제작한 포트폴리오의 일부 코드 모음입니다.

[마지막 페이지에 포트폴리오 영상 URL이 있습니다.]

1.1 텔레포트 스킬

```
223     void TelePortCheck()
224     {
225         if (Input.GetMouseButtonDown(0))
226         {
227             Vector3 buildingPos;
228             if (isTeleportOnOff)
229             {
230                 Ray ray = cam.ScreenPointToRay(Input.mousePosition);
231                 float enter;
232                 Vector3 meetPoint;
233                 float dist = 30f;
234                 plane = new Plane(-cam.transform.forward, cam.transform.position + cam.transform.forward * dist);
235                 RaycastHit hit;
236 
237                 if (Physics.Raycast(ray, out hit, dist, buildingLayer))
238                         buildingPos = hit.point;
239                 else
240                         buildingPos = Vector3.zero;
241 
242                 if (plane.Raycast(ray, out enter))
243                 {
244                     meetPoint = ray.GetPoint(enter);
245                     if (buildingPos != Vector3.zero)
246                     {
247                     Vector3 pos = buildingPos - transform.position;
248                     float moveDist = Vector3.Distance(buildingPos, transform.position);
249                     pos.Normalize();
250                     _animationActionPlay.Teleport += () => rigidbody.MovePosition(transform.position + (pos * (moveDist - 1f)));
251                     curCastingSkill.GetComponent<SKillTeleport>().SkillAnimSetp1();
252                     curCastingSkill.GetComponent<SKillTeleport>().SkillAnimSetp2();
253                     }
254                     else
255                     {
256                         Vector3 pos = meetPoint - transform.position;
257                         pos.Normalize();
258                         _animationActionPlay.Teleport += () => rigidbody.MovePosition(transform.position + (pos * dist));
259                         curCastingSkill.GetComponent<SKillTeleport>().SkillAnimSetp1();
260                         curCastingSkill.GetComponent<SKillTeleport>().SkillAnimSetp2();
261                     }
262                 }
263             }
264         }
265     }
```

플레이어가 바라보는 방향으로 일정 거리 앞에 보이지 않는 Plane을 생성시키고 해당 지점으로 플레이어 캐릭터를 이동시키는 방식입니다.

만약 Plane과 캐릭터 사이에 건물이 있는 경우 캐릭터가 건물을 통과하지 못하도록 텔레포트 사용 시 RayCast로 건물과 플레이어의 거리를 확인 하여 건물 바로 앞에서 텔레포트 이동이 멈추도록 했습니다.

2.1 몬스터 타겟팅

```
void Update()
{
    if (Input.GetMouseButtonDown(0))
    {
        Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition);
        RaycastHit hit;
        Transform headTr;
        Transform leftHandTr;
        Transform rightHandTr;
        Transform leftFootTr;
        Transform rightFootTr;

        //Ray가 몬스터 레이어를 통과하면
        if (Physics.Raycast(ray, out hit, 100.0f, Monster))
        {
            //직전에 선택된 몬스터는 타겟팅 테두리 효과 적용 해제
            if(prevSelectTarget != null && prevSelectTarget != hit.transform)
            {
                prevSelectTarget.GetComponentInChildren<GetMaterial>().GetmyMaterial[1].SetFloat("Boolean_AB71AB7D", 0.0f);
                Destroy(prevSelectTarget.GetComponentInChildren<MonsterState>().hpbarObj);
                prevSelectTarget = hit.transform;
            }
            else
            {
                prevSelectTarget = hit.transform;
            }

            GetIsSelect = true;
            GetselectTarget = hit.transform;


            headTr = hit.transform.GetComponentInChildren<GetHeadPosition>().tr;
            leftHandTr = hit.transform.GetComponentInChildren<GetLeftHandPosition>().tr;
            rightHandTr = hit.transform.GetComponentInChildren<GetRightHandPosition>().tr;
            leftFootTr = hit.transform.GetComponentInChildren<GetLeftFootPosition>().tr;
            rightFootTr = hit.transform.GetComponentInChildren<GetRightFootPosition>().tr;

            //몬스터 타겟팅 이미지 출력 코루틴
            if (isTargetingFollow != null) StopCoroutine(isTargetingFollow);
            isTargetingFollow = StartCoroutine(TargetingFollow(hit.transform, targetImage, headTr, leftHandTr, rightHandTr, leftFootTr, rightFootTr));
        }
    }
}
```

예시 이미지

몬스터 타켓팅 인지방식은 몬스터 모델링 외곽 부분을 Material로 붉게 변화를 주는 것과 따로 제작한 이미지로 마치 몬스터가 락온(Lock on)된 것처럼 보이는 것. 총 2가지로 적용했습니다.

Ray가 몬스터를 통과 후 첫번째 조건문은 현재 선택된 몬스터가 직전에 선택된 몬스터와 동일 몬스터인지를 판단하여 동일하다면 타겟 표시가 유지되도록 아니라면 직전에 선택된 몬스터는 타겟 표시가 해제되도록 했습니다.

그 다음 락온에 사용할 이미지들을 몬스터에 위치시키기 위해 현재 선택된 몬스터의 머리, 양 손,양 발 끝의 위치를 가져오고 이후 락온 이미지들이 몬스터의 움직임에 맞게 실시간으로 위치가 변할 수 있도록 코루틴을 이용했습니다.

2.2 몬스터 타겟팅

```
Vector3 setPos = Vector3.zero;
target.GetComponentInChildren<GetMaterial>().GetmyMaterial[1].SetFloat("Boolean_AB71AB7D", 1.0f);
target.GetComponentInChildren<MonsterState>().InstantiateHPbar();
hpbar = target.GetComponentInChildren<MonsterState>().myhpbar;
targetUIGroup.SetActive(true);
MonsterData targetData = target.GetComponent<MonsterData>();
hpbar.maxValue = targetData.MaxHp;
hpbar.value = targetData.CurHp;
nameText.text = targetData.MonsterName;
nameText.outlineColor = new Color(255,0,0);

float targetingMaxDistance =  70.0f;
float targetDistance = 0.0f;

List<float> screenPosX = new List<float>();
List<float> screenPosY = new List<float>();

Vector3 headPos;
Vector3 leftHandPos;
Vector3 rightHandPos;
Vector3 leftFootPos;
Vector3 rightFootPos;

Vector3 headScreenPos;
Vector3 leftHandScreenPos;
Vector3 rightHandScreenPos;
Vector3 leftFootScreenPos;
Vector3 rightFootScreenPos;

float getMinX;
float getMinY;
float getMaxX;
float getMaxY;


float outRange = 15.0f;

//몬스터의 모델링 움직임에 따라 락온 이미지의 위치도 이에 맞춰서 함께 조절
while (target != null && GetIsSelect)
{
    hpbar.value = targetData.CurHp;
    float dist = (transform.position - target.position).magnitude;
    if (dist >= outRange)
    {
        //몬스터와 플레이어의 거리가 멀어지면 타겟 해제
        GetIsSelect = false;
        target.GetComponentInChildren<GetMaterial>().GetmyMaterial[1].SetFloat("Boolean_AB71AB7D", 0.0f);
        Destroy(hpbar.gameObject);
        targetUIGroup.SetActive(false);
        target = null;
    }
    else
```

```
headScreenPos = Camera.main.WorldToScreenPoint(headPos);
leftHandScreenPos = Camera.main.WorldToScreenPoint(leftHandPos);
rightHandScreenPos = Camera.main.WorldToScreenPoint(rightHandPos);
leftFootScreenPos = Camera.main.WorldToScreenPoint(leftFootPos);
rightFootScreenPos = Camera.main.WorldToScreenPoint(rightFootPos);

headScreenPos.x -= halfsize.x;
headScreenPos.y -= halfsize.y;
leftHandScreenPos.x -= halfsize.x;
leftHandScreenPos.y -= halfsize.y;
rightHandScreenPos.x -= halfsize.x;
rightHandScreenPos.y -= halfsize.y;
leftFootScreenPos.x -= halfsize.x;
leftFootScreenPos.y -= halfsize.y;
rightFootScreenPos.x -= halfsize.x;
rightFootScreenPos.y -= halfsize.y;


screenPosX.Add(headScreenPos.x);
screenPosX.Add(leftHandScreenPos.x);
screenPosX.Add(rightHandScreenPos.x);
screenPosX.Add(leftFootScreenPos.x);
screenPosX.Add(rightFootScreenPos.x);

screenPosY.Add(headScreenPos.y);
screenPosY.Add(leftHandScreenPos.y);
screenPosY.Add(rightHandScreenPos.y);
screenPosY.Add(leftFootScreenPos.y);
screenPosY.Add(rightFootScreenPos.y);

screenPosX.Sort();
screenPosY.Sort();

getMinX = screenPosX[0];
getMinY = screenPosY[0];
getMaxX = screenPosX[4];
getMaxY = screenPosY[4];

targetDistance = Mathf.Abs(screenPosX[4] - screenPosX[0]);

//각 방향의 최대,최소값을 확인해서 타겟 이미지의 포지션에 적용
if (targetDistance < targetingMaxDistance)
{
    images[0].transform.localPosition = new Vector3(getMinX - 20.0f, getMaxY, 0);
    images[1].transform.localPosition = new Vector3(getMaxX + 20.0f, getMaxY, 0);
    images[2].transform.localPosition = new Vector3(getMaxX + 20.0f, getMinY, 0);
    images[3].transform.localPosition = new Vector3(getMinX - 20.0f, getMinY, 0);


}
```

코루틴 내부 반복문입니다.

플레이어와 선택된 몬스터 사이의 거리가 일정 간격 이내라면 락온이 유지되도록 했습니다.

선택된 몬스터의 머리, 양 발,손 끝 위치를 월드 좌표에서 스크린 좌표로 변환해주고 락온을 위해 만든 4개의 이미지를 해당 좌표로 대입해 줬습니다.

마지막으로 락온 이미지가 몬스터의 각 사이드에서 너무 멀어지거나 좁혀지지 않도록 최소, 최대값을 지정해줬습니다.

3.1 방위 표시

```
public class CardinalPoints : MonoBehaviour
{
    public Transform playerTr;
    public Image cardinalPoints;
    public Image selectPoint;
    Material myMaterial;
    float algle;
    float setoffSet;
    float targetsetoffSet;
    public bool isSelect = false;

    void Start()
    {
        myMaterial = cardinalPoints.materialForRendering;
    }

    // Update is called once per frame
    void Update()
    {
        algle = playerTr.eulerAngles.y;
        setoffSet = (algle / 360.0f)+0.94f;
        if(setoffSet > 1)
        {
            setoffSet -= 1.0f;
        }
        myMaterial.SetTextureOffset("_MainTex", new Vector2(setoffSet,0));
    }

    public void SelectPoint(Transform selectTarget)
    {
        StartCoroutine(PointCheck(selectTarget));
    }

    IEnumerator PointCheck(Transform selectTarget)
    {
        while(isSelect)
        {
            Vector3 dir = (selectTarget.position - playerTr.position).normalized;
            float rot = Vector3.Dot(dir, playerTr.forward);
            rot = Mathf.Acos(rot);
            rot = (rot * 180.0f) / Mathf.PI;

            if (Vector3.Dot(playerTr.right, dir) < 0.0f)
            {
                rot *= -1.0f;
            }

            targetsetoffSet = (400f * rot) / 180f + 0.94f;

            selectPoint.rectTransform.localPosition = new Vector2(targetsetoffSet, 0);
            yield return null;
        }
    }
```

예시 이미지

화면 상단에 현재 플레이어가 바라보고 있는 방향과 미니맵에서 선택한 NPC의 위치를 확인 할 수 있도록 방위를 적용했습니다.

Material의 Offset을 이용하여 플레이어의 Y축 회전에 따라 값이 변하도록 했습니다.

또한 내적을 통해 플레이어를 기준으로 선택한 NPC의 방향을 파악하여 방위 이미지 위에 함께 표시되도록 했습니다.

4.1 회복 아이템

```
public abstract class Item : ScriptableObject
{
    public int ItemID = 0;
    public string ItemName = "";
    public Sprite ItemIcon = null;

    public abstract void Remove();
    public abstract void Use();
}
```

```
[CreateAssetMenu(fileName = "PotionItem", menuName = "Item / Potion Data", order  = int.MaxValue)]
public class PotionItem : Item
{

    public float Health = 0;

    public override void Use()
    {
        float maxHp = PlayerData.GetInstance().maxHp;

        if (PlayerData.GetInstance().curHp + Health >= maxHp)
        {
            PlayerData.GetInstance().curHp = maxHp;
            PlayerData.GetInstance().hpBar.value = maxHp;
            PlayerData.GetInstance().mainHpbar.value = maxHp;
        }
        else
        {
            PlayerData.GetInstance().curHp += Health;
            PlayerData.GetInstance().hpBar.value += Health;
            PlayerData.GetInstance().mainHpbar.value += Health;
        }

        Remove();
    }

    public override void Remove()
    {
        SelectSlot.GetInstance().curSelectSlot.RemoveItem();
    }
}
```

베리에이션을 고려하여 상속을 통해 각각 특징별로 오버라이드 하여 활용할 수 있도록 적용했습니다.

회복아이템 사용시 인스턴스를 통해 해당 아이템을 사용하는 플레이어의 데이터에 바로 접근하여 사용되도록 했습니다.

감사합니다.

https://www.youtube.com/watch?v=TEgRdujNk8w

포트폴리오 영상 URL