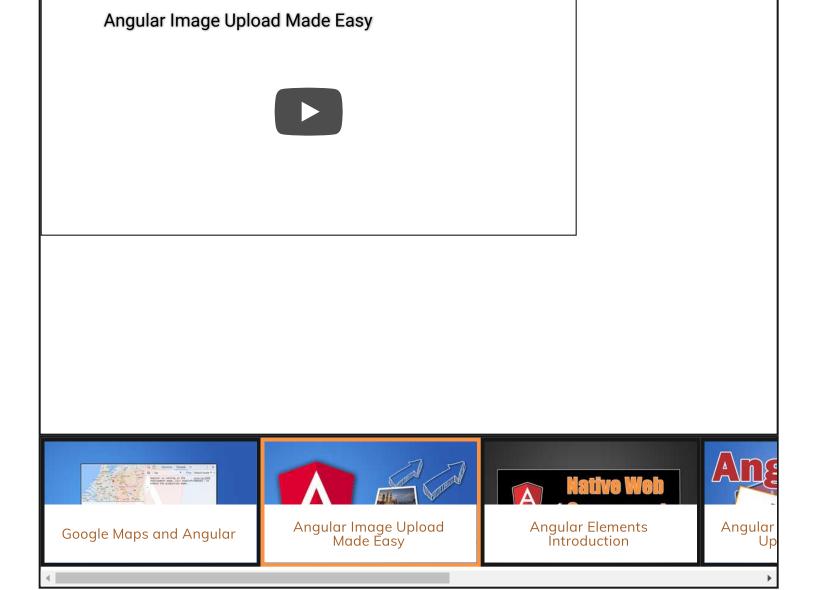


Special Offer: Boost your career with our all-access membership and get all our current and future courses for one price!

EXPLORE NOW!





Created by Maximilian Schwarzmüller Last Updated on February 12, 2018





Angular Image Upload

Uploading images basically is a two-step process:

- 1. Select a file
- Send it to a server



Angular - The Complete Guide



Use it with the MEAN Stack

Select a File

Selecting a file requires us to have a <input type="file"> element in the Angular component template.

The key thing is to react to file changes made by the user. A change occurs if the user selects a file which differs from the currently selected one (null initially).

We can react to changes like this:

<input type="file" (change)="onFileChanged(\$event)">



```
export class MyFileUploadComponent {
  onFileChanged(event) {
    const file = event.target.files[0]
  }
}
```

If you want to take a more elegant route and hide the file picker, you could implement the following code in your template:

```
<input
   style="display: none"
   type="file" (change)="onFileChanged($event)"
   #fileInput>
<button (click)="fileInput.click()">Select File</button>
```

You can of course also add an additional button which should then start the upload process:

```
<input
   style="display: none"
   type="file" (change)="onFileChanged($event)"
   #fileInput>
<button (click)="fileInput.click()">Select File</button>
<button (click)="onUpload()">Upload!</button>
```

In the component TypeScript file, you could handle it like this:

```
export class MyFileUploadComponent {
  selectedFile: File

  onFileChanged(event) {
    this.selectedFile = event.target.files[0]
  }

  onUpload() {
    // upload code goes here
```



Send the File to the Server

We have the file, to send it to the server, we can use the Angular HttpClient.

We can either send the file as a binary or as part of a FormData object - whatever your REST API endpoint supports/ expects.

Send as binary data

```
onUpload() {
   // this.http is the injected HttpClient
   this.http.post('my-backend.com/file-upload', this.selectedFile)
        .subscribe(...);
}
```

Send as FormData

```
onUpload() {
   // this.http is the injected HttpClient
   const uploadData = new FormData();
   uploadData.append('myFile', this.selectedFile, this.selectedFile.name);
   this.http.post('my-backend.com/file-upload', uploadData)
        .subscribe(...);
}
```

Listen to upload progress

In both cases, you can listen to the upload progress by editing the above code like this:

```
onUpload() {
    ...
    this.http.post('my-backend.com/file-upload', uploadData, {
        reportProgress: true,
        observe: 'events'
```



```
});
```

Make sure to watch the video above this article to see it all in action and to see the finished code.



Angular - The Complete Guide



Use it with the MEAN Stack

IMPRESSUM & DATENSCHUTZ (DE)

IMPRINT & DATA PRIVACY (EN)