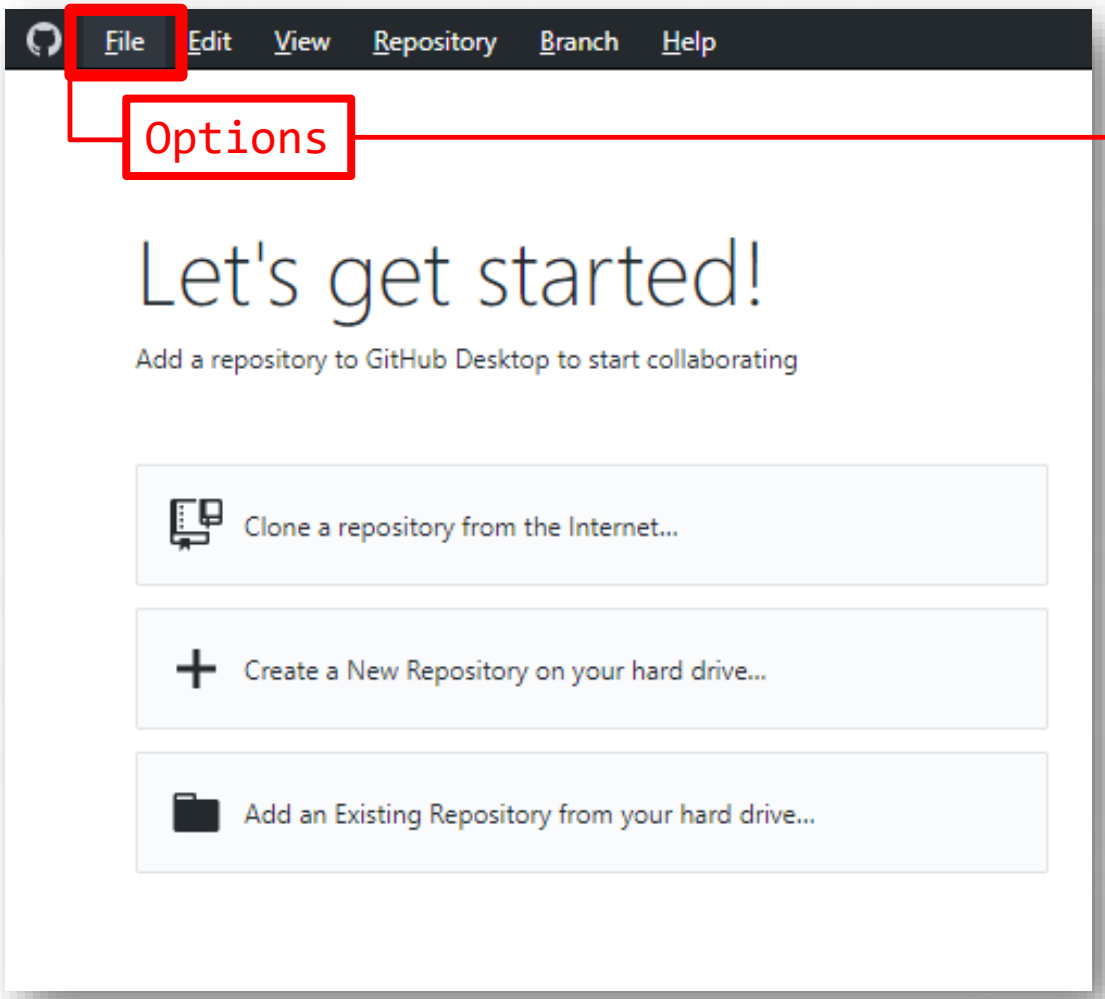
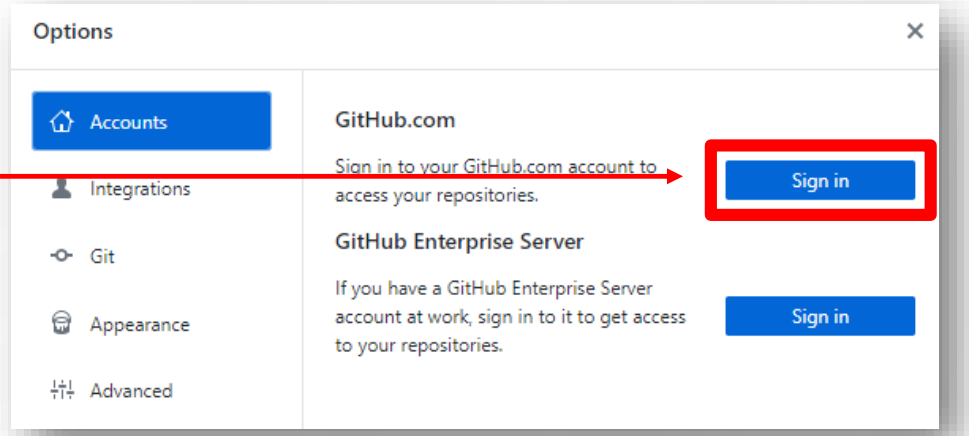
[7주차] 2D RunGame-4

인천정보과학고등학교 3학년 전산과 게임프로그래밍2

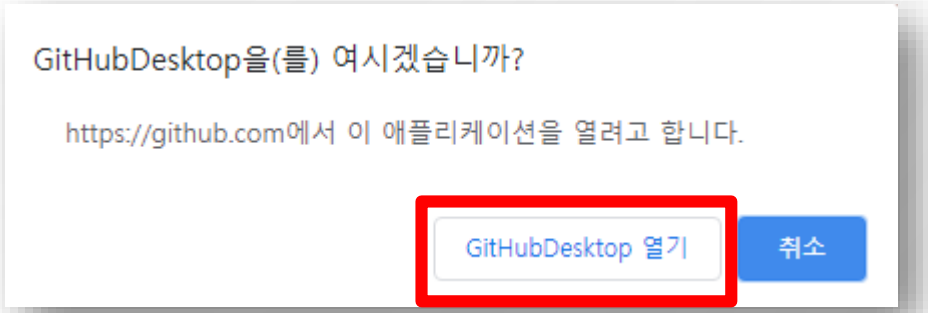
GITHUB CLONE

Git Desktop Login

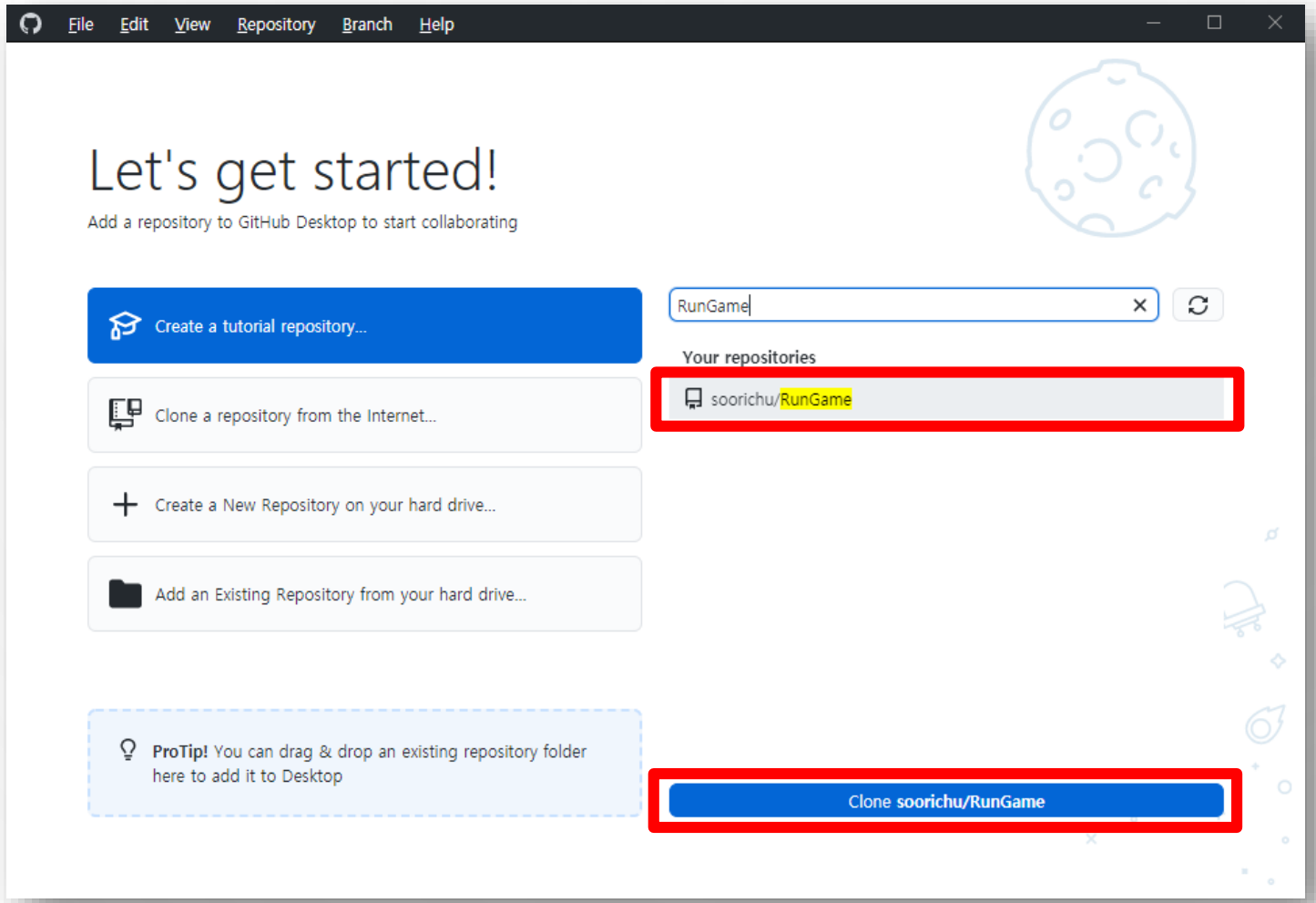




깃헙 로그인 진행
크롬에서 다음과 같은 메시지가 나오면



Clone Repository

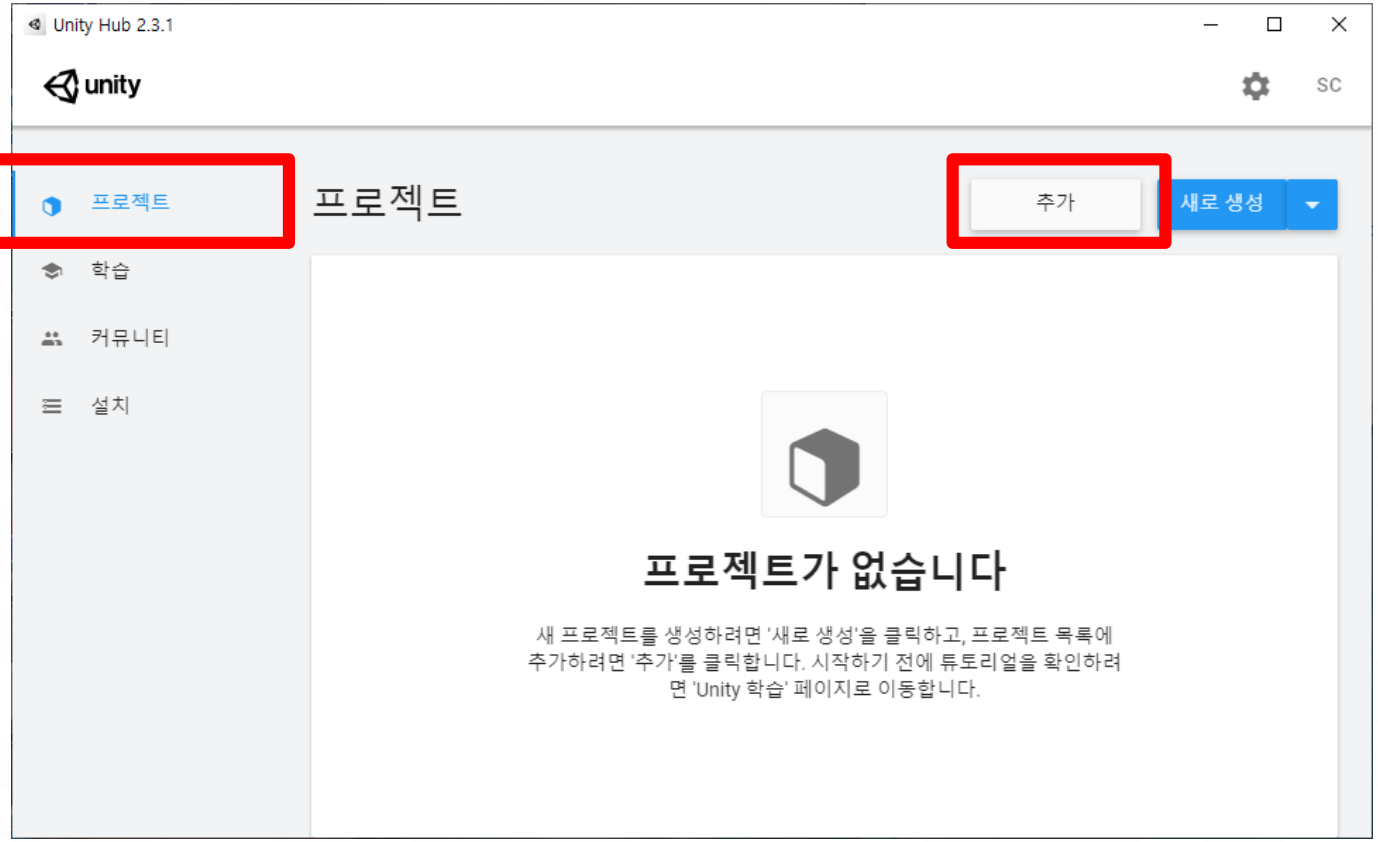


<user>/RunGame 선택

Clone <user>/RunGame 클릭

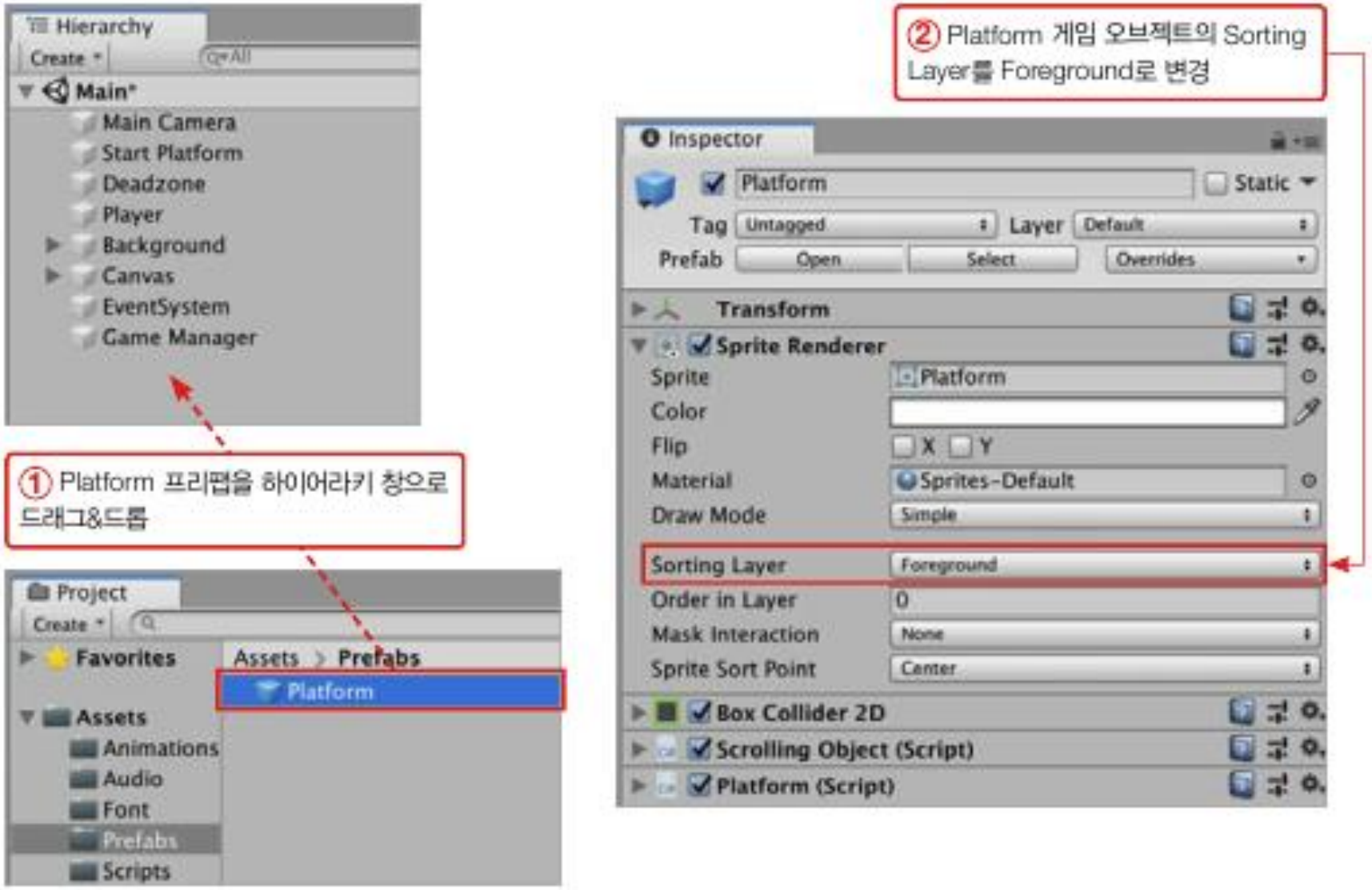
Project Open

- Unity Hub에서 프로젝트 > 추가 > RunGame 폴더 선택

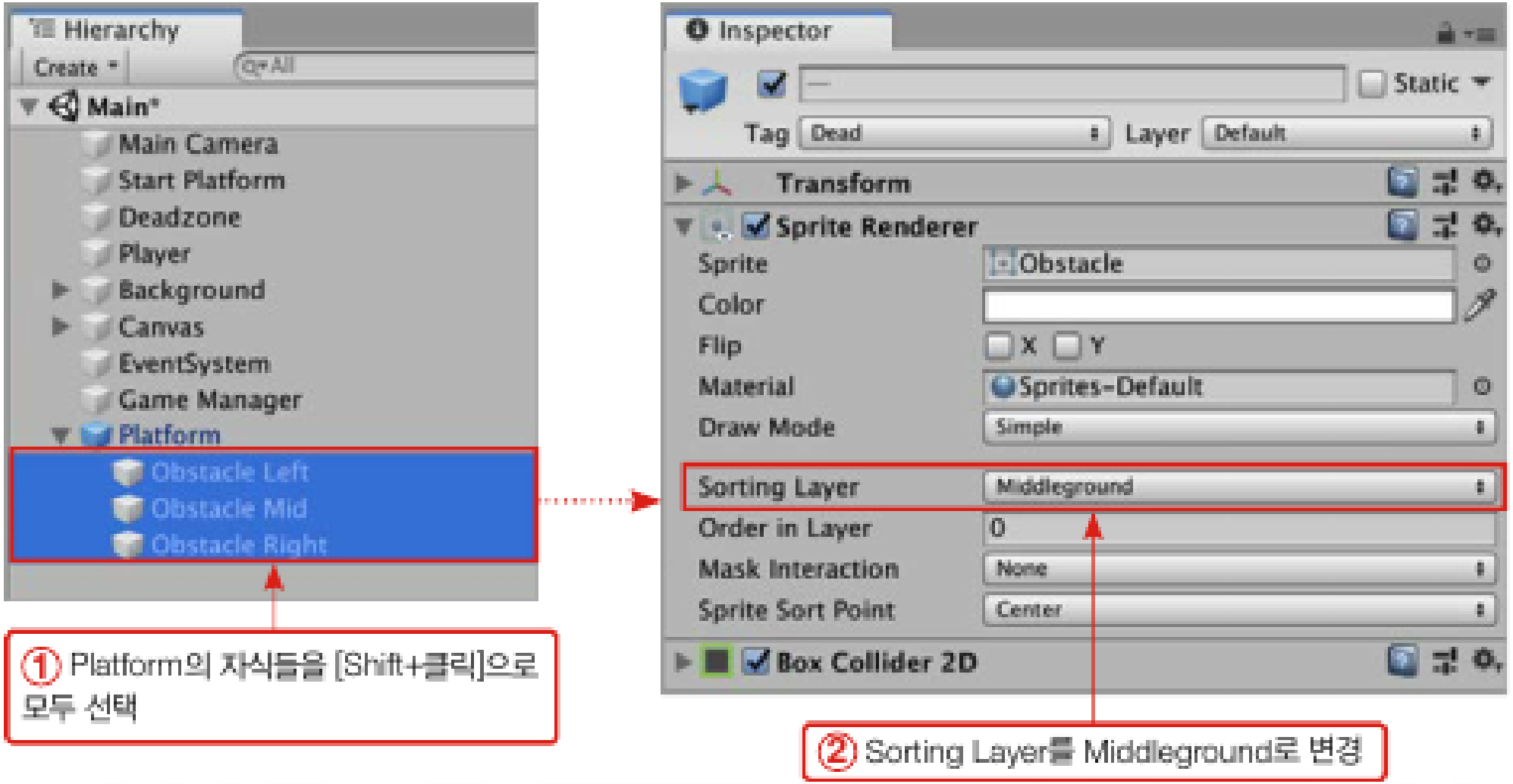


발판 만들기

Platform 프리팹 배치하기(1)



Platform 프리팹 배치하기(2)



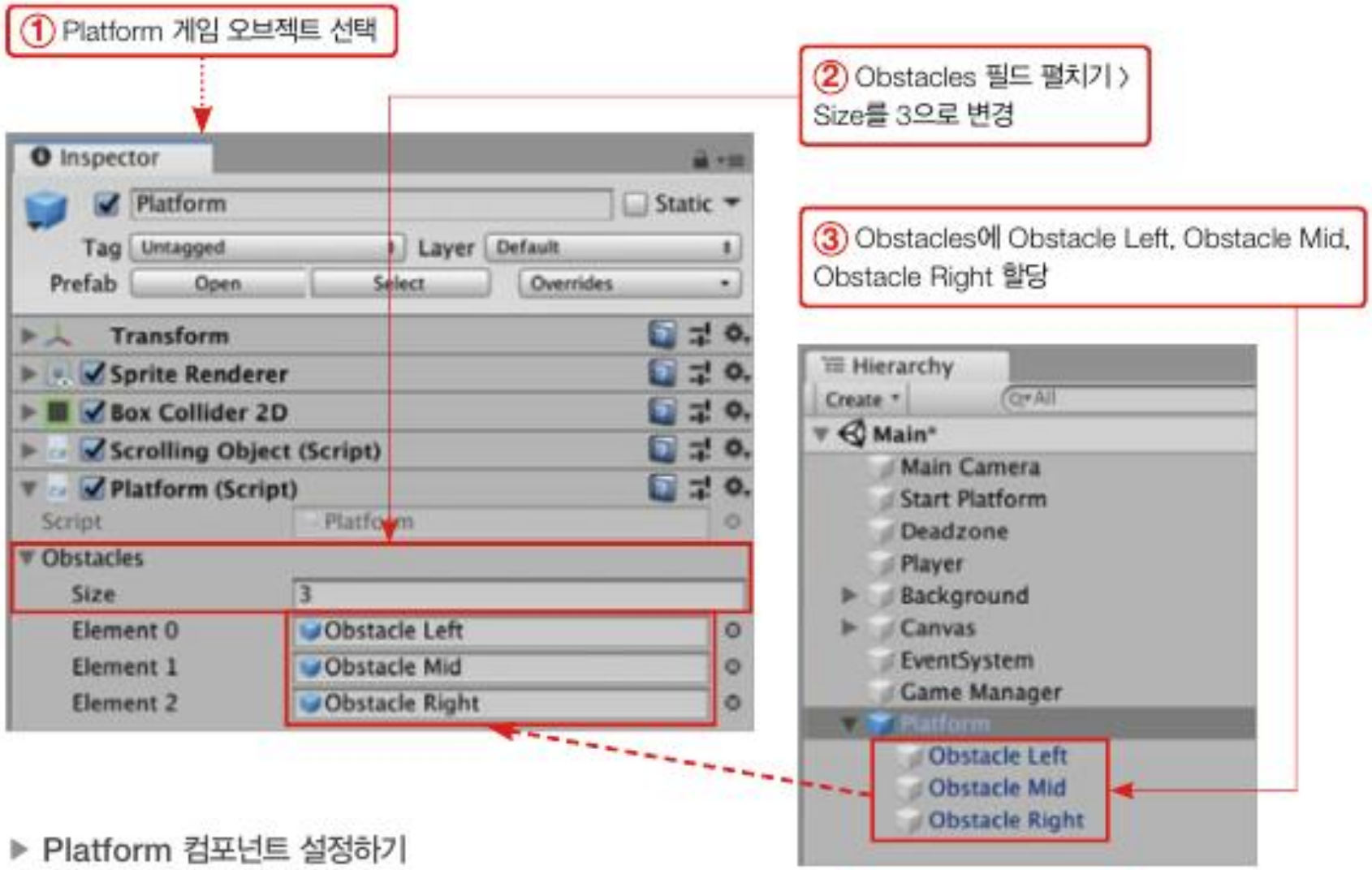
Platform 스크립트(1)

```
using UnityEngine;

// 발판으로서 필요한 동작을 담은 스크립트
public class Platform : MonoBehaviour {
    public GameObject[] obstacles; // 장애물 오브젝트들
    private bool stepped = false; // 플레이어 캐릭터가 밟았는가

    // 컴포넌트가 활성화될 때마다 매번 실행되는 메서드
    private void OnEnable() {
        // 밟힘 상태를 리셋
        stepped = false;

        // 장애물의 수만큼 루프
        for (int i = 0; i < obstacles.Length; i++)
        {
            // 현재 순번의 장애물을 1/3의 확률로 활성화
            if (Random.Range(0, 3) == 0)
            {
                obstacles[i].SetActive(true);
            }
```

Platform 스크립트(2)

```
            else
            {
                obstacles[i].SetActive(false);
            }
        }
    }
    void OnCollisionEnter2D(Collision2D collision) {
        // 충돌한 상대방의 태그가 Player이고 이전에 플레이어 캐릭터가 밟지 않았다면
        if (collision.collider.tag == "Player" && !stepped)
        {
            // 점수를 추가하고, 밟힘 상태를 참으로 변경
            stepped = true;
            GameManager.instance.AddScore(1);
        }
    }
}
```

Platform 컴포넌트 설정하기

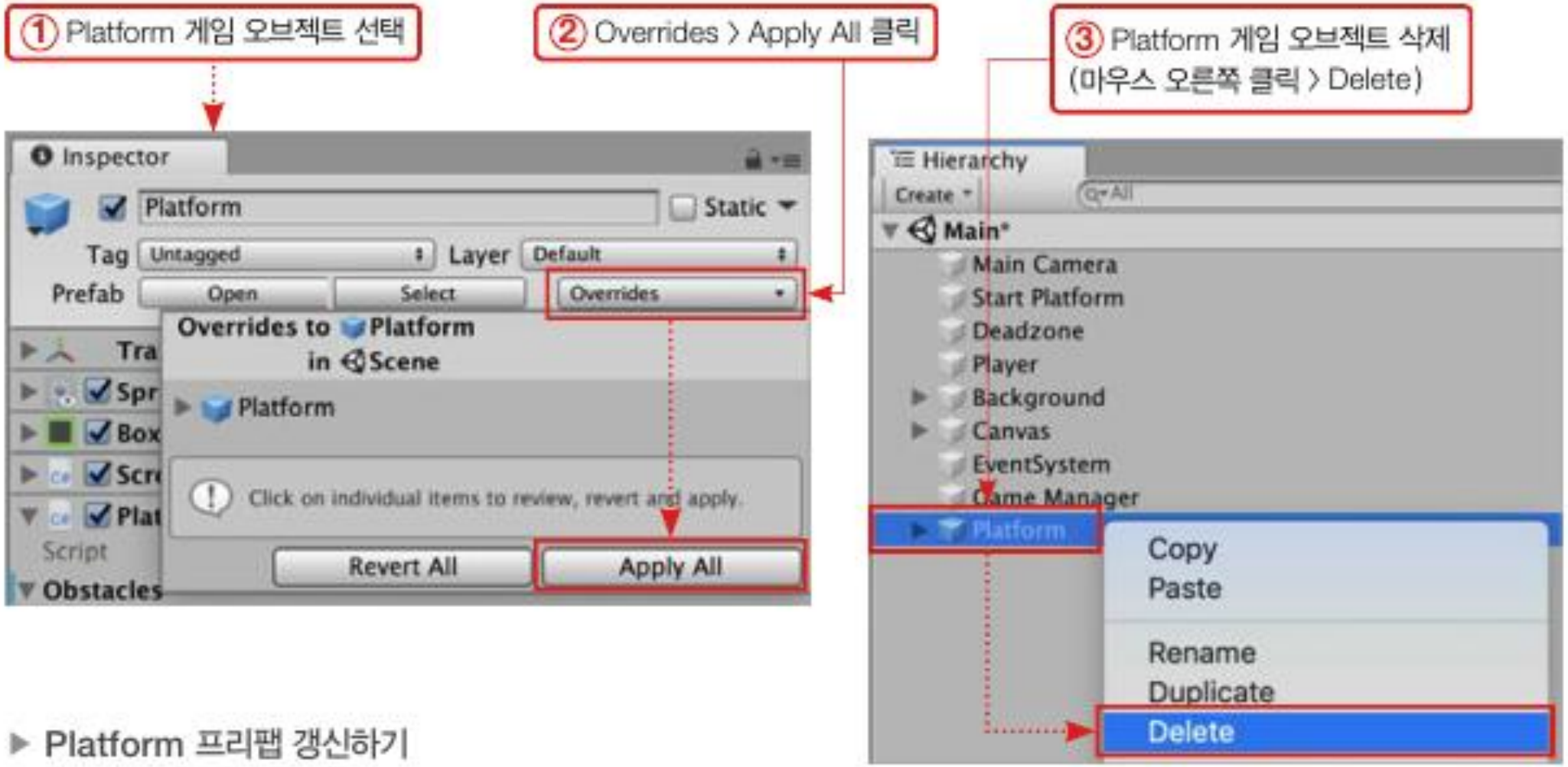


▶ Platform 컴포넌트 설정하기

ScrollingObject 스크립트 완성하기

```
using UnityEngine;

// 게임 오브젝트를 계속 왼쪽으로 움직이는 스크립트
public class ScrollingObject : MonoBehaviour {
    public float speed = 10f; // 이동 속도

    private void Update() {
        // 초당 speed의 속도로 왼쪽으로 평행이동
        transform.Translate(Vector3.left * speed * Time.deltaTime);
    }
}
```

Platform 프리팹 갱신하기

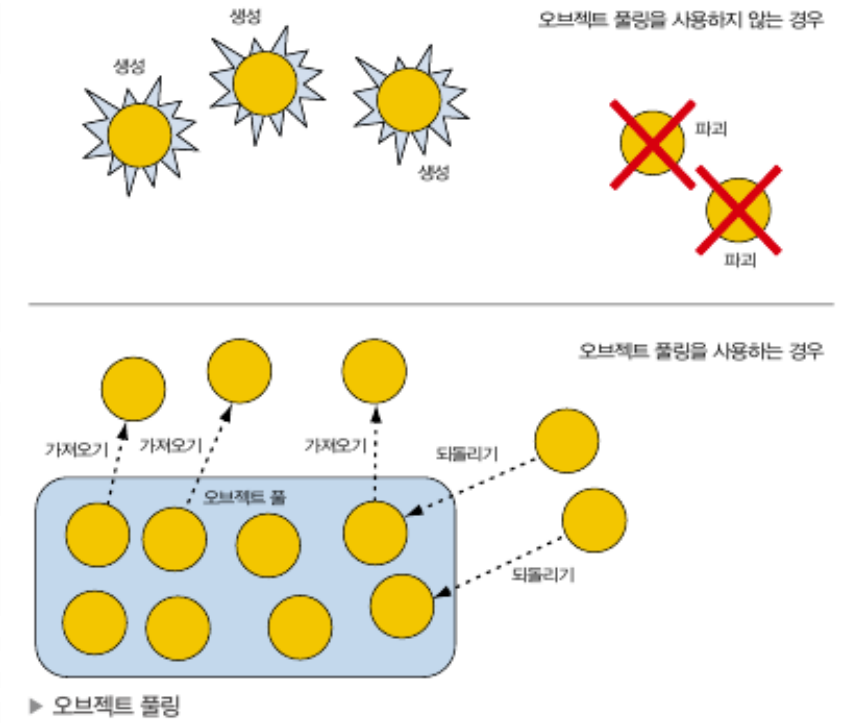


▶ Platform 프리팹 갱신하기

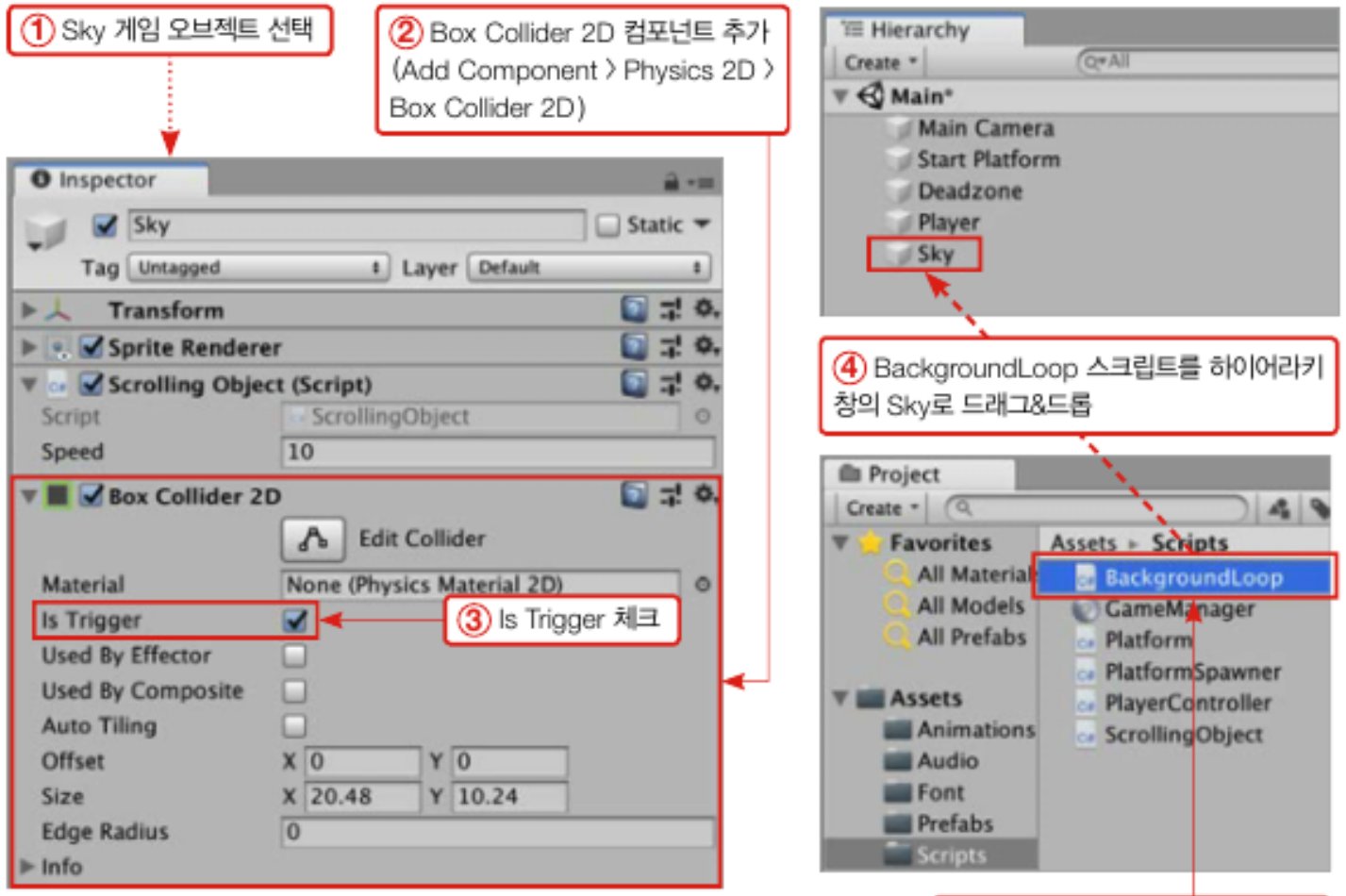
오브젝트 풀링

오브젝트 풀링이란?

- 초기에 필요한 만큼 오브젝트를 만들어 풀(Pool)에 쌓아두는 방식
- Inistantiate()나 Destroy()는 실시간으로 생성/파괴 하므로 메모리를 많이 잡아 먹음
- 오브젝트 풀링을 사용하면 게임 끊김 현상이 줄어 들음.



BoxCollider2D와 BackgroundLoop



▶ 박스 콜라이더 2D와 BackgroundLoop 준비하기

발판 무한 배치 과정

발판 무한 배치 과정

게임 화면

PlatformSpawner 스크립트(1)

```
using UnityEngine;

// 발판을 생성하고 주기적으로 재배치하는 스크립트
public class PlatformSpawner : MonoBehaviour {
    public GameObject platformPrefab; // 생성할 발판의 원본 프리팹
    public int count = 3; // 생성할 발판 수

    public float timeBetSpawnMin = 1.25f; // 다음 배치까지의 시간 간격 최솟값
    public float timeBetSpawnMax = 2.25f; // 다음 배치까지의 시간 간격 최댓값
    private float timeBetSpawn; // 다음 배치까지의 시간 간격

    public float yMin = -3.5f; // 배치할 위치의 최소 y 값
    public float yMax = 1.5f; // 배치할 위치의 최대 y 값
    private float xPos = 20f; // 배치할 위치의 x 값

    private GameObject[] platforms; // 미리 생성한 발판들
    private int currentIndex = 0; // 사용할 현재 순번의 발판

    // 초반에 생성한 발판을 화면 밖에 숨겨둘 위치
    private Vector2 poolPosition = new Vector2(0, -25);
    private float lastSpawnTime; // 마지막 배치 시점
```

PlatformSpawner 스크립트(2)

```
// 변수를 초기화하고 사용할 발판을 미리 생성
void Start() {
    // count만큼의 공간을 가지는 새로운 발판 배열 생성
    platforms = new GameObject[count];

    // count만큼 루프하면서 발판 생성
    for (int i = 0; i < count; i++)
    {
        // platformPrefab을 원본으로 새 발판을 poolPosition 위치에 복제 생성
        // 생성된 발판을 platform 배열에 할당
        platforms[i] = Instantiate(platformPrefab, poolPosition, Quaternion.identity);
    }

    // 마지막 배치 시점 초기화
    lastSpawnTime = 0f;
    // 다음번 배치까지의 시간 간격을 0으로 초기화
    timeBetSpawn = 0f;
}
```

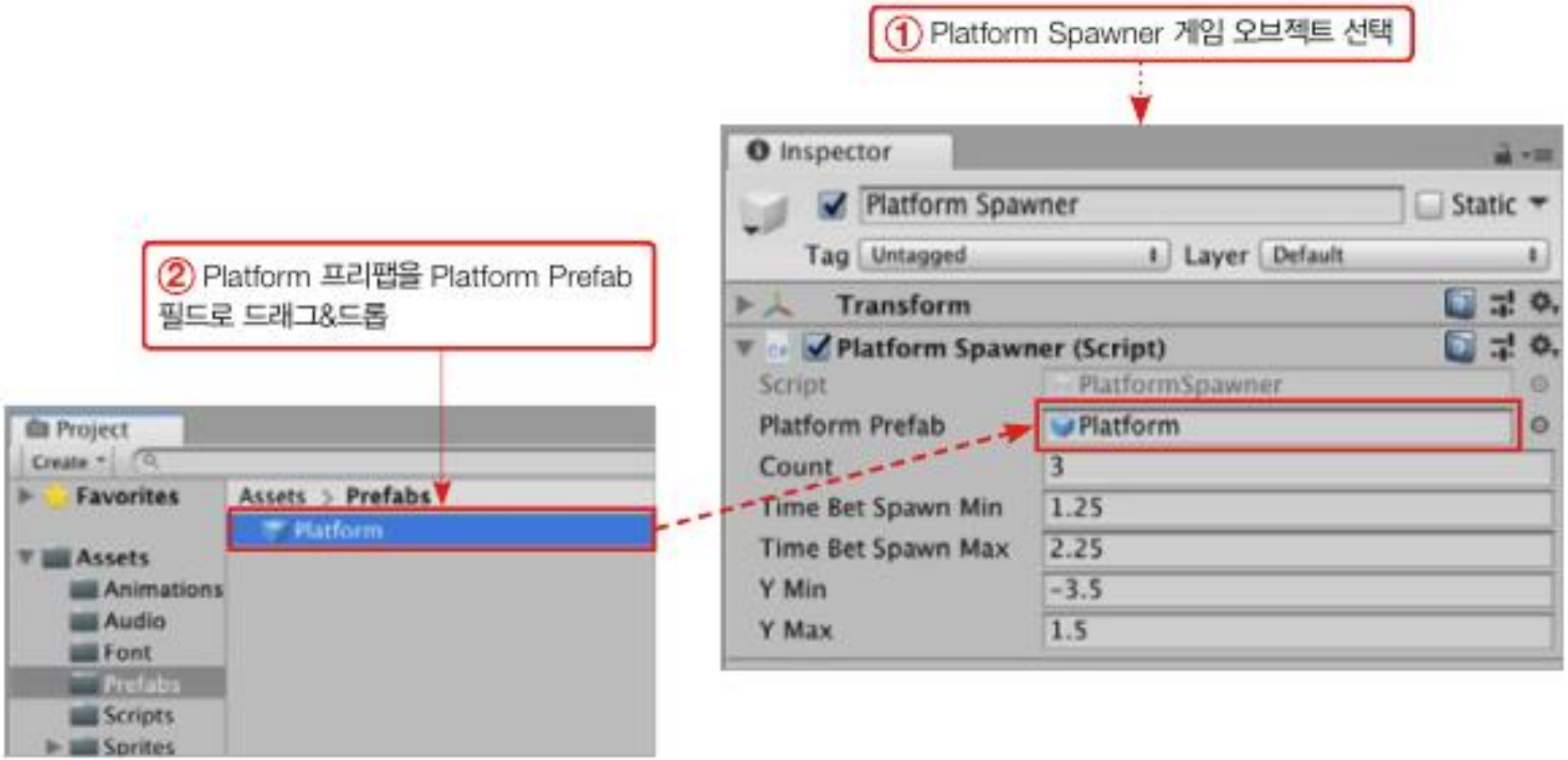
PlatformSpawner 스크립트(3)

```
void Update() {
    // 게임오버 상태에서는 동작하지 않음
    if (GameManager.instance.isGameover)
    {
        return;
    }

    // 마지막 배치 시점에서 timeBetSpawn 이상 시간이 흘렀다면
    if (Time.time >= lastSpawnTime + timeBetSpawn)
    {
        // 기록된 마지막 배치 시점을 현재 시점으로 갱신
        lastSpawnTime = Time.time;

        // 다음 배치까지의 시간 간격을 timeBetSpawnMin, timeBetSpawnMax 사이에서 랜덤 설정
        timeBetSpawn = Random.Range(timeBetSpawnMin, timeBetSpawnMax);

        // 배치할 위치의 높이를 yMin과 yMax 사이에서 랜덤 설정
        float yPos = Random.Range(yMin, yMax);
```

PlatformSpawner 스크립트(4)

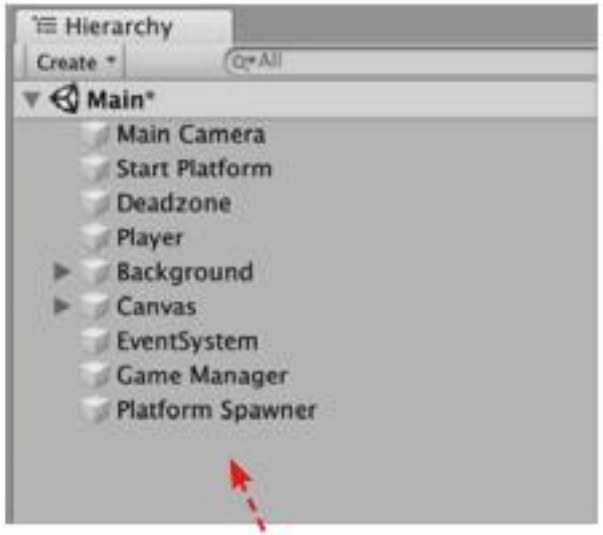
```
            // 사용할 현재 순번의 발판 게임 오브젝트를 비활성화하고 즉시 다시 활성화
            // 이때 발판의 Platform 컴포넌트의 OnEnable 메서드가 실행됨
            platforms[currentIndex].SetActive(false);
            platforms[currentIndex].SetActive(true);

            // 현재 순번의 발판을 화면 오른쪽에 재배치
            platforms[currentIndex].transform.position = new Vector2(xPos, yPos);
            // 순번 넘기기
            currentIndex++;

            // 마지막 순번에 도달했다면 순번을 리셋
            if (currentIndex >= count)
            {
                currentIndex = 0;
            }
        }
    }
}
```

PlatformSpawner 컴포넌트

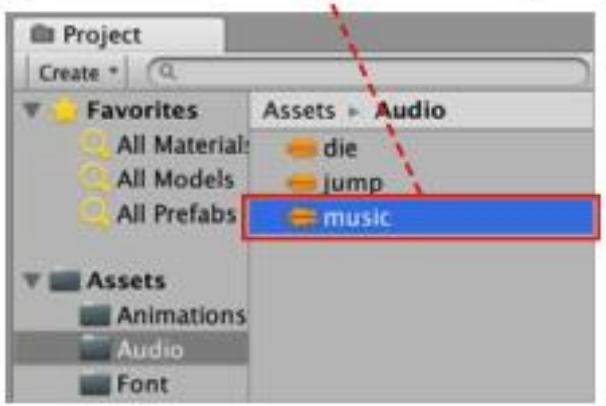


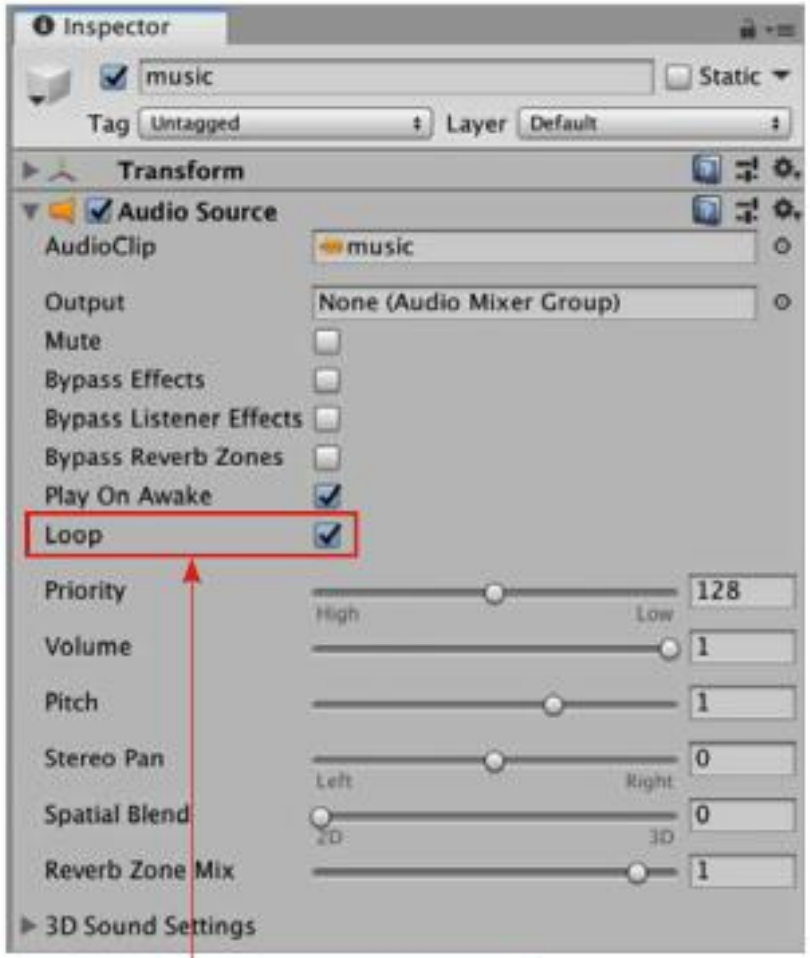
배경 음악 추가와 빌드하기

배경음악 추가하기



① music 오디오 클립을 하이어라키 창으로 드래그&드롭



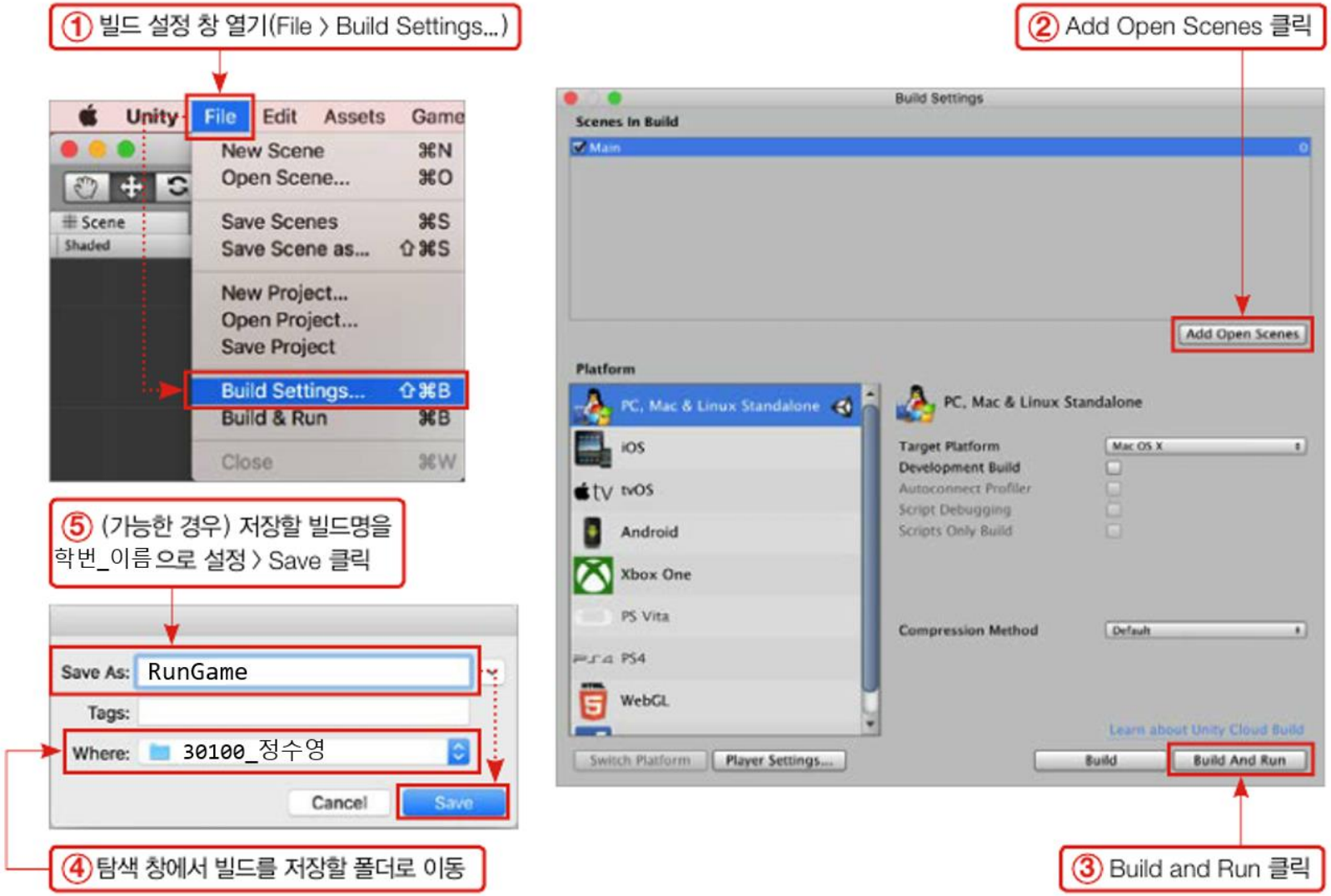


② music 게임 오브젝트의 Audio Source 컴포넌트의 Loop 체크

테스트하기

빌드하고 제출하기

빌드하기



빌드 폴더 제출

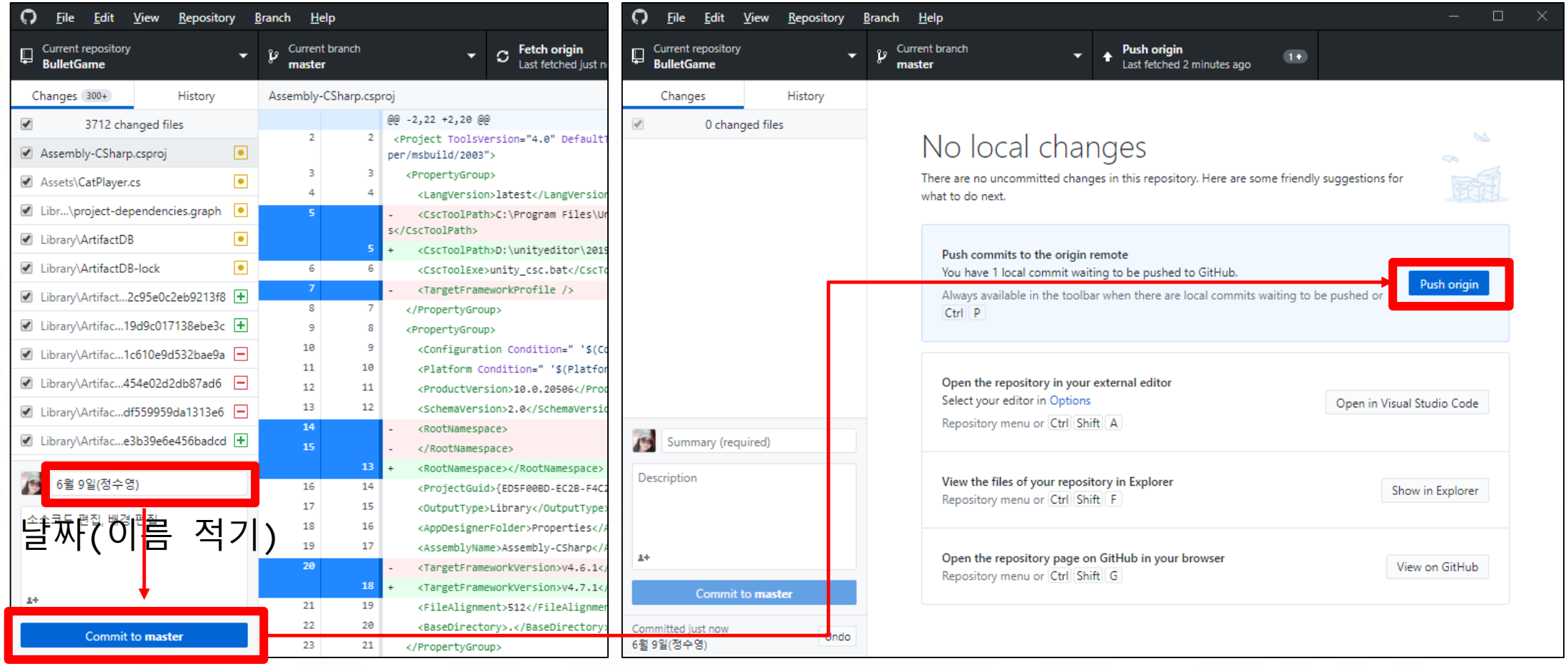
위와 같이 빌드 파일이 들어 있는 폴더를 압축한 후 과제 제출창에 제출합니다.

마지막에 깃허브에 푸쉬하는 것도 잊지마세요!

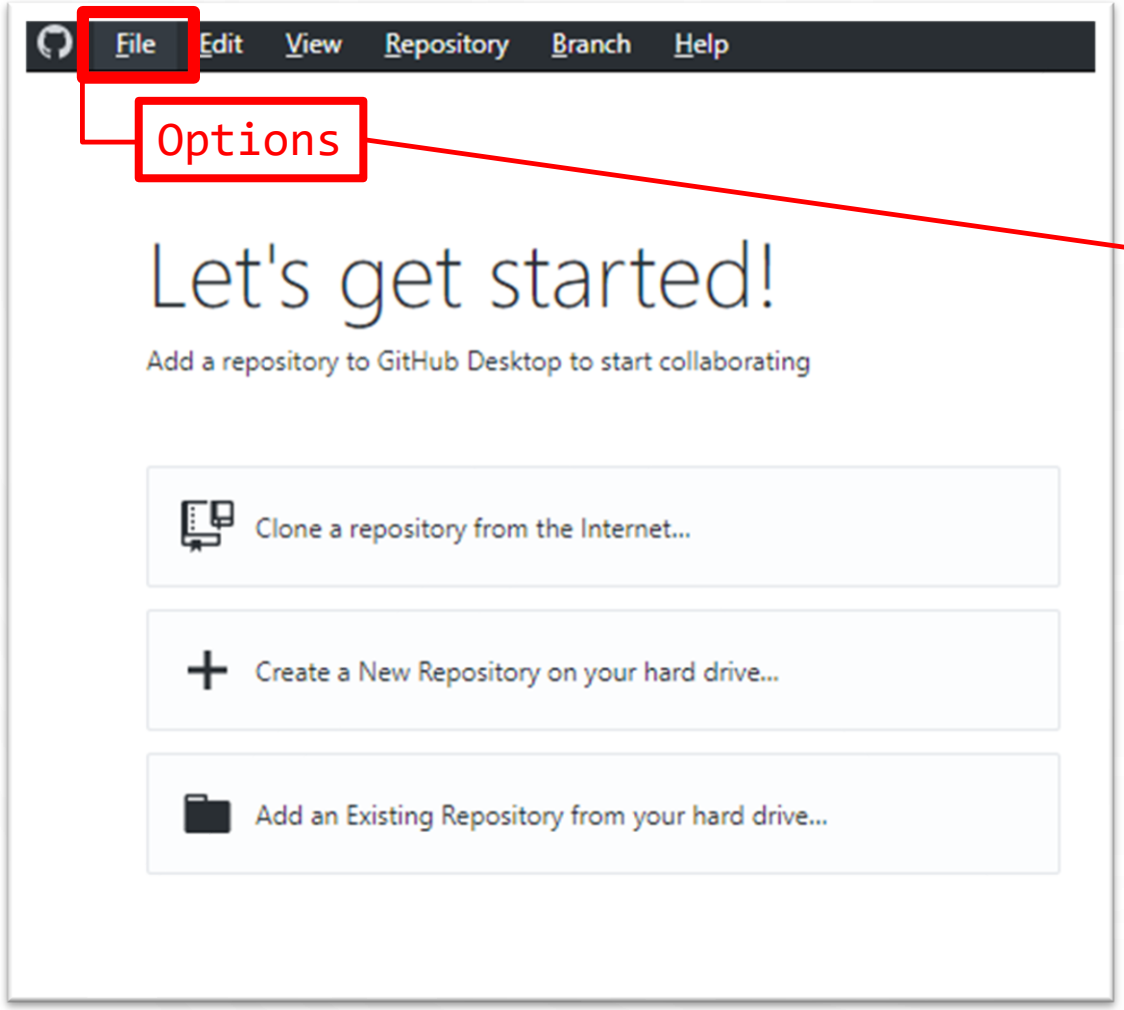
GITHUB PUSH

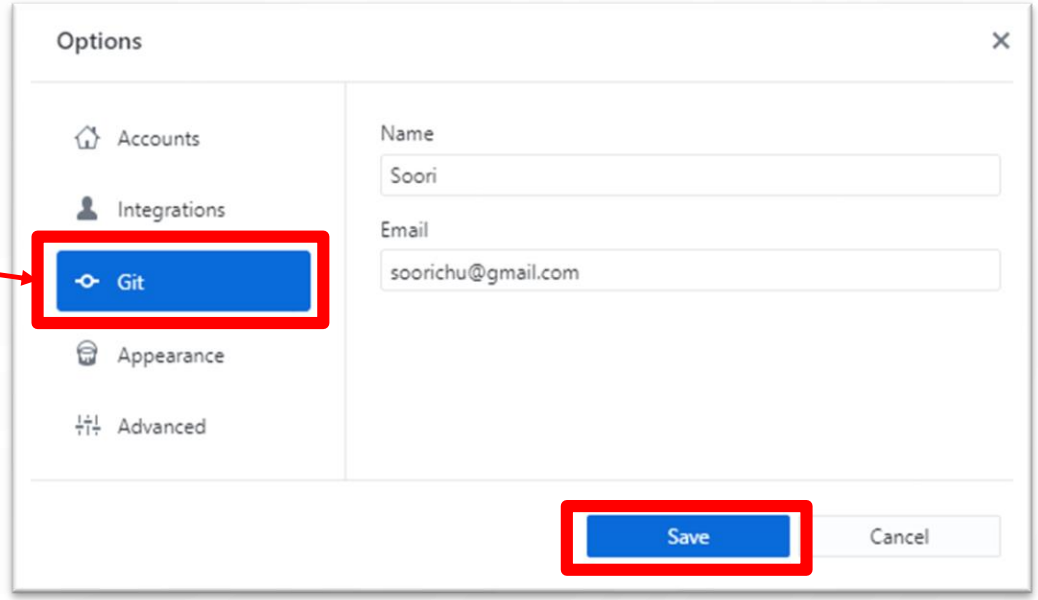
깃허브 올리기

- Unity 프로젝트 저장 후 모두 닫고 Git Desktop 열기



에러 날 때!





그냥 Name과 Email이 잘 채워진 것만 확인하고 Save 버튼 누르세요.
이제 다시 Commit > Push 시도해보세요. ^^

END

참고 : 레트로의 유니티 게임 프로그래밍 에센스