










FileuploaderService

[FileuploaderService.php](#) : 14

Table of Contents

-  [\\$configuration](#) : array<string|int, mixed>
-  [\\$key](#) : string
-  [\\$processors](#) : array<string|int, [FileuploaderProcessorInterface](#)>
-  [__construct\(\)](#) : mixed
-  [handlePreload\(\)](#) : array<string|int, mixed>
-  [handleRemove\(\)](#) : array<string|int, mixed>
Handles removing files
-  [handleUpload\(\)](#) : array<string|int, mixed>
Handles uploading files
-  [isMultiple\(\)](#) : bool
-  [validate\(\)](#) : bool

Properties

\$configuration

[FileuploaderService.php](#) : 24

```
protected array<string|int, mixed> $configuration
```

\$key

[FileuploaderService.php](#) : 19

```
protected string $key
```

\$processors

[FileuploaderService.php](#) : 29

```
protected array<string|int, FileuploaderProcessorInterface> $processors
```

Methods

__construct()

[FileuploaderService.php](#) : 31

```
public __construct(string $key, array<string|int, mixed> $configuration) :  
mixed
```

Parameters

\$key : string

\$configuration : array<string|int, mixed>

Return values

mixed —

handlePreload()

[FileuploaderService.php](#) : 48

```
public handlePreload(array<string|int, string> $files) : array<string|int,  
mixed>
```

Parameters

\$files : array<string|int, string>

Tags

throws

[FileuploaderProcessingException](#)

throws

[FileuploaderFileNotFoundException](#)

Return values

array<string|int, mixed> —

handleRemove()

[FileuploaderService.php](#) : 116

Handles removing files

```
public handleRemove(string $fileToRemove, array<string|int, mixed> $uploadedFiles, string $checksum) : array<string|int, mixed>
```

Parameters

\$fileToRemove : string

\$uploadedFiles : array<string|int, mixed>

\$checksum : string

Tags

throws

[FileuploaderInvalidChecksumException](#)

throws

[FileuploaderFileNotFoundException](#)

throws

[FileuploaderFileNotContainedException](#)

Return values

array<string|int, mixed> —

handleUpload()

[FileuploaderService.php](#) : 74

Handles uploading files

```
public handleUpload(array<string|int, mixed> $files, array<string|int, mixed>|null $previouslyUploadedFiles, string|null $checksum) : array<string|int, mixed>
```

Parameters

\$files : array<string|int, mixed>

An array of files to be uploaded

\$previouslyUploadedFiles : array<string|int, mixed>|null

An array of file paths of previously uploaded files

\$checksum : string|null

The checksum of the previously uploaded files

Tags

throws

[FileuploaderInvalidChecksumException](#)

throws

[FileuploaderProcessingException](#)

throws

[FileuploaderFileNotFoundException](#)

Return values

array<string|int, mixed> —

isMultiple()

[FileuploaderService.php](#) : 159

```
public isMultiple() : bool
```

Return values

bool —

validate()

[FileuploaderService.php](#) : 141

```
public validate(array<string|int, mixed> $uploadedFiles, string $checksum)  
: bool
```

Parameters

\$uploadedFiles : array<string|int, mixed>

\$checksum : string

Tags

throws

[FileuploaderFileNotFoundException](#)

Return values

bool —

Class Fileuploader

The core fileuploader. Holds settings and state and handled all actions on a specific fileuploader.

Hierarchy

- Fileuploader

Index

Constructors

[constructor](#)

Properties

config	inputElement	url
dispose	key	
hiddenInput	store	

Methods

afterUpload	on	sendRequest
beforeUpload	remove	sendUploadRequest
initPreloadedFiles	removeFile	upload
initialize	sendRemoveRequest	validate

Constructors

constructor

[new](#) Fileuploader(inputElement: *HTMLInputElement*): *Fileuploader*

Defined in src/Fileuploader.ts:99

Parameters

· **inputElement:** *HTMLInputElement*

Returns *Fileuploader*

Properties

Private **Readonly** **config**

🔗 **config**: *Configuration*

Defined in `src/Fileuploader.ts:66`

The fileuploader's configuration. Comes from the backend and is located in the `data-config` attribute of the file input.

readonly

dispose

🔗 **dispose**: `() => void`

Defined in `src/Fileuploader.ts:99`

The dispose function to destroy this instance

Type declaration

```
Ⓢ (): void
```

Returns *void*

Private **Readonly** **hiddenInput**

🔗 **hiddenInput**: *HTMLInputElement*

Defined in `src/Fileuploader.ts:81`

The corresponding `input[type="hidden"]` of this Fileuploader instance. Hold the stringified files array in its value which gets sent with the form on submit.

readonly

Private **Readonly** **inputElement**

🔗 **inputElement**: *HTMLInputElement*

Defined in `src/Fileuploader.ts:73`

The corresponding `input[type="file"]` of this Fileuploader instance.

readonly

Readonly **Private** key

`key: string`

Defined in `src/Fileuploader.ts:58`

The unique fileuploader key. Comes from the backend and is located in the data-key attribute.

`readonly`

Private **Readonly** store

`store: FileuploaderStore`

Defined in `src/Fileuploader.ts:95`

The store that holds the fileuploaders' state

`readonly`

Private **Readonly** url

`url: string`

Defined in `src/Fileuploader.ts:89`

The url where to upload the files via AJAX. Comes from the backend.

`readonly`

Methods

Private afterUpload

`afterUpload(files: FileuploaderItem[], responseFiles: ResponseFile[]): void`

Defined in `src/Fileuploader.ts:350`

Process files after uploading

Parameters

- `files: FileuploaderItem[]`
- `responseFiles: ResponseFile[]`

Returns `void`

Private beforeUpload

`beforeUpload(files: FileuploaderItem[]): void`

Defined in `src/Fileuploader.ts:337`

Process files before uploading

Parameters

• **files:** [FileuploaderItem](#)[]

Returns *void*

Private initPreloadedFiles

`initPreloadedFiles(): void`

Defined in `src/Fileuploader.ts:154`

Initialize preloaded files

Returns *void*

Private initialize

`initialize(): () => void`

Defined in `src/Fileuploader.ts:124`

Initialize method Initialize the files store and the upload handler

Returns `() => void`

⦿ `()`: *void*

Returns *void*

on

```
on(subscribers: Subscribers): () => void
```

Defined in `src/Fileuploader.ts:308`

The public API to add different listeners. Available are `pending`, `change`, `progress` and `error`.

- `uploading` – gets triggered when an upload action is pending or has finished
- `removing` – gets triggered when a remove action is pending or has finished
- `pending` – gets triggered when an upload/remove action is pending or has finished
- `change` – gets triggered when files are added
- `progress` – gets triggered when the progress of an upload changes
- `error` – gets triggered when files are invalid or an error occurs

[see](#) <https://mobx.js.org/reactions.html>

[see](#) [Subscribers](#)

[example](#)

```
// Using ES6 modules
import {getFileuploader} from 'form4-fileuploader';
const myFileuploader = getFileuploader('my_fileuploader');
const unsubscribe = myFileuploader.on({
  uploading(uploading) {
    // Show/Hide a progressbar
  },
  removing(removing) {
    // Something that could happen while removing a file
  },
  pending(pending) {
    // Do something e.g showing a loading spinner
  },
  change(files) {
    // Update the UI
  },
  progress(progress) {
    // Show/Update a progressbar
  },
  error(error) {
    if(error.files) {
      // Tell the user that some files are invalid
    }
    else if(error.cause) {
      // Some specific error occurred
    }
    else {
      // Oops something went wrong...
    }
  }
});
// later ...
// This will dispose the subscription
unsubscribe();
```

example

```
// Using the bundled version
const myFileuploader =
Form4Fileuploader.getFileuploader('my_fileuploader');
const unsubscribe = myFileuploader.on({
  uploading(uploading) {
    // Show/Hide a progressbar
  },
  removing(removing) {
    // Something that could happen while removing a file
  },
  pending(pending) {
    // Do something e.g showing a loading spinner
  },
  change(files) {
    // Update the UI
  },
  progress(progress) {
    // Show/Update a progressbar
  },
  error(error) {
    if(error.files) {
      // Tell the user that some files are invalid
    }
    else if(error.cause) {
      // Some specific error occurred
    }
    else {
      // Oops something went wrong...
    }
  }
});
// later ...
// This will dispose the subscription
unsubscribe();
```

Parameters

· subscribers: [Subscribers](#)

Returns () => void

Ⓢ(): void

Returns void

Private remove

```
remove(file: FileuploaderItem): Promise<string>
```

Defined in src/Fileuploader.ts:414

The remove handler

async

Parameters

· **file:** [FileuploaderItem](#)

Returns *Promise<string>*

removeFile

```
removeFile(id: string): void
```

Defined in src/Fileuploader.ts:328

Public API for removing a file.

Parameters

· **id:** *string*

The identifier of the file

Returns *void*

Private sendRemoveRequest

```
sendRemoveRequest(file: string): Promise<RemoveResponse>
```

Defined in src/Fileuploader.ts:461

Sends the remove request to the server with the file path.

Parameters

· **file:** *string*

The file path on the server

Returns *Promise<[RemoveResponse](#)>*

Private sendRequest

```
sendRequest(formData: FormData): Promise<Response>
```

Defined in src/Fileuploader.ts:476

A helper method for making requests

Parameters

• **formData:** *FormData*

Returns *Promise<Response>*

Private sendUploadRequest

```
sendUploadRequest(files: FileuploaderItem[]):  
  Promise<UploadResponse>
```

Defined in src/Fileuploader.ts:447

Send files to the server files

async

Parameters

• **files:** *FileuploaderItem*[]

Returns *Promise<UploadResponse>*

Private upload

```
upload(files: FileuploaderItem[]): Promise<string>
```

Defined in src/Fileuploader.ts:375

Prepares files for uploading and sends valid files

async

Parameters

• **files:** *FileuploaderItem*[]

Returns *Promise<string>*

validate

```
validate(): void
```

Defined in src/Fileuploader.ts:361

Process files on submit

throws Error

Returns *void*