API Reference

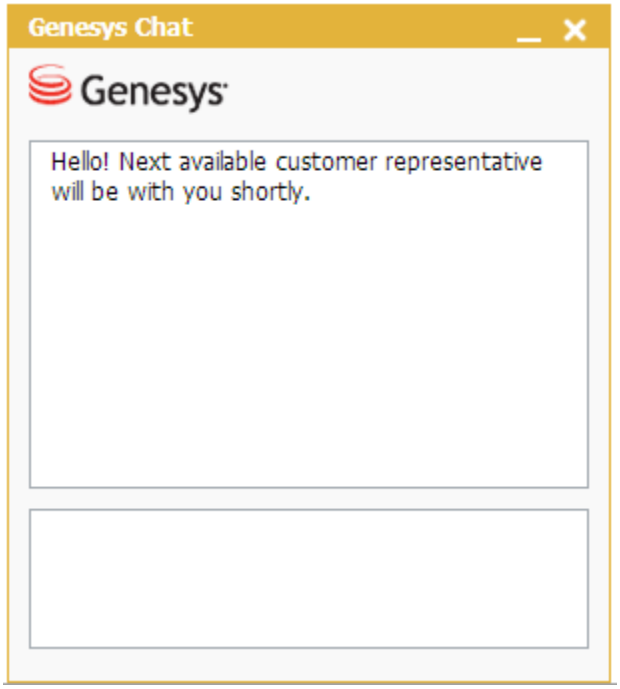
Chat API

5/4/2025

Contents

Chat API

Co-browse is shipped with a built-in chat widget. Out-of-the-box, the chat widget looks like this:



To configure the chat widget, see Integrated JavaScript Application#Configuring Chat.

To get access to the Chat Widget API, see Configuration API# Accessing the Co-browse and Chat APIs. You generally will not need to access the Chat Widget API as configuration can be done in instrumentation. The Chat Widget API can be used to get access to the lower lever Chat API. See Advanced Usage below for more details.

For a full Chat Widget API reference, see Chat Widget JS API.

Advanced Usage of the Chat API

Getting Access to the Lower Level Chat API

The Chat Widget API is built on top of the Chat Service JS API. The Chat Service API allows you to perform actions such as programmatically sending messages to a chat or subscribing to session events such as `agentConnected` and `messageReceived`.

To get access to the Chat Service JS API, you will have to:

1. Get access to the Chat Widget API.
2. Use the Chat Widget API to obtain the Chat Service API.

The following code example shows how you can access the Chat Service API. Note that this example is a bit simplistic in that it starts chat unconditionally on every page load and does not handle errors.

```
var _genesys = {
    chat: {
        // 1. Tell the integrated application not to call restoreChat(),
        //    because you will call it manually.
        autoRestore: false,
        // 2. Subscribe to chat widget's "ready" event
        //    to get access to the widget API.
        onReady: function(chat) {
            // 3. Use chat widget API to get access to the service API.
            chat.restoreChat().done(function(session) {
                // Chat session is restored, use
                // chat service API here. E.g.:
                // session.sendMessage('Hello World!');
            }).fail(function() {
                chat.startChat().done(function(session) {
                    // Chat session was not restored so
                    // we started a new one. Use the API here. E.g.:
                    // session.sendMessage('Hello World!');
                });
            });
        }
    }
};
```

Using the Lower Level Chat API as a Separate JavaScript Bundle

If you want to build your own chat widget on top of the Genesys Chat API, you can use the special JavaScript file shipped with Co-browse. This file is available at the following URL:

`http(s)://<COBROWSE_SERVER>/cobrowse/js/chatAPI.min.js`

When loaded in a browser, this file exports the Chat Service JS API as a global `chat` variable. The size of this file is 113 KB (~35 KB gzipped) and it does not require any dependencies.

Another version of this file is available at `http(s)://<COBROWSE_SERVER>/cobrowse/js/chatAPI-noDeps.min.js`. The size of this file is 23 KB (~8 KB gzipped), but it requires that you have the following libraries globally available:

- `$` for jQuery (v. 1.8.1 or higher)
- `_` for underscore (v. 1.5.0 or higher) or lodash (v. 2.0.0 or higher)
- `org.cometd` for Cometd (v. 2.8.0)

> ### Important
> If you choose to implement your own chat widget using the Chat Service JS API in the form of a seperate JS file, your chat widget `will not be automatically integrated`

with Co-browse. Integration consists of two features:

- Co-browse automatically determines if the user is on chat when the user starts a Co-browse session.
- The Co-browse session token is automatically passed to an agent.

To support these integration features, you will also have to implement the External Media Adapter API for your chat widget and pass the implementation object to the Configuration API `primaryMedia` option.