForce**

In order to add a force in a particular direction, you would use the .AddForce() function.

**rigidbody.addForce(vector3)**

In order to add a rotational force around an axis, you would use the .AddTorque() function.

**rigidBody.AddTorque(Vector3, Force Mode)**

ForceModes:

* **Acceleration:** Applies a force that increases at a constant rate.
* **Force:** Default, gradually applies a force accounting for its mass.
* **Impulse:** Applies an instant force instead of one that gradually builds up over time.
* **VelocityChange:** Applies instant forces in different directions. Disregards mass

Updates:

* **Update**: Called once per frame.
* **FixedUpdate**: Called multiple times per frame. Most physics calculations will be called in FixedUpdate. The time between calculations is fixed, as the name implies.