



[스파르타코딩클럽] 게임개발 종합반 - 1주차



매 주차 강의자료 시작에 PDF파일을 올려두었어요!

▼ PDF 파일

[수업 목표]

1. 유니티 다뤄보기
2. C# 기본 문법 익히기
3. 유니티 기본 사용법 익히기

[목차]

- 01. 1주차 오늘 배울 것
- 02. 유니티 설치하기
- 03. 기본 씬 구성하기
- 04. 애니메이션 맛보기
- 05. 캐릭터 움직이기
- 06. 빔방을 내리게 하기
- 07. 빔방을 랜덤하게 나타나게 하기
- 08. 빔방을 계속 나오게 하기
- 09. 점수 올라가게 하기
- 10. 게임 끝내기
- 11. 속제 - 빨강 빔방을 만들기
- HW. 1주차 속제 해설



모든 토글을 열고 닫는 단축키

Windows : **Ctrl** + **alt** + **t**

Mac : **⌘** + **⌥** + **t**

01. 1주차 오늘 배울 것

▼ 1) 게임개발종합반 수업의 목표와 범위



기본적으로 게임개발 종합반 수업은 "스스로 찾을 수 있는 단계"로 만드는 데에 있습니다.

"여차피 게임 개발자들도 모든 유니티 코드를 외우고 있지 않습니다.

결국, 어떻게 동작하는지 대략적인 기능을 이해하고,
내가 필요한 부분을 찾아서 만들 수 있는 단계로 오르는 것이 중요합니다."

- **핵심:** 유니티는 안 어렵습니다! 그런데, 처음에 사용법을 깔끔하게 설명해둔 곳이 없습니다!
- 4~5개를 만들어보게 되면, 결국 코드는 돌고 돈다-는 것을 알게 되실 거예요.

- C# 이라는 프로그래밍 언어를 사용하는데요, 이것은 하면서! 알려드리겠습니다. 😊

▼ 2) raindrop - 친환경 게임: 빗물 받는 르탄이

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/12970db0-a2fa-4090-8721-dabc106195c7/빗물_받는_르탄이.mp4

▼ 3) 5주 강의 구성



여러번 반복 숙달로 익숙해질 수 있게 구성하였습니다. **사실 게임 만들기는 쉽다니까요!**

- 1주차 - **빗물 받는 르탄이** : 유니티 세팅, 기초 문법 연습
- 2주차 - **풍선을 구해라! 백만 다운로드 게임 따라만들기** : 유니티 기초 복습
- 3주차 - **고양이 밥주기 게임** : hp바, 레벨 연습하기
- 4주차 - **르탄이 카드 뒤집기 게임** : 보드 게임 기초 구현하기
- 5주차 - **주변 기능 학습** : 스플래시 화면 구성, 광고붙이기, 배포하기, 무료 에셋 구경하기

▼ 4) 오늘 만들 순서

- (1) 유니티 - 기본 세팅, 씬 구성하기
- (2) 캐릭터 왔다 갔다 하게 하기 + 클릭 시 방향 전환 구현
- (3) 비 내리기 구현
- (4) 비 충돌 구현
- (5) UX (남은 시간 / 숫자합) 구현
- (6) 게임 오버(팝업) 구현

02. 유니티 설치하기

▼ 1) ([다운로드 링크](#))를 클릭해서 Unity-hub를 다운받습니다.



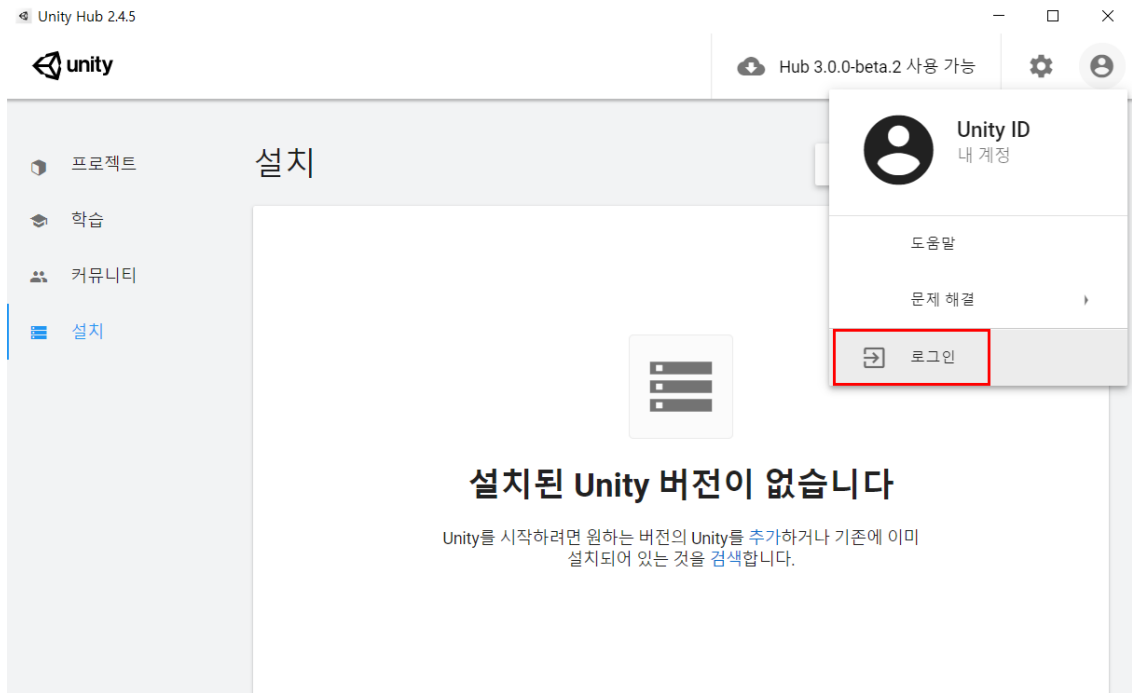
Unity Hub는 일종의 프로그램 설치/프로젝트 시작을 총괄하는 입장 안내데스크

▼ [코드스니펫] 유니티허브 다운로드

https://public-cdn.cloud.unity3d.com/hub/prod/UnityHubSetup.exe?_ga=2.197600431.1066071928.1631537679-831002153.1627910894

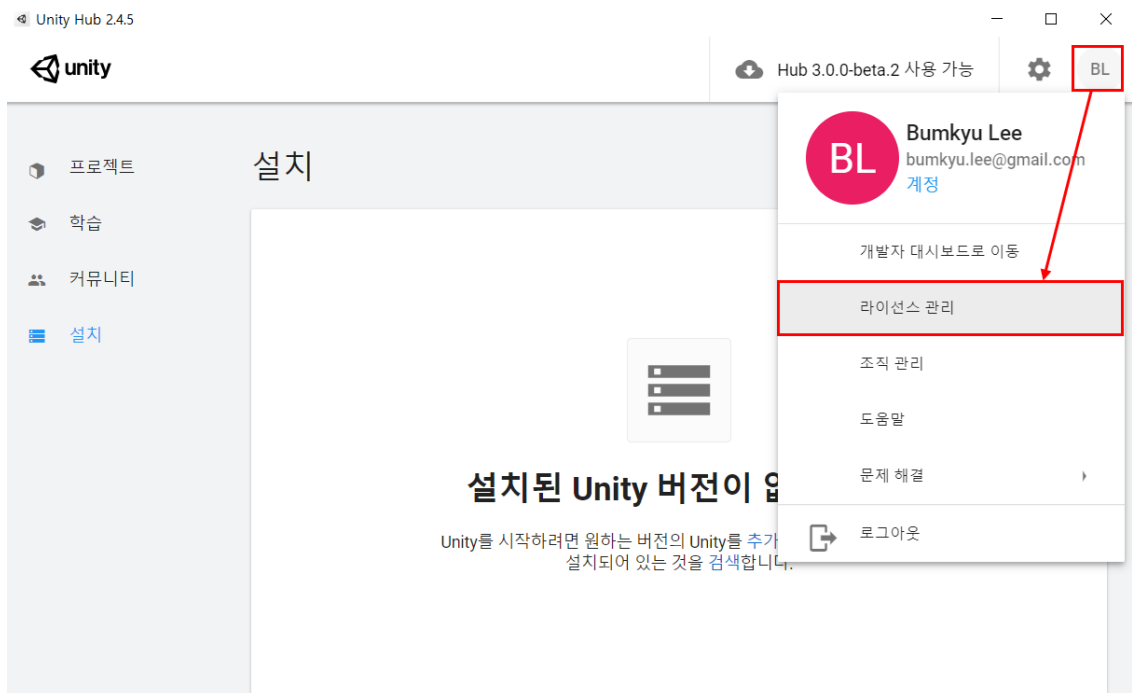
▼ 2) 로그인 진행

1. 로그인하기 클릭 후 회원가입 진행 → 구글로 로그인

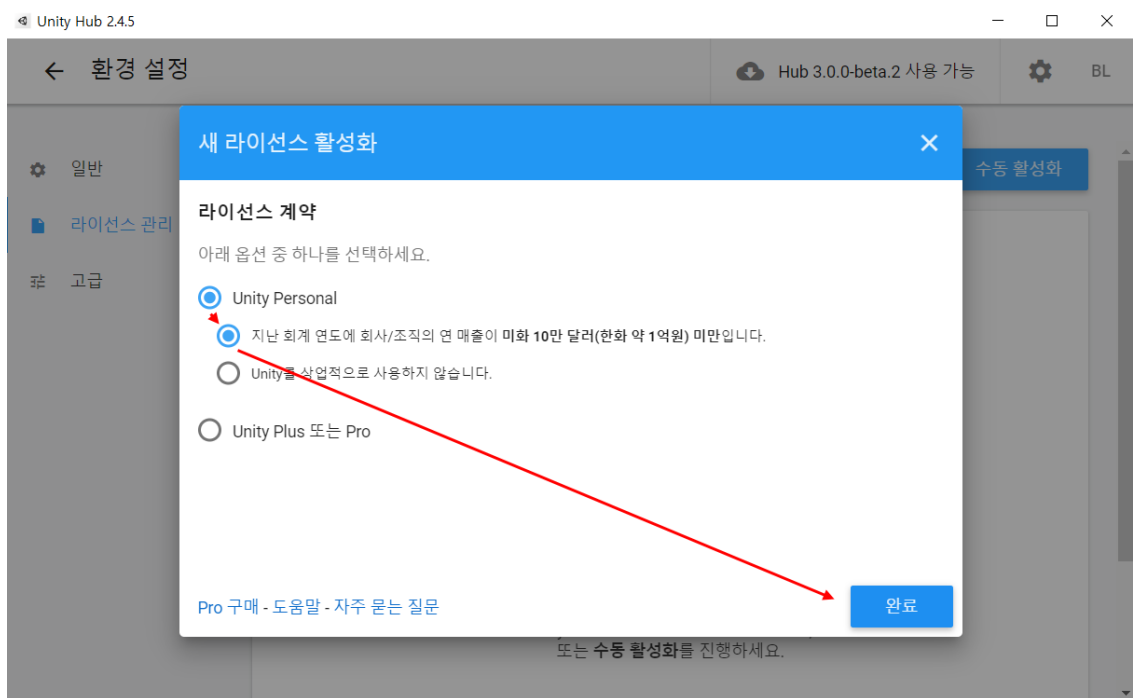
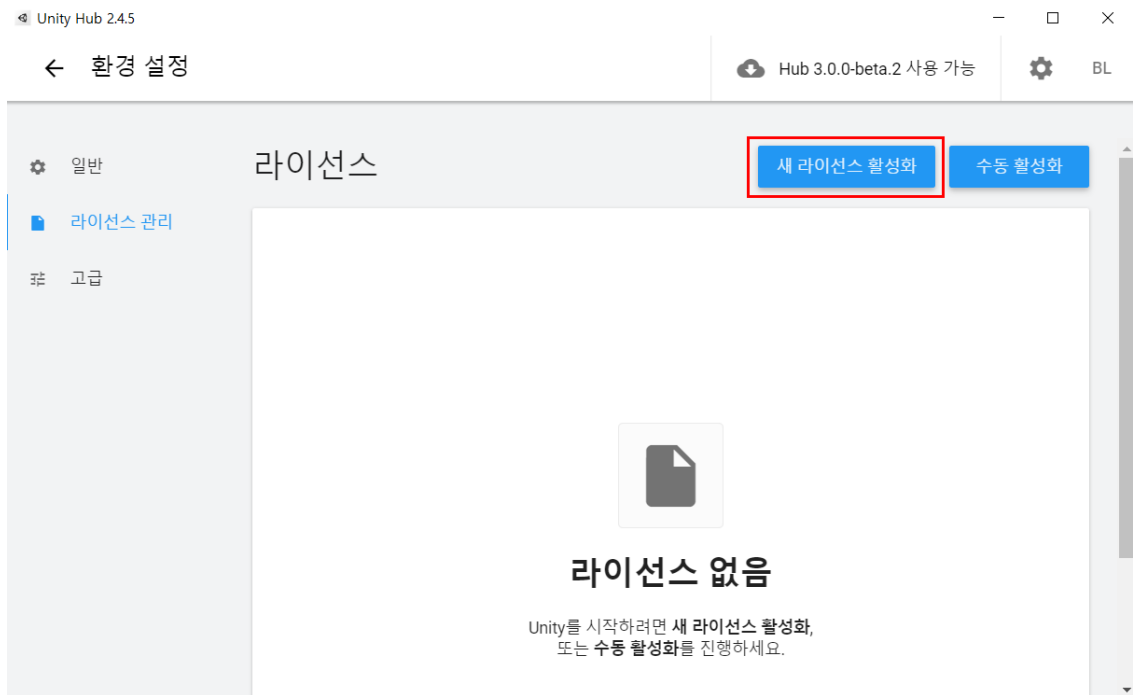


▼ 3) 라이선스 발급받기

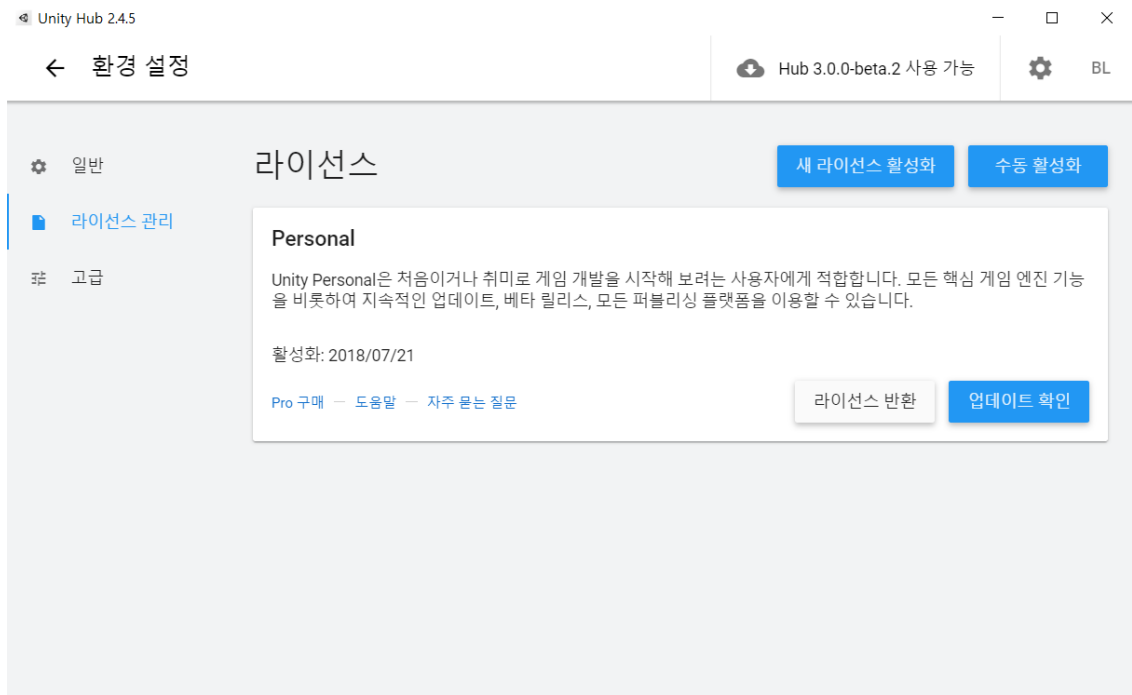
1. 라이선스 관리 클릭



2. 새 라이선스 활성화 클릭 → Personal → 완료

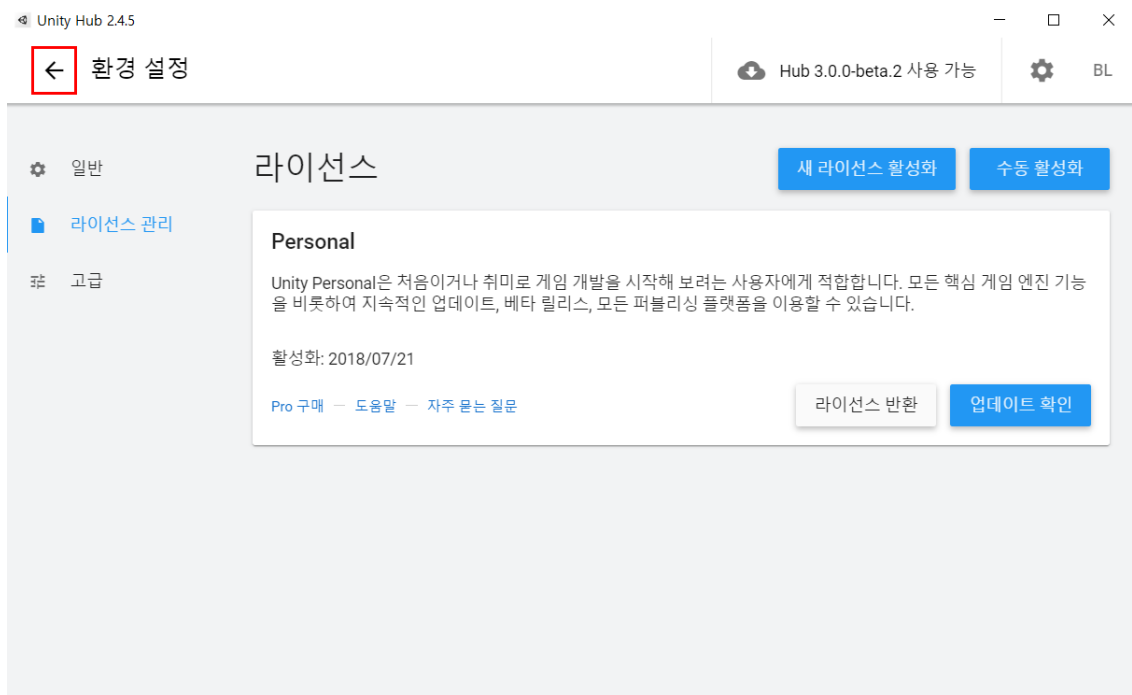


3. 아래와 같은 화면이 나오면 발급 완료!

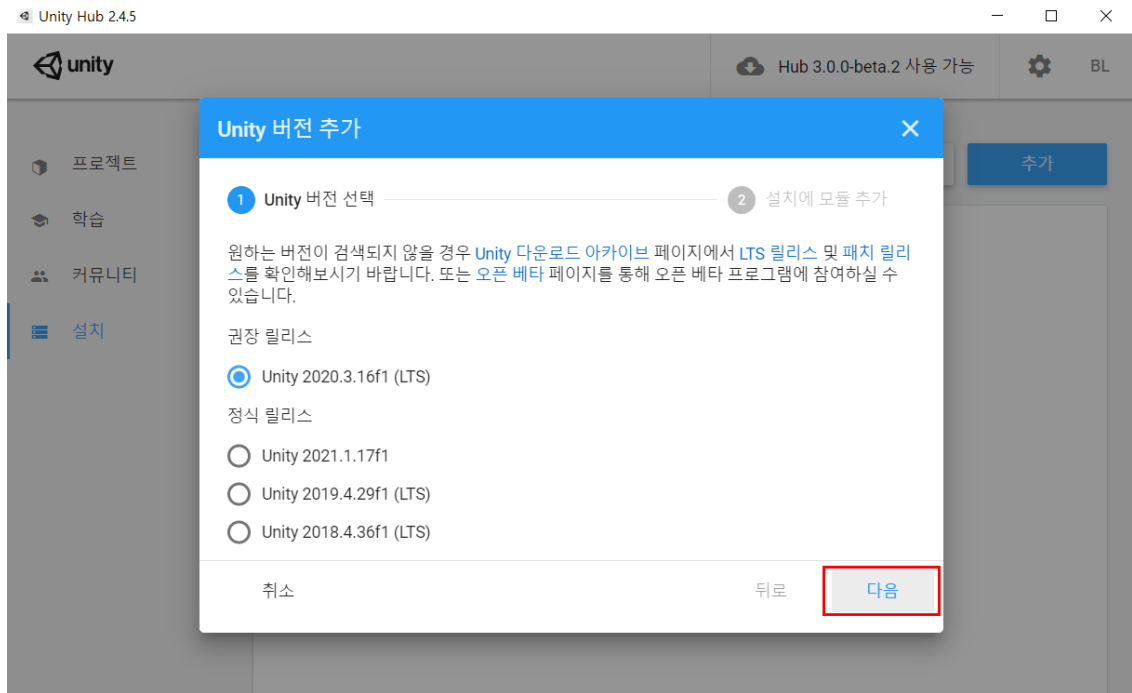


▼ 4) 유니티 설치하기

1. 뒤로가기 눌러서 메인으로 돌아온 뒤 → 설치 → '추가' 클릭



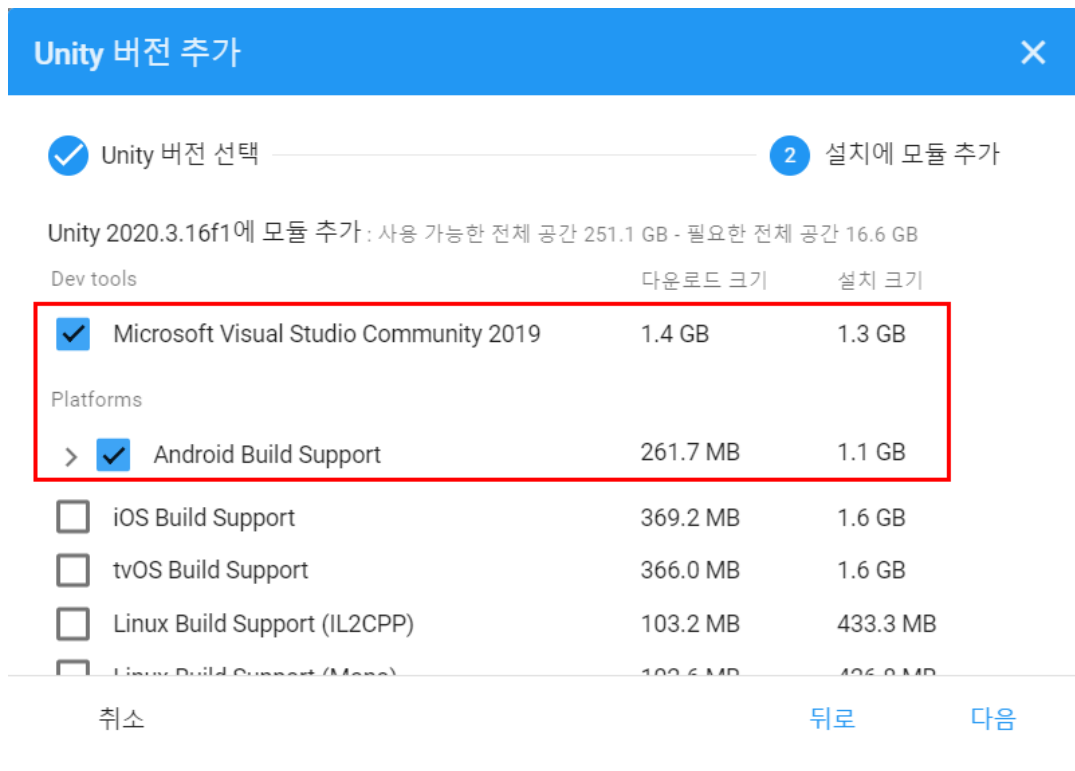
2. 아래 화면에서 '다음' 클릭



3. VisualStudio 클릭 + Android build support 클릭

+ 맥의 경우 'Mac build support' / 윈도우의 경우 'Windows build support' 클릭

(모듈은 추후에도 추가할 수 있으니 잘못 체크했을까봐 너무 걱정 마세요!)



Unity 버전 추가

✕

<input type="checkbox"/>	tvOS Build Support	366.0 MB	1.6 GB
<input type="checkbox"/>	Linux Build Support (IL2CPP)	103.2 MB	433.3 MB
<input type="checkbox"/>	Linux Build Support (Mono)	102.6 MB	426.8 MB
<input type="checkbox"/>	Mac Build Support (Mono)	318.3 MB	1.8 GB
<input type="checkbox"/>	Universal Windows Platform Build Support	287.4 MB	2.1 GB
<input type="checkbox"/>	WebGL Build Support	318.2 MB	1.1 GB
<input checked="" type="checkbox"/>	Windows Build Support (IL2CPP)	73.5 MB	374.7 MB
<input type="checkbox"/>	Lumin OS (Magic Leap) Build Support	159.7 MB	870.9 MB

Documentation

<input type="checkbox"/>	Documentation	284.0 MB	579.4 MB
--------------------------	---------------	----------	----------

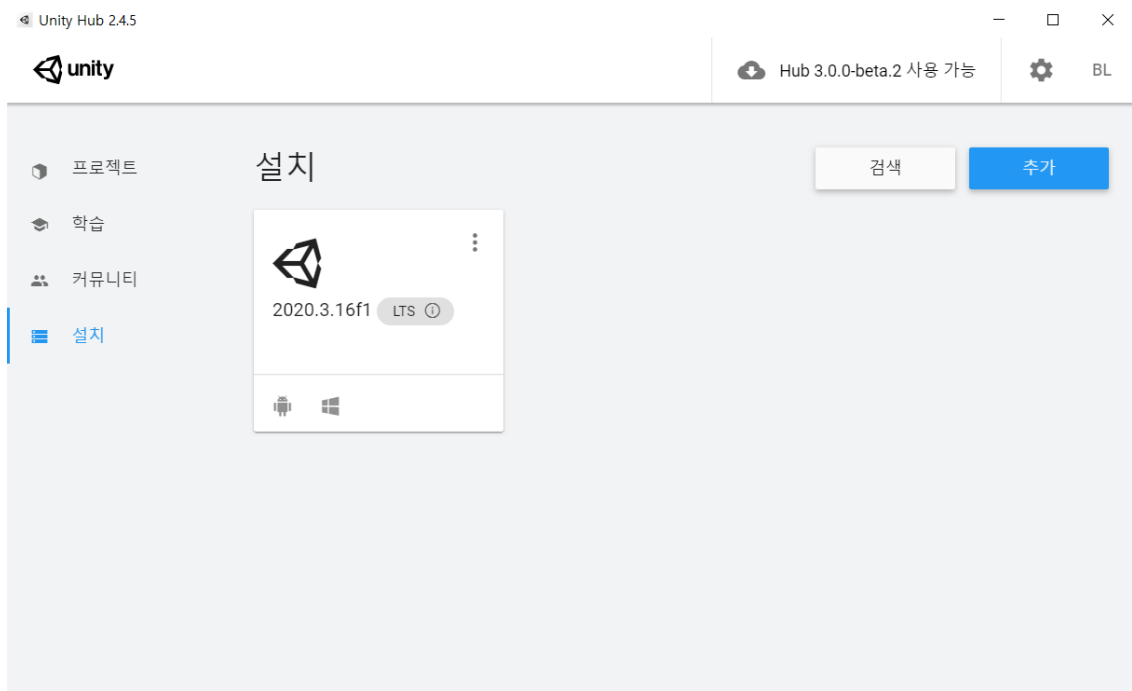
Language packs (Beta)

취소

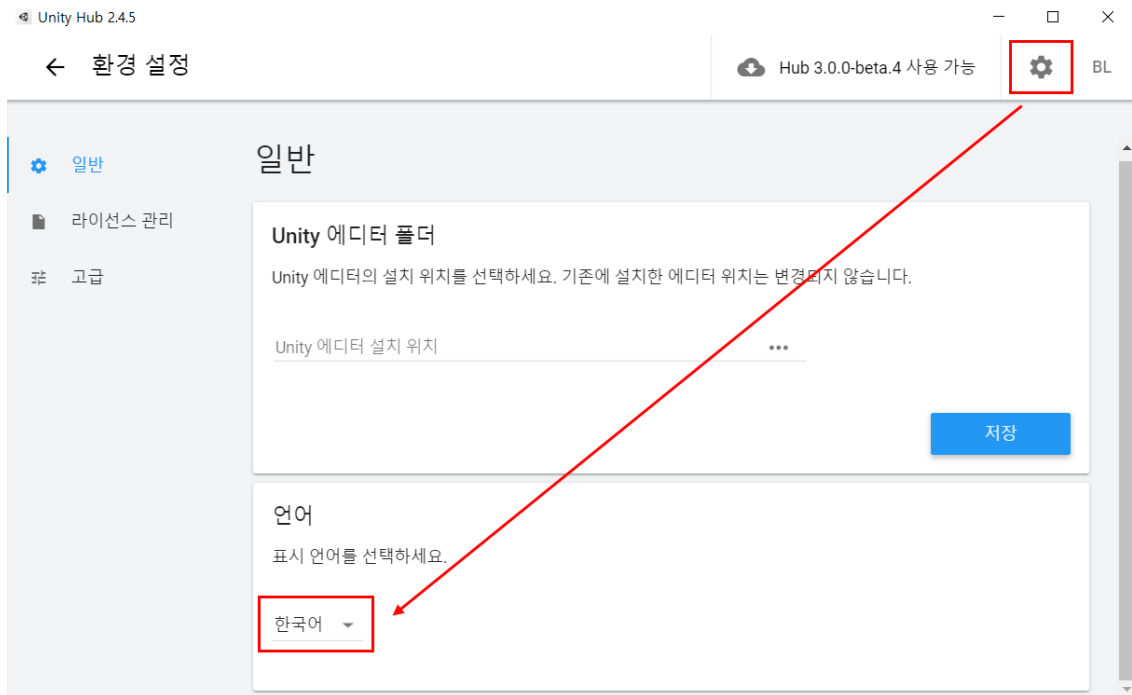
뒤로

다음

4. 동의 → 동의 → 완료 클릭. 꽤 오랜 시간 (최장 20~30분까지) 기다리면 설치 완료!



5. 마지막으로, 한국어 세팅하면 끝!

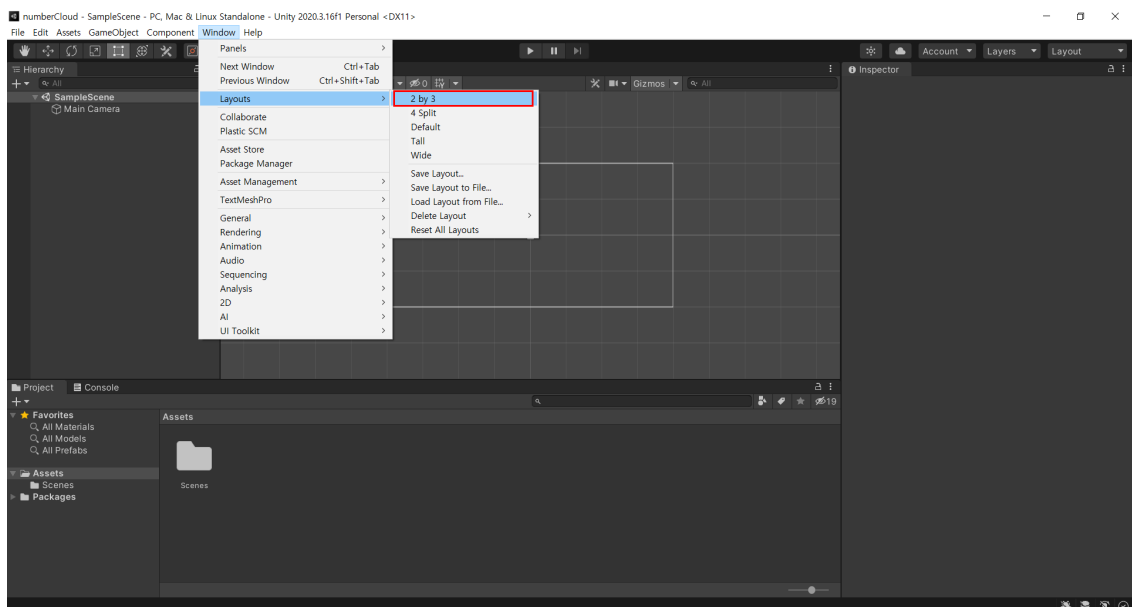


03. 기본 씬 구성하기

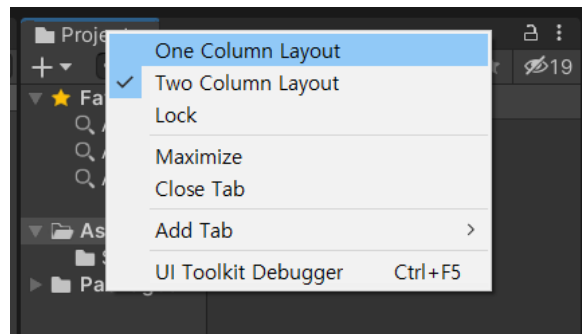
▼ 1) 유니티에서 개발하기

👉 **유니티란?** 충돌/중력/탄성/카메라 이런 것들을 쉽게 코딩 할 수 있는 툴

- 게임 개발에 최적화된 개발 환경. 특히 2D 게임은 거의 100% 유니티로 개발한 것으로 생각하면 됨. 최근엔 대놓고 unity 로고를 보이는 게임들도 많음. 그림판 vs 포토샵.
- 프로젝트 생성 후 window → 아래와 같이 뷰 환경을 세팅!



project → 오른쪽 클릭 → one column layout 클릭



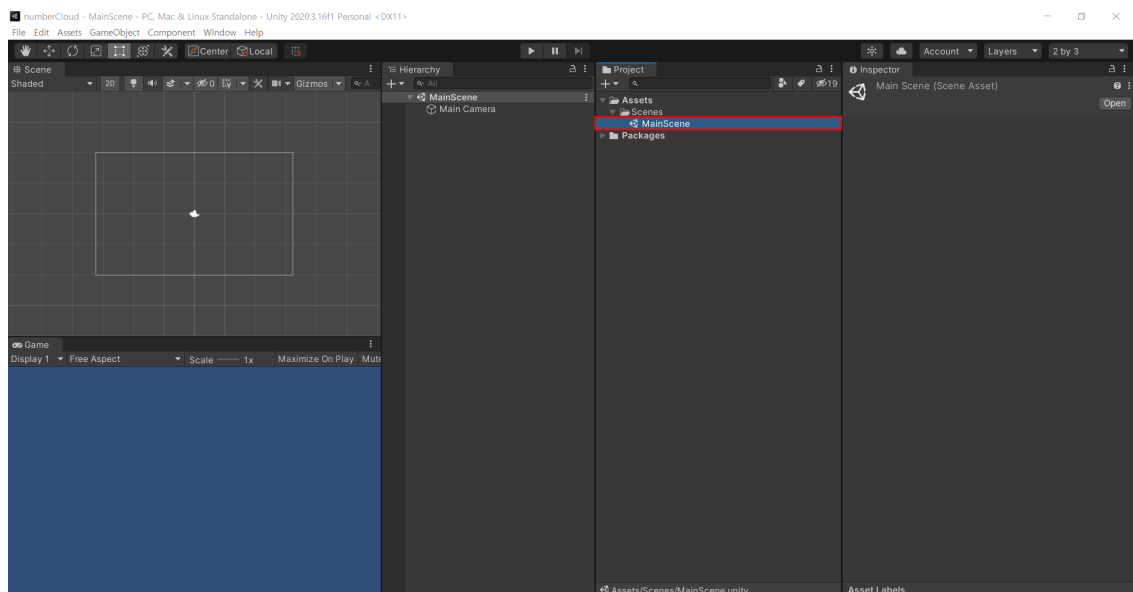
▼ 2) 각각 뭐 하는 뷰일까?

- Scene : 실제 게임의 구성요소를 보는 곳. 실질적인 게임 개발 씬
- Game : 게임이 실제로 보여지는 곳. play 버튼 클릭 후 볼 수 있음
- Hierarchy : 게임 내 구성요소를 볼 수 있는 곳. 개발 시 자주 필요
- Project : 이 프로젝트에 포함된 파일들을 모아볼 수 있는 곳
- Inspector : 클릭한 요소의 속성과 정보를 보여주는 곳(차차 보면 알게 됨!)

▼ 3) 배경 세팅하기

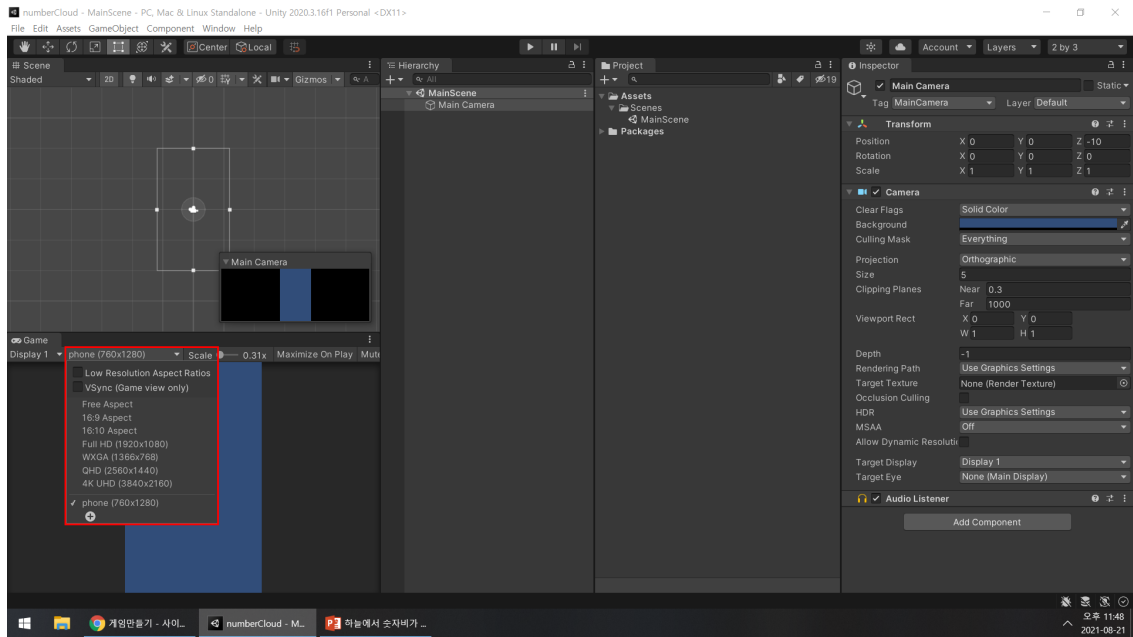
1. 메인 씬 이름 바꾸기

→ project에서 오른쪽 클릭 후 MainScene으로 변경



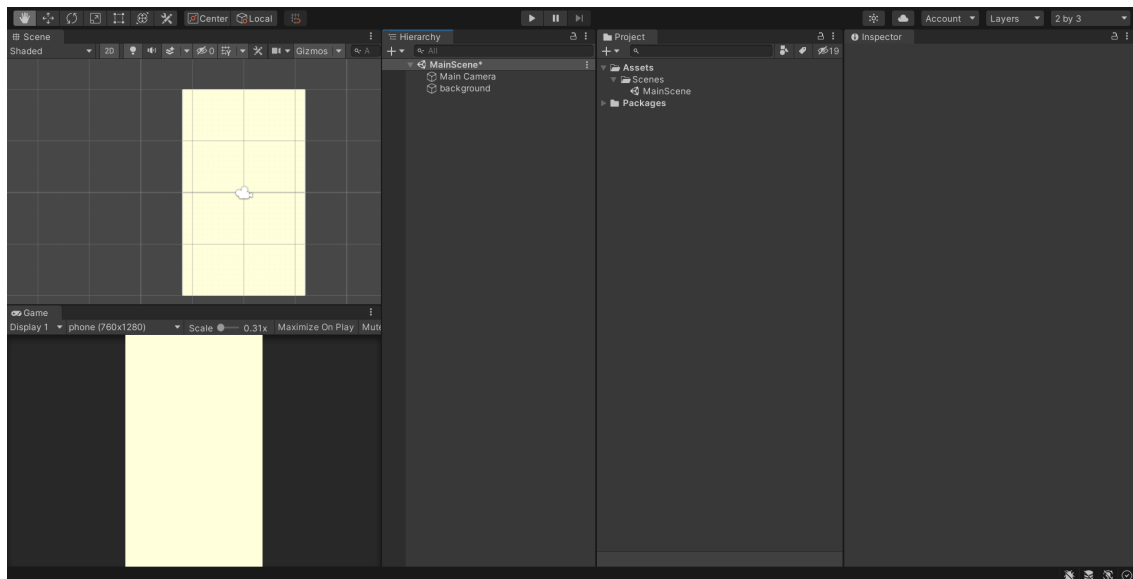
2. Game 씬 사이즈 바꾸기

→ + 버튼을 클릭하고 760 x 1280 Phone을 입력 → Phone 으로 변경



3. 배경 입히기

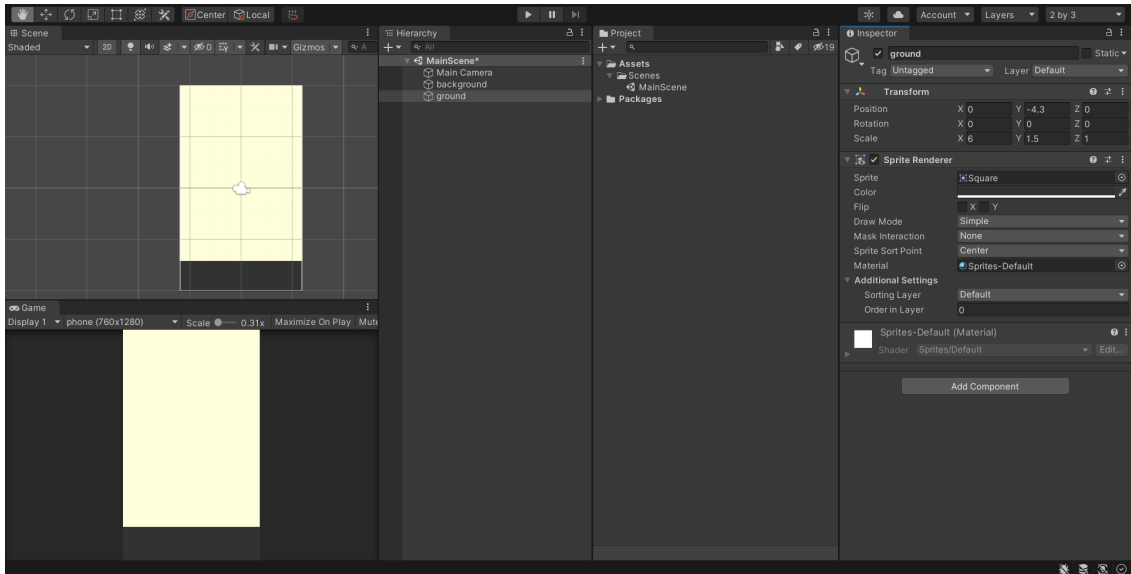
- 2D Object → Sprite → Square 클릭 → background로 이름 바꾸기
- 색을 255, 255, 220, 255 로 맞추기
- Scale을 X:6, Y:10 으로 맞추기



▼ 4) UI박스(점수) 세팅하기

1. 검은색 박스 만들기

- 2D Object → Sprite → Square 클릭 → ground로 이름 바꾸기
- 색을 50, 50, 50, 255 로 맞추기
- Scale을 X:6, Y:1.5 으로 맞추기 + Position은 Y:-4.3 으로 맞추기
- order in layer를 1로 맞추기



2. '에셋'에 캐릭터 넣어두기

▼ [코드스니펫] 르탄이 이미지

https://s3.ap-northeast-2.amazonaws.com/materials.spartacodingclub.kr/game_new/week01/rtan.zip

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/50d75f15-09cc-4ecb-a4d8-4894c6a90112/rtan.zip>

→ Assets 에서 Images 폴더 생성 → 르탄이 이미지 압축 풀고 끌어다놓기

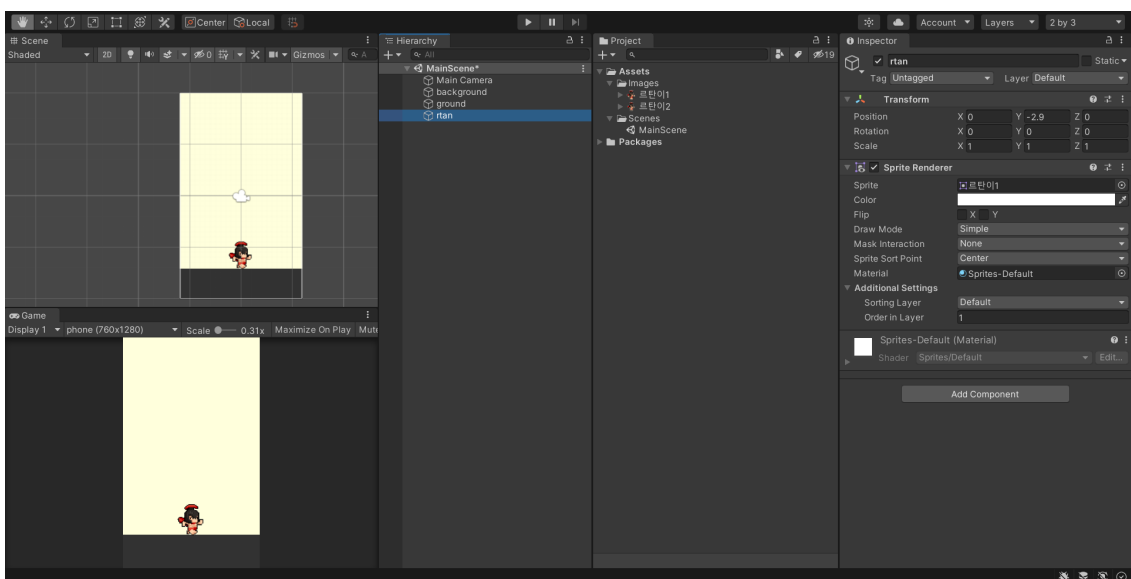
3. 르탄이 캐릭터 만들기

→ 2D Object → Sprite → Square 클릭 → rtan으로 이름 바꾸기

→ Sprite 부분에 르탄이1 이미지 끌어다놓기

→ Order in Layer를 1로 바꾸기

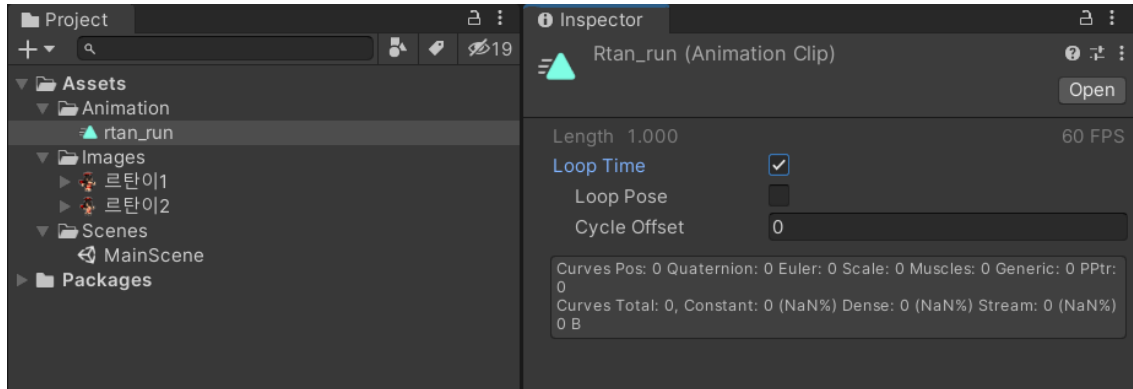
→ position Y: -2.9



04. 애니메이션 맛보기

▼ 1) 간단한 애니메이션을 입혀보기

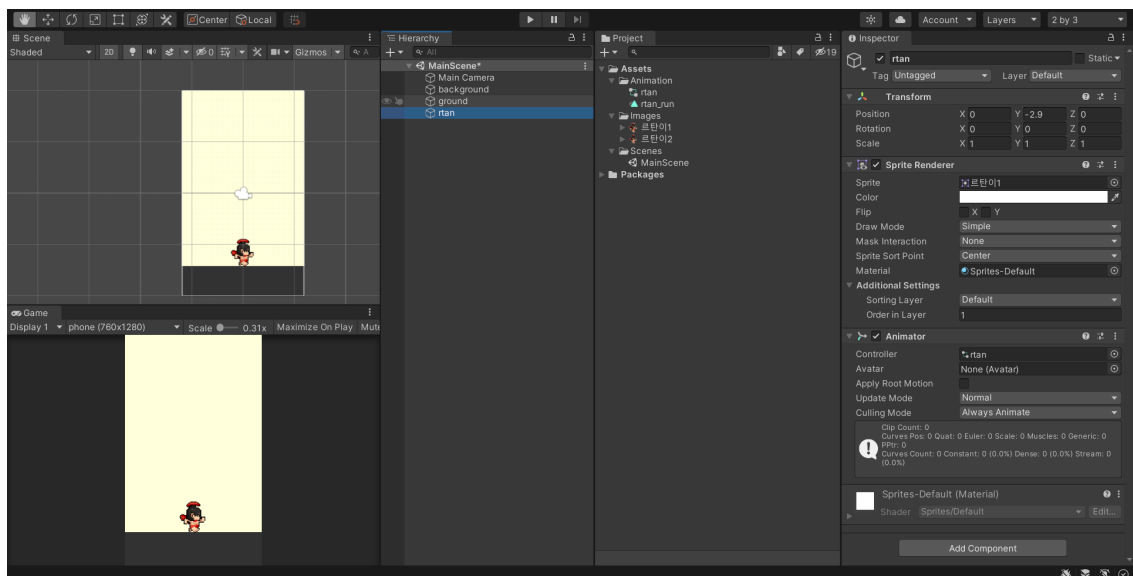
1. 애니메이션 폴더 만들기 (Asset → Animation)
2. 애니메이션 파일을 만들고, Loop Time에 체크



3. 이것을 만들어둔 르탄 캐릭터에 sprite에 끌어다놓기

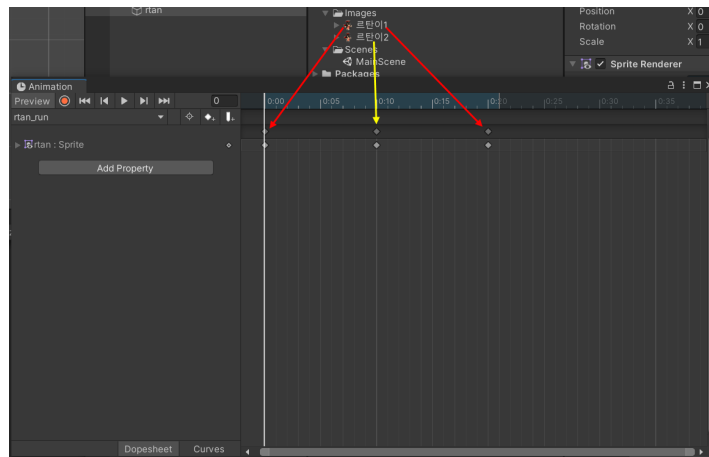
4. Controller가 생긴 것을 확인!

- Controller는 : Animation을 컨트롤 하는 것 (예 - 보통 상태 / 맞을 때 / 땀 때 어떤 애니메이션을 써라)
- Animation은 : 동작 파일



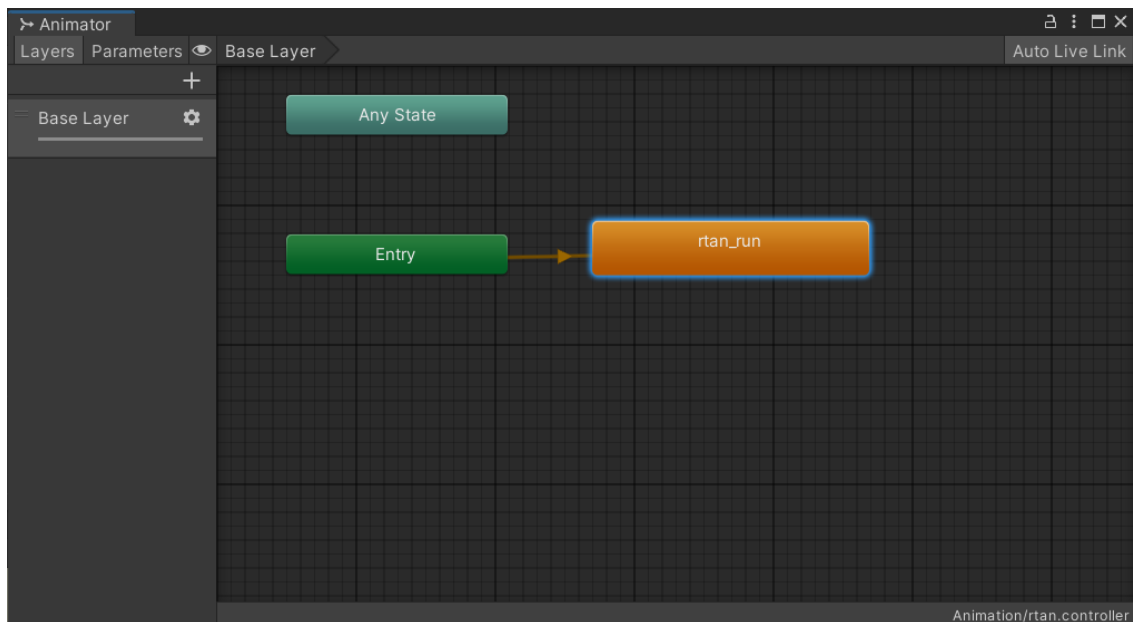
▼ 2) 기본 Animation 만들어보기

1. `rtan_run.anim` 더블 클릭 후 → 르탄이 캐릭터 클릭
2. 르탄이1, 2파일을 적당한 시간 간격으로 끌어다두기



3. Animator 간단 설명

- 시작하면 무조건 rtan_run을 실행하게 되어있고
- rtan_run은 끝이 없는 애니메이션임

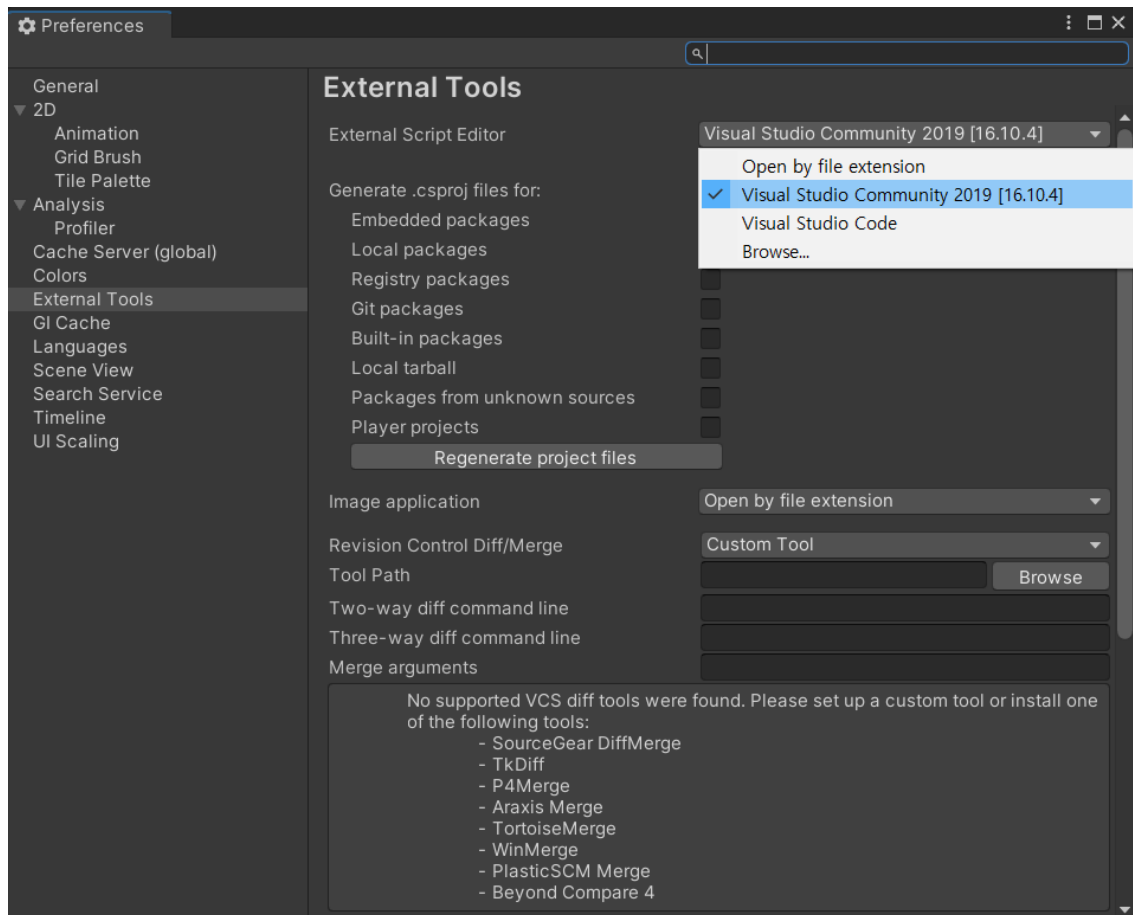


4. 실행해보면, 움직인다!

05. 캐릭터 움직이기

▼ 1) 먼저 세팅하기 : Visual Studio

- 윈) Edit → Preferences → External Tools → Visual Studio Community 2019로 맞추기
- 맥) Unity → Preferences → External Tools → Visual Studio for mac

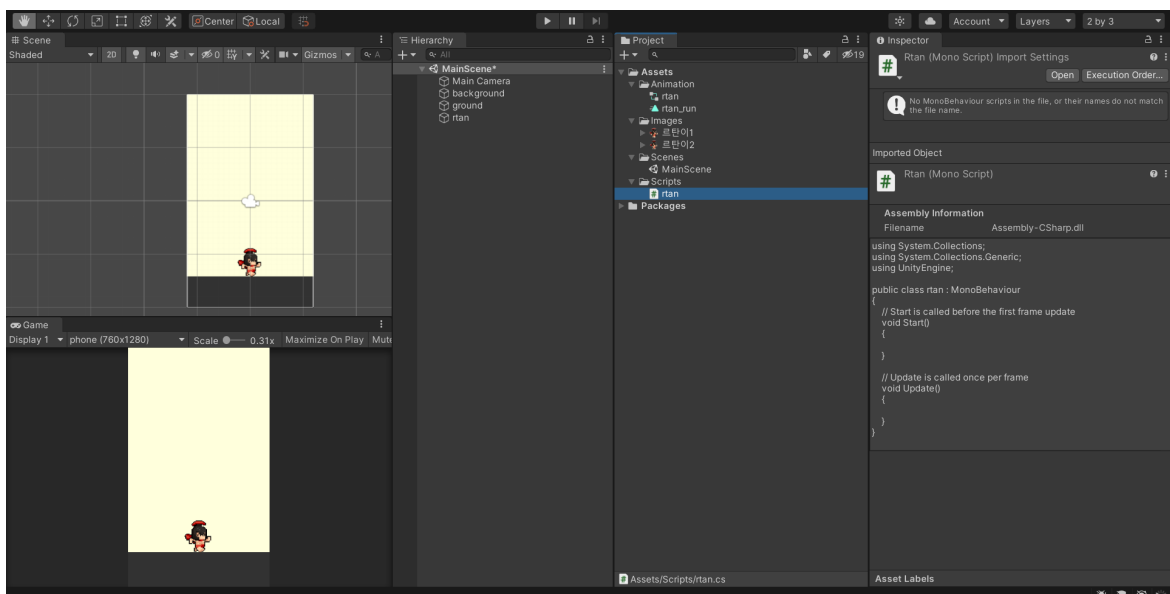


▼ 2) 캐릭터에 코딩을 더하는 법

- 유니티에서는, 캐릭터가 코드를 갖고 있을 수 있음
- 즉, 캐릭터에 코드를 입히는 것. "너는 태어날 때 여기서 태어나고, 매 순간 이렇게 작동해라"
- 가장 중요한 두 가지 함수가 있음. start (너는 태어날 때) / update (매 순간 이렇게 해라)

▼ 3) Script 만들기

- Assets 우클릭 → Create → Folder (이름 Scripts) → Create → C# script (이름 rtan)
- C#은? Microsoft가 개발한 코딩 개발 언어. 희한하게 유니티에서만 주류로 쓰이고 있다.



▼ 4) 캐릭터 좌우 움직임 코딩하기

- 1) 캐릭터 오른쪽으로 이동하기

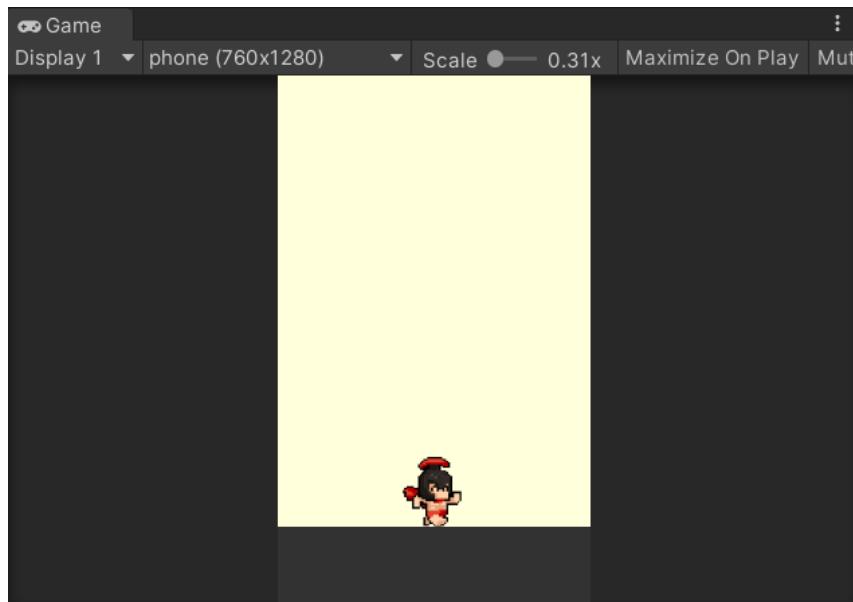
→ 아래와 같이 입력하고 캐릭터에 스크립트를 끌어다 놓기

▼ [코드스니펫] 캐릭터 오른쪽으로 이동하기

```
void Update()
{
    transform.position += new Vector3(0.05f, 0, 0);
}
```

```
void Update()
{
    transform.position += new Vector3(0.05f, 0, 0);
}
```

→ Play 버튼을 눌러보기 (캐릭터가 오른쪽으로 이동한다!)



→ transform의 의미: 캐릭터의 위치와 중, position을 계속 바꿔달라는 것

→ `transform.position += new Vector3(0.05f, 0, 0);`

→ 트랜스폼 안의 포지션을, Vector3 방향으로 계속 더해주세요

→ float 란? 소수점을 나타내는 자료형. 즉, 소수를 쓰고 싶으면 뒤에 f 를 붙여줘야 함

→ 위에 변수를 선언해서 이렇게 쓸 수도 있음!

```
float direction = 0.05f;

// Start is called before the first frame update
void Start()
{
}

// Update is called once per frame
void Update()
{
    transform.position += new Vector3(direction, 0, 0);
}
```

- 2) 캐릭터가 벽에 닿으면 다른 방향을 보게 하기

→ 2-0) Debug.Log 로 위치 보기

▼ [코드스니펫] Debug.log

```
Debug.Log(transform.position.x);
```

```
Debug.Log(transform.position.x);
```

C#

→ 2-1) 760보다 클 때 다른 방향 보게 하기

▼ [코드스니펫] 760보다 클 때 다른 방향 보게 하기

```
float direction = 0.05f;

// Start is called before the first frame update
void Start()
{

}

// Update is called once per frame
void Update()
{
    if (transform.position.x > 2.8f)
    {
        direction = -0.05f;
    }
    transform.position += new Vector3(direction, 0, 0);
}
```

```
float direction = 0.05f;

// Start is called before the first frame update
void Start()
{

}


// Update is called once per frame
void Update()
{
    if (transform.position.x > 2.8f)
    {
        direction = -0.05f;
    }
    transform.position += new Vector3(direction, 0, 0);
}
```

→ 2-2) 0보다 작을 때 다른 방향 보게 하기

```
float direction = 0.05f;
// Start is called before the first frame update
void Start()
{

}

// Update is called once per frame
void Update()
{
    if (transform.position.x > 2.8f)
    {
        direction = -0.05f;
    }
    if (transform.position.x < -2.8f)
    {
        direction = 0.05f;
    }
    transform.position += new Vector3(direction, 0, 0);
}
```

→  (직접 해보기) 2-3) 방향 전환하기

(힌트1: "유니티 2d 좌우반전하기"로 검색)

(힌트2: 이런 코드를 만나면 굿! `transform.localScale = new Vector3(-1, 1, 1);`)

▼ [코드스니펫] 방향 좌우반전하기

```
transform.localScale = new Vector3(-1, 1, 1);
```

▼ 펼치면 답!

```
float direction = 0.05f;
// Start is called before the first frame update
void Start()
{
}

// Update is called once per frame
void Update()
{
    if (transform.position.x > 2.8f)
    {
        direction = -0.05f;
        transform.localScale = new Vector3(-1, 1, 1);
    }
    if (transform.position.x < -2.8f)
    {
        direction = 0.05f;
        transform.localScale = new Vector3(1, 1, 1);
    }
    transform.position += new Vector3(direction, 0, 0);
}
```

▼ 5) 클릭 시 움직임 바꾸기

- 위의 코드를 조금만 예쁘게 다듬고,

```
float direction = 0.05f;
float toward = 1.0f;
// Start is called before the first frame update
void Start()
{
}

// Update is called once per frame
void Update()
{
    if (transform.position.x > 2.8f)
    {
        direction = -0.05f;
        toward = -1.0f;
    }
    if (transform.position.x < -2.8f)
    {
        direction = 0.05f;
        toward = 1.0f;
    }
    transform.localScale = new Vector3(toward, 1, 1);
    transform.position += new Vector3(direction, 0, 0);
}
```

- 마우스 클릭하면 → 움직이는 방향/이미지 방향 바꾸기

▼ [코드스니펫] 마우스 클릭시 방향 바꾸기

```
if (Input.GetMouseButtonDown(0))
{
    toward *= -1;
    direction *= -1;
}
```

```
if (Input.GetMouseButtonDown(0))
{
```

```
toward *= -1;
direction *= -1;
}
```

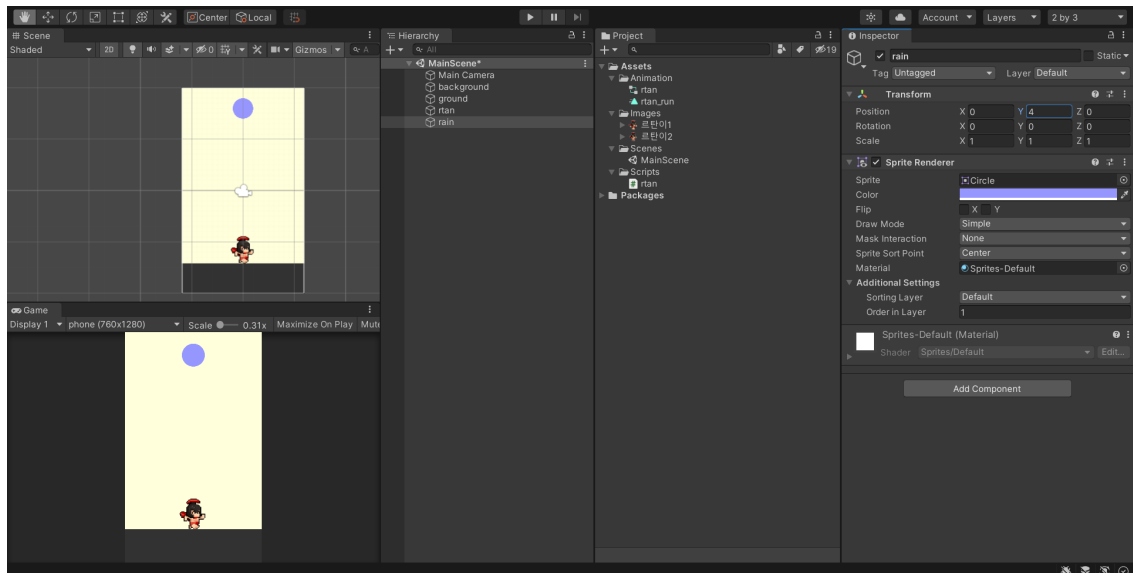
06. 빗방울 내리게 하기

▼ 1) 빗방울 특징

- 빗방울은 "하늘 랜덤한 위치에서 내림"
- 큰 / 중간 / 작은 빗방울 존재 (3-2-1점)
- 캐릭터와 부딪히면 점수 더하기

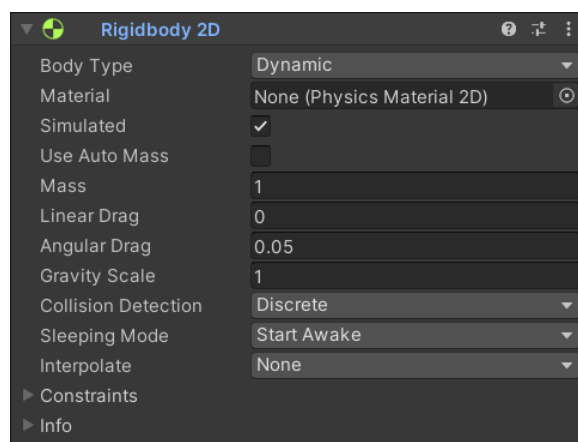
▼ 2) 빗방울 그리기

- Sprite → Circle 클릭 → rain 으로 이름 바꾸기
→ 색을 150,150,255,255 으로 맞추기
→ Position Y:4 세팅하기



▼ 3) 빗방울 떨어지게 하기

- rigidbody 2D를 달아 중력의 영향을 받게 하기



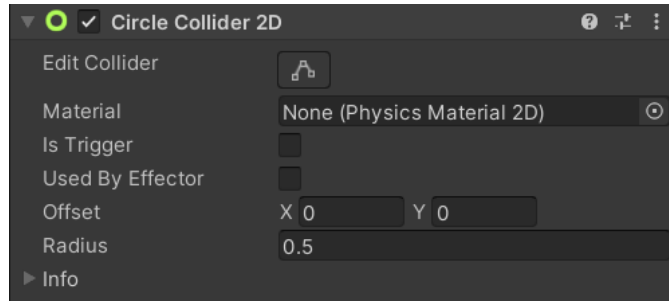
▼ 4) 땅에 닿으면 없어지게 하기(충돌 세팅)



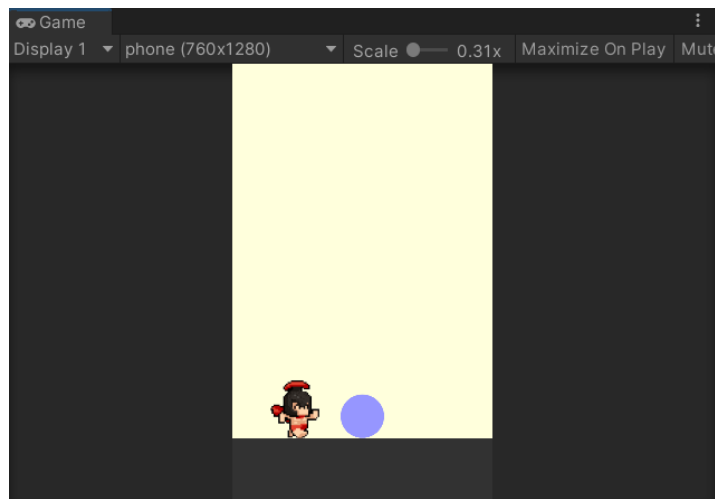
충돌의 기본 조건

: 둘 다 Collider 가 있어야 한다 / 둘 중 하나는 Rigidbody가 있어야 한다.

1. circle collider 2d를 달고, 반경 조정, 자세히 보면 초록색 선이 보임!

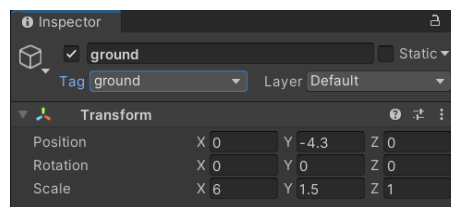
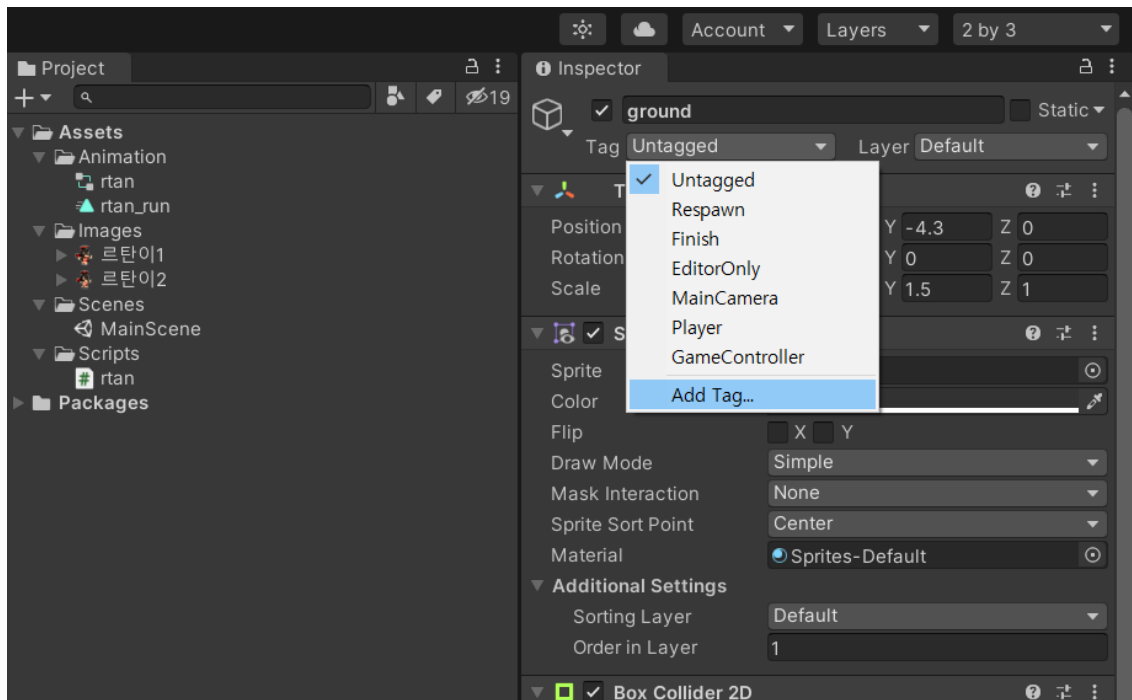


2. 바닥에도 box collider 2d를 달아주기
3. 게임을 실행하면 땅과 충돌을 합니다.



▼ 5) 땅에 닿으면 없어지게 하기(충돌 조작)

1. "땅"인지 알 수 있게, ground 라고 tag를 주기



2. 땅에 닿았는지 확인하기

- **rain** 스크립트 만들고, 빗방울에 붙이기
- **OnCollisionEnter2D** 함수는 다른 콜라이더에 부딪혔을 때 실행되는 내장함수
- coll (부딪힌 것의) tag 가 ground 이면!

▼ [코드스니펫] 땅에 닿았는지 확인하기

```
void OnCollisionEnter2D(Collision2D coll)
{
    if (coll.gameObject.tag == "ground")
    {
        Debug.Log("땅이다!");
    }
}
```

```
void OnCollisionEnter2D(Collision2D coll)
{
    if (coll.gameObject.tag == "ground")
    {
        Debug.Log("땅이다!");
    }
}
```

3. 비가 없어지게 하기

- Debug.Log 대신, **Destroy(gameObject)**
- gameObject는 나 자신

▼ [코드스니펫] 비가 없어지게 하기

```
Destroy(gameObject);
```

```

void OnCollisionEnter2D(Collision2D coll)
{
    if (coll.gameObject.tag == "ground")
    {
        Destroy(gameObject);
    }
}

```

07. 빗방울 랜덤하게 나타나게 하기

▼ 1) 랜덤하게 위치 잡아주기

- start() 함수에 랜덤 position 세팅하기

```

void Start()
{
    float x = Random.Range(-2.7f, 2.7f);
    float y = Random.Range(3.0f, 5.0f);
    transform.position = new Vector3(x, y, 0);
}

```

▼ 2) 랜덤하게 사이즈(큰/중간/작은) 잡아주기

- 어떤 사이즈로 나올지 생각하고 →

사이즈 변경: `transform.localScale = new Vector3(size, size, 0);`

색 변경: `GetComponent<SpriteRenderer>().color = new Color(100 / 255f, 100 / 255f, 255 / 255f, 255 / 255f);` (참고 : 255.0f 로 나눠주는 게 핵심!)

```

int type;
float size;
int score;

// Start is called before the first frame update
void Start()
{
    float x = Random.Range(-2.7f, 2.7f);
    float y = Random.Range(3.0f, 5.0f);
    transform.position = new Vector3(x, y, 0);

    type = Random.Range(1, 4);

    if (type == 1)
    {
        size = 1.2f;
        score = 3;
        GetComponent<SpriteRenderer>().color = new Color(100 / 255f, 100 / 255f, 255 / 255f, 255 / 255f);
    }
    else if (type == 2)
    {
        size = 1.0f;
        score = 2;
        GetComponent<SpriteRenderer>().color = new Color(130 / 255f, 130 / 255f, 255 / 255f, 255 / 255f);
    }
    else
    {
        size = 0.8f;
        score = 1;
        GetComponent<SpriteRenderer>().color = new Color(150 / 255f, 150 / 255f, 255 / 255f, 255 / 255f);
    }

    transform.localScale = new Vector3(size, size, 0);
}

```

▼ [코드스니펫] type이 1일 때

```
size = 1.2f;
score = 3;
GetComponent<SpriteRenderer>().color = new Color(100 / 255f, 100 / 255f, 255 / 255f, 255 / 255f);
```

▼ [코드스니펫] type이 2일 때

```
size = 1.0f;
score = 2;
GetComponent<SpriteRenderer>().color = new Color(130 / 255f, 130 / 255f, 255 / 255f, 255 / 255f);
```

▼ [코드스니펫] type이 3일 때

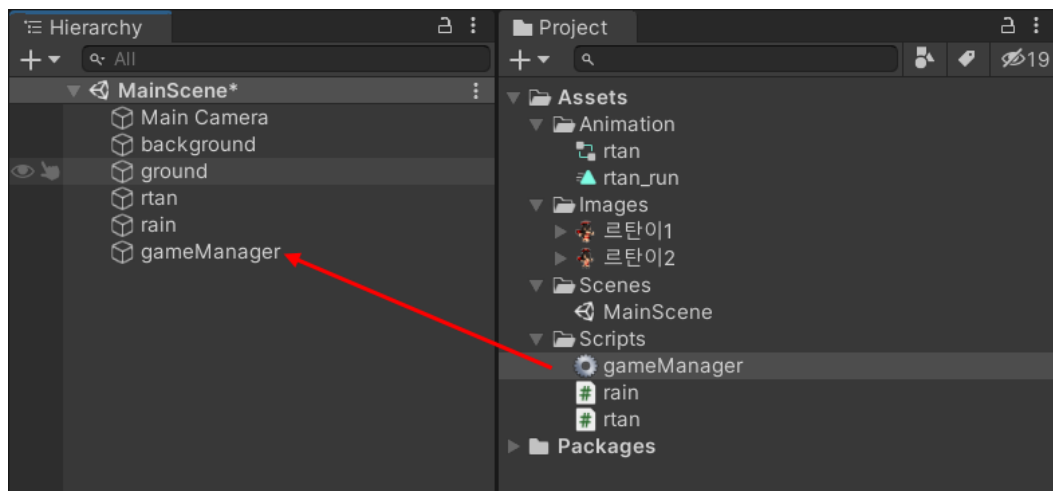
```
size = 0.8f;
score = 1;
GetComponent<SpriteRenderer>().color = new Color(150 / 255f, 150 / 255f, 255 / 255f, 255 / 255f);
```

08. 빗방울 계속 나오게 하기

▼ 1) GameManager 만들기

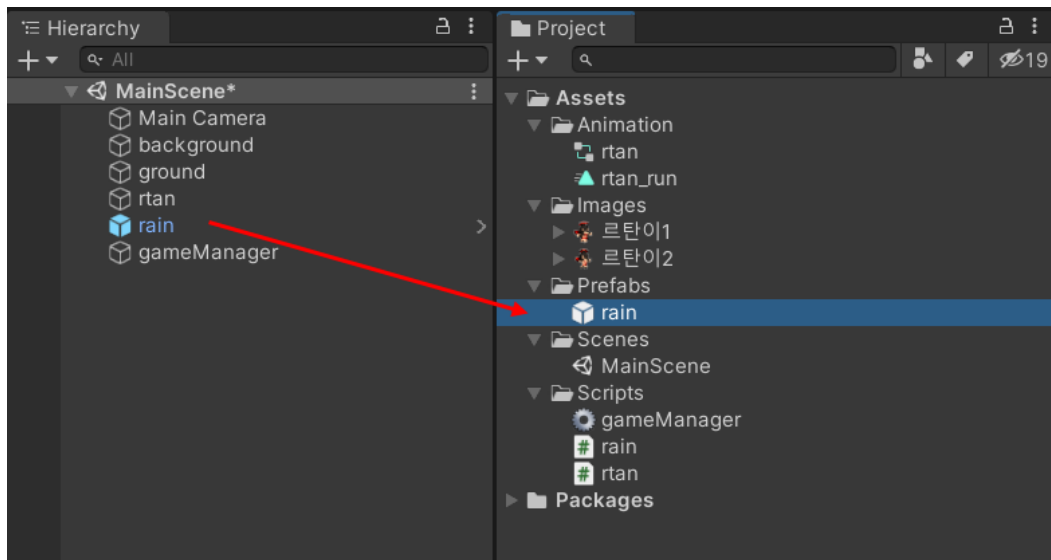
👉 GameManager란? 게임 전체를 조율하는 오브젝트!

- 예) 점수 / 다시 시작 / 3번 째 다시 시작에 부스터 / 광고보기 등
- 빈 곳에 object를 만들고 "gameManager"로 만들어둡니다.
- 마찬가지로 스크립트도 만들어 붙입니다. (어떻게 알았는지 아이콘 모양이 다르네요!)



▼ 2) 빗방울 복제하기 - Prefabs

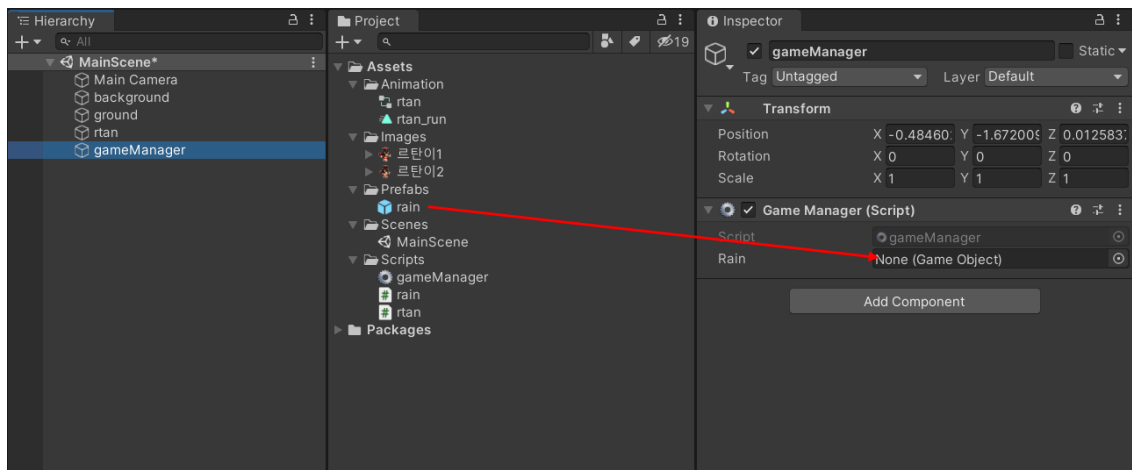
- 프리팹 구현하기 (폴더 만들고 끌어다 놓기 & 오브젝트는 삭제)



▼ 3) 빗방울 복제하기 - Instantiate

1. gameManager에서 : 빗방울을 받기 + 프리팹 끌어다놓기

```
public GameObject rain;
```



2. 0.5초마다 한번씩 실행되는 코드

▼ [코드스니펫] InvokeRepeating 함수

```
InvokeRepeating("makeRain", 0, 0.5f);
```

▼ [코드스니펫] makeRain 함수

```
void makeRain()
{
    Debug.Log("비를 내려라!");
}
```

```
void Start()
{
    InvokeRepeating("makeRain", 0, 0.5f);
}

void makeRain()
```

```
{
    Debug.Log("비를 내려라!");
}
```

3. 빗방울 프리팹을 복제하기

▼ [코드스니펫] Instantiate 함수

```
Instantiate(rain);
```

```
void makeRain()
{
    Instantiate(rain);
}
```

09. 점수 올라가게 하기

▼ 1) 점수 보드 만들기



[알고 가야 할 것]

UI는 Canvas라는 도화지 위에 그려지고, 카메라 위치와는 관계가 없이 보여집니다.

→ 버튼 / 텍스트 / 순위를 보여줄 때에만 써줍니다. 😊

1. 폰트 적용하기

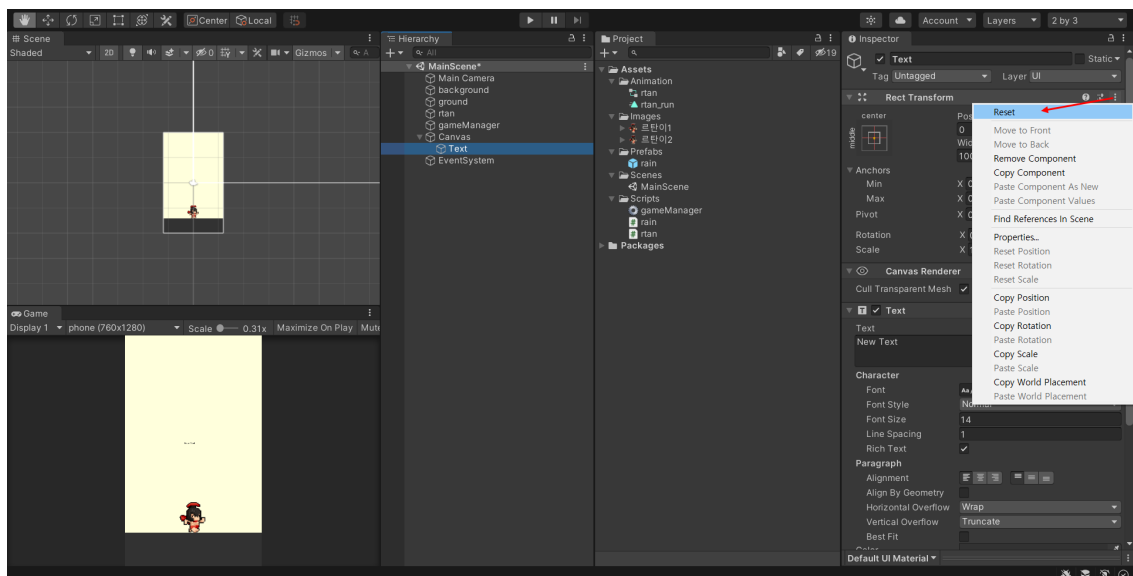
→ Assets에 fonts 폴더 만들고 옮겨두기

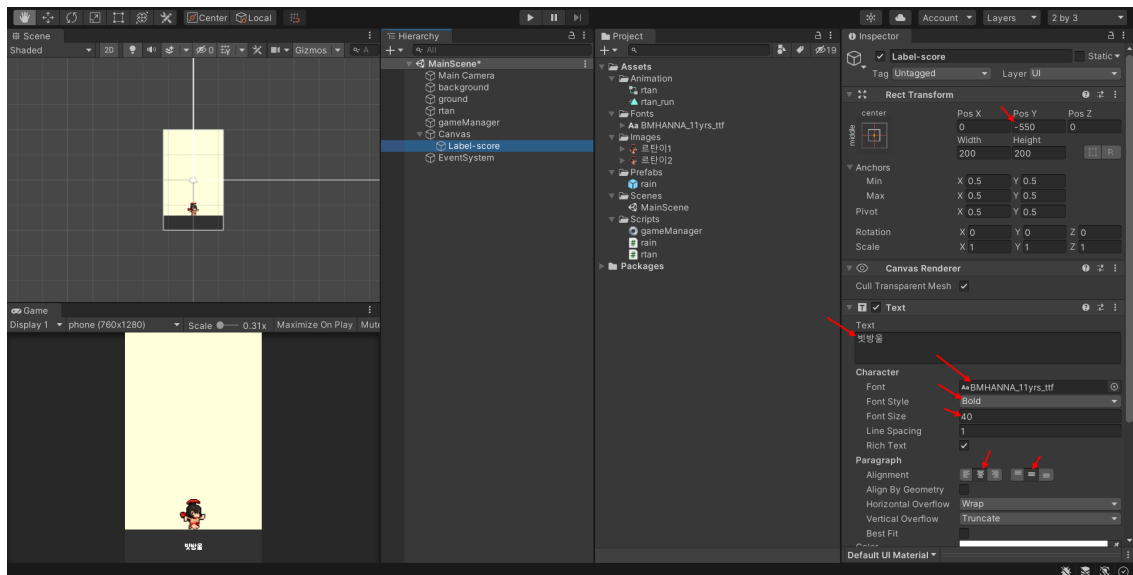
▼ [코드스니펫] 배민-한나체

http://pop.baemin.com/fonts/hanna11yrs/BMHANNA_11yrs_ttf.ttf

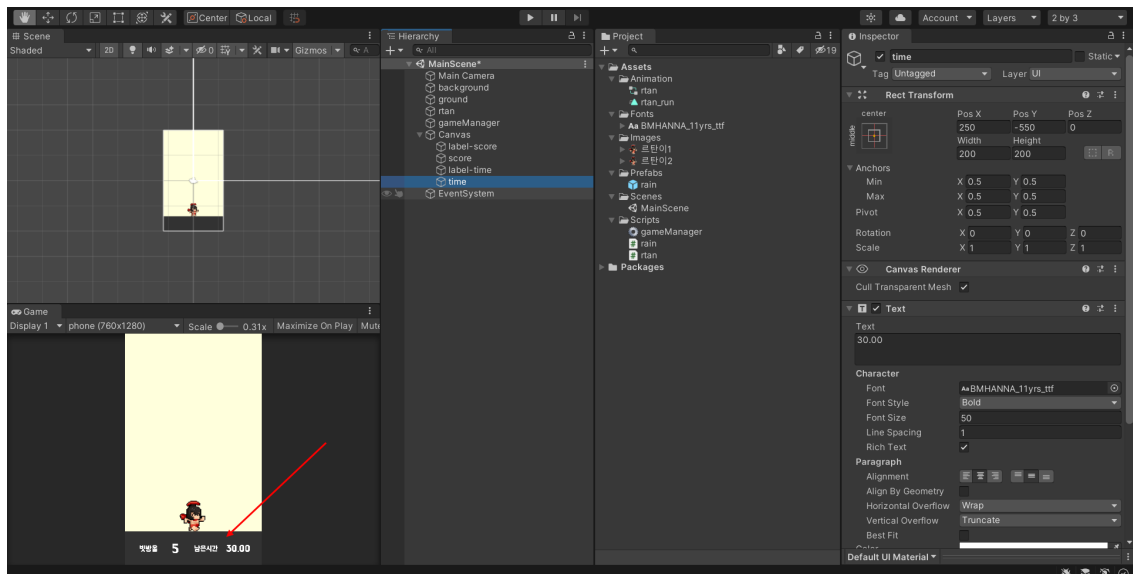
2. Sprite vs UI 그리고 Canvas

→ UI → Text 클릭 → 아래 설정을 따라하기 (폰트사이즈, 위치 등)





3. text를 네 번 복사 → 붙여넣기 해서 아래와 같이 맞추기



▼ 2) gameManager - 싱글톤 화

👉 싱글톤이란? 어디서든 부를 수 있는 '하나'로 만들어주는 것! 곧 보게 되실 거예요!

▼ [코드스니펫] 싱글톤 화

```
public static GameManager I;

void Awake()
{
    I = this;
}
```

```
public static GameManager I;

void Awake()
{
    I = this;
}
```

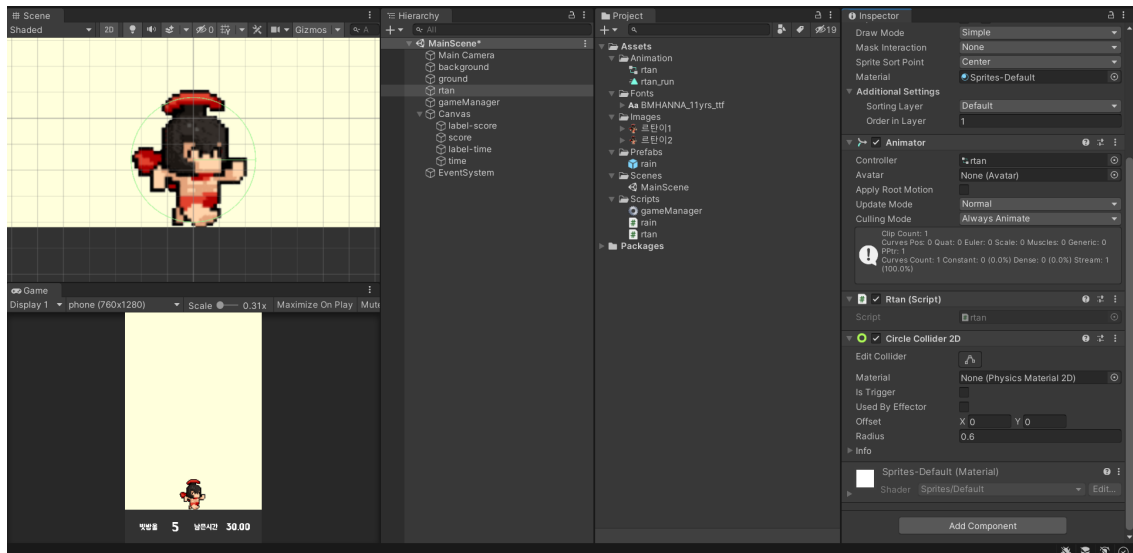
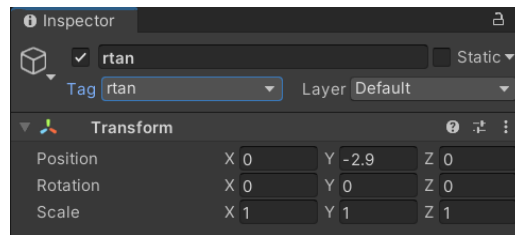
▼ 3) GameManager - 점수 올라가는 함수 만들기

```
int totalScore = 0;

public void addScore(int score)
{
    totalScore += score;
}
```

▼ 4) 빗방울 - 캐릭터에 맞으면 점수 올라가게 하기

1. 캐릭터에 tag 주기 + collider 주기

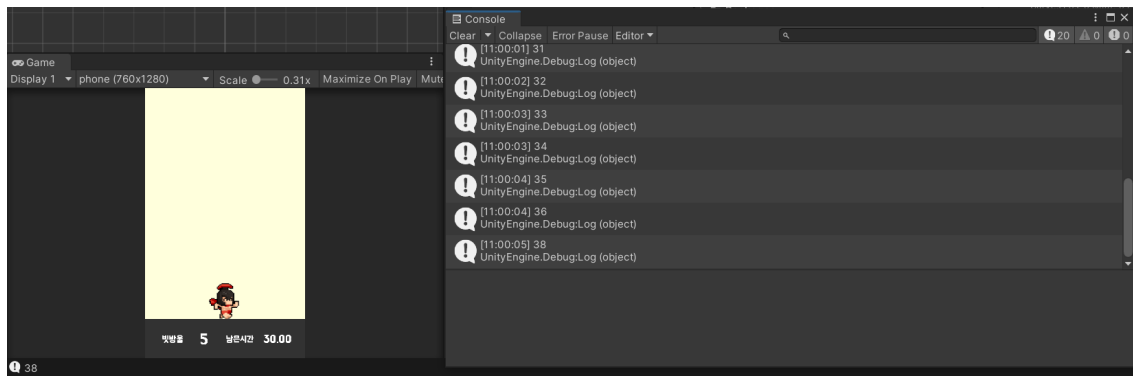


2. 빗방울 - 캐릭터에 맞으면 점수 올라가고 + 사라지기

```
void OnCollisionEnter2D(Collision2D coll)
{
    if (coll.gameObject.tag == "ground")
    {
        Destroy(gameObject);
    }

    if (coll.gameObject.tag == "rtan")
    {
        GameManager.I.addScore(score);
        Destroy(gameObject);
    }
}
```

3. GameManager - addScore 함수에 Debug.Log를 걸어서 확인 → 잘된다!



▼ 5) gameManager - 올라가는 점수 표기하기

1. UI Text 받기

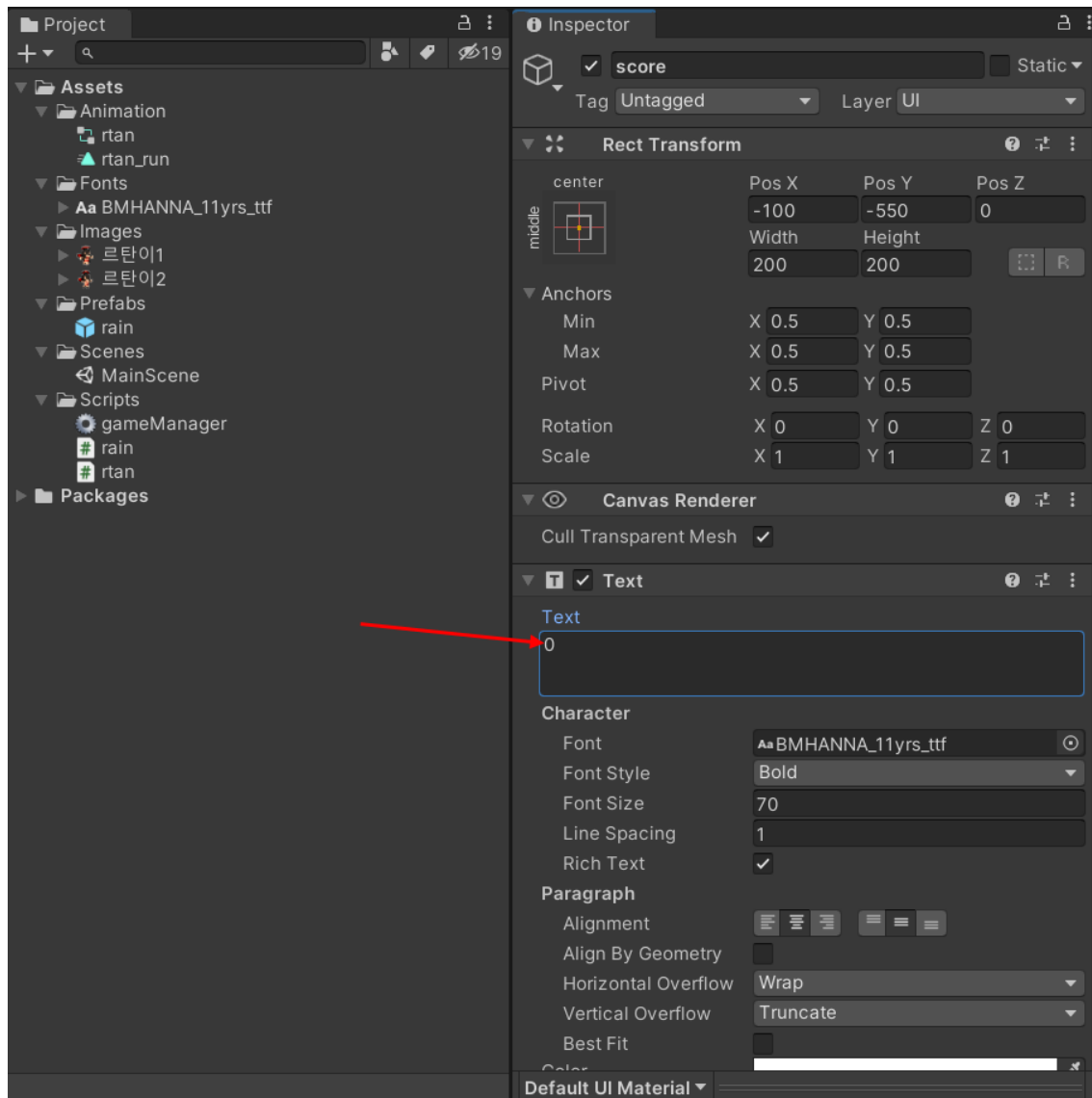
```
using UnityEngine.UI;
```

```
public Text scoreText;
```

2. Text 바꿔주기

```
public void addScore(int score)
{
    totalScore += score;
    scoreText.text = totalScore. ();
}
```

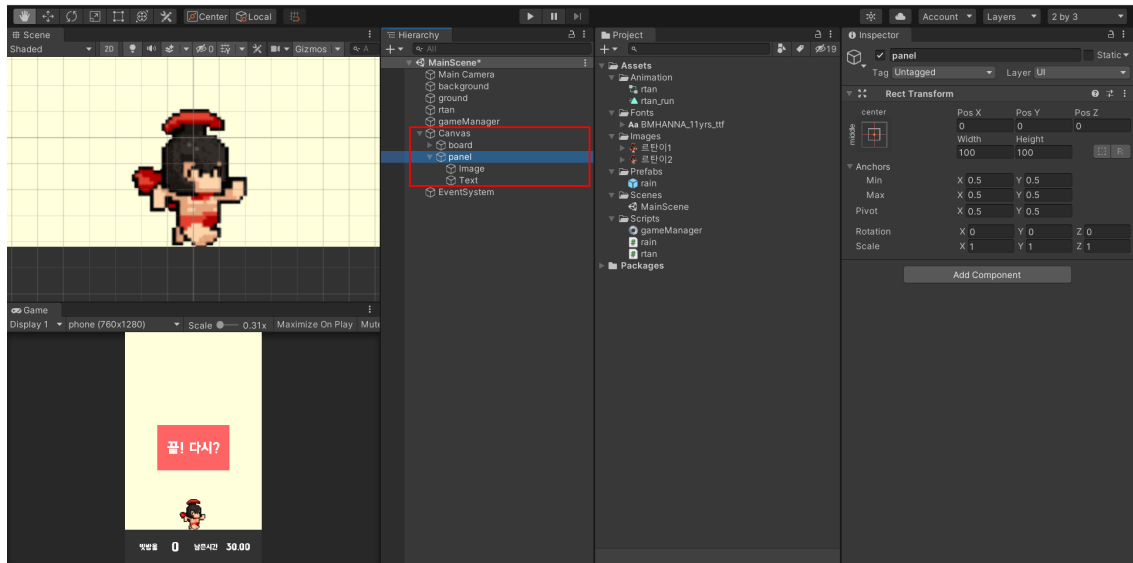
3. 처음 스코어는 0으로 만들어주기



10. 게임 끝내기

▼ 1) Retry 패널 만들기

- 정리하기 : 아래와 같이 세팅하기
 - image 사이즈: 400 / 250
 - txt 사이즈: 80
 - 글자 색상 (255, 255, 255, 255)
 - 배경 색상 (232, 52, 78, 255)
 - Inactive로 만들어두기



▼ 2) gameManager - 시간이 가게 하기

1. 시간이 흐르게 하기

```
void Update()
{
    limit -= Time.deltaTime;
    timeTxt.text = timeLimit.ToString("N2");
}
```

2. 멈추게 하기

```
void Update()
{
    limit -= Time.deltaTime;
    if (limit < 0)
    {
        Time.timeScale = 0.0f;
        limit = 0.0f;
    }
    timeTxt.text = timeLimit.ToString("N2");
}
```

▼ 3) 0초에 Retry 패널 나오게하기

1. Panel 받기

```
public GameObject panel;
```

2. Panel 나오게 하기

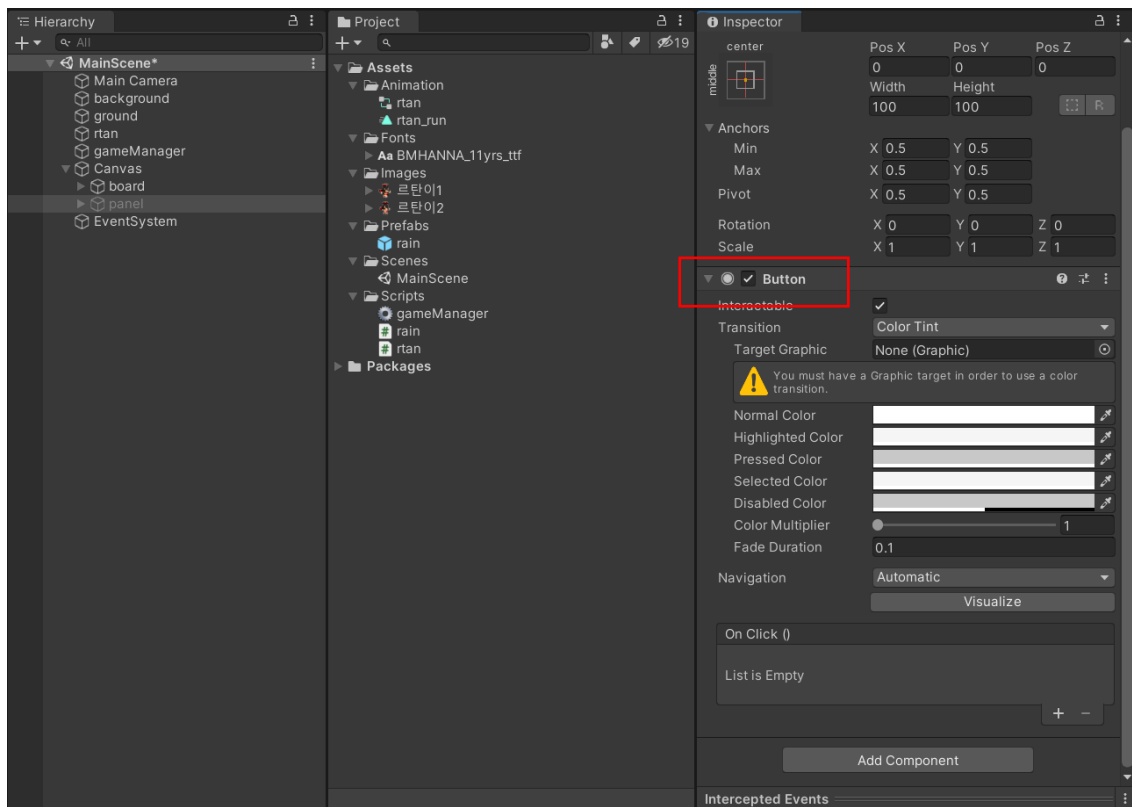
```
void Update()
{
    limit -= Time.deltaTime;

    if (limit < 0)
    {
        limit = 0.0f;
        panel.SetActive(true);
        Time.timeScale = 0.0f;
    }

    timeText.text = limit.ToString("N2");
}
```

▼ 4) 판넬 클릭하면 다시 시작하게 하기

1. 판넬에 button 달기



2. 씬 불러오는 것은 중앙에서 해야할 일! (gameManager.cs)

```
using UnityEngine.SceneManagement;

public void retry()
{
    SceneManager.LoadScene("MainScene");
}
```

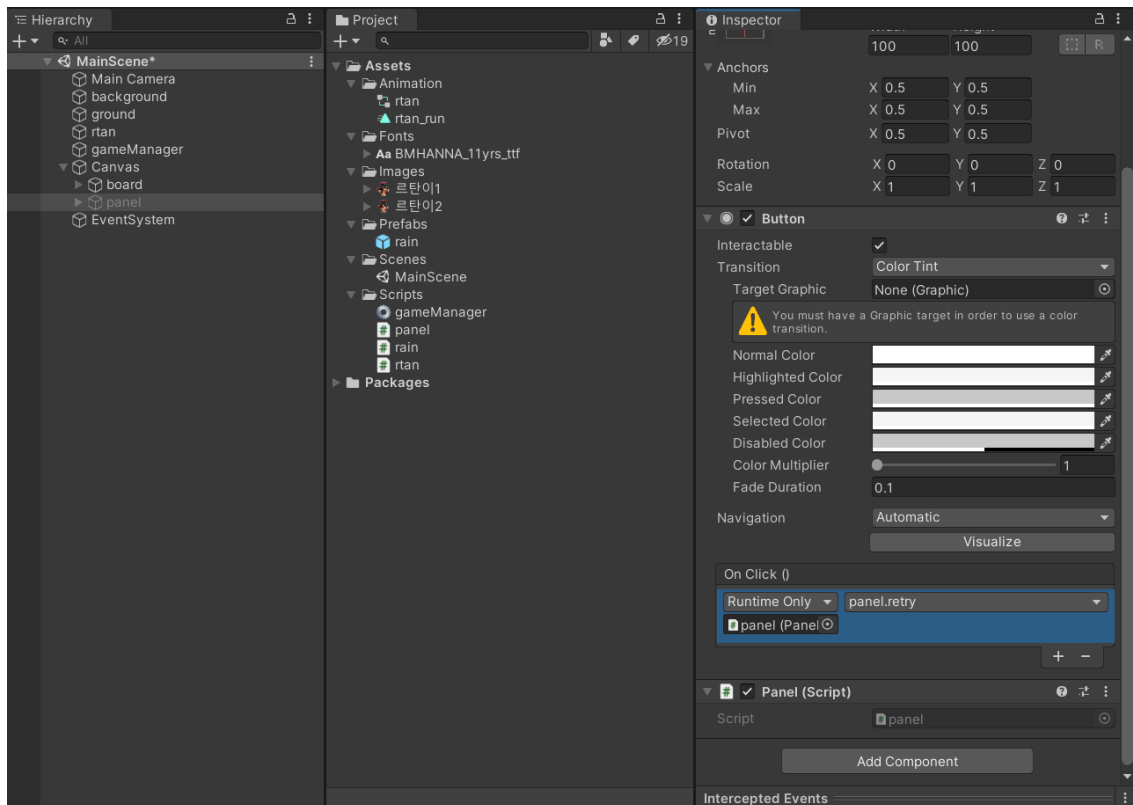
3. 클릭하면 작동 할 함수 만들기 (panel.cs)

▼ [코드스니펫] panel.cs

```
public void retry()
{
    gameManager.I.retry();
}
```

```
public void retry()
{
    gameManager.I.retry();
}
```

4. onclick 연결하기



▼ 5) gameManager - 초기화 함수를 만들기

- 초기화 해야 할 요소들
→ timeScale, timeLimit, totalScore

```
void Start()
{
    InvokeRepeating("makeRain", 0, 0.5f);
    initGame();
}
void initGame()
{
    Time.timeScale = 1.0f;
    totalScore = 0;
    limit = 30.0f;
}
```

▼ 6) 수업 전체 코드

👉 Github에서 pull 받아서 unity에서 실행하기

- <https://github.com/bumkyulee/sparta-raindrop.git>

11. 숙제 - 빨강 빗방울 만들기

📄 맞으면 -5 점이 되는 빨간(rgb = 255,100,255) 빗방울을 만들어보세요!

- 사이즈는 0.8로 해주세요!
- 색은 `new Color(255 / 255.0f, 100.0f / 255.0f, 100.0f / 255.0f, 255.0f / 255.0f);` 이렇게!
- ▼ 이렇게 되면 완성!

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/def83cfb-9cae-4d9f-9253-bec5d0e0f8b0/빗물받는르탄이_숙제완성.mp4

▼ 힌트요정 - 🧚‍♀️

- `rain.cs` 만 수정하면 된답니다!
- `type = Random.Range(..` 여기부터, `if else` 까지!

HW. 1주차 숙제 해설

▼ new Color

```
new Color(255 / 255.0f, 100.0f / 255.0f, 100.0f / 255.0f, 255.0f / 255.0f);
```

▼ `rain.cs` 코드

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

public class rain : MonoBehaviour
{
    int type;
    float size;
    int score;

    // Start is called before the first frame update
    void Start()
    {
        float x = Random.Range(-2.7f, 2.7f);
        float y = Random.Range(3.0f, 5.0f);

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

        type = Random.Range(1, 5);
        if (type == 1)
        {
            size = 1.2f;
            score = 3;

            GetComponent().color = new Color(100.0f / 255.0f, 100.0f / 255.0f, 255.0f / 255.0f, 255.0f / 255.0f);
        }
        else if (type == 2)
        {
            size = 1.0f;
            score = 2;

            GetComponent().color = new Color(130.0f / 255.0f, 130.0f / 255.0f, 255.0f / 255.0f, 255.0f / 255.0f);
        }
        else if (type == 3)
        {
            size = 0.8f;
            score = 1;

            GetComponent().color = new Color(150.0f / 255.0f, 150.0f / 255.0f, 255.0f / 255.0f, 255.0f / 255.0f);
        }
        else
        {
            size = 0.8f;
            score = -5;

            GetComponent().color = new Color(255 / 255.0f, 100.0f / 255.0f, 100.0f / 255.0f, 255.0f / 255.0f);
        }
        transform.localScale = new Vector3(size, size, 0);
    }

    // Update is called once per frame
    void Update()
    {
    }

    void OnCollisionEnter2D(Collision2D coll)
    {
    }
}
```



```
        if (coll.gameObject.tag == "ground")
        {
            Destroy(gameObject);
        }
        if (coll.gameObject.tag == "rtan")
        {
            gameManager.I.addScore(score);
            Destroy(gameObject);
        }
    }
}
```

[이전 주차](#)

[다음 주차](#)

Copyright © TeamSparta All rights reserved.