

# | Unity Backend

“게임 유저 관리” 기능을 이용한 로그인, 계정 생성, 이메일 설정, 아이디/비밀번호 찾기, 유저 정보 출력, 닉네임 설정

Created in 2023-01-26  
Last Updated 2023-02-01  
Unity Version 2022.2.2f1

## *Index*

- ◆ 뒤끝 함수 호출 방식과 비동기 호출
- ◆ 게임 유저 관리

# 뒤끝 함수 호출 방식과 비동기 호출

- 뒤끝 함수 호출 방식
- 비동기 호출을 위한 처리



# 뒤끝 함수 호출 방식과 비동기 호출

## ■ 뒤끝 함수 호출 방식

- 뒤끝에서 제공하는 함수는 동기, 비동기, SendQueue로 호출 가능

### ■ 동기

- 요청 시점에서 시간이 얼마나 걸리던지 요청한 자리에서 결과가 주어지는 함수
- 함수를 호출하면 리턴 값이 반환될 때까지 메인 스레드의 동작이 중지
- 비동기 함수와 다르게 순차적으로 함수가 호출되어 순서가 섞일 가능성이 없으며, 작업 도중 다른 스레드에 의해 오류가 발생할 경우가 없기 때문에 확실한 결과 제공

```
11 // 동기
12 var bro = Backend.BMember.CustomLogin(ID, PW);
13 // 로그인에 성공했을 때 처리
14 if ( bro.IsSuccess() )
15 {
16 }
17 // 로그인에 실패했을 때 처리
18 else
19 {
20     // statusCode의 값에 따라 실패 원인을 알 수 있다.
21     int statusCode = int.Parse(bro.GetStatusCode());
22 }
```



# 뒤끝 함수 호출 방식과 비동기 호출

## ■ 비동기

- 동기 함수와 다르게 호출 시점에서 실행 결과를 기다리지 않는 함수
- 비동기 함수의 경우 비동기 함수를 호출한 스레드에서 해당 함수 호출에 대한 결과를 기다리지 않고 바로 다음 작업을 수행할 수 있다.
- 비동기 함수의 콜백 함수
  - **비동기 IO 스레드 내에서 실행**하는 방법
    - 유니티의 정책에 따라 별도의 스레드에서 **유니티 MonoBehaviour 객체에 접근 불가능**
    - 콜백 함수 내에서 유니티 객체, UI 객체 등에 접근할 수 없기 때문에 비동기 함수의 요청 결과에 따른 처리를 위해 별도의 **Dispatcher 사용 필요**
  - 콜백 함수 풀링을 사용해 **메인 스레드에서 실행**하는 방법
    - 콜백 함수 풀링 기능을 사용하면 콜백 함수 내에서도 **유니티 MonoBehaviour 객체에 접근 가능**
    - 콜백 함수 내에서 유니티 객체, UI 객체 등에 접근 가능
    - **뒤끝 초기화 시 useAsynPoll=true 설정과 Update() 메소드에서 AsyncPoll() 메소드 호출 필요**



# 뒤끝 함수 호출 방식과 비동기 호출

## ■ 비동기 - 콜백 폴링 함수

```
24 // 비동기 - 콜백 폴링 함수
25 // 별도의 큐에 저장된 콜백 함수를 실행하려면 AsyncPoll() 메소드 호출이 필요하다.
26 // https://developer.thebackend.io/unity3d/guide/Async/AsyncFuncPoll/
27 Backend.BMember.CustomLogin(ID, PW, callback =>
28 {
29     // 로그인에 성공했을 때
30     if ( callback.IsSuccess() )
31     {
32     }
33     // 로그인에 실패했을 때
34     else
35     {
36         // statusCode의 값에 따라 실패 원인을 알 수 있다.
37         int statusCode = int.Parse(callback.GetStatusCode());
38     }
39 });
```



# 뒤끝 함수 호출 방식과 비동기 호출

## ■ SendQueue

- 비동기 함수 호출 시 바로 호출하지 않고 큐에 적재한 후 순차적으로 함수를 호출
- 동기의 장점인 순차적으로 호출되는 점과 비동기의 장점인 함수 호출의 결과를 기다리지 않고 바로 다음 작업이 수행되며, 프로그램이 중지되지 않는 것처럼 보인다는 점이 합쳐져 있다.

```
41 // SendQueue
42 // SendQueue의 Enqueue()에 호출할 메소드를 등록하고 별도의 시작 처리가 필요하다.
43 // https://developer.thebackend.io/unity3d/guide/Async/SendQueueDetail/
44 // SendQueue.Enqueue\(Backend.BMember.CustomLogin, ID, PW, callback =>
45 {
46     // 로그인에 성공했을 때
47     if ( callback.IsSuccess() )
48     {
49     }
50     // 로그인에 실패했을 때
51     else
52     {
53         // statusCode의 값에 따라 실패 원인을 알 수 있다.
54         int statusCode = int.Parse(callback.GetStatusCode());
55     }
56 });
```

SendQueue는 메인 스레드가 아닌  
별도의 스레드에서 동작



# 뒤끝 함수 호출 방식과 비동기 호출

- 동기, 비동기 호출 예시



동기



비동기 - 콜백 풀링 함수





# 뒤끝 함수 호출 방식과 비동기 호출

## ■ 비동기 호출을 위한 처리

### ■ 비동기 메소드 호출을 위해 AsyncPoll() 호출

#### □ BackendManager Script 수정

```
1  using UnityEngine;
2  using BackEnd;    // 뒤끝 SDK
3
4  public class BackendManager : MonoBehaviour
5  {
6      private void Awake()
7      {
8          // Update() 메소드의 Backend.AsyncPoll(); 호출을 위해 오브젝트를 파괴하지 않는다
9          DontDestroyOnLoad(gameObject);
10
11         // 뒤끝 서버 초기화 AsyncPoll() 메소드를 호출해야 별도의 큐에 저장된 콜백 함수를 호출할 수 있기
12         BackendSetup();           때문에 모든 씬에서 삭제되지 않고 계속 남아있도록 DontDestroyOnLoad()를 호출
13     }
14
15     private void Update()
16     {
17         // 서버의 비동기 메소드 호출(콜백 함수 폴링)을 위해 작성
18         // 참고 : https://developer.thebackend.io/unity3d/guide/Async/AsyncFuncPoll/
19         if ( Backend.IsInitialized )
20         {
21             Backend.AsyncPoll();
22         }
23     }
24
25     private void BackendSetup()...
```



# 뒤끝 함수 호출 방식과 비동기 호출

- Login 씬 생성
  - File - New Scene

The screenshot shows the Unity interface with the 'File' menu open and the 'New Scene' dialog box displayed. The 'File' menu is highlighted with a red box, and the 'New Scene' option is also highlighted. In the 'New Scene' dialog, the 'Basic 2D (Built-in)' template is selected and highlighted with a red box. The 'Create' button at the bottom right of the dialog is also highlighted with a red box.

File Edit Assets GameObject Component Services

New Scene Ctrl+N

Open Scene Ctrl+O

Open Recent Scene >

Save Ctrl+S

Save As... Ctrl+Shift+S

Save As Scene Template...

New Project...

Open Project...

Save Project

Build Settings... Ctrl+Shift+B

Build And Run Ctrl+B

Exit

New Scene

Scene Templates in Project

2D Basic 2D (Built-in)

3D Basic 3D (Built-in)

Empty

Basic 2D (Built-in)

Description

Contains an orthographic camera setup for 2D games. Works with built-in renderer.

To begin using a template, create a template from an existing scene in your project. Click to see Scene template documentation.

Load additively

Create Cancel



# 뒤끝 함수 호출 방식과 비동기 호출

## ■ 카메라 설정

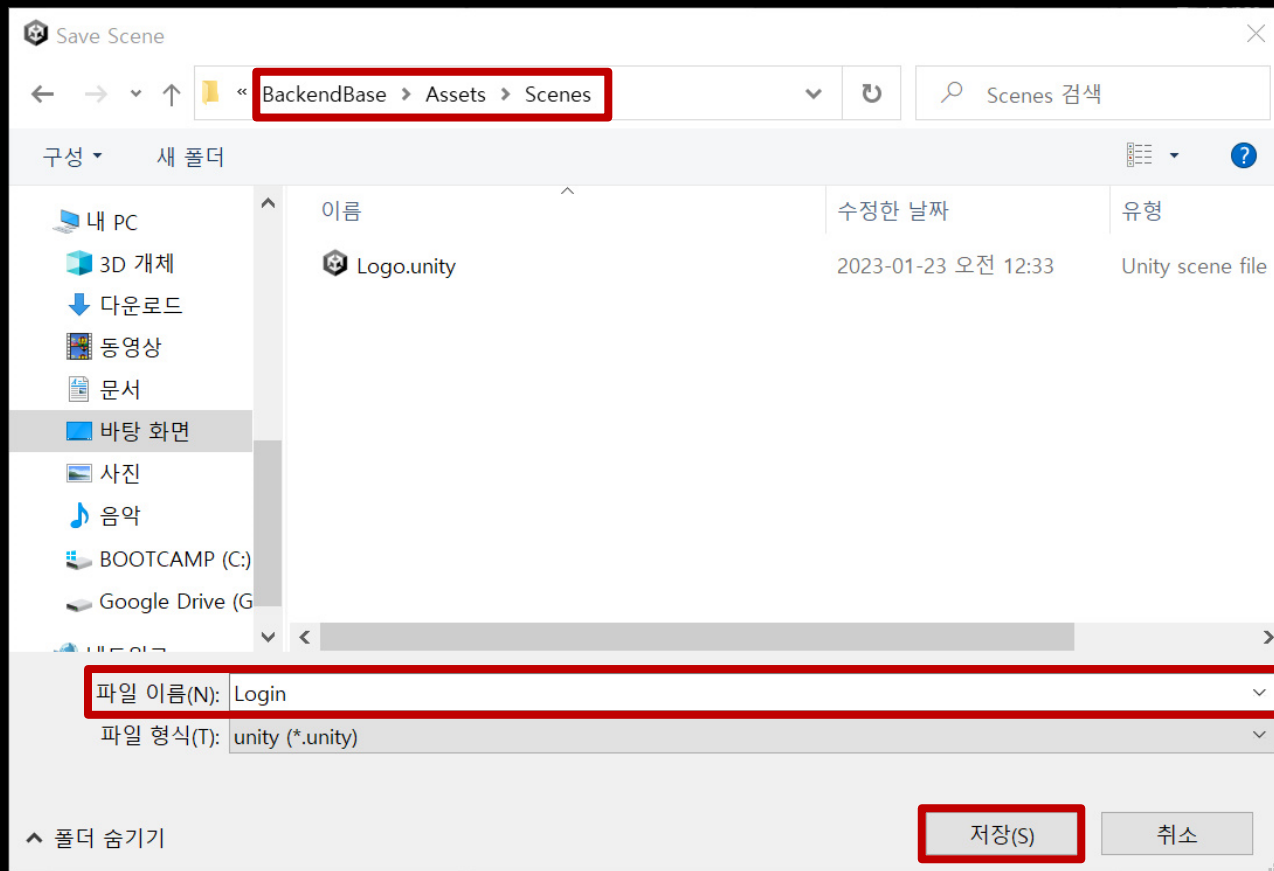
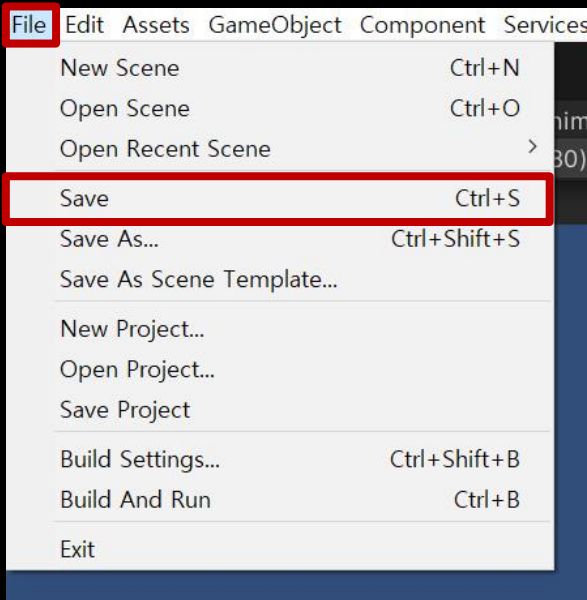
The screenshot shows the Unity Inspector window with the following components and settings:

- Hierarchy:** Shows 'Main Camera' selected under 'Untitled\*'.
- Inspector:**
  - Main Camera:** Tag: MainCamera, Layer: Default.
  - Transform:** Position (X: 0, Y: 0, Z: -10), Rotation (X: 0, Y: 0, Z: 0), Scale (X: 1, Y: 1, Z: 1).
  - Camera (highlighted):**
    - Clear Flags: Solid Color
    - Background: (highlighted with red dashed box)
    - Culling Mask: Everything, Color (0, 0, 0, 0)
    - Projection: Orthographic
    - Size: 5
    - Clipping Planes: Near: 0.3, Far: 1000
    - Viewport Rect: X: 0, Y: 0, W: 1, H: 1
    - Depth: -1
    - Rendering Path: Use Graphics Settings
    - Target Texture: None (Render Texture)
    - Occlusion Culling:
    - HDR: Use Graphics Settings
    - MSAA: Use Graphics Settings
    - Allow Dynamic Resol:
    - Target Display: Display 1
  - Audio Listener:**



# 뒤끝 함수 호출 방식과 비동기 호출

- Login 실패 저장
  - File - Save





# 뒤끝 함수 호출 방식과 비동기 호출

- Scenes In Build에 씬 등록
  - File - Build Settings

The screenshot displays the Unity Build Settings interface. In the 'Scenes In Build' section, the 'Scenes/Login' entry is checked and highlighted with a red box. The 'Assets' panel on the right shows the 'Login' scene file under 'Assets > Scenes' also highlighted with a red box. The 'Platform' section shows 'Windows, Mac, Linux' selected. The 'Target Platform' is set to 'Windows' and 'Architecture' is 'Intel 64-bit'. The 'Build' button is visible at the bottom right.



# 뒤끝 함수 호출 방식과 비동기 호출

- 씬 전환과 같은 유틸 메소드를 정의하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "Utils"로 변경

```
1 using UnityEngine.SceneManagement;
2
3 public enum SceneNames { Logo=0, Login, }
4
5 public static class Utils
6 {
7     public static string GetActiveScene()
8     {
9         return SceneManager.GetActiveScene().name;
10    }
11
12    public static void LoadScene(string sceneName="")
13    {
14        if ( sceneName == "" )
15        {
16            SceneManager.LoadScene(GetActiveScene());
17        }
18        else
19        {
20            SceneManager.LoadScene(sceneName);
21        }
22    }
23
24    public static void LoadScene(SceneNames sceneName)
25    {
26        // SceneNames 열거형으로 매개변수를 받아온 경우 ToString() 처리
27        SceneManager.LoadScene(sceneName.ToString());
28    }
29 }
```



# 뒤끝 함수 호출 방식과 비동기 호출

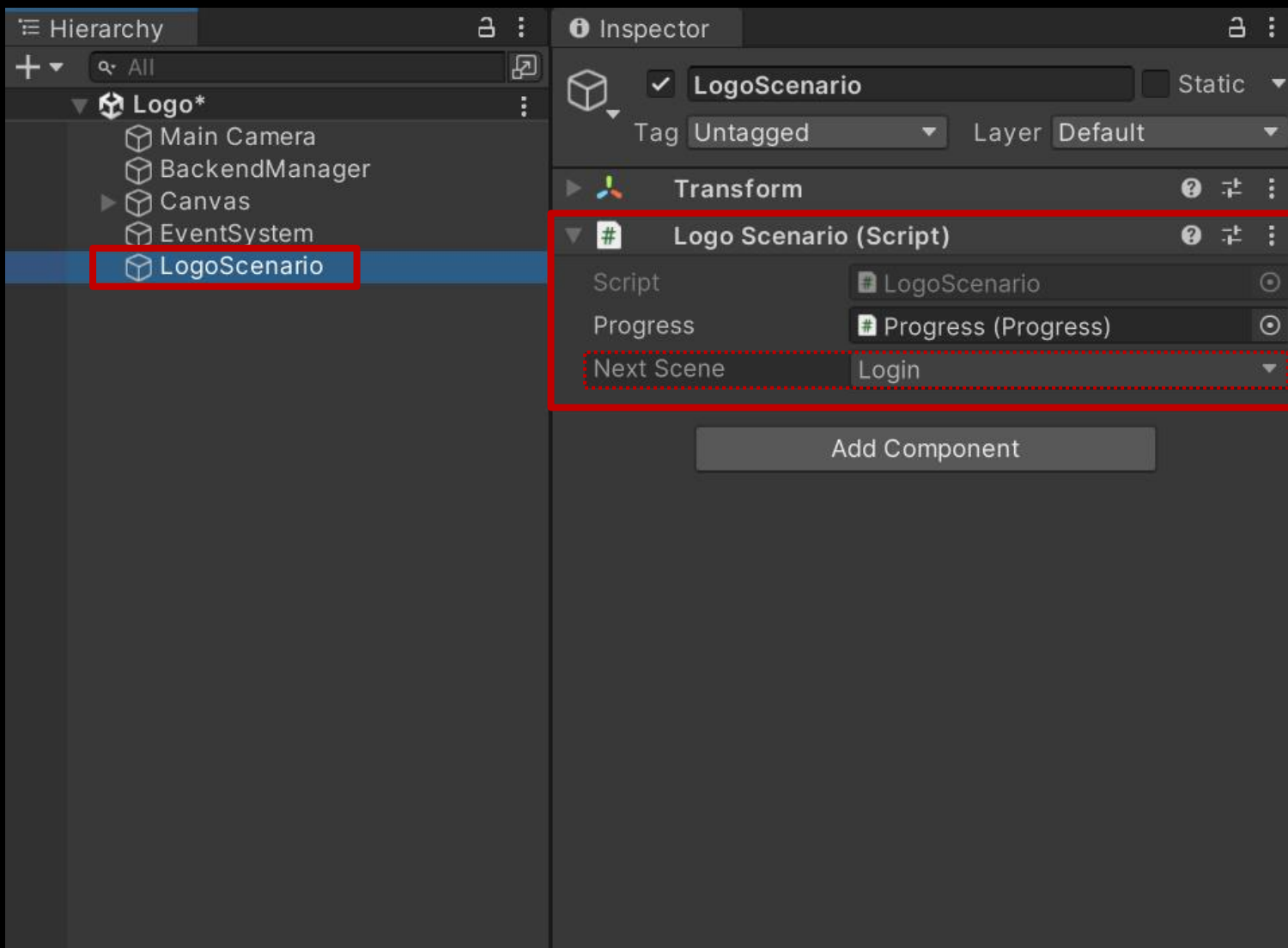
- 로딩이 완료되면 "Login" 씬으로 이동
  - LogoScenario Script 수정

```
1  using UnityEngine;
2
3  public class LogoScenario : MonoBehaviour
4  {
5      [SerializeField]
6      private Progress    progress;
7      [SerializeField]
8      private SceneNames nextScene;
9
10     private void Awake()...
14
15     private void SystemSetup()...
31
32     private void OnAfterProgress()
33     {
34         Utils.LoadScene(nextScene);
35     }
36 }
```



# 뒤끝 함수 호출 방식과 비동기 호출

- LogoScenario 오브젝트의 "LogoScenario" 컴포넌트 변수 설정

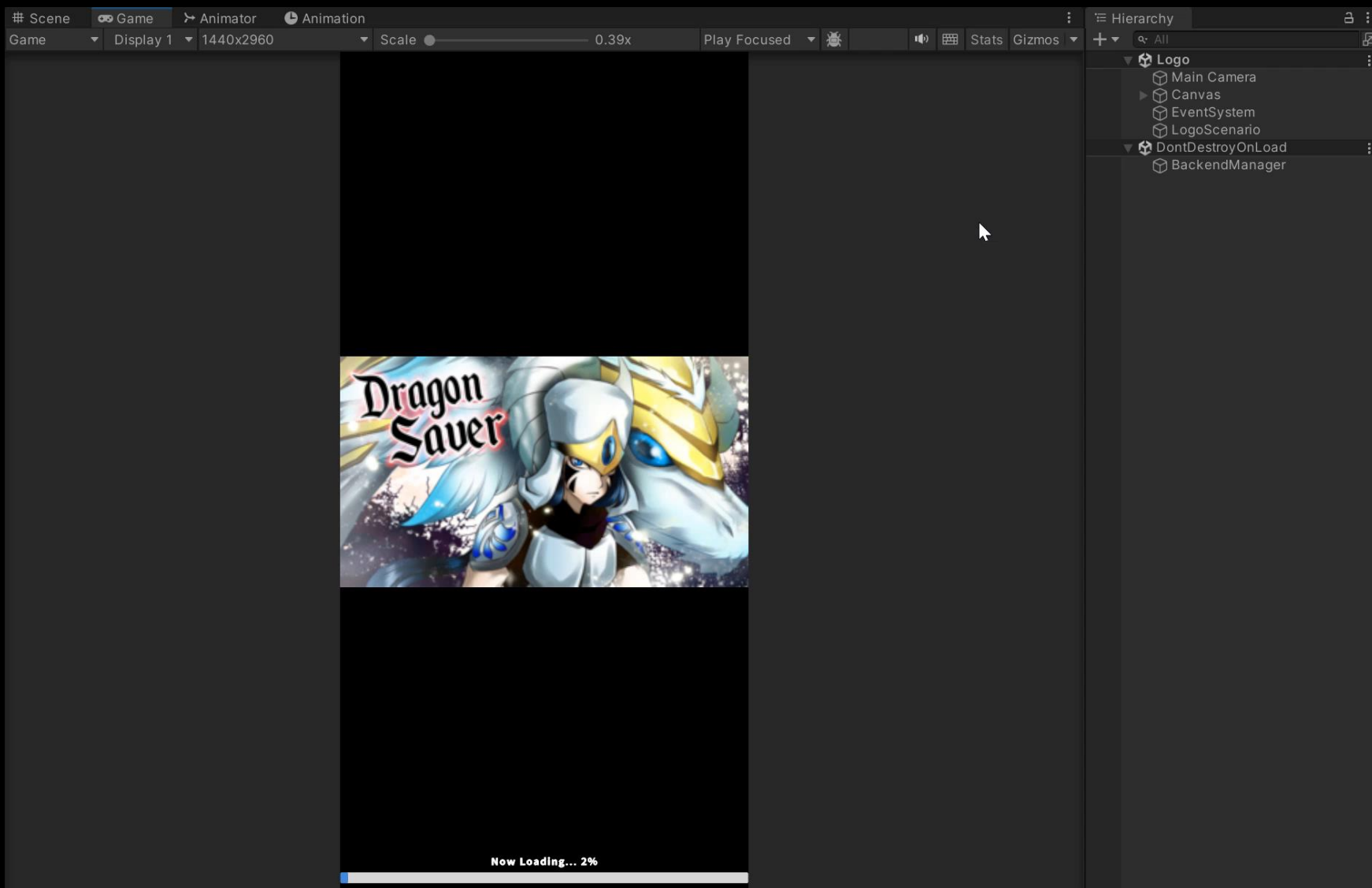






# 뒤끝 함수 호출 방식과 비동기 호출

## ■ 결과 화면



# 게임 유저 관리

- 개요
- 로그인
- 계정 생성, 이메일 설정
- 아이디 찾기
- 비밀번호 찾기
- 유저 정보 출력
- 닉네임 설정



# 게임 유저 관리

## ■ 개요

- 회원가입, 이메일 설정, 로그인, 아이디/비밀번호 찾기, 닉네임 설정 메소드

```
1 using UnityEngine;
2 using Backend;
3
4 public class LoginSample : MonoBehaviour
5 {
6     private void Awake()
7     {
8         string ID = "user01";
9         string PW = "1234";
10        string email = "user01@gmail.com";
11        string nickname = "첫번째유저";
12
13        /// 회원가입
14        Backend.BMember.CustomSignUp(ID, PW);
15
16        /// 이메일 설정
17        Backend.BMember.UpdateCustomEmail(email);
18
19        /// 로그인
20        Backend.BMember.CustomLogin(ID, PW);
21
22        /// 아이디 찾기
23        Backend.BMember.FindCustomID(email);
24
25        /// 비밀번호 찾기
26        Backend.BMember.ResetPassword(ID, email);
27
28        /// 닉네임 설정
29        // 닉네임이 없을 때 최초 닉네임 설정
30        Backend.BMember.CreateNickname(nickname);
31        // 이미 있는 닉네임을 수정 (만약 닉네임이 없으면 CreateNickname()이 호출된다.
32        Backend.BMember.UpdateNickname(nickname);
33    }
34 }
```

회원가입, 로그인 등과 같이 뒤끝 서버의 게임 유저를 관리하기 위해서는 Backend 클래스에 정의되어 있는 BackendMember 타입의 BMember 정적 변수에 접근해 설정할 수 있습니다.



# 게임 유저 관리

## 로그인

- 팝업 윈도우에 출력하는 UI들을 관리하는 Panel UI 생성 및 설정
  - GameObject - UI - Panel

The screenshot displays the Unity Inspector window for a selected UI Panel. The Hierarchy panel on the left shows the object structure: Login\* (Main Camera, Canvas, Panel, EventSystem). The Canvas component is selected, and the Inspector shows the following settings:

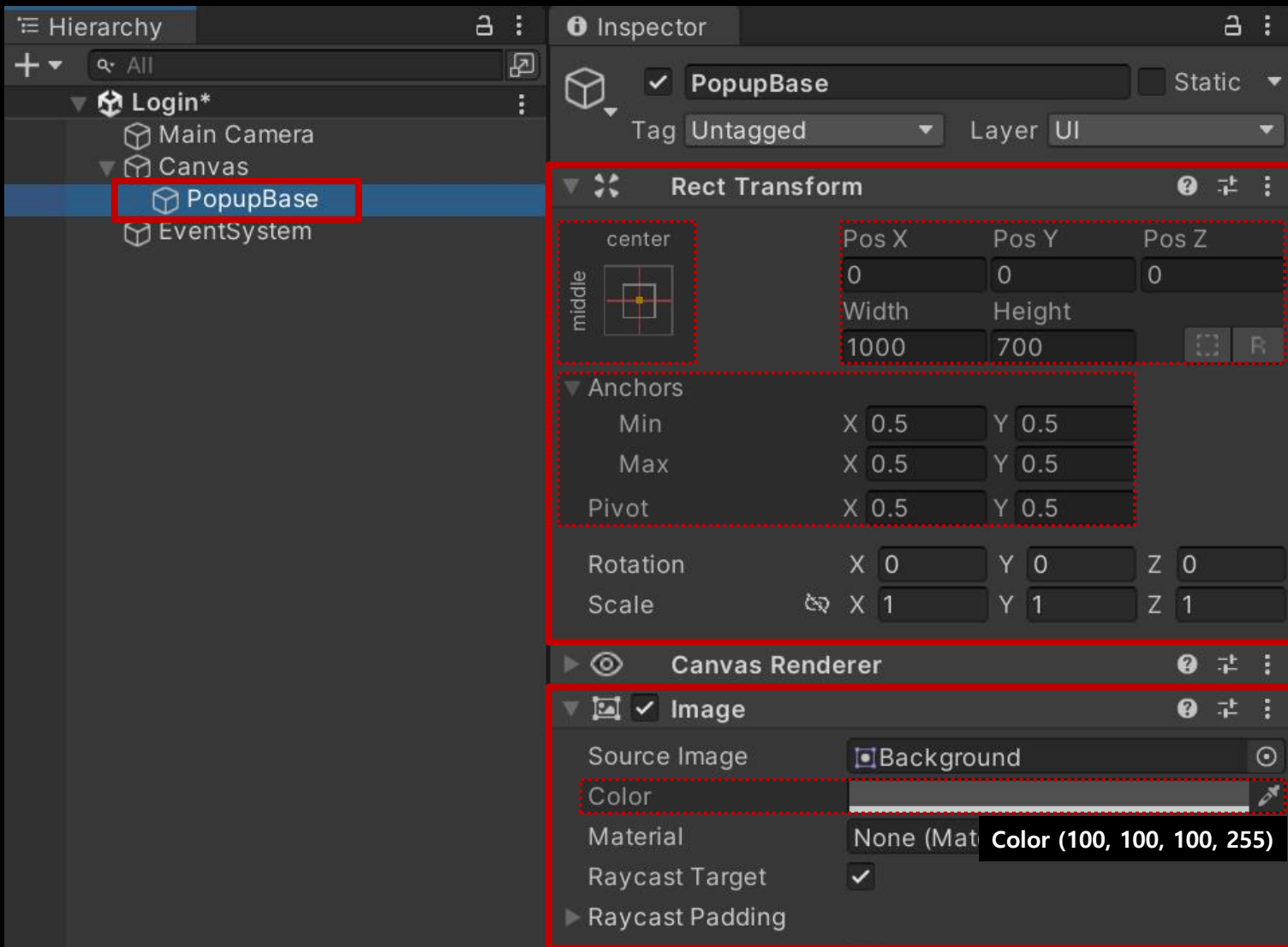
- Canvas**: Tag: Untagged, Layer: UI
- Rect Transform**
- Canvas**
- Canvas Scaler** (highlighted with a red dashed box):
  - UI Scale Mode: Scale With Screen Size
  - Reference Resolution X: 1440, Y: 2960
  - Screen Match Mode: Match Width Or Height
  - Match: Width (slider), Height (0.5)
  - Reference Pixels Per Unit: 100
- Graphic Raycaster**

An "Add Component" button is visible at the bottom of the Inspector.



# 게임 유저 관리

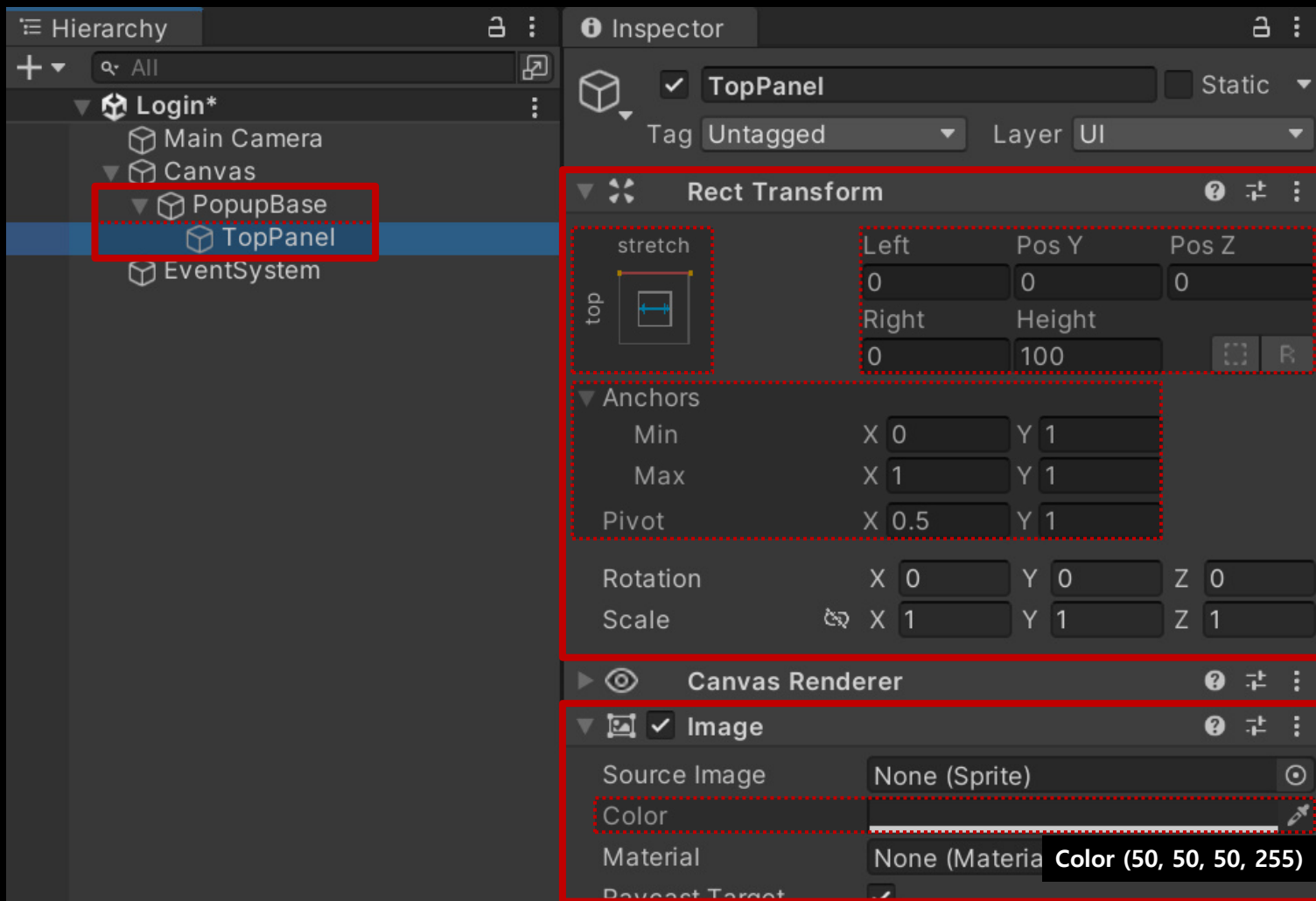
- 팝업 윈도우에 출력하는 UI들을 관리하는 Panel UI 생성 및 설정 (계속)





# 게임 유저 관리

- 윈도우 상단 배경을 출력하는 Image UI 생성 및 설정
  - GameObject - UI - Image





# 게임 유저 관리

- 윈도우 상단에 제목을 출력하는 "Text - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Text - TextMeshPro"

The screenshot shows the Unity interface. In the Hierarchy panel, the 'Title' object is selected under 'TopPanel'. The Inspector panel displays the 'Title' object's properties. The 'Rect Transform' component is highlighted with a red dashed box, showing the following settings:

stretch	Left	Pos Y	Pos Z
middle	100	0	0
	Right	Height	
	100	100	

The 'Anchors' component is also highlighted with a red dashed box, showing the following settings:

Min	X	Y
	0	0.5
Max	X	Y
	1	0.5
Pivot	X	Y
	0.5	0.5

The 'Canvas Renderer' component is also visible, showing the 'TextMeshPro - Text (UI)' component. The 'TextMeshPro - Text (UI)' component is highlighted with a red dashed box, showing the following settings:

Font Asset
F NotoSansKR-Bold SDF (TMP_Fc

The screenshot shows the TextMeshPro Inspector panel. The 'Main Settings' section is highlighted with a red dashed box, showing the following settings:

Font Asset	Material Preset	Font Style	Font Size
F NotoSansKR-Bold SDF (TMP_Fc	NotoSansKR-BoId SDF Material	B I U S ab AB SC	40

The 'Font Style' section is also highlighted with a red dashed box, showing the following options:

Font Style
B I U S ab AB SC

The 'Alignment' section is highlighted with a red dashed box, showing the following options:

Alignment
Left Center Right



# 게임 유저 관리

- 윈도우를 종료하는 "Button - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Button - TextMeshPro"

“X” 버튼에 텍스트는 필요 없기 때문에 삭제

**Hierarchy Panel:**

- Login\*
  - Main Camera
  - Canvas
    - PopupBase
      - TopPanel
        - Title
        - Exit
        - Text (TMP)

**Inspector Panel:**

- Exit (Static)
  - Tag: Untagged, Layer: UI
  - Rect Transform
    - right: Pos X: -10, Pos Y: 0, Pos Z: 0; Width: 80, Height: 80
    - middle: [Visual representation]
    - Anchors: Min (X: 1, Y: 0.5), Max (X: 1, Y: 0.5), Pivot (X: 1, Y: 0.5)
    - Rotation: X: 0, Y: 0, Z: 0
    - Scale: X: 1, Y: 1, Z: 1
  - Canvas Renderer
  - Image
    - Source Image: UI\_Button\_Exit
    - Color: [Color picker]
    - Material: None (Material)
    - Raycast Target: [checked]
    - Raycast Padding: [checked]
    - Maskable: [checked]
    - Image Type: Simple

**Project Panel:**

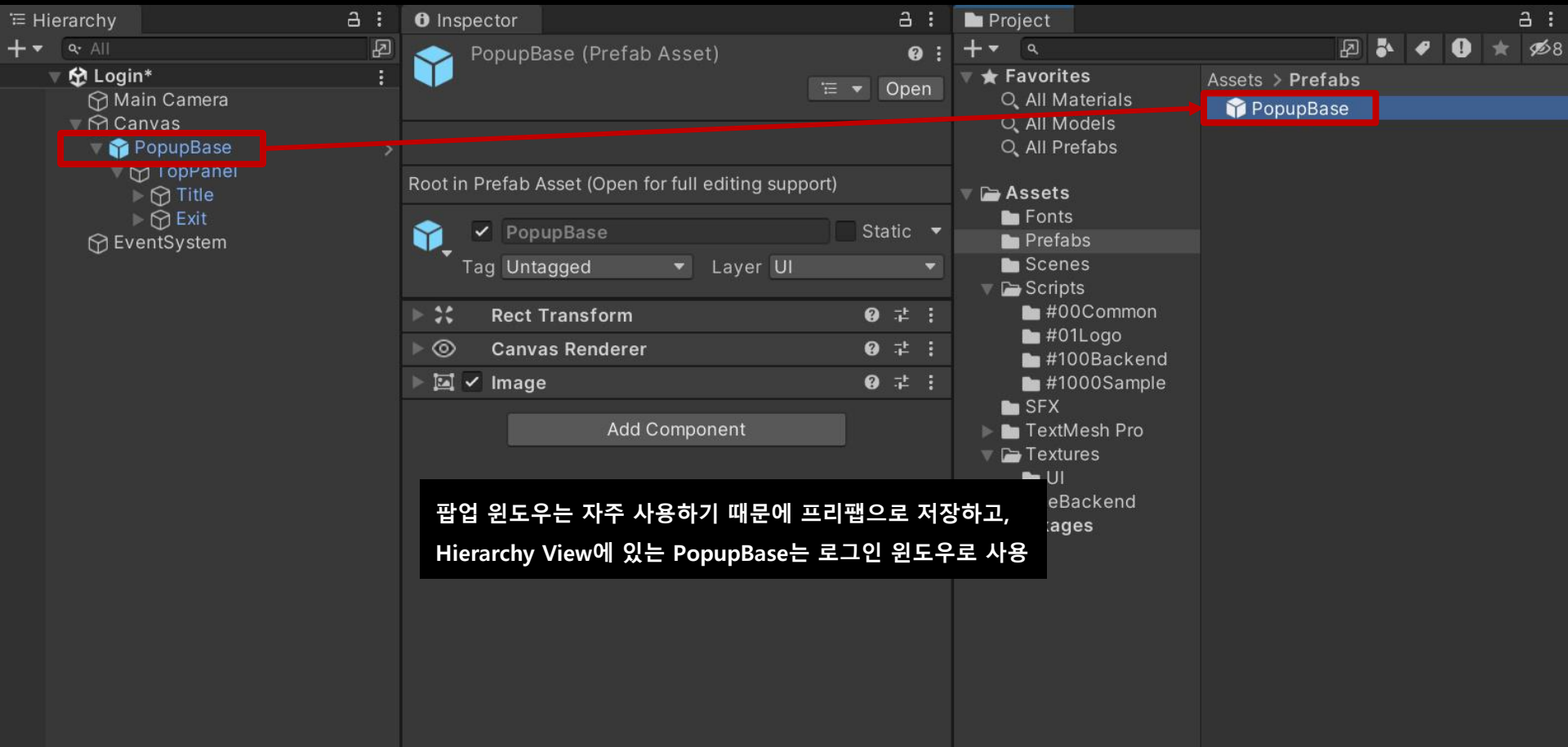
- Assets > Textures > UI
  - UI\_Button\_Base
  - UI\_Button\_Edit
  - UI\_Button\_Exit
  - UI\_Button\_Menu
  - UI\_Icon\_ProfileBackground
  - UI\_Octagon\_Empty
  - UI\_Octagon\_Full
  - UI\_RoundedRect\_Empty
  - UI\_RoundedRect\_Full





# 게임 유저 관리

- 팝업 윈도우 오브젝트 Prefab 생성
  - Hierarchy View의 "PopupBase" 오브젝트를 Project View로 Drag & Drop





# 게임 유저 관리

- 로그인 팝업 제작을 위한 오브젝트 설정

1. PopupBase 오브젝트의 이름을 "PopupLogin"으로 변경  
2. Title 오브젝트 "TextMeshPro - Text" 컴포넌트의 Text에 "로그인" 입력  
3. 로그인 윈도우는 닫을 수 없도록 Exit 오브젝트 삭제



# 게임 유저 관리

- 텍스트 입력을 위한 "Input Field - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Input Field - TextMeshPro"

The screenshot displays the Unity development environment with three panels:

- Hierarchy Panel:** Shows a tree structure under 'Login\*' with 'Main Camera', 'Canvas', 'PopupLogin', 'TopPanel', 'InputFieldBase', 'Text Area', 'Placeholder', and 'Text'. 'InputFieldBase' is selected and highlighted with a red box.
- Inspector Panel:** Shows the 'InputFieldBase' component with 'Tag: Untagged' and 'Layer: UI'. The 'Rect Transform' component is expanded and highlighted with a red dashed box, showing:
  - stretch: top
  - Left: 100, Pos Y: -170, Pos Z: 0
  - Right: 100, Height: 80
  - Anchors: Min (X: 0, Y: 1), Max (X: 1, Y: 1), Pivot (X: 0.5, Y: 1)
  - Rotation: X: 0, Y: 0, Z: 0
  - Scale: X: 1, Y: 1, Z: 1
- TextMeshPro - Input Field Panel:** Shows the 'TextMeshPro - Input Field' component with various settings:
  - Interactable: checked
  - Transition: Color Tint
  - Target Graphic: InputFieldBase (Image)
  - Normal Color, Highlighted Color, Pressed Color, Selected Color, Disabled Color: color pickers
  - Color Multiplier: 1
  - Fade Duration: 0.1
  - Navigation: Automatic
  - Text Viewport: Text Area (Rect Transform)
  - Text Component: Text (Text Mesh Pro UGUI)
  - Input Field Settings: Font Asset (NotoSansKR-Regular SDF), Point Size (40), Character Limit (0)

"TextMeshPro - Input Field" 컴포넌트에서 Font Asset, Point Size를 설정하면 Placeholder, Text 오브젝트의 설정이 바뀐다.



# 게임 유저 관리

- 텍스트 입력을 위한 "Input Field - TextMeshPro" UI 생성 및 설정 (계속)

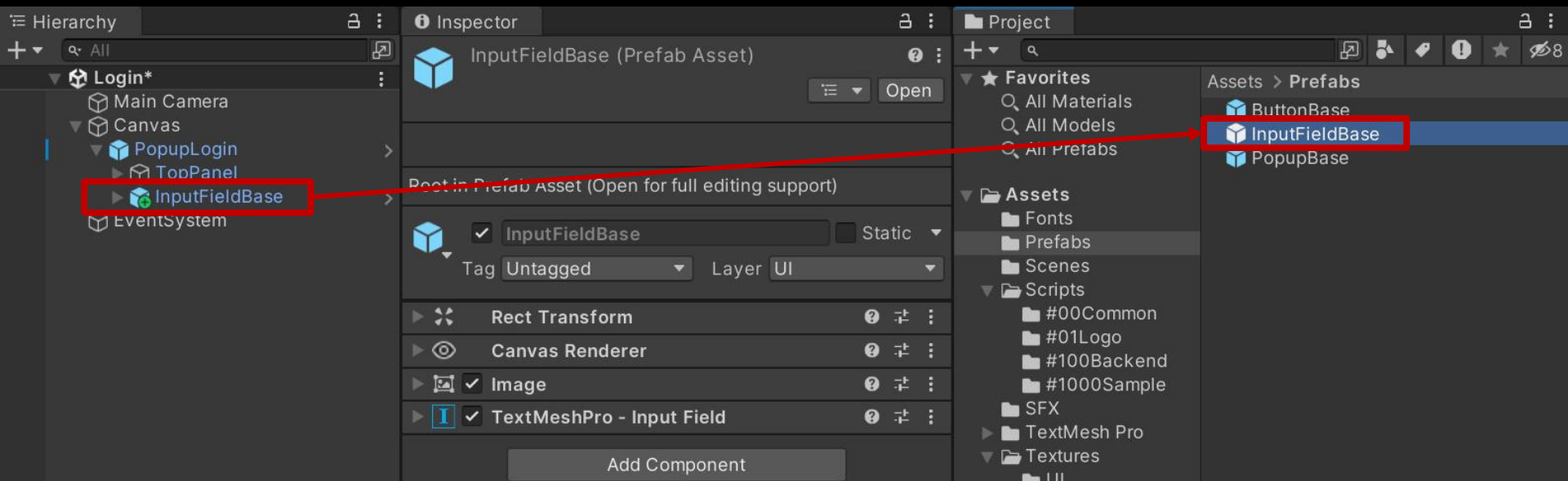
The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Placeholder' object under 'InputFieldBase' is selected. The Inspector panel shows the following settings for the 'TextMeshPro - Text (UI)' component:

- Placeholder** (checked), **Static** (unchecked)
- Tag: **Untagged**, Layer: **UI**
- Rect Transform** (expanded)
- Canvas Renderer** (expanded)
- TextMeshPro - Text (UI)** (expanded)
- Text Input** (with **Enable RTL Editor** checkbox)
- Enter text... (input field)
- Text Style: **Normal**
- Main Settings** section:
  - Font Asset: **NotoSansKR-Regular SDF (TMF)**
  - Material Preset: **NotoSansKR-Regular SDF Material**
  - Font Style: **B I U S ab AB SC** (with **I** selected)
  - Font Size: **40**
  - Auto Size: (unchecked)
- Vertex Color: (color picker)
- Color Gradient: (checkbox) **Color (50, 50, 50, 255)**



# 게임 유저 관리

- 텍스트 입력 오브젝트 Prefab 생성
  - Hierarchy View의 "InputFieldBase" 오브젝트를 Project View로 Drag & Drop



텍스트 입력은(ID, PW, E-mail 등) 자주 사용하기 때문에 프리팹으로 저장하고,  
Hierarchy View에 있는 InputFieldBase는 아이디 입력용으로 사용



# 게임 유저 관리

- 아이디 입력 필드 제작을 위한 오브젝트 설정

1. InputFieldBase 오브젝트의 이름을 "ID"로 변경  
2. Placeholder 오브젝트 "TextMeshPro - Text" 컴포넌트의 Text에 "아이디" 입력



# 게임 유저 관리

- 비밀번호 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot shows the Unity interface with three main panels:

- Hierarchy Panel:** Shows a scene hierarchy under 'Login\*'. A red box highlights the 'PW' object under 'Canvas' > 'PopupLogin' > 'ID'. A red dashed box indicates the 'PW' object is being dragged.
- Inspector Panel:** Shows the properties of the selected 'PW' object. The 'Name' is 'PW'. The 'Prefab' is 'InputFieldBase'. The 'Rect Transform' component is expanded, showing 'Left: 100', 'Pos Y: -270', 'Pos Z: 0', 'Right: 100', and 'Height: 80'. The 'Anchors' section shows 'Min' (X: 0, Y: 1) and 'Max' (X: 1, Y: 1). The 'Rotation' is (0, 0, 0) and 'Scale' is (1, 1, 1). Below this, the 'Image' component is visible, and the 'TextMeshPro - Input Field' component is selected.
- Hierarchy View Panel:** Shows the 'TextMeshPro - Input Field' component. The 'Content Type' is set to 'Password' (highlighted with a red dashed box). Other settings include 'Font Asset: NotoSansKR-Regular SDF', 'Point Size: 40', and 'Placeholder: Placeholder (Text Mesh Pro L)'. The 'Text' field is empty.

**InputFieldBase 오브젝트의 이름을 "PW"로 변경**



# 게임 유저 관리

## ■ 비밀번호 입력 필드 생성 및 설정 (계속)

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure with 'Login\*' as the root. Underneath, there is a 'Canvas' containing a 'PopupLogin' object. Inside 'PopupLogin', there are several sub-objects: 'TopPanel', 'ID', 'PW', 'Text Area', and 'Placeholder'. The 'Placeholder' object is currently selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected 'TextMeshPro - Text (UI)' component. The 'Text Input' field is visible, containing the Korean text '비밀번호'. Below this, the 'Main Settings' section is expanded, showing the following configurations:

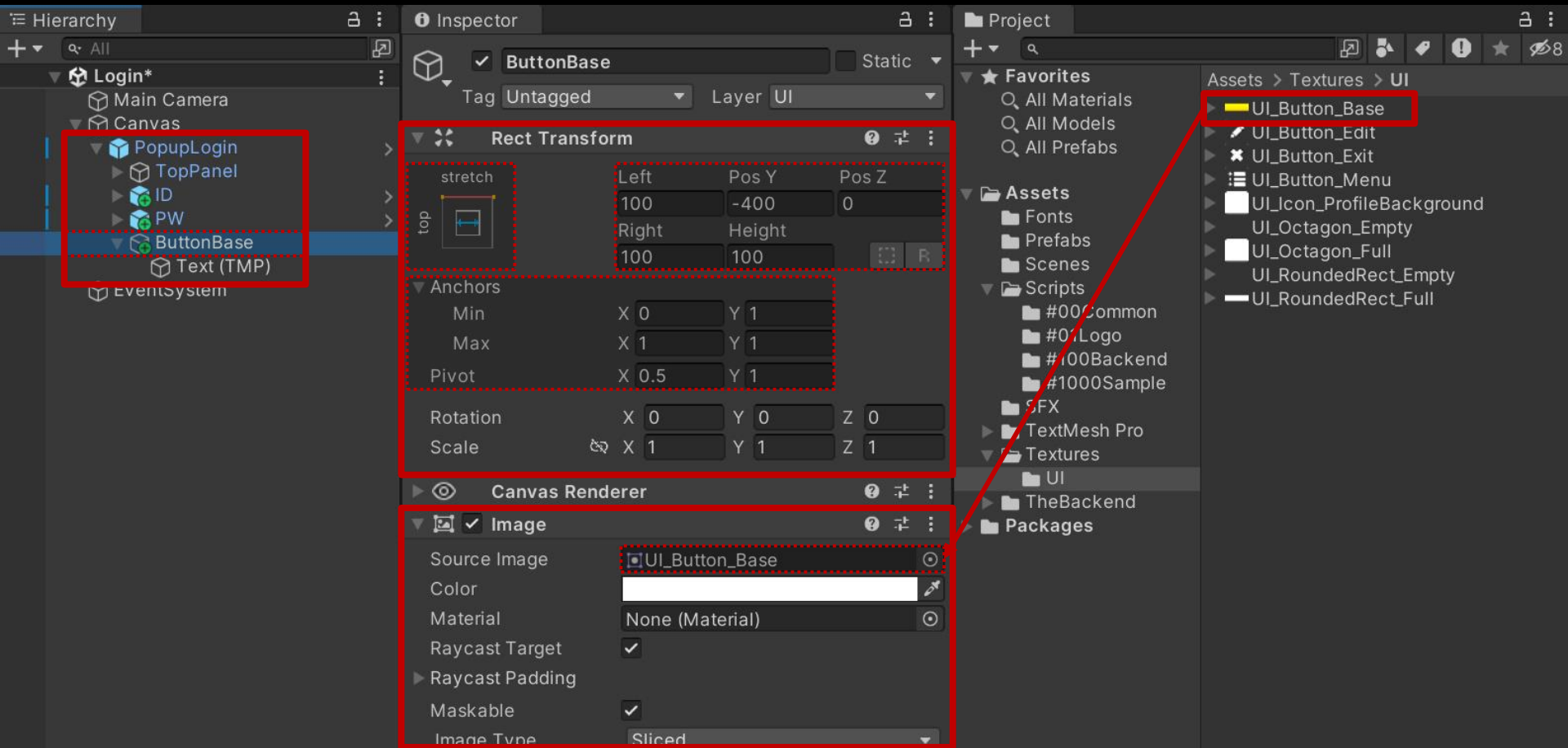
- Text Style: Normal
- Font Asset: NotoSansKR-Regular SDF (TMF)
- Material Preset: NotoSansKR-Regular SDF Material
- Font Style: B I U S ab AB SC (The 'I' style is currently selected)
- Font Size: 40
- Auto Size:
- Vertex Color:
- Color Gradient:





# 게임 유저 관리

- 클릭 가능한 버튼 "Button - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Button - TextMeshPro"





# 게임 유저 관리

- 클릭 가능한 버튼 "Button - TextMeshPro" UI 생성 및 설정 (계속)

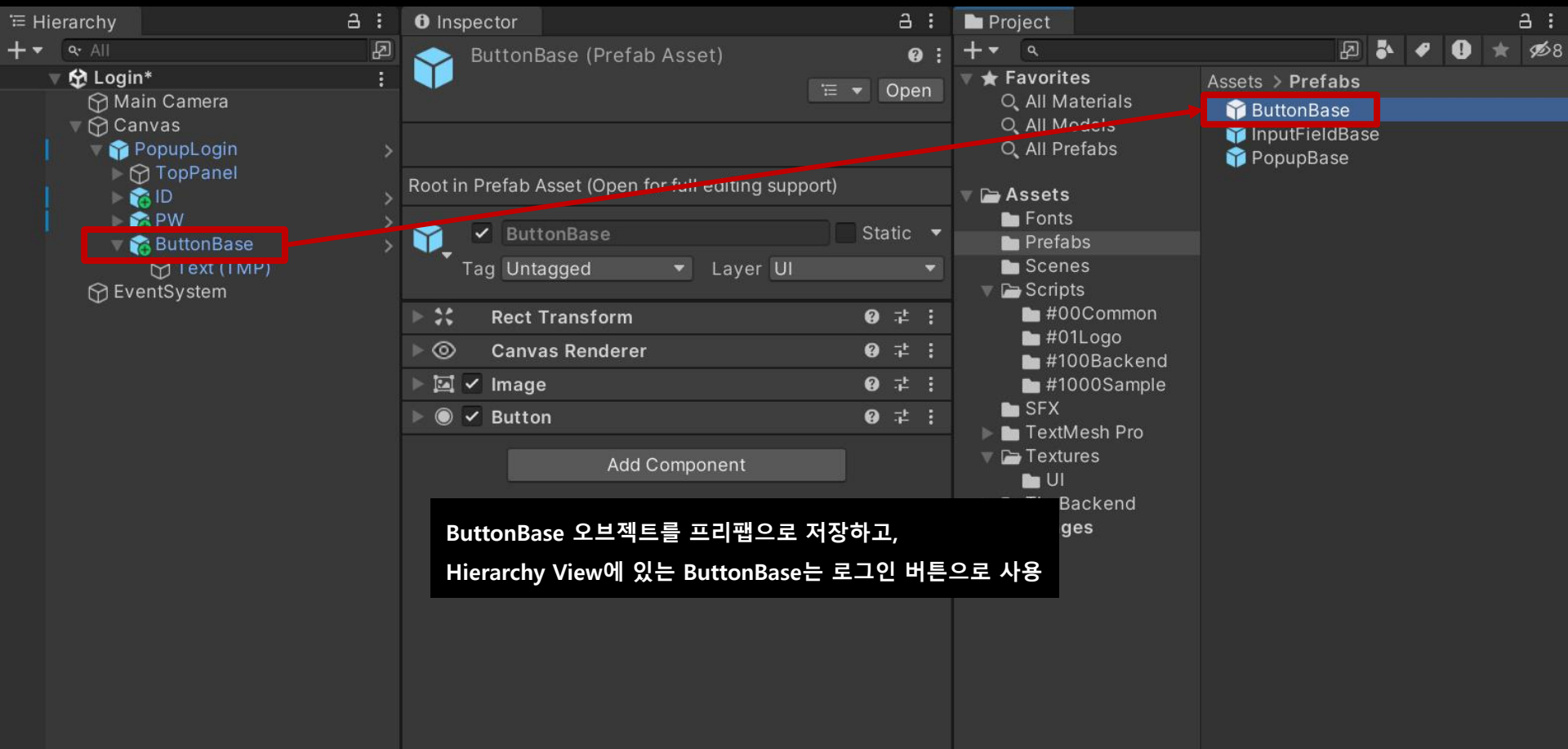
The image shows the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows a tree structure under 'Login\*' with 'Canvas' containing 'PopupLogin', 'TopPanel', 'ID', 'PW', 'ButtonBase', and 'Text (TMP)'. The 'Text (TMP)' object is highlighted with a red box. The Inspector panel on the right shows the properties for the selected 'Text (TMP)' object. The 'TextMeshPro - Text (UI)' component is selected, and its settings are displayed. The 'Text Input' field is empty. The 'Text Style' is set to 'Normal'. The 'Main Settings' section includes:

- Font Asset: NotoSansKR-Bold SDF (TMP\_Fc)
- Material Preset: NotoSansKR-Bold SDF Material
- Font Style: B (Bold) is selected
- Font Size: 40
- Auto Size: unchecked
- Vertex Color: Color (0, 0, 0, 255)



# 게임 유저 관리

- 버튼 오브젝트 Prefab 생성
  - Hierarchy View의 "ButtonBase" 오브젝트를 Project View로 Drag & Drop





# 게임 유저 관리

- “로그인” 버튼 제작을 위한 오브젝트 설정

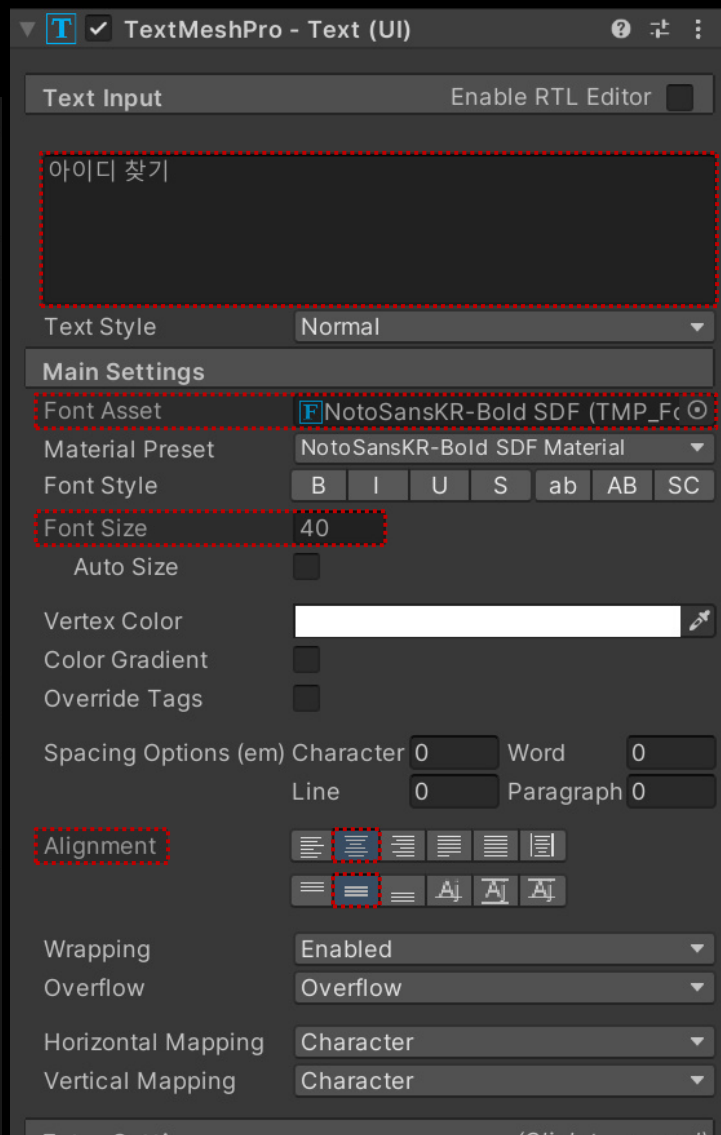
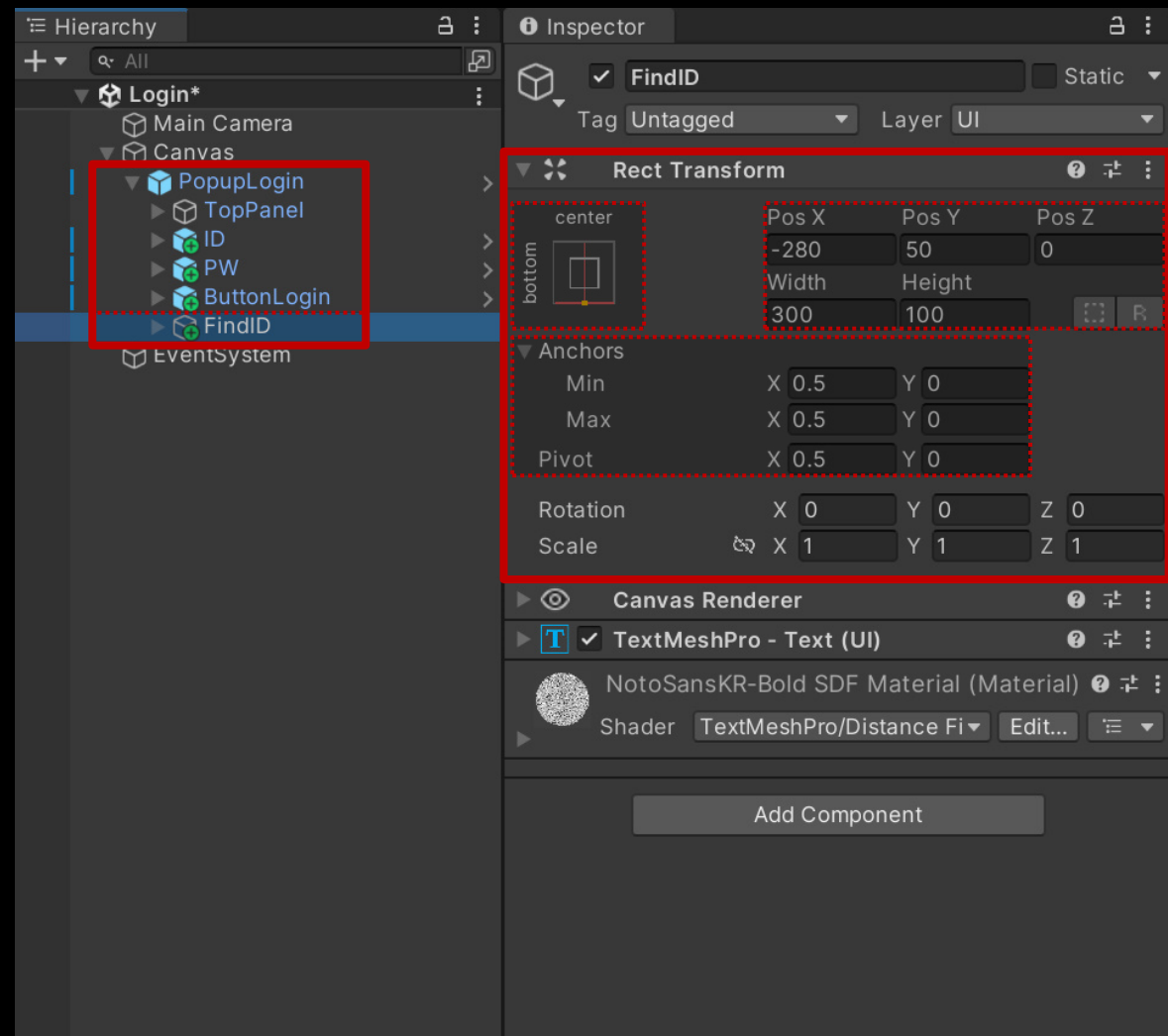
1. ButtonBase 오브젝트의 이름을 “ButtonLogin”으로 변경

2. Text (TMP) 오브젝트 “TextMeshPro - Text” 컴포넌트의 Text에 “로그인” 입력



# 게임 유저 관리

- “아이디 찾기” 텍스트를 출력하는 “Text - TextMeshPro” UI 생성 및 설정
  - GameObject - UI - “Text - TextMeshPro”





# 게임 유저 관리

- “비밀번호 찾기” 텍스트를 출력하는 “Text - TextMeshPro” UI 생성 및 설정
  - FindID 오브젝트를 Ctrl+D로 복제한 후 이름을 FindPW로 변경

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Login\*' folder is expanded, showing a sub-folder 'PopupLogin' which contains 'TopPanel', 'InputFieldBase', 'ID', 'PW', 'ButtonLogin', 'FindID', and 'FindPW'. The 'FindPW' object is selected. The Inspector panel shows the 'FindPW' object with the following properties:

- Rect Transform**
  - Pivot: center (bottom center)
  - Pos X: 260, Pos Y: 50, Pos Z: 0
  - Width: 300, Height: 100
  - Min: X 0.5, Y 0
  - Max: X 0.5, Y 0
  - Pivot: X 0.5, Y 0
  - Rotation: X 0, Y 0, Z 0
  - Scale: X 1, Y 1, Z 1
- Canvas Renderer**
- TextMeshPro - Text (UI)**
  - Text Input: 비밀번호 찾기
  - Enable RTL Editor:



# 게임 유저 관리

- “계정 생성” 텍스트를 출력하는 “Text - TextMeshPro” UI 생성 및 설정
  - FindID 오브젝트를 Ctrl+D로 복제한 후 이름을 RegisterAccount로 변경

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Login\*'. The 'RegisterAccount' object is selected and highlighted in blue. On the right, the Inspector panel shows the properties of the 'RegisterAccount' object. The 'Rect Transform' component is expanded, showing a pivot point at the bottom center and a bounding box with dimensions 300x100. The 'TextMeshPro - Text (UI)' component is also expanded, showing the text '계정 생성'.



# 게임 유저 관리

- 시스템 메시지를 출력하는 "Text - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Text - TextMeshPro"

The screenshot shows the Unity Inspector for a 'Message' object. The 'Rect Transform' component is highlighted with a red dashed box. The 'stretch' section shows the following values: Left: 0, Pos Y: -450, Pos Z: 0, Right: 0, Height: 100. The 'Anchors' section shows: Min: X 0, Y 0.5; Max: X 1, Y 0.5; Pivot: X 0.5, Y 0.5. Below this, the 'Canvas Renderer' and 'TextMeshPro - Text (UI)' components are visible.

The screenshot shows the Unity Inspector for the 'TextMeshPro - Text (UI)' component. The 'Text Input' field is empty. The 'Text Style' is set to 'Normal'. The 'Main Settings' section is highlighted with a red dashed box, showing 'Font Asset' as 'NotoSansKR-Bold SDF (TMP\_Fc...', 'Material Preset' as 'NotoSansKR-Bold SDF Material', 'Font Style' as 'B I U S ab AB SC', and 'Font Size' as '40'. The 'Alignment' section is also highlighted with a red dashed box, showing the 'Center' alignment icon selected. Other settings like 'Wrapping' (Enabled), 'Overflow' (Overflow), 'Horizontal Mapping' (Character), and 'Vertical Mapping' (Character) are visible.





# 게임 유저 관리

- 게임 유저 관리에서 사용하는 UI들을 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "LoginBase"로 변경

```
1  using UnityEngine;
2      using UnityEngine.UI;
3      using TMPro;
4
5  public class LoginBase : MonoBehaviour
6  {
7      [SerializeField]
8      private TextMeshProUGUI textMessage;
9
10     /// <summary>
11     /// 메세지 내용, InputField 색상 초기화
12     /// </summary>
13     protected void ResetUI(params Image[] images)
14     {
15         textMessage.text = string.Empty;
16
17         for ( int i = 0; i < images.Length; ++ i )
18         {
19             images[i].color = Color.white;
20         }
21     }
22 }
```



# 게임 유저 관리

- 게임 유저 관리에서 사용하는 UI들을 제어하는 스크립트 생성 및 작성 (계속)

```
23  /// <summary>
24  /// 매개변수에 있는 내용을 출력
25  /// </summary>
26  protected void SetMessage(string msg)
27  {
28      textMessage.text = msg;
29  }
30
31  /// <summary>
32  /// 입력 오류가 있는 InputField 색상 변경
33  /// 오류에 대한 메시지 출력
34  /// </summary>
35  protected void GuideForIncorrectlyEnteredData(Image image, string msg)
36  {
37      textMessage.text = msg;
38      image.color      = Color.red;
39  }
40
```



# 게임 유저 관리

- 게임 유저 관리에서 사용하는 UI들을 제어하는 스크립트 생성 및 작성 (계속)

```
41  /// <summary>
42  /// 필드 값이 비어있는지 확인 (image:필드 색상, field:내용, result:출력될 내용)
43  /// </summary>
44  protected bool IsFieldDataEmpty(Image image, string field, string result)
45  {
46      if ( field.Trim().Equals("") )
47      {
48          GuideForIncorrectlyEnteredData(image, $"{result}\ " 필드를 채워주세요.");
49
50          return true;
51      }
52
53      return false;
54  }
55 }
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 로그인을 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "Login"으로 변경

```
1 using System.Collections;
2     using UnityEngine;
3     using UnityEngine.UI;
4     using TMPro;
5     using BackEnd;
6
7 public class Login : LoginBase
8 {
9     [SerializeField]
10    private Image          imageID;          // ID 필드 색상 변경
11    [SerializeField]
12    private TMP_InputField inputFieldID;     // ID 필드 텍스트 정보 추출
13    [SerializeField]
14    private Image          imagePW;         // PW 필드 색상 변경
15    [SerializeField]
16    private TMP_InputField inputFieldPW;    // PW 필드 텍스트 정보 추출
17
18    [SerializeField]
19    private Button         btnLogin;        // 로그인 버튼 (상호작용 가능/불가능)
20
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 로그인을 제어하는 스크립트 생성 및 작성 (계속)

```
21  /// <summary>
22  /// "로그인" 버튼을 눌렀을 때 호출
23  /// </summary>
24  public void OnClickLogin()
25  {
26      // 매개변수로 입력한 InputField UI의 색상과 Message 내용 초기화
27      ResetUI(imageID, imagePW);
28
29      // 필드 값이 비어있는지 체크
30      if ( IsFieldDataEmpty(imageID, inputFieldID.text, "아이디") ) return;
31      if ( IsFieldDataEmpty(imagePW, inputFieldPW.text, "비밀번호") ) return;
32
33      // 로그인 버튼을 연타하지 못하도록 상호작용 비활성화
34      btnLogin.interactable = false;
35
36      // 서버에 로그인을 요청하는 동안 화면에 출력하는 내용 업데이트
37      // ex) 로그인 관련 텍스트 출력, 톱니바퀴 아이콘 회전 등
38      StartCoroutine(nameof(LoginProcess));
39
40      // 뒤끝 서버 로그인 시도
41      ResponseToLogin(inputFieldID.text, inputFieldPW.text);
42  }
43
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 로그인을 제어하는 스크립트 생성 및 작성 (계속)

```
44  /// <summary>
45  /// 로그인 시도 후 서버로부터 전달받은 message를 기반으로 로직 처리
46  /// </summary>
47  private void ResponseToLogin(string ID, string PW)
48  {
49      // 서버에 로그인 요청
50      Backend.BMember.CustomLogin(ID, PW, callback =>
51      {
52          StopCoroutine(nameof(LoginProcess));
53
54          // 로그인 성공
55          if ( callback.IsSuccess() )
56          {
57              SetMessage($"{inputFieldID.text}님 환영합니다.");
58          }
59          // 로그인 실패
60          else ...
93      });
94  }
95
```

뒷장



# 게임 유저 관리

- 뒤끝 서버와 연동해 로그인을 제어하는 스크립트 생성 및 작성 (계속)

```
59 // 로그인 실패
60 else
61 {
62     // 로그인에 실패했을 때는 다시 로그인을 해야하기 때문에 "로그인" 버튼 상호작용 활성화
63     btnLogin.interactable = true;
64
65     string message = string.Empty;
66
67     switch ( int.Parse(callback.GetStatusCode()) )
68     {
69         case 401: // 존재하지 않는 아이디, 잘못된 비밀번호
70             message = callback.GetMessage().Contains("customId") ? "존재하지 않는 아이디입니다." : "잘못된 비밀번호 입니다.";
71             break;
72         case 403: // 유저 or 디바이스 차단
73             message = callback.GetMessage().Contains("user") ? "차단당한 유저입니다." : "차단당한 디바이스입니다.";
74             break;
75         case 410: // 탈퇴 진행중
76             message = "탈퇴가 진행중인 유저입니다.";
77             break;
78         default:
79             message = callback.GetMessage();
80             break;
81     }
82
83     // StatusCode 401에서 "잘못된 비밀번호 입니다." 일 때
84     if ( message.Contains("비밀번호") )
85     {
86         GuideForIncorrectlyEnteredData(imagePW, message);
87     }
88     else
89     {
90         GuideForIncorrectlyEnteredData(imageID, message);
91     }
92 }
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 로그인을 제어하는 스크립트 생성 및 작성 (계속)

```
96     private IEnumerator LoginProcess()  
97     {  
98         float time = 0;  
99  
100        while ( true )  
101        {  
102            time += Time.deltaTime;  
103  
104            SendMessage($"로그인 중입니다... {time:F1}");  
105  
106            yield return null;  
107        }  
108    }  
109 }
```





# 게임 유저 관리

- PopupLogin 오브젝트에 "Login" 컴포넌트 추가 및 설정

The screenshot displays the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows the scene structure, with the **PopupLogin** object selected. The Inspector panel on the right shows the properties of the selected object. The **Login (Script)** component is expanded, and its fields are being configured. Red boxes and arrows highlight the configuration process:

- Message:** Set to **Message (Text Mesh Pro UGUI)**
- ID:** Set to **ID (Image)**
- Input Field ID:** Set to **ID (TMP\_Input Field)**
- Image PW:** Set to **PW (Image)**
- Input Field PW:** Set to **PW (TMP\_Input Field)**
- Btn Login:** Set to **ButtonLogin (Button)**

The **Message** component in the Hierarchy panel is also highlighted with a red box, and an arrow points to the **Message** field in the Inspector. The **Message (Text Mesh Pro UGUI)** component in the Inspector is highlighted with a red dotted box. The **ID (Image)**, **ID (TMP\_Input Field)**, **PW (Image)**, and **PW (TMP\_Input Field)** components are also highlighted with red dotted boxes. The **ButtonLogin (Button)** component is highlighted with a red dotted box.



# 게임 유저 관리

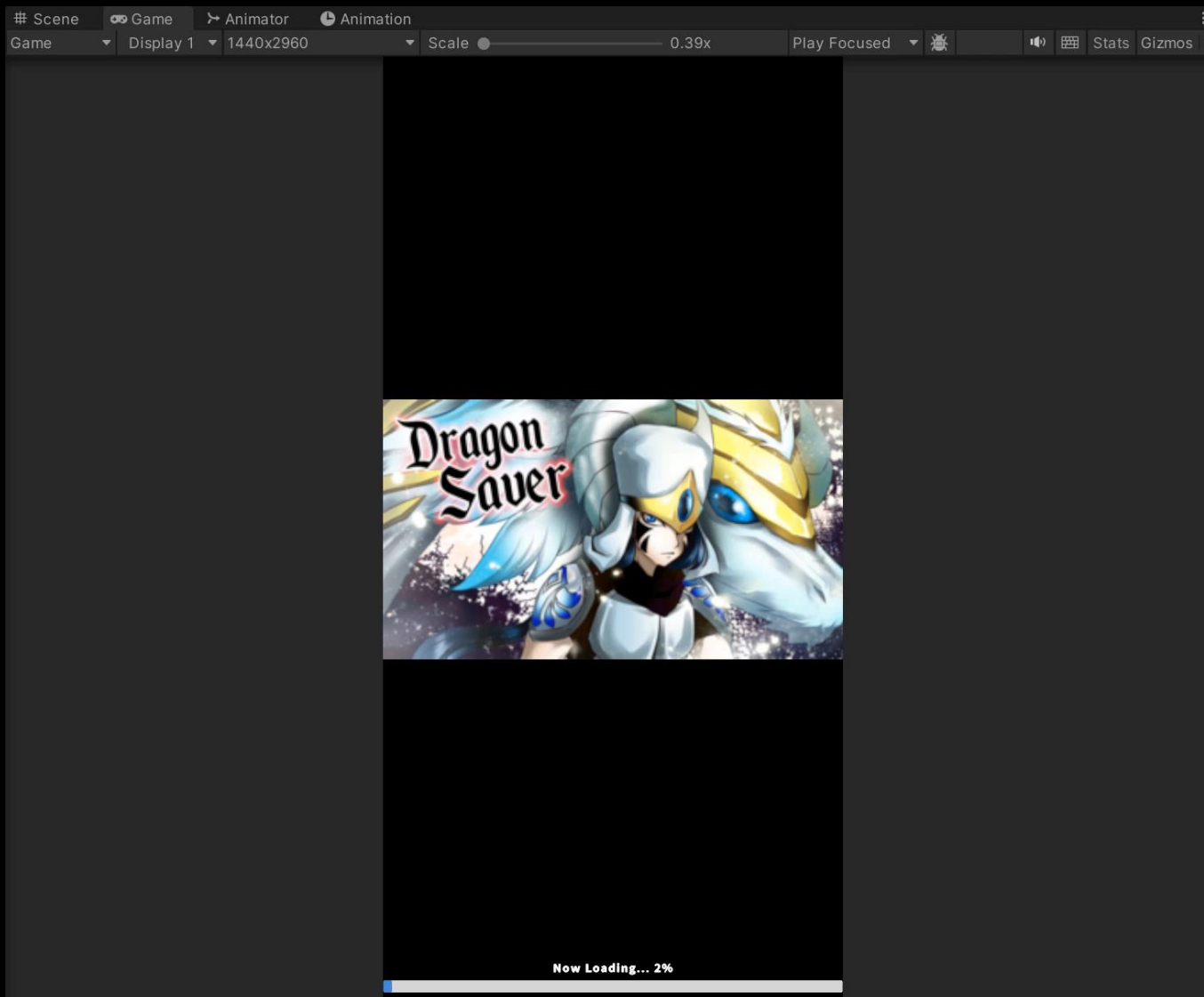
- ButtonLogin 오브젝트의 "Button" 컴포넌트 OnClick() 이벤트 설정

The screenshot displays the Unity Inspector interface. On the left, the Hierarchy panel shows a tree structure under 'Login' > 'Canvas' > 'PopupLogin', with 'ButtonLogin' selected and highlighted in blue. A red box highlights the 'ButtonLogin' object in the hierarchy, and a red arrow points from it to the 'Button' component in the Inspector. The Inspector shows the 'ButtonLogin' component with various properties. The 'Button' component is expanded, showing the 'On Click ()' event list. The event is set to 'Runtime Only' and 'Login.OnClickLogin'. A red dashed box highlights the 'Login.OnClickLogin' event name, and another red dashed box highlights the 'PopupLog' icon in the event list. The 'Button' component also shows 'Interactable' checked, 'Transition' set to 'Color Tint', and 'Target Graphic' set to 'ButtonLogin (Image)'. The 'Normal Color', 'Highlighted Color', 'Pressed Color', and 'Selected Color' are all set to white. The 'Disabled Color' is set to a light gray. The 'Color Multiplier' is set to 1, and the 'Fade Duration' is set to 0.1. The 'Navigation' is set to 'Automatic'. The 'Visualize' button is visible at the bottom of the 'Button' component.



# 게임 유저 관리

## ■ 결과 화면





# 게임 유저 관리

- Backend Console에서 계정 생성
  - "유저 관리" 탭 - "게임 유저 생성" 버튼

The screenshot displays the Backend Console interface for 'ProjectA'. The 'User Management' (유저 관리) tab is active, and the 'Game User Creation' (게임 유저 생성) button is highlighted. A modal dialog box is open, prompting for 'User ID' (유저 아이디\*) and 'Password' (비밀번호\*). The 'User ID' field contains 'test01', and the 'Password' field contains '....'. The 'Confirm' (확인) button is highlighted. The background shows a table of existing users with columns for 'No.' (번호), 'User ID' (유저 아이디), 'Nickname' (닉네임), 'Email' (이메일), 'Phone' (전화번호), 'Registration Date' (등록일), 'Last Login' (최근 로그인), 'OS' (OS), 'Status' (상태), and 'User ID' (유저 ID).

번호	유저 아이디	닉네임	이메일	전화번호	등록일	최근 로그인	OS	상태	유저 ID
1	test01							정상	UUID
2	user10						Windows 10 &#4010...	정상	UUID
3	user09						Windows 10 &#4010...	정상	UUID
4	user08						Windows 10 &#4010...	정상	UUID
5	user07						Windows 10 &#4010...	정상	UUID
6	user06						Windows 10 &#4010...	정상	UUID
7	user05	닉네임없다	-		2023.06.26 15:45	2023.08.05 00:13	Windows 10 &#4010...	정상	UUID
8	user04	user04	-		2023.06.26 15:45	2023.08.02 01:56	Windows 10 &#4010...	정상	UUID
9	user03	user03	-		2023.06.26 15:43	2023.08.02 01:56	Windows 10 &#4010...	정상	UUID
10	user02	닉네임	-		2023.06.11 22:13	2023.08.06 22:24	Windows 10 &#4010...	정상	UUID



# 게임 유저 관리

## Backend Console에서 계정 생성 (계속)

Backend Console << ProjectA [Settings] [7] 관리자

ProjectA [경고] >

★ 유저 관리 + 게임 유저 생성 삭제 SDK 5.11.0 이상 SDK 문서 콘솔 가이드

대상: 유저 UUID [ ]

상세 검색: [ ]

[검색] [검색 조건 초기화]

<input type="checkbox"/>	번호	유저 아이디	닉네임	국가	가입일	최근 접속일	최근 접속 OS	상태	유저 UUID
<input type="checkbox"/>	1	<a href="#">test01</a>	-	-	2023.08.13 21:28	2023.08.13 21:28	thebackend_console	정상	UUID [ ] [ ]
<input type="checkbox"/>	2	<a href="#">user10</a>	user10	-	2023.08.03 14:18	2023.08.06 22:45	Windows 10 &#4010...	정상	UUID [ ] [ ]
<input type="checkbox"/>	3	<a href="#">user09</a>	user09	-	2023.08.03 14:18	2023.08.06 22:27	Windows 10 &#4010...	정상	UUID [ ] [ ]
<input type="checkbox"/>	4	<a href="#">user08</a>	user08	-	2023.08.03 14:17	2023.08.06 22:27	Windows 10 &#4010...	정상	UUID [ ] [ ]
<input type="checkbox"/>	5	<a href="#">user07</a>	user07	-	2023.08.03 14:17	2023.08.03 14:17	Windows 10 &#4010...	정상	UUID [ ] [ ]
<input type="checkbox"/>	6	<a href="#">user06</a>	user06	-	2023.06.26 20:36	2023.08.03 14:16	Windows 10 &#4010...	정상	UUID [ ] [ ]
<input type="checkbox"/>	7	<a href="#">user05</a>	닉네임없다	-	2023.06.26 15:45	2023.08.05 00:13	Windows 10 &#4010...	정상	UUID [ ] [ ]
<input type="checkbox"/>	8	<a href="#">user04</a>	user04	-	2023.06.26 15:45	2023.08.02 01:56	Windows 10 &#4010...	정상	UUID [ ] [ ]
<input type="checkbox"/>	9	<a href="#">user03</a>	user03	-	2023.06.26 15:43	2023.08.02 01:56	Windows 10 &#4010...	정상	UUID [ ] [ ]
<input type="checkbox"/>	10	<a href="#">user02</a>	닉네임	-	2023.06.11 22:13	2023.08.06 22:24	Windows 10 &#4010...	정상	UUID [ ] [ ]

< 1 > 10개씩 보기

회사소개 이용약관 서비스수준협약 개인정보처리방침 © AFI, Inc. All rights reserved.



# 게임 유저 관리

## ■ 계정 생성, 이메일 설정

### ■ 계정 생성 팝업 윈도우 생성 및 설정

- PopupBase 프리팹을 Hierarchy View로 Drag & Drop

PopupBase 오브젝트의 이름을  
"PopupRegisterAccount"로 변경

Property	X	Y	Z
Pos X	0	0	0
Pos Y	0	0	0
Pos Z	0	0	0
Width	1000		
Height		800	
Min X	0.5		
Min Y		0.5	
Max X	0.5		
Max Y		0.5	
Pivot X	0.5		
Pivot Y		0.5	
Rotation X	0		
Rotation Y		0	
Rotation Z			0
Scale X	1		
Scale Y		1	
Scale Z			1



# 게임 유저 관리

- 계정 생성 팝업 윈도우 상단 이름 설정

The image shows the Unity development environment interface. On the left is the Hierarchy panel, and on the right is the Inspector panel.

**Hierarchy Panel:** Shows a tree view of the scene's objects. The 'Login\*' object is expanded, showing sub-objects: Main Camera, Canvas, PopupLogin, PopupRegisterAccount, TopPanel, Title (highlighted with a red box), Exit, Message, and EventSystem.

**Inspector Panel:** Shows the properties of the selected 'Title' object. The component 'TextMeshPro - Text (UI)' is active. The 'Text Input' field contains the Korean text '계정 생성'. Below this, the 'Main Settings' section is visible, including: Font Asset (NotoSansKR-Bold SDF (TMP\_Fc)), Material Preset (NotoSansKR-BoId SDF Material), Font Style (B, I, U, S, ab, AB, SC), Font Size (40), and Vertex Color (white).



# 게임 유저 관리

- 아이디 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot shows the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'ID' object is selected under the 'PopupRegisterAccount' object. The Inspector panel shows the 'Rect Transform' component with the following properties:

Property	Value
Left	100
Right	100
Pos Y	-170
Height	80
Pos Z	0
Min X	0
Min Y	1
Max X	1
Max Y	1
Pivot X	0.5
Pivot Y	1
Rotation X	0
Rotation Y	0
Rotation Z	0
Scale X	1
Scale Y	1
Scale Z	1

Below the Inspector panel, the following components are visible:

- Canvas Renderer
- Image
- TextMeshPro - Input Field

**InputFieldBase 오브젝트의 이름을 "ID"로 변경**





# 게임 유저 관리

- 아이디 입력 필드 생성 및 설정 (계속)

The image shows the Unity development environment. On the left, the Hierarchy panel displays a scene structure under 'Login\*'. A 'Placeholder' object is highlighted under the 'ID' folder. On the right, the Inspector panel shows the properties for the selected 'TextMeshPro - Text (UI)' component. The 'Text Input' field is set to '아이디' (ID). The 'Main Settings' section shows the font asset is 'NotoSansKR-Regular SDF (TMF)' and the font size is 40. The 'Placeholder' component is also visible in the Inspector above the text component, with 'Tag' set to 'Untagged' and 'Layer' set to 'UI'.



# 게임 유저 관리

- 비밀번호 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot shows the Unity interface with three main panels:

- Hierarchy Panel:** Shows a tree structure under 'Login\*' with 'Canvas' > 'PopupLogin' > 'PopupRegisterAccount' > 'TopPanel' > 'ID' > 'PW' selected. A red box highlights the 'PW' object and its children: 'Text Area', 'Placeholder', and 'Text'.
- Inspector Panel:** Shows the 'PW' object selected. The 'Rect Transform' component is highlighted with a red dashed box. Its properties are: stretch (top), Left: 100, Pos Y: -270, Pos Z: 0, Right: 100, Height: 80. The 'Anchors' component is also highlighted with a red dashed box, showing Min (X: 0, Y: 1), Max (X: 1, Y: 1), and Pivot (X: 0.5, Y: 1). Below it, the 'TextMeshPro - Input Field' component is visible.
- Hierarchy Panel (Right):** Shows the 'TextMeshPro - Input Field' component. The 'Content Type' dropdown is highlighted with a red dashed box and set to 'Password'.

**InputFieldBase 오브젝트의 이름을 "PW"로 변경**



# 게임 유저 관리

## ■ 비밀번호 입력 필드 생성 및 설정 (계속)

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a scene structure under 'Login\*'. A 'Placeholder' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected 'TextMeshPro - Text (UI)' component. The 'Text Input' field is visible, containing the Korean text '비밀번호' (Password). The 'Main Settings' section is expanded, showing the following configurations:

- Font Asset: NotoSansKR-Regular SDF (TMF)
- Material Preset: NotoSansKR-Regular SDF Material
- Font Style: I (Italic)
- Font Size: 40
- Auto Size: Unchecked

The 'Text Input' field and the 'Main Settings' section are also highlighted with a red box.



# 게임 유저 관리

- 비밀번호 확인 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

**InputFieldBase 오브젝트의 이름을 "ConfirmPW"로 변경**

Property	Value
Left	100
Pos Y	-370
Pos Z	0
Right	100
Height	80
Min X	0
Min Y	1
Max X	1
Max Y	1
Pivot X	0.5
Pivot Y	1
Rotation X	0
Rotation Y	0
Rotation Z	0
Scale X	1
Scale Y	1
Scale Z	1

Property	Value
Font Asset	F NotoSansKR-Regular SDF (T1)
Point Size	40
Character Limit	0
Content Type	Password
Placeholder	T Placeholder (Text Mesh Pro L)



# 게임 유저 관리

- 비밀번호 확인 입력 필드 생성 및 설정 (계속)

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Login\*'. The 'ConfirmPW' folder is expanded, and the 'Placeholder' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected 'TextMeshPro - Text (UI)' component. The 'Text Input' field is visible, containing the text '비밀번호 확인'. Below this, the 'Text Style' is set to 'Normal'. The 'Main Settings' section includes 'Font Asset' set to 'NotoSansKR-Regular SDF (TMF)', 'Material Preset' set to 'NotoSansKR-Regular SDF Material', 'Font Style' with 'B', 'I', 'U', 'S', 'ab', 'AB', 'SC' buttons, and 'Font Size' set to '40'. The 'Vertex Color' and 'Color Gradient' options are also visible at the bottom.



# 게임 유저 관리

- 메일 주소 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot displays the Unity development environment with three main panels:

- Hierarchy Panel:** Shows a tree structure of objects. The 'MailAddress' object is selected and highlighted in blue. A red dashed box highlights the 'MailAddress' object and its children: 'Text Area', 'Placeholder', and 'Text'.
- Inspector Panel:** Shows the properties of the selected 'MailAddress' object. The 'Rect Transform' component is expanded, showing its position and size. A red dashed box highlights the 'Rect Transform' component and its 'Anchors' and 'Pivot' properties.
- Hierarchy Panel:** Shows the 'TextMeshPro - Input Field' component. A red dashed box highlights the 'Content Type' property, which is set to 'Email Address'.

**InputFieldBase 오브젝트의 이름을 "MailAddress"로 변경**



# 게임 유저 관리

## ■ 메일 주소 입력 필드 생성 및 설정 (계속)

The image shows the Unity development environment. On the left is the Hierarchy panel, and on the right is the Inspector panel.

**Hierarchy Panel:**

- Login\*
  - Main Camera
  - Canvas
    - PopupLogin
    - PopupRegisterAccount
      - TopPanel
      - ID
      - PW
      - ConfirmPW
      - MailAddress
        - Text Area
          - Placeholder** (Selected)
          - Text

- Message
- EventSystem

**Inspector Panel:**

- Placeholder** (Static) | Tag: Untagged | Layer: UI
- Rect Transform
- Canvas Renderer
- TextMeshPro - Text (UI)**
  - Text Input | Enable RTL Editor:
  - 메일주소
  - Text Style: Normal
  - Main Settings
    - Font Asset: NotoSansKR-Regular SDF (TMF)
    - Material Preset: NotoSansKR-Regular SDF Material
    - Font Style: B I U S ab AB SC
    - Font Size: 40
    - Auto Size:
    - Vertex Color:
    - Color Gradient:



# 게임 유저 관리

- “계정 생성” 버튼 생성 및 설정
  - ButtonBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot shows the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'ButtonRegisterAccount' object is selected under the 'PopupRegisterAccount' hierarchy. In the Inspector panel, the 'Rect Transform' component is highlighted, showing its position and size settings. The 'Anchors' section is also visible, showing the object's position relative to the canvas.

**ButtonBase** 오브젝트의 이름을  
“ButtonRegisterAccount”로 변경

Property	Value
Left	100
Right	100
Pos Y	-600
Height	100
Pos Z	0
Min X	0
Min Y	1
Max X	1
Max Y	1
Pivot X	0.5
Pivot Y	1
Rotation X	0
Rotation Y	0
Rotation Z	0
Scale X	1
Scale Y	1
Scale Z	1





# 게임 유저 관리

- “계정 생성” 버튼 생성 및 설정 (계속)

The image shows the Unity development environment. On the left is the Hierarchy panel, and on the right is the Inspector panel.

**Hierarchy Panel:** Shows a tree structure of game objects. The selected object is **Text (TMP)**, which is a child of **ButtonRegisterAccount**. Other objects include **Login\***, **Main Camera**, **Canvas**, **PopupLogin**, **PopupRegisterAccount**, **TopPanel**, **ID**, **PW**, **ConfirmPW**, **MailAddress**, **Message**, and **EventSystem**.

**Inspector Panel:** Shows the properties of the selected **Text (TMP)** object. The **TextMeshPro - Text (UI)** component is active. The text input field contains the Korean text **계정 생성** (Account Creation). The font asset is set to **NotoSansKR-Bold SDF (TMP\_Fc)** and the font size is **40**.



# 게임 유저 관리

- 뒤끝 서버와 연동해 계정 생성을 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "RegisterAccount"로 변경

```
1 using UnityEngine;
2     using UnityEngine.UI;
3     using TMPro;
4     using BackEnd;
5
6 public class RegisterAccount : LoginBase
7 {
8     [SerializeField]
9     private Image          imageID;          // ID 필드 색상 변경
10    [SerializeField]
11    private TMP_InputField  inputFieldID;    // ID 필드 텍스트 정보 추출
12    [SerializeField]
13    private Image          imagePW;          // PW 필드 색상 변경
14    [SerializeField]
15    private TMP_InputField  inputFieldPW;    // PW 필드 텍스트 정보 추출
16    [SerializeField]
17    private Image          imageConfirmPW;   // Confirm PW 필드 색상 변경
18    [SerializeField]
19    private TMP_InputField  inputFieldConfirmPW; // Confirm PW 필드 텍스트 정보 추출
20    [SerializeField]
21    private Image          imageEmail;       // E-mail 필드 색상 변경
22    [SerializeField]
23    private TMP_InputField  inputFieldEmail; // E-mail 필드 텍스트 정보 추출
24
25    [SerializeField]
26    private Button          btnRegisterAccount; // "계정 생성" 버튼 (상호작용 가능/불가능)
27
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 계정 생성을 제어하는 스크립트 생성 및 작성 (계속)

```
28  /// <summary>
29  /// "계정 생성" 버튼을 눌렀을 때 호출
30  /// </summary>
31  public void OnClickRegisterAccount()
32  {
33      // 매개변수로 입력한 InputField UI의 색상과 Message 내용 초기화
34      ResetUI(imageID, imagePW, imageConfirmPW, imageEmail);
35
36      // 필드 값이 비어있는지 체크
37      if ( IsFieldDataEmpty(imageID, inputFieldID.text, "아이디" )           return;
38      if ( IsFieldDataEmpty(imagePW, inputFieldPW.text, "비밀번호" )         return;
39      if ( IsFieldDataEmpty(imageConfirmPW, inputFieldConfirmPW.text, "비밀번호 확인" ) return;
40      if ( IsFieldDataEmpty(imageEmail, inputFieldEmail.text, "메일 주소" )  return;
41
42      // 비밀번호와 비밀번호 확인의 내용이 다를 때
43      if ( !inputFieldPW.text.Equals(inputFieldConfirmPW.text) )
44      {
45          GuideForIncorrectlyEnteredData(imageConfirmPW, "비밀번호가 일치하지 않습니다.");
46          return;
47      }
48
49      // 메일 형식 검사
50      if ( !inputFieldEmail.text.Contains("@" ) )
51      {
52          GuideForIncorrectlyEnteredData(imageEmail, "메일 형식이 잘못되었습니다.(ex. address@xx.xx)");
53          return;
54      }
55
56      // 계정 생성 버튼의 상호작용 비활성화
57      btnRegisterAccount.interactable = false;
58      SetMessage("계정 생성중입니다.");
59
60      // 뒤끝 서버 계정 생성 시도
61      CustomSignUp();
62  }
63
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 계정 생성을 제어하는 스크립트 생성 및 작성 (계속)

```
64  // <summary>
65  // 계정 생성 시도 후 서버로부터 전달받은 message를 기반으로 로직 처리
66  // </summary>
67  private void CustomSignUp()
68  {
69      Backend.BMember.CustomSignUp(inputFieldID.text, inputFieldPW.text, callback =>
70      {
71          // "계정 생성" 버튼 상호작용 활성화
72          btnRegisterAccount.interactable = true;
73
74          // 계정 생성 성공
75          if ( callback.IsSuccess() )
76          {
77              // E-mail 정보 업데이트
78              Backend.BMember.UpdateCustomEmail(inputFieldEmail.text, callback =>
79              {
80                  if ( callback.IsSuccess() )
81                  {
82                      SetMessage($"계정 생성 성공. {inputFieldID.text}님 환영합니다.");
83                  }
84              });
85          }
86          // 계정 생성 실패
87          else...
115  });
116  }
117  }
```

뒷장



# 게임 유저 관리

- 뒤끝 서버와 연동해 계정 생성을 제어하는 스크립트 생성 및 작성 (계속)

```
86 // 계정 생성 실패
87 else
88 {
89     string message = string.Empty;
90
91     switch ( int.Parse(callback.GetStatusCode()) )
92     {
93         case 409: // 중복된 customId 가 존재하는 경우
94             message = "이미 존재하는 아이디입니다.";
95             break;
96         case 403: // 차단당한 디바이스일 경우
97             message = callback.GetMessage();
98             break;
99         case 401: // 프로젝트 상태가 '점검'일 경우
100        case 400: // 디바이스 정보가 null일 경우
101        default:
102            message = callback.GetMessage();
103            break;
104        }
105
106        if ( message.Contains("아이디") )
107        {
108            GuideForIncorrectlyEnteredData(imageID, message);
109        }
110        else
111        {
112            SetMessage(message);
113        }
114    }
```



# 게임 유저 관리

- PopupRegisterAccount 오브젝트에 "RegisterAccount" 컴포넌트 추가 및 설정

The screenshot displays the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows the scene structure, with 'PopupRegisterAccount' selected and its sub-objects (ID, PW, ConfirmPW, MailAddress, ButtonRegisterAccount, Message, EventSystem) listed. The Inspector panel on the right shows the properties of the selected object, including 'Register Account (Script)'. The 'Register Account (Script)' component is expanded, showing a list of fields with their corresponding values and types. Red boxes highlight the 'Register Account (Script)' component and its fields, and red arrows point from the Hierarchy panel to the Inspector panel, indicating the mapping of the sub-objects to the script fields.

Field Name	Value	Type
Script	RegisterAccount	Script
Text Message	Message (Text Mesh Pro UGUI)	Text Mesh Pro UGUI
Image ID	ID (Image)	Image
Input Field ID	ID (TMP_Input Field)	Input Field
Image PW	PW (Image)	Image
Input Field PW	PW (TMP_Input Field)	Input Field
Image Confirm PW	ConfirmPW (Image)	Image
Input Field Confirm PW	ConfirmPW (TMP_Input Field)	Input Field
Image Email	MailAddress (Image)	Image
Input Field Email	MailAddress (TMP_Input Field)	Input Field
Btn Register Account	ButtonRegisterAccount (Button)	Button



# 게임 유저 관리

- ButtonRegisterAccount 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 설정

The screenshot displays the Unity Inspector for the **ButtonRegisterAccount** object. The Hierarchy panel on the left shows the object structure, with **ButtonRegisterAccount** selected. The Inspector panel on the right shows the **Button** component settings. The **On Click ()** event list is visible, showing a single event with the following configuration:

- Event Type: **Runtime Only**
- Event Name: **RegisterAccount.OnClickRegister**
- Target: **PopupReg**



# 게임 유저 관리

- PopupRegisterAccount 오브젝트 비활성화

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Login\*'. The 'PopupRegisterAccount' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected object. The 'PopupRegisterAccount' component is highlighted with a red box, and its 'Active' checkbox is unchecked, indicating that the object is disabled. Other components listed include 'Rect Transform', 'Canvas Renderer', 'Image', and 'Register Account (Script)'. The 'Add Component' button is visible at the bottom of the Inspector panel.





# 게임 유저 관리

- Text UI 상호작용을 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "UITextInteraction"으로 변경

```
1 using UnityEngine;
2 using UnityEngine.EventSystems;
3 using UnityEngine.Events;
4 using TMPro;
5
6 public class UITextInteraction : MonoBehaviour, IPointerEnterHandler, IPointerExitHandler, IPointerClickHandler
7 {
8     [System.Serializable]
9     private class OnClickEvent : UnityEvent { }
10
11     // Text UI를 클릭했을 때 호출하고 싶은 메소드 등록
12     [SerializeField]
13     private OnClickEvent    onClickEvent;
14
15     // 색상이 바뀌고, 터치가 되는 TextMeshProGUI
16     private TextMeshProUGUI text;
17
```



# 게임 유저 관리

- Text UI 상호작용을 제어하는 스크립트 생성 및 작성 (계속)

```
18 private void Awake()  
19 {  
20     text = GetComponent<TextMeshProUGUI>();  
21 }  
22  
23 public void OnPointerEnter(PointerEventData eventData)  
24 {  
25     text.fontStyle = FontStyle.Bold;  
26 }  
27  
28 public void OnPointerExit(PointerEventData eventData)  
29 {  
30     text.fontStyle = FontStyle.Normal;  
31 }  
32  
33 public void OnPointerClick(PointerEventData eventData)  
34 {  
35     onClickEvent?.Invoke();  
36 }  
37 }
```



# 게임 유저 관리

- RegisterAccount 오브젝트에 "UITextInteraction" 컴포넌트 추가 및 설정

로그인 팝업 윈도우에서 "계정 생성" 텍스트를 클릭하면  
로그인 팝업은 비활성화되고, 계정 생성 팝업이 활성화된다.



# 게임 유저 관리

- Exit 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 설정

The screenshot shows the Unity Hierarchy and Inspector panels. In the Hierarchy, the 'Exit' object under 'PopupRegisterAccount' is selected. The Inspector shows the 'Button' component with the following settings:

- Tag: Untagged
- Layer: UI
- Rect Transform: (empty)
- Interactable:
- Transition: Color Tint
- Target Graphic: Exit (Image)
- Normal Color: (white)
- Highlighted Color: (white)
- Pressed Color: (white)
- Selected Color: (white)
- Disabled Color: (grey)
- Color Multiplier: 1
- Fade Duration: 0.1
- Navigation: Automatic

The 'On Click ()' section contains two events:

- Event 1: Runtime Only, Target: PopupLog, Method: GameObject.SetActive (checked)
- Event 2: Runtime Only, Target: PopupReg, Method: GameObject.SetActive (unchecked)

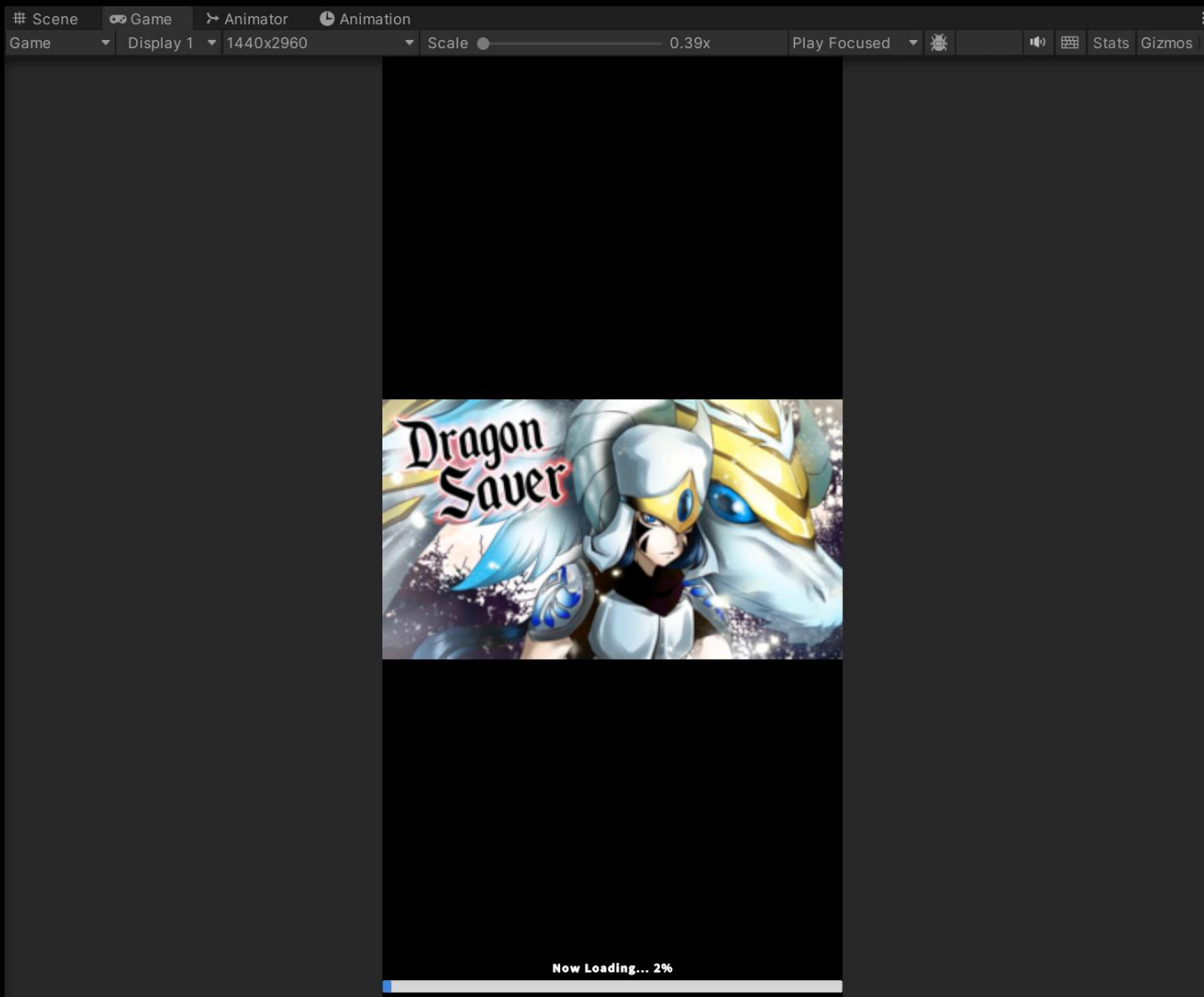
Red dashed boxes highlight the 'Runtime Only' dropdowns, the target objects 'PopupLog' and 'PopupReg', and the 'GameObject.SetActive' method for both events. Red arrows point from the 'Exit' object in the Hierarchy to the event configuration area in the Inspector.

계정 생성 팝업 윈도우에서 "X" 버튼을 클릭하면  
계정 생성 팝업은 비활성화되고, 로그인 팝업이 활성화된다.



# 게임 유저 관리

## ■ 결과 화면





# 게임 유저 관리

## ■ 아이디 찾기

### ■ 아이디 찾기 팝업 윈도우 생성 및 설정

- PopupBase 프리팹을 Hierarchy View로 Drag & Drop

PopupBase 오브젝트의 이름을 "PopupFindID"로 변경



# 게임 유저 관리

- 아이디 찾기 팝업 윈도우 상단 이름 설정

The image shows the Unity Inspector interface for a TextMeshPro text element. The Hierarchy panel on the left shows the following structure:

- Login\*
  - Main Camera
  - Canvas
    - PopupLogin
    - PopupRegisterAccount
    - PopupFindID
      - TopPanel
      - Title** (selected)
      - EXIT
    - Message
    - EventSystem

The Inspector panel on the right shows the following settings for the selected 'TextMeshPro - Text (UI)' component:

- Text Input: 아이디 찾기
- Text Style: Normal
- Main Settings
  - Font Asset: NotoSansKR-Bold SDF (TMP\_...
  - Material Preset: NotoSansKR-BoId SDF Material
  - Font Style: B I U S ab AB SC
  - Font Size: 40
  - Auto Size:
  - Vertex Color:
  - Color Gradient:



# 게임 유저 관리

- 메일 주소 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot shows the Unity interface with three panels: Hierarchy, Inspector, and Hierarchy (right). The Hierarchy panel on the left shows a tree structure under 'Login\*' with 'MailAddress' selected. The Inspector panel in the middle shows the properties of the 'MailAddress' object, including 'Rect Transform' and 'TextMeshPro - Input Field'. The Hierarchy panel on the right shows the 'TextMeshPro - Input Field' settings, including 'Content Type' set to 'Email Address'.

**InputFieldBase** 오브젝트의 이름을 "MailAddress"로 변경

Property	Value
Tag	Untagged
Layer	UI
Prefab	InputFieldBase
Overrides	Select Open
Rect Transform - stretch	top
Rect Transform - Left	100
Rect Transform - Pos Y	-170
Rect Transform - Pos Z	0
Rect Transform - Right	100
Rect Transform - Height	80
Rect Transform - Anchors - Min X	0
Rect Transform - Anchors - Min Y	1
Rect Transform - Anchors - Max X	1
Rect Transform - Anchors - Max Y	1
Rect Transform - Anchors - Pivot X	0.5
Rect Transform - Anchors - Pivot Y	1
Rect Transform - Rotation X	0
Rect Transform - Rotation Y	0
Rect Transform - Rotation Z	0
Rect Transform - Scale X	1
Rect Transform - Scale Y	1
Rect Transform - Scale Z	1
TextMeshPro - Input Field - Content Type	Email Address





# 게임 유저 관리

## ■ 메일 주소 입력 필드 생성 및 설정 (계속)

The image shows the Unity development environment with the Hierarchy and Inspector panels. The Hierarchy panel on the left shows a tree structure under 'Login\*' with 'Placeholder' selected and highlighted by a red box. The Inspector panel on the right shows the properties for the selected 'Placeholder' object, which is a 'TextMeshPro - Text (UI)'. The text content is '메일주소' (Email Address). The Inspector panel is also highlighted with a red box.

**Hierarchy Panel:**

- Login\*
  - Main Camera
  - Canvas
    - PopupLogin
    - PopupRegisterAccount
    - PopupFindID
    - TopPanel
    - MailAddress
      - Text Area
      - Placeholder**
      - Text
    - Message
    - EventSystem

**Inspector Panel:**

- Placeholder (Static) [Tag: Untagged, Layer: UI]
- Rect Transform
- Canvas Renderer
- TextMeshPro - Text (UI)**
  - Text Input [Enable RTL Editor: Off]
  - 메일주소
  - Text Style: Normal
  - Main Settings
    - Font Asset: NotoSansKR-Regular SDF (TMF)
    - Material Preset: NotoSansKR-Regular SDF Material
    - Font Style: B I U S ab AB SC
    - Font Size: 40
    - Auto Size: Off
    - Vertex Color: [Color Picker]
    - Color Gradient: [Color Gradient]



# 게임 유저 관리

- “아이디 찾기” 버튼 생성 및 설정
  - ButtonBase 프리팹을 Hierarchy View로 Drag & Drop

ButtonBase 오브젝트의 이름을  
"ButtonFindID"로 변경

Property	X	Y	Z
Left	100	-300	0
Right	100	100	
Min	0	1	
Max	1	1	
Pivot	0.5	1	
Rotation	0	0	0
Scale	1	1	1



# 게임 유저 관리

- “아이디 찾기” 버튼 생성 및 설정 (계속)

The image shows the Unity development environment. On the left, the Hierarchy panel displays a tree structure under 'Login\*'. The 'Text (TMP)' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected 'TextMeshPro - Text (UI)' object. The text input field contains '아이디 찾기'. The font style is set to 'Normal'. The Inspector panel is also highlighted with a red box.

**Hierarchy Panel:**

- Login\*
  - Main Camera
  - Canvas
    - PopupLogin
    - PopupRegisterAccount
    - PopupFindID
      - TopPanel
      - MailAddress
      - ButtonFindID
      - Text (TMP)**
- Message
- EventSystem

**Inspector Panel:**

- Text (TMP) [Static]
- Tag: Untagged, Layer: UI
- Rect Transform
- Canvas Renderer
- TextMeshPro - Text (UI)**
- Text Input: 아이디 찾기 (Enable RTL Editor: Off)
- Text Style: Normal
- Main Settings
  - Font Asset: NotoSansKR-Bold SDF (TMP\_Fc)
  - Material Preset: NotoSansKR-BoId SDF Material
  - Font Style: B I U S ab AB SC
  - Font Size: 40 (Auto Size: Off)
  - Vertex Color: [Color Picker]
  - Color Gradient: [Color Gradient]



# 게임 유저 관리

- 뒤끝 서버와 연동해 아이디 찾기를 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "FindID"로 변경

```
1 using UnityEngine;
2     using UnityEngine.UI;
3     using TMPro;
4     using BackEnd;
5
6 public class FindID : LoginBase
7 {
8     [SerializeField]
9     private Image          imageEmail;          // E-mail 필드 색상 변경
10    [SerializeField]
11    private TMP_InputField  inputFieldEmail;    // E-mail 필드 텍스트 정보 추출
12
13    [SerializeField]
14    private Button         btnFindID;          // "아이디 찾기" 버튼 (상호작용 가능/불가능)
15
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 아이디 찾기를 제어하는 스크립트 생성 및 작성 (계속)

```
16 public void OnClickFindID()
17 {
18     // 매개변수로 입력한 InputField UI의 색상과 Message 내용 초기화
19     ResetUI(imageEmail);
20
21     // 필드 값이 비어있는지 체크
22     if ( IsFieldDataEmpty(imageEmail, inputFieldEmail.text, "메일 주소") ) return;
23
24     // 메일 형식 검사
25     if ( !inputFieldEmail.text.Contains("@") )
26     {
27         GuideForIncorrectlyEnteredData(imageEmail, "메일 형식이 잘못되었습니다.(ex. address@xx.xx)");
28         return;
29     }
30
31     // "아이디 찾기" 버튼의 상호작용 비활성화
32     btnFindID.interactable = false;
33     SetMessage("메일 발송중입니다.");
34
35     // 뒤끝 서버 아이디 찾기 시도
36     FindCustomID();
37 }
38
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 아이디 찾기를 제어하는 스크립트 생성 및 작성 (계속)

```
39  /// <summary>
40  /// 아이디 찾기를 위해 이메일 발송 시도 후 서버로부터 전달받은 message를 기반으로 로직 처리
41  /// </summary>
42  private void FindCustomID()
43  {
44      // 아이디 정보를 이메일로 발송
45      Backend.BMember.FindCustomID(inputFieldEmail.text, callback =>
46      {
47          // "아이디 찾기" 버튼 상호작용 활성화
48          btnFindID.interactable = true;
49
50          // 메일 발송 성공
51          if ( callback.IsSuccess() )
52          {
53              SetMessage($"{inputFieldEmail.text} 주소로 메일을 발송하였습니다.");
54          }
55          // 메일 발송 실패
56          else ...
57      }
58  });
59  }
```

뒷장



# 게임 유저 관리

- 뒤끝 서버와 연동해 아이디 찾기를 제어하는 스크립트 생성 및 작성 (계속)

```
55 // 메일 발송 실패
56 else
57 {
58     string message = string.Empty;
59
60     switch ( int.Parse(callback.GetStatusCode()) )
61     {
62         case 404: // 해당 이메일의 게이머가 없는 경우
63             message = "해당 이메일을 사용하는 사용자가 없습니다.";
64             break;
65         case 429: // 24시간 이내에 5회 이상 같은 이메일 정보로 아이디/비밀번호 찾기를 시도한 경우
66             message = "24시간 이내에 5회 이상 아이디/비밀번호 찾기를 시도했습니다.";
67             break;
68         default:
69             // statusCode : 400 => 프로젝트 명에 특수문자가 추가된 경우 (안내 메일 미발송 및 에러 발생)
70             message = callback.GetMessage();
71             break;
72     }
73
74     if ( message.Contains("이메일") )
75     {
76         GuideForIncorrectlyEnteredData(imageEmail, message);
77     }
78     else
79     {
80         SetMessage(message);
81     }
82 }
```



# 게임 유저 관리

- PopupFindID 오브젝트에 "FindID" 컴포넌트 추가 및 설정

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Login\*' folder is expanded, and the 'PopupFindID' object is selected. Below it, the 'MailAddress' and 'ButtonFindID' objects are also visible. In the Inspector panel, the 'Find ID (Script)' component is selected, and its 'FindID' property is set to 'FindID'. The 'Message (Text Mesh Pro UGUI)', 'MailAddress (Image)', 'MailAddress (TMP\_Input Field)', and 'ButtonFindID (Button)' properties are also visible and highlighted with red dashed boxes. Red arrows point from the 'MailAddress' and 'ButtonFindID' objects in the Hierarchy panel to their respective entries in the Inspector panel.

**Hierarchy Panel:**

- Login\*
  - Main Camera
  - Canvas
    - PopupLogin
    - PopupRegisterAccount
    - PopupFindID**
      - TopPanel
        - MailAddress
        - ButtonFindID
      - Message
    - EventSystem

**Inspector Panel:**

- PopupFindID
  - Tag: Untagged
  - Layer: UI
  - Prefab: PopupBase
  - Overrides: [Dropdown]
  - Select
  - Open
  - Rect Transform
  - Canvas Renderer
  - Image
  - Find ID (Script)**
    - Script: FindID
    - Text Message: Message (Text Mesh Pro UGUI)
    - Image Email: MailAddress (Image)
    - Input Field Email: MailAddress (TMP\_Input Field)
    - Btn Find ID: ButtonFindID (Button)
  - Default UI Material (Material)
    - Shader: UI/Default

**Add Component**





# 게임 유저 관리

- ButtonFindID 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 설정

The screenshot displays the Unity Inspector interface for a **ButtonFindID** object. The Hierarchy panel on the left shows the object structure, with **ButtonFindID** selected under **PopupFindID**. The Inspector panel on the right shows the **Button** component settings. The **On Click ()** event is configured with the **Runtime Only** dropdown and the **FindID.OnClickFindID** event. The **PopupFindID** event is also visible in the list.



# 게임 유저 관리

## ■ PopupFindID 오브젝트 비활성화

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Login\*'. The 'PopupFindID' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected object. The 'Find ID (Script)' component is highlighted with a blue bar, and a red box highlights the 'Find ID (Script)' component in the Inspector panel.



# 게임 유저 관리

- FindID 오브젝트에 "UITextInteraction" 컴포넌트 추가 및 설정

로그인 팝업 윈도우에서 "아이디 찾기" 텍스트를 클릭하면  
로그인 팝업은 비활성화되고, 아이디 찾기 팝업이 활성화된다.



# 게임 유저 관리

- Exit 오브젝트의 "Button" 컴포넌트 OnClick() 이벤트 설정

The screenshot displays the Unity Inspector for the 'Exit' object. The Hierarchy panel on the left shows the scene structure, with 'Exit' selected. The Inspector panel shows the Button component's settings, including the On Click () event list. Two events are configured: 'Runtime Only' and 'PopupLog', and 'Runtime Only' and 'PopupFinc'. Red dashed boxes highlight the event names and the 'Runtime Only' dropdowns. Red arrows point from the 'Exit' object in the Hierarchy panel to the Button component in the Inspector panel.

아이디 찾기 팝업 윈도우에서 "X" 버튼을 클릭하면  
아이디 찾기 팝업은 비활성화되고, 로그인 팝업이 활성화된다.



# 게임 유저 관리

## ■ 결과 화면

The screenshot displays a game engine interface with a notification and a login form. The notification, highlighted with a red dashed border, contains the following text:

☆ 요청하신 아이디 찾기 안내입니다.  
보낸사람 : "ProjectA" <no-reply@thebackend.io> 주소록추가 수신자단 23-01-27 12:10

요청하신 아이디는 다음과 같습니다.  
user03, user02  
감사합니다.

Below the notification is a login form titled "로그인" (Login). The form includes:

- An input field for "아이디" (ID) with the placeholder text "아이디".
- A password input field, currently masked with black dots.
- A yellow "로그인" (Login) button.
- Three links: "아이디 찾기" (Find ID), "계정 생성" (Create Account), and "비밀번호 찾기" (Find Password).

The background of the interface is dark gray, and a mouse cursor is visible on the right side.



# 게임 유저 관리

## ■ 비밀번호 찾기

- 비밀번호 찾기 팝업 윈도우 생성 및 설정
  - PopupBase 프리팹을 Hierarchy View로 Drag & Drop

PopupBase 오브젝트의 이름을  
"PopupFindPW"로 변경



# 게임 유저 관리

- 비밀번호 찾기 팝업 윈도우 상단 이름 설정

The image shows the Unity development environment interface. On the left is the Hierarchy panel, and on the right is the Inspector panel.

**Hierarchy Panel:** Shows a tree structure of game objects. The 'Login\*' folder is expanded, showing sub-objects like 'Main Camera', 'Canvas', 'PopupLogin', 'PopupRegisterAccount', 'PopupFindID', 'PopupFindPW', 'TopPanel', 'Title', 'Exit', 'Message', and 'EventSystem'. The 'Title' object is selected and highlighted with a red box.

**Inspector Panel:** Shows the properties for the selected 'TextMeshPro - Text (UI)' object. The 'Text Input' field contains the text '비밀번호 찾기', which is also highlighted with a red box. Other visible properties include 'Text Style' (Normal), 'Main Settings' (Font Asset: NotoSansKR-Bold SDF (TMP\_Fc), Material Preset: NotoSansKR-BoId SDF Material, Font Style: B, I, U, S, ab, AB, SC, Font Size: 40), and 'Vertex Color' (white).



# 게임 유저 관리

- 아이디 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

**InputFieldBase** 오브젝트의 이름을 "ID"로 변경

Property	Value
ID	Static
Tag	Untagged
Layer	UI
Prefab	InputFieldBase
Overrides	Select Open
stretch	Left: 100, Right: 100, Pos Y: -170, Height: 80, Pos Z: 0
Anchors	Min: X 0, Y 1; Max: X 1, Y 1; Pivot: X 0.5, Y 1
Rotation	X 0, Y 0, Z 0
Scale	X 1, Y 1, Z 1





# 게임 유저 관리

## ■ 아이디 입력 필드 생성 및 설정 (계속)

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a scene named 'Login\*' with a 'Canvas' containing several UI elements. The 'Placeholder' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected 'Placeholder' object. The 'TextMeshPro - Text (UI)' component is expanded and highlighted with a red box. The text input field contains the Korean characters '아이디' (ID). The font asset is set to 'NotoSansKR-Regular SDF (TMF)' and the font size is 40.

**Hierarchy Panel:**

- Login\*
  - Main Camera
  - Canvas
    - PopupLogin
    - PopupRegisterAccount
    - PopupFindID
    - PopupFindPW
    - TopPanel
    - ID
      - Text Area
      - Placeholder**
      - Text
    - Message
    - EventSystem

**Inspector Panel (TextMeshPro - Text (UI)):**

- Text Input: Enable RTL Editor
- Text: 아이디
- Text Style: Normal
- Main Settings
  - Font Asset: **F** NotoSansKR-Regular SDF (TMF)
  - Material Preset: NotoSansKR-Regular SDF Material
  - Font Style: B **I** U S ab AB SC
  - Font Size: 40
  - Auto Size:
  - Vertex Color:
  - Color Gradient:



# 게임 유저 관리

- 메일 주소 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

**InputFieldBase 오브젝트의 이름을 "MailAddress"로 변경**

Property	X	Y	Z
Left	100	-270	0
Right	100	80	
Height		80	
Min	0	1	
Max	1	1	
Pivot	0.5	1	
Rotation	0	0	0
Scale	1	1	1

**Input Field Settings**

Content Type	Email Address
Placeholder	Placeholder (Text Mesh Pro L)



# 게임 유저 관리

## ■ 메일 주소 입력 필드 생성 및 설정 (계속)

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Placeholder' object under the 'Text Area' folder is selected. The Inspector panel shows the properties for the 'TextMeshPro - Text (UI)' component. The 'Text Input' section is expanded, showing the text '메일주소' (Email Address) entered in the input field. The 'Main Settings' section is also visible, showing the font asset set to 'NotoSansKR-Regular SDF (TMF)' and the font size set to 40.

**Hierarchy Panel:**

- Login\*
  - Main Camera
  - Canvas
    - PopupLogin
    - PopupRegisterAccount
    - PopupFindID
    - PopupFindPW
    - TopPanel
    - ID
    - MailAddress
    - Text Area
      - Placeholder (Selected)
      - Text
  - Message
  - EventSystem

**Inspector Panel:**

- Placeholder (Static) [Tag: Untagged, Layer: UI]
- Rect Transform
- Canvas Renderer
- TextMeshPro - Text (UI)**
  - Text Input: Enable RTL Editor [Off]
  - Text: 메일주소
  - Text Style: Normal
  - Main Settings
    - Font Asset: NotoSansKR-Regular SDF (TMF)
    - Material Preset: NotoSansKR-Regular SDF Material
    - Font Style: B I U S ab AB SC
    - Font Size: 40
    - Auto Size: [Off]
    - Vertex Color: [White]
    - Color Gradient: [Off]



# 게임 유저 관리

- “비밀번호 찾기” 버튼 생성 및 설정
  - ButtonBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot shows the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows a tree structure under 'Login\*'. The 'PopupFindPW' object is selected, and its children are visible: 'TopPanel', 'ID', 'MailAddress', 'ButtonFindPW', and 'Text (TMP)'. The Inspector panel on the right shows the 'ButtonFindPW' component. The 'Rect Transform' component is expanded, showing the 'stretch' and 'Anchors' settings. The 'stretch' settings are: Left: 100, Right: 100, Pos Y: -400, Height: 100. The 'Anchors' settings are: Min: X 0, Y 1; Max: X 1, Y 1; Pivot: X 0.5, Y 1. The 'Rotation' settings are: X 0, Y 0, Z 0. The 'Scale' settings are: X 1, Y 1, Z 1. A text box at the bottom left says: 'ButtonBase 오브젝트의 이름을 "ButtonFindPW"로 변경'.

ButtonBase 오브젝트의 이름을  
"ButtonFindPW"로 변경



# 게임 유저 관리

- “비밀번호 찾기” 버튼 생성 및 설정 (계속)

The image shows the Unity development environment. On the left is the Hierarchy panel, and on the right is the Inspector panel.

**Hierarchy Panel:**

- Root: Login\*
- Canvas
  - PopupLogin
  - PopupRegisterAccount
  - PopupFindID
  - PopupFindPW
    - TopPanel
    - ID
    - MailAddress
    - ButtonFindPW
    - Text (TMP)** (Selected)
- Message
- EventSystem

**Inspector Panel:**

- Component: Text (TMP) [Static]
- Tag: Untagged, Layer: UI
- Rect Transform
- Canvas Renderer
- TextMeshPro - Text (UI)** (Selected)
  - Text Input: 비밀번호 찾기
  - Enable RTL Editor: [Off]
  - Text Style: Normal
  - Main Settings
    - Font Asset: NotoSansKR-Bold SDF (TMP\_Fc)
    - Material Preset: NotoSansKR-BoId SDF Material
    - Font Style: B I U S ab AB SC
    - Font Size: 40
    - Auto Size: [Off]
    - Vertex Color: [Color Picker]
    - Color Gradient: [Color Gradient]



# 게임 유저 관리

- 뒤끝 서버와 연동해 비밀번호 찾기를 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "FindPW"로 변경

```
1  using UnityEngine;
2  using UnityEngine.UI;
3  using TMPro;
4  using BackEnd;
5
6  public class FindPW : LoginBase
7  {
8      [SerializeField]
9      private Image          imageID;          // ID 필드 색상 변경
10     [SerializeField]
11     private TMP_InputField  inputFieldID;    // ID 필드 텍스트 정보 추출
12     [SerializeField]
13     private Image          imageEmail;      // E-mail 필드 색상 변경
14     [SerializeField]
15     private TMP_InputField  inputFieldEmail; // E-mail 필드 텍스트 정보 추출
16
17     [SerializeField]
18     private Button          btnFindPW;      // "비밀번호 찾기" 버튼 (상호작용 가능/불가능)
19
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 비밀번호 찾기를 제어하는 스크립트 생성 및 작성 (계속)

```
20 public void OnClickFindPW()
21 {
22     // 매개변수로 입력한 InputField UI의 색상과 Message 내용 초기화
23     ResetUI(imageID, imageEmail);
24
25     // 필드 값이 비어있는지 체크
26     if ( IsFieldDataEmpty(imageID, inputFieldID.text, "아이디") ) return;
27     if ( IsFieldDataEmpty(imageEmail, inputFieldEmail.text, "메일 주소") ) return;
28
29     // 메일 형식 검사
30     if ( !inputFieldEmail.text.Contains("@") )
31     {
32         GuideForIncorrectlyEnteredData(imageEmail, "메일 형식이 잘못되었습니다.(ex. address@xx.xx)");
33         return;
34     }
35
36     // "비밀번호 찾기" 버튼의 상호작용 비활성화
37     btnFindPW.interactable = false;
38     SetMessage("메일 발송중입니다.");
39
40     // 뒤끝 서버 비밀번호 찾기 시도
41     FindCustomPW();
42 }
43
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 비밀번호 찾기를 제어하는 스크립트 생성 및 작성 (계속)

```
44  // <summary>
45  // 비밀번호를 리셋하기 위해 이메일 발송 시도 후 서버로부터 전달받은 message를 기반으로 로직 처리
46  // </summary>
47  private void FindCustomPW()
48  {
49      // 리셋된 비밀번호 정보를 이메일로 발송
50      Backend.BMember.ResetPassword(inputFieldID.text, inputFieldEmail.text, callback =>
51      {
52          // "비밀번호 찾기" 버튼 상호작용 활성화
53          btnFindPW.interactable = true;
54
55          // 메일 발송 성공
56          if ( callback.IsSuccess() )
57          {
58              SetMessage($"{inputFieldEmail.text} 주소로 메일을 발송하였습니다.");
59          }
60          // 메일 발송 실패
61          else ...
62      });
63  }
```

뒷장





# 게임 유저 관리

- 뒤끝 서버와 연동해 비밀번호 찾기를 제어하는 스크립트 생성 및 작성 (계속)

```
60 // 메일 발송 실패
61 else
62 {
63     string message = string.Empty;
64
65     switch ( int.Parse(callback.GetStatusCode()) )
66     {
67         case 404: // 해당 이메일의 게이머가 없는 경우
68             message = "해당 이메일을 사용하는 사용자가 없습니다.";
69             break;
70         case 429: // 24시간 이내에 5회 이상 같은 이메일 정보로 아이디/비밀번호 찾기를 시도한 경우
71             message = "24시간 이내에 5회 이상 아이디/비밀번호 찾기를 시도했습니다.";
72             break;
73         default:
74             // statusCode : 400 => 프로젝트 명에 특수문자가 추가된 경우 (안내 메일 미발송 및 에러 발생)
75             message = callback.GetMessage();
76             break;
77     }
78
79     if ( message.Contains("이메일") )
80     {
81         GuideForIncorrectlyEnteredData(imageEmail, message);
82     }
83     else
84     {
85         SetMessage(message);
86     }
87 }
```



# 게임 유저 관리

- PopupFindPW 오브젝트에 "FindPW" 컴포넌트 추가 및 설정

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Login\*' object is expanded to show 'Canvas', which contains 'PopupLogin', 'PopupRegisterAccount', 'PopupFindID', and 'PopupFindPW'. The 'PopupFindPW' object is selected, and its children 'ID', 'MailAddress', and 'ButtonFindPW' are also highlighted with red boxes. Red arrows point from these boxes to the corresponding fields in the 'Find PW (Script)' component in the Inspector panel. The Inspector panel shows the 'FindPW' script component with the following fields:

- Script: FindPW
- Text Message: Message (Text Mesh Pro UGUI)
- Image ID: ID (Image)
- Input Field ID: ID (TMP\_Input Field)
- Image Email: MailAddress (Image)
- Input Field Email: MailAddress (TMP\_Input Field)
- Btn Find PW: ButtonFindPW (Button)

The Inspector panel also shows other components like 'Rect Transform', 'Canvas Renderer', and 'Image' for the selected object. At the bottom, there is a 'Default UI Material (Material)' section with a 'Shader' dropdown set to 'UI/Default' and an 'Add Component' button.



# 게임 유저 관리

- ButtonFindPW 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 설정

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Login\*' with 'Canvas' containing several popups. 'PopupFindPW' is selected, and its 'ButtonFindPW' child is highlighted with a red box. A red line connects this box to the Inspector panel on the right. The Inspector panel shows the 'ButtonFindPW' object with its 'Button' component selected. The 'Button' component's 'On Click ()' event is configured with 'Runtime Only' and 'FindPW.OnClickFindPW'. A red dashed box highlights the event name, and another red dashed box highlights the 'PopupFindPW' icon in the event list, with a red arrow pointing from the Hierarchy panel to it. The 'Button' component also shows various visual and navigation settings like 'Color Tint', 'Normal Color', and 'Navigation'.



# 게임 유저 관리

## ■ PopupFindPW 오브젝트 비활성화

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Login\*'. The 'PopupFindPW' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected object. The 'Static' checkbox is checked and highlighted with a red box, indicating that the object is inactive. The Inspector also shows the object is a Prefab of 'PopupBase' and lists components: Rect Transform, Canvas Renderer, Image, and Find PW (Script).

**Hierarchy Panel:**

- Login\*
  - Main Camera
  - Canvas
    - PopupLogin
    - PopupRegisterAccount
    - PopupFindID
    - PopupFindPW**
    - Message
  - EventSystem

**Inspector Panel:**

- Static:
- Tag: Untagged
- Layer: UI
- Prefab: PopupBase
- Overrides: [Dropdown]
- Select: [Button]
- Open: [Button]
- Rect Transform
- Canvas Renderer
- Image
- Find PW (Script)
- Add Component



# 게임 유저 관리

- FindPW 오브젝트에 "UITextInteraction" 컴포넌트 추가 및 설정

로그인 팝업 윈도우에서 "비밀번호 찾기" 텍스트를 클릭하면  
로그인 팝업은 비활성화되고, 비밀번호 찾기 팝업이 활성화된다.



# 게임 유저 관리

- Exit 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 설정

The screenshot shows the Unity Hierarchy and Inspector panels. In the Hierarchy, the 'Exit' object is selected under the 'PopupFindPW' folder. The Inspector shows the 'Exit' component with the following settings:

- Tag: Untagged
- Layer: UI
- Rect Transform: (empty)
- Interactable: checked
- Transition: Color Tint
- Target Graphic: Exit (Image)
- Normal Color, Highlighted Color, Pressed Color, Selected Color, Disabled Color: (color pickers)
- Color Multiplier: 1
- Fade Duration: 0.1
- Navigation: Automatic
- Visualize: (button)
- On Click ():
  - Runtime Only: checked, PopupLog (checked)
  - Runtime Only: checked, PopupFinc (unchecked)

Red boxes highlight the 'Exit' object in the Hierarchy, the 'Exit' component in the Inspector, and the 'On Click ()' event list. Red arrows point from the 'Exit' object in the Hierarchy to the 'Exit' component in the Inspector, and from the 'Exit' component to the 'On Click ()' event list.

비밀번호 찾기 팝업 윈도우에서 "X" 버튼을 클릭하면  
비밀번호 찾기 팝업은 비활성화되고, 로그인 팝업이 활성화된다.



# 게임 유저 관리

## ■ 결과 화면

The screenshot shows a game engine interface with a notification message and a login form. The notification message is highlighted with a red dashed border and contains the following text:

☆ 요청하신 비밀번호 초기화 안내입니다.  
보낸사람: "ProjectA" <no-reply@thebackend.io> 주소록추가 수신저단 23-01-27 12:31

요청하신 비밀번호 초기화 결과 다음과 같이 임시 비밀번호가 발급되었습니다.  
q7072q0k  
임시 비밀번호로 로그인하신 후, 새로운 비밀번호로 변경해주세요.  
감사합니다.

Below the notification, there is a login form titled "로그인" (Login). The form has two input fields: "아이디" (ID) and "비밀번호" (Password). The password field is masked with black dots. Below the input fields is a yellow "로그인" (Login) button. At the bottom of the form, there are three links: "아이디 찾기" (Find ID), "계정 생성" (Create Account), and "비밀번호 찾기" (Find Password).

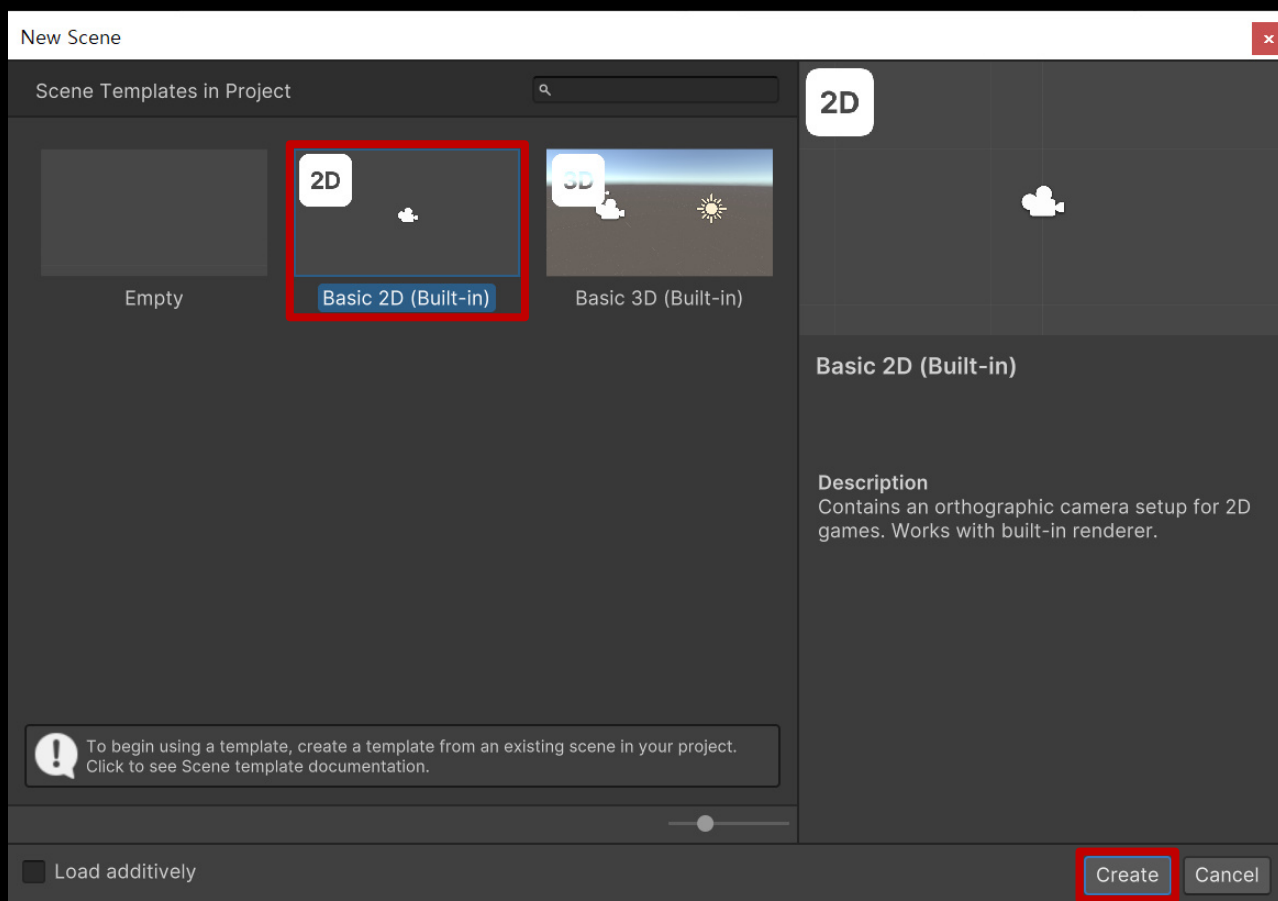
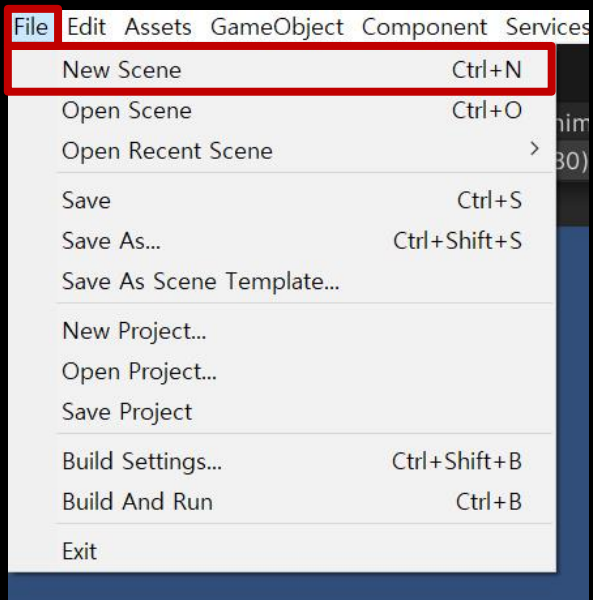


# 게임 유저 관리

## ■ 유저 정보 출력

### ■ Lobby 씬 생성

#### □ File - New Scene







# 게임 유저 관리

## ■ 카메라 설정

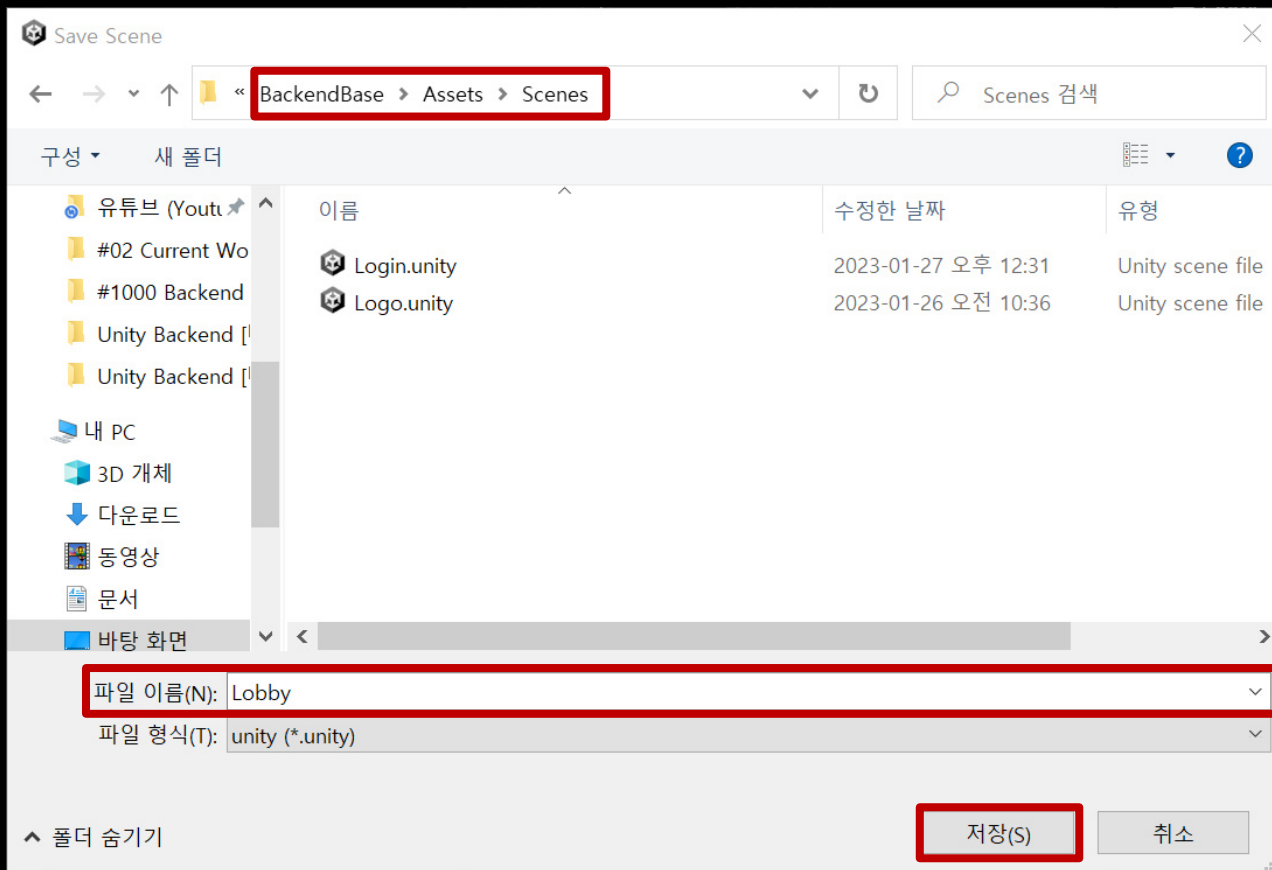
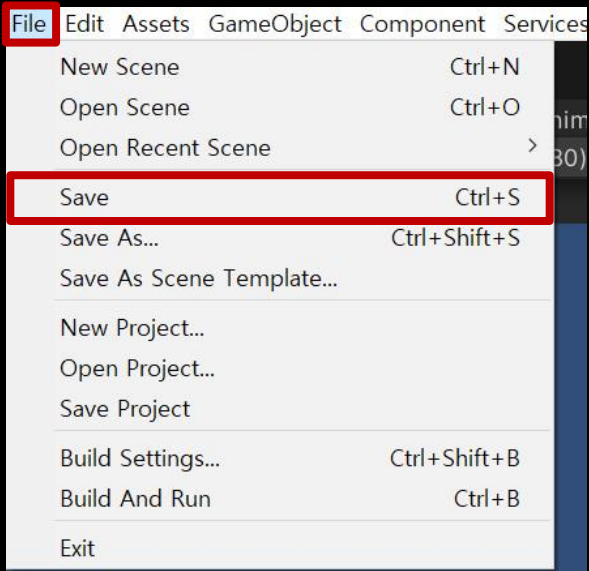
The screenshot displays the Unity Inspector interface for a 'Main Camera' object. The Hierarchy panel on the left shows the 'Main Camera' object selected. The Inspector panel on the right shows the 'Main Camera' component with the following settings:

- Tag: MainCamera
- Layer: Default
- Transform: Position (X: 0, Y: 0, Z: -10), Rotation (X: 0, Y: 0, Z: 0), Scale (X: 1, Y: 1, Z: 1)
- Camera component (highlighted with a red box):
  - Clear Flags: Solid Color
  - Background: (highlighted with a red dashed box)
  - Culling Mask: Everything
  - Projection: Orthographic
  - Size: 5
  - Clipping Planes: Near (0.3), Far (1000)
  - Viewport Rect: X (0), Y (0), W (1), H (1)
  - Depth: -1
  - Rendering Path: Use Graphics Settings
  - Target Texture: None (Render Texture)
  - Occlusion Culling:
  - HDR: Use Graphics Settings
  - MSAA: Use Graphics Settings
  - Allow Dynamic Resol:
  - Target Display: Display 1
- Audio Listener component (highlighted with a red box):



# 게임 유저 관리

- Lobby 씬 저장
  - File - Save





# 게임 유저 관리

- Scenes In Build에 씬 등록
  - File - Build Settings

The screenshot displays the Unity interface with two main panels:

- Build Settings:** The 'Scenes In Build' list is visible, showing three entries: 'Scenes/Logo' (count 0), 'Scenes/Login' (count 1), and 'Scenes/Lobby' (count 2). The 'Scenes/Lobby' entry is highlighted with a blue background and a red border.
- Project Hierarchy:** The 'Assets > Scenes' folder is expanded, showing three scene files: 'Lobby', 'Login', and 'Logo'. The 'Lobby' scene is highlighted with a red box, and a red arrow points from this box to the '2' in the Build Settings panel.

Below the Build Settings panel, the 'Platform' dropdown is set to 'Windows, Mac, Linux'. The 'Target Platform' is 'Windows' and the 'Architecture' is 'Intel 64-bit'. Other settings like 'Copy PDB files', 'Create Visual Studio Solution', 'Development Build', 'Autoconnect Profiler', 'Deep Profiling', 'Script Debugging', and 'Compression Method' are also visible.



# 게임 유저 관리

- SceneNames 열거형에 Lobby 추가
  - Utils Script 수정

```
1     using UnityEngine.SceneManagement;
2
3     public enum SceneNames { Logo=0, Login, Lobby, }
4
5     public static class Utils
6     {
7         public static string GetActiveScene()...
11
12         public static void LoadScene(string sceneName="")...
23
24         public static void LoadScene(SceneNames sceneName)...
29     }
```



# 게임 유저 관리

- 로그인에 성공했을 때 Lobby 씬으로 이동
  - Login Script 수정

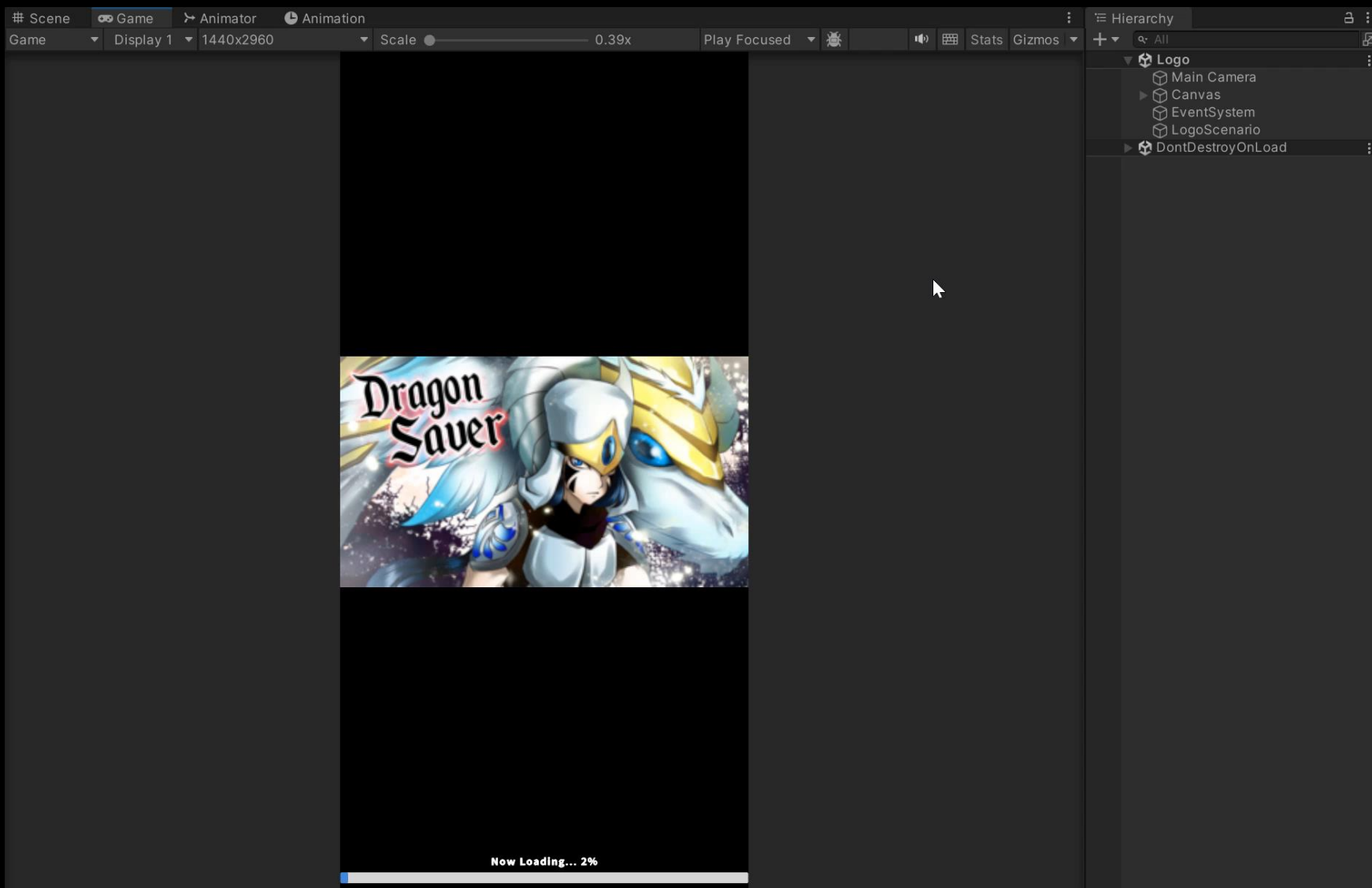
```
21  +  /// <summary> "로그인" 버튼을 눌렀을 때 호출
24  +  public void OnClickLogin()...
43
44  +  /// <summary> 로그인 시도 후 서버로부터 전달받은 message를 기반으로 로직 처리
47  -  private void ResponseToLogin(string ID, string PW)
48  {
49      // 서버에 로그인 요청
50      Backend.BMember.CustomLogin(ID, PW, callback =>
51      {
52          StopCoroutine(nameof(LoginProcess));
53
54          // 로그인 성공
55          if ( callback.IsSuccess() )
56          {
57              SetMessage($"{inputFieldID.text}님 환영합니다.");
58
59              // Lobby 씬으로 이동
60              Utils.LoadScene(SceneNames.Lobby);
61          }
62          // 로그인 실패
63          else...
96      });
97  }
98
99  +  private IEnumerator LoginProcess()...
112 }
```





# 게임 유저 관리

## ■ 결과 화면





# 게임 유저 관리

- Lobby 씬 상단의 프로필, 재화 UI들을 관리하는 Panel UI 생성 및 설정
  - GameObject - UI - Panel

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Canvas' object under the 'Lobby\*' scene is selected and highlighted with a red box. The Inspector panel shows the 'Canvas Scaler' component selected, with its settings highlighted by a red dashed box. The settings include:

- UI Scale Mode: Scale With Screen Size
- Reference Resolution X: 1440, Y: 2960
- Screen Match Mode: Match Width Or Height
- Match: A slider set to 0.5, with 'Width' and 'Height' labels.
- Reference Pixels Per Unit: 100

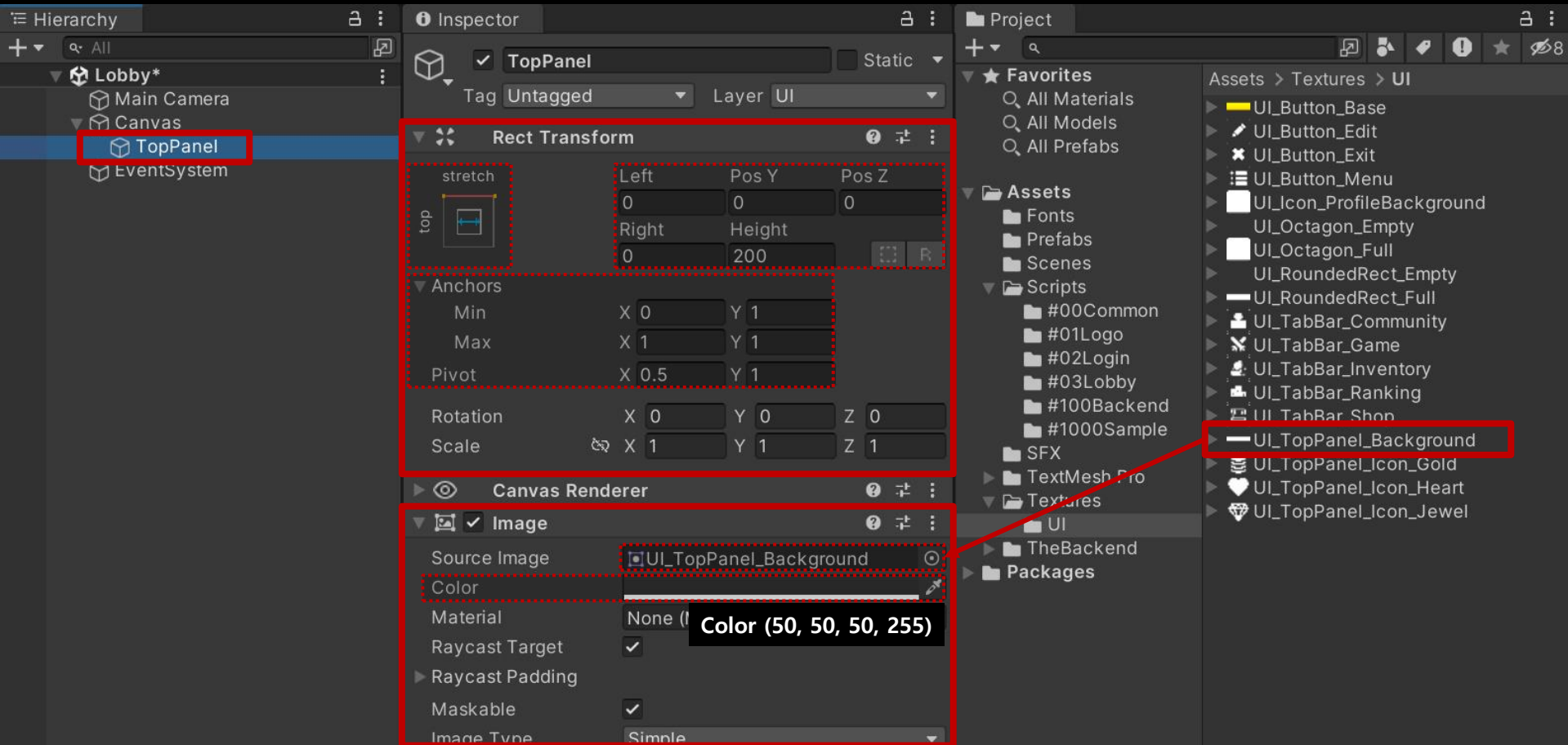
Below the Canvas Scaler component, the 'Graphic Raycaster' component is also visible. At the bottom of the Inspector panel, there is an 'Add Component' button.





# 게임 유저 관리

- Lobby 씬 상단의 프로필, 재화 UI들을 관리하는 Panel UI 생성 및 설정 (계속)





# 게임 유저 관리

- 유저 프로필 정보 UI들을 관리하는 Panel UI 생성 및 설정
  - GameObject - UI - Panel

화면에 출력하지 않도록  
CanvasRenderer, Image 컴포넌트 삭제



# 게임 유저 관리

- 프로필 "Button - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Button - TextMeshPro"

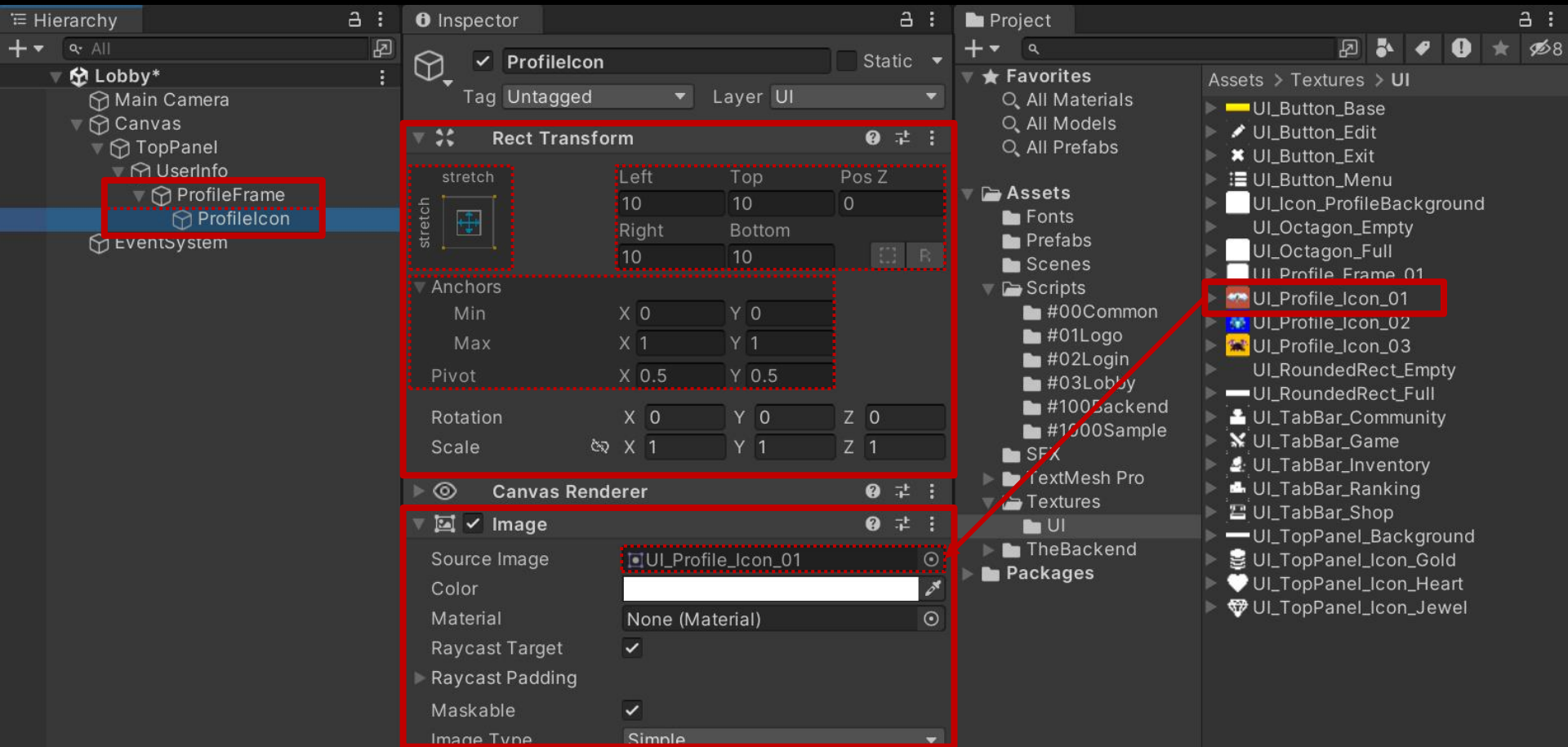
The screenshot displays the Unity development environment. The Hierarchy panel on the left shows a tree structure under 'Lobby\*' with 'UserInfo' containing 'ProfileFrame' and 'Text (TMP)'. The Inspector panel in the center shows the 'ProfileFrame' selected, with its 'Rect Transform' and 'Image' components highlighted by red dashed boxes. The 'Rect Transform' component shows a 'left' anchor and a width of 160. The 'Image' component shows the 'Source Image' set to 'UI\_Profile\_Frame\_01'. The Project panel on the right shows the 'Assets' folder structure, with 'UI\_Profile\_Frame\_01' highlighted in the 'Assets > Textures > UI' path.

**버튼과 함께 생성되는  
Text (TMP) 오브젝트는 삭제**



# 게임 유저 관리

- 프로필 아이콘 Image UI 생성 및 설정
  - GameObject - UI - Image





# 게임 유저 관리

## ■ 프로필 오브젝트 Prefab 생성

- Hierarchy View의 "ProfileFrame" 오브젝트를 Project View로 Drag & Drop

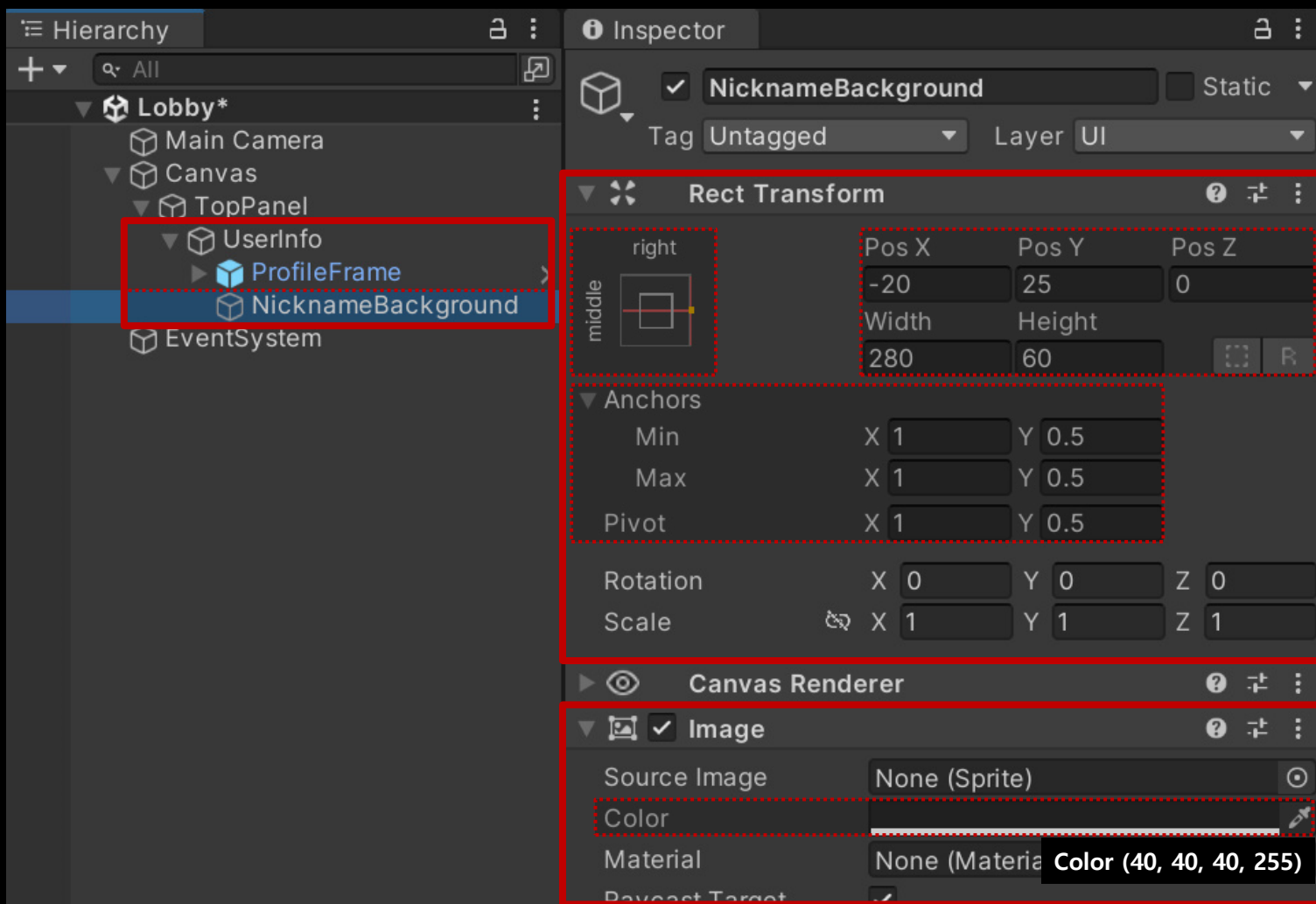
The screenshot shows the Unity interface with three panels: Hierarchy, Inspector, and Project. In the Hierarchy panel, the 'ProfileFrame' object under 'UserInfo' is highlighted with a red box. A red arrow points from this box to the 'ProfileFrame' prefab in the Project panel's 'Assets > Prefabs' folder, which is also highlighted with a red box. The Inspector panel shows the 'ProfileFrame' (Prefab Asset) with its properties, including 'Tag: Untagged' and 'Layer: UI'. A text box in the bottom left corner explains the reason for this action.

현재 UserInfo에도 프로필 정보가 필요하기  
때문에 오브젝트를 삭제하지 않고 사용



# 게임 유저 관리

- 닉네임 배경 Image UI 생성 및 설정
  - GameObject - UI - Image





# 게임 유저 관리

- 닉네임을 출력하는 "Text - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Text - TextMeshPro"

The screenshot shows the Unity interface. In the Hierarchy panel, the 'Nickname' object is selected under 'NicknameBackground'. The Inspector panel shows the 'Rect Transform' component with a stretch box and 'Anchors' set to (0,0,1,1). Below it, the 'TextMeshPro - Text (UI)' component is visible, showing the font asset and material.

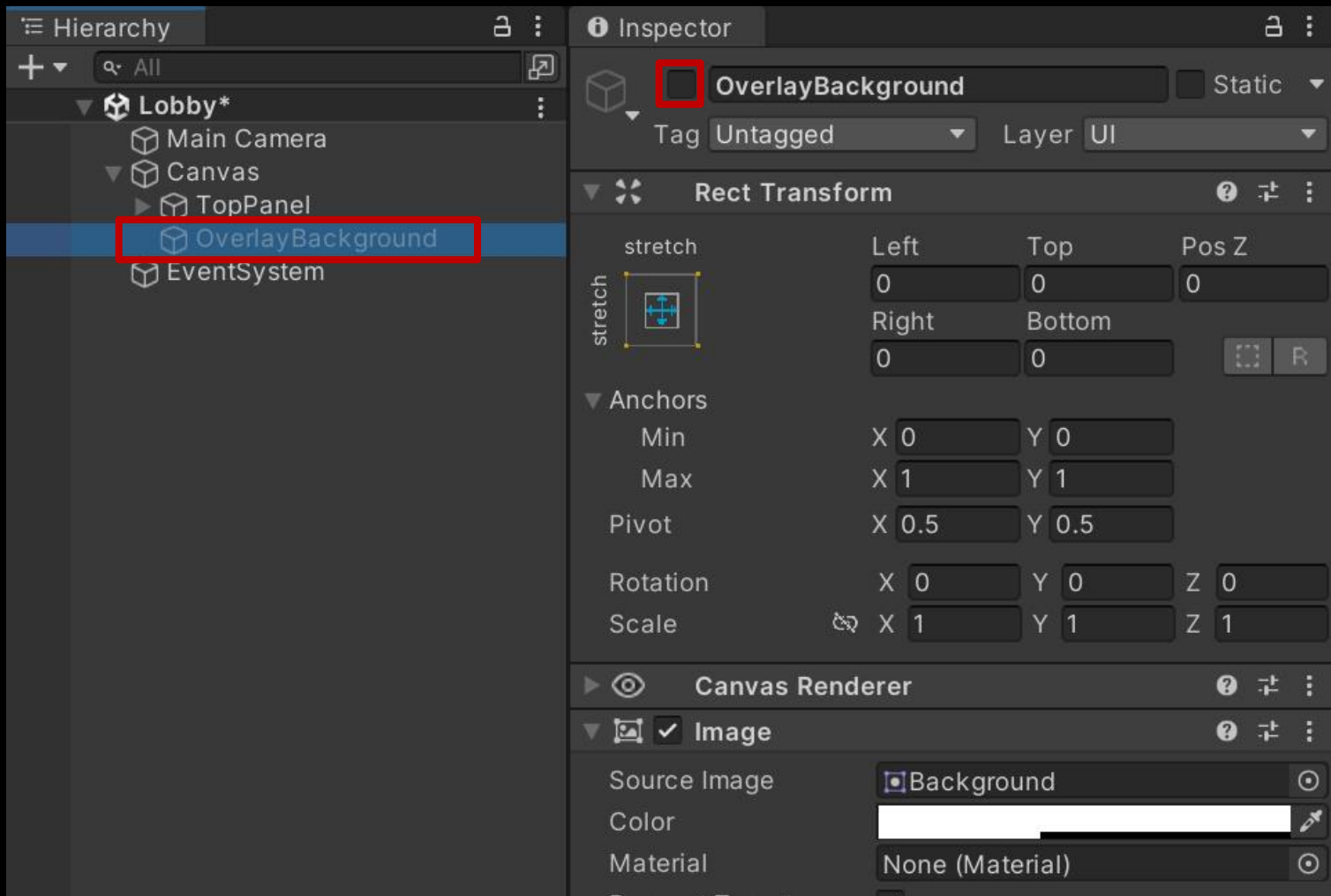
The screenshot shows the TextMeshPro Inspector panel. The 'Text Input' field contains the Korean text '닉네임'. The 'Main Settings' section shows the font asset 'NotoSansKR-Bold SDF (TMP\_...)'. The 'Font Style' section shows 'B I U S ab AB SC' with 'B' selected. The 'Alignment' section shows 'Left' selected. The 'Wrapping' section shows 'Enabled'. The 'Overflow' section shows 'Ellipsis'.

The screenshot shows two overflow options in a UI. The first option is 'Overflow : Ellipsis' and the second option is 'Overflow : Overflow'. Both options are highlighted with a red box.



# 게임 유저 관리

- 제일 앞에 활성화되는 팝업 UI 뒤에 배치하는 배경 Panel UI 생성 및 설정
  - GameObject - UI - Panel







# 게임 유저 관리

- 현재 로그인한 유저의 정보를 불러오는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "UserInfo"로 변경

```
1  using UnityEngine;
2  using BackEnd;
3  using LitJson;
4
5  public class UserInfo : MonoBehaviour
6  {
7      [System.Serializable]
8      public class UserInfoEvent : UnityEngine.Events.UnityEvent { }
9      public UserInfoEvent onUserInfoEvent = new UserInfoEvent();
10
11     private static UserInfoData data = new UserInfoData();
12     public static UserInfoData Data => data;
13
```



# 게임 유저 관리

- 현재 로그인한 유저의 정보를 불러오는 스크립트 생성 및 작성 (계속)

```
14 public void GetUserInfoFromBackend()  
15 {  
16     // 현재 로그인한 사용자 정보 불러오기  
17     // https://developer.thebackend.io/unity3d/guide/bmember/userInfo/  
18     Backend.BMember.GetUserInfo(callback =>  
19     {  
20         // 정보 불러오기 성공  
21         if ( callback.IsSuccess() ) ...  
45         // 정보 불러오기 실패  
46         else  
47         {  
48             // 유저 정보를 기본 상태로 설정  
49             // Tip. 일반적으로 오프라인 상태를 대비해 기본적인 정보를 저장해두고 오프라인일 때 불러와서 사용  
50             data.Reset();  
51             Debug.LogError(callback.GetMessage());  
52         }  
53  
54         // 유저 정보 불러오기에 성공했을 때 onUserInfoEvent에 등록되어 있는 이벤트 메소드 호출  
55         onUserInfoEvent?.Invoke();  
56     });  
57 }  
58 }  
59
```



# 게임 유저 관리

- 현재 로그인한 유저의 정보를 불러오는 스크립트 생성 및 작성 (계속)

```
20 // 정보 불러오기 성공
21 if ( callback.IsSuccess() )
22 {
23     // JSON 데이터 파싱 성공
24     try
25     {
26         JsonData json = callback.GetReturnValuetoJSON()["row"];
27
28         data.gamerId           = json["gamerId"].ToString();
29         data.countryCode      = json["countryCode"]?.ToString();
30         data.nickname         = json["nickname"]?.ToString();
31         data.inDate           = json["inDate"].ToString();
32         data.emailForFindPassword = json["emailForFindPassword"]?.ToString();
33         data.subscriptionType  = json["subscriptionType"].ToString();
34         data.federationId     = json["federationId"]?.ToString();
35     }
36     // JSON 데이터 파싱 실패
37     catch ( System.Exception e )
38     {
39         // 유저 정보를 기본 상태로 설정
40         data.Reset();
41         // try-catch 에러 출력
42         Debug.LogError(e);
43     }
44 }
```



# 게임 유저 관리

- 현재 로그인한 유저의 정보를 불러오는 스크립트 생성 및 작성 (계속)

```
60 public class UserInfoData
61 {
62     public string  gamerId;           // 유저의 gamerID
63     public string  countryCode;      // 국가코드. 설정 안했으면 null
64     public string  nickname;        // 닉네임. 설정 안했으면 null
65     public string  inDate;          // 유저의 inDate
66     public string  emailForFindPassword; // 이메일주소. 설정 안했으면 null
67     public string  subscriptionType; // 커스텀, 페더레이션 타입
68     public string  federationId;     // 구글, 애플, 페이스북 페더레이션 ID. 커스텀 계정은 null
69
70     public void Reset()
71     {
72         gamerId           = "Offline";
73         countryCode       = "Unknown";
74         nickname          = "Noname";
75         inDate            = string.Empty;
76         emailForFindPassword = string.Empty;
77         subscriptionType  = string.Empty;
78         federationId      = string.Empty;
79     }
80 }
```



# 게임 유저 관리

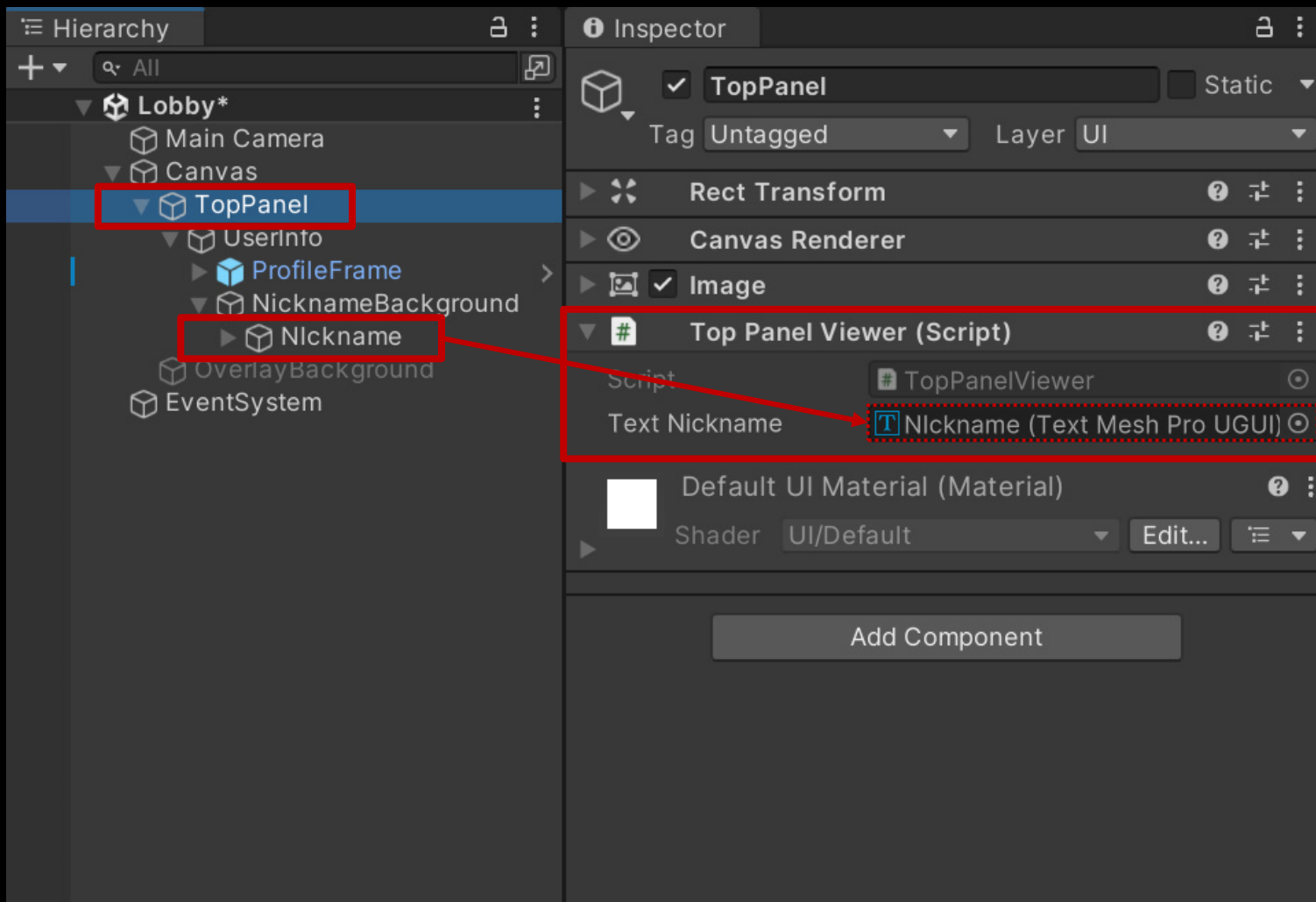
- Top Panel에 출력하는 UI 정보를 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "TopPanelViewer"로 변경

```
1 using UnityEngine;
2 using TMPro;
3
4 public class TopPanelViewer : MonoBehaviour
5 {
6     [SerializeField]
7     private TextMeshProUGUI textNickname;
8
9     public void UpdateNickname()
10    {
11        // 닉네임이 없으면 gamer_id를 출력하고, 닉네임이 있으면 닉네임 출력
12        textNickname.text = UserInfo.Data.nickname == null ?
13            UserInfo.Data.gamerId : UserInfo.Data.nickname;
14    }
15 }
```



# 게임 유저 관리

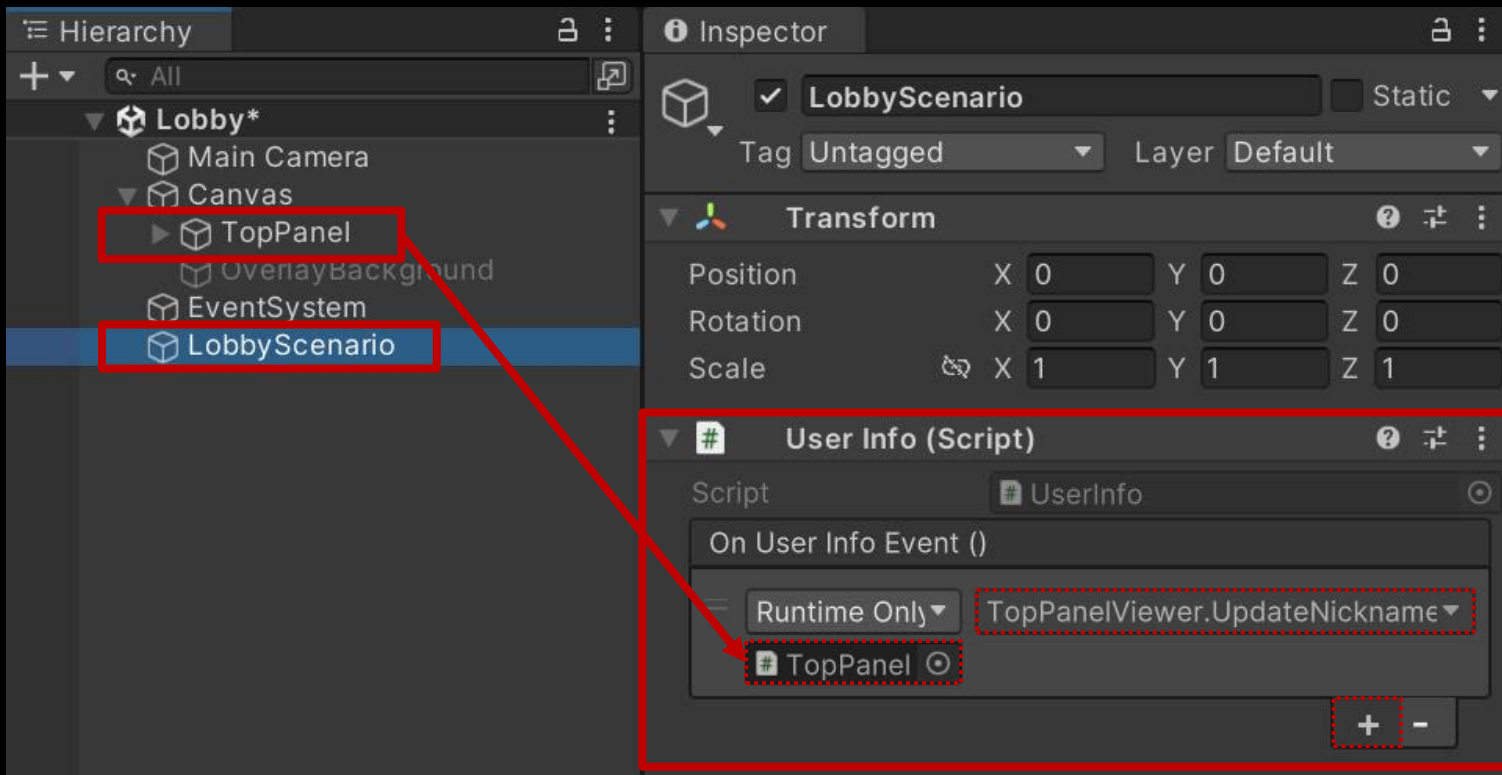
- TopPanel 오브젝트에 "TopPanelViewer" 컴포넌트 추가 및 설정





# 게임 유저 관리

- LobbyScenario 오브젝트에 "UserInfo" 컴포넌트 추가 및 설정
  - GameObject - Create Empty



UserInfo.GetUserInfoFromBackend() 메소드를 호출해 서버로부터 유저 정보를 불러오면 TopPanel에 있는 Nickname UI에 닉네임 정보를 출력합니다.



# 게임 유저 관리

- Lobby 씬에서 정보 갱신 등을 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "LobbyScenario"로 변경

```
1 using UnityEngine;
2
3 public class LobbyScenario : MonoBehaviour
4 {
5     [SerializeField]
6     private UserInfo user;
7
8     private void Awake()
9     {
10         user.GetUserInfoFromBackend();
11     }
12 }
```





# 게임 유저 관리

- LobbyScenario 오브젝트에 "LobbyScenario" 컴포넌트 추가 및 설정

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree view under 'Lobby\*'. The 'LobbyScenario' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties of the selected 'LobbyScenario (Script)'. The 'User' field is set to 'LobbyScenario (User Info)', which is also highlighted with a red dashed box. A red arrow points from the 'LobbyScenario' in the Hierarchy to the 'LobbyScenario (User Info)' in the Inspector. An 'Add Component' button is visible at the bottom of the Inspector panel.



# 게임 유저 관리

## ■ 결과 화면

로그인

아이디

비밀번호

로그인

아이디 찾기   계정 생성   비밀번호 찾기

로그인

아이디

비밀번호

로그인

아이디 찾기   계정 생성   비밀번호 찾기



# 게임 유저 관리

## ■ 닉네임 설정

- 플레이어 정보 팝업 윈도우 생성 및 설정
  - PopupBase 프리팹을 Hierarchy View로 Drag & Drop

PopupBase 오브젝트의 이름을 "PopupUpdateProfile"로 변경



# 게임 유저 관리

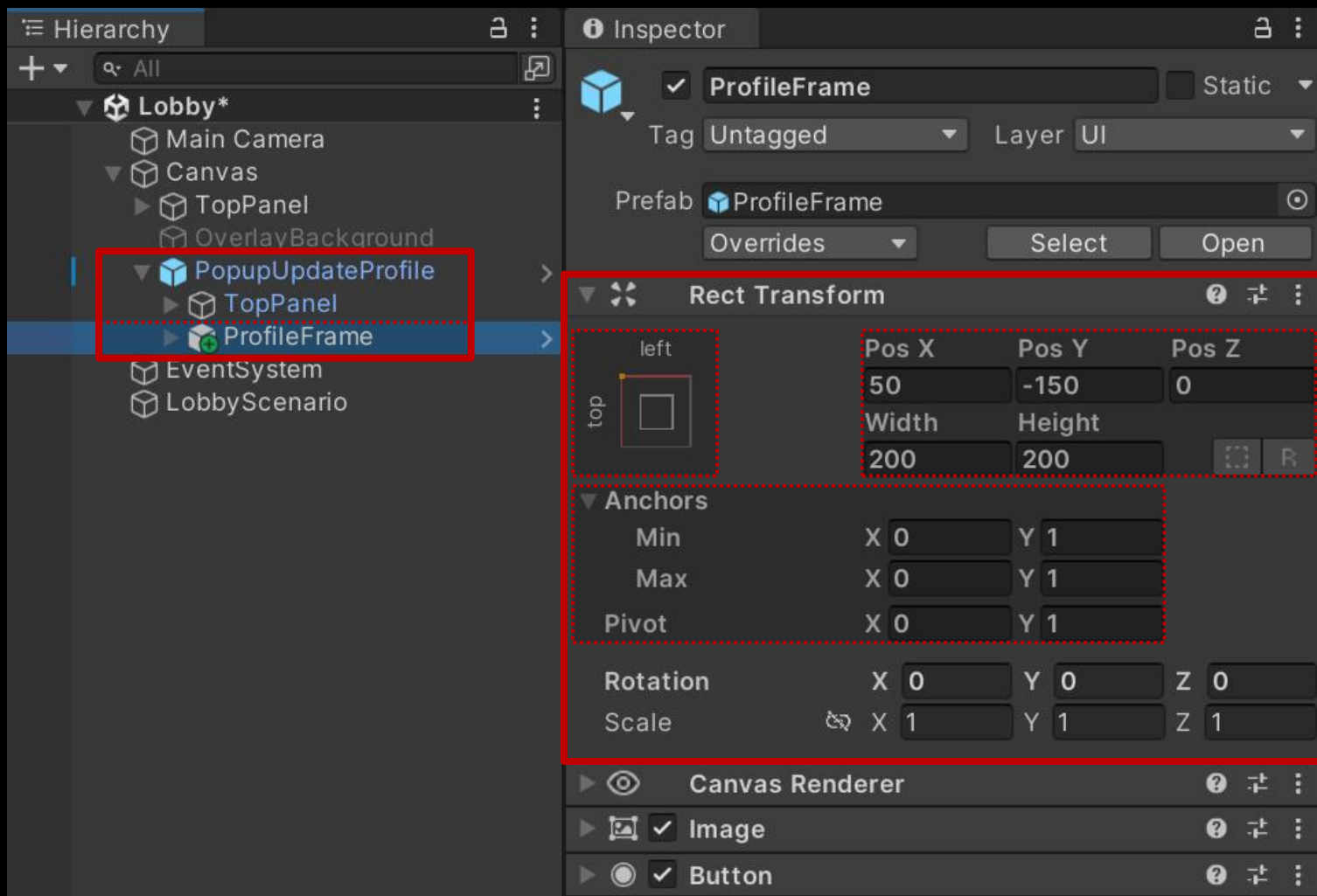
- 플레이어 정보 팝업 윈도우 상단 이름 설정

The image shows the Unity development environment. On the left, the Hierarchy panel displays a scene structure. Under the 'Lobby\*' hierarchy, there is a 'Canvas' containing 'TopPanel', 'OverlayBackground', and 'PopupUpdateProfile'. Inside 'PopupUpdateProfile', there is another 'TopPanel' which contains a 'Title' object. The 'Title' object is highlighted with a red box. On the right, the Inspector panel shows the properties for the selected 'TextMeshPro - Text (UI)' component. The 'Text Input' field contains the Korean text '플레이어 정보'. Below this, various text settings are visible, including 'Text Style' set to 'Normal', 'Font Asset' set to 'NotoSansKR-Bold SDF (TMP\_...', 'Material Preset' set to 'NotoSansKR-Bold SDF Material', 'Font Style' with buttons for 'B', 'I', 'U', 'S', 'ab', 'AB', 'SC', 'Font Size' set to '40', and 'Vertex Color' set to white.



# 게임 유저 관리

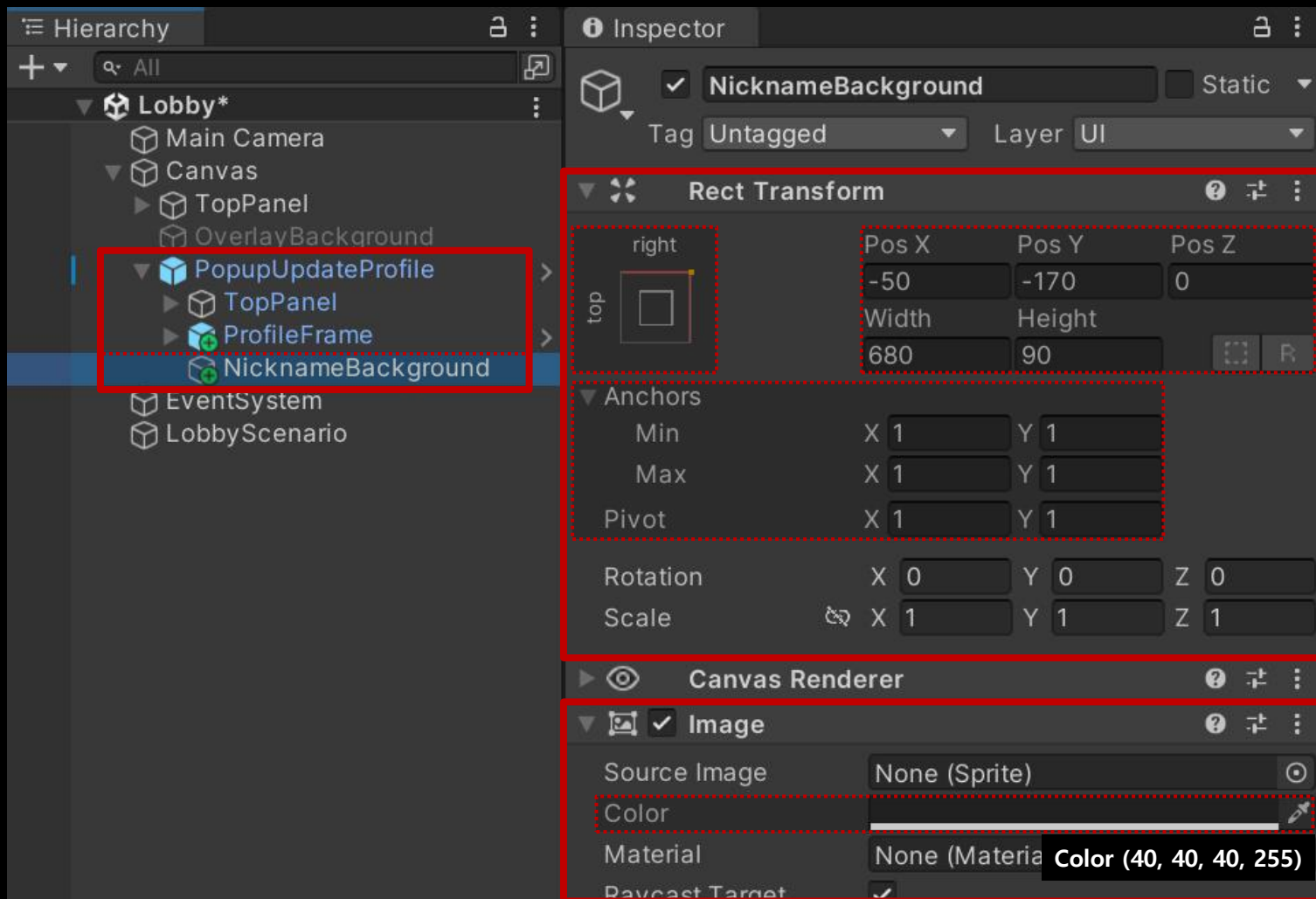
- 프로필 아이콘 생성 및 설정
  - ProfileFrame 프리팹을 Hierarchy View로 Drag & Drop





# 게임 유저 관리

- 닉네임 배경 Image UI 생성 및 설정
  - GameObject - UI - Image





# 게임 유저 관리

- 닉네임을 출력하는 "Text - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Text - TextMeshPro"

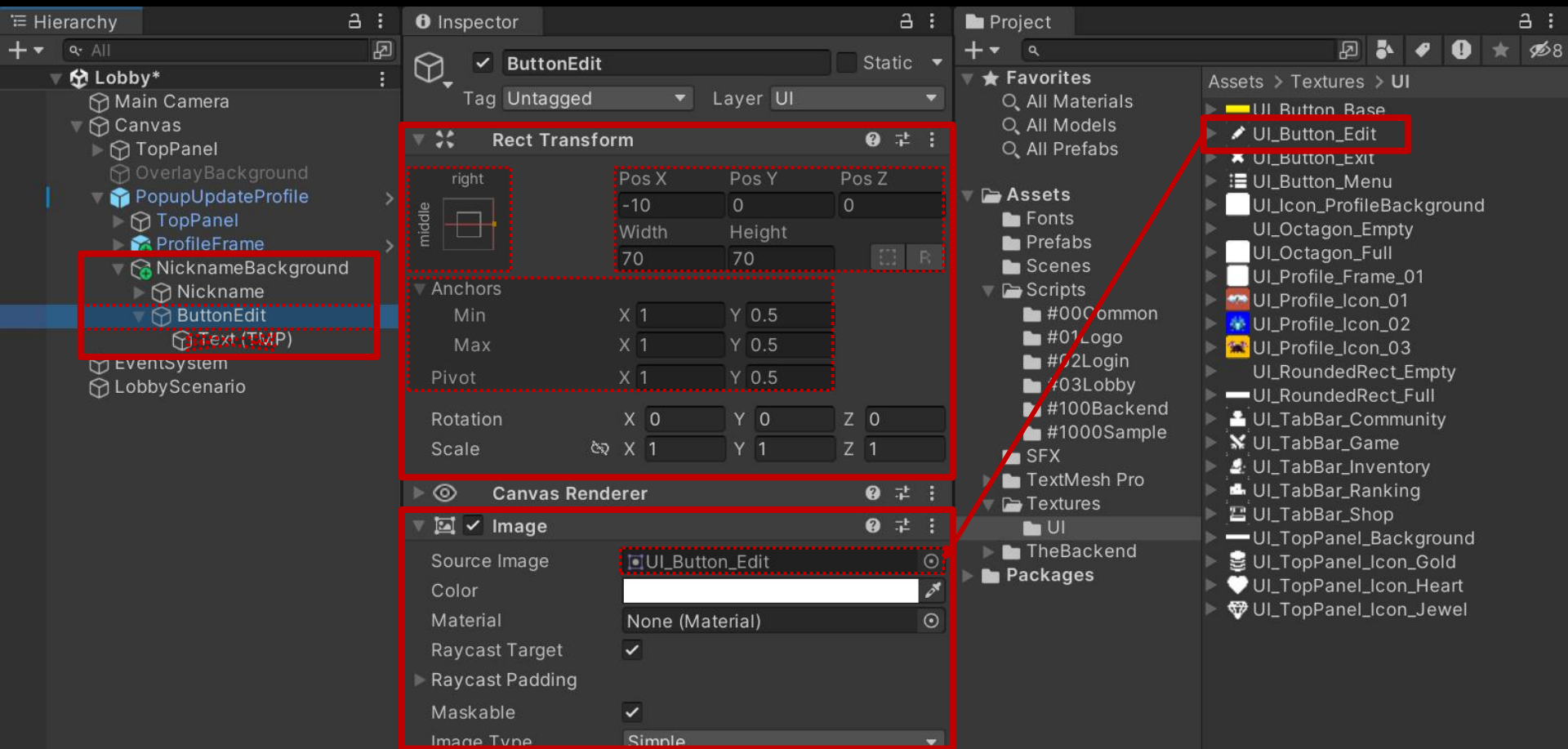
The screenshot shows the Unity development environment. On the left, the Hierarchy panel displays a tree structure under 'Lobby\*'. The 'NicknameBackground' object is selected, and its 'Nickname' child is highlighted with a red box. The Inspector panel on the right shows the 'TextMeshPro - Text (UI)' component. The 'Rect Transform' section is expanded, showing a 'stretch' handle and a table of values: Left (10), Top (0), Pos Z (0), Right (90), and Bottom (0). The 'Anchors' section shows Min (X: 0, Y: 0), Max (X: 1, Y: 1), and Pivot (X: 0.5, Y: 0.5). The 'Canvas Renderer' and 'TextMeshPro - Text (UI)' sections are also visible.

The screenshot shows the 'TextMeshPro - Text (UI)' Inspector panel. The 'Text Input' field contains the Korean text '닉네임'. The 'Text Style' is set to 'Normal'. The 'Main Settings' section includes 'Font Asset' (NotoSansKR-Regular SDF (TMF)), 'Material Preset' (NotoSansKR-Regular SDF Material), 'Font Style' (B, I, U, S, ab, AB, SC), and 'Font Size' (36). The 'Vertex Color' is set to white. The 'Spacing Options' are set to 0 for Character, Word, Line, and Paragraph. The 'Alignment' is set to 'Left'. The 'Wrapping' is set to 'Enabled'. The 'Overflow' is set to 'Ellipsis'. The 'Horizontal Mapping' and 'Vertical Mapping' are both set to 'Character'.



# 게임 유저 관리

- 닉네임 설정 "Button - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Button - TextMeshPro"







# 게임 유저 관리

- 아이디(gamer\_id)를 출력하는 "Text - TextMeshPro" UI 생성 및 설정
  - GameObject - UI - "Text - TextMeshPro"

Hierarchy: Lobby\* > Canvas > TopPanel > OverlayBackground > PopupUpdateProfile > ID

Inspector: Rect Transform

Property	Value
Pos X	-50
Pos Y	-270
Pos Z	0
Width	670
Height	60
Min X	1
Min Y	1
Max X	1
Max Y	1
Pivot X	1
Pivot Y	1
Rotation X	0
Rotation Y	0
Rotation Z	0
Scale X	1
Scale Y	1
Scale Z	1

TextMeshPro - Text (UI)

Material: NotoSansKR-Regular SDF Material

Shader: TextMeshPro/Distance F...

Text Input: This is Player ID.

Text Style: Normal

Main Settings

Font Asset: NotoSansKR-Regular SDF (TMF)

Material Preset: NotoSansKR-Regular SDF Material

Font Style: B I U S ab AB SC

Font Size: 30

Auto Size:

Vertex Color:

Color Gradient:

Override Tags:

Spacing Options (em): Character 0, Word 0, Line 0, Paragraph 0

Alignment: Left

Wrapping: Enabled

Overflow: Ellipsis

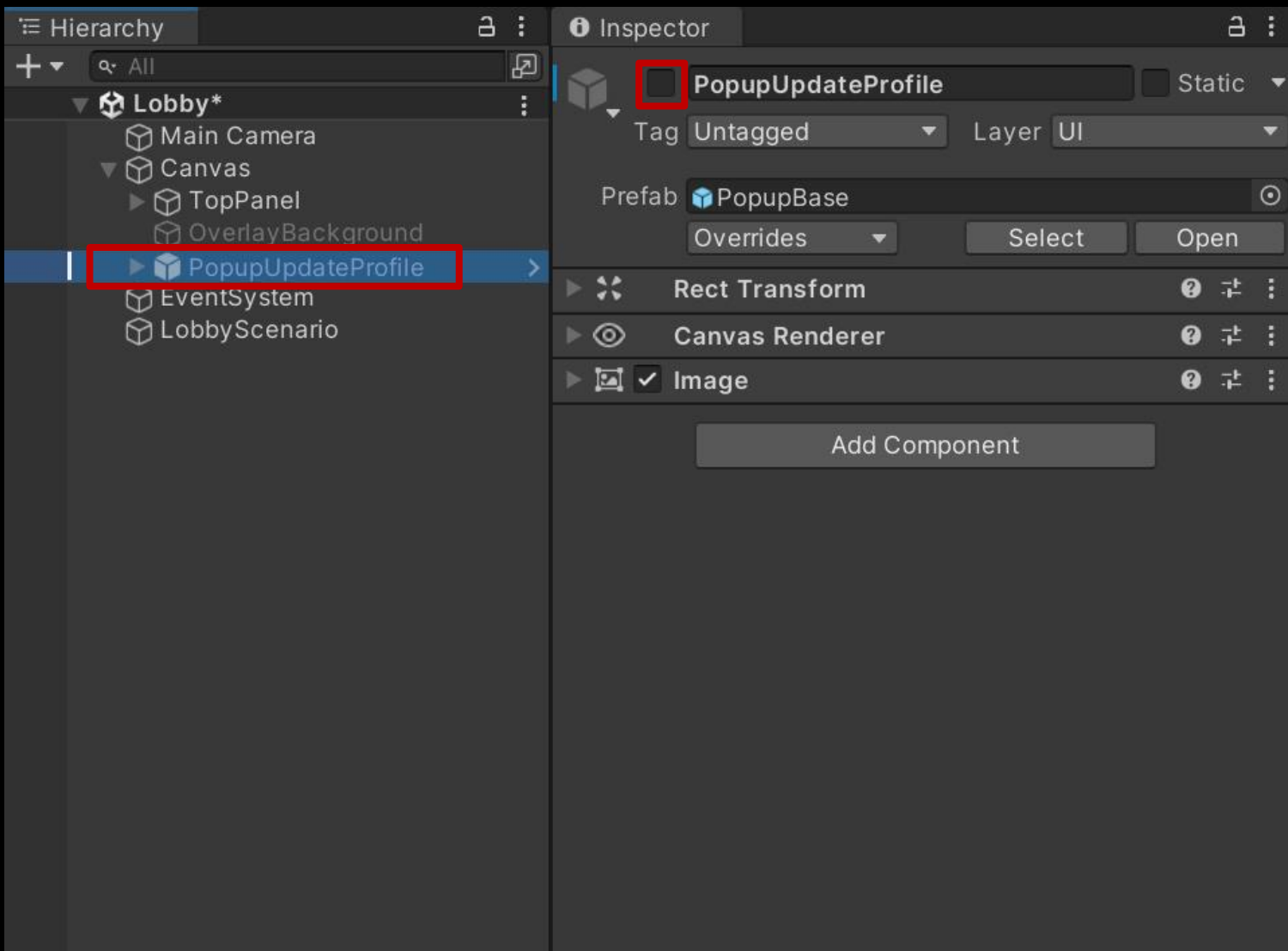
Horizontal Mapping: Character

Vertical Mapping: Character



# 게임 유저 관리

- PopupUpdateProfile 오브젝트 비활성화





# 게임 유저 관리

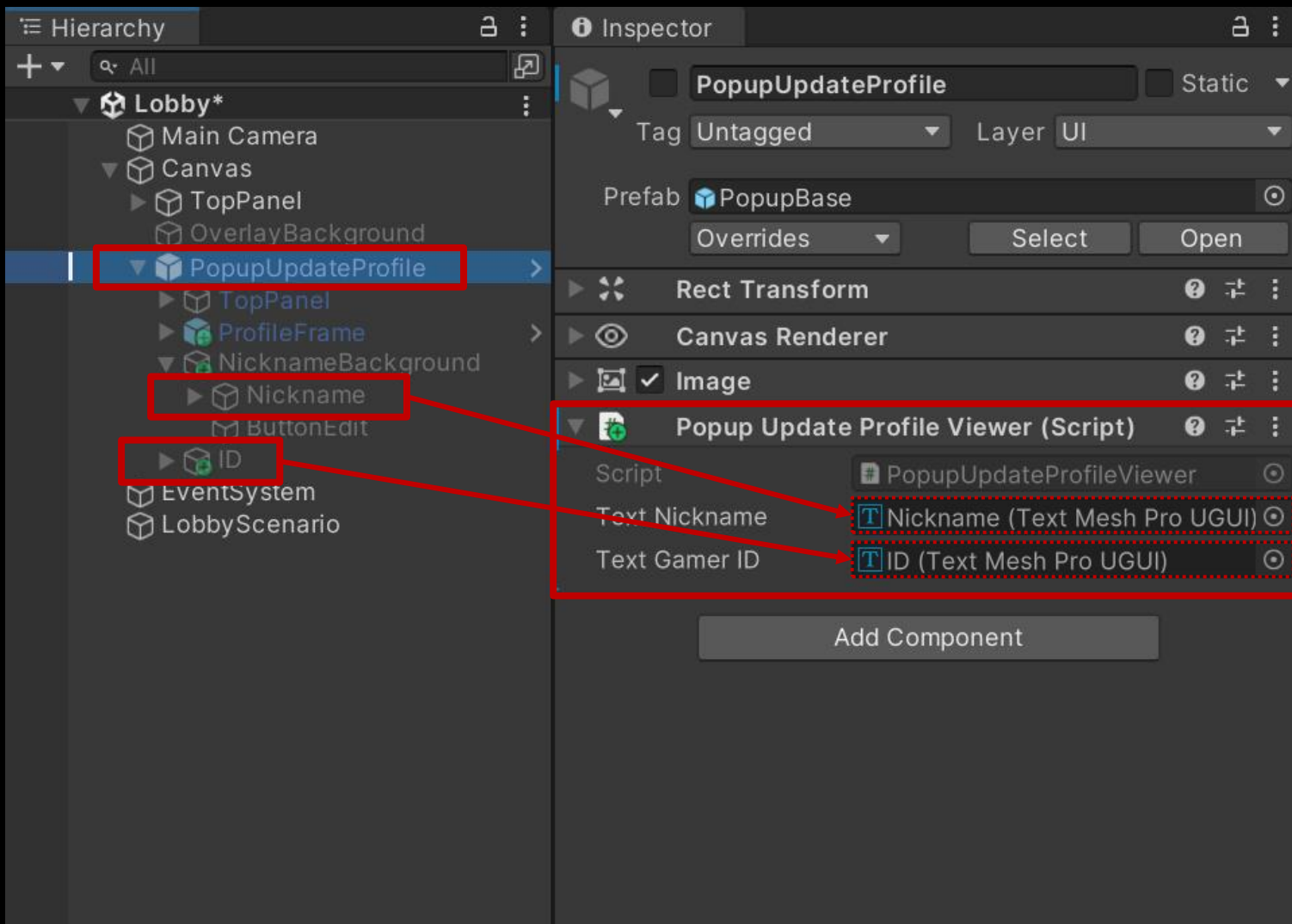
- PopupUpdateProfile에 출력하는 UI 정보를 제어하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "PopupUpdateProfileViewer"로 변경

```
1 using UnityEngine;
2 using TMPro;
3
4 public class PopupUpdateProfileViewer : MonoBehaviour
5 {
6     [SerializeField]
7     private TextMeshProUGUI textNickname;
8     [SerializeField]
9     private TextMeshProUGUI textGamerID;
10
11     public void UpdateNickname()
12     {
13         // 닉네임이 없으면 gamer_id를 출력하고, 닉네임이 있으면 닉네임 출력
14         textNickname.text = UserInfo.Data.nickname == null ?
15             UserInfo.Data.gamerId : UserInfo.Data.nickname;
16
17         // gamer_id 출력
18         textGamerID.text = UserInfo.Data.gamerId;
19     }
20 }
```



# 게임 유저 관리

- PopupUpdateProfile 오브젝트에 컴포넌트 추가 및 설정





# 게임 유저 관리

- LobbyScenario 오브젝트의 "UserInfo" 컴포넌트 변수 설정

The screenshot shows the Unity Inspector for the 'LobbyScenario' object. The 'UserInfo' script component is selected, and its 'On User Info Event ()' event is configured. The event is set to 'Runtime Only' and targets 'PopupUpdateProfileViewer.Update'. A red box highlights the 'UserInfo' component and its event configuration, with a red arrow pointing from the 'PopupUpdateProfile' object in the Hierarchy panel to the event configuration.

UserInfo.GetUserInfoFromBackend() 메소드를 호출해 서버로부터 유저 정보를 불러온 이후 PopupUpdateProfile에 있는 닉네임, gamer\_id를 출력



# 게임 유저 관리

## ■ ProfileFrame 오브젝트의 "Button" 컴포넌트 OnClick() 이벤트 설정

“프로필” 버튼을 누르면

OverlayBackground 오브젝트를 Canvas의 마지막 자식으로 설정한 후 활성화하고,  
PopupUpdateProfile 오브젝트를 Canvas의 마지막 자식으로 설정한 후 활성화한다.

OnClick ()

- Runtime Only | RectTransform.SetAsLastSibling() | OverlayBa | [x]
- Runtime Only | GameObject.SetActive() | OverlayBa | [x]
- Runtime Only | RectTransform.SetAsLastSibling() | PopupUpc | [x]
- Runtime Only | GameObject.SetActive() | PopupUpc | [x]



# 게임 유저 관리

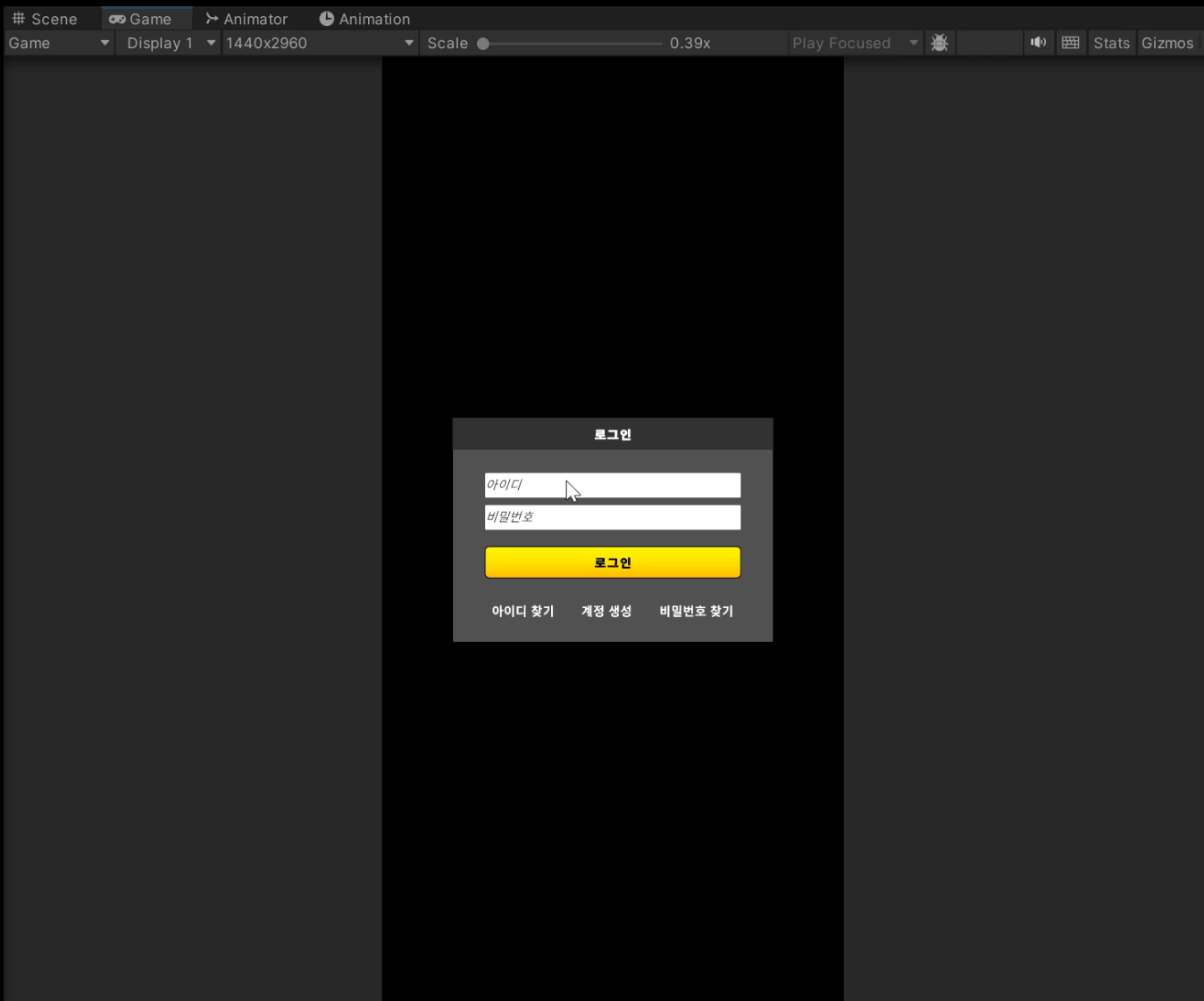
- Exit 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 설정

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Exit' object is selected, and its parent 'PopupUpdateProfile' is also highlighted. The Inspector panel shows the 'Button' component's configuration. The 'On Click ()' event list contains two entries, both set to 'Runtime Only' and 'GameObject.SetActive'. The first entry is linked to the 'OverlayBa' object, and the second entry is linked to the 'PopupUpc' object. Red dashed boxes highlight these event entries and their target objects. Red arrows point from the 'Exit' object in the Hierarchy to the 'OverlayBa' and 'PopupUpc' objects in the event list.



# 게임 유저 관리

## ■ 결과 화면







# 게임 유저 관리

- 닉네임 변경 팝업 윈도우 생성 및 설정
  - PopupBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot shows the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows a tree view with 'PopupUpdateNickname' selected. The Inspector panel on the right shows the properties of the selected object, including its name, tag, layer, and transform properties.

**Hierarchy View:**

- Lobby\*
  - Main Camera
  - Canvas
    - TopPanel
    - OverlayBackground
    - PopupUpdateProfile
    - PopupUpdateNickname**
    - TopPanel
      - Title
      - Exit
  - EventSystem
  - LobbyScenario

**Inspector View:**

- Name: **PopupUpdateNickname**
- Static:
- Tag: Untagged
- Layer: UI
- Prefab: PopupBase
- Overrides:
- Rect Transform**
  - Pivot: center
  - Pos X: 0, Pos Y: 0, Pos Z: 0
  - Width: 800, Height: 500
  - Anchors**
    - Min: X 0.5, Y 0.5
    - Max: X 0.5, Y 0.5
    - Pivot: X 0.5, Y 0.5
  - Rotation: X 0, Y 0, Z 0
  - Scale: X 1, Y 1, Z 1
- Canvas Renderer
- Image
- Default UI Material (Material)

**Text Box:**

PopupBase 오브젝트의 이름을 "PopupUpdateNickname"으로 변경



# 게임 유저 관리

- 닉네임 변경 팝업 윈도우 상단 이름 설정

The image shows the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Title' object under the 'PopupUpdateNickname' hierarchy is selected and highlighted with a red box. The Inspector panel shows the properties for the selected 'TextMeshPro - Text (UI)' component. The text content is '닉네임 변경' (Nickname Change), which is also enclosed in a red dashed box. The font asset is set to 'NotoSansKR-Bold SDF (TMP\_Fc)' and the font size is 40.

**Hierarchy Panel:**

- Lobby\*
  - Main Camera
  - Canvas
    - TopPanel
      - OverlayBackground
      - PopupUpdateProfile
      - PopupUpdateNickname
        - TopPanel
          - Title**
          - Exit
- EventSystem
- LobbyScenario

**Inspector Panel:**

- Title** (checked) Static (unchecked)
- Tag: Untagged Layer: UI
- Rect Transform
- Canvas Renderer
- T** **TextMeshPro - Text (UI)** (checked)
- Text Input Enable RTL Editor (unchecked)
- 닉네임 변경
- Text Style: Normal
- Main Settings**
- Font Asset: **F** NotoSansKR-Bold SDF (TMP\_Fc)
- Material Preset: NotoSansKR-Bold SDF Material
- Font Style: **B** I U S ab AB SC
- Font Size: 40
- Auto Size: (unchecked)
- Vertex Color: (white)
- Color Gradient: (unchecked)



# 게임 유저 관리

- “닉네임을..” 텍스트를 출력하는 “Text - TextMeshPro” UI 생성 및 설정
  - GameObject - UI - “Text - TextMeshPro”

Hierarchy

- Lobby\*
- Main Camera
- Canvas
  - TopPanel
  - OverlayBackground
  - PopupUpdateProfile
  - PopupUpdateNickname**
  - TopPanel
  - TextMessage
- EventSystem
- LobbyScenario

Inspector

TextMessage

Tag Untagged Layer UI

Rect Transform

stretch	Left	Pos Y	Pos Z
	0	80	0
	Right	Height	
	0	100	

Anchors

Min	X	0	Y	0.5
Max	X	1	Y	0.5
Pivot	X	0.5	Y	0.5

Rotation

X	0	Y	0	Z	0
---	---	---	---	---	---

Scale

X	1	Y	1	Z	1
---	---	---	---	---	---

Canvas Renderer

TextMeshPro - Text (UI)

NotoSansKR-Bold SDF Material (Material)

Shader TextMeshPro/Distance Fi

Add Component

TextMeshPro - Text (UI)

Text Input Enable RTL Editor

닉네임을 입력하세요

Text Style Normal

Main Settings

Font Asset NotoSansKR-Bold SDF (TMP\_Fc)

Material Preset NotoSansKR-Bold SDF Material

Font Style B I U S ab AB SC

Font Size 36

Auto Size

Vertex Color

Color Gradient

Override Tags

Spacing Options (em) Character 0 Word 0 Line 0 Paragraph 0

Alignment

Wrapping Enabled

Overflow Overflow

Horizontal Mapping Character

Vertical Mapping Character



# 게임 유저 관리

- 닉네임 입력 필드 생성 및 설정
  - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop

InputFieldBase 오브젝트의 이름을 "Nickname"으로 변경

Property	Value
Pos X	0
Pos Y	-30
Pos Z	0
Width	600
Height	80
Min X	0.5
Min Y	0.5
Max X	0.5
Max Y	0.5
Pivot X	0.5
Pivot Y	0.5
Rotation X	0
Rotation Y	0
Rotation Z	0
Scale X	1
Scale Y	1
Scale Z	1



# 게임 유저 관리

- “닉네임 변경” 버튼 생성 및 설정
  - ButtonBase 프리팹을 Hierarchy View로 Drag & Drop

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Lobby\*'. A red box highlights the 'PopupUpdateNickname' object, which contains 'TopPanel', 'TextMessage', 'Nickname', and 'ButtonOK'. The 'ButtonOK' object is selected. On the right, the Inspector panel shows the 'ButtonOK' component. The 'Prefab' is set to 'ButtonBase'. Below this, the 'Rect Transform' component is visible, with its properties: Pos X (0), Pos Y (30), Pos Z (0), Width (300), and Height (100). The 'Anchors' section shows Min (X 0.5, Y 0) and Max (X 0.5, Y 0). The 'Pivot' is set to (X 0.5, Y 0). The 'Rotation' is (X 0, Y 0, Z 0) and the 'Scale' is (X 1, Y 1, Z 1). A black box at the bottom left contains the text: 'ButtonBase 오브젝트의 이름을 "ButtonOK"로 변경'.

ButtonBase 오브젝트의 이름을  
"ButtonOK"로 변경



# 게임 유저 관리

- “닉네임 변경” 버튼 생성 및 설정 (계속)

The image shows the Unity development environment. On the left is the Hierarchy panel, and on the right is the Inspector panel.

**Hierarchy Panel:** Shows a tree structure under 'Lobby\*'. The 'Text (TMP)' object is highlighted with a red box. Other objects include Main Camera, Canvas, TopPanel, OverlayBackground, PopupUpdateProfile, PopupUpdateNickname, TextMessage, Nickname, ButtonOK, EventSystem, and LobbyScenario.

**Inspector Panel:** Shows the properties for the selected 'Text (TMP)' object. The 'TextMeshPro - Text (UI)' component is expanded and highlighted with a red box. The text content is 'OK'. The 'Font Size' property is set to 60 and is also highlighted with a red dashed box. Other visible properties include Text Style (Normal), Font Asset (NotoSansKR-Bold SDF), Material Preset (NotoSansKR-BoId SDF Material), and Font Style (B, I, U, S, ab, AB, SC).



# 게임 유저 관리

- PopupUpdateNickname 오브젝트 비활성화

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Lobby\*'. The 'PopupUpdateNickname' object is selected and highlighted with a red box. On the right, the Inspector panel shows the 'PopupUpdateNickname' component. The 'Active' checkbox is unchecked, and this checkbox is also highlighted with a red box. Below the component name, the 'Tag' is set to 'Untagged' and the 'Layer' is 'UI'. The 'Prefab' is set to 'PopupBase'. Below these settings, the 'Rect Transform', 'Canvas Renderer', and 'Image' components are listed. The 'Image' component has a checkmark next to it. At the bottom of the Inspector panel, there is an 'Add Component' button.



# 게임 유저 관리

- 뒤끝 서버와 연동해 닉네임을 변경하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "Nickname"으로 변경

```
1 using UnityEngine;
2 using UnityEngine.UI;
3 using TMPro;
4 using BackEnd;
5
6 public class Nickname : LoginBase
7 {
8     [System.Serializable]
9     public class NicknameEvent : UnityEngine.Events.UnityEvent { }
10    public NicknameEvent onNicknameEvent = new NicknameEvent();
11
12    [SerializeField]
13    private Image imageNickname; // 닉네임 필드 색상 변경
14    [SerializeField]
15    private TMP_InputField inputFieldNickname; // 닉네임 필드 텍스트 정보 추출
16
17    [SerializeField]
18    private Button btnUpdateNickname; // "닉네임 설정" 버튼 (상호작용 가능/불가능)
19
```





# 게임 유저 관리

- 뒤끝 서버와 연동해 닉네임을 변경하는 스크립트 생성 및 작성 (계속)

```
20 private void OnEnable()
21 {
22     // 닉네임 변경에 실패해 에러 메시지를 출력한 상태에서
23     // 닉네임 변경 팝업을 닫았다가 열 수 있기 때문에 상태를 초기화
24     ResetUI(imageNickname);
25     SetMessage("닉네임을 입력하세요");
26 }
27
28 public void OnClickUpdateNickname()
29 {
30     // 매개변수로 입력한 InputField UI의 색상과 Message 내용 초기화
31     ResetUI(imageNickname);
32
33     // 필드 값이 비어있는지 체크
34     if ( IsFieldDataEmpty(imageNickname, inputFieldNickname.text, "Nickname") ) return;
35
36     // "닉네임 변경" 버튼의 상호작용 비활성화
37     btnUpdateNickname.interactable = false;
38     SetMessage("닉네임 변경중입니다..");
39
40     // 뒤끝 서버 닉네임 변경 시도
41     UpdateNickname();
42 }
43
```



# 게임 유저 관리

- 뒤끝 서버와 연동해 닉네임을 변경하는 스크립트 생성 및 작성 (계속)

```
44 private void UpdateNickname()  
45 {  
46     // 닉네임 설정  
47     Backend.BMember.UpdateNickname(inputFieldNickname.text, callback =>  
48     {  
49         // "닉네임 변경" 버튼의 상호작용 활성화  
50         btnUpdateNickname.interactable = true;  
51  
52         // 닉네임 변경 성공  
53         if ( callback.IsSuccess() )  
54         {  
55             SetMessage($"{inputFieldNickname.text}(으)로 닉네임이 변경되었습니다.");  
56  
57             // 닉네임 변경에 성공했을 때 onNicknameEvent에 등록되어 있는 이벤트 메소드 호출  
58             onNicknameEvent?.Invoke();  
59         }  
60         // 닉네임 변경 실패  
61         else ...  
80     });  
81 }  
82 }
```

뒷장



# 게임 유저 관리

- 뒤끝 서버와 연동해 닉네임을 변경하는 스크립트 생성 및 작성 (계속)

```
60 // 닉네임 변경 실패
61 else
62 {
63     string message = string.Empty;
64
65     switch ( int.Parse(callback.GetStatusCode()) )
66     {
67         case 400: // 빈 닉네임 혹은 string.Empty, 20자 이상의 닉네임, 닉네임 앞/뒤에 공백이 있는 경우
68             message = "닉네임이 비어있거나 | 20자 이상 이거나 | 앞/뒤에 공백이 있습니다.";
69             break;
70         case 409: // 이미 중복된 닉네임이 있는 경우
71             message = "이미 존재하는 닉네임입니다.";
72             break;
73         default:
74             message = callback.GetMessage();
75             break;
76     }
77
78     GuideForIncorrectlyEnteredData(imageNickname, message);
79 }
```



# 게임 유저 관리

- PopupUpdateNickname 오브젝트에 "Nickname" 컴포넌트 추가 및 설정

The screenshot displays the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows the 'Lobby\*' scene structure, with 'PopupUpdateNickname' selected. The Inspector panel on the right shows the properties of the selected object, including the 'Nickname (Script)' component. Red boxes and arrows highlight the configuration steps:

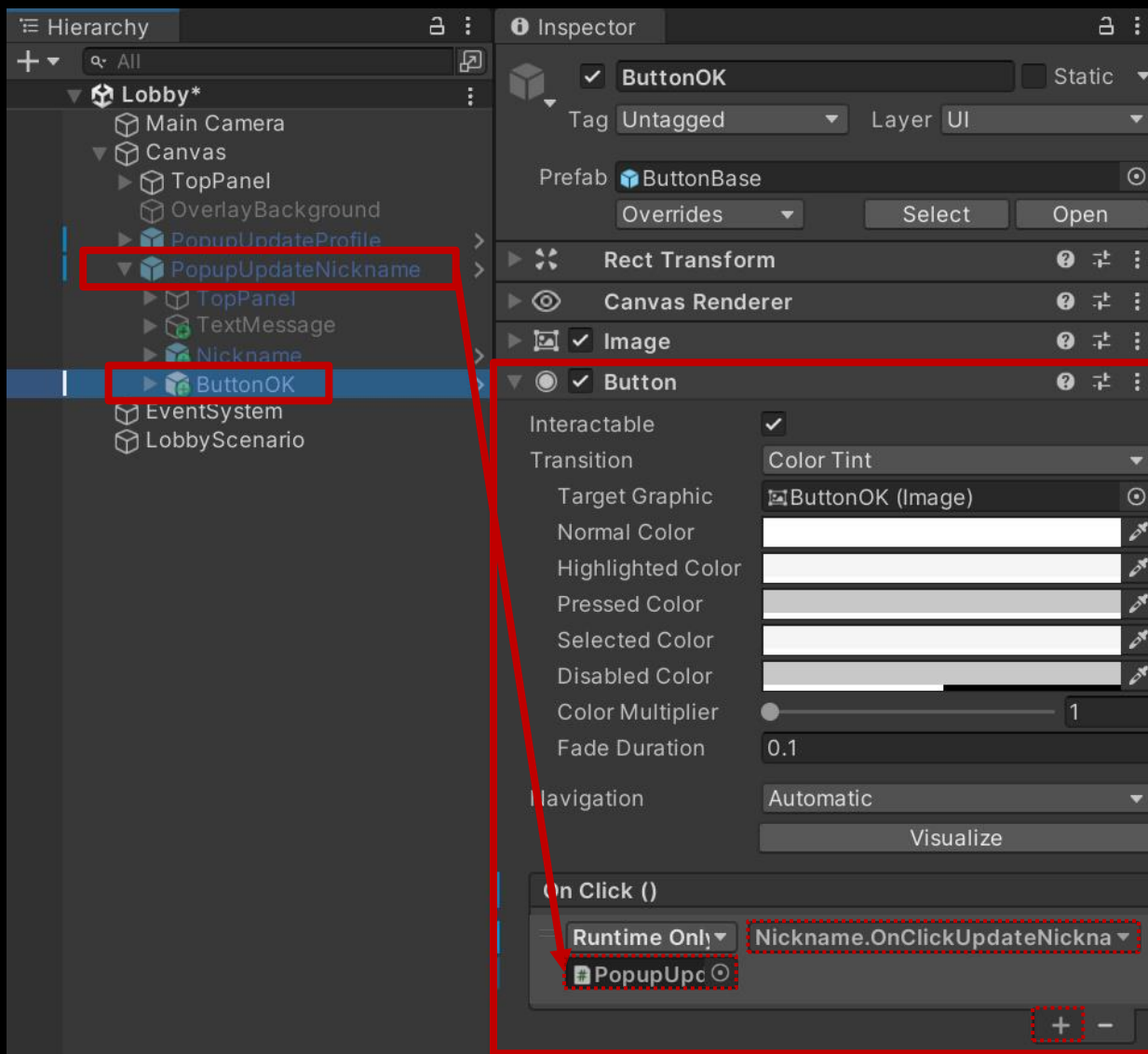
- Text Message:** Set to 'TextMessage (Text Mesh Pro UC)'.
- On Nickname Event ():** Set to 'Runtime Only' and 'UserInfo.GetUserInfoFromBacken'.
- Image Nickname:** Set to 'Nickname (Image)'.
- Input Field Nickname:** Set to 'Nickname (TMP\_Input Field)'.
- Btn Update Nickname:** Set to 'ButtonOK (Button)'.

The 'Add Component' button is visible at the bottom of the Inspector panel.



# 게임 유저 관리

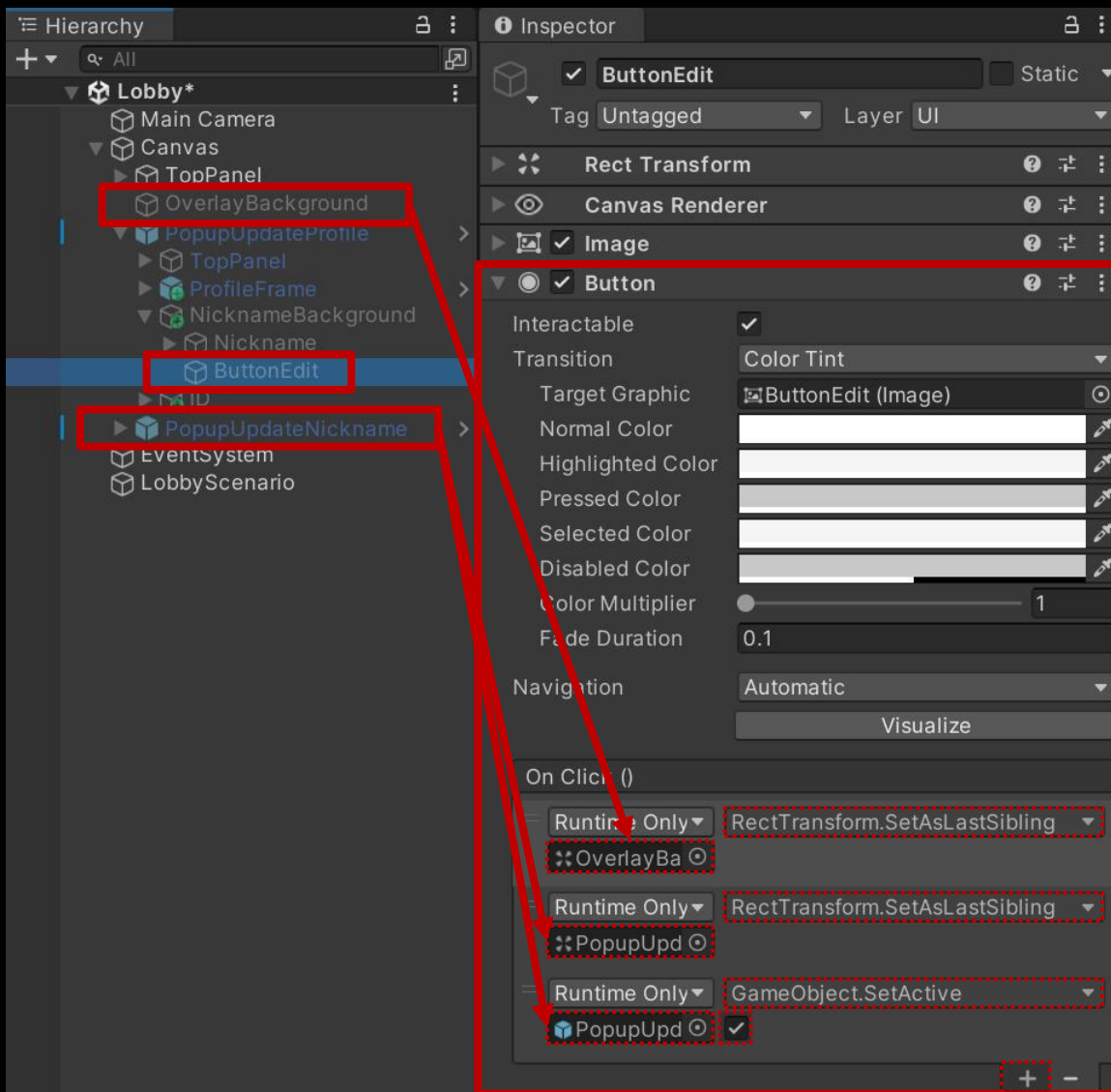
- ButtonOK 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 설정





# 게임 유저 관리

- ButtonEdit 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 설정





# 게임 유저 관리

- Exit 오브젝트의 "Button" 컴포넌트 OnClick() 이벤트 설정

The screenshot displays the Unity Inspector window for an 'Exit' button component. The Hierarchy panel on the left shows the 'Exit' object selected under the 'Lobby\*' scene. The Inspector panel on the right shows the 'Button' component's settings. The 'On Click ()' section is expanded, showing two event entries:

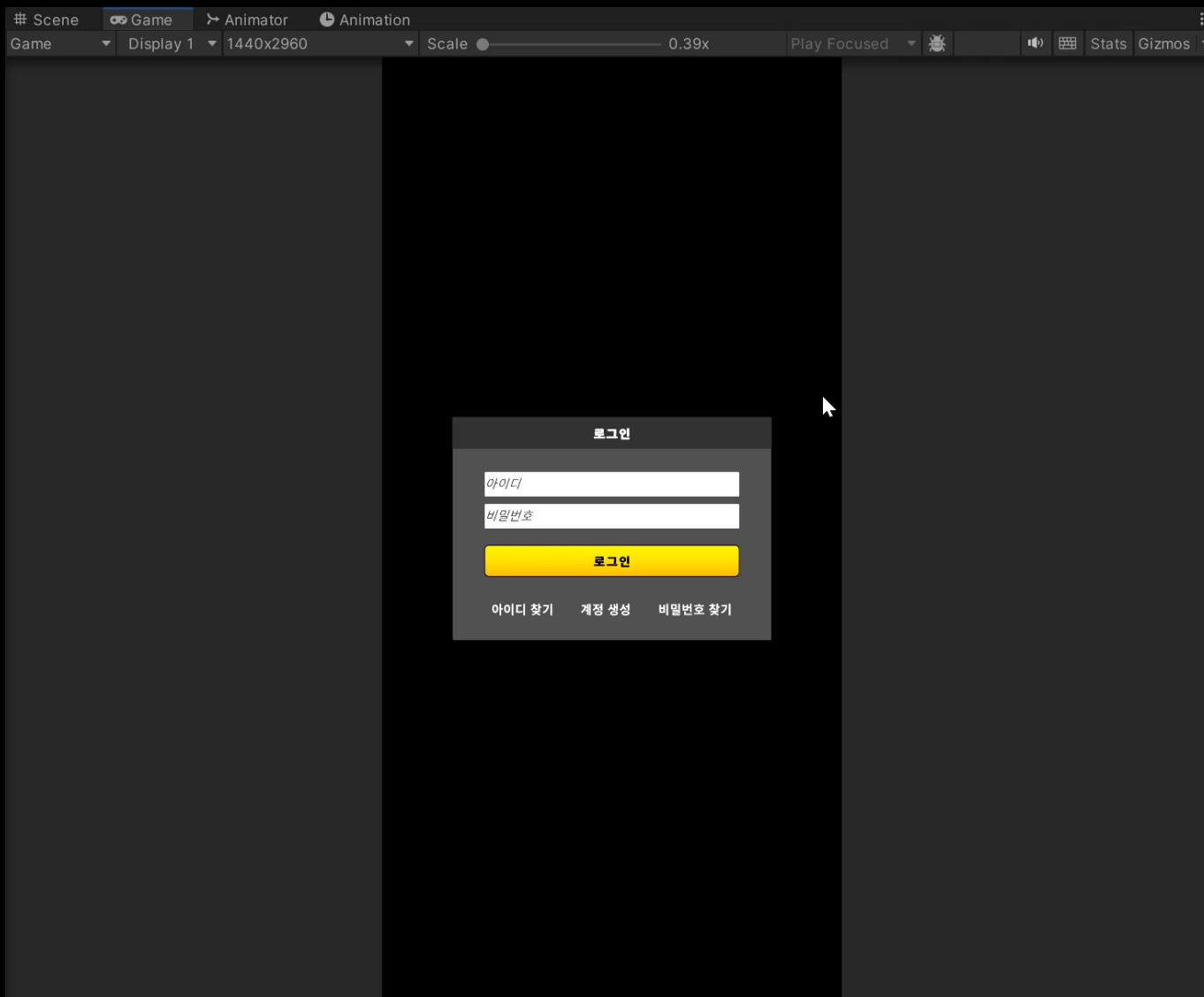
- Event 1: Runtime Only, Target: PopupUpc, Method: GameObject.SetActive
- Event 2: Runtime Only, Target: PopupUpc, Method: RectTransform.SetAsLastSibling

Red dashed boxes highlight the 'Runtime Only' dropdowns, the 'PopupUpc' target objects, and the method names. Red arrows point from the 'Exit' object in the Hierarchy to the 'Image' and 'Button' components in the Inspector, and from the 'PopupUpc' targets in the event list back to the 'Exit' object in the Hierarchy.



# 게임 유저 관리

## ■ 결과 화면







# 게임 유저 관리

## ■ 결과 화면

Backnd Console << ProjectA [Settings] [Notifications] [Help] [0] 관리자 [Profile]

★ 유저 관리 [Game User Gen] [Delete]

SDK 문서 [Console Guide]

대상: 회원번호 [Search] [Reset Search]

상세 검색 ▶

**로그인에 사용하는 아이디** → 회원번호

**로그인하면 접속 일시를 기록** → 최종 접속일

번호	회원번호	회원 아이디	닉네임	가입일	최종 접속일	국가	접속 OS
1	<a href="#">8715cf60-9d88-11ed-891a-a3bb595a3744</a>	user06	고박사2	2023.01.26 23:48	2023.01.27 16:15	-	Windows 10 &#4010...
2	<a href="#">570df900-9d88-11ed-931d-433b52bcbe82</a>	user05	고박사3	2023.01.26 23:47	2023.01.27 16:15	-	Windows 10 &#4010...
3	<a href="#">5ae8b8b0-9cb8-11ed-aa09-15179ce39577</a>	user03	-	2023.01.25 22:58	2023.01.25 22:58	-	Windows 10 &#4010...
4	<a href="#">06c1e7c0-9cb8-11ed-91ff-21ce2c0cebaa</a>	user02	고박사	2023.01.25 22:56	2023.01.27 15:54	-	Windows 10 &#4010...

1 > [10개씩 보기]

**gamer\_id** → 회원번호

**닉네임을 생성하거나 변경했을 때** → 닉네임