

 [Yaxian](#) / [vue3-dropzone](#) Public

HTML5 drag-drop zone with vue3

 MIT License


 30 stars  3 forks


 Star



 Notifications

 **Code**

 Issues 1


 Pull requests

 Actions


 Projects

 Wiki

 Security

 main ▾

Go to file

 **Yaxian** docs: update doc ...

on Nov 5, 2021 ⌚ 23

[View code](#)

 README.md

Releases 5

vue3-dropzone



v0.0.7

Latest

on Mar 4, 2021

It's inspired by [react-dropzone](#) and implemented with vue3.
+ 4 releases

Installation

Pack

```
npm install --save vue3-dropzone
```

No pa

or

User'

```
yarn add vue3-dropzone
```

Usage



`acceptFiles` is an array returned in the same format as [FileList](#) where all the dropped files are turned into a [File class](#) before saving to the array.

Cont `<template>`



```
<div>
  <div v-bind="getRootProps()">
    <input v-bind="getInputProps()" />
    <p v-if="isDragActive">Drop the files here ...</p>
    <p v-else>Drag 'n' drop some files here, or click to select files</p>
  </div>
  <button @click="open">open</button>
</div>
```

Lang `</template>`

```
• Tj <script>
import { useDropzone } from "vue3-dropzone";

export default {
  name: "UseDropzoneDemo",
  setup() {
    function onDrop(acceptFiles, rejectReasons) {
      console.log(acceptFiles);
      console.log(rejectReasons);
    }

    const { getRootProps, getInputProps, ...rest } = useDropzone({ onDrop });

    return {
      getRootProps,
      getInputProps,
      ...rest,
    };
  },
};
</script>
```

🔗 Save multiple files

Save multiple files through [axios requests](#) and [FormData](#). You will need a backend to loop through the received files and save them individually in the loop.


```
<template>
  <div>
    <div v-bind="getRootProps()">
      <input v-bind="getInputProps()" />
      <p v-if="isDragActive">Drop the files here ...</p>
      <p v-else>Drag 'n' drop some files here, or click to select files</p>
    </div>
    <button @click="open">open</button>
  </div>
</template>

<script>
import { useDropzone } from "vue3-dropzone";
import axios from "axios";

export default {
  name: "UseDropzoneDemo",
  setup() {
    const url = "{your_url}"; // Your url on the server side
    const saveFiles = (files) => {
      const formData = new FormData(); // pass data as a form
      for (var x = 0; x < files.length; x++) {
        // append files as array to the form, feel free to change the array name
        formData.append("images[]", files[x]);
      }

      // post the formData to your backend where storage is processed. In the bac

      axios
        .post(url, formData, {
          headers: {
            "Content-Type": "multipart/form-data",
          },
        })
        .then((response) => {
          console.info(response.data);
        })
        .catch((err) => {
          console.error(err);
        });
    };

    function onDrop(acceptFiles, rejectReasons) {
      saveFiles(acceptFiles); // saveFiles as callback
      console.log(rejectReasons);
    }
  }
}
```

```
const { getRootProps, getInputProps, ...rest } = useDropzone({ onDrop });

return {
  getRootProps,
  getInputProps,
  ...rest,
};
},
};
</script>
```

API

```
const result = useDropzone(options)
```

options

property	type	description
onDrop	Function	Cb for when the drop event occurs. Note that this callback is invoked after the <code>getFilesFromEvent</code> callback is done.
accept	String / Array<*String>	Set accepted file types. See https://github.com/okonet/attr-accept for more information.
disabled	Boolean	Enable/disable the dropzone
maxSize	Number	Maximum file size (in bytes)
minSize	Number	Minimum file size (in bytes)
multiple	Number	Allow of multiple files
maxFiles	Number	Maximum accepted number of files The default value is 0 which means there is no limitation to how many files are accepted

property	type	description
getFilesFromEvent	Function	Use this to provide a custom file aggregator
onDragEnter	Function	Cb for when the dragenter event occurs.
onDragenter	Function	Cb for when the dragenter event occurs.
onDragOver	Function	Cb for when the dragover event occurs
onDragover	Function	Cb for when the dragover event occurs
onDragLeave	Function	Cb for when the dragleave event occurs
onDragleave	Function	Cb for when the dragleave event occurs
onDropAccepted	Function	Cb for when the drop event occurs. Note that if no files are accepted, this callback is not invoked.
onDropRejected	Function	Cb for when the drop event occurs. Note that if no files are rejected, this callback is not invoked.
onFileDialogCancel	Function	Cb for when closing the file dialog with no selection
preventDropOnDocument	Boolean	If <code>false</code> , allow dropped items to take over the current browser window
noClick	Boolean	If <code>true</code> , disables click to open the native file selection dialog
noKeyboard	Boolean	If <code>true</code> , disables SPACE/ENTER to open the native file selection

property	type	description
		dialog. Note that it also stops tracking the focus state.
noDrag	Boolean	If <code>true</code> , disables drag 'n' drop
noDragEventsBubbling	Boolean	If <code>true</code> , stops drag event propagation to parents

[↗](#) result

property	type	description
isFocused	Ref<Boolean>	
isFileDialogActive	Ref<Boolean>	
isDragActive	Ref<Boolean>	
isDragAccept	Ref<Boolean>	
isDragReject	Ref<Boolean>	
draggedFiles	Ref<Array>	dragged files
acceptedFiles	Ref<Array>	accepted files
fileRejections	Ref<Array>	files rejections
getRootProps	Function	Function to generate element props which contains input
getInputProps	Function	Function to generate input props
rootRef	Ref<*HTMLElement>	ref a dom element
inputRef	Ref<*HTMLElement>	ref a input element
open	Function	Open file selection dialog

[↗](#) Run example

```
cd examples  
yarn install  
yarn dev
```