



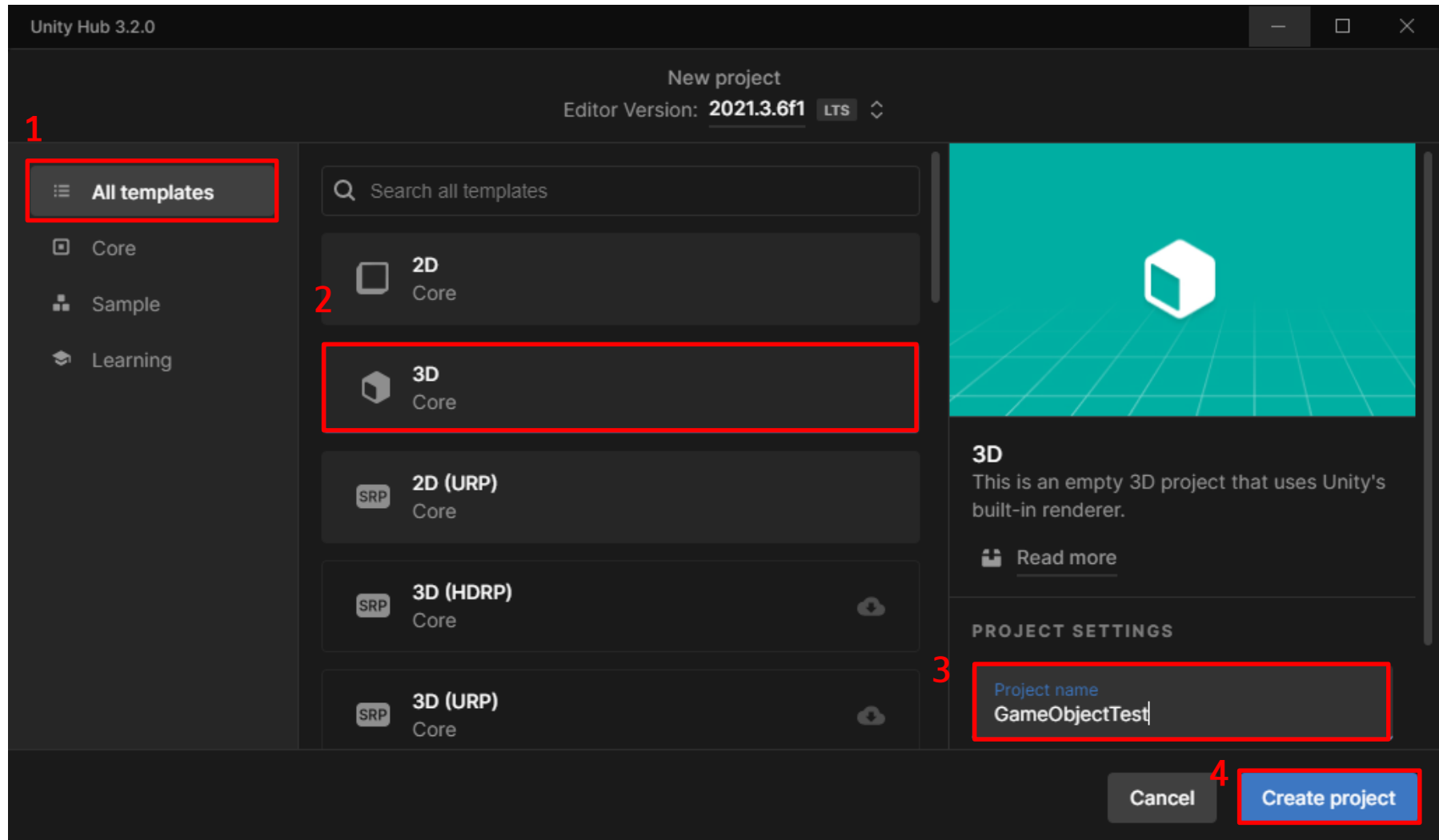
Game Programming

GameObject & Script

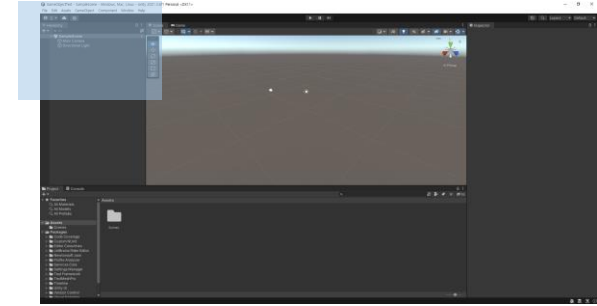
2021.3.6f1



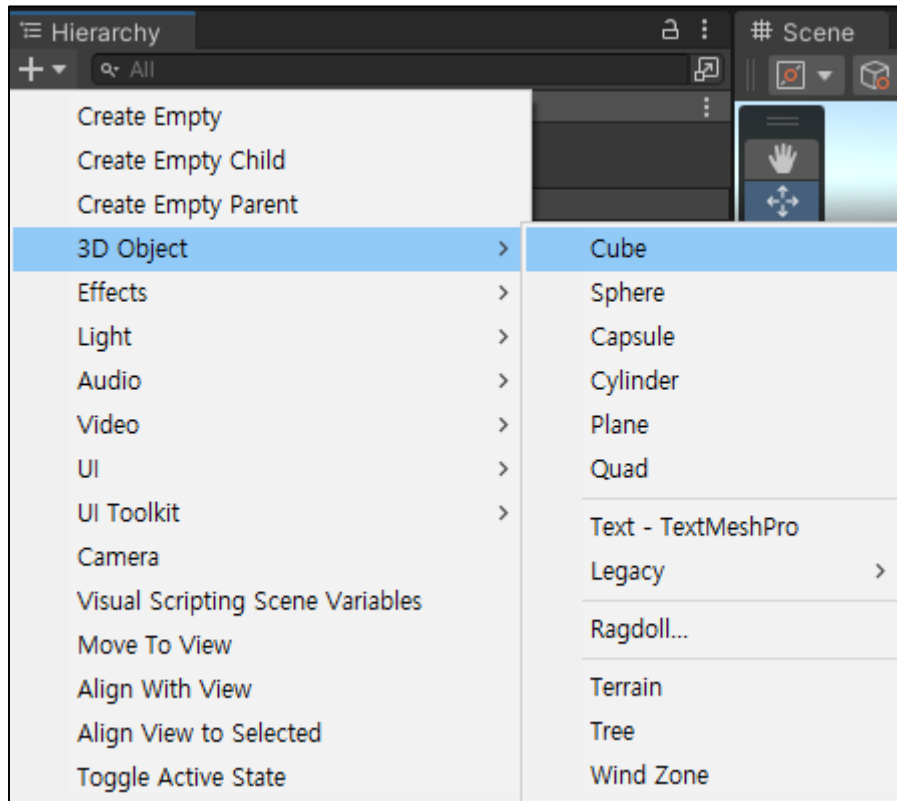
KYUNGSUNG UNIVERSITY SINCE 1955



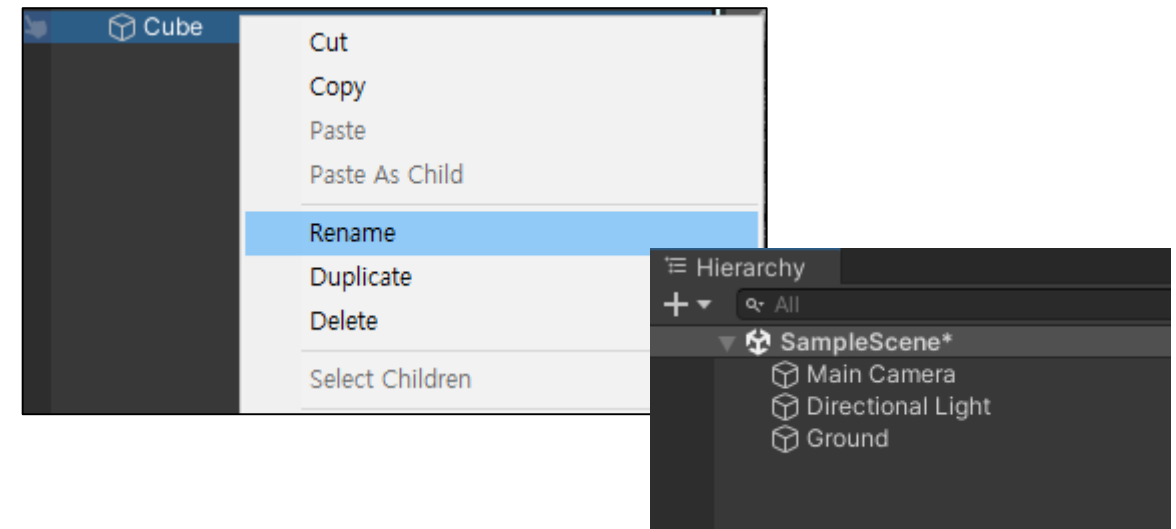
스테이지 만들기



1. Hierarchy – 3D Object – Cube

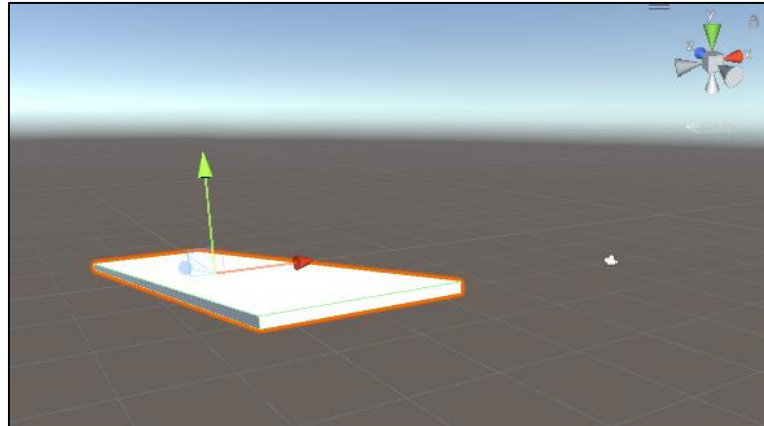
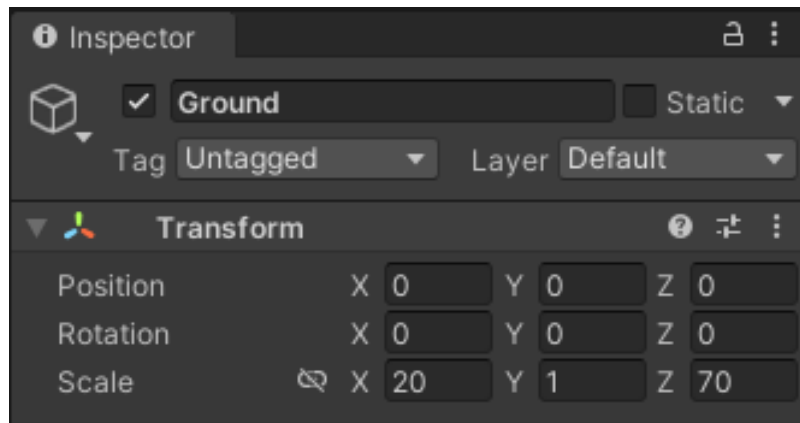
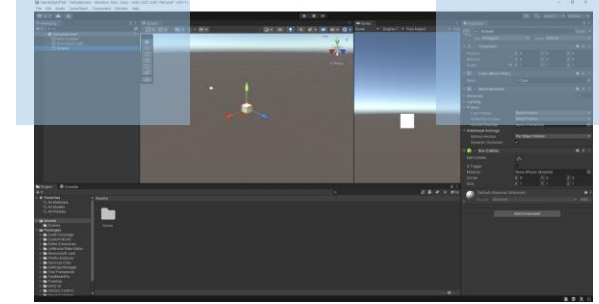


2. Hierarchy – Cube – Rename – “Ground”



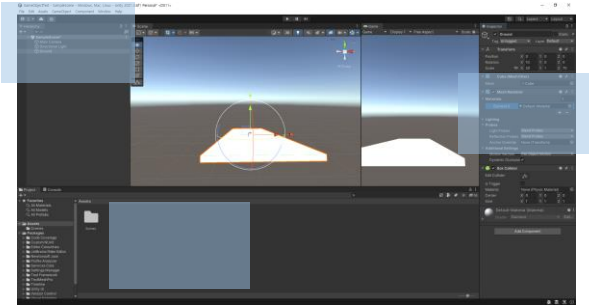
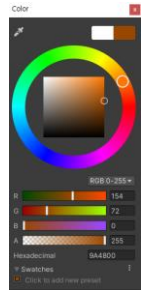
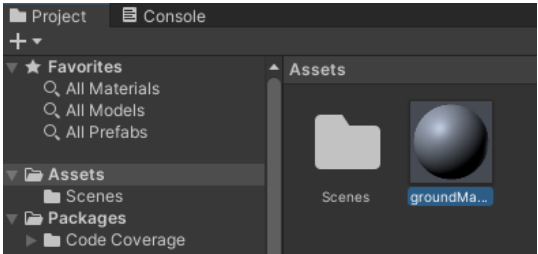
스테이지 만들기

3. Hierarchy - Main Camera - Inspector - Position 0,11,-52
4. Hierarchy - Directional Light - Inspector - Position 40,23,40 - Rotation 50,-30,1
5. Hierarchy - Ground - Inspector - Transform - Rotation 10,0,0 - Scale 20,1,70



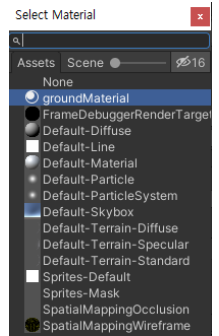
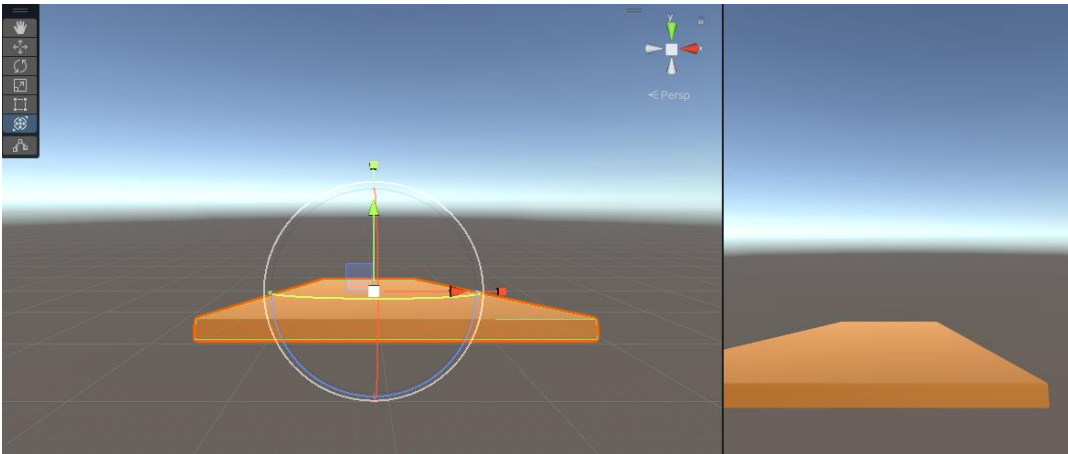
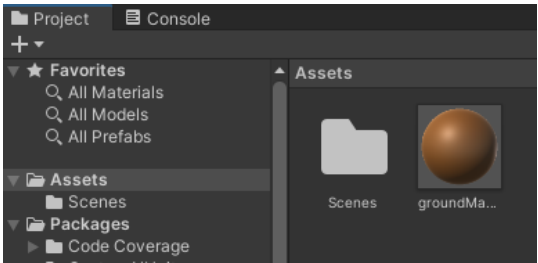
스테이지 만들기

6. Project – Assets – Create – Material – “GroundMaterial”



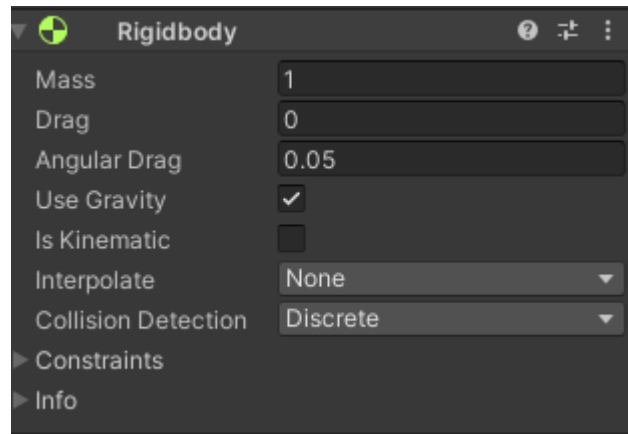
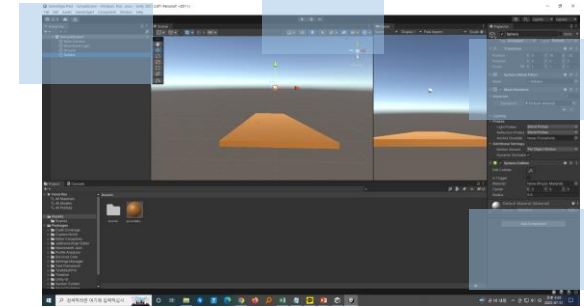
7. Project – Assets – GroundMaterial – Inspector – Main Maps – Albedo – “원하는 색 선택”

8. Project – Assets – GroundMaterial을 Hierarchy – Ground로 DnD(Drag and Drop)

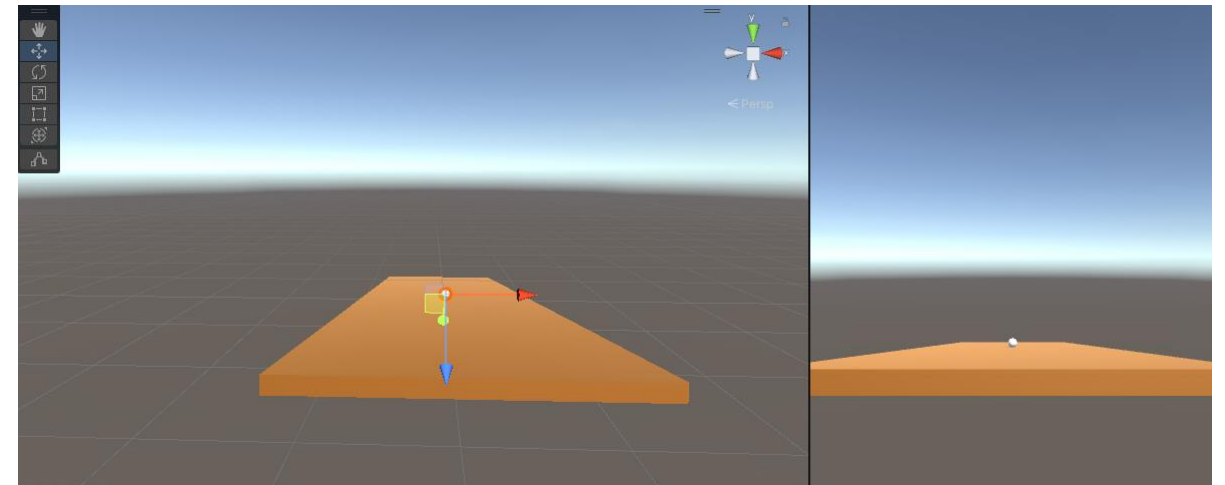
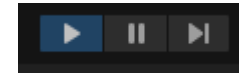


볼 만들기

1. Hierarchy – 3D Object – Sphere – Rename – “Ball”
2. Hierarchy – Ball – Inspector – Position 0,15,-30
3. Hierarchy – Ball – Inspector – Add Component – Physics – Rigidbody



4. Play



Rigidbody : 게임 오브젝트가 물리력의 영향을 받도록 하는 컴포넌트



위치 출력하기

1. Project – Assets – Create – C# script – Rename – “PrtPosition”
2. Project – Assets – PrtPosition 더블 클릭
3. Coding

```
using UnityEngine;

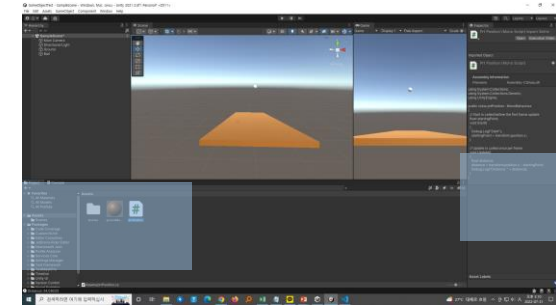
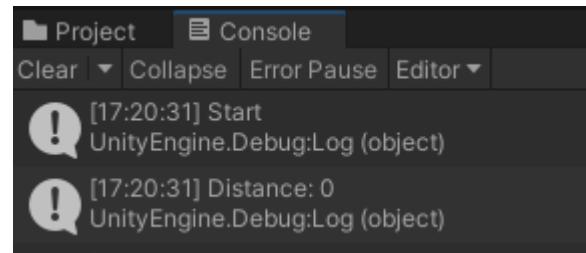
public class PrtPosition : MonoBehaviour
{
    // Start is called before the first frame update
    float startingPoint;
    void Start()
    {
        Debug.Log("Start");
        startingPoint = transform.position.z;
    }

    // Update is called once per frame
    void Update()
    {
        float distance;
        distance = transform.position.z - startingPoint;
        Debug.Log("Distance: " + distance);
    }
}
```

4. Hierarchy – Ball – Inspector – Add Component – script –
“PrtPosition” or script파일을 게임오브젝트로 드래그앤드롭

5. Play 

6. Project – Console Click



if문을 사용하여 위치 출력하기

1. Project – Assets – PrtPosition 더블 클릭

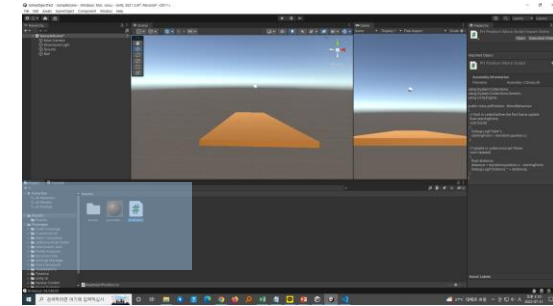
2. Coding

```
using UnityEngine;

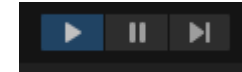
public class PrtPosition : MonoBehaviour
{
    // Start is called before the first frame update
    float startingPoint;
    void Start()
    {
        Debug.Log("Start");
        startingPoint = transform.position.z;
    }

    // Update is called once per frame
    void Update()
    {
        float distance;
        distance = transform.position.z - startingPoint;

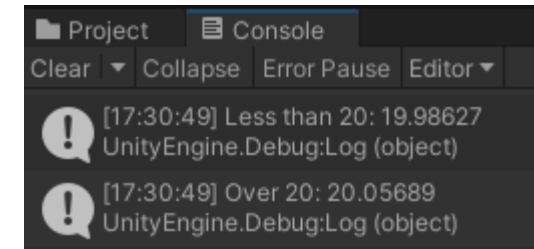
        if (distance > 40)
            Debug.Log("Over 40: " + distance);
        else if (distance > 20)
            Debug.Log("Over 20: " + distance);
        else
            Debug.Log("Less than 20: " + distance);
    }
}
```



4. Play



5. Project – Console Click



조건에 맞을 때 위치 한 번만 출력하기

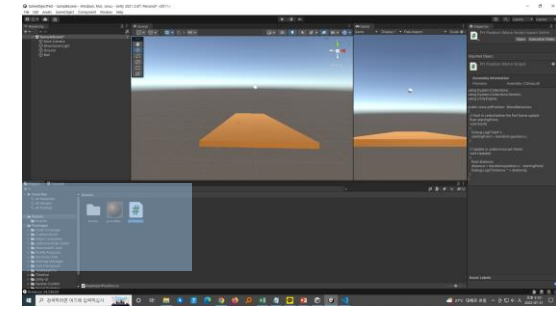
1. Project – Assets – PrtPosition 더블 클릭

2. Coding

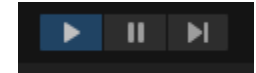
```
public class PrtPosition : MonoBehaviour
{
    // Start is called before the first frame update
    float startingPoint;
    bool isOver20 = true;
    bool isOver40 = true;
    void Start()
    {
        startingPoint = transform.position.z;
    }
}
```

```
// Update is called once per frame
void Update()
{
    float distance;
    distance = transform.position.z - startingPoint;

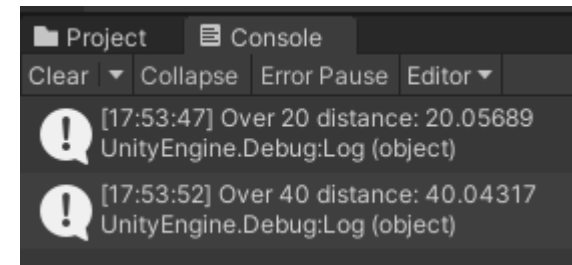
    if (distance > 40)
    {
        if (isOver40)
        {
            Debug.Log("Over 40 distance: " + distance);
            isOver40 = false;
        }
    }
    else if (distance > 20)
    {
        if (isOver20)
        {
            Debug.Log("Over 20 distance: " + distance);
            isOver20 = false;
        }
    }
}
}
```



3. Play



4. Project – Console Click



볼의 반지름 변경하기

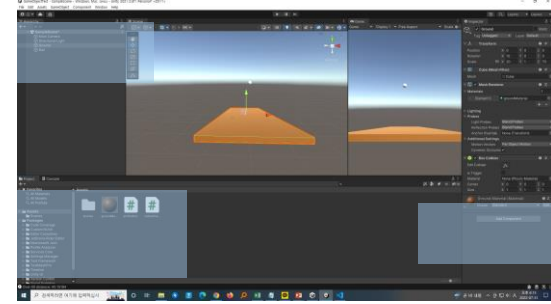
1. Project – Assets – Create – C# script – Rename – “RadiusChange”
2. Project – Assets – RadiusChange 더블 클릭
3. Coding

```
using UnityEngine;

public class RadiusChange : MonoBehaviour
{
    SphereCollider myCollider = new SphereCollider();
    // Start is called before the first frame update
    void Start()
    {
        Rigidbody myRigidbody = GetComponent<Rigidbody>();
        Debug.Log("UseGravity: " + myRigidbody.useGravity);
        myCollider = GetComponent<SphereCollider>();
    }

    // Update is called once per frame
    void Update()
    {
        myCollider.radius = myCollider.radius + 0.01f;
    }
}
```

4. Hierarchy – Ball – Inspector – Add Component – script – “RadiusChange”
5. Play 
6. Hierarchy – Ball – Inspector – RadiusChange(Script) – Kebab Menu – Remove Component



오브젝트 따라가는 카메라

1. Project – Assets – Create – C# script – Rename – “CameraFollow”
2. Project – Assets – CameraFollow 더블 클릭
3. Coding

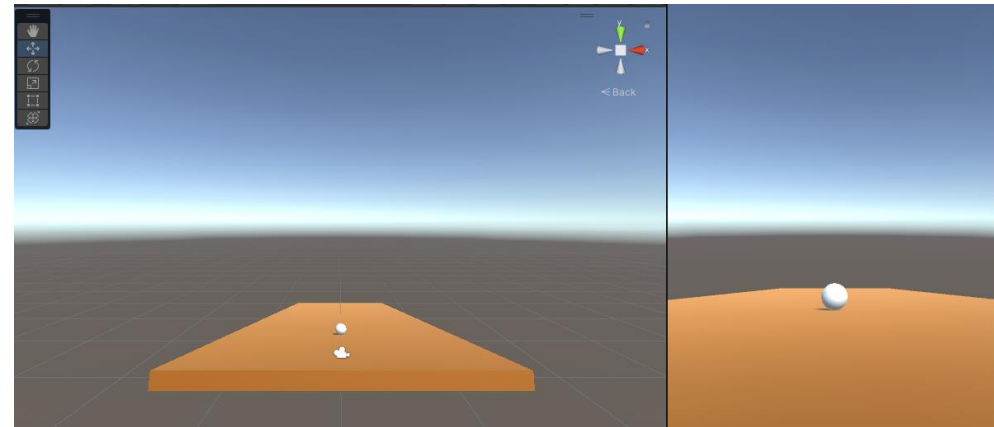
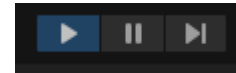
```
using UnityEngine;

public class CameraFollow : MonoBehaviour
{
    GameObject ball;
    // Start is called before the first frame update
    void Start()
    {
        ball = GameObject.Find("Ball");
    }

    // Update is called once per frame
    void Update()
    {
        transform.position = new Vector3(0,
                                           ball.transform.position.y + 3,
                                           ball.transform.position.z - 14);
    }
}
```

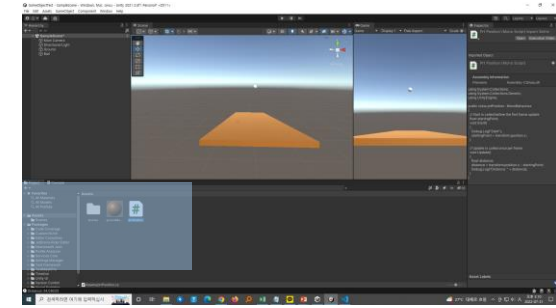
4. Hierarchy – Main Camera – Inspector – Add Component – script – “CameraFollow”

5. Play



Ground 움직이기(←, → 키 입력)

1. Project – Assets– Create – C# script – Rename – “GroundMove”
2. Project – Assets – Assets – GroundMove 더블 클릭
3. Coding



```
using UnityEngine;

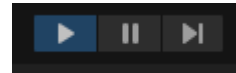
public class GroundMove : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

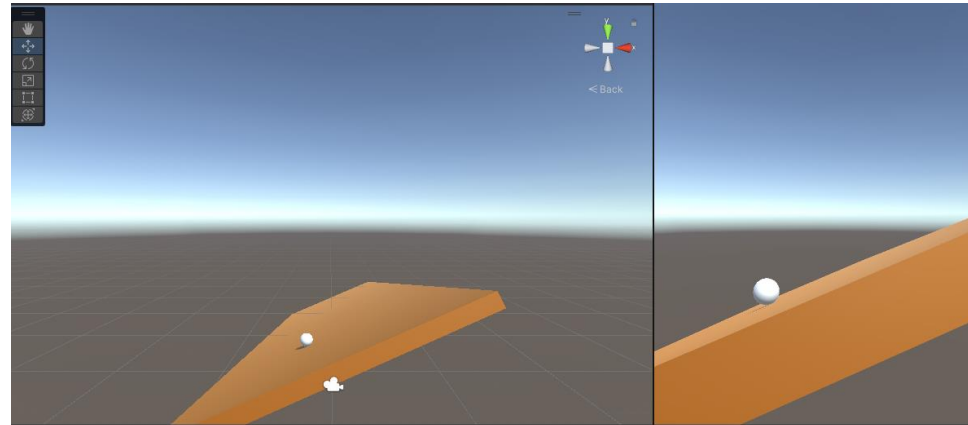
    // Update is called once per frame
    void Update()
    {
        //Debug.Log(Input.GetAxis("Horizontal"));
        //Debug.Log(Input.GetAxis("Vertical"));
        float zRotation = transform.localEulerAngles.z;
        zRotation = zRotation - Input.GetAxis("Horizontal")*0.1f;
        transform.localEulerAngles = new Vector3(10, 0, zRotation);
    }
}
```

4. Hierarchy – Ground – Inspector – Add Component – script – “GroundMove” or script파일을 게임오브젝트로 드래그앤드롭

5. Play



←, → 키 입력

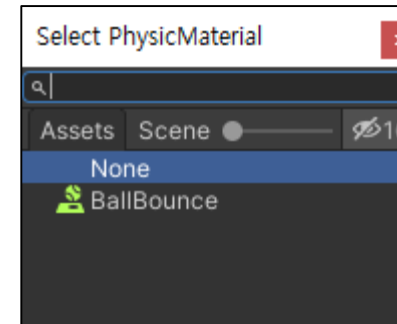
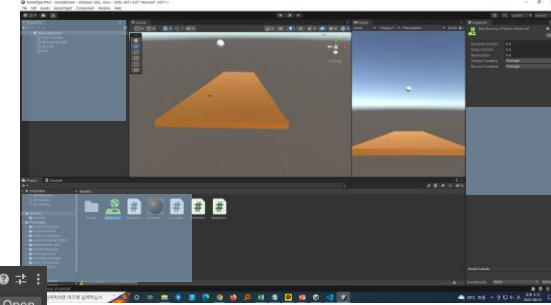
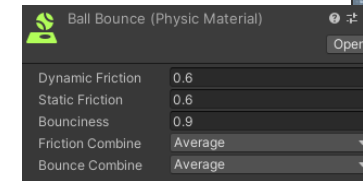
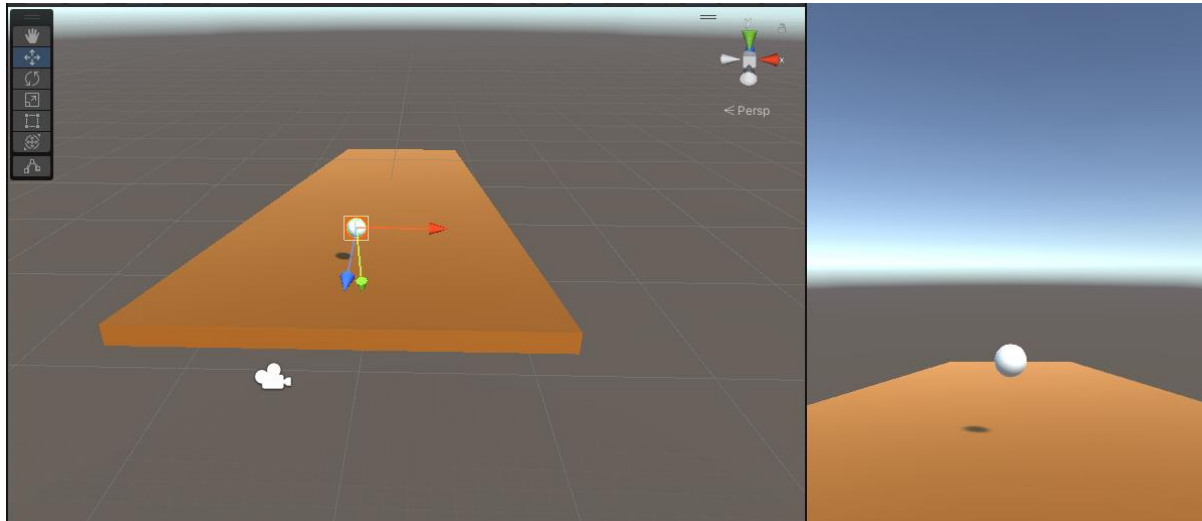
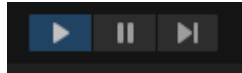


공 튀어 오르기

1. Project – Assets– Create – Physic Material – Rename – “BallBounce”
2. Project – Assets– BallBounce – Inspector – Bounciness : 0.9
3. Hierarchy – Ball – Inspector – Sphere Collider – Material –

BallBounce or BallBounce를 Ball로 DnD

4. Play



5. Hierarchy – Ground – Inspector – Box Collider – Material – BallBounce(Bounciness : 0.6)

6. Play



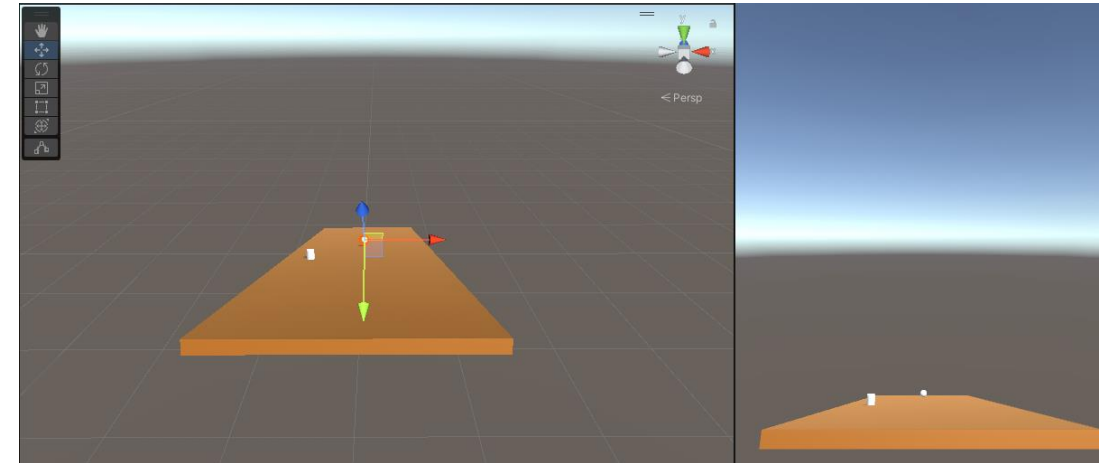
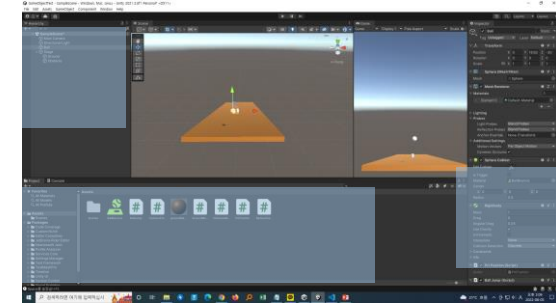
키 입력으로 공 튀어 오르기(space 키)

1. Project – Assets – Create – C# Script – Rename – BallJump
2. Project – Assets – BallJump – 더블 클릭
3. Coding
4. Hierarchy – Ball – Inspector – Add Component – Scripts – BallJump
5. Play

```
using UnityEngine;

public class BallJump : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {
    }

    // Update is called once per frame
    void Update()
    {
        if (Input.GetKeyDown(KeyCode.Space))
        {
            GetComponent<Rigidbody>().AddForce(Vector3.up * 300);
        }
    }
}
```



마우스로 Ground 움직이기

1. Project – Assets – GroundMove – 더블 클릭

2. Coding

3. Play

```
using UnityEngine;
```

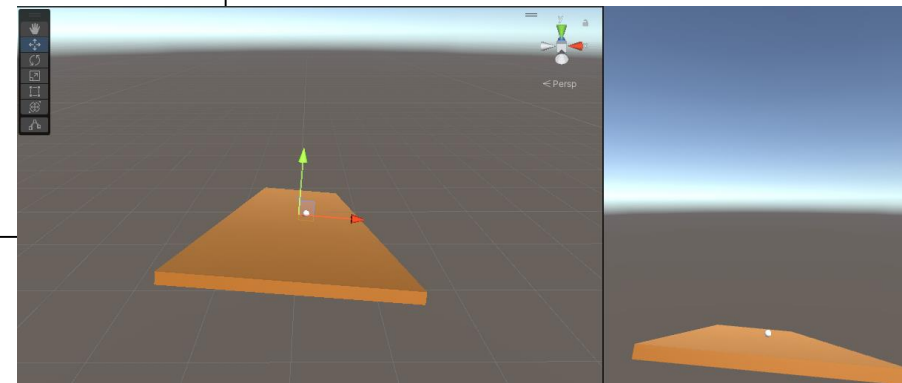
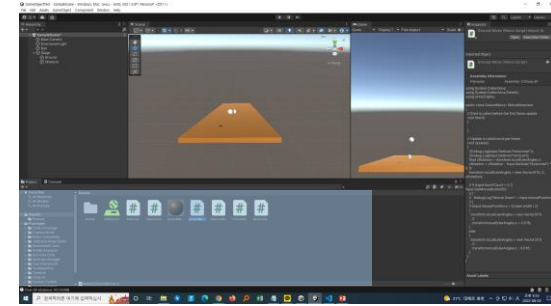
```
public class GroundMove : MonoBehaviour
```

```
{  
    // Start is called before the first frame update  
    void Start()  
    {  
  
    }  
}
```

```
// Update is called once per frame  
void Update()  
{
```

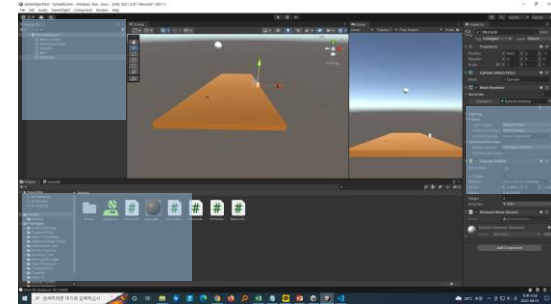
```
    //Debug.Log(Input.GetAxis("Horizontal"));  
    //Debug.Log(Input.GetAxis("Vertical"));  
    float zRotation = transform.localEulerAngles.z;  
    zRotation = zRotation - Input.GetAxis("Horizontal") * 0.1f;  
    transform.localEulerAngles = new Vector3(10, 0, zRotation);  
}
```

```
if (Input.touchCount > 0 || Input.GetMouseButton(0))  
{  
    Debug.Log("Mouse Down" + Input.mousePosition);  
    if (Input.mousePosition.x < Screen.width / 2)  
    {  
        transform.localEulerAngles = new Vector3(10  
            , 0  
            , transform.localEulerAngles.z + 0.05f);  
    }  
    else  
    {  
        transform.localEulerAngles = new Vector3(10  
            , 0  
            , transform.localEulerAngles.z - 0.05f);  
    }  
}  
}
```



장애물 설치 및 좌우 이동하기

1. Hierarchy – 3D Object – Cylinder – Rename – “Obstacle”
2. Hierarchy – Obstacle – Inspector – Position 0,2,-7
3. Play
4. Project – Assets – Create – C# Script – Rename – “ObstacleMove”
5. Project – Assets – ObstacleMove 더블클릭
6. Coding
7. Hierarchy – Obstacle – Inspector – Add Component – Script – ObstacleMove
8. Play



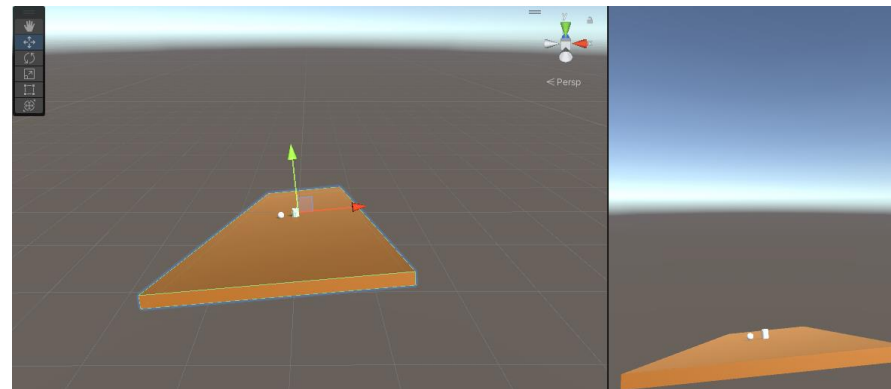
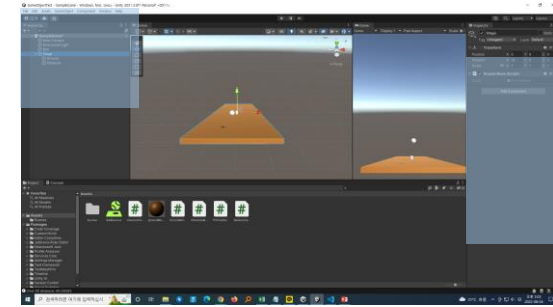
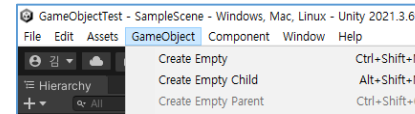
```
public class ObstacleMove : MonoBehaviour
{
    float delta = 0.01f;
    // Start is called before the first frame update
    void Start()
    {
    }
}
```

```
// Update is called once per frame
void Update()
{
    float newXPosition = transform.position.x + delta;
    transform.position = new Vector3(newXPosition, 2, -7);
    if (transform.position.x < -9) {
        delta = 0.01f;
    }
    else if (transform.position.x > 9) {
        delta = -0.01f;
    }
}
}
```



장애물과 그라운드 합치기

1. GameObject – Create Empty – Rename – “Stage”
2. Hierarchy – Stage – Inspector – Position 0,0,0 – Rotation 10,0,0
3. Hierarchy에 있는 Ground와 Obstacle을 Stage로 드래그 앤 드롭
4. Hierarchy – Stage – Ground – Inspector – Position 0,0,0 – Rotation 10,0,0
5. Hierarchy – Stage – Ground – Inspector – Ground Move(Script) – Kebab Menu – Remove Component
6. Hierarchy – Stage – Obstacle – Inspector – Position 0,2,-7 – Scale 1,1,1
7. Hierarchy – Stage – Obstacle – Inspector – Obstacle Move(Script) – Kebab Menu – Remove Component
8. Hierarchy – Stage – Inspector – Add Component – Scripts – Ground Move
9. Play



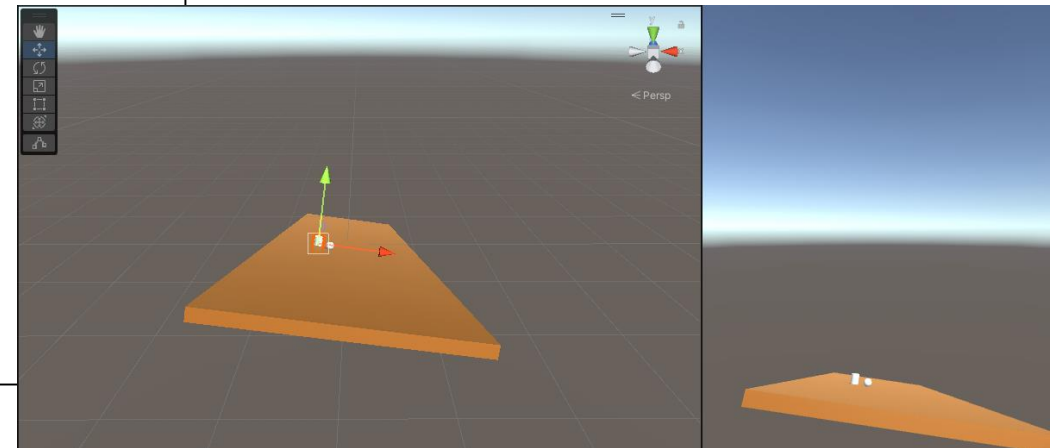
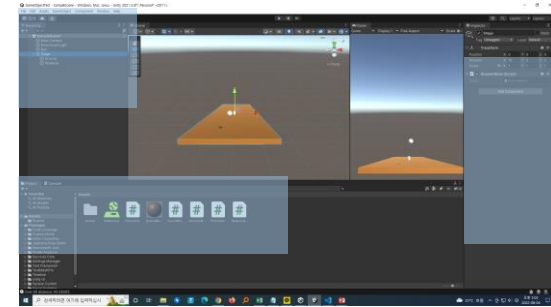
장애물과 그라운드 같이 움직이기

1. Hierarchy – Stage – Obstacle – Inspector – Add Component – Scripts – Obstacle Move
2. Play
3. Project – Assets – ObstacleMove 더블클릭
4. Coding
5. Play

using UnityEngine;

```
public class ObstacleMove : MonoBehaviour{
    float delta = 0.01f;
    // Start is called before the first frame update
    void Start() {
    }
    // Update is called once per frame
```

```
// Update is called once per frame
void Update()
{
    float newXPosition = transform.localPosition.x + delta;
    transform.localPosition = new Vector3(newXPosition,
    transform.localPosition.y,
    transform.localPosition.z);
    if (transform.localPosition.x < -9)
    {
        delta = 0.01f;
    }
    else if (transform.localPosition.x > 9)
    {
        delta = -0.01f;
    }
}
}
```



position(global position)을 localPosition으로 변경
Stage에 대한 Obstacle의 상대적 위치



C# Method & parameter

1. Project – Assets – ObstacleMove 더블클릭

2. Coding

3. Play

```
using UnityEngine;
```

```
public class ObstacleMove : MonoBehaviour
```

```
{
```

```
    void TestMethod(string name, int a)
```

```
    {
```

```
        float distance = Vector3.Distance(GameObject.Find(name).transform.position, transform.position);  
        Debug.Log(name + "까지 거리: " + distance);
```

```
    }
```

```
    float delta = 0.01f;
```

```
    // Start is called before the first frame update
```

```
    void Start() {
```

```
    }
```

```
    // Update is called once per frame
```

```
    void Update() {
```

```
        TestMethod("Ball", 0);
```

```
        float newXPosition = transform.localPosition.x + delta;
```

```
        transform.localPosition = new Vector3(newXPosition, transform.localPosition.y,  
        transform.localPosition.z);
```

```
        if (transform.localPosition.x < -9) {  
            delta = 0.01f;
```

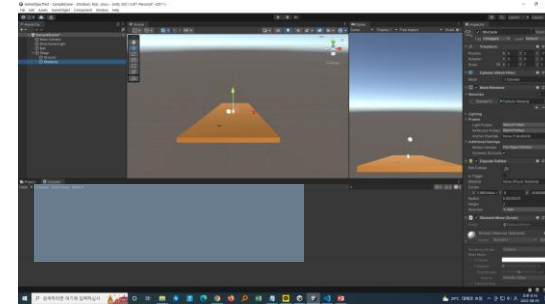
```
        }
```

```
        else if (transform.localPosition.x > 9) {  
            delta = -0.01f;
```

```
        }
```

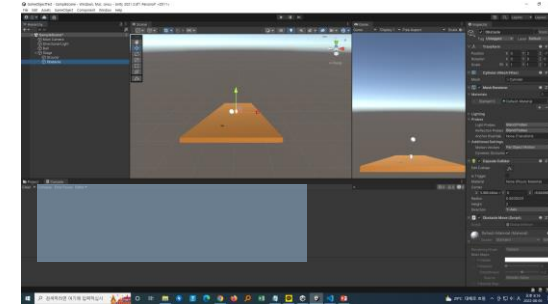
```
    }
```

```
}
```



물체의 충돌 알아보기

1. Project – Assets – ObstacleMove 더블클릭
2. Coding
3. Play



```
using UnityEngine;

public class ObstacleMove : MonoBehaviour
{
    void OnCollisionEnter(Collision collision)
    {
        //Debug.Log(collision.gameObject.name);
        Vector3 direction = transform.position -
            collision.gameObject.transform.position;
        //나의 위치에서 상대의 위치를 빼면 방향이 결정
        direction = direction.normalized * 1000; //힘 결정
        collision.gameObject.GetComponent<Rigidbody>().AddForce(direction);
    }
    float delta = 0f;
    // Start is called before the first frame update
    void Start()
    {
    }
}
```

```
// Update is called once per frame
void Update()
{
    float newXPosition = transform.localPosition.x + delta;
    transform.localPosition = new Vector3(newXPosition
        , transform.localPosition.y
        , transform.localPosition.z);
    if (transform.localPosition.x < -9)
    {
        delta = 0.01f;
    }
    else if (transform.localPosition.x > 9)
    {
        delta = -0.01f;
    }
}
```

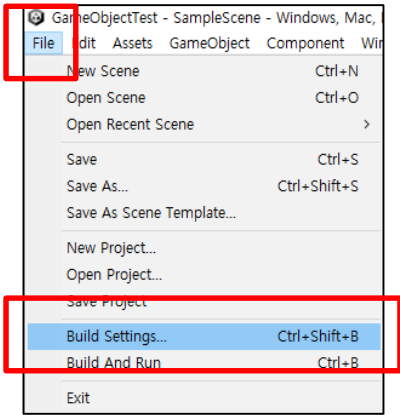
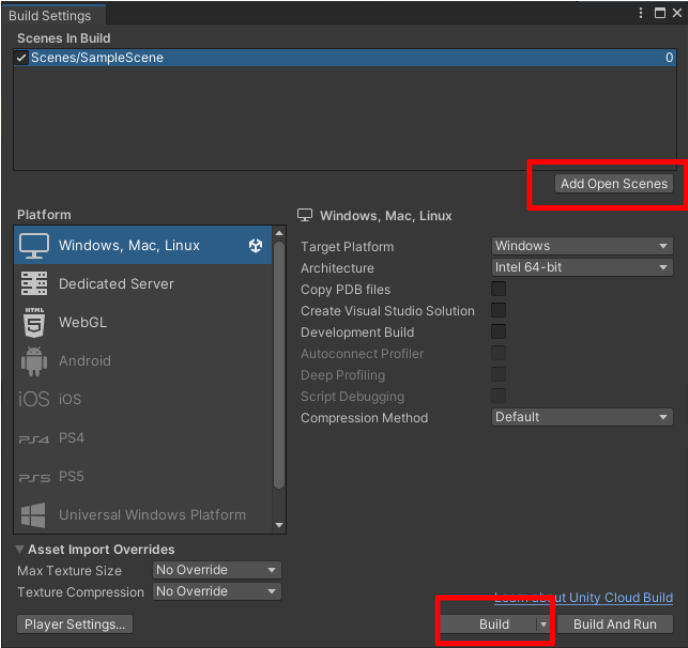
Stage 기울기 변경 실행 & 객체의 Collider 변경 실행

충돌하면 호출되는 method – OnCollisionEnter(Collision col)
col : 충돌체 정보

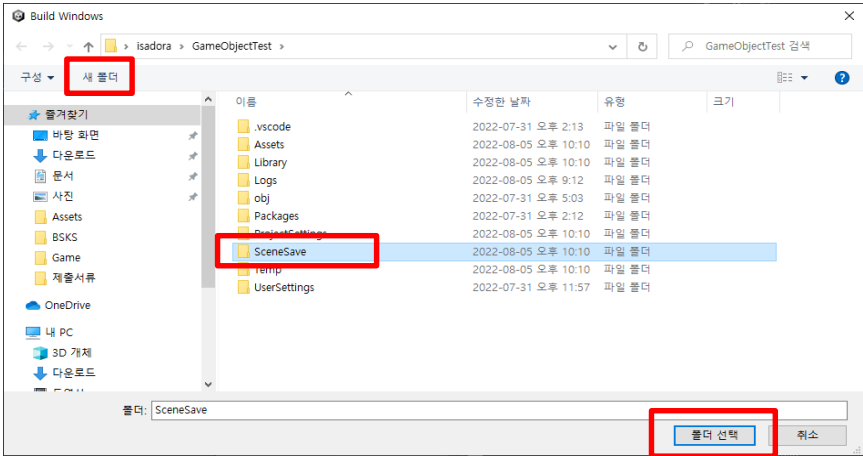


볼 낙하 후 게임 재 시작하기 - 1

- 1. File – Build Settings
- 2. Build Settings – Add Open Scenes – Build

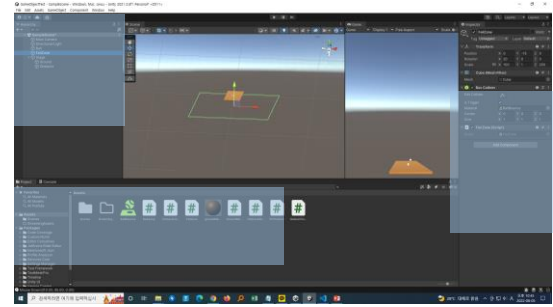


3. 새폴더 – “SceneSave” – 폴더 선택



볼 낙하 후 게임 재 시작하기 - 2

1. Hierarchy – Stage – Ground – ^C & ^V
2. Hierarchy – Stage – Ground(1) – 상위 레벨로 드래그 앤 드롭
3. Hierarchy – Ground(1) – Rename – FailZone
4. Hierarchy – FailZone – Inspector – Position 0,-15,0
5. Hierarchy – FailZone – Inspector – Rotation 20,0,0
6. Hierarchy – FailZone – Inspector – Scale 100,1,200
7. Hierarchy – FailZone – Inspector – Box Collider – Is Trigger – check
8. Hierarchy – FailZone – Inspector – Mesh Renderer – Kebab Menu – Remove Component
9. Project – Assets – Create – C# Script – Rename – FailZone
10. Hierarchy – FailZone – Inspector – Add Component – Scripts – Failzone



볼 낙하 후 게임 재 시작하기 - 3

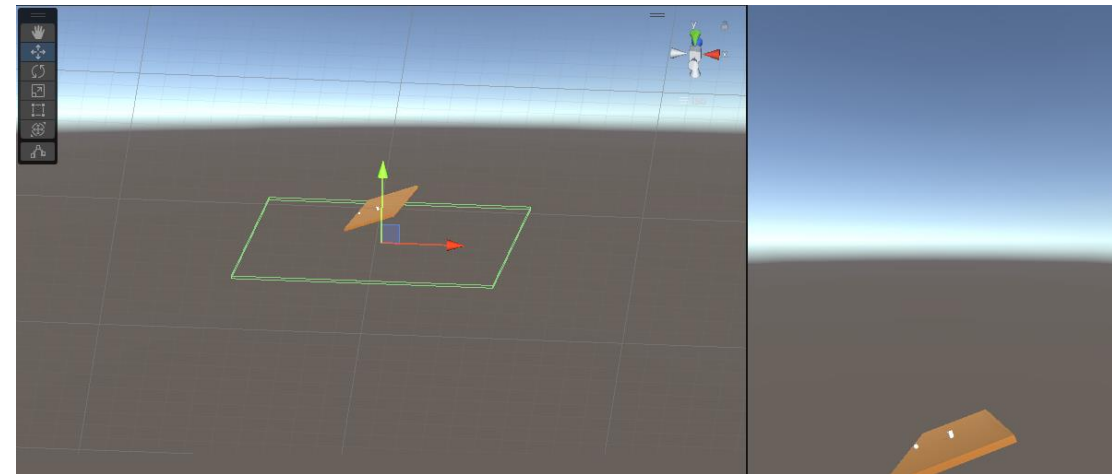
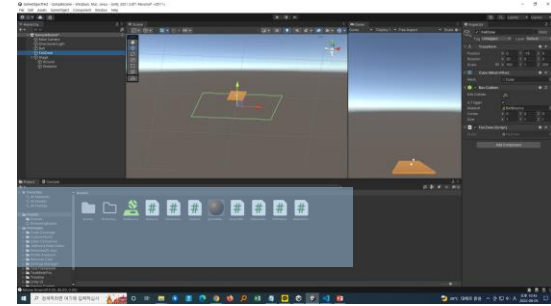
11. Project – Assets – FailZone 더블클릭

12. Coding

13. Play

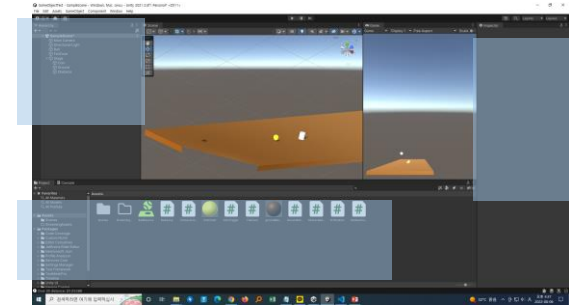
```
using UnityEngine;
using UnityEngine.SceneManagement;
public class FailZone : MonoBehaviour
{
    void OnTriggerEnter(Collider collider)
    {
        //Debug.Log(collider.gameObject.name);
        if (collider.gameObject.name == "Ball")
        {
            SceneManager.LoadScene(0);
        }
    }
    // Start is called before the first frame update
    void Start()
    {
    }

    // Update is called once per frame
    void Update()
    {
    }
}
```



코인 표시 및 코인 없애기 - 1

1. Hierarchy – 3D Object – Cylinder – Rename – “Coin”
2. Hierarchy – Coin – Inspector – Position 0,5,-10
3. Hierarchy – Coin – Inspector – Rotation 110,0,0
4. Hierarchy – Coin – Inspector – Scale 1,0.15,1
5. Project – Assets – Create – Material – Rename – “CoinColor”
6. Project – Assets – CoinColor – Inspector – Main Maps – Albedo – “Yellow 계열”
7. Project – Assets – CoinColor를 Hierarchy – Coin으로 DnD
8. Hierarchy – Coin – Inspector – Capsule Collider – Is Trigger – check
9. Hierarchy – Coin을 Hierarchy – Stage로 DnD



코인 표시 및 코인 없애기 - 2

10.Project – Assets – Create – C# Script – Rename – “CoinTrigger”

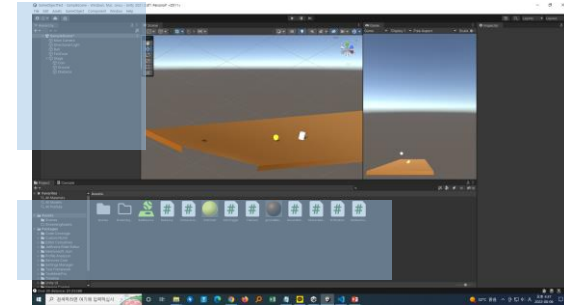
11.Coding

12.Project – Assets – CoinTrigger를 Hierarchy – Stage – Coin으로 DnD

13.Play

14.Hierarchy – Stage – Coin – ^C, ^V 5번 – 각 코인 위치 조정

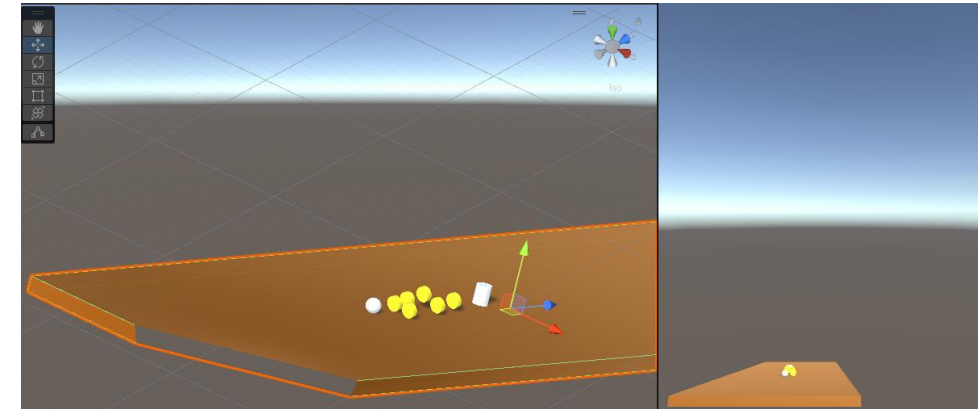
15.Play



```
using UnityEngine;
```

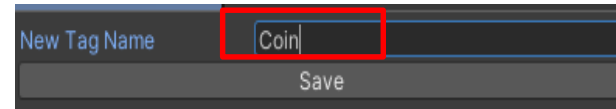
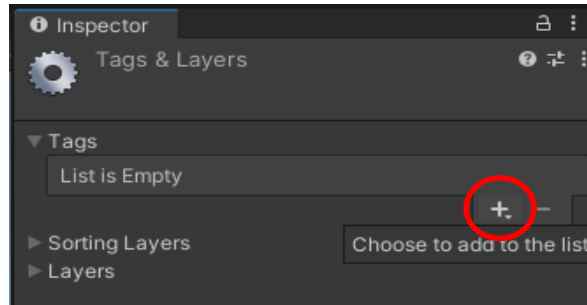
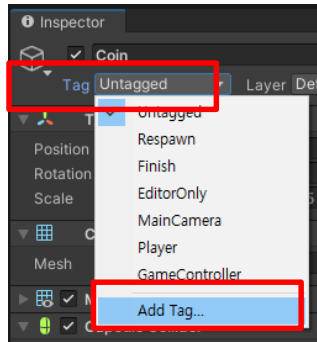
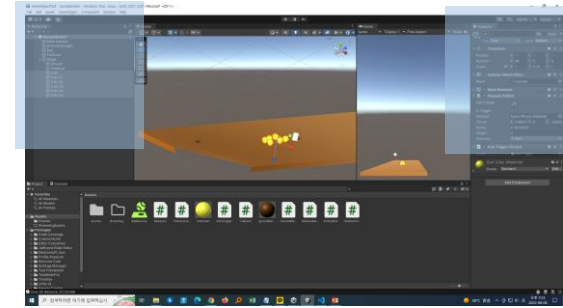
```
public class CoinTrigger : MonoBehaviour  
{  
    void OnTriggerEnter(Collider collider)  
    {  
        if (collider.gameObject.name == "Ball")  
        {  
            Destroy(gameObject);  
        }  
    }  
}
```

```
// Start is called before the first frame update  
void Start()  
{  
}  
  
// Update is called once per frame  
void Update()  
{  
}  
}
```

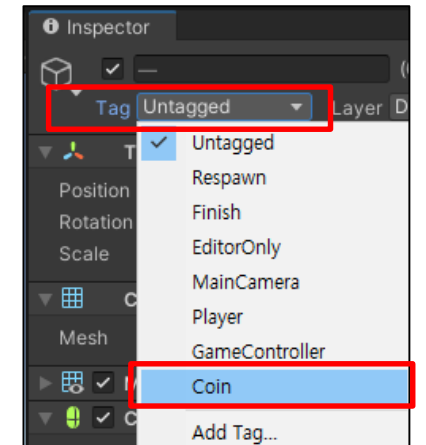
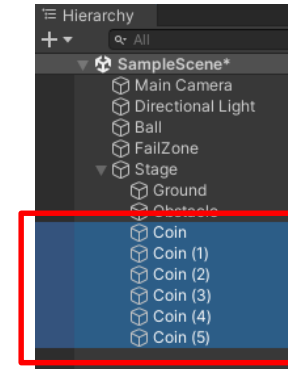


Tag 사용하기 - 1

1. Hierarchy – Stage – Coin – Inspector – Tag(UnTagged) – Add Tag
– + – “Coin” – Save



2. Hierarchy – Stage – Coin들 모두 선택(Shift or Ctrl 이용) –
Inspector – Tag(UnTagged) – Coin



Tag : 여러 개의 게임오브젝트를 하나의 이름으로 관리

Tag 사용하기 - 2

3. Project – Assets – CameraFollow – 더블클릭

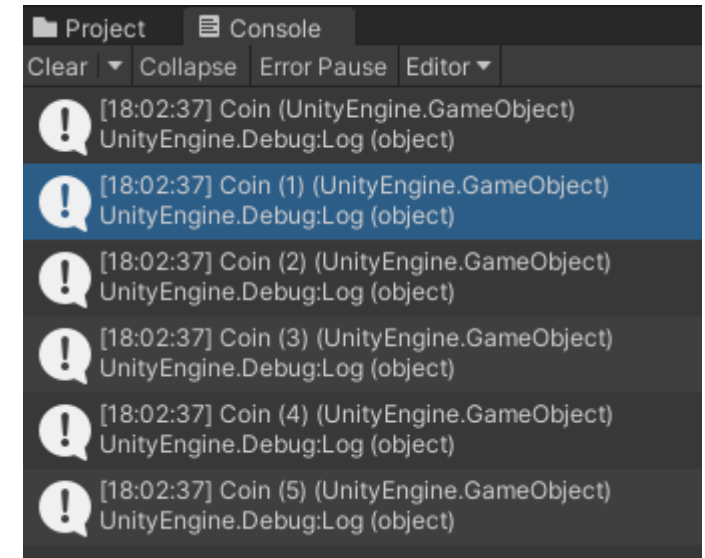
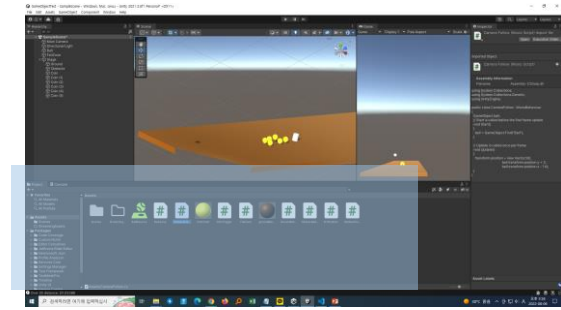
4. Coding

5. Play

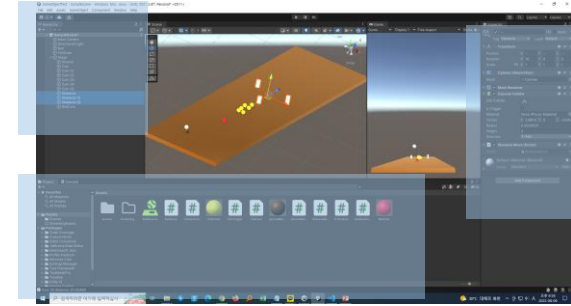
```
using UnityEngine;

public class CameraFollow : MonoBehaviour
{
    GameObject ball;
    // Start is called before the first frame update
    void Start()
    {
        ball = GameObject.Find("Ball");
        GameObject[] coins = GameObject.FindGameObjectsWithTag("Coin");
        for (int i = 0; i < coins.Length; i++)
        {
            Debug.Log(coins[i].name);
        }
    }

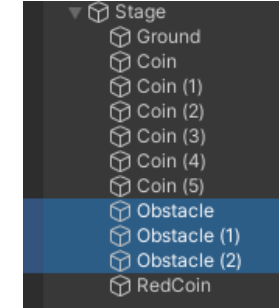
    // Update is called once per frame
    void Update()
    {
        transform.position = new Vector3(0,
                                           ball.transform.position.y + 3,
                                           ball.transform.position.z - 14);
    }
}
```



아이템 먹고 장애물 없애기 - 1



1. Hierarchy – Stage – Obstacle – Inspector – Tag(UnTagged) – Add Tag
– + – “Obstacle” – Save
2. Hierarchy – Stage – Obstacle – ^C, ^V 3번 – 자리배치
3. Hierarchy – Stage – Obstacle들 모두 선택(Shift or Ctrl 이용) – Inspector – Tag(UnTagged)
– Obstacle
4. Hierarchy – Stage – Coin – ^C, ^V – 자리 배치
5. Hierarchy – Stage – Coin(6) – Rename – “RedCoin”
6. Project – Assets – Create – Material – Rename – “RedCoin”
7. Project – Assets – RedCoin을 Hierarchy – Stage – RedCoin으로 DnD
9. Project – Assets – Create – C# Script – Rename – “RedCoinItem”
10. Project – Assets – RedCoinItem – 더블클릭



아이템 먹고 장애물 없애기 - 2

11.Coding

12.Project – Assets – RedCoinItem 을 Hierarchy – RedCoin으로 DnD

13.Play



```
using UnityEngine;
```

```
public class RedCoinItem : MonoBehaviour
```

```
{
    void OnTriggerEnter(Collider collider)
    {
        if (collider.gameObject.name == "Ball")
        {
            DestroyObstacles();
            Destroy(gameObject);
        }
    }
}
```

```
void DestroyObstacles()
```

```
{
    GameObject[] obstacles = GameObject.FindGameObjectsWithTag("Obstacle");
    for (int i = 0; i < obstacles.Length; i++)
    {
        Destroy(obstacles[i]);
    }
}
```

```
// Start is called before the first frame update
```

```
void Start()
```

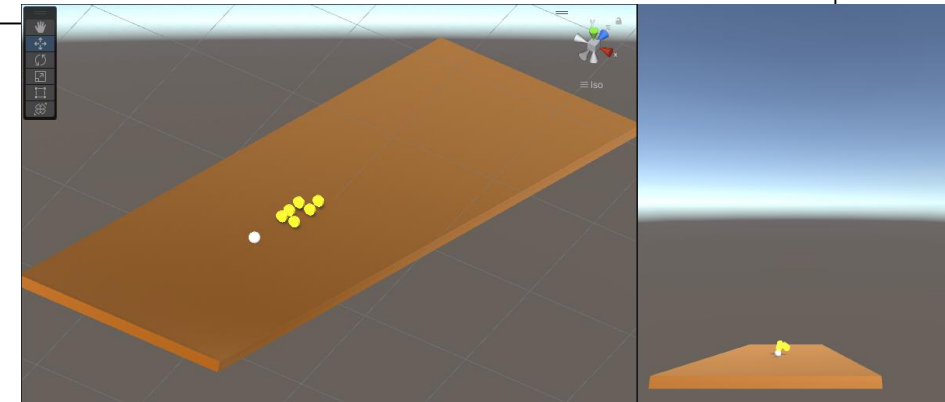
```
{
}
```

```
// Update is called once per frame
```

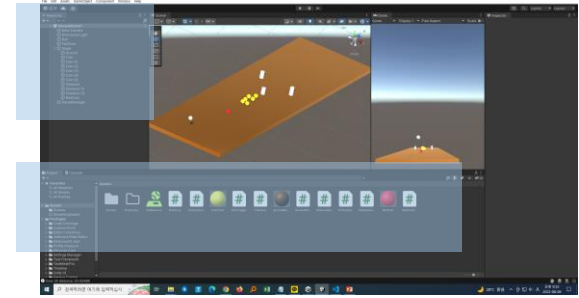
```
void Update()
```

```
{
}
```

```
}
```



GameManager - 1



1. Hierarchy – Create Empty – Rename – “GameManager”
2. Project – Assets – Create – C# Script – Rename – “GameManager”
3. Project – Assets – GameManager를 Hierarchy – GameManager로 DnD
4. Project – Assets – GameManager 더블클릭
5. Coding(코드 추가)

```
using UnityEngine.SceneManagement;

void RestartGame()
{
    SceneManager.LoadScene(0);
}
```

FailZone의 code를 GameManager로 이동
FailZone에서 GameManager의 메소드 호출

6. Project – Assets – FailZone 더블클릭

7. Coding(코드 변경)

```
void OnTriggerEnter(Collider collider)
{
    //Debug.Log(collider.gameObject.name);
    if (collider.gameObject.name == "Ball")
    {
        GameObject.Find("GameManager").SendMessage("RestartGame");
    }
}
```

8. Play

게임의 전반적인 제어를 위해 사용



GameManager - 2

9. Project – Assets – GameManager 더블클릭

10.Coding(코드 추가)

```
void RedCoinStart(){
    DestroyObstacles();
}
void DestroyObstacles()
{
    GameObject[] obstacles = GameObject.FindGameObjectsWithTag("Obstacle");
    for (int i = 0; i < obstacles.Length; i++)
    {
        Destroy(obstacles[i]);
    }
}
```

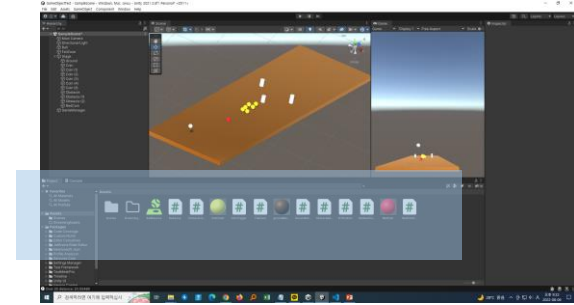
RedCoinItem에서 ^x, ^v로 가져온 코드

11.Project – Assets – RedCoinItem 더블클릭

12.Coding(코드 변경)

```
void OnTriggerEnter(Collider collider)
{
    if (collider.gameObject.name == "Ball")
    {
        GameObject.Find("GameManager").SendMessage("RedCoinStart");
        Destroy(gameObject);
    }
}
//void DestroyObstacles() 메소드 삭제
```

13.Play



획득한 동전 개수 세기

1. Project – Assets – GameManager 더블클릭

2. Coding(코드 추가)

```
int coinCount = 0;
void GetCoin()
{
    coinCount++;

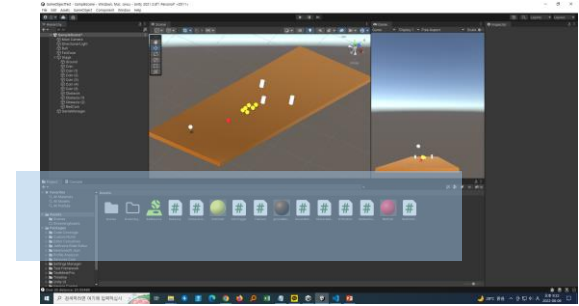
    Debug.Log("동전 : " + coinCount);
}
```

3. Project – Assets – CoinTrigger 더블클릭

4. Coding(코드 추가)

```
void OnTriggerEnter(Collider collider)
{
    if (collider.gameObject.name == "Ball")
    {
        GameObject.Find("GameManager").SendMessage("GetCoin");
        Destroy(gameObject);
    }
}
```

5. Play



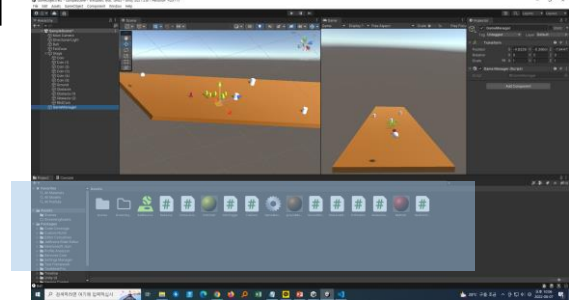
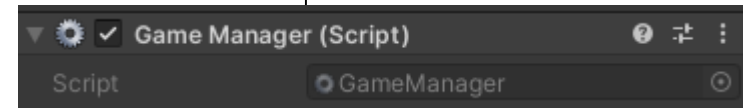
다른 클래스의 필드와 메소드 사용하기 - 1

1. Project – Assets – FailZone 더블클릭

2. Coding

```
void OnTriggerEnter(Collider collider)
{
    Debug.Log(collider.gameObject.name);
    if (collider.gameObject.name == "Ball")
    {
        //GameObject.Find("GameManager").SendMessage("RestartGame");
        GameObject gm = GameObject.Find("GameManager");
        GameManager gmComponent = gm.GetComponent<GameManager>();
        //GameManager gmComponent = GameObject.Find("GameManager").GetComponent<GameManager>();
        gmComponent.RestartGame();
    }
}
```

접근할 수 없어서 error



3. Project – Assets – GameManager 더블클릭

4. Coding

```
public void RestartGame()
{
    SceneManager.LoadScene(0);
}
```

void RestartGame()을 public으로 변경

5. Play

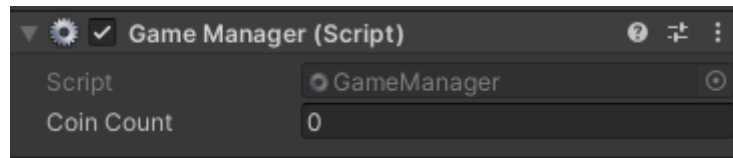
다른 클래스에서의 접근을 허용하려면 필드나 메소드 앞에 public



다른 클래스의 필드와 메소드 사용하기 - 2

1. Project – Assets – GameManager 더블클릭
2. Coding(int coinCount 변수를 public으로)

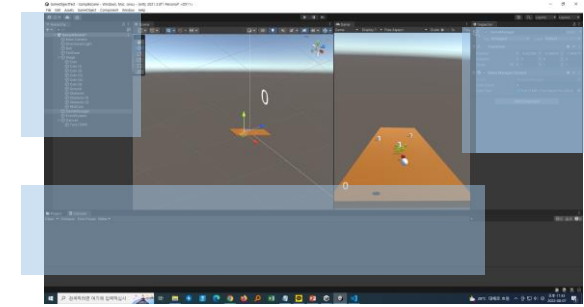
```
public int coinCount = 0;
```



필드를 public으로 하면 Inspector 창에 필드 명 나타남
값을 입력하고 실행하면 반영이 됨
필요한 경우 void Start()에서 초기화



획득한 동전의 개수 표시하기



1. Hierarchy – UI – Text(-TextMeshPro) – Import TMP Essentials

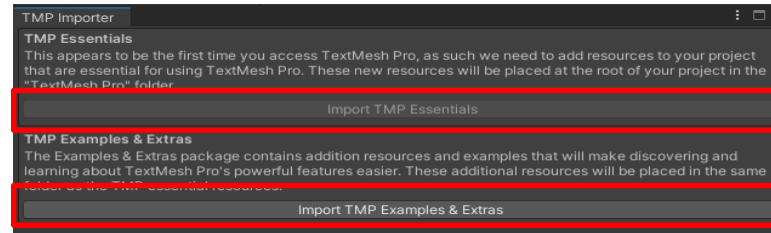
클릭 – Import TMP Examples & Extras 클릭

2. Project – Assets – GameManager 더블 클릭

3. Coding(코드 추가)

```
using TMPro;

public class GameManager : MonoBehaviour
{
    public int coinCount = 0;
    public TextMeshProUGUI coinText;
    void GetCoin()
    {
        coinCount++;
        coinText.text = coinCount + "";
        Debug.Log("동전 : " + coinCount);
    }
}
```



한글 포함하면 error, 한글 따로 처리해야 함

4. Hierarchy – GameManager – Inspector – Game Manager(Script) –

Coin Text – Text(TMP)

5. Play

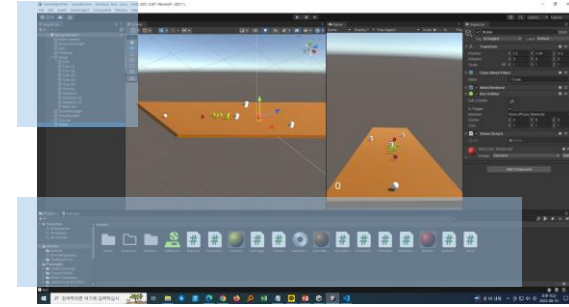


돌 던지기 - 1

1. Hierarchy – 3D Object – Cube – Rename – “Stone”
2. Project – Assets – Create – C# Script – Rename – “Stone”
3. Coding

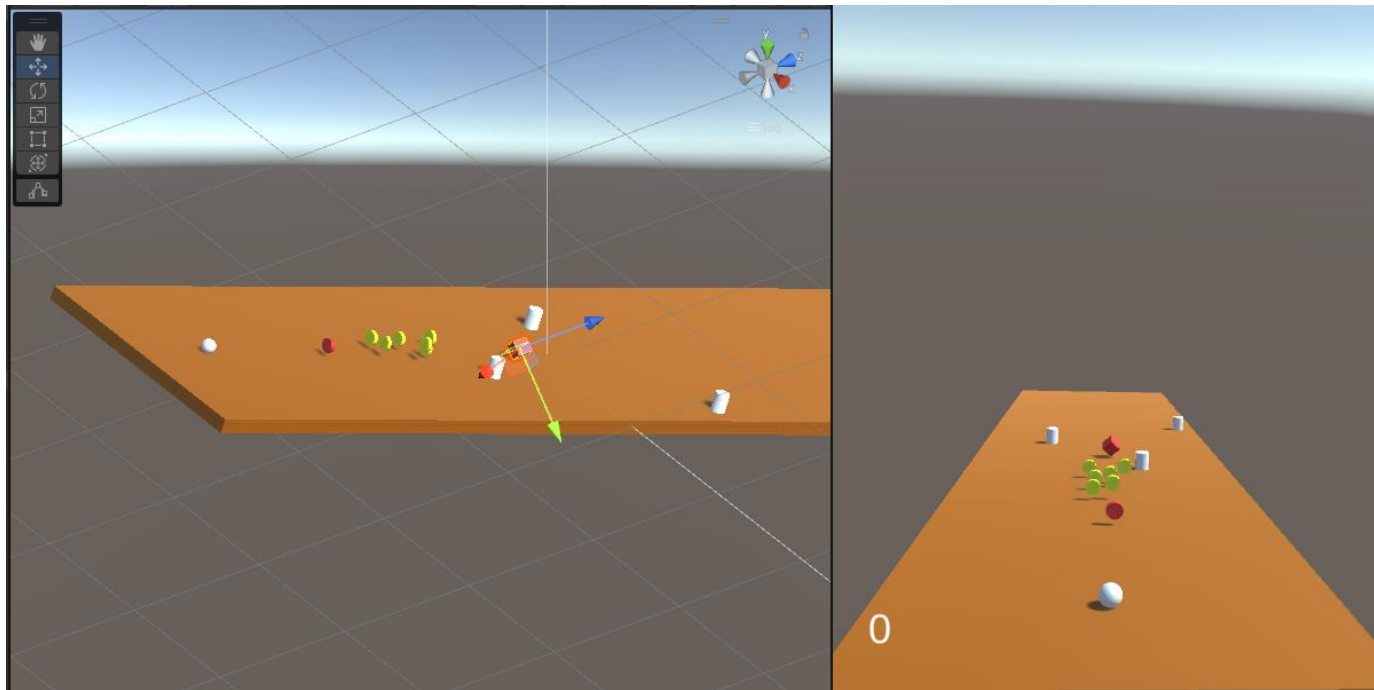
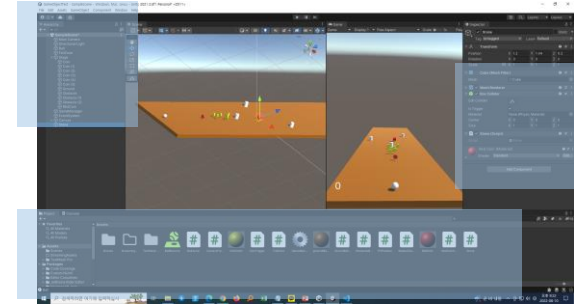
```
using UnityEngine;
public class Stone : MonoBehaviour
{
    Vector3 target;
    // Start is called before the first frame update
    void Start() {
        target = GameObject.Find("Ball").transform.position; //위치 지정
    }

    // Update is called once per frame
    void Update() {
        transform.position = Vector3.MoveTowards(transform.position, target, 0.01f); //target 방향으로 움직이기
        transform.Rotate(new Vector3(0, 0, 5)); //돌 회전하기
    }
    void OnTriggerEnter(Collider collider) {
        Debug.Log(collider.gameObject.name);
        if (collider.gameObject.name == "Ball")
        {
            GameManager gmComponent = GameObject.Find("GameManager").GetComponent<GameManager>();
            gmComponent.RestartGame();
        }
    }
}
```



돌 던지기 - 2

1. Hierarchy – Stone – Inspector – Box Collider – Is Trigger – check
2. Project – Assets – RedCoin을 Hierarchy – Stone으로 DnD
3. Hierarchy – Stone과 Ball을 Stage의 상단에 위치시킴
4. Play

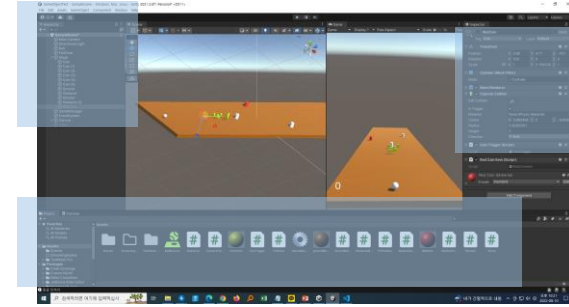


일정 간격으로 메시지 출력하기 - 1

1. Hierarchy – Stage – Obstacle(1) – Rename – “Shooter”
2. Hierarchy – Stage – Shooter – Inspector – Obstacle Move(Script) – Kebab Menu – Remove Component
3. Project – Assets – RedCoin을 Hierarchy – Stage – Shooter로 DnD
4. Project – Assets – Create – C# Script – Rename – “Shooter” – 더블클릭
5. Coding

```
using UnityEngine;

public class ObstacleMove : MonoBehaviour
{
    void OnCollisionEnter(Collision collision)
    {
        //Debug.Log(collision.gameObject.name);
        Vector3 direction = transform.position - collision.gameObject.transform.position;
        //나의 위치에서 상대의 위치를 빼면 방향이 결정
        direction = direction.normalized * 1000; //힘 결정
        collision.gameObject.GetComponent<Rigidbody>().AddForce(direction);
    }
}
```



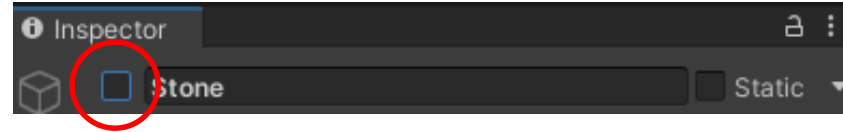
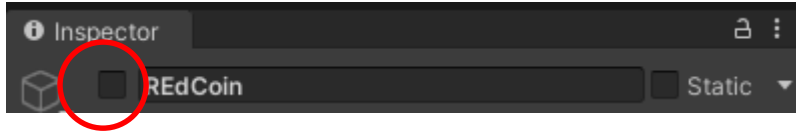
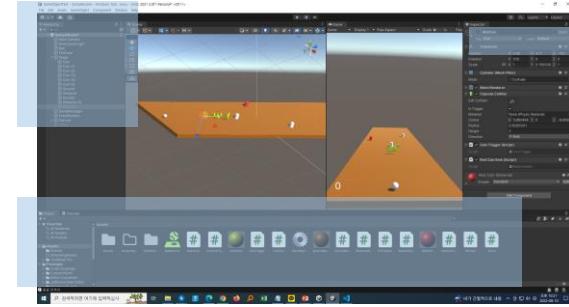
일정 간격으로 메시지 출력하기 - 2

```
float timeCount = 0;
// Update is called once per frame
void Update()
{
    timeCount += Time.deltaTime;
    if (timeCount > 3)
    {
        Debug.Log("돌을 던져라");
        timeCount = 0;
    }
    float newXPosition = transform.localPosition.x + delta;
    transform.localPosition = new Vector3(newXPosition, transform.localPosition.y,
    transform.localPosition.z);
    if (transform.localPosition.x < -9)
    {
        delta = 0.01f;
    }
    else if (transform.localPosition.x > 9)
    {
        delta = -0.01f;
    }
}
}
```

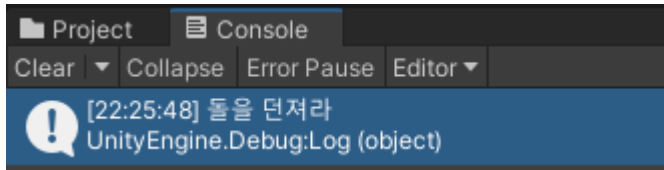


일정 간격으로 메시지 출력하기 - 3

1. Project - Assets - Shooter를 Hierarchy - Stage - Shooter로 DnD
2. Hierarchy - Stage - RedCoin - Inspector - uncheck(비활성화)
3. Hierarchy - Stone - Inspector - uncheck(비활성화)

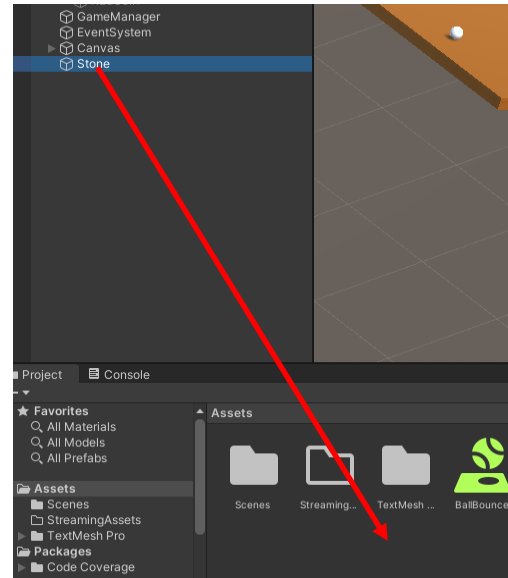


4. Play



Prefab 생성 및 사용하기 - 1

1. Hierarchy – Stone – Inspector – check(활성화)
2. Hierarchy – Stone을 Project – Assets으로 DnD
3. Hierarchy – Stone – Delete(삭제)



4. Project – Assets – Shooter – 더블클릭
5. Coding(코드 추가)

```
using UnityEngine;

public class Shooter : MonoBehaviour
{
    public GameObject stone;

    void OnCollisionEnter(Collision collision)
```

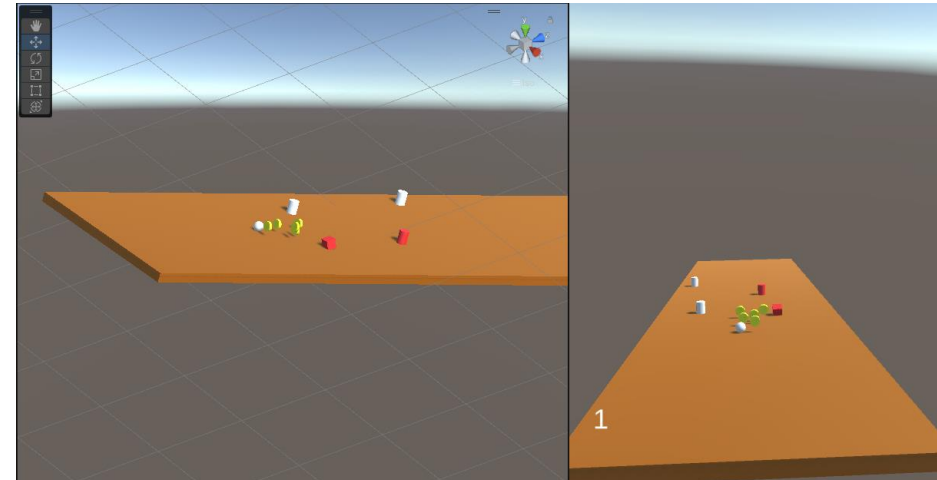
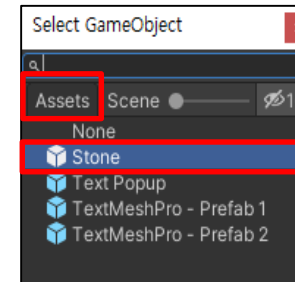
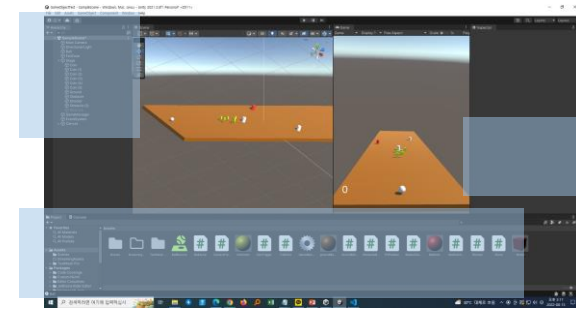


Prefab 생성 및 사용하기 - 2

6. Hierarchy – Stage – Shooter – Inspector – Shooter(Script) – Stone
– Stone 혹은 Project – Assets – Stone(Prefab)을 Hierarchy – Stage
– Shooter – Inspector – Shooter(Script) – Stone으로 DnD
7. Project – Assets – Shooter – 더블클릭
8. Coding

```
void Update()
{
    timeCount += Time.deltaTime;
    if (timeCount > 3)
    {
        //Debug.Log("돌을 던져라");
        Instantiate(stone, transform.position, Quaternion.identity);
        timeCount = 0;
    }
}
```

9. Play



상속

1. Project – Assets – ObstacleMove(Script) – 더블클릭
2. Coding(키워드 추가)

void Update() → protected void Update()

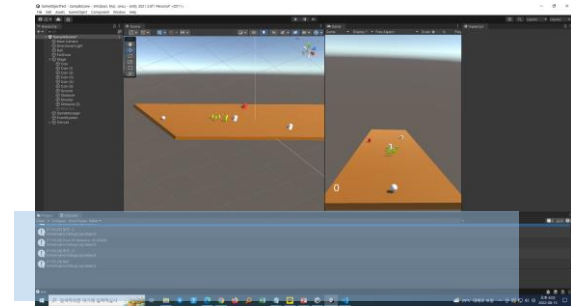
3. Project – Assets – Shooter(Script) – 더블클릭
4. Coding
5. Play

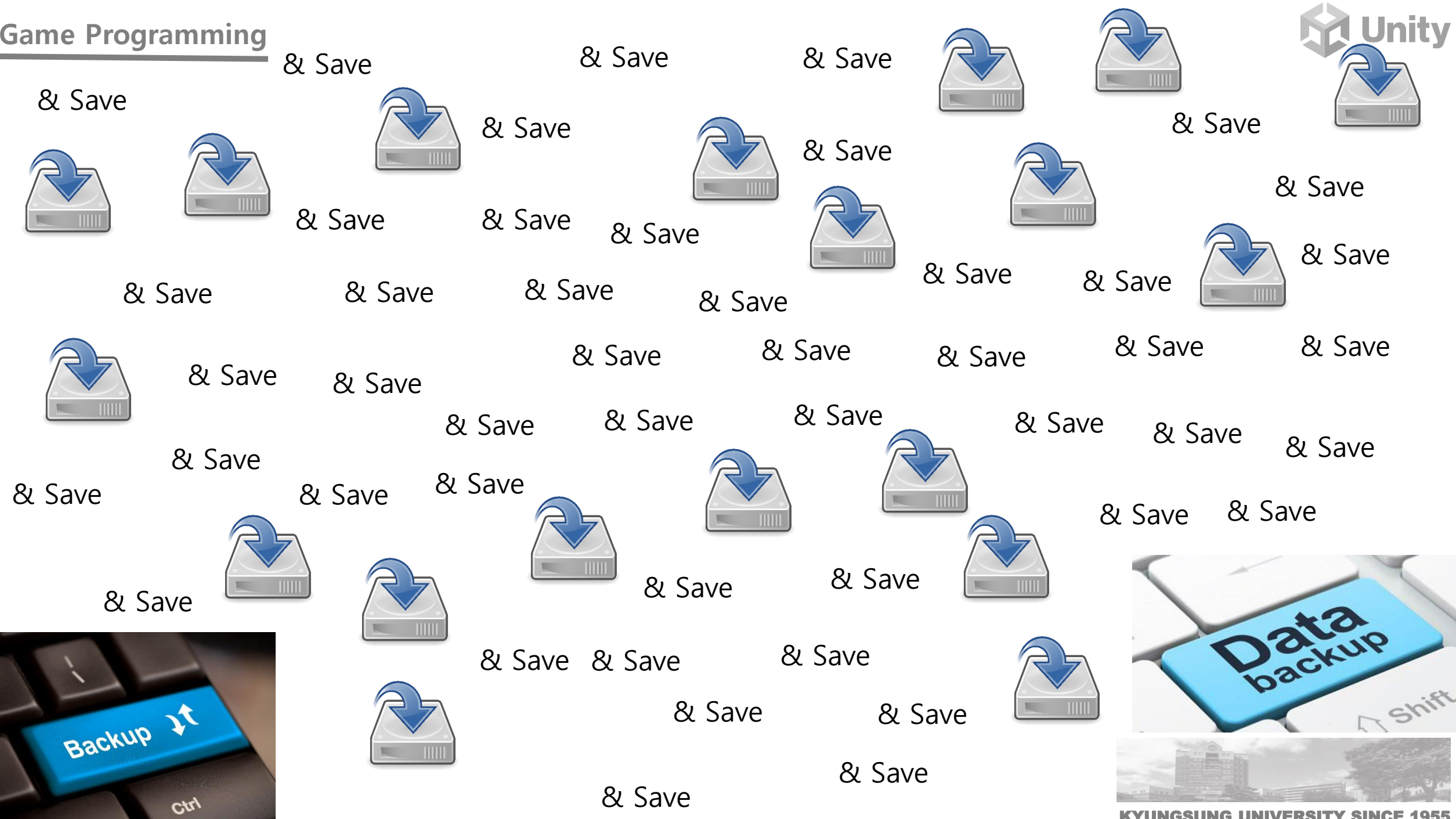
```
using UnityEngine;

public class Shooter : ObstacleMove
{
    public GameObject stone;

    float timeCount = 0;
    // Update is called once per frame
    void Update()
    {
        base.Update();
        timeCount += Time.deltaTime;
        if (timeCount > 3)
        {
            //Debug.Log("돌을 던져라");
            Instantiate(stone, transform.position, Quaternion.identity);
            timeCount = 0;
        }
    }
}
```

ObstacleMove 상속 받음
ObstacleMove에 있는 코드 삭제
Shooter가 움직이기 위해 ObstacleMove의 Update() 메소드 호출
base.Update() 경고, new base.Update()로 변경





Reference

- ✓ <https://unity.com/>
- ✓ <https://docs.unity3d.com/kr/2022.1/Manual/UnityManual.html>
- ✓ <https://school.programmers.co.kr/learn/courses/1>, 정두식
- ✓ https://fiftiesstudy.tistory.com/category/%EC%B7%A8%EB%AF%B8%EB%A1%9C%20%ED%95%98%EB%8A%94%20%EA%B2%8C%EC%9E%84%EC%BD%94%EB%94%A9_gameCodingAsHobby/%EC%9C%A0%EB%8B%88%ED%8B%B0unity%EB%A1%9C%20%EA%B2%8C%EC%9E%84%20%EB%A7%8C%EB%93%A4%EA%B8%B0

