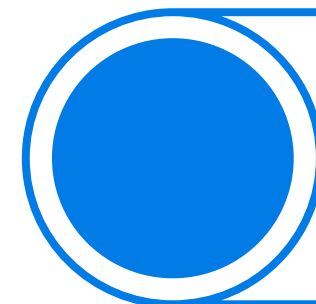




INTERACTION을 활용한

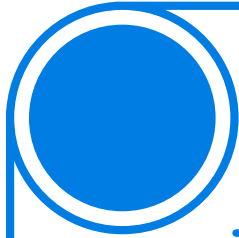
# 유니티 프로젝트



20204043 문벼리



## 1. 기획



### 게임 스토리

코코와 나나는 사랑스러운 커플입니다. 그들은 꽃과 나비, 달콤한 노래와 장미 향기가 가득한 숲에서 살고 있습니다. 어느 날, 코코는 나나에게 결혼 프로포즈를 하려고 마음 먹었습니다. 나나를 위해 완벽한 프로포즈를 준비하고 싶었습니다.

프로포즈를 위해 코코는 자신의 사랑을 표현할 수 있는 엘프 버섯을 찾으려고 결심합니다. 그러나 그 버섯은 도시 어딘가에 있다는 소문뿐이며, 코코는 나나를 위해 버섯을 찾아 떠나기로 결심해 혼자 도시로 향합니다.

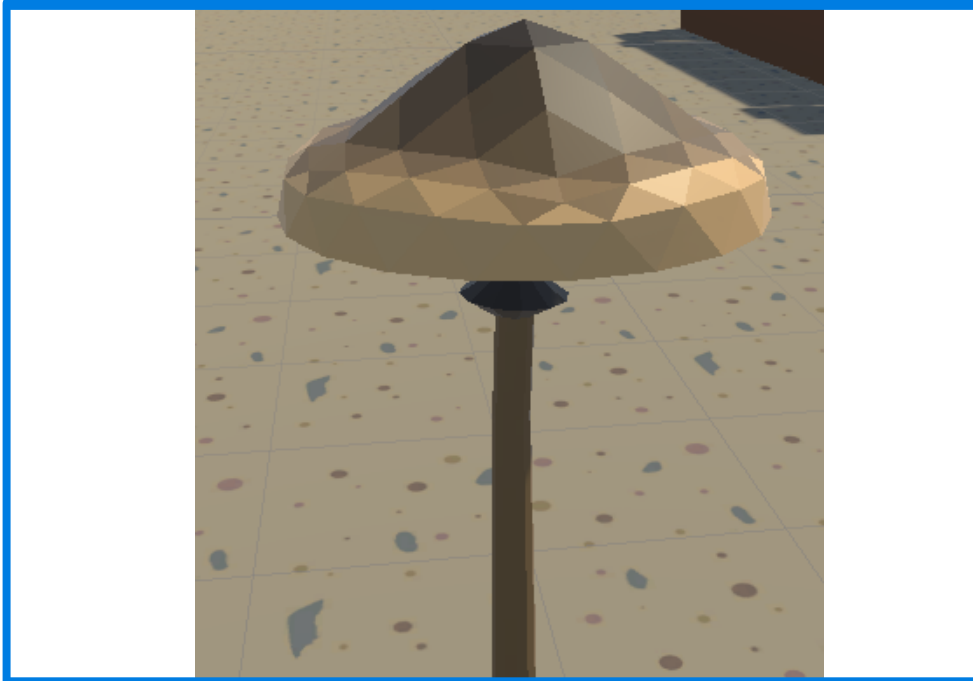
코코는 엘프 버섯을 찾아 나나에게 프로포즈를 성공 할 수 있을까요?

## 1. 기획

## 게임 아이템

01

방망이버섯



- ▶ 버섯을 획득하면 울타리나 상자를 부술 수 있다.

02

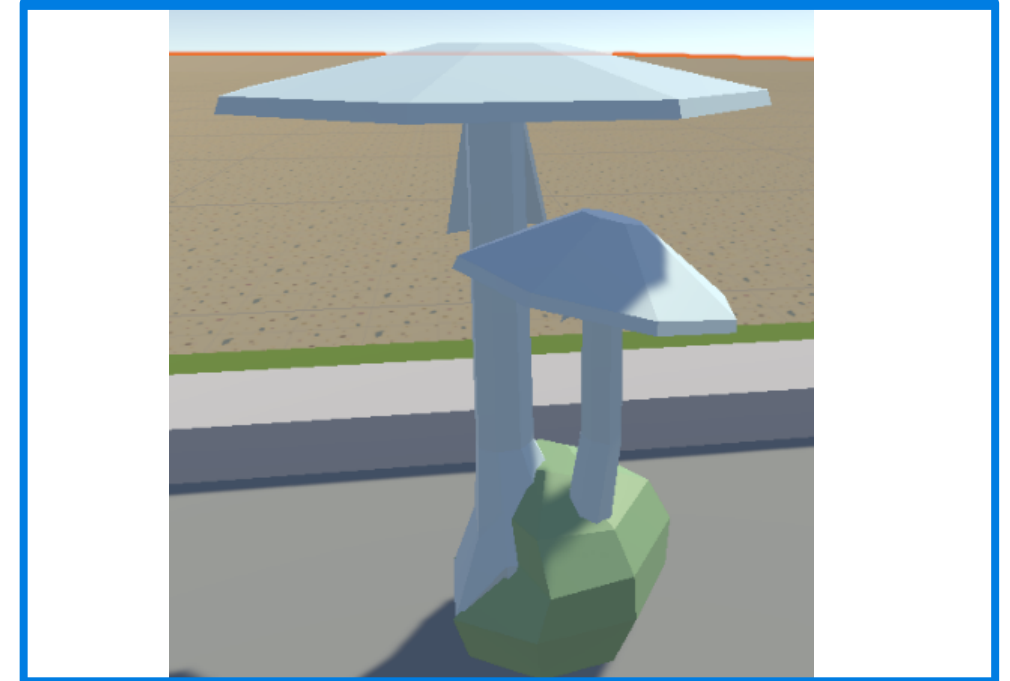
파워버섯



- ▶ 버섯을 섭취하면 점프와 대쉬가 가능해진다. 버섯을 여러개 먹을수록 점점 점프력이 강해진다.

03

엘프버섯



- ▶ 코코가 프로포즈를 위해 찾는 버섯. 찾으면 게임이 클리어된다.

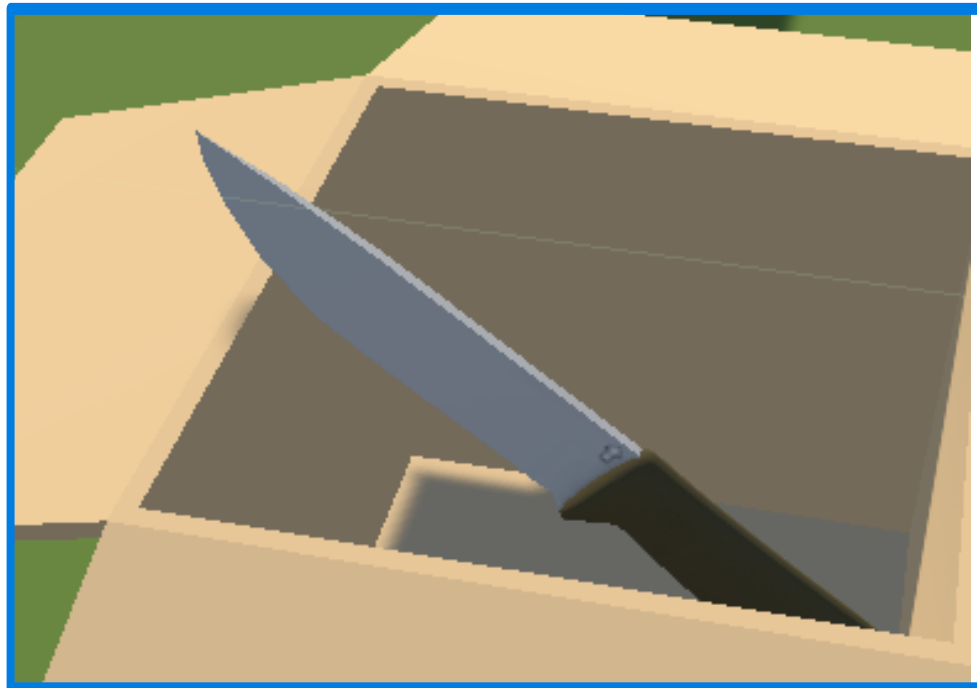


## 1. 기획

# 게임 아이템

04

칼



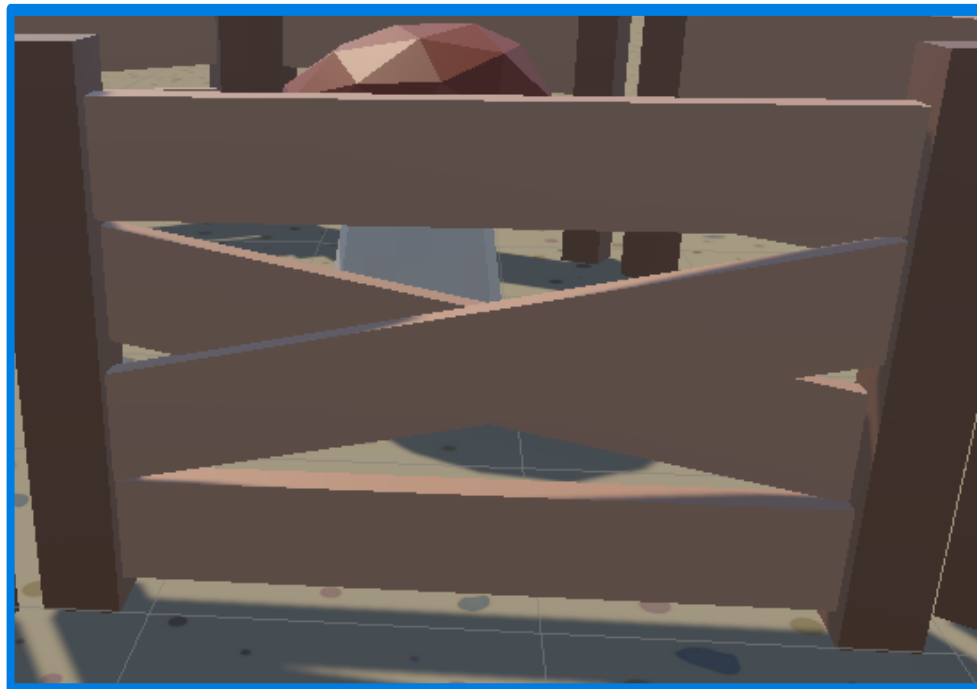
› 칼을 획득하면 박스를 찢을 수 있다.

## 1. 기획

# 파괴 가능 오브젝트

01

울타리



▶ 방망이버섯으로 부술 수 있다.

02

박스

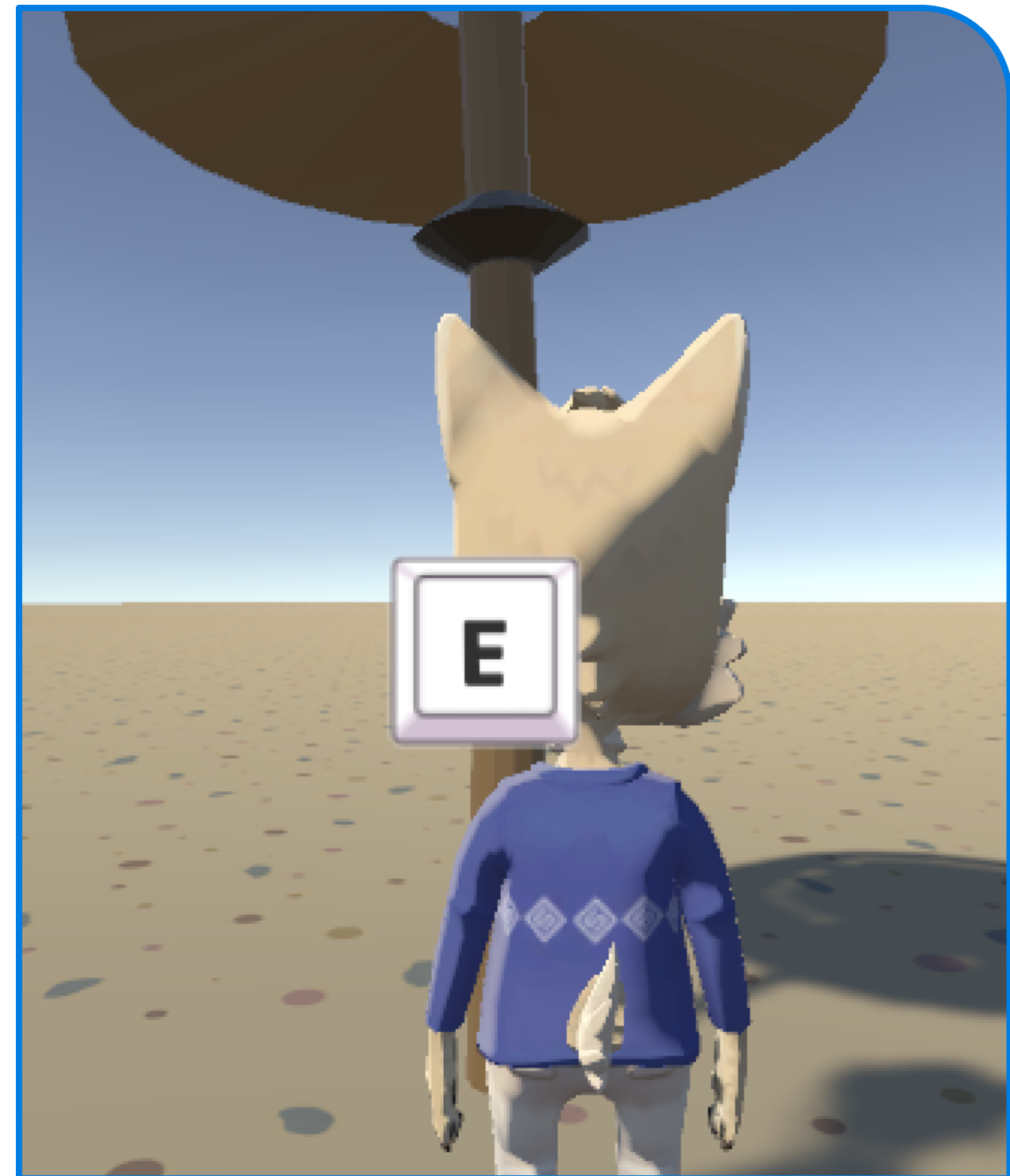


▶ 칼로 찢을 수 있다. 운이 좋으면 파워버섯이 나올지도?

## 1. 기획

### 아이템 획득

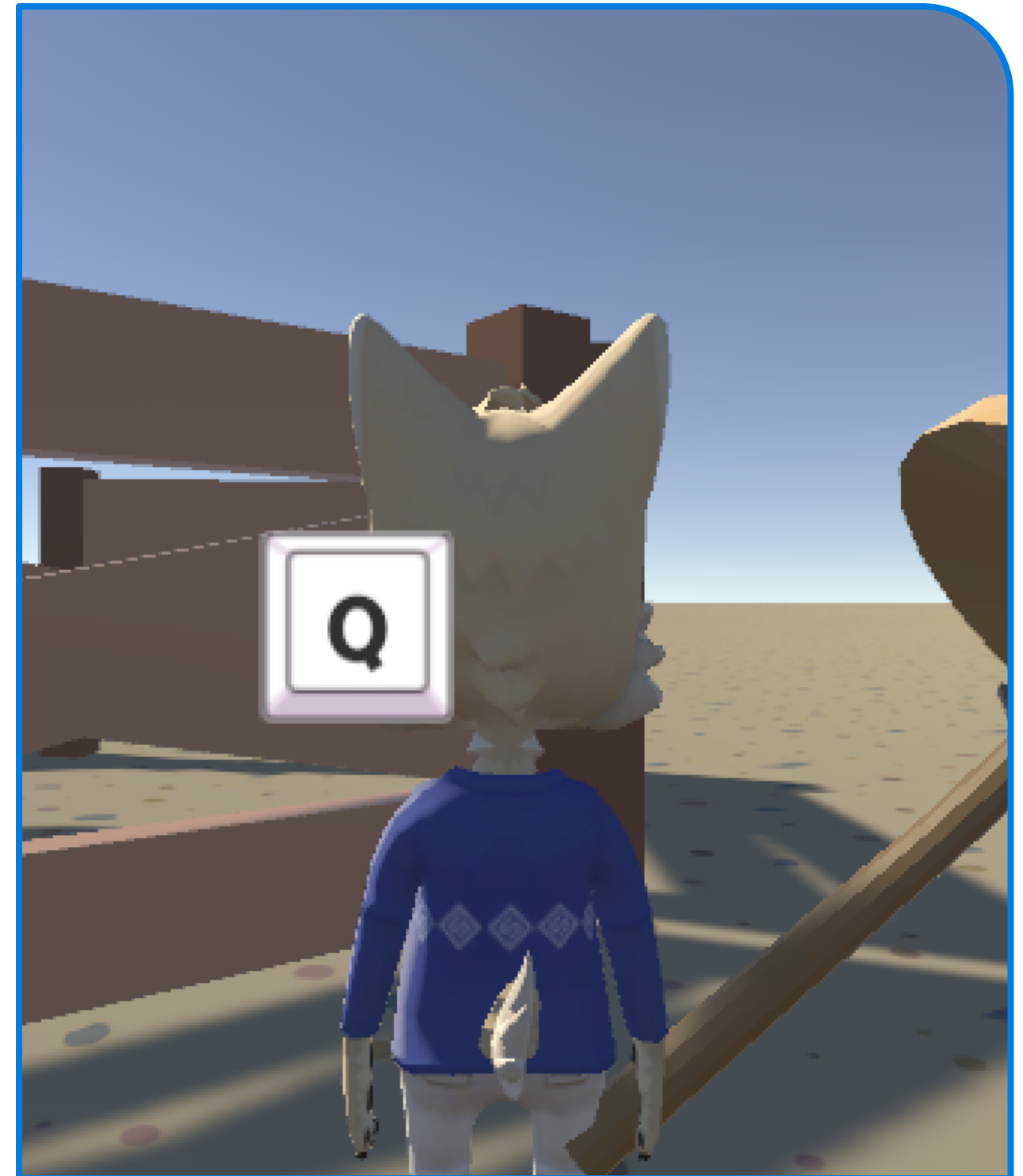
- 오브젝트에 가까이 다가가면 E키를 눌러서 획득할 수 있다.
- 아이템이라는 것을 표현하기 위해 트리거 안에 가까이 다가가갈 경우 UI로 눌러야하는 키를 띄워준다.
- 섭취 아이템과 장착 아이템을 구분해서 제작하였다.
- 장착 가능 오브젝트를 획득하면 자동으로 해당 오브젝트가 플레이어에 장착된다.
- 오브젝트가 장착되어있는지 여부는 플레이어에 붙어있는 PlayerItem 스크립트로 관리해 주었고, 오브젝트를 획득하는 것은 아이템에 붙어있는 Collectable로 관리해주었다.



## 1. 기획

### 오브젝트 파괴

- 오브젝트에 가까이 다가가면 Q키를 눌러서 파괴할 수 있다.
- 파괴 가능 아이템이라는 것을 표현해주기 위해 아이템 파괴를 위한 오브젝트를 장착하고 트리거 안에 가까이 다가갈 경우, UI로 눌러야하는 키를 띄워준다.
- DotTween을 활용해서 DOSHAKE 이후 쓰러지게 하는 애니메이션 효과를 추가했다.
- 파괴할 때, 플레이어가 때리는 모션을 하도록 하는 트리거를 포함하였다.

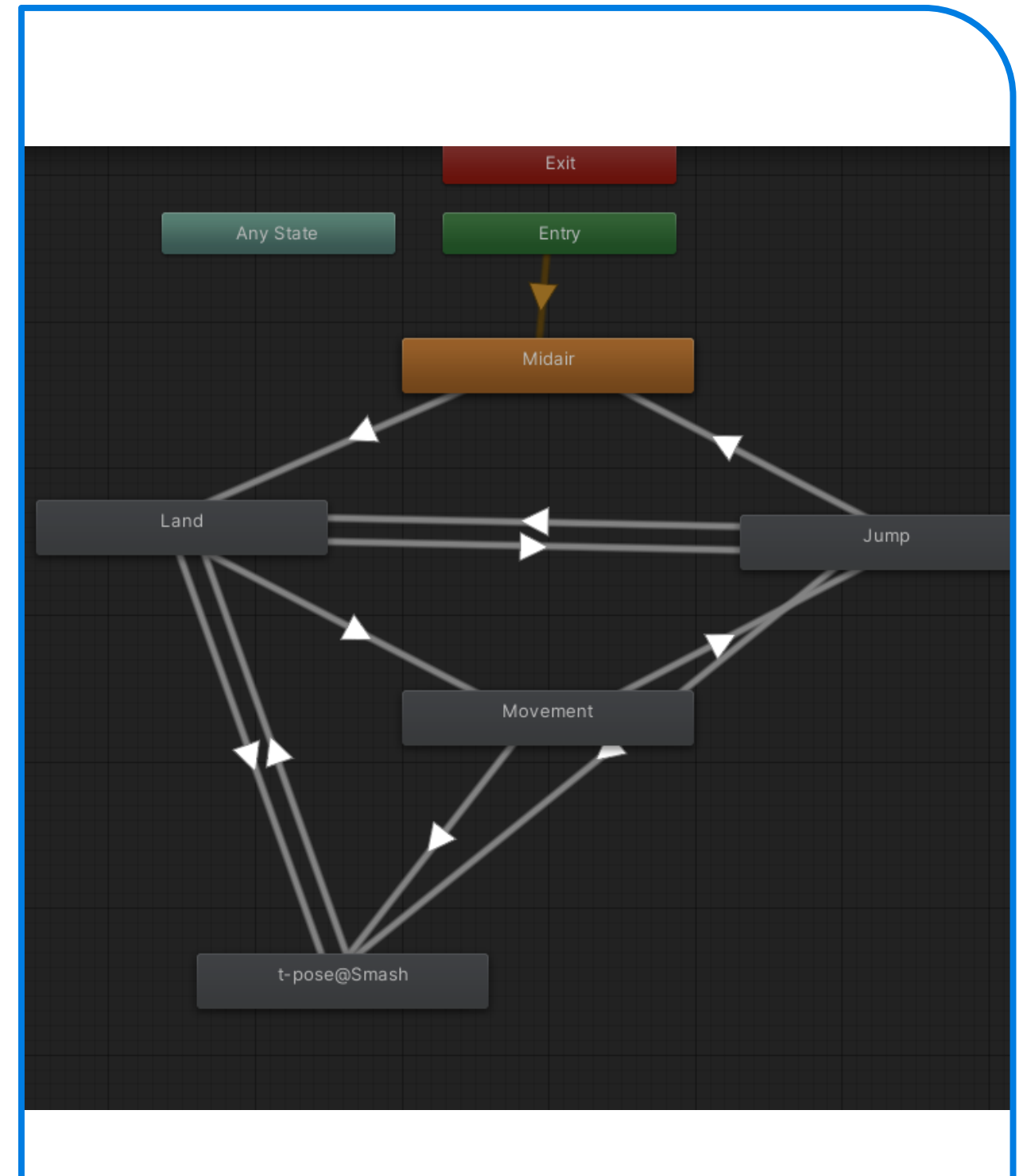




## 2. 구현방법

### 애니메이터

- 서있을 때 Idle 모션, 뛸 때 Jump 모션, 움직일 때 Movement 모션, 도구를 이용해 부술 때 Smash 모션이 나오도록 설정했다.
- Smash모션은 믹사모에서 직접 매핑해서 넣었고, 나머지는 플레이어 오브젝트 에셋에 추가되어 있던 모션을 그대로 사용했다.



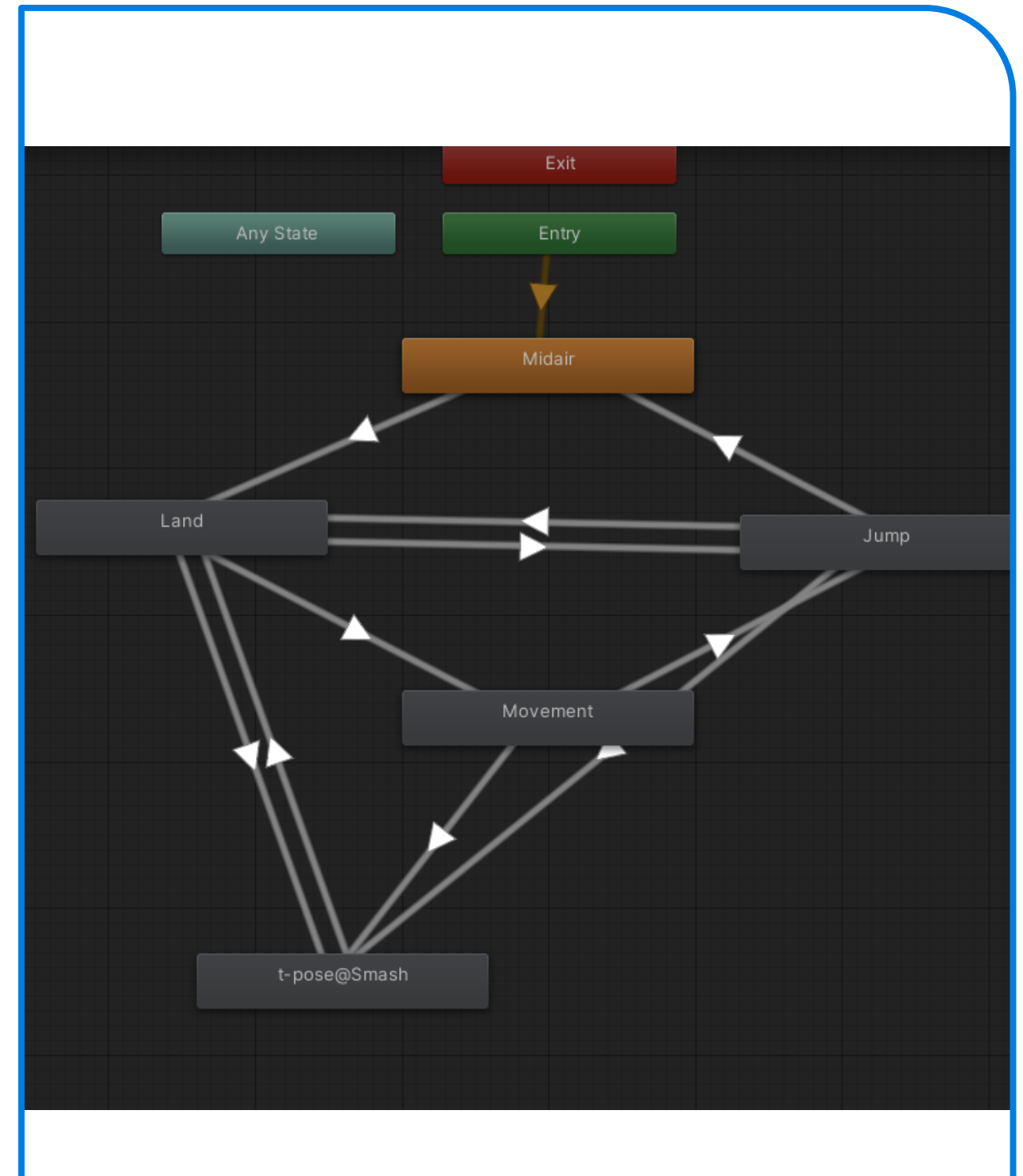




## 2. 구현방법

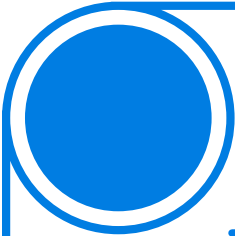
### 애니메이터

- 서있을 때 Idle 모션, 뛸 때 Jump 모션, 움직일 때 Movement 모션, 도구를 이용해 부술 때 Smash 모션이 나오도록 설정했다.
- Smash모션은 믹사모에서 직접 매핑해서 넣었고, 나머지는 플레이어 오브젝트 에셋에 추가되어 있던 모션을 그대로 사용했다.





## 2. 구현방법



### 상호작용 구현

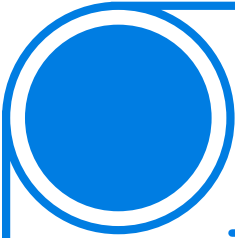
```
private void OnTriggerEnter(Collider other)
{
    isActive = true;
    if (other.gameObject.tag == "Player")
    {
        StartCoroutine(
            IEnumKeyShow(
                new Vector2(UnityEngine.Random.Range(-30, 30), UnityEngine.Random.Range(-30, 30)), pressAction,
                4.0f
            ));
    }
}

private void OnTriggerExit(Collider other)
{
    if (other.gameObject != this && other.gameObject.tag == "Player")
    {
        isCancel = true;
        isActive = false;
    }
}
```

OnTriggerStay를 사용해서 if문을 계속 호출한다면 성능 이슈가 생길 것이라고 생각했고, isActive라는 boolean값이 TriggerEnter할 땐 true, TriggerExit할땐 false가 되도록 하여 이 bool값으로 트리거 범위 안에 있는지 판단하였다.



## 2. 구현방법



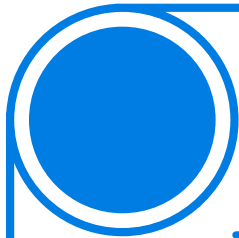
### GetItem 상호작용 구현

- 획득할 아이템 오브젝트에 붙일 Collectable 클래스를 만들었다.
- 오브젝트를 구별할 Item을 enum으로 분류해서 지정하였다.

```
public enum Item
{
    power,
    bat,
    knife,
    Elf,
}
```



## 2. 구현방법



### GetItem 상호작용 구현

아이템별로 다른 기능들을 작성해주었다.

- 방망이버섯 -> 방망이 활성화
- 칼 -> 칼 활성화
- 파워버섯 -> 점프력 상승, 대쉬속도 상승
- 엘프버섯 -> 게임 클리어 UI 활성화

```
// Update is called once per frame
void Update()
{
    if (isActive && Input.GetKeyDown(KeyCode.E))
    {
        isKeyPressed = true;
        transform.DOShakePosition(1f, 0.5f, 5, 0, false, true).OnComplete(() =>
        {
            if (item == Item.bat)
            {
                player.GetComponent<PlayerItem>().batMushroom = true;
                player.GetComponent<PlayerItem>().knifeItem = false;
                player.GetComponent<PlayerItem>().knife.SetActive(false);
                player.GetComponent<PlayerItem>().bat.SetActive(true);
            }

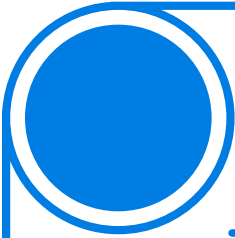
            else if (item == Item.power)
            {
                player.GetComponent<PlayerItem>().powerMushRoom = true;
                player.GetComponent<PlayerController>().m_jumpForce += 9.0f;
                player.GetComponent<PlayerController>().m_walkScale += 3.0f;
            }

            else if (item == Item.knife)
            {
                player.GetComponent<PlayerItem>().knifeItem = true;
                player.GetComponent<PlayerItem>().batMushroom = false;
                player.GetComponent<PlayerItem>().bat.SetActive(false);
                player.GetComponent<PlayerItem>().knife.SetActive(true);
            }

            else if (item == Item.Elf)
            {
                clearUI.SetActive(true);
            }
        });
        Destroy(this.gameObject);
    }
}
```



## 2. 구현방법



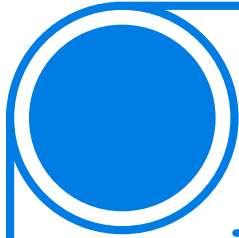
### Destroy 상호작용 구현

- 획득할 아이템 오브젝트에 붙일 DestroyableObject 클래스를 만들었다.
- 무기에 따라 오브젝트를 구분하여 enum으로 분류해서 지정하였다.

```
public enum Destroyable
{
    batDestroyable,
    knifeDestroyable
}
```



## 2. 구현방법



### Destroy 상호작용 구현

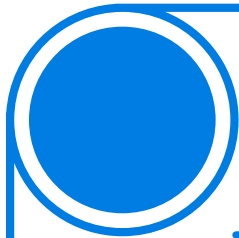
필요한 무기에 따라 해당 아이템을 장착하고 있는지 확인하고, 장착하고 있다면 코드를 실행한다

- 플레이어 애니메이터에서 부수는 모션의 Trigger 활성화
- DOTween툴을 활용해 오브젝트가 쓰러지는 애니메이션 재생한 이후 오브젝트 파괴

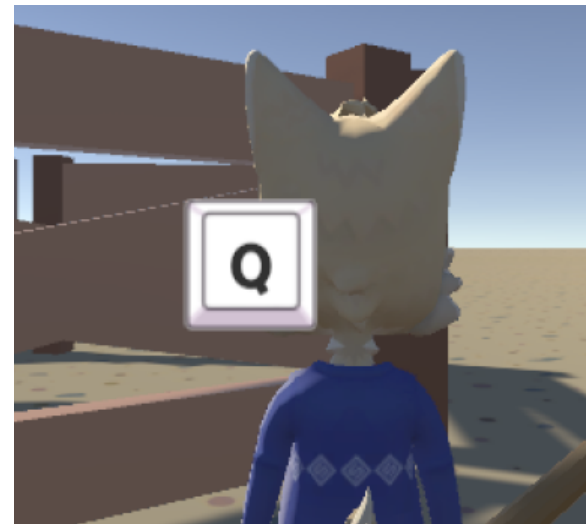
```
// Update is called once per frame
void Update()
{
    if (isActive && Input.GetKeyDown(KeyCode.Q))
    {
        isKeyPressed = true;
        player.GetComponent<PlayerController>().m_animator.SetTrigger("Smash");
        transform.DOShakePosition(1f, 0.5f, 5, 0, false, true).OnComplete(() =>
        {
            transform.DORotate(new Vector3(-90, 0, 0), 1).SetRelative(true).SetEase(Ease.InBack).OnComplete(() =>
            {
                Destroy(this.gameObject);
            });
        });
    }
}
```



## 2. 구현방법



### 범위 안에 들어갈 경우 UI 켜기



해당 키가 눌리거나  
범위 바깥으로 나가기 전까지  
계속 화면에 키 UI를 띄워 주는 코드이다.  
DOTween을 사용해서  
UI의 움직임을 부드럽게 만들었다.

```
public IEnumerator IEnumKeyShow(Vector2 pos, Action pressAction, float showTime = 4.0f)
{
    isKeyPressed = false;
    isCancel = false;
    uiKeyPanel.keyText.text = "Q";
    uiKeyPanel.DOKill();
    uiKeyPanel.GetComponent<RectTransform>().anchoredPosition = pos;
    uiKeyPanel.transform.localScale = Vector3.one;
    uiKeyPanel.keyRect.DOKill();
    uiKeyPanel.keyRect.anchoredPosition = Vector2.zero;
    uiKeyPanel.keyRect.DOLocalMoveY(-12f, 0.2f).SetLoops(-1, LoopType.Yoyo).SetUpdate(true);

    while (!isKeyPressed && !isCancel)
    {
        yield return null;
    }


    uiKeyPanel.transform.DORewind();
    uiKeyPanel.transform.DOScale(Vector3.zero, 0.4f).SetEase(Ease.InBack).SetUpdate(true);

    if (pressAction != null && isKeyPressed) { pressAction(); }
}
```





# 사용한 에셋



FLOAT3D

[Free] Modern Combat K...

4.4 MB


구매 시간: 2023년 10월 5일

Organization: molang9876(Perso...

최근 업데이트: 2020년 9월 11일 • 버전: 1.0

First release

[레이블 추가](#) [에셋 숨기기](#)



BROKEN VECTOR

Low Poly Fence Pack

230.5 KB


구매 시간: 2023년 10월 4일

Organization: molang9876(Perso...

최근 업데이트: 2018년 7월 10일 • 버전: 1.1

Updated to Unity 5.6

[레이블 추가](#) [에셋 숨기기](#)



PALMOV ISLAND

Low Poly Houses Free Pa...

1.5 MB


구매 시간: 2023년 10월 4일

Organization: molang9876(Perso...

최근 업데이트: 2023년 2월 16일 • 버전: 1.0.0

First release

[레이블 추가](#) [에셋 숨기기](#)



PAPERSY

Low Poly Mushrooms Pack

541.8 KB


구매 시간: 2023년 10월 4일

Organization: molang9876(Perso...

최근 업데이트: 2021년 12월 3일 • 버전: 1.0

First release

[레이블 추가](#) [에셋 숨기기](#)



DRAFTPUNK STUDIOS

Low Poly Simple Urban C...

2.1 MB


구매 시간: 2023년 10월 4일

Organization: molang9876(Perso...

최근 업데이트: 2023년 1월 3일 • 버전: 1.0.0

First release

[레이블 추가](#) [에셋 숨기기](#)



BOKI

BOKI - Low Poly Nature

98.7 MB

구매 시간: 2023년 10월 4일

Organization: molang9876(Perso...

최근 업데이트: 2021년 12월 6일 • 버전: 1.0

First release

[레이블 추가](#) [에셋 숨기기](#)



SUPERCYAN

Character Pack: Free Ani...

13.1 MB


구매 시간: 2023년 10월 4일

Organization: molang9876(Perso...

최근 업데이트: 2022년 1월 7일 • 버전: 1.0.0

CHANGELOG v1.0.0 (2021-09-28) - First release

[레이블 추가](#) [에셋 숨기기](#)



UNITY TECHNOLOGIES

Particle Pack

137.2 MB

구매 시간: 2023년 2월 6일


Organization: molang9876(Perso...

최근 업데이트: 2023년 3월 8일 • 버전: 1.7

**Asset is under Unity Companion License.**

[more](#)

[레이블 추가](#) [에셋 숨기기](#)



DIGITAL RUBY (JEFF JOHNSON)

Fire & Spell Effects

26.5 MB


구매 시간: 2023년 2월 6일

Organization: molang9876(Perso...

최근 업데이트: 2022년 5월 18일 • 버전: 1.0.3

Fix typo

[레이블 추가](#) [에셋 숨기기](#)



DEMIGIANT

DOTween (HOTween v2)

221.6 KB

구매 시간: 2023년 1월 14일

Organization: molang9876(Perso...

최근 업데이트: 2023년 8월 11일 • 버전: 1.2.745

**See FULL CHANGELOG**

[more](#)

[레이블 추가](#) [에셋 숨기기](#)





**감사합니다 :)**

