팀 프로젝트

이름	이종찬
이메일	ok7148@naver.com
전화번호	010-9183-4226

포트폴리오 링크 주소

https://youtu.be/tWiCQQzsY2c

스폰매니저

```
EndTime += Time.deltaTime;
if (EndTime < TimeManager.Instance.DeadLine)
{

    if (BlockChk == false) // Npc 생성위치에 아무런 방해물이 없어서 정상일 때
    {
        if (curTime >= spawnTime && hostCount < maxCount)
        {
            AddvInpc(); // 가중치 값 세팅
            if (first)
            {
                for (int i = 0; i < vInpcs.Count; i++)
                {
                    total += vInpcs[i].weight;
                }
                first = false;
            }
            SpawnHost(); // 콜라이더에서 불값을 전달하고 이프문에서 불값을 검사함
        }
    }
}

curTime += Time.deltaTime;
if (EndTime >= TimeManager.Instance.OneDay)
{
    QuestManager.Instance.EndQuestClear();
    EndTime = 0.0f;
    curTime = 0.0f;
}
```

- 최대 수를 정해줘서 일정 수 이상이 되면 더 이상 스폰되지 않게 해주었고 마감시간을 넘어가도 스폰이 불가능하게 해주었습니다.

- 하루가 지나면 시간을 다시 0으로 초기화 해주었습니다.

사운드

```
private void Start()
{
    Scene scene = SceneManager.GetActiveScene();
    if (scene.name == "Tutorial1")
    {
        SoundList.volume = PlayerPrefs.GetFloat("volume");
    }
    else
    {
        BGMSound.volume = PlayerPrefs.GetFloat("volume");
    }
    foreach (AudioSource source in Eff)
    {
        source.volume = PlayerPrefs.GetFloat("eff");
    }
}
public void SetVolume()
{
    PlayerPrefs.SetFloat("volume", BGMvolume.value);
    BGM.volume = PlayerPrefs.GetFloat("volume");
    PlayerPrefs.SetFloat("eff", Effvolume.value);
    foreach (AudioSource source in Eff)
    {
        source.volume = PlayerPrefs.GetFloat("eff");
    }
}
```

- 슬라이더로 사운드를 조절했습니다.
- PlayerPrefs를 사용해서 볼륨크기를 저장하고 다른 씬에서 저장한 볼륨 크기를 가져옵니다.

가중치랜덤

```
if (GameManager.Instance.Fame >= 0)
{
    VLarray[0].NpcJob = VLNpc.NPCJOB.COMMONS;
    VLarray[1].NpcJob = VLNpc.NPCJOB.NOBILLITY;
    VLarray[2].NpcJob = VLNpc.NPCJOB.ROYALTY;

    VLarray[0].weight = 100;
    VLarray[1].weight = 0;

    for (int i = 0; i < (GameManager.Instance.Fame / 100); i++)
    {
        if (VLarray[0].weight != 0)
        {
            VLarray[0].weight -= 5;
            VLarray[1].weight += ((VLarray[1].weight >= 20) ? 4 : 5);
        }
        else
        {
            if (VLarray[1].weight != 0)
            {
                VLarray[1].weight -= 4;
            }
            else if (VLarray[1].weight == 0)
            {
                break;
            }
        }
    }
    VLarray[2].weight = 100 - (VLarray[0].weight + VLarray[1].weight);
}
vlnpcs.Clear();
for (int i = 0; i < 3; i++)
{
    vlnpcs.Add(VLarray[i]);
}
```

- 처음에는 일반 시민만 스폰이 되게하고 명성이 높아지고 일반 시민이 스폰이 될수록 귀족이 스폰될 확률이 올라가고 더 진행할수록 왕족의 스폰확률을 높였습니다.

AI 자동전투

```
switch (_adnpc.adtype)//직업별 애니메이션 실행
{
    case 0: // 여자 궁수
        _anim.SetTrigger("ArrowAttack");
        CreateArrow();
        break;
    case 1: // 여자 도적
        _anim.SetTrigger("ThiefAttack");
        break;
    case 2: // 여자 마법사
        _anim.SetTrigger("MagicAttack");
        break;
    case 3: // 남자 궁수
        _anim.SetTrigger("ArrowAttack");
        CreateArrow();
        break;
    case 4: // 남자 도적
        _anim.SetTrigger("ThiefAttack");
        break;
    case 5: // 남자 마법사
        _anim.SetTrigger("MagicAttack");
        break;
    case 6: // 남자 전사
        _anim.SetTrigger("WarriorAttack");
        break;
}
```

```
public void FindTarget(Transform target)
{
    myTarget = target;
    StopAllCoroutines();
    ChangeState(STATE.Battle);
}

참조 1개
public void LostTarget()
{
    myTarget = null;
    StopAllCoroutines();
    _anim.SetBool("IsMoving", false);
    ChangeState(STATE.Idle);
}
```

- 직업별로 공격이 다 다르게 했습니다.
- 적을 찾으면 유한상태기계 STATE를 Battle로 바꾸어 주었고 적이 죽어서 없어지면 Idle상태로 변하게 해주었습니다.

줄세우기

```
IEnumerator ForwardCheck()// 앞에 누가있는지 구분하여 상태를 Wait 또는 Order로 바꿔줌
{
    while (true)
    {
        if (Physics.SphereCast(transform.position, 7.0f, transform.forward, out RaycastHit hitinfo, 7.0f, layerMask)) // 내 앞에 누군가 있을 경우
        {
            agent.ResetPath();
            agent.velocity = Vector3.zero; //가속도 = 0
            _anim.SetBool("IsMoving", false); // 걷는 애니 중지

            if (hitinfo.collider.gameObject.layer == 6) // layer 6 = Host, layer 3 = Staff // 앞에 Host가 있을 경우
            {
                if (hitinfo.collider.gameObject.GetComponent<Host>().LineChk == true)
                {
                    LineChk = true;
                }
                ChangeState(STATE.Wait);
            }
            else // 앞에 Staff가 있을 경우
            {
                myStaff = hitinfo.collider.gameObject; // myStaff를 각 구역에 맞게 설정해줌 ( 스마일 아이콘 발동 시키기 위해 )
                ChangeState(STATE.Order); // Order로 상태변화
            }
        }
        else // 앞에 있는 누군가가 비켰을 경우
        {
            ChangeState(STATE.Moving);
        }
        yield return null;
    }
}
```

Teleport 이동

```
void Teleport(GameObject host, int num, bool chk)
{
    if (chk)
    {
        host.GetComponent<ADNpc>().AI_Per.SetActive(true);
        if (host.GetComponent<ADNpc>().myStat.npcJob == CharState.NPCJOB.ACHER)
        {
            host.GetComponent<Host>().myBow.SetActive(true);
        }
    }
    //빈자리인지 확인
    host.GetComponent<NavMeshAgent>().Warp((questchk ? QuestPos1[num] : QuestPos2[num]).transform.position);
    if (host.GetComponent<QuestInformation>().myQuest.questname == "채집")
    {
        host.GetComponent<Host>().StateFarming();
    }
}
public void Teleport(GameObject host)
{
    host.GetComponent<NavMeshAgent>().Warp(spawnPoints[0].transform.position);
    host.transform.parent = spawnPoints[0].transform;
}
```

- 네비게이션을 사용해 일반 텔레포트가 아닌 Warp함수를 사용했습니다.

스텟에 따른 등급

```
public CharState.GRADE Grade(int main)
{
    CharState.GRADE grade = new CharState.GRADE();
    if (main <= 140) // 기본스텟 40
    {
        grade = CharState.GRADE.F;
    }
    else if (main <= 640 && main > 140)
    {
        grade = CharState.GRADE.E;
    }
    else if (main <= 1640 && main > 640)
    {
        grade = CharState.GRADE.D;
    }
    else if (main <= 4640 && main > 1640)
    {
        grade = CharState.GRADE.C;
    }
    else if (main <= 9640 && main > 4640)
    {
        grade = CharState.GRADE.B;
    }
    else if (main > 9640)
    {
        grade = CharState.GRADE.A;
    }
    return grade;
}
```

▶ 자신의 스텟에 따라서 등급을 정해주었습니다.

패널티

```
// 실패시 감소 연산
  GameManager.Instance.ChangeGold(-myQuest.rewardgold);
  if (SpawnManager.Instance.EndTime >= TimeManager.Instance.DeadLine)
  {
      GameManager.Instance.ChangeFame(-myQuest.rewardfame);
  }
  _childHost.onAngry = true;
```

▶ 퀘스트에 실패하면 돈과 명예가 줄게 했습니다.

감사합니다!