

## 프로젝트 최종 결과 보고서

1815060 문 정현

### 1. 프로젝트 계획서 요약


Title	Adventure of chicken
Genre	3D, Action, Adventure
Platform	Window, Mac, Linux
Description	<ul style="list-style-type: none"><li>• 3개의 스테이지로 구성되어 있습니다.</li><li>• 플레이어의 기본 목숨은 5개로 각 스테이지를 추락할 때 목숨 1개가 깎이고 남은 목숨이 없는 경우 게임오버가 됩니다.</li><li>• 1스테이지는 7개의 장애물 중 랜덤하게 선택된 4개의 장애물로 구성된 스테이지입니다.</li><li>• 2스테이지는 랜덤하게 생성된 미로가 배치되어 플레이어는 움직이는 발판을 타고 미로를 탈출하는 스테이지입니다.</li><li>• 3스테이지는 떨어지는 돌들을 피해 보스를 총으로 물리쳐야 게임에서 승리하는 최종 스테이지입니다.</li><li>• 스테이지 간 이동은 포탈을 통해 이뤄지고 매 스테이지를 통과하면 자동으로 세이브 구간이 갱신됩니다.</li></ul>

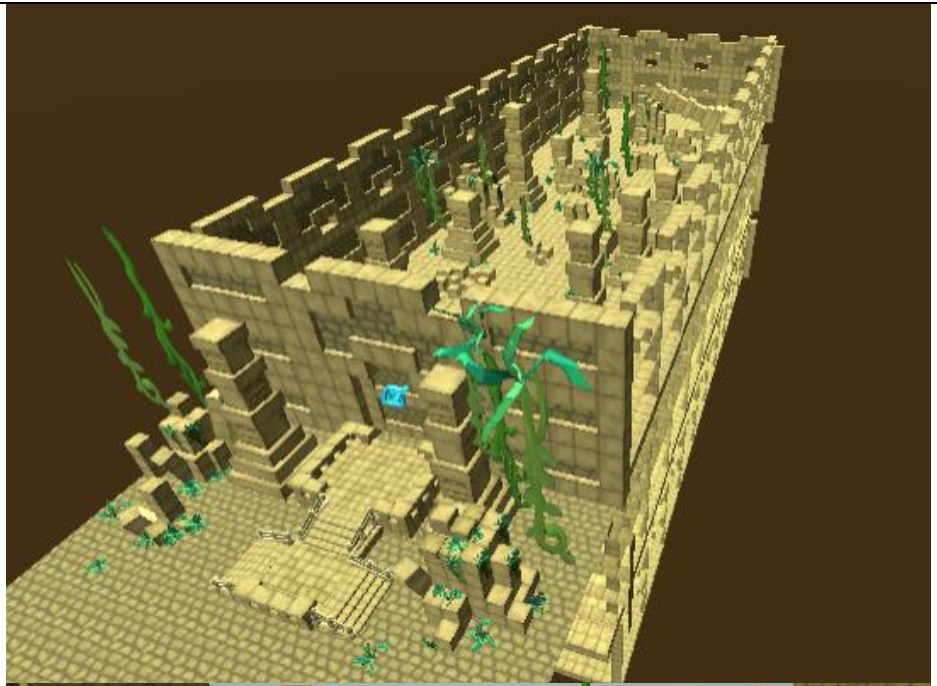
### 2. 중간 결과보고서 대비 추가 구현 현황 비교

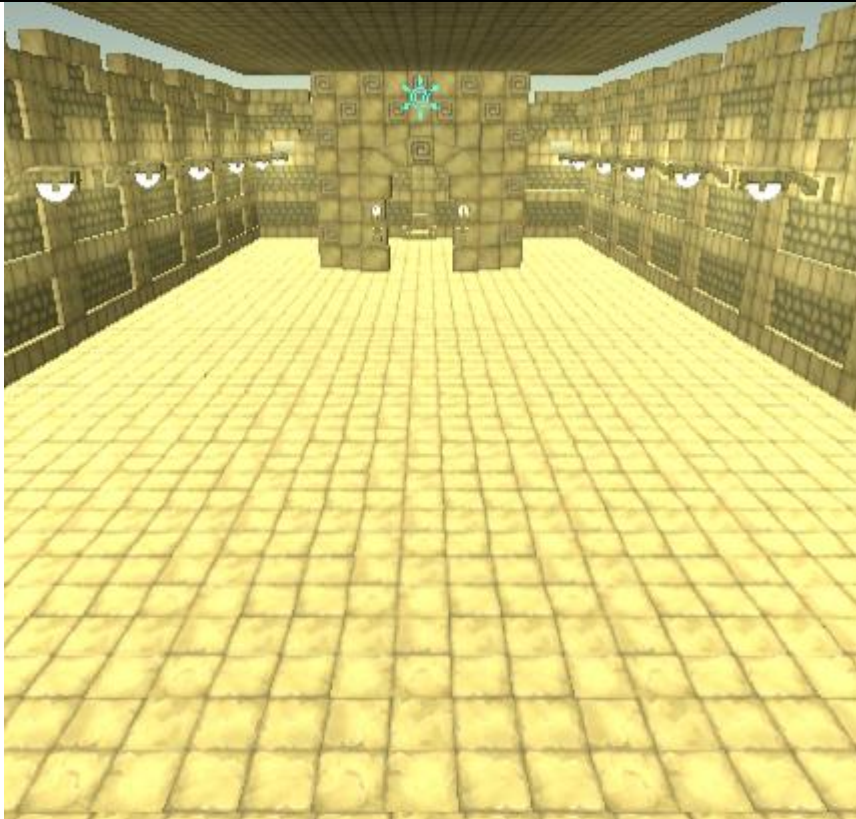
Camera	Cinemachine 컴포넌트 추가로 플레이어에 대한 3인칭 카메라
Chracter	스킨과 애니메이션

Background	사원의 느낌을 주는 스킨 적용
UI	플레이어의 체력과 플레이 타임을 기록하는 타이머 총알의 갯수를 확인할 수 있는 UI 추가 설명 문구 UI
Menu	게임 종료와 게임 클리어 시 돌아가게 되는 화면과 게임 시작 버튼
Stage2	랜덤으로 미로가 생성, 움직이는 발판과 일정 거리 내 플레이어를 감지하고 총알을 발사하는 적 생성되는 스테이지
Stage3	랜덤으로 돌과 아이템(체력, 총알)이 떨어지는 맵으로 최종 보스를 총으로 싸우는 마지막 스테이지
Sound	게임 효과음

### 3. 최종 구현 내용의 기능별 실행 화면 및 설명

Menu	
게임 타이틀 화면과 시작 버튼이 있어 버튼을 누르면 게임이 시작됩니다.	
	
Background	
유적 속 사원을 탐험하는 분위기를 위한 배경으로 연출되었습니다.	

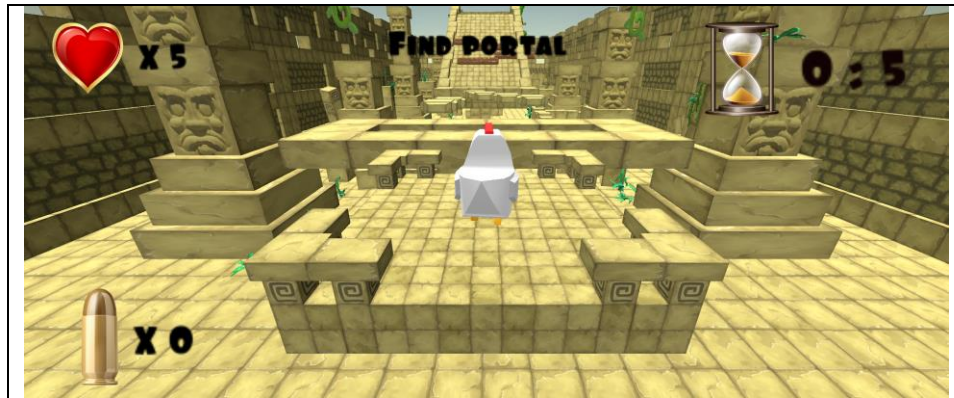




## UI

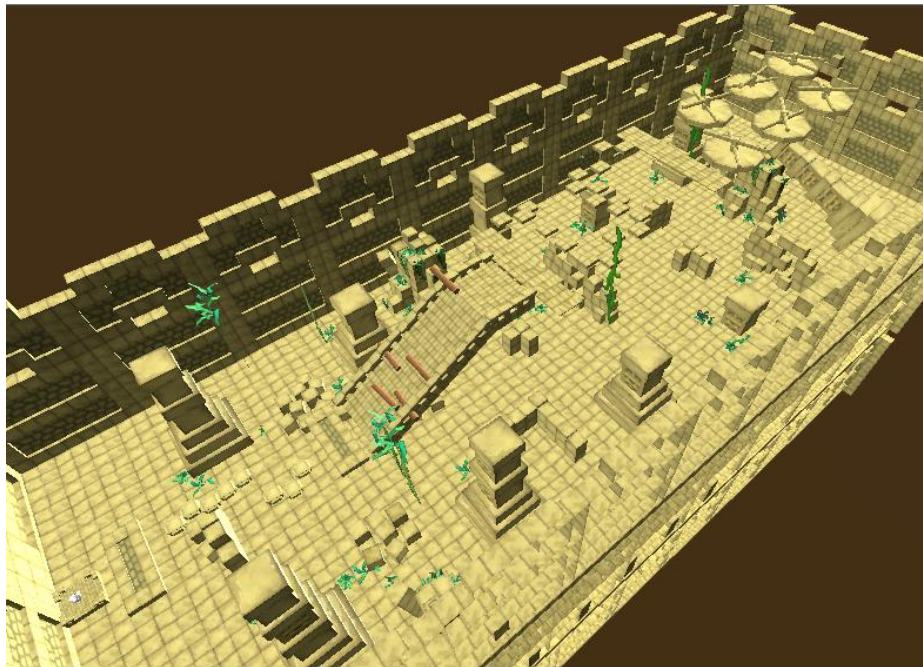
목숨의 개수, 스테이지 목표, 경과 시간 , 총알의 개수를 보여 주기 위한 UI가 있습니다.

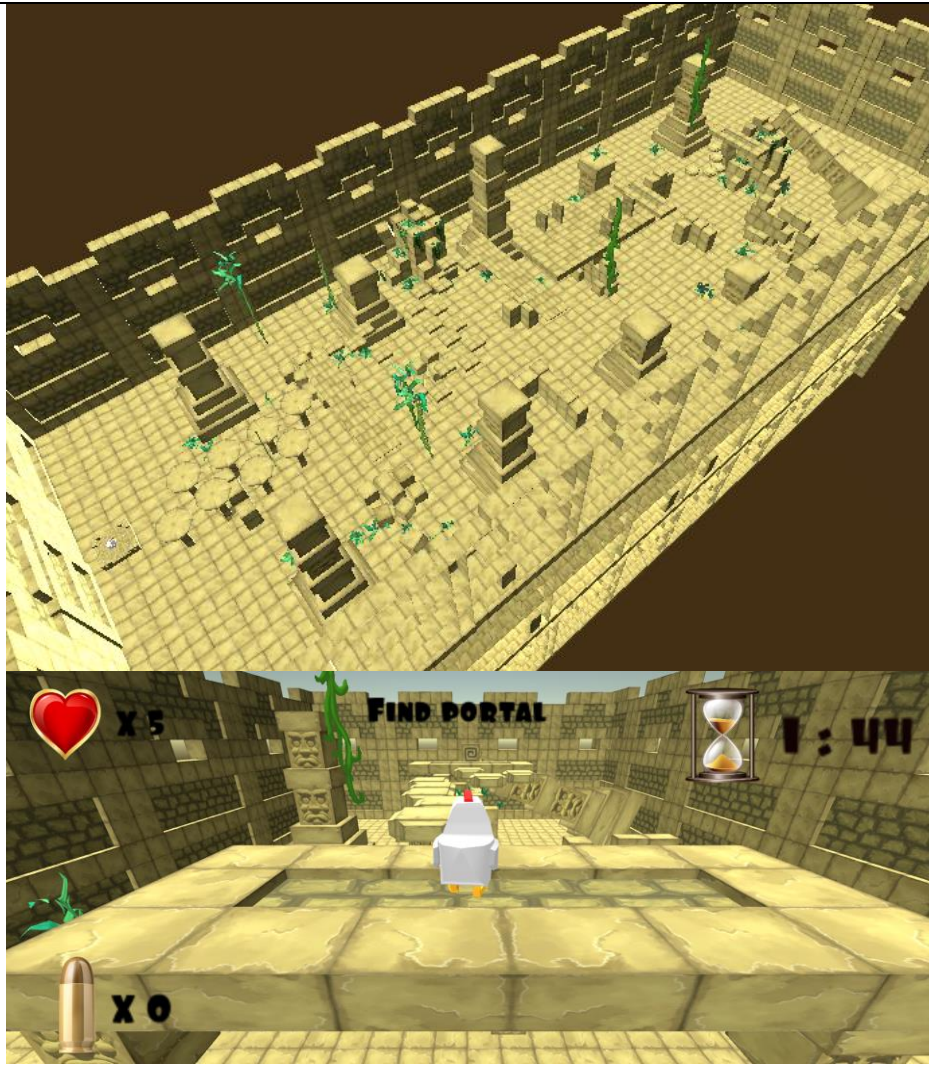




### Stage1

7개의 장애물 중 4개의 장애물이 랜덤으로 배치되 스테이지 마지막 부분에는 다음 스테이지로 가기 위한 포탈이 있습니다.

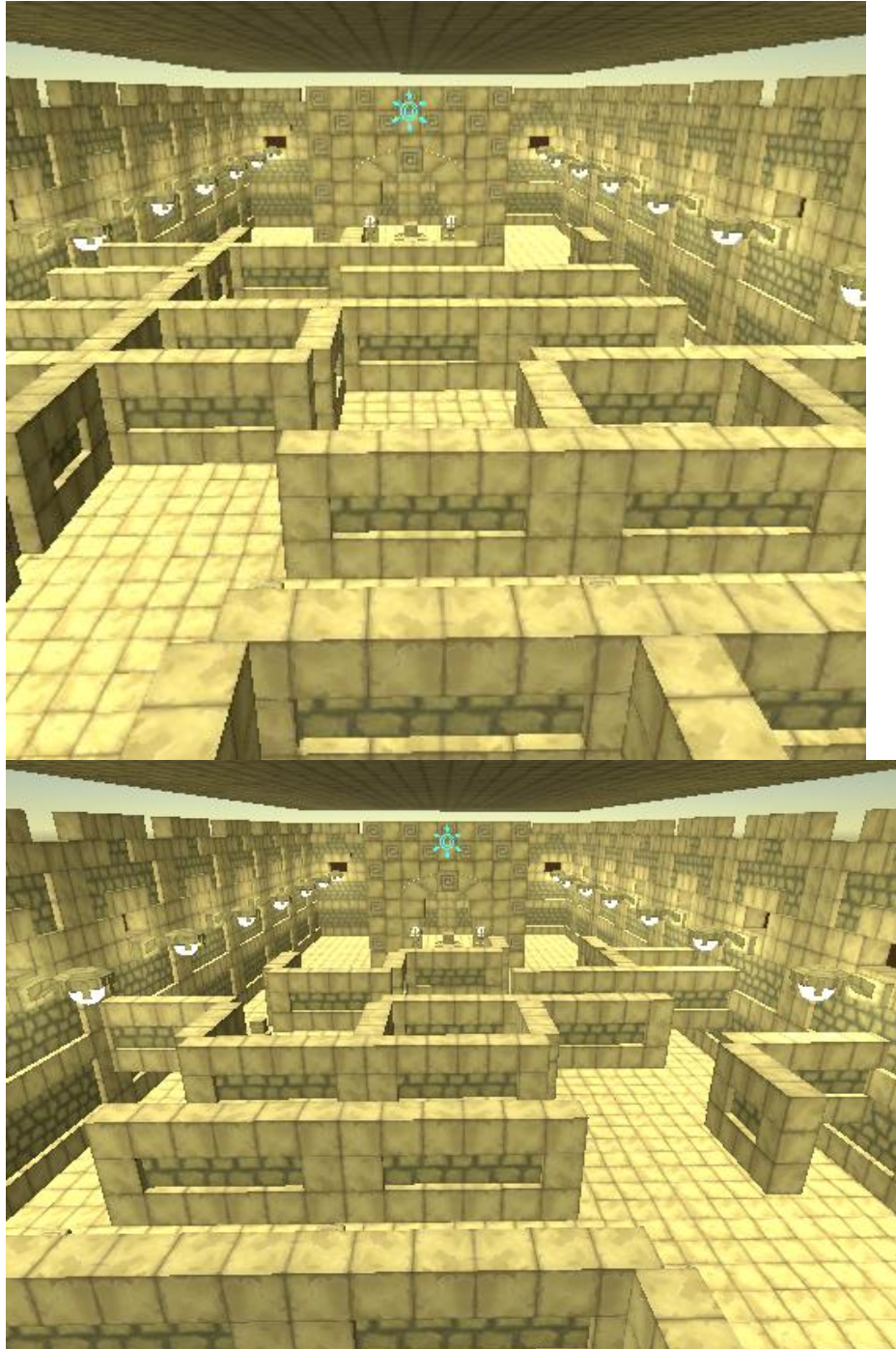


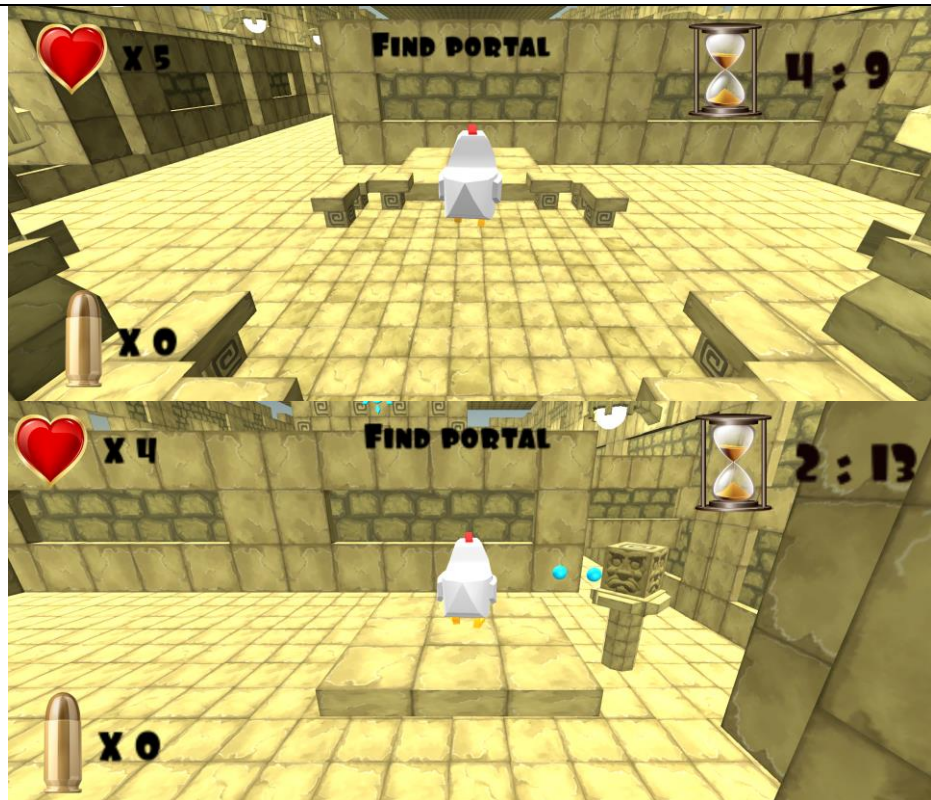


Stage2



랜덤으로 생성된 미로와 플레이어가 있는 방향에 따라 움직이는 발판이 있고 일정거리 내 플레이어를 감지할 경우 플레이어 방향으로 구체를 쏘는 적을 배치하여 플레이어는 다음 스테이지로 가기 위해 움직이는 발판을 조정해 적들을 피하며 떨어지지 않고 가는 스테이지로 구성하였습니다.

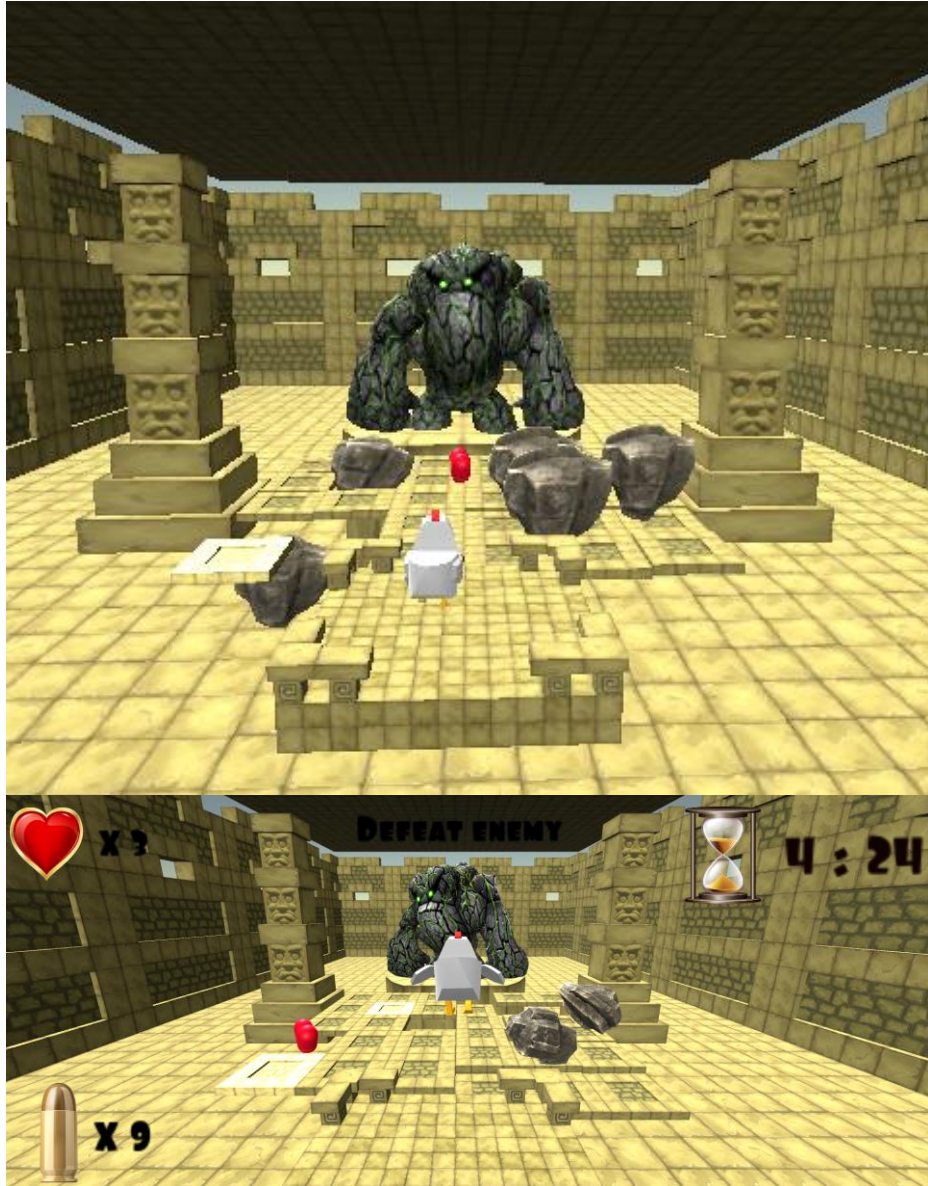




Stage3



일정 시간마다 체력 회복 아이템과 총알 아이템, 돌이 떨어지고 돌과 충돌한 발판은 떨어져 사라진 뒤 일정 시간 뒤 다시 리스폰되는 구조물로 플레이어는 앞의 적을 총으로 물리치는 최종 스테이지로 구현됐습니다. 적이 죽으면 게임 승리 문구와 함께 메뉴로 이동합니다.





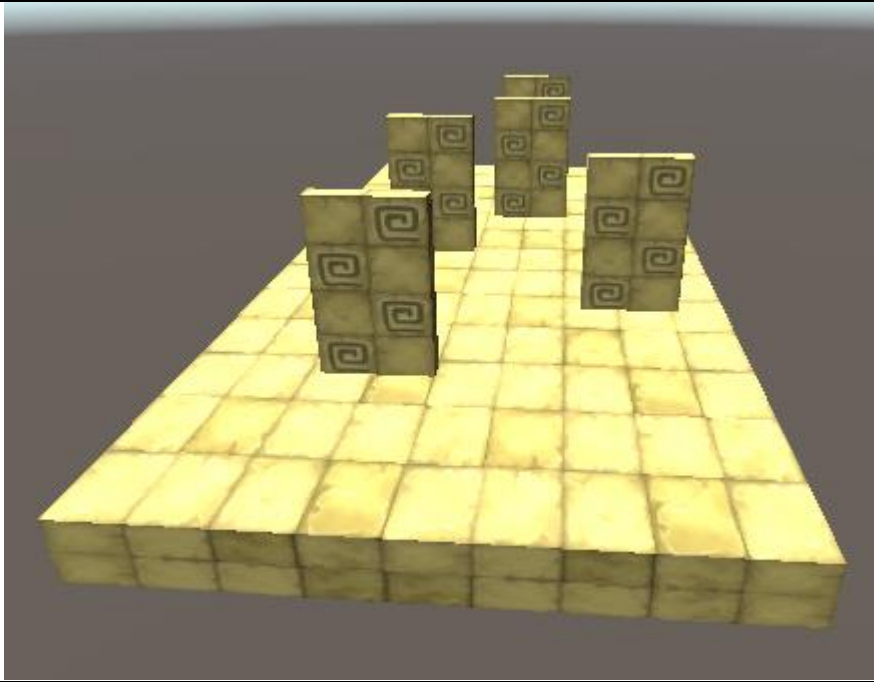
### Stage1 #1

각 원기둥의 속도가 랜덤으로 돌아가며 플레이어의 진행을 어렵게 하도록 구현했습니다.



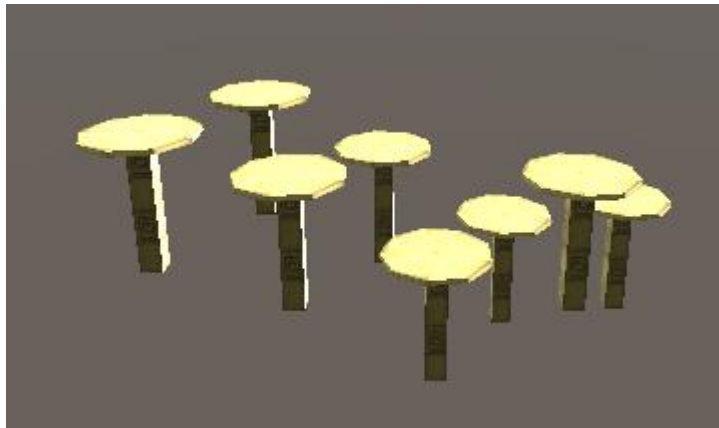
### Stage1 #2

랜덤한 위치에 사각 구조물이 앞으로 다가와 플레이어의 진행을 어렵게 하도록 구현했습니다.



Stage1 #3

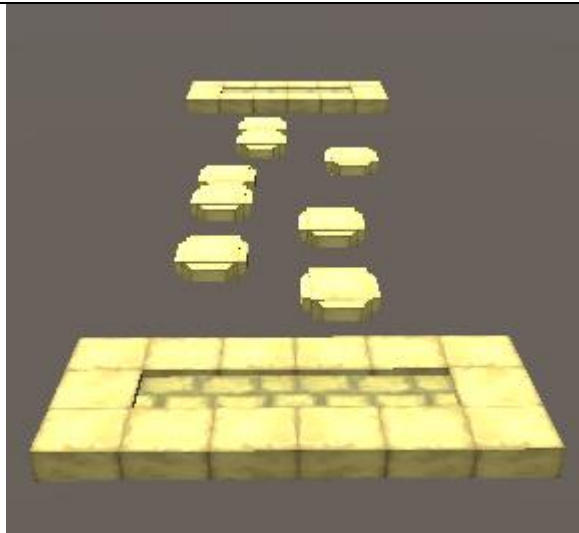
랜덤한 속도로 원판이 위아래로 이동하여 플레이어의 진행을 어렵게 하도록 구현했습니다.



Stage1 #4

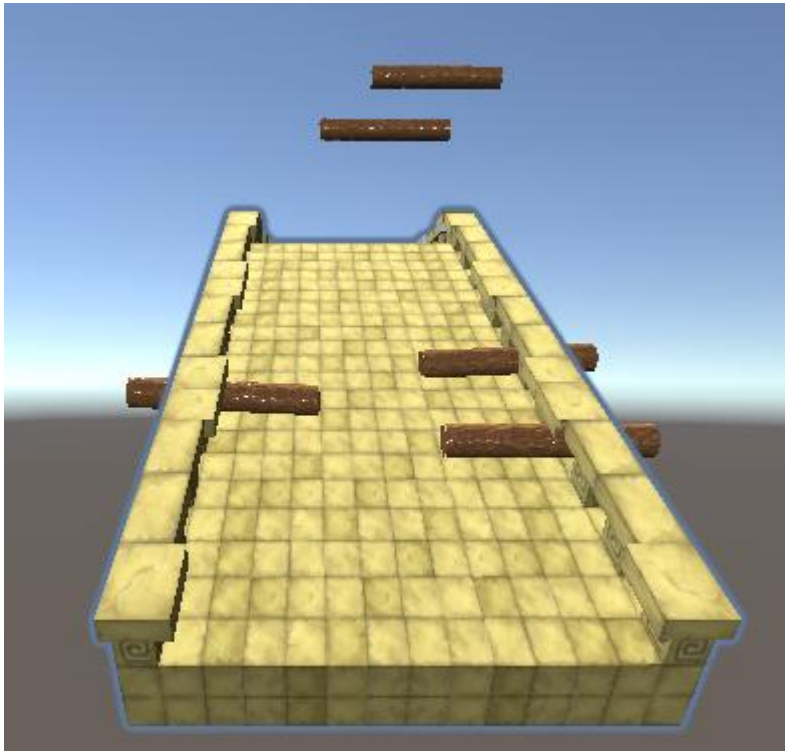
움직이는 발판이 이동하여 플레이어가 발판을 밟고 앞으로 나아가게 하여 플레이어의 진행을 어렵게 하도록 구현했습니다.





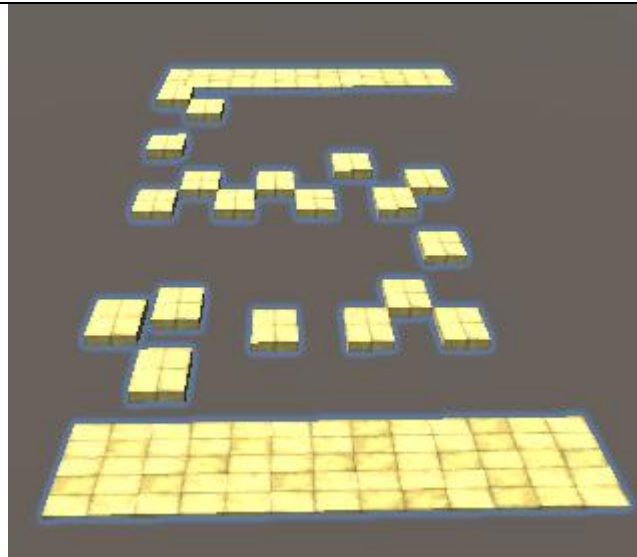
Stage1 #5

일정한 속도로 통나무가 내려와 플레이어를 방해하는 경사 구조물로 플레이어의 진행을 어렵게 하도록 구현했습니다.



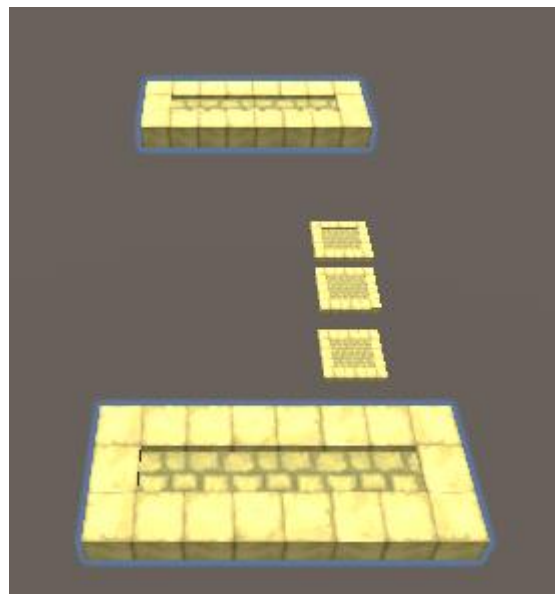
Stage1 #6

일정 시간을 간격으로 발판이 생기다 사라지는 것을 방해하는 구조물로 플레이어의 진행을 어렵게 하도록 구현했습니다.



Stage1 #7

발판이 앞으로 생기면서 사라지는 형태로 플레이어의 진행을 어렵게 하도록 구현했습니다.



#### 4. 사용된 외부 에셋 및 자체 구현 에셋 현황

- 외부 에셋

BackGround	<p>Cartoon Temple Building Kit Lite  <a href="https://assetstore.unity.com/packages/3d/environments/dungeons/cartoon-temple-building-kit-lite-110397">https://assetstore.unity.com/packages/3d/environments/dungeons/c</a>  <a href="https://assetstore.unity.com/packages/3d/environments/dungeons/cartoon-temple-building-kit-lite-110397">aroon-temple-building-kit-lite-110397</a></p> <p>Rock  <a href="https://assetstore.unity.com/packages/3d/props/exterior/rock-package-118182">https://assetstore.unity.com/packages/3d/props/exterior/rock-</a>  <a href="https://assetstore.unity.com/packages/3d/props/exterior/rock-package-118182">package-118182</a></p>
Player Chracter & Animation	<p>Meshtint Free Chicken Mega Toon Series  <a href="https://assetstore.unity.com/packages/3d/characters/animals/meshtint-free-chicken-mega-toon-series-151842">https://assetstore.unity.com/packages/3d/characters/animals/meshti</a>  <a href="https://assetstore.unity.com/packages/3d/characters/animals/meshtint-free-chicken-mega-toon-series-151842">nt-free-chicken-mega-toon-series-151842</a></p>
Boss Character & Animation	<p>Mini Legion Rock Golem PBR HP Polyart  <a href="https://assetstore.unity.com/packages/3d/characters/humanoids/fantasy/mini-legion-rock-golem-pbr-hp-polyart-94707">https://assetstore.unity.com/packages/3d/characters/humanoids/fan</a>  <a href="https://assetstore.unity.com/packages/3d/characters/humanoids/fantasy/mini-legion-rock-golem-pbr-hp-polyart-94707">tasy/mini-legion-rock-golem-pbr-hp-polyart-94707</a></p>
B.G.M	<p>The Legend of Zelda – Skyward Sword OST  (Lanayru Mining Facility, Ancient Cistern, Eldin Eruption, Koloktos Appears)</p>
Sound Effect	<p>New Super Mario Bros. Wii (Game Over)</p>
	<p>Super Mario Series - Super Smash Bros. Ultimate  (Victory!)</p>
	<p>Spelunky HD OST (sound effect)</p>
UI Image	<p>Heart  <a href="https://en.wikipedia.org/wiki/Heart_symbol#/media/File:Heart_coraz%C3%B3n.svg">https://en.wikipedia.org/wiki/Heart_symbol#/media/File:</a>  <a href="https://en.wikipedia.org/wiki/Heart_symbol#/media/File:Heart_coraz%C3%B3n.svg">Heart_coraz%C3%B3n.svg</a></p> <p>bullet  <a href="https://webiconspng.com/icon/73848">https://webiconspng.com/icon/73848</a></p>



	Clock <a href="https://toppng.com/hourglass-png-PNG-free-PNG-Images_138150">https://toppng.com/hourglass-png-PNG-free-PNG-Images_138150</a>
UI Fonts	20 Logo Templates with customizable PSD vector sources <a href="https://assetstore.unity.com/packages/2d/gui/icons/20-logo-templates-with-customizable-psd-vector-sources-174999">https://assetstore.unity.com/packages/2d/gui/icons/20-logo-templates-with-customizable-psd-vector-sources-174999</a>

- 자체 구현 에셋

Stage1	랜덤으로 선택된 구조물 1~7의 개별 동작 스크립트 구조물 배치 스크립트 포탈 스크립트
Stage2	랜덤 미로 및 적 생성 스크립트 적의 개별 동작 스크립트
Stage3	적 행동 및 애니메이션 제어 스크립트 구조물 자동 생성 스크립트 하트, 총알 아이템
Player	플레이어 행동 및 애니메이션 제어 스크립트
Main Menu	게임 시작 제어 스크립트
UI / Sound	UI 및 사운드 제어 스크립트

## 5. 주요 스크립트 코드( 메인게임, 스테이지1, 2, 3, 플레이어, 보스)

MainGame.cs

```

public class MainGame : MonoBehaviour
{
    //노래를 제어하기 위한 변수
    public AudioSource stage1,stage2,stage3,menu;
    public AudioSource gameOver,win;
    public AudioSource die, hit, attack;
    public AudioSource heart,fire,enemy;
    public static Vector3 savePoint;

    //스테이지와 메뉴 ,플레이어를 제어하기 위한 변수
    GameObject Stage1, Stage2, Stage3, Player1,Menu;
    void Start() { |
        Stage1 = GameObject.FindWithTag("stage1");
        Stage2 = GameObject.FindWithTag("stage2");
        Stage3 = GameObject.FindWithTag("stage3");
        Player1 = GameObject.FindWithTag("Player");
        Menu = GameObject.FindWithTag("menu");
        Stage1.SetActive(false);
        Stage2.SetActive(false);
        Stage3.SetActive(false);
        Player1.SetActive(false);
        music(11);
    }

    //메뉴에서 게임 시작 버튼을 누르면 실행되는 함수
    //게임스테이지가 생성되고 플레이어가 스테이지1에서
    //게임이 시작됩니다.
    public void gameStart()
    {
        GameObject.FindWithTag("MainGame").GetComponent<MainGame>().music(0);
        Stage1.SetActive(true);
        Stage2.SetActive(true);
        Stage3.SetActive(true);
        Player1.SetActive(true);
        Menu.SetActive(false);
    }
}

```

//게임이 끝나는 경우 실행되는 함수  
 //게임 설정을 디폴트로 바꾸고 스테이지를 비활성화  
 //한 뒤 노래는 메뉴의 노래를 실행합니다.

```

public void gameEnd()
{
    Player.frame = 0;
    Player.min = 0;
    Player.sec = 0;
    Player.life = 5;
    Player.bullet = 0;
    Boss.hp = 300;
    Stage1.SetActive(false);
    Stage2.SetActive(false);
    Stage3.SetActive(false);
    Player1.SetActive(false);
    Menu.SetActive(true);
    music(11);
}

```

//상황에 맞는 노래를 실행하는 함수

```
public void music(int num)
{
```

```
    switch (num)
```

```
    {
```

```
        case 0:
```

```
            menu.Stop();
```

```
            stage2.Stop();
```

```
            stage3.Stop();
```

```
            stage1.Play();
```

```
            stage1.loop = true;
```

```
            break;
```

```
        case 1:
```

```
            menu.Stop();
```

```
            stage1.Stop();
```

```
            stage3.Stop();
```

```
            stage2.Play();
```

```
            stage2.loop = true;
```

```
            break;
```

```
        case 2:
```

```
            menu.Stop();
```

```
            stage1.Stop();
```

```
            stage2.Stop();
```

```
            stage3.Play();
```

```
            stage3.loop = true;
```

```
            break;
```

```
        case 3:
```

```
            stage1.Stop();
```

```
            stage2.Stop();
```

```
            stage3.Stop();
```

```
            gameOver.Play();
```

```
            break;
```

```
        case 4:
```

```
            stage1.Stop();
```

```
            stage2.Stop();
```

```
            stage3.Stop();
```

```
            win.Play();
```

```
            break;
```

```
        case 5:
```

```
            die.Play();
```

```
            break;
```

```
        case 6:
```

```
            hit.Play();
```

```
            break;
```

```
        case 7:
```

```
            attack.Play();
```

```
            break;
```

```
        case 8:
```

```
            heart.Play();
```

```
            break;
```

```
        case 9:
```

```
            enemy.Play();
```

```
            break;
```

```
        case 10:
```

```
            fire.Play();
```

```
            break;
```

```
        case 11:
```

```
            menu.Play();
```

```
            menu.loop = true;
```

```
            break;
```



## Player.cs

```
public class Player : MonoBehaviour
{
    private Animator anime;
    private GameObject prefab;
    public static int life, bullet, frame, min, sec;
    private bool jump, gameOverFlag, heartFlag;
    private GameObject main;
    private Text hpCount, bulletCount, timeCount;

    // Start is called before the first frame update
    void Start()
    {
        frame = 0;
        min = 0;
        sec = 0;
        life = 5;
        bullet = 0;
        jump = false;
        gameOverFlag = true;
        heartFlag = true;
        main = GameObject.FindWithTag("MainGame");
        anime = gameObject.GetComponent<Animator>();
        prefab = (GameObject)Resources.Load("playerFire");
        hpCount = GameObject.FindWithTag("heartUi").GetComponent<Text>();
        bulletCount = GameObject.FindWithTag("bulletUi").GetComponent<Text>();
        timeCount = GameObject.FindWithTag("clockUi").GetComponent<Text>();
    }

    void FixedUpdate()
    {
        //UI창 표시를 위한 명령
        frame++;
        min = frame / (50 * 60);
        sec = (frame / 50) % 60;
        hpCount.text = "X " + life.ToString();
        bulletCount.text = "X " + bullet.ToString();
        timeCount.text = min.ToString() + " : " + sec.ToString();
        if ((life < 0) && (gameOverFlag))
        {
            //게임 오버 시 수행되는 명령
            life = 0;
            gameOverFlag = false;
            GameObject.FindWithTag("main").GetComponent<Text>().text = "Game Over";
            main.GetComponent<MainGame>().music(3);
            Invoke("returnToTitle", 5f);
        }
        if (transform.position.y < MainGame.savePoint.y - 5f)
        {
            //각 스테이지 별 낙하 판정 뒤 세이브 포인트로 리스폰 되고
            //폭숨이 하나 감소하게 됩니다.
            if (transform.position.y < -22f)
            {
                GameObject.FindWithTag("Struct8").GetComponent<Struct8>().defaultPos();
            }
            transform.position = MainGame.savePoint;
            life--;
        }
        float h = Input.GetAxis("Horizontal");
        float v = Input.GetAxis("Vertical");
    }
}
```

```

//애니메이션 제어를 위한 조건문과 명령어
if((h == 0) && (v == 0))
{
    anime.SetBool("Idle", true);
    anime.SetBool("Jump", false);
    anime.SetBool("Walk", false);
}
if ((h != 0) || (v != 0))
{
    anime.SetBool("Walk", true);
    anime.SetBool("Idle", false);
    anime.SetBool("Jump", false);
}
if (jump)
{
    anime.SetBool("Jump", true);
    anime.SetBool("Walk", false);
    anime.SetBool("Idle", false);
}
//플레이어의 행동을 구현하기 위한 명령어
transform.Translate(new Vector3(h,0,v)* Time.deltaTime * 5f);
if ((!jump)&&Input.GetKeyDown(KeyCode.Space))
{
    gameObject.GetComponent<Rigidbody>().AddForce(Vector3.up*300f);
}
//총알이 있고 마우스 왼쪽 버튼을 누를 경우 총이 발사합니다.
if (bullet > 0)&&(Input.GetMouseButtonDown(0)))
{
    main.GetComponent<MainGame>().music(10);
    Instantiate(prefab, new Vector3(transform.position.x, transform.position.y, transform.position.z-2f), Quaternion.identity);
    bullet--;
}

//이단 점프를 방지하기 위한 명령
void OnCollisionStay()
{
    jump = false;
}
void OnCollisionExit()
{
    jump = true;
}
void OnTriggerEnter(Collider other)
{
    //portal과 접촉시 해당하는 세이브 포인트로 스폰됩니다.
    if (other.gameObject.CompareTag("portal"))
    {
        GameObject.FindWithTag("objectiveUi").GetComponent<Text>().text = "Find portal";
        transform.position = new Vector3(0,-21,-8);
        MainGame.savePoint = new Vector3(0,-21,-8);
        main.GetComponent<MainGame>().music(1);
    }
    if (other.gameObject.CompareTag("portal1"))
    {
        GameObject.FindWithTag("objectiveUi").GetComponent<Text>().text = "Defeat enemy";
        transform.position = new Vector3(0, -48, -80);
        MainGame.savePoint = new Vector3(0, -48, -80);
        main.GetComponent<MainGame>().music(2);
        GameObject.FindWithTag("Boss").GetComponent<Boss>().stageStart();
    }
    //총알 아이템을 먹을 경우 총알 갯수가 20개 증가합니다.
    if (other.gameObject.CompareTag("bullet"))
    {
        bullet += 20;
    }
    //하트 아이템을 먹은 경우 목숨이 1개 증가합니다.
    if (other.gameObject.CompareTag("heart"))
    {
        if (heartFlag) { main.GetComponent<MainGame>().music(8); }
        life +=10;
    }
}

```

setStage1

```

public class setStage1 : MonoBehaviour
{
    //프리팜 배열
    GameObject[] prefabs;
    float[] height;
    float[] depth;
    GameObject tmp;

    void Start()
    {
        //랜덤 배치로 스테이지1이 구성되기 때문에
        //각 구조물들의 높이와 너비를 저장합니다
        GameObject.FindWithTag("MainGame").GetComponent<MainGame>().music(0);
        prefabs = new GameObject[7];
        height = new float[10];
        depth = new float[10];
        MainGame.savePoint = new Vector3(0, 11, -13);
        prefabs[0] = (GameObject)Resources.Load("Struct1");
        prefabs[1] = (GameObject)Resources.Load("Struct2");
        prefabs[2] = (GameObject)Resources.Load("Struct3");
        prefabs[3] = (GameObject)Resources.Load("Struct4");
        prefabs[4] = (GameObject)Resources.Load("Struct5");
        prefabs[5] = (GameObject)Resources.Load("Struct6");
        prefabs[6] = (GameObject)Resources.Load("Struct7");

        height[0] = 0f;
        height[1] = 0f;
        height[2] = 0f;
        height[3] = 0f;
        height[4] = 11f;
        height[5] = 5f;
        height[6] = 0f;

        depth[0] = 20f;
        depth[1] = 11f;
        depth[2] = 12f;
        depth[3] = 14f;
        depth[4] = 13f;
        depth[5] = 16.5f;
        depth[6] = 11f;

        SetStage1();
    }

    void SetStage1()
    {
        List<int> structList = new List<int>();

        //랜덤으로 호출
        //인스턴스화 후 배치
        float z = -16.5f;
        float y = 10.5f;
        //랜덤으로 4개의 구조물을 선택
        for (int i = 0; i < 4;)
        {
            int x = Random.Range(0, 7);
            if (structList.Contains(x))
            {
                continue;
            }
            else
            {
                structList.Add(x);
                i++;
            }
        }
        //계산된 위치에 나란히 구조물을 배치합니다.
        foreach (int x in structList)
        {
            y += height[x];
            z -= depth[x];
            tmp = Instantiate(prefabs[x], new Vector3(0, y, z), Quaternion.identity);
            tmp.transform.parent = transform;
            z -= depth[x];
        }
        tmp = (GameObject)Resources.Load("portal");
        //스테이지2를 위한 portal
        Instantiate(tmp, new Vector3(0, y+1f, z - 0.5f), Quaternion.identity);
    }
}

```



## SetStage2.cs

```
public class setStage2 : MonoBehaviour
{
    private GameObject prefab1;
    private GameObject prefab2;
    private GameObject prefab3;
    private GameObject tmp;
    // Start is called before the first frame update
    void Start()
    {
        prefab1 = (GameObject)Resources.Load("horizontalBlock");
        prefab2 = (GameObject)Resources.Load("verticalBlock");
        prefab3 = (GameObject)Resources.Load("Enemy");

        //x [24,-24]
        //z [-20,-70]
        int[] xArray = new int[10];
        int i;

        //가로축으로 통로로 만들 위치저장
        for(i = 0; i < 9; i++)
        {
            xArray[i] = Random.Range(0,5);
        }

        for (i = 20; i < 100; i += 5)
        {
            //가로 구조물 생성
            if (i % 10 == 0) {
                for (int j = 0; j < 5; j++)
                {
                    //선택된 좌표에 대한 가로 블록 생성 스킵
                    if (j == xArray[i/10-2]) { continue; }
                    //20%로 블록 생성 스킵
                    if (Random.Range(0, 5) != 0)
                    {
                        tmp = Instantiate(prefab1, new Vector3(24-12*j, -23, -i), Quaternion.identity);
                        tmp.transform.parent = transform;
                    }
                }
            }
            //세로 구조물 생성
            else
            {
                int a = System.Math.Min(xArray[i / 10 - 2], xArray[i / 10 - 1]);
                int b = System.Math.Max(xArray[i / 10 - 2], xArray[i / 10 - 1]);
                //세로 [a,b] 에 대한 세로 블록 생성 스킵
                for (int j = 0; j < 4; j++) {
                    if ((j >= a) && (j < b)) {
                        //비어 있는 확정 구간에 50프로의 확률로 적을 배치
                        if (Random.Range(0,2) == 0 )
                        {
                            tmp = Instantiate(prefab3, new Vector3(18 - 12 * j, -21, -i), Quaternion.identity);
                            tmp.transform.parent = transform;
                        }
                        continue;
                    }
                    //20프로로 블록 생성 스킵
                    if(Random.Range(0,5) != 0)
                    {
                        tmp = Instantiate(prefab2, new Vector3(18-12*j, -23, -i), Quaternion.identity);
                        tmp.transform.parent = transform;
                    }
                }
            }
        }
    }
}
```

## Stage3.cs

```

public class setStage3 : MonoBehaviour
{
    GameObject[] prefabs;
    GameObject tmp;
    int num,x,z;
    // Start is called before the first frame update
    void Start()
    {
        prefabs = new GameObject[2];
        prefabs[0] = (GameObject)Resources.Load("heart");
        prefabs[1] = (GameObject)Resources.Load("bullet");
        //5초 후 6초마다 하트나 총알을 랜덤으로 리스폰하는 함수
        InvokeRepeating("createItem", 5f, 6f);
    }

    // Update is called once per frame
    void createItem()
    {
        //랜덤한 위치에 하트나 총알이 생성되어 떨어집니다.
        x = Random.Range(-4, 5);
        z = Random.Range(-91, -83);
        num = Random.Range(0, 2);
        tmp = Instantiate(prefabs[num],new Vector3(x,-42,z),Quaternion.identity);
        tmp.transform.parent = transform;
    }
}

```

## Boss.cs

```

public class Boss : MonoBehaviour
{
    Animator anime;
    GameObject prefab,tmp;
    int num, x, z;
    public static int hp;
    bool hitFlag,dieFlag;
    // Start is called before the first frame update
    void Start()
    {
        hitFlag = true;
        dieFlag = true;
        anime = transform.GetComponent<Animator>();
        hp = 300;
        prefab = (GameObject)Resources.Load("stone");
    }
    //8초 간격으로 돌을 만들어 냅니다.
    public void stageStart()
    {
        InvokeRepeating("createStone", 0f, 8f);
    }
    void Update()
    {
        //보스몬스터가 죽을 경우 게임 우승 문구가 출력되고 승리 사운드가 나온 뒤 메뉴 화면으로 돌아갑니다
        if ((hp < 0)&&(dieFlag))
        {
            dieFlag = false;
            anime.SetBool("Die", true);
            GameObject.FindWithTag("MainGame").GetComponent<MainGame>().music(7);
            GameObject.FindWithTag("main").GetComponent<Text>().text = "The End";
            GameObject.FindWithTag("MainGame").GetComponent<MainGame>().music(4);
            Invoke("returnToTitle", 8f);
        }
    }
}

```

```

//메뉴로 돌아가기 전 게임의 초기 시작 설정을 실행하고 게임 메뉴로 돌아갑니다.
void returnToTitle()
{
    GameObject.FindWithTag("main").GetComponent<Text>().text = "";
    MainGame.savePoint = new Vector3(0, 11, -13);
    GameObject.FindWithTag("Player").transform.position = new Vector3(0, 11, -13);
    GameObject.FindWithTag("MainGame").GetComponent<MainGame>().gameEnd();
}

// Update is called once per frame
void createStone()
{
    if (dieFlag)
    {
        //보스몬스터의 울음소리가 연출됩니다.
        GameObject.FindWithTag("MainGame").GetComponent<MainGame>().music(7);
        anime.SetBool("GetHit", false);
        for (int i = 0; i < 5; i++)
        {
            //랜덤한 위치에 5개의 돌이 생성되어 떨어집니다.
            x = Random.Range(-4, 5);
            z = Random.Range(-91, -83);
            tmp = Instantiate(prefab, new Vector3(x, -42, z), Quaternion.identity);
            tmp.transform.parent = transform;
        }
    }
}

public void hit()
{
    //플레이어의 총에 맞을 경우 체력이 5씩 감소합니다.
    hitFlag = false;
    if (hitFlag)
    {
        GameObject.FindWithTag("MainGame").GetComponent<MainGame>().music(7);
        anime.SetBool("GetHit", true);
        hitFlag = true;
    }
    hp -= 5;
}

```

## Stage1의 구조물(#1~#7)

```

public class Struct1 : MonoBehaviour
{
    private int speed;
    void Start()
    {
        speed = Random.Range(0, 10)+3;
    }
    //랜덤한 속도로 y축 중심의 회전
    void FixedUpdate()
    {
        transform.Rotate(Vector3.up, Time.deltaTime * 7*speed);
    }
}

public class Struct2 : MonoBehaviour
{
    private GameObject prefab;
    private Vector3 pos;
    // Start is called before the first frame update

    //1초마다 블록 생성
    void Start()
    {
        prefab = (GameObject)Resources.Load("sstruct2");
        InvokeRepeating("makeBlock", 0f, 1f);
    }

    //랜덤한 위치에 앞으로 다가오는 블록을 생성
    void makeBlock()
    {
        int w = Random.Range(0, 4);
        int x = Random.Range(0, 3);
        pos = new Vector3(transform.position.x-3f+2.25f*w, transform.position.y+1*x*0.2f, transform.position.z-8f);
        Instantiate(prefab, pos, Quaternion.identity);
    }
}

public class Struct3 : MonoBehaviour
{
    private int state;
    private int speed;
    //2초마다 원판의 이동 방향을 바꿉니다.
    void Start()
    {
        speed = Random.Range(0, 10) + 1;
        state = 1;
        InvokeRepeating("stateChange", 0, 2);
    }
    //state에 따라 원판이 위나 아래로 이동합니다.
    void FixedUpdate()
    {
        gameObject.GetComponent<Transform>().Translate(state * Vector3.up * Time.deltaTime*speed*0.5f);
    }

    void stateChange()
    {
        state *= -1;
    }
}

public class Struct4 : MonoBehaviour
{
    private GameObject prefab;
    private Vector3 pos;
    //1초마다 블록을 생성합니다.
    void Start()
    {
        prefab = (GameObject)Resources.Load("sstruct4");
        InvokeRepeating("makeBlock",0f,1f);
    }

    //랜덤한 위치로 앞으로 오는 블록을 생성합니다.
    void makeBlock()
    {
        int w = Random.Range(0, 5);
        pos = new Vector3(transform.position.x - 2.5f + 1.2f*w, transform.position.y, transform.position.z - 8f);
        Instantiate(prefab, pos, Quaternion.identity);
    }
}

```

```

public class Struct5 : MonoBehaviour
{
    private GameObject prefab,prefab2;
    private Vector3 pos;
    private int blockCount,blockCount2;
    // Start is called before the first frame update
    //0.5초마다 통나무를 생성하는 함수를 호출
    void Start()
    {
        blockCount2 = 5;
        prefab2 = (GameObject)Resources.Load("sstruct52");
        InvokeRepeating("makeBlock2", 0f, 0.5f);
    }
    //랜덤한 위치로 통나무를 생성합니다.
    void makeBlock2()
    {
        if (blockCount2 > 0)
        {
            int w = Random.Range(0, 4);
            pos = new Vector3(transform.position.x - 3f + 2 * w, transform.position.y + 5f, transform.position.z - 4f);
            Instantiate(prefab2, pos, Quaternion.Euler(new Vector3(0,0,-90)));
            --blockCount2;
        }
    }
    public void plusBlockCount2()
    {
        ++blockCount2;
    }
}

public class Struct6 : MonoBehaviour
{
    private bool state1 = true;
    private bool state2 = false;

    //4초마다 활성 상태를 변화합니다
    // Start is called before the first frame update
    void Start()
    {
        InvokeRepeating("changeState", 0f, 4f);
    }
    //활성상태에 따라 블럭이 나타나거나 사라집니다.
    // Update is called once per frame
    void FixedUpdate()
    {
        transform.GetChild(0).gameObject.SetActive(state1);
        transform.GetChild(1).gameObject.SetActive(state2);
    }

    void changeState()
    {
        state1 = !state1;
        state2 = !state2;
    }
}

```



```

public class Struct7 : MonoBehaviour
{
    private GameObject prefab;
    private Vector3 pos;
    private int idx=0;
    private int num;
    private int[,] array = { { 0, 0 }, { -1, 0 }, { -1, 0 }, { 0, -1 },
                              { -1, 0 }, { 0, -1 }, { 1, 0 }, { 0, -1 }, { 1, 0 }, { 1, 0 }, { 0, -1 }, { -1, 0 }, { 0, -1 } };

    // Start is called before the first frame update
    // 1초마다 블럭 생성하는 함수 호출
    void Start()
    {
        prefab = (GameObject)Resources.Load("sstruct7");
        InvokeRepeating("makeBlock", 0f, 1f);
    }

    // 좌표에 찍힌 방향으로 아래로 천천히 떨어지는 블럭을 생성합니다.
    void makeBlock()
    {
        num = Random.Range(0, 3) + 2;
        if (idx==0)
        {
            pos = new Vector3(transform.position.x + 4f, transform.position.y, transform.position.z + 6f);
        }
        pos = new Vector3(pos.x+array[idx,0]*2.25f, pos.y, pos.z+array[idx,1]*2f);
        Instantiate(prefab, new Vector3(pos.x,pos.y+0.3f*num,pos.z), Quaternion.Euler(Vector3.right*90));
        idx = (idx+1)%14;
    }
}

```

## Stage3의 구조물

```
public class Struct9 : MonoBehaviour
{
    GameObject prefab;
    private Vector3 tmp;
    //다시 생성해야 할 위치를 저장하기 위한 큐
    private Queue<Vector3> schedule;

    void Start()
    {
        schedule = new Queue<Vector3>();
        prefab = (GameObject)Resources.Load("sstruct9");
    }
    //8초가 지난 뒤 사라진 발판을 원 위치에 생성하는 함수를 호출합니다.
    public void create(Vector3 pos)
    {
        //큐 값 저장
        schedule.Enqueue(pos);
        Invoke("func1", 8f);
    }
    //
    //먼저 저장된 발판 위치에 해당하는 곳에 발판을 다시 생성합니다.
    void func1()
    {
        tmp = schedule.Dequeue();
        Instantiate(prefab,tmp,Quaternion.Euler(Vector3.right*90)).transform.parent = transform;
    }
}
```