

08.가상현실 제작 실습 7/9

한 일

- FPSController + UI 의 마우스 이슈
- Light
- ~~Audio & Video~~

할 일

- 팀 빌딩
- Audio & Video
- Scene 전환과 데이터 관리

전달

- 출석 인정 신청 방법
 - 학칙이 허용하는 사유(증빙 서류를 제출할 수 있는 사유) + 생리공결 --> 유레카를 통해 신청
 - 기타 사유(증빙 서류를 제출할 수 없는 사유) --> 사이버 캠퍼스의 구글 양식을 통해 신청
 - 출석 인정 신청 승인은 월말마다 진행함
- 출석 인정 신청시 주의사항
 - 기타 사유로 인한 출석 인정 횟수는 학기당 2 회임
 - 생리공결 신청 횟수는 학기당 4 회 이내로 할 것
 - 4 회를 초과할 경우, 동점자 발생시 등급을 하향 조정할 수 있음
 - 생리공결 신청 횟수 만큼 기타 사유 횟수는 차감함
 - 가령, 생리공결 2 회를 신청했고 경우 기타 사유 1 회 신청했을 경우 출석 2 회(생리공결), 결석 1 회(기타 사유 불인정)로 처리함

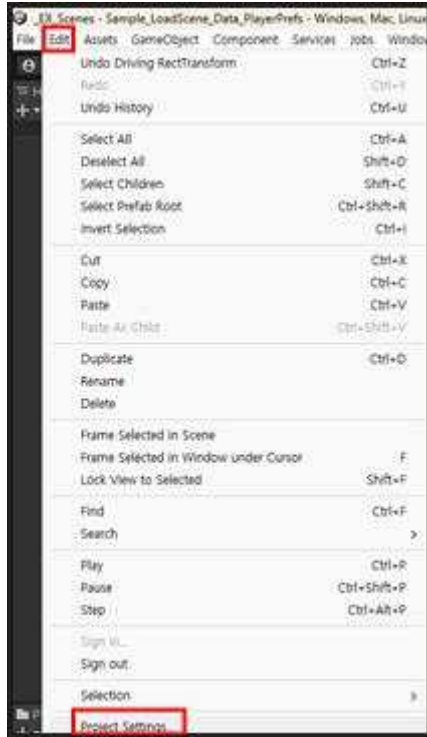
과제

- 제목
 - 실내 공간이 있는 씬 만들기
- 내용
 - 실내 공간을 제작하고 Light, Audio, Video 를 제어할 수 있도록 함
- 요구사항
 - 과제 폴더를 만들고 관련 파일들(Scene, Terrain, Script 등)을 해당 폴더에 모을 것
 - 과제 폴더의 이름은 다음의 형식을 따를 것
 - 예) 학번이 [1234567890]이고 이름이 [홍길동]일 경우, 폴더 이름은 [HW02_1234567890_HGD]으로 정할 것
 - 실내 공간 씬과 실외 공간 씬을 제작할 것(기존 Terrain 이 있는 씬 재활용 가능)
 - 실내 공간에 라디오(오디오 제어), TV(동영상 제어), 전등(Light 제어) 등을 추가할 것
 - 실외 공간에 일상 소음(오디오 제어), 전광판(동영상 제어), 가로등(Light 제어) 등을 추가할 것
 - 하나의 씬에서 다른 씬으로 전환(LoadScene)할 수 있도록 할 것
 - 이전 씬의 데이터(가령, 라디오, TV 등의 SetActive 정보 등)를 유지할 수 있도록 할 것
 - 작업물을 Unity 의 패키지 내보내기 기능을 사용하여 파일(*.unitypackage)로 저장할 것
 - Play 모드에서 Scene 의 특징점을 잘 보여줄 수 있도록 조작하며 화면 캡처를(동영상) 할 것
- 제출물
 - 다음 2 가지를 제출할 것
 - 패키지 파일
 - 과제 폴더를 [Export Package]로 내보내고 Zip 포맷으로 압축하여 사이버 캠퍼스에 제출할 것
 - 용량이 제출 허용치를 초과할 경우, 다운로드 가능한 링크를 제출해도 무방함
 - 화면 캡처(동영상) 파일
 - Play 모드를 동영상으로 캡처하고 mp4 포맷으로 저장하여 제출
 - 캡션이나 안내, 설명 등은 넣을 필요 없음
 - 스마트폰으로 촬영해도 무방함

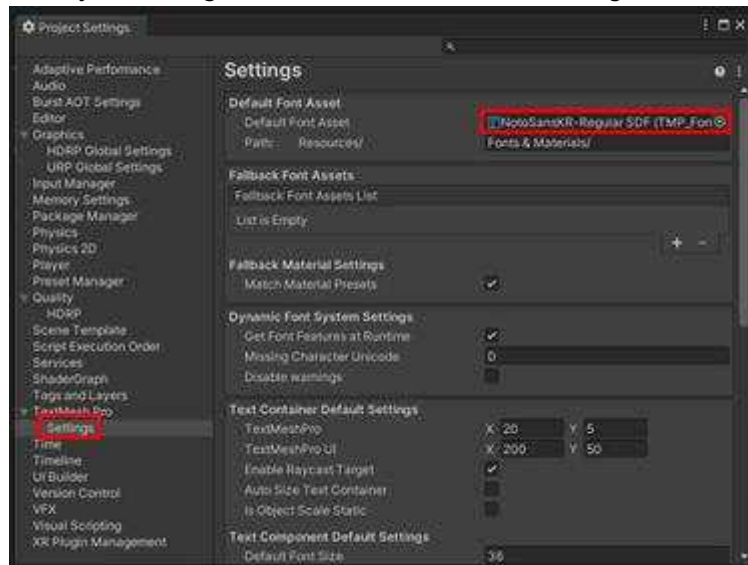
- 제출 기한
 - 일요일 23:59
- 제출 방법
 - 사이버 캠퍼스
- 주의
 - 제출한 패키지가 제대로 열리는지 반드시 확인할 것
 - 가령, 새로운 유니티 프로젝트를 만들고 해당 프로젝트에서 자신의 패키지가 제대로 열리는지 확인할 것
 - 만약 열리지 않을 경우 미제출로 처리함
 - 링크를 제출할 경우, 교수자가 다운로드 가능한 링크인지 확인할 것
 - 만약 다운로드가 불가하다면 미제출로 처리함

TMPro 기본 폰트 설정하기

- 한글을 지원하는 폰트를 기본 폰트로 설정함
- 유니티 메뉴 > [File] > [Project Settings...]



- [Project Settings] 창 > [TextMesh Pro] > [Settings] > [Default Font Asset] > [Default Font Asset] > 원하는 폰트 선택

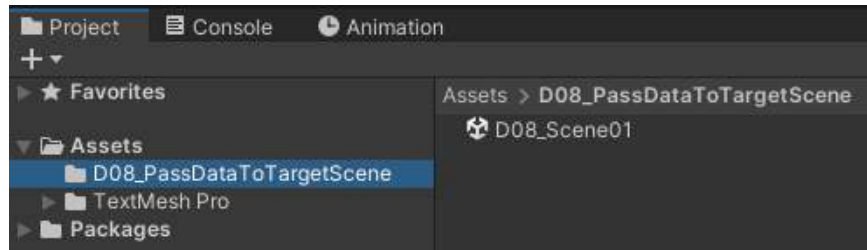


다른 씬에 데이터 전달하기

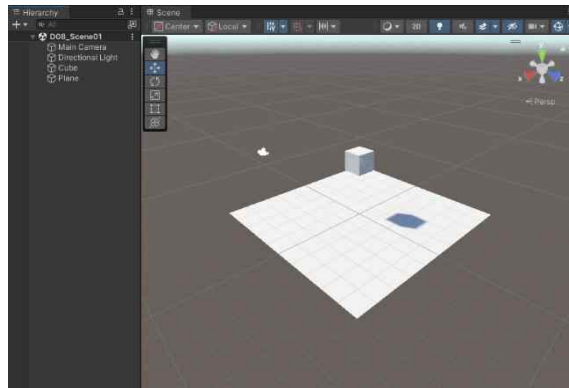
- 씬을 새로 열리면 기존 씬의 모든 게임 오브젝트는 소멸됨(Destroy)
- 따라서 변수의 값도 소멸함
- 다른 씬으로 데이터를 전달하는 방법 중에 PlayerPrefs 를 활용하는 기법이 있음

실습 환경 구축하기

- 새로운 작업 폴더와 씬을 만들

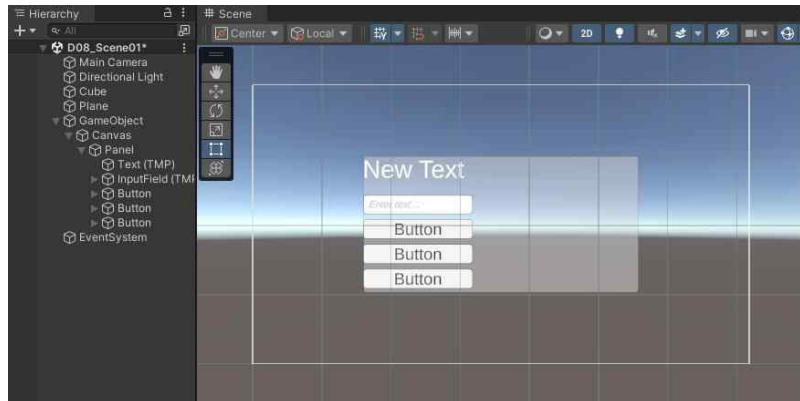


- 첫 번째 씬을 다음과 같이 구성
 - 3D Object
 - Cube (Rigidbody 컴포넌트 추가)
 - Plane

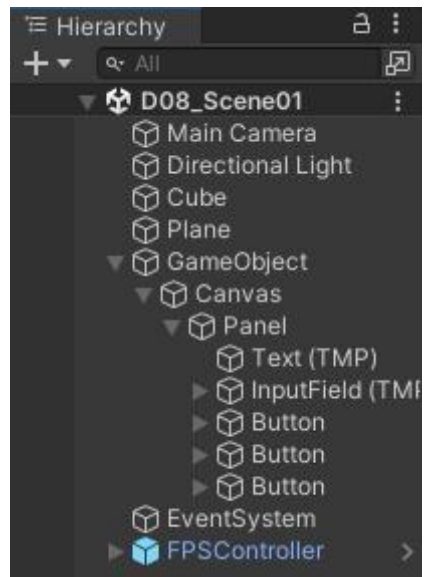


○ UI

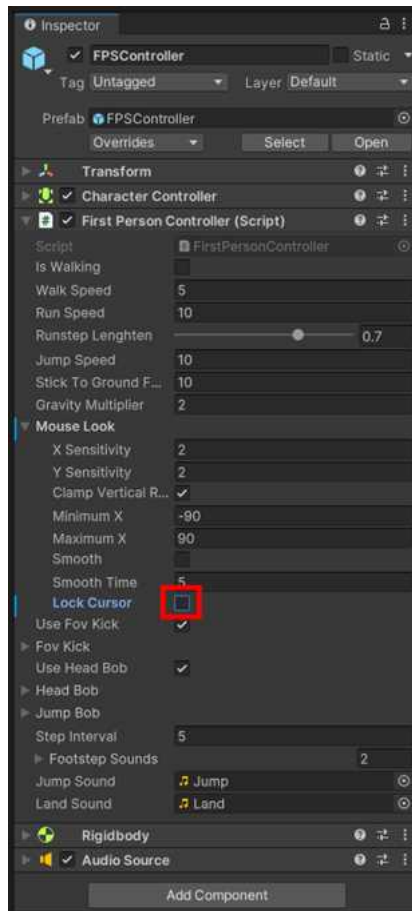
- Script 컴포넌트를 포함하고 UI 들을 그루핑 할 수 있는 Empty 게임 오브젝트 [Create Empty]
- 레이아웃을 관리할 수 있는 [Panel]
- 메시지를 표시할 수 있는 [Text - TextMeshPro]
- 사용자의 입력치를 받을 수 있는 [Input Field - TextMeshPro]
- 사용자의 입력치를 메시지 게임 오브젝트에 표시하는 기능을 가진 [Button - TextMeshPro]
- 사용자의 입력치를 메시지 게임 오브젝트에 표시하고 또한 PlayerPrefs 에 저장도 하는 기능을 가진 [Button - TextMeshPro]
- Scene 을 Loading 하는 기능을 가진 [Button - TextMeshPro]



- FPSController



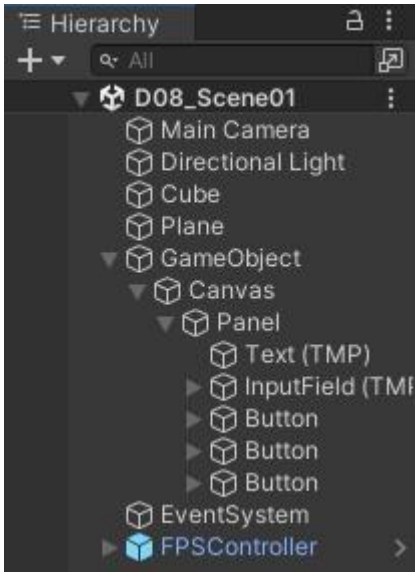
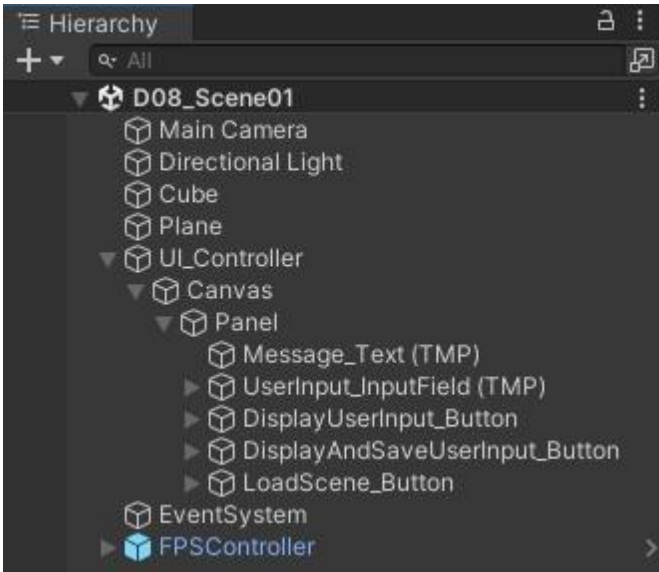
- FPSController 의 [Lock Cursor] 기능 해제



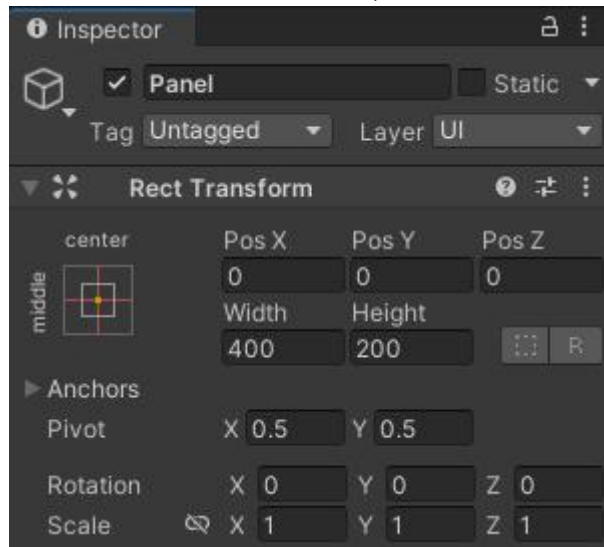
- 게임 창의 모습



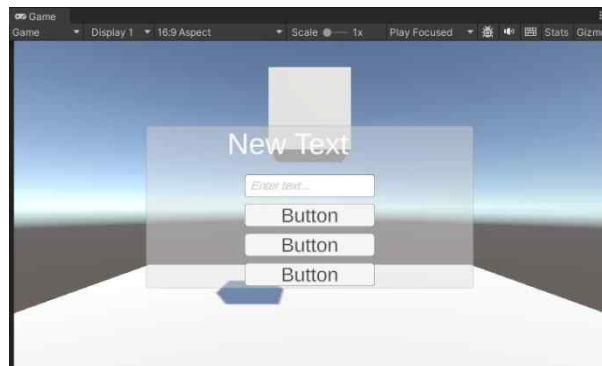
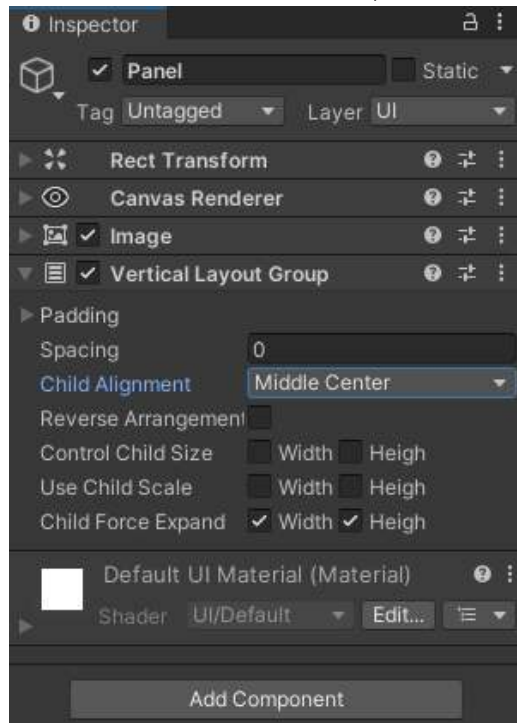
- 게임 오브젝트의 이름 수정

수정 전	수정 후
 <p>Unity Hierarchy panel showing the initial object structure. The hierarchy under D08_Scene01 is as follows:</p> <ul style="list-style-type: none"> Main Camera Directional Light Cube Plane GameObject <ul style="list-style-type: none"> Canvas <ul style="list-style-type: none"> Panel <ul style="list-style-type: none"> Text (TMP) InputField (TMP) Button Button Button EventSystem FPSController 	 <p>Unity Hierarchy panel showing the object structure after renaming. The hierarchy under D08_Scene01 is as follows:</p> <ul style="list-style-type: none"> Main Camera Directional Light Cube Plane UI_Controller <ul style="list-style-type: none"> Canvas <ul style="list-style-type: none"> Panel <ul style="list-style-type: none"> Message_Text (TMP) UserInput_InputField (TMP) DisplayUserInput_Button DisplayAndSaveUserInput_Button LoadScene_Button EventSystem FPSController

- [Panel] 게임 오브젝트의 [Inspector] > [Rect Transform]을 다음과 같이 설정



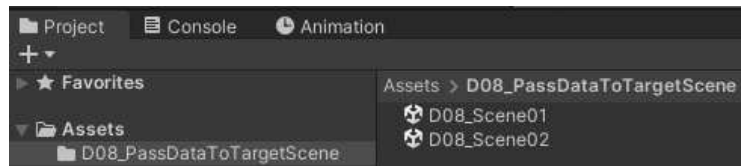
- [Panel] 게임 오브젝트의 [Inspector] > [Vertical Layout Group] 컴포넌트 추가 후 [Child Alignment]를 [Middle Center]로 변경



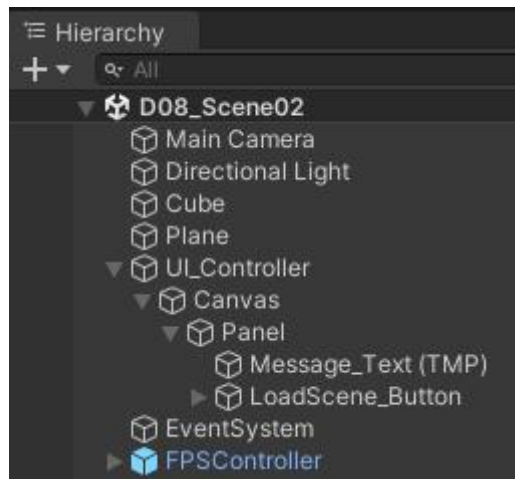
- 버튼의 레이블 수정



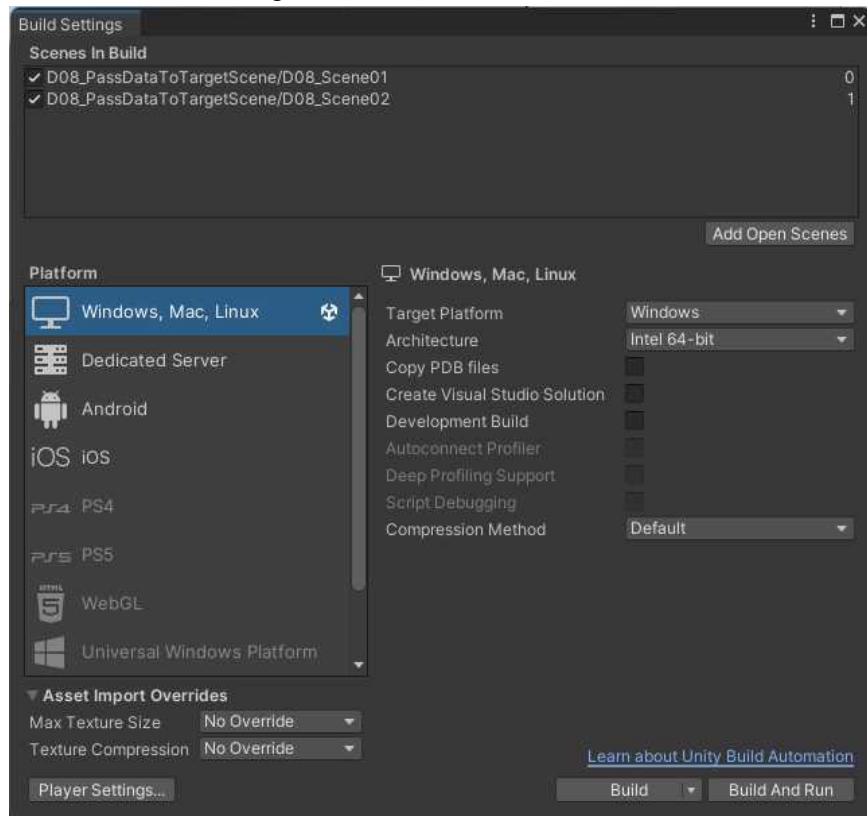
- 첫 번째 씬을 [Save As...] 하여 두 번째 씬으로 저장



- 두 번째 씬에서 불필요한 버튼 제거



- 두 씬을 Build Settings 에 등록



첫 번째 씬 스크립팅

- 스크립트 파일을 생성하고 다음과 같이 작성
- D08_Scene01_Controller.cs

```
using System.Collections;
using System.Collections.Generic;
using TMPro;
using UnityEngine;
using UnityEngine.SceneManagement;
using UnityEngine.UI;

public class D08_Scene01_Controller : MonoBehaviour
{
    string UserInput = "";

    void Update()
    {
        if (Input.GetKeyDown(KeyCode.X))
        {
            PlayerPrefs.DeleteAll();
        }
    }

    public void OnClick_AssignData(TMP_InputField InputField)
    {
        UserInput = InputField.text;
    }

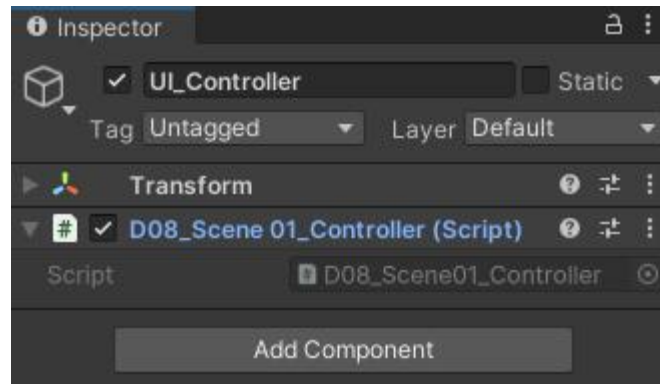
    public void OnClick_Display_UserInput(TMP_Text Message)
    {
        Message.text = $"{UserInput}";
    }

    public void OnClick_DisplayAndSet_UserInput(TMP_Text Message)
    {
        Message.text = $"{UserInput}";
    }
}
```

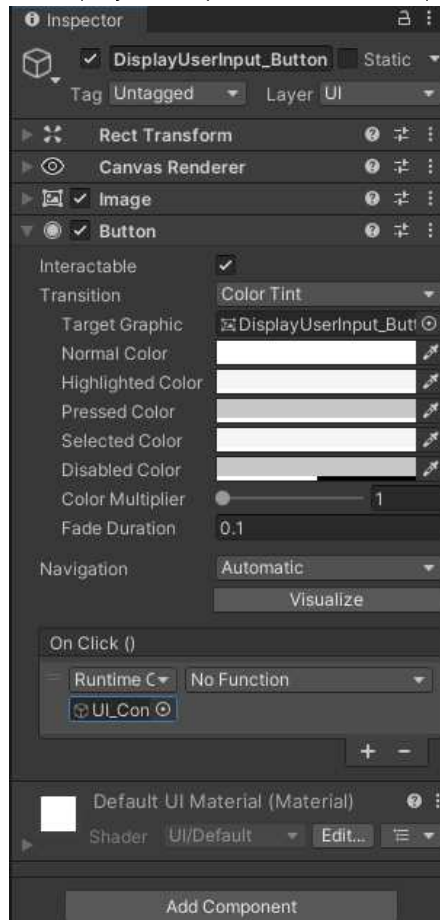
```
        PlayerPrefs.SetString("Input", Message.text);
    }

    public void OnClick_LoadScene(Object SceneObject)
    {
        SceneManager.LoadScene(SceneObject.name);
    }
}
```

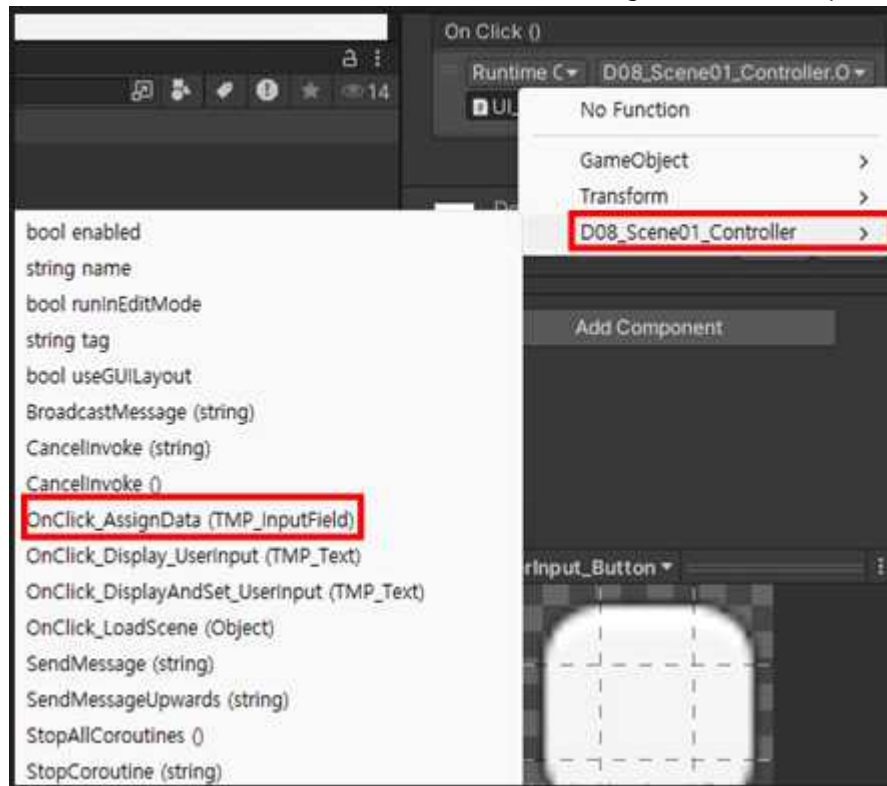
- 스크립트를 [UI_Controller]의 컴포넌트로 추가



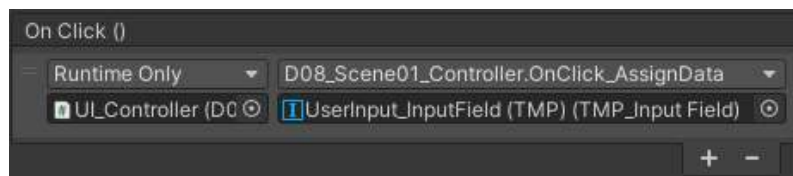
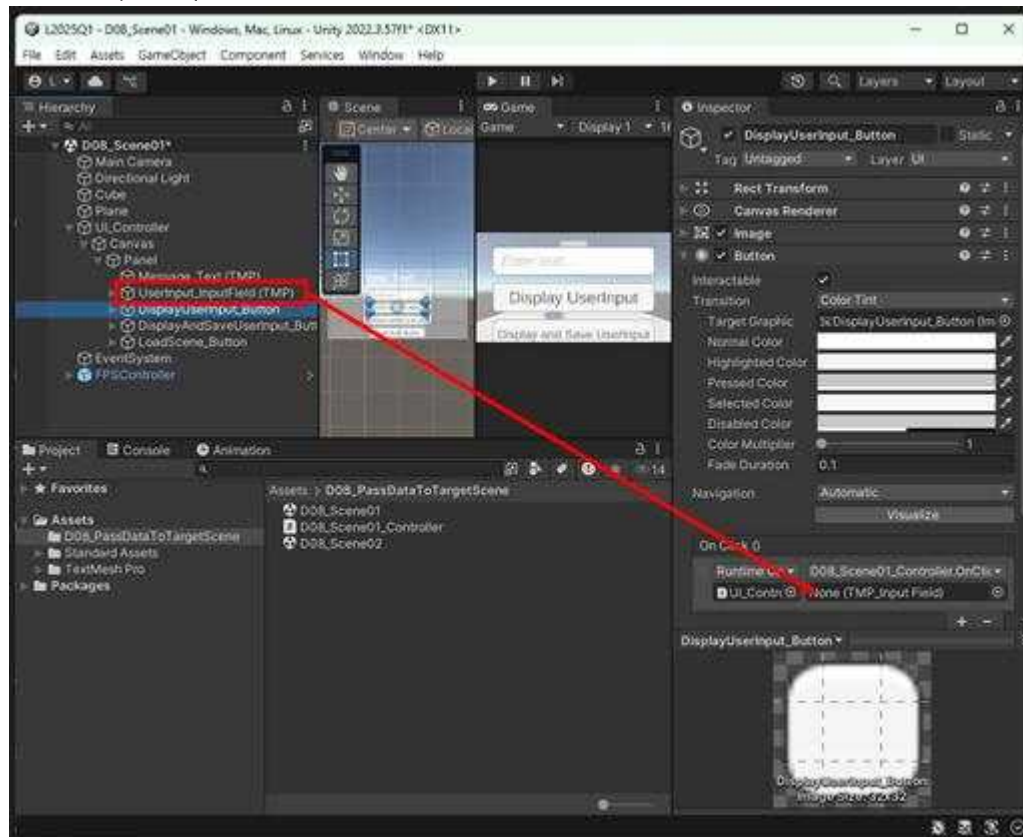
- [DisplayUserInput_Button]의 [Inspector] > [Button] > [OnClick] > [+]를 하고 [UI_Controller] 게임 오브젝트를 [Object]에 할당



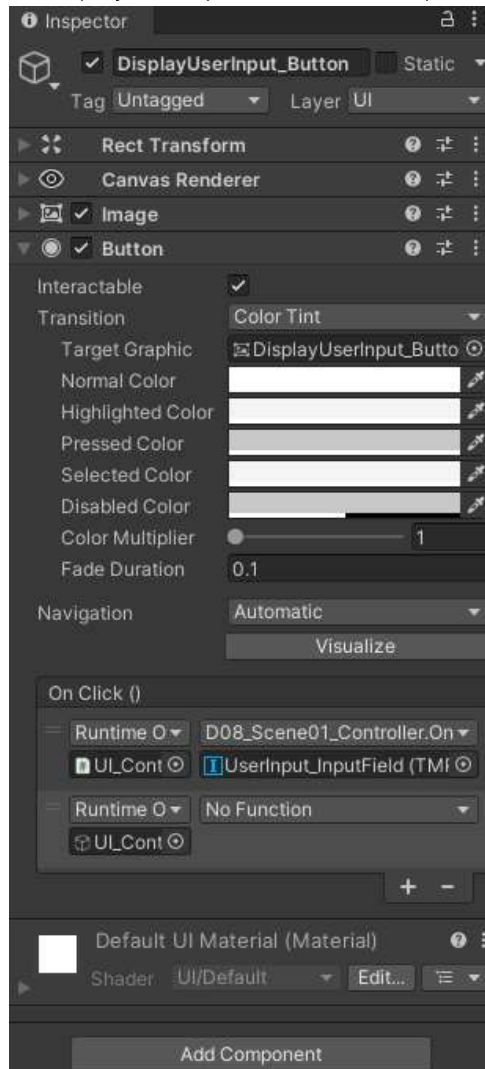
- 함수를 [D08_Scene01_Controller] > [OnClick_AssignData (TMP_InputField)]로 설정



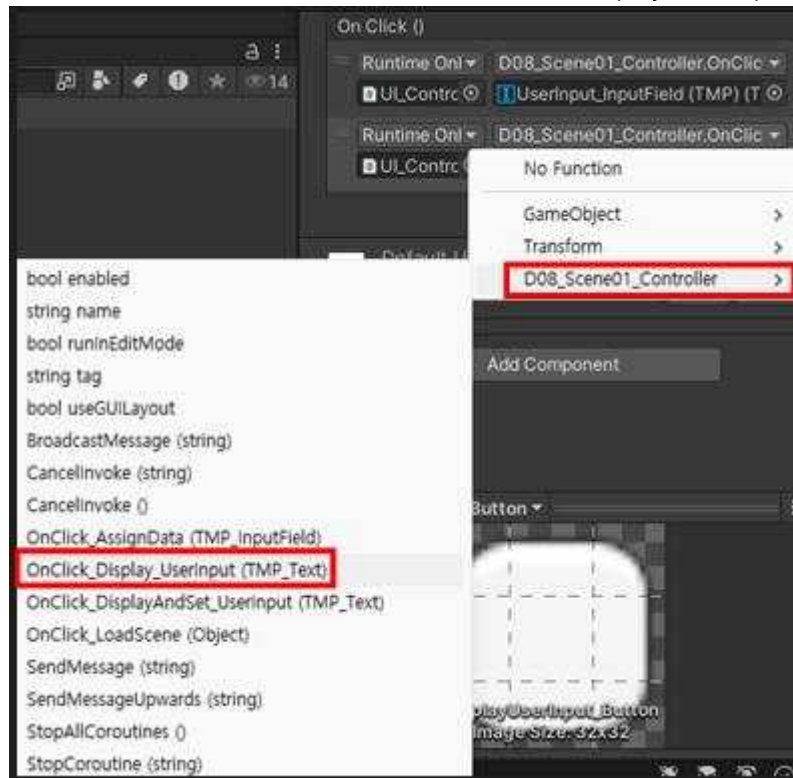
- [UserInput_InputField (TMP)] 게임 오브젝트를 매개변수로 할당



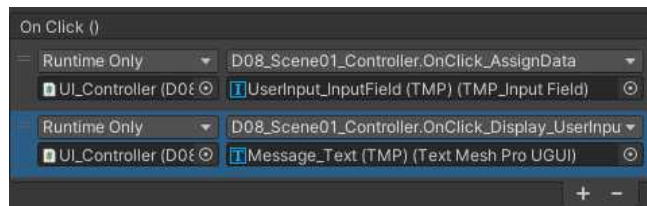
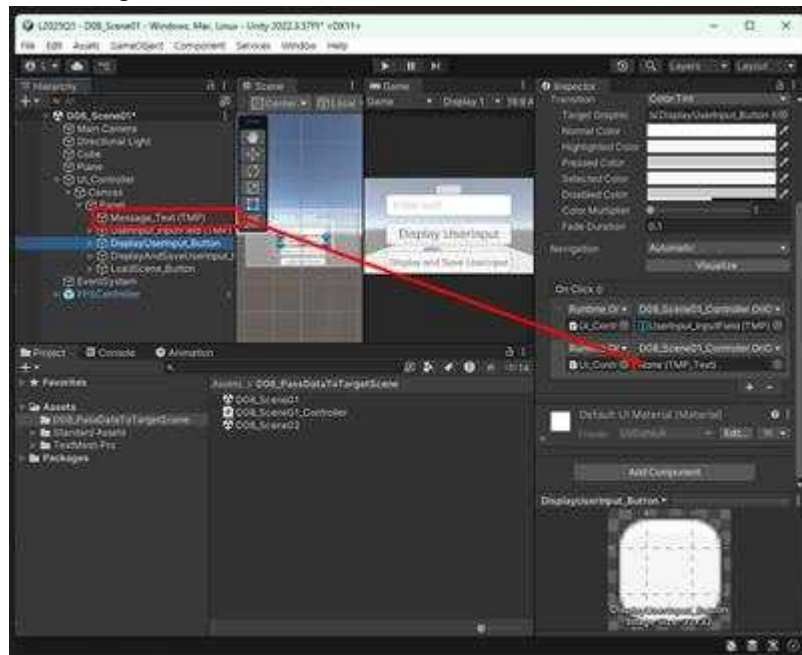
- [DisplayUserInput_Button]의 [Inspector] > [Button] > [OnClick] > [+]를 하고 [UI_Controller] 게임 오브젝트를 [Object]에 할당



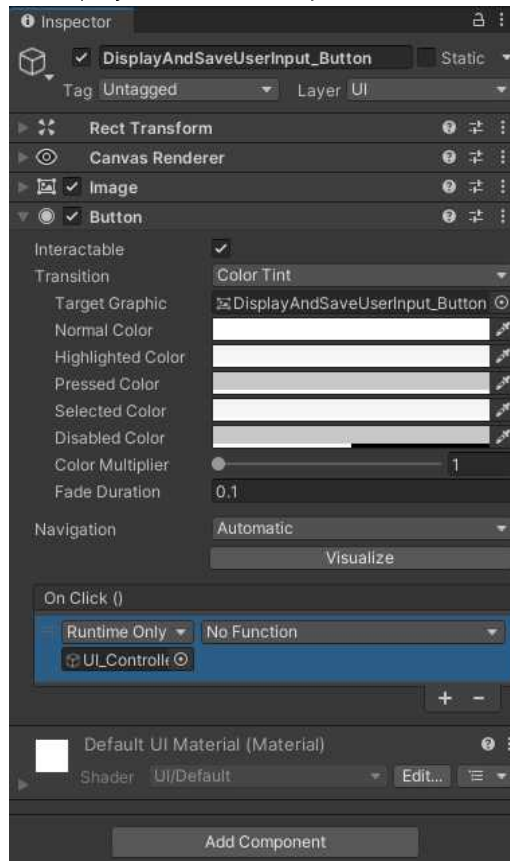
- 함수를 [D08_Scene01_Controller] > [OnClick_Display_UserInput (TMP_Text)]로 설정



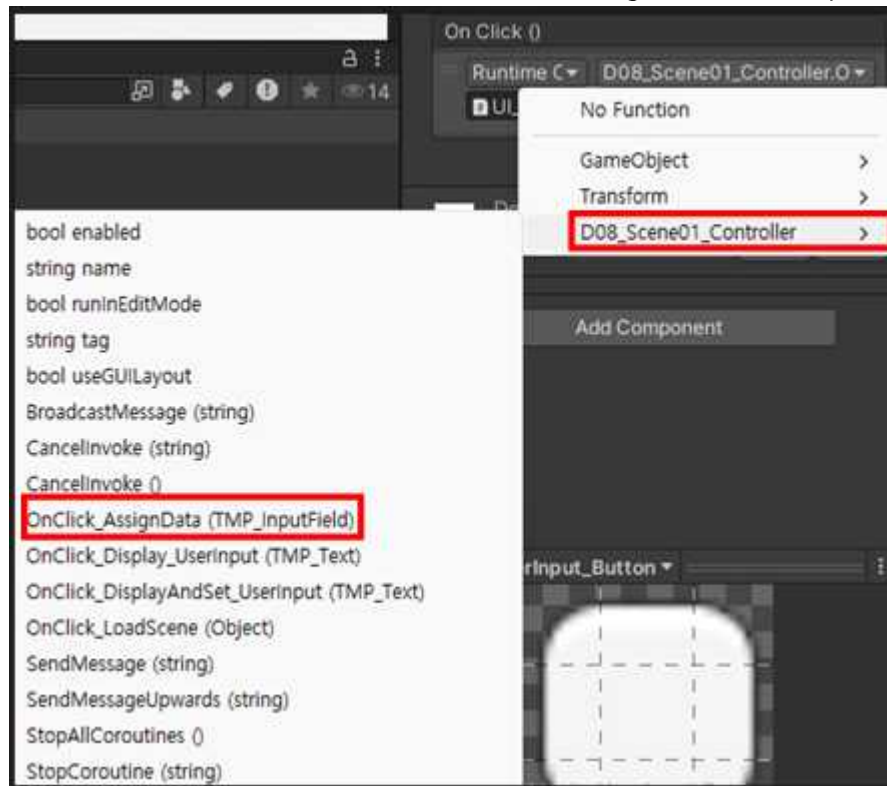
- [Message_Text (TMP)] 게임 오브젝트를 매개변수로 할당



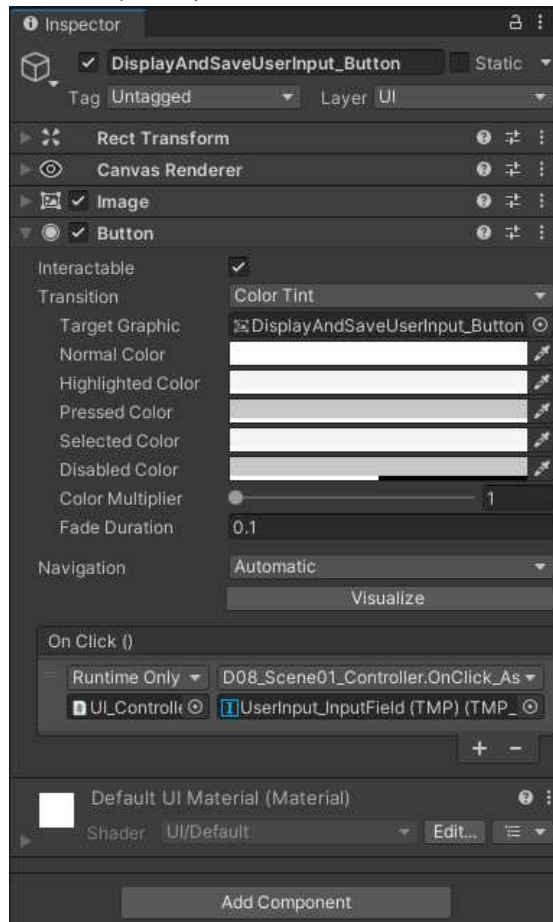
- [DisplayAndSaveUserInput_Button]의 [Inspector] > [Button] > [OnClick] > [+]를 하고 [UI_Controller] 게임 오브젝트를 [Object]에 할당



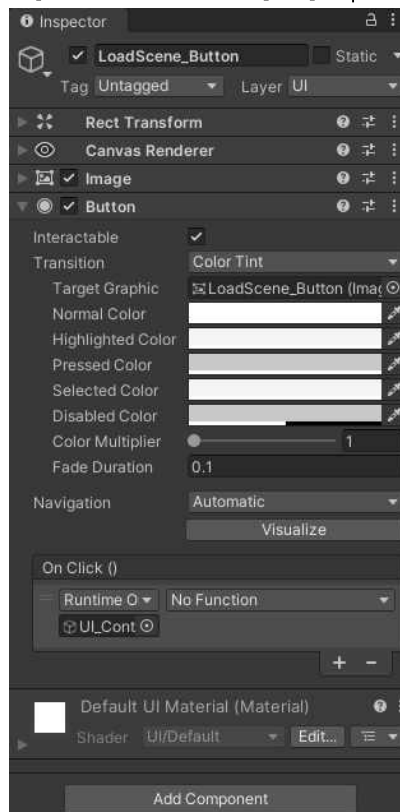
- 함수를 [D08_Scene01_Controller] > [OnClick_AssignData (TMP_InputField)]로 설정



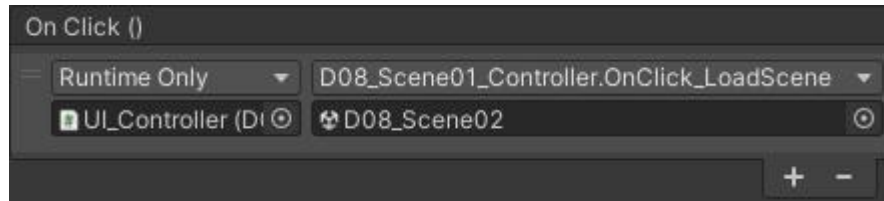
- [UserInput_InputField (TMP)] 게임 오브젝트를 매개변수로 할당



- [DisplayAndSaveUserInput_Button]의 [Inspector] > [Button] > [OnClick] > [+]를 하고 [UI_Controller] 게임 오브젝트를 [Object]에 할당
- 함수를 [D08_Scene01_Controller] > [OnClick_DisplayAndSet_UserInput (TMP_Text)]로 설정
- [Message_Text (TMP)] 게임 오브젝트를 매개변수로 할당
- [LoadScene_Button]의 [Inspector] > [Button] > [OnClick] > [+]를 하고 [UI_Controller] 게임 오브젝트를 [Object]에 할당



- 함수를 [D08_Scene01_Controller] > [OnClick_LoadScene (Object)]로 설정
- [D08_Scene02] 씬 파일을 매개변수로 할당



두 번째 씬 스크립팅

- 두 번째 스크립트 파일을 생성하고 다음 스크립트 작성

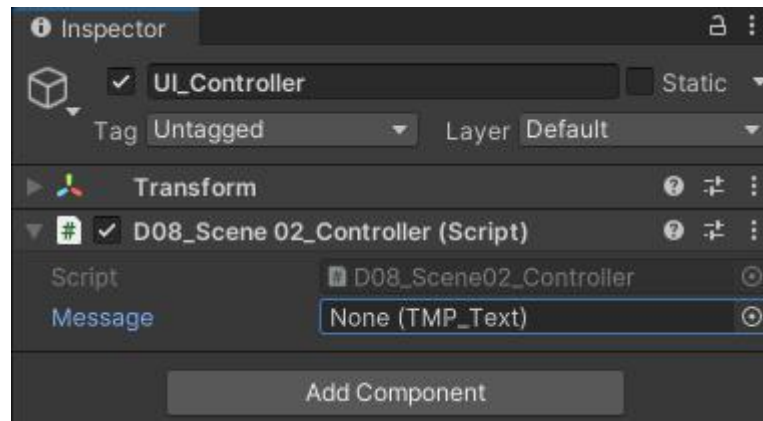
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro;
using UnityEngine.SceneManagement;

public class D08_Scene02_Controller : MonoBehaviour
{
    string UserInput;
    public TMP_Text Message;

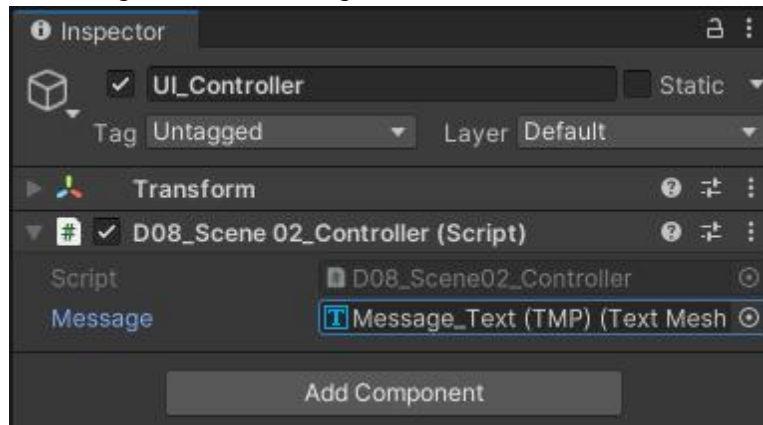
    void Start()
    {
        UserInput = PlayerPrefs.GetString("Input");
        if (string.IsNullOrEmpty(UserInput))
        {
            Message.text = "PlayerPrefs 에 저장한 데이터가 없습니다.";
        }
        else
        {
            Message.text = UserInput;
        }
    }

    public void OnClick_LoadScene(Object SceneObject)
    {
        SceneManager.LoadScene(SceneObject.name);
    }
}
```

- 스크립트를 [UI_Controller]의 컴포넌트로 추가

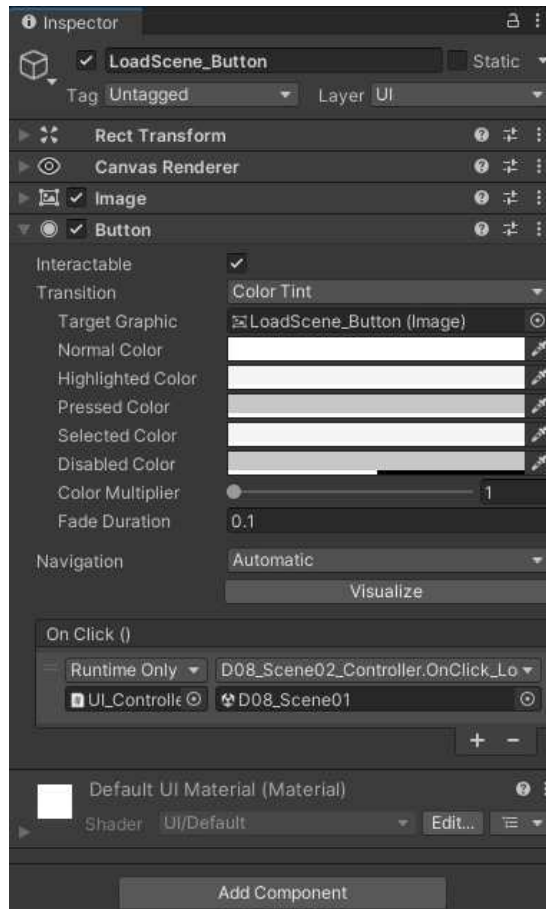


- [Message] 변수에 [Message_Text (TMP)] 게임 오브젝트 할당



- [LoadScene_Button]의 [Inspector] > [Button] > [OnClick()] > [+]를 하고 [UI_Controller] 게임 오브젝트를 [Object]에 할당
- 함수를 [D08_Scene02_Controller] > [OnClick_LoadScene (Object)]로 설정

- [D08_Scene01] 씬 파일을 매개변수로 할당



테스트

- [D08_Scene01] 씬에서 Input Field 에 텍스트를 입력하고 [DisplayUserInput_Button]이나 [DisplayAndSaveUserInput_Button]을 클릭하면 [Message_Text (TMP)]에 입력한 내용이 나타남
- [DisplayUserInput_Button]으로 표시한 데이터는 새로운 씬에 전달되지 않음
- [DisplayAndSaveUserInput_Button]으로 표시한 데이터는 새로운 씬에 전달됨
- PlayerPrefs 의 SetString(). GetString()은 string 데이터를 key 와 함께 저장하는 기능을 제공함