캡스톤디자인(종합설계) 결과보고서

소속학부(과)	디지털콘텐츠공학과	팀명	ALTF4
개설 연도 및 학기	2022 학년도 ■1학기 □2학기	교과목명	캡스톤디자인2(기업연계프로젝트)
과제명	교육용 미니게임		
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	*19800*	19.8천원
	인쇄비		천원
	학생여비	자세히 작성	
	학생회의비	( )천원 × ( )인 × ( )회	천원
			천원
	총액		천원

위와 같이 캡스톤디자인(종합설계) 결과보고서를 제출합니다.

첨부 : 캡스톤디자인(종합설계) 과제 상세 결과보고서[별첨 1호]

2022 년 06 월 18 일

지원학생(팀장)

사업책임자(지도교수) (인)

참여기업 담당자 (인)

원광대학교 LINC 3.0 사업단장 귀하

1. 연구 필요성

(1) 게임화의정의

게임화(Gamification)라는 용어가 처음 인터넷 검색 어로 등장한2008년 이래로 조회 수는 기하급수적으로 증가했으며 이제는 한 달에1,000,000회 이상 검색되고 있다.

이러한 관심에 비해 아직 게임화에 대한 학계의 일치된 정의나 개념은 형성되고 있는 과도기적 상태이지만 관련된 연구들을 종합하면게임화는 다음 두 가지 관점에서 논의되고 있음을 확인할 수 있다.

첫째, 게임화란 게임 이외의 상황에서 게임디자인 요소를 활용하는 것이다. 혁신, 마케팅, 교육, 사회 변화와 같은 비 게임 환경에 게임 메커니즘을 사용하거나 원하는 동작을 촉진하기 위해 게임을 사용하는 것이 이에 해당한다.

둘째, 게임화는 게임과 같은 사고와 방법을 활용해 체험자를 몰입하게 만들어 문제를 해결하는 과정이다. 이 경우 체험자의 몰입을 위해 참여와 즉각적인 피드백을 제공하며 생산성을 향상시키는 과정 이 중점적으로 다루어진다.

위 두 관점은 모두 게임화가 게임 자체만이 아니라 게임을 통해 흥미를 유발하고 이를 다른 활동에 통합시키는 것임을 강조한다.[1]

(2) 교육용기능성 게임의 정의 및 효과

이렇게 게임의 순기능에 대한 연구가 활발한 가운데, 교육용 기능성 게임(Serious Game)을 교육 현장에서 적극적으로 활용하는 시도가 많이 이루어진다.

교육용 기능성 게임이란 동기유발과 자발성을 끌어내기에 유용한 게임의 특성을 이용하여, 사용자가 여러가지 교육적 효과를 거둘 수 있도록 의도적으로 설계한 게임을 말한다.[2]

즉 게임의 특성인 재미와 몰입, 학습에 필요한 내용과 방법이 모두 구현되어 있다.

이러한 교육용 기능성 게임은 학생들의 학업성취도, 학습동기, 학습흥미, 학습효능감에 긍정적인 영향을 주는 것으로 나타났다.

Test	Achievement		Motivation		Interest		Efficacy	
	EXP	CON	EXP	CON	EXP	CON	EXP	CON
Pre	5.3	5.9	6.9	6.4	6.7	6.5	6.1	5.7
Post	9.1	7.1	9.7	6.6	9.5	7.2	9.6	6.4
T (P)	2.690 (0.009)**	2.192 (0.017)*	1.615 (0.003)**	3.618 (0.483)	3.684 (0.001)***	2.772 (0.235)	3.583 (0.001)***	1.811 (0.413)

<그림 1 교육용 기능성 게임의 효과 비교 표>

학업성취도, 학습동기, 학습흥미, 학습효능감으로 구분되는 전체 구인에서 게임화로 수업한 실험집단(EXP)이 비교 집단(CON)에 비해 높은 교육적 효과를 거둔 것으로 나타났다. 또한 이러한 사전·사후 변화에 대한 유의미 정도를 측정한 결과 실험집단의 경우 모든 구인에서 유의미하게 긍정적인 교육적 변화가 일어났음을 확인할 수 있었다. 비교집단 역시 사전에 비하면 사후 모든 영역이 상승했지만 실험집단에 비해서는 상승폭이나 유의미 정도에서 상대적으로 부진한 결과가 나타났다.[3]

이러한 실험 결과를토대로 일반적인 학습보다 게임적 요소를 더한 게임화 수업이 더욱 효과가 좋다는 것을 알 수 있다.

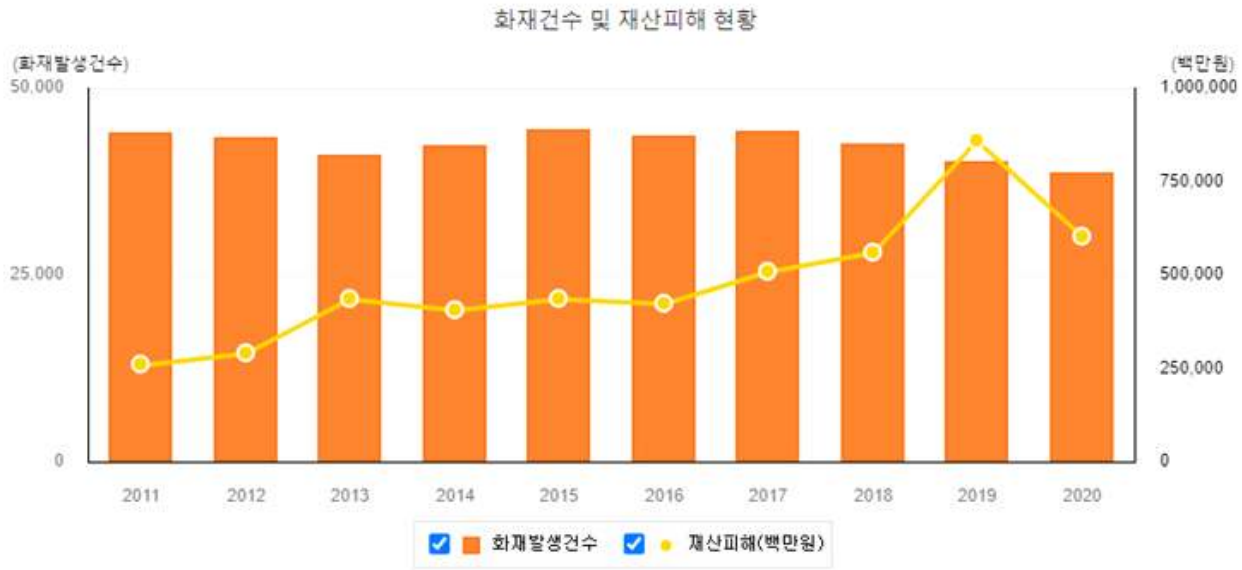
(3) 재난소방 안전 교육용 기능성 게임[4]

교육에 게임을 접목시킨 교육용 기능성 게임을 어떤 주제로 제작해야 할지 조원들끼리 논의 해본 결과 우리 일상에서 가장 가깝다고느낀 안전 교육을 주제로 잡았다.

안전 교육은 안전에 대한 바람직한 행동의 변화와 태도 및 능력을 목표로 하는 교육이다. 현재 소방행정에 있어 소방업무의 활동영역은 화재 예방, 진압, 구조, 구급, 산불 진화, 건물 붕괴, 지진, 폭발, 화생방, 해일, 풍수해 등 거의 모든 면에서 광범위한 업무 영역을 포함하고 있다.

따라서 소방 안전 교육이란 각종 재난 재해를 사전에 예방하고 그 대처 방법을 교육함으로써 재난으로 인한 불행을 예방하고안전을 도모할 수 있도록 변화시키는 것이다.

현대 사회는 급속한 변화와 발전 속에서 언제 어디서나 도처에 각종 사고의 위험이 항상 존재하고 있으며, 각종 대형사고가 발생하여 수많은 인명 피해와 재산피해가 발생하고 있다.



<그림2,3 화재발생건수 및 재산, 인명피해 그래프[5] >

우리 사회가 성장위주로 빠르게 발전하는 가운데, 결과 중심의 성과주의로 인해 절차와 과정이 무시되면서 안전하도록 지켜야 할 규정을 어기고, 그렇게 해도 아무 거리낌이 없는'안전 불감증' 이란 병을 얻었다.이런 의식들이 우리 사회 곳곳에서 대형 사고를 유발시켰으며, 국민들을 불안에 떨게 하였다.

그동안의 안전 교육이 있음에도 불구하고 항상 안전 불감증이 존재하는 근본적인 이유는안전교육을 어려서부터 성인까지 지속적으로 실시하지 않아서 그렇다. 그러므로 우리는 안전교육을 사용자들에게 더욱 친근하고 편하게 접근시킬 수 있도록 소방안전교육용 기능성 게임을 제작하기로 하였다.

<그림 9 Unity>

과제의 기본적인 틀은 유니티로 제작할 예정이다. 유니티는 게임 제작에 특화 되어있고, 다양한asset들이 존재하여 필요한 자료들을 쉽고 빠르게 획득 할 수 있을 것이다.
기본적으로 2D로 제작되는 게임이므로 그래픽의 느낌은 카툰그래픽으로 제작 할 예정이다. 현실감 있는 그래픽 보다는 더 따뜻한 느낌을 주기 때문에 더욱 이용자에게 친근하게 접근 할 수 있을 것이다.'

2. 연구 목표

- 이용자들이 소방안전교육을 더욱 쉽게 접근하게 하는 기능성 게임제작.
- 게임을 즐기며 안전교육을 상세한 내용을 확인 할 수 있는 도감메뉴 제작
- 카툰그래픽 제작으로 인한 그래픽 제작

3. 관련 연구 분석

(1)유사 컨텐츠 및 아이디어/소프트웨어

1. 말랑말랑 두뇌교실

<그림말랑말랑 두뇌교실>

-닌텐도DS의 기능성 게임으로 다양한 미니게임을 이용해 기억력, 사고력 등의 두뇌에 도움이 되는 효과를 가진 게임이다.

2.메이드 인 와리오

<그림메이드 인 와리오>

- 닌텐도의 메이드 인 와리오 시리즈는 간단한 조작으로 이루어진미니게임을5초 이내로 클리어 하며 연속으로 도전하는 게임이다.

우리는 소방안전교육용 기능성 게임을 제작하기에 앞서 어떻게 하면이용자에게 더욱 쉽게 접근 할 수 있을지 고민했다. 그렇게 나온 아이디어가 '미니게임'이다. 앞서 제시한 두개의 사례에서는 매우 간단한 조작을 이용해 이용자에게 쉽고 빠르게 게임을 이해 시킨다. 우리는 그 점에 주목하게 되었고 과제에 접목시키기로 하였다.

4. 연구 수행 내용

<게임 컨셉 기획>

게임을 제작하기 앞서 다양한 게임들이 제작되어야하는 과제의 특성 덕분에 게임 컨셉들을 먼저기획 하였다.

<그림 10. 컨셉 기획 초안>

우선적으로 나온 컨셉은 아래와 같다.

No	게임 컨셉	비고
1	비상구 찾기	화재
2	대피 시 요령	화재
3	소화기, 소화전 작동법	화재
4	표지판 퀴즈	표지판
5	화재 진압(소화기)	화재
6	소화기 검사하기	화재
7	화재 초기 진압(담요 및 밟기)	화재
8	소화기 보관 요령	화재
9	비상벨 누르기	대피

10	대중교통 비상탈출(버스, 지하철)	대피
11	완강기 작동	대피
12	투척 소화기	화재
13	구조 신호 보내기	대피
14	건물에서 떨어진 공터로 대피	대피
15	문고리 온도 체크 후 대피	화재

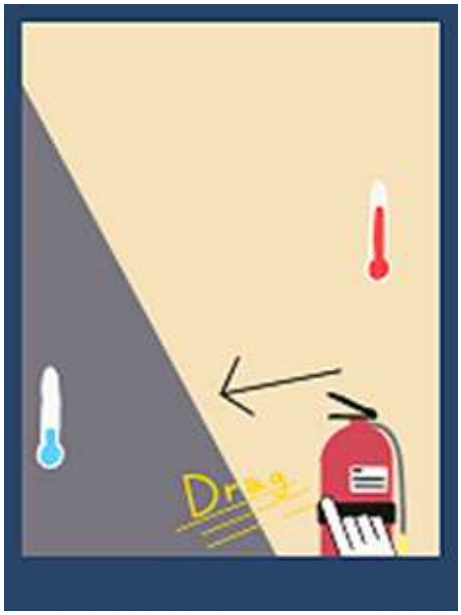
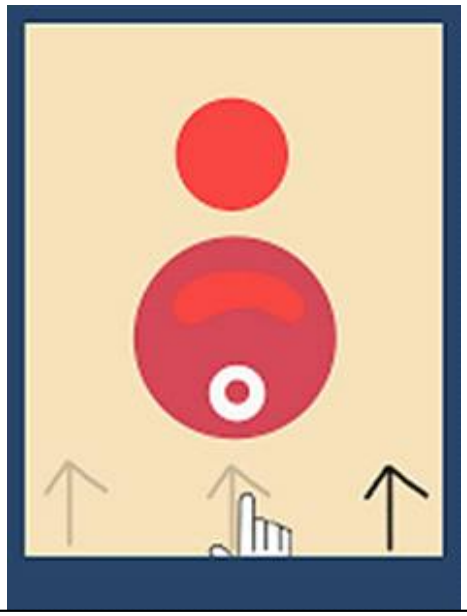
컨셉구체화시킨 내용을 아래 표에 정리하였다.

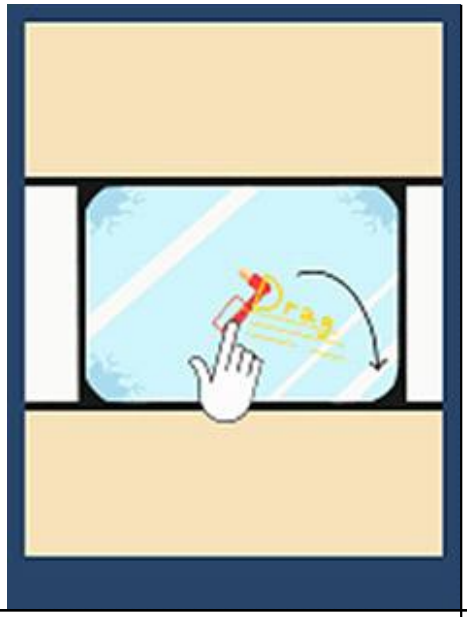
NO	이미지	설명
1		<비상구 찾기> - 출입구나 출입문을 제시해주며 출구위에 있는 비상구 표지를 보고 올바른 비상구를 선택하는 게임 - 난이도 차별점 1레벨: 표지판만 제시 2레벨: 다른 네온 사인 표지판 추가로 헷갈리게 설치

2		<대피 시 요령> - 화재 대피시에는 벽을 짚고 코와 입을 젖은 천으로 막고 저자세로 대피한다. - 화재 현장에 서있는 캐릭터의 허리를 숙이거나 젖은 천을 가져다주며 클리어 - 난이도 차별점 1레벨: 캐릭터의 허리를 숙여 저자세로 만들어 클리어 2레벨: 캐릭터의 허리를 숙이고 젖은 천을 전달해 클리어
3		<소화기, 소화전 작동법> - 소화기 사용법 1. 손잡이를 잡은 상태에서 소화기를 든다. 2. 소화기 밑바닥을 손으로 받쳐 든다. 3. 바람을 등지고 화점 부근으로 접근한다. 4. 안전핀을 뽑는다. 5. 노즐이 화점을 향하게 한다. 6. 손잡이를 강하게 누른다. 7. 소화가 완전히 되었는지 확인한다. - 소화전 사용법 1. 소화전함 위의 기동용 버튼 또는 발신기를 누른다. 2. 소화전함을 열고 호스를 꺼내어 불이 난곳까지 꼬이지 않게 펼친다. 3. 소화전 밸브를 왼쪽 방향으로 돌리면서 서서히 연다. 4. 호스 끝부분을 두 손으로 잡고 불이 난 곳을 향하여 불을 끈다.

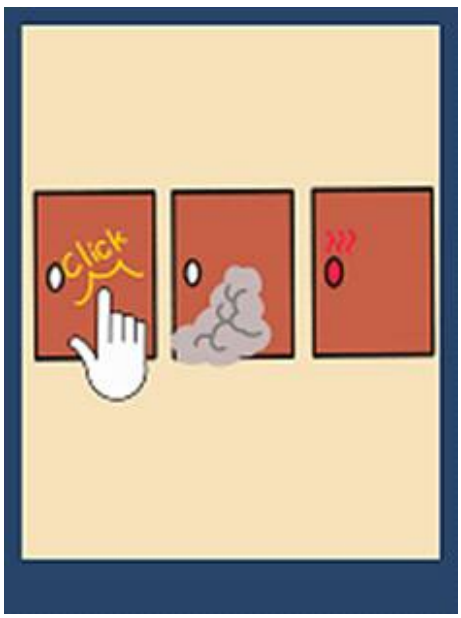
		-난이도 차별점 이 게임에서는 사용법에만 집중하여 게임 제작 소화기: 안전핀 뽑기조준손잡이 누르기 소화전: 소화전 열기꺼내기돌리기소화
4		<표지판 퀴즈> - 표지판3개 제시하여 다른 표지판 선택하는 퀴즈 게임
5		<화재 진압(소화기)> - 손을 대면 소화기 발사, 불을 끈다. 불을 끌 때는 빗질하듯이 바닥을 쓸며 끈다. -난이도 차별점 1레벨: 소화기 이용해서 불을 꺼서 클리어 2레벨: 바람 방향에 유의하여 불을 끈다.
6		<소화기 점검하기> - 점검목록

		안전핀 상태 확인 손잡이 및 봉인줄 상태 확인 용기 상태 용기 받침대 상태 호스 및 노즐 상태 압력게이지 정상 여부 - 난이도 차별점 소화기의 문제 있는 부분을 한군데 만들어서 틀린그림 찾기처럼 클릭하여 클리어
7		<화재 초기 진압(불씨 밟기)> - 담배 꽁초와 같은 작은 불씨로 인해서 화재가 발생하는 경우가 있다. - 담배와 같은 불씨를 연속해서 터치하여 불을 끈다.
8		<소화기 보관 요령> - 높은 온도와 습기, 직사광선을 피해 배치 - 올바른 위치에 배치시켜 게임을 클리어

		
9		<비상벨 누르기> - 화재 최초 발견시 “불이야“를 외치며 비상벨을 눌러 화재 사실 전파 - 빠르게 비상벨을 눌러 게임 클리어
10		<대중교통 비상탈출(버스, 지하철)> - 버스 내부 비상탈출 망치 혹은 대체 물품으로 창문 아래에서10cm 부근을 가격해 밀면 창문을 부수고 나갈 수 있다. - 지하철 출입문 옆 의자 아래 위치한 비상시 커버를 열고 손잡이를 앞으로 당기면 수동으로 문을 열 수 있다. -난이도 차별점

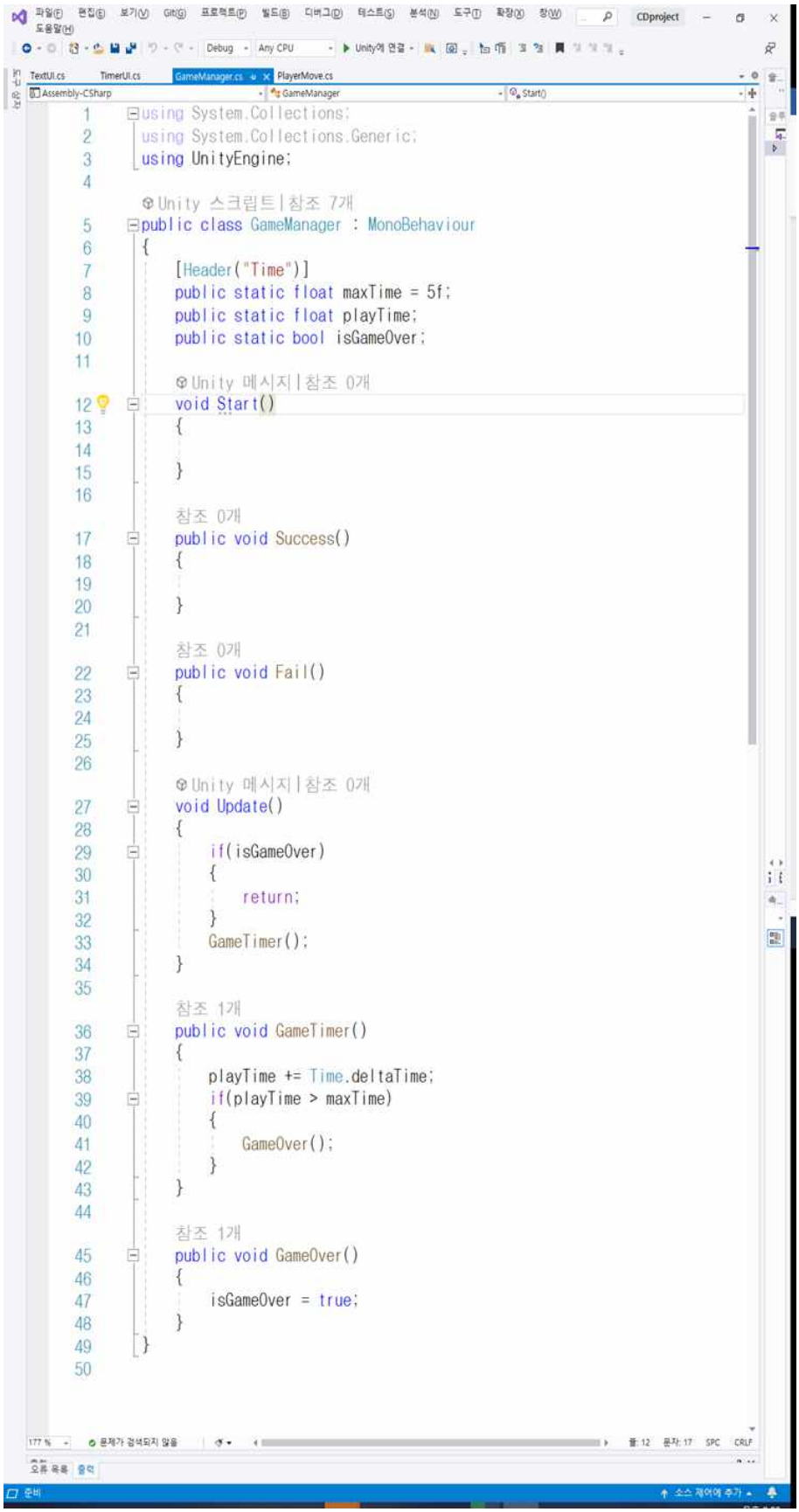
		- 버스: 창문을 연타하여 부숴 탈출 - 지하철: 문 개폐하여 탈출
11		<완강기 작동> - 완강기 작동법 1. 완강기 보관함에서 구성품을 모두 꺼냄 2. 외부 고정 지지대와 조속기를 연결용 고리를 이용해 연결 3. 구조용 안전 벨트 착용 4. 고정 지지대를 하강할 곳으로 이동시키고 로프릴을 하강시킨다. 5. 안전하게 하강
12		<투척 소화기> - 투척용 소화기 사용법 1. 2미터 이상 떨어진곳에서 투척 2. 깨지지 않는 곳에서 투척하지 말 것 3. 일반화재, 목재화재는 발화점을 향해 투척 4. 유류화재는 발화점이 아닌, 주변 바닥이나 벽에 투척

		5. 투척후 안전한 위치로 대피 -난이도 차별점 1레벨: 화재 지점에 투척으로 클리어 2레벨: 화재 지점에 소파나 침대를 배치해 정확한 위치에 던지게 제작
13	 - SOS - 모스부호 ?	<구조 신호 보내기> -SOS 짧은 신호3번긴 신호3번짧은 신호3번 -퀴즈 형식으로 제작하여 다양한 구조 신호를 문제로 제시한다.
14		<건물에서 떨어진 공터로 대피> - 지진이나 화재 대피시 위에서 떨어질 낙하물에 대비해 건물에서 떨어진 공터로 대피한다. - 건물에 있는 사람 무리들을 넓은 공터로 이동시킨다. - 난이도 차별점 건물과 공터의 위치를 변경해 매번 같은 게임을 하는 느낌을 감소시킨다.

		
15		<문고리 온도 체크 후 대피> - 화재 대피시 문에 손을 대어 본 후 연기와 화기가 없다고 생각이 들때에는 어깨로 문을 떠받친 다음 문쪽의 반대 방향으로 고개를 돌리고 숨을 멈춘후 조심해서 비상구나 출입문을 열고 대피한다. - 문의 문고리를 빨갛게 해서 열기를 표현한다거나, 문아래나 위에서 연기를 나오게 하여 연기를 표현해 그런점이 없는 문을 고르게 게임 제작

<게임제작>
-기본세팅
유니티버전: 2020.3.33f

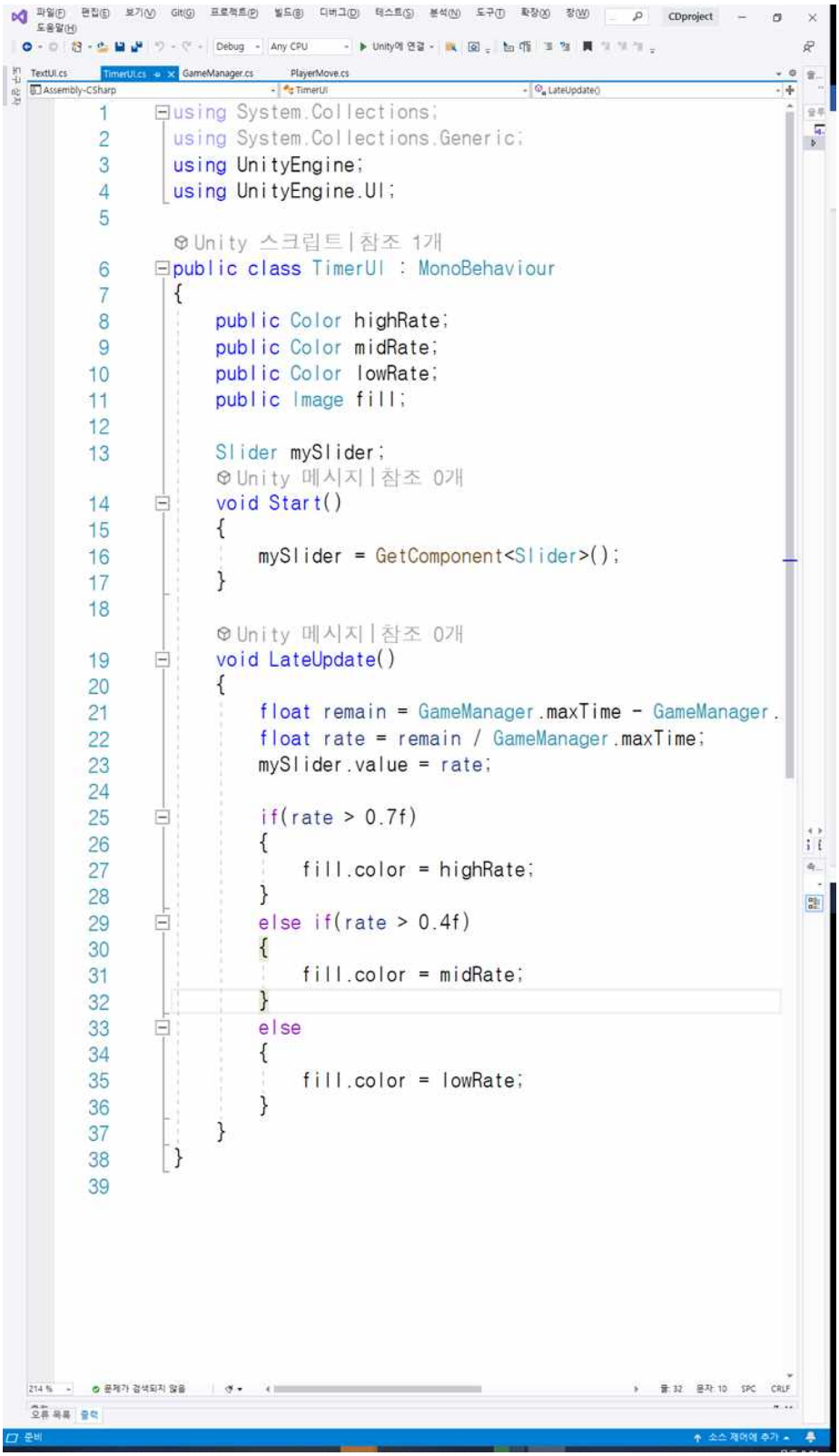
0. 기본적인 기능
0-1. 타이머



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

Unity 스크립트 | 참조 7개
public class GameManager : MonoBehaviour
{
    [Header("Time")]
    public static float maxTime = 5f;
    public static float playTime;
    public static bool isGameOver;

    Unity 메시지 | 참조 0개
    void Start()
    {

    }

    참조 0개
    public void Success()
    {

    }

    참조 0개
    public void Fail()
    {

    }

    Unity 메시지 | 참조 0개
    void Update()
    {
        if(isGameOver)
        {
            return;
        }
        GameTimer();
    }

    참조 1개
    public void GameTimer()
    {
        playTime += Time.deltaTime;
        if(playTime > maxTime)
        {
            GameOver();
        }
    }

    참조 1개
    public void GameOver()
    {
        isGameOver = true;
    }
}
```

<GameManager.cs>

시간관리와 게임의 성공 실패를 처리하는 코드.

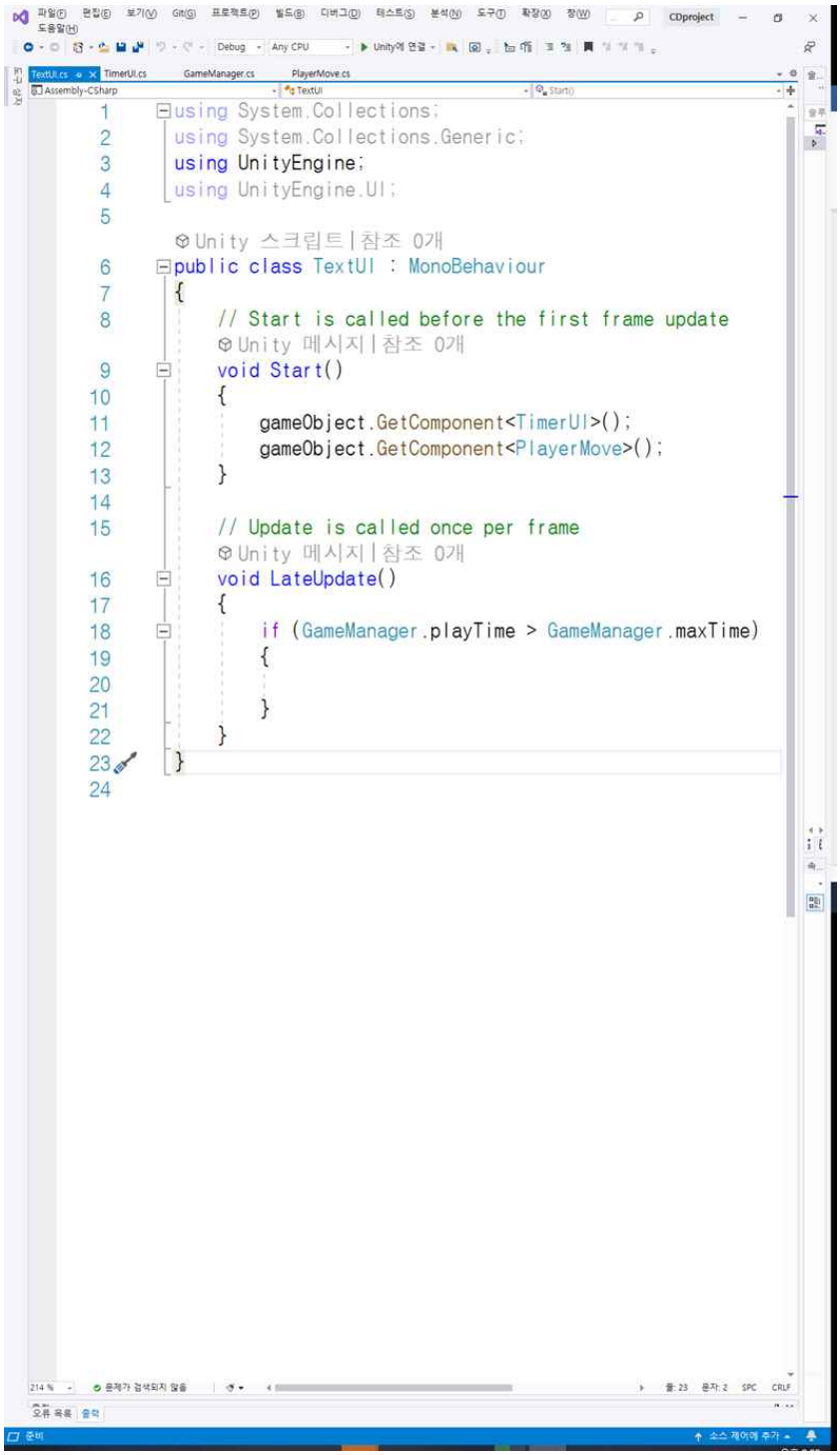


```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

Unity 스크립트 | 참조 1개
public class TimerUI : MonoBehaviour
{
    public Color highRate;
    public Color midRate;
    public Color lowRate;
    public Image fill;

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

    Unity 메시지 | 참조 0개
    void LateUpdate()
    {
        float remain = GameManager.maxTime - GameManager.
        float rate = remain / GameManager.maxTime;
        mySlider.value = rate;

        if(rate > 0.7f)
        {
            fill.color = highRate;
        }
        else if(rate > 0.4f)
        {
            fill.color = midRate;
        }
        else
        {
            fill.color = lowRate;
        }
    }
}
```

<TimerUI.cs>

GameManger에서 시간처리를 받아와 슬라이더를 이용해 시간바를 추가하였다.

시간바는시간의 잔여 여부에 따라 3단계로 색을 변하게 제작하였다.

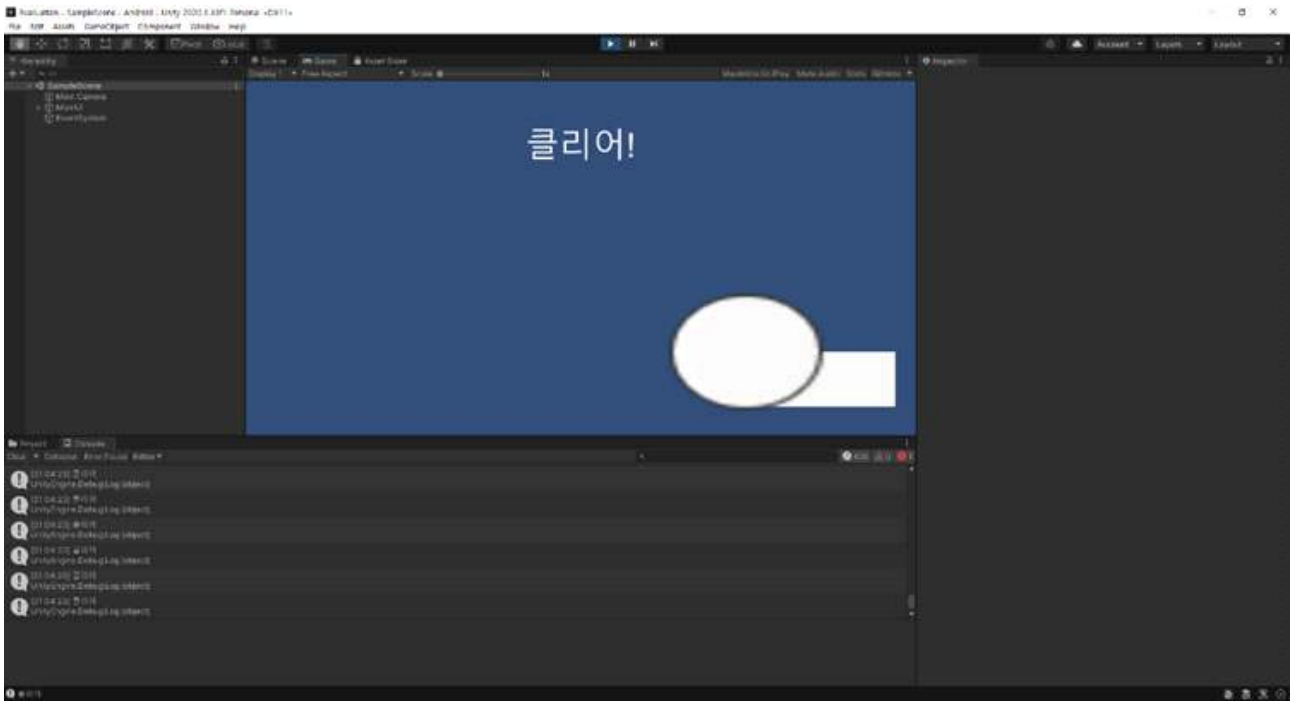


```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

Unity 스크립트 | 참조 0개
public class TextUI : MonoBehaviour
{
    // Start is called before the first frame update
    Unity 메시지 | 참조 0개
    void Start()
    {
        gameObject.GetComponent<TimerUI>();
        gameObject.GetComponent<PlayerMove>();
    }

    // Update is called once per frame
    Unity 메시지 | 참조 0개
    void LateUpdate()
    {
        if (GameManager.playTime > GameManager.maxTime)
        {

        }
    }
}
```

<TextUI.cs>

게임의성공과 실패를 띄워주는 코드.

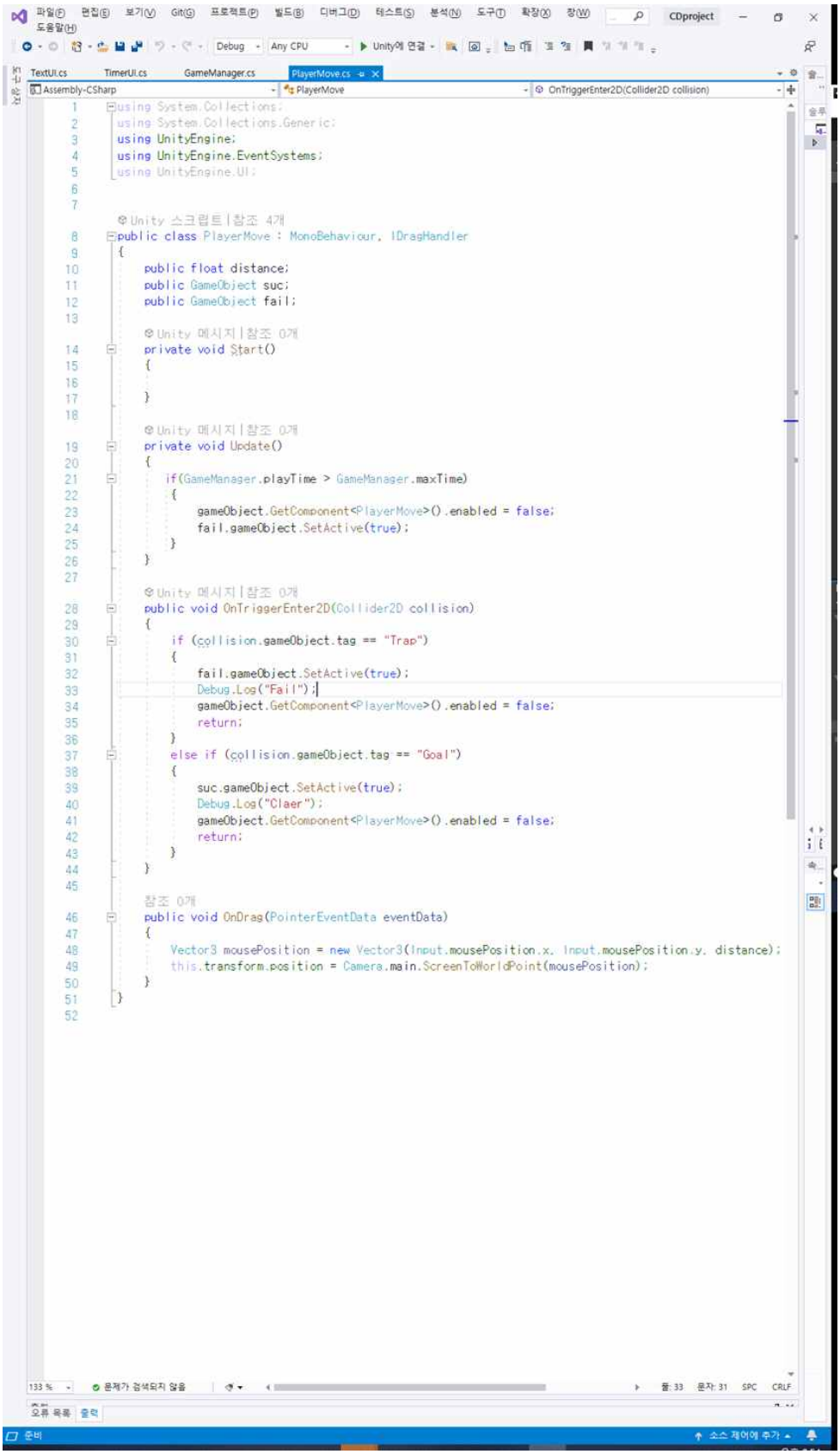
1. 대피시 요령



대피시 요령 내용중'화재 시 몸을 저자세로 낮추고 이동한다'를 게임으로 제작하였다. 서있는 캐릭터(동그라미)를 아래로 내려 몸을 낮추게 만들면 되는 게임5초안에 몸을 내리면 클리어하고 몸을 내리지 못하면 실패하는 게임으로 제작하였다.
차후어울리는 이미지로 대체하여 제작하면 될 것이다.

14.건물에서떨어진 곳으로 대피

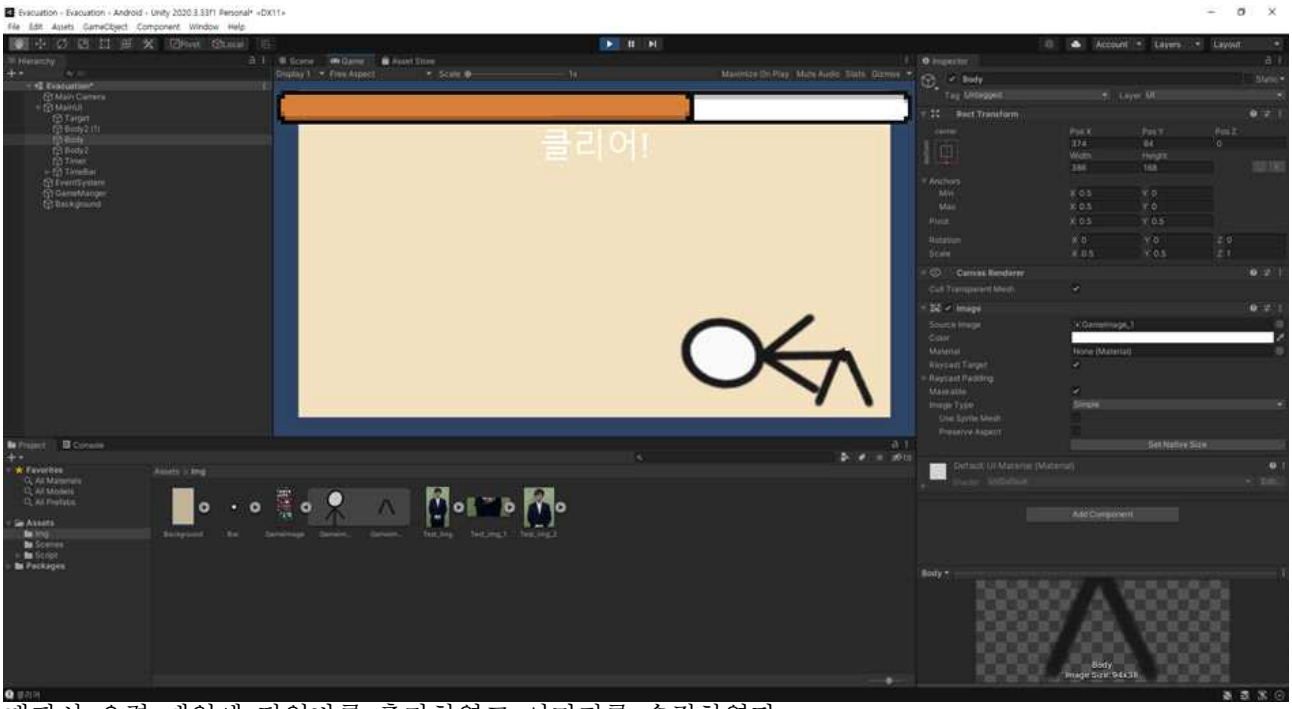
건물에서떨어진 공터로 대피하는 내용을 담은 게임을 제작하였다. 아래의 사람 무리를SAFE존으로 옮겨 클리어하는 게임이다. 땅에 있는 금에 캐릭터가 닿을 시 게임에서 패배한다.



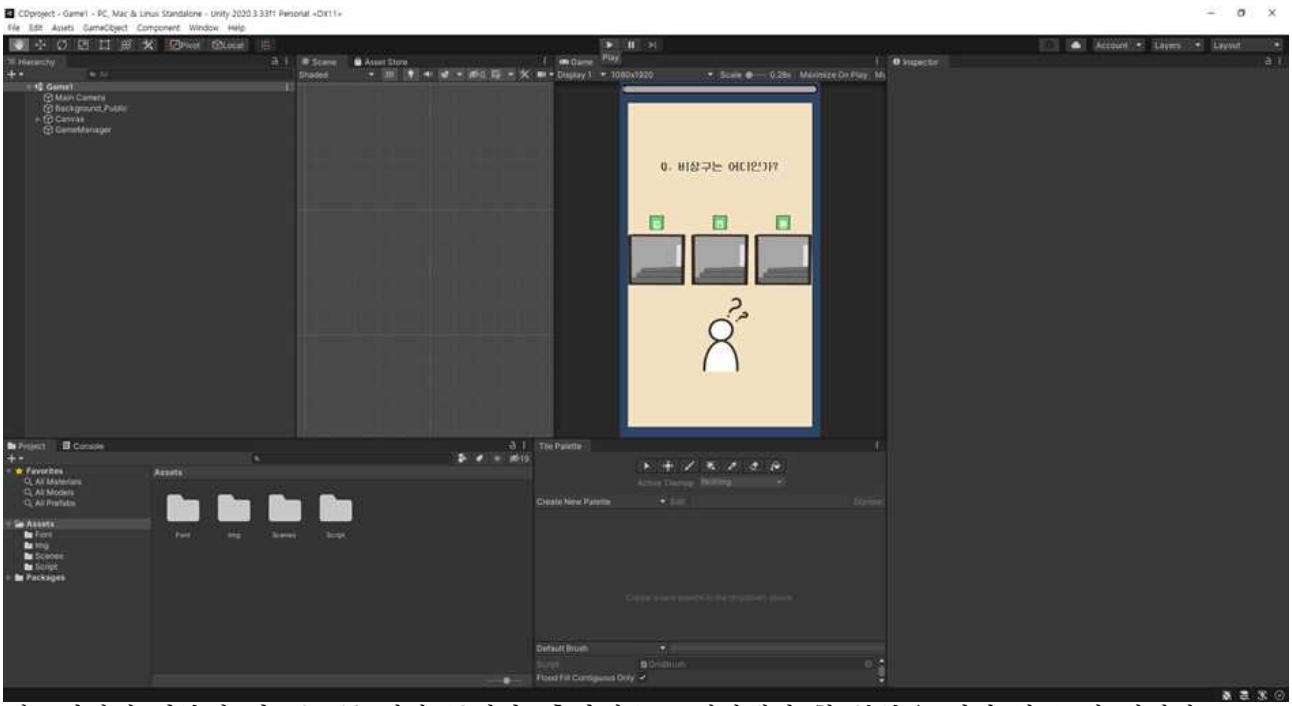
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.UI;


Unity 스크립트 | 참조 4개
public class PlayerMove : MonoBehaviour, IDragHandler
{
    public float distance;
    public GameObject suc;
    public GameObject fail;

    Unity 메시지 | 참조 0개
    private void Start()
    {

    }

    Unity 메시지 | 참조 0개
    private void Update()
    {
        if(GameManager.playTime > GameManager.maxTime)
        {
            gameObject.GetComponent<PlayerMove>().enabled = false;
            fail.gameObject.SetActive(true);
        }
    }

    Unity 메시지 | 참조 0개
    public void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.gameObject.tag == "Trap")
        {
            fail.gameObject.SetActive(true);
            Debug.Log("Fail");
            gameObject.GetComponent<PlayerMove>().enabled = false;
            return;
        }
        else if (collision.gameObject.tag == "Goal")
        {
            suc.gameObject.SetActive(true);
            Debug.Log("Claer");
            gameObject.GetComponent<PlayerMove>().enabled = false;
            return;
        }
    }

    참조 0개
    public void OnDrag(PointerEventData eventData)
    {
        Vector3 mousePosition = new Vector3(Input.mousePosition.x, Input.mousePosition.y, distance);
        this.transform.position = Camera.main.ScreenToWorldPoint(mousePosition);
    }
}
```

<사람무리 옮기는 코드>

터치위치를 받아 캐릭터 위치를 옮기고, Trap 태그에 닿을 시 패배,Goal 태그에 닿을 시 클리어 하는 구조로 구성하였다.



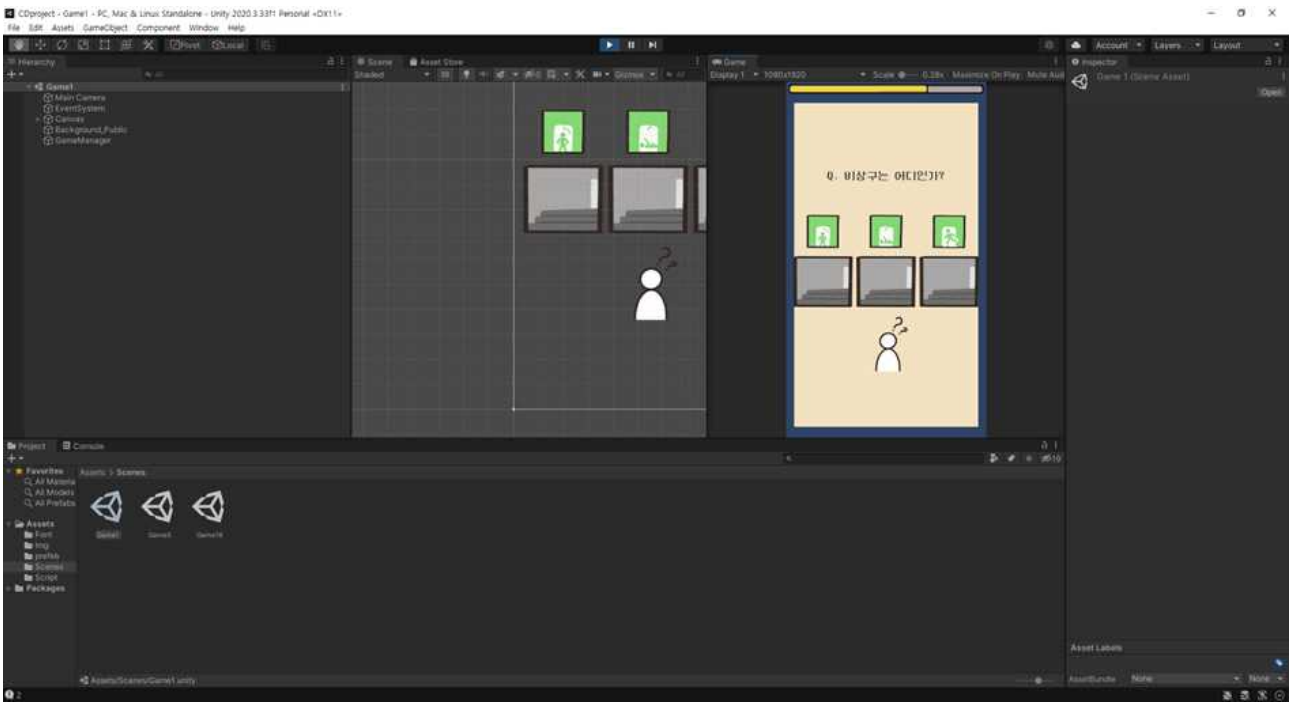
대피시 요령 게임에 타임바를 추가하였고 이미지를 수정하였다.



퀴즈게임에 기본이 될 구조를 제작 중이다. 추가적으로 제작해야 할 부분은 랜덤 한 문제 제시하는 코드, 지문 위치 섞어주는 코드 제작할 예정이다.

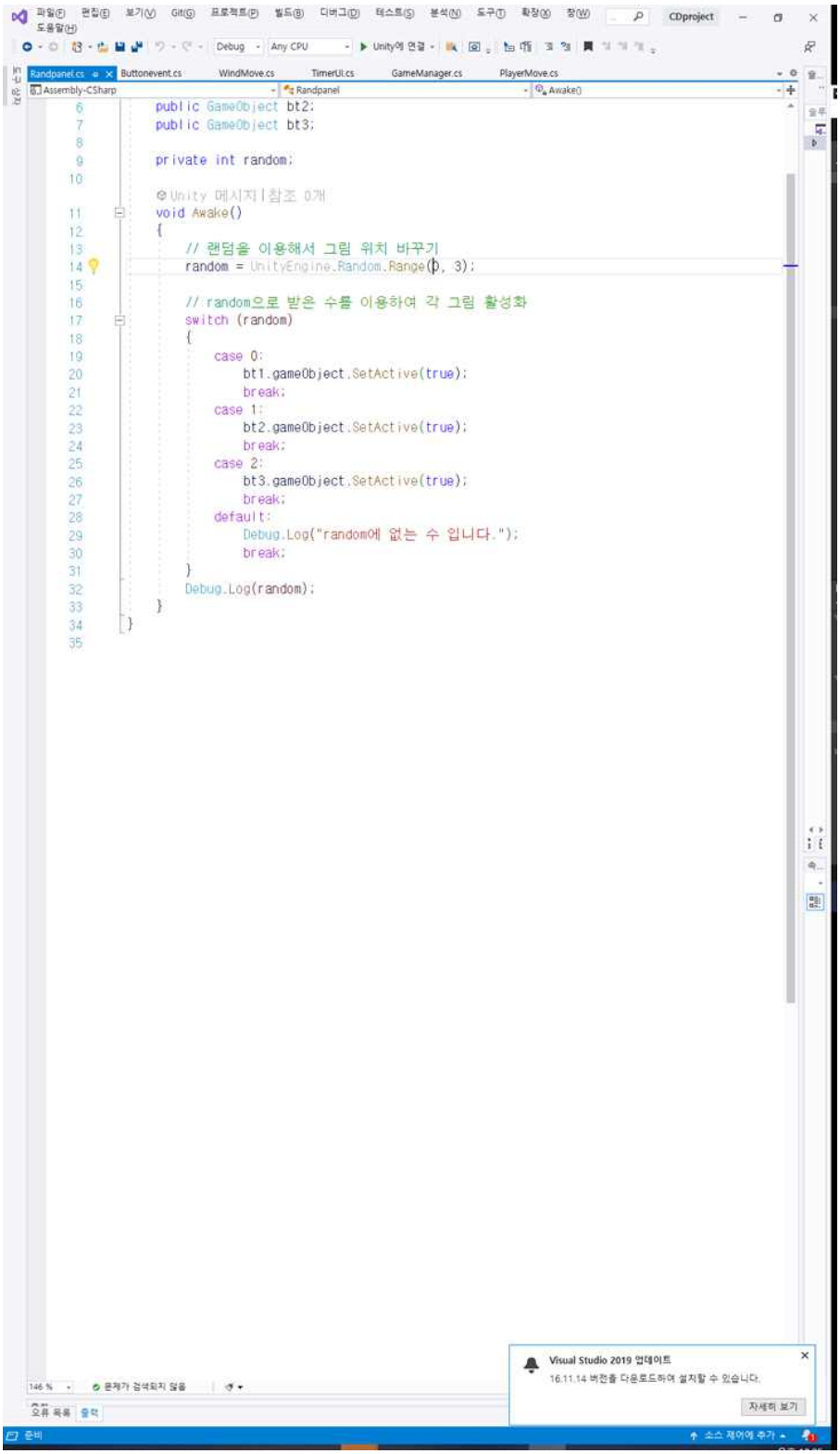
메인화면 초안과 버튼 이미지 초안도 제작 중이다.

<11주차>



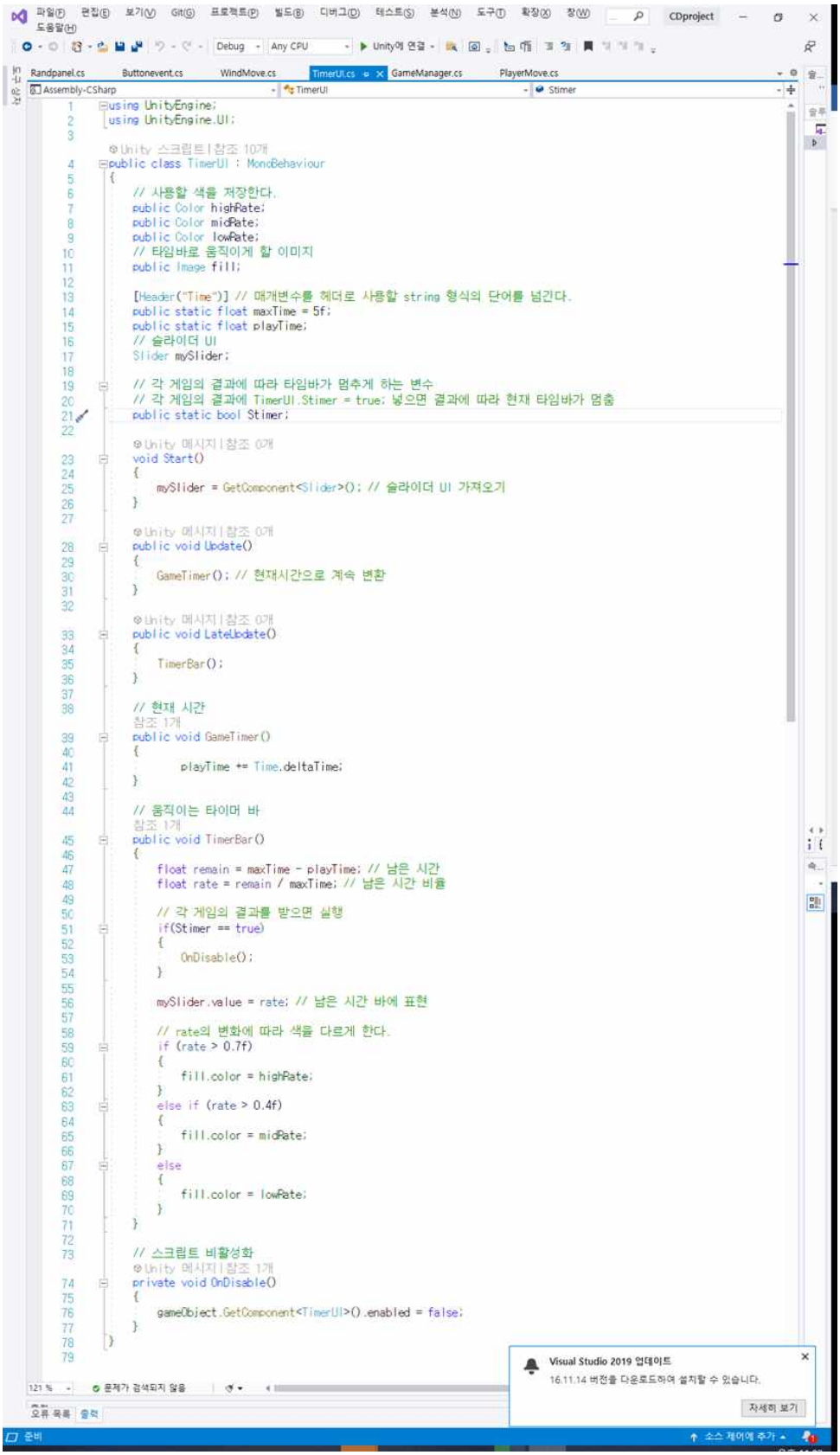
게임1의 기본 기능 작성 및 타이머 코드 수정

올바른탈출구의 표지판을 찾아 누르면 성공하는 게임을 제작.



```
public GameObject bt2;
public GameObject bt3;

private int random;

Unity 메시지 | 참조 0개
void Awake()
{
    // 랜덤을 이용해서 그림 위치 바꾸기
    random = UnityEngine.Random.Range(0, 3);

    // random으로 받은 수를 이용하여 각 그림 활성화
    switch (random)
    {
        case 0:
            bt1.gameObject.SetActive(true);
            break;
        case 1:
            bt2.gameObject.SetActive(true);
            break;
        case 2:
            bt3.gameObject.SetActive(true);
            break;
        default:
            Debug.Log("random에 없는 수 입니다.");
            break;
    }
    Debug.Log(random);
}
}
```

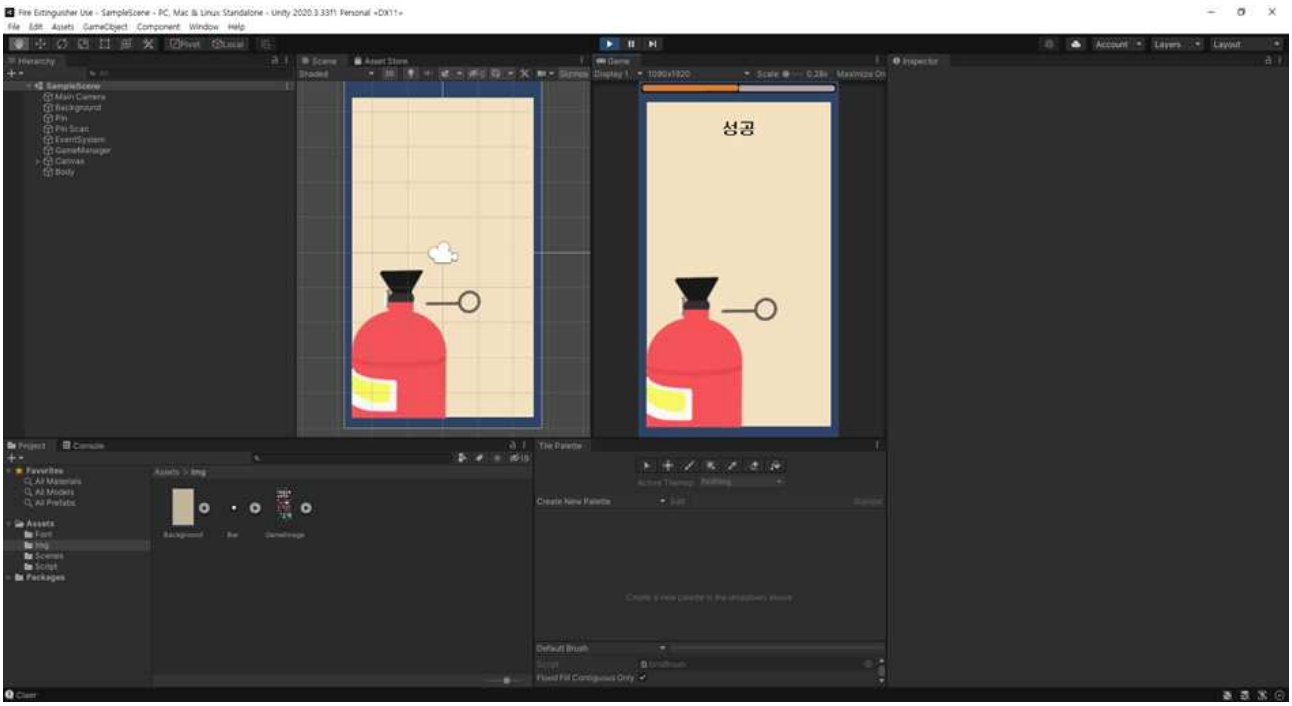
퀴즈지문을 랜덤 한 위치에 섞어주는 코드이다. switch (random)을 이용해 랜덤하게 위치를 바꿔 생성하는 코드



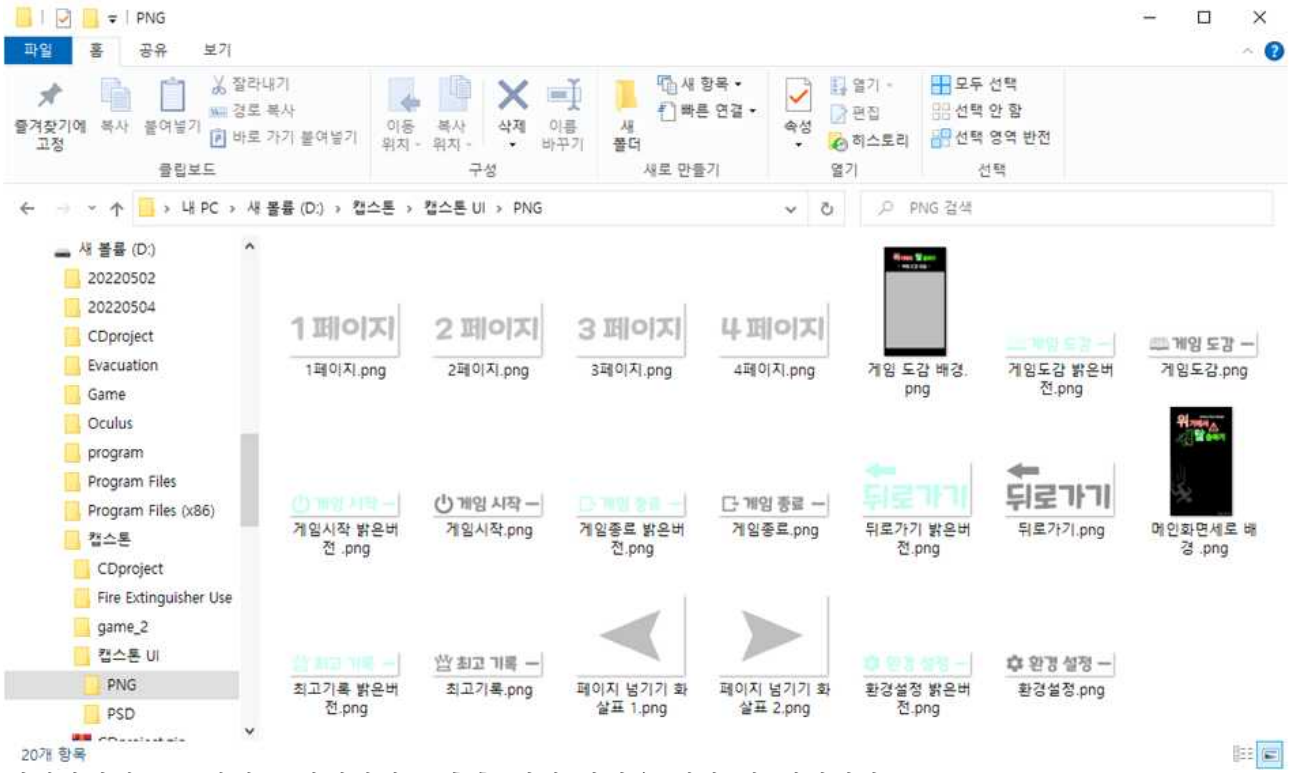
```
using UnityEngine;
using UnityEngine.UI;

Unity 스크립트 | 참조 10개
public class TimerUI : MonoBehaviour
{
    // 사용할 색을 저장한다.
    public Color highRate;
    public Color midRate;
    public Color lowRate;
    // 타임바로 움직이게 할 이미지
    public Image fill;

    [Header("Time")] // 매개변수를 헤더로 사용할 string 형식의 단어를 넘긴다.
    public static float maxTime = 5f;
    public static float playTime;
    // 슬라이더 UI
    Slider mySlider;

    // 각 게임의 결과에 따라 타임바가 멈추게 하는 변수
    // 각 게임의 결과에 TimerUI.Stimer = true; 넣으면 결과에 따라 현재 타임바가 멈춤
    public static bool Stimer;

    Unity 메시지 | 참조 0개
    void Start()
    {
        mySlider = GetComponent<Slider>(); // 슬라이더 UI 가져오기
    }

    Unity 메시지 | 참조 0개
    public void Update()
    {
        GameTimer(); // 현재시간으로 계속 변환
    }

    Unity 메시지 | 참조 0개
    public void LateUpdate()
    {
        TimerBar();
    }

    // 현재 시간
    참조 1개
    public void GameTimer()
    {
            playTime += Time.deltaTime;
    }

    // 움직이는 타이머 바
    참조 1개
    public void TimerBar()
    {
        float remain = maxTime - playTime; // 남은 시간
        float rate = remain / maxTime; // 남은 시간 비율

        // 각 게임의 결과를 받으면 실행
        if(Stimer == true)
        {
            OnDisable();
        }

        mySlider.value = rate; // 남은 시간 바에 표현

        // rate의 변화에 따라 색을 다르게 한다.
        if (rate > 0.7f)
        {
            fill.color = highRate;
        }
        else if (rate > 0.4f)
        {
            fill.color = midRate;
        }
        else
        {
            fill.color = lowRate;
        }
    }

    // 스크립트 비활성화
    Unity 메시지 | 참조 1개
    private void OnDisable()
    {
        gameObject.GetComponent<TimerUI>().enabled = false;
    }
}
```

TimerUI.Stimer = true;를 추가해 타임바를 멈추게 만든다

게임3 소화기 사용법 게임 제작 과정

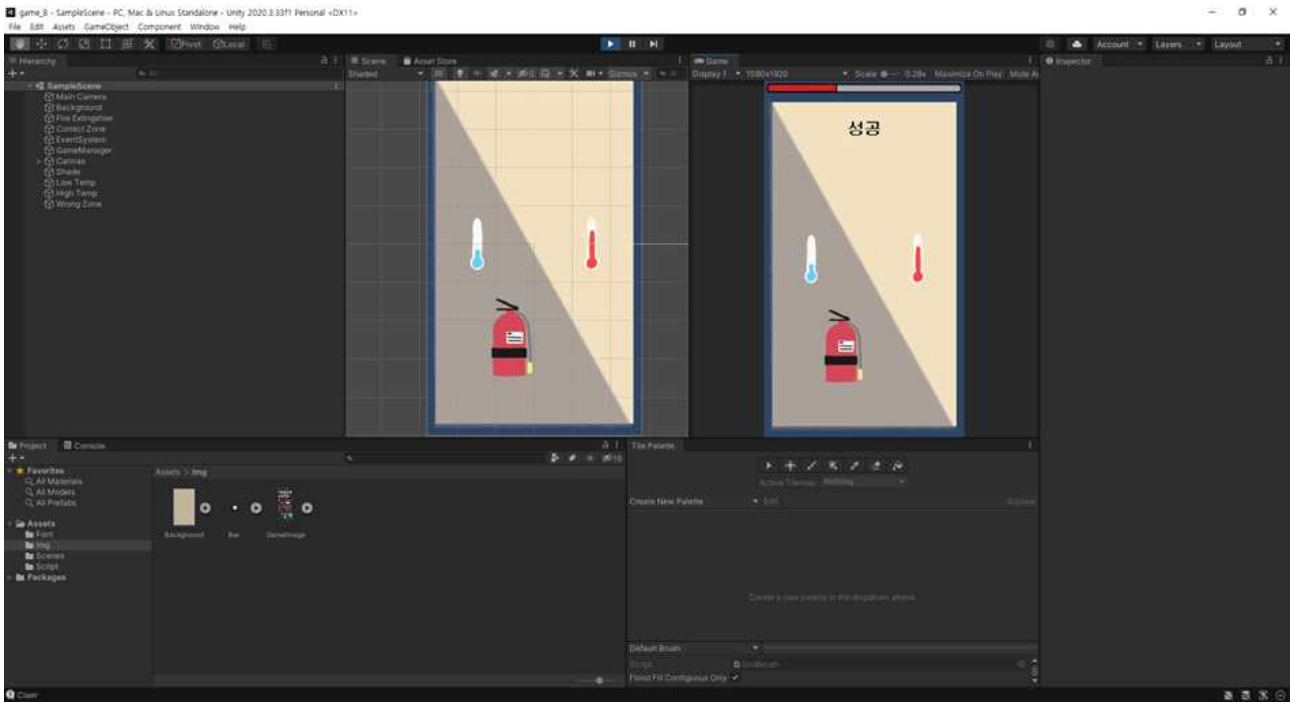


소화기의핀을 뽑으면 클리어 하는 기능까지 작성, 기본적인 코드는 14번 게임과 같다.
14번게임과의 차별점을 두기위해서 손잡이를 움켜지는 기능을 추가할 예정이다.
게임UI 및 메인 화면



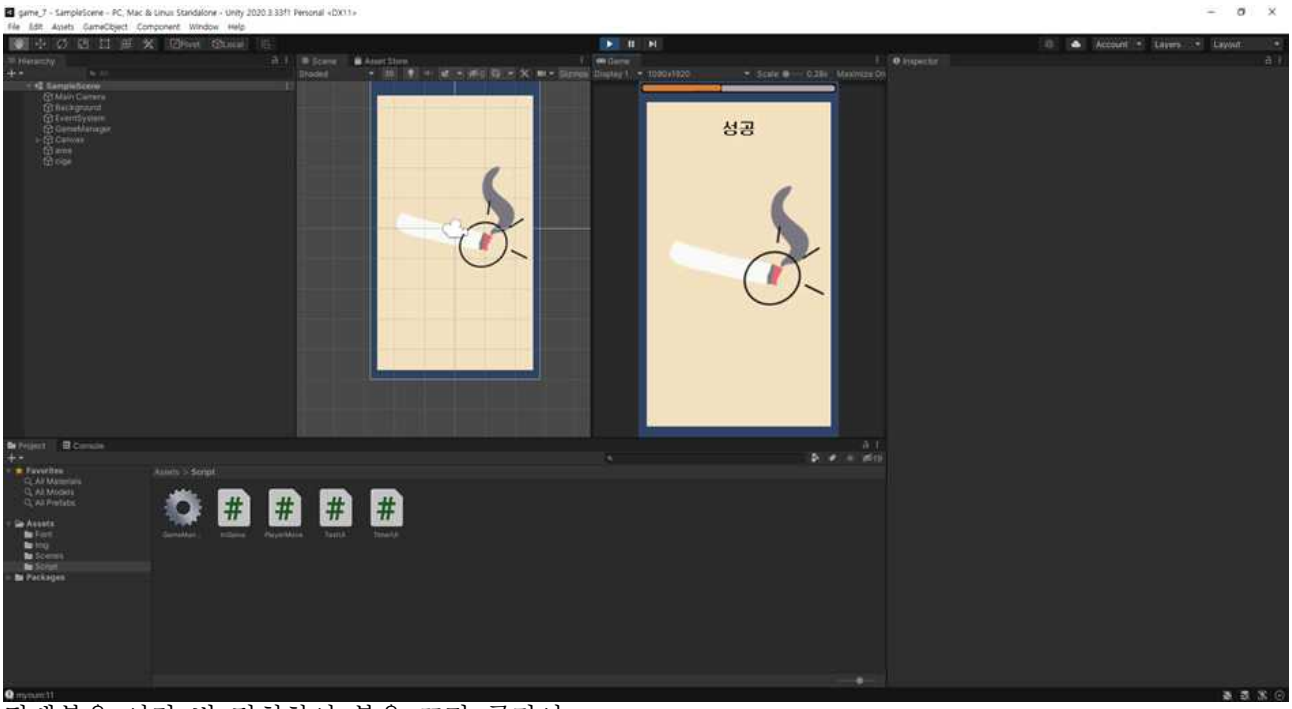
메인화면의PNG 파일로 작성하였고 추후 메인 화면을 제작 할 예정이다.

<12주차>
게임8 소화기 보관요령

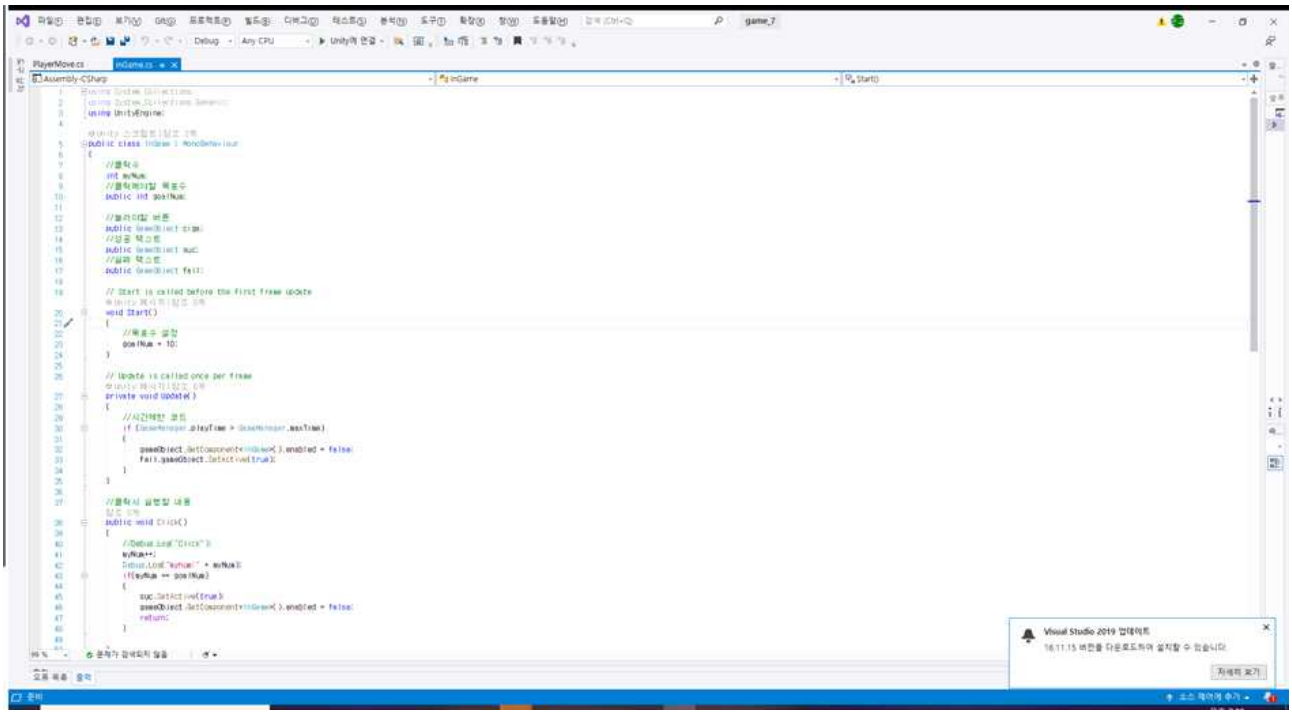


소화기를높은 온도 와 습기, 직사 광선을 피하여 배치하면 클리어
14번게임의 코드를 기반으로 제작하였음

게임7번 화재 초기진압

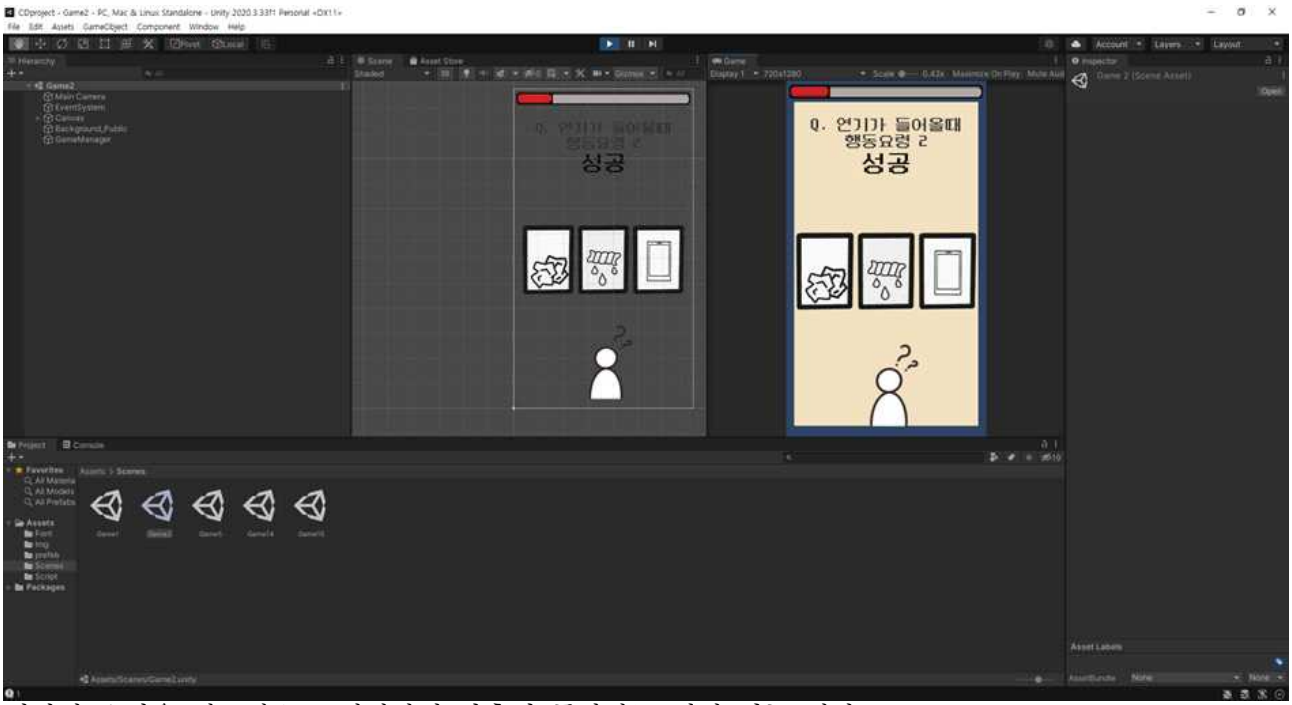


담배불을 여러 번 터치하여 불을 끄면 클리어
클리커게임을 기반으로 제작하였음



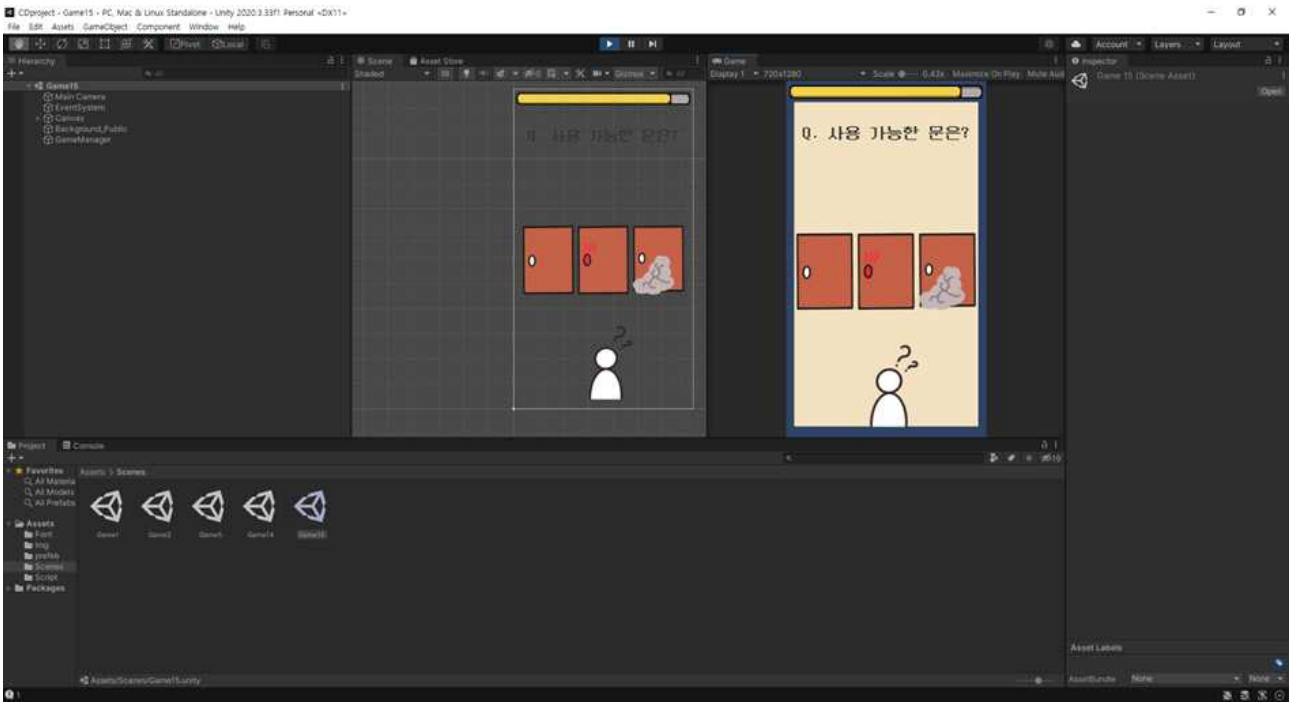
담배불의 위치를 클릭하여 지정된 수만큼 클릭하면 게임을 클리어하는 코드

게임2 대피 시 요령



대피시 요령을 퀴즈식으로 제시하여 맞추면 통과하는 게임 기능 제작.
게임1의 코드를 기반으로 제작

게임15 문고리 온도 체크



대피상황 때에 문 너머의 상황을 파악하는 방법을 이해하여 올바른 문을 고르면 통과하는 게임 기능 제작

게임1의 코드를 기반으로 제작

게임UI – 게임 도감 예시

게임도감 UI 예시 제작

<13주차>

```
using System.Collections;
using System;

using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;

참조 1개
public class sceneLoad : MonoBehaviour
{
    public static sceneLoad instance = null;

    private int sceneNum;
    private List<int> randomList;

    public static Action Load;

    //public bool check = true;
    참조 0개
    private void Awake()
    {
        if (instance == null)
            instance = this;
        else if (instance != this)
            Destroy(gameObject);
        DontDestroyOnLoad(gameObject);


        //randomList에 값 넣어주기
        ResetList();
    }

    참조 0개
    public void Start()
    {
        Load = () => { SceneLoad(); };
    }
    참조 0개
    private void Update()
    {



    }

    //reset List
    참조 2개
    private void ResetList()
    {
        //random하게 불러올 씬 넘버
        randomList = new List<int>() {  1,2, 3, 4 ,5,6,7,8,9,10,11};
    }

    //chk current sceneNum
    참조 0개
    private int ChkSceneNum()
    {
        sceneNum = SceneManager.GetActiveScene().buildIndex;
        return sceneNum;
    }

    //shuffle
    참조 1개
    public void Mix()
    {
        Debug.Log("Mix");
        List<int> list = new List<int>();
        int count = randomList.Count;
        for (int i = 0; i < count; i++)
        {
            int rand = UnityEngine.Random.Range(0, randomList.Count);
            list.Add(randomList[rand]);
            randomList.RemoveAt(rand);
        }
        randomList = list;
    }

    참조 1개
    public void SceneLoad()
    {

        if (randomList.Count == 0 || randomList == null)
        {
            SceneManager.LoadScene(12);
            //SceneManager.LoadScene(0);
            ResetList();
            SceneManager.LoadScene(randomList[0]);
            Debug.Log(randomList.Count);
            randomList.RemoveAt(0);
        }
        else
        {
            SceneManager.LoadScene(12);


            Mix();
            SceneManager.LoadScene(randomList[0]);
            Debug.Log(randomList.Count);
            randomList.RemoveAt(0);
        }
    }

}
```

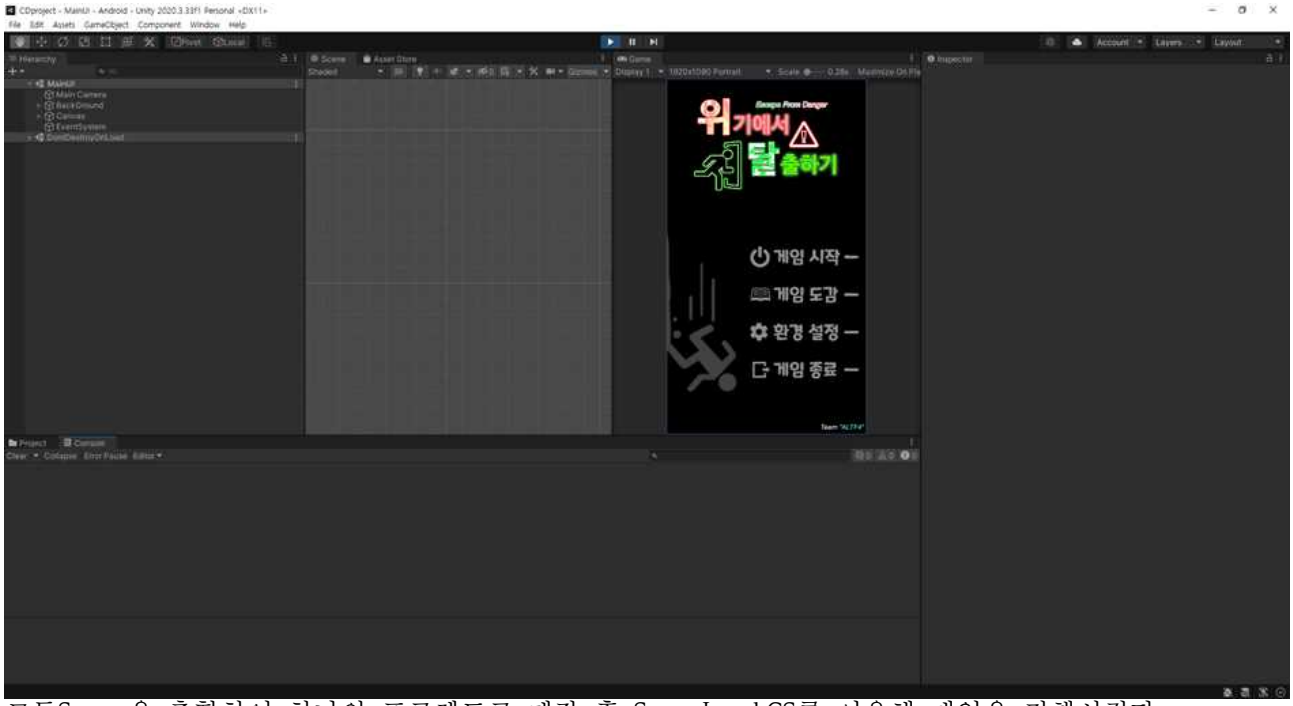
<SceneLoad.CS>

랜덤한Scene 로드를 위한 코드이다. 게임 플레이 버튼을 눌렀을 때 랜덤하게 돌릴 Scene을 리스트에 넣

고 섞어서 순차적으로 실행시킨 후 모든 리스트를 작동하였으면 다시 리스트를 초기화 시켜 반복하는 코드이다.

게임데모 제작

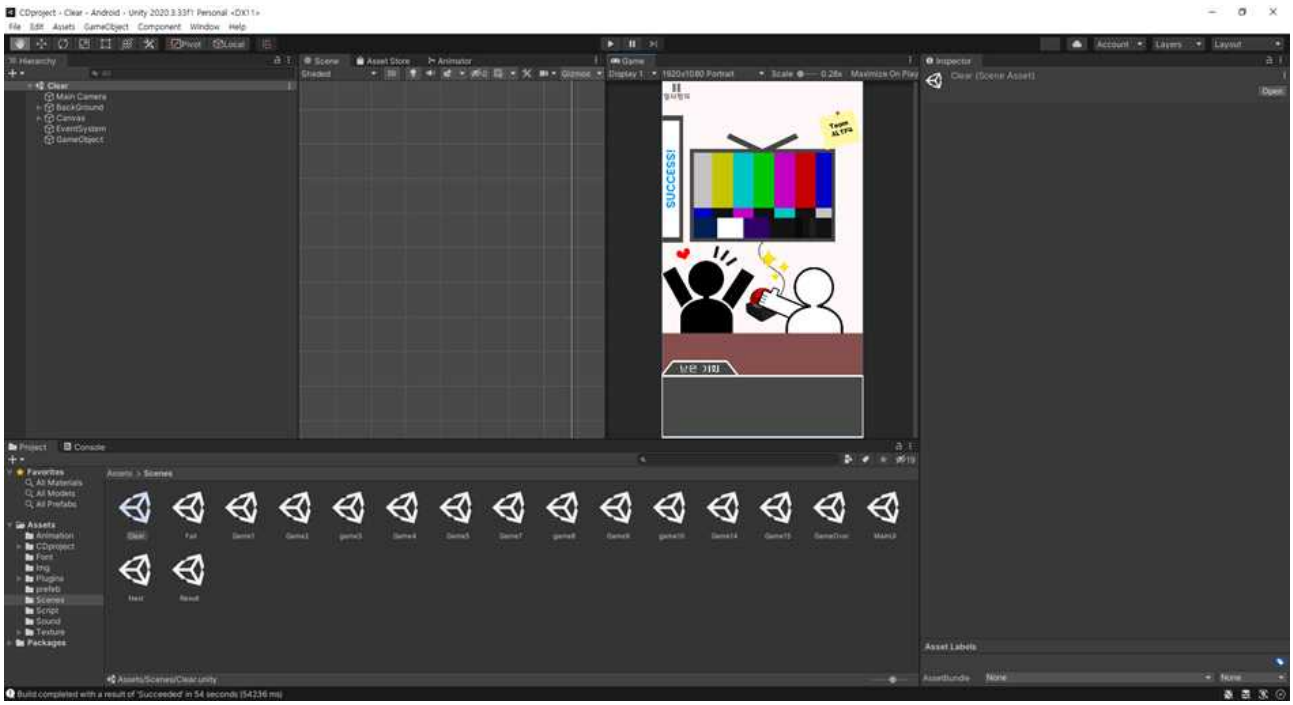
13주차의주된 목표는 게임 데모 제작이다.



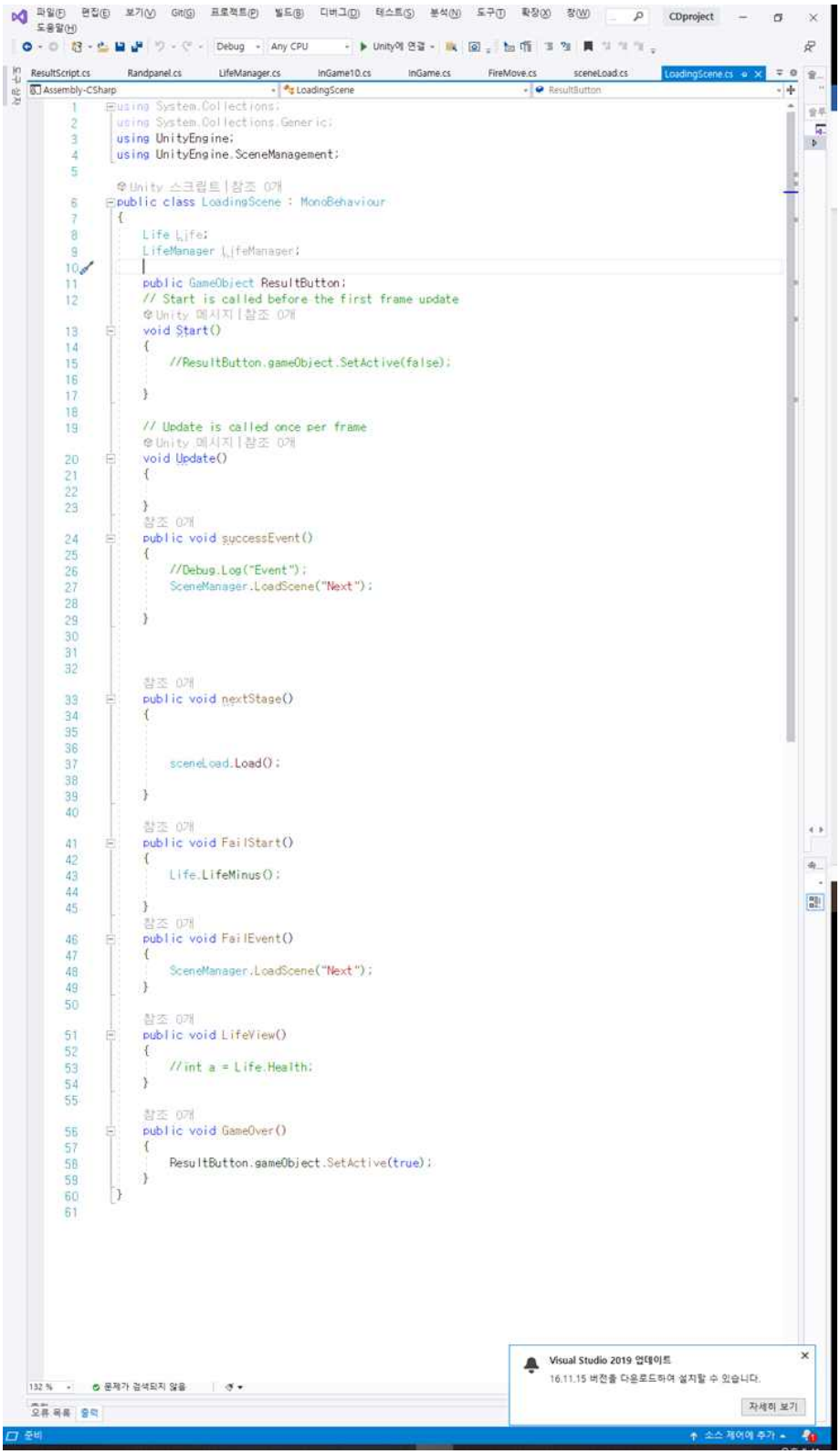
모든Scene을 총합하여 하나의 프로젝트로 제작 후 SceneLoad.CS를 이용해 게임을 진행시켰다.
추후작업해야 할 내용은 결과 창, 스테이지가 넘어갈 때 보이는 로딩 창,체력과 게임 속도 조정이 필요하다.

<14주차>

게임성공, 실패 여부 창 및 로딩 창 제작



로딩창과성공여부 확인창을 제작하였다.



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

Unity 스크립트 | 참조 0개
public class LoadingScene : MonoBehaviour
{
    Life Life;
    LifeManager LifeManager;

    public GameObject ResultButton;
    // Start is called before the first frame update
    Unity 메시지 | 참조 0개
    void Start()
    {
        //ResultButton.gameObject.SetActive(false);

    }

    // Update is called once per frame
    Unity 메시지 | 참조 0개
    void Update()
    {

    }
    참조 0개
    public void successEvent()
    {
        //Debug.Log("Event");
        SceneManager.LoadScene("Next");

    }



    참조 0개
    public void nextStage()
    {


        sceneLoad.Load();

    }

    참조 0개
    public void FailStart()
    {
        Life.LifeMinus();

    }
    참조 0개
    public void FailEvent()
    {
        SceneManager.LoadScene("Next");
    }

    참조 0개
    public void LifeView()
    {
        //int a = Life.Health;
    }

    참조 0개
    public void GameOver()
    {
        ResultButton.gameObject.SetActive(true);
    }
}
```

<LoadingScene.cs>

성공및 로딩창의 애니메이션이 끝나면 다음Scene으로 넘어가는 코드를 작성하였다.

애니메이션끝에 이벤트를 넣어 작동시킨다.

<Life.cs>

플레이어의체력을 관리하는 스크립트, 실패Scene으로 갔을 시 라이프 하나를 감소하며 게임 진행 라이프가0가 되었을 시 게임오버 Scene으로 이동시킨다.

또한메인 메뉴로 가면 체력변수를 모두 초기화 시킨다.

7. 연구 결과 및 고찰

메인 화면에서 게임 시작을 눌렀을 시, 게임이 실행되고 모든 라이프가 없어졌을 시 게임을 종료하고 결과를 출력한다.

또한 도감 메뉴를 추가하여 게임에 담겨있는 안전 지식의 관한 내용과 게임 진행 방식에대해 설명하였다.

향후 개선 사항으로는더 많은 게임을 추가하거나 데이터베이스를 활용하여 결과를 저장하는 것이 좋을것으로 보인다.

8. 예산 집행 현황

구분	일자	사용 내역	금액
합계			

---

[1] 김영현:"교육용 게임을 활용한 초등학교 게이미피케이션" Journal of KoreaGame Society 2020 Oct; 20(5): 21-30
[2] 교육용게임 - 위키백과, 우리 모두의백과사전 (wikipedia.org)
[3]김영현: "교육용 게임을 활용한 초등학교 게이미피케이션" Journal of Korea Game Society 2020 Oct; 20(5): 21-30
방재와 보험143호- 특집3 - 소방안전교육의 강화 필요성[4]
[5]e-나라지표, 화재 발생 현황2011 - 2020

캡스톤디자인 지도 실적 보고서(지도교수용)

캡스톤디자인 교과목명 (교과목코드)	캡스톤디자인2(기업연계프로젝트)			
캡스톤디자인 과제명	교육용 미니게임			
지도학생				
지도개요	교육용 미니게임 개발에 조언 및 지도			
지도교수	소속	디지털콘텐츠공학과	성명	
세부 지도내용	1. 주제 선정 및 주제 관련 사례 조사 2. 개발 초안 작성 3.게임 개발에 대한 조언 및 지도 4. 게임 그래픽 디자인에 대한 조언 및 지도 5. 결과물 도출에 대한 조언 및 지도			
수행기간	2022 년 03월 08일 ~ 2022 년 06월 12일			

위와 같이 캡스톤디자인(과제명)의 실적 보고서를 제출합니다.

2022 년 06 월 12 일

지도교수 : (인)

원광대학교 LINC 3.0 사업단장 귀하

캡스톤디자인 산학연계 교육 실적보고서

캡스톤디자인 교과목명 (교과목코드)	캡스톤디자인2(기업연계프로젝트)			
캡스톤디자인 과제명	게임용 미니게임			
교육기간	2022 년 03월 08일 ~ 2022 년 06월 12일			
교육개요	교육용 미니게임 개발			
기업체전문가	소속	(주)펀웨이브	성명	
교육내용	1. 주제 선정 및 주제 관련 사례 조사 2. 개발 초안 작성 3.게임 개발에 대한 조언 및 지도 4. 게임 그래픽 디자인에 대한 조언 및 지도 5. 결과물 도출에 대한 조언 및 지도			
교육운영결과	교육용 미니게임 개발에 대한 조언을 전달 하여 게임의 퀄리티 향상			

위와 같이 캡스톤디자인(과제명) 산학연계 교육 실적보고서를 제출합니다.

2022 년 06월 12 일

기업체전문가 : (인)

원광대학교 LINC 3.0 사업단장 귀하