



Introduction to Clouddinary's CLI

Overview for Developers and Command Line Interface Users



Jen Brissman
Technical Curriculum Engineer

Topics

- What is a CLI?
- Setting Up Your Environment
- CLI Helper Tools
- Uploading Assets
- Cloudinary URLs
- Managing Assets
- Sync (Local File Management)
- Transforming Assets
- Next Steps for Further Support



What is a CLI?



Command Line Interface



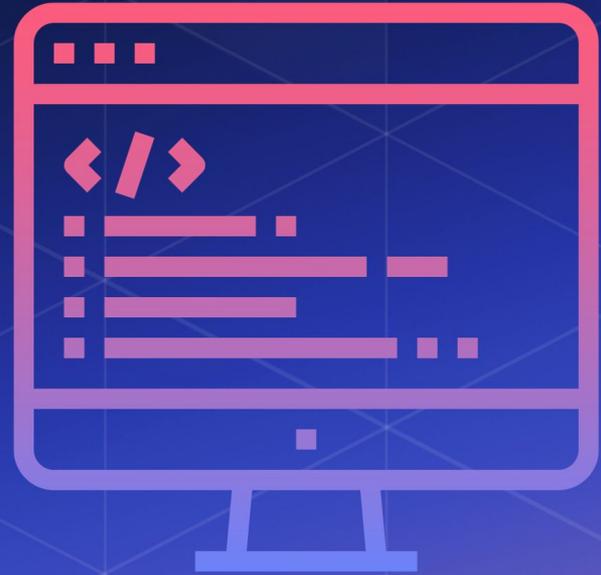
CLI stands for *Command Line Interface*

- Text-based interface used for interacting with software and computer systems by entering commands into a terminal or console.
- In a CLI, users interact with the system by typing commands and receiving text-based feedback.
- Alternative to Graphical User Interfaces - GUIs, where users interact with software through graphical elements like windows, icons, and buttons.

Cloudinary CLI



Setting Up Your Environment



Install Python

To run Cloudinary's CLI, you need to install Python, **version 3.6 or higher**.

You can download directly from python.org/downloads

Download Python 3.12.0

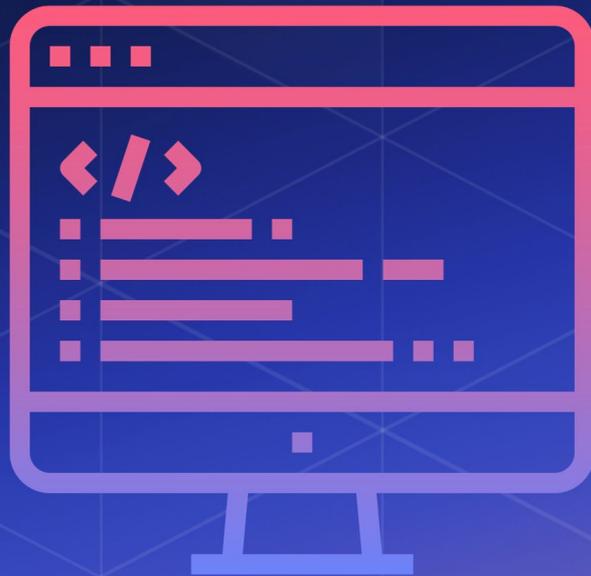
You can also download and install through Homebrew (Mac)

```
$ brew update && brew upgrade
```

```
$ brew install python3
```

```
$ python3 --version
```

```
> Python 3.9.6
```



Install Cloudinary

Next, you will need to install the Cloudinary CLI Package.

```
$ pip3 install cloudinary-cli
```

Make sure you have the most updated package.

```
$ pip3 install cloudinary-cli --upgrade
```



Install Cloudinary

Next, you will need to install the Cloudinary CLI Package.

```
$ pip3 install cloudinary-cli
```

Make sure you have the most updated package.

```
$ pip3 install cloudinary-cli --upgrade
```



Clouddinary Credentials



Clouinary_URL (API Environment Variable)

You can locate your CLOUDINARY_URL on your Clouinary Dashboard under Product Environment Credentials.

The screenshot shows the Clouinary Dashboard interface. At the top, a blue banner reads "Welcome to your Clouinary Dashboard" with a subtext: "Find all the information you need about your plan, usage, and how to get the most out of, or even impact, Clouinary features." To the right of the banner is a "Transformations" chart showing Image (874,658), Video (142,305), and Total (1,016,963) usage over time.

Below the banner, the "Product Environment Credentials" section is highlighted with a blue arrow. It contains three credential cards:

- Cloud Name:** jen-brissman
- API Key:** 963449183515596
- API Secret:** *****

At the bottom, the "API Environment variable" section is highlighted with a blue border. It displays the variable name and its value:

```
<> API Environment variable  
CLOUDINARY_URL=cloudinary://*****.*****@jen-brissman
```

On the right side of this section, there are icons for a clipboard, an eye (visibility toggle), and a "More Info" button.

Cloudinary_URL (API Environment Variable)

The CLOUDINARY_URL contains your cloud name, API Key, and API Secret.

The screenshot displays a 'Product Environment Credentials' interface. It features three credential cards: 'Cloud Name' (jen-brissman), 'API Key' (963449183515596), and 'API Secret' (masked with asterisks). Below these is an 'API Environment variable' card showing the resulting URL: `CLOUDINARY_URL=cloudinary://*****:*****@jen-brissman`. Blue lines connect the individual credential values to their respective parts in the final URL string.

Important!

Do not expose your API Secret in your client-side code, as it creates a security risk and potentially allows others to access your account.

API Environment Variable Configuration

Then **Export**(Mac) or **Set** (Windows) your CLOUDINARY_URL via the CLI.



```
export CLOUDINARY_URL=cloudinary://API_KEY:API_SECRET@cloud_name
```



```
set CLOUDINARY_URL=cloudinary://API_KEY:API_SECRET@cloud_name
```



API Environment variable

```
CLOUDINARY_URL=cloudinary://*****.*****@jen-brissman
```



More Info

Check Cloudinary Configuration

Check your Cloudinary configuration in the CLI

```
$ cld config  
cloud_name:  cloud_name  
api_key:     API_KEY  
api_secret:  *****CRET  
private_cdn: False
```

from_url Configuration

You can also configure from the CLI using `clid config --from_url`, which will save your configuration for use beyond a single session.

However, future commands would need to be prefixed with `clid -C cloudname`, instead of `clid`.

```
$ clid config --from_url cloudinary://API_KEY:API_SECRET@cloud_name  
Config cloud_name saved!
```

```
$ clid -C cloud_name config  
cloud_name: cloud_name  
api_key: API_KEY  
api_secret: *****CRET
```

You can use `clid config --from_url` to set up **multiple** account configurations

Digital Asset Management (DAM)

The image shows a screenshot of a Digital Asset Management (DAM) interface with several callout boxes pointing to specific features:

- Show/Hide Folder Pane**: Points to the left sidebar navigation.
- Media Library Tabs**: Points to the top navigation tabs (Assets, Folders, Collections, Recently Uploaded).
- Toggle Grid/List View**: Points to the view toggle icons.
- Global Search**: Points to the search bar at the top right.
- Upload Assets**: Points to the 'Upload' button.
- Asset Toolbar**: Points to the toolbar above the asset preview.
- Rename Asset**: Points to the '...' menu option in the toolbar.
- Show/Hide Preview Pane**: Points to the right-hand preview pane.
- Preview Pane - Metadata**: Points to the 'Metadata' tab in the preview pane.
- Preview Pane - Summary**: Points to the 'Summary' tab in the preview pane.
- Jump to Root Folder**: Points to the root folder icon in the sidebar.
- Sorting Options**: Points to the 'Last replaced' dropdown menu.

The interface itself displays a 'Media Library' view with a sidebar showing folders like 'Product A', 'Product B', and 'samples'. The main area shows a grid of assets, with 'elephants_jzgp57' selected. The right-hand preview pane shows a video of elephants with metadata such as 'Location: samples', 'Format: MP4', 'File size: 37.06 MB', and 'Duration: 00:48'.

Helper Tools



Helper Tools

If you are looking for help while using the CLI, you can use the following command:

```
$ cld --help # see list of commands (below)
```

Usage: cld [OPTIONS] COMMAND [ARGS]...

Options:

- `--help` Show this message and exit.
- `--version` Show the version and exit.
- `-c, --config TEXT` Tell the CLI which account to run the command on by specifying an account environment variable.
- `-C, --config_saved TEXT` Tell the CLI which account to run the command on by specifying a saved configuration
- `-v, --verbosity LVL` Either CRITICAL, ERROR, WARNING, INFO or DEBUG

Commands:

- `admin` Run any methods that can be called through the admin API.
- `config` Display the current configuration, and manage additional...
- `make` Return template code for implementing the specified Cloudinary widget.
- `migrate` Migrate files using an existing auto-upload mapping and a file of URLs.
- `provisioning` Run any methods that can be called through the provisioning API.
- `regen_derived` Regenerate all derived assets pertaining to a named transformation, or transformation string.
- `search` Run the admin API search method.
- `sync` Synchronize between a local directory and a Cloudinary folder.
- `upload_dir` Upload a folder of assets, maintaining the folder structure.
- `uploader` Run any methods that can be called through the upload API.
- `url` Generate a Cloudinary URL, which you can optionally open...
- `utils` Call Cloudinary utility methods.

Basic Commands

To help you get started, here are some basic commands you can use.

```
$ cld --version # see versions of Clouinary CLI, underlying Clouinary Python SDK and Python
```

```
$ cld admin # see available Admin API methods
```

```
$ cld uploader # see available Upload API methods
```

```
$ cld search --help # see usage for the Search API
```

```
$ cld utils # see available utility methods
```

```
$ cld admin --doc # opens the Admin API reference in a browser
```

```
$ cld upload --doc # opens the Upload API reference in a browser
```

Uploading



Uploading

- Upload Method
- Fetch
- Public ID
- Resource Type
- Folders
- JSON Response
- Upload Widget



Upload Method

```
$ cld upload
```

The *upload method* performs an authenticated upload API call over HTTPS.

Try performing a basic upload of your own from a local folder.

- Navigate within the CLI to the local folder that contains the asset you would like to upload.
- Enter the command `cld` and the **name of the asset** you would like to upload.

```
$ cld upload hiking.jpg
```

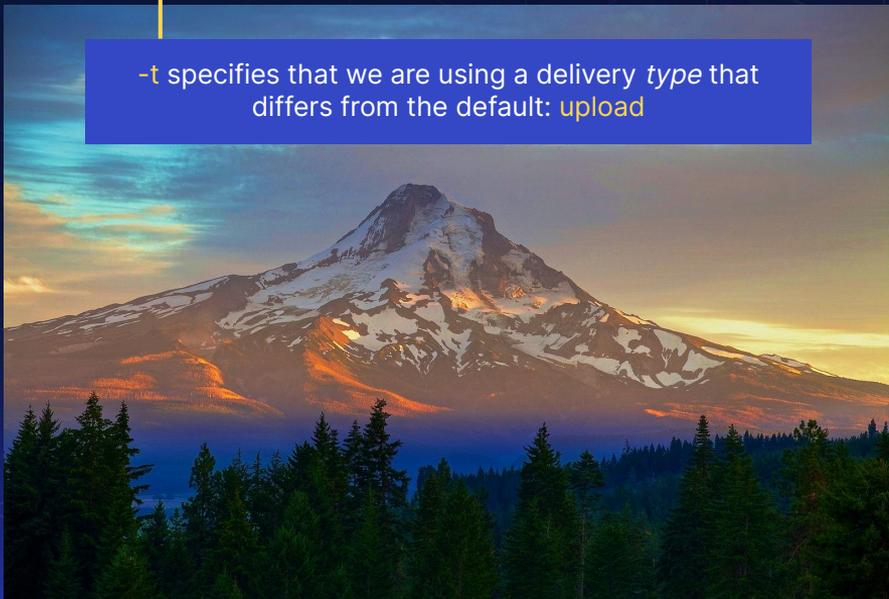
*I am using a file called `hiking.jpg`

Fetch

You can fetch images from remote sources and upload them straight to Clouinary entering the command `url -t fetch` and the **URL** of the image.

```
cld url -t fetch 'https://upload.wikimedia.org/wikipedia/commons/d/d5/Mt._Hood_%288081466807%29.jpg'
```

↑
`-t` specifies that we are using a delivery *type* that differs from the default: `upload`



- Supported for images only.
- Cached on your Clouinary account for performance reasons.
- Remote images are checked on a regular basis, and if the remote image changes, the cached image is updated accordingly.

Public ID

A unique identifier for your asset that appears in the URL, which is used to reference the uploaded resource as well as for building dynamic delivery and transformation URLs.

If you don't supply a Public ID in the upload API call, one will be randomly assigned.

If you **DO** supply a Public ID, **DO NOT** include the file extension.

```
$ cld upload mountains.jpg
```

Public ID: "vr1kukfernlipukpw3hr"

```
$ cld upload mountains.jpg use_filename=true unique_filename=false
```

Public ID: "mountains"

```
$ cld upload mountains.jpg use_filename=true unique_filename=true
```

Public ID: "mountains_yqnfod"

```
$ cld upload mountains.jpg public_id=i-love-hiking
```

Public ID: "i-love-hiking"

Public ID in the URL

```
https://res.cloudinary.com/demo/image/upload/mountains.jpg
```

Resource Type

Used to indicate whether you want to upload an image, video or raw file.

You can let Cloudinary determine the type by setting the parameter to “auto”.

```
$ cld upload water-bottle.mp4 resource_type=auto
```

Or you can manually define for each file.

```
$ cld upload water-bottle.mp4 resource_type=video
```

Resource Type in the URL

```
https://res.cloudinary.com/demo/video/upload/water-bottle.mp4
```

Folders

Simply put, folders are a way to organize and divide your files within Cloudinary.

- Folder paths can be included in the **Public ID** parameter

```
$ cld upload dolomites.jpg public_id=italy/dolomites
```

Public ID: "italy/dolomites"

*TIP: A folder will **automatically** be created for you with this call, if it did not already exist.

- Folder paths can also be created in a separate **Folder** parameter

```
$ cld upload dolomites.jpg folder=italy public_id=dolomites
```

Public ID: "italy/dolomites"

- This works in the same way for sub folders

```
$ cld upload dolomites.jpg use_filename=true unique_filename=false folder=mountains/italy
```

Public ID: "mountains/italy/dolomites"

Folders in the URL

```
https://res.cloudinary.com/jen-brissman/image/upload/mountains/italy/dolomites.jpg
```

JSON Response

Cloudinary returns a JSON response that includes HTTP and HTTPS URLs for accessing the uploaded image, as well as additional image information.

```
{ public_id: mountains-123,  
  version: '1312461204',  
  width: 864,  
  height: 564,  
  format: 'jpg',  
  created_at: '2017-08-10T09:55:32Z',  
  resource_type: 'image',  
  tags: [],  
  bytes: 9597,  
  type: 'upload',  
  etag: 'd1ac0ee70a9a36b14887aca7f7211737',  
  url: 'http://res.cloudinary.com/demo/image/upload/v1312461204/sample.jpg',  
  secure_url: 'https://res.cloudinary.com/demo/image/upload/v1312461204/sample.jpg',  
  signature: 'abcdefgc024acceb1c1baa8dca46717137fa5ae0c3',  
  original_filename: 'sample'}
```

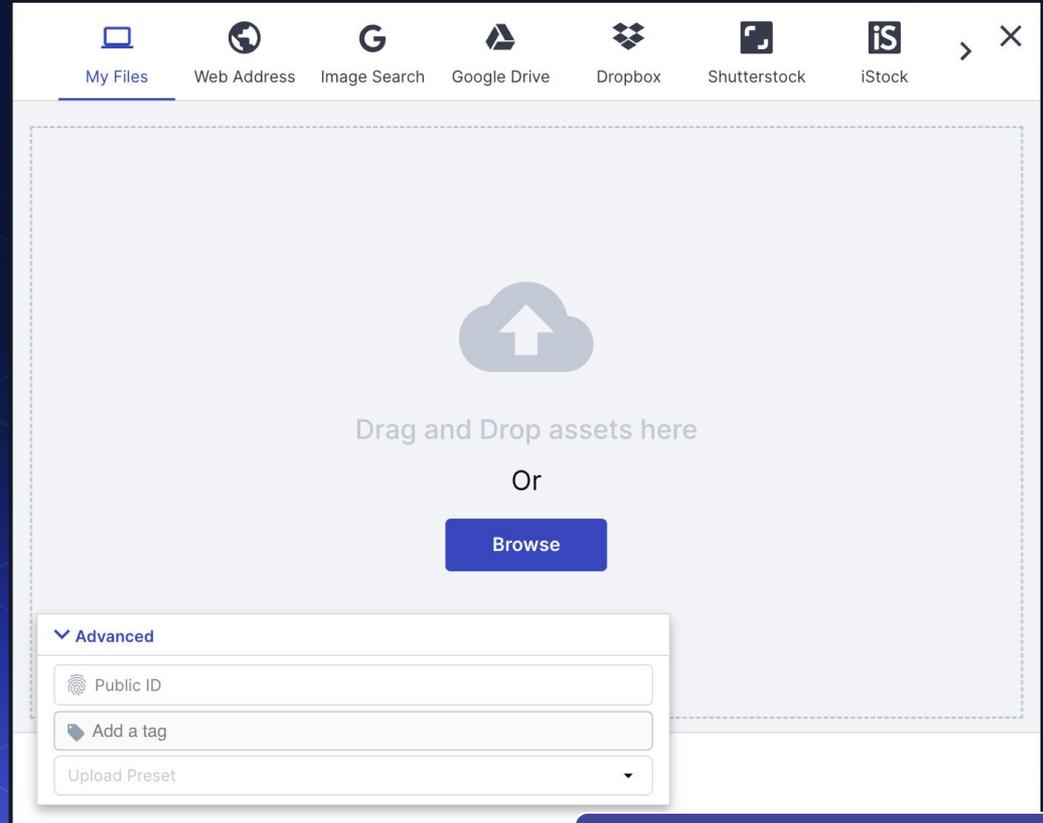
Upload Widget

You can implement Cloudinary's Upload Widget to easily upload assets from the frontend of a web/mobile browser directly to your Cloudinary account, without involving any servers in the process.

The widget shows upload progress and offers thumbnail-preview, as well as interactive-cropping capabilities.

Create an Upload Widget HTML using the CLI

```
$ cld make upload widget
```



Upload Widget Tutorial

Cloudinary URLs



Cloudinary Delivery URL Structure

Domain	<code>https://res.cloudinary.com/</code>	
Cloud Name	<code><cloud_name>/</code>	
Resource Type	<code><resource_type>/</code>	<code>https://res.cloudinary.com/jen-brissman/image/upload/v1702185884/mountains.jpg</code>
Delivery Type	<code><type>/</code>	<code><domain>/<cloud_name>/<resource_type>/<type>/<version>/<public_id>.<format></code>
Transformations	<code><transformations>/</code>	
Version	<code><version>/</code>	
Public ID and Format	<code><public_id>.<format></code>	

Creating Cloudinary URLs

There are two main ways to generate a Cloudinary delivery URL in the CLI. Let's create a URL with the public ID of "mountains" and scale it to a width of 300 pixels.

The Helper command, `url`, generates a Cloudinary URL.

Note: Unless the URL is opened, the derived asset is not generated.

```
$ cld url mountains w_300,c_scale
```

The `utils` command enables you to call Cloudinary utility methods. One of these is the `cloudinary_url` method, for embedding images in web pages using SDKs.

```
$ cld utils cloudinary_url mountains width=300 crop=scale
```

Same exact URL output for both:

```
https://res.cloudinary.com/demo/image/upload/c_scale,w_300,mountains
```

Managing Assets





Upload API



Admin API

Managing Assets

- List Resources
- Search
- Rename
- Tags
- Delete
- Invalidate and Versioning



List Resources

Using Admin API methods with the Cloudinary CLI, you can list all of your uploaded assets by many different sets of criteria.

- List all images

```
$ cld admin resources
```

- List asset with a given public ID

```
$ cld admin resource mountain-123
```

- List all images with a given tag

```
$ cld admin resources tags="blurry"
```

- List all images with a given prefix

```
$ cld admin resources type=upload prefix=sample
```

Search

The search command runs the Admin API search method. This method allows you to filter and retrieve information on all the assets in your product environment with the help of query expressions in a Lucene-like query language.

- Search for images with the tag “blurry”

```
$ cld admin search tags=blurry
```

- Lucene-like query

```
$ cld admin search -f tags -n 30 -s public_id asc "resource_type:image AND  
tags=bird AND uploaded_at>1d AND bytes>100k"
```

Rename

Renaming immediately and permanently updates the assets in your cloud storage.

```
$ cld upload rename [old public ID] [new public ID]
```



```
$ cld upload rename mountains tetons
```

Any existing URLs of renamed assets and their associated derived assets are no longer valid.



Tags

With tags you can categorize and organize your assets. You can add tags to your assets at any time, whether during upload or after the asset is in your account.

```
$ cld upload tent.jpg public_id=tent tags="tent, stars, grass"
```

```
$ cld add_tag italy public_ids=tent
```

You can also remove tags to uploaded assets

```
$ cld remove_tag grass public_ids=tent
```

```
$ cld remove_all_tags tent
```

You can also use our AI auto tagging add-ons to further automate the tagging process

```
$ cld uploader upload "tent.jpg" categorization=aws_rek_tagging,google_tagging auto_tagging=".95"
```



Delete

The **destroy** method of the Upload API immediately destroys and permanently updates the assets in your cloud storage.

```
$ cld uploader destroy tent
```

You can use the **delete_resources** method of the Admin API delete multiple assets at one time by their Public ID or even a prefix in the Public ID with our Admin API.

```
$ cld admin delete_resources tetons,i-love-hiking
```

```
$ cld admin delete_resources_by_prefix mountains
```

Invalidate and Versioning

Prevent users from accessing deleted or renamed assets by sending an invalidation request.

- This instructs the CDN to remove **cached** copies of the old asset.
- The old cached media asset **can remain** on the CDN servers for up to 30 days.
- The next request for the asset will pull the newest copy from your Cloudinary storage, or will return an error if the asset no longer exists.
- This same approach can be done with versioning too, bypassing the CDN cached version and forcing delivery of the newest asset.

```
$ cld destroy sample invalidate=true
```

Note: From within the UI/DAM, an invalidate request is **automatically** included whenever you delete, rename, or overwrite media assets.

To check versioning:

```
$ cld admin resource sample versions=true
```

Sync (Local File Management)



Sync (Local File Management)

Syncing Local Files and Folders to Cloudinary. Upload from a local directory to the cloud or download a cloud directory to your local file system. This is unique to the CLI, it doesn't exist in SDKs.

Similar to **git** syntax, sync allows you to **push** to the cloud and **pull** from the cloud.

Keep in mind that in this case sync does not mean constantly syncing or “synchronizing”.

For perpetual file syncing check out:
Cloudy Desktop from Cloudinary Labs

Sync (Local File Management)

Sync example - If you have a local directory under your root, “~/Pictures/images” that you want to upload to a folder in your Cloudinary account named “test-images”, you can use the **push** option.

```
$ cld sync --push ~/Desktop/cli-course test-assets
```

If you want to download the “test-assets” folder from Cloudinary to another local directory, for example “~/Desktop/pulled-from-cld”, you can use the **pull** option.

```
$ cld sync --pull ~/Desktop/pulled-from-cld test-assets
```



Cloudinary Transformations



Cloudinary Transformations

Cloudinary Transformations																			
Fn ¹ Custom Function function		<table border="1"> <tr> <td>aesthetic</td> <td>attention</td> <td>audio</td> <td>audio-video</td> </tr> <tr> <td>custom</td> <td>effect</td> <td>layer</td> <td>optimization</td> </tr> <tr> <td>photoshop</td> <td>resize</td> <td>sprite</td> <td>template</td> </tr> </table>				aesthetic	attention	audio	audio-video	custom	effect	layer	optimization	photoshop	resize	sprite	template	P ² Public ID	
aesthetic	attention	audio	audio-video																
custom	effect	layer	optimization																
photoshop	resize	sprite	template																
Ac ³ Audio Codec m...	Af ⁴ Audio Frequency d...																		
B ¹¹ Background b...	Bo ¹² Border bc...	Co ¹³ Color cc...	Cs ¹⁴ Color Space cs...	R ¹⁵ Radius r...	F ⁵ Format f...	Q ⁶ Quality q...	Dr ⁷ DPR dpr...	If ⁸ If if...	T ⁹ Named Transformations t...	\$ ¹⁰ Variable \$.									
L ²² Layer l...	Un ²³ Under Layer u...	Pg ²⁴ X position x...	Y ²⁵ Y position y...	Pg ²⁶ Font Family font...	Fs ²⁷ Font Size -font	Fw ²⁸ Font Weight -bold	Tx ²⁹ Text txt...	Eo ³⁰ End Offset eo...	Du ³¹ Duration du...	So ³² Start Offset so...	Br ³³ Bit Rate br...	DI ³⁴ Delay di...	Fp ³⁵ Frames per Second fp...	Pg ³⁶ Keyframe Integer k...	Sp ³⁷ Stream Profile sp...	Vc ³⁸ Video Codec vc...	Vs ³⁹ Video Sample vs...		
Aa ⁴⁰ Accelerate e-accelerate	Ay ⁴¹ Advanced Red Eye e-adv-red-eye	Av ⁴² Anti Removal e-anti-removal	At ⁴³ Art e-art	Au ⁴⁴ Auto Brightness e-auto-brightness	Al ⁴⁵ Auto Color e-auto-color	As ⁴⁶ Auto Contrast e-auto-contrast	Ab ⁴⁷ Color Blind Assist e-assist-colorblind	Bg ⁴⁸ Background Removal e-background	Bw ⁴⁹ Black White e-black-white	Bl ⁵⁰ Blur e-blur	Bu ⁵¹ Blur e-blur	Bf ⁵² Blur Faces e-blur-faces	Bn ⁵³ Blur Region e-blur-region	Bn ⁵⁴ Boomerang e-boomerang	Bg ⁵⁵ Brightness e-brightness	Bh ⁵⁶ Brightness HSB e-brightness-hsb	Ca ⁵⁷ Cartoonify e-cartoonify		
Cz ⁵⁸ Colorize e-colorize	Cn ⁵⁹ Contrast e-contrast	Cu ⁶⁰ Cut Out e-cut-out	De ⁶¹ Denoise e-denoise	Di ⁶² Distort e-distort	Dp ⁶³ Displace e-displace	Fa ⁶⁴ Fade e-fade	Fg ⁶⁵ Fill Light e-fill-light	Ga ⁶⁶ Gamma e-gamma	Gr ⁶⁷ Gradient Fade e-gradient-fade	Gy ⁶⁸ Grayscale e-grayscale	Ge ⁶⁹ Green e-green	Hu ⁷⁰ Hue e-hue	Im ⁷¹ Improve e-improve	Li ⁷² Lightroom e-lightroom	Lo ⁷³ Loop e-loop	Mt ⁷⁴ Make Transparent e-make-transparent	Mu ⁷⁵ Multiply e-multiply		
Ne ⁷⁶ Negate e-negate	No ⁷⁷ Noise e-noise	Oi ⁷⁸ Oil Paint e-oil-paint	Ot ⁷⁹ Opacity Threshold e-opacity-threshold	Or ⁸⁰ Ordered Dither e-ordered-dither	Ou ⁸¹ Outline e-outline	Ov ⁸² Overlay e-overlay	Pi ⁸³ Pixelate e-pixelate	Pf ⁸⁴ Pixelate Faces e-pixelate-faces	Pr ⁸⁵ Pixelate Region e-pixelate-region	Mu ⁸⁶ Preview e-preview	Po ⁸⁷ Progress Bar e-progress-bar	Re ⁸⁸ Recolor e-recolor	Rd ⁸⁹ Red e-red	Ry ⁹⁰ Redeye e-redeye	Rc ⁹¹ Replace Color e-replace-color	Rv ⁹² Reverse e-reverse	Sa ⁹³ Saturation e-saturation		
Se ⁹⁴ Sepia e-sepia	Sc ⁹⁵ Screen e-screen	Sh ⁹⁶ Shadow e-shadow	Sp ⁹⁷ Sharpen e-sharpen	Sr ⁹⁸ Sharpen e-sharp	Si ⁹⁹ Simulate Color Blind e-simulate-color-blind	St ¹⁰⁰ Style Transfer e-style-transfer	Th ¹⁰¹ Theme e-theme	Ti ¹⁰² Text e-text	Tr ¹⁰³ Transition e-transition	Tm ¹⁰⁴ Trim e-trim	Um ¹⁰⁵ Unsharp Mask e-unsharp-mask	Ve ¹⁰⁶ Vibrance e-vibrance	Vi ¹⁰⁷ Vibrance e-vibrance	Vt ¹⁰⁸ Vignette Correct e-vignette-correct	Vg ¹⁰⁹ Vignette e-vignette	Vo ¹¹⁰ Volume e-volume			
Ar ¹¹¹ Aspect Ratio ar...	H ¹¹² Height h...	W ¹¹³ Width w...	Fi ¹¹⁴ Crop Fill c-fill	Fd ¹¹⁵ Crop Fill Pad c-fill-pad	Ft ¹¹⁶ Crop Fit c-fit	Ic ¹¹⁷ Image Crop c-image-crop	Is ¹¹⁸ Image Scale c-image-scale	Lf ¹¹⁹ Crop Limit Fill c-limit	Li ¹²⁰ Crop Limit c-limit	Lp ¹²¹ Crop Limit Pad c-limit-pad	Mf ¹²² Crop Minimum Fit c-min	Mp ¹²³ Crop Minimum Pad c-min-pad	Pa ¹²⁴ Crop Pad c-pad	Sc ¹²⁵ Crop Scale c-scale	Tb ¹²⁶ Crop Thumb c-thumb				

Example Assets

Let's use this photo to perform various Cloudinary transformations via the CLI.

First, we'll upload two assets to our account with designated Public IDs of "hiker" and "mountains"

Then we can easily reference the asset by this Public ID throughout this transformations section.

```
$ cld upload hiker.jpg public_id=hiker
```

```
$ cld upload mountains.jpg public_id=mountains
```



Resize

You can resize any image or video by editing its width (`w_`) and/or height (`h_`)

Decreasing the width and/or height of the asset will commonly decrease the file size, optimizing it for a specific project.

```
$ cld url hiker w_300
```

The default is scale. If you don't provide one of the parameters it will automatically scale for you.

You can also make transformations in the URL.

```
http://res.cloudinary.com/jen-brissman/image/upload/w_300/hiker
```



1.05 MB



15.88 KB

Format (f_auto)

Use Cloudinary to automatically convert assets to other formats for displaying in your web site or application (output formats).

Specify image and video format, e.g. on native mobile, based on device capabilities.

Allow Cloudinary to deliver the optimal format for web delivery scenarios with **f_auto** for images and video.

```
$ cld url hiker f_auto
```

```
http://res.cloudinary.com/jen-brissman/image/upload/f\_auto/hiker
```



JPG - 1.05 MB



AVIF - 492.4 KB

Crop with Gravity

You can use Gravity to specify a location in an image or video that is used as the focus for another transformation.

Let's make a 1:1 aspect ratio by adjusting the height and width to be 200 each. Then perform a thumb crop.

We can utilize `g_auto` to make sure the face is centered in the frame.

```
$ cld url hiker w_500,h_500,c_thumb,g_auto
```

```
https://res.cloudinary.com/jen-brissman/image/upload/w_500,h_500,c_thumb_g_auto,hiker
```



Rounding for Aesthetics

For rounding, you can use the radius parameter to control the degree of rounding applied to the image.

Here we will use `r_max` to create a complete circle.

We will also use the grayscale effect, `e_grayscale`.

Rounded images are commonly used for circular profile pictures, icons, or buttons in user interfaces.

If you have a design theme that involves rounded elements, rounding images can help maintain a consistent visual style throughout your application or website.



```
$ cld url hiker w_500,h_500,c_thumb,g_auto,r_max,e_grayscale,f_auto
```

```
https://res.cloudinary.com/jen-brissman/image/upload/w\_500,h\_500,c\_thumb,g\_auto,r\_max,e\_grayscale,f\_auto/hiker
```

Quality (q_auto)

Quality controls the visual quality and compression level of assets.

With `q_auto`, you allow Cloudinary to deliver the optimal quality of images and videos for each viewing device.

It compresses your asset without a visual difference, but a big difference in the file size as you can see.

```
$ cld url mountains q_auto
```

```
http://res.cloudinary.com/jen-brissman/image/upload/q\_auto/mountains
```

No visual
difference



8.53 MB - JPG



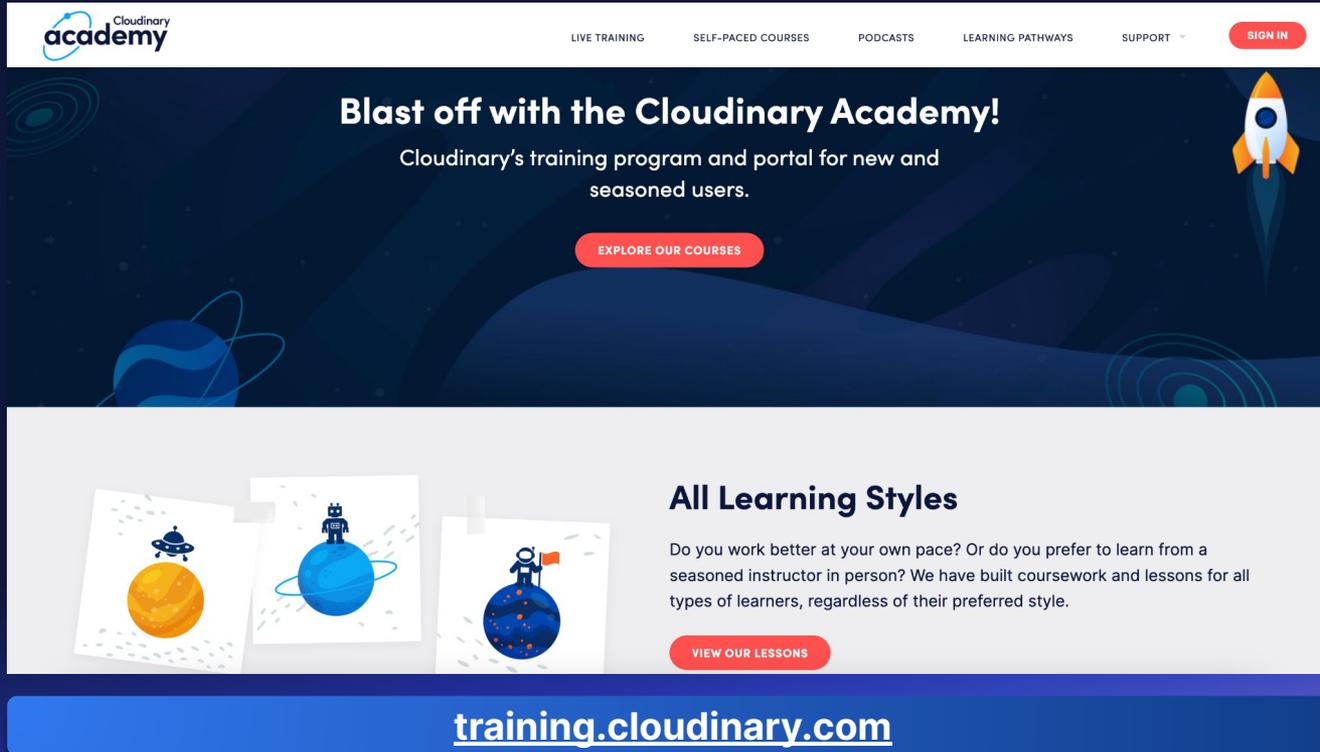
3.88 MB - JPG

Next Steps and Further Support



Cloudinary Academy

The Cloudinary Academy offers both self-paced courses and live classes taught by Cloudinary experts.



The screenshot shows the Cloudinary Academy website. At the top left is the Cloudinary Academy logo. The navigation menu includes 'LIVE TRAINING', 'SELF-PACED COURSES', 'PODCASTS', 'LEARNING PATHWAYS', 'SUPPORT', and a 'SIGN IN' button. The main heading is 'Blast off with the Cloudinary Academy!' with a subtext 'Cloudinary's training program and portal for new and seasoned users.' and an 'EXPLORE OUR COURSES' button. Below this is a section titled 'All Learning Styles' with three icons representing different learning methods: self-paced (planet and UFO), live class (robot on planet), and in-person (astronaut on planet). A 'VIEW OUR LESSONS' button is also present. At the bottom is a blue banner with the URL 'training.cloudinary.com'.

Cloudinary
academy

LIVE TRAINING SELF-PACED COURSES PODCASTS LEARNING PATHWAYS SUPPORT [SIGN IN](#)

Blast off with the Cloudinary Academy!

Cloudinary's training program and portal for new and seasoned users.

[EXPLORE OUR COURSES](#)

All Learning Styles

Do you work better at your own pace? Or do you prefer to learn from a seasoned instructor in person? We have built coursework and lessons for all types of learners, regardless of their preferred style.

[VIEW OUR LESSONS](#)

training.cloudinary.com

Recommended Courses

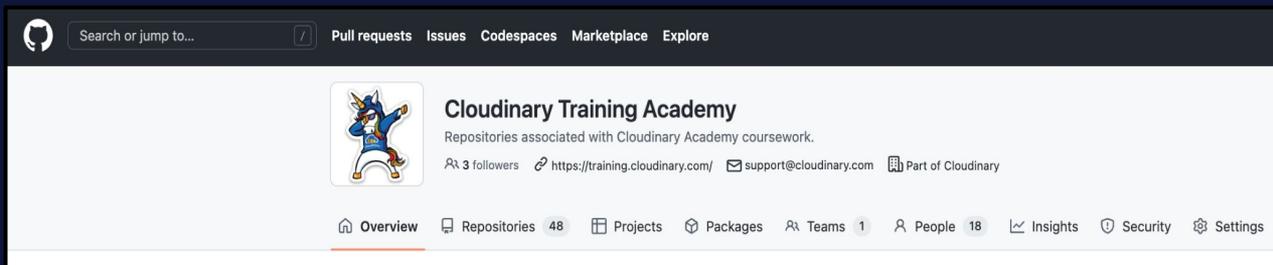
Those who are new to Cloudinary's APIs can benefit from a variety of helpful, self-paced courses that provide comprehensive learning resources.

- [Introduction to Cloudinary Programmable Media \(90-Minutes\)](#)
- [Understanding Cloudinary Programmable Media Terminology \(30-Minutes\)](#)
- [Cloudinary JumpStart for New Developer Users \(~40-Minutes\)](#)
- [Advanced Concepts for Developers \(~16 hours\)](#)
- [Introduction for Node.js Developers \(90 minutes\)](#)
- [Fundamentals for Developers \(~9 hours\)](#)



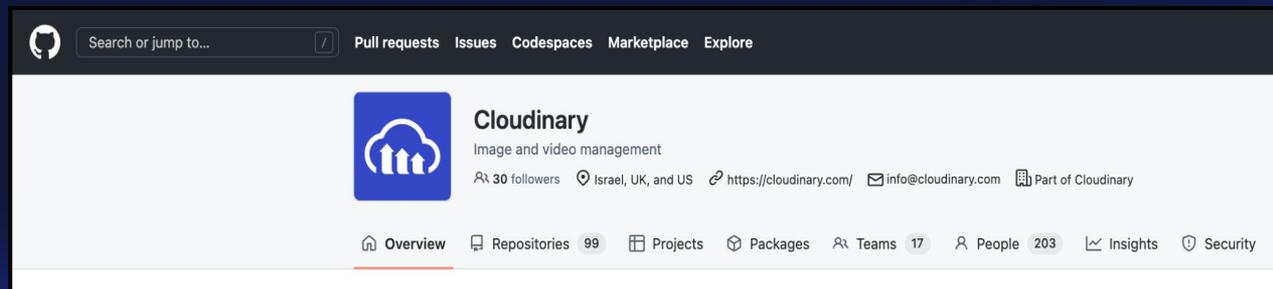
Review our GitHub Repositories

Access our sample projects to help you with building your own work using Cloudinary.



The screenshot shows the GitHub profile page for 'Cloudinary Training Academy'. At the top, there is a search bar and navigation links for 'Pull requests', 'Issues', 'Codespaces', 'Marketplace', and 'Explore'. The profile header includes the organization's name, a description 'Repositories associated with Cloudinary Academy coursework', and statistics: 3 followers, a website link 'https://training.cloudinary.com/', an email 'support@cloudinary.com', and a 'Part of Cloudinary' badge. Below the header, there are navigation tabs for 'Overview', 'Repositories (48)', 'Projects', 'Packages', 'Teams (1)', 'People (18)', 'Insights', 'Security', and 'Settings'. The 'Overview' tab is currently selected.

github.com/cloudinary-training

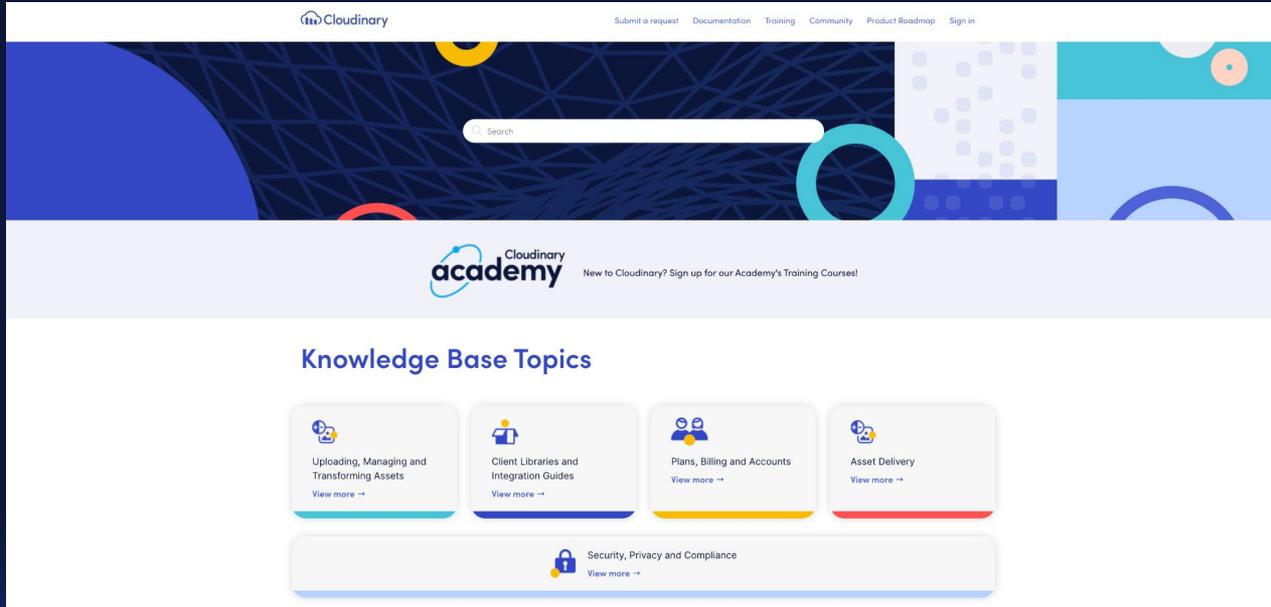


The screenshot shows the GitHub profile page for 'Cloudinary'. At the top, there is a search bar and navigation links for 'Pull requests', 'Issues', 'Codespaces', 'Marketplace', and 'Explore'. The profile header includes the organization's name, a description 'Image and video management', and statistics: 30 followers, location 'Israel, UK, and US', a website link 'https://cloudinary.com/', an email 'info@cloudinary.com', and a 'Part of Cloudinary' badge. Below the header, there are navigation tabs for 'Overview', 'Repositories (99)', 'Projects', 'Packages', 'Teams (17)', 'People (203)', 'Insights', and 'Security'. The 'Overview' tab is currently selected.

github.com/cloudinary

Engineering Support

We are always happy to answer your questions, as we have dedicated support staff for our developer community.



The screenshot shows the Cloudinary Academy website. At the top, there is a navigation bar with the Cloudinary logo and links for "Submit a request", "Documentation", "Training", "Community", "Product Roadmap", and "Sign in". Below the navigation bar is a search bar with the text "Search". The main content area features the Cloudinary Academy logo and the text "New to Cloudinary? Sign up for our Academy's Training Courses!". Below this is a section titled "Knowledge Base Topics" with five topic cards: "Uploading, Managing and Transforming Assets", "Client Libraries and Integration Guides", "Plans, Billing and Accounts", "Asset Delivery", and "Security, Privacy and Compliance". Each card has a "View more" link.

Cloudinary
Submit a request Documentation Training Community Product Roadmap Sign in

Search

Cloudinary
academy
New to Cloudinary? Sign up for our Academy's Training Courses!

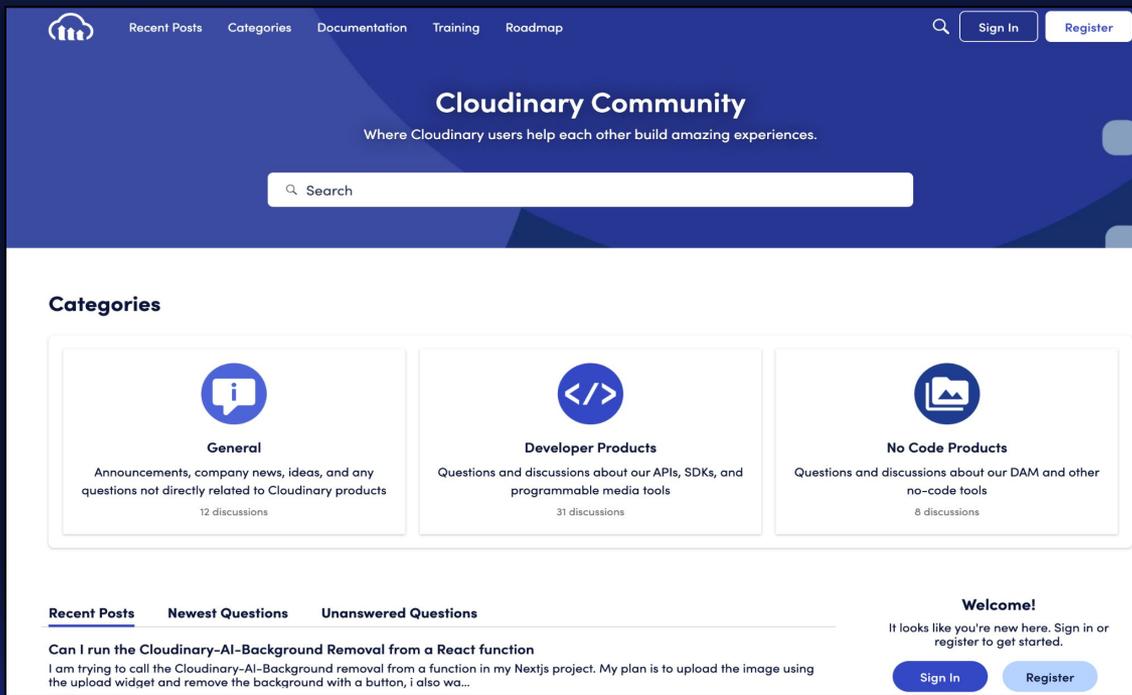
Knowledge Base Topics

- Uploading, Managing and Transforming Assets
View more →
- Client Libraries and Integration Guides
View more →
- Plans, Billing and Accounts
View more →
- Asset Delivery
View more →
- Security, Privacy and Compliance
View more →

support.cloudinary.com/hc/en-us/requests/new

Join Our Community Forums

Ask questions to staff or other users in our dedicated communities.



The screenshot shows the Cloudinary Community forum homepage. At the top, there is a navigation bar with the Cloudinary logo, links for 'Recent Posts', 'Categories', 'Documentation', 'Training', and 'Roadmap', a search icon, and buttons for 'Sign In' and 'Register'. The main header features the 'Cloudinary Community' title and the tagline 'Where Cloudinary users help each other build amazing experiences.' Below this is a search bar. The 'Categories' section contains three cards: 'General' (12 discussions), 'Developer Products' (31 discussions), and 'No Code Products' (8 discussions). At the bottom, there are tabs for 'Recent Posts', 'Newest Questions', and 'Unanswered Questions'. A 'Welcome!' message is displayed, along with 'Sign In' and 'Register' buttons. A sample forum post is visible under the 'Recent Posts' tab.

Recent Posts Categories Documentation Training Roadmap

Search Sign In Register

Cloudinary Community

Where Cloudinary users help each other build amazing experiences.

Search

Categories


General
Announcements, company news, ideas, and any questions not directly related to Cloudinary products
12 discussions


Developer Products
Questions and discussions about our APIs, SDKs, and programmable media tools
31 discussions


No Code Products
Questions and discussions about our DAM and other no-code tools
8 discussions

Recent Posts Newest Questions Unanswered Questions

Can I run the Cloudinary-AI-Background Removal from a React function
I am trying to call the Cloudinary-AI-Background removal from a function in my Nextjs project. My plan is to upload the image using the upload widget and remove the background with a button, I also wa...

Welcome!
It looks like you're new here. Sign in or register to get started.

Sign In Register

community.cloudinary.com

Join Our Discord Discussions

Connect with Cloudinary users and staff on Discord!

The screenshot shows the Discord interface for the 'Cloudinary Community' server, specifically the '# cloudinary-apis' channel. The interface is dark-themed. On the left, there's a sidebar with server navigation options like 'GET STARTED', 'NEWS AND ANNOUNCEMENTS', 'COURSES', 'MEDIA FLOWS', and 'PRODUCT DISCUSSIONS'. The main chat area shows a message from 'SamBrace' dated 04/08/2022, discussing a new DevJams podcast episode. Below it, a message from 'Thunderstorm' dated 07/30/2022 asks for help with a code snippet. The right sidebar shows a list of online and offline users, including 'Becky Peltz', 'Emily.lee', and 'colbyfayock'. At the bottom, there's a message input field with a 'Message # cloudinary-apis' label and various icons for attachments and emojis.

discord.gg/cloudinary

Thank you