DIVISION OF
DIGITAL CONTENTS
DONGSEO UNIVERSITY

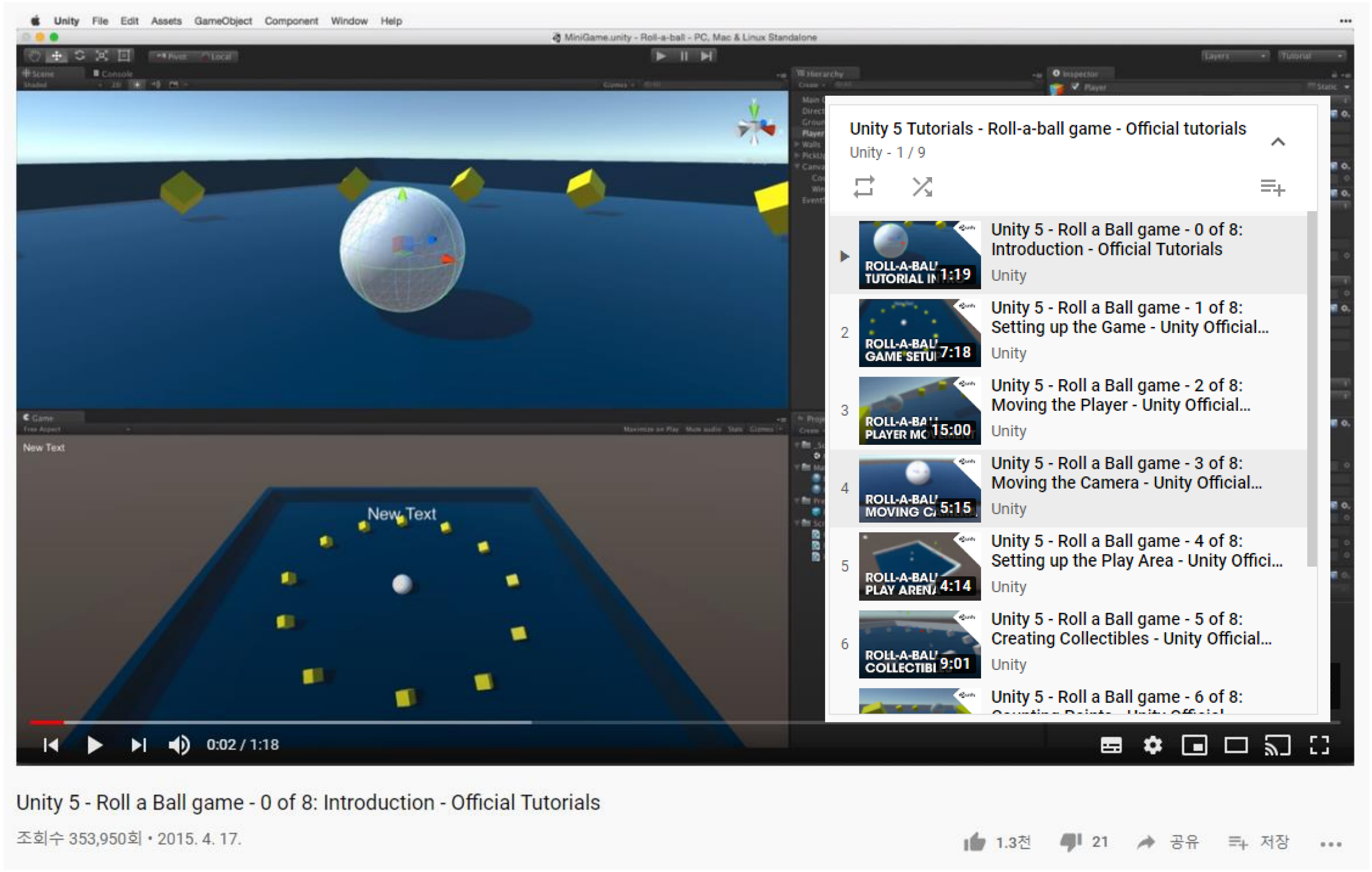
Unity Physics

Add Force to Rigidbody

jintaeks@dongseo.ac.kr

May, 2020

Demo:
Unity Roll-A-Ball Project


Unity File Edit Assets GameObject Component Window Help
MiniGame.unity - Roll-a-ball - PC, Mac & Linux Standalone
New Text
New Text
0:02 / 1:18

Unity 5 Tutorials - Roll-a-ball game - Official tutorials
Unity - 1 / 9

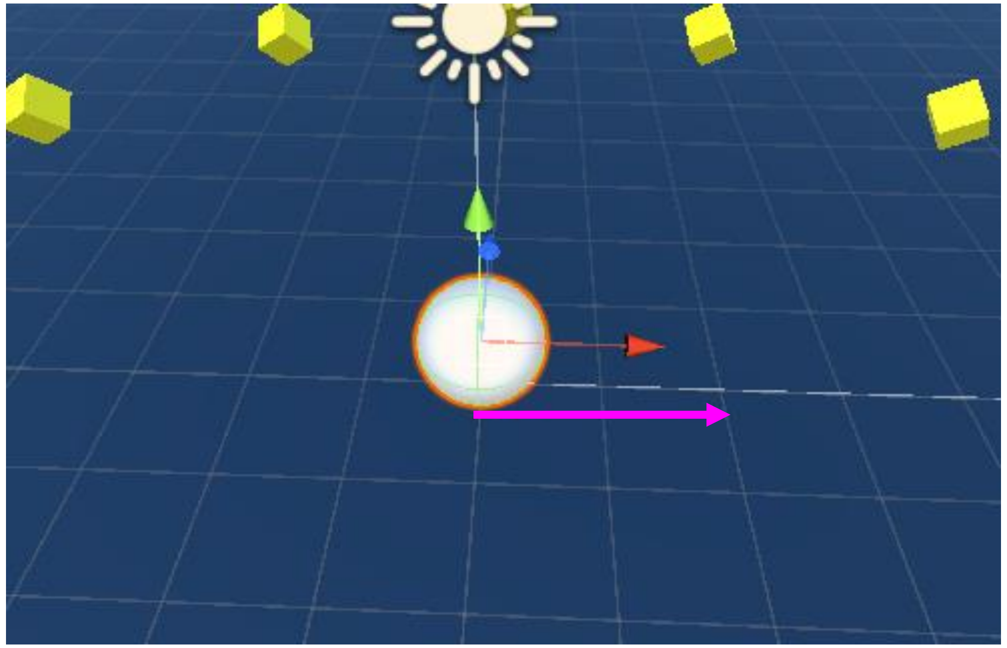
▶	ROLL-A-BALL TUTORIAL IN… 1:19	Unity 5 - Roll a Ball game - 0 of 8: Introduction - Official Tutorials Unity
2	ROLL-A-BALL GAME SETU… 7:18	Unity 5 - Roll a Ball game - 1 of 8: Setting up the Game - Unity Official... Unity
3	ROLL-A-BALL PLAYER MC… 15:00	Unity 5 - Roll a Ball game - 2 of 8: Moving the Player - Unity Official... Unity
4	ROLL-A-BALL MOVING C… 5:15	Unity 5 - Roll a Ball game - 3 of 8: Moving the Camera - Unity Official... Unity
5	ROLL-A-BALL PLAY AREN… 4:14	Unity 5 - Roll a Ball game - 4 of 8: Setting up the Play Area - Unity Offici... Unity
6	ROLL-A-BALL COLLECTIBI… 9:01	Unity 5 - Roll a Ball game - 5 of 8: Creating Collectibles - Unity Official... Unity
		Unity 5 - Roll a Ball game - 6 of 8:

Unity 5 - Roll a Ball game - 0 of 8: Introduction - Official Tutorials

조회수 353,950회 • 2015. 4. 17.

1.3천 21 공유 저장

How can we move the ball with 2 units when we pressed a key?



- ✓ **Linear momentum**, **translational momentum**, or simply **momentum**(pl. momenta) is the product of the mass and velocity of an object.
- ✓ It is a vector quantity, possessing a magnitude and a direction in three-dimensional space.
- ✓ If *m* is an object's mass and **v** is the velocity (also a vector), then the momentum is

$$p = mv$$

Many particles

- ✓ The momentum of a system of particles is the vector sum of their momenta. If two particles have respective masses m_1 and m_2, and velocities v_1 and v_2, the to tal momentum is

$$\begin{aligned} p &= p_1 + p_2 \\ &= m_1 v_1 + m_2 v_2 \,. \end{aligned}$$

- ✓ The momenta of more than two particles can be added more generally with th e following:

$$p = \sum_i m_i v_i \,.$$

$$\vec{a} = \frac{\Delta \vec{v}}{\Delta t}$$

$$dt = \Delta t$$

$$d\vec{v} = \Delta \vec{v}$$

$$\vec{a} = \frac{d\vec{v}}{dt}$$

$$\vec{v} = \vec{a}t$$

$$\vec{a} = \frac{\Delta \vec{v}}{\Delta t}$$

$$dt = \Delta t$$

$$d\vec{v} = \Delta \vec{v}$$

$$\vec{a} = \frac{d\vec{v}}{dt}$$

$$\vec{v} = \vec{a} t$$

$$\vec{F} = m\vec{a}$$

$$\vec{p} = m\vec{v}$$

$$\vec{F} = \frac{d\vec{p}}{dt}$$

$$\vec{F} dt = d\vec{p}$$

$$\vec{F} = \frac{d\vec{p}}{dt} = \frac{d(m\vec{v})}{dt}$$

$$\vec{a} = \frac{\Delta \vec{v}}{\Delta t}$$

$$\vec{F} = m\vec{a}$$

$$dt = \Delta t$$

$$\vec{p} = m\vec{v}$$

$$d\vec{v} = \Delta \vec{v}$$

$$\vec{F} = \frac{d\vec{p}}{dt}$$

$$\vec{a} = \frac{d\vec{v}}{dt}$$

$$\vec{F}dt = d\vec{p}$$

$$\vec{v} = \vec{a}t$$

$$\vec{F} = \frac{d\vec{p}}{dt} = \frac{d(m\vec{v})}{dt}$$

$$\vec{F} = m\frac{d\vec{v}}{dt}$$

$$\vec{F} = m\vec{a}$$

$$\vec{F}dt = d\vec{p}$$

$$\vec{I} = \int_{t_1}^{t_2} \vec{F}dt$$

$$\vec{I} = \vec{F}dt = d\vec{p}$$

$$\vec{I} = \vec{F}dt$$

$$\vec{I} = \vec{F}dt$$

$$\frac{\vec{I}}{m} = \frac{\vec{F}dt}{m} = \frac{m\vec{a}dt}{m} = \vec{a}dt = \vec{v}$$

$$\frac{d\vec{v}}{dt} = \vec{a}$$

$$\vec{I} = \vec{F}dt$$

```
_rigidbody.AddForce(f, ForceMode.Force);

_rigidbody.AddForce(f*time, ForceMode.Impulse);
```

$$\frac{\vec{I}}{m} = \frac{\vec{F}dt}{m} = \frac{m\vec{a}dt}{m} = \vec{a}dt = \vec{v}$$

```
Vector3 v = f*time / m;
_rigidbody.AddForce(v, ForceMode.VelocityChange);
```

$$\frac{d\vec{v}}{dt} = \vec{a}$$

```
_rigidbody.AddForce(v / time, ForceMode.Acceleration);
```

- ✓ If the net force *F* applied to a particle is constant, and is applied for a time inte rval Δ*t*, the momentum of the particle changes by an amount

$$\Delta p = F \Delta t\,.$$

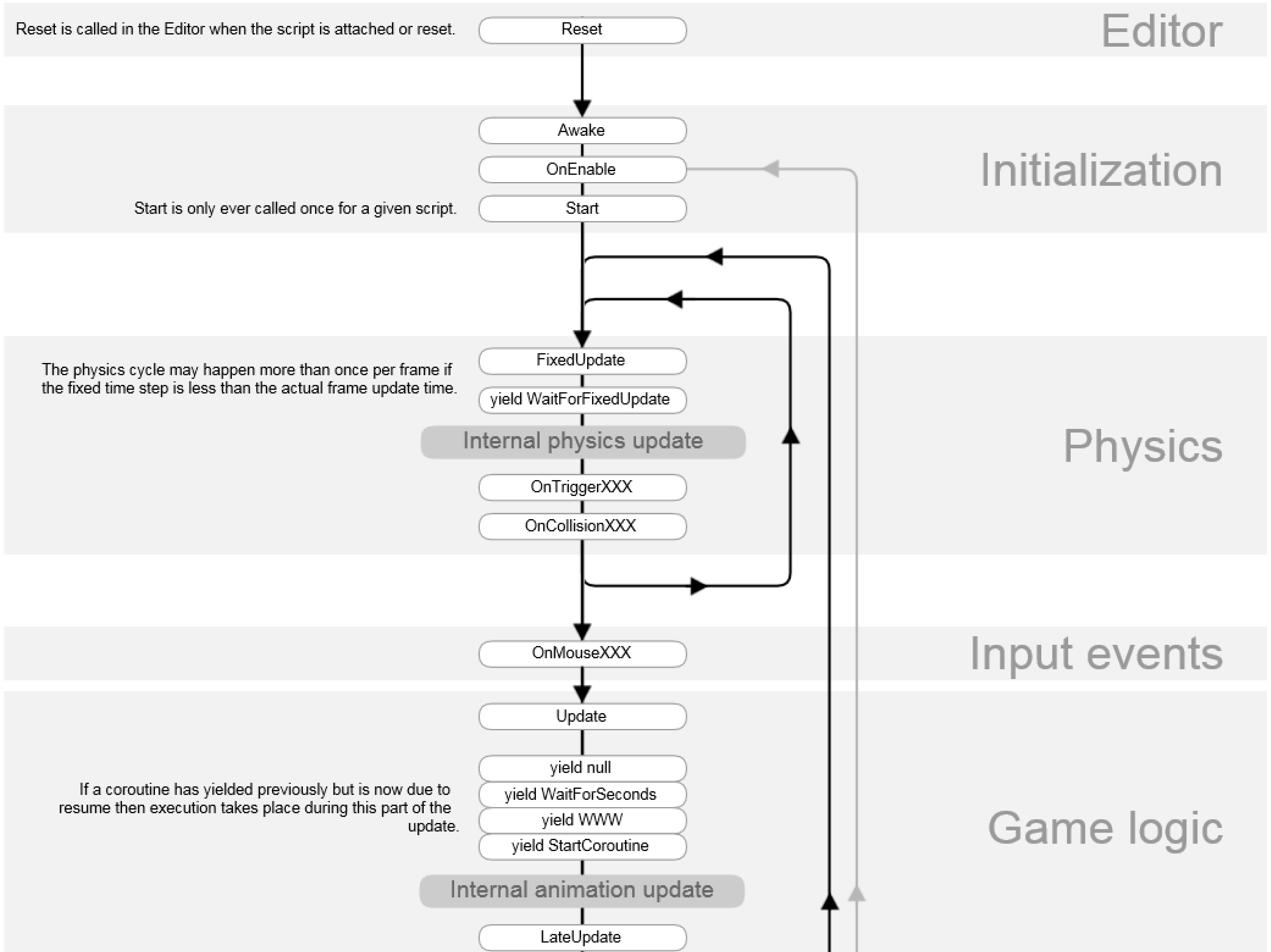
- ✓ In differential form, this is Newton's second law; the rate of change of the mom entum of a particle is equal to the instantaneous force *F* acting on it,

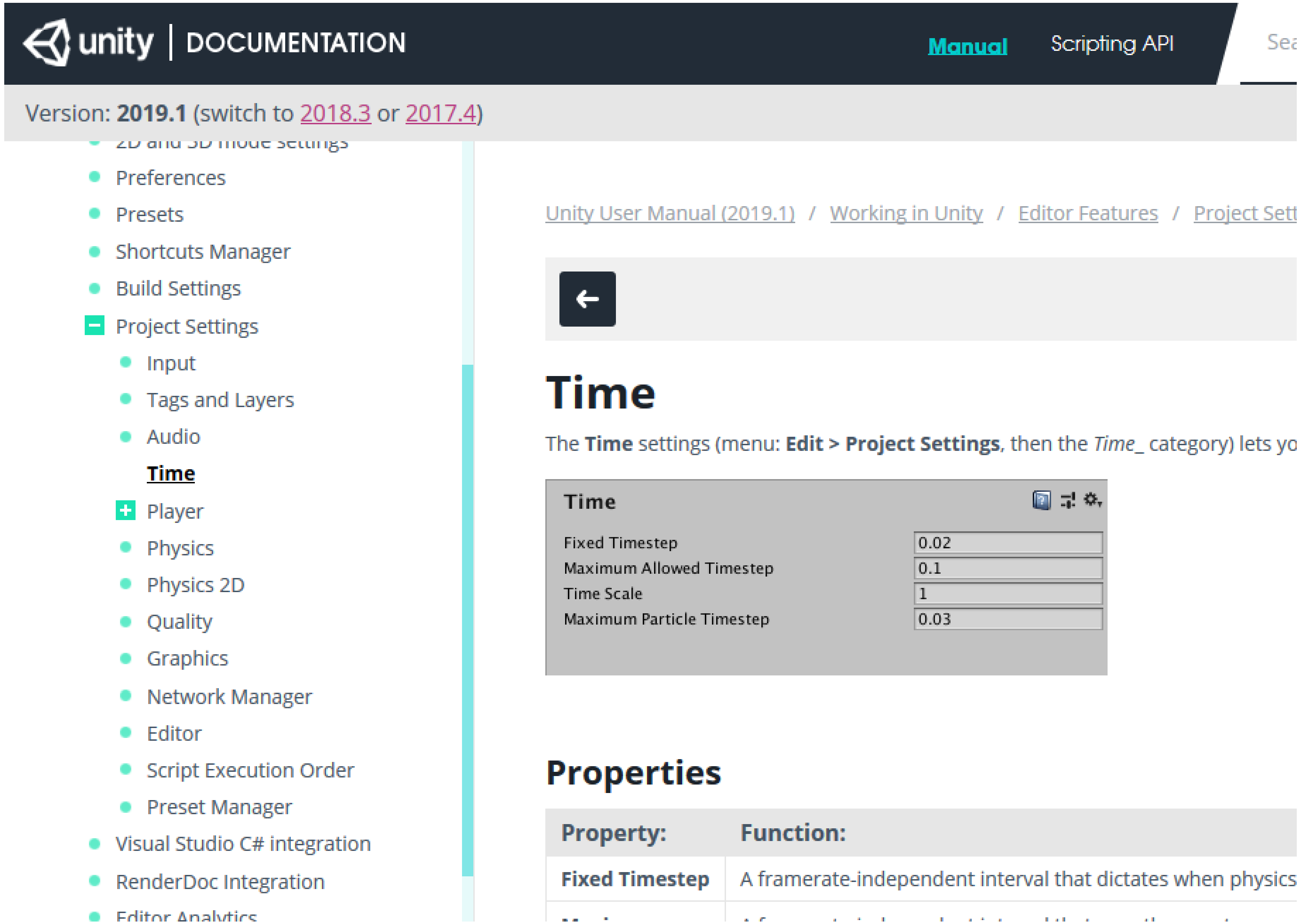
$$F = \frac{dp}{dt}.$$

- ✓ If the net force experienced by a particle changes as a function of time, *F(t)*, th e change in momentum (or impulse *J*) between times t_1 and t_2 is

$$\Delta p = J = \int_{t_1}^{t_2} F(t)\, dt\,.$$

Unity Demo


Editor
Reset is called in the Editor when the script is attached or reset.
Reset
Initialization
Awake
OnEnable
Start is only ever called once for a given script.
Start
Physics
The physics cycle may happen more than once per frame if the fixed time step is less than the actual frame update time.
FixedUpdate
yield WaitForFixedUpdate
Internal physics update
OnTriggerXXX
OnCollisionXXX
Input events
OnMouseXXX
Game logic
Update
If a coroutine has yielded previously but is now due to resume then execution takes place during this part of the update.
yield null
yield WaitForSeconds
yield WWW
yield StartCoroutine
Internal animation update
LateUpdate


unity | DOCUMENTATION

Manual   Scripting API   Sea

Version: **2019.1** (switch to 2018.3 or 2017.4)

- 2D and 3D mode settings
- Preferences
- Presets
- Shortcuts Manager
- Build Settings
- Project Settings
  - Input
  - Tags and Layers
  - Audio
  - **Time**
  - Player
  - Physics
  - Physics 2D
  - Quality
  - Graphics
  - Network Manager
  - Editor
  - Script Execution Order
  - Preset Manager
- Visual Studio C# integration
- RenderDoc Integration
- Editor Analytics

Unity User Manual (2019.1) / Working in Unity / Editor Features / Project Sett

Time

The **Time** settings (menu: **Edit > Project Settings**, then the *Time_* category) lets yo

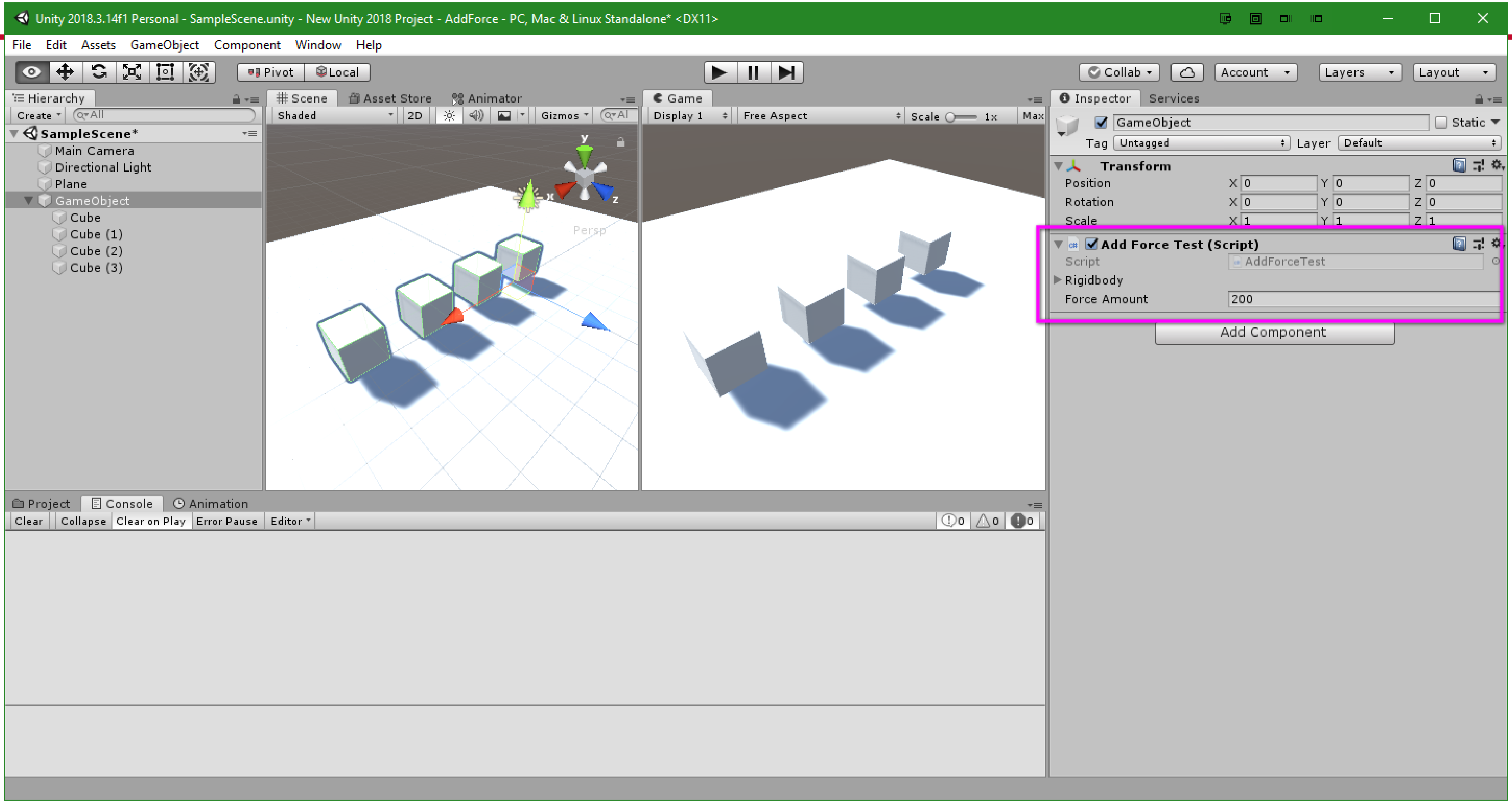
Time

Fixed Timestep	0.02
Maximum Allowed Timestep	0.1
Time Scale	1
Maximum Particle Timestep	0.03

Properties

Property:	**Function:**
Fixed Timestep	A framerate-independent interval that dictates when physics

Demo



Unity 2018.3.14f1 Personal - SampleScene.unity - New Unity 2018 Project - AddForce - PC, Mac & Linux Standalone* <DX11>

GameObject Static
Tag Untagged Layer Default

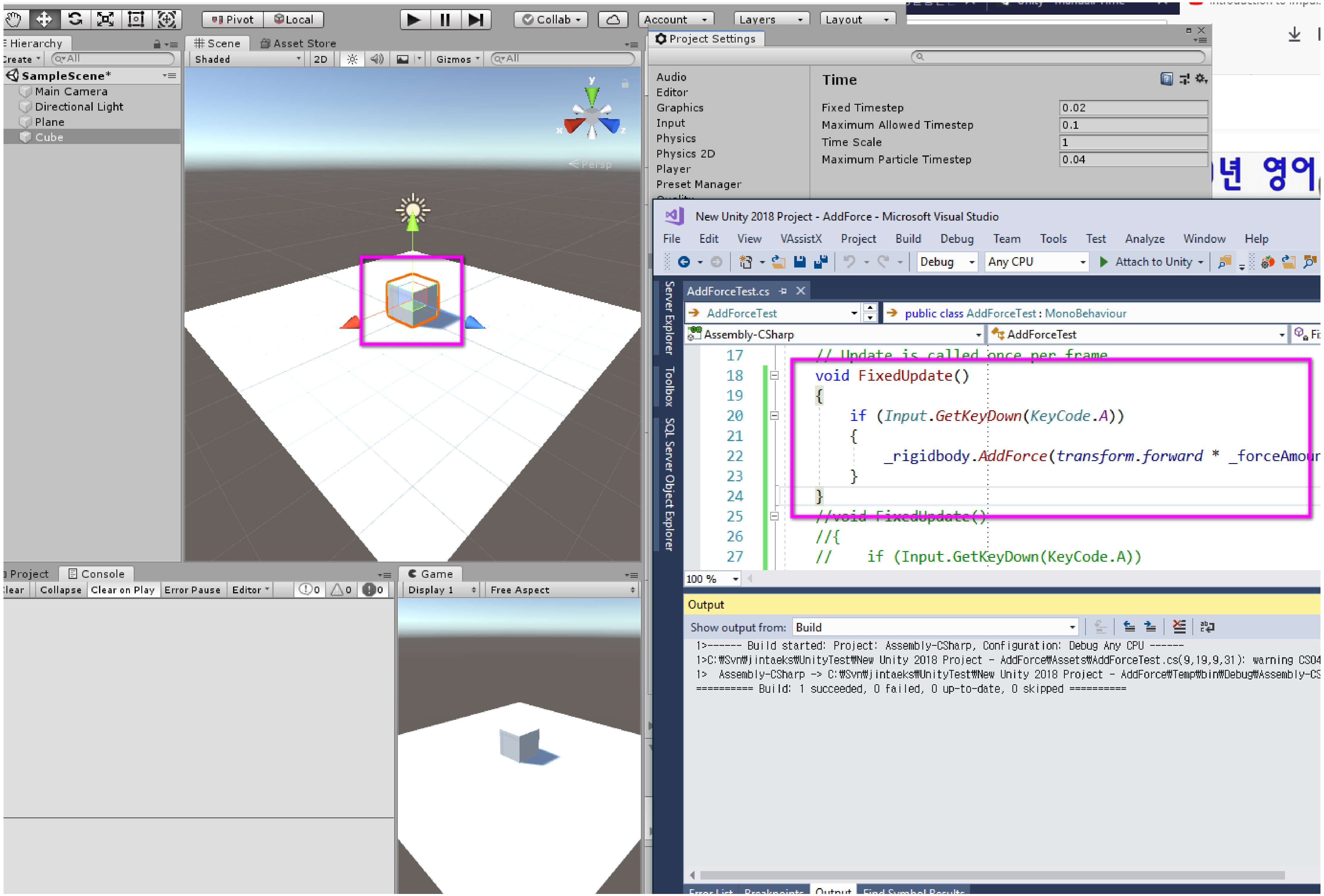
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Add Force Test (Script)
Script AddForceTest
Rigidbody
Force Amount 200

Add Component

```
public class AddForceTest : MonoBehaviour
{
    public Rigidbody[] _rigidbody;
    public float _forceAmount = 100;

    void Start()
    {
        _rigidbody = new Rigidbody[4];
        int childindex = 0;
        foreach (Transform child in transform)
        {
            _rigidbody[childindex] = child.gameObject.GetComponent<Rigidbody>();
            childindex += 1;
        }
    }

    void _ApplyForce()
    {
        _rigidbody[0].AddForce(transform.forward * _forceAmount, ForceMode.Force);
        _rigidbody[1].AddForce(transform.forward * _forceAmount * Time.fixedDeltaTime,
ForceMode.Impulse);
        Vector3 v = transform.forward * _forceAmount * Time.fixedDeltaTime / _rigidbody[1].mass;
        _rigidbody[2].AddForce(v, ForceMode.VelocityChange);
        _rigidbody[3].AddForce(v / Time.fixedDeltaTime, ForceMode.Acceleration);
    }

    // Update is called once per frame
    void FixedUpdate()
    {
        if (Input.GetKeyDown(KeyCode.A)) {
            _ApplyForce();
        }
    }
```

How to add consistent force?



Time

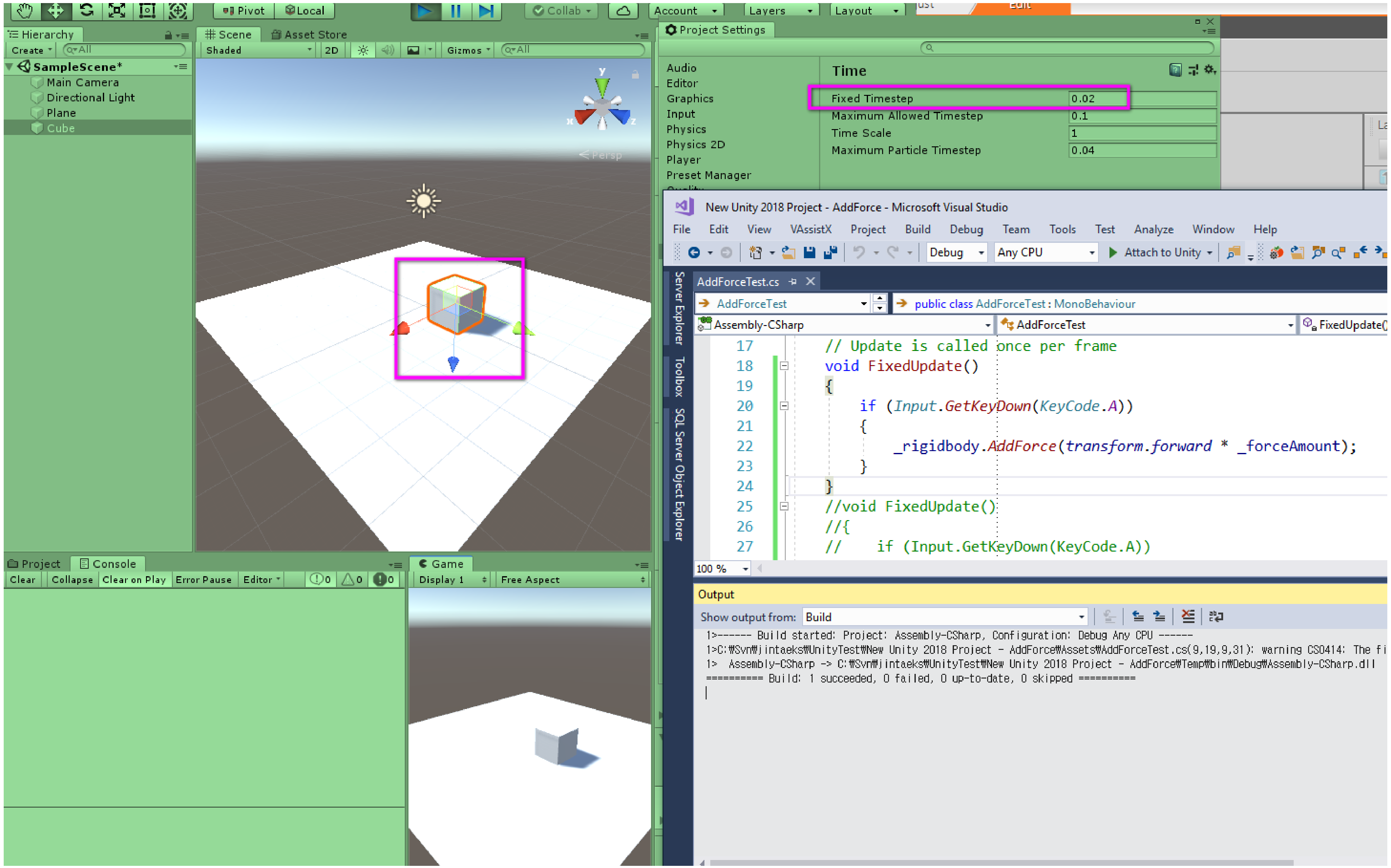
Fixed Timestep	0.02
Maximum Allowed Timestep	0.1
Time Scale	1
Maximum Particle Timestep	0.04

```
    // Update is called once per frame
    void FixedUpdate()
    {
        if (Input.GetKeyDown(KeyCode.A))
        {
            _rigidbody.AddForce(transform.forward * _forceAmour
        }
    }
    //void FixedUpdate()
    //{
    //    if (Input.GetKeyDown(KeyCode.A))
```

Output

Show output from: Build

```
1>------ Build started: Project: Assembly-CSharp, Configuration: Debug Any CPU ------
1>C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Assets₩AddForceTest.cs(9,19,9,31): warning CS04
1>  Assembly-CSharp -> C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Temp₩bin₩Debug₩Assembly-CS
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```



Time

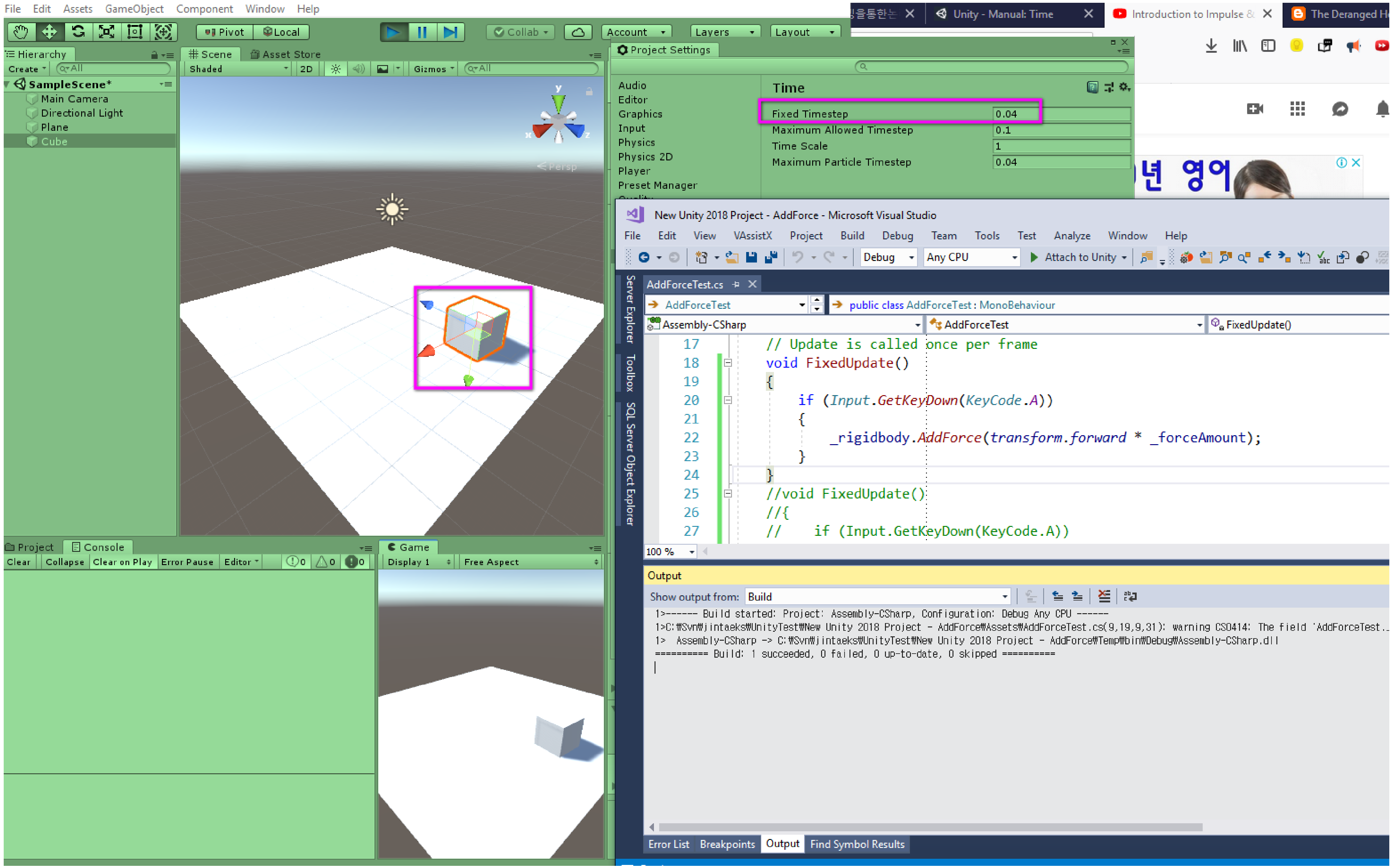
Fixed Timestep	0.02
Maximum Allowed Timestep	0.1
Time Scale	1
Maximum Particle Timestep	0.04

```
    // Update is called once per frame
    void FixedUpdate()
    {
        if (Input.GetKeyDown(KeyCode.A))
        {
            _rigidbody.AddForce(transform.forward * _forceAmount);
        }
    }
    //void FixedUpdate()
    //{
    //    if (Input.GetKeyDown(KeyCode.A))
```

Output

Show output from: Build

```
1>------ Build started: Project: Assembly-CSharp, Configuration: Debug Any CPU ------
1>C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Assets₩AddForceTest.cs(9,19,9,31): warning CS0414: The fi
1>  Assembly-CSharp -> C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Temp₩bin₩Debug₩Assembly-CSharp.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```



Time

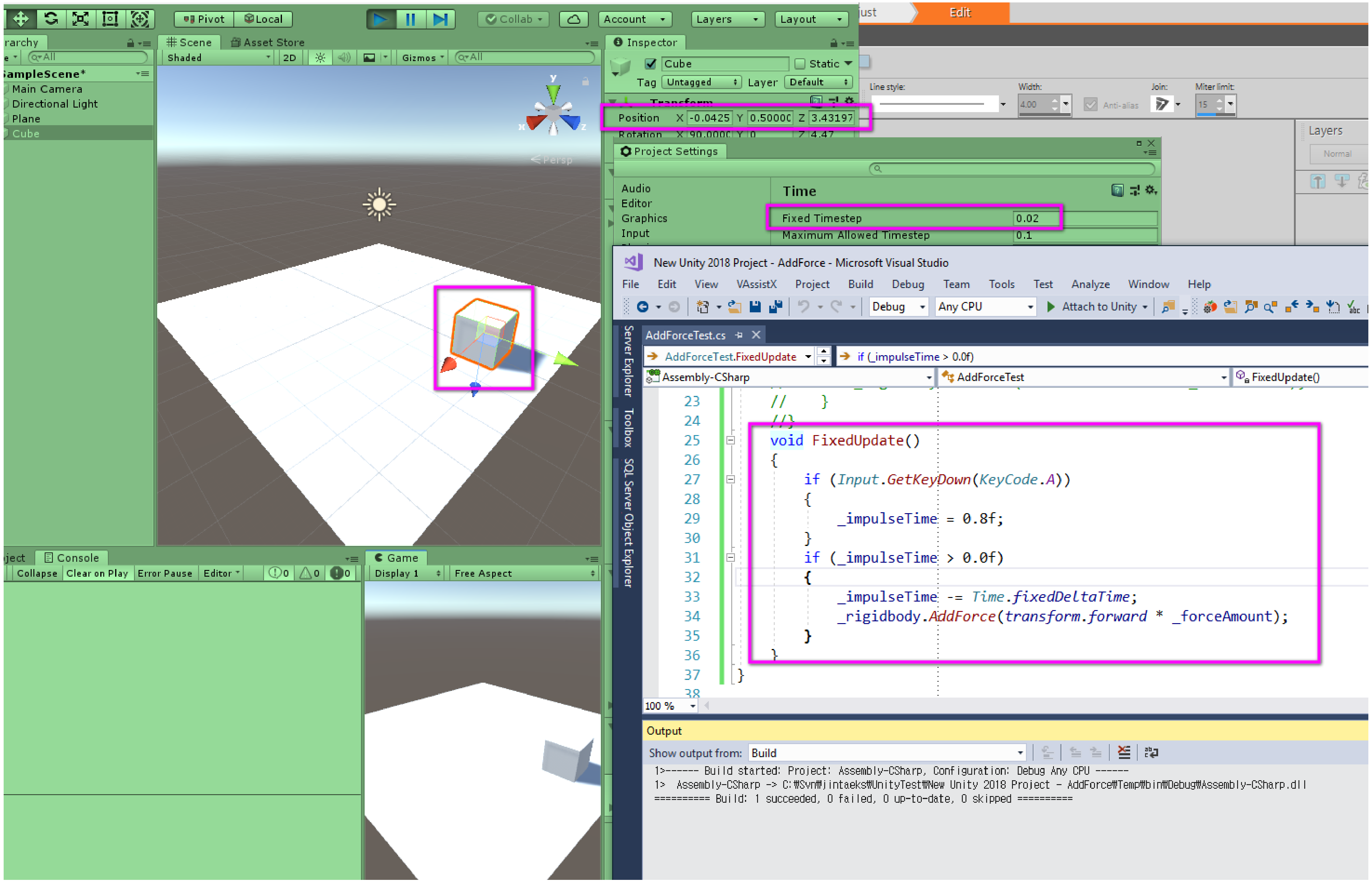
Fixed Timestep	0.04
Maximum Allowed Timestep	0.1
Time Scale	1
Maximum Particle Timestep	0.04

```
    // Update is called once per frame
    void FixedUpdate()
    {
        if (Input.GetKeyDown(KeyCode.A))
        {
            _rigidbody.AddForce(transform.forward * _forceAmount);
        }
    }
    //void FixedUpdate()
    //{
    //    if (Input.GetKeyDown(KeyCode.A))
```

Output

Show output from: Build

```
1>------ Build started: Project: Assembly-CSharp, Configuration: Debug Any CPU ------
1>C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Assets₩AddForceTest.cs(9,19,9,31): warning CS0414: The field 'AddForceTest..
1>  Assembly-CSharp -> C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Temp₩bin₩Debug₩Assembly-CSharp.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```



Cube Static
Tag Untagged Layer Default
Transform
Position X -0.0425 Y 0.50000 Z 3.43197
Rotation X 90.0000 Y 0 Z 4.47

Project Settings
Audio
Editor
Graphics
Input

Time
Fixed Timestep 0.02
Maximum Allowed Timestep 0.1

New Unity 2018 Project - AddForce - Microsoft Visual Studio

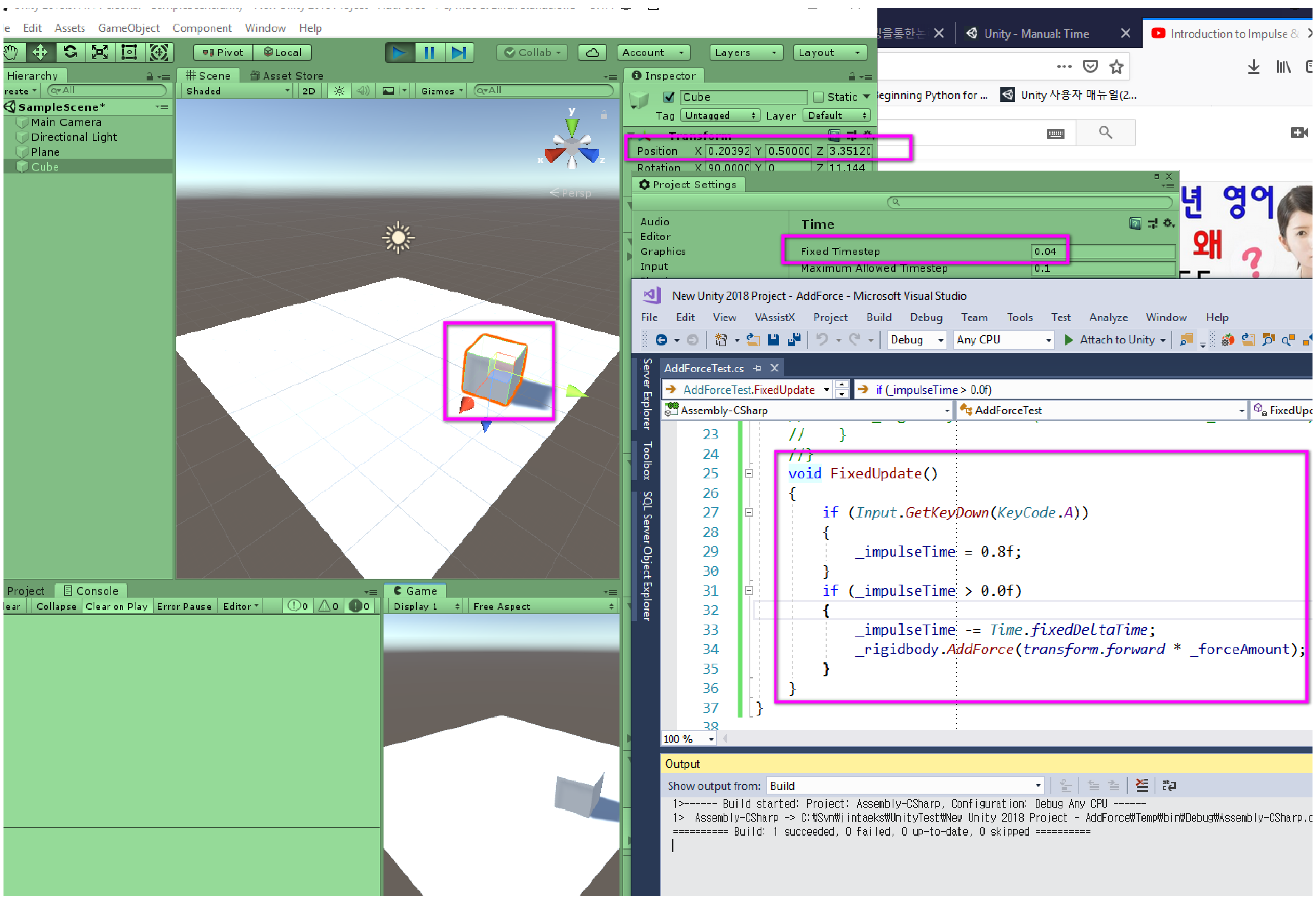
AddForceTest.cs
AddForceTest.FixedUpdate → if (_impulseTime > 0.0f)
Assembly-CSharp AddForceTest FixedUpdate()

```
    //    }
    //}
    void FixedUpdate()
    {
        if (Input.GetKeyDown(KeyCode.A))
        {
            _impulseTime = 0.8f;
        }
        if (_impulseTime > 0.0f)
        {
            _impulseTime -= Time.fixedDeltaTime;
            _rigidbody.AddForce(transform.forward * _forceAmount);
        }
    }
}
```

100 %

Output
Show output from: Build

```
1>------ Build started: Project: Assembly-CSharp, Configuration: Debug Any CPU ------
1>  Assembly-CSharp -> C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Temp₩bin₩Debug₩Assembly-CSharp.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```



Cube
Tag Untagged   Layer Default
Transform
Position X 0.20392 Y 0.50000 Z 3.35120
Rotation X 90.0000 Y 0 Z 11.144

Time
Fixed Timestep 0.04
Maximum Allowed Timestep 0.1

New Unity 2018 Project - AddForce - Microsoft Visual Studio

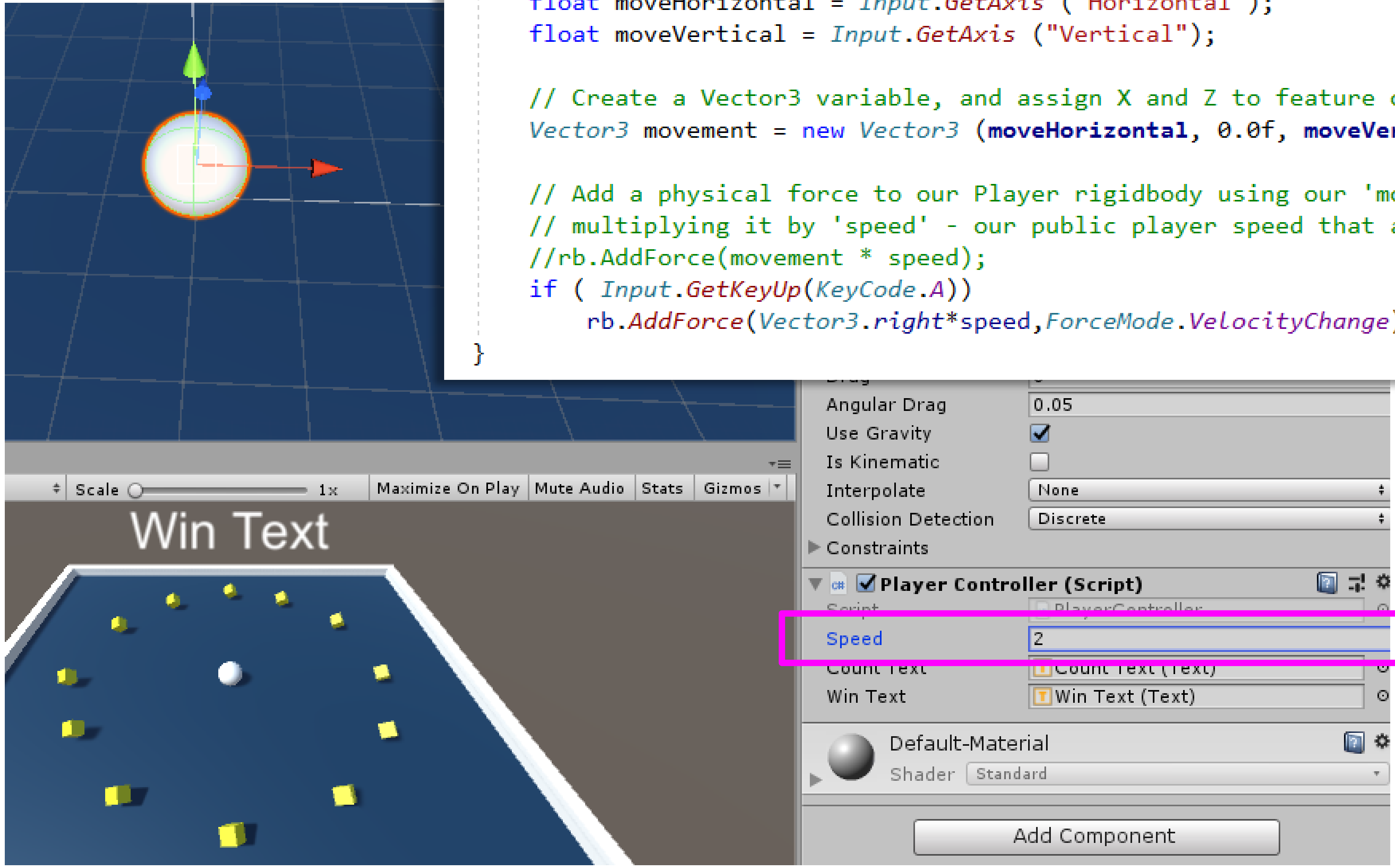
AddForceTest.cs
AddForceTest.FixedUpdate → if (_impulseTime > 0.0f)
Assembly-CSharp   AddForceTest

```
//    }
//}
void FixedUpdate()
{
    if (Input.GetKeyDown(KeyCode.A))
    {
        _impulseTime = 0.8f;
    }
    if (_impulseTime > 0.0f)
    {
        _impulseTime -= Time.fixedDeltaTime;
        _rigidbody.AddForce(transform.forward * _forceAmount);
    }
}
}
```

Output
Show output from: Build

```
1>------ Build started: Project: Assembly-CSharp, Configuration: Debug Any CPU ------
1>  Assembly-CSharp -> C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Temp₩bin₩Debug₩Assembly-CSharp.d
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Answer for the Question



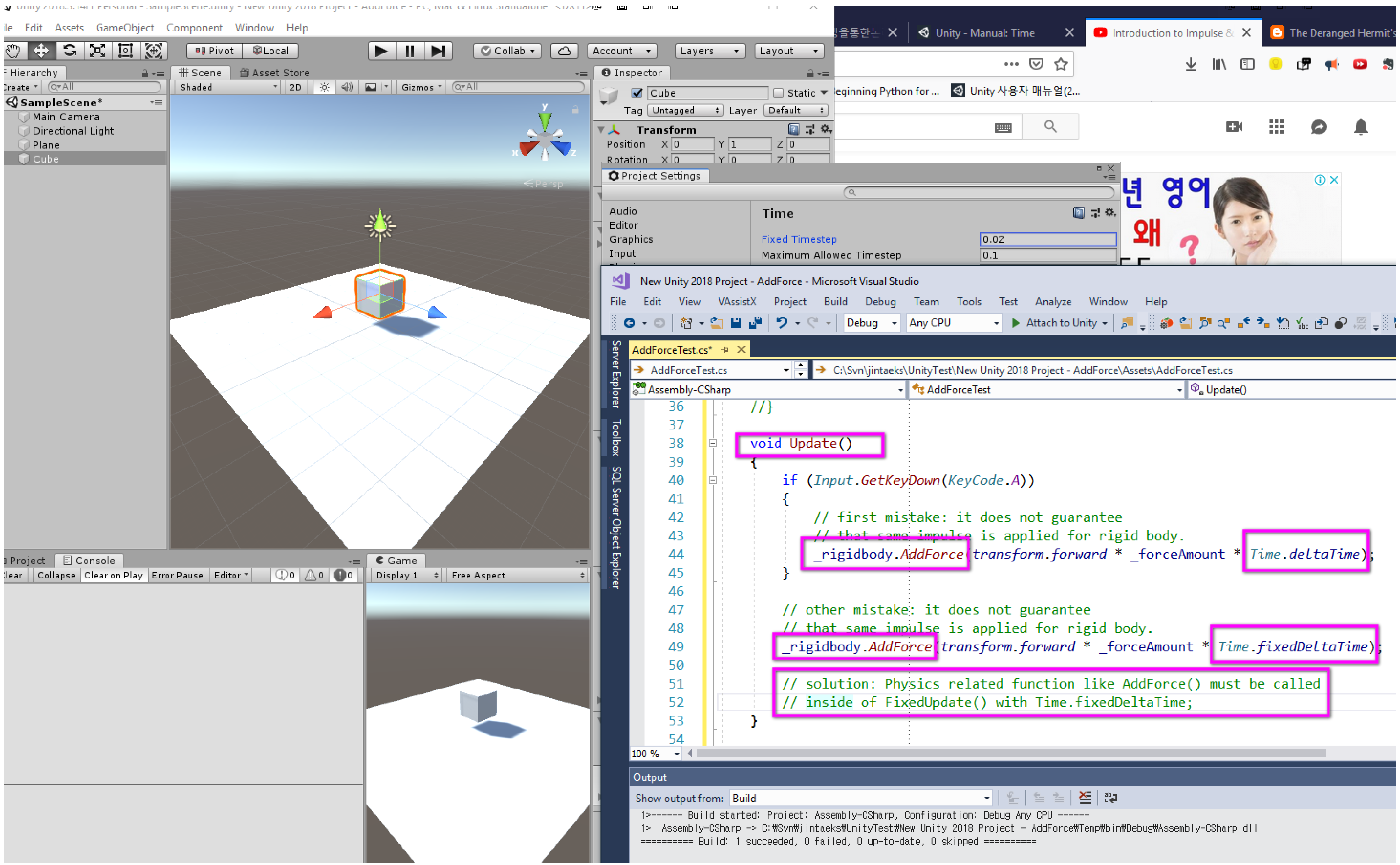
```
// Each physics step..
void FixedUpdate ()
{
    // Set some local float variables equal to the value of our Horizonta
    float moveHorizontal = Input.GetAxis ("Horizontal");
    float moveVertical = Input.GetAxis ("Vertical");

    // Create a Vector3 variable, and assign X and Z to feature our hori
    Vector3 movement = new Vector3 (moveHorizontal, 0.0f, moveVertical);

    // Add a physical force to our Player rigidbody using our 'movement'
    // multiplying it by 'speed' - our public player speed that appears
    //rb.AddForce(movement * speed);
    if ( Input.GetKeyUp(KeyCode.A))
        rb.AddForce(Vector3.right*speed,ForceMode.VelocityChange);
}
```

Angular Drag	0.05
Use Gravity	✓
Is Kinematic	
Interpolate	None
Collision Detection	Discrete
Constraints	

Player Controller (Script)

Script	PlayerController
Speed	2
Count Text	Count Text (Text)
Win Text	Win Text (Text)

Wrong Usage Example1



Time

Fixed Timestep: 0.02

Maximum Allowed Timestep: 0.1

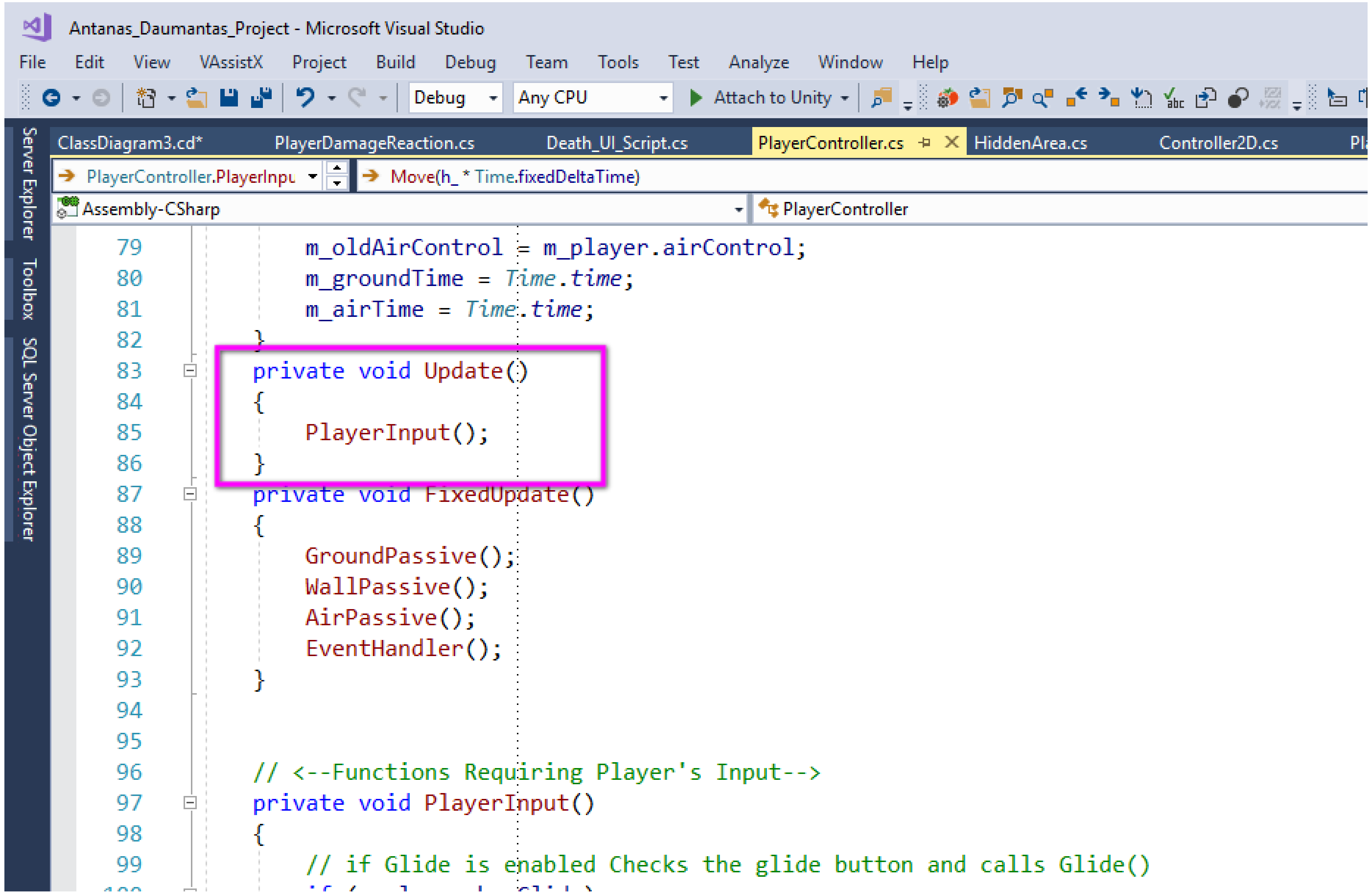
AddForceTest.cs — C:\Svn\jintaeks\UnityTest\New Unity 2018 Project - AddForce\Assets\AddForceTest.cs

```
        //}

    void Update()
    {
        if (Input.GetKeyDown(KeyCode.A))
        {
            // first mistake: it does not guarantee
            // that same impulse is applied for rigid body.
            _rigidbody.AddForce(transform.forward * _forceAmount * Time.deltaTime);
        }

        // other mistake: it does not guarantee
        // that same impulse is applied for rigid body.
        _rigidbody.AddForce(transform.forward * _forceAmount * Time.fixedDeltaTime);

        // solution: Physics related function like AddForce() must be called
        // inside of FixedUpdate() with Time.fixedDeltaTime;
    }
```

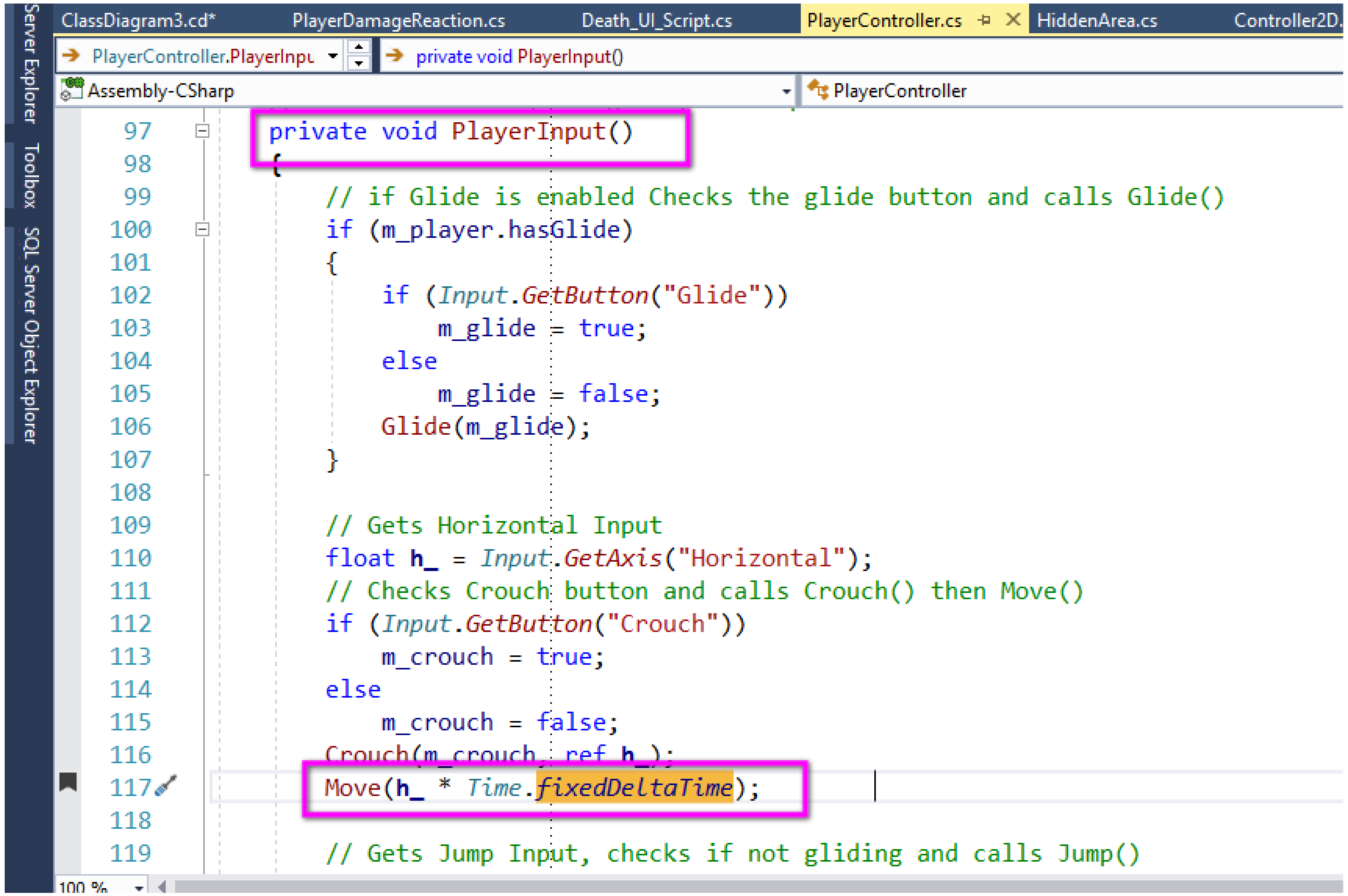
Output — Show output from: Build

```
1>------ Build started: Project: Assembly-CSharp, Configuration: Debug Any CPU ------
1>  Assembly-CSharp -> C:₩Svn₩jintaeks₩UnityTest₩New Unity 2018 Project - AddForce₩Temp₩bin₩Debug₩Assembly-CSharp.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

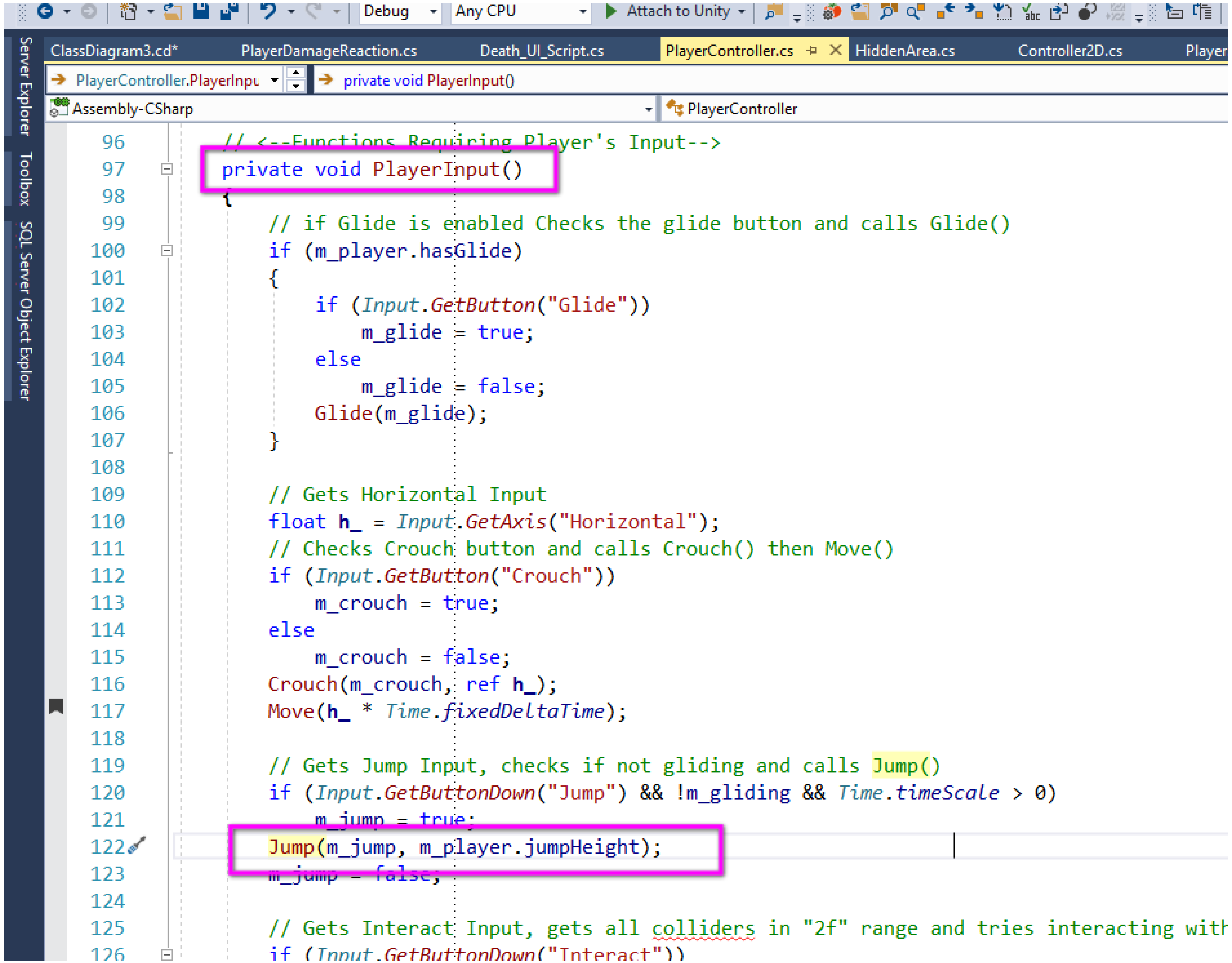
Wrong Usage Example2


Antanas_Daumantas_Project - Microsoft Visual Studio
File Edit View VAssistX Project Build Debug Team Tools Test Analyze Window Help
Debug
Any CPU
Attach to Unity
Server Explorer
Toolbox
SQL Server Object Explorer
ClassDiagram3.cd*
PlayerDamageReaction.cs
Death_UI_Script.cs
PlayerController.cs
HiddenArea.cs
Controller2D.cs
PlayerController.PlayerInpu
Move(h_ * Time.fixedDeltaTime)
Assembly-CSharp
PlayerController

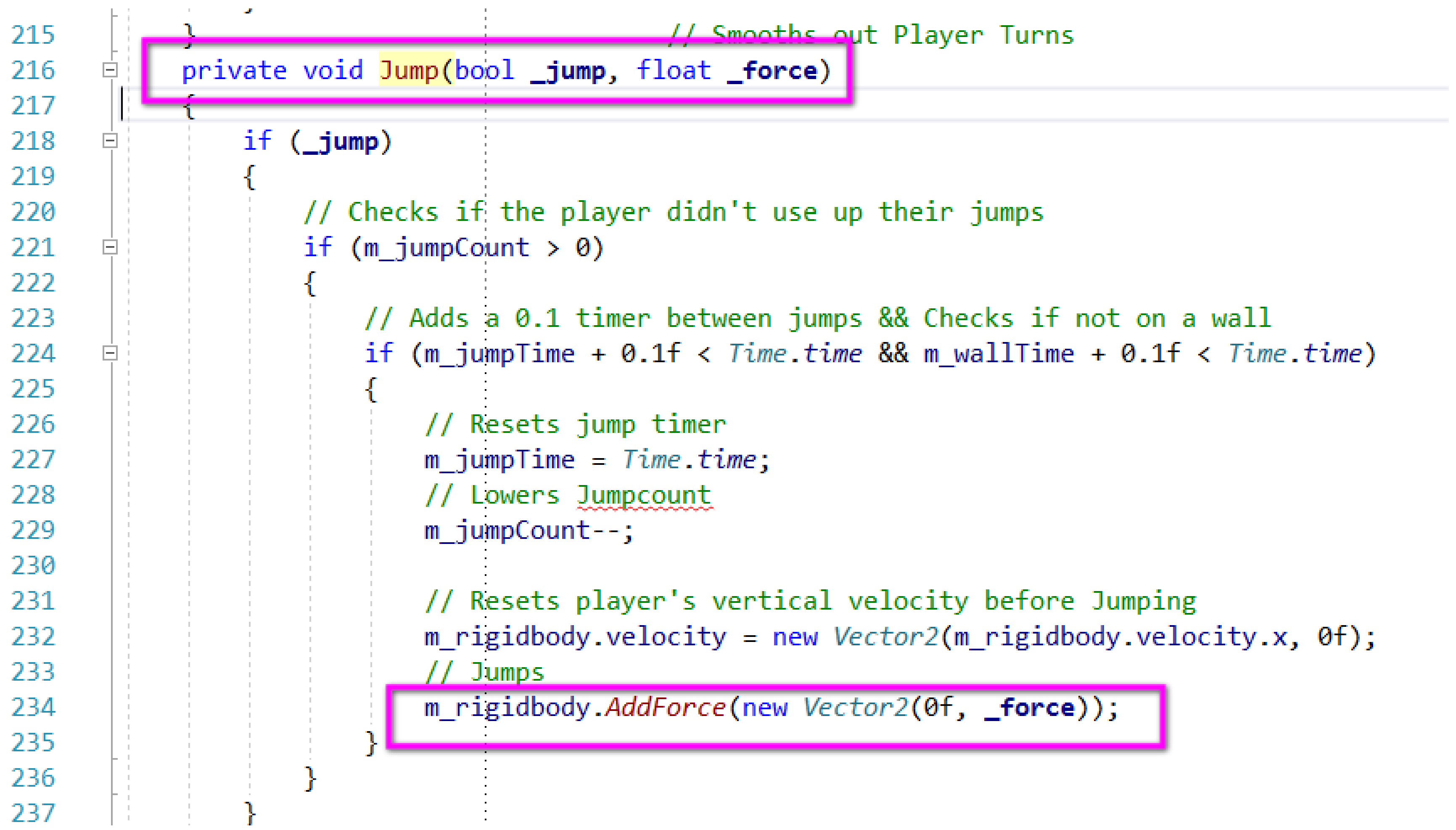
```
        m_oldAirControl = m_player.airControl;
        m_groundTime = Time.time;
        m_airTime = Time.time;
    }
    private void Update()
    {
        PlayerInput();
    }
    private void FixedUpdate()
    {
        GroundPassive();
        WallPassive();
        AirPassive();
        EventHandler();
    }


    // <--Functions Requiring Player's Input-->
    private void PlayerInput()
    {
        // if Glide is enabled Checks the glide button and calls Glide()
```



```
private void PlayerInput()
{
    // if Glide is enabled Checks the glide button and calls Glide()
    if (m_player.hasGlide)
    {
        if (Input.GetButton("Glide"))
            m_glide = true;
        else
            m_glide = false;
        Glide(m_glide);
    }

    // Gets Horizontal Input
    float h_ = Input.GetAxis("Horizontal");
    // Checks Crouch button and calls Crouch() then Move()
    if (Input.GetButton("Crouch"))
        m_crouch = true;
    else
        m_crouch = false;
    Crouch(m_crouch, ref h_);
    Move(h_ * Time.fixedDeltaTime);

    // Gets Jump Input, checks if not gliding and calls Jump()
```



```
    // <--Functions Requiring Player's Input-->
    private void PlayerInput()
    {
        // if Glide is enabled Checks the glide button and calls Glide()
        if (m_player.hasGlide)
        {
            if (Input.GetButton("Glide"))
                m_glide = true;
            else
                m_glide = false;
            Glide(m_glide);
        }

        // Gets Horizontal Input
        float h_ = Input.GetAxis("Horizontal");
        // Checks Crouch button and calls Crouch() then Move()
        if (Input.GetButton("Crouch"))
            m_crouch = true;
        else
            m_crouch = false;
        Crouch(m_crouch, ref h_);
        Move(h_ * Time.fixedDeltaTime);

        // Gets Jump Input, checks if not gliding and calls Jump()
        if (Input.GetButtonDown("Jump") && !m_gliding && Time.timeScale > 0)
            m_jump = true;
        Jump(m_jump, m_player.jumpHeight);
        m_jump = false;

        // Gets Interact Input, gets all colliders in "2f" range and tries interacting with
        if (Input.GetButtonDown("Interact"))
```



```
}                                // Smooths out Player Turns
private void Jump(bool _jump, float _force)
{
    if (_jump)
    {
        // Checks if the player didn't use up their jumps
        if (m_jumpCount > 0)
        {
            // Adds a 0.1 timer between jumps && Checks if not on a wall
            if (m_jumpTime + 0.1f < Time.time && m_wallTime + 0.1f < Time.time)
            {
                // Resets jump timer
                m_jumpTime = Time.time;
                // Lowers Jumpcount
                m_jumpCount--;

                // Resets player's vertical velocity before Jumping
                m_rigidbody.velocity = new Vector2(m_rigidbody.velocity.x, 0f);
                // Jumps
                m_rigidbody.AddForce(new Vector2(0f, _force));
            }
        }
    }
```

References

- ✓ https://en.wikipedia.org/wiki/Momentum
- ✓ https://docs.unity3d.com/ScriptReference/ForceMode.Impulse.html
- ✓ https://answers.unity.com/questions/713217/exact-difference-between-fixeddeltatime-and-deltat.html

QnA

MY BRIGHT FUTURE

DSU Dongseo University 동서대학교