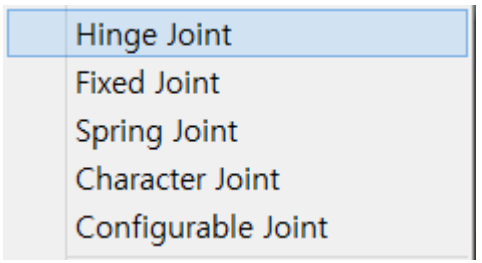
물리엔진의 활용

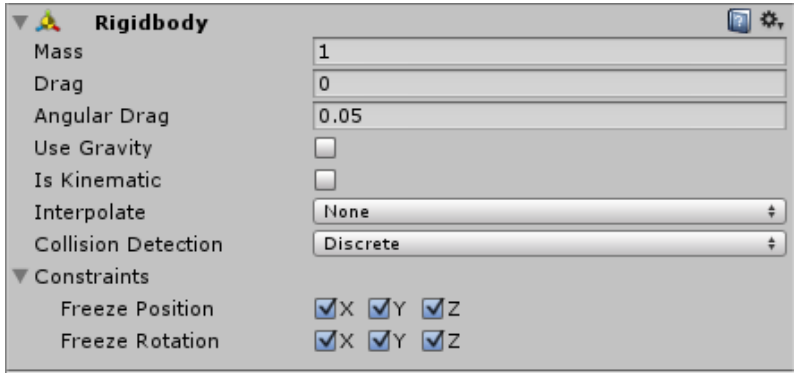
관절

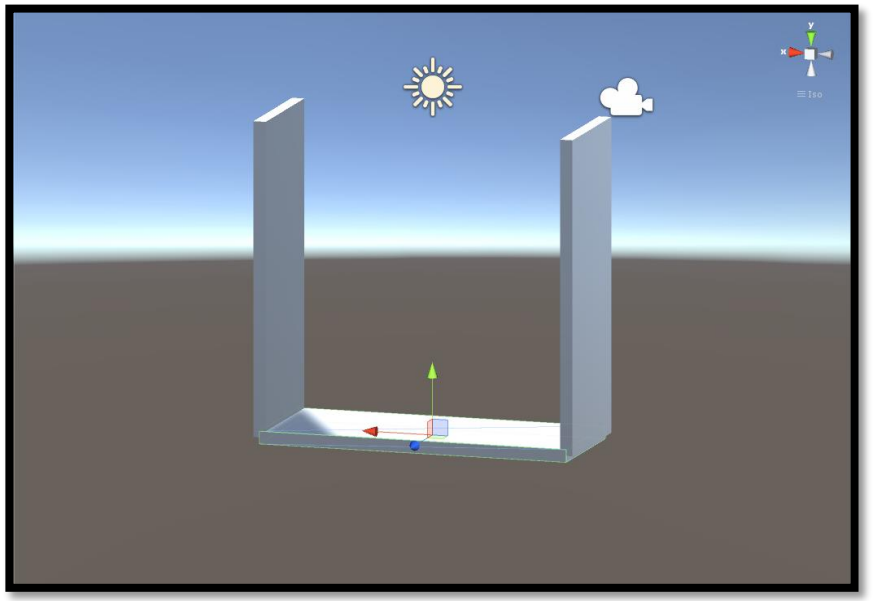
- **관절(Joint)**
  - 강체 오브젝트들을 다른 오브젝트에 연결
  - 일반적으로 회전 축에 제약이나 연결 부분의 움직임에 따라 다양한 종류의 관절 컴포넌트 제공
    - Hinge Joint - 한 축으로만 움직임
    - Fixed Joint - 연결된 게임 오브젝트에 고정된 상태로 움직임
    - Spring Joint - 연결 부분이 스프링 처럼 움직임
    - Character Joint - 캐릭터의 관절, 랙돌Ragdoll 효과 구현에 유용
    - Configurable Joint - 회전이나 가속을 설정 할 수 있는 관절



경첩 관절(Hinge Joint)의 구현

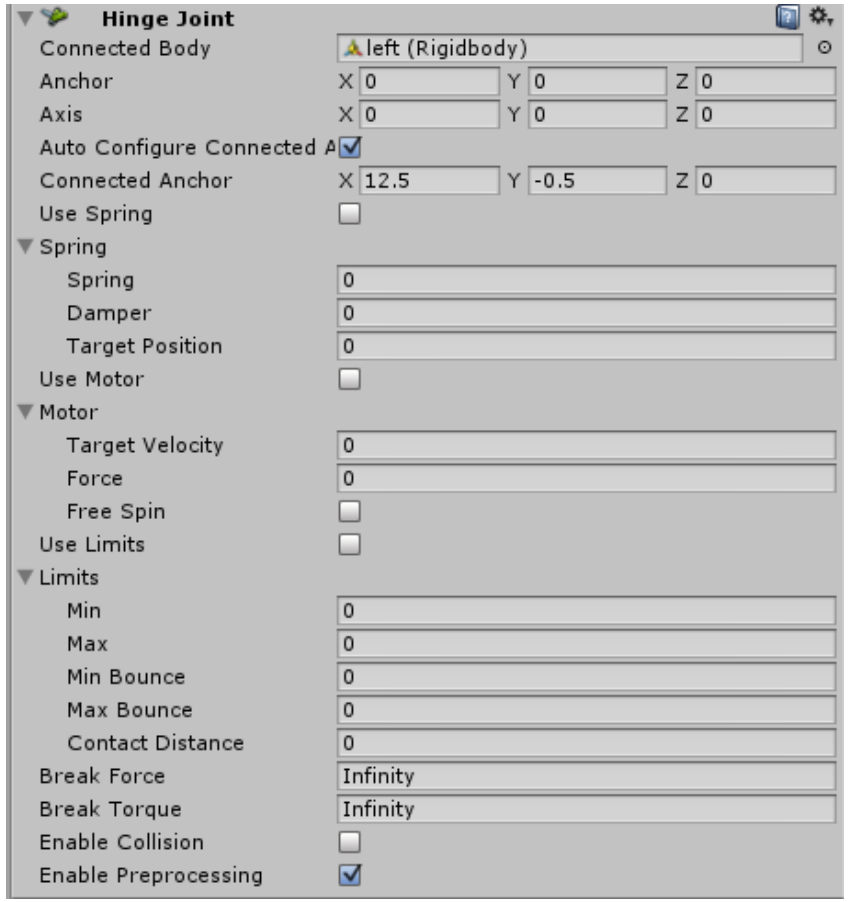
- 큐브에 RigidBody 컴포넌트를 추가하여 좌 우 바닥을 구성하는 세 개의 벽 생성
- 벽 이므로 중력의 영향을 받지 않도록 설정을 끈다
- 움직이지 않도록 X, Y, Z 모두 Freeze





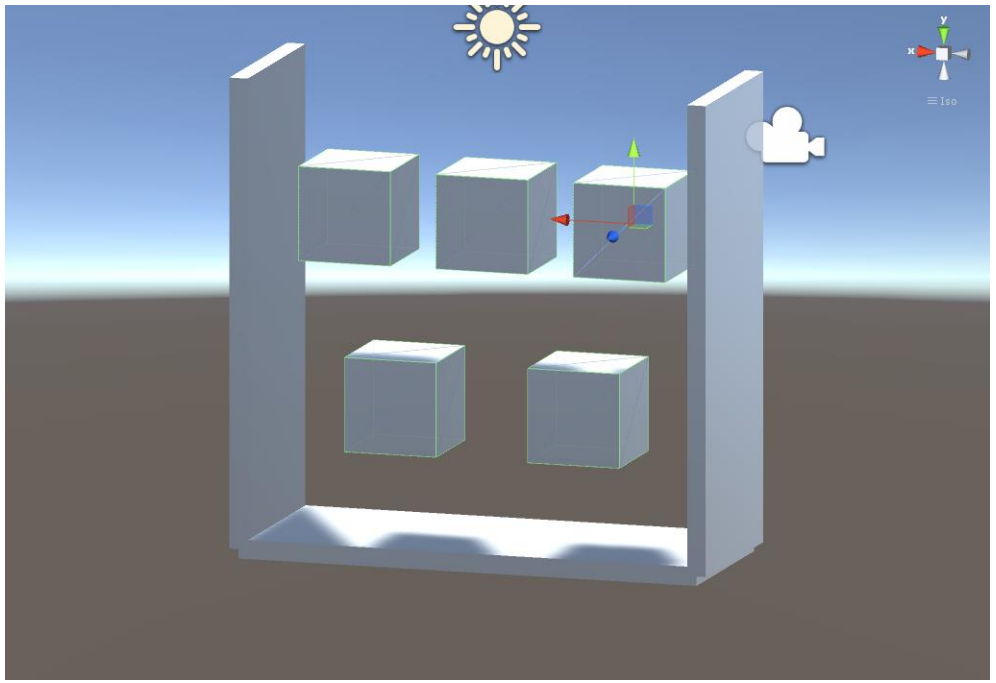
경첩 관절(Hinge Joint)의 구현

- bottom 게임 오브젝트에 Hinge Joint 컴포넌트 추가
  - **[메뉴 > Component > Physics > Hinge Joint]**
- 최초에 Hinge Joint가 작동하지 않도록 설정



경첩 관절(Hinge Joint)의 구현

- **경첩 관절(Hinge Joint)의 구현**
  - Hinge Joint를 작동시킬 큐브 추가
  - 큐브 생성 후 Rigidbody 컴포넌트 추가
  - bottom에 힘이 전달 될 수 있도록 위쪽에 배치



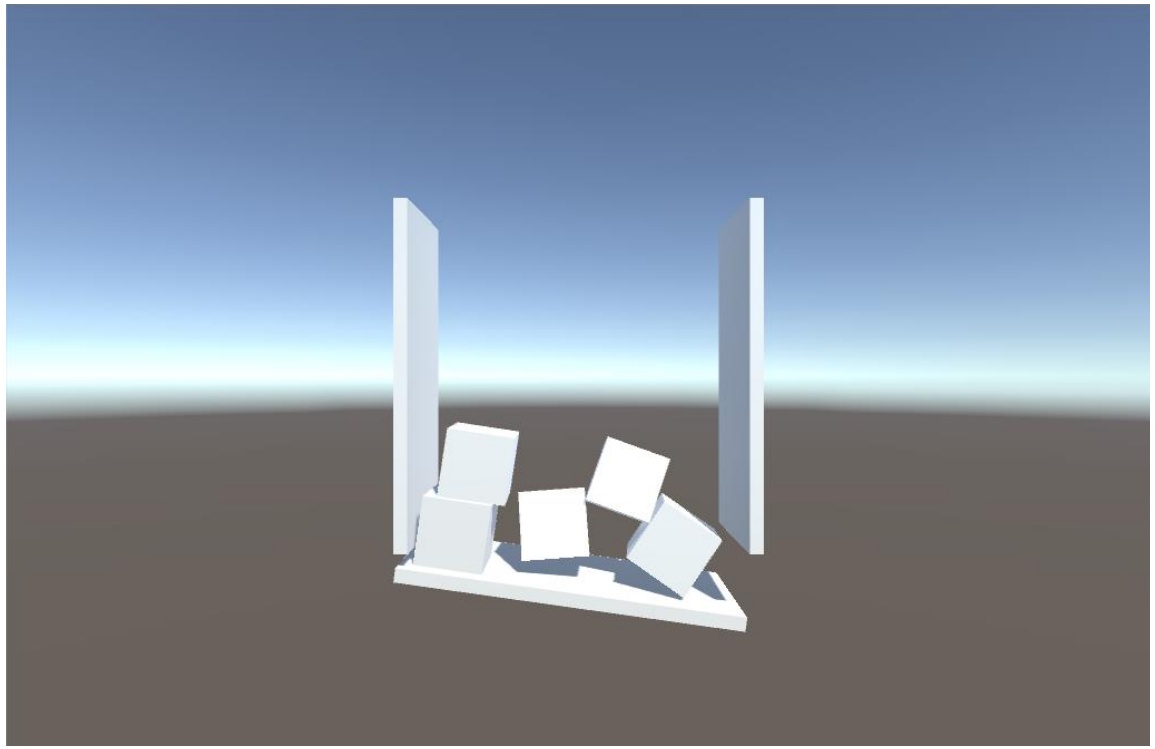
경첩 관절(Hinge Joint)의 구현

- **경첩 관절(Hinge Joint)의 구현**
  - Bottom 클릭 시 경첩 관절을 움직일 스크립트 작성. 작성 후 bottom 게임 오브젝트에 추가

```
using UnityEngine;
using System.Collections;

public class PickSelect : MonoBehaviour {

    void Update () {
        try {
            if (Input.GetMouseButton (0)) {
                Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
                RaycastHit hit = new RaycastHit();

                if (Physics.Raycast(ray, out hit, Mathf.Infinity))
                {
                    hit.collider.gameObject.GetComponent<HingeJoint>().useMotor = true;
                    hit.collider.gameObject.GetComponent<HingeJoint>().anchor = new Vector3(-0.5f, 0.0f, 0.0f);
                    hit.collider.gameObject.GetComponent<HingeJoint>().axis = new Vector3(0.0f, 0.0f, 1.0f);
                    hit.collider.gameObject.GetComponent<Rigidbody>().constraints = RigidbodyConstraints.None;
                }
            }
        } catch(MissingComponentException e) {
            Debug.Log(e.Message);
        }
    }
}
```

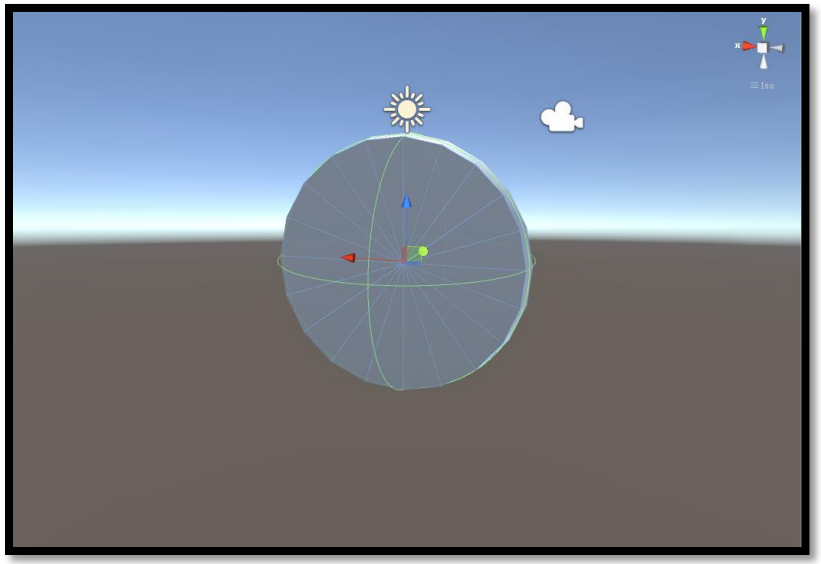
경첩 관절(Hinge Joint)의 구현

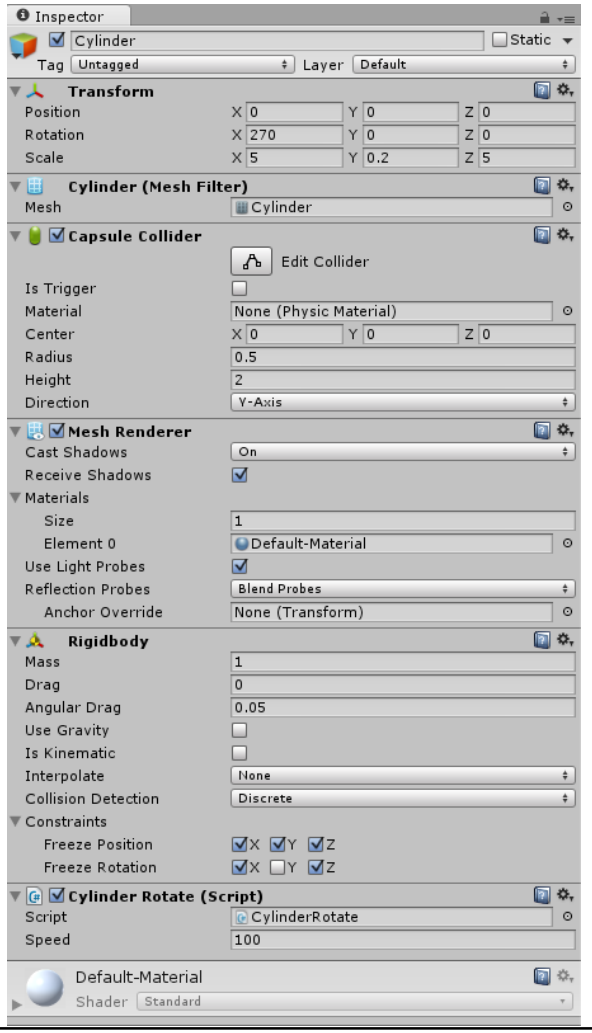
- 실행 후 bottom 게임 오브젝트를 클릭



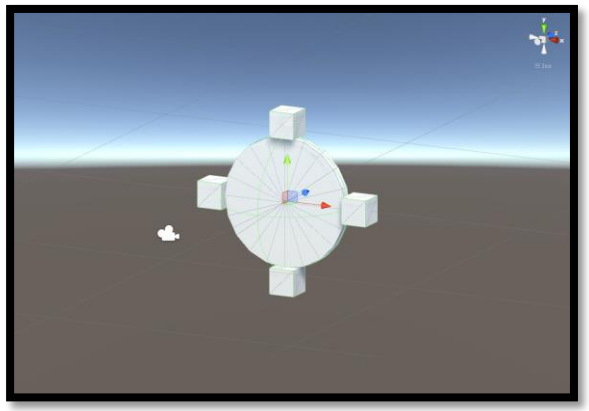
고정 관절(Hinge Joint)의 구현

- 실린더 게임 오브젝트 생성 후 Rigidbody 추가
- Inspector 뷰에서 Transform과 Rigidbody 설정

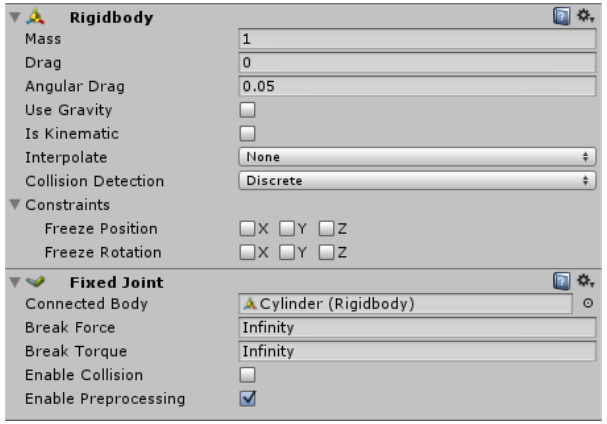


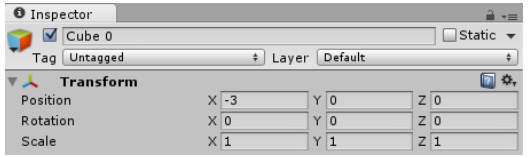


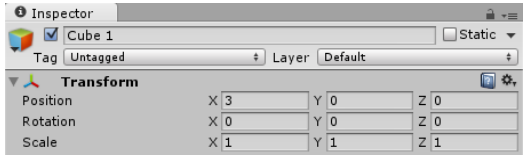
고정 관절(Hinge Joint)의 구현

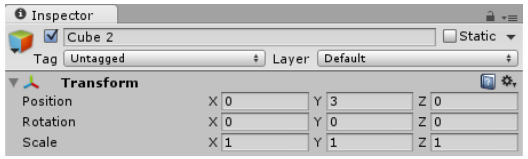


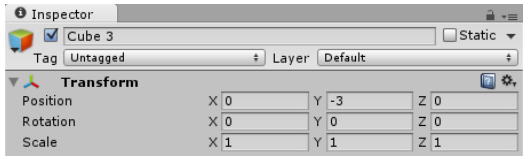
- 앞서 만든 실린더에 연결할 4개의 큐브 생성
- 큐브에 Fixed Joint 추가 후 값 설정
  - **[메뉴 > Component > Physics > Fixed Joint]**
- 설정 후 빈 게임 오브젝트를 생성하여 실린더와 큐브들을 자식 노드로 추가 후 이름을 leftGear로 수정





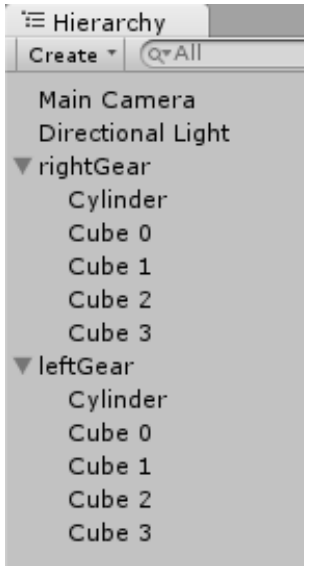


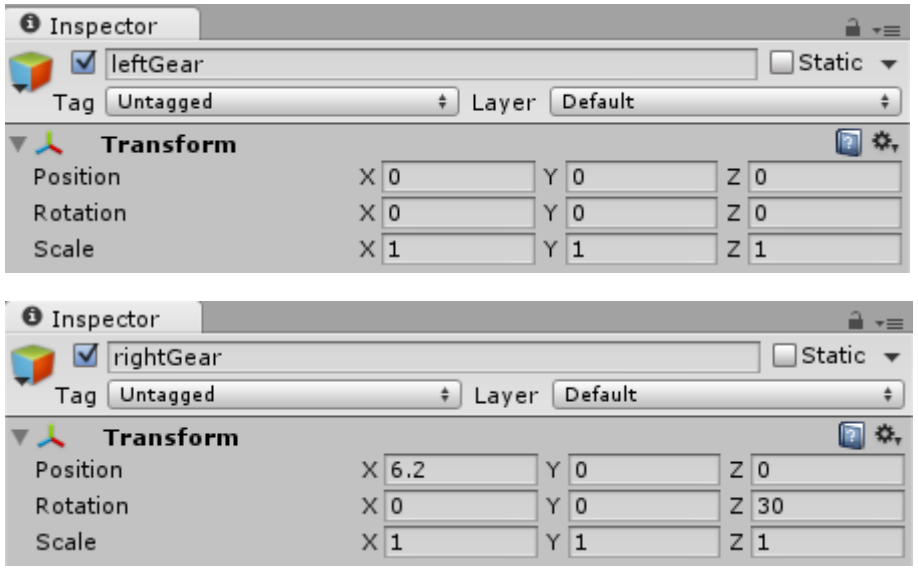


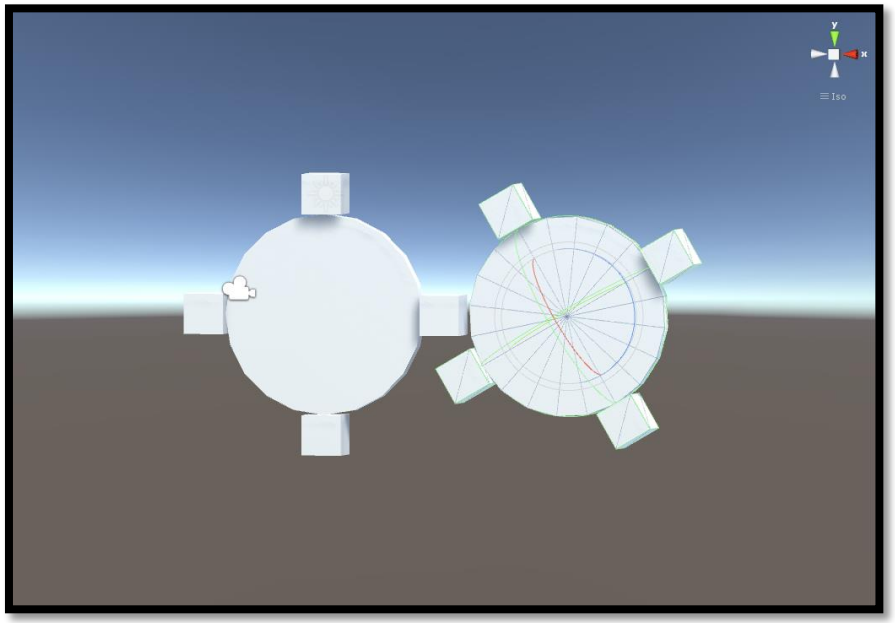


고정 관절(Hinge Joint)의 구현

- leftGear를 복제하여 rightGer로 이름 수정 후 오른쪽에 배치







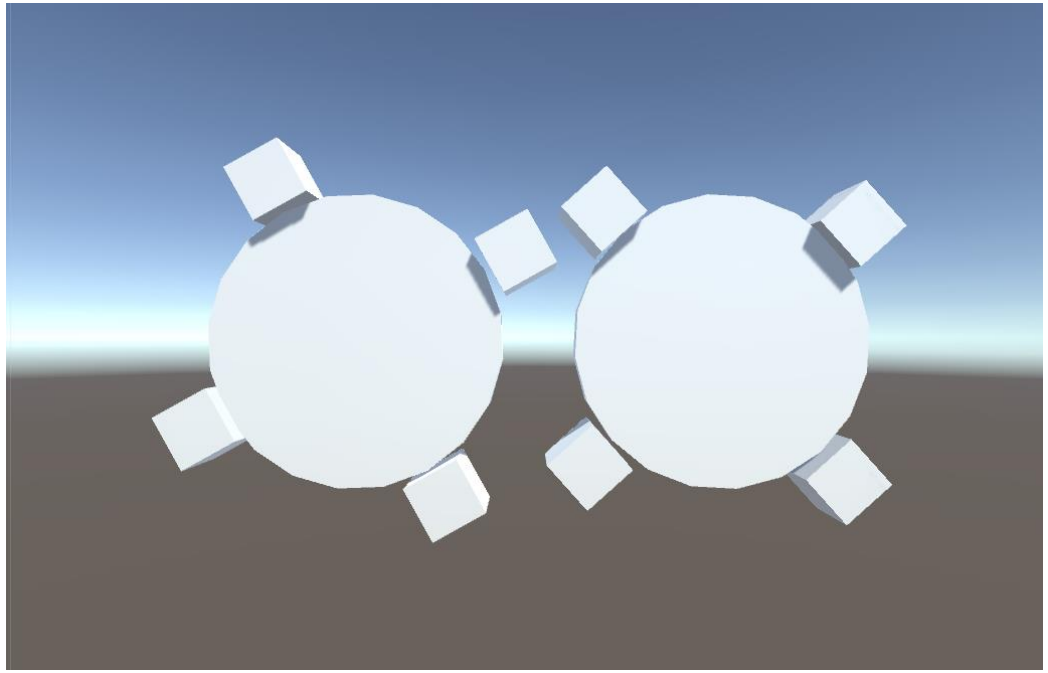
고정 관절(Hinge Joint)의 구현

- leftGear를 회전시키는 스크립트 작성 후 leftGear의 Cylinder 게임 오브젝트에 추가

```
using UnityEngine;
using System.Collections;

public class CylinderRotate : MonoBehaviour {
    public float speed = 100.0f;

    void FixedUpdate () {
        GetComponent<Rigidbody> ().AddTorque (new Vector3 (0.0f, 0.0f, 1.0f * speed), ForceMode.Impulse);
    }
}
```

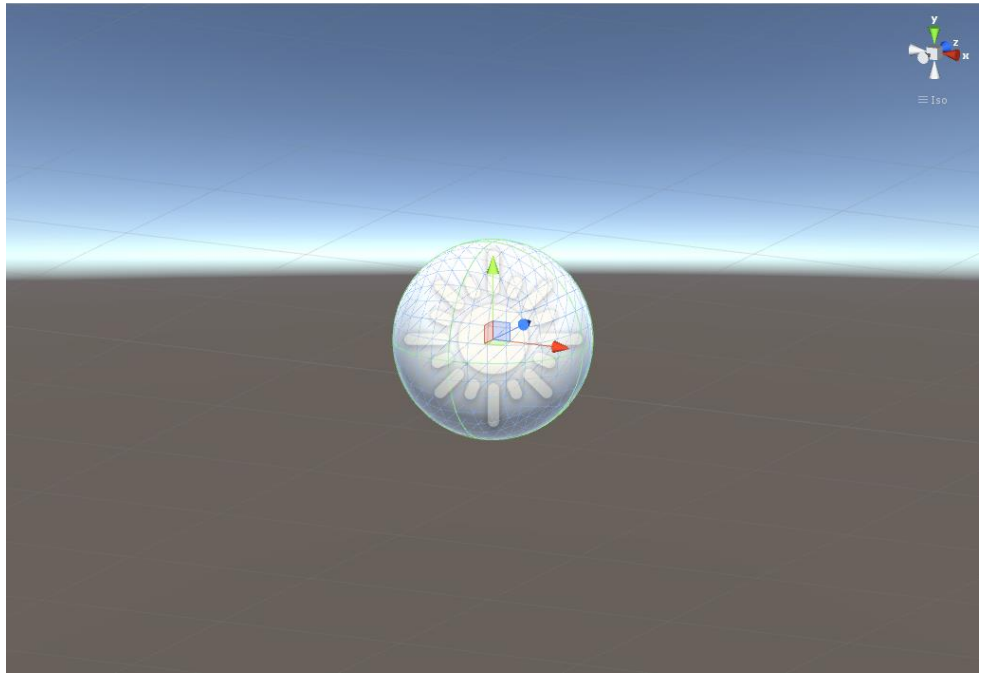
고정 관절(Hinge Joint)의 구현

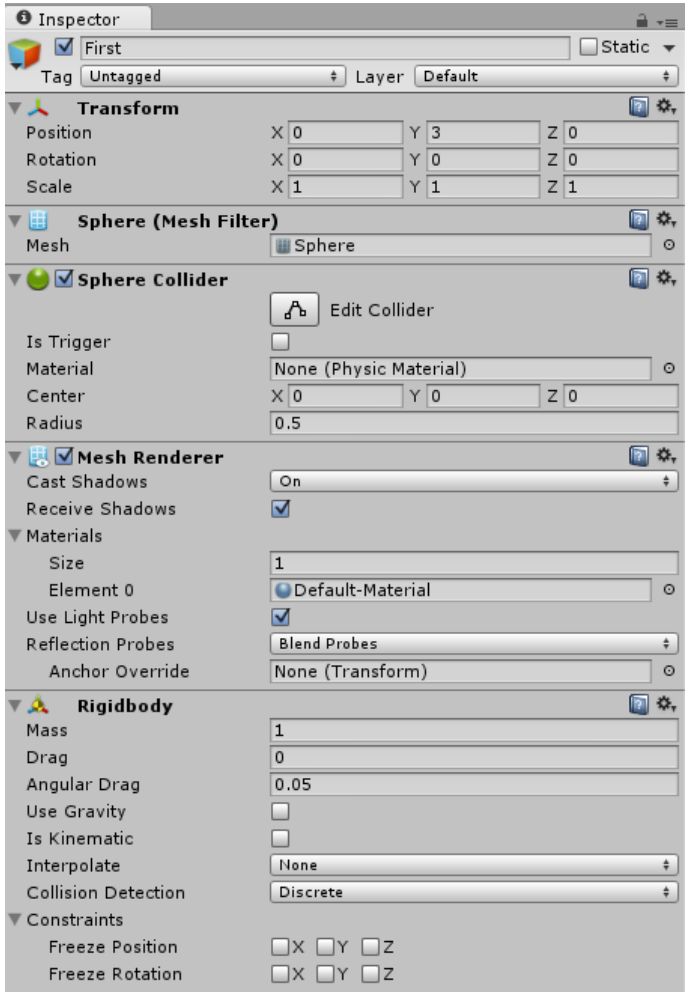
- 실행 결과
  - 왼쪽 톱니바퀴가 회전하면서 오른쪽 톱니바퀴를 회전 시킴



스프링 관절(Spring Joint)의 구현

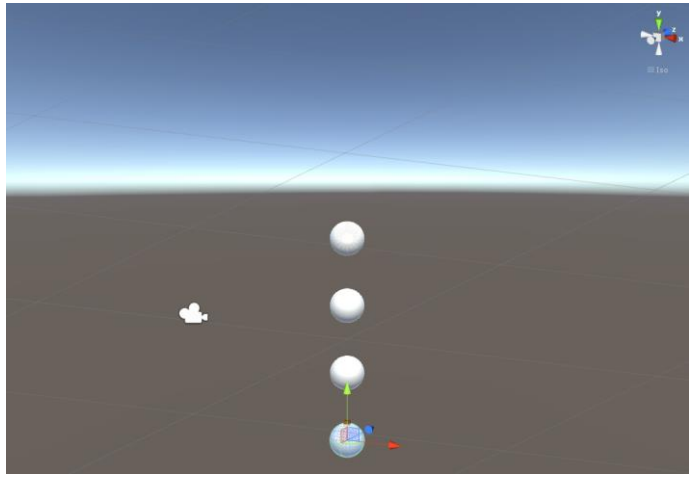
- 구 게임 오브젝트 생성 후 이름을 First 로 변경
- Rigidbody 컴포넌트 추가 & 값 설정

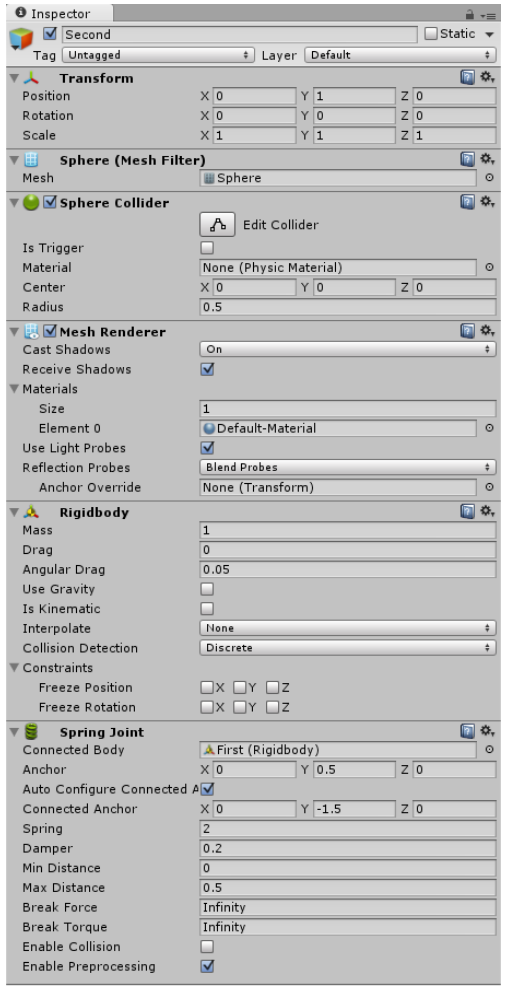




스프링 관절(Spring Joint)의 구현

- 앞서 생성한 구를 복제 후 이름을 Second로 변경
- Spring Joint 컴포넌트 추가
- Connected Body에 First 연결 & 값 설정
- 동일한 방식으로 Third & Forth 추가 후 Transform 위치 조정





스프링 관절(Spring Joint)의 구현

- First 게임 오브젝트에 힘을 가하는 스크립트 작성
- 작성후 First 게임 오브젝트에 추가

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class SphereMovement : MonoBehaviour {
5     void Start () {
6         GetComponent<Rigidbody> ().AddForce (Vector3.up * 10.0f, ForceMode.Impulse);
7     }
8 }
```

스프링 관절(Spring Joint)의 구현

- 실행 결과
  - 움직임 확인을 위해 카메라 위치 조정 후 실행