

Unity Backend

“친구” 기능을 이용한 친구 요청 및 친구 요청 대기 목록

Created in 2023-06-30
Last Updated 2023-06-30
Unity Version 2022.2.2f1

Index

- ◆ 친구 요청
- ◆ 친구 요청 대기 목록

친구 요청

- 소셜 설정
- 친구 요청
- 커뮤니티 - 친구 요청 탭 제작



친구 요청

■ 소셜 설정

■ Backend Console에서 친구 보유 수 설정

Backend Console

ProjectA

관리자

ProjectA **개발 모드**

- 즐거찾기
- 대시보드
- 서버 설정
- 프로젝트
- 인증 정보
- 푸시
- 스토어 정보
- 소셜**
- 예산 알림
- 요금제
- 뒤끝베이스
- 유저 관리
- 유저 접근 관리
- 게임 정보 관리
- 랭킹 관리
- 우편 관리
- 푸시 관리
- 쿠폰 관리

★ 소셜

콘솔 가이드

친구	최대 보유 수	0	변경
쪽지	최대 보유 수	0	변경
	글자 수 제한	0	변경
실시간 알림	기능 소개	친구, 쪽지, 우편, 길드 등 게임 내 유저 간 특정 상황 발생 시 실시간으로 확인이 가능합니다.	
	사용 방법	자세한 사용 방법은 개발문서를 참고해 주세요.	
		<button>실시간 알림 활성화</button>	



친구 요청

■ Backend Console에서 친구 보유 수 설정 (계속)

Backend Console << ProjectA [Settings] [50 관리자]

ProjectA [개발 모드]

- 즐거찾기
- 대시보드
- 서버 설정
- 프로젝트
- 인증 정보
- 푸시
- 스토어 정보
- 소셜**
- 예산 알림
- 요금제
- 뒤끝베이스
- 유저 관리
- 유저 접근 관리
- 게임 정보 관리
- 랭킹 관리
- 우편 관리
- 푸시 관리
- 쿠폰 관리
- 차트 관리

콘솔 가이드

친구	최대 보유 수	<input type="text" value="100"/>	<input type="button" value="적용"/>
쪽지	최대 보유 수	<input type="text" value="0"/>	<input type="button" value="변경"/>
	글자 수 제한	<input type="text" value="0"/>	<input type="button" value="변경"/>

실시간 알림

- 기능 소개: 친구, 쪽지, 우편, 길드 등 게임 내 유저 간 특정 상황 발생 시 실시간으로 확인이 가능합니다.
- 사용 방법: 자세한 사용 방법은 [개발문서](#)를 참고해 주세요.

회사소개 | 이용약관 | 서비스수준협약 | 개인정보처리방침

© AFI, Inc. All rights reserved.



친구 요청

■ 친구 요청

■ 유저 정보 조회 메소드

```
GetUserInfoByNickName(string nickname);
```

해당 닉네임(nickname)을 가지고 있는 유저의 유저정보 조회

■ 조회된 유저 정보 JsonData

```
{
  "row":
  {
    "nickname": "고박사" // 유저의 닉네임
    "inDate": "2023-06-26T02:22:22.022Z" // 유저의 inDate
    "lastLogin": "2023-06-26T02:22:22.022Z" // 유저의 마지막 로그인 시각
    "guildName": "UnityNote" // 유저의 길드명 (없으면 null)
  }
}
```



친구 요청

- 서버와 통신해 친구 관련 시스템을 제어하는 스크립트 생성 및 작성
 - C# Script 생성 후 스크립트의 이름을 "BackendFriendSystem"으로 변경

```
1  using UnityEngine;
2  using System;
3  using BackEnd;
4
5  public class BackendFriendSystem : MonoBehaviour
6  {
7      private string GetUserInfoBy(string nickname)
8      {
9          // 해당 닉네임(nickname)의 유저가 존재하는지 여부는 동기로 진행
10         var bro = Backend.Social.GetUserInfoByNickName(nickname);
11         string inDate = string.Empty;
12
13         if ( !bro.IsSuccess() )
14         {
15             Debug.LogError($"유저 검색 도중 에러가 발생했습니다. : {bro}");
16             return inDate;
17         }
18     }
19 }
```



친구 요청

- 서버와 통신해 친구 관련 시스템을 제어하는 스크립트 생성 및 작성 (계속)

```
19 // JSON 데이터 파싱 성공
20 try
21 {
22     LitJson.JsonData jsonData = bro.GetFlattenJSON()["row"];
23
24     // 받아온 데이터의 개수가 0이면 데이터가 없는 것
25     if ( jsonData.Count <= 0 )
26     {
27         Debug.LogWarning("유저의 inDate 데이터가 없습니다.");
28         return inDate;
29     }
30
31     inDate = jsonData["inDate"].ToString();
32
33     Debug.Log($"{nickname}의 inDate 값은 {inDate} 입니다.");
34 }
35 // JSON 데이터 파싱 실패
36 catch ( Exception e )
37 {
38     // try-catch 에러 출력
39     Debug.LogError(e);
40 }
41
42 return inDate;
43 }
44
```




친구 요청

- 서버와 통신해 친구 관련 시스템을 제어하는 스크립트 생성 및 작성 (계속)

```
45 public void SendRequestFriend(string nickname)
46 {
47     // RequestFriend() 메소드를 이용해 친구 추가 요청을 할 때 해당 친구의 inDate 정보가 필요
48     string inDate = GetUserInfoBy(nickname);
49
50     // inDate 정보를 가진 유저에게 "친구 요청"을 보낸다.
51     Backend.Friend.RequestFriend(inDate, callback =>
52     {
53         if ( !callback.IsSuccess() )
54         {
55             Debug.LogError($"{nickname} 친구 요청 도중 에러가 발생했습니다. : {callback}");
56             return;
57         }
58
59         Debug.Log($"친구 요청에 성공했습니다. : {callback}");
60     });
61 }
62 }
```

== Return Error cases ==

403 : 뒤끝 콘솔의 소셜관리 메뉴에서 친구 최대보유수 설정값이 0인 경우
412

- [maxRequestFriend..] : 받는 사람의 request가 꽉 찬 경우

- [maxSendFriendRequest..] : 보내는 사람의 request가 꽉 찬 경우

409 : 이미 친구 요청한 사람에게 다시 요청한 경우



친구 요청

- BackendSystem 오브젝트에 "BackendFriendSystem" 컴포넌트 추가

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree view of the scene objects. The 'BackendSystem' object is selected and highlighted with a red box. On the right, the Inspector panel shows the properties of the selected object. The 'BackendSystem' component is active, and the 'Backend Friend System (Script)' component is highlighted with a red box. Below the component list, the 'Script' dropdown is set to 'BackendFriendSystem', and the 'Add Component' button is visible.



친구 요청

■ 커뮤니티 - 친구 요청 탭 제작

- 커뮤니티 페이지를 관리하는 Panel UI 생성 및 설정
 - GameObject - UI - Panel

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Lobby*'. The 'CommunityPage' object is selected and highlighted in blue. A red box highlights the 'CommunityPage' object in the Hierarchy. On the right, the Inspector panel shows the 'CommunityPage' component. The 'Rect Transform' component is expanded, and the 'stretch' option is selected for the Left, Top, Right, and Bottom properties. A red dashed box highlights the 'stretch' selection and the 'Rect Transform' component. Below the 'Rect Transform' component, the 'Canvas Renderer' and 'Image' components are visible. A red dashed box highlights the 'Canvas Renderer' and 'Image' components. The 'Add Component' button is visible at the bottom.



친구 요청

- 친구 요청 페이지를 관리하는 Panel UI 생성 및 설정
 - GameObject - UI - Panel

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Lobby*'. The 'CommunityPage' folder is expanded, and 'SentRequestPage' is selected. On the right, the Inspector panel shows the 'SentRequestPage' component. The 'Rect Transform' component is highlighted with a red dashed box, showing the 'stretch' option selected for both width and height. The 'Canvas Renderer' component is also highlighted with a red dashed box. The 'Image' component is visible below it. The 'Add Component' button is at the bottom.

Property	Value
Left	0
Top	250
Right	0
Bottom	0
Pos Z	0
Min X	0
Min Y	0
Max X	1
Max Y	1
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



친구 요청

- 친구 요청 UI를 관리하는 Panel UI 생성 및 설정
 - GameObject - UI - Panel

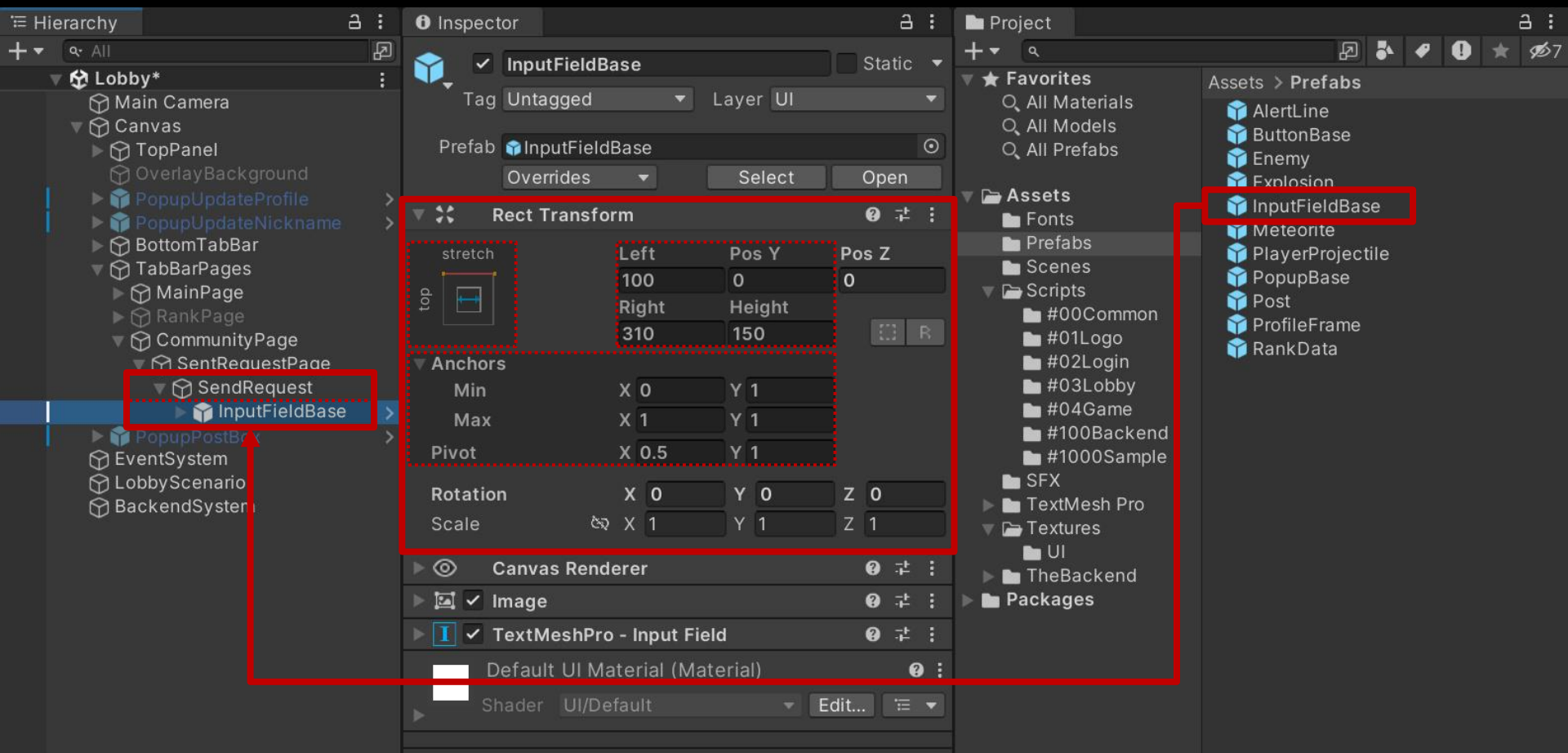
The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Lobby*'. The 'SentRequestPage' and 'SendRequest' objects are highlighted with a red box. On the right, the Inspector panel shows the 'SendRequest' component selected. The 'Rect Transform' component is highlighted with a red dashed box, showing its 'stretch' and 'top' anchors, and its position and scale values. The 'Canvas Renderer' component is also highlighted with a red dashed box, showing its 'Image' sub-component. The 'Add Component' button is visible at the bottom of the Inspector panel.

Property	Value
Tag	Untagged
Layer	UI
Left	0
Pos Y	-50
Pos Z	0
Right	0
Height	250
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



친구 요청

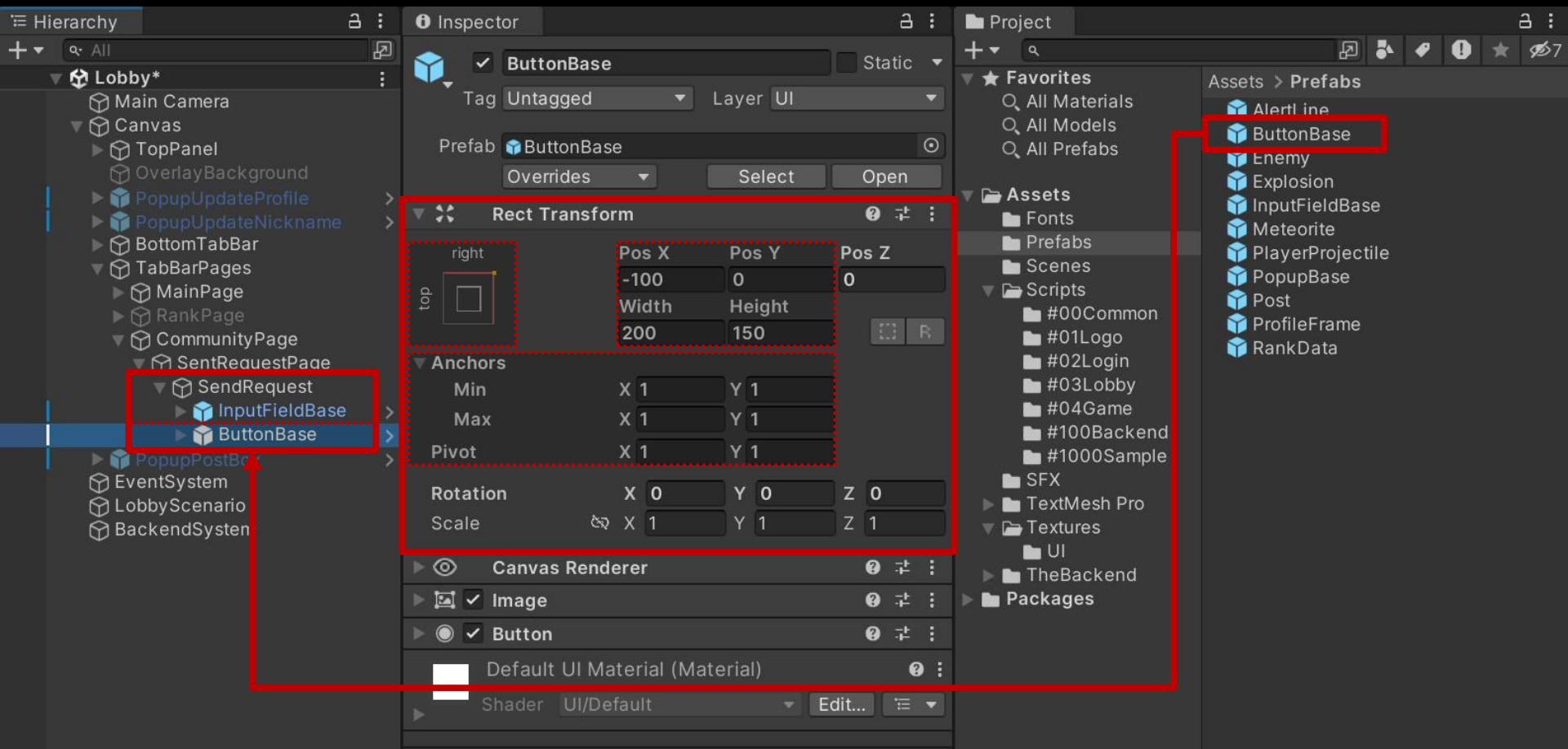
- 친구 닉네임 입력 필드 생성 및 설정
 - InputFieldBase 프리팹을 Hierarchy View로 Drag & Drop





친구 요청

- 친구 요청 Button UI 생성 및 설정
 - ButtonBase 프리팹을 Hierarchy View로 Drag & Drop





친구 요청

■ 친구 요청 Button UI 생성 및 설정 (계속)

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a scene named 'Lobby*' with various UI components. A 'Text (TMP)' object is highlighted with a red box. On the right, the Inspector panel shows the properties for the selected 'TextMeshPro - Text (UI)' object. The text content is '친구요청'. The font asset is set to 'NotoSansKR-Bold SDF (TMP_Fc)'. The font size is 40. The text is bolded. The Inspector panel is also highlighted with a red box.

Hierarchy Panel:

- Lobby*
 - Main Camera
 - Canvas
 - TopPanel
 - OverlayBackground
 - PopupUpdateProfile
 - PopupUpdateNickname
 - BottomTabBar
 - TabBarPages
 - MainPage
 - RankPage
 - CommunityPage
 - SentRequestPage
 - SendRequest
 - InputFieldBase
 - ButtonBase
 - Text (TMP)**
 - PopupPostBox
 - EventSystem
 - LobbyScenario
 - BackendSystem

Inspector Panel:

- Text (TMP) [Static]
- Tag: Untagged, Layer: UI
- Rect Transform
- Canvas Renderer
- TextMeshPro - Text (UI)**
 - Text Input: 친구요청 [Enable RTL Editor]
 - 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]
 - Vertex Color: [Color Picker]
 - Color Gradient: [Color Gradient]
 - Override Tags: [Override Tags]
 - Spacing Options (em): Character 0, Word 0, Line 0, Paragraph 0
 - Alignment: [Alignment Icons]
 - Wrapping: Enabled



친구 요청

- 친구 요청 결과 텍스트를 출력하는 "Text - TextMeshPro" UI 생성 및 설정
 - GameObject - UI - "Text - TextMeshPro"

Hierarchy: Lobby > Canvas > TopPanel > OverlayBackground > PopupUpdateProfile > PopupUpdateNickname > BottomTabBar > TabBarPages > MainPage > RankPage > CommunityPage > SentRequestPage > SendRequest > InputFieldBase > ButtonBase > TextResult

Inspector: TextResult (Tag: Untagged, Layer: UI)

Rect Transform:

stretch	Left	Pos Y	Pos Z
bottom	130	0	0
	Right	Height	
	50	100	

Anchors:

Min	X	Y
	0	0
Max	X	Y
	1	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), Fade Effect_TMP (Script)

Fade Effect_TMP (Script): Script: FadeEffect_TMP, Effect Time: 1.5

Text Input: Enable RTL Editor

결과 텍스트 출력

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: Left Center Right Justified Full

Wrapping: Enabled

Overflow: Overflow

Horizontal Mapping: Character

Vertical Mapping: Character



친구 요청

- 친구 요청 페이지를 제어하는 스크립트 생성 및 작성
 - C# Script 생성 후 스크립트의 이름을 "FriendSentRequestPage"로 변경

```
1  using UnityEngine;
2  using TMPro;
3
4  public class FriendSentRequestPage : MonoBehaviour
5  {
6      [SerializeField]
7      private BackendFriendSystem backendFriendSystem;
8
9      [Header("Send Request Friend")]
10     [SerializeField]
11     private TMP_InputField    inputFieldNickname;
12     [SerializeField]
13     private FadeEffect_TMP    textResult;
14
```



친구 요청

- 친구 요청 페이지를 제어하는 스크립트 생성 및 작성 (계속)

```
15 public void OnClickRequestFriend()
16 {
17     string nickname = inputFieldNickname.text;
18
19     if ( nickname.Trim().Equals("") )
20     {
21         textResult.FadeOut("친구 요청을 보낼 닉네임을 입력해주세요.");
22         return;
23     }
24
25     inputFieldNickname.text = "";
26
27     backendFriendSystem.SendRequestFriend(nickname);
28 }
29 }
```



친구 요청

- SentRequestPage 오브젝트에 "FriendSentRequestPage" 컴포넌트 추가 및 설정

The screenshot displays the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows the scene structure, with **SentRequestPage** selected under the **CommunityPage** folder. The Inspector panel on the right shows the properties of the selected **SentRequestPage** object. The **Friend Sent Request Page (Script)** component is highlighted with a red box. The script's properties are also highlighted with red dashed boxes:

- Backend Friend System:** BackendSystem (Backend Friend)
- Input Field Nickname:** InputFieldBase (TMP_Input Field)
- Text Result:** TextResult (Fade Effect_TMP)

Red arrows point from the **BackendSystem**, **InputFieldBase**, and **TextResult** components in the Hierarchy panel to their respective values in the Inspector panel. The **Add Component** button is visible at the bottom of the Inspector panel.



친구 요청

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

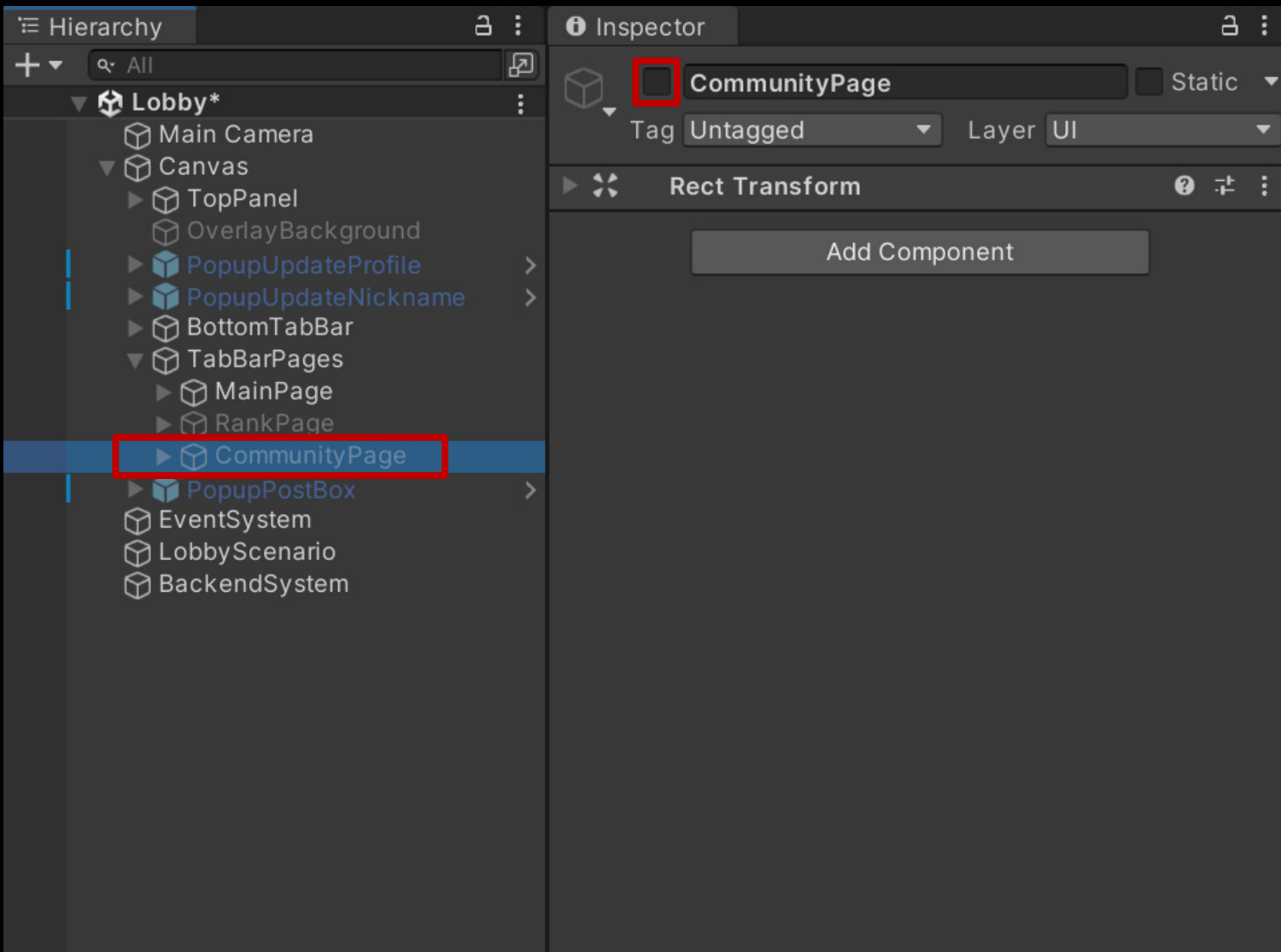
The screenshot displays the Unity Inspector window for a ButtonBase component. The Hierarchy panel on the left shows the scene structure, with the ButtonBase component selected under the SentRequestPage object. The Inspector panel shows the following configuration:

- ButtonBase** (Static): Tag: Untagged, Layer: UI, Prefab: ButtonBase.
- Rect Transform**
- Canvas Renderer**
- Image**
- Button** (Interactable):
 - Transition: Color Tint
 - Target Graphic: ButtonBase (Image)
 - Normal Color, Highlighted Color, Pressed Color, Selected Color, Disabled Color: Color pickers.
 - Color Multiplier: 1
 - Fade Duration: 0.1
 - Navigation: Automatic
- On Click ()**: Runtime Only, FriendSentRequestPage.OnClick, SentRequestPage.OnClick.



친구 요청

- CommunityPage 오브젝트 비활성화





친구 요청

- Community 오브젝트의 "Toggle" 컴포넌트 OnValueChanged() 이벤트 등록

The screenshot shows the Unity Hierarchy and Inspector panels. In the Hierarchy, the 'Community' object is selected under the 'Lobby*' folder. The Inspector panel shows the 'Toggle' component selected. The 'On Value Changed (Boolean)' event is configured with the following settings:

- Runtime Only:
- Method: `GameObject.SetActive`
- Target: `Community`

Red boxes and arrows highlight the 'Community' object in the Hierarchy, the 'Toggle' component in the Inspector, and the event configuration details.



친구 요청

■ 결과 화면

The screenshot shows a Unity game console with a dark theme. The console window is open, displaying several log messages. A red box highlights a specific message: "[13:47:21] user03의 inDate 값은 2023-06-26T06:43:54.899Z 입니다. UnityEngine.Debug:Log (object)". Below this, another message reads "[13:47:22] 친구 요청에 성공했습니다. : statusCode : 204 message : Success". The game view in the background shows a login screen with the title "로그인" (Login), a text input field containing "user01", a password input field with four asterisks, and a yellow "로그인" button. At the bottom of the login screen are three links: "아이디 찾기" (Find ID), "계정 생성" (Create Account), and "비밀번호 찾기" (Find Password).

친구 요청 대기 목록

- 친구 요청 대기 목록 조회
- 커뮤니티 페이지 TabBar 제작
- 친구 요청 대기 목록 UI 제작
- 메모리풀 (MemoryPool)
- 친구 요청 대기 목록 UI 출력
- 요청 대기 중인 친구 UI 데이터 연동
- 친구 요청 취소
- 만료시간 이후 친구 요청 자동 취소



친구 요청 대기 목록

■ 친구 요청 대기 목록 조회

- 친구 한 명의 데이터를 관리하는 스크립트 생성 및 작성
 - C# Script 생성 후 스크립트의 이름을 "FriendData"로 변경

```
1 public class FriendData
2 {
3     public string nickname; // 닉네임
4     public string inDate; // 해당 유저의 inDate
5     public string createdAt; // 친구 요청 보낸 시간 / 친구 요청 받은 시간 / 친구가 된 시간
6
7     public override string ToString()
8     {
9         string result = string.Empty;
10        result += $"닉네임 : {nickname}\n";
11        result += $"inDate : {inDate}\n";
12        result += $"createdAt : {createdAt}\n";
13
14        return result;
15    }
16 }
```



친구 요청 대기 목록

■ 친구 요청 대기 목록 조회 메소드

`GetSentRequestList(int limit=100, int offset=0);`

`RequestFriend()` 메소드를 호출해 친구 요청을 보낸 [요청대기] 목록 조회
친구 요청을 받은 유저가 수락/거절하면 해당 목록에서 삭제

`limit` : 불러올 친구 요청 대기 목록의 수 (default : 100)

`offset` : 불러올 친구 요청 대기 기준 숫자

ex) `GetSentRequestList(5);` -> 친구 요청을 보낸 5명 목록 조회 (1-5)

ex) `GetSentRequestList(5, 5);` -> 친구 요청을 보낸 처음 5명 이후의 5명 조회 (6-10)



친구 요청 대기 목록

■ 친구 요청 대기 목록 JsonData

```
{
  "rows":
  [
    // 닉네임이 있는 유저
    {
      "nickname": "고박사" // 유저의 닉네임
      "inDate": "2023-06-26T02:22:22.022Z" // 유저의 inDate
      "createdAt": "2023-06-26T02:22:22.022Z" // 친구 요청 보낸 시간
    },
    // 닉네임이 없는 유저
    {
      "inDate": "2023-06-26T02:22:22.022Z" // 유저의 inDate
      "createdAt": "2023-06-26T02:22:22.022Z" // 친구 요청 보낸 시간
    },
  ]
}
```



친구 요청 대기 목록

- GetSentRequestList() 메소드를 호출해 친구 [요청대기] 목록 조회
 - BackendFriendSystem Script 수정

```
1  using UnityEngine;
2  using System;
3  using BackEnd;
4
5  public class BackendFriendSystem : MonoBehaviour
6  {
7      private string GetUserInfoBy(string nickname) ...
44
45     public void SendRequestFriend(string nickname)
46     {
47         // RequestFriend() 메소드를 이용해 친구 추가 요청을 할 때 해당 친구의 inDate 정보가 필요
48         string inDate = GetUserInfoBy(nickname);
49
50         // inDate 정보를 가진 유저에게 "친구 요청"을 보낸다.
51         Backend.Friend.RequestFriend(inDate, callback =>
52         {
53             if ( !callback.IsSuccess() )
54             {
55                 Debug.LogError($"{nickname} 친구 요청 도중 에러가 발생했습니다. : {callback}");
56                 return;
57             }
58
59             Debug.Log($"친구 요청에 성공했습니다. : {callback}");
60
61             // 친구 요청에 성공하면 친구 요청 대기 목록 불러오기
62             GetSentRequestList();
63         });
64     }
65 }
```



친구 요청 대기 목록

- BackendFriendSystem Script 수정 (계속)

```
66 public void GetSentRequestList()
67 {
68     Backend.Friend.GetSentRequestList(callback =>
69     {
70         if ( !callback.IsSuccess() )
71         {
72             Debug.LogError($"친구 요청 대기 목록 조회 도중 에러가 발생했습니다. : {callback}");
73             return;
74         }
75
76         // JSON 데이터 파싱 성공
77         try
78         {
79             LitJson.JsonData jsonData = callback.GetFlattenJSON()["rows"];
80
81             // 받은 데이터의 개수가 0이면 데이터가 없는 것
82             if ( jsonData.Count <= 0 )
83             {
84                 Debug.LogWarning("친구 요청 대기 목록 데이터가 없습니다.");
85                 return;
86             }
87
```



친구 요청 대기 목록

□ BackendFriendSystem Script 수정 (계속)

```
88     foreach ( LitJson.JsonData item in jsonData )
89     {
90         FriendData friendData = new FriendData();
91
92         //friend.nickname = item.ContainsKey("nickname") == true ? item["nickname"].ToString() : "NONAME";
93         friendData.nickname = item["nickname"].ToString().Equals("True") ? "NONAME" : item["nickname"].ToString();
94         friendData.inDate = item["inDate"].ToString();
95         friendData.createdAt = item["createdAt"].ToString();
96
97         Debug.Log(friendData.ToString());
98     }
99 }
100 // JSON 데이터 파싱 실패
101 catch ( Exception e )
102 {
103     // try-catch 에러 출력
104     Debug.LogError(e);
105 }
106 });
107 }
108 }
```




친구 요청 대기 목록

■ 결과 화면

```
Console
Clear Collapse Error Pause Editor
UnityEngine.Debug:Log (object)
[14:08:31] Level : 10, Max Exp : 51200,Reward Gold : 100000
UnityEngine.Debug:Log (object)
[14:08:31] 게임 정보 데이터 불러오기에 성공했습니다. : statusCode : 200
message : Success
[14:08:35] user03의 inDate 값은 2023-06-26T06:43:54.899Z 입니다.
UnityEngine.Debug:Log (object)
[14:08:35] 친구 요청에 성공했습니다. : statusCode : 204
message : Success
[14:08:36] 닉네임 : user03
inDate : 2023-06-26T06:43:54.899Z
닉네임 : user03
inDate : 2023-06-26T06:43:54.899Z
createdAt : 2023-06-30T05:08:36.119Z
UnityEngine.Debug:Log (object)
BackendFriendSystem/<>c:<GetSentRequestList>b 2 0 (BackendBackendReturnObject) (at Assets/Scripts/#100Backend/BackendFriendSystem.cs:97)
```



친구 요청 대기 목록

■ 커뮤니티 페이지 TabBar 제작

- 커뮤니티 페이지의 TabBar UI를 관리하는 Panel UI 생성 및 설정
 - GameObject - UI - Panel

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Lobby*'. The 'CommunityPage' folder is expanded, showing 'TabBar' and 'SentRequestPage' as child objects. The 'TabBar' object is selected. On the right, the Inspector panel shows the 'TabBar' component. The 'Rect Transform' component is highlighted with a red dashed box, showing its 'stretch' property set to 'top' and its 'Anchors' (Min, Max, Pivot) and 'Scale' properties. The 'Canvas Renderer' component is also highlighted with a red dashed box.

Property	Value
stretch	top
Left	0
Pos Y	0
Pos Z	0
Right	0
Height	200
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



친구 요청 대기 목록

- TabBar 오브젝트에 컴포넌트 추가 및 설정

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'TabBar' object is selected under the 'Lobby*' folder. The Inspector panel shows the following configuration for the 'TabBar' component:

- TabBar** (Static:)
 - Tag: Untagged
 - Layer: UI
- Rect Transform**
 - Horizontal Layout Group
 - Padding: Left: 0, Right: 0, Top: 0, Bottom: 0, Spacing: 0
 - Child Alignment: Upper Left
 - Reverse Arrangement:
 - Control Child Size: Width, Height
 - Use Child Scale: Width, Height
 - Child Force Expand: Width, Height
 - Toggle Group
 - Script: ToggleGroup
 - Allow Switch Off:

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



친구 요청 대기 목록

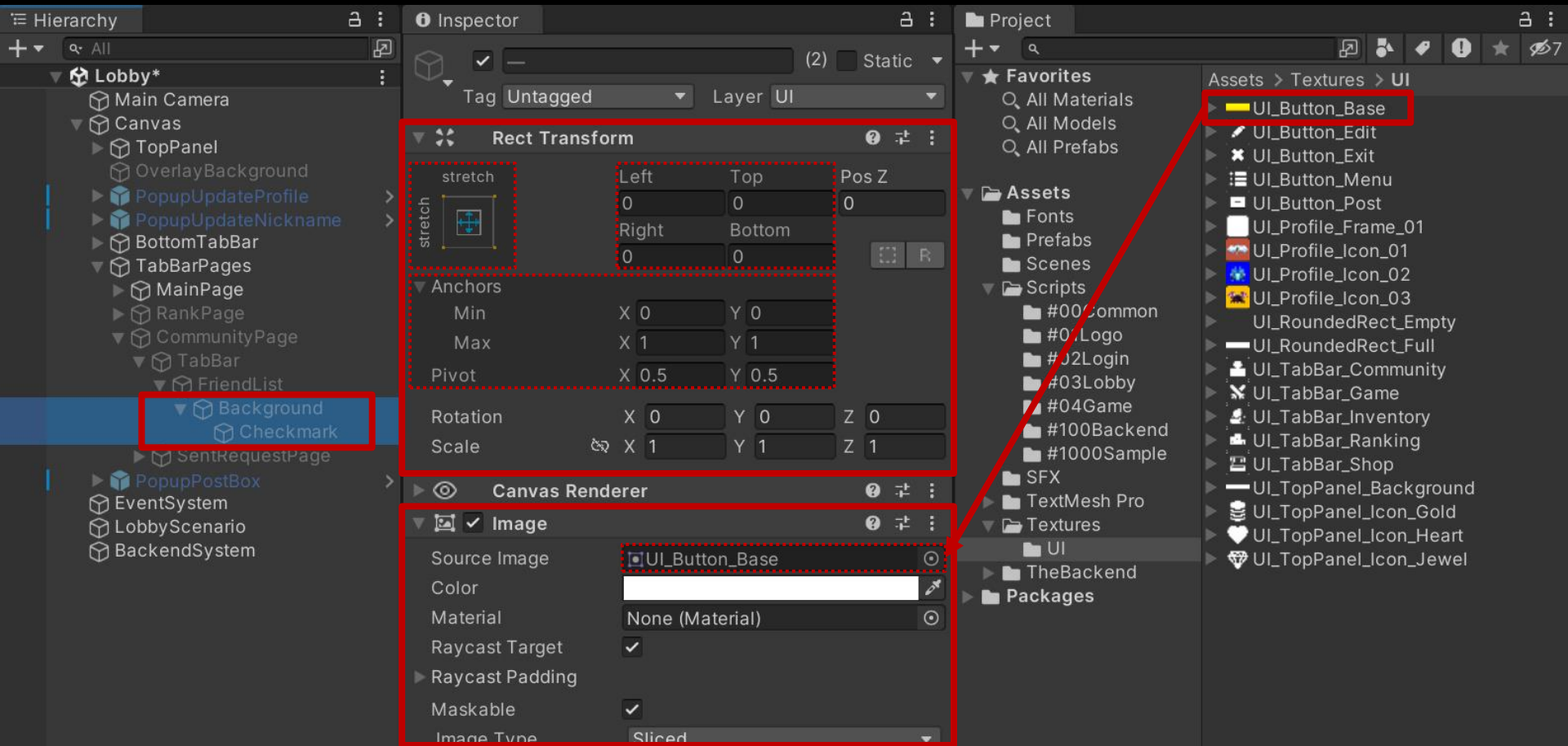
- 친구목록 Toggle UI 생성 및 설정
 - GameObject - UI - Toggle

The image shows the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows a tree structure under 'Lobby*' with 'FriendList' selected under 'TabBar'. A red box highlights the 'FriendList' object and its children: 'Background', 'Checkmark', and 'Label'. A black box with white text below the Hierarchy panel reads 'Label은 사용하지 않기 때문에 삭제' (Delete Label because it is not used). The Inspector panel on the right shows the properties for the selected 'FriendList' object, including 'Rect Transform' and 'Toggle' components. The 'Toggle' component is expanded, showing settings for 'Interactable', 'Transition' (Color Tint), 'Target Graphic' (Background (Image)), and various color swatches (Normal, Highlighted, Pressed, Selected, Disabled). Other settings include 'Color Multiplier' (1), 'Fade Duration' (0.1), 'Navigation' (Automatic), 'Is On' (checked), 'Toggle Transition' (Fade), and 'Graphic' (Checkmark (Image)).



친구 요청 대기 목록

■ FriendList Toggle UI 생성 및 설정 (계속)





친구 요청 대기 목록

■ FriendList Toggle UI 생성 및 설정 (계속)

The screenshot displays the Unity development environment. On the left, the Hierarchy panel shows a tree structure under 'Lobby*'. The 'Background' component under 'FriendList' is selected and highlighted with a red box. On the right, the Inspector panel shows the properties for the selected 'Image' component. The 'Color' property is highlighted with a red dashed box and has a tooltip showing 'Color (100, 100, 100, 255)'. Other visible properties include 'Source Image' (UI_Button_Base), 'Material' (None (Mat)), 'Raycast Target' (checked), 'Maskable' (checked), and 'Image Type' (Sliced).

Property	Value
Background	Static
Tag	Untagged
Layer	UI
Rect Transform	
Canvas Renderer	
Image	
Source Image	UI_Button_Base
Color	Color (100, 100, 100, 255)
Material	None (Mat)
Raycast Target	checked
Raycast Padding	
Maskable	checked
Image Type	Sliced
Fill Center	checked
Pixels Per Unit Mul	1



친구 요청 대기 목록

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

Rect Transform	
stretch	Left: 0, Top: 0, Right: 0, Bottom: 0, Pos Z: 0
stretch	
Anchors	
Min	X: 0, Y: 0
Max	X: 1, Y: 1
Pivot	X: 0.5, Y: 0.5
Rotation	X: 0, Y: 0, Z: 0
Scale	X: 1, Y: 1, Z: 1

Text Input: Enable RTL Editor

친구목록

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: 80

Auto Size:

Vertex Color:

Color Gradient:

Override Tags:

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

Alignment: Center Left Right Justified Full Justified

Wrapping: Enabled

Overflow: Overflow

Horizontal Mapping: Character

Vertical Mapping: Character



친구 요청 대기 목록

- SentRequestList, ReceivedRequestList Toggle UI 생성 및 설정
 - FriendList 오브젝트를 Ctrl+D로 복제

The screenshot displays the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows the scene structure under 'Lobby*'. The Inspector panel on the right shows the properties of the selected 'TextMeshPro - Text (UI)' object. A red box highlights the 'TextMeshPro - Text (UI)' object in the Hierarchy and its corresponding Inspector settings. The Inspector shows the text input field with the text 'SentRequestList : "요청대기"' and 'ReceivedRequestList : "수락대기"'. The font asset is set to 'NotoSansKR-Bold SDF (TMP_Fc)' and the material preset is 'NotoSansKR-Bold SDF Material'. The font style is set to 'B' (Bold) and the font size is 80.

Hierarchy Panel:

- Lobby*
 - Main Camera
 - Canvas
 - TopPanel
 - OverlayBackground
 - PopupUpdateProfile
 - PopupUpdateNickname
 - BottomTabBar
 - TabBarPages
 - MainPage
 - RankPage
 - CommunityPage
 - TabBar
 - FriendList
 - SentRequestList
 - Background
 - Text (TMP)
 - ReceivedRequestList
 - Background
 - Text (TMP)
 - SentRequestPage
 - PopupPostBox

- EventSystem
- LobbyScenario
- BackendSystem



친구 요청 대기 목록

- Toggle 오브젝트의 "Toggle" 컴포넌트 Group 설정

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Lobby*' object is expanded to show its children, including 'Canvas', 'TopPanel', 'OverlayBackground', 'PopupUpdateProfile', 'PopupUpdateNickname', 'BottomTabBar', 'TabBarPages', and 'TabBar'. The 'TabBar' object is selected, and its children 'FriendList', 'SentRequestList', and 'ReceivedRequestList' are visible. In the Inspector panel, the 'Toggle' component is selected, and its properties are shown. The 'Group' property is set to 'TabBar (Toggle Group)'. A red box highlights the 'Toggle' component and its 'Group' property, with a red arrow pointing from the 'TabBar' object in the Hierarchy to the 'Group' property in the Inspector.

Hierarchy Panel:

- Lobby*
 - Main Camera
 - Canvas
 - TopPanel
 - OverlayBackground
 - PopupUpdateProfile
 - PopupUpdateNickname
 - BottomTabBar
 - TabBarPages
 - MainPage
 - RankPage
 - CommunityPage
 - TabBar
 - FriendList
 - SentRequestList
 - ReceivedRequestList
 - SentRequestPage
 - PopupPostBox
 - EventSystem
 - LobbyScenario
 - BackendSystem

Inspector Panel:

- Rect Transform
 - Toggle
 - Interactable:
 - Transition: Color Tint
 - Target Graphic: —
 - Normal Color: [Color Picker]
 - Highlighted Color: [Color Picker]
 - Pressed Color: [Color Picker]
 - Selected Color: [Color Picker]
 - Disabled Color: [Color Picker]
 - Color Multiplier: 1
 - Fade Duration: 0.1
 - Navigation: Automatic
 - Visualize
 - Is On:
 - Toggle Transition: Fade
 - Graphic: —
 - Group: TabBar (Toggle Group)



친구 요청 대기 목록

- SentRequestList 오브젝트의 "Toggle" 컴포넌트 OnValueChanged() 이벤트 등록

The screenshot shows the Unity Inspector window with the following configuration for the **SentRequestList** object:

- Component:** Toggle
- Interactable:**
- Transition:** Color Tint
 - Target Graphic: Background (Image)
 - Normal Color: [Color Picker]
 - Highlighted Color: [Color Picker]
 - Pressed Color: [Color Picker]
 - Selected Color: [Color Picker]
 - Disabled Color: [Color Picker]
 - Color Multiplier: 1
 - Fade Duration: 0.1
- Navigation:** Automatic
- Is On:**
- Toggle Transition:** Fade
- Graphic:** Checkmark (Image)
- Group:** TabBar (Toggle Group)
- On Value Changed (Boolean):**
 - Event Type: Runtime Only
 - Event Name: GameObject.SetActive
 - Target: SentRequestList



친구 요청 대기 목록

- Content 오브젝트에 컴포넌트 추가 및 설정

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Content' object is selected under the 'Lobby*' hierarchy. The Inspector panel shows the following configuration:

- Content**: Content, Static, Tag: Untagged, Layer: UI
- Rect Transform**: Rect Transform
- Vertical Layout Group**: Vertical Layout Group
 - Padding: Left: 20, Right: 20, Top: 20, Bottom: 20
 - Spacing: 20
 - Child Alignment: Upper Center
 - Reverse Arrangement:
 - Control Child Size: Width, Height
 - Use Child Scale: Width, Height
 - Child Force Expand: Width, Height
- Content Size Fitter**: Content Size Fitter
 - Horizontal Fit: Unconstrained
 - Vertical Fit: Preferred Size

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



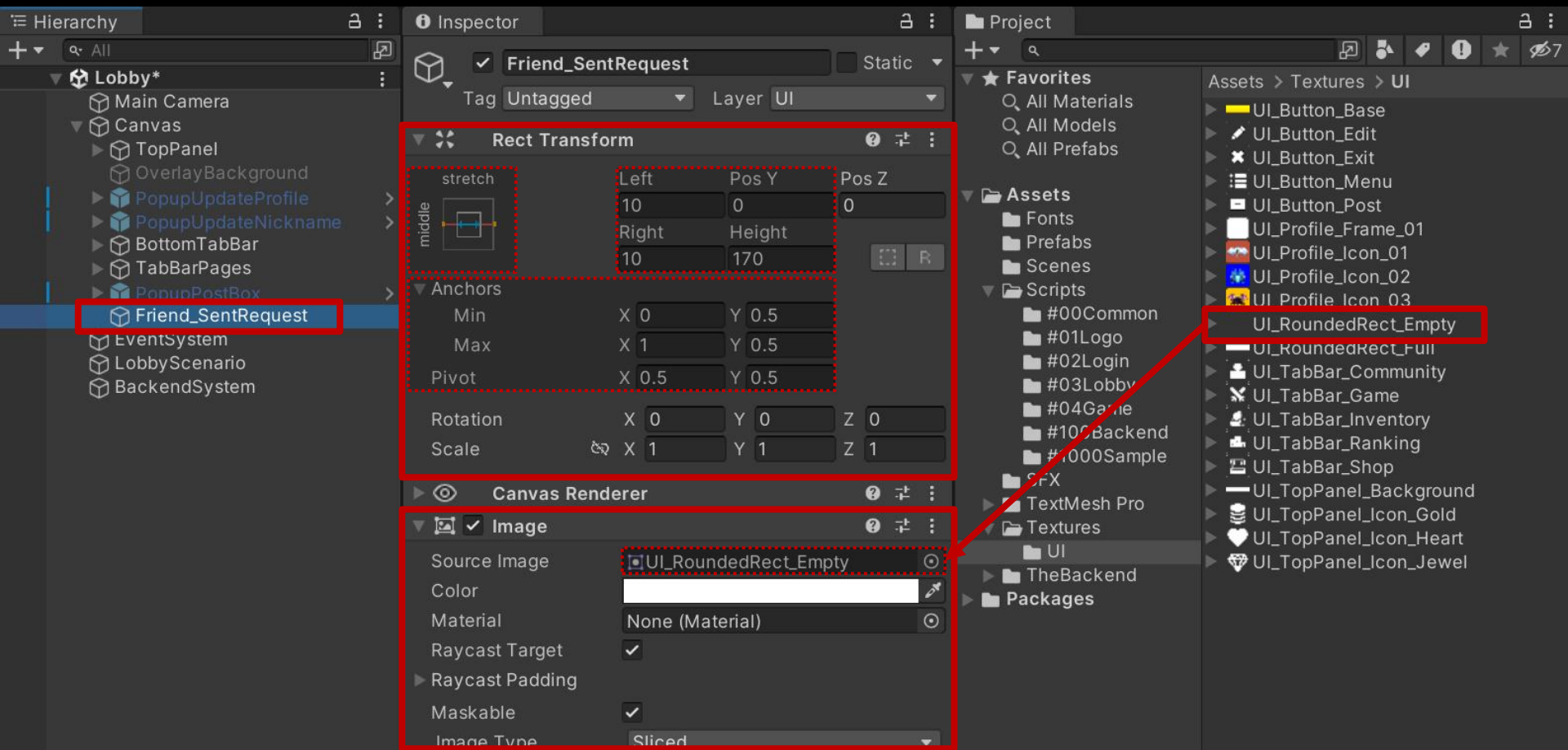
친구 요청 대기 목록

- “[친구 요청..” 텍스트를 출력하는 “Text - TextMeshPro” UI 생성 및 설정
 - GameObject - UI - “Text - TextMeshPro”



친구 요청 대기 목록

- 요청 대기 중인 친구 UI를 관리하는 Image UI 생성 및 설정
 - GameObject - UI - Image

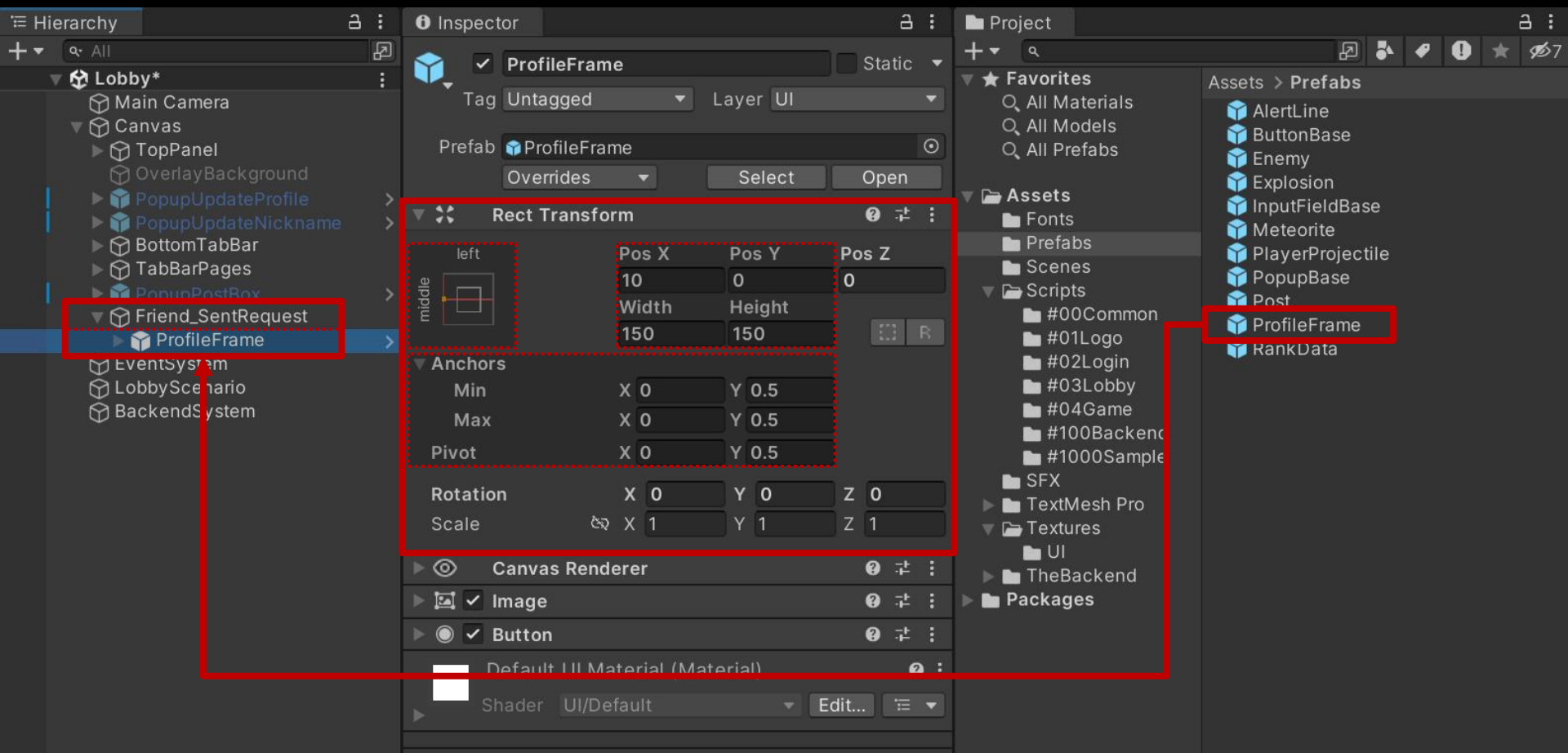




친구 요청 대기 목록

■ 프로필 아이콘 생성 및 설정

- ProfileFrame 프리팹을 Hierarchy View로 Drag & Drop





친구 요청 대기 목록

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

Hierarchy: Lobby* > Canvas > TopPanel > OverlayBackground > PopupUpdateProfile > PopupUpdateNickname > BottomTabBar > TabBarPages > Friend_SentRequest > ProfileFrame > Nickname

Inspector: Nickname (Static) | Tag: Untagged | Layer: UI

Rect Transform:

Anchor	Pos X	Pos Y	Pos Z
left	180	0	0
middle	400	75	

TextMeshPro - Text (UI): NotoSansKR-Bold SDF Material (Material) | Shader: TextMeshPro/Distance Fi

Text Input: Enable RTL Editor

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: 60
- Auto Size:
- Vertex Color:
- Color Gradient:
- Override Tags:
- Spacing Options (em): Character 0, Word 0, Line 0, Paragraph 0
- Alignment: Left Center Right Justified Full
- Wrapping: Enabled
- Overflow: Overflow
- Horizontal Mapping: Character
- Vertical Mapping: Character



친구 요청 대기 목록

- 남은 시간을 출력하는 "Text - TextMeshPro" UI 생성 및 설정
 - GameObject - UI - "Text - TextMeshPro"

Hierarchy: Lobby* > Friend_SentRequest > ExpirationDate

Inspector: Rect Transform

Property	Value
Left	600
Pos Y	0
Pos Z	0
Right	280
Height	75

Inspector: TextMeshPro - Text (UI)

Property	Value
Font Asset	NotoSansKR-Bold SDF (TMP_Fc)
Font Style	B
Font Size	36

Text Input: 72시간 남음

Text Style: Normal

Main Settings

Property	Value
Font Asset	NotoSansKR-Bold SDF (TMP_Fc)
Material Preset	NotoSansKR-Bold SDF Material
Font Style	B
Font Size	36

Alignment: Center

Wrapping: Enabled

Overflow: Overflow



친구 요청 대기 목록

- 친구 요청 취소 "Button - TextMeshPro" UI 생성 및 설정
 - GameObject - UI - "Button - TextMeshPro"

The screenshot displays the Unity Hierarchy and Inspector panels. The Hierarchy panel on the left shows a tree structure under 'Lobby*' with 'Friend_SentRequest' selected, containing 'ProfileFrame', 'Nickname', 'ExpirationDate', and 'ButtonCancelRequest'. The Inspector panel on the right shows the properties for 'ButtonCancelRequest'.

Rect Transform

Property	Value
Pos X	-10
Pos Y	0
Pos Z	0
Width	250
Height	150

Anchors

Property	Value
Min X	1
Min Y	0.5
Max X	1
Max Y	0.5
Pivot X	1
Pivot Y	0.5

Rotation

Property	Value
X	0
Y	0
Z	0

Scale

Property	Value
X	1
Y	1
Z	1

Canvas Renderer

Image

Property	Value
Source Image	UISprite
Color	Color (73, 105, 139, 255)
Material	None (Mat)
Raycast Target	✓



친구 요청 대기 목록

- 친구 요청 취소 "Button - TextMeshPro" UI 생성 및 설정 (계속)

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the 'Text (TMP)' object under the 'Friend_SentRequest' folder is selected and highlighted with a red box. The Inspector panel shows the configuration for the 'TextMeshPro - Text (UI)' component, which is also highlighted with a red box. The text content is '요청취소'. The 'Font Asset' is set to 'NotoSansKR-Bold SDF (TMP_Fc)'. The 'Font Style' is set to 'B' (Bold). The 'Font Size' is set to 40. The 'Color' is set to (255, 255, 255, 255).

Hierarchy Panel:

- Lobby*
 - Main Camera
 - Canvas
 - TopPanel
 - OverlayBackground
 - PopupUpdateProfile
 - PopupUpdateNickname
 - BottomTabBar
 - TabBarPages
 - PopupPostBox
 - Friend_SentRequest
 - ProfileFrame
 - Nickname
 - ExpirationDate
 - ButtonCancelRequest
 - Text (TMP)**
 - EventSystem
 - LobbyScenario
 - BackendSystem

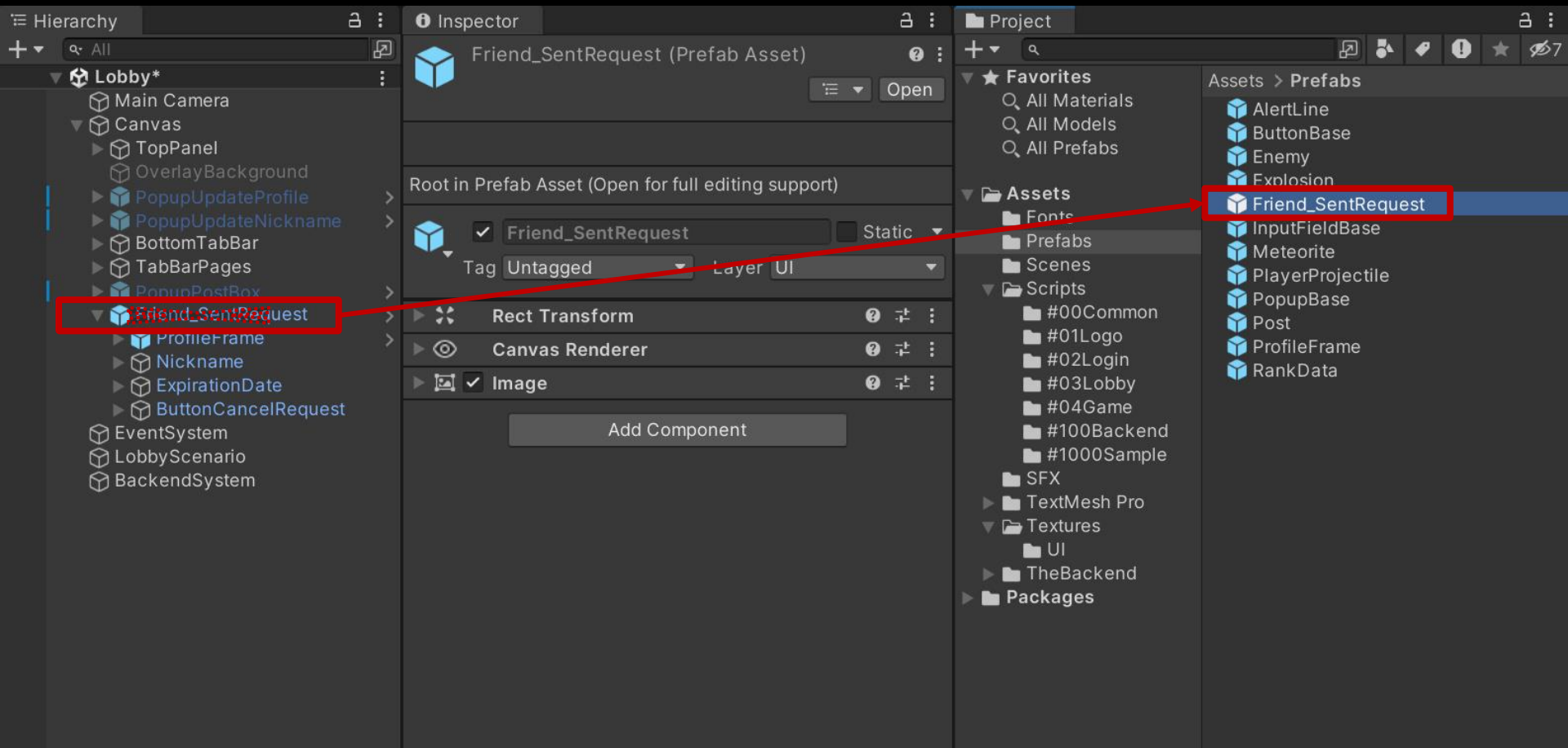
Inspector Panel:

- Text (TMP) [Static]
- Tag: Untagged, Layer: UI
- Rect Transform
- Canvas Renderer
- TextMeshPro - Text (UI)**
 - Text Input: Enable RTL Editor [Off]
 - 요청취소
 - 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: [Off]
 - Vertex Color: [Color Picker]
 - Color Gradient: [Off] **Color (255, 255, 255, 255)**



친구 요청 대기 목록

- Friend_SentRequest 오브젝트 Prefab 생성
 - Hierarchy View의 "Friend_SentRequest" 오브젝트를 Project View로 Drag & Drop
 - Hierarchy View에 있는 "Friend_SentRequest" 오브젝트 삭제





친구 요청 대기 목록

■ 메모리풀 (MemoryPool)

- 메모리를 할당하는 **new**는 사용하지만 반납하는 **Delete**는 사용하지 않는 것
 - 유니티 엔진은 오브젝트를 다루기 때문에 오브젝트 풀(Object Pool)이라고 한다
 - 유니티 엔진의 경우
 - 오브젝트를 생성하는 **Instantiate()** 함수는 사용하지만
 - 오브젝트를 삭제하는 **Destroy()** 함수는 사용하지 않는 것
- 메모리 풀은 왜 사용하는 것인가?
 - 모노.Net 환경(C#, others)을 사용할 경우 메모리를 해제할 때 가비지 컬렉션 (Garbage Collection)이 발생하며, 이 때 **성능이 저하되어 끊김 현상**이 나타난다.
 - 유니티에서 오브젝트의 생성/삭제가 빈번하게 일어나는 게임의 경우 가비지 컬렉션이 최소한으로 발생하게 하기 위해 메모리 풀(오브젝트 풀)을 이용



친구 요청 대기 목록

- 메모리 풀은 생성과 삭제가 빈번하게 발생하는 모든 경우 사용
 - 비행 슈팅 게임의 "발사체", "다양한 이펙트"
 - RPG 게임의 "적 캐릭터", "아이템", 전투에서 발생하는 "체력 증가/감소 UI" 등
- 메모리 풀은 정적 크기와 동적 크기의 두 가지 경우로 설계 함
 - 정적 크기
 - 필요한 최대 크기 만큼 할당 받아 두고, 그 안에서만 활용
 - 메모리를 확보하는 new 행위를 한번만 수행
 - 현재 사용중인 메모리 개수를 알기 어렵다
 - 동적 크기
 - 사용자가 지정한 작은 수의 개수 만큼 메모리를 할당
 - 현재 할당된 개수 보다 더 많은 메모리가 필요할 때마다 조금씩 추가로 메모리를 할당
 - 메모리를 확보하는 new 행위를 여러 번 수행
 - 현재 사용중인 메모리 개수를 알기 쉽다



친구 요청 대기 목록

- 오브젝트를 관리하는 메모리 풀 스크립트 생성 및 작성
 - C# Script 생성 후 스크립트의 이름을 "MemoryPool"로 변경

```
1 using UnityEngine;
2 using System.Collections.Generic;
3
4 public class MemoryPool
5 {
6     // 메모리 풀로 관리되는 오브젝트 정보
7     private class PoolItem
8     {
9         public bool    isActive;        // "gameObject"의 활성화/비활성화 정보
10        public GameObject gameObject;    // 화면에 보이는 실제 게임오브젝트
11    }
12
13    private int increaseCount = 5;        // 오브젝트가 부족할 때 Instantiate()로 추가 생성되는 오브젝트 개수
14    private int maxCount;                // 현재 리스트에 등록되어 있는 오브젝트 개수
15    private int activeCount;             // 현재 게임에 사용되고 있는(활성화) 오브젝트 개수
16
17    private GameObject poolObject;       // 오브젝트 풀링에서 관리하는 게임 오브젝트 프리팹
18    private Transform poolParent;        // 생성하는 게임 오브젝트의 부모 Transform
19    private List<PoolItem> poolItemList; // 관리되는 모든 오브젝트를 저장하는 리스트
20
21    public int MaxCount => maxCount;     // 외부에서 현재 리스트에 등록되어 있는 오브젝트 개수 확인을 위한 프로퍼티
22    public int ActiveCount => activeCount; // 외부에서 현재 활성화 되어 있는 오브젝트 개수 확인을 위한 프로퍼티
23
```




친구 요청 대기 목록

- 오브젝트를 관리하는 메모리 풀 스크립트 생성 및 작성 (계속)

```
24 public MemoryPool(GameObject poolObject, Transform poolParent=null)
25 {
26     maxCount        = 0;
27     activeCount     = 0;
28     this.poolObject = poolObject;
29     this.poolParent = poolParent;
30
31     poolItemList    = new List<PoolItem>();
32
33     InstantiateObjects();
34 }
35
```



친구 요청 대기 목록

- 오브젝트를 관리하는 메모리 풀 스크립트 생성 및 작성 (계속)

```
36  // <summary>
37  // increaseCount 단위로 오브젝트를 생성
38  // </summary>
39  public void InstantiateObjects()
40  {
41      maxCount += increaseCount;
42
43      for ( int i = 0; i < increaseCount; ++ i )
44      {
45          PoolItem poolItem = new PoolItem();
46
47          poolItem.isActive = false;
48          poolItem.gameObject = GameObject.Instantiate(poolObject);
49          poolItem.gameObject.SetActive(false);
50          poolItem.gameObject.transform.SetParent(poolParent);
51
52          poolItemList.Add(poolItem);
53      }
54  }
55
```



친구 요청 대기 목록

- 오브젝트를 관리하는 메모리 풀 스크립트 생성 및 작성 (계속)

```
56  /// <summary>
57  /// 현재 관리중인(활성/비활성) 모든 오브젝트를 삭제
58  /// </summary>
59  public void DestroyObjects()
60  {
61      if ( poolItemList == null ) return;
62
63      int count = poolItemList.Count;
64      for ( int i = 0; i < count; ++ i )
65      {
66          GameObject.Destroy(poolItemList[i].gameObject);
67      }
68
69      poolItemList.Clear();
70  }
71
```



친구 요청 대기 목록

■ 오브젝트를 관리하는 메모리 풀 스크립트 생성 및 작성 (계속)

```
72     /// <summary>
73     /// poolItemList에 저장되어 있는 오브젝트를 활성화해서 사용
74     /// 현재 모든 오브젝트가 사용중이면 InstantiateObjects()로 추가 생성
75     /// </summary>
76     public GameObject ActivatePoolItem()
77     {
78         if ( poolItemList == null ) return null;
79
80         // 현재 생성해서 관리하는 모든 오브젝트 개수와 현재 활성화 상태인 오브젝트 개수 비교
81         // 모든 오브젝트가 활성화 상태이면 새로운 오브젝트 필요
82         if ( maxCount == activeCount ) InstantiateObjects();
83
84         int count = poolItemList.Count;
85         for ( int i = 0; i < count; ++ i )
86         {
87             PoolItem poolItem = poolItemList[i];
88
89             if ( poolItem.isActive == false )
90             {
91                 activeCount ++;
92
93                 poolItem.isActive = true;
94                 poolItem.gameObject.SetActive(true);
95
96                 return poolItem.gameObject;
97             }
98         }
99
100        return null;
101    }
102
```



친구 요청 대기 목록

- 오브젝트를 관리하는 메모리 풀 스크립트 생성 및 작성 (계속)

```
103  /// <summary>
104  /// 현재 사용이 완료된 오브젝트를 비활성화 상태로 설정
105  /// </summary>
106  public void DeactivatePoolItem(GameObject removeObject)
107  {
108      if ( poolItemList == null || removeObject == null ) return;
109
110      int count = poolItemList.Count;
111      for ( int i = 0; i < count; ++ i )
112      {
113          PoolItem poolItem = poolItemList[i];
114
115          if ( poolItem.gameObject == removeObject )
116          {
117              activeCount --;
118
119              poolItem.isActive = false;
120              poolItem.gameObject.SetActive(false);
121
122              return;
123          }
124      }
125  }
126
```



친구 요청 대기 목록

- 오브젝트를 관리하는 메모리 풀 스크립트 생성 및 작성 (계속)

```
127     /// <summary>
128     /// 현재 활성화되어 있는 오브젝트 하나를 비활성화 상태로 설정
129     /// </summary>
130     public GameObject DeactivatePoolItem()
131     {
132         if ( poolItemList == null ) return null;
133
134         int count = poolItemList.Count;
135         for ( int i = 0; i < count; ++ i )
136         {
137             PoolItem poolItem = poolItemList[i];
138
139             if ( poolItem.gameObject.activeSelf == true )
140             {
141                 activeCount --;
142
143                 poolItem.isActive = false;
144                 poolItem.gameObject.SetActive(false);
145
146                 return poolItem.gameObject;
147             }
148         }
149
150         return null;
151     }
152
```



친구 요청 대기 목록

- 오브젝트를 관리하는 메모리 풀 스크립트 생성 및 작성 (계속)

```
153  /// <summary>
154  /// 게임에 사용중인 모든 오브젝트를 비활성화 상태로 설정
155  /// </summary>
156  public void DeactivateAllPoolItems()
157  {
158      if ( poolItemList == null ) return;
159
160      int count = poolItemList.Count;
161      for ( int i = 0; i < count; ++ i )
162      {
163          PoolItem poolItem = poolItemList[i];
164
165          if ( poolItem.gameObject != null && poolItem.isActive == true )
166          {
167              poolItem.isActive = false;
168              poolItem.gameObject.SetActive(false);
169          }
170      }
171
172      activeCount = 0;
173  }
174 }
```



친구 요청 대기 목록

■ 친구 요청 대기 목록 UI 출력

- 커뮤니티 페이지에 있는 페이지들이 상속받는 스크립트 생성 및 작성
 - C# Script 생성 후 스크립트의 이름을 "FriendPageBase"로 변경

```
1 using System.Collections.Generic;
2 using UnityEngine;
3
4 public class FriendPageBase : MonoBehaviour
5 {
6     [Header("Common")]
7     [SerializeField]
8     protected BackendFriendSystem backendFriendSystem;
9
10    [Header("Friend Page Base")]
11    [SerializeField]
12    private GameObject friendPrefab; // 해당 페이지에 출력하는 개별 친구 UI
13    [SerializeField]
14    private Transform parentContent; // UI가 배치되는 ScrollView의 Content
15    [SerializeField]
16    private GameObject textSystem; // 해당 페이지가 비어있을 때 출력하는 Text UI
17
18    private MemoryPool memoryPool;
19
```




친구 요청 대기 목록

- 커뮤니티 페이지에 있는 페이지들이 상속받는 스크립트 생성 및 작성 (계속)

```
20 private void Awake()
21 {
22     memoryPool = new MemoryPool(friendPrefab, parentContent);
23 }
24
25 public void Activate(FriendData friend)
26 {
27     if ( textSystem.activeSelf ) textSystem.SetActive(false);
28
29     memoryPool.ActivatePoolItem();
30 }
31
32 public void ActivateAll(List<FriendData> friendList)
33 {
34     for ( int i = 0; i < friendList.Count; ++ i )
35     {
36         Activate(friendList[i]);
37     }
38 }
39
```



친구 요청 대기 목록

- 커뮤니티 페이지에 있는 페이지들이 상속받는 스크립트 생성 및 작성 (계속)

```
40 public void DeactivateAll()
41 {
42     textSystem.SetActive(true);
43
44     memoryPool.DeactivateAllPoolItems();
45 }
46
47 public void Deactivate(GameObject friend)
48 {
49     memoryPool.DeactivatePoolItem(friend);
50
51     if ( memoryPool.ActiveCount == 0 )
52     {
53         textSystem.SetActive(true);
54     }
55 }
56 }
```



친구 요청 대기 목록

- 친구 요청 대기 목록 UI 출력을 위해 FriendPageBase 클래스 상속
 - FriendSentRequestPage Script 수정

```
1 using UnityEngine;
2 using TMPro;
3
4 public class FriendSentRequestPage : FriendPageBase /*MonoBehaviour*/
5 {
6     /*[SerializeField]
7     private BackendFriendSystem backendFriendSystem;*/
8
9     [Header("Send Request Friend")]
10    [SerializeField]
11    private TMP_InputField    inputFieldNickname;
12    [SerializeField]
13    private FadeEffect_TMP    textResult;
14
15    private void OnEnable()
16    {
17        // [친구 요청 대기] 목록 불러오기
18        backendFriendSystem.GetSentRequestList();
19    }
20
21    private void OnDisable()
22    {
23        DeactivateAll();
24    }
25
26    public void OnClickRequestFriend()...
27
28
29
30
31
32
33
34
35
36
37
38
39
40 }
```



친구 요청 대기 목록

- SentRequestPage 오브젝트의 "FriendSentRequestPage" 컴포넌트 변수 설정

The screenshot displays the Unity Inspector window for the **SentRequestPage** object. The **Friend Sent Request Page (Script)** component is selected, and its variables are being configured. Red boxes and arrows highlight the following settings:

- Backend Friend System:** BackendSystem (Backend Friend)
- Friend Page Base:** Friend_SentRequest (This prefab is also highlighted in the Assets > Prefabs panel on the right)
- Parent Content:** Content (Rect Transform)
- Text System:** TextSystem
- Send Request Friend:** InputFieldNickname (InputFieldBase (TMP_Input Field))
- Text Result:** TextResult (Fade Effect_TMP)

The Hierarchy panel on the left shows the **SentRequestPage** object selected under the **TabBarPages** folder. The Project panel on the right shows the **Assets > Prefabs** folder containing the **Friend_SentRequest** prefab.



친구 요청 대기 목록

- sentRequestPage.Activate() 메소드를 호출해 친구 요청 대기 목록 UI 출력
 - BackendFriendSystem Script 수정

```
1  +using ...
4
5  -public class BackendFriendSystem : MonoBehaviour
6  {
7      [SerializeField]
8      private FriendSentRequestPage sentRequestPage;
9
10     +private string GetUserInfoBy(string nickname)...
47
48     +public void SendRequestFriend(string nickname)...
68
69     -public void GetSentRequestList()
70     {
71         -Backend.Friend.GetSentRequestList(callback =>
72         {
73             +if ( !callback.IsSuccess() )...
78
79             // JSON 데이터 파싱 성공
80             try
81             {
82                 LitJson.JsonData jsonData = callback.GetFlattenJSON()["rows"];
83
84                 // 받아온 데이터의 개수가 0이면 데이터가 없는 것
85                 +if ( jsonData.Count <= 0 )...
90
```



친구 요청 대기 목록

□ BackendFriendSystem Script 수정 (계속)

```
91 // 친구 요청 대기 목록에 있는 모든 UI 비활성화
92 sentRequestPage.DeactivateAll();
93
94 foreach ( LitJson.JsonData item in jsonData )
95 {
96     FriendData friendData = new FriendData();
97
98     //friend.nickname = item.ContainsKey("nickname") == true ? item["nickname"].ToString() : "NONAME";
99     friendData.nickname = item["nickname"].ToString().Equals("True") ? "NONAME" : item["nickname"].ToString();
100     friendData.inDate = item["inDate"].ToString();
101     friendData.createdAt = item["createdAt"].ToString();
102
103     //Debug.Log(friendData.ToString());
104     // 현재 friend 정보를 바탕으로 친구 요청 대기 UI 활성화
105     sentRequestPage.Activate(friendData);
106 }
107
108 // JSON 데이터 파싱 실패
109 catch ( Exception e )
110 {
111     // try-catch 에러 출력
112     Debug.LogError(e);
113 }
114 });
115 }
116 }
```



친구 요청 대기 목록

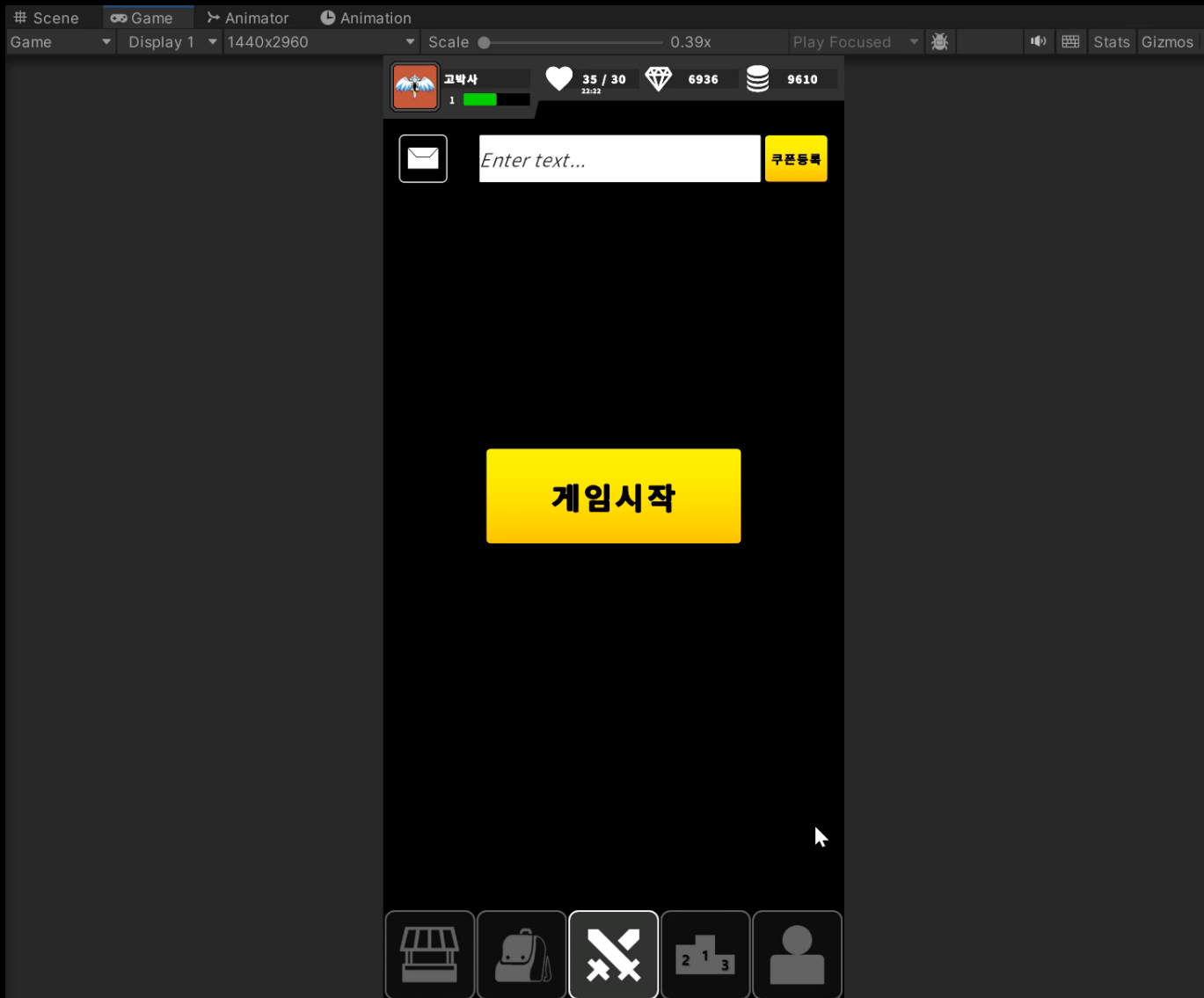
- BackendSystem 오브젝트의 "BackendFriendSystem" 컴포넌트 변수 설정

The screenshot displays the Unity Hierarchy and Inspector panels. In the Hierarchy panel, the **BackendSystem** object is selected at the bottom, and its **SentRequestPage** child is highlighted with a red box. An arrow points from this box to the Inspector panel. In the Inspector panel, the **BackendSystem** component is selected, and its **Backend Friend System (Script)** component is expanded. The **Sent Request Page** variable is set to **SentRequestPage (Friend Sent F**, which is also highlighted with a red dashed box. An **Add Component** button is visible below the Inspector panel.



친구 요청 대기 목록

■ 결과 화면





친구 요청 대기 목록

■ 요청 대기 중인 친구 UI 데이터 연동

■ 친구 요청대기, 수락대기 만료 기간

□ Constants Script 수정

```
1 public static class Constants
2 {
3     public static readonly string USER_DATA_TABLE = "USER_DATA";
4     public static readonly string DAILY_RANK_UUID = "0fa14490-085a-11ee-b506-7d5eaba96eff";
5     public static readonly int MAX_RANK_LIST = 20;
6
7     public static readonly string LEVEL_CHART = "82683";
8
9     public static readonly string GOODS_CHART_NAME = "재화차트";
10
11     public static readonly int EXPIRATION_DAYS = 3;
12 }
```



친구 요청 대기 목록

- 친구, 요청대기, 수락대기 UI가 상속받는 스크립트 생성 및 작성
 - C# Script 생성 후 스크립트의 이름을 "FriendBase"로 변경

```
1 using UnityEngine;
2     using TMPro;
3     using BackEnd;
4     using System;
5
6 public class FriendBase : MonoBehaviour
7 {
8     [Header("Friend Base")]
9     [SerializeField]
10    private TextMeshProUGUI textNickname;    // 닉네임
11    [SerializeField]
12    private TextMeshProUGUI textTime;        // 만료시간, 접속시간 등의 시간 정보
13
14    protected BackendFriendSystem backendFriendSystem;
15    protected FriendPageBase friendPage;
16    protected FriendData friendData;
17
```



친구 요청 대기 목록

- 친구, 요청대기, 수락대기 UI가 상속받는 스크립트 생성 및 작성 (계속)

```
18 public virtual void Setup(BackendFriendSystem friendSystem, FriendPageBase friendPage, FriendData friendData)
19 {
20     backendFriendSystem = friendSystem;
21     this.friendPage      = friendPage;
22     this.friendData      = friendData;
23
24     textNickname.text    = friendData.nickname;
25 }
26
27 public void SetExpirationDate()
28 {
29     // GetServerTime() - 서버 시간 불러오기
30     Backend.Utils.GetServerTime(callback =>
31     {
32         if ( !callback.IsSuccess() )
33         {
34             Debug.LogError($"서버 시간 불러오기에 실패했습니다. : {callback}");
35             return;
36         }
37     }
```



친구 요청 대기 목록

- 친구, 요청대기, 수락대기 UI가 상속받는 스크립트 생성 및 작성 (계속)

```
38 // JSON 데이터 파싱 성공
39 try
40 {
41     // createdAt 시간으로부터 3일 뒤의 시간
42     DateTime after3Days = DateTime.Parse(friendData.createdAt).AddDays(Constants.EXPIRATION_DAYS);
43     // 현재 서버 시간
44     string serverTime = callback.GetFlattenJSON()["utcTime"].ToString();
45     // 만료까지 남은 시간 = 만료 시간 - 현재 서버 시간
46     TimeSpan timeSpan = after3Days - DateTime.Parse(serverTime);
47
48     // timeSpan.TotalHours로 남은 기간을 시(hour)로 표현
49     textTime.text = $"{timeSpan.TotalHours:F0}시간 남음";
50 }
51 // JSON 데이터 파싱 실패
52 catch ( Exception e )
53 {
54     // try-catch 에러 출력
55     Debug.LogError(e);
56 }
57 });
58 }
59 }
```



친구 요청 대기 목록

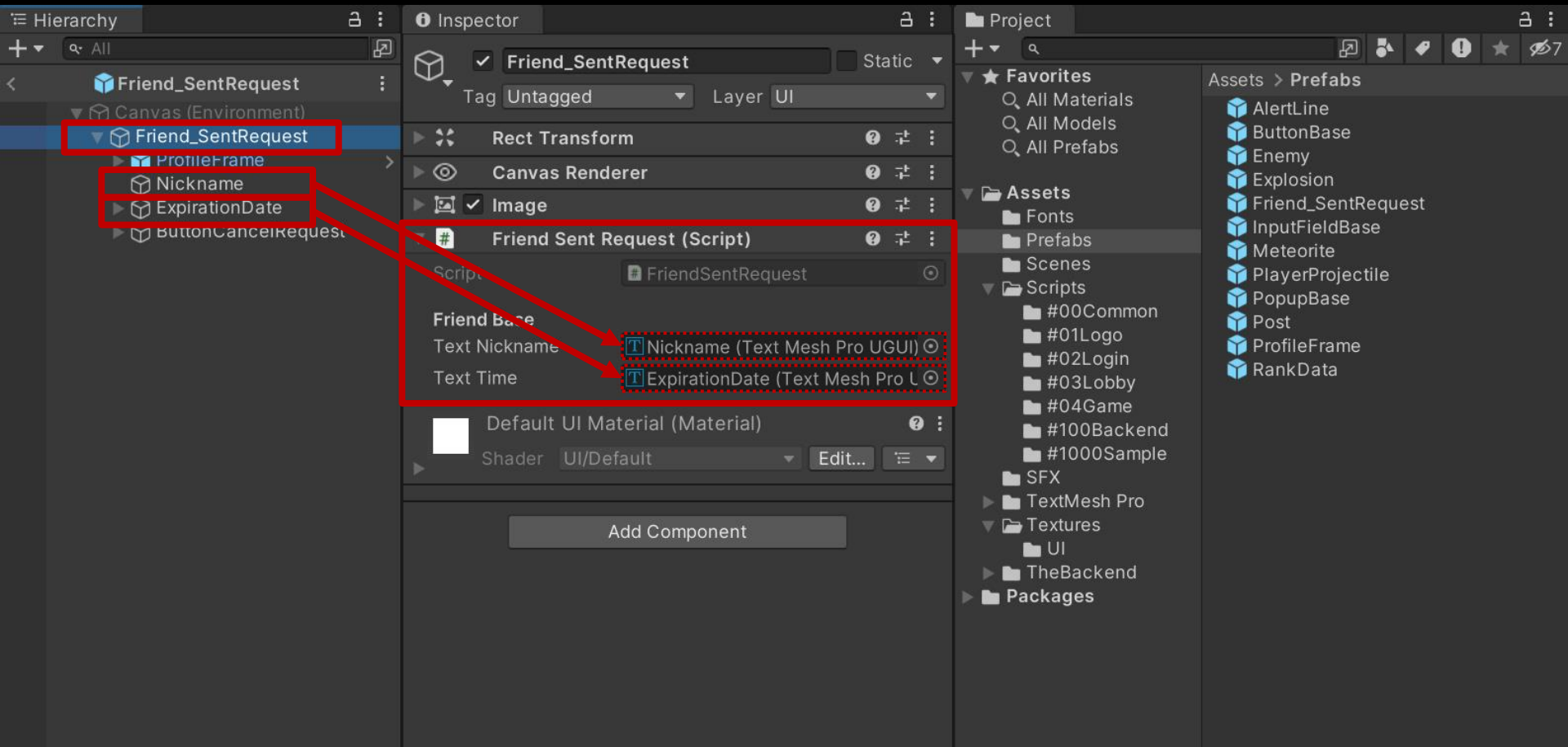
- 요청 대기 중인 친구 UI를 제어하는 스크립트 생성 및 작성
 - C# Script 생성 후 스크립트의 이름을 "FriendSentRequest"로 변경

```
1 public class FriendSentRequest : FriendBase
2 {
3     public override void Setup(BackendFriendSystem friendSystem, FriendPageBase friendPage, FriendData friendData)
4     {
5         base.Setup(friendSystem, friendPage, friendData);
6         base.SetExpirationDate();
7     }
8 }
```



친구 요청 대기 목록

- Friend_SentRequest 프리팹에 "FriendSentRequest" 컴포넌트 추가 및 설정





친구 요청 대기 목록

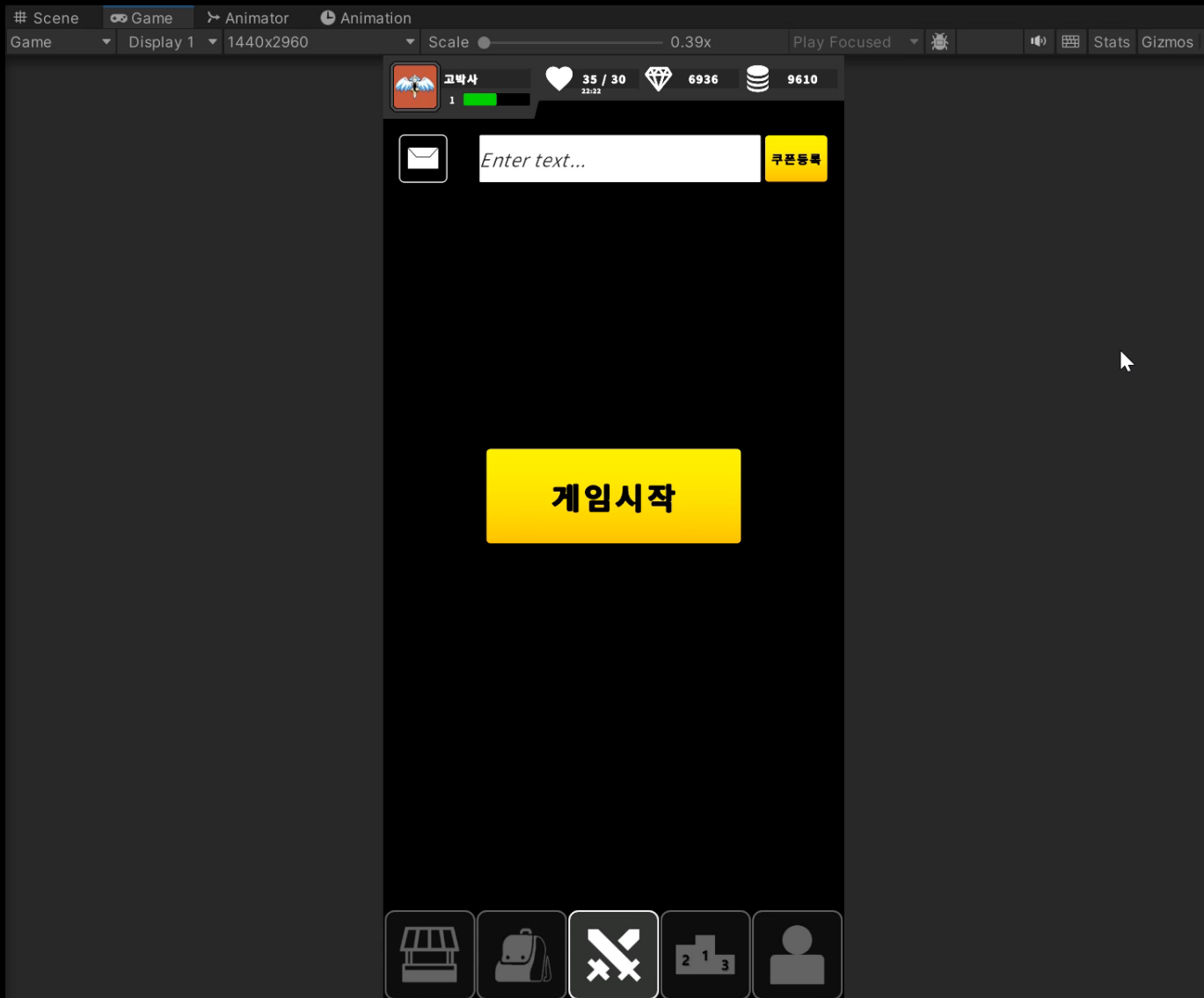
- 요청 대기 중인 친구 UI를 활성화할 때 FriendBase.Setup() 메소드 호출
 - FriendPageBase Script 수정

```
4 public class FriendPageBase : MonoBehaviour
5 {
6     [Header("Common")]
7     [SerializeField]
8     protected BackendFriendSystem backendFriendSystem;
9
10    [Header("Friend Page Base")]
11    [SerializeField]
12    private GameObject friendPrefab; // 해당 페이지에 출력하는 개별 친구 UI
13    [SerializeField]
14    private Transform parentContent; // UI가 배치되는 ScrollView의 Content
15    [SerializeField]
16    private GameObject textSystem; // 해당 페이지가 비어있을 때 출력하는 Text UI
17
18    private MemoryPool memoryPool;
19
20    private void Awake()...
24
25    public void Activate(FriendData friend)
26    {
27        if ( textSystem.activeSelf ) textSystem.SetActive(false);
28
29        GameObject item = memoryPool.ActivatePoolItem();
30        item.GetComponent<FriendBase>().Setup(backendFriendSystem, this, friend);
31    }
32
33    public void ActivateAll(List<FriendData> friendList)...
40
```



친구 요청 대기 목록

■ 결과 화면





친구 요청 대기 목록

■ 친구 요청 취소

- RevokeSentRequest() 메소드를 호출해 친구 요청 취소

- BackendFriendSystem Script 수정

```
1  +using ...
4
5  -public class BackendFriendSystem : MonoBehaviour
6  {
7      [SerializeField]
8      private FriendSentRequestPage    sentRequestPage;
9
10     +private string GetUserInfoBy(string nickname)...
47
48     +public void SendRequestFriend(string nickname)...
68
69     +public void GetSentRequestList()...
115
116     -public void RevokeSentRequest(string inDate)
117     {
118         Backend.Friend.RevokeSentRequest(inDate, callback =>
119         {
120             if ( !callback.IsSuccess() )
121             {
122                 Debug.LogError($"친구 요청 취소 도중 에러가 발생했습니다. : {callback}");
123                 return;
124             }
125
126             Debug.Log($"친구 요청 취소에 성공했습니다. : {callback}");
127         });
128     }
129 }
```



친구 요청 대기 목록

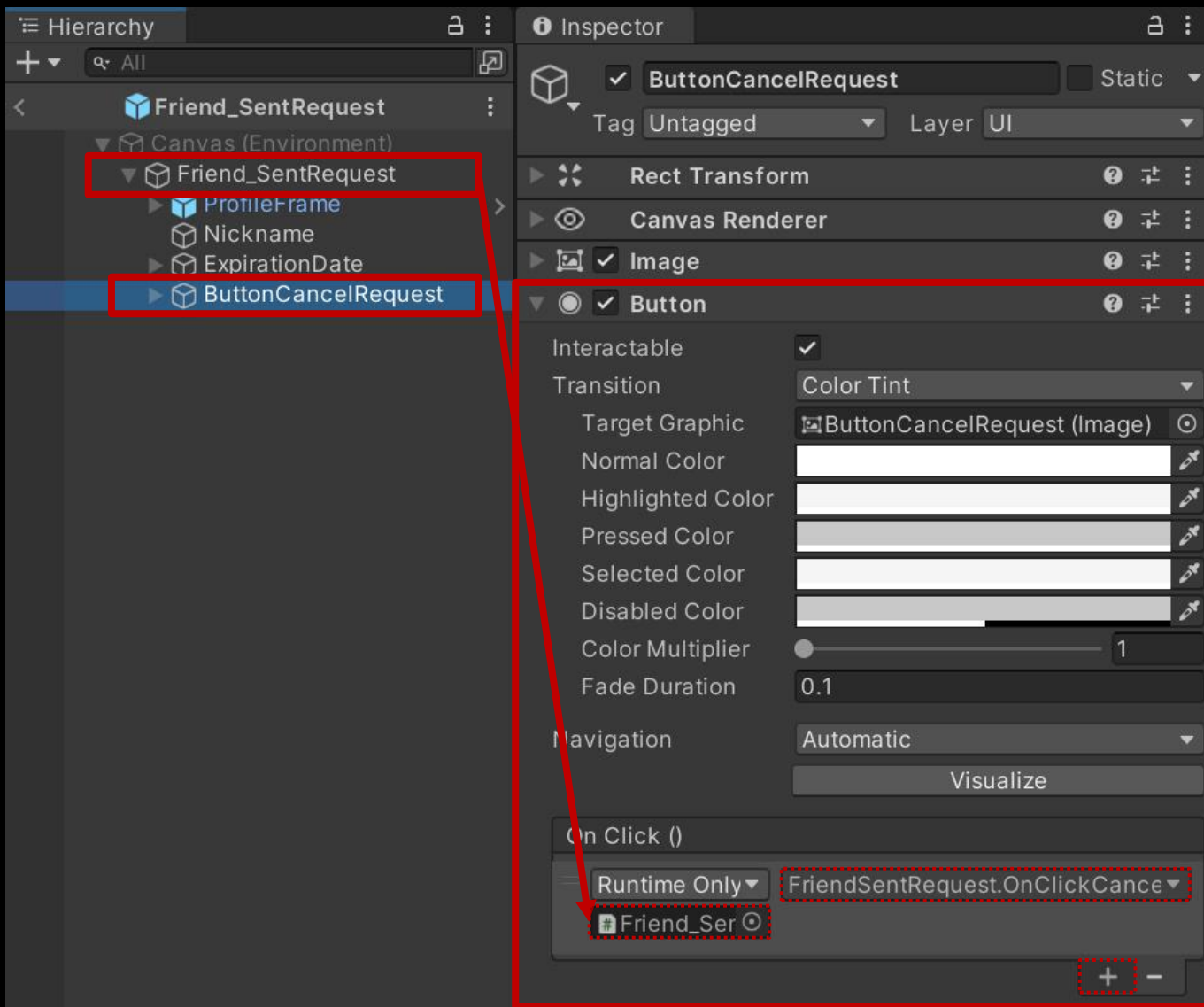
- “요청취소” 버튼을 눌렀을 때 호출할 `OnClickCancelRequest()` 메소드 정의
 - `FriendSentRequest` Script 수정

```
1 public class FriendSentRequest : FriendBase
2 {
3     public override void Setup(BackendFriendSystem friendSystem, FriendPageBase friendPage, FriendData friendData) ...
8
9     public void OnClickCancelRequest()
10    {
11        // 친구 UI 오브젝트 비활성화
12        friendPage.Deactivate(gameObject);
13        // 친구 요청 취소
14        backendFriendSystem.RevokeSentRequest(friendData.inDate);
15    }
16 }
```



친구 요청 대기 목록

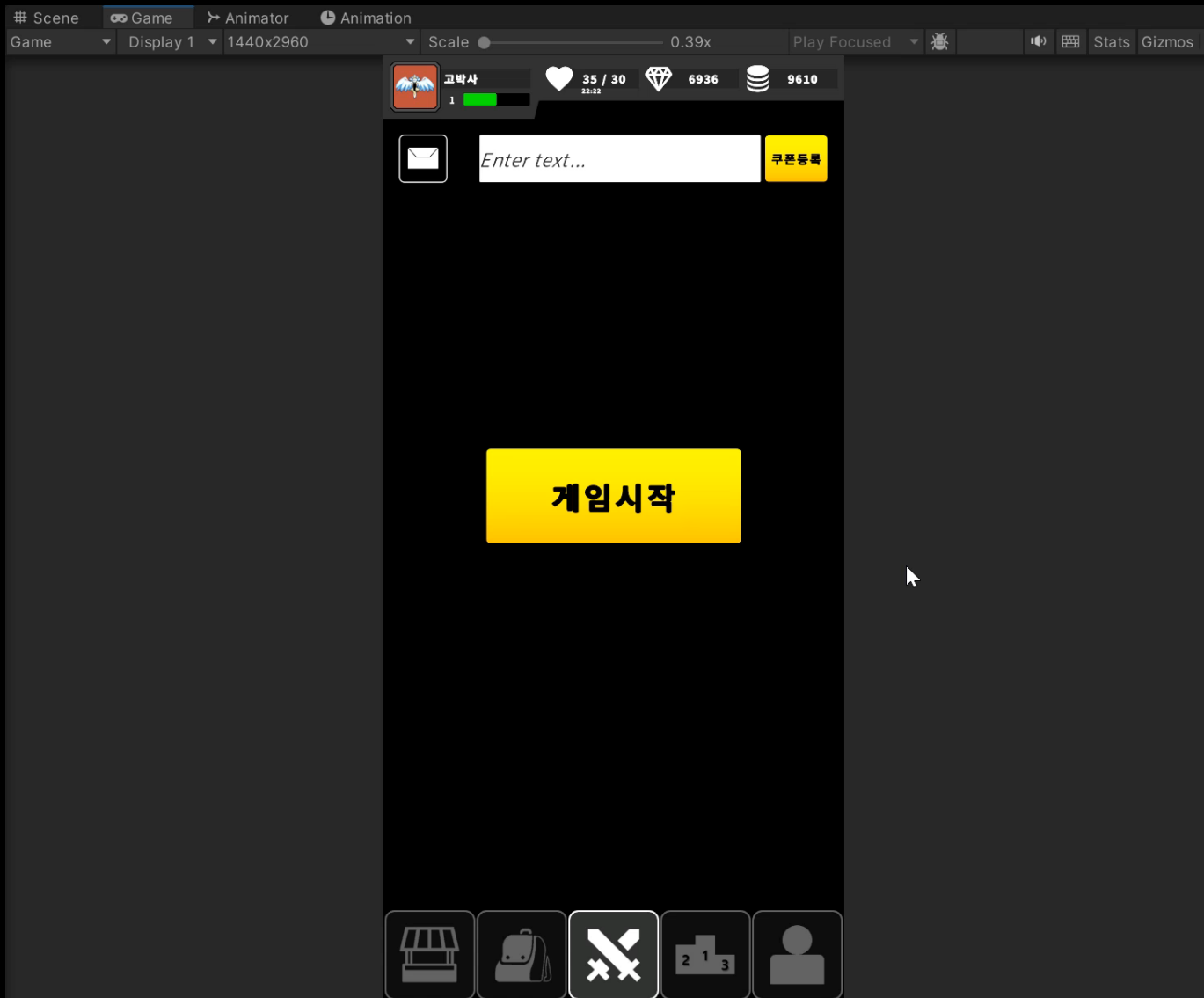
- ButtonCancelRequest 오브젝트의 "Button" 컴포넌트 onClick() 이벤트 등록





친구 요청 대기 목록

■ 결과 화면





친구 요청 대기 목록

■ 만료시간 이후 친구 요청 자동 취소

■ 요청 대기중인 친구 UI를 생성할 때 만료 기간 여부 검사

□ BackendFriendSystem Script 수정

```
69 public void GetSentRequestList()
70 {
71     Backend.Friend.GetSentRequestList(callback =>
72     {
73         if ( !callback.IsSuccess() )...
74     }
75     // JSON 데이터 파싱 성공
76     try
77     {
78         LitJson.JsonData jsonData = callback.GetFlattenJSON()["rows"];
79
80         // 받은 데이터의 개수가 0이면 데이터가 없는 것
81         if ( jsonData.Count <= 0 )...
82
83         // 친구 요청 대기 목록에 있는 모든 UI 비활성화
84         sentRequestPage.DeactivateAll();
85
86         foreach ( LitJson.JsonData item in jsonData )
87         {
88             FriendData friendData = new FriendData();
89
90             //friend.nickname = item.ContainsKey("nickname") == true ? item["nickname"].ToString() : "NONAME";
91             friendData.nickname = item["nickname"].ToString().Equals("True") ? "NONAME" : item["nickname"].ToString();
92             friendData.inDate = item["inDate"].ToString();
93             friendData.createdAt = item["createdAt"].ToString();
94
95             // [친구 요청]을 보낸 시간으로부터 일정 기간이 지났다면 자동으로 친구 요청 취소
96             if ( IsExpirationDate(friendData.createdAt) )
97             {
98                 RevokeSentRequest(friendData.inDate);
99                 continue;
100             }
101
102             // 현재 friend 정보를 바탕으로 친구 요청 대기 UI 활성화
103             sentRequestPage.Activate(friendData);
104         }
105     }
106 }
```



친구 요청 대기 목록


BackendFriendSystem Script 수정 (계속)

```
123 public void RevokeSentRequest(string inDate)...
136
137 private bool IsExpirationDate(string createdAt)
138 {
139     // GetServerTime() - 서버 시간 불러오기
140     var bro = Backend.Utills.GetServerTime();
141
142     if ( !bro.IsSuccess() )
143     {
144         Debug.LogError($"서버 시간 불러오기에 실패했습니다. : {bro}");
145         return false;
146     }
147
148     // JSON 데이터 파싱 성공
149     try
150     {
151         // createdAt 시간으로부터 3일 뒤의 시간
152         DateTime after3Days = DateTime.Parse(createdAt).AddDays(Constants.EXPIRATION_DAYS);
153         // 현재 서버 시간
154         string serverTime = bro.GetFlattenJSON()["utcTime"].ToString();
155         // 만료까지 남은 시간 = 만료 시간 - 현재 서버 시간
156         TimeSpan timeSpan = after3Days - DateTime.Parse(serverTime);
157
158         if ( timeSpan.TotalHours < 0 )
159         {
160             return true;
161         }
162     }
163     // JSON 데이터 파싱 실패
164     catch ( Exception e )
165     {
166         // try-catch 에러 출력
167         Debug.LogError(e);
168     }
169
170     return false;
171 }
172 }
```




친구 요청 대기 목록


■ 결과 화면


고박사
❤️ 35 / 30
22:22
💎 6936
🏠 9610

친구목록
요청대기
수락대기

친구요청


user04
72시간 남음
요청취소


user03
-12시간 남음
요청취소


고박사
❤️ 35 / 30
22:22
💎 6936
🏠 9610

친구목록
요청대기
수락대기

친구요청


user04
72시간 남음
요청취소



< 적용 전 >



< 적용 후 >