

## o Start

### o 1 ~ 2P : Unity Object Life Cycle

- 3 ~ 4P : Keyboard, mouse - Move Object
- 5P : 물体质 이동
- 6P : 델타타임(이동거리 증정)
- 7P : 실제와 같은 물체를 만들기 - 물리관련 Component
- 8P : 힘을 이용하여 물체 움직여보기

### • 9P ~ 10P : 물리 충돌 이벤트

### • 11P ~ 12P : UG UI 기초

• 유튜브 : [ 7초만으로 놀라운 3D 게임 만들기 ] BE1. In YouTub

• 13P : 2D 프로젝트 준비하기 ( 2D 캐릭터 TIP )

• 14P : 2D 아틀라스와 애니메이션

• 15P ~ 17P : 2D 플레이어 이동구현 : Animation 조정(FadeIn) 등 )) 연계

• 18P ~ 20P : 2D 플레이어 점프구현 : Jump, RayCast 관련

• 21P : TileMap으로 Platform 만들기

• 22P ~ 23P : 몬스터 AI 구현

• 24P ~ 25P : 플레이어 죽음 Event

• 유튜브 : [ 2D Platform 완성하기 ] BE2

• 26 ~ 27P : 짜증 같은 탑다운 RPG 쉽게 준비하기

• 28 ~ 30P : 짜증 식 액션 구현하기 - 대맞은 이동금지, 아웃라이언(쓰러짐), 죽기 액션(RAY)

• 31P ~ 33P : 대화창 UI 구현하기 - 대화창 코드, 대화관련 액션 ..

• 34 ~ 37P : 대화 시스템 구현 ~~XLG~~.

화이팅!: 계획구조

• RPG 쿠스트 시스템 구현

Debug.Log();

# C# 정복

여러자료형 0

같은자료형 만

## ① 컬렉션 : ArrayList, List, Queue

HashTable, Dictionary  
Queue, Stack

아주 자료형이나 도구 (내부에선, 변수)

크기 고정, 틸드연산자 계속해서 사용 가능 ↑

과부화

ArrayList arrayList = new ArrayList();

] → 배열과 유사. index.

List<int> list = new List<int>();

HashTable hashTable = new HashTable(); → 딕셔너리 유사. key 키

hashTable[key] = value, Add ("key", value)

Dictionary<String, int> dictionary = new Dictionary<String, int>();

HashTable의 Key와 Value가 자료형을 지정. 자료형 변수

명시적 타입대로. 안쓰면 됨. 하지만 관리하는 데 더 좋다

Queue<int> queue = new Queue<int>(); // FIFO. enqueue, dequeue

Stack<int> stack = new Stack<int>(); // LIFO. Push, Pop

→ 오류제거 (설치, 초기화)

포션다이얼

\* TIP

Void 함수 (~~, String stringX = "C# 툴트립")

어려운 드래그 베이스 놓으면  
놓고자 하는  
안 놓으면  
지정한 거 들어감

# C# 정복 네임스페이스

[네임스페이스?]: 누군가 만든 Class, Method, Function을 꽂아 쓰는거.

using Test.Studio; 4

{  
    using Test; 42

NameSpace Test

{ Public Class Youtube

{ Public int SubScribe;

{ NameSpace Studio

{

Public Class Youtube

{

int like;

Public Void SetLike(int value);

{

like = Value;

3

Public bool IsLike()

{

Return like != 0;

3

8

public Class Test\_Name : MonoBehaviour {

Test.Youtube Sni;

Void Start()

{

Sni = new Test.Youtube();

Sni.Subscribe (5);

Print (Sni.Subscribe);

3

이름정지면 사용하고 싶은거에 대해서

using Test.Studio;

Test.Youtube

Test.Studio.Youtube

오해쓰기?

접두, 다른 프로젝트, 외부 라이브러리

{  
코드 이름중복?

NameSpace?

중복확인

# C# 정보 - 구조체 + 열거형 자료

Public Struct Youtube

변수선언 O  
함수선언 O

Class로 차이없음.

직접값 대입 불가.  
int a ≠ 5;

왜 굳이??

{ Public int a;

Public void GetA(int v)

{

a = v

}

~ 단순한 그릇 역할

값 넣는 빌.

1. Public int, ... 변수

2. 초기화하는 형태

Struct - 구조체



Class - 발전되어 C# 등

Class로 발전.

Object  
Instance 그려서 반환용(있을지) 객체

Struct를 남겨둠.

상속 가능

Public Class Test : MonoBehaviour

{

Youtube Shii;

Void Start()

{

Shii.a = 5;

Shii.GetA(5);

Struct는 Value Type

Class는 Reference Type (주소값)

Vector3 h = new Vector3(~)  
이 이유는

가변 주소 주소. 아직 풀의 상태  
(class)

생성자.

## 열거

Public enum Item

{

Weapon,

Shield,

Potion

3

남겨둠

정정한 값이 만들어짐

선택지 (중복)

자료 선택지

Public Class ~~~

Item item;

Void Start()

{

item = Item.Weapon;

}

# Object Life Cycle . By Gold Metal

- Life Cycle 단계 관리 X 실행순서

## 초기화 영역

- 「 Awake : Game Object 생성 시 최초실행 ( 이미 있는건도 실행됨 )」

## 활성화 영역

- 「 OnEnable : Awake보다 뒤에, Start보다 빠르기」

Game Object 활성화 되었을 때.

※ 최초 회실행이 아니라 ON, OFF 시마다 실행됨」

- Start : Update 시작직후 최초실행



」

## 물리연산 영역

- 「 FixedUpdate : 물리연산 업데이트 . 물리연산하기 전 업데이트」

↳ 고정된 실행주기로 컴퓨터 사용 영향 줄고 CPU 사용량 ↓  
(보통 50/5)

↳ Time.deltaTime  
안해도 됨.

」



## 게임 로직 영역

- 「 Update : 게임로직 업데이트 . 물리제외 주기적 .」

↳ 환경에 따라 실행주기 달라질 수 있음 보통 60ms

- Late Update : 모든 업데이트 끝난 후 실행

\* 주로 캐릭터를 따라가는  
카메라 or Logic의 후처리

• 비활성화 영역

On Disable : 비활성화 . Object 코드를 깃다쳤다.

• 해제

On Destroy : 게임 오브젝트가 삭제될 때 . 이 오브젝트가 실제로 갖고  
있었던 낭비하고 삭제되는 느낌 .

- 예시 쓰것 -

- Debug.Log(" " );

Awake : Player data 준비

OnEnable : Player Login

Start : 사냥장비 청기기

Fixed Update : ~0.1초

Update : 주소 ~ 사냥

Late Update : 경지 낙을

OnDisable : Player Logout

OnDestroy : Player data 초기화

# Key Board & mouse - Input Object

Input: 게임내 입력 관리 Class (마우스, 키보드 ~)

return type : Bool

L Input . AnyKey Down : 아무 입력을 최초로 냈을 때 True

Input . Any Key : 아무 입력을 냈을 때 True

o 입력 방식 : 누를 때, 누르고 있을 때, 떼 때

Input . Get Key Down (KeyCode, Return) → 0이 Enter, Enter는 아로 있고, 121은 Enter인  
KeypadEnter는 아로

II Get Key (KeyCode. LeftArrow)

II Get Key Up (KeyCode. RightArrow) ↗ 키 상태

→ .keyCode는 class임

함수의 내용은 해당함수로  
전달된 값은 기준기준에 따른 번수

△ 마우스는? : Get Mouse (마우스 입력을 냈으면 True)

Get Mouse Button Down (0) → 0은 마우스 왼쪽 버튼 1은 마우스 오른쪽

Get Mouse Button (1) → 마우스 버튼이 0, 1

Get Mouse Button Up (0)

△ 대중적인 Button 사용법?

Edit → Project Setting → Input Manager 각 Button 설정 가능

보통 Horizontal (4자) A, D, Right Arrow, Left Arrow

The Left

Vertical (4자) W, S, Up Arrow, Down Arrow

버튼 만들기  
사용법

```

Input . GetButton ("Jump") // Button을 이용 , Fire1 등등
" " . GetButton Down (" " )
" " . GetButton Up (" " )

```

⑥ PC 말고 훈련할 때 GetButton

## ⑥ About

Get Axis : A정, 4직선은 입력을 받으면 float(반환)

ex) Input.GetAxis ("Horizontal")

↓  
좌우로 번갈아가며 1 or -1로 기준으로 오는 (like 가속도)

GetAxisRaw : 가중치 0이 1 or -1이 반환

→ → 동시에 누르면 0. 가만히도 0

TIP

Object는 네임

Transform은

항상 가지고 있음

오브젝트 정의에

대는 기본 청도보트

## ⑦ move

transform.Translate ( Vector3 );

: Vector3을 3차원 위치로 드는다. Vector3은 만큼 이동해주심.

Vector3 move = new Vector3 ( Input.GetAxis ("Horizontal"),
Input.GetAxis ("Vertical"),
0 );

UPdate

transform.Translate ( move );

# 목표지점 이동

- Vector3로 위치와 이동 방식 4가지

Vector3 target = new Vector3(8, 1.5f, 0);

①

Move Towards : 풍속 이동

ex) transform.position = Vector3.MoveTowards (현재위치, 목표위치, 속도);  
 transform. target  
 position

②

Smooth Damp : 부드러운 감속 이동

ex) Vector3 Velo = Vector3.zero; [속도=0이라 가정] / Vector3 Velo = Vector3.Lerp(0, Vector3.zero, 5f);

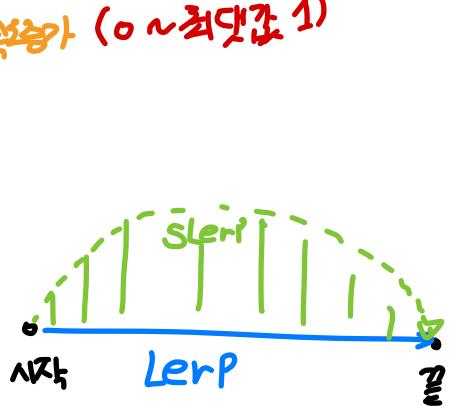
... = Vector3.SmoothDamp (현재위치, 목표위치, 참조속도, 0.1f);  
 transform. target  
 position

Pass by Value or Reference

③

Lerp: 선형보간, Smooth Damp 보다 감속 시간이 짧

ex) ... = Vector3.Lerp (현재위치, 목표위치, 속도);  
 transform. target  
 position



④ SLerp: 구면 선형보간

ex) ... = Vector3.Slerp (현재위치, 목표위치, 속도);  
 transform. target  
 position

\* **델타타임** : Time.deltaTime - 이전 프레임의 완료까지 걸린 시간

○ 사용법: Translate : Vector의 증가  
translate.Translate (Vec \* Time.deltaTime);

Vector 함수 : 시간 단위별로의 증가  
Vector3.Lerp (Vec1, Vec2, T \* Time.deltaTime);

• 설명 void Update ()

{

```
Vector3 vec = new Vector3(  
    Input.GetAxisRaw ("Horizontal") * Time.deltaTime,  
    Input.GetAxisRaw ("Vertical") * Time.deltaTime, 0);  
transform.Translate (vec);
```

}

fps 10fps → Update 10번동  
fps 6fps → Update 6번동  
Translate 주기, 속도 등 달라짐  
} → deltaTime 값은 프레임이 적으면 크고, 프레임이 많으면 작음  
fps가 크면 작은 값으로, 공정하게 만들

# 힘을 이용하여 물체 움직여보기 - Rigid body Component

① 각각 // 힘 1개씩 넣기

TIP: Rigid body  
Fixed update.

Public Class MyBall : MonoBehaviour {

Rigidbody rigid;

```
void Start () {  
    rigid = GetComponent<Rigidbody>(); // Component 탐색  
    rigid.Type = Type.Slow; // 속도 조절  
    rigid.Velocity = Vector3.right; // 초기 속도 설정  
    rigid.AddForce(Vector3.right); // 초기 속도 설정  
}
```

물체의 속도  
바꾸는 방법

```
void FixedUpdate() {  
    if (Input.GetButtonDown("Jump")) {  
        rigid.AddForce(Vector3.up * 5, ForceMode.Impulse);  
        ↓  
        계속 누르면 F↑, a↑, s↑, 물리  
        ↴  
        물체 주는 방식  
        (가속, 무게 반영)  
    }  
}
```

```
Vector3 vec = new Vector3(Input.GetAxisRaw("Horizontal"),  
                           0,  
                           Input.GetAxisRaw("Vertical"));  
rigid.AddForce(vec, ForceMode.Impulse);
```

③ 힘을 찾고 회전하고  
 rigid.AddTorque(Vector3.back); →  
 (Vector3.up); ↪  
 y↑, up, 회전하기로 정의됨

3

# 실제와 같은 물체 만들기

: 프로그래밍 X, 그냥 editor 내 조작

• 물체 필수 요소: Mesh, Material, Collider, RigidBody

→ RigidBody, Sound ... etc

○ 유니티는 **.addComponent** 기반 게임 엔진

1. **중력 적용**: Rigid body 컴포넌트(물체를 블리우는 Component. 중력관련)

설정: Mass: 무게 → 수치가 높을수록 충돌이 무거워짐 (= 강해짐)

움직이는 향정 만들때 유용  
① Use Gravity: 중력 유/무  
② Is Kinematic: 외부 물리효과를 무시. Script을 통해서만 이동시키겠다!

2. **충돌 영역 정의**: Collider 물리학적 물체를 관리하는 Component

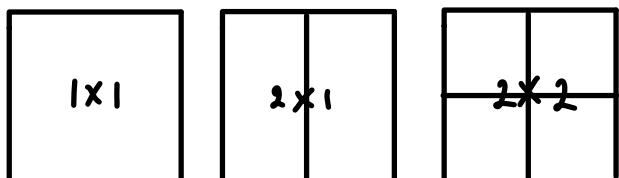
3. **재질 설정**: Material: Object 재질 고르기.

○ 순서: Create → Material로 재질 생성 후 편집해서 사용 (Object까지 재질 편집은 불가능)

- Albedo: RGB
- Metallic: 금속 재질수준
- Smoothness: 빛 반사수준

- Texture: 재질에 들어가는 이미지
- 그림 Object에 집어 넣기
- Albedo 옆 □에 그림 드래그해서 넣으면 됨

• Tiling: Texture 반복타일 개수 (X, Y)



4. 물리 재질 만들기 (Create → Physics Material)

• 단순화 마찰을 다른 물리적인 재질 [여기  
[넣으면 Collider Material이 들어감]

○ 설정 Bounciness: 탄성률 (0 ~ 1)

Bounciness Combine: 다음 탄성을 계산하는 방식

Average: bounce 이후 평균(평균) 값

Maximum: 최대값.

• Emission: Texture 발광(발기) 조절  
└ 빛이 물리적인 것은 X(자체 발광 X)

Friction: 마찰력 [ Dynamic : 움직임 중 마찰력  
Static : 정지 시 마찰력

Friction Combine: 다음 마찰력을 계산하는 방식

○ 물리값은 양식 Bounciness Combine: Max  
Friction Combine: Min

# 물리충돌 대비 - Collision VS Collider

① 끝이 훑힐때면 사각형 보호

네트워크 Script

public class ...

{

    MeshRenderer mesh;

    Material mat;

void Start()

{

    mesh = GetComponent<MeshRenderer>();

    mat = mesh.Material;

}

적어도 1번정도

놓는 위치

On이벤트

물리충돌이 시작할 때 호출되는 함수  
("충돌 정보 Class")

private void OnCollisionEnter(Collision collision)

    " Exit : 물리충돌을 끝날 때 호출함수,

    " Stay : " 지속 시 "

{

    if(collision.gameObject.name == "My Ball")

        mat.Color = new Color(기본값) (0,0,0);  
        <sup>회색</sup> Color(255,255,255)

}

private void OnCollisionExit(Collision collision)

{

    if(collision.gameObject.name == "My Ball")

        mat.Color = new Color(기본값) (1,1,1);

흰색

}

Tip. 사용 툴 경고 메시지

① Rendering Mode는  
Transparent

② 알파값 A값 조정

## ① Unity 이벤트

MyBall Script.cs ( 힘을 이용하여 물체 움직여보기 단원)

Collider 간의 충돌이벤트  
물체를 움직일 수 있음

private void OnTriggerStay(Collider other)

{

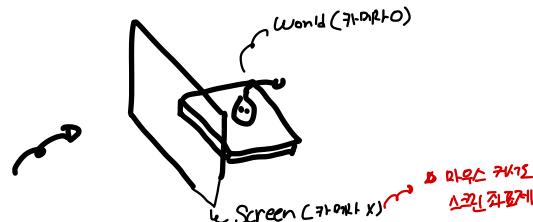
if (other.name == "Cube")

    rigid.AddForce(Vector3.up \* 2, ForceMode.Impulse);

# UGUI 7초

① Canvas : UI 가 그려지는 화면의 영역의 Component

② Screen : 게임의 표시되는 화면. 해상도로 크기 결정. 하얀 □



· Text : 문자열을 표시하는 UI

Font

Font : 사용 가능한 자판과 폰트 확인 (辈인의 폰트는 주제에 맞지 않음)

Horizontal Overflow : Wrap (감싸다. 허용 크기만 가능)

Vertical

Overflow (넘어갈 때) → but UI에는 드래그 가능을 허용할 경우에만

Line Spacing : 글자 간격을 조정

- 이미지 4종-

· Image : 이미지를 표시하는 UI

Texture Type : 원본 Default.  
이미지 파일을 Sprite로 설정하면 UI 편집 가능

① Simple : width, height (가로/세로) 고정 높이 넣는 것

② Sliced : 양 옆으로 잘라서 넣거나 혹은 가운데 잘라서 넣기 → Button이나 상자 툴

Image Type : ③ Tiled : 평면, 틀을 크기 만큼 넣어

④ Filled : Fill Method 방식에 따라 Fill Amount로 이미지 넣는 방식

Ex: Filled 기능으로 큐브에 텍스처 적용 가능 - 미리 그림 위에 넣은 그림 두께  
설정한 넣은 그림 Amount 0.5000  
설정 1.0

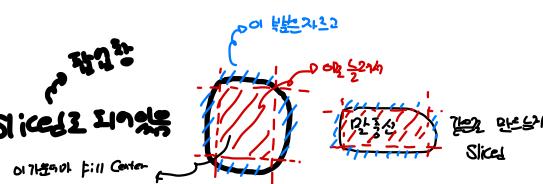
Preserve Aspect : 비율정

Set Native size : 원본사이즈로

TIP. 큐브는 허아라기  
사용할때는 면마다  
그림을 넣어야 한다.

· Button UI : ① Image

Image Type : Sliced로 표시됨



② Button (Script)

Interactable  : 누를 때/여기

Translation : 높을 낮기 : Color Tint : 221, 47, 147

State Swap

Animation

Normal Color: #32  
Highlighted: # : 0.050 원래 색상에  
Pressed: # : 0.054  
Disabled: # : 버튼 비활성화 [Interactable]  
Fade Duration: 색 바뀌는 시간  
Color Multiplier: 색 강도(0.0~1.0), 높을 1.0이면  
Navigation: Tab 키, 등록된 다른 버튼으로 이동 시킬 때, TAB, Shift 키로

Public void Button Function 사용 가능

Public void Jump()

{  
void.AddForce(Vector3.up \* 2, ForceMode.Impulse);  
}

OnClick(): 버튼 클릭시 힘으로 이동할 함수

Routine Only No function  
(None Selected)

③ Object (My ball) 풀링

④ My ball Script

void.AddForce(Vector3.up \* 2, ForceMode.Impulse);

}

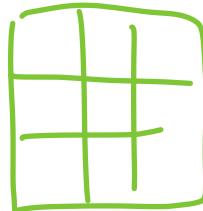
⑤ 가중치 Jump() 풀링 사용

# \* UI Screen 해상도 맞추기 - 고정

- UI의 Rect Transform 옵션



이거 누른다.



9개 둘

앵커 - 빨간점 : 컨테스에서 기준점



- Shift를 누르면 이미지의 고리점이 생김



앵커 - 고리점 : Component 이하의 기준점 → 빨 고리 둘다 하면 이미지를 둘 끌어

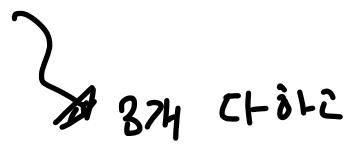


기준을 둘 끌어

으면 안이

- Alt를 누르면 중앙 네모 모양

앵커 - 중앙네모 : 커스텀의 위치



3개 다하고

Pos X, Y 수정하는가 위치

# 2D 프로젝트 준비하기

TIP: 게임 씬 내부 관리  
Gizmos

## 1. Project 설정 (2D)

## 2. 2D Object - Sprite

## 3. Render 순서

Camera - Projection : Orthographic : 원근법이 아닌 평행 技巧  
Perspective : 원근법 O

2D sprite 순서 설정.

1. Z정 사용

2. Order in Layer 사용 (같이 높을수록 위로 정렬)

3. Sorting Layer는 Layer 항목으로 만들어 순서지정



• 그래픽(프트) 번진기 : 사진 높으면 Filter mode에 Bilinear, Trilinear 는 그래픽을 압축시키는 품질.(잘보는).

그냥 Point3D 해

• !! 색 번진기 : !! Compression을 None으로  
(압축방식)

• 유니티 도트 1칸이 안맞음 : 사진, Pixels Per Unit : 1칸이 꽉찬 들어가는 개수. Unity 32x32면 32개면됨

## 4. 물리작용

(충돌 예측)

Box Collider 2D : • Collider 속성 : Project Setting → Physics 2D → Default Contact offset은 0으로 두기!

## Rigidbody 2D

# 2D 아틀라스와 애니메이션

## ① 아틀라스 (그림 끊임없는거)

Sprite Mode : Single : 풀체이드

Multiple : Sprite Editor로 자르고 싶다.



💡 왜 할까? 게임의 경우에 Batches 2번에 있는지  
Batch : 그룹을 묶어서 여러번 CPU를 사용한 흐름(초기화, 모비일스 풀식)

Tip - 2번에 위 RGB 누적  
0번 차례차 내용

TYPE - Automatic : 자동으로

Grid by CellSize : 같은 사이즈로 자르기 (설정하기)

" Count : 몇번 깨끗하게

Padding(여백)  
아틀라스에서 그림판 이미지

## ② 애니메이션

I. 2D Sprite를 Drag&Drop하면 파일이 생기는가, Asset을 Animation 파일 만들거나

자동화된 애니메이션 자동생성

[반복 x 1회 LoopTime]

II. Window → Animation창에서 조작

• Set as Layer Default State : 끌고다를 때

Bool, float 등 파라미터로 만들고

List에서 추가



etc

은 다음부분

# 2D 플레이어 이동구현

## ① 물리이동

```
public class PlayerMove : MonoBehaviour
```

```
{
```

```
    Rigidbody2D rigid;
```

```
    public float maxSpeed;
```

```
void Awake()
```

```
{
```

```
    rigid = GetComponent<Rigidbody2D>();
```

```
}
```

```
void FixedUpdate()
```

```
{ //move speed
```

```
    float h = Input.GetAxisRaw("Horizontal");
```

```
    rigid.AddForce(Vector2.right * h, ForceMode2D.Impulse);
```

```
//maxSpeed
```

```
    if(rigid.Velocity.x > maxSpeed)
```

```
        rigid.Velocity = new Vector2(maxSpeed, rigid.Velocity.y);
```

```
    if(rigid.Velocity.x < maxSpeed * (-1))
```

```
        rigid.Velocity = new Vector2(maxSpeed * (-1), rigid.Velocity.y);
```

```
}
```

```
}
```

## ● 오른 막길 마찰력을 없애기 (굴리갈 수 있게)

Physics Material 2D : 마찰력 (Friction) = 0



## ② 저항 설정 : Player Rigidbody

Linear Drag : 공기 저항, 이동 속도 (속도  $\downarrow$ ) -  $B_{air} \times \ln2$

Angular Drag : 회전, 보통은  $Touch \times$

프로세스 : Fixed Update() 쭉쓰기

단발적인 뉴트로는 Update() 쭉쓰기, 물리엔진 초기화 때문에 키가 쓰일수도 있어서

Void Update()  
} [우상 코드이 쓰기]

{

//Stop Speed

if (Input.GetButtonUp("Horizontal")) {

(단위 벡터, 물체의 위치를 초기화 상태)

Rigid.Velocity = new Vector2 (Rigid.Velocity.Normalized.X \* 0.5f

, Rigid.Velocity.y); // Button UP 시 빠르게 멈추기

○ Rigidbody - Constraints - Freeze Rotation Z키 : 오브젝트 회전 앤드는 옵션에서 2D에서 안쪽면 Player가 늘어짐

### ③ Animation

#### I. Sprite Renderer - Flip (스프라이트를 뒤집는 옵션)

전역 변수이 Sprite Renderer Sprite Renderer;

Awake() Sprite Renderer = GetComponent<Sprite Renderer>();

//Direction Sprite

Update() if (Input.GetButtonDown("Horizontal"))

Sprite Renderer. flipX = Input.GetAxisRaw("Horizontal") == -1;

#### II. Animator

Parameters: float, int, bool, Trigger

/ Inspector-창의 Transition Phases  
Conditions

줄이면 드물  
1 안쓰기  
V.V --

is Walking (bool)

OnAnimatorBool  
사용

(애니메이션 끝드릴때 .20s X)

Settings & Timeline 풀리는 구조 같아고

↓  
알맞은 Script 부기  
잘 끼여 놓기

Has Exit Time (애니메이션 끝난 때 까지 상태 유지)

↓  
풀리기 전까지 다음 State로 안 넘긴다

골드아워 걸기 모드 내에서만

전역 변수 Animator anim, Awake() 초기화

(수동 관찰 형식)

Mathf.Abs (rigid.velocity.X)  
(속도) < 0.3

Void Update() Sprite 멀기

if (rigidbody.velocity.normalized.X == 0)  
anim.SetBool("isWalking", False);

속도 절대값으로 애니메이션 관리

else

anim.SetBool("isWalking", True);

# 2D Player Jump

## 1. Jump

① ↴ 27H Player Move의 추가

- public float jumpPower;
- Update on

//Jump

if (Input.GetButtonDown("Jump"))

    rigid.AddForce (Vector2.up \* jumpPower, ForceMode2D.Impulse);

↓ 문제점: Jump는 AddForce, 중력으로 뛰어가는 것 위 공기저항 때문에 느려서 뛰어감

② Project Setting > Physic2D에서 중력값 설정 가능 ( 기본: 9.81 ) [ 전반적인 중력 ]

or

Rigidbody 2D의 Gravity Scale 조정 [ 오브젝트의 개별 중력 비율 ]  
[ 오브젝트에 적용되는 중력 비율 ]

## 2. Animation

1. Animation API : isJumping (bool) 사용하여 Jump Animation을 조정 등 ..

if (Input.GetButtonDown("Jump")) {

    rigid.AddForce (Vector2.up \* jumpPower, ForceMode2D.Impulse);

    anim.SetBool ("isJumping", true);

↓ 문제점

isJumping API 탈출X.

애니메이션 감속

해답: RayCast2D FixedUpdate에서 낙하 애니메이션 실행

### 3. Ray Cast : Object 경로를 위해 Ray를 쓰는 방식

방 쓰는 위치      방 방향  
[연습]

Void FixedUpdate() or Debug.DrawRay (rigid.Position, Vector3.down, new Color(0,1,0));  
(어디에 상어가 Ray를 그려주는지)

로 Ray 확인 가능(효과X 둘다만)

\* 실습 사용

Ray이 닿은 오브젝트

Void FixedUpdate() or RayCast Hit 2D rayHit = Physics2D.Raycast (rigid.Position,  
Vector3.down, 1);  
Ray가 물체가 있는 Ray를 초기화  
Ray가 사각형

if(rayHit.Collider != null){

Debug.Log(rayHit.Collider.name);

}

3  
Player 은 why?

Player Physics는 중앙점, Collider 원

3  
어디로 그쳐

Layer Mask : 물체효과를 구별하는 정수값 → 그중 우측 맨위 Layer

Layer와 Platform 만들기 (땅바닥 용)

rayHit의 Layer만 됨

RayHit = Physics2D.Raycast (rigid.Position, Vector3.down, 1, LayerMask.GetMask  
("Platform"));

이제 Layer  
정수값 return

// Landing Platform  
if (rigid. Velocity. Y < 0) {

    Void FixedUpdate() on RayCast Hit 2D rayHit = Physics2D.Raycast(rigid.Position,  
  Vector3.down,  
  1);  
    if (rayHit.Collider != null) {  
        Ray가 땅을 대신해  $\frac{1}{2}$  \*  $\frac{1}{2}$   
        if (rayHit.distance < 0.5f)  
            Anim.SetBool("isJumping", false);  
    }  
}

#### 4. 1단점조

점조 횟수 int 놓면 ... 있지만  
    ↳ 빠른 때로 안놓여도됨!

OPK 1단점조만 하는건



Jump 중이면 추가 점조

//Jump

if (Input.GetButtonDown("Jump") && !anim.GetBool("isJumping"))  
    추가.

TIP: Debug로 Ray 선

표시해보기!

# Tile Map 으로 Platform 만들기

## 1. 타일팔레트 & Tile Map

- ① Window → 2D → Tile Palette (타일을 사용하기 위해 모아둔 프리팹)
- ② 팔레트 창에서 Create New palette 를 Palette 프리팹 생성 (이름은 자유롭고, 나머지 Touch X)
- ③ 지형관련 Sprite 선택해서 Drag & Drop 해서 저장  
↳ Palette의 Edit 을 누르면 Palette Tile 추가 가능
- ④ Create → 2D Object → TileMap 을 누르면 Grid 가 나온다.  
자신이 TileMap이 있는게 타일을 일정해지 깔아주는 Component.
- ⑤ Palette 끄기.

## 2. 물리설정

- 결국 이 TileMap은 도와지 않음. 즉 TileMap이 전조기작용 Tile Map Collider 2D 가 아니라 이동 : Logic, Raycast이 자신...  
↓  
경사 Collider 가 이상하면
- Sprite Editor에서 편집
- ① Editor에서 Sprite Editor 를 Custom Physics Shape 를
  - ② Generate 를 Colider 를 만들며 수정 (delete로 삭제, 추가도 가능)
  - ③ Apply
  - ④ 기존 경사 팔레트 지우고 수정한 Sprite 가져옴

# 몬스터 AI 구현하기

## Enemy Move Script

```
Rigidbody2D rigid;
Animator anim;
SpriteRenderer spriteRenderer;

public int nextMove; // 행동지표를 결정할 변수

void Awake()
{
    rigid = GetComponent<Rigidbody2D>();
    anim = GetComponent<Animator>();
    spriteRenderer = GetComponent<SpriteRenderer>();
    Invoke("Think", 5);
}

void FixedUpdate()
{
    // move
    rigid.velocity = new Vector2(nextMove, rigid.velocity.y);

    // 전방 낭비하지 않음
    Vector2 frontVec = new Vector2(rigid.position.x + nextMove, rigid.position.y);
    RayCastHit2D rayHit = Physics2D.Raycast(frontVec, Vector3.down, 1, LayerMask.GetMask("Platform"));

    if (rayHit.collider == null)
        Turn();
}

}
}
```

// 행동지침을 바꾸는 함수

Void Think()  
{  
 nextMove = Random.Range(-1, 2);  
 // (-1 ~ 1) 사이의 정수  
}

float nextThinkTime = Random.Range(2f, 5f); // 2~5초의 시간 범위

Invoke("Think", nextThinkTime); → 자동 + 딜레이.

anim.SetInteger("WalkSpeed", nextMove); // Bool이거나 int3 anim값  
if(nextMove != 0)  
 spriteRenderer.flipX = nextMove == 1;

}

Void Turn()

{  
 nextMove \*= -1;  
 spriteRenderer.flipX = nextMove == 1;  
}

CancelInvoke(); [현재 작동하는 모든 Invoke를 멈춰놓기]  
Invoke("Think", 5);

}

# 플레이어 이벤트 Event 구현

: 몬스터, 농장 등 인물

## 1. 향정 맵 추가

- ① 가시밭장을 위한 TileMap을 Grid로 하나 더 만들기 + 배치
- ② 주인 대신 Tag와 Layer를 (Tag, Layer - Enemy) 가시밭장, 몬스터 2개. 누가 충돌 했느냐는 tag가 중요
- ③ Player는 Layer 2개 사용 예상 - Player, Player Damage

## 2. Layer 설정

- ① Edit → Project Settings의 Physics2D의 맨아래  → 여기 있는데 Layer간 충돌 유/무 설정.
- ② Player가 대적지 입체화 때 잠시 무적상태로

해주기도 좋다.



Player Damage 일 때는 Enemy 충돌 X로 무적화

## 3. 무적시간

### PlayerMove Script

```
private void OnCollisionEnter2D(Collision2D collision)
```

{      if gameObject 담은 물체의 tag가 Enemy면

```
if (Collision.gameObject.tag == "Enemy")  
    OnDamaged(collision.transform.position);
```

}

충돌한 놈 위치



// 무적화 함수 생성

```
void OnDamaged(Vector2 targetPos)
```

{

```
    gameObject.layer = 11;
```

R G B F

```
    SpriteRenderer.color = new Color(1, 1, 1, 0.4f); // 투명하게 만들어서 막으라
```

```
    int dinc = transform.position.x - targetPos.x > 0 ? 1 : -1; // 막은 뒤 백설과 방향
```

```
    rigid.AddForce(new Vector2(dinc, 1) * 7, ForceMode2D.Impulse); // 끌어
```

```
    anim.SetBool("do Damaged"); // 5. Animation을 트리거
```

```
    Invoke("OffDamaged", 3);
```

}

## 4. 부작 애니

```
Void OffDamaged ()  
{  
    gameObject.layer = 10;  
    spriteRenderer.color = new Color(1,1,1,1);  
}
```

## 5. 애니메이션

① 점프 애니메이션 재활용 : 코루틴 Sprite Damaged 쓰기

- Alt + 마우스 휠 : Animator ZoomIn, ZoomOut (애니메이션 확장)

- Trigger 카드미터 추가 // Trigger : 반복적 액션의 마지막번, 값이 없다는 것이 특징

- AnyState



Damaged : onanim DoDamaged Trigger 추가. Has Exit Time ✕ (OneShot(나만 적용))  
LoopTime ✗



Exit

- AnyState → Exit : 현재상태 상관없이 실행 후 끝내기

# 쭈끄족 같은 학다운 RPG 이렇게 준비하기

## 1. 풀어그인 설치

RoleTile : 규칙을 정할 수 있는 타일 : 유니티 자체 제공 X. 따로 가져와야 함.

① GitHub 2D Extra 검색 → GitHub 클릭

② Releases 클릭 → 유니티 버전 Zip 다운로드  
(배포하는 뜻)

③ 앱축풀기 → Assets 드래그



## 2. Role Tile

① Create 가 더 많아짐. Tile 보면 많아졌는데 Rule Tile 생김

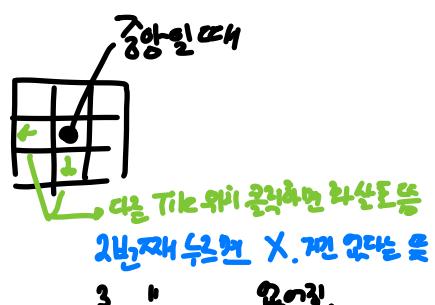
② Rule Tile Inspector창 보면 기본 Sprite 설정해주고,

Tiling Rules 앤드 아래 + 버튼으로 블 타일 추가 (영상에선 3x3 → 9개 됨.)

→ 각 방향 Sprite를 Rule Tile의 지정

후 팔레트에 넣음

③ 스크레이프 모양이 맞게 방향규칙 설정. 자기 Sprite가



TIP: 타일 맵 2개 이상 작업할 때는

Focus On 기능 활용

[현재 그린 Tile 배경 오른 흑/백 등]

### 3. 오토마이션 타일

- ① Rule Tile 생성
- ② Output을 Animation으로 설정
- ③ Size 조작2 Sprite 놓으면 끝

### 4. 외부정제 (쪽지 맵 밖으로 안나가게)

- Colider 추가 안보이게, 일정한 Composite 쪽에 안보이게  
Composite 두드려됨
- ① TileMap 생성, Colider 추가 → 타일맵 출판에는 복합 콜라이더(Composite Collider2D)와 혼용 Good  
(Used by Composite)

Rigid body 2D - Simulated ✕

- ② 타일을 맵 외부에 끌고, 타일맵 편집기에 Mask Interaction을 Mask 안보이게 있음 빛이지가 되는지  
Mask가 없으니 그냥 안보임 드렁抨

### 5. 디자인 퍼펙트 - 폭발 깨끗이 선 생겼을 때

- ① Window → Package Manager (기본 외 플러그인 패키지 관리창)

- ② 윈쪽 Advanced → Show preview packages  
[내다보면 패키지까지 보겠다]

- ③ 윈쪽 검색 2D Pixel Perfect 등의 생김. Install → 2019.2 Ver은 도메지가 아니라  
기본 기능으로 들어간거 같은  
폭발 깨끗이 화려하게 처리해주는 것이라 끝났어

- ④ Main Toolbar Pixel Perfect Component 추가 + Pixel per Unit 해상도 맞춤.

# 쓰글리식 액션 구현하기

1. 쓰글리 스크립트 위한 대체선 이동 구조. 십자 이동

Player Action Script

{

```
public float Speed;  
Rigidbody2D rigid;  
float h;  
float v;  
bool isHorizonMove;
```

void Awake()

{

```
    rigid = GetComponent<Rigidbody2D>();
```

}

void Update()

```
{  
    h = Input.GetAxisRaw("Horizontal");  
    v = Input.GetAxisRaw("Vertical");  
    bool hDown = Input.GetButtonDown("Horizontal"); T  
    bool vDown = Input.GetButtonDown("Vertical");  
    bool hUp = Input.GetButtonUp("Horizontal"); Check Button  
    bool vUp = Input.GetButtonUp("Vertical"); J
```

if (hDown || vUp) T

isHorizonMove = true; Check Horizontal Move

else if (vDown || hUp)

isHorizonMove = false; J

}

void FixedUpdate()

```
{  
    Vector2 moveVec = isHorizonMove ? new Vector2(h, 0) : new Vector2(0, v); ← 십자 이동  
    rigid.velocity = moveVec * Speed;
```

}

## 2. OHUD가 아닌

방향변화 transition 한 번만 실행되니

① 4축, 4동 같은 블록 int 차원에서, OHUD에 이전용 bool 사용해라

② OHUD의 Settings의 Transition Duration = 0.0f (없애기)

③ Any State → Player\_

Up  
Down  
Left  
Right



\_Walk → Player\_ : \_Idle

상수처럼 이동하면 OHUD로 → 이동끝나면 그 방향 Idle



④ UpdateOn if (anim.GetInteger("hAxisRaw") != h) {

    anim.SetBool("isChange", true);     // 명시적 형변환 안좋아하므로 구현 외

    anim.SetInt("hAxisRaw"), (int) h);

}

cde if (anim.GetInteger("vAxisRaw") != v) {

    anim.SetBool("isChange", false);

    anim.SetInt("vAxisRaw"), (int) v);

}

Transition을 연속적으로 허용할 OHUD에만이 작동X → 여기서 OHUD는 동시작동

↓  
⇒ 2번 ←→ 한꺼번에 누르면 문제!

Update Check Horizontal Move를

if (hDown)

    isHorizonMove = true;

else if (vDown)

    473

    isHorizonMove = false;

else if (hUP || vUP)

    isHorizonMove = h != 0;

### 3. 조사 액션 (상자 등...): Ray3

① 전역변수나 내가 바라보는 방향을 Vector3 dirVec;  
조사용 GameObject Scan Object

#### ② Update()

```
//Direction  
if (vDown && v == 1)  
    dirVec = Vector3.up;  
  
else if (vDown && v == -1)  
    dirVec = Vector3.down;  
  
else if (hDown && h == -1)  
    dirVec = Vector3.left;  
  
else if (hDown && h == 1)  
    dirVec = Vector3.right;  
  
//ScanObject  
if (Input.GetButtonDown("Jump") && ScanObject != null)  
    Debug.Log("this is :" + ScanObject.name);
```

#### Void FixedUpdate()

##### { //Move

```
Vector2 moveVec = isHorizontal ? new Vector2(h, 0) : new Vector2(0, v);  
rigid.velocity = moveVec * speed;
```

##### //Ray

```
Debug.DrawRay(rigid.position, dirVec + 0.7f, new Color(0, 1, 0));
```

```
RaycastHit2D rayHit = Physics2D.Raycast(rigid.position, dirVec, 0.7f,
```

지나가는 Object  $\hookrightarrow$  LayerMask.GetMask  
( "Object" ));

같이 있다 = 사용하겠다

LayerMask

↑

```
if (rayHit.collider != null) {
```

ScanObject = rayHit.collider.gameObject;  $\rightarrow$  Raycast가 오브젝트를 뒤져보기 때문에 찾을

}

```
else
```

ScanObject = null;

}

# 대화창 UI 구축하기 [단순 Ver.]

## 1. 대화창 UI

### ① UI 애니 모드 Canvas - Pixel Perfect

② 만들기 높은 정사각형 대화창을 3 ½ 틀 : Image Type Sliced

7. Sprite와 위치를 외우고 블록은 Border 값 입력 (변화 X 부분). Editor Apply  
[Left, Top, Right, Bottom]

L. Image Type - Sliced → 크기 변경으로 인한 딕시를 개정 해방

③ Anchor을 alt 키로 조작. (alt 키) 누르면 주변 화살표가 막기 때문에 놓침

\* L을 치면 Transform의 Left, Right로 바뀌는데 예쁘다.

TIP: Anchor와 Shift 누르면 Image의 기준점이 조정 가능. Image 자간색 온에 기준점

L을 예쁘게 만들어 사용할 수 있다 대화창 만들기

### ④ Image 자식으로 Text.(크기전 무로 font)

L. Text는 Anchor로 처리 + 텍스트 높이보면서 조절

## 2. Data 전달 → 대화창을 재활용하기 좋음. 데이터 전달을 위한 PlayerPrefs 사용

① GameManager 클래스도 만들고 C# Script 쓰기.

### Game Manager Script

```
using UnityEngine;
using UnityEngine.UI;
public class GameManager : MonoBehaviour
{
    public Text talkText; // 대화창
    public GameObject scanObject; // scandit OB
```

```
public void Action(GameObject scanObj)
{
    scanObject = ScanObj;
    talkText.text = scanObject.name + "를 찾았."
}
```

### - Player Action Script

```
public GameManager manger;
```

### Update 부분

```
//ScanObject
if (Input.GetButtonDown("Jump") 위에 끌어내리기 && scanObject != null)
    manger.Action(scanObject);
```

② Script : public 키가 GameManager의 텍스트, PlayerAction은 GameManager

### 3. 상태 전환 : 카메라 움직이기

#### 01 카메라 움직이기

```
public GameObject talkPanel;
public Text talkText;
public GameObject scanObject;
public bool isAction
```

```
public void Action()
{
    if (isAction) { //Enter Action
        isAction = false;
    }
    else { //Exit Action
        isAction = true;
        talkPanel.SetActive(true);
        scanObject = ScanObj;
        talkText.text = scanObject.name + "를 찾았.";
    }
    talkPanel.SetActive(isAction);
}
```

## — Change Player Position

Player Action Script on Update Move 관련 코드

// Move Value

h = manager.isAction ? 0 : Input.GetAxisRaw("Horizontal");

v = manager.isAction ? 0 : Input.GetAxisRaw("Vertical");

bool hDown = manager.isAction ? false : Input.nu

: vDown, hUp, vUp 등 키값이

## 4. OHU 디자인

① Image 만든다. 그 위에 이미지 만들고 그 위에 (→ 회색으로)

→ 원본 이미지 Set Native Size, Apply, Scale 조정이 있다

② AnchorZ 위치조정 (위로) → Animation

③ 이미지에 Animator 끌고, Animation 켜기 Create - Animation Controller 만들기 (영상과 End Cursor)

Animation 만들기 (영상과 Cursor Move)

④ Animation Add Property 끌고 Anchored Position 선택  
(Anchor 설정된 UI)

⑤ Y축 조정하기 ▶ ▲ ▾ 투명 → 다른 애니메이션  
Loop Time 설정하는지?

# 대화 시스템 구현

## 1. 오브젝트 관리

① Create C# Script : Obj Data → 모든 Obj이 개체로 같은 느낌?

```
public class ObjData : MonoBehaviour  
{  
    public int id; → 각 Obj마다 고유한 ID로 1000 ~ 2000까지 NPC, 2000 ~ 4000은 물체 ...  
    public bool isNPC;
```

## 2. 대화 시스템

① Create TalkManager : Obj Data 관리하는 Script + Obj

```
public class TalkManager : MonoBehaviour  
{  
    Dictionary<int, string[]> talkData;  
}
```

```
void Awake()  
{  
    talkData = new Dictionary<int, string[]>();  
    GenerateData();  
  
}  
  
void GenerateData()  
{  
    NPC_ID  
    talkData.Add(1000, new string[] {"안녕?", "이곳에 처음 왔구나?"});  
    talkData.Add(2000, new string[] {"우?", "아니면 거기다."});  
    talkData.Add(4000, new string[] {"정말로 나육상자인."});  
}
```

```
String[] talkData[int id, int talkIndex]  
{  
    return talkData[id][talkIndex];  
}
```

}

## ② Game Manager(액션)

1. CHAT하기 2. CHAT하기 다른나라 액션이 끝나기

### Game Manager Script

using UnityEngine;

using UnityEngine.UI;

public class Game Manager : MonoBehavior

{

    public Talk Manager talkManager;

    public GameObject talkPanel;

    public Text talkText;

    public GameObject scanObject;

    public bool isAction;

    public int talkIndex;

    public void Action(m)

    {  
        scanObject = scanObj;

        ObjData objData = scanObject.GetComponent<ObjData>();

        TalkObjData.id, objData.isNPC);

        talkPanel.SetActive(isAction);

}

void Talk(int id, bool isNPC)

{

    String talkData = talkManager.GetTalk(id, talkIndex);

    if (talkData == null) {

        isAction = false;

        talkIndex = 0;

        return; // 허술한 대화문이 있으면

        즉, voidalk return;을

    }

    강제종료역할

    if (isNPC) {

        talkText.Text = talkData;

    }

    else {

        talkText.Text = talkData;

    }

    isAction = true;

    talkIndex++;

}

## ③ CHAT하기 Action, TalkManager

String[] talkData Index

public String GetTalk(int id, int talkIndex)

{  
    if (talkIndex == talkData[id].Length)

        return null;

    else  
        //聊天 Get → talkIndex의 친구는 talkData[id][talkIndex];

        return talkData[id][talkIndex];

}

### 3. 초상화 : 대화시 대화창 얼굴

① 초상화 Sprite 가져와서 정의 (오기, 자크 등)

② 초상화를 보여줄 TalkSet(대화창) 지속으로 Image UI 생성

③ Image UI 를 통해 번역 생성 & 홀딩

7.炳기 : GameManager.on Public Image PortraitImg;

L. 할당 [GameManager]

if(isNPC){

    talkText.Text = talkData;

    PortraitImg.Color = new Color(1,1,1,1); : NPC 일때만 이색이가 난다기

}

else{

    talkText.Text = talkData;

    PortraitImg.Color = new Color(1,1,1,0); : NPC 일때만 이색이가 난다기

}

### ④ Sprite(포장) 번역을 정정

7. Talk Manager

Dictionary<int, Sprite> PortraitData; // 초기 Sprite를 추가  
Public Sprite[] PortraitArr; // 초상화 sprite의 번역을 정정한 번역 생성. Component의 Size 72mm 45mm

Void Awake() on

PortraitData = new Dictionary<int, Sprite>(); // 초기화

Void GenerateData() onAwake 추가

포장.37H

(그림에 맞는 포장 번호)

PortraitData.Add(1000+0, PortraitArr[0]);  
PortraitData.Add(1000+1, PortraitArr[1]);  
PortraitData.Add(1000+2, PortraitArr[2]);

// 마지막 초상화 sprite의 번역을 번역 생성

Public Sprite GetPortrait(int id, int PortraitIndex)

{

    Return PortraitData[id + PortraitIndex];

}

Portrait Index는 CH와 같은 숫자가

Talk Manager의 기준 CH와

```
Void GenerateData()
{
    NPC_ID
    talkData.Add(1000, new String[] {"안녕?", "이 곳이 처음 찾구나?"});
    talkData.Add(100, new String[] {"경험한 나무상자다."});
}
```

3

"안녕?:0", "이 곳이 처음 찾구나?:1"

:은/는 경우

대화의 시작

안들어가는)

아니거나

초상화 구별용

4

L. Game Manager로 돌아 와서 오류 수정.

if(isNPC){

{ 구별자를 통하여 뒤에 있는 나누주는 문자를 참조

talkText.Text = talkData.Split(':')[0]; → CHAT : 텍스트

PortraitImg.Sprite = talkManager.GetPortrait(id, int.Parse(talkData.Split(':')[1]));

PortraitImg.Color = new Color(1,1,1,1);

Parse(): 문자열 해당 부분으로 나누주는 함수

3

else{

talkText.Text = talkData;

PortraitImg.Color = new Color(1,1,1,0);

3

# RPG 쿼스트 시스템

## 1. 쿼스트 DB

### ① 쿼스트 DB(오브젝트 & 스크립트) 생성

```
Public Class QuestManager : MonoBehaviour
```

```
{
```

```
    Public int questId; // 진행중인 쿼스트 ID
```

QuestID ↗ 구조체

```
Dictionary<int, QuestData> questList;
```

// 쿼스트 데이터 저장용 Dictionary 변수

```
Void Awake()
```

```
{
```

```
    questList = new Dictionary<int, QuestData>();
```

```
    GenerateData();
```

```
}
```

```
Void GenerateData()
```

```
{
```

```
    questList.Add(10, new QuestData("인디언족", new int[] {1000, 2000}));
```

```
}
```

↑ NPC ID를 빙고 쿼스트 번호를 반환

```
Public int GetQuestTalkIndex(int id)
```

```
{
```

```
    Return questId;
```

```
}
```

오브젝트가 놓여있을 때마다 네트워크에 ~~Mark~~ Rel-

①-2 : 쿼스트 Data Script 작성

```
Public Class QuestData
```

```
{
```

```
    Public String questName;
```

```
    Public int[] npcId; // 관련 NPC
```

↑ 구조체 생성을 위해 대개 변수 생성자를 작성

```
    Public QuestData(String name, int[] npc)
```

```
{
```

```
        questName = name;
```

```
        npcId = npc;
```

```
}
```

## ② Game Manager 4장: A. 쿼스트 ID

public QuestManager questManager; // 퀘스트 생성, 쿼스트 번호(ID) 관리

void Talk(int id, bool isNPC) 함수

int questTalkIndex = questManager.GetQuestTalkIndex(id); // NPC ID를 대체해주기 추가  
String talkData = talkManager.GetTalk(id + questTalkIndex, talkIndex); // NPCID + 쿼스트 번호 = 쿼스트 대화 GIVE\_ID

## 2. 쿼스트 진행

### ③ Talk Manager 4장

void GenerateData()의 틀을 대체 한 일

talkData.Add(10 + 1000, new String[] { "아무말:0", "이미 물어 전화가 있음:1", "42 가족:2" });  
talkData.Add(10 + 2000, new String[] { "무엇?:0", "아, 놀라운 전화?:1", "일은 왜 주연 알려줄까:2" });  
↓  
구현: NPC 퀘스트 번호와 함께 진행

### ④ 쿼스트 순서 정하기 - QuestManager

A. public int questActionIndex; // 쿼스트 대화 순서 번호 생성

B. GetQuestTalkIndex(int id)

{  
return questId + questActionIndex; // 추가 쿼스트 번호 + 쿼스트 대화 순서 = 쿼스트 대화 ID  
}

C. 대화전용을 위해 쿼스트 대화 순서를 옮기는 함수 생성

public void CheckQuest()

{

questActionIndex++;

}

D. 대화가 끝난 때 questActionIndex += 2로 바꾸기

GameManager의 void Talk의 talkIndex = 0;

문제

questManager.CheckQuest(); 추가

↓

\* 쿼스트 ID 번호의

talkData.Add(10 + 1000, ...)

talkData.Add(10 + 2000, ...)  
10 112 change

E. 순서에 맞게 대화를 때면 쿠스트 대화순서를 움직임을 작성

```
Public void CheckQuest (int id)
{
    if (id == questList [questId].npcId [questActionIndex])
        questActionIndex++;
}
```

F. 쿠스트 2개 만들고 향상 : 멀티Q

- QuestManager의 GenerateData에서  
questList.Add(10, new QuestData("대화형Quest", new int[] {1000, 2000}));  
questList.Add(20, new QuestData("동전찾아주기", new int[] {5000, 2000}));

• void NextQuest ()

```
{  
    questId += 10;  
    questActionIndex = 0; // 기존 쿠스트 초기화  
}
```

• 수정

```
Public void CheckQuest (int id)
{
    if (id == questList [questId].npcId [questActionIndex])
        questActionIndex++;

    if (questActionIndex == questList [questId].npcId .Length) // 쿠스트 대화순서가 끝나면  
        NextQuest(); // 쿠스트 번호 증가
}
```

### 3. 쿼이스트 오브젝트

① 쿼스트 번호 20·중간단계로 등록 Obj 추가. Obj Data는 초기 - 단, 동전은 안 뜨기. 쿼스트 번호로 뜨여지

#### ② Quest Manager.cs

• Public GameObject[] questObject; // 쿼스트 Obj(중전 등)을 저장할 변수 생성  
    └ Components.cs size 7012

• 쿼스트 Obj를 관리할 함수 생성

Void ControlObject()

{

    Switch (questID) {

        Case 10:

            if (questActionIndex == 2)  
                questObject[0].SetActive (true); // 대화 끝나면 ON

            break;

        Case 20:

            if (questActionIndex == 1)

                questObject[0].SetActive (false); // 대화 시작하면 OFF

            break;

    }

}

↓ Control Object 언제 실행?

• CheckQuest(), 대화가 끝난 후에 실행

Public Void CheckQuest (int id)

{

    // Next Talk Target:

    if (id == questList [questId].npcId [questActionIndex])

        questActionIndex ++;

    // Control Quest Object

    ControlObject();

    // Talk Complete & Next Quest

    if (questActionIndex == questList [questId].npcId .Length) // 쿼스트 대화 수가 끝나면 다음 대화

        NextQuest(); // 쿼스트 번호 증가

## • 20번 중간찾기 쿠스트 Text(중)

Talk Manager의

talkData.(20 + 1000, new String[] { "중간찾아줘!: ", "한수역 극도의 중간여야:", " } );

talkData.(20 + 2000, new String[] { "한수역 꽃 줄 가져다 주!: ", " } );

talkData.(20 + 5000, new String[] { "한수역 중간을 찾았다. ", " } );

↳ 중간 Obi

talkData.(21 + 2000, new String[] { "와! 고마워 !! : ", " } );

△ Quest 퀘스팅 : 대화 Q → 중간찾아줘 Q → Error: 쿠스트 오류. Array 초과

• Quest Manager의

QuestList.Add(30, new QuestData("퀘스트 을 물어보기", new int[] { 0 }));

Alg. [ 101Q . Error 발생 예상, 대화한 대로 ]

↓ 21 쿠스트는 1000번이 물어보면 Error. 왜? 관계자가 다른데

4. 예외처리 : ↗ 224번 예외처리

① TalkManager의 GetTalkById, 관리자 있는지

Get Talk 가장 앞이

다시 브레이크가 있는지

if( ! talkData.ContainsKey(id) ) { // 쿠스트가 이후 대사X

if( ! talkData.ContainsKey(id - id%10) ) { // 이전이 쿠스트 관계X. 쿠스트 관계 대사 전부

if( talkIndex == talkData[id - id%10].Length )

return null;

// 쿠스트 번호를 대사마지 않은 때 (사용 등)

else

return talkData[id - id%10][talkIndex];

↳ 쿠스트 번호까지 처리

3

else { // 224번 쿠스트 대사가 대사가 있

if( talkIndex == talkData[id - id%10].Length )

return null;

0[7] 1 4000번

else

return talkData[id - id%10][talkIndex];

1021 - 1 = 1020 : id가 10으로 쿠스트 대사에서 처리 후 재생성

책상로 대사 등

## 5. 로직 다듬기 : 27장 35문

: 내용은 API GI  
개별자 → User 간의  
고민으로 ↑  
자기에게 시간↑? 등

- GetTalk() 꽂아 Logic은 같은데 값이 다름. 자세히

//: 수정

```
public String GetTalk(int id, int talkIndex)
```

```
{ if( ! talkData.containsKey( id ) ) {
```

```
    if( ! talkData.containsKey( id - id%10 ) ) {
```

```
        if( talkIndex == talkData[id - id%10].Length )  
            return null;
```

```
    else
```

```
        return talkData[id - id%10][talkIndex];
```

```
    }
```

반환 값이 있는지 체크하는 걸  
가져KnockOut.

```
{
```

→ Return GetTalk(id - id%  
10, talkIndex);

//Get First Talk

```
else {
```

```
    if( talkIndex == talkData[id - id%10].Length )  
        return null;
```

```
    else
```

```
        return talkData[id - id%10][talkIndex];
```

→ Return GetTalk(id - id%  
10, talkIndex);

//Get First Quest Talk

```
3
```

```
if( talkIndex == talkData[id].Length )  
    return null;
```

Basic Logic

```
else
```

```
    return talkData[id][talkIndex];
```

```
3
```

```
3
```

6. 쿼스트 ID를 입력하기 → 콘솔창에 quest ID를 쓰는가. 위에는 내용 X. 관련 내용 없음

## ① Game Manager.cs

Void Start()

{  
    questManager.CheckQuest();

}

을 추가. → DTMF 번수는?

DTMF 번수가 있는 CheckQuest(); 상/하

↓

Public String CheckQuest();  
{

    return questList[questId].QuestName; (길이를 but DTMF 번수에 따라 길이)

}

Console 창 확인 필요 없다. 상자박스

안녕 있음.

기존 DTMF 번수 int id 있으면서 DTMF 번수 없어

함수 몇개 필요하거나 남김

Overloading : DTMF 번수에 따라 함수  
호출하는 기술

# CH&L OH4 UI Effect 느낀점

## 1. CH&L Effect

① Animator 속성 사용하여 [CH&L UI]의 허가  
Bool ? b. isShow 상태

② Animation 2DH 사용하여 Animator ? Show

③ Window → Animation 창 열고, 대화 UI 를 찾았을 때까지 이동 (안드로이드)

All Property → Rect Transform → Anchored Position 속성  
y축만 변경 : CH&L UI 위, 아래 이동  
Show: ↑, hide: ↓ 이동



④ Game Manager이 기준 창은 Action의 talkPanel.SetActive(isAction);  
으로 추가함.



I. Public Animator talkPanel

II. 기준 Set Active(isAction) 노출로 talkPanel.SetActive("isShow", isAction);

2. 초상화 Effect : 표정 변화하면 초상화가 움직임. OODL는



Entry

Empty

Has exit time ↓ ↑ Has Exit Time

PortraitEffect

기본적으로 솔루션 인해도  
Animation 끝나면  
자동으로 돌아감

① Animator, Animation | 개별 초상화 Image의 상태

Trigger ? b. doEffect 실행

② Anchored Position - y축 조정. OODL은 2D로 3D. ↑ down, up

③ Game Manager Public Animator portraitAnim;

OBJ Sprite의 경우에 A.S. → def Talk on M

→ ODE Sprite 재정, 뒤로하고  
Anim 실행

I. Public Sprite PrevPortrait 속성

I. if (PrevPortrait != PortraitImg.Sprite)

PortraitAnim.SetActive("doEffect");  
PrevPortrait = PortraitImg.Sprite

### 3. El01 티피 Effect

① C# Script - TypeEffect . C#: Text의 편집

Class TypeEffect

using UnityEngine.UI;

```
Public int CharPerSeconds; // 글자 지속 시간을 변수  
Public GameObject EndCursor; // Effecting End. cursor는 글자 끝에 표시되는 커서  
String targetMsg; // 표시할 대화문장을 변수 저장  
Text msgText;  
int index;  
float interval;  
  
Private void Awake()  
{  
    msgText = GetComponent<Text>(); // GetComponent<Text>() // GetComponent<Text>()  
}  
  
Public void SetMsg(String msg) // 대화문을 변수로 받는다  
{  
    targetMsg = msg;  
    EffectStart();  
}  
}  
// 시작 - 지속 - 종료  
void EffectStart()  
{  
    msgText.text = " "; // 공백으로 초기화  
    index = 0;  
    EndCursor.SetActive(false);  
  
    // Start Animation  
    interval = 1.0f / CharPerSecond;  
    Invoke("Effecting", interval);  
}
```

1글자가 나오는 것

```

Void Effecting()
{
    if(msgText.text == targetMsg)
    {
        EffectEnd();
        return;
    }

    msgText.text += targetMsg[index];
    index++;
}

Invoke("Effecting", interval);
}

Void EffectEnd()
{
    EndCursor.SetActive(true);
}

```

## (2) Game Manager 4<sup>章</sup>

: GM에서 Talk을 ese - talkText.text = talkData; 한다



(Script)

I. Public TypeEffect talk; // 가진 talkText

### II. Talk 설정

if (isNPC)
 talk.SetMsg(talkData.Split(':')[0]);

o Portrait Sprite, Main

else:
 talk.SetMsg(talkData);

### ③ Text Sound, Chat Skip 막기

I. Text on AudioSource 속성 : 헤더에 있는 Asset의 DM-Cos-21.mp3.  
Play on Awake

#### II. Type Effect의 흡

- AudioSource audioSource;
- Awake() on AudioSource = GetComponent< AudioSource >();
- Void Effecting() // 자주 쓰는거면 Play()

##### Void Effecting()

```
{ if (msgText.text == targetMsg) {  
    EffectEnd();  
    return;  
}
```

msgText.text += targetMsg[index];

• 음성 출력, .은 시운드 제작

if (targetMsg[index] != ' ' || targetMsg[index] != '.')  
 AudioSource.Play();

index ++;

Invoke("Effecting", interval);

}

Skip Dlg : Chat에 대화 중에는 다른 대화가 걸리지 않는지

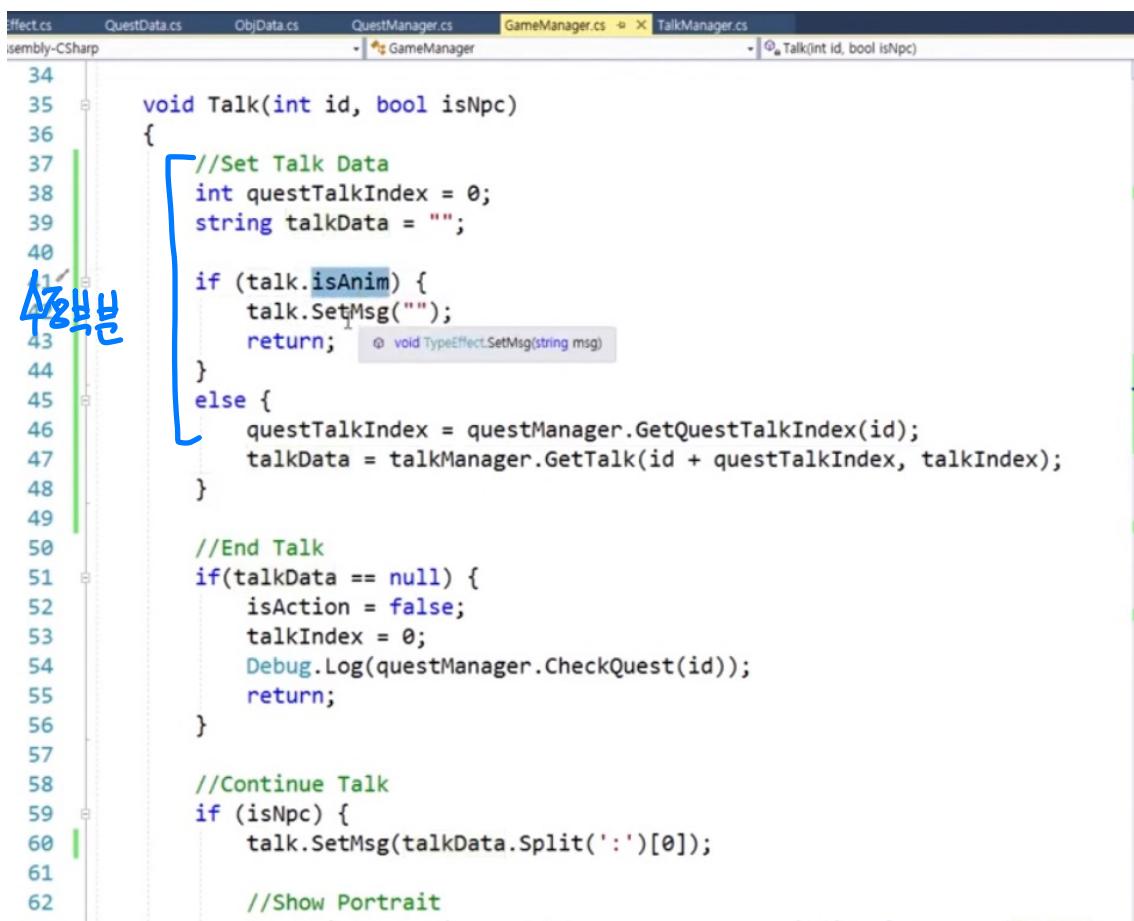
I. Public bool isAnimation; //

Text Effect  
II. Effect Start() when isAnimation = true;  
Effect End() when isAnimation = false;

III. Set Msg on 맨앞에 추가

```
if (isAnimation) {  
    MSGText.text = targetMsg; →  
    CancelInvoke();  
    EffectEnd();  
}  
else {  
    원래 있던  
    }  
}
```

## IV. GameManager 7/16



```
34
35     void Talk(int id, bool isNpc)
36     {
37         //Set Talk Data
38         int questTalkIndex = 0;
39         string talkData = "";
40
41         if (talk.isAnim) {
42             talk.SetMsg("");
43             return;    @ void TypeEffect.SetMsg(string msg)
44         }
45         else {
46             questTalkIndex = questManager.GetQuestTalkIndex(id);
47             talkData = talkManager.GetTalk(id + questTalkIndex, talkIndex);
48         }
49
50         //End Talk
51         if(talkData == null) {
52             isAction = false;
53             talkIndex = 0;
54             Debug.Log(questManager.CheckQuest(id));
55             return;
56         }
57
58         //Continue Talk
59         if (isNpc) {
60             talk.SetMsg(talkData.Split(':')[0]);
61
62             //Show Portrait
```

# 서브메뉴 만들기 + 저장기능

## 1. UI 구축

- ① Canvas → UI. Image 추가 → Anchor 가 아니라 가로형으로 크게 설정 + 투명한 경계색으로 설정
- ② 자식으로 버튼을 감싸줄 이미지 생성. 버튼은 생성하고 적절히 모양잡으면 완료
- ③ 제작하기, 저장하기, 종료하기 + 전통화면 쿼드로 표시. 총 4개 이미지를 결정



## 2. 제작하기

### ① Game Manager

```
public GameObject menuSet;
```

```
void Update()
{
    //Sub Menu
    if (Input.GetButtonDown("Cancel"))
    {
        if (menuSet.activeSelf)
            menuSet.SetActive(false);
    }
    else
        menuSet.SetActive(true);
}
```

↑ 이런 단축키는 Input에 해당.  
↑ Project Setting에 있는지 확인. Escape가 ESC

↑ UI 구조 ① 번 obj

- ② 제작하기 버튼에 가서 On Click() 추가해 Menu Set 끝에 No Function 을 누르면  
Game Object 뒷면에는 Inspector창에서 바로 할당 가능  
여기선 SetActive 끝에 완료.

### 3. 쿼이스트 확인

① GameManager의 QuestText 할당 : Public Text questTalk;

Void Start()

{  
    questTalk.text = questManager.CheckQuest(); // 추가 : 쿼이스트 내용 전달  
}

Void Talk (int id, bool isNPC) 함수의 내용과 같은 부분에 기존 Debug.Log 지우고

if (talkData == null)

{  
    isAction = false;  
    talkIndex = 0;

    QuestText.text = questManager.CheckQuest (id); // Debug.Log를 QuestText에 넣기

    return;

}

### 4. 종료하기

① OnClick() 추가. 여기서 끝나는 텍스트를 만들어 주야함.

GameManager의 새로운 함수

Public Void GameExit()

{ // 추가. Application.Quit()은 Editor에서 실행되지 않음.

    Application.Quit();

}  *추가된 Class*

② Build Setting

- Scene In Build의 Build에 들어갈 Scene을 꺼내 추가해야함. (Add Open Scene)

5. 저장하기 : Hand 저장해야될지, 어떤 RPGA가 있는지? Quest id, Quest ActionIndex, PlayerPref

① Game Manager on Save, load, 상수

Public GameObject Player;

어디 저장? PlayerPrefs, Company, Product Name?  
레지스터에 저장

Public void GameSave()  
{ → 전달할 Data 저장 기능을 가진다는 Class

PlayerPrefs.SetFloat("Player X", player.transform.position.x);

PlayerPrefs.SetFloat("Player Y", player.transform.position.y);

PlayerPrefs.SetInt("QUEST ID", questManager.questId);

PlayerPrefs.SetInt("QUEST Action Index", questManager.questActionIndex);

PlayerPrefs.Save();

}

Public void GameLoad()  
{ 이전에 저장되었는가?

if (!PlayerPrefs.HasKey("Player X")) // 저장 한적 없으면 return;  
return;

float x = PlayerPrefs.GetFloat("Player X");

float y = PlayerPrefs.GetFloat("Player Y");

int questId = PlayerPrefs.GetInt("QUEST ID");

int questActionIndex = PlayerPrefs.GetInt("QUEST Action Index");

Player.transform.position = new Vector3(x, y, 0);

questManager.questId = questId;

questManager.questActionIndex = questActionIndex;

3

↓ Save, Load 연동 불가능か?

Void Start()

{

GameLoad();

questText.text = questManager.CheckQuest();

}

GameSave는 나중으로

## ② Load 하는 때 당시의 쿼스트 순서

연결된 Obj 관리하기

obj는 Quest Manager에서

그는 ControlObject Public으로 바꾸고

Case 2d:

```
if (Quest ActionIndex == 0)  
    QuestObject [0]. SetActive (true);  
  
else if (Quest ActionIndex == 2)  
    :  
  
기준이
```

Game Load() 딱지 액션



Quest Manager. ControlObject(); // 추가

# 모바일 UI & 안드로이드 빌드

## 1. UI 구조 (모바일용)

① Canvas → Image → 비주얼 기능이 되는 이미지 생성, 아래로 앵커 설정

② Empty Object 생성 → 안에 내용은 Button이나 생성

③ 빈상도 대응을 위한 앵커 설정 + 크기조정

→ Mobile OK, HTML은 한 화면도 확인

④ Empty Object → 액션키 생성 + 앵커, 크기조정

↓

⑤ 기대도 대응이 가능한 HTML UI 액션이나 이벤트 대응

## 2. 방향키

① Player Action 대응

• 추가

2. Public void ButtonDown(String type)  
{

    switch(type){

        Case "U":

            break;

        Case "D":

            break;

        Case "R":

            break;

        Case "L":

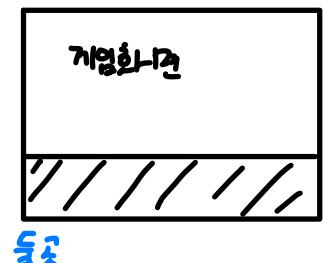
            break;

}

b. Public void ButtonUp(String type)

{

    동일 }



② Event 관리 추가

\* (UI 이벤트 관리함수)

이동 버튼의 Event Trigger 추가

Why? Button Up, Down은 (set,

Down) 대만 Button 누르면

Logic 문제 생길 확률 ↑

So 이 Component의 Pointer Down, Up 추가

그리고 일반 버튼처럼 Mapping

RIGHt 버튼 추가 - L, R, U, D

### ③ 기초 이동 설정

2. 기초 이동 맵고, 버튼 입력을 받을 변수 12H 쌍수 ( $7k + \text{Down} + \text{Up}$ ) X 4

```
int up_Value  
int down_Value  
:  
int up_UP  
int down_UP  
int left_UP  
:  
int Right_Down
```

추가  
→

```
public void ButtonUP (String type)  
{
```

Switch (type) {

Case "U":

```
    UP_Value = 0;  
    UP_UP = true;  
    break;
```

Case "D":

```
    down_Value = 0;  
    down_UP = true;  
    break;
```

Case "R":

```
    right_Value = 0;  
    right_UP = true;  
    break;
```

Case "L":

```
    left_Value = 0;  
    left_UP = true;  
    break;
```

3

```
public voidButtonDown (String type)  
{
```

Switch (type) {

Case "U":

```
    UP_Value = 1;  
    UP_Down = true;  
    break;
```

Case "D":

```
    down_Value = 1;  
    down_Down = true;  
    break;
```

Case "R":

```
    right_Value = 1;  
    right_Down = true;  
    break;
```

Case "L":

```
    left_Value = 1;  
    left_Down = true;  
    break;
```

3

3

## b. 빙수 사용

Void Update() {

    // Move Value

    // PC

    h = manager.isAction? 0 : Input.GetAxisRaw("Horizontal");  
    v = manager.isAction? 0 : Input.GetAxisRaw("Vertical");

Tip? 2개 API 쓰기

Mobile + right\_value + left\_value;  
API 활용.

    // Mobile

    h = manager.isAction? 0 : right\_value + left\_value;  
    v = manager.isAction? 0 : up\_value + down\_value;

    // Check Button Down & UP

    // PC

    bool hDown = manager.isAction? false : Input.GetAxisRaw("Horizontal");

    bool vDown =        •

    bool hUp =         •

    bool vUp =         •

Tip? AND 조건 true가 1개 있으면  
OR 조건 끝에 있으면.

    // Mobile

    bool hDown = manager.isAction? false : right\_down || left\_down;

    bool vDown =        •

    bool hUp =         •

    bool vUp =         •

## C. 3D 바일 빙수가 전역 변수니

Update()에 대해서만 Logic 끝까지 False로 초기화

    // Mobile var Init

        up\_up = false;  
        up\_down = false;  
        down\_up = false;

        •  
        •  
        •

### 3. 액션 키

① the Down 키에 액션 조건 추가

Switchcase 추가

Case "A": *기존 액션코드 복사*

if ( scanObject != null )

Manager. Action ( scanObject );

③ Cancel 키  
break;

Case "C":

manager. SubMenu Active();

break;

② ESC 키, Cancel 키로 블리하여 흐름으로 작성

• Game Manager의 Update() 또는 меню에 `public void SubMenuActive()` 넣기

④ Event Trigger, Pointer Down 추가. 어떤 UI든 활성화하니

### 4. UI 스케일링

① UI Canvas의 Scaler Mode를 Screen Size로 설정

→ Reference Resolution X 높이 480, Match Width로. *스크린 크기 설정*

### 5. 모바일 Build

① Build Setting with Android로 설정

② Player Setting 설정

a. Icons은 24x 48px

b. Allowed Orientations for Auto Rotation은

712 모드인 Landscape 를 제외(Right, Left, Both)

OR 위가 있는 Orientation은 Default Orientation으로 설정. - Portrait Up Side Down

### C. Splash Images 화면 이미지 유동화 (게임 시작할 때)

d. Other Settings - Identification - Package Name : Com.한국인증.프로젝트명

e. || - Configurations - Scripting Backend: Mono로 되어있는데 IL2CPP로 변경  $\rightarrow$  이제 구글이 선택한  
모드 **64비트 APK형** 받아서



f. Publishing Setting, Key Setting은 비슷한데

비슷한 허аниц이

Target Architectures는 비슷!

x86 or 32-bit . 64bit

ARM 64, ARM v7 만족

### ③ Build . APK생성 .

finish holy moly

구글 개발자 등록 25일로 1회. 오타 X

# 4. 쇼팅게임 - 1. 이동구현하기 Ver. Shooting Game

(종 스크립트)

## D. 준비하기

### ① 빌드기 - Player Sprite [영상과는 차이]

- Pixel Pen Unit은 Sprite 크기와 맞지
- Filter Mode는 Point (no filter)
- 압축모드 - Compression은 None. 압축 X

이후 자를거나 Single → Multi Plat  
한후 Sprite Editor로 자를.



## I. 풀려있던 이동

### C# Script - Player

```
public float Speed;
```

```
void Update()           -1, 0, 1 까지만
{
    float h = Input.GetAxisRaw("Horizontal");
    float v = Input.GetAxisRaw("Vertical");
    Vector3 curPos = transform.position; // Player 위치
    스クリプ트 Monobehavior 퀘
}
```

```
Vector3 nextPos = new Vector3(h, v, 0) * Speed * Time.deltaTime;
```

↳ transform object 뒤에  
Time.deltaTime 사용하기

```
transform.position = curPos + nextPos;
```

## 2. 해상도 조절

16:9일 수 있군 등..

- ① 게임창 Display 열기 Free Aspect: 자유비율 (카메라 크기 = Game 뷰 크기)  
에서 Add.

Type의 Fixed Resolution은 1920x1080, 등 정해진 거

Aspect Ratio는 비율. 예를 9:16으로. 첫 번째 9:16 추가

## 3. 경계 설정 - 화면 밖 안내하기

- ① Player한테 Box Collider 2D 넣기. 표면 벽과 물체로 쓰니 크게 험도 가

- ② Create Empty - Border → 안에 자식으로 4개 Create Empty 한개 주변 벽으로  
Box Collider 넣기

Tip? - Component 육수상단 툴바에 있는 Copy + Paste Component New ~로 Copy 후  
다른 Object에서 Transform 툴바에 있는 ↓로 붙여 넣기 가능

- ③ 물리연산용 Rigidbody2D 추가.

Player - 물리연산 많이 안쓰는 타입은 Kinematics  
Dynamic은 빠르면 빠르고  
Kinematic은 Script에  
그대로 움직이겠다.  
혹 4개 - Rigidbody2D 타입 Static(고정)

4 Tip: transform 이동 + 물리충돌은 떨림 현상이 발생함. - Player가 Dynamic인 경우

C# Script - Player : 4종

부수 - Public bool isTouchTop;  
Public bool isTouchBottom; 추가  
:  
]

```

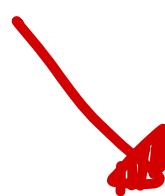
private void OnTriggerEnter2D(Collider2D collision)
{
    if(collision.gameObject.tag == "Border")
    {
        switch(collision.gameObject.name)
        {
            case "Top":
                isTouchTop = true;
                break;

            case "Bottom":
                isTouchBottom = true;
                break;

            case "Right":
                isTouchRight = true;
                break;

            case "Left":
                isTouchLeft = true;
                break;
        }
    }
}

```



→ Trigger  $\frac{1}{2}$  틈강  
 막으니 Is Trigger  $\frac{1}{2}$  죽음  
 (벽면)  
 → Update식 사용

OnTriggerExit2D로  
 같은 logic으로  
 플래그 지우기 - false

```

Void Update()           ←, 0, 1 371쪽
{
    float h = Input.GetAxisRaw("Horizontal");
    if((isTouchRight && h==1) || (isTouchLeft && h == -1)) //→가
        h=0;

    float v = Input.GetAxisRaw("Vertical");
    if((isTouchTop && v == 1) || (isTouchBottom && v == -1)) //↑가
        v=0;
}

```

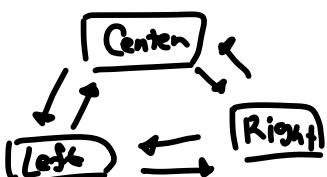
Vector3 CurPos = transform.Position; // Player 위치  
 ↗ 손잡이 위치

Vector3 nextPos = new Vector3(h, v, 0) \* Speed \* Time.deltaTime;  
 transform.Position = CurPos + nextPos;

}

## 4. OH4 미리보기

① 제작 OH4 미리보기 잘라내 [중간에 흑] Player한테 드래그. 뒤에 CreateOn/ 만들면 드러



② Animation On/Off 조정. 삼각형 서로 맞다같다 가능하기

③ 드래그해 int3, Input.GetAxisRaw 사용  
 이를 Input

Right는 1, Left는 -1, Center는 0

④ 키 입력 즉각 반응은 Has ExitTime 끄고, Setting의 3D용 뉴트로 모션은 Duration 0으로.

#### ④ 플레이어 사용: Player C# Script

I. Animator anim; 추가 후

Void Awake() 추가 후 초기화

{

Anim = GetComponent<Animation>(); // 초기화

}

II. 사용 - Update 맨 위

if (Input.GetButton Down ("Horizontal")) ||

Input.GetButton Up ("Horizontal"))

{

Anim.SetInten("Input", (int) h);

}

# B28 슈팅총알 발사 구현

## 0. 준비하기

제공 스프라이트 자르기. Same

## 1. Prefabs

① 단한 Sprite, Object로 만들고 충돌이벤트용 끌라이더, Rigid Body 생성  
중력 안 받으니 Gravity Scale = 0.

② Prefab : 재활용을 위해 예전으로 저장된 게임 Obj 으로 만들기 (단한)

## 2 오브젝트 삭제 (나온 단한)

C# Script - Bullet

```
OnTriggerEnter2D(Collision2D collision)
{
    if (collision.gameObject.tag == "Border Bullet")
        Destroy(gameObject);
}
```

Bullet 을 Border 만들. Object.tag

3

## 3. 오브젝트 생성 (총알 만드는 logic)

C# Script - Player 4장

- public float Gun Shoot Delay; // 총알 딜레이
- public float max Shoot Delay; // 4정 딜레이. 0.2초
- Public GameObject bulletObjA;
- Public GameObject bulletObjB;
- Public float Power; // bulletObjB 를. 크기↑

▲ Update 함수 logic 캡슐화 ( 다른 함수로 넣기 )

Update()
{
 Move();
 Fire();
 Reload();
}

기존 Update logic은 Private void Move()로 캡슐화

3

```
Void Fire()
```

```
{
```

```
if (!Input.GetButton("Fire 1"))  
    return;
```

② 충돌계가 충돌함. Bullet TM Is Trigger ✓

```
if (CurShootDelay < maxShootDelay)  
    return;
```

```
// Power3 탄환 번한 전코드
```

이미지 A Obj를 생성

```
// GameObject bullet = Instantiate(bulletObjA, transform.position, transform.rotation);
```

위쪽 오버로드 많음.  
다양

```
Switch (Power){
```

Case 1:

```
GameObject bullet = Instantiate(bulletObjA, transform.position, transform.rotation);  
Rigidbody2D rigid = bullet.GetComponent< Rigidbody2D>();  
rigid.AddForce(Vector2.up * 10, ForceMode2D.Impulse);  
break;
```

Case 2: //Power2 방향 2D

+ Vector3.right \* 0.1

```
GameObject bulletR = Instantiate(bulletObjA, transform.position, transform.rotation);  
Rigidbody2D rigid = bulletR.GetComponent< Rigidbody2D>();  
rigid.AddForce(Vector2.up * 10, ForceMode2D.Impulse); + Vector3.left * 0.1  
	GameObject bulletL = Instantiate(bulletObjA, transform.position, transform.rotation);  
Rigidbody2D rigid = bulletL.GetComponent< Rigidbody2D>();  
rigid.AddForce(Vector2.up * 10, ForceMode2D.Impulse);  
break;
```

Case 3: //그린화

```
GameObject bullet = Instantiate(bulletObjB, transform.position, transform.rotation);  
Rigidbody2D rigid = bullet.GetComponent< Rigidbody2D>();  
rigid.AddForce(Vector2.up * 10, ForceMode2D.Impulse);  
break;
```

```
CunShootDelay = 0;
```

```
}
```

```
Void Reload()
```

```
{
```

설정시간

```
    CunShootDelay += Time.deltaTime;
```

```
}
```

# 적 뇌회기 만들기 (+다운로드)

O를 넣기 → Sprite 만들기

나기증

- Collider 생성(트리미언트풀) - Polygon Collider 2D Custom Physics Shape 2  
이내 끝내 Shape하면 물리법칙 조절 가능

- Rigidbody 2D (속도) Gravity Scale = 0.

## I. 적 조립

C# Script - Enemy 추가 후 조립

```
Public float Speed;  
Public int health;  
public Sprite[] sprites; // [0]은 2D Sprite, [1]은 3D Sprite
```

```
Rigidbody2D rigid;  
SpriteRenderer spriteRenderer;
```

```
void Awake()
```

```
{ // 초기화  
    spriteRenderer = GetComponent<SpriteRenderer>();  
    rigid = GetComponent<Rigidbody2D>();
```

```
    rigid.velocity = Vector2.down * Speed;
```

```
}
```

```

Void OnHit (int dmg)
{
    health -=dmg;
    SpriteRenderer.Sprite = Sprites[1]; //Sprite Change .30%41
    Invoke ("ReturnSprite", 0.1f);
    if (health <= 0)
    {
        Destroy(gameObject);
    }
}

```

Void ReturnSprite()

```

{
    SpriteRenderer.Sprite = Sprites[0];
}

```

↑n) 21 32 막기로 10%  
Is Trigger.

Void OnTriggerEnter2D (Collider2D collision)

```

{
    if (collision.gameObject.tag == "Bor den Bullet")
        Destroy(gameObject);
}

```

else if (collision.gameObject.tag == "Player Bullet") // Player Bullet tag 37%

Bullet bullet = collision.gameObject.GetComponent<Bullet>();  
Bullet Script

OnHit(bullet dmg);

↳ Bullet Script 가 7%

public int dmg; // 7%

Destroy (collision.gameObject); // 블록하기 7%

}

< Prefab 저장 > + transform 초기화

## 2. 적 생성 - DHTML

① 적 생성을 관리할 Game Manager 캐스팅 (Create Empty, C# Script) 이를 통해 빈 GameObject를  
'아이언'한 바꾸.

② C# Script - Game Manager

Public GameObject[] enemyObjs; // 적 빙고

Public Transform[] spawnPoints; // 생성위치 빙고

Public float maxSpawnDelay; // 딜레이

Public float curSpawnDelay; // 현재까지의 시간

Void Update()

{

CurSpawnDelay += Time.deltaTime;

if (CurSpawnDelay > maxSpawnDelay); {

SpawnEnemy();

maxSpawnDelay = Random.Range(0.5f, 3f); // Random.Range는 반드시  
0과 1 사이 (0.5f ~ 3f). 예상

CurSpawnDelay = 0;

}

}

Void SpawnEnemy()

by 선택 0.5초스보고 알아서 = 0.45초

{

int ranEnemy = Random.Range(0, 3);

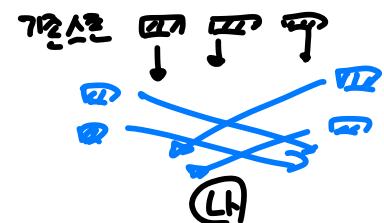
int ranPoint = Random.Range(0, 5); // Spawn 위치 5개

Instantiate(enemyObjs[ranEnemy], spawnPoints[ranPoint].Position,  
spawnPoints[ranPoint].Rotation); // 소환

}

[Public 할당해주기 (위치)] - Object 이는 물체를  
Icon상상. 네 오브젝트 브라우저

# 적 전투 + 파티 이벤트



## I. 적동장 강화

- ① Spawn 지점 높낮이 + 좌우로 바꿔 → 경비망 대나무 스픈지점 추가  
↳ 오른쪽과 왼쪽에 쪼개
- ② C# Script - Game Manager 478.

```
Void SpawnEnemy()
```

```
{ int ranEnemy = Random.Range(0,3);
```

```
int ranPoint = Random.Range(0, 9);
```

```
GameObject enemy = Instantiate(enemyObj[ranEnemy],
```

```
SpawnPoints[ranPoint].Position,
```

```
SpawnPoints[ranPoint].Rotation);
```

```
Rigidbody2D rigid = enemy.GetComponent< Rigidbody2D >();
```

```
Enemy enemyLogic = enemy.GetComponent< Enemy >();
```

```
if(ranPoint == 5 || ranPoint == 6) // 오른쪽이나 유행 스픈지역
```

```
{ rigid.Velocity = new Vector2(enemy.Logic.Speed * (-1), 1);
```

```
enemy.transform.Rotate(Vector3.back * 90); // 90도회전
```

```
 } // Right Spawn
```

```
else if(ranPoint == 7 || ranPoint == 8) // 오른쪽이나 유행 스픈지역
```

```
{ rigid.Velocity = new Vector2(enemy.Logic.Speed, 1);
```

```
enemy.transform.Rotate(Vector3.front * 90);
```

```
 } // Left Spawn
```

```
else
```

```
{ rigid.Velocity = new Vector2(0, enemy.Logic.Speed * (-1));
```

```
 } // Front Spawn
```

## 2. 적 공격

- ① 기존 프리팹이나 블록 → Open Prefab. 하위 수정
  - Name, Sprite, Collider ...

- ② Enemy Bullet (123자)

C# Script - Enemy : Player Logic 재활용.

Fire(), Reload() 뷰로 버튼이 아니라 기존 캐릭터 API 사용.

GetButton, 총알 Player 사용

초기화 [  
    Public String enemyName;  
    Public GameObject Player;

Fire()

if (enemyName == 'S')  
{  
    (기존 방식 ~)

    목표 방향      목적 위치

    Vector3 dirVec = Player.transform.position - transform.position;

    rigid.AddForce (dirVec.normalized \* 10, ForceMode2D.Impulse);

    ↳ 단위 벡터로 만든 벡터. 같은 이름 잘려도 있음 (Normalized)

    }  
    else if (enemyName == 'L')

    }{  
    Large Enemy, 큰 높은 총알 2개 방식

    };  
    방법 2.

위치가 고정하기

GameManager Script ①

Public GameObject Player; // 블록

Spawn Enemy() ①

    enemyLogic.Player = Player; // 적 생성 직후에 플레이어 번수를 넣겨줌

\* Prefab은 Only Scene!

클라우드 오브젝트에 접근 불가능

들여는 가능하나 인스턴스화 안되어 작동 X.

단!! Prefab은 Prefab은 사용 가능.

### 3. 이걸 이걸

#### ① C# Script - Player

```
public GameManager manager;
```

void OnTriggerEnter2D() { Border에 카운팅이 추가. }

```
if ~  
{ Border ~
```

```
else if (collision.gameObject.tag == "Enemy" ||  
         collision.gameObject.tag == "Enemy Bullet")
```

```
{
```

```
    manager.RespawnPlayer();
```

gameObject.SetActive(false); // 죽고는 말고 비활성화 → Invoke 사용 불가.

GameManager에서 실행될 때 부른다.

#### ② C# Script. GameManager

```
public void RespawnPlayer()
```

```
{
```

```
    Invoke("RespawnPlayerExe", 2f);
```

```
}
```

```
void RespawnPlayerExe()
```

```
{
```

```
    Player.transform.position = Vector3.down * 3.5f;
```

```
    Player.SetActive(true);
```

```
}
```

