Broadcast (and receive) a message | WeWeb Documentation

|  |  |
| --- | --- |
| URL source | <https://docs.weweb.io/websockets/supabase-realtime/broadcast-message.html> |
| Date de scraping | 2025-04-08T15:56:14.451504+00:00 |

## Broadcast messages with Supabase realtime ​

## Pre-requisites ​

Before users of your WeWeb app can send and receive Supabase realtime broadcast messages, they need to subscribe to a realtime broadcast channel .  
Once that's done, they can start sending & receiving realtime broadcast messages.

## Broadcast a message ​

To allow users to send messages in a channel, use the Broadcast a message action:  
  
In the example above, you can see:  
We triggered the Broadcast a message action. To send a message to a channel we named room1 . We said these messages should be called a chat event. We configured the payload of the event to be an object with the user's name and the text from a chat input.  
Note that the Event name can be whatever you like. It's important you remember it if you want to filter on specific events later, when you start listening and reacting to events.  
When the Broadcast a message action is successful, the workflow Logs will simply inform you that the message was sent:  
  
The action sends the realtime event to the channel but that's it.  
Receiving (and displaying) messages is a separate matter.

### Images

<https://docs.weweb.io/assets/supabase-realtime-broadcast-message.CzSaS67q.png>

<https://docs.weweb.io/assets/supabase-realtime-broadcast-message-error.fxgARhg5.png>

<https://docs.weweb.io/assets/supabase-realtime-broadcast-message-success.Bi70X6kr.png>

## Receive broadcast events ​

If we want users in the channel to receive messages from other users, we need to create a page or app workflow that is triggered when there's a new realtime broadcast event in a channel.  
In the example below, we setup a workflow on a page with a chatroom:  
The workflow is triggered when there's a new broadcast event in the room1 channel. To help us debug things, we used the Logs action and bound it to the entire workflow Event object. Then we updated a variable with the payload of the workflow Event .

### Images

<https://docs.weweb.io/assets/supabase-realtime-listen-broadcast.CAXamZWA.png>

<https://docs.weweb.io/assets/supabase-realtime-broadcast-message.CzSaS67q.png>

## Display broadcast events ​

Broadcast events are ephemereal by design (see Supabase docs on the topic).  
To display these events in your WeWeb app, you'll want to update a variable every time there's a new event and bind that variable to an item on your page. That way, your UI can react dynamically to the event it receives.  
For example, in the case of a live collaboration tool that tracks cursor positions, you might have a cursors array variable with a seperate objet for every active user. Each object would include the user's cursor position. On your page, you could use that variable to move cursor icons around the page based on each user's position.

## Realtime debugger ​

In Supabase, you can open the Realtime page to check if messages go through ok:

### Images

<https://docs.weweb.io/assets/supabase-realtime-debug.C5gfD-22.gif>