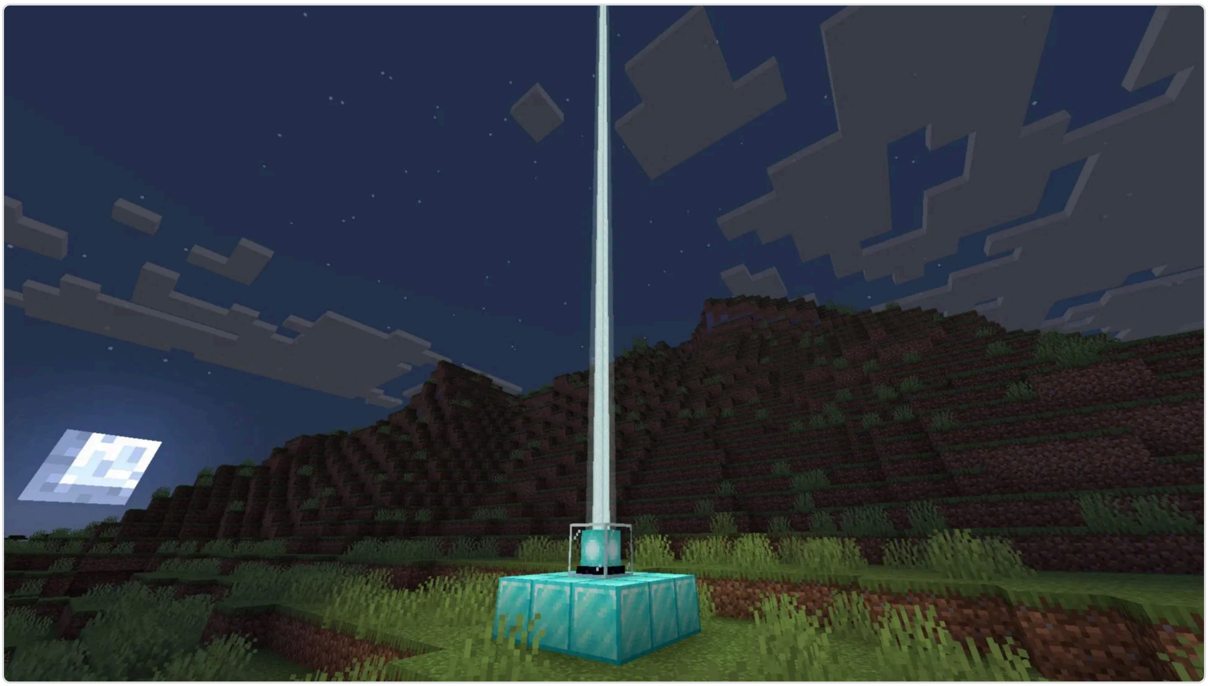
Hemath's Blog ☘️

# Say bye with JavaScript Beacon

Sun Aug 24, 2025 Engineering Learning JavaScript



Sometimes we want to send a piece of data to our servers when user leaves our website or webapp. Maybe it's for for analytics or even auto-logout when they leave the website. But do you know what is a reliable way of doing it?

Most of you might say use XMLHTTPRequest (or fetch) in **`beforeunload`** or **`unload`** events. Like,

```
window.addEventListener("beforeunload", () => {
  fetch('/analytics', {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify({event: 'page_leave'})
    })
  });
```

Well, in my experience, this is not that reliable. Sometimes it works, sometimes it doesn't. There must be a reliable way to do such a trivial task.

You might ask, why not we have an event listener for the browser tab? We can easily listen for that event and do our operations right?

Because, users can leave your website by just changing the URL to some other website or clicking a bookmark. Don't expect users to be polite :)

But why beforeunload doesn't work reliably? The reason is that browser cannot wait for some JavaScript to execute before moving on. It will be a bad experience for user to have delay when they close the tab or move to a different website. So it gives sometime and hardly stops the exeuction of JavaScript.

So when we send an network request to our backend, there is a high chance that browser either will not send it or will cancel it.

That's where Beacon API in Javascript comes in handy. It is like, fire-and-forget. Browser doesn't need to wait for anything, once the request is sent, the JavaScript execution will be done immediately, nothing will be kept in the memory.

The API itself is very easy to work with. Look at the following example,

```
  navigator.sendBeacon('/analytics', JSON.stringify({ event: 'page_leave' }));
```

That's really all there is to it. No callbacks or promises.

However, there is some limitations. Only a tiny amount of data can be sent through it. And it only supports POST request.

But hey, for it’s intended use, tiny important message sent as the page closes, it works beautifully.

And the great thing is that not only need to be used when closing the website. It can send data to server anytime. Very good for syncing backend with frontend.

Say an user types something and that needs to be stored in the server somewhere. If we use regular XMLHTTPRequest (or fetch, or axios), the browser will wait for the response. And that might be unnecessary for our usecase.

Take analytics as an example. We don’t really need to wait for the response, we just send data to the server and that’s it.

It’s subtle and reliable!

For more info about Beacon API, check out MDN - https://developer.mozilla.org/en-US/docs/Web/API/Beacon_API

---

BRAIN MADE.ORG