Fundamental Functions
(Chapter 7)

Jin-Mo Kim

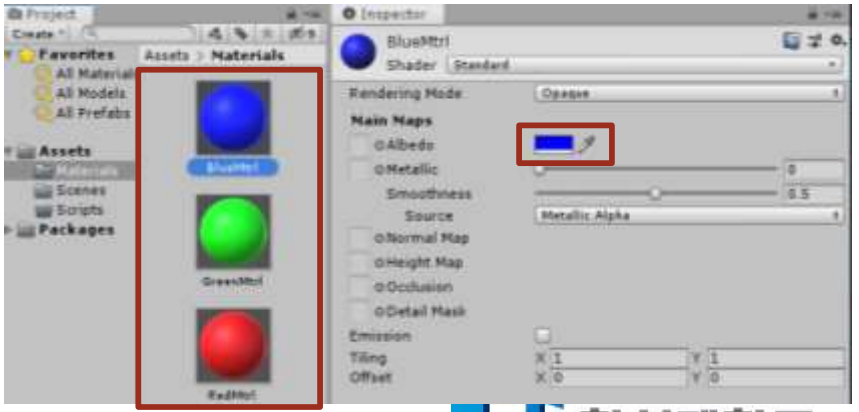
jinmo.kim@hansung.ac.kr

배경 설정

- 폴더 생성
  - Materials
  - Scripts

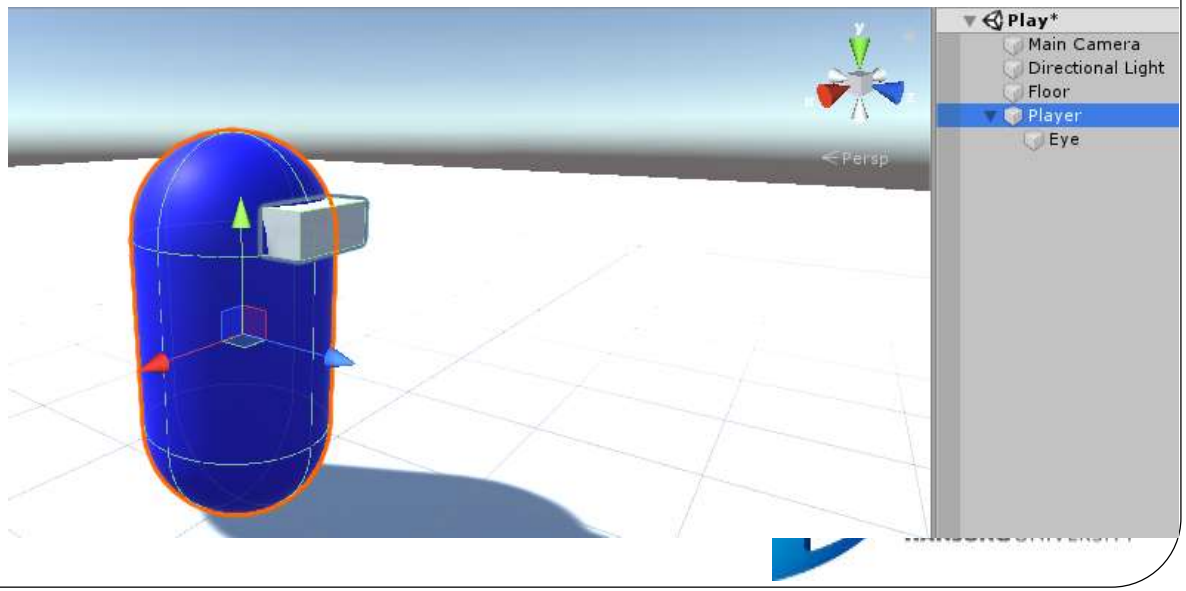
- 바닥
  - 3D Object → Plane
    - 이름: Floor
    - Scale: 5, 1, 5

- 재질 추가 (Materials 폴더)
  - Project → Material
    - RedMtrl, GreenMtrl, BlueMtrl 재질 생성



Movement (이동)

- 플레이어 설정
  - 3D Object → Capsule
    - 이름: Player
    - Position: 0, 1, 0
    - BlueMtrl 재질 적용
    - 플레이어 방향 식별을 위한 자식 객체 생성
    - 3D Object → Cube
      - 이름: Eye
      - Position: 0, 0.6, 0.5
      - Scale: 0.5, 0.2, 0.2



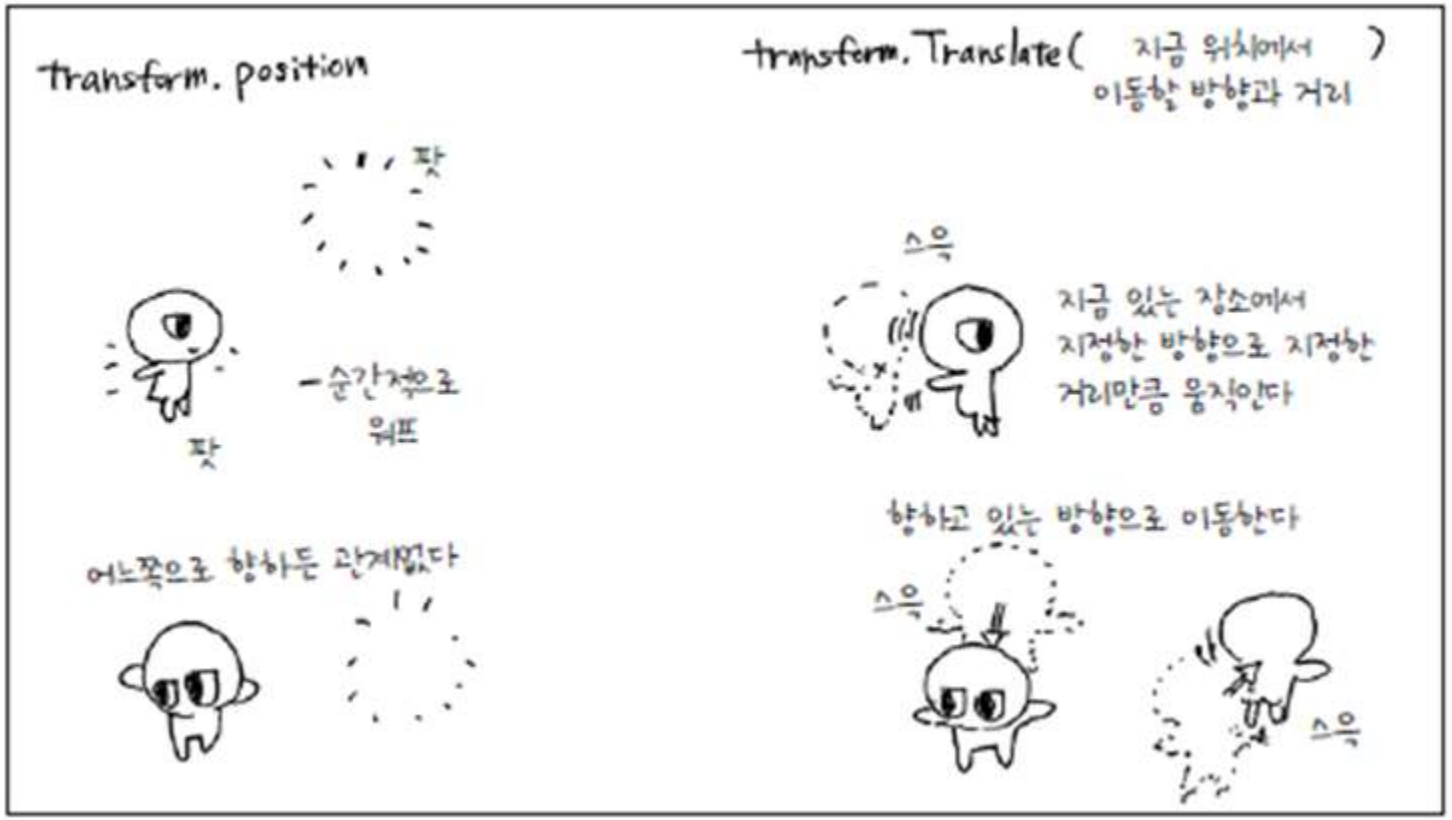
Movement (이동)

- 스크립트 생성 및 등록
  - 스크립트 폴더: cshPlayerController.cs 파일 생성

```
public class cshPlayerController : MonoBehaviour
{
    public float m_moveSpeed = 2.0f;

    void Update()
    {
        PlayerMove();
    }

    void PlayerMove()
    {
        float h = Input.GetAxis("Horizontal");
        float v = Input.GetAxis("Vertical");

        Vector3 moveHorizontal = Vector3.right * h;
        Vector3 moveVertical = Vector3.forward * v;
        Vector3 velocity = (moveHorizontal + moveVertical).normalized;

        transform.LookAt(transform.position + velocity);

        transform.Translate(velocity * m_moveSpeed * Time.deltaTime, Space.World);
    }
}
```

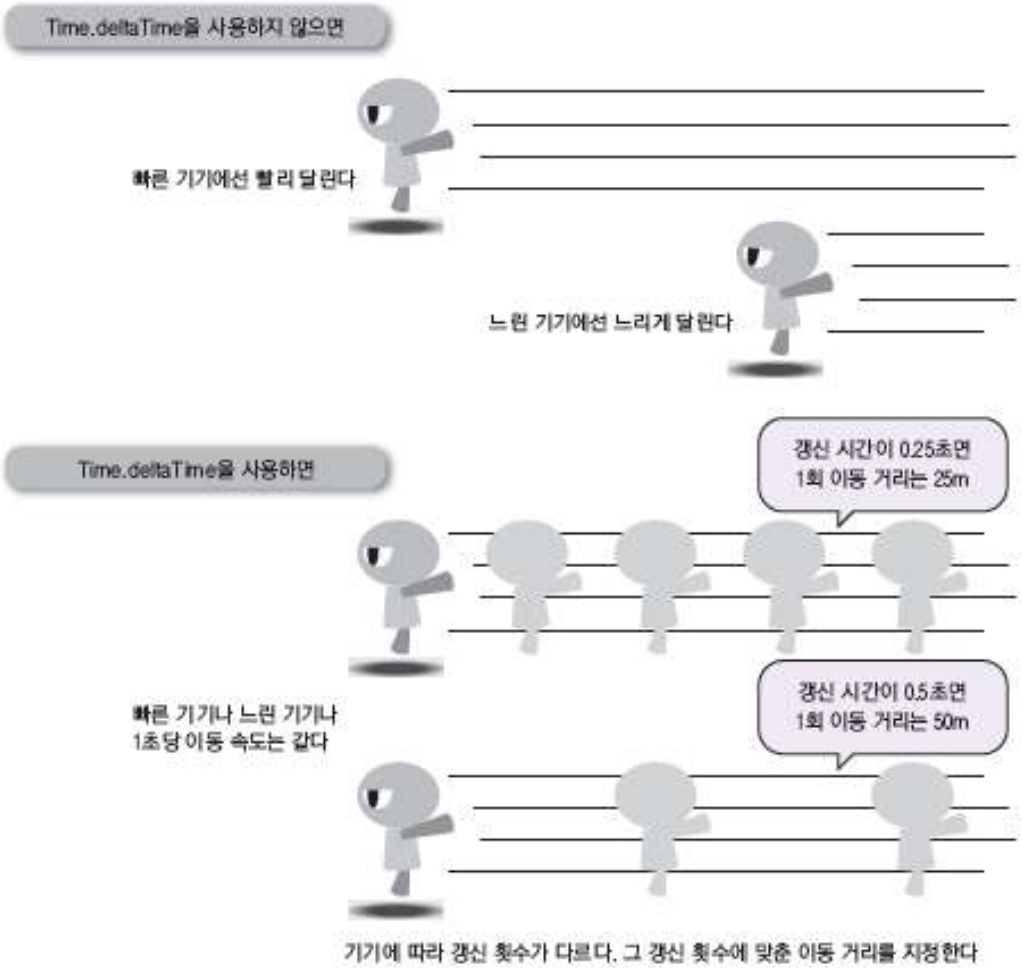
Movement (이동)

- Transform 과 Translate



Movement (이동)

- Time.deltaTime



Movement (이동)

- Time.deltaTime
  - 기기의 성능에 따라 게임의 갱신빈도가 다르다.
  - 기기 별 성능의 차이를 메워주는 것

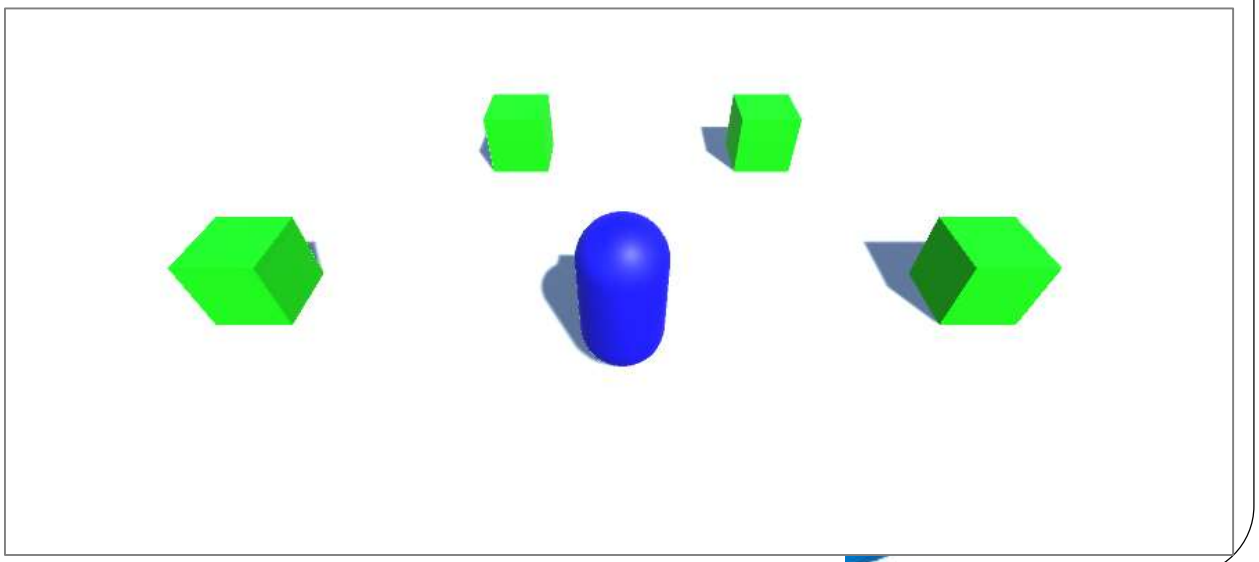
  → **Time.deltaTime**

  - 초당 이동 거리와 회전량 지정

  - 빠른 기기 : 1회 이동거리 ↓ 갱신 빈도 ↑
  - 느린 기기 : 1회 이동거리 ↑ 갱신 빈도 ↓

Selection (선택)

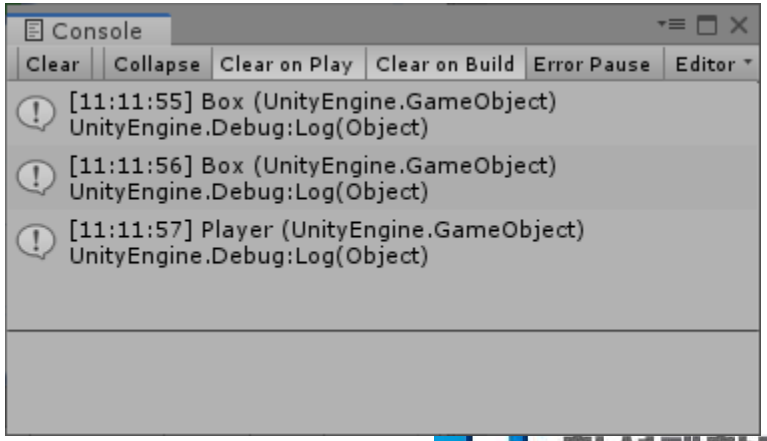
- 선택 객체 생성
  - 3D Object → Cube
    - 이름: Box
    - GreenMtrl 재질 적용
    - 임의의 위치에 적당히 배치
    - **객체 선택을 위해서는 충돌체 속성이 반드시 필요!!** Collider
- 자연스러운 선택을 위해 카메라 위치, 방향 조정
  - Main Camera
    - Position: 0, 5, -4
    - Rotation: 45, 0, 0



Selection (선택)

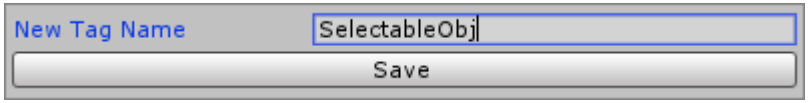
- 스크립트 생성 및 등록
  - 스크립트 폴더: cshSelectionObject.cs 파일 생성
    - Player 객체에 등록

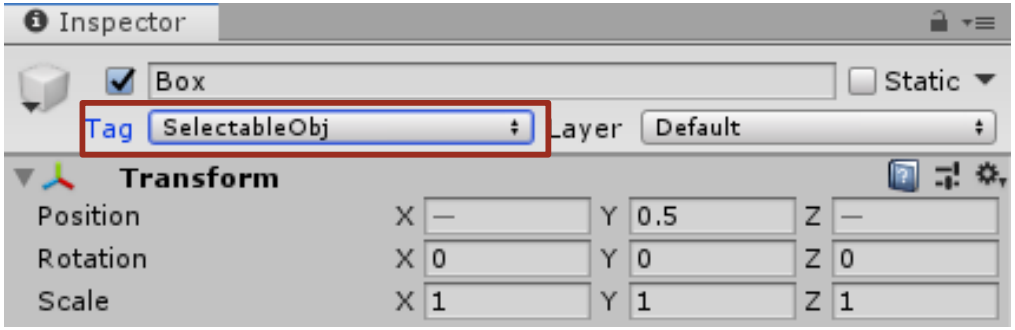
```
public class cshSelectionObject : MonoBehaviour
{
    public Camera cam;

    void Update()
    {
        if (Input.GetMouseButtonDown(0))
        {
            RaycastHit hit;
            Ray ray = cam.ScreenPointToRay(Input.mousePosition);
            if (Physics.Raycast(ray, out hit, Mathf.Infinity))
            {
                Debug.Log(hit.transform.gameObject);
            }
        }

    }
}
```



Selection (선택)

- 태그 설정
  - 한 개 이상의 게임 오브젝트 에 할당할 수 있는 레퍼런스 단어
    - 같은 목적을 가지는 다양한 오브젝트들을 태그를 활용하여 관리
    - Inspector → Tag → Add Tag
      - SelectableObj 태그 추가
      - Box 객체들의 태그를 SelectableObj로 변경





Selection (선택)

- 스크립트 수정
  - 특정 태그를 가진 오브젝트만 선택 가능하도록 수정
  - cshSelectionObject.cs 코드 추가

```
public class cshSelectionObject : MonoBehaviour
{
    public Camera cam;

    void Update()
    {
        if (Input.GetMouseButtonDown(0))
        {
            RaycastHit hit;
            Ray ray = cam.ScreenPointToRay(Input.mousePosition);
            if (Physics.Raycast(ray, out hit, Mathf.Infinity))
            {
                if(hit.transform.gameObject.tag =="SelectableObj")
                    Debug.Log(hit.transform.gameObject);
            }
        }

    }
}
```

Physics (물리)

- 물리 속성
  - 외부 힘에 의한 물리적 움직임 처리를 위한 강체 (Rigidbody) 속성 추가
  - 박스 객체들
    - Add Component → Rigidbody

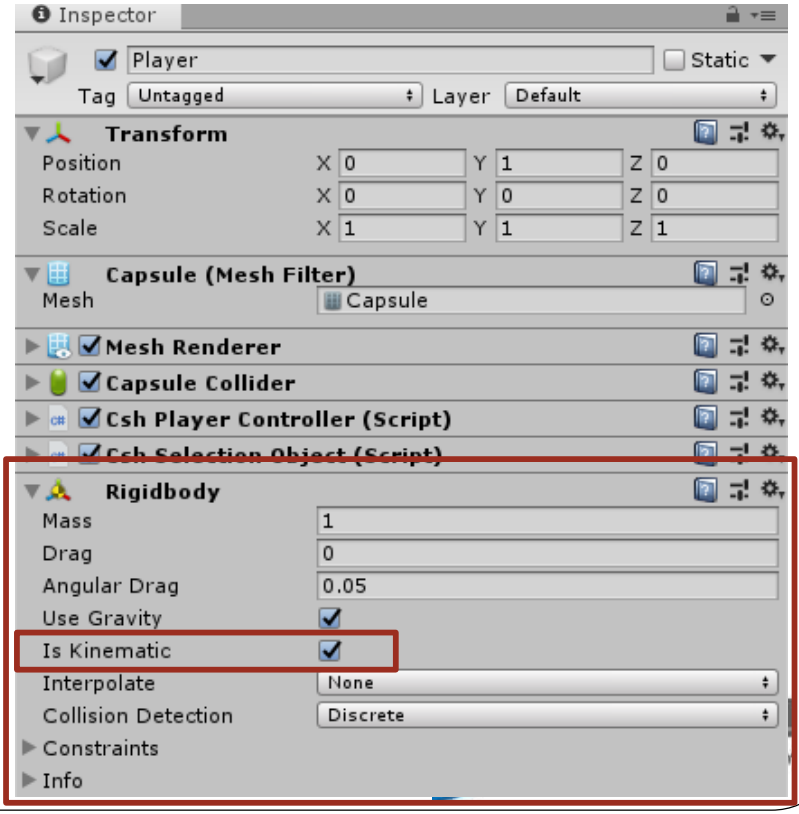
Physics (물리)

- 스크립트 수정
  - cshSelectionObject.cs 코드 추가 : 강체 속성에 힘 추가 (AddForce)

```
public class cshSelectionObject : MonoBehaviour
{
    public Camera cam;

    void Update()
    {
        if (Input.GetMouseButtonDown(0))
        {
            RaycastHit hit;
            Ray ray = cam.ScreenPointToRay(Input.mousePosition);
            if (Physics.Raycast(ray, out hit, Mathf.Infinity))
            {
                if(hit.transform.gameObject.tag == "SelectableObj")
                {
                    Debug.Log(hit.transform.gameObject);
                    SetForce(hit.transform.gameObject);
                }
            }
        }
    }

    void SetForce(GameObject obj)
    {
        float power = Random.Range(500.0f, 1000.0f);
        Vector3 dir = new Vector3(Random.Range(-1.0f, 1.0f), Random.Range(0.0f, 1.0f), Random.Range(0.0f, 1.0f));
        dir = dir.normalized;
        obj.GetComponent<Rigidbody>().AddForce(dir * power);
    }
}
```

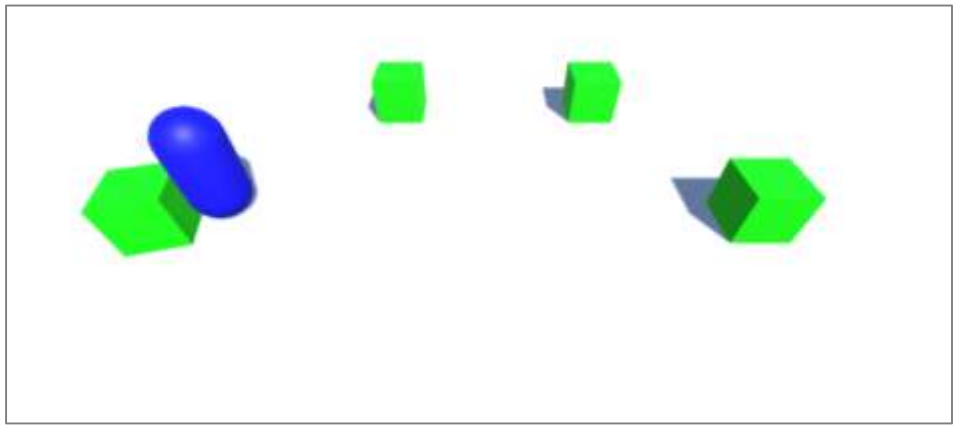
Collision (충돌)

- 충돌체 설정
  - 객체에 충돌체가 없는 경우
    - Component → Physics → Box/Sphere/Capsule Collider를 선택하여 추가
    - **객체들 사이의 충돌 검사를 위해서 충돌체 속성은 필수!!**
    - **충돌 검사 함수 실행을 위해서는 움직이는 객체에 강체(Rigidbody) 속성은 필수!!**
    - → Player 객체에 Rigidbody 속성 추가
      - Rigidbody의 IsKinematic : 체크



Collision (충돌)

- 충돌 확인

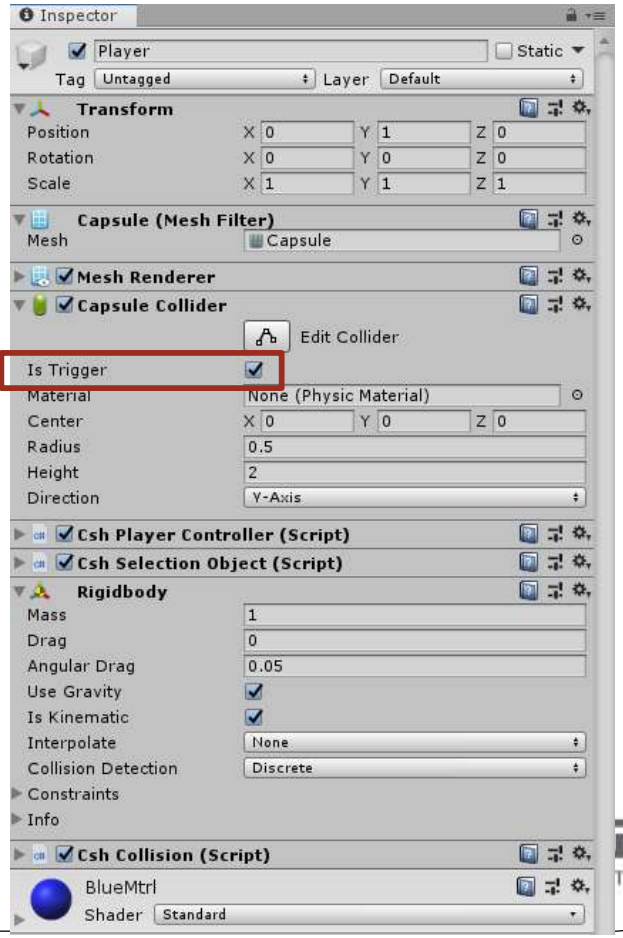


- 스크립트 생성 및 등록
  - 스크립트 폴더: cshCollision.cs 파일 생성
    - Player 객체에 등록

```
public class cshCollision : MonoBehaviour
{
    private void OnCollisionEnter(Collision collision)
    {
        if(collision.gameObject.tag == "SelectableObj")
        {
            Destroy(collision.gameObject);
        }
    }
}
```

Collision (충돌)

- 트리거
  - 객체들 간의 물리적 연산을 하지 않고 충돌을 감지
    - 물체가 접촉하였을 때 서로 튕겨져 나가지 않고 그냥 통과
    - 트리거에 의한 충돌 처리를 위한 조건
      - **충돌이 발생하는 두 객체 중 하나는 isTrigger에 체크**
      - **Player, Box 중 하나는 isTrigger 체크 필요**



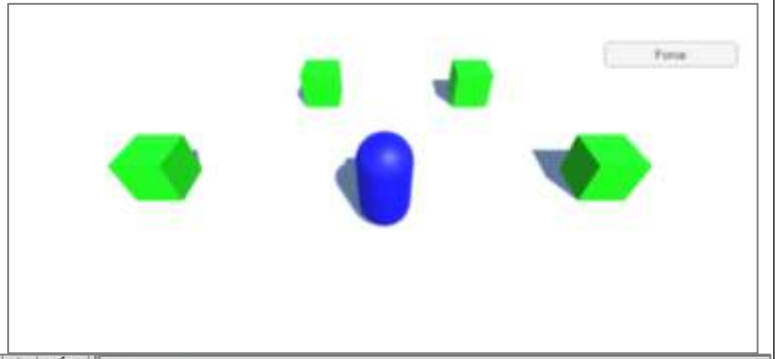
Collision (충돌)

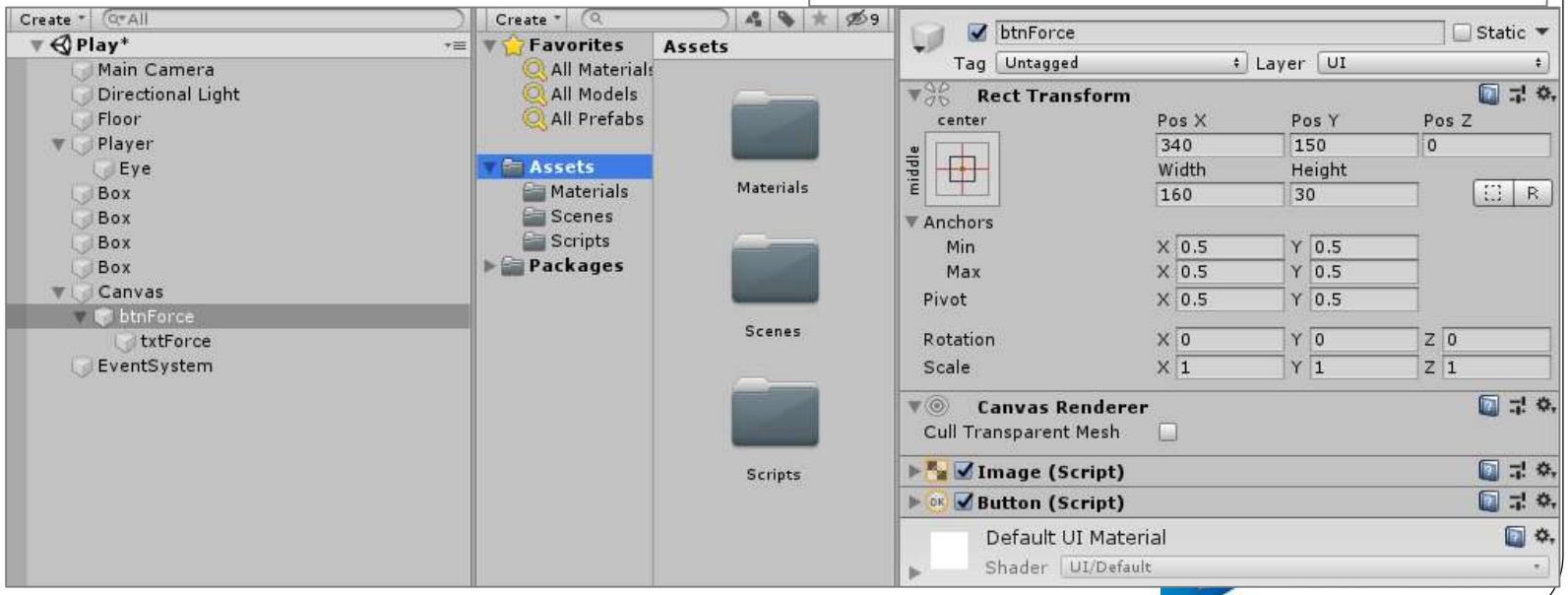
- 스크립트 수정
  - cshCollision.cs 코드 수정

```
public class cshCollision : MonoBehaviour
{
    /*
    private void OnCollisionEnter(Collision collision)
    {
        if(collision.gameObject.tag == "SelectableObj")
        {
            Destroy(collision.gameObject);
        }
    }*/

    private void OnTriggerEnter(Collider other)
    {
        if (other.gameObject.tag == "SelectableObj")
        {
            Destroy(other.gameObject);
        }
    }
}
```

GUI (Graphical User Interface)

- GUI 설정
  - 버튼, 이미지, 텍스트 등으로 구성된 사용자 인터페이스 처리
  - Create → Button
    - 버튼 명: btnForce
    - 글자 명: txtForce
    - PosX, PosY: 340, 150





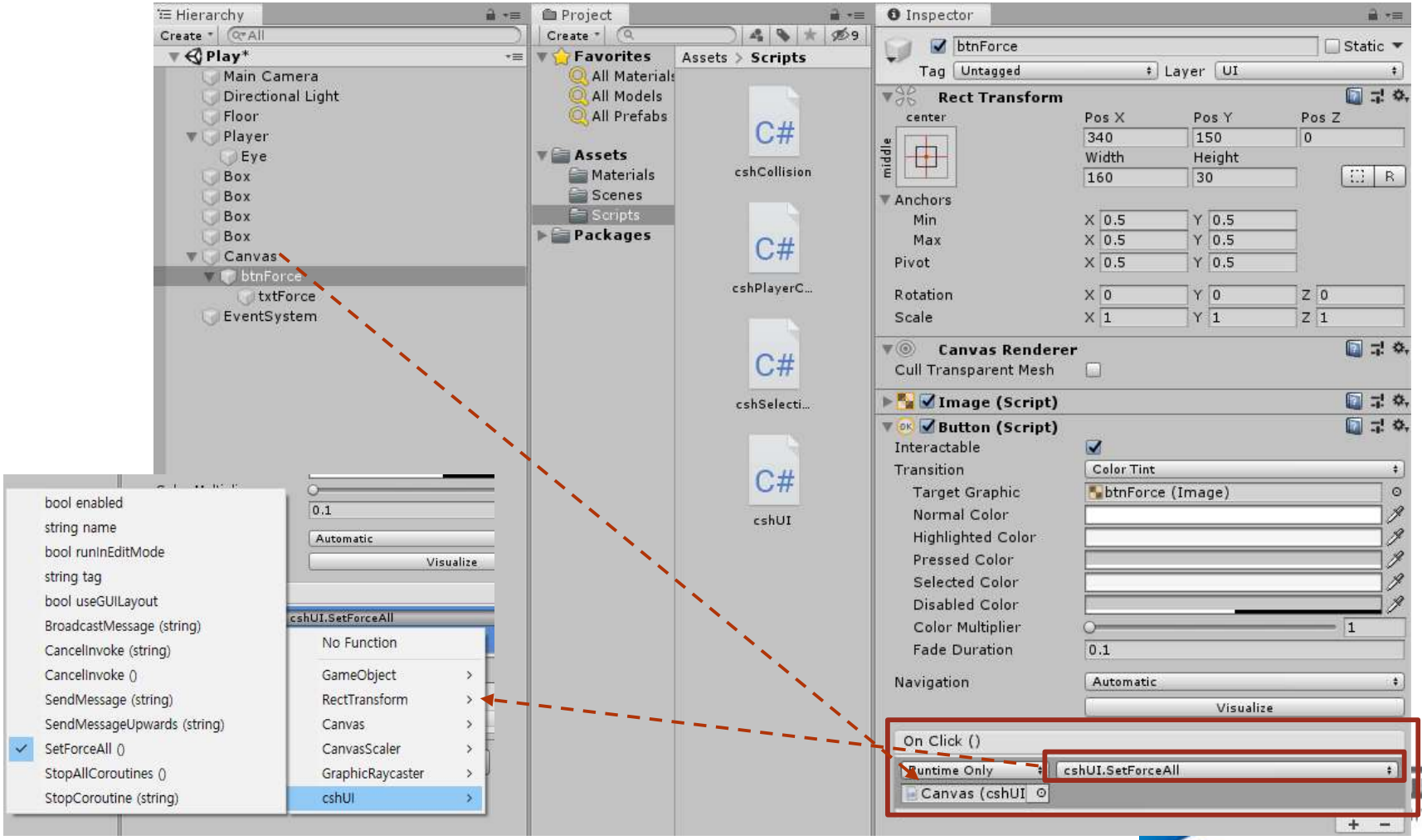
GUI (Graphical User Interface)

- 이벤트 처리
  - 스크립트 생성
    - 스크립트 폴더: cshUI.cs 파일 생성
      - Canvas 객체에 등록

```
public class cshUI : MonoBehaviour
{
    public void SetForceAll()
    {
        GameObject[] boxes = GameObject.FindGameObjectsWithTag("SelectableObj");
        for(int i = 0; i < boxes.Length; i++)
        {
            float power = Random.Range(500.0f, 1000.0f);
            Vector3 dir = new Vector3(Random.Range(-1.0f, 1.0f), Random.Range(0.0f, 1.0f), Random.Range(0.0f, 1.0f));
            dir = dir.normalized;
            boxes[i].GetComponent<Rigidbody>().AddForce(dir * power);
        }

    }
}
```

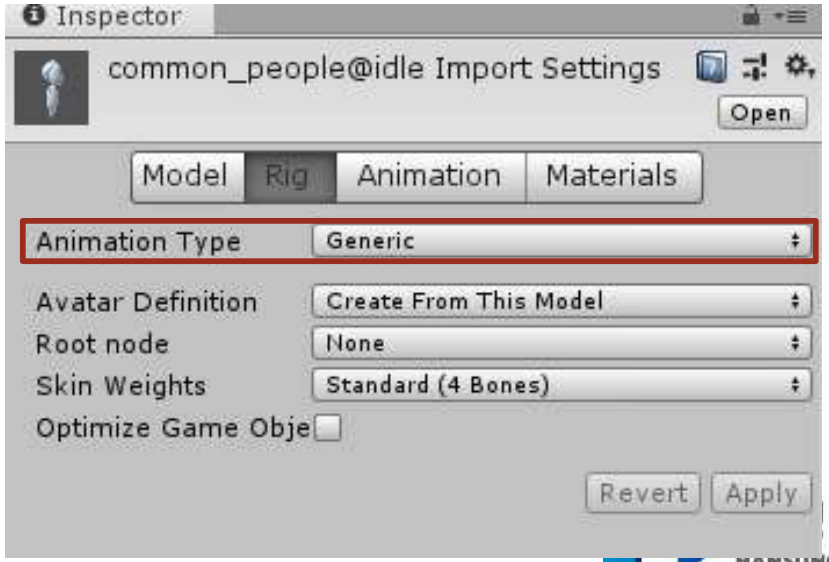
GUI (Graphical User Interface)

- 이벤트 설정



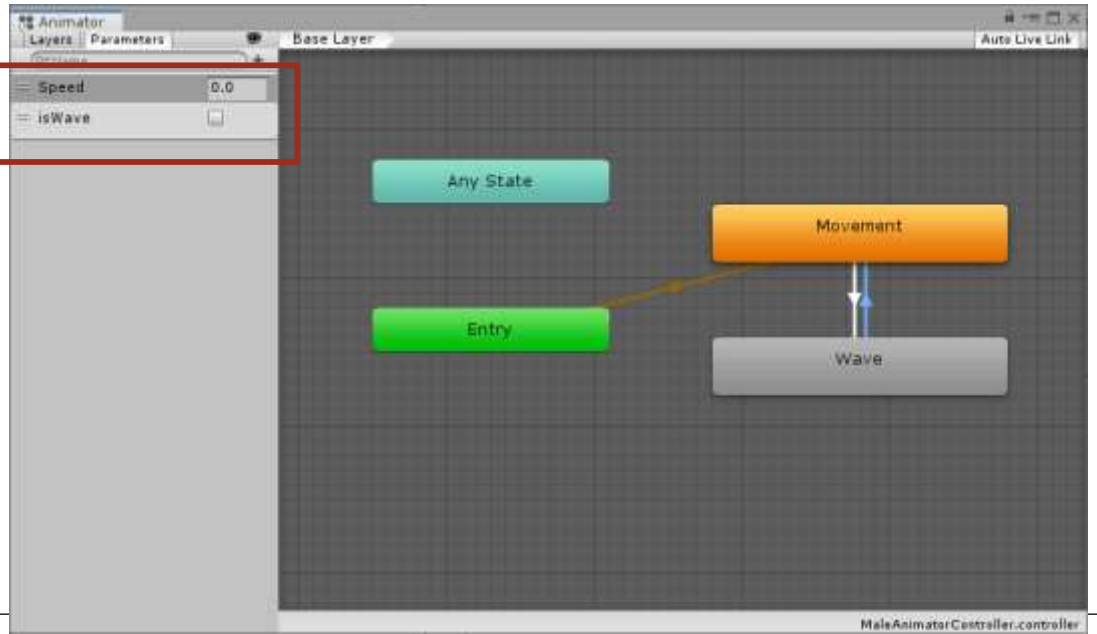
Animation (애니메이션)

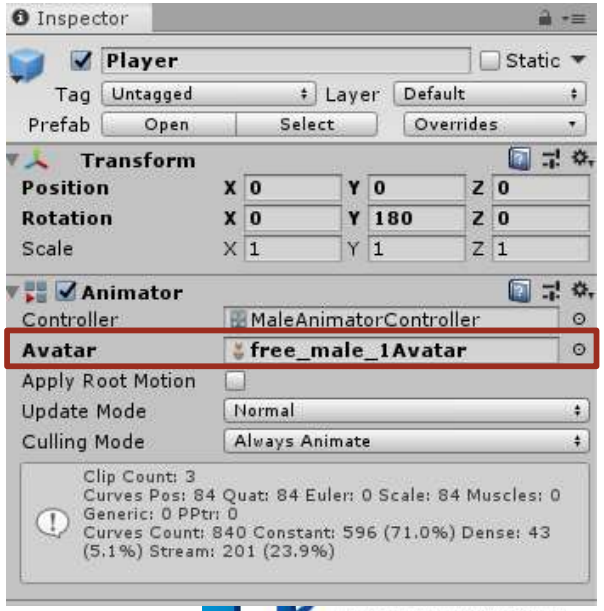
- 애니메이션 설정
  - Character Model → Models → free_male_1
  - Character Model → Animations
    - → common_people@idle
    - → common_people@run
    - → common_people@wave
  - Rig → Animation Type : Generic으로 변경



Animation (애니메이션)

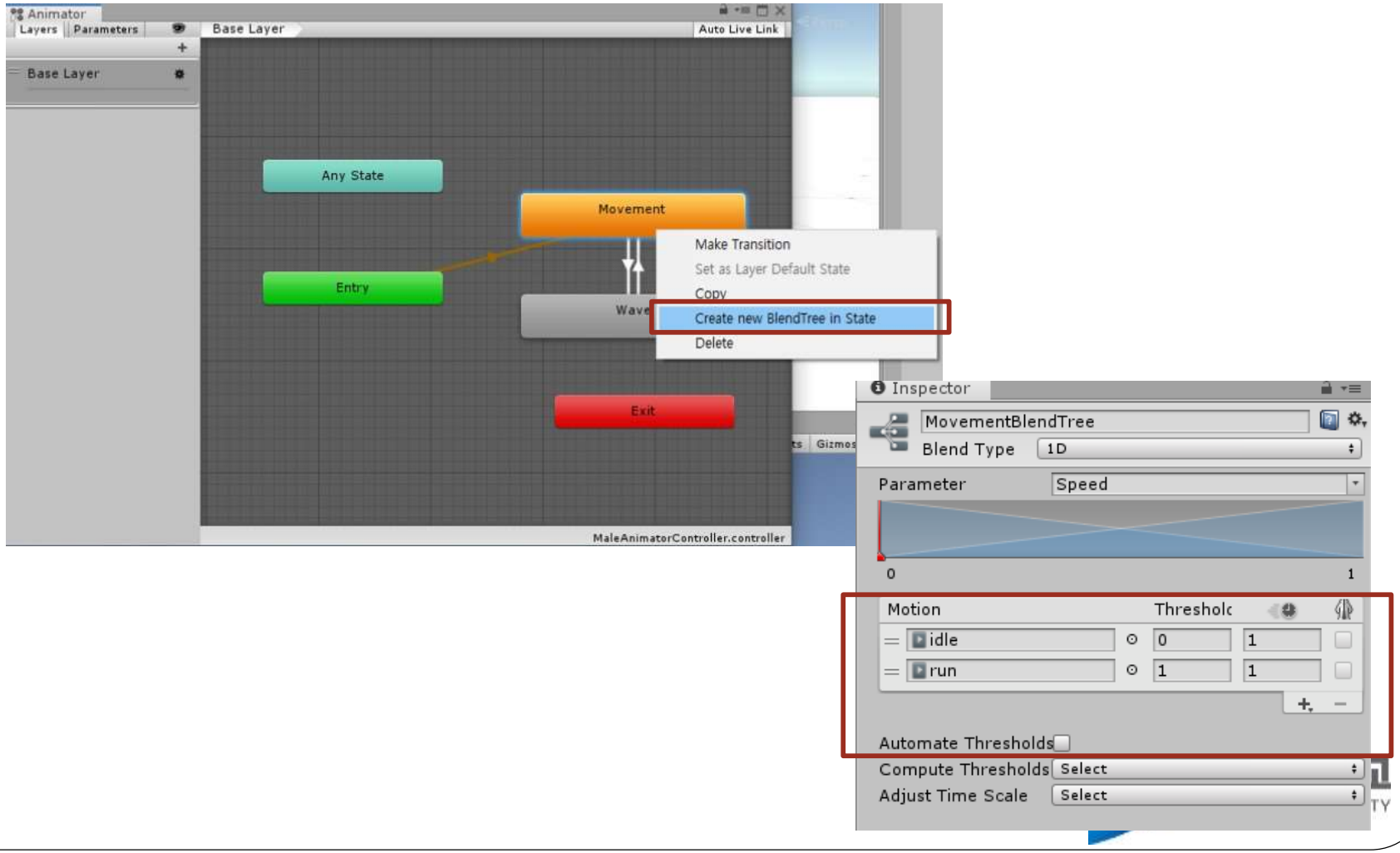
- 캐릭터 설정
  - Character Model → Prefabs → Base → High Quality
    - MaleFree1 모델 등록
      - 이름: Player
      - Rotation: 0, 180, 0
  - Project → Create → Animator Controller
    - 이름: MaleAnimatorController





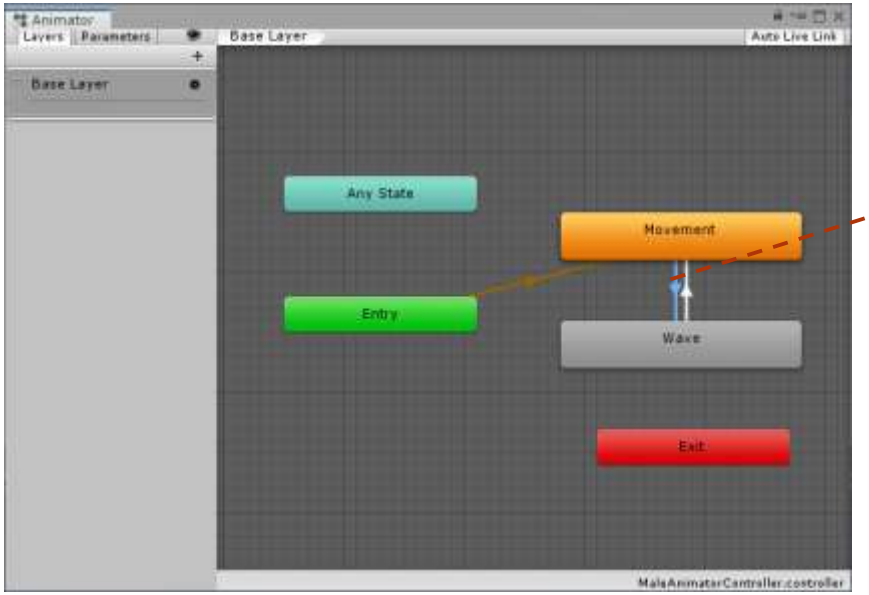
Animation (애니메이션)

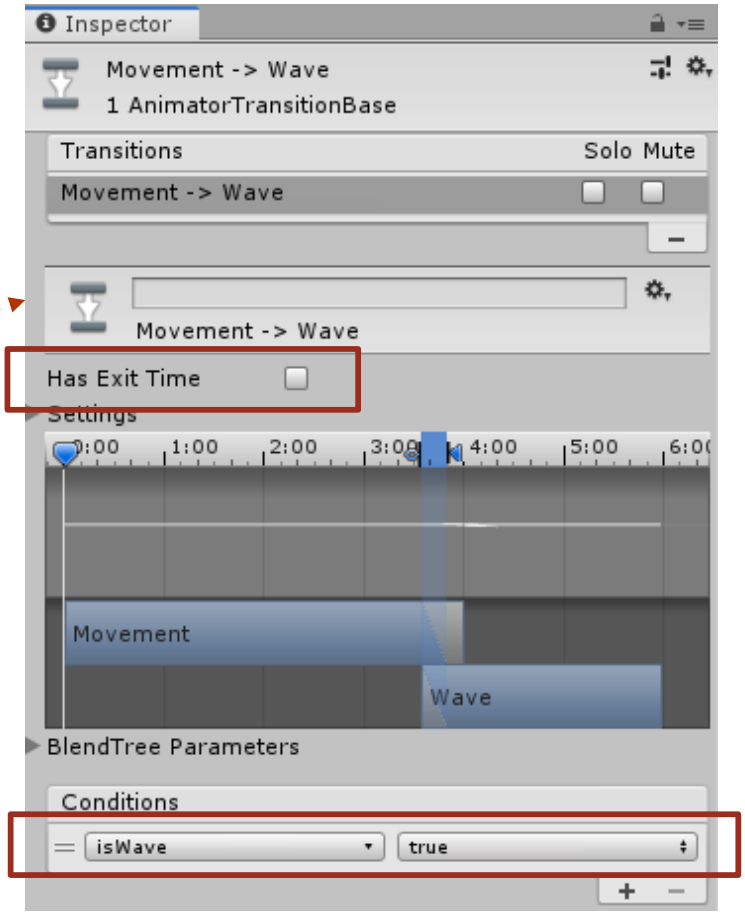
- 애니메이션 설정



Animation (애니메이션)

- 애니메이션 설정





Animation (애니메이션)

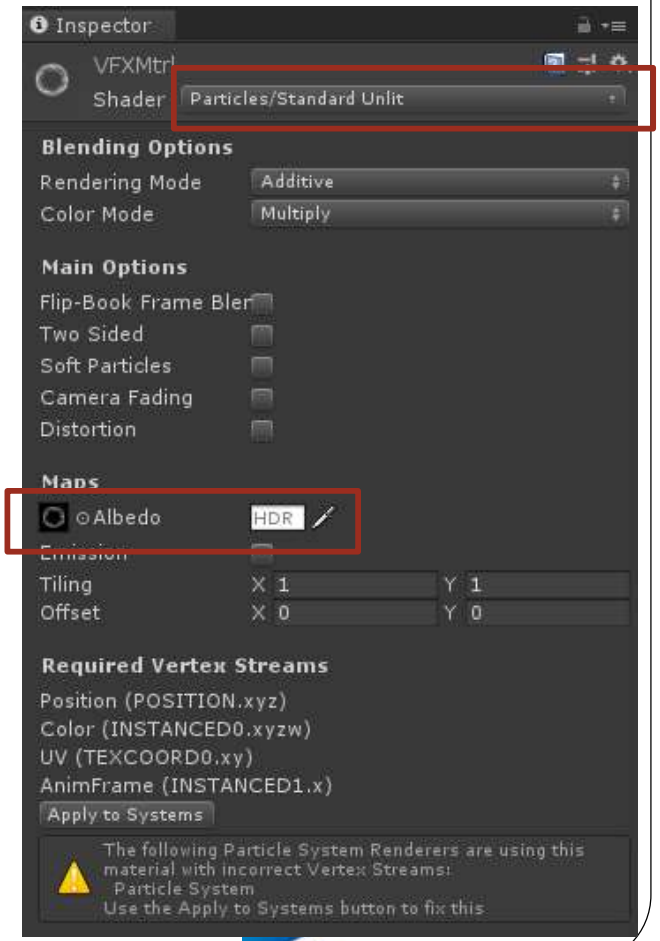
- 스크립트 생성 및 등록
  - 스크립트 폴더: cshAnimation.cs 파일 생성
    - Player 객체에 등록

```
public class cshAnimation : MonoBehaviour
{
    private Animator ani;
    private float speed = 0.0f;
    void Start()
    {
        ani = GetComponent<Animator>();
    }

    void Update()
    {
        if (Input.GetMouseButton(0))
        {
            speed += 0.01f;
            if (speed > 1.0f) speed = 1.0f;
        }else if (Input.GetMouseButtonUp(0))
        {
            speed = 0.0f;
        }else if (Input.GetMouseButtonDown(1))
        {
            ani.SetBool("isWave", true);
        }else if (Input.GetMouseButtonUp(1))
        {
            ani.SetBool("isWave", false);
        }
        ani.SetFloat("Speed", speed);
    }
}
```

Particle (파티클)

- 파티클 시스템
  - GameObject → Effect → Particle System
  - Project → Create → Material : VFXMtrl

Texture 선택



Particle (파티클)

- 파티클 시스템
  - 파티클 속성 변경

색상 선택