캡스톤디자인(종합설계) 결과보고서

소속학부(과)	디지털콘텐츠공학과	팀명	BHK2
개설 연도 및 학기	2022 학년도 ☑1학기 □2학기	교과목명	캡스톤디자인
과제명	공포게임_미련		
과제유형	☑기업연계형 캡스톤디자인	□기술이전형 캡스톤디자인	
희망금액		(기술이전금액)천원	

참여기업현황					
기업	기업명		소재지	
	사업자번호		주요생산품목	
담당자	성명		소속부서	
	H.P		E-mail	

참여 학생 현황

구분	이름	학부(과)	학년	학번	H.P	E-mail
팀장		디지털콘텐츠공학과	4			
팀원1		디지털콘텐츠공학과	4			
팀원2		디지털콘텐츠공학과	4			
팀원3		디지털콘텐츠공학과	4			
팀원4						
팀원5						
팀원6						
팀원7						

집행경비내역	비목	집행내역	금액
	재료비	*게임 제작에 필요한 에셋 구매*	250천원
	인쇄비		0천원
	학생여비	자세히 작성	
	학생회의비	( )천원 × ( )인 × ( )회	0천원
			천원
	총액		250천원

위와 같이 캡스톤디자인(종합설계) 결과보고서를 제출합니다.

첨부 : 캡스톤디자인(종합설계) 과제 상세 결과보고서[별첨 1호]

2022년 06월 17일

지원학생(팀장) (인)

사업책임자(지도교수) (인)

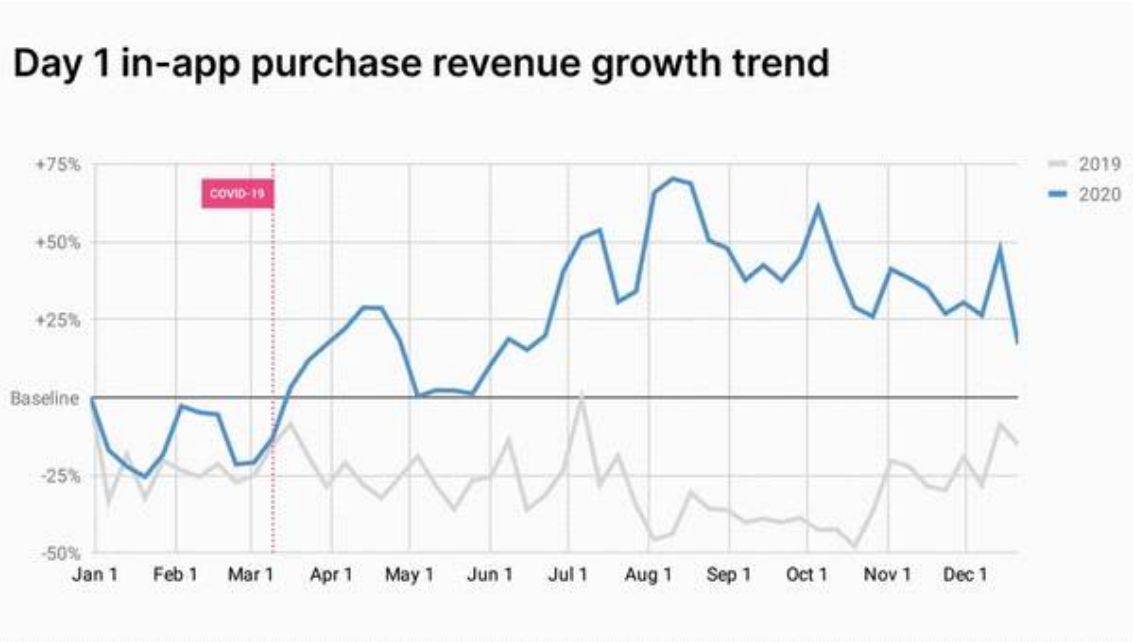
참여기업 담당자 (인)

원광대학교 LINC 3.0 사업단장 귀하

3. 개발의 필요성

a. 코로나19 팬데믹으로 인한 게임이용자 증가

코로나 19로 장기화로 인해 집안에 주거하는 시간이 증가하면서 게임 시장은 호황을 이끌고 있다.



출처: http://www.itdaily.kr/news/articleView.html?idxno=202085

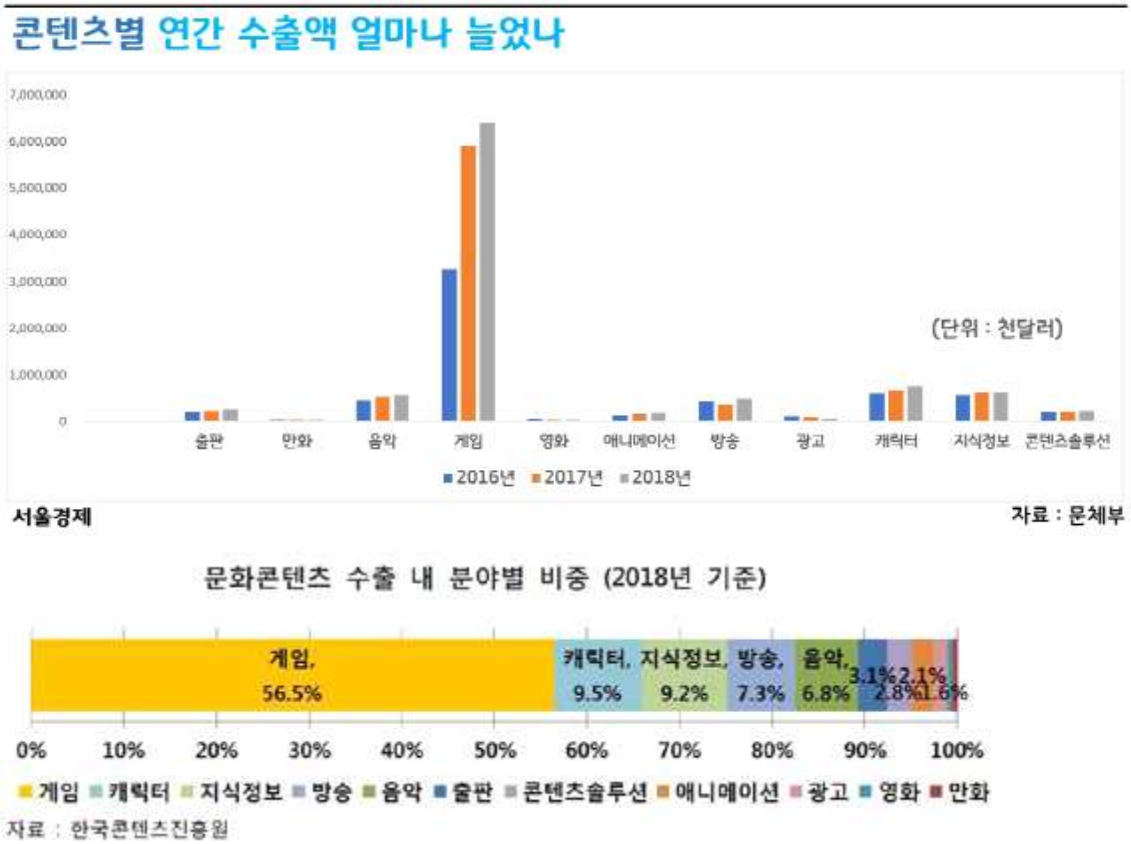
유니티의 펜데믹 보고서에 따르면 코로나로 인해 게임 이용자들의 플레이 시간이 증가했다는 조사 결과가 나왔다.

b. 콘텐츠 매출 1위인 게임

출처: https://www.sedaily.com/NewsView/1Z2T84CQK3

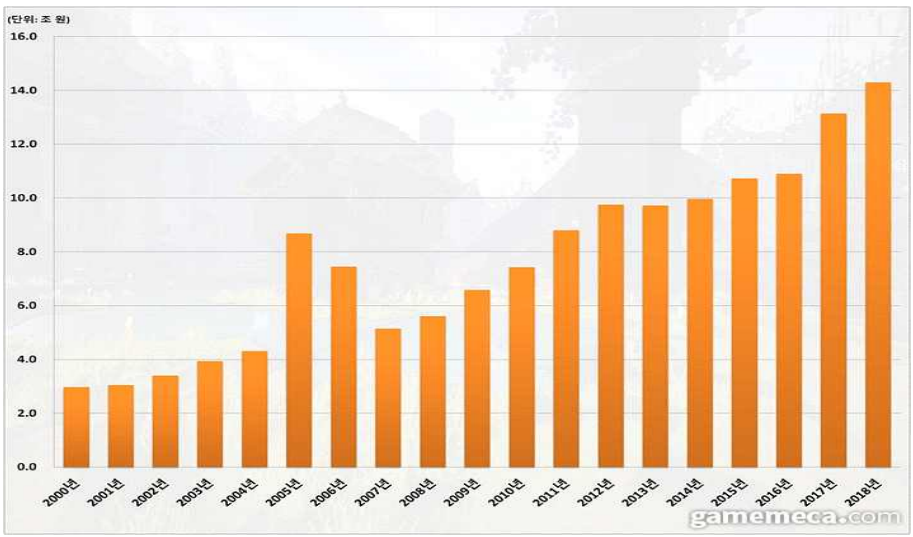
문체부, 2019년 콘텐츠산업 통계조사에 의하면 한국의 콘텐츠별 수출 중 가장 많이 한 콘텐츠는 '게임'이라고 조사되었다.

게임은 문화콘텐츠 수출 내 분야별 비중에 56.5%를 차지한 것으로 나타났다.



출처:https://m.segye.com/view/20190619502814

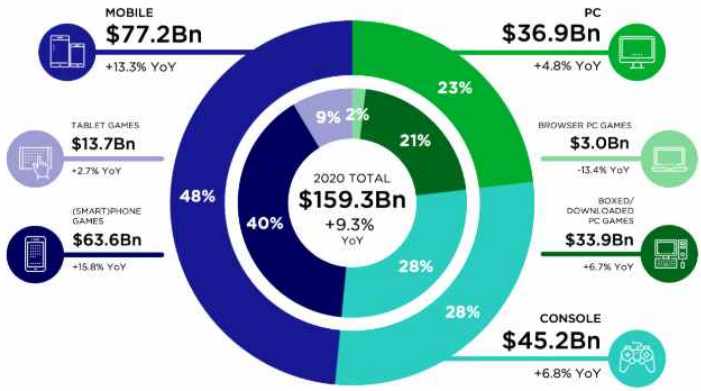
c. 게임 산업의 성장



출처: https://www.gamemeca.com/view.php?gid=1613418

해당 그래프를 보면 2000년, 국내 게임시장 전체 매출 규모는 약 3조원 수준이었다. 그러나 2018년에는 5배 가까이 성장한 14조원을 기록했으며 안정적인 성장세를 유지하고 있다.

d. 전 세계 게임시장 현황



출처:https://newzoo.com/insights/articles/newzoo-games-market-numbers-revenues-and-audience-2020-2023/

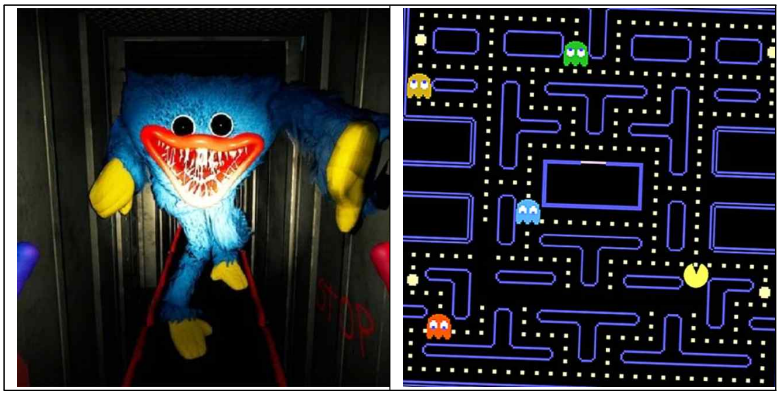
Newzoo가 조사한 글로벌 게임 시장 보고서에 의하면 코로나로 인해 2020년 글로벌 게임 시장이 1600억 달러를 돌파하여 2023년에는 약 2000억 달러를 돌파할 것이라고 보고 있다.
PC 및 콘솔 게임보다 모바일이 더 성장세가 높으며, 콘솔과 PC 시장도 안정적인 성장세를 유지하고 있다.

이를 근거로 게임의 시장성이 높고 수익성 또한 증가하고 있다. 따라서 개발의 필요성을 느끼게 되었다.

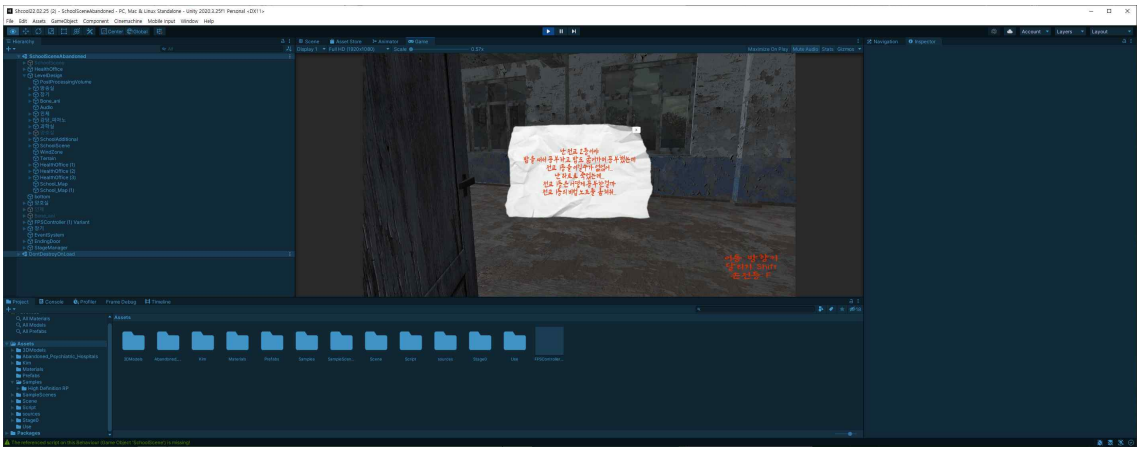
4, 개발 목표

유니티를 활용한 공포 탈출 게임

방안에 갇힌 플레이어는 스토리의 진행에 따라 제공되는 힌트를 이용하여 방안에 있는 퍼즐들을 풀어



가며 학교 탈출을 최종 개발 목표로 한다.



이 게임을 통해 플레이어는 퍼즐을 풀어가는 재미와 문제를 추리하는 재미를 느낄 수 있도록 한다. 또한 플레이어에게 게임 스토리를 직접적으로 묘사하는 것이 아닌 주변 환경요소와 퍼즐에 스토리 요소를 녹여 내어 플레이어 스스로 스토리를 추리할 수 있다.

5. 관련 개발 분석

A. Poppy Playtime

MOB Games에서 개발한 게임이며 1인칭 생존 호러 게임으로, '그랩 팩'이라는 독특한 퍼즐 기믹과 챕터 1의 적 '몬스터'의 추격 신이 굉장히 긴박하고 임팩트 있게 연출되어서 인기가 빠르게 오르고 있다

<링크>

B. Escape Simulator

방탈출 카페를 게임으로 만든 작품이며 단독 또는 온라인 협동 플레이가 가능한 1인칭 퍼즐 게임이다. 이 게임의 특징은 책이나 유물 등을 조사하면서 단서를 찾고 그 단서를 이용해서 퍼즐을 푸는 게임이다. 또 다른 특징은 직접 방탈출과 퍼즐을 디자인할 수 있는 편집 기능이 포함되어 있다.

C. 반교

대만의 게임 제작사 레드 캔들 게임즈가 제작한 호러 게임이며, 사이드뷰 형식의 2D 호러 어드벤처이다. 1960년 그 시절 대만의 시대상을 높은 전달력과 스토리로 공산주의의 어두운 면을 공포게임으로

잘 전달했다.

6. 연구 수행 내용

플레이어가 아이템과 접촉하면 사라지는 작업을 실시

```
5   public class GetCoin : MonoBehaviour
6   {
        Unity 메시지 | 참조 0개
7       private void OnTriggerEnter(Collider other)
8       {
9           //Debug.Log("누군가 들어왔다." + other.tag);
10          if(other.tag=="Player")
11          {
12              //CoinCount_TxT.text = (Coincount - 1).ToString();
13              CoinCheck.Instance.Destroy_Coin();
14              Destroy(gameObject);
15          }
16      }
17  }
18  /*
19  만일 코인이 통과하면 코인 사라지고 남은 코인 개수가 -1 됨
20   */
21
```

코인안에 만약 플레이어 태그가 들어왔을 경우 CoinCheck Destroy_Coin() 함수를 실행시킨 뒤 게임 오브젝트인 코인을 파괴시킴

```
41
42      //싱글톤으로 코인 스크립트에서 코인 먹을때 쓰임
        참조 1개
43      public void Destroy_Coin()
44      {
45          CoinSound.Play();
46          CoinCount -= 1;
47          Coin_Count_TxT.text = CoinCount.ToString();
48      }
49  }
50
```

CoinCheck Destroy_Coin() 함수는 코인 먹는 사운드를 실행시킴, 남은 코인 개수를 하나 줄인 뒤 오른쪽 위에 있는 UI에 숫자를 넘겨줌

```
6   public class CoinCheck : MonoBehaviour
7   {
8       public static CoinCheck Instance;   //싱글톤
9       public int CoinCount = 0;   //전체 코인 개수
10
11      [SerializeField]
12      private AudioSource CoinSound;
13
14      [SerializeField]
15      Text Coin_Count_TxT;
16
        @Unity 메시지|참조 0개
17      private void Awake()
18      {
19          Instance = this;
20      }
        @Unity 메시지|참조 0개
21      void Start()
22      {
23          Invoke("Find_CoinCount", 0.1f);
24      }
25
26
        참조 0개
27      void Find_CoinCount()
28      {
29          for (int i = 0; i < transform.childCount; i++)
30          {
31              if (transform.GetChild(i).gameObject.name == "CoinPrefab(Clone)")
32              {
33                  CoinCount++;
34              }
35          }
36
37          Coin_Count_TxT.text = CoinCount.ToString();
38
39          //Debug.Log(CoinCount);
40      }
```

맵 안에서 존재하는 코인을 찾아 전체 코인 개수를 파악한 뒤 오른쪽 위에 있는 UI에 코인 개수를 출력시킴

게임 제작 시 플레이어가 목표를 달성하면 클리어, 클리어 타이머 기능의 필요성을 느끼게 되어서 제작하게 되었음

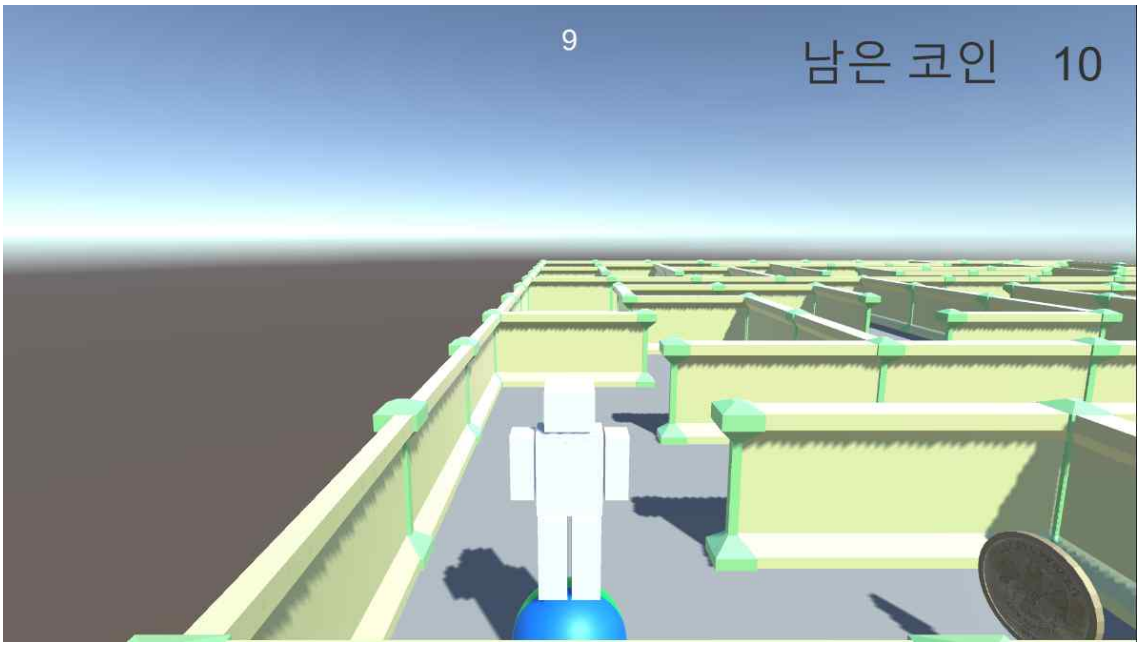
클리어 기능 구현

게임 목표 달성 시 화면에 UI가 등장하게 하였음


```
public void Destroy_Coin()
{
    CoinSound.Play();
    CoinCount -= 1;
    Coin_Count_TxT.text = CoinCount.ToString();

    if (CoinCount == 0)
    {
        gameClearPannel.SetActive(true);
        finishTime.text = CntTimerCheck.timer.ToString("N0");
    }
}
```

해당 스크립트를 보면 코인 삭제 스크립트 안에 만일 마지막 코인이 삭제되고 남은 코인이 하나도 남지 않는다면 게임 클리어 패널을 띄워주고 클리어 시간을 출력하게 해주었음

다시 시작 구현



ReStart 버튼을 누르면 해당 사진과 같이 다시 게임이 재시작 하는 것을 확인할 수 있음


```
public void GameComplte_ReStartGame()
{
    SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex);
}
```

해당 버튼 안에 들어가는 스크립트

타이머 기능 구현

타이머 스크립트

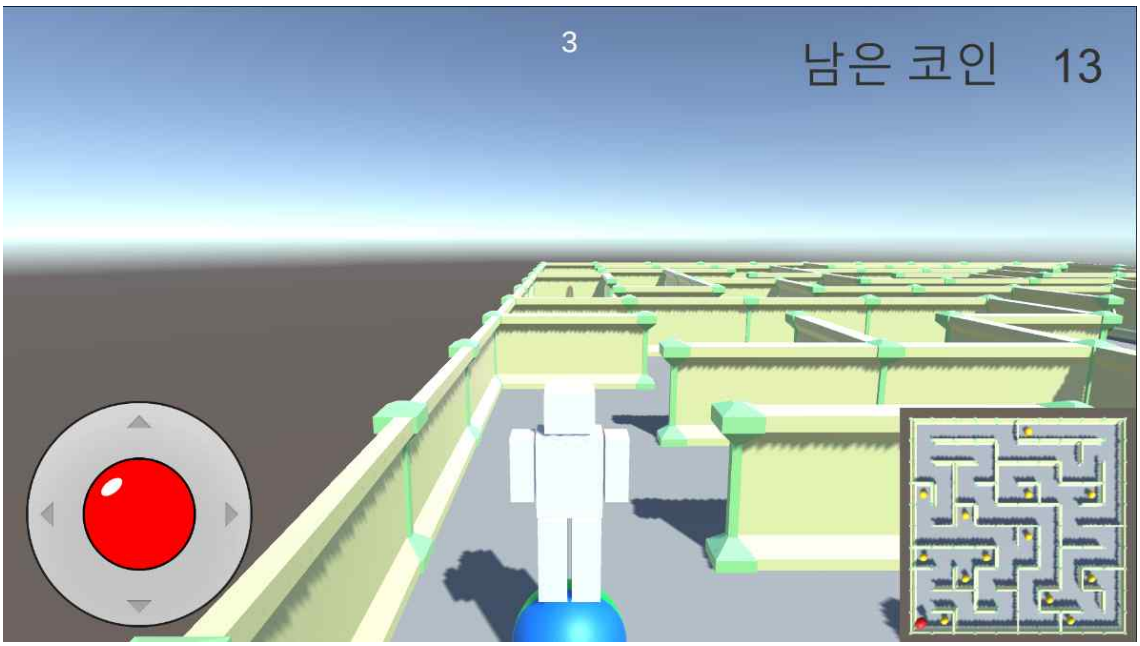
```
public class CntTimerCheck : MonoBehaviour
{
    public TextMeshProUGUI cntText;

    public static float timer;

    // Update is called once per frame
    Unity 메시지 | 참조 0개
    void Update()
    {
        timer += Time.deltaTime;
        cntText.text = timer.ToString("N0");
    }
}
```

실시간으로 시간을 받아오고 소수점을 버리게 만들었습니다.

모바일에서 동작시키기 위한 조이스틱 기능을 구현해보았습니다.

UI 구성

좌측에는 플레이어가 움직일 수 있는 조이스틱을, 화면을 터치 드래그 하여 보는 방향을 바꿀 수 있다.



화면 드래그 스크립트

```
public class CameraDragRotation : MonoBehaviour, IBeginDragHandler, IDragHandler
{
    [SerializeField]
    public Transform cameraArm;

    Vector3 FirstPoint;
    Vector3 SecondPoint;
    public float xAngle = 0f;
    public float yAngle = 55f;
    float xAngleTemp;
    float yAngleTemp;

    [SerializeField]
    private float limit_yAngle_lest = 0;
    [SerializeField]
    private float limit_yAngle_MAX = 70;

    참조 0개
    public void OnBeginDrag(PointerEventData eventData)
    {
        BeginDrag(eventData.position);
    }

    참조 0개
    public void OnDrag(PointerEventData eventData)
    {
        OnDrag(eventData.position);
    }

    참조 1개
    public void BeginDrag(Vector2 a_FirstPoint)
    {
        FirstPoint = a_FirstPoint;
        xAngleTemp = xAngle;
        yAngleTemp = yAngle;
    }

    참조 1개
    public void OnDrag(Vector2 a_SecondPoint)
    {
        SecondPoint = a_SecondPoint;
        xAngle = xAngleTemp + (SecondPoint.x - FirstPoint.x) * 180 / Screen.width;
        yAngle = yAngleTemp - (SecondPoint.y - FirstPoint.y) * 90 * 3f / Screen.height; // Y값 변화가 좀 느려서 3배 곱해줌.

        // 회전값을 40~85로 제한
        if (yAngle < limit_yAngle_lest)
            yAngle = limit_yAngle_lest;
        if (yAngle > limit_yAngle_MAX)
            yAngle = limit_yAngle_MAX;

        cameraArm.rotation = Quaternion.Euler(yAngle, xAngle, 0.0f);
    }
}
```

좌측 하단 조이스틱 스크립트

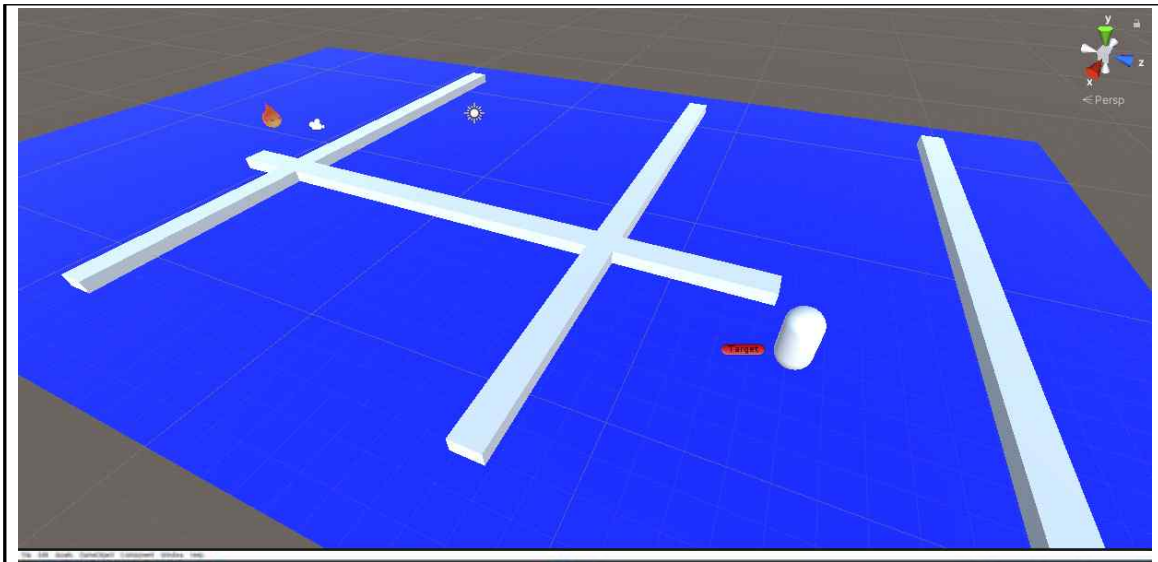
```
6
7       //아래 3개의 인터페이스를 선언하면 에러가 날텐데 컨트롤 + . 으로 바로 선언 가능
        유Unity 스크립트(자산 참조 5개)|참조 0개
8       public class VirtualJoystick : MonoBehaviour, IBeginDragHandler, IDragHandler, IEndDragHandler
9       {
10          [SerializeField]
11          private RectTransform lever;
12          private RectTransform rectTransform;
13
14          [SerializeField, Range(10,150)]
15          private float leverRange;
16
17          // 플레이어 조이스틱 관련
18          private Vector2 inputDirection;
19          private bool isInput;
20
21          [SerializeField]
22          private TPSCharacterController controller;
23
            참조 5개
24          public enum JoystickType { Move,Rotate}
25          public JoystickType joystickType;
26
27          public float sensitivity = 1f;
28
            유Unity 메시지|참조 0개
29          private void Awake()
30          {
31              rectTransform = GetComponent<RectTransform>();
32          }
33          //드래그를 시작할 때
            참조 0개
34          public void OnBeginDrag(PointerEventData eventData)
35          {
36              ControlJoystickLever(eventData);
37              isInput = true;
38          }
39          // 오브젝트를 클릭해서 드래그 하는 도중에 들어오는 이벤트
40          // 하지만 클릭을 유지한 상태로 마우스를 멈추면 이벤트가 들어오지 않음
            참조 0개
41          public void OnDrag(PointerEventData eventData)  // 드래그 중
42          {
43              ControlJoystickLever(eventData);
44          }
```

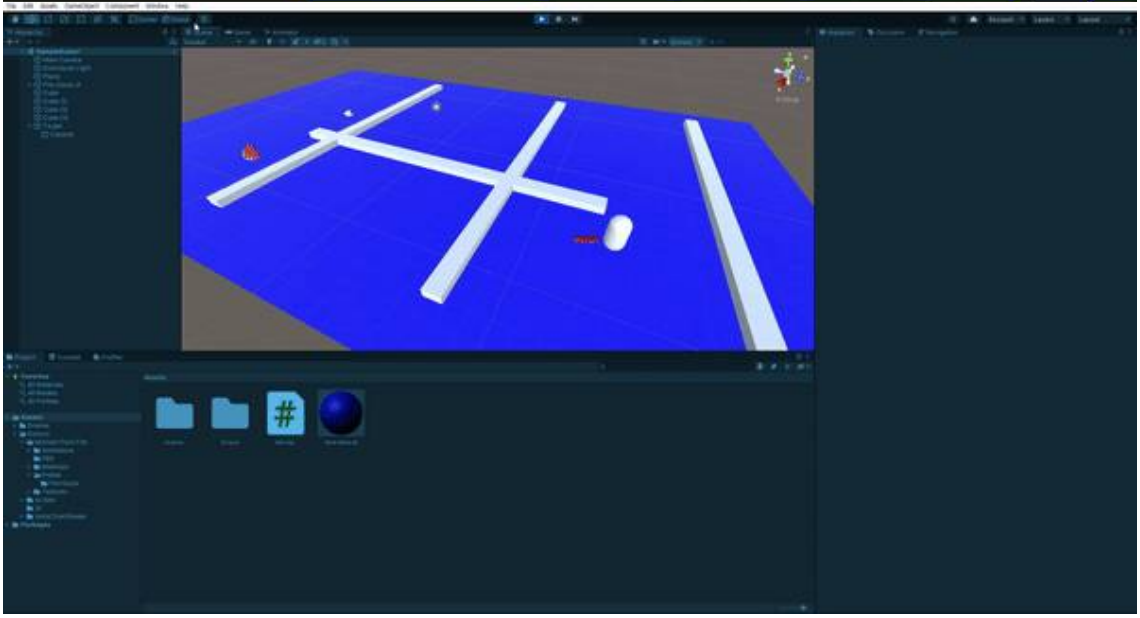
```
    //드래그를 끝냈을 때
    참조 0개
    public void OnEndDrag(PointerEventData eventData)
    {
        // 드래그가 끝나면 다시 중심으로 돌아가게 초기화
        lever.anchoredPosition = Vector2.zero;
        isInput = false;
        //controller.Move(Vector2.zero);

        switch (joystickType)
        {
            case JoystickType.Move:
                controller.Move(Vector2.zero);
                break;
            case JoystickType.Rotate:
                break;
        }
    }

    //1~3: 조이스틱 움직임, 밖으로 안나가게 관련, 4:조이스틱 움직임에 따라 나온 변수 0~1
    참조 2개
    private void ControlJoystickLever(PointerEventData eventData)
    {
        //레버가 있어야할 위치를 구함
        var inputPos = eventData.position - rectTransform.anchoredPosition;
        // inputPos와 leverRange 비교 짧으면 inputPos, 길면 inputpos 정규화 후 leverRange 곱하고 넣는다.
        var inputVector = inputPos.magnitude < leverRange ? inputPos : inputPos.normalized * leverRange;
        lever.anchoredPosition = inputVector;

        //입력값을 0~1로 정규화해서 움직이게 하기 위함 또한 해상도가 변경되었을 경우 입력값이 달라짐
        inputDirection = inputVector / leverRange;
    }

    참조 1개
    private void InputControlVector()
    {
        switch(joystickType)
        {
            case JoystickType.Move:
                controller.Move(inputDirection * sensitivity);
                break;
            case JoystickType.Rotate:
                controller.LookAround(inputDirection * sensitivity);
                break;
        }

        //Debug.Log(inputDirection.x + " / " + inputDirection.y);
    }

    유Unity 메시지 | 참조 0개
    void Update()
    {
        if(isInput)
        {
            InputControlVector();
        }
    }
}
```

몬스터가 타겟을 찾아 스스로 최단 경로로 이동하는 기능을 구현함

몬스터가 플레이어를 추적하는 시스템이 필요하여 해당 씬을 구축하고 테스트하였음

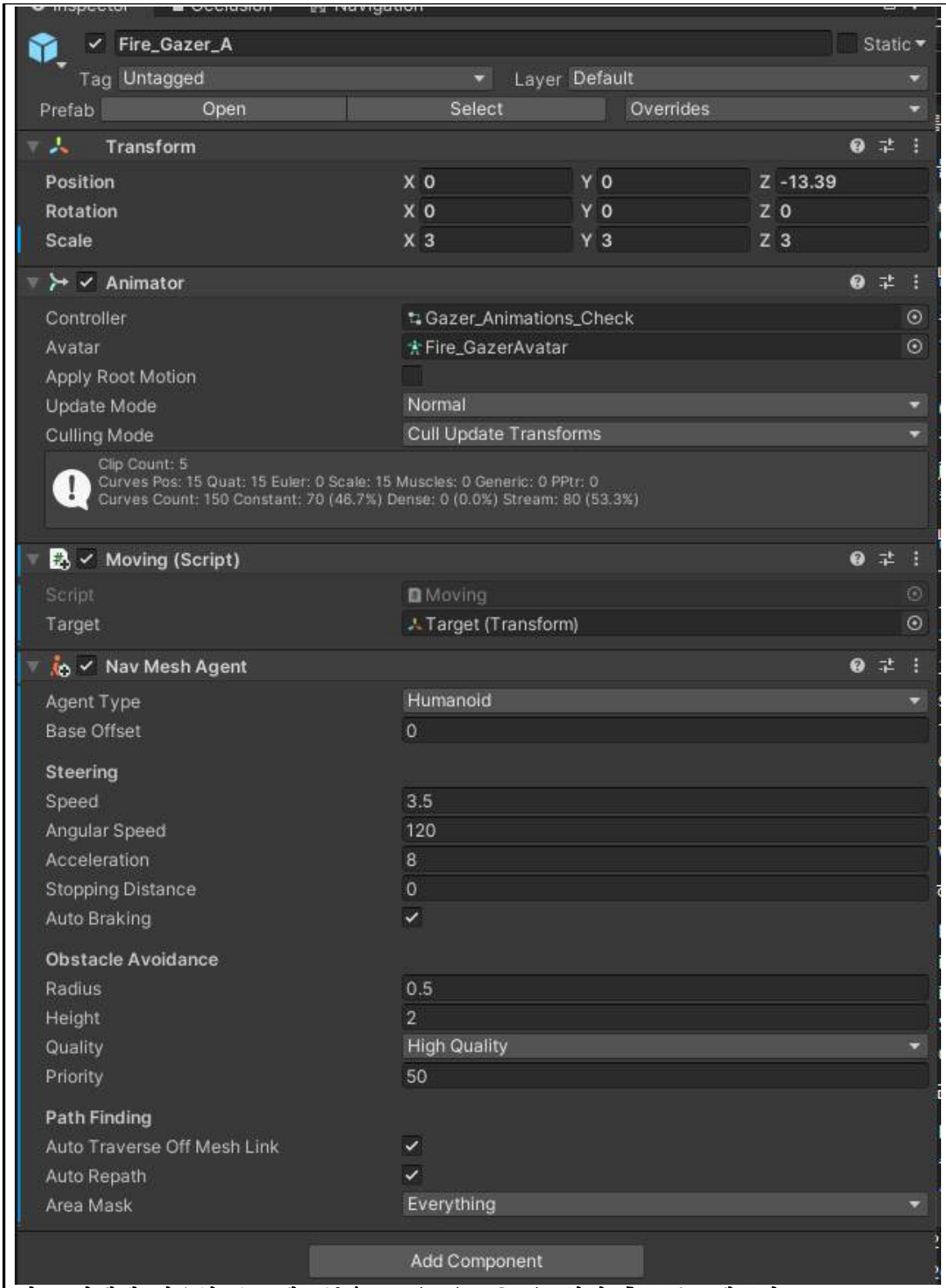
플레이어를 향해 자동으로 경로를 탐색하여 가는 것을 확인할 수 있음





제작 방법

추척할 몬스터에 Nav Mesh Agent, Moving 스크립트를 추가한다.

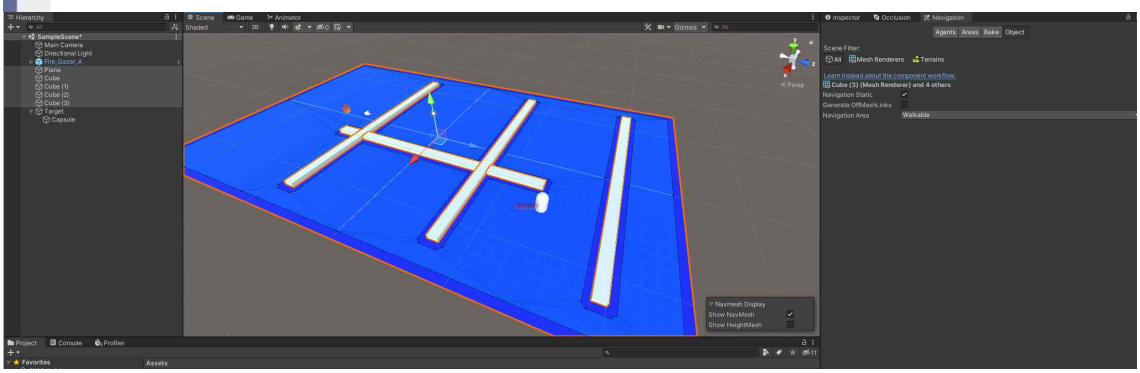


경로 탐색에 필요한 오브젝트들을 Navigation Static 설정 후 Bake 해준다.

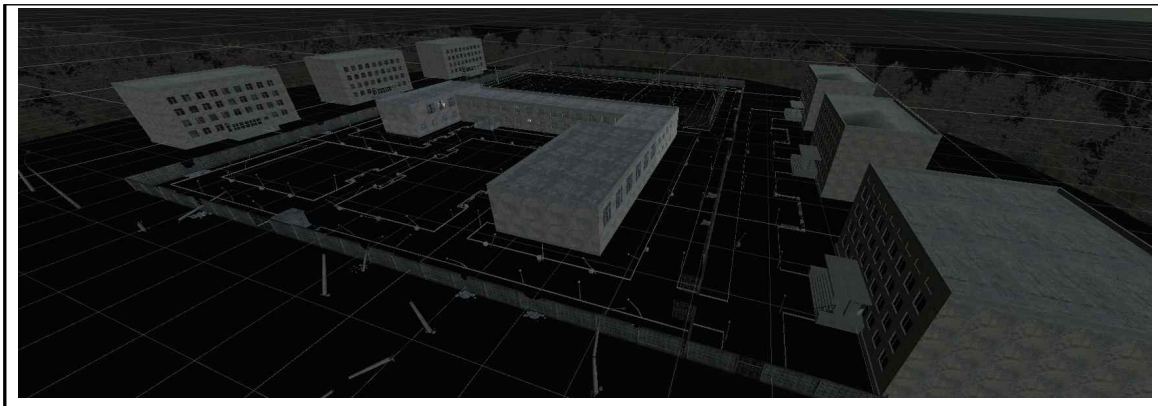
Moving.cs

Assembly-CSharp

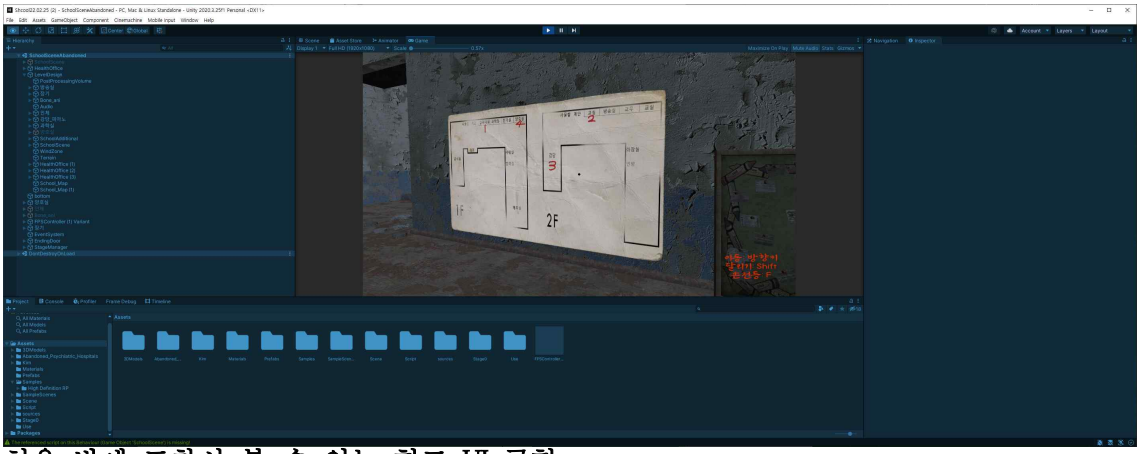
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.AI;

Unity 스크립트(자산 참조 1개)|참조 0개
public class Moving : MonoBehaviour
{
    NavMeshAgent agent;

    [SerializeField]

    Transform target;

    Unity 메시지|참조 0개
    void Start()
    {
        agent = GetComponent<NavMeshAgent>();
    }

    Unity 메시지|참조 0개
    void Update()
    {
        agent.SetDestination(target.position);
    }
}
```



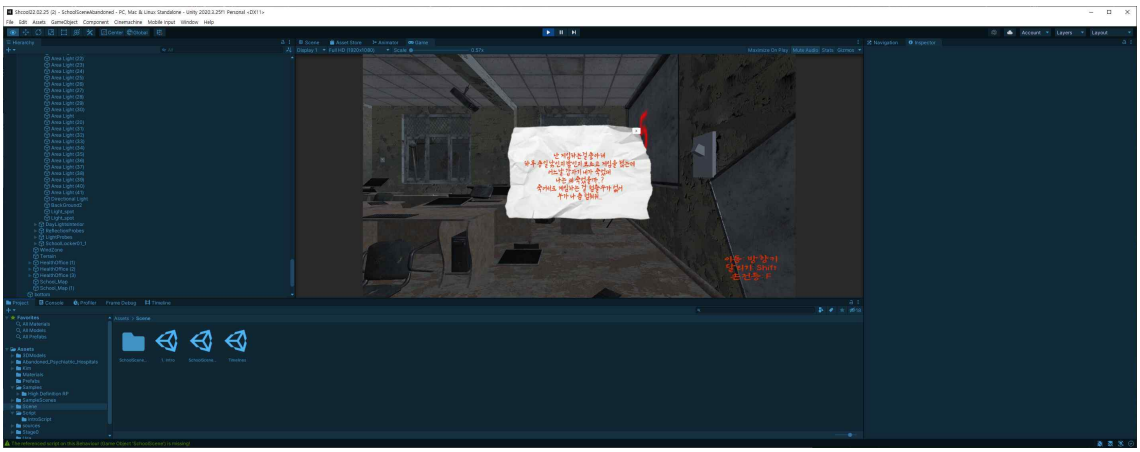
유니티 에셋스토어에서 맵을 구매, 기존에 구현하였던 기능들을 추가하여 프로토타입을 제작하였음



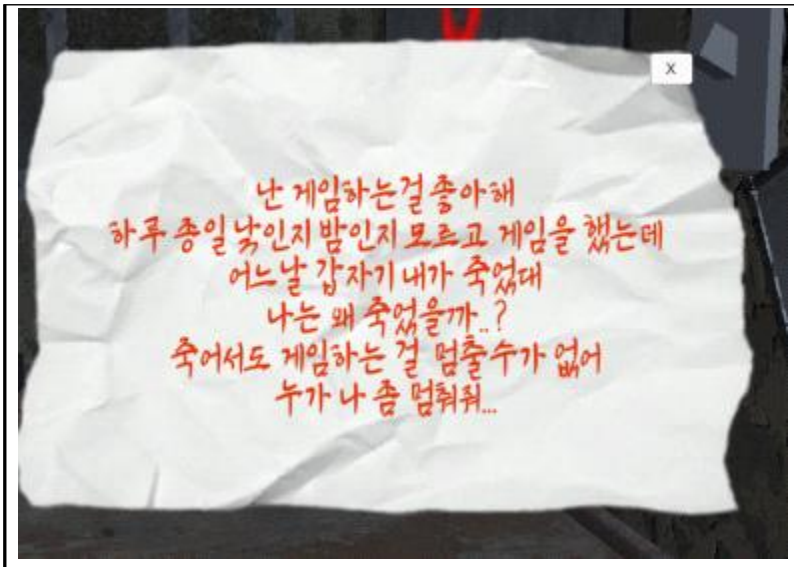
포토샵으로 지도 리소스를 만들어 전체적인 스테이지 흐름을 표시해줌



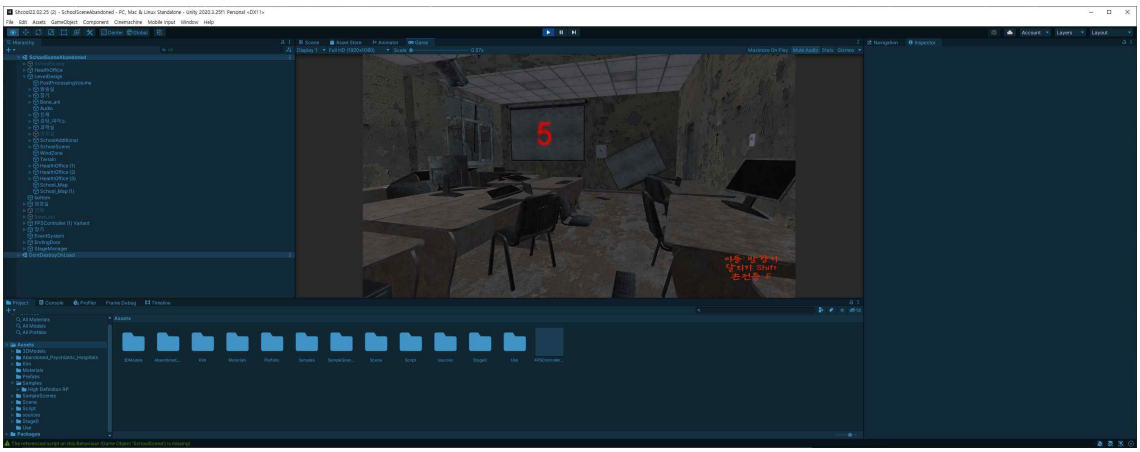
처음 방에 도착시 볼 수 있는 힌트 UI 구현

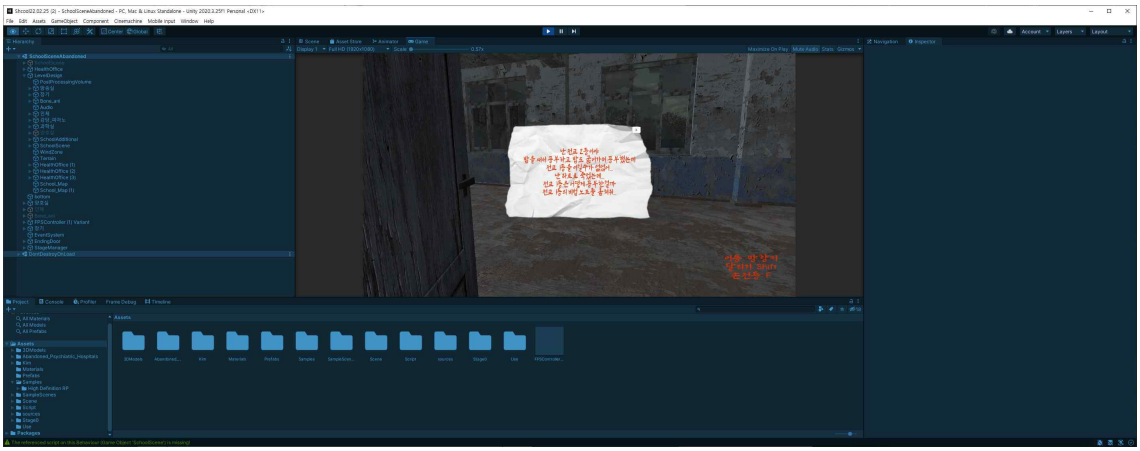


포토샵을 이용하여 힌트 UI 제작



Stage 1 : Stage 안에 있는 모든 스위치를 내리면 칠판에 있는 카운터가 내려감,
카운터가 0이되면 방문이 열림





방을 나오면 다음 스테이지의 힌트 UI가 출력이 됨
RayCast를 이용한 아이템 획득 구현

Camera Vector를 기준으로 Camera Front Normal Vector를 가져와 RayCast를 생성

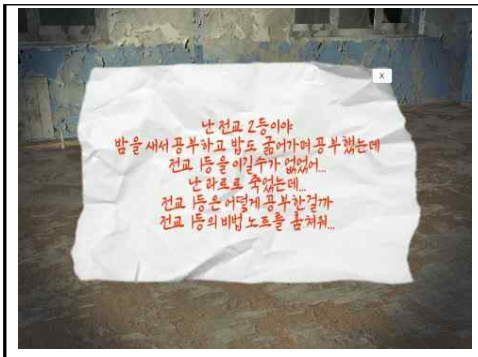
RayCast가 Object의 Collider에 닿으면 Trigger 이벤트가 발생하여 아이템 줍기 UI가 화면에 표시된다.

키보드 이벤트(E)를 누르면 아이템을 획득한다.

SpotLight를 이용하여 손전등을 구현하였음

13주차에 수행했던 프로토타입 제작을 계속 진행하였음

이전에 사용했던 UI를 다시 재사용함



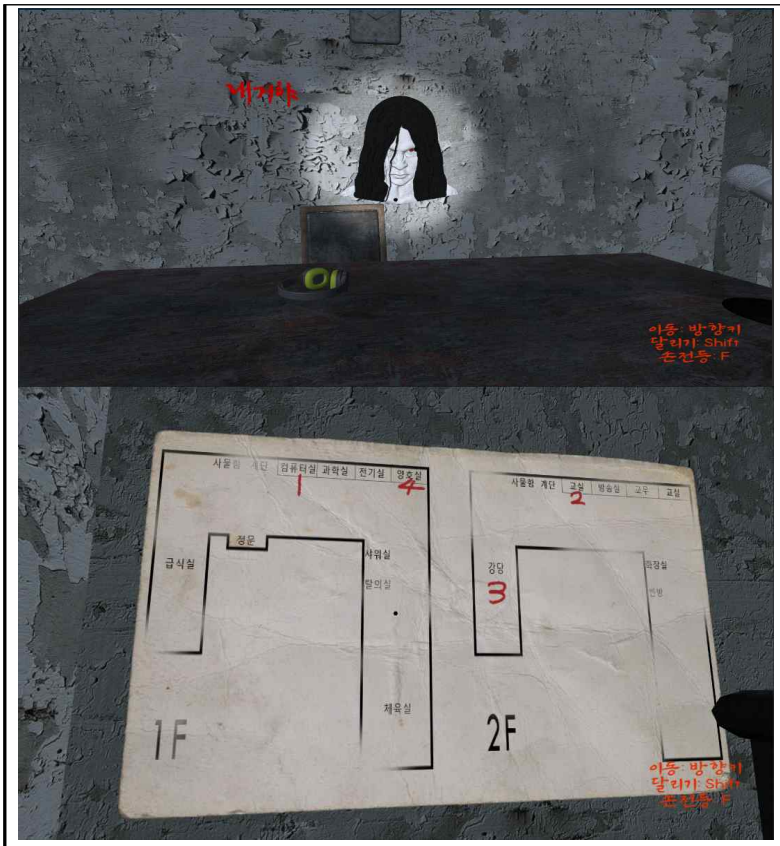
bool 변수를 사용하여 열쇠를 소지하고 있을 시 true로 변경
RayCast가 문에 닿았을 시 Trigger 이벤트가 발생하여 기존에 만들었던 UI를 출력함

그 후 E 버튼을 눌렀을 경우 우측 사진과 같이 문이 열리는 애니메이션이 작동함

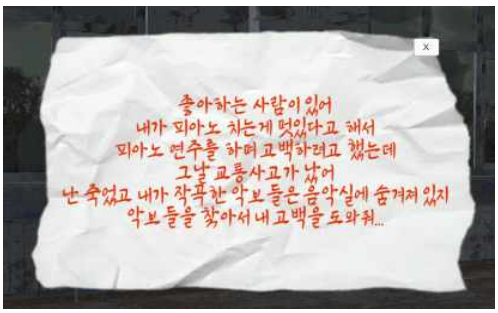
방 안에 들어왔을 경우 Trigger가 발생하여 문이 닫히는 애니메이션이 작동함

그리고 책상 위에 놓여진 책에 RayCast가 닿았을 시 줍기 UI를 출력하여 E키를 눌렀을 시 하단 사진처럼 오디오와 함께 갑자기 귀신이 튀어나오는 연출을 하였음

이전 단계 스테이지를 무시하고 다른 스테이지를 먼저 갔을 경우 나중 스테이지가
먼저 실행되지 않게 하였음

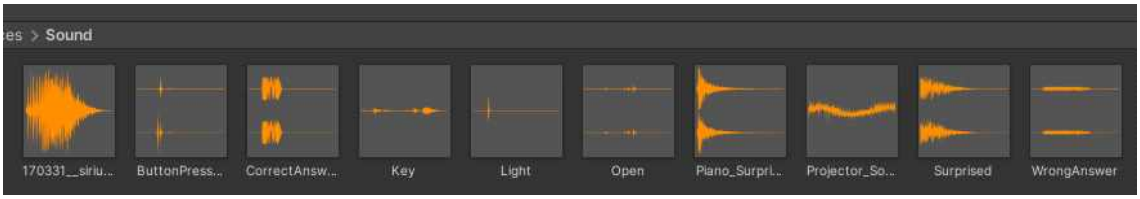


그 후 방에서 나오면 이전에 사용했던 쪽지 UI를 출력하여 Stage3 힌트를 줌



그리고 Stage3 음악실에 입장 시 문이 닫히면서 밑의 gif처럼 괴물이 가까이 왔다 돌아가는 애니메이션을 실행함

그 외 BGM과 효과음을 추가하였음



7. 연구 결과 및 고찰

추후 그래픽 업그레이드를 위해 렌더링 파이프 교체 및 프레임 저하 현상 해결을 위한 최적화 기법 적용 현재 PC밖에 되지 않지만 VR, 스마트폰 등 여러 가지 기기에서도 플레이할

수 있게 변경 예정임.

어색해보이는 애니메이션 수정 및 플레이어를 추적하는 AI 구현

8. 예산 집행 현황

구분	사용 내역	금액
유니티 에셋 구매	Skill Icon Pack, RPG Monster Wave PBR, Horror FPS KIT, Monsters Forest Pack, Quirky Series - Animals Mega Pack, Character Mechanic, Deadly Dungeons: Modern Traps, QTE Lockpicking, Themed Key Unlock System	250,000
합계		250,000

캡스톤디자인 산학연계 교육 실적보고서

캡스톤디자인 교과목명 (교과목코드)	374120			
캡스톤디자인 과제명	캡스톤디자인2			
교육기간	2022년 03월 09일 ~ 2022년 06월 14일			
교육개요	수행내용 지도			
기업체전문가	소속		성명	
교육내용	**과제 프로젝트 진행사항 확인 및 발표** **프로젝트 보안점 및 개선사항 피드백** **향후 일정 및 외부 공모전 탐색 및 계획 수립** **다른 프로젝트 발표 참관을 통한 학습**			
교육운영결과	**과제 프로젝트 진행사항 확인 및 발표를 하였음** **프로젝트 보안점 및 개선사항 피드백을 받음** **향후 일정 및 외부 공모전 탐색 및 계획 수립하였음** **다른 프로젝트 발표 참관을 통한 학습하였음**			

위와 같이 캡스톤디자인(과제명) 산학연계 교육 실적보고서를 제출합니다.

2022년 06월 21일

기업체전문가 : (인)

원광대학교 LINC 3.0 사업단장 귀하

캡스톤디자인 지도 실적 보고서(지도교수용)

캡스톤디자인 교과목명 (교과목코드)	374120			
캡스톤디자인 과제명	캡스톤디자인2			
지도학생				
지도개요	수행내용 지도			
지도교수	소속	디지털콘텐츠공학과	성명	
세부 지도내용	**과제 프로젝트 진행사항 확인 및 발표** **프로젝트 보안점 및 개선사항 피드백** **향후 일정 및 외부 공모전 탐색 및 계획 수립** **다른 프로젝트 발표 참관을 통한 학습**			
수행기간	2022년 03월 09일 ~ 2022년 06월 14일			

위와 같이 캡스톤디자인(과제명)의 실적 보고서를 제출합니다.

2022년 06월 17일

지도교수 : (인)

원광대학교 LINC 3.0 사업단장 귀하