# Unity Lecture 01

GetComponent() 사용하기

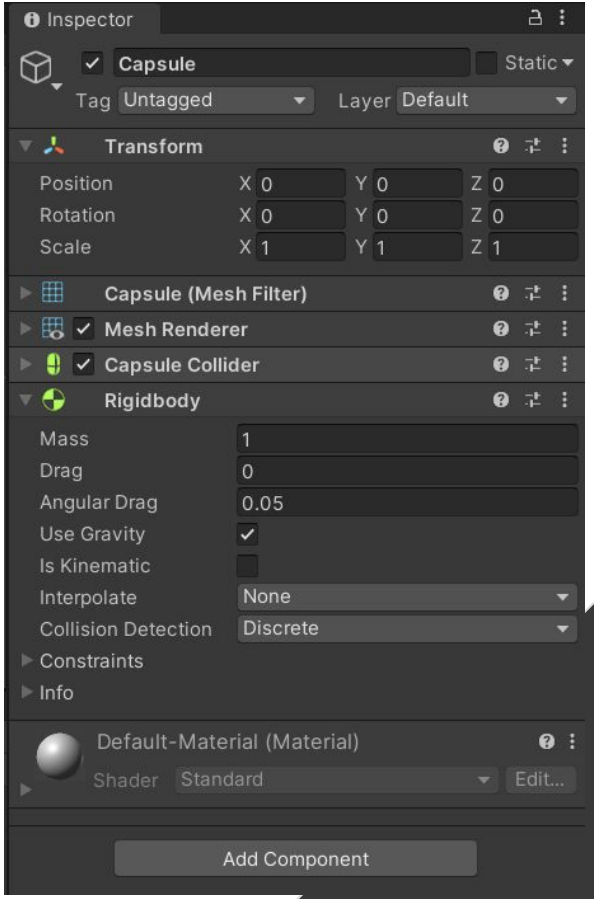
# COMPONENT 란?

유니티에서 기능을 가지고 있는 부품이라고 할 수 있습니다.

기계에는 부품들이 들어가듯이 게임오브젝트GameObject 에도 컴포넌트가 들어가서 동작하게 됩니다. 오른쪽 Inspector 창에서 탭으로 분리되어있는 것들 하나하나가 컴포넌트입니다.

Transform은 좌표, 회전, 크기 등을 나타내고 Mesh Renderer는 어떻게 그릴지, Capsule Collider는 충돌 확인을 어떻게 할지, Rigidbody는 물리 시뮬레이션을 해주는 역할을 합니다.

컴포넌트를 넣고 싶을때는 그냥 밑의 Add Component 버튼을 사용하면 끝이 나고, 그 컴포넌트의 기능들을 쓸 수 있지만 만약 코드(스크립트) 에서 그 기능들을 사용하고 싶다면 어떻게 해야할까요?



# GetComponent 사용하기

오른쪽 사진처럼, 여러번 쓰일 컴포넌트는 해당 컴포넌트(클래스) 타입의 변수를 선언하고, 그 변수에 GetComponent<컴포넌트(클래스)명>(); 과 같이 사용합니다. 그리고 이렇게 불러오는 것의 장점은, 컴포넌트 창에는 보이지 않는 속성들, x,y,z 축의 현재 속도를 담당하는 Rigidbody의 velocity 라는 속성을 가지고 오고, 수정할 수 있다는 점입니다.

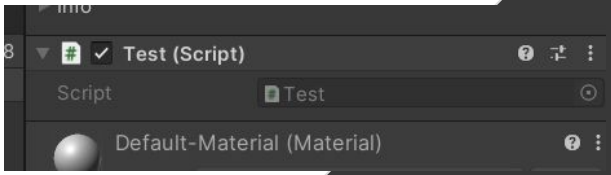
오른쪽 코드는 내가 가지고있는 Rigidbody 컴포넌트를 가져와서 현재 속도값을 (0, 10, 0) 으로 설정합니다.


```
public class Test : MonoBehaviour
{
    Rigidbody rgd;
    void Start()
    {
        rgd = GetComponent<Rigidbody>();
        rgd.velocity = new Vector3(0, 10, 0);
    }
}
```

# 내가 만드는 컴포넌트

방금 위에서 작성된 Test는 이제 게임 오브젝트에 넣어줘야지 작동하게 됩니다. 그런데 넣어준다고요? 네 맞습니다. 우리가 작성한 C# 클래스, 즉 스크립트들은 컴포넌트로 등록이 되고 사용이 됩니다. 그래서 스크립트를 게임오브젝트에 드래그해서 넣으면 Inspector 창에 해당 스크립트가 컴포넌트로 있는 것을 볼 수 있죠.

그래서 내가 만든 스크립트는 똑같이 GetComponent<>() 로 가져올 수 있고, 해당 스크립트(클래스) 의 public 메서드와 변수들을 사용할 수 있습니다.



# 스크립트 컴포넌트 사용하기 - 1

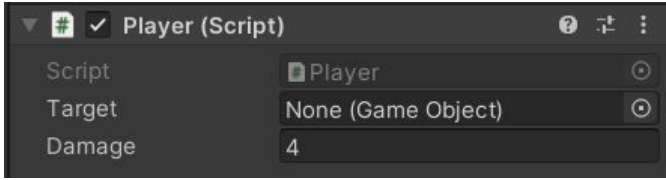
예시로 간단한 스크립트를 작성했습니다. 두 스크립트를 서로 다른곳이든, 같은곳이든 아무 게임 오브젝트에나 넣어보세요.

```
public class Player : MonoBehaviour
{
    public GameObject target;
    public int damage = 4;

    void Update()
    {
        if (Input.GetMouseButtonDown(0))
            target.GetComponent<Entity>().OnDamaged(this.damage);
    }
}
```

```
public class Entity : MonoBehaviour
{
    int health = 10;
    public void OnDamaged(int dmg)
    {
        this.health -= dmg;
        Debug.Log("남은 체력: " + this.health);

        if (this.health <= 0)
            Destroy(gameObject);
    }
```

# 스크립트 컴포넌트 사용하기 - 2

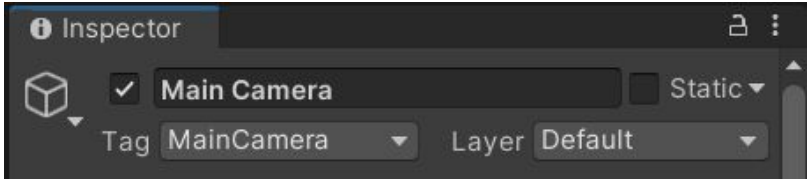
public 으로 선언된 변수들은 오른쪽처럼 Inspector 창에 보이게 되는데, Target에 Entity 스크립트(클래스)를 적용한 게임오브젝트를 드래그해서 할당한 다음에, 실행해서 좌클릭을 일정 횟수 하면 Target 게임오브젝트가 사라지는 것을 볼 수 있습니다!




```
public Entity target;
public int damage = 4;

void Update()
{
    if (Input.GetMouseButtonDown(0))
        target.OnDamaged(this.damage);
}
```

참고로, 왼쪽과 같이 애초부터 Entity 타입으로 입력받으면 GetComponent를 생략할 수 있습니다.

# GAMEOBJECT 컴포넌트




GameObject는 모든 게임오브젝트들이 가지고 있는 컴포넌트(클래스) 이며, 오른쪽처럼 맨 위에 해당하는 부분입니다. 게임 오브젝트의 이름, Tag, Layer나 활성화/비활성화를 할 수 있는데, 이 컴포넌트는 모든 게임오브젝트에 있다는 것이 자명하기 때문에 GetComponent를 할 필요 없이 gameObject 라는 이름으로 바로 접근할 수 있게 해주었습니다.

```
void Start()
{
    gameObject.SetActive(false);
}
```

여기서 Tag와 Layer는 비교에 주로 사용되며, SetActive(bool) 는 안에 true를 넣어주면 활성화, false를 넣어주면 비활성화됩니다.

# TRANSFORM 컴포넌트

Transform 컴포넌트 또한 모든 게임 오브젝트가 필수적으로 지니고 있는 컴포넌트입니다. transform이라는 이름으로 접근할 수 있고 좌표, 회전, 크기, 부모/자식의 정보 등 수많은 정보를 담고 있습니다.

Unity - Scripting API: Transform (unity3d.com)

위의 공식 API를 참고하면 모든 Public 메서드와 변수들을 확인할 수 있습니다.

# 마치며

궁금한 점은 꼭 질문해주세요!

카카오톡 | 디스코드

sigening | Sigening#6088