

Uppy

npm v1.27.0




Tests	Tests passing	Companion passing	End-to-end tests failing
Deploys	CDN passing	Companion Deploy failing	Deploy uppy.io passing

Uppy is a sleek, modular JavaScript file uploader that integrates seamlessly with any application. It's fast, easy to use and lets you worry about more important problems than building a file uploader.

- **Fetch** files from local disk, remote URLs, Google Drive, Dropbox, Box, Instagram or snap and record selfies with a camera
- **Preview** and edit metadata with a nice interface
- **Upload** to the final destination, optionally process/encode

Drop files here, paste, [browse](#)
or import from



Powered by  Uppy

[Read the docs](#) | [Try Uppy](#)



Developed by
Transloadit

Uppy is being developed by the folks at [Transloadit](#), a versatile file encoding service.

Example

Code used in the above example:

```
const Uppy = require('@uppy/core')
const Dashboard = require('@uppy/dashboard')
const GoogleDrive = require('@uppy/google-drive')
const Instagram = require('@uppy/instagram')
const Webcam = require('@uppy/webcam')
const Tus = require('@uppy/tus')

const uppy = new Uppy({ autoProceed: false })
  .use(Dashboard, { trigger: '#select-files' })
  .use(GoogleDrive, { target: Dashboard, companionUrl: 'https://companion.uppy.io' })
  .use(Instagram, { target: Dashboard, companionUrl: 'https://companion.uppy.io' })
  .use(Webcam, { target: Dashboard })
```

```
.use(Tus, { endpoint: 'https://tusd.tusdemo.net/files/' })
.on('complete', (result) => {
  console.log('Upload result:', result)
})
```

[Try it online](#) or [read the docs](#) for more details on how to use Uppy and its plugins.

Features

- Lightweight, modular plugin-based architecture, easy on dependencies :zap:
- Resumable file uploads via the open [tus](#) standard, so large uploads survive network hiccups
- Supports picking files from: Webcam, Dropbox, Box, Google Drive, Instagram, bypassing the user's device where possible, syncing between servers directly via [@uppy/companion](#)
- Works great with file encoding and processing backends, such as [Transloadit](#), works great without (just roll your own Apache/Nginx/Node/FFmpeg/etc backend)
- Sleek user interface :sparkles:
- Optional file recovery (after a browser crash) with [Golden Retriever](#)
- Speaks multiple languages (i18n) :earth_africa:
- Built with accessibility in mind
- Free for the world, forever (as in beer 🍺, pizza 🍕, and liberty 🗽)
- Cute as a puppy, also accepts cat pictures :dog:

Installation

```
$ npm install @uppy/core @uppy/dashboard @uppy/tus
```

We recommend installing from npm and then using a module bundler such as [Webpack](#), [Browserify](#) or [Rollup.js](#).

Add CSS [uppy.min.css](#), either to your HTML page's `<head>` or include in JS, if your bundler of choice supports it — transforms and plugins are available for Browserify and Webpack.

Alternatively, you can also use a pre-built bundle from Transloadit's CDN: Edgly. In that case `Uppy` will attach itself to the global `window.Uppy` object.

⚠ The bundle currently consists of most Uppy plugins, so this method is not recommended for production, as your users will have to download all plugins when you

are likely using just a few.

```
<!-- 1. Add CSS to `<head>` -->
<link href="https://releases.transloadit.com/uppy/v1.27.0/uppy.min.css" rel="styl

<!-- 2. Add JS before the closing `</body>` -->
<script src="https://releases.transloadit.com/uppy/v1.27.0/uppy.min.js"></script>

<!-- 3. Initialize -->
<div class="UppyDragDrop"></div>
<script>
  var uppy = Uppy.Core()
  uppy.use(Uppy.DragDrop, { target: '.UppyDragDrop' })
  uppy.use(Uppy.Tus, { endpoint: '//tusd.tusdemo.net/files/' })
</script>
```

Documentation

- [Uppy](#) — full list of options, methods and events
- [Plugins](#) — list of Uppy plugins and their options
- [Companion](#) — setting up and running a Companion instance, which adds support for Instagram, Dropbox, Box, Google Drive and remote URLs
- [React](#) — components to integrate Uppy UI plugins with React apps
- [Architecture & Writing a Plugin](#) — how to write a plugin for Uppy

Plugins

[List of plugins and their common options](#)

UI Elements

- [Dashboard](#) — universal UI with previews, progress bars, metadata editor and all the cool stuff. Required for most UI plugins like Webcam and Instagram
- [Progress Bar](#) — minimal progress bar that fills itself when upload progresses
- [Status Bar](#) — more detailed progress, pause/resume/cancel buttons, percentage, speed, uploaded/total sizes (included by default with [Dashboard](#))
- [Informer](#) — send notifications like “smile” before taking a selfie or “upload failed” when all is lost (also included by default with [Dashboard](#))

Sources

- `Drag & Drop` — plain and simple drag and drop area
- `File Input` — even plainer “select files” button
- `Webcam` — snap and record those selfies 📷
- © `Google Drive` — import files from Google Drive
- © `Dropbox` — import files from Dropbox
- © `Box` — import files from Box
- © `Instagram` — import images and videos from Instagram
- © `Facebook` — import images and videos from Facebook
- © `OneDrive` — import files from Microsoft OneDrive
- © `Import From URL` — import direct URLs from anywhere on the web

The © mark means that `@uppy/companion`, a server-side component, is needed for a plugin to work.

Destinations

- `Tus` — resumable uploads via the open `tus` standard
- `XHR Upload` — regular uploads for any backend out there (like Apache, Nginx)
- `AWS S3` — plain upload to AWS S3 or compatible services
- `AWS S3 Multipart` — S3-style “Multipart” upload to AWS or compatible services

File Processing

- `Robodog` — user friendly abstraction to do file processing with Transloadit
- `Transloadit` — support for `Transloadit`’s robust file uploading and encoding backend

Miscellaneous














- `Golden Retriever` — restores files after a browser crash, like it’s nothing
- `Thumbnail Generator` — generates image previews (included by default with `Dashboard`)
- `Form` — collects metadata from `<form>` right before an Uppy upload, then optionally appends results back to the form
- `Redux` — for your emerging `time traveling` needs


React

- `React` — components to integrate Uppy UI plugins with React apps

- [React Native](#) — basic Uppy component for React Native with Expo

Browser Support

 Android	 Firefox	 Chrome	 IE	 Edge	 Safari
6.0  * ✓	80  10 ✓	80  10 ✓	10  8 ✓	85  10 ✓	11  10.13 ✓
			11  10 ✓		

TESTING POWERED BY 

We aim to support IE11 and recent versions of Safari, Edge, Chrome, Firefox and Opera.

We still run end-to-end tests with IE10, but we are not actively supporting it or fixing visual / minor issues.

Polyfills

Uppy heavily uses Promises. If your target environment [does not support Promises](#), use a polyfill like `es6-promise` before initializing Uppy.

When using remote providers like Google Drive or Dropbox, the Fetch API is used. If your target environment does not support the [Fetch API](#), use a polyfill like `whatwg-fetch` before initializing Uppy. The Fetch API polyfill must be loaded *after* the Promises polyfill, because Fetch uses Promises.

With a module bundler, you can use the required polyfills like so:

```
npm install es6-promise whatwg-fetch
```

```
require('es6-promise/auto')
require('whatwg-fetch')
const Uppy = require('@uppy/core')
```

If you're using Uppy from CDN, `es6-promise` and `whatwg-fetch` are already included in the bundle, so no need to include anything additionally:

```
<script src="https://releases.transloadit.com/uppy/v1.27.0/uppy.min.js"></script>
```

FAQ

Why not just use `<input type="file">` ?

Having no JavaScript beats having a lot of it, so that's a fair question! Running an uploading & encoding business for ten years though we found that in cases, the file input leaves some to be desired:

- We received complaints about broken uploads and found that resumable uploads are important, especially for big files and to be inclusive towards people on poorer connections (we also launched tus.io to attack that problem). Uppy uploads can survive network outages and browser crashes or accidental navigate-aways.
- Uppy supports editing meta information before uploading (and e.g. cropping is planned).
- There's the situation where people are using their mobile devices and want to upload on the go, but they have their picture on Instagram, files in Dropbox or just a plain file URL from anywhere on the open web. Uppy allows to pick files from those and push it to the destination without downloading it to your mobile device first.
- Accurate upload progress reporting is an issue on many platforms.
- Some file validation — size, type, number of files — can be done on the client with Uppy.
- Uppy integrates webcam support, in case your users want to upload a picture/video/audio that does not exist yet :)
- A larger drag and drop surface can be pleasant to work with. Some people also like that you can control the styling, language, etc.
- Uppy is aware of encoding backends. Often after an upload, the server needs to rotate, detect faces, optimize for iPad, or what have you. Uppy can track progress of this and report back to the user in different ways.
- Sometimes you might want your uploads to happen while you continue to interact on the same single page.

Not all apps need all of these features. An `<input type="file">` is fine in many situations. But these were a few things that our customers hit / asked about enough to spark us to develop Uppy.

Why is all this goodness free?

Transloadit's team is small and we have a shared ambition to make a living from open source. By giving away projects like tus.io and [Uppy](https://uppy.io), we're hoping to advance the state of the art, make life a tiny little bit better for everyone and in doing so have rewarding jobs and get some eyes on our commercial service: [a content ingestion & processing platform](https://transloadit.com).

Our thinking is that if just a fraction of our open source userbase can see the appeal of hosted versions straight from the source, that could already be enough to sustain our work. So far this

is working out! We're able to dedicate 80% of our time to open source and haven't gone bankrupt yet. :D

Does Uppy support React?

Yep, we have Uppy React components, please see [Uppy React docs](#).

Does Uppy support S3 uploads?

Yes, there is an S3 plugin, please check out the [docs](#) for more.

Do I need to install a special service/server for Uppy? Can I use it with Rails/Node/Go/PHP?

Yes, whatever you want on the backend will work with `@uppy/xhr-upload` plugin, since it just does a `POST` or `PUT` request. Here's a [PHP backend example](#).

If you want resumability with the Tus plugin, use [one of the tus server implementations](#)

And you'll need `@uppy/companion` if you'd like your users to be able to pick files from Instagram, Google Drive, Dropbox or via direct URLs (with more services coming).

Contributions are welcome







- Contributor's guide in `.github/CONTRIBUTING.md`
- Changelog to track our release progress (we aim to roll out a release every month): `CHANGELOG.md`







Used by







Uppy is used by: [Photobox](#), [Issuu](#), [Law Insider](#), [Cool Tabs](#), [Soundoff](#), [Scrumi](#), [Crive](#) and others.







Use Uppy in your project? [Let us know!](#)

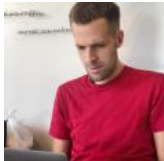





Contributors





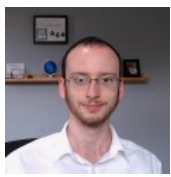

					
arturi	goto-bus-stop	kvz	ifedapoolarewaju	hedgerh	AJvanLoon

					
nqst	lakesare	kiloreux	sadovnychi	samuelayo	richardwillars







					
ajkachnic	zcallan	tim-kos	janko	wilkoklak	dependabot[bot]


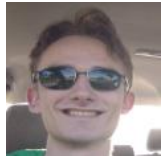



					
oliverpool	Botz	mcallistertyler	mokutsu-coursera	DJWassink	taoqf






					
mifi	tuoxiansp	elenalape	gavboulton	berth-zero	tranvansang



					
ap--	mrbatista	MikeKovarik	pauln	szh	toadkicker


					
ofhope	dominiceden	mejiaej	johnnyperkins	dargmuesli	manuelkiessling






					
nndevstudio	ogtfaber	sksavant	suchoproduction	sunil-shrestha	timodwhit

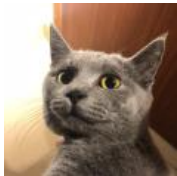





					
yonahforst	stephentuso	mskelton	ahmedkandel	btrice	behnammodi







					
Burkes	craigjennings11	davekiss	DenysNosov	ethanwillis	frobinsonj





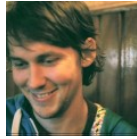
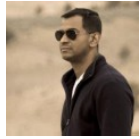
					
geertclerx	jasonbosco	jedwood	dogrocker	lamartire	Mactaivsh







					
maferland	Martin005	martiuslim	MatthiasKunnen	msand	richartkeil







					
richmeij	rosenfeld	jrschumacher	ThomasG77	sparanoid	zhuangya

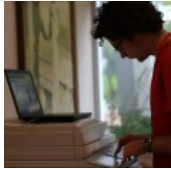





					
allenfantasy	Zyclotrop-j	fortrieb	jarey	muhammadInam	rettgerst





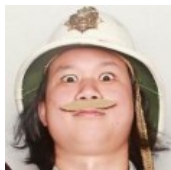

					
mkabatek	jukakoski	olemoign	ajschmidt8	superhawk610	abannach





					
adamelmore	ajh-sr	adamvigneault	adritasharma	asmt3	alexnj





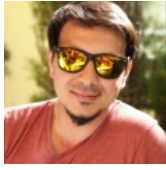

					
aalepis	Dogfalo	tekacs	amitport	functino	radarhere





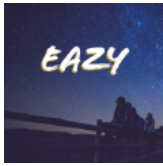

					
superandrew213	andychongyz	anthony0030	aduh95	Abourass	arthurdenne





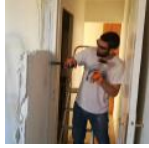

					
apuyou	bochkarev-artem	atsawin	ayhankesicioglu	azeemba	azizk




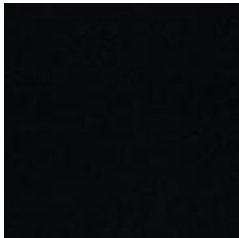


					
bducharme	Quorafind	wbaaron	bedgerotto	cyu	cartfisk







					
cellvinchung	chao	csprance	Aarbel	cbush06	czj

					
ardeois	sercraig	danmichaelo	mrboomer	akizor	davilima6







					
DennisKofflard	jeetiss	sweetro	efbautista	yoldar	eliOcs







					
EnricoSottile	Gkleinereva	fgallinari	ferdiusa	dtrucs	geoffappleford





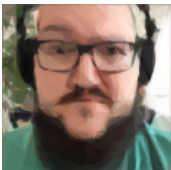

					
gjungb	roenschg	HughbertD	HussainAlkhalifah	huydod	ishendyweb



					
NaxYo	ghasfakhri	intenzive	JacobMGEvans	jdssem	Jbithell


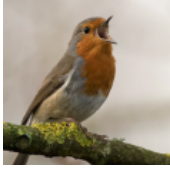



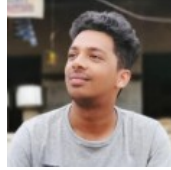
					
jcjmclean	janklimo	janwilts	vith	jessica-coursera	Jmales

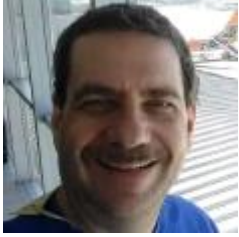

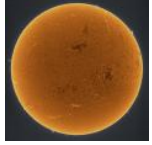



					
theJoeBiz	profsmallpine	jonathanarbely	jderrough	jonathanly	jorgeepc




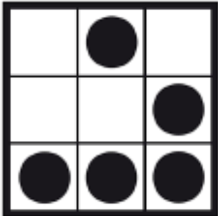


					
julianocomg	firesharkstudios	elkebab	kyleparisi	lafe	leaanthony






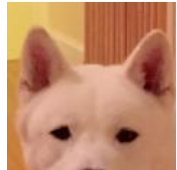
					
larowlan	dviry	galli-leo	leods92	louim	lucaperret



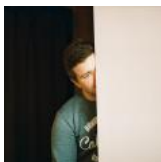



					
mperrando	marcosthejew	marcusforsberg	Acconut	mattfik	matthewhartstong







					
hrsh	mhulet	mkopinsky	achmiral	mnafees	shahimclt







					
pleasespammelater	naveed-ahmad	nicojones	coreprocess	leftdevel	cryptic022

					
patricklindsay	pedrofs	phillipalexander	ppadmavilasom	Pzoco	eman8519






					
luarmr	SxDx	phobos101	romain-preston	scherroman	rart






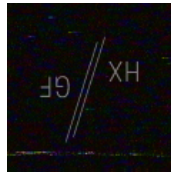
					
fortunto2	samuelcolburn	sergei-zelinsky	SpazzMarticus	waptik	steverob



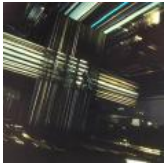
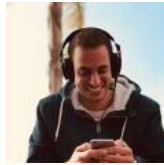
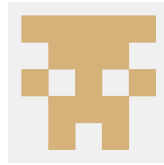

					
taj	Tashows	twarlop	tmaier	tomsaleeba	tvaliasek







					
vially	nagyv	willycamargo	xhocquet	yaegor	YehudaKremer




					
zachconner	zacharyl原因son	agreene-coursera	alfatv	anark	arggh

					
avalla	bdirito	c0b41	canvasbh	craigcbrunner	darthf1

					
dkisic	fingul	franckl	gaelicwinter	green-mike	hxgf

					
johnmanjiro13	kode-ninja	magumbo	ninesalt	phil714	lunta

					
rhymes	rebosse	rtaieb	slawexxx44	thanhthot	tinny77

		
vedran555	yoann-hellopret	olitomas

Software

 BrowserStack We use Browserstack for manual testing

License

The MIT License.