Documentation

Easy Discord Sign-In

Version 1.0

Quick Start

Welcome to Easy Discord Sign-In for Unity! With just a few simple steps, you can enable your players to sign in with their Discord account.

Discord Dev Portal

To get started, you will need to set up a Discord application and obtain a Discord API key:

1. Navigate to the Discord Developer Portal (https://discord.com/developers/applications).
2. Sign in with your Discord account. If you don't have a Discord account, you can create one at https://discord.com.
3. Click the "New Application" button.
4. Give your application a name and click "Create".
5. On the left side of the page, click the "OAuth2" tab.
6. Click the "Copy" button next to the "Client ID" field to copy the client ID to your clipboard. You will need this later when setting up the asset in Unity.
7. Click the "Generate a New Secret" button to create a new client secret. Click the "Copy" button next to the "Client Secret" field to copy the client secret to your clipboard. You will also need this later when setting up the asset in Unity.
8. Under "Redirects", add a Redirect URL. This is the URL that Discord will redirect to after a user has successfully signed in. The Redirect URL should point to the index.html file included in the package.

Landing Page Setup

The way the asset works is it opens up a browser window for Discord's authentication flow, using the client information and scopes specified in the inspector window. Once the user has granted authorization, they are redirected to your redirect URL which also has a **code** parameter. This code is necessary for our subsequent access token request. Typically, games and apps will run a server instance that handles the passing of the code from the browser back to the client application. To avoid the need for a server instance, we utilize the user's clipboard for passing the code.

The included landing page index.html file handles grabbing the code from the URL and showing a Confirm button, which upon being clicked, copies the code to the user's clipboard and asks them to return to the game. Once back at the game (Where the Easy Discord script listens for the OnApplicationFocus event), we grab the code from the clipboard and continue the authorization process from within the Unity app.

There are two things to note:

- We use a button for copying the code rather than directly calling the clipboard.writeText function to avoid permission popups in most browsers. When the user makes an intentional interaction, most browsers will not prompt the user any further.
- On mobile, both iOS and Android require the website to use SSL (https) in order for the copy-to-clipboard function to work. For security reasons, we recommend all users use SSL regardless.

Unity Editor Setup

Once you have completed all the previous steps, simply add the Easy Discord Sign-In script to a gameobject in your scene. Enter in your Discord client credentials into the fields.

Pay close attention to the redirect url, since this needs to perfectly match the redirect url you set up on your Discord application page.

For scopes, select the ones your application requires. On its own, this asset only uses Identify, Email and (optionally) Guilds. The other scopes are included for convenience. With the scopes chosen, create a button and add a Click event triggering the SignIn() method in the script. Alternatively, you can call it from a separate script of your own. The Easy Discord Sign In class is a Singleton and accessible via EasyDiscordSignIn.Instance.

Enabling the toggle “Print Debug Logs” will let you know if your calls are working correctly. This can be useful when initially setting up the asset.

Finally, the expandable events tab gives you some easy-access events for

onSignInBegin,
onSignInEnd,
onSignInSuccess
and **onSignInFailed**.

You can use these events to act on the results of the sign-in process; Such as showing / hiding different views. For a more in-depth example of how this might look, check out the included Demo scene.

The demo scene was also designed to be able to be integrated directly into your project. You can enter your Discord details into the Easy Discord Sign In component, and then simply include the Demo scene (you may rename it) in your Build Settings.

** By default, the demo log-in button will try to go to the next available scene index.*

Components

EDUserInfoDisplay

Easy Discord User Info Display component is a script that helps you easily grab and display user info in your scene, without having to write additional code. Once authorized, this component can grab and display user info such as email, username etc passed through a UnityEvent.

EDAvatarDisplay

Easy Discord Avatar Display component is a script that helps you easily grab and display the user's avatar image, without having to write additional code. This also uses a UnityEvent that allows you to easily set a Texture or Sprite. Using the Sprite event incurs a small, additional performance cost because it will create a new sprite from the fetched avatar texture.
We therefore recommend using the Raw Image component and the Texture2D event over the Image component and Sprite event whenever possible.

EDEnforceGuild

Easy Discord Enforce Guild component is a script that allows you to easily check if the user is a member of one or more Discord communities (Guilds). Unlike the previously mentioned Display components, this one doesn't run automatically but instead requires you to call the public EnforceGuild() method, either via script or another event handler.

For the Demo scene, one example of use could be final Log-In button calling EnforceGuild, which then depending on the outcome, invokes either

onUserInGuild

or **onUserNotInGuild**.

The onUserInGuild event could then be set up to call yet another method that continues to the next scene, and onUserNotInGuild could be set up to show an error message.

** This helper component does very simple client-side enforcing of the Discord Guild and could easily be bypassed by those with the technical know-how. For truly secure verification, you would want to run it on a separate server instance somewhere.*

** The demo scene is set up with the EDEnforceGuild component, but it is not being called by default. Instead the Log-In button calls the scene switching method directly.*

For Advanced Users

If you are an experienced user, you probably won't find much use for the included UnityEvents and helper components, but instead want to call the various methods directly yourself.

The way the code is structured, the static DiscordRequest class offers some generic methods for interfacing with the Discord REST API. The lowest level includes Get, Post, Patch and Delete, where you can call it with the class type representing the return object.
You can find JSON examples of the return data in many places in the official Discord developer documentation, and there are many tools online for converting JSON objects to C# classes that may save you time if you are not familiar with the structure of JSON already and how Newtonsoft JSONConvert handles them.

The DiscordRequest class also has some more higher level methods specifically around the authorization flow. These were primarily written to help simplify the flow within the EasyDiscordSignIn class, but might also come in handy if you are customizing your own sign-in flow.