How to paste files

- Using the HTMLElement's paste event
- Further reading
- Demo

Pasting files into the browser consists of using the `paste` event of the `HTMLElement`.

Using the `HTMLElement`'s `paste` event

As a first step, you add an event listener for the `paste` event at the desired element, commonly this is at the `document` level, so no specific element needs to be focused. Next, you use the Clipboard API, which gives you access to the `clipboardData` field of the `HTMLElement`'s `paste` event, whose `files` list you can then iterate over. Based on the MIME type of each of the files pasted, you can decide whether to render it to the screen, as in the case of an image or video, or to paste the text contents of the file into, for example, a `textarea` element, in the case of a text file.

{% BrowserCompat 'api.Clipboard' %}

```
document.addEventListener('paste', async (e) => {
  // Prevent the default behavior, so you can code your own logic.
  e.preventDefault();
  if (!e.clipboardData.files.length) {
    return;
  }
  // Iterate over all pasted files.
  Array.from(e.clipboardData.files).forEach(async (file) => {
    // Add more checks here for MIME types you're interested in,
    // such as `application/pdf`, `video/mp4`, etc.
    if (file.type.startsWith('image/')) {
```

```
      // For images, create an image and append it to the `body`.
      const img = document.createElement('img');
      const blob = URL.createObjectURL(file);
      img.src = blob;
      document.body.append(img);
    } else if (file.type.startsWith('text/')) {
      // For text files, read the contents and output it into a `textarea`.
      const textarea = document.createElement('textarea');
      textarea.value = await file.text();
      document.body.append(textarea);
    }
  });
});
```

Further reading

- Clipboard API
- Unblocking clipboard access

Demo

□□□ #2
□ □□□ □□□ 2026-10-31 23:01:10 UTC
□ □□□ □□□ 2026-10-31 23:01:11 UTC