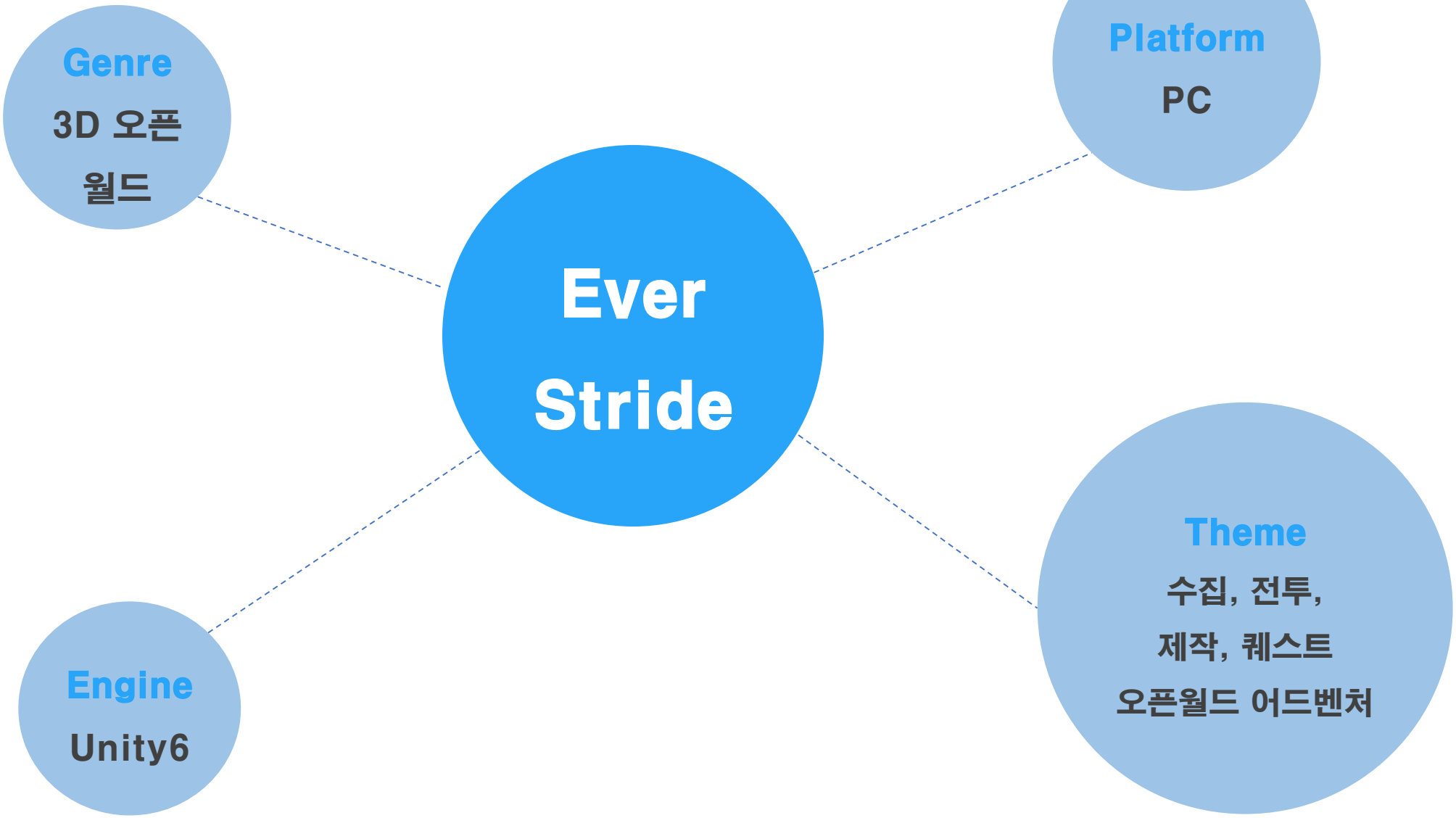
hjhmoon61@naver.com
https://github.com/HaJinhyeok/3D_EverStride.git
https://youtu.be/5QFXvFK136M

0. 목차

EverStride

1. 게임 개요


Genre
3D 오픈
월드
Platform
PC
Ever
Stride
Engine
Unity6
Theme
수집, 전투,
제작, 퀘스트
오픈월드 어드벤처

2. 게임 진행 게임 시작

Input.anyKeyDown으로

입력 대기

Start와 Exit 버튼 배치

2. 게임 진행 수집

벌목

채광

벌목과 채광을 통해
나무와 돌 자원을 수집

2. 게임 진행 제작

제작대

제작판

마을에 위치한 제작대를

통해 아이템 제작

2. 게임 진행 전투

레이드 입장

보스 전투

마을에 위치한 포탈을
통해 보스 레이드 입장

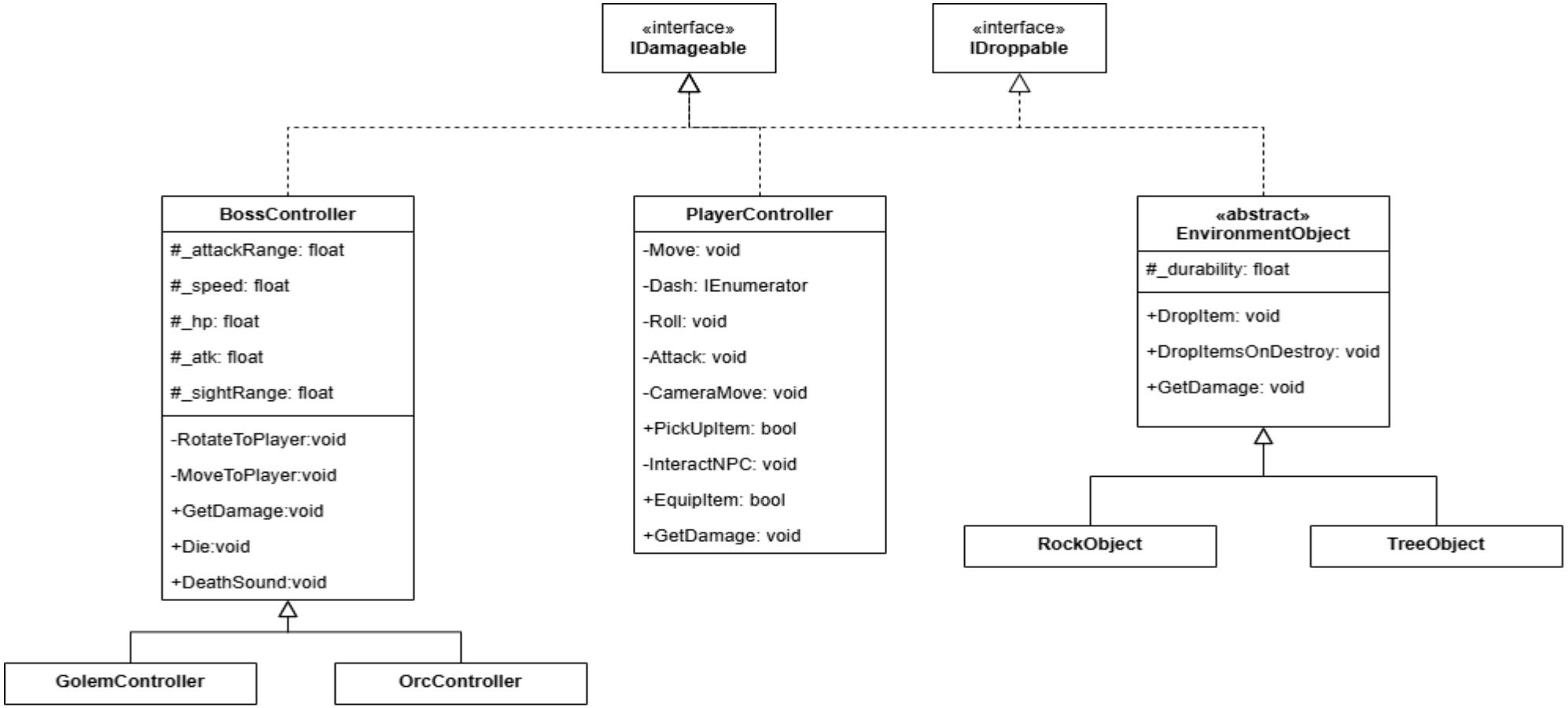
2. 게임 진행 퀘스트

상호작용

퀘스트 생성

마을의 NPC와 상호작용
몬스터 처치, 아이템 수집

3. 구조 및 설계 주요 클래스 상속도



```
using UnityEngine;

참조 9개
public interface IDamageable
{
    참조 12개
    public void GetDamage(GameObject attacker, float damage, int bonusDamage = 1, Vector3 hitPos = default);
}
```

대상으로부터 입는 대미지 계산

```
using UnityEngine;


public interface IAttackable
{

    public bool DoAttack(GameObject target, float damage, Vector3 hitPos = default);
}
```

대상에게 가하는 대미지 계산

```
public interface IDroppable
{
    참조 9개
    public void DropItem();

    참조 5개
    public void DropItemsOnDestroy();
}
```

대미지를 입는 경우 아이템 드롭 기능

3. 구조 및 설계 카메라

플레이어가 화면의 중앙보다
왼쪽에 배치된 3인칭 시점

CamAxis와 MainCamera를
Parent-Child 관계로 설정

```
// * 카메라
_camera = Camera.main;
_camAxis = new GameObject(Define.CamAxis).transform;
_camera.transform.parent = _camAxis;
_camera.transform.localPosition = _camOffset;
ConversationCamera?.gameObject.SetActive(false);

// * 그 외
```

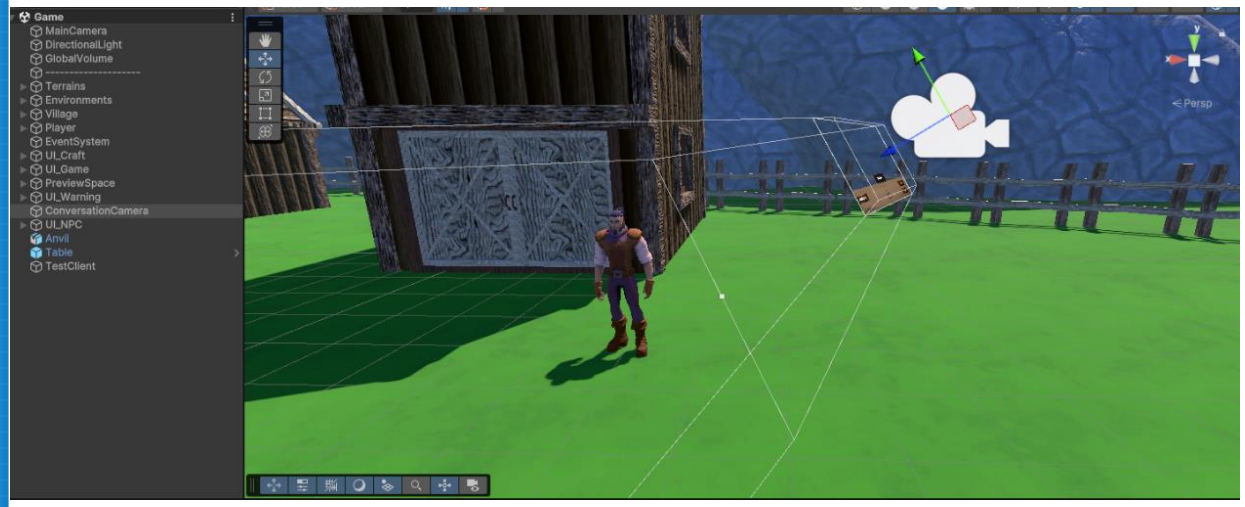
CamAxis를 축으로 회전시켜
MainCamera가 일정한
localPosition에 존재

```
void CameraMove()
{
    _mouseX += Input.GetAxis(Define.MouseX);
    _mouseY -= Input.GetAxis(Define.MouseY);

    if (_mouseY > 10)
        _mouseY = 10;
    if (_mouseY < 0)
        _mouseY = 0;

    _camAxis.rotation = Quaternion.Euler(new Vector3(_camAxis.rotation.x + _mouseY, _camAxis.rotation.y + _mouseX, 0) * _camSpeed);
}
```

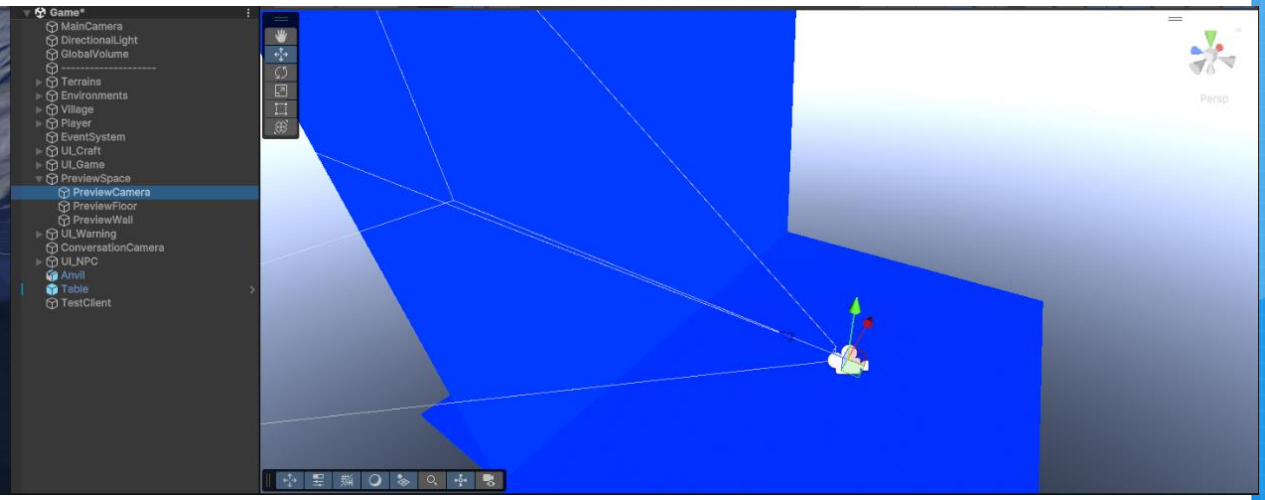
3. 구조 및 설계 카메라



ConversationCamera

별도의 카메라를 두어 필요 시
활성화/비활성화하여 화면 전환

CraftCamera



PreviewCamera

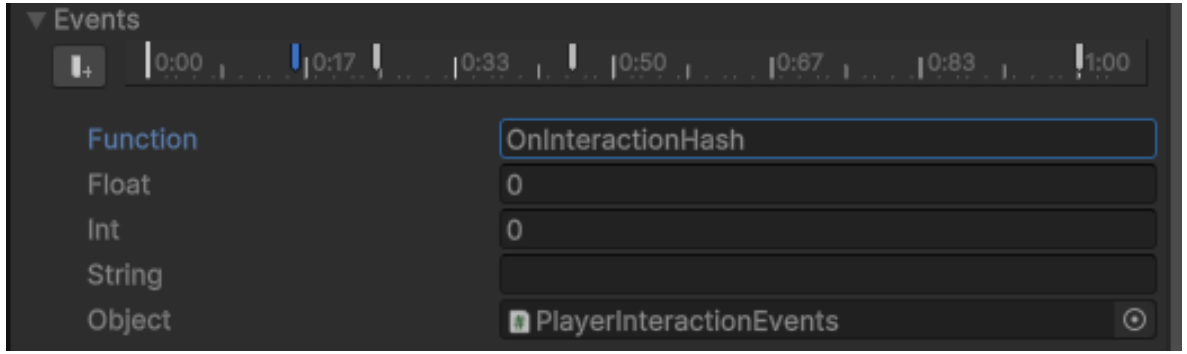
3. 구조 및 설계 캐릭터

```
참조 0개
public void OnInteractionHash()
{
    animator.SetBool(Define.InteractionHash, true);
}

참조 0개
public void OffInteractionHash()
{
    animator.SetBool(Define.InteractionHash, false);
}

참조 0개
public void OffIsAttacking()
{
    animator.SetBool(Define.IsAttacking, false);
}

참조 0개
public void OnWeaponSound()
{
    weaponPos.GetChild(0).GetComponent<AudioSource>().Play();
}
```



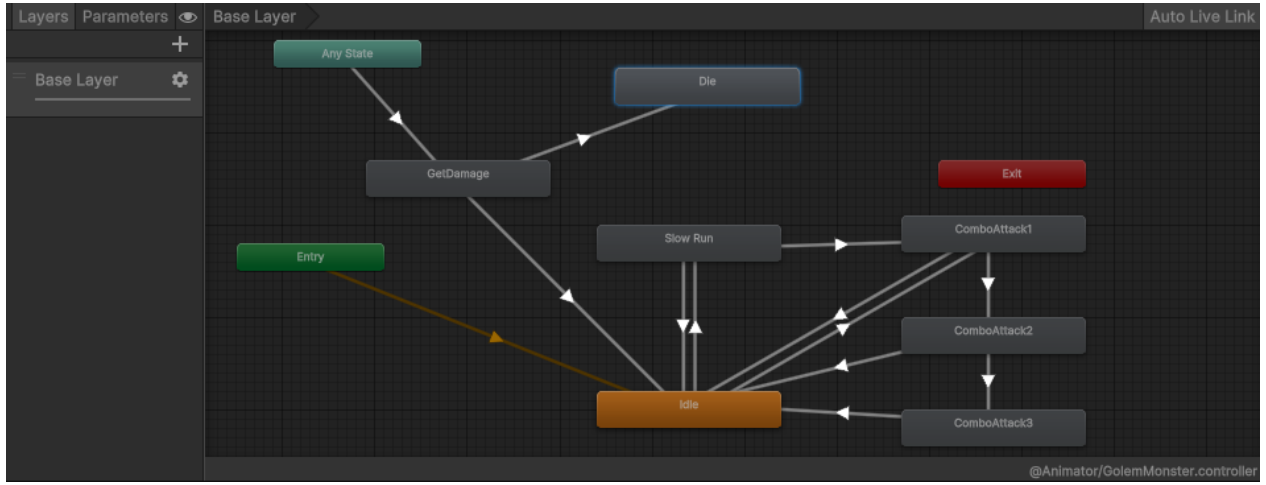
공격 모션 시에만 대미지 효과가 발동하도록 애니메이션 클립 이벤트 설정

3. 구조 및 설계

캐릭터 - 몬스터

BossAnimator - 3단 콤보 공격

플레이어 추격



3. 구조 및 설계

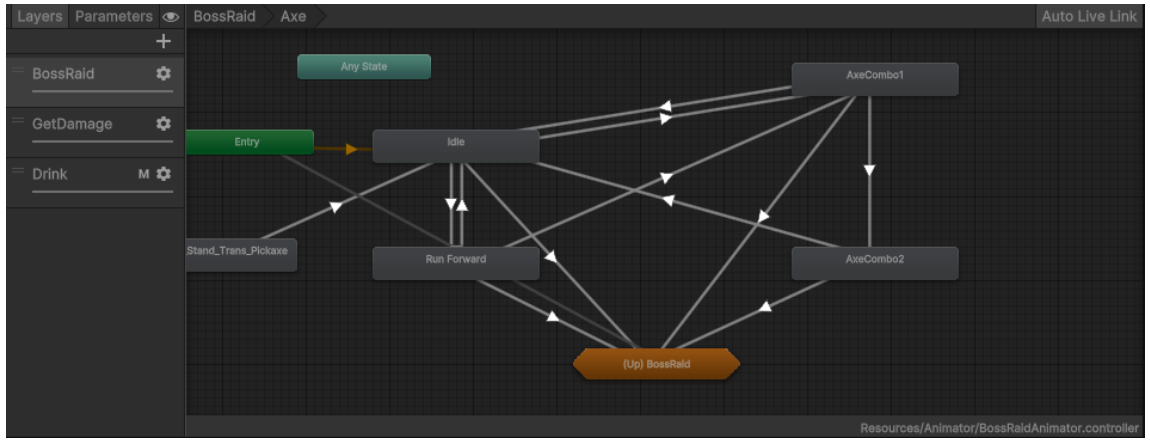
캐릭터 - 플레이어

Sword - 공격 시 trail 생성

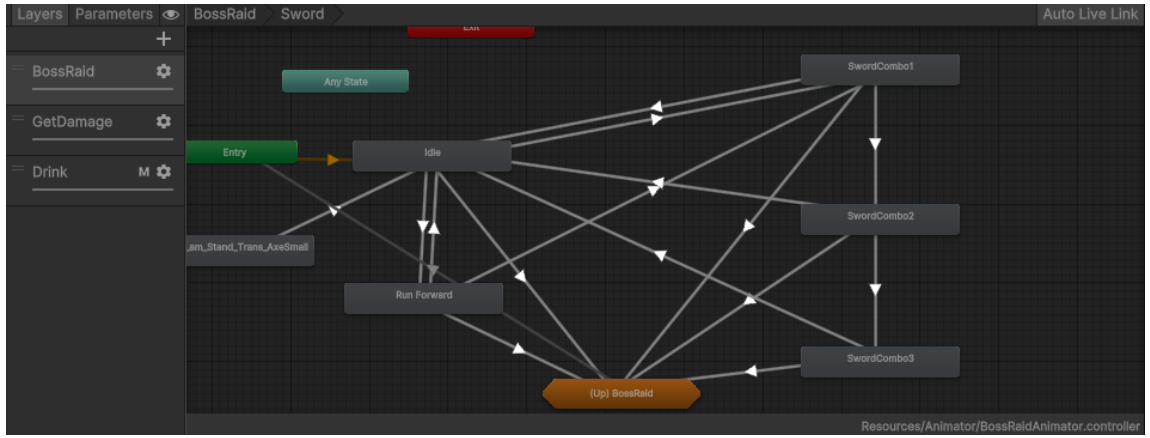
Stamina - 평시엔 Dash,
전투 시 Roll 행동에 사용

Roll(구르기) – 구르는 중엔 무적 판정

3. 구조 및 설계 캐릭터 - 플레이어



Axe – 2단 콤보



Sword - 3단 콤보

무기에 따라 다른 애니메이션과
콤보 공격 횟수

```
public Define.WeaponType UnequipWeapon()
{
    if (WeaponPos.transform.childCount == 0)
    {
        Debug.Log("무기를 장착하고 있지 않습니다");
        UI_Warning.OnWarningEffect?.Invoke(Define.NoWeapon);
        return Define.WeaponType.None;
    }
    GameObject weapon = WeaponPos.transform.GetChild(0).gameObject;
    if (weapon == null)
        return Define.WeaponType.None;
    Define.WeaponType weaponType = weapon.GetComponent<WeaponItem>().GetWeaponType();
    Destroy(weapon);
    // 맨손
    WeaponTypeHash = -1;
    GameManager.Instance.OnWeaponChanged?.Invoke();

    return weaponType;
}
```

무기 장착(퀵슬롯) 및 해제(x버튼)

```
public bool EquipItem(Slot currentSlot)
{
    // 아이템 사용해서 개수 줄어들면 true, 아니면 false
    if (currentSlot.ItemData == null)
        return false;
    // 무기 아이템일 경우
    if (currentSlot.ItemData.ItemType == Define.ItemType.Weapon)
    {
        if (WeaponPos.transform.childCount > 0)
        {
            // 장착해제
            UnequipWeapon();
        }
        GameObject newWeapon = Instantiate(GameManager.Instance.Weapons[(int)
          currentSlot.ItemData.WeaponType], WeaponPos.transform);
        WeaponTypeHash = (int)currentSlot.ItemData.WeaponType;
        GameManager.Instance.OnWeaponChanged?.Invoke();
        return false;
    }
```

무기 사용법

이미 무기 장착 중일 경우, 해제 후 장착

3. 구조 및 설계 캐릭터 - 플레이어

```
// 소비 및 재료 아이템일 경우
else if (currentSlot.ItemData.ItemType == Define.ItemType.Countable)
{
    if (currentSlot.ItemData.ItemName == "포션")
    {
        // 포션 사용 및 포션 먹는 애니메이션
        if (_hp >= Define.PlayerMaxHp)
        {
            UI_Warning.OnWarningEffect?.Invoke(Define.AlreadyFullHP);
            return false;
        }
        else
        {
            _animator.SetTrigger(Define.Drink);
            GameObject gameObject = Instantiate
              (GameManager.Instance.ConsumptionItems[0], ItemPos.transform);
            gameObject.GetComponent<Collider>().isTrigger = true;
            return true;
        }
    }
```

포션 애니메이션 실행

애니메이션 종료 시점에
포션 제거 이벤트 추가

포션 사용법

```
void OnPotionAnimationExit()
{
    GameManager.Instance.Player.PlayerHp = Mathf.Min(Define.PlayerMaxHp,
      GameManager.Instance.Player.PlayerHp + 20f);
    // 포션 아이템 제거, 인벤토리에서 포션 -1
    GameObject currentItem =
      GameManager.Instance.Player.ItemPos.transform.GetChild(0).gameObject;
    GameManager.Instance.Player.Inventory.AddItem
      (currentItem.GetComponent<Item>(), -1);
    Destroy(currentItem);
}
```

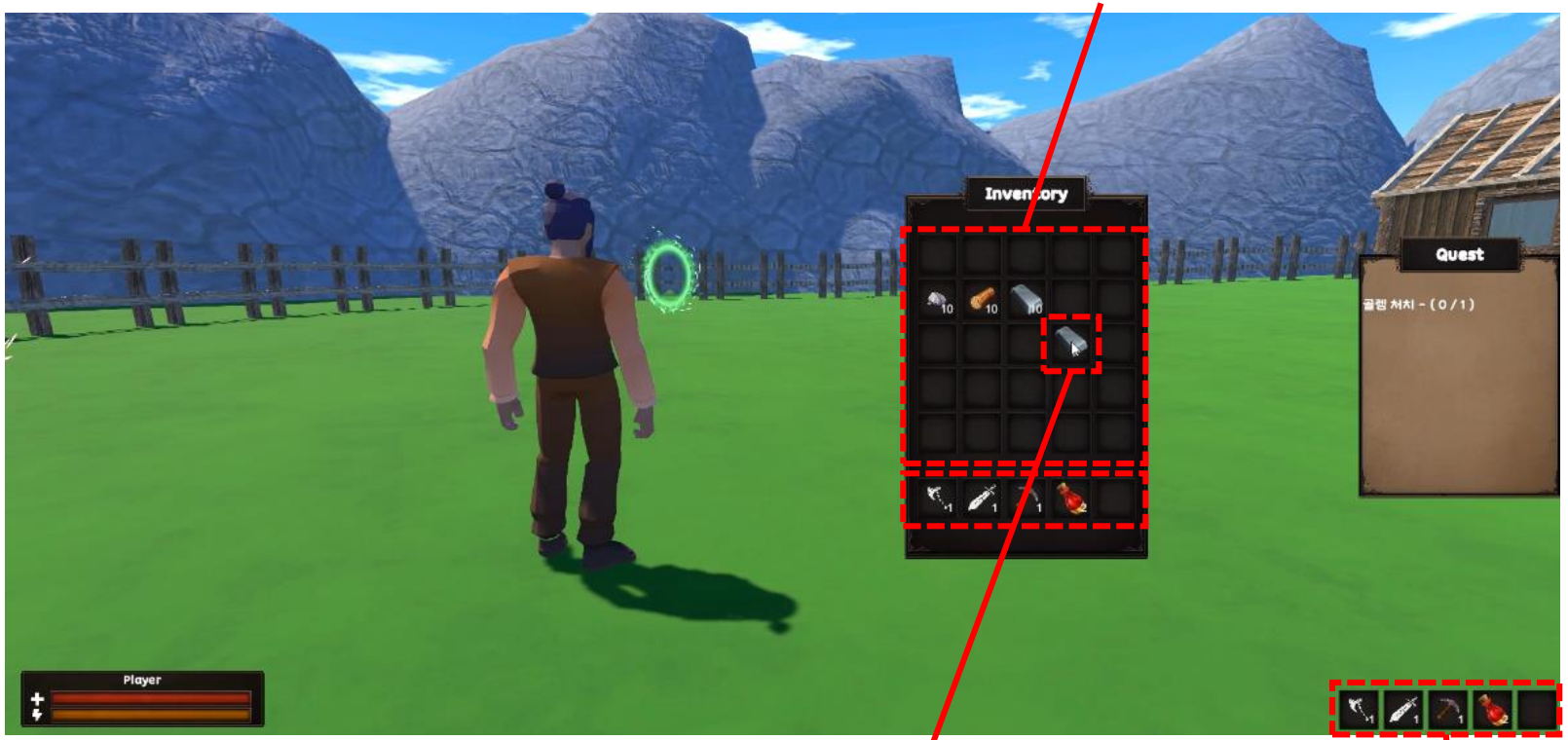
EverStride

```
// 장비 아이템일 경우
else if (currentSlot.ItemData.ItemType == Define.ItemType.Equipment)
{
    // GameManager에서 현재 플레이어의 장비 장착 상태 확인 후 업그레이드 가능하면 장착시켜줌
    switch (currentSlot.ItemData.Prefab.layer)
    {
        case (int)Define.EquipmentType.Helmet:
            // 기존 부위의 장비보다 높은 등급이면 장착
            if (GameManager.Instance.PlayerEquipment.helmet !=
             Define.EquipmentStatus.Iron)
            {
                if (GameManager.Instance.PlayerEquipment.helmet ==
                 Define.EquipmentStatus.Base)
                {
                    BaseClothes.transform.Find
                      ("Starter_Helmet").gameObject.SetActive(false);
                }
                PlateClothes.transform.Find(Define.PlateSetPrefix +
                  "Helmet").gameObject.SetActive(true);
                GameManager.Instance.PlayerEquipment.helmet =
                  Define.EquipmentStatus.Iron;
            }
            // 이미 같은 등급 장착 중이면 이미 장착 중이라고 표시
            else
            {
                UI_Warning.OnWarningEffect?.Invoke(Define.AlreadyEquipSameGrade);
                return false;
            }
            break;
```

장비 사용법

GameManager에서
플레이어 장비 상태 관리

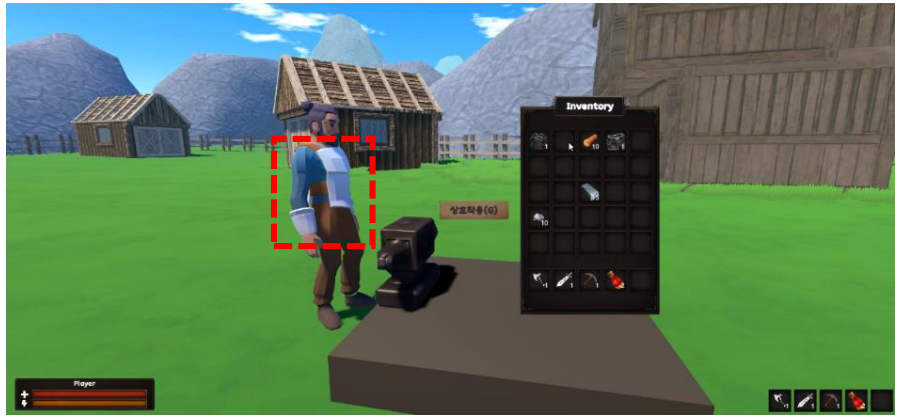
인벤토리 25칸 + 퀵슬롯 5칸



드래그 – 아이템 드롭
및 슬롯 위치 교체

퀵슬롯 인게임
UI

3. 구조 및 설계 인벤토리



우클릭

퀵슬롯 사용

아이템 사용 및 장착

3. 구조 및 설계 퀘스트

마을 내 NPC 근처 상호작용

수락 시 퀘스트 UI 생성

인벤토리 업데이트 및 목표 몬스터 처치 시 업데이트

3. 구조 및 설계 퀘스트

Coroutine을 활용한 NPC 대사 출력

```
IEnumerator CoWriteComment(string[] comment, bool yesFlag, bool noFlag)
{
    ConversationText.text = "";
    _currentContext = comment;
    for (int i = 0; i < comment.Length; i++)
    {
        foreach (char ch in comment[i].ToCharArray())
        {
            ConversationText.text += ch;
            yield return new WaitForSeconds(0.1f);
        }
        if (i != comment.Length - 1)
        {
            yield return new WaitUntil(() => (Input.GetKeyDown(KeyCode.Space) || Input.GetKeyDown(KeyCode.G)));
        }
    }

    YesButton.gameObject.SetActive(yesFlag);
    NoButton.gameObject.SetActive(noFlag);
    yield return new WaitUntil(() => (Input.GetKeyDown(KeyCode.Space) || Input.GetKeyDown(KeyCode.G)));
}
```

3. 구조 및 설계 퀘스트

```
public class Quest
{
    public Define.QuestName name;
    public string context;
    public int requireNum;
    public int currentNum;
    public bool isComlete;
    참조 1개
    public Quest(Define.QuestName name, string context, int requireNum)
    {
        this.name = name;
        this.context = context;
        this.requireNum = requireNum;
        this.currentNum = 0;
        this.isComlete = false;
    }

    참조 4개
    public void AddNum(int account)
    {
        this.currentNum += account;
        QuestPanel.OnQuestPanelUpdate?.Invoke();
        if (this.currentNum >= this.requireNum)
        {
            this.isComlete = true;
        }
    }
}
```

퀘스트 클래스

```
public void MakeQuest(Define.QuestName name, string context, int num)
{
    Quest quest = new Quest(name, context, num);
    if (!QuestDictionary.ContainsKey(name))
    {
        QuestDictionary.Add(name, quest);
        if (name == Define.QuestName.Wood)
        {
            // 혹시 인벤토리에 이미 나무가 있으면 개수 더해줌
            Slot slot = player.Inventory.FindItemInInventory(Ingredients[1].GetComponent<Item>());
            if (slot != null)
            {
                QuestDictionary[name].AddNum(slot.Amount);
            }
        }
        QuestPanel.OnQuestPanelUpdate?.Invoke();
    }
    else
    {
        UI_Warning.OnWarningEffect?.Invoke(Define.DuplicatedQuest);
    }
}
```

퀘스트 생성

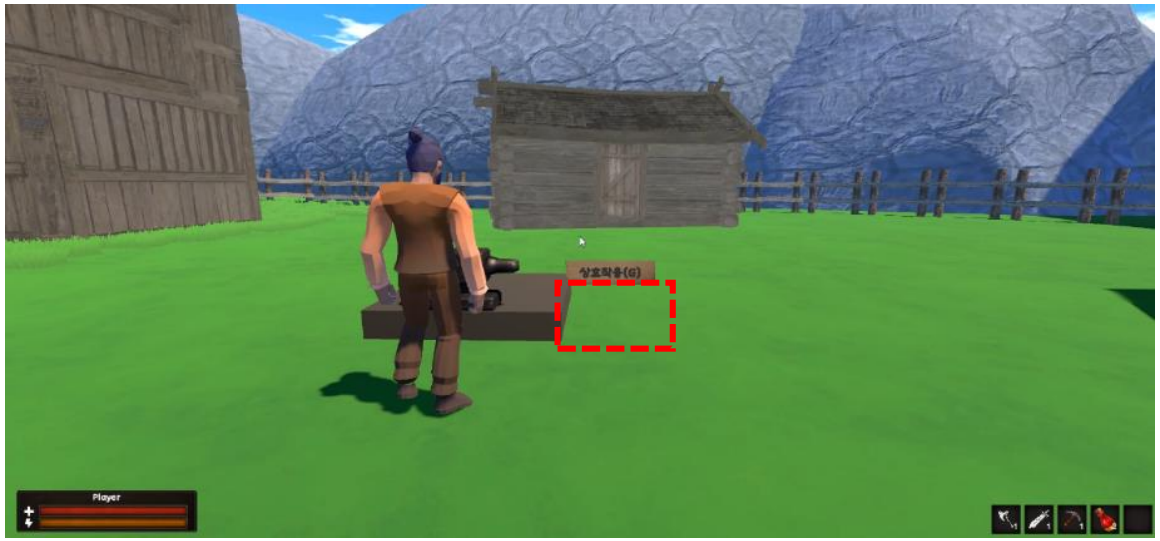
퀘스트 완료

```
public void CompleteQuest(Define.QuestName name)
{
    // 보상 지급 후
    if (name == Define.QuestName.Wood)
    {
        // 나무 10개 차감 후
        player.Inventory.FindItemInInventory(Ingredients[1].GetComponent<Item>()).AddAmount(-10);
        // 돌멩이 10개로 바꿔주자
        GameManager.Instance.Player.Inventory.AddItem(Ingredients[0].GetComponent<Item>(), 10);
    }
    else if (name == Define.QuestName.Golem)
    {
        GameManager.Instance.Player.Inventory.AddItem(Ingredients[2].GetComponent<Item>(), 3);
    }
    else if (name == Define.QuestName.Orc)
    {
        GameManager.Instance.Player.Inventory.AddItem(Ingredients[2].GetComponent<Item>(), 5);
    }
    // 퀘스트 삭제
    DeleteQuest(name);
}
```

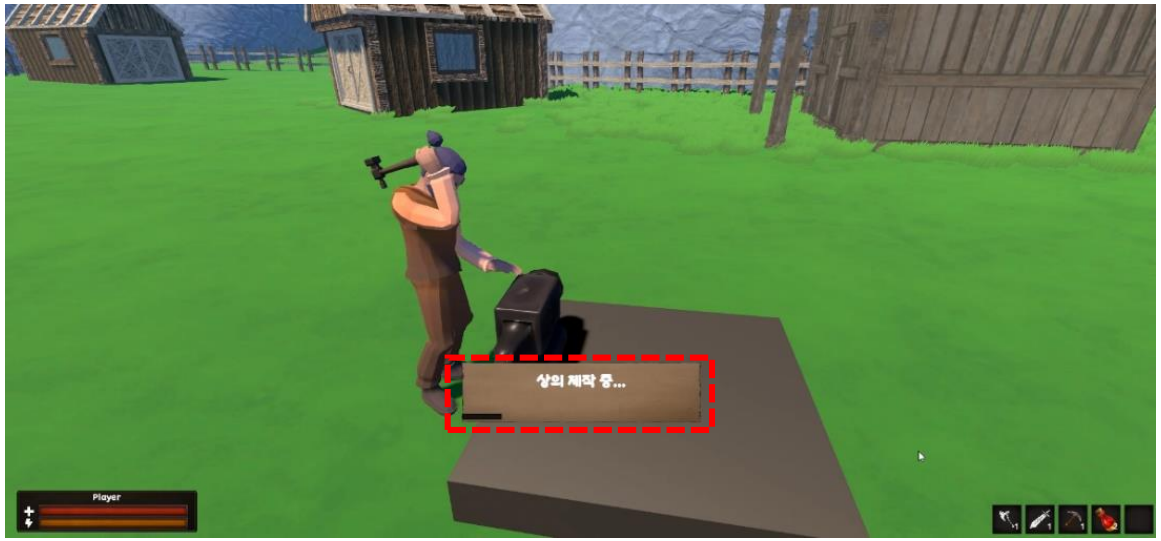
퀘스트 삭제

```
public void DeleteQuest(Define.QuestName name)
{
    int len = QuestDictionary.Count;
    foreach (KeyValuePair<Define.QuestName, Quest> quest in QuestDictionary)
    {
        if (quest.Value.name == name)
        {
            QuestDictionary.Remove(name);
            break;
        }
    }
    // 삭제되었으면 업데이트
    if (len != QuestDictionary.Count)
    {
        QuestPanel.OnQuestPanelUpdate?.Invoke();
    }
}
```

3. 구조 및 설계 제작 시스템



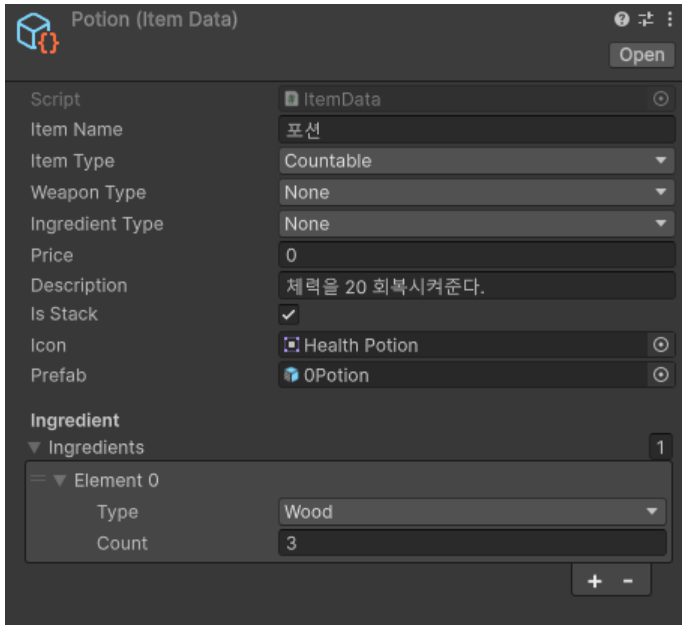
마을 내 제작대 근처 상호작용



제작 버튼 클릭 시,
정해진 시간 동안 제작 모션 실행

아이템 목록에서 버튼 클릭 시,
우측 Preview를 통해 확인
및 필요 재료 출력

3. 구조 및 설계 제작 시스템




```
// 장비 아이템일 경우, 프리뷰캐릭터를 생성하고 해당 부위의 레이어값만 렌더링 레이어에 넣어준다
if (currentItem.ItemData.ItemType == Define.ItemType.Equipment)
{
    previewObject = Instantiate(GameManager.Instance.PreviewCharacter, PreviewSpace.transform);
    CraftTable.PreviewCameraSetting(1 << LayerMask.NameToLayer("PreviewObject") | 1 << currentItem.ItemData.Prefab.layer);
    switch (currentItem.ItemData.Prefab.layer)
    {
        case (int)Define.EquipmentType.Helmet:
            previewObject.transform.localPosition = new Vector3(0f, -1.5f, 0f);
            break;
        case (int)Define.EquipmentType.Chest:
            previewObject.transform.localPosition = new Vector3(0f, -1f, 1f);
            break;
        case (int)Define.EquipmentType.Shoulders:
            previewObject.transform.localPosition = new Vector3(0f, -1.3f, 0.5f);
            break;
        case (int)Define.EquipmentType.Gloves:
            previewObject.transform.localPosition = new Vector3(0f, -1.3f, 1.5f);
            break;
        case (int)Define.EquipmentType.Pants:
            previewObject.transform.localPosition = new Vector3(0f, 0f, 1f);
            break;
        case (int)Define.EquipmentType.Boots:
            previewObject.transform.localPosition = new Vector3(0f, 0.05f, 0f);
            break;
        default:
            break;
    }
}
```

ItemData:
ScriptableObject로
아이템데이터 저장

장비 Preview:
캐릭터 오브젝트 생성 - 장비별 레이어 적용 -
PreviewCamera CullingMask 값 조절 -
장비별 localPosition 각각 설정

감사합니다

EverStride