Unity를 이용한 2D 게임프로그래밍

Lecture 10
게임 매니저

강영민

동명대학교 게임공학과

학습목표

- 싱글톤 패턴
- 게임 매니저의 구현

싱글톤 패턴으로 구현하는 게임 매니저

- 싱글톤 – 유일한 객체를 만들기 위한 디자인 패턴
  - 게임관리자
    - 두 개의 관리자가 있을 이유가 없음
- 게임 매니저
  - 씬의 전환
  - 스코어의 관리
  - 옵션의 관리
- 통합적으로 게임을 관리하는 하나의 매니저만 필요
  - 단 하나만 존재하는 유일성과 모두가 사용할 수 있는 접근성이 특징
  - 싱글톤 패턴으로 구현하는 것이 바람직

싱글톤 유일성 구현 방법

- 유일성을 유지하기 위한 변수 설정

```
public class GameManager : MonoBehaviour {
        private static GameManager instance;
```

```
void Awake()
{
        if(instance != this)
        {
                Destroy(gameObject);          유일성 유지
        }
}
```

싱글톤 접근성을 지원하는 게터

```
public class GameManager : MonoBehaviour
{
	private static GameManager instance;
	//싱글톤 접근용 프로퍼티
	public static GameManager Instance
	{
		get{
			if(instance == null)
			{
				return null;
			}
			return instance;
		}
	}
```

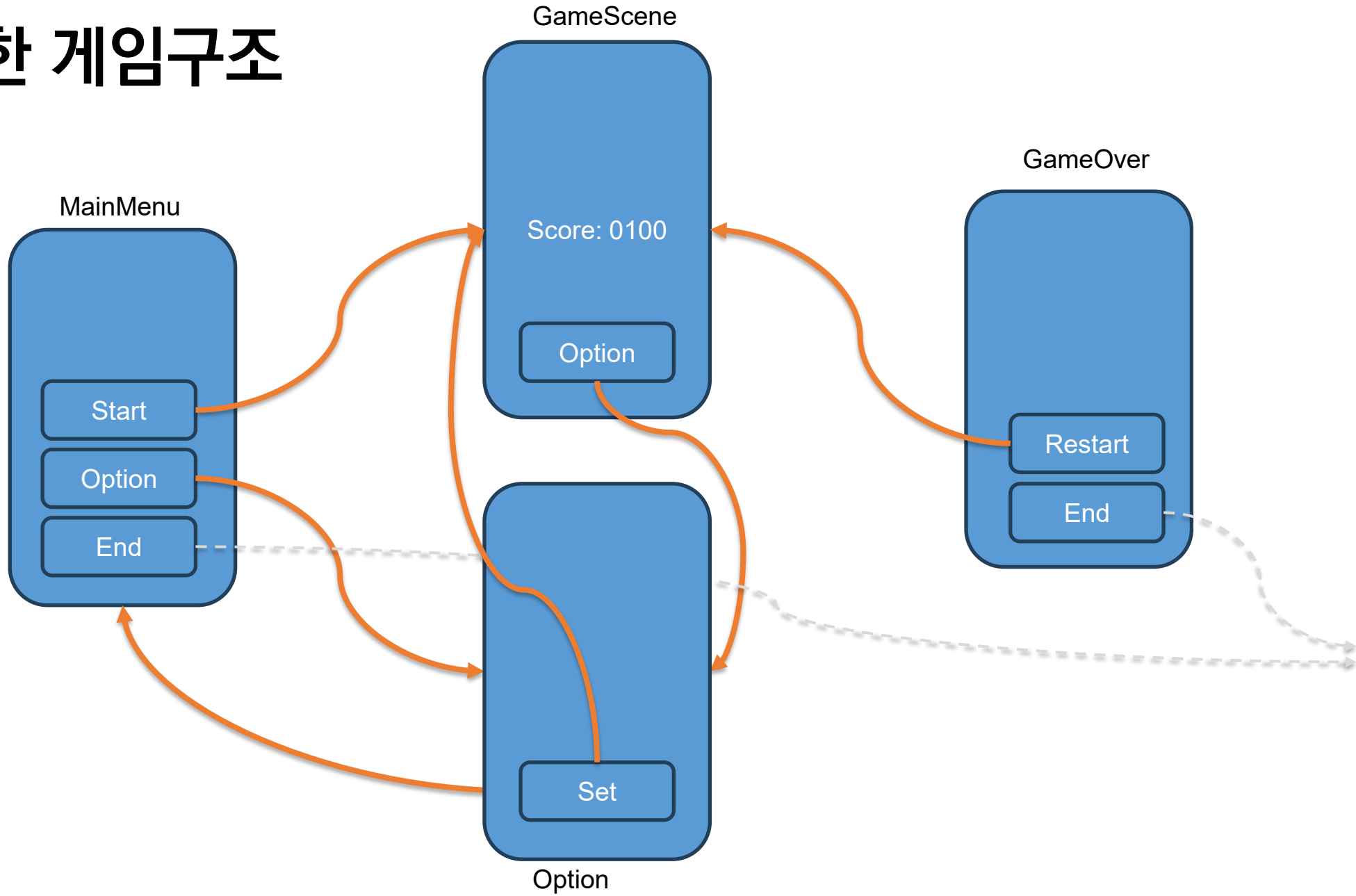
싱글톤 접근성 구현 방법

- 외부 호출 변수를 public으로 구현

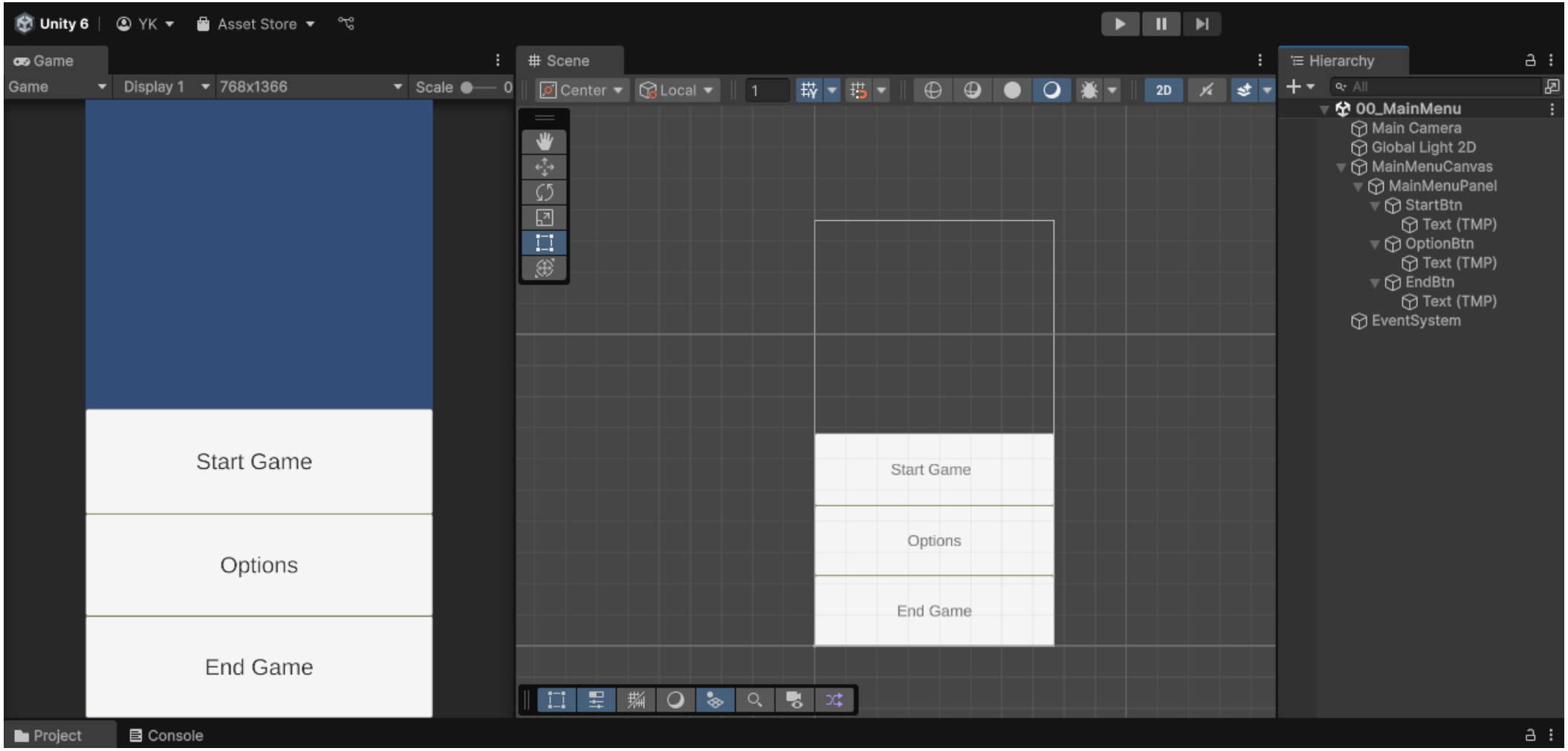
```
public void DoSomething(… args …)
{
    …. Do something
}
```

```
public class PlayerCtrl : MonoBehaviour
{
        public void  NeedSomething ()
        {
                GameManager.Instance.DoSomething(…);          어디서든 접근
        }
}
```

간단한 게임구조


GameScene
Score: 0100
Option
GameOver
MainMenu
Start
Option
End
Restart
End
Set
Option

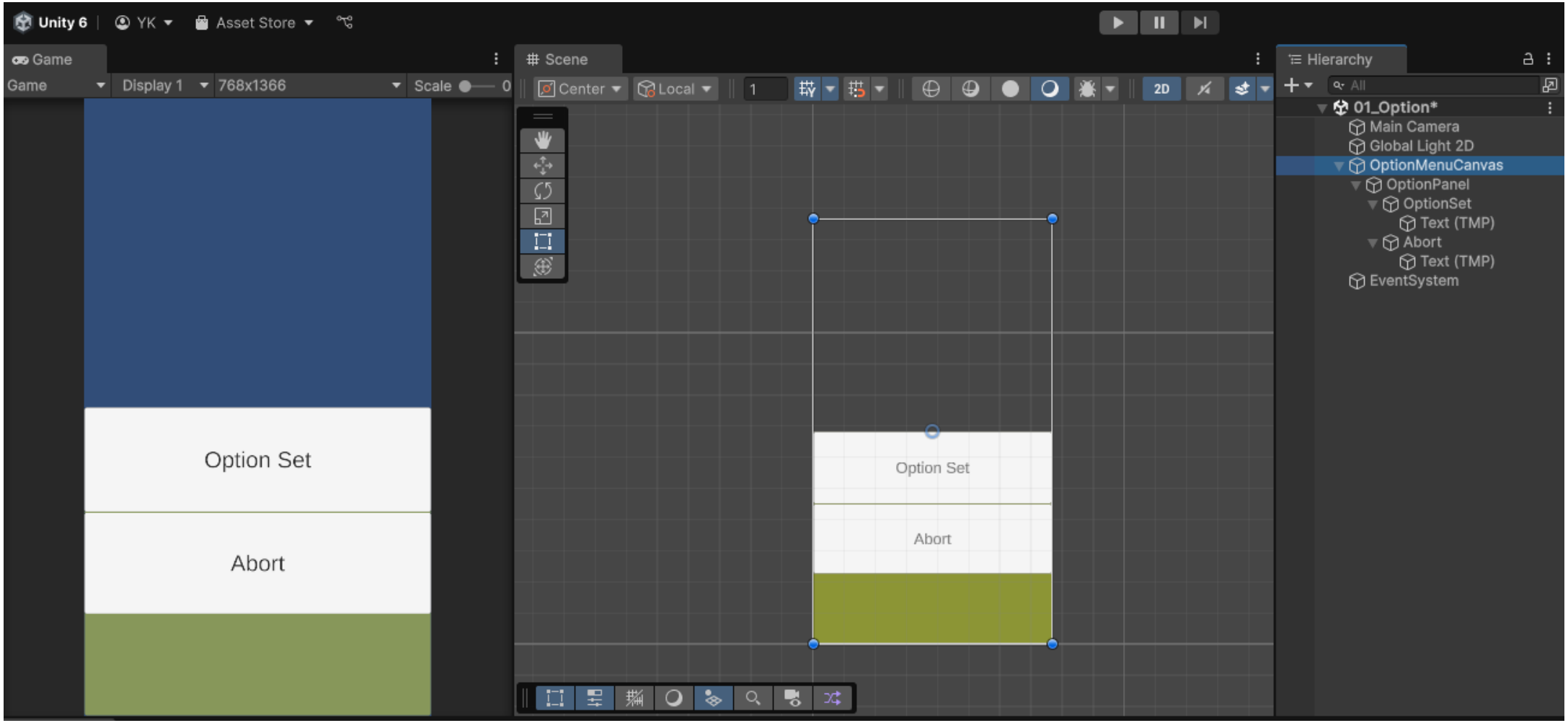
Main Menu를 만들어 보자



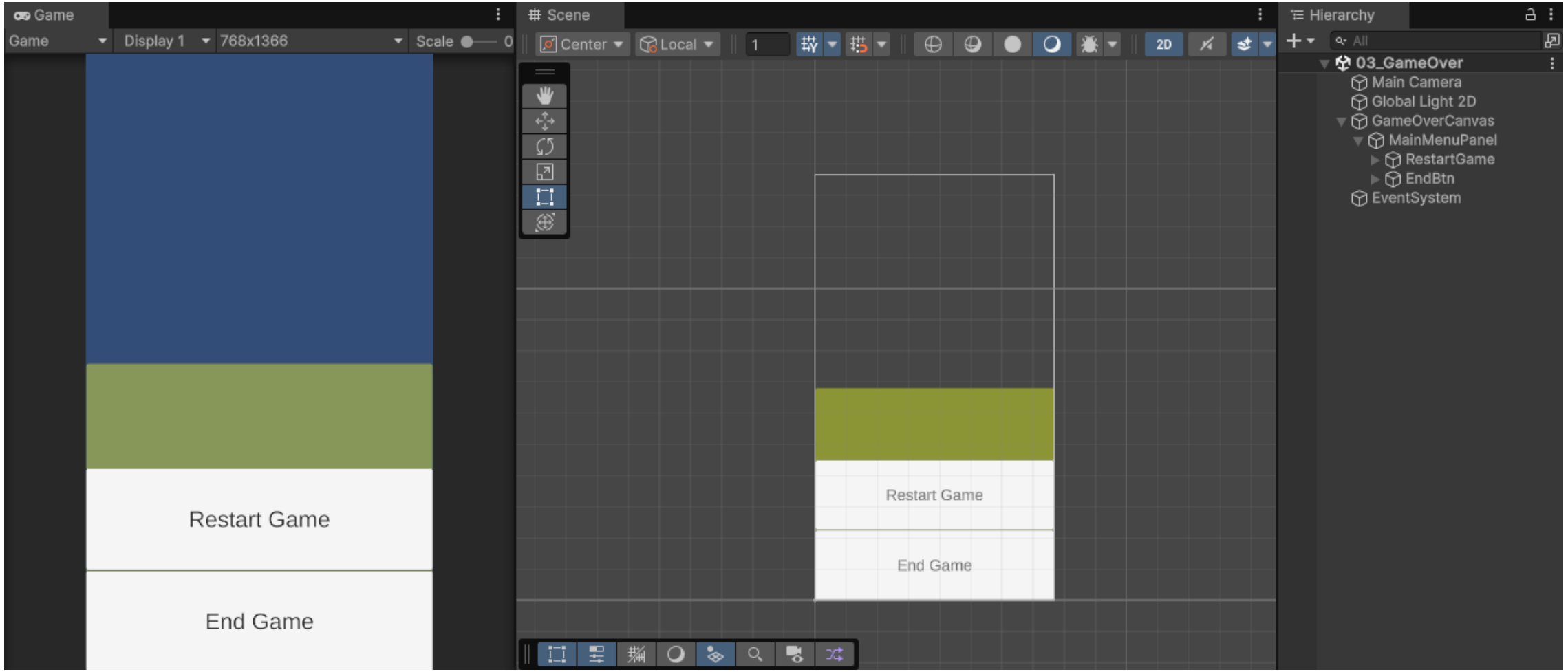
Scene의 이름을 변경하자

Scene을 복사해서 여러 씬을 만들자 (저장하고 복사!)

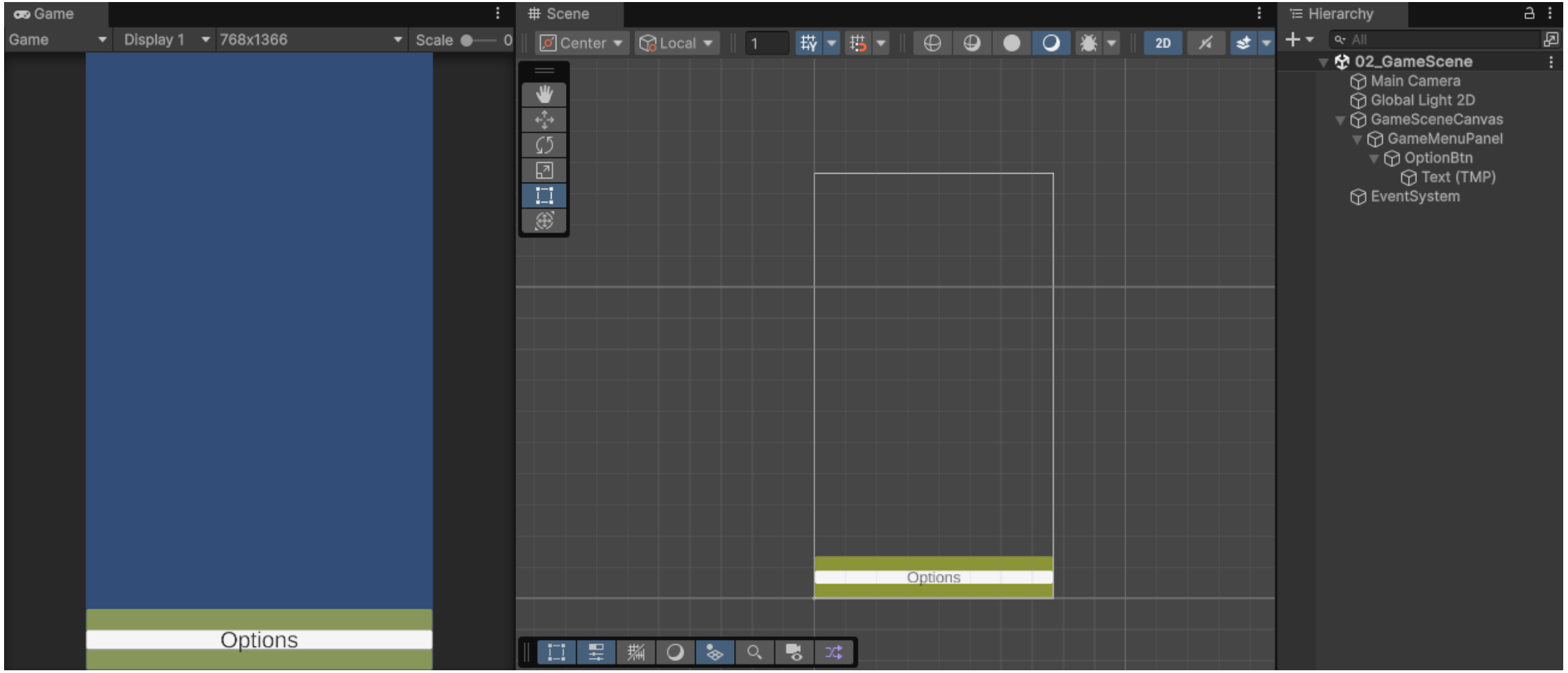
Option 장면 구성


Unity 6
YK
Asset Store
Game
Display 1
768x1366
Scale
Option Set
Abort
Scene
Center
Local
2D
Option Set
Abort
Hierarchy
01_Option*
Main Camera
Global Light 2D
OptionMenuCanvas
OptionPanel
OptionSet
Text (TMP)
Abort
Text (TMP)
EventSystem

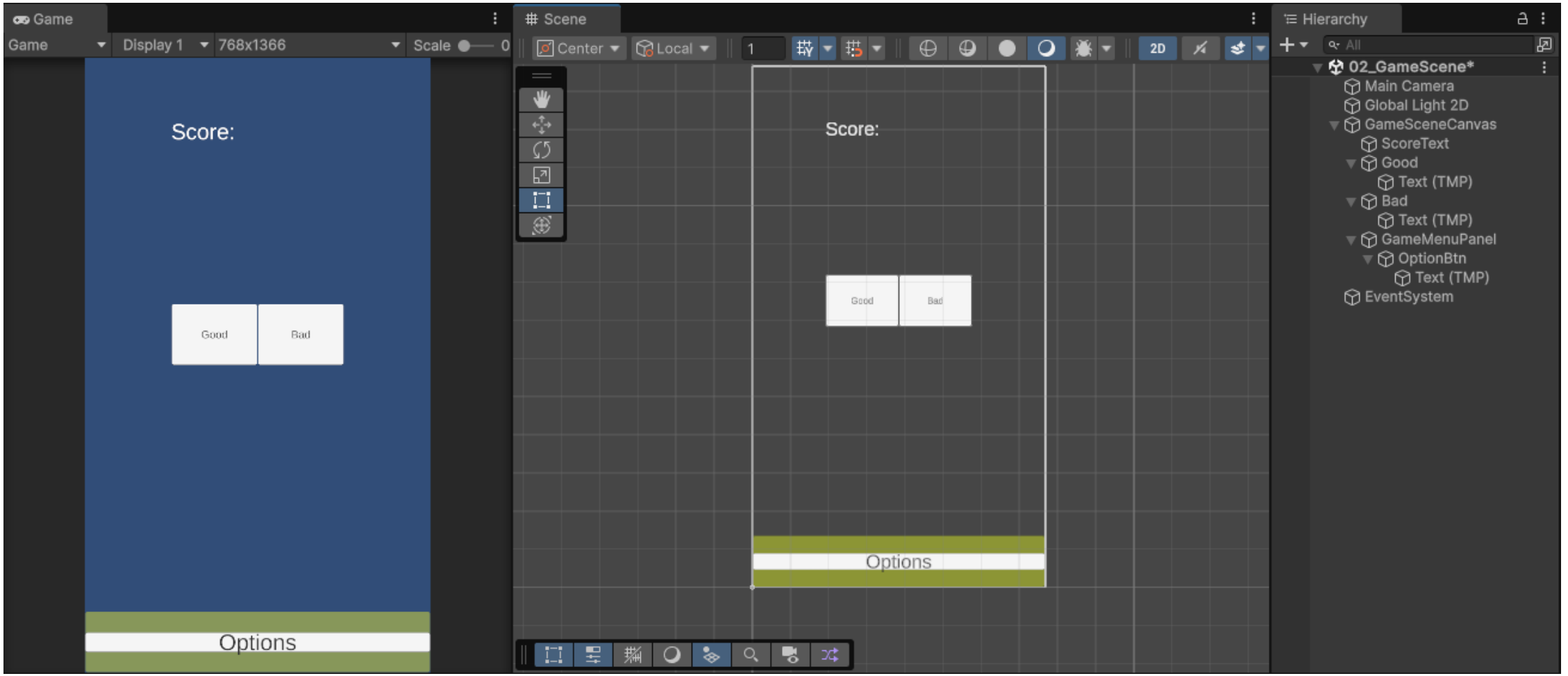
GameOver 장면 구성



GameScene 구성

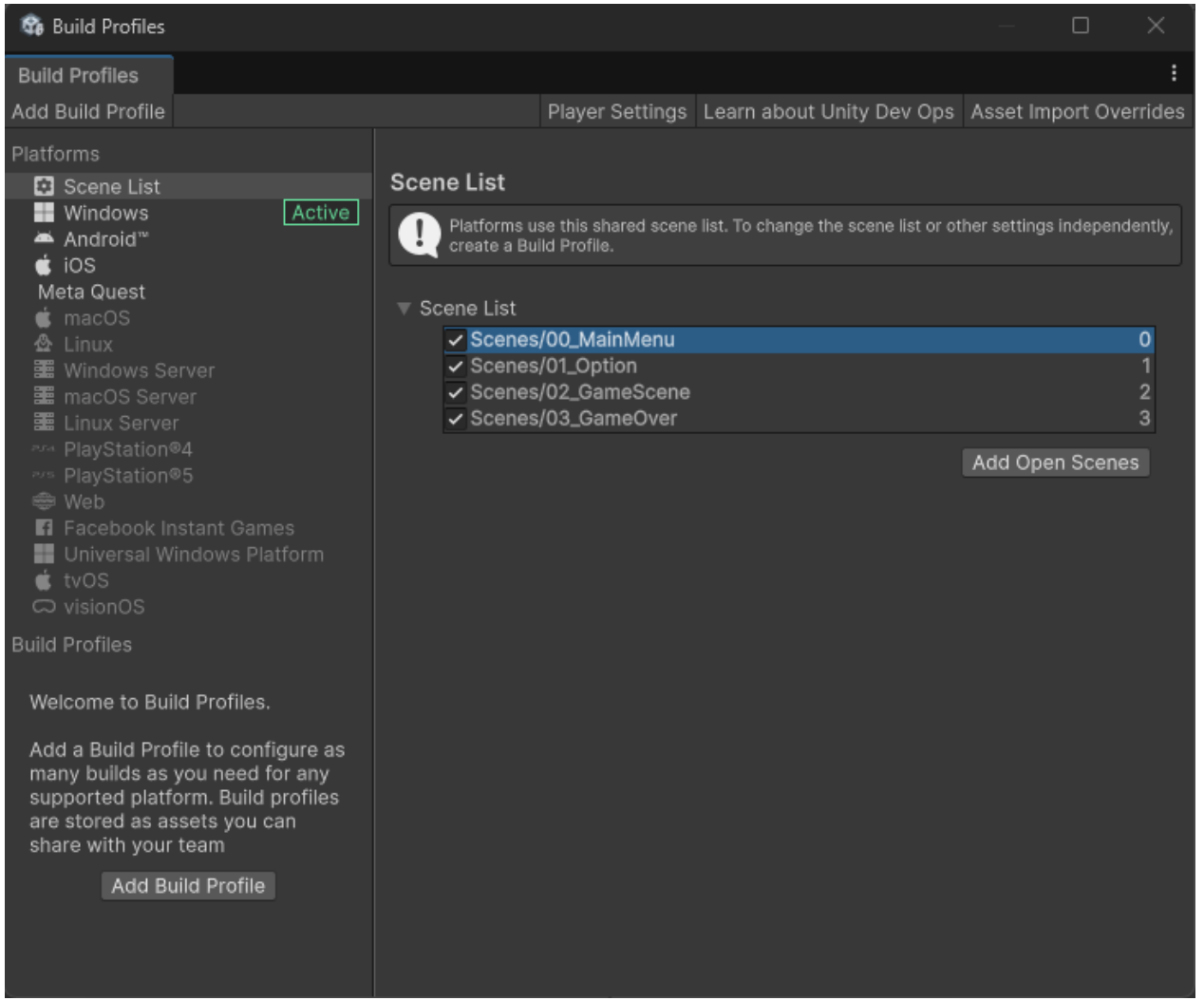


GameScene 추가

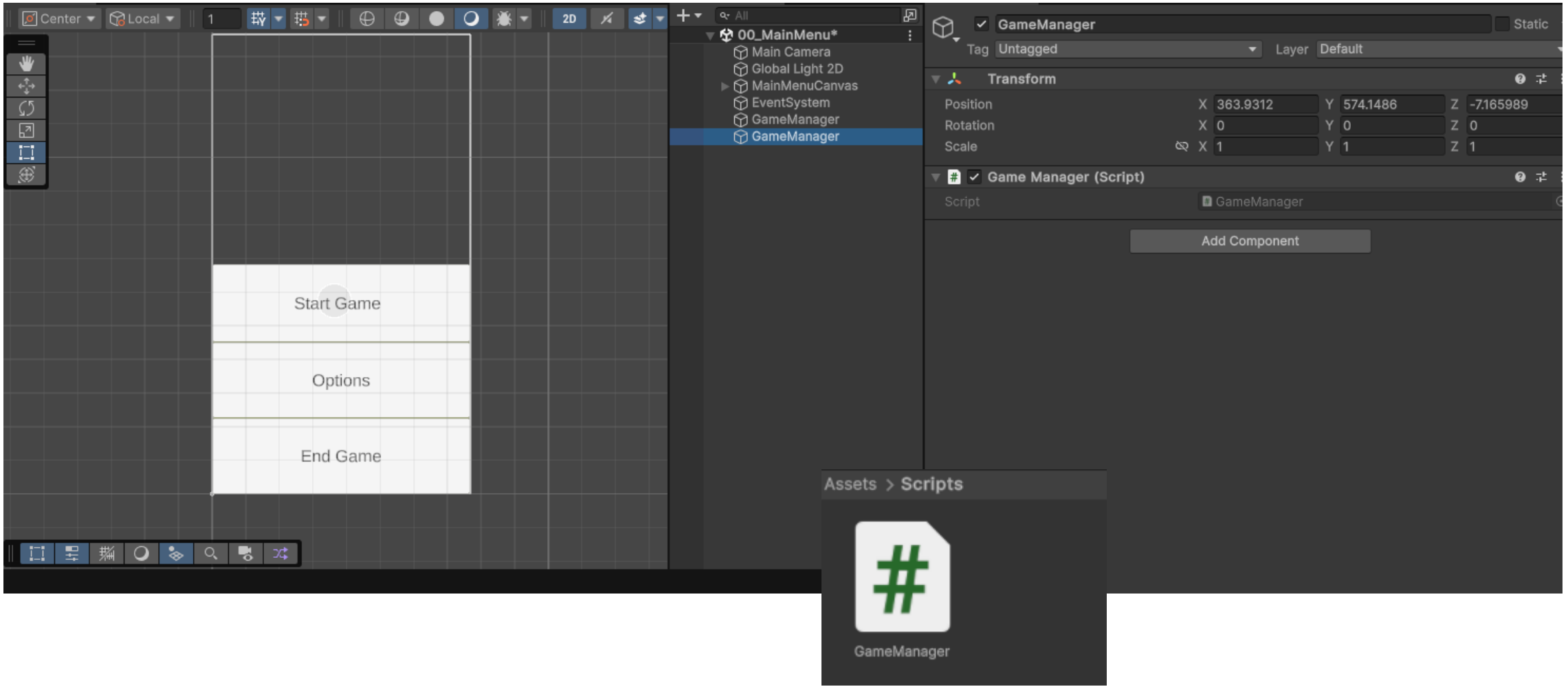


File → Build Profiles

- Scene List 추가



Game Manager 생성



GameManager.cs

```
public class GameManager : MonoBehaviour
{
    private static GameManager instance;
    public static GameManager Instance
    {
        get
        {
            if (instance == null)
            {
                return null;
            }
            return instance;
        }
    }
```

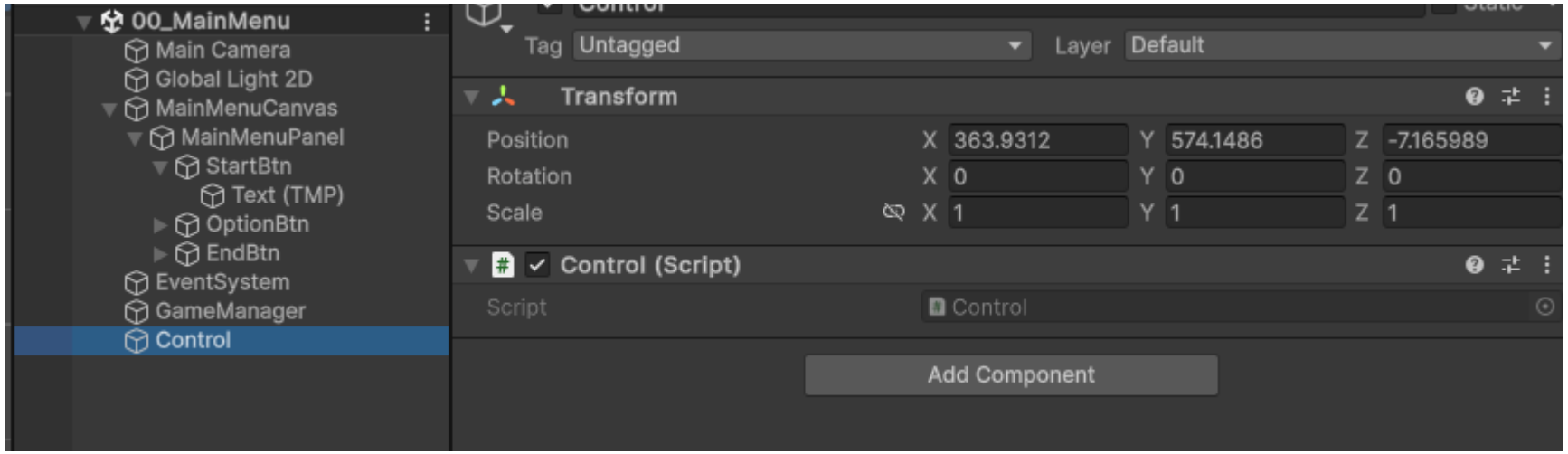
```
    private void Awake()
    {
        if (instance == null)
        {
            instance = this;
            DontDestroyOnLoad(gameObject);
        }
        else
        {
            Destroy(gameObject);
        }
    }
```

Game Manager에 장면 전환 기능 추가

```
using UnityEngine.SceneManagement;
```

```
public void LoadScene(string sceneName)
{
    SceneManager.LoadScene(sceneName);
}
```

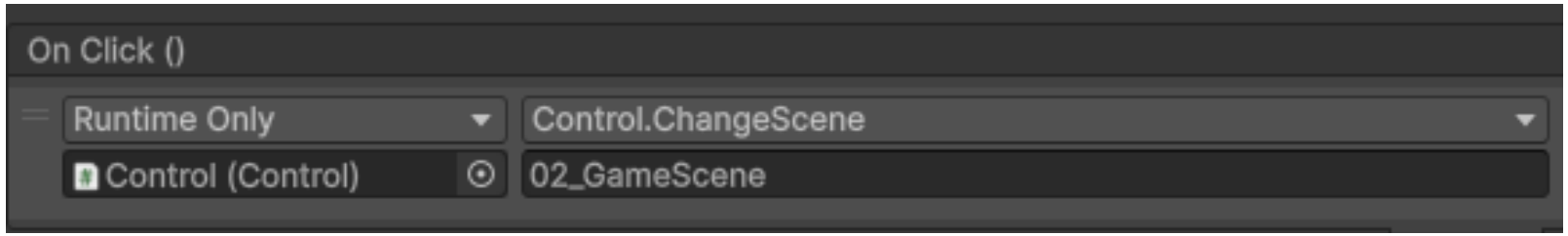
MainMenu에 Control 추가



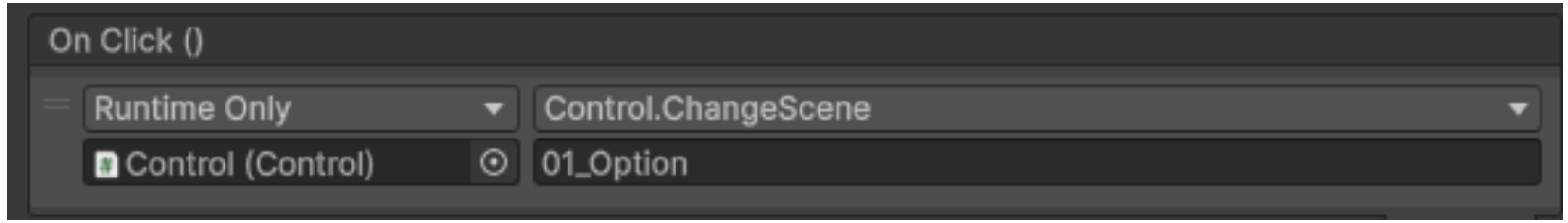
Control.cs 연결

```
public class Control : MonoBehaviour
{
    // Start is called once before the first execution of Update after the
    0 references
    public void ChangeScene(String sceneName)
    {
        if (GameManager.Instance != null)
        {
            GameManager.Instance.LoadScene(sceneName);
        }
        else
        {
            Debug.LogError("GameManager instance is not available.");
        }
    }

    0 references
    void Start()
```

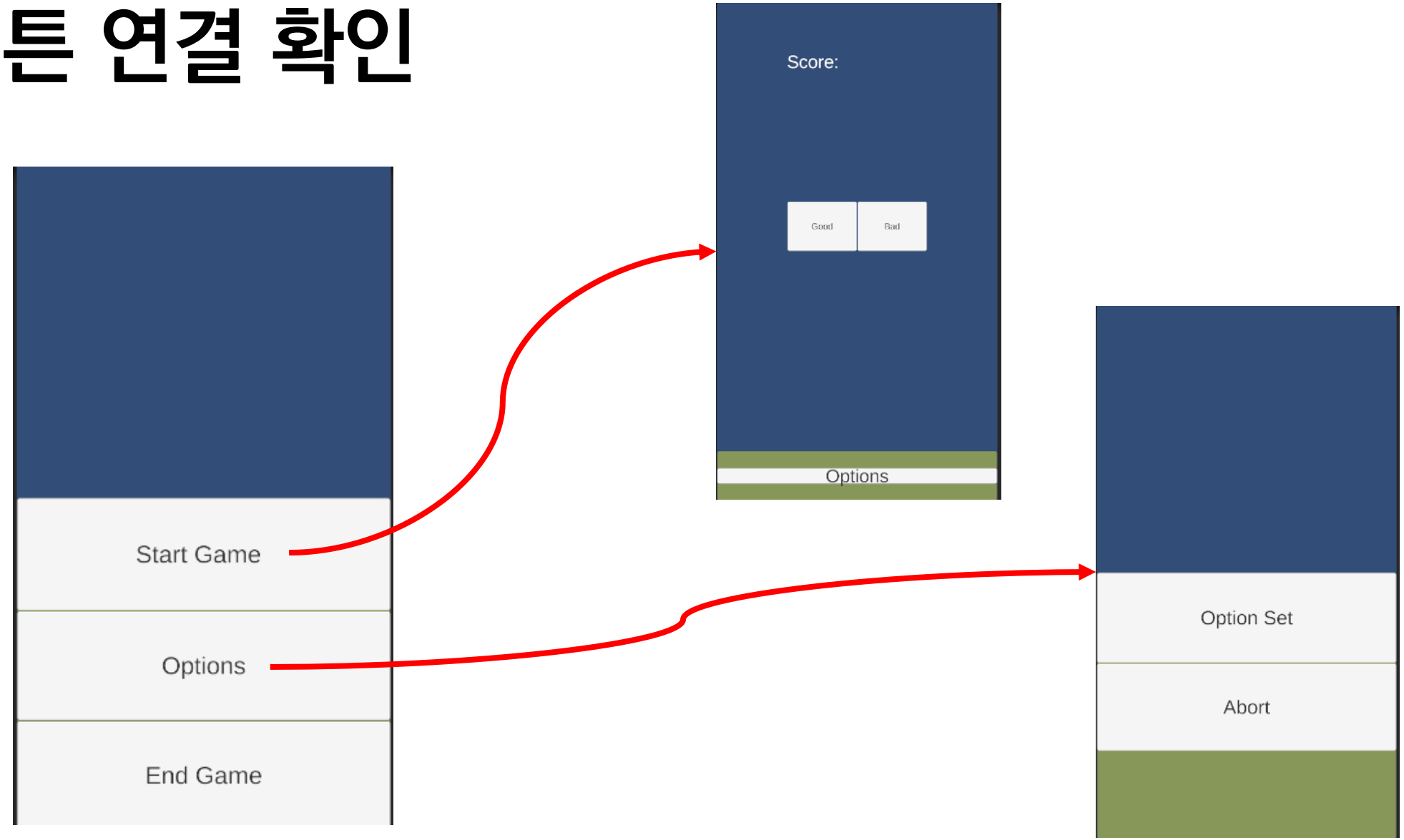
Start Game 버튼과 장면 전환 함수 연결



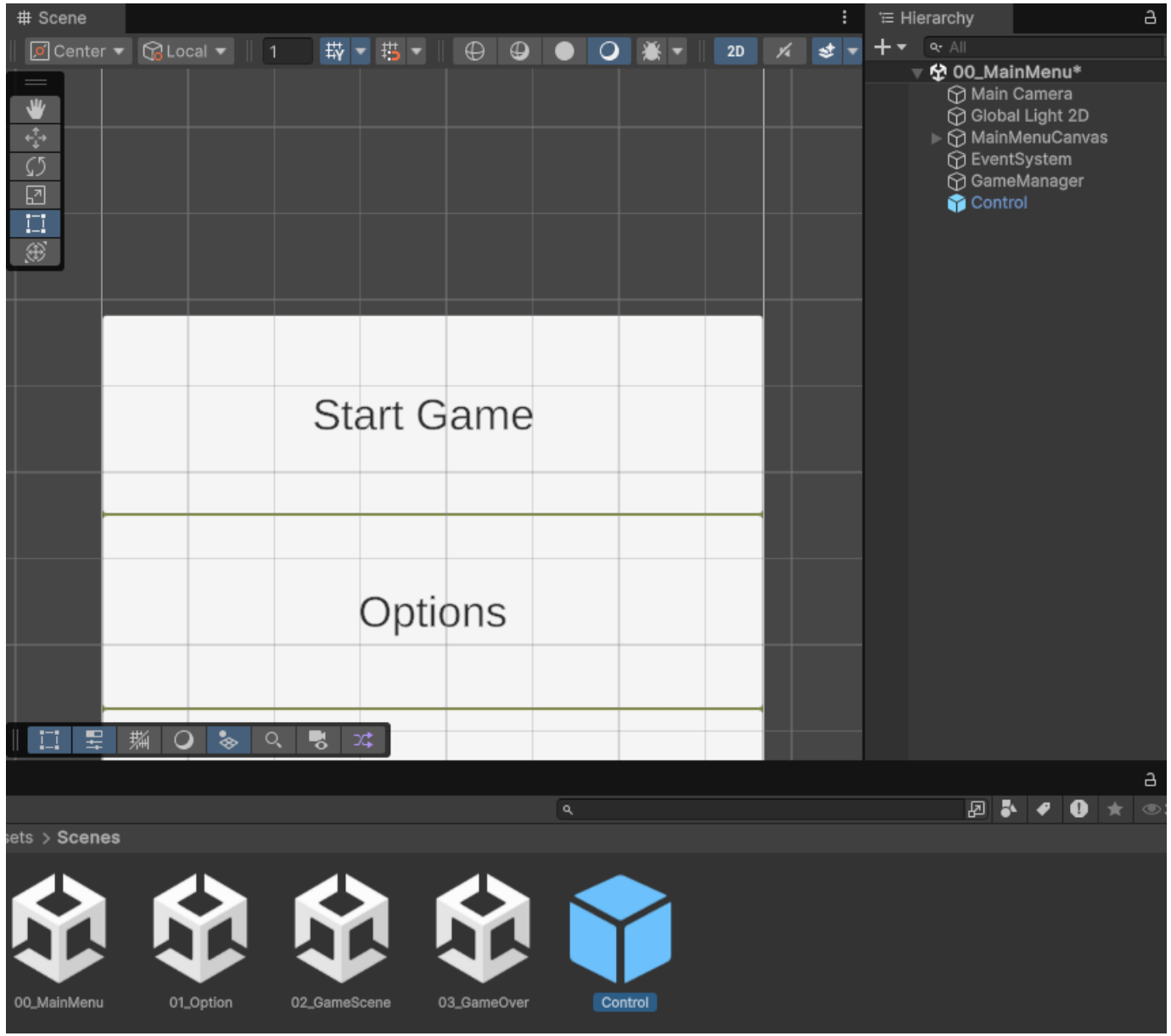
Options 버튼도 연결 추가



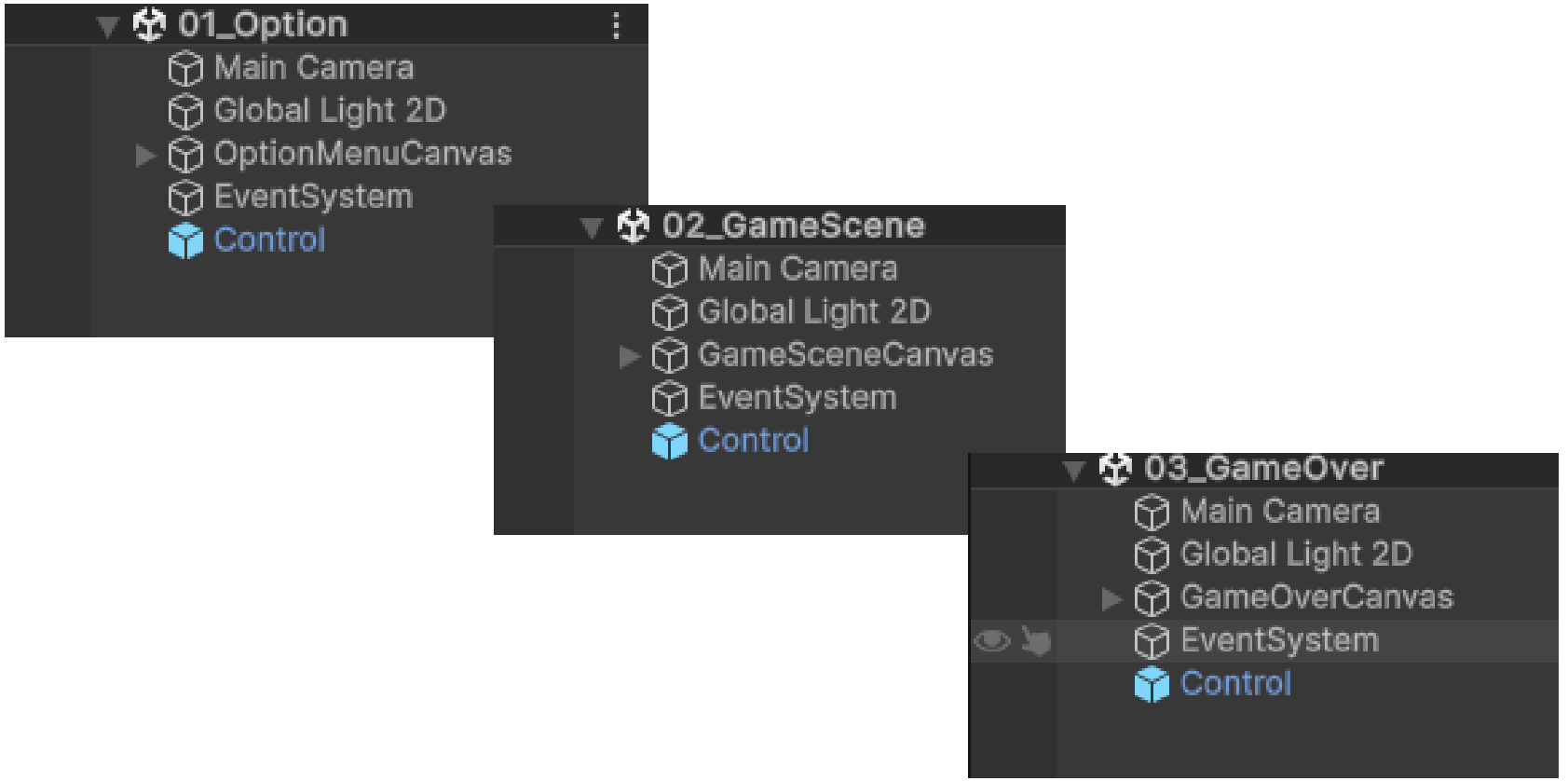
버튼 연결 확인


Start Game
Options
End Game
Score:
Good
Bad
Options
Option Set
Abort

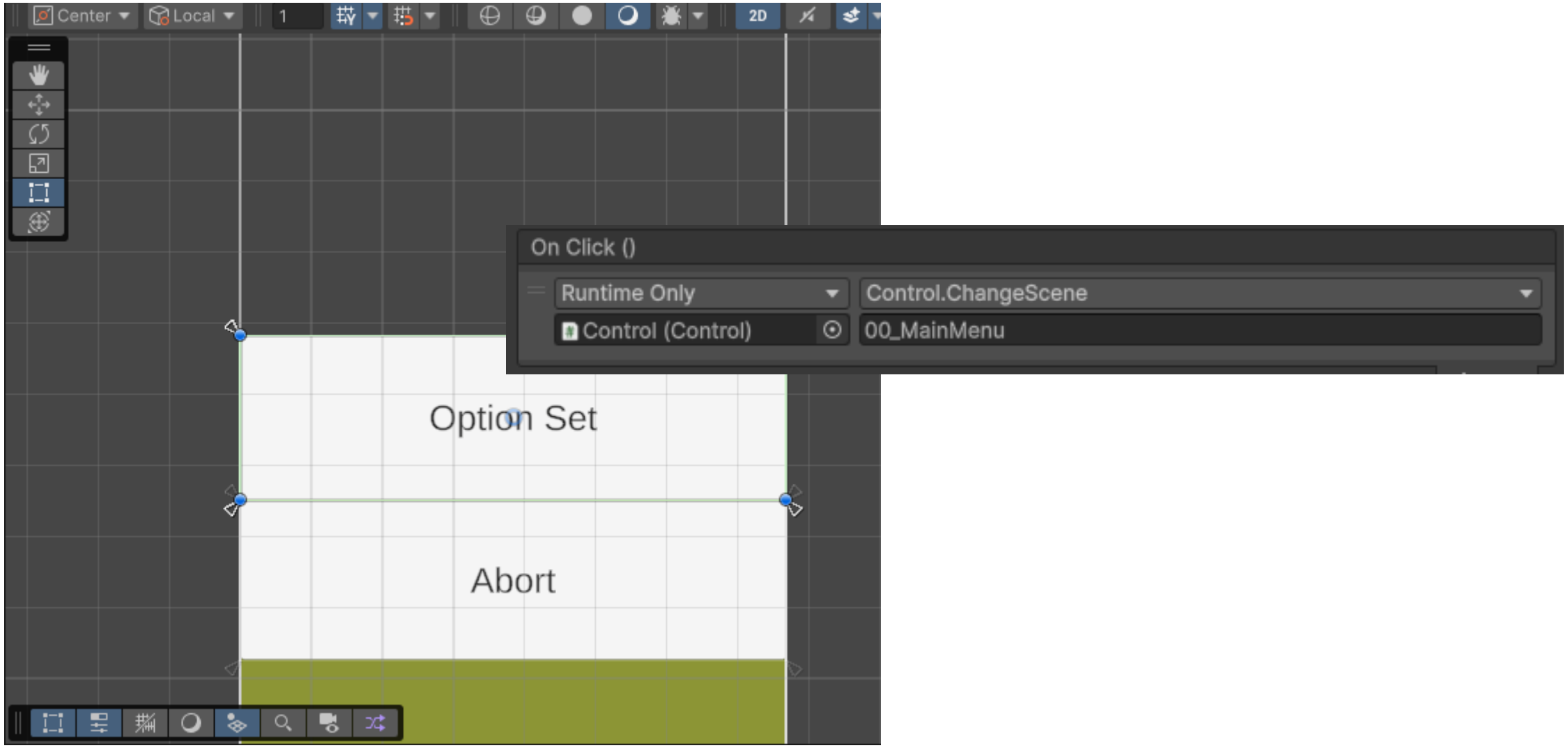
Control을 Prefab으로



다른 장면에도 이 Control 객체 달기



01_Option에서 01_MainMenu로 돌아가기



게임 매니저가 생성되는 곳

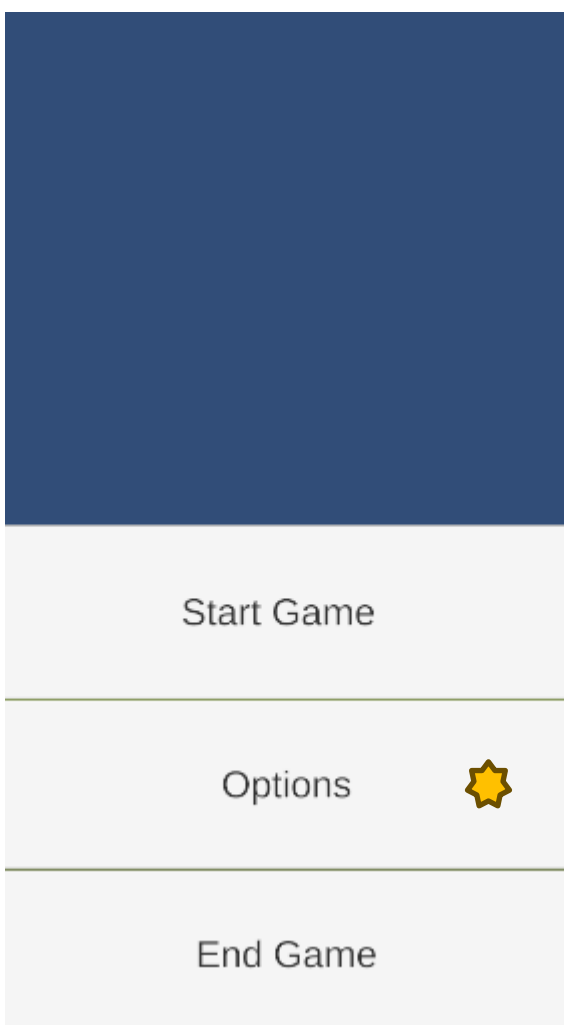
00_MainMenu Scene

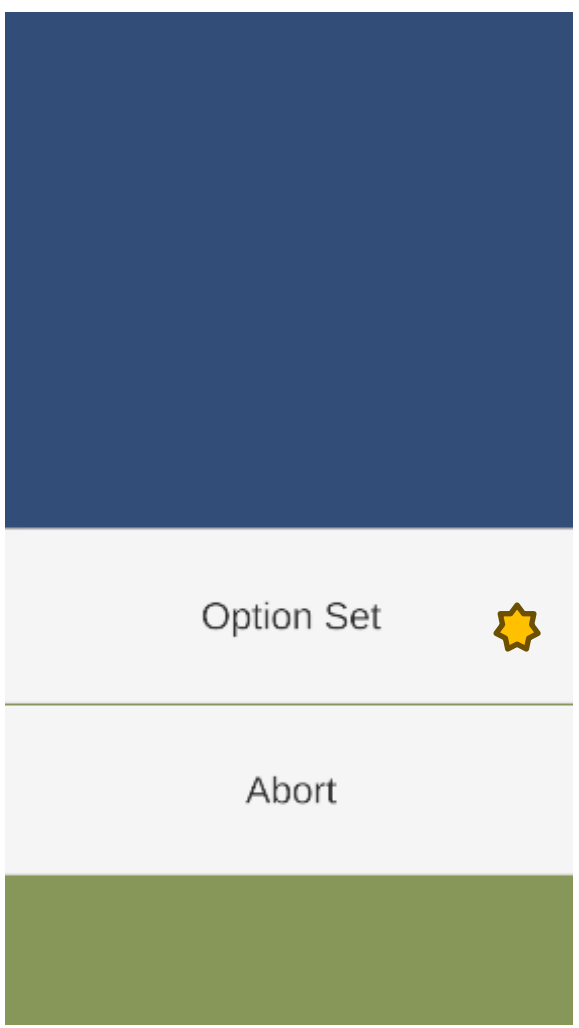
- 가장 먼저 이 장면이 실행되어야 함

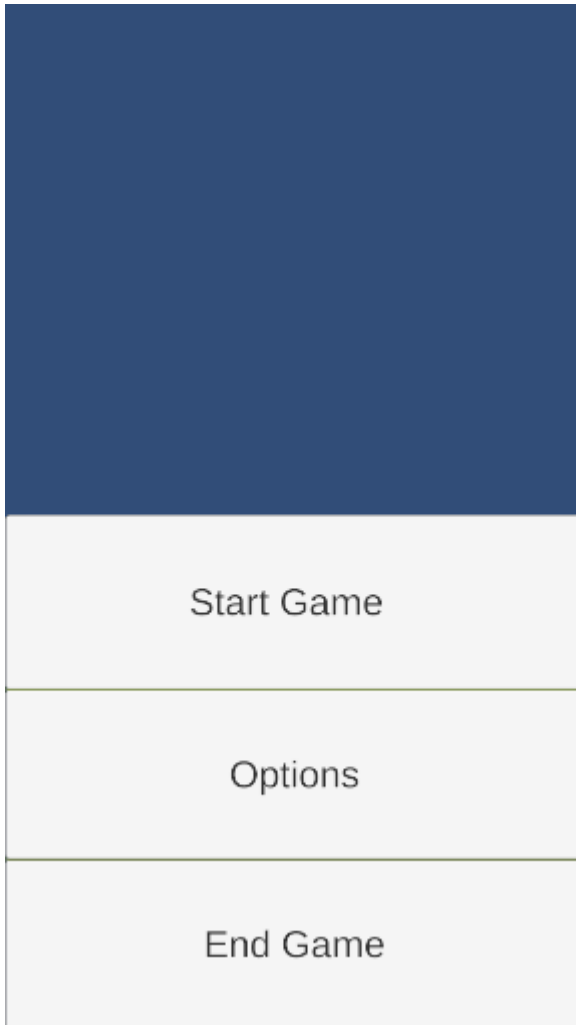
- 다른 장면이 로드되면 객체들이 Destroy됨
- 이를 막기 위해 GameManager.cs에서 다음과 같이 지정하였음

```
private void Awake()
{
    if (instance == null)
    {
        instance = this;
        DontDestroyOnLoad(gameObject);
    }
    else
    {
        Destroy(gameObject);
    }
}
```

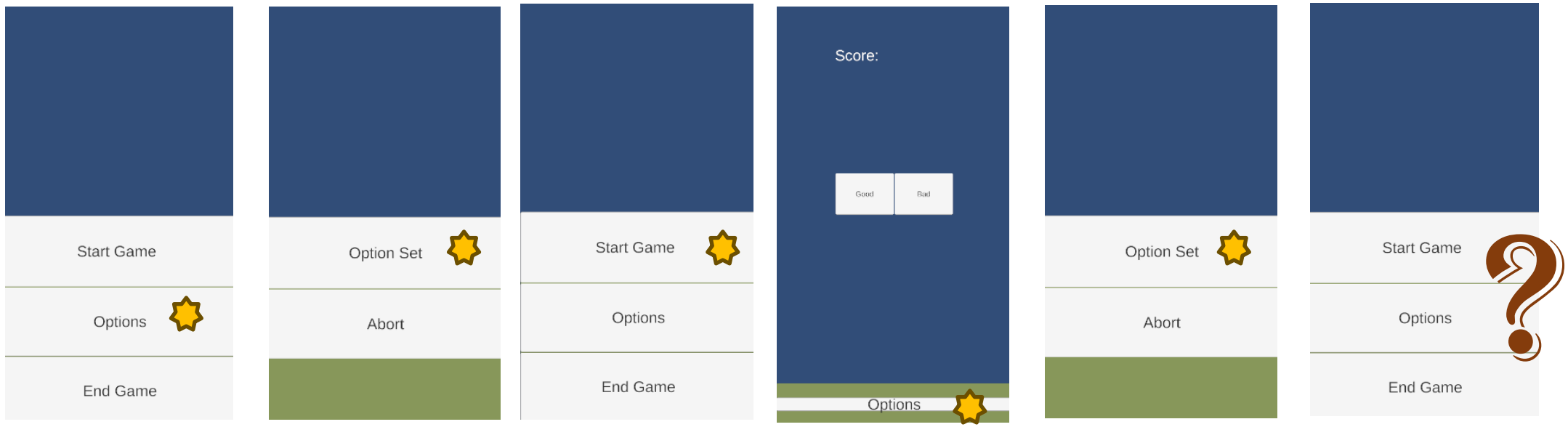
이제 옵션 설정을 마치면 Main Menu로 돌아감







게임에서도 옵션으로 들어갈 수 있음



Option 장면을 부를 때 추가 정보 저장

```
using System;
using UnityEngine;
using UnityEngine.SceneManagement;

9 references
public class GameManager : MonoBehaviour
{
    4 references
    private static GameManager instance;
    3 references
    public String caller;
```

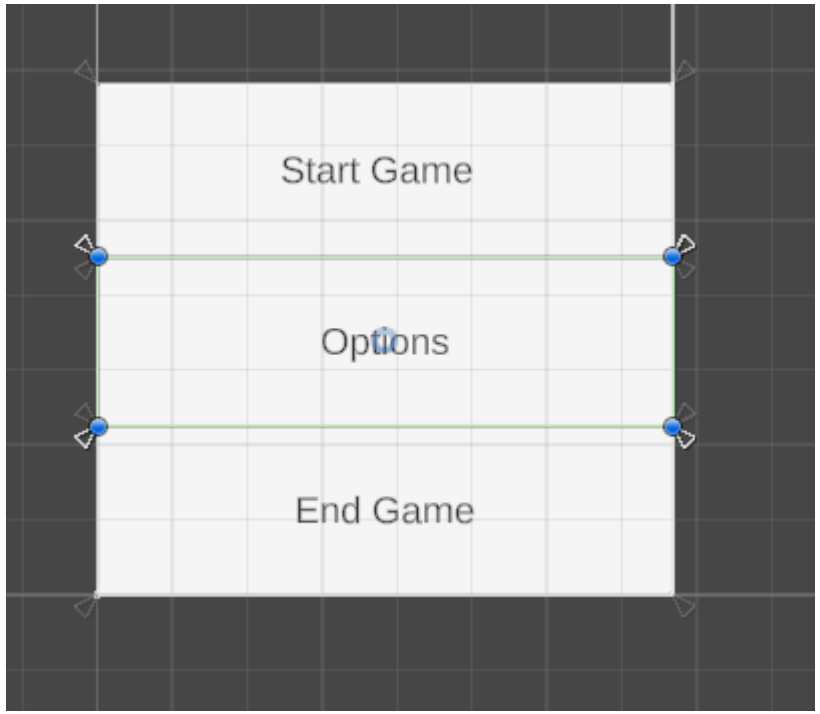
Game Manager에 추가 정보 저장 변수 추가

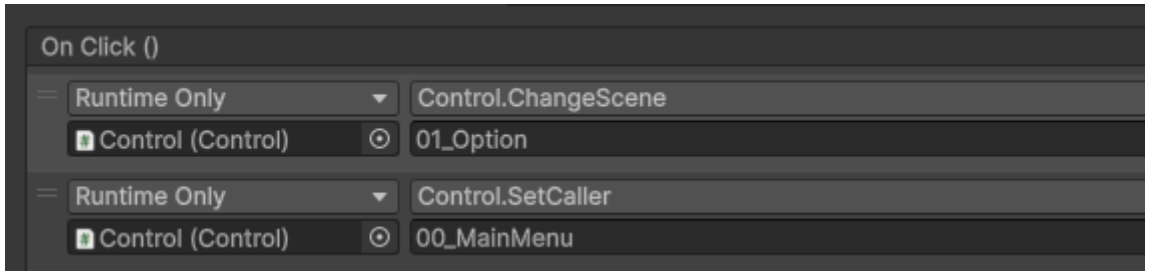
Option 장면을 부를 때 추가 정보 저장

```
public class Control : MonoBehaviour
{
    // Start is called once before the first execution of Update after the MonoBehaviour is
    0 references
    public void ChangeScene(String sceneName)
    {
        if (GameManager.Instance != null)
        {
            GameManager.Instance.LoadScene(sceneName);
            GameManager.Instance.caller = sceneName;
        }
        else
        {
            Debug.LogError("GameManager instance is not available.");
        }
    }
    0 references
    public void SetCaller(String sceneName)
    {
        if (GameManager.Instance != null)
        {
            GameManager.Instance.caller = sceneName;
        }
        else
        {
            Debug.LogError("GameManager instance is not available.");
        }
    }
```

Control.cs에서 추가정보 저장

장면 전환과 함께 부른 장면 기록





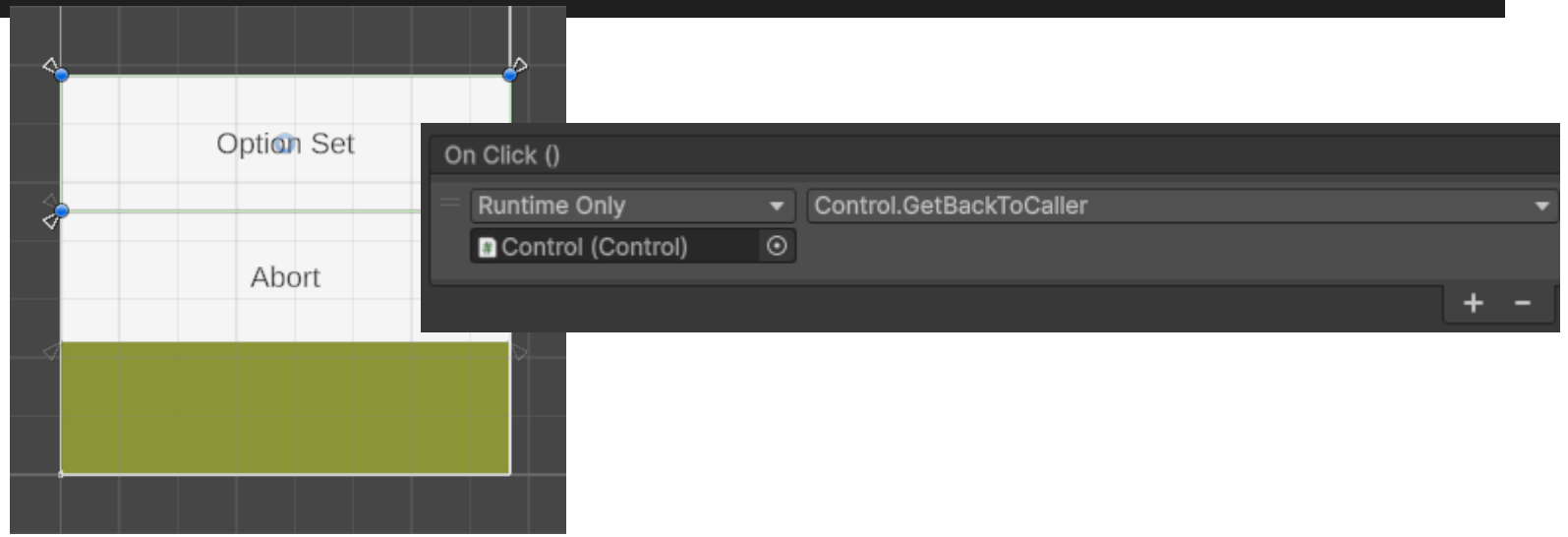
Game Manager에 추가 정보 저장 변수 추가

게임 씬에서 부를 때도 저장

Game Manager에 추가 정보 저장 변수 추가

Option 장면에서
→ 돌아갈 때 부른 장면으로 가기

```
public void GetBackToCaller()
{
    if (GameManager.Instance != null && !string.IsNullOrEmpty(GameManager.Instance.caller))
    {
        GameManager.Instance.LoadScene(GameManager.Instance.caller);
    }
    else
    {
        Debug.LogError("GameManager instance is not available or caller is not set.");
    }
}
```

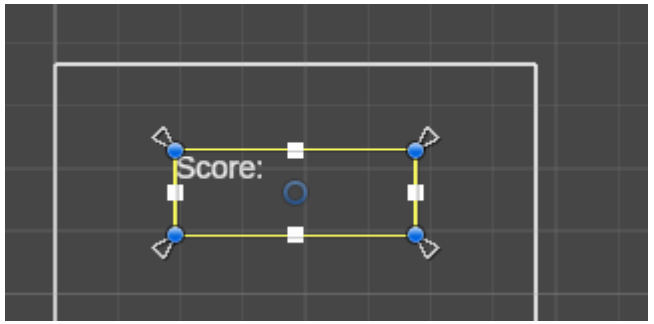


Score 관리는? 게임 매니저에서

```
public class GameManager : MonoBehaviour
{
    4 references
    private static GameManager instance;
    4 references
    public String caller;
    1 reference
    public int score;
```

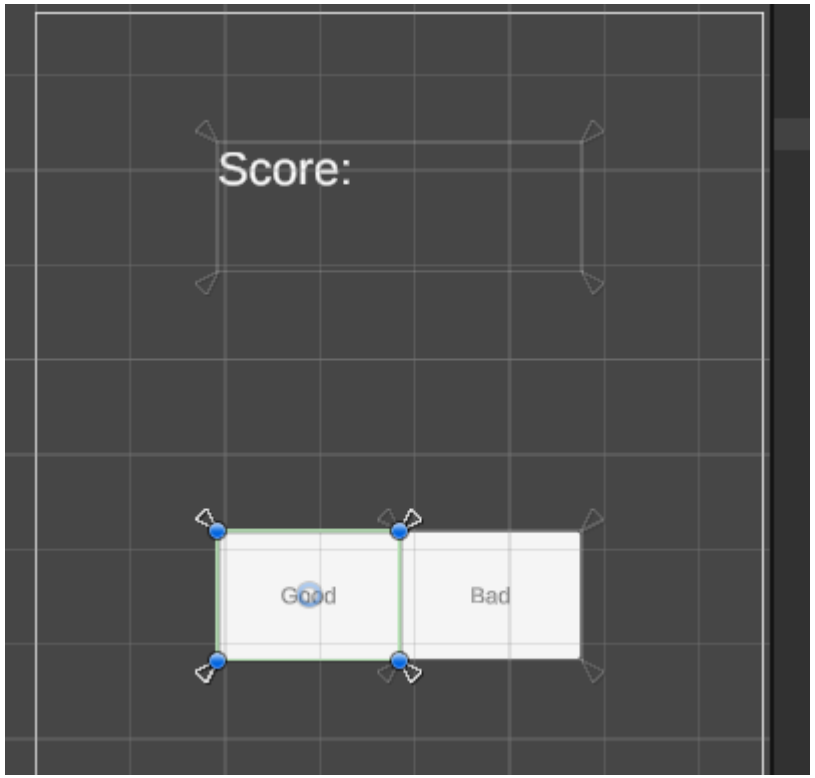
```
private void Awake()
{
    if (instance == null)
    {
        instance = this;
        score = 0; // Initialize score
        DontDestroyOnLoad(gameObject);
    }
    else
    {
        Destroy(gameObject);
    }
}
```

Score Text에 ScoreDisplay.cs 연결

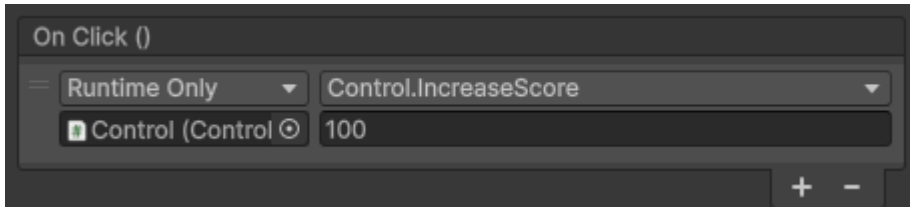



```
using TMPro;
using UnityEngine;

0 references
public class ScoreDisplay : MonoBehaviour
{
    // Start is called once before the first execution of Update after the MonoBehaviour is created
    0 references
    void Start()
    {

    }

    // Update is called once per frame
    0 references
    void Update()
    {
        if (GameManager.Instance == null)
        {
            Debug.LogError("GameManager instance is not available.");
            return;
        }
        gameObject.GetComponent<TextMeshProUGUI>().text = "Score: " + GameManager.Instance.score.ToString();
    }
}
```

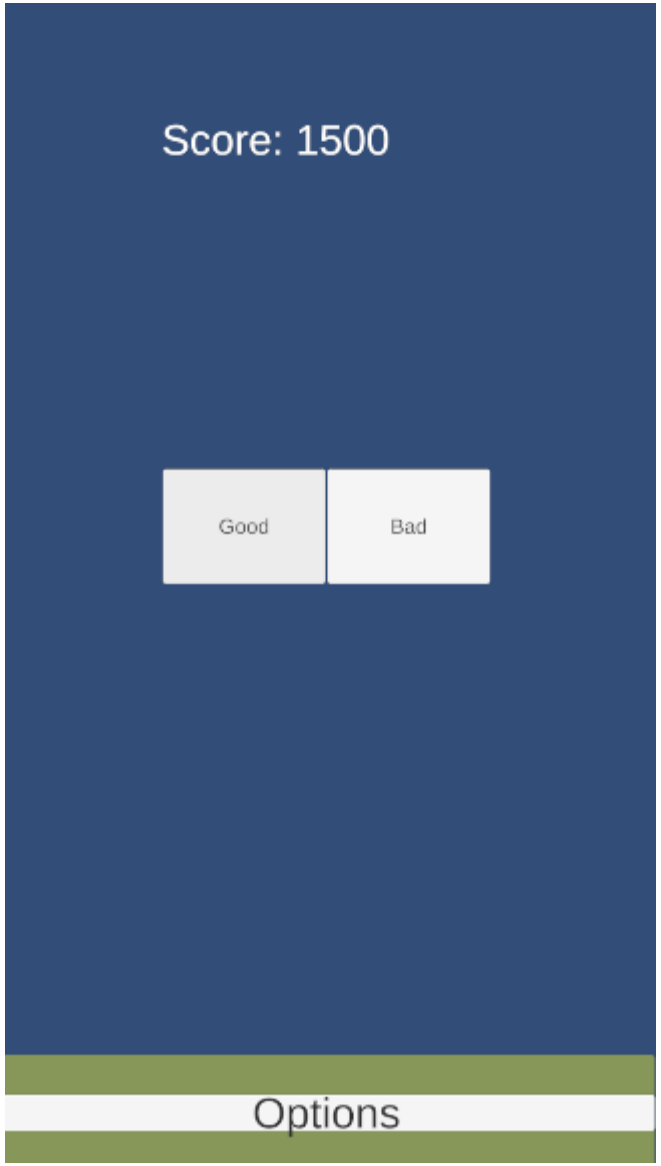
점수 증가




```
0 references
public void IncreaseScore(int amount)
{
    if (GameManager.Instance != null)
    {
        GameManager.Instance.score += amount;
    }
    else
    {
        Debug.LogError("GameManager instance is not available.");
    }
}
```



Score 관리가 이루어짐



도전

1. GameScene에서 Bad 버튼을 누르면 GameOver가 된다
2. GameOver에서 Restart를 누르면 점수가 초기화되고 다시 게임이 시작한다.
3. GameScene에서 Option 버튼을 누르고 옵션 설정을 하고 돌아오면 게임 점수는 유지된다.

복잡한 게임을
설계할 수 있나요