

BDG Chat for Unity

User Guide and Script Reference, Version 1.0.0-alpha

Publisher: XStreamIO. Product site: <https://bgdchat.com>. Live API: <https://aponteplace.io>. License: MIT.

This document is the offline setup guide required for Asset Store packages that include scripts. It is written in English and is intended for Unity developers who want to add hosted real-time chat to a game.

Table of Contents

1. What this package does
2. Requirements
3. Importing the package
4. Running the demo (step by step)
5. Using your own studio session
6. Integrating BDGChatClient into a game
7. Minting player tokens on a game server
8. Script reference
9. WebSocket protocol
10. Error codes
11. Limitations and support

1. What this package does

BDG Chat is a Unity client for hosted real-time chat. You do not run a chat server in the game. Your studio account owns one or more apps. Each app has a public identifier called **app_id**. Players connect with a short-lived player JWT. Rooms and message history are isolated per **app_id**, so two games never share a lobby.

Do not put an **sk_live_** secret in a shipped player build. That secret belongs on a trusted game server. The Unity client stores only the player JWT and talks to <wss://aponteplace.io/ws>.

2. Requirements

- 1 Unity 2021.3 or newer. This package was tested on Unity 6 (6000.5).
- 2 A project that can compile C# scripts. No extra packages are required for the demo (it uses built-in IMGUI).
- 3 Editor or standalone player (Windows, macOS, or Linux). WebGL is not supported in this version.
- 4 An internet connection so the client can reach <https://aponteplace.io>.

- 5 Optional: a studio account at <https://bgdchat.com> if you want private rooms instead of the public demo.

3. Importing the package

- 1 Create or open a Unity project that matches the versions in section 2.
- 2 Import the .unitypackage, or copy the BDGChat and BDGChatDemo folders into your Assets directory.
- 3 Wait for Unity to compile. The Console must show no errors related to XStreamIO.BDGChat.
- 4 Confirm these folders exist: Assets/BDGChat (runtime) and Assets/BDGChatDemo (demo scene and prefab).

4. Running the demo (step by step)

- 1 In the Project window, open BDGChatDemo/DemoScene (or PlayDemo if that scene is present).
- 2 Press Play.
- 3 The demo requests a public app_id and player JWT from <https://aponteplace.io/demo/player-token>. Wait until the log says that a player JWT was received.
- 4 Click Connect, then Join. The default room is lobby.
- 5 Type a message and click Send. You should see your line in the log. Other people using the public demo share that lobby.
- 6 When you are finished, click Leave or Close, then stop Play mode.

The on-screen panel states that this is a shared public demo. To use your own session, go to bgdchat.com to subscribe, or buy a license to run the server on your own machines. The Open bgdchat.com button opens the pricing page.

5. Using your own studio session

- 1 Open <https://bgdchat.com/account> in a browser.
- 2 Register with email, password, and display name, or log in if you already have an account.
- 3 Copy your app_id from the Apps list. It looks like app_ followed by hexadecimal characters.
- 4 Save the sk_live_ secret if it is shown. It is displayed only once, when the app is created. Never paste that secret into the Unity client.
- 5 Mint a player JWT on a trusted machine (see section 7), or for a quick editor test use the login JWT from the account page (browser console: `copy(localStorage.getItem("bdg_token"))`).
- 6 Paste the app_id and the JWT into the demo fields, or assign them on BDGChatClient in the Inspector, then Connect and Join.

6. Integrating BDGChatClient into a game

- 1 Create an empty GameObject in your gameplay scene.
- 2 Add the BDGChatClient component (Add Component, search for BDGChatClient).
- 3 Set App Id and Player Token from your game server at runtime. Do not type `sk_live_` into those fields.
- 4 Subscribe to events before you connect, then call `ConnectToChat()`. When `Connected` fires, call `Join` with your room name.
- 5 Call `SendChat` to post a line. Call `Leave` or `Close` when the player exits the room or the match.

```
var chat = gameObject.AddComponent<XStreamIO.BDGChat.BDGChatClient>();
chat.AppId = "app_XXXXXXX";
chat.PlayerToken = playerJwtFromYourGameServer;
chat.Connected += () => chat.Join("lobby");
chat.ChatReceived += (text, name, id, ts, room) =>
    Debug.Log(name + ": " + text);
chat.ErrorReceived += (code, message) =>
    Debug.LogWarning(code + ": " + message);
chat.ConnectToChat();
```

7. Minting player tokens on a game server

Your game server (or an editor tool you do not ship) calls the HTTP API with the studio secret. The Unity helper `BDGPlayerTokens.Mint` wraps that call. You may also use `curl` or any HTTP client.

```
POST https://aponteplace.io/apps/{app_id}/player-tokens
Authorization: Bearer sk_live_...
Content-Type: application/json

{ "display_name": "PlayerOne", "player_id": "steam_123" }
```

```
// C#
var minted = await XStreamIO.BDGChat.BDGPlayerTokens.Mint(
    appId: "app_XXXXXXX",
    secret: "sk_live_...",
    displayName: "PlayerOne",
    playerId: "steam_123");
// minted.token is the player JWT for BDGChatClient
```

Player JWTs expire after 24 hours. Developer login JWTs expire after 7 days. Mint a new player token when a session starts.

8. Script reference

8.1 Namespace

XStreamIO.BDGChat — runtime client and token helper. XStreamIO.BDGChat.Samples — editor demo only.

8.2 BDGChatClient (MonoBehaviour)

Add this component to a GameObject. It opens a WebSocket to the hosted API, sends join and chat messages, and raises C# events on the main thread. It pings every 25 seconds while connected.

Member	Type	Description
WsUrl	string	WebSocket URL. Default: wss://aponteplace.io/ws
AppId	string	Public app identifier from your studio account.
PlayerToken	string	Player JWT. Must not start with sk_live_.
IsSocketOpen	bool	True while the WebSocket is open.
CurrentRoom	string	Room name after a successful join.
ConnectToChat()	void	Opens the WebSocket. Fails if app_id or token is missing.
Join(string room)	void	Sends a join message. Empty room becomes lobby.
SendChat(string text)	void	Sends a chat line. Ignored if text is empty.
Leave()	void	Leaves the current room.
Close()	void	Closes the WebSocket.

Event	Arguments	When it fires
Connected	none	WebSocket handshake succeeded.
Disconnected	int code, string reason	Socket closed or connect failed.
Joined	room, user_id, display_name, users_online	Server accepted join.
ChatReceived	text, display_name, user_id, ts, room	A chat line arrived.
UserJoined	user_id, display_name	Another user entered the room.
UserLeft	user_id, display_name	Another user left the room.
ErrorReceived	code, message	Client or server error.
RawEvent	string json	Every inbound JSON payload.

8.3 BDGPlayerTokens

Static helper for trusted servers. `Mint(appId, secret, displayName, playerId, apiUrl)` returns a `PlayerTokenResponse` with `token`, `app_id`, `player_id`, and `display_name`. Throws `InvalidOperationException` if the HTTP call fails.

8.4 PlayerTokenResponse

Serializable class with fields `token`, `app_id`, `player_id`, `display_name`, and `error`. Used by `Mint` and by the demo credential fetch.

9. WebSocket protocol

Connect: `wss://aponteplace.io/ws`

```
Join: { "type":"join", "room":"lobby", "app_id":"app_xxx", "token":"PLAYER_JWT" }
Chat: { "type":"chat", "text":"hello" }
Leave: { "type":"leave" }
Ping: { "type":"ping" }
```

Inbound types: `joined`, `chat`, `user_joined`, `user_left`, `error`, `pong`

10. Error codes

Code	Meaning
<code>app_required</code>	<code>AppId</code> was empty on <code>ConnectToChat</code> .
<code>auth_required</code>	<code>PlayerToken</code> was empty.
<code>secret_in_client</code>	You pasted an <code>sk_live_</code> secret. Use a player JWT instead.
<code>connect_failed</code>	The WebSocket could not open.
<code>not_connected</code>	<code>Join</code> or <code>SendChat</code> was called before the socket was open.
<code>invalid_token</code>	The JWT is invalid or expired.
<code>app_not_found</code>	The <code>app_id</code> does not exist on the server.
<code>app_forbidden</code>	The token does not own that <code>app_id</code> .
<code>app_mismatch</code>	A player token was used with a different <code>app_id</code> .
<code>plan_limit</code>	The app hit its plan limit for connections, rooms, or messages.

11. Limitations and support

- 1 WebGL is not supported. `ClientWebSocket` is not available in that player.
- 2 The public demo lobby is shared. Do not put production secrets or private player data there.
- 3 This package talks to the hosted API at `aponteplace.io` unless you change `WsUrl` after buying a self-host license.

- 4 Documentation on the web: <https://bgdchat.com/docs>
- 5 Godot addon (same protocol): <https://store.godotengine.org/asset/xstreamio/bdg-chat/>
- 6 Company: <https://xstreamio.com>

Copyright (c) 2026 XStreamIO / Misael Aponte. Released under the MIT License.