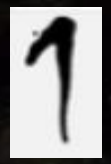
폐허 탈출

팀 명: **PRIA**

이대건, 최광현, 이창호, 박재홍

목차



스토리 라인

게임 실행

구현 목표 리스트

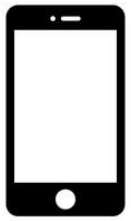
질문 & 소감

1 스토리 라인

1

2

3



4

5

6

2 게임 실행

구현리스트	
1.	공포분위기
2.	스마트폰 UI
3.	스크린 샷
4.	스크린샷에 특정 오브젝트가 있는지 판단 → Mission 달성
5.	미션 달성 후 보상
6.	탈출을 위한 오브젝트
7.	포톤 클라우드

3 구현 목표 리스트

1. 공포분위기

배경음악

Post Process활성화

3 구현 목표 리스트

2. 스마트폰 UI

전화 수신

메뉴 화면

보조 화면

3 구현 목표 리스트

3. 스크린 샷 (1)

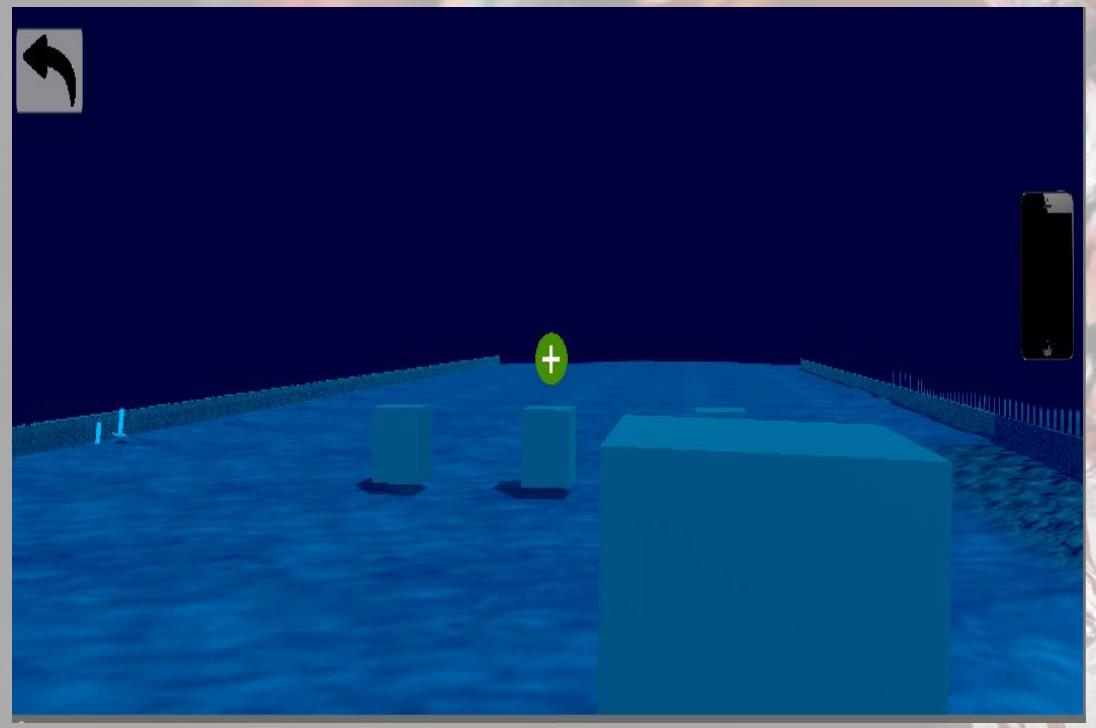


사진 촬영

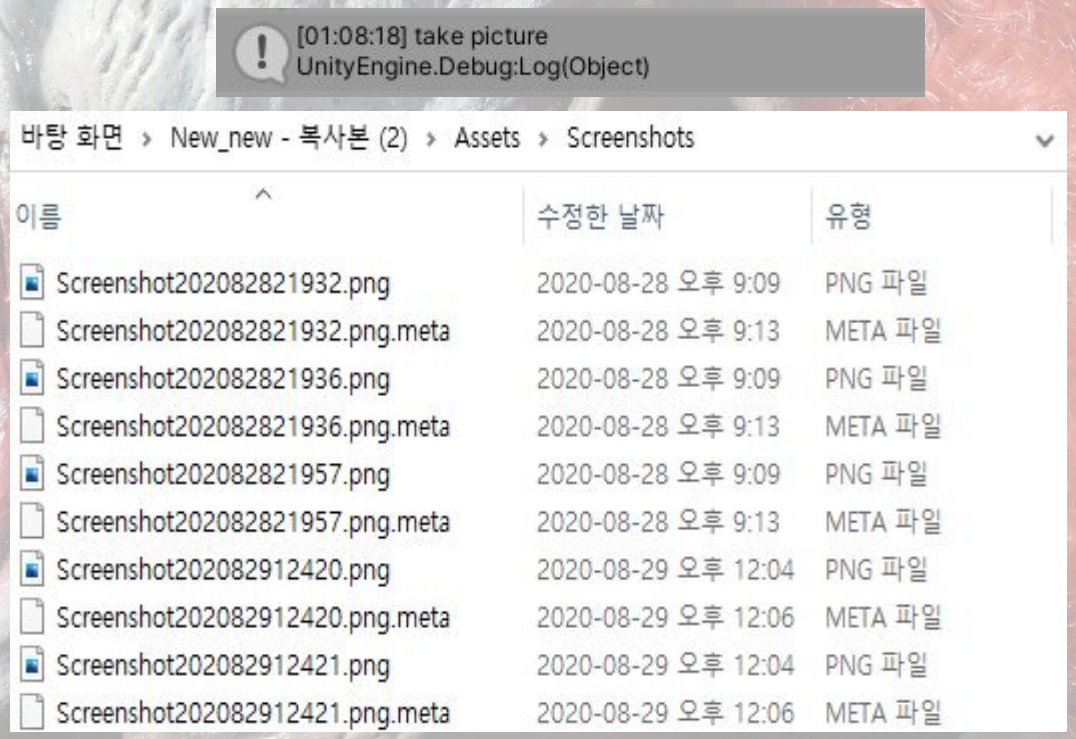


사진 저장

3 구현 목표 리스트

3. 스크린 샷 (2)

```
ScreenCapture.CaptureScreenshot(path);
```

```
dir = new DirectoryInfo(Application.dataPath + "/Screenshots");
info = dir.GetFiles("*.png");
```


```
byte[] data = File.ReadAllBytes(info[i].ToString());
Texture2D texture = new Texture2D(2, 2);
texture.LoadImage(data);
Sprite sprite = Sprite.Create(texture, new Rect(0, 0, texture.width, texture.height), new Vector2(0.5f, 0.5f), 100.0f);
```

스크립트

3 구현 목표 리스트

4. 스크린샷에 특정 오브젝트가 있는지 판단 → Mission 달성 (1)

```
if(Input.GetButtonDown("Jump") && csScreenShot.isCameraOn==true) {
    Debug.Log("Find Target");
    for(int i = 0; i < target.Length; i++) {
        Vector3 viewPos = mainCamera.WorldToViewportPoint(target[i].tran
        if((0<viewPos.x && viewPos.x<1) && (0 < viewPos.y && viewPos.y <
            Debug.Log("Target : " + target[i].name);
            for(int j=0; j<csMissionManager.mission.Count; j++) {
                if(target[i].name == csMissionManager.mission[j].mission
                    Debug.Log("Mission : " + target[i].name);
                    csMissionManager.createDroneIdx = csMissionManager.m
                    csMissionManager.deleteMission(j);
                }
            }
        }
    }
}
```

```
[01:08:18] Find Target
UnityEngine.Debug:Log(Object)
[01:08:18] Target : Cube1
UnityEngine.Debug:Log(Object)
[01:08:18] Mission : Cube1
UnityEngine.Debug:Log(Object)
[01:08:18] Target : Cube2
UnityEngine.Debug:Log(Object)
[01:08:18] Mission : Cube2
UnityEngine.Debug:Log(Object)
```

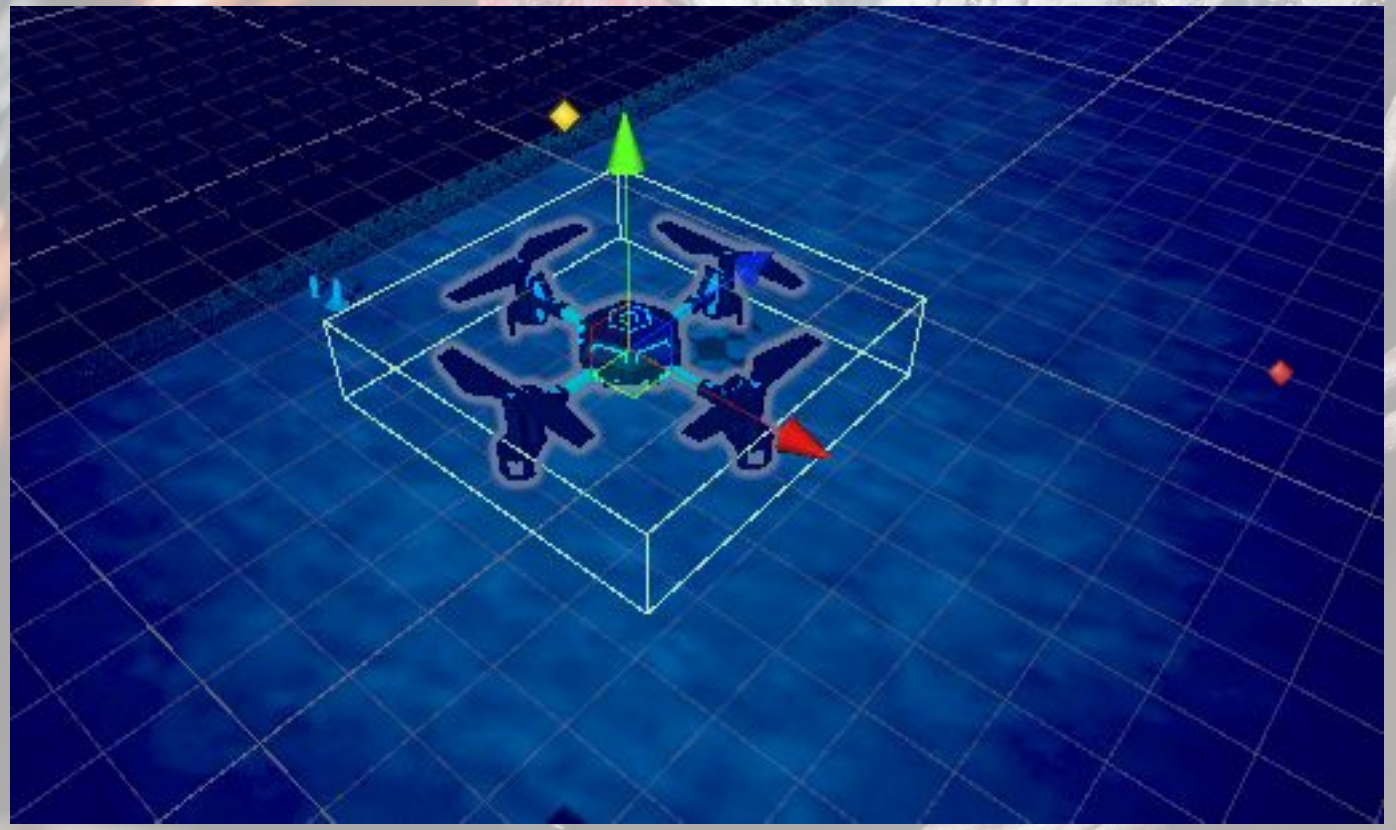
스크립트

3 구현 목표 리스트

4. 스크린샷에 특정 오브젝트가 있는지 판단 → Mission 달성 (2)

3 구현 목표 리스트

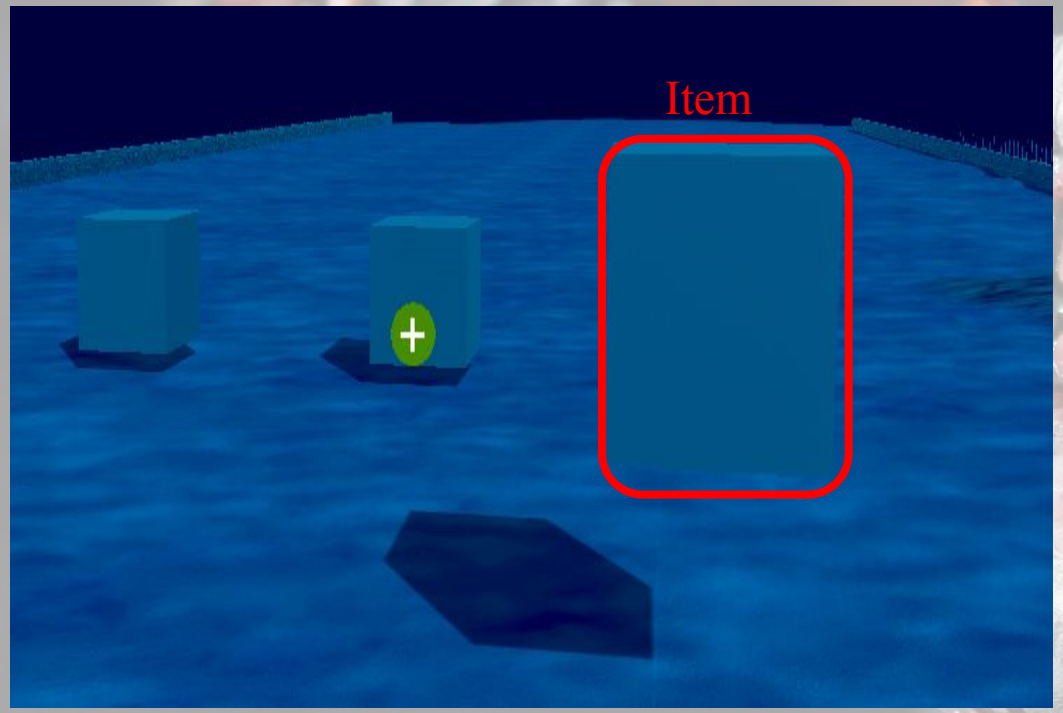
5. 미션 달성 후 보상

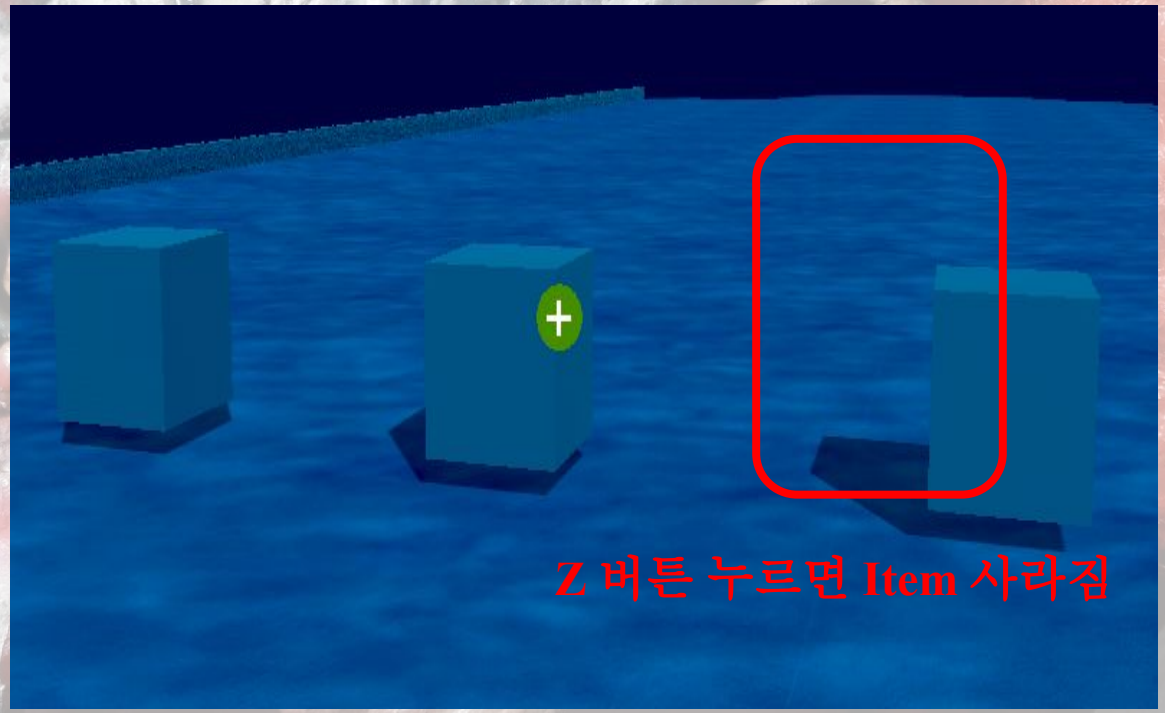


Drone (Game Manager / Mission Manager)

3 구현 목표 리스트

5. 미션 달성 후 보상





Items

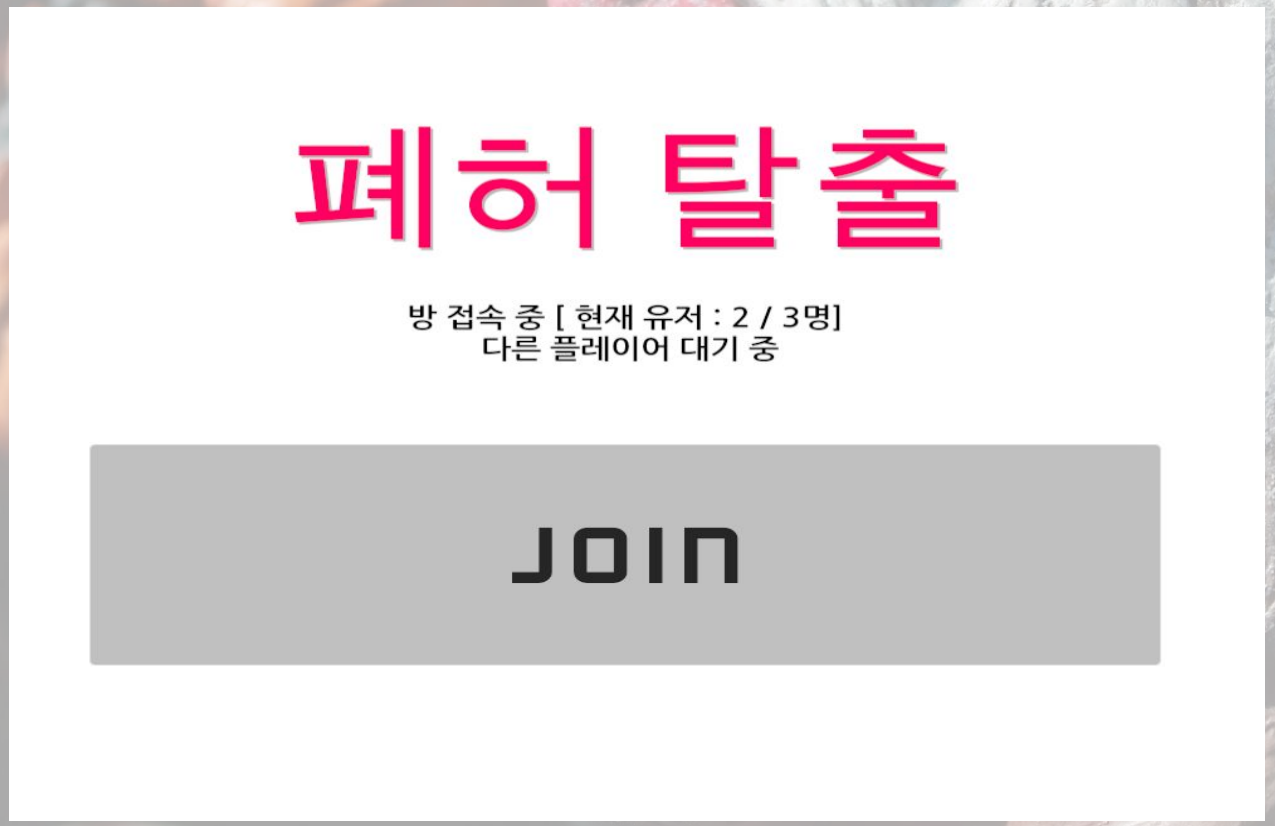
3 구현 목표 리스트

6. 탈출을 위한 오브젝트

열기구 상승 (탈출)

3 구현 목표 리스트

7. 포톤 클라우드



3 구현 목표 리스트

7. 포톤 클라우드

```
public override void OnJoinRandomFailed(short returnCode, string message) {
    connectionInfoText.text = "빈 방이 없음, 새로운 방 생성...";
    PhotonNetwork.CreateRoom(null, new RoomOptions { MaxPlayers = 3 });
}
```

```
[PunRPC]
참조 0개
public void ChkRoomUser(int gameUser)
{
    this.gameUser = gameUser;
    connectionInfoText.text = $"방 접속 중 [ 현재 유저 : {this.gameUser} / {chkUser}명]" + "\n 다른 플레이어 대기 중";
}
```

[PunRPC]

원격 절차 호출 **Remote Procedure Calls**

3 구현 목표 리스트

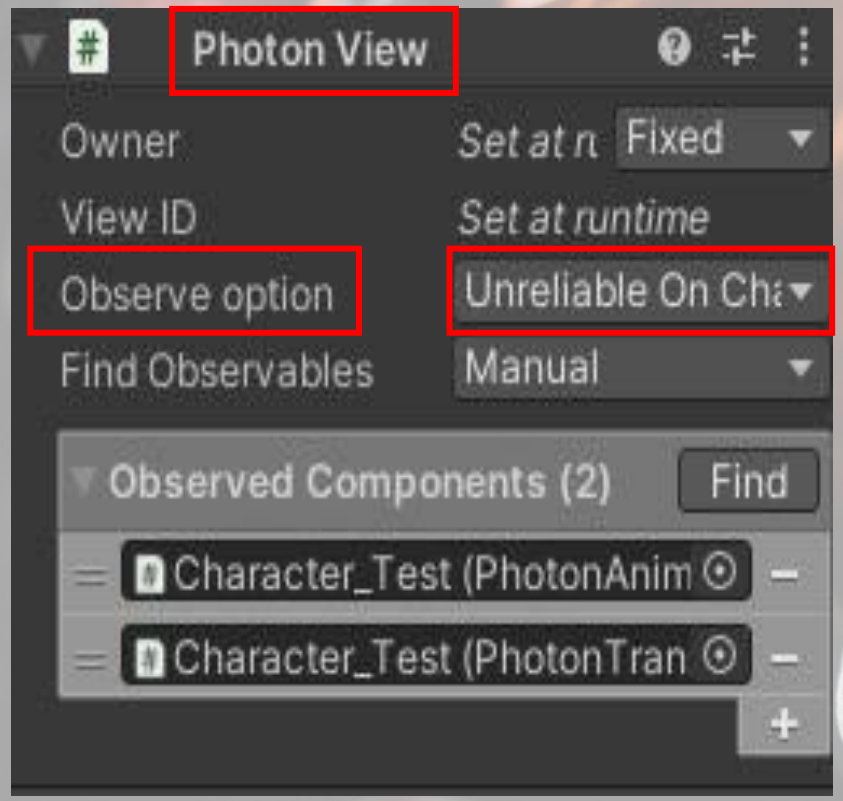
7. 포톤 클라우드

움직이는 대상에 대한 포톤 적용

Particle에 대한 포톤 처리

3 구현 목표 리스트

7. 포톤 클라우드



Photon View
컴퓨터들의 다양한 인스턴스들을 하나로 연결해 주며 어떤 컴포넌트들을 관찰하고 어떻게 관찰 할지를 정의

Observe option
업데이트를 보내는 방법과시기

Unreliable on Change

각 업데이트에서 변경 사항을 확인합니다. 전송 된 다음 다시 변경되지 않는 한 소유자가 업데이트 전송을 중지합니다.한동안 추가 업데이트를 생성하지 않는 게임 오브젝트에 유용합니다.

4 질문 & 소감