Use image CDNs to optimize images

Why use an image CDN?

Image content delivery networks (CDNs) are excellent at optimizing images. Switching to an image CDN can yield a 40–80% savings in image file size. In theory, it's possible to achieve the same results using only build scripts, but it's rare in practice.

What's an image CDN?

Image CDNs specialize in the transformation, optimization, and delivery of images. You can also think of them as APIs for accessing and manipulating the images used on your site. For images loaded from an image CDN, an image URL indicates not only which image to load, but also parameters like size, format, and quality. This makes it easy to create variations of an image for different use cases.

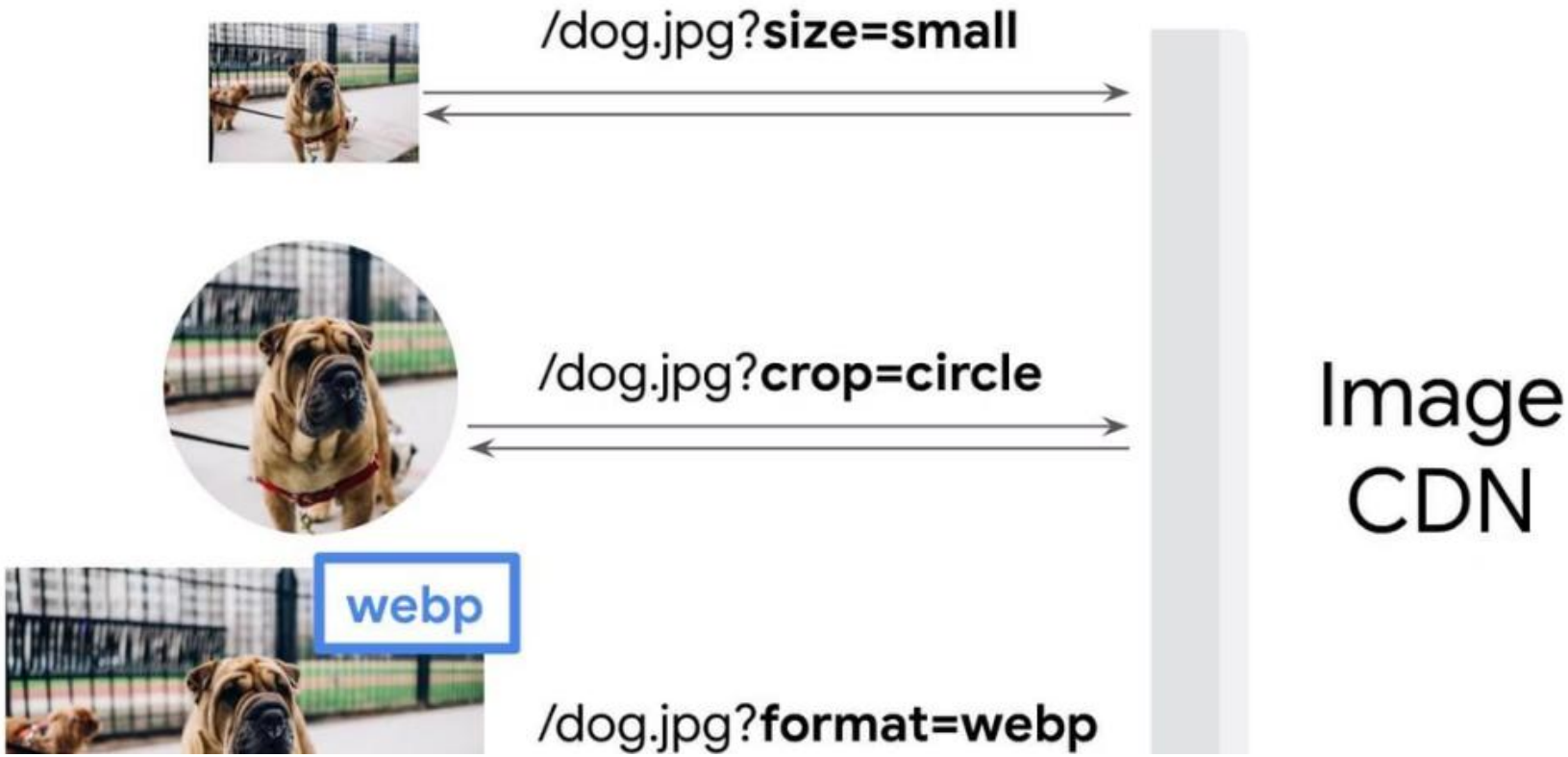


Image CDNs are different from build-time image optimization scripts in that they create new versions of images as they're needed. As a result, CDNs are generally better suited to creating images that are heavily customized for each individual client than build scripts are.

Security key

A security key prevents other people from creating new versions of your images. If this feature is enabled, each new version of an image requires a unique security key. If someone tries to change the parameters of the image URL but doesn't provide a valid security key, they won't be able to create a new version. Your image CDN will take care of the details of generating and tracking security keys for you.


https://my-site.example-cdn.com/daisy.jpg?key=s18dfgKd&quality=auto&size=300w400h&format=webp
Origin
Image
Security Key
Transformations

Transformations

Image CDNs offer tens, and in some cases hundreds, of different image transformations. These transformations are specified via the URL string, and there are no restrictions on using multiple transformations at the same time. In the context of web performance, the most important image transformations are size, pixel density, format, and compression. These transformations are the reason why switching to an image CDN typically results in a significant reduction in image size.

There tends to be an objectively best setting for performance transformations, so some image CDNs support an "auto" mode for these transformations. For example, instead of specifying that images be transformed to the WebP format, you could allow the CDN to automatically select and serve the optimal format. Signals that an image CDN can use to determine the best way to transform an image include:

For example, the image CDN might serve JPEG XR to an Edge browser, WebP to a Chrome browser, and JPEG to a very old browser. Auto settings are popular because they allow you to take advantage of image CDNs' significant expertise in optimizing images without the need for code changes to adopt new technologies once they're supported by the image CDN.

Types of Image CDNs

Image CDNs can be broken down into two categories: self-managed and third-party-managed.

Self-managed image CDNs

Self-managed CDNs can be a good choice for sites with engineering staff who are comfortable maintaining their own infrastructure.

Thumbor is the only self-managed image CDN available today. While it is open-source and free to use, it generally has fewer features than most commercial CDNs, and its documentation is somewhat limited. Wikipedia, Square, and 99designs are three sites that use Thumbor. See the How to install the Thumbor image CDN post for instructions on setting it up.

Third-party image CDNs

Third-party image CDNs provide image CDNs as a service. Just as cloud providers provide servers and other infrastructure for a fee; image CDNs provide image optimization and delivery for a fee. Because third-party image CDNs maintain the underlying technology, getting started is fairly simple and can usually be accomplished in 10-15 minutes, although a complete migration for a large site might take far longer. Third-party image CDNs are typically priced based on usage tiers, with most image CDNs providing either a free tier or a free trial to give you an opportunity to try out their product.

Choosing an image CDN

There are many good options for image CDNs. Some will have more features than others, but all of them will probably help you save bytes on your images and therefore load your pages faster. Besides feature sets, other factors to consider when choosing an image CDN are cost, support, documentation, and ease of setup or migration.

Trying them out yourself before making a decision can also be helpful. Below you can find codelabs with instructions on how to quickly get started with several image CDNs.

The right way to manage & deliver images on the web

Image CDN with automatic optimization, real-time transformation, and storage that you can integrate with existing setup in minutes.

imagekit.io

GET STARTED FOR FREE

25%

Customer experience is of paramount importance at BigBasket. We looked to solve image optimization and delivery as a part of our experience optimization exercise across BigBasket's apps and website. While we were looking at other tools in the market and deliberating on building a solution internally, ImageKit came in just at the right time. It solved our main concerns - excellent performance and optimization, the right visual quality, an almost zero-effort integration, and great pricing. We didn't have to think of any other alternative after going through ImageKit's demo.

Rupesh Kumar, Director of Engineering, BigBasket

READ FULL STORY

TESTIMONIALS

84% reduction in image bandwidth

ImageKit allows us to deliver optimized images across web and mobile with minimal effort. We don't have to think much about compressing the images to the right level or delivering it in the right format. And it has made life simpler for our developers.

Jon Arne Sæterås / APR 26, 2017 / 13 comments

Let The Content Delivery Network Optimize Your Images

QUICK SUMMARY ↬ *Sometimes you have to step back and ask why a tradition exists. In mobile-first design, serving an image in three sizes — one for smartphones, one for tablets and one for desktops — using media queries and responsive images has become a tradition. But is it the best solution? It's most likely better than doing nothing, but how well does it actually work? And is there room for improvement? In this article, we'll look closely at how well the one-size-per-form-factor approach really works and how we can use smart content delivery networks to improve image performance.*

It's most likely better than doing nothing, but how well does it actually work? And is there room for improvement? In this article, we'll look closely at how well the one-size-per-form-factor approach really works and how we can use smart content delivery networks to improve image performance.

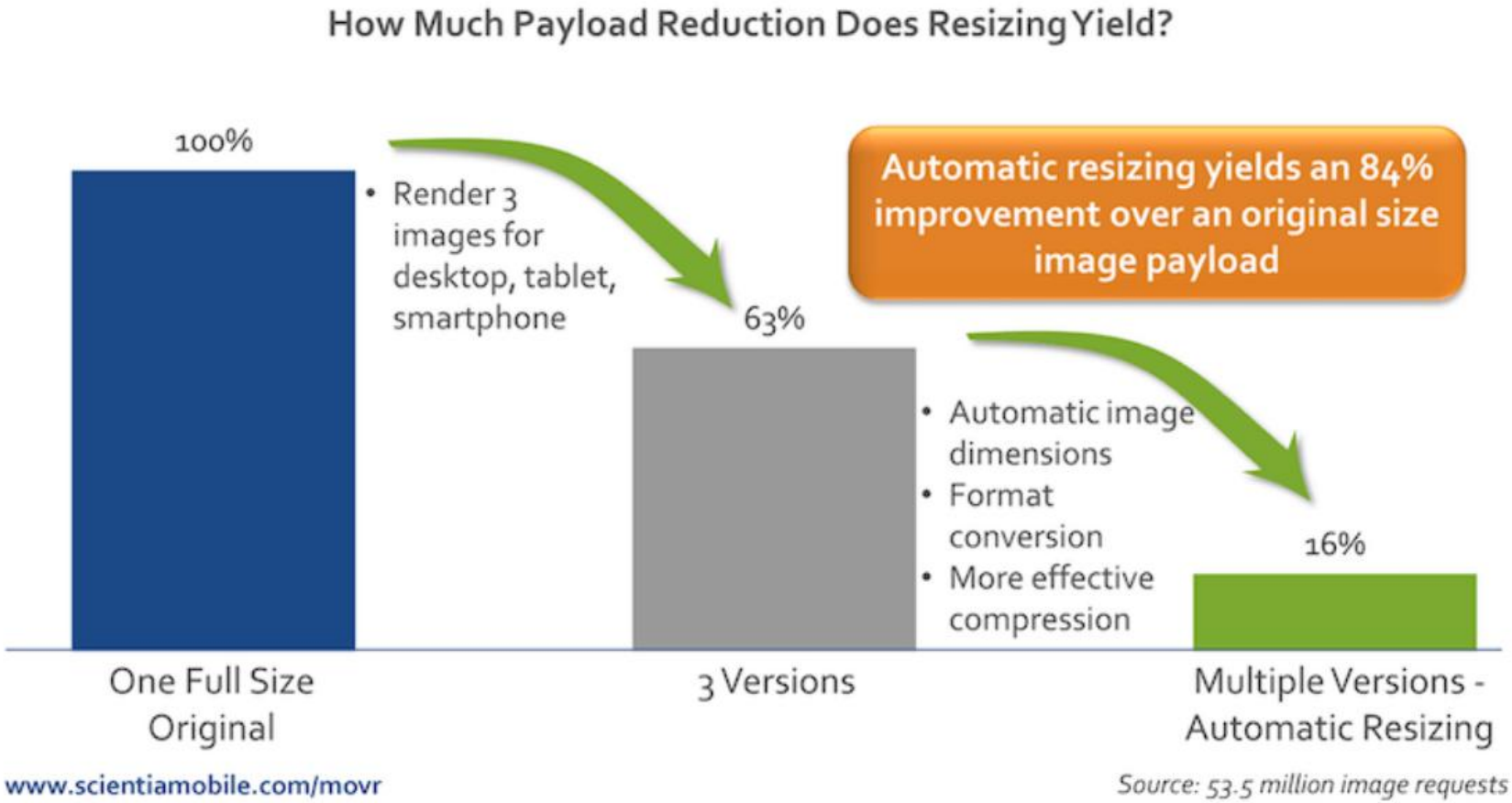
Over time, websites have become more image-heavy. Research by Radware indicates that the average e-commerce web page is over 1.3 MB; 64% of that payload comes from images. More than half of your users (and potential customers) will abandon your website if it takes longer than three seconds to load. Literally millions of dollars are at stake in this question of how images affect web performance.

The latter was the way to go until responsive images came around. The idea was that the browser shouldn't load the images by reading the HTML, but instead should load and execute JavaScript to figure out which images to download. Usually, it would query the viewport's width, compare that width to the static breakpoints and then select the best fit out of a few predefined image sizes. As we now know, this approach has two major problems. First, it breaks browser preloading; the browser can't start loading images referenced in the markup right away, rather having to wait for the JavaScript to execute. Secondly, there is a significant risk of an oversized version of an image being downloaded because you have only a few predefined image sizes to pick from.

Replacing this JavaScript-based image-loading with the new responsive images specification addresses the preloading issue. However, the risk of an oversized version of an image being downloaded is just as significant if you have breakpoints for only three image sizes.

Picking the right, and right number of, breakpoints is not an easy task. Even if tools exist to help you in the process, breakpoints tend to be a moving target. The ideal breakpoints today could change tomorrow due to new screen sizes and form factors.

An experiment done by ScientiaMobile found that it takes on average only eight requests for a given image to surpass three breakpoints. The experiment collected data over four months and compared the size of the image actually served to the optimal size for the particular device and screen size. Due to the wide diversity of devices of different forms and shapes accessing the web over time, statistically, the ninth image request will require a size that does not exist and most likely will get the performance penalty of downloading a larger and heavier image than necessary. The more requests you get for a given image's URL, the more fragmented will be the devices making the requests: At 180 requests, you will surpass 11 breakpoints. At 1,000 image requests, you will surpass 20 breakpoints.


How Much Payload Reduction Does Resizing Yield?
100%
• Render 3 images for desktop, tablet, smartphone
63%
Automatic resizing yields an 84% improvement over an original size image payload
• Automatic image dimensions
• Format conversion
• More effective compression
16%
One Full Size Original
3 Versions
Multiple Versions - Automatic Resizing
www.scientiamobile.com/movr
Source: 53.5 million image requests

Smart Content Delivery Networks

As we've seen, displaying an image on a web page might not be as easy as it sounds if your performance budget is tight. There are many factors to account for. Picking the best breakpoints, image formats, image sizes, compression rate and implementation is challenging enough, but the solution should also be future-friendly. As new devices and form factors are launched into an already diverse landscape, supporting legacy devices becomes just as important as supporting the latest fashion. Automating this process seems to be worthy of serious consideration.

We are starting to see several turnkey solutions that optimize images in this manner. These solutions are smart content delivery networks (CDNs), and they have intelligence built in at the edge of the network. These CDNs offer device-aware image optimization by combining three elements: device detection, real-time image optimization and a classic CDN cache geographically close to the end user. Unlike regular CDNs that do not have an interest in minimizing your payload, smart CDNs seek to improve web performance by tailoring your payload to the capabilities of the device and even network conditions.

geographically close to the end user. Unlike regular CDNs that do not have an interest in minimizing your payload, smart CDNs seek to improve web performance by tailoring your payload to the capabilities of the device and even network conditions.

What options do you have if you want to implement a dynamic image-optimization CDN on your website? Be aware that there are a lot of image-manipulation services for performing static operations such as resizing, cropping and filtering. This is not what we're looking for. Nor are we looking for a service that uses JavaScript to determine the best size and format of an image.

We're looking for a CDN that implements server-side logic, or "edge logic," using client hints or device detection to determine the best image size and format. This is known as content negotiation. This makes the list significantly shorter. Let's compare the most prominent contenders:

	Client hints	Device detection	Push or pull

	Client hints	Device detection	Push or pull
Cloudinary	yes	no	push/pull
imgix	yes	no	push
ImageEngine	yes	yes	pull

All of the above use client hints to determine the best size. Client hints is a fairly new initiative, currently implemented only in Blink-based browsers, and it includes some additional information about preferred image sizes in the HTTP request. The server can use this information to generate a properly sized image. Even though support and adoption of client hints are growing, only about 3% of image requests come with client hints. This number is expected to grow. In the near future, however, it would be a good idea to pick a CDN with device detection built in, so that all images are optimized. If you want to play around with the concept a bit, look for a pull-based CDN (which requires less configuration) with a trial option or a free tier.

this information to generate a properly sized image. Even though support and adoption of client hints are growing, only about 3% of image requests come with client hints. This number is expected to grow. In the near future, however, it would be a good idea to pick a CDN with device detection built in, so that all images are optimized. If you want to play around with the concept a bit, look for a pull-based CDN (which requires less configuration) with a trial option or a free tier.

All of the CDNs mentioned above require registration. Cloudinary have a free tier. ImageEngine a and Imgix has a trial concept, which is convenient if you want to try before you buy. Let's have a closer look at ImageEngine. ImageEngine is pull-based, which means you don't have to upload your images anywhere before you start. With ImageEngine, you simply keep images on your server, and ImageEngine will pull images from there on demand.

Once registered for ImageEngine, you will get a dedicated hostname, which is your CDN's URL. The only thing left to do is prefix all of your image sources with that hostname.

Say your original image tag is like this:

Your new image tag with ImageEngine would look like this:

```
<img src="//{personal-token}.imgeng.in/https://mysite.xyz/dog.png" alt="My dog">
```

In this example, ImageEngine makes use of client hints and device detection to determine the optimal pixel size, compression ratio and image format for each device requesting the image. If you want to be more specific, all services listed above support explicit "commands" to override any automatically detected parameters. For ImageEngine, requesting a 360-pixel-wide image in WebP format would look like this:

```
<img src="//{personal-token}.imgeng.in/w_360/f_webp/https://mysite.xyz/dog.png" alt="My dog">
```

Of course, to make use of client hints, remember to enable it in your markup. Put this in

In this example, ImageEngine makes use of client hints and device detection to determine the optimal pixel size, compression ratio and image format for each device requesting the image. If you want to be more specific, all services listed above support explicit "commands" to override any automatically detected parameters. For ImageEngine, requesting a 360-pixel-wide image in WebP format would look like this:

Of course, to make use of client hints, remember to enable it in your markup. Put this in your `<head>` tag:

```
<meta http-equiv="Accept-CH" content="DPR, Viewport-Width, Width">
```

ImageEngine even has a WordPress plugin, which handles all of this automatically for you.

maintain. But we must not forget why we do this. We're not doing this for ourselves as developers, but for our end users. Hence, automating this process if we can makes sense. This is not a job for humans. Offloading this task is a win-win: easier maintenance and less data transfer.

Smart CDNs give you core CDN functionality, as well as dynamic and automatic image optimization for any size and form factor by using client hints and device detection at the edge of the network. Experiments suggest that payload savings can run as high as 84% compared to serving one static image, and run around 75% compared to the common one-size-per-form-factor approach.

Luckily, a few smart CDNs are out th s fairly easy to get started and measure the difference. Once you've created an account, you can put up a simple page and run it through WebPagetest to see the difference.

(da, vf, al, il)

Explore more on

cloudinary.com/blog/delivering_all_your_websites_images_through_a_cdn/

Apps Gmail XL 1 XL 2 trillo 3000 Drive Utube bucket W Git T f suffix Reading list

Cloudinary PRODUCTS SOLUTIONS RESOURCES ABOUT US PRICING

Contact Support Documentation Login

CDN for Images: Optimize and D...

+ Add tags

Share Manage Highlights

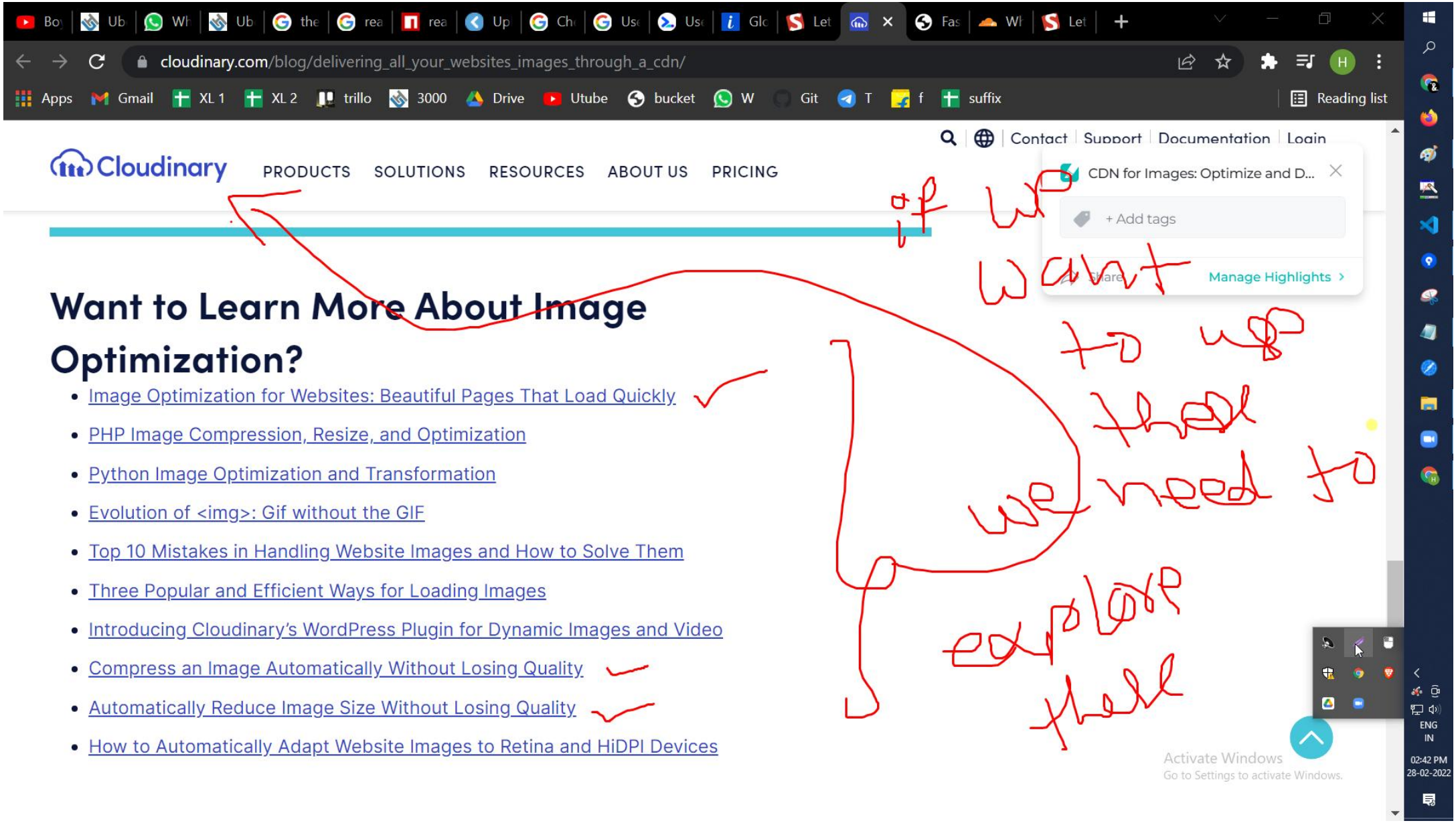
server, and show it on your website:
http://upload.wikimedia.org/wikipedia/commons/4/46/Jennifer_Lawrence_at_the_83rd_Academy_Awards.jpg

- Now, simply replace this image's URL with a URL with Cloudinary as a prefix:
https://res.cloudinary.com/demo/image/fetch/http://upload.wikimedia.org/wikipedia/commons/4/46/Jennifer_Lawrence_at_the_83rd_Academy_Awards.jpg

Both URLs return the exact same image, only the second one is cached and delivered through fast, localized CDNs and not via your local web server. Better experience to your users. Lower load on your server. Lower hosting costs.

Activate Windows
Go to Settings to activate Windows.

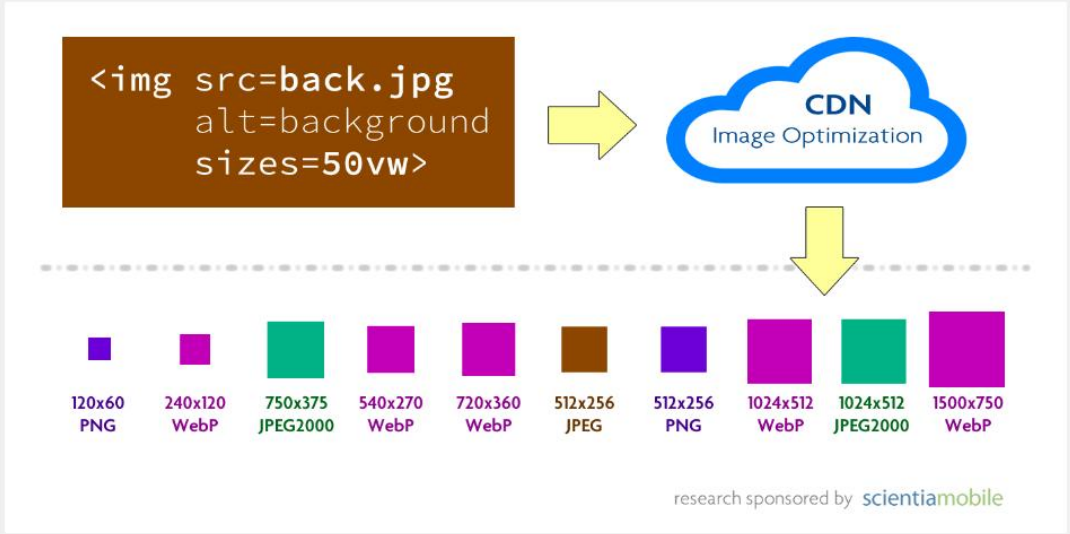
ENG IN
02:41 PM
28-02-2022


cloudinary.com/blog/delivering_all_your_websites_images_through_a_cdn/
Cloudinary
PRODUCTS
SOLUTIONS
RESOURCES
ABOUT US
PRICING
Contact
Support
Documentation
Login
CDN for Images: Optimize and D...
+ Add tags
Manage Highlights
Want to Learn More About Image Optimization?
Image Optimization for Websites: Beautiful Pages That Load Quickly
PHP Image Compression, Resize, and Optimization
Python Image Optimization and Transformation
Evolution of <img>: Gif without the GIF
Top 10 Mistakes in Handling Website Images and How to Solve Them
Three Popular and Efficient Ways for Loading Images
Introducing Cloudinary's WordPress Plugin for Dynamic Images and Video
Compress an Image Automatically Without Losing Quality
Automatically Reduce Image Size Without Losing Quality
How to Automatically Adapt Website Images to Retina and HiDPI Devices
if we want to up that we need to explore that
Activate Windows
Go to Settings to activate Windows.
ENG
IN
02:42 PM
28-02-2022

Maximiliano Firtman (firt.dev)

Sep 17, 2019 · 15 min read

Faster Paint Metrics with Responsive Image Optimization CDNs



Maximiliano Firtman (firt.dev)
9.7K Followers

mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps

research sponsored by scientiamobile

When you are interested in web performance, image compression and optimization are key aspects to consider that will impact many user-centric metrics, such as First Meaningful Paint and Speed Index. I've researched the image optimization solutions offered by the main CDN providers in order to identify the most relevant techniques in use today to map the differences among providers and help you pick one in case you need them. I also looked at what opportunities for improvement are still there.

Pictures are worth a thousand words… if they load

On average 51% transferred bytes on a page load are images, and more than half of those bytes are used above the fold, so they impact performance metrics. With thousands of different devices browsing the web today, offering the best image for each context is challenging.

Today, the context to decide the right image includes layout dimensions, pixel density (dpr), the best format supported (such as WebP, APNG, JPEG2000, or

mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps

of those bytes are used above the fold, so they impact performance metrics. With thousands of different devices browsing the web today, offering the best image for each context is challenging.

Today, the context to decide the right image includes layout dimensions, pixel density (dpr), the best format supported (such as WebP, APNG, JPEG2000, or just classic web formats), user's data-saving preferences and connection speeds.

A default website without a truly responsive images solution typically involves the creation of 1 to 5 different versions of one image, but to provide the best possible experience and best performance results we need to create *n* different variants of the same image. That's were image optimization CDN providers come in handy in addition to their traditional function of caching content at the edge, they take care of image compression, reformat, resize and other techniques in the cloud, as a service, with a Content Delivery Network in place that will serve that file closer to the user.

Why image optimization CDNs help in Web Performance?

When you run a lighthouse test, typically you see a list of opportunities to help the user-centric metrics. Many of those opportunities are around image

Maximiliano Firtman (firt.dev)
9.7K Followers

mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps

Related

Optimizing fonts for the web
This post was originally published on my blo…

Exploring Chakra UI—Part 2: Setting up axios, react-query and react-router-dom.

4 stages you need to test with React Que
Testing Library

How to create a Theme in React/Typescript (Context API) with styled-components

thousands of different devices browsing the web today, offering the best image for each context is challenging.

Today, the context to decide the right image includes layout dimensions, pixel density (dpr), the best format supported (such as WebP, APNG, JPEG2000, or just classic web formats), user's data-saving preferences and connection speeds.

A default website without a truly responsive images solution typically involves the creation of 1 to 5 different versions of one image, but to provide the best possible experience and best performance results we need to create *n* different variants of the same image. That's were image optimization CDN providers come in handy in addition to their traditional function of caching content at the edge, they take care of image compression, reformat, resize and other techniques in the cloud, as a service, with a Content Delivery Network in place that will serve that file closer to the user.

Why image optimization CDNs help in Web Performance?

When you run a lighthouse test, typically you see a list of opportunities to help the user-centric metrics. Many of those opportunities are around image

Maximiliano Firt
9.7K Followers
mobile+web devel
Check firt.dev for n
experiences | PWA
Mobile Apps

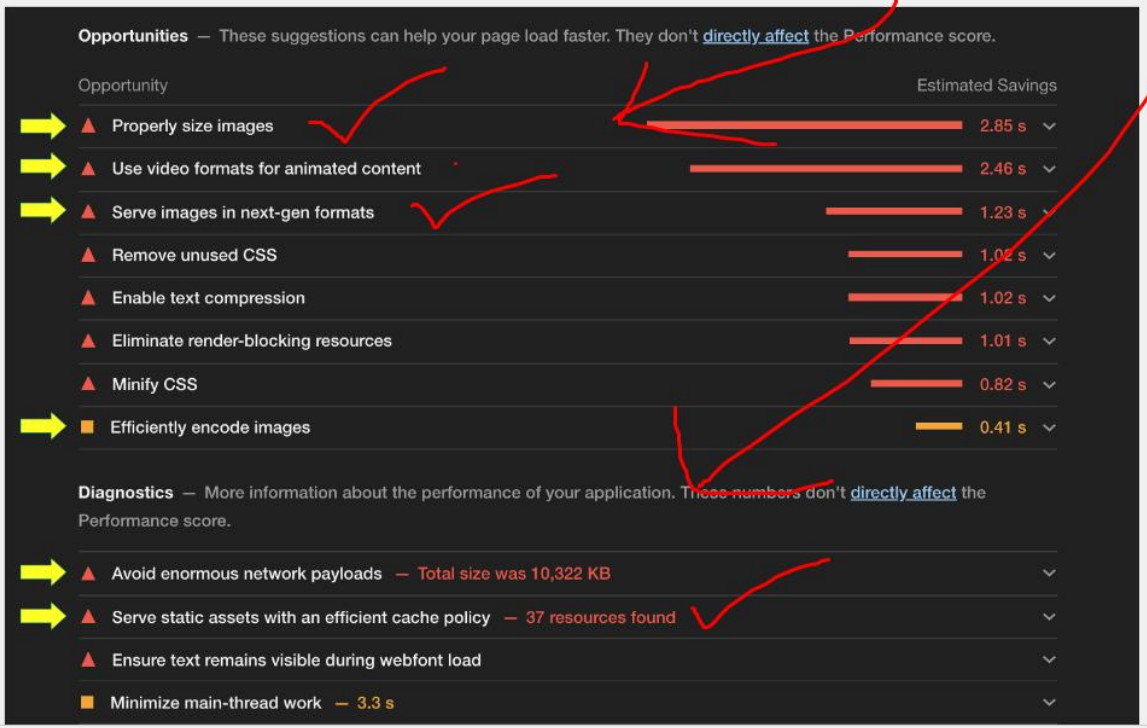
Related


Optimizi
This pos


Explorin
react-qu


4 stages
Testing I

When you run a lighthouse test, typically you see a list of opportunities to help the user-centric metrics. Many of those opportunities are around image manipulation and image optimization CDNs can help you with those without too much work on your side.



312 1

mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps

Follow

Related


Optimizing fonts for the web
This post was originally published on my blo…


Exploring Chakra UI—Part 2: Setting up axios, react-query and react-router-dom.


4 stages you need to test with React Query + Testing Library

How to create a Theme in React/Typescript (Context API) with styled-components

Providers in this research

The research includes well-known names in the web industry, such as Akamai's Image Manager, Cloudflare's Polish & Mirage, and ScientiaMobile's Image Engine. In addition to those, I've added some new names in the field, such as ImgIX, Fastly's ImageOptimizer and CloudImage. Some providers not offering CDN services (such as ImageKit, Cloudinary) were worth mentioning as they are addressing the same use-case, albeit with a different approach.

I identified three kinds of providers:

- **General Purpose CDNs**, such as Akamai, Fastly or Cloudflare. They offer image compression services as an add-on or a product on top of their main solutions. Typically these solutions involve changing your main DNS entry or directly the DNS server in your domain registrar. Usually, we don't need to change our HTML image references to deploy these solutions. These cases involve a bigger deal, as they affect more than just image loading.
- **Image Optimization CDNs** such as ImgIX, CloudImage or ImageEngine that are mainly focused on images and can be used with any architecture. These solutions typically involve referencing to a different domain (such as images.mydomain.com) which may require changing all the image references in the HTML either manually or through an SDK.
- **Image Optimization Services with external CDNs**: Some providers in the research are not offering CDN services, only mage optimization services, but they might use an external basic CDN service in their solutions, such as Amazon CloudFront. In those cases, you can also integrate other general-purpose CDN on top to have a full or better CDN coverage.

Characteristics

Characteristics

Here you can see a comparison of the main features available from each CDN providers, before seeing results from testing.

	Provider	DNS Firs...	CDN	Custom do...	Documentation	Ir
1	Akamai	Yes	Yes	N/A	★★★	
2	ImageEngine	No	Yes	Yes	★★★★	
3	Cloudflare	Yes	Yes	N/A	★★★	
4	Fastly IO	Yes	Yes	N/A	★	
5	Imgix	No	Yes	No	★★★★★	
6	CloudImage	No	Yes	No	★★★	
7	ImageKit	No	No	Yes	★★★	
8	Cloudinary	No	No	Yes	★	
9						

9 records

Sum 23

Web Performance Summary

If you are out of time, let's see a summary of the impact that each provider might have on your web performance by optimizing responsive images.

	Provider	CDN	Device Libr...	Client-Hints	Save-Data	We
1	Akamai IM*	Yes	Unknown	No	No	Ye
2	Cloudflare	Yes	No	No	No	Ye
3	Fastly IO	Yes	No	No	No	Ye
4	ImageEngine	Yes	Yes	Yes	Yes	Ye
5	Imgix	Yes	No	Yes	Yes	Ye
6	CloudImage	Yes	No	No	No	Ye
7	ImageKit	External	No	Yes	No	Ye
8	Cloudinary	External	Partial	Yes	Yes	Ye
9						

Detection mechanism

There are generally two ways to obtain images in optimized format or dimensions with these providers:

- **Explicit directives**: where we specify width, height, format, etc. of the image that we would like to obtain back from the CDN or service. This is typically paired with certain automatism, so requesting a page with a given width will cause the height to be automatically calculated to maintain the aspect ratio for example.
- **Automatic transformations**: relying on the CDN to understand the context of the request and apply the most suitable parameters for the transformation.

When doing responsive images, we'd like to let the CDN pick the best format and dimensions based on the context and the placeholder for the image to keep our HTML semantic and without the need of client-side code, so I will focus more on automatic transformations in my testing. When doing native apps and you have full control of image loading, you might want to see more on the explicit directives side of CDNs.

Device Library Support

ImageEngine seems to be the only provider publicly known to employ a device detection library within its decision algorithm. ImageEngine uses the well-known WURFL library to detect the device from the User-Agent string and other HTTP headers, to then make a decision regarding screen width, screen density and best-supported format for a given image for a given device, even if no Accept header is explicitly supporting that selection.

Client Hints

Client Hints is one of the current useful (but less well-known by the web community) features offered by browsers. In fact, finding accurate information about current compatibility and support was hard. There is a good update article explaining Client Hints in developers.google, but it doesn't provide info about compatibility.

Things get even more complicated in the case of image CDN solutions as we are loading images from cross-origin servers (i.e. images.mydomain.com) instead of from first-origin (i.e. mydomain.com). Because of some privacy concerns, Chromium has decided to limit Client Hints to cross origins, and I couldn't find updated information on the matter. Therefore, it was time fa little research.

Client Hints are sometimes even better than using a server-side library as they can, for example, take current zoom level on a desktop or a user's preference in scale on mobile devices into consideration. For example, a Retina MacBook with a DPR of 2 will send a DPR or 2.50 to the server if you zoom in your browser by 125%. On Android, a Pixel 3A with a DPR of 2.5 will expose a DPR of 3.375 if

Browser Availability

Current available Client Hints are: *viewport-width*, *dpr* (device pixel ratio or pixel density), *width* (when the browser knows the exact width needed for an image), *ECT* (effective connection type, such as 2g, 3g or 4g), *RTT* (round trip time, related to the network latency), *downlink* (estimated download bandwidth), *device-memory* (expressed in GiB rounded to a limited set such as 0.5, 1, 2, 4 or 8) and *save-data* that will expose user's intent to reduce bandwidth usage. With this information, the server can make smarter decisions regarding image transformation and send the best image to the client.

As of September, 2019 this is the compatibility of Client Hints support for first-origin images (such as when you use Cloudflare or Akamai) and cross-origin images (such as when you use ImageEngine or CloudImage).

	Name	Chrome Android	Samsung Internet
1	Client Hints Support	Yes	Yes
2	Save-Data Support	Yes	No
3	First Origin: Width, DPR an...	Yes	Yes
4	First Origin: ECT, Downlink, ...	Yes	Yes
5	First Origin: Device-Memory	Yes	Yes
6	First Origin: Save-Data	Yes	N/A
7	Cross Origin: Width, DPR a...	Yes	No
8	Cross Origin: ECT, Downlin...	No	No
9	Cross Origin: Device-Memo...	Yes	No
10	Cross Origin: Save-Data	Yes	N/A
11			
12			

12 records

Airtable Download CSV View larger version

Based on that compatibility, today if an Image CDN supports Client Hints, these are the percentage of visitors that can receive more accurate images to their situation:

312 1

Maximiliano Firtman (firt.dev)

9.7K Followers

mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps

Follow

Related

Optimizing fonts for the web

This post was originally published on my blo…

Exploring Chakra UI—Part 2: Setting up axios, react-query and react-router-dom.

4 stages you need to test with React Que… Testing Library

How to create a Theme in React/Typescript (Context API) with styled-components

2	Save-Data Support	Yes
3	First Origin: Width, DPR and Viewport-Width	Yes
4	First Origin: ECT, Downlink, RTT	Yes
5	First Origin: Device-Memory	Yes
6	First Origin: Save-Data	Yes
7	Cross Origin: Width, DPR and Viewport-Width	Yes
8	Cross Origin: ECT, Downlink, RTT	No
9	Cross Origin: Device-Memory	Yes
10	Cross Origin: Save-Data	Yes
11		
12		

12 records

Airtable Download CSV View larger version

Based on that compatibility, today if an Image CDN supports Client Hints, these are the percentage of visitors that can receive more accurate images to their situation:

- First-origin CDNs —73% of desktop users and 67% of mobile users
- Cross-origin CDNs — No desktop users and 61% of mobile users

Maximiliano Firtman (firt.dev)
9.7K Followers

mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps

The Save-Data Client Hint

The Save-Data is the only hint that doesn't need being opted-in by the developer. If the user enables Save Data in settings, a compatible browser will send that HTTP header automatically. For example, iOS 13 adds a "Low Data Mode" on cellular networks, so Safari can add the Save-Data header, even if they are not supporting Client Hints yet. Unfortunately, Safari is not honoring that setting yet.

According to the recent article from Tim Kadlec on Save-Data Usage, between 4% and 20% of your users might have the data saver settings enabled, based on your audience geography and type. When we blend those stats with current compatibility of the feature, we can conclude that supporting Save Data to compress our images even further, will increase performance and honor user's choice for 1% to 6% of your total visits. The impact of these numbers is even bigger when you realize that typically those users are browsing the web under poor network conditions.

CDN Availability

Now it's time to look at Client-Hints support by Image Compression CDNs based

Maximiliano Firtman (firt.dev)
9.7K Followers
mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps

Related

Optimizing fonts for the web
This post was originally published on my blo…

Exploring Chakra UI—Part 2: Setting up axios, react-query and react-router-dom.

4 stages you need to test with React Que
Testing Library

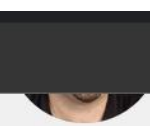
How to create a Theme in React/Typescript (Context API) with styled-components

poor network conditions.

CDN Availability

Now it's time to look at Client-Hints support by Image Compression CDNs based on my testing.

	CDN	Client-Hints	DPR	Viewport-...	Width	Width+D
1	Cloudflare	No	N/A	N/A	N/A	N/A
2	Imgix	Yes	Yes	No	Yes	Yes
3	ImageEngine	Yes	Yes	No	Yes	Yes
4	Fastly IO	No	N/A	N/A	N/A	N/A
5	CloudImage	No	N/A	N/A	N/A	N/A
6	Cloudinary	Yes	Yes	No	Yes	Yes
7	ImageKit	Yes	No	No	No	No
8	Akamai	No	N/A	N/A	N/A	N/A



Maximiliano Firtman (firt.dev)
9.7K Followers

mobile+web developer, author, trainer, spea
Check firt.dev for more articles and learning
experiences | PWAs, Web Performance, Java
Mobile Apps

Follow

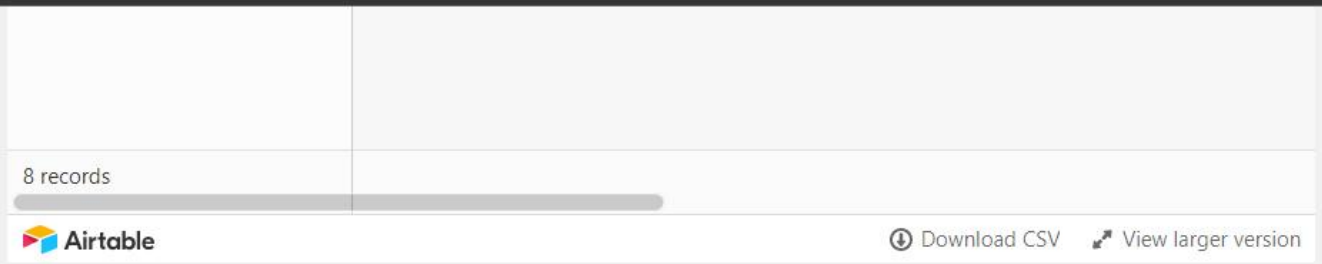
Related

Optimizing fonts for the web
This post was originally published o

Exploring Chakra UI—Part 2: Settin
react-query and react-router-dom.

4 stages you need to test with React
Testing Library

How to create a Theme in React/Typ
(Context API) with styled-componer



We can see that no CDN takes advantage of the network-related client-hint. In this case, only first-origin CDNs could take advantage of it, but none is actually doing it. The idea is that if the RTT is too high, or if the ECT is actually 2g or slow-2g we know for sure we must send a high-compressed version of the image.

Also, no image CDN is taking advantage of the device memory client hint, even if all of them are able to access that data as long as the developer opts-in into it. In cases like this, we could make a difference between a high-end Android and a low-end Android, or between an old desktop with 1GiB of RAM vs a high-end laptop with +8GiB of RAM, even if both are using the same desktop OS and browser.

Input Formats diversity

Every provider accepts the well-known PNG and JPEG files for optimizing, but not all of them accept SVG files or animations as input. CDNs that do not support a given file format will simply ignore the file and just bypass it through the CDN without any compression or transformation.

Output Formats diversity

Every provider supports converting images into WebP format if the *Accept* header is present. That's basic for all CDNs; however, only ImageEngine goes beyond PNG, JPEG, and WebP.

Sometimes picking the format only based on the browser might lead to a bigger file; for example, **ImageKit** delivers WebP files to compatible browsers only if the WebP file is smaller than the same image in PNG or JPEG. So the same image to the same browser might be delivered in WebP or PNG for example based on the screen size to maximize data transfer opportunities.

screen size to maximize data transfer opportunities.

ImageEngine is the only provider in the test offering JPEG-2000 format (JP2 files); Akamai says the format is available based on the documentation, but I couldn't test it. This format is useful for Safari on macOS, and for all iOS and iPadOS browsers. That is a great advantage over the other CDNs, as WebP is not available on these browsers yet, and JPEG-2000 files will typically be smaller than a normal JPEG with the same visual quality. This is done thanks to the usage of a device detection library because, for some reason, WebKit does not expose image/jp2 support on the *Accept* header; CDNs that rely exclusively on HTTP headers, won't know that JPEG-2000 is available and will rely on classic compressed PNGs or JPEGs.

JPEG-XR was important for a while to support IE and Edge, but I didn't test this as I think in terms of market share it's not important and it will be less important in the future with Edge moving to Chromium and WebP support.

Similarly, ImageEngine is the only CDN being able to convert animated GIFs into Animated WebP for Android and Chrome mainly, and into MPEG-4 videos, supported now on <img> elements on Safari for macOS, iOS and iPadOS.

as I think in terms of market share it's not important and it will be less important in the future with Edge moving to Chromium and WebP support.

Similarly, ImageEngine is the only CDN being able to convert animated GIFs into Animated WebP for Android and Chrome mainly, and into MPEG-4 videos, supported now on <img> elements on Safari for macOS, iOS and iPadOS.

Cloudinary knows about JPEG-2000, but what they said is: *"JPEG 2000 is better only if you need transparency or for huge images. Encoding JPEG 2000 is slower and CPU intensive and also with f_auto it will add another transformation. So we do not support automatic delivery of JPEG 2000"*

Compression Comparison

Compression Comparison

Optimizing Images without any manual sizing

Every provider let you provide arguments to define the exact width and height that you prefer for the image, and also the resulting format (JPEG, PNG, WebP, etc.). While that ability might be good for image management in general and for native apps, it's not something interesting for web design as we just want to load the best possible image for the current context, without the need to define every argument explicitly.

Optimizing Images without any manual sizing

Every provider let you provide arguments to define the exact width and height that you prefer for the image, and also the resulting format (JPEG, PNG, WebP, etc.). While that ability might be good for image management in general and for native apps, it's not something interesting for web design as we just want to load the best possible image for the current context, without the need to define every argument explicitly.

In this case, I analyze what happens in each CDN provider with bitmaps when we have an *img* element without any manual size declaration with something like:

<img src="https://cloud-url/image" sizes="100vw" alt="no size declaration in URL">

	Name	Score	Resize Image	Desktop Saving:
1	Imgix	★★★★★★★	Mobile-only	
2	ImageEngine	★★★★★★★★★★	Yes	
3	Cloudflare	★★★★★	No	

As you can see ImageEngine beats all the other providers in this category by far. The main reason is the usage of a device detection library that let it resize the image to the maximum available width for each device. That's also something you might not want, so use with care. Finally, iPhones won't get the right size or resolution because Apple decided to mask the iPhone model from the user agent, so a generic iPhone (the least powerful hardware supporting that iOS version) is always assumed by ImageEngine.

CDNs with a JavaScript SDK available will reformat these URLs and include all the sizing and dpr data from the current client context, however, moving image loading from being HTML-based to JavaScript-based is often not ideal for websites, so I'm leaving that testing out.

Optimizing Vector Graphics

SVG files are often ignored by Image Compression CDNs, but here Cloudinary and CloudImage exhibit a difference: they convert the SVG into a bitmap. The final bitmap image has different sizes for different devices, but I couldn't find a consistent answer to how they calculate those sizes. Cloudinary seems to define the final size based on a device library, and CloudImage just picked a 1800px width for the bitmap version for all devices. To be honest, I'm not sure this is something web designers will want for SVG-compatible browsers; in fact, for some devices, the WebP result from the CDN is bigger in size than the original SVG served with GZIP or Brotli.

Most CDNs, such as ImageEngine or Cloudflare, are serving the file compressed with either GZIP or Brotli based on browser compatibility as SVG is the only image format that is text-based.

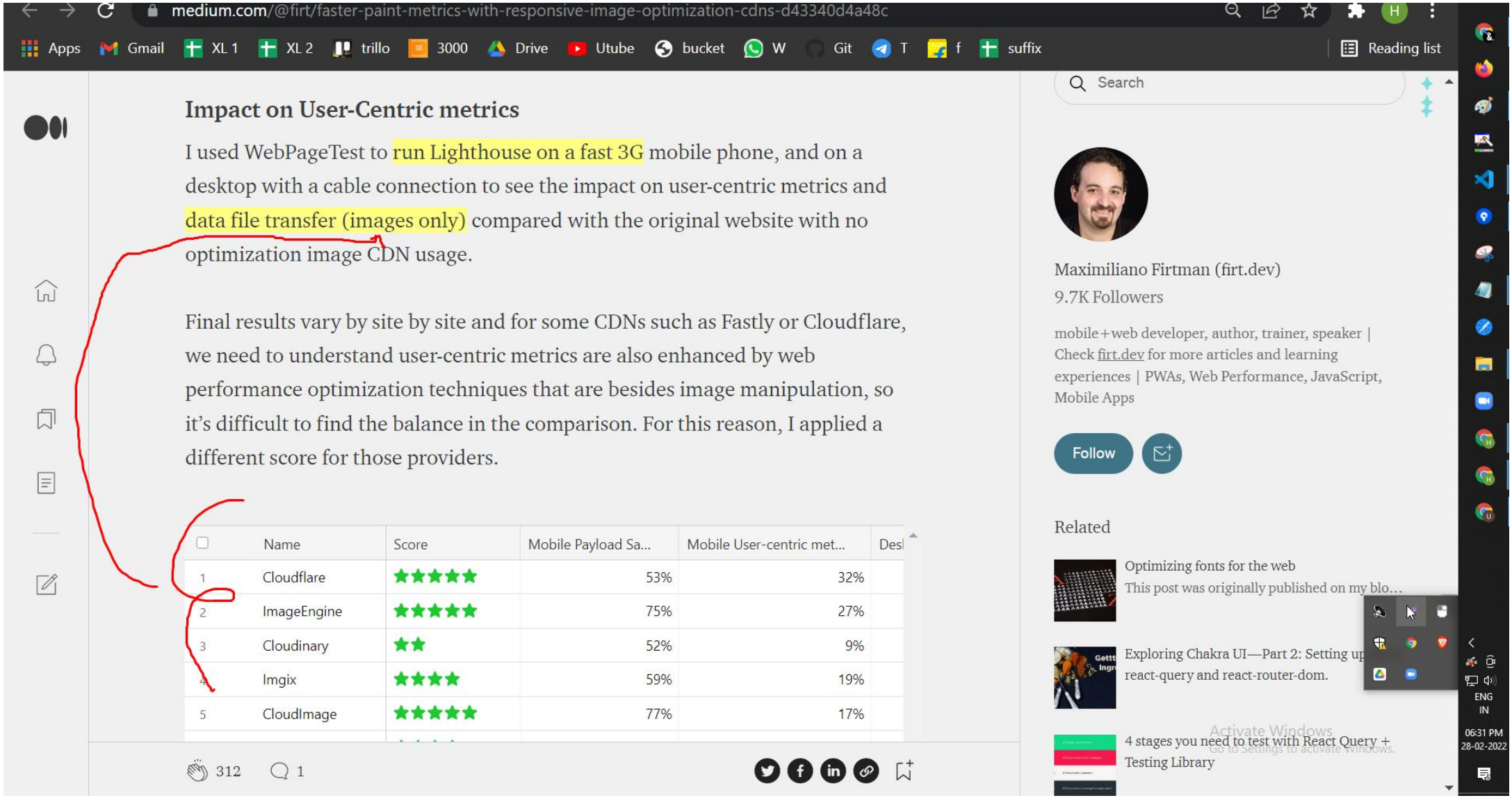
Maximiliano Firtman (firt.dev)
9.7K Followers

mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps

Optimizing Animations

Animated GIFs can be really painful for web performance because it's a very old and unoptimized format. In these tests, I analyzed each CDN and what they do with these files, including format conversion (APNG, AWebP or MP4) and resizing.

	Name	Supports GIF	Compression	To Animated WebP
1	ImageEngine	Yes	Yes	Yes
2	Cloudflare	Yes	Yes	No
3	Imgix	No		
4	Akamai IO	Yes		
5	Fastly IO	No		
6	CloudImage	No		
7	ImageKit	Yes	Yes	Yes
8	Cloudinary	No		
9				


medium.com/@firt/faster-paint-metrics-with-responsive-image-optimization-cdns-d43340d4a48c
Apps Gmail XL 1 XL 2 trillo 3000 Drive Utube bucket W Git T f suffix Reading list
Impact on User-Centric metrics
I used WebPageTest to run Lighthouse on a fast 3G mobile phone, and on a desktop with a cable connection to see the impact on user-centric metrics and data file transfer (images only) compared with the original website with no optimization image CDN usage.
Final results vary by site by site and for some CDNs such as Fastly or Cloudflare, we need to understand user-centric metrics are also enhanced by web performance optimization techniques that are besides image manipulation, so it's difficult to find the balance in the comparison. For this reason, I applied a different score for those providers.
Name Score Mobile Payload Sa... Mobile User-centric met... Desl
1 Cloudflare 53% 32%
2 ImageEngine 75% 27%
3 Cloudinary 52% 9%
4 Imgix 59% 19%
5 CloudImage 77% 17%
312 1
Search
Maximiliano Firtman (firt.dev)
9.7K Followers
mobile+web developer, author, trainer, speaker | Check firt.dev for more articles and learning experiences | PWAs, Web Performance, JavaScript, Mobile Apps
Follow
Related
Optimizing fonts for the web
This post was originally published on my blo…
Exploring Chakra UI—Part 2: Setting up react-query and react-router-dom.
4 stages you need to test with React Query + Testing Library
Activate Windows
Go to Settings to activate Windows.
ENG IN
06:31 PM 28-02-2022

What's next

If you have a website with an average or high amount of images, using a responsive image CDN solution might get mandatory if you want to increase your user-centric paint metrics. Testing the CDNs wasn't easy; there are many challenges and differences in adopted practices but I hope you can now have a better idea of the different solutions in the market, their differences and their technical advantages.

Device-detection is definitely something nice to see in an image compression CDN because Client-Hints are not available today on all browsers, not everything can be detected with Accept headers and web designers want to define responsive images in the simplest way possible. Most of the CDNs lack support for this. ImageEngine is the only one that I could confirm use a device library to make decisions server-side.

But there are also some other features not seen on any CDNs that might impact

mobile+web developer, author, traine
Check firt.dev for more articles and lea
experiences | PWAs, Web Performance
Mobile Apps

Disclaimer about the tests

For most providers, testing the service was simple: registering for a trial account, checking documentation and testing in a couple of minutes. General Purpose CDNs such as Cloudflare took a bit more time, including changing the DNS server in the registrar (and even paying the Pro account — no image compression service on a free tier).

For testing I've used one of the most downloaded free creative commons HTML templates including their included images and some other images of my own, trying to be as diverse as possible: backgrounds, logos, icons, images with people, animations and an SVG file. Tests were made in September 2019, so if you are reading this in the future, the CDNs might have changed what they offer.

I've tested with desktop browsers and a diverse set of mobile devices: including an iPhone, a high-end Android, a low-end Android, a kaiOS feature phone, an iPad, and an Android tablet. On compatible browsers, I enabled Client Hints to see if there was any reaction, and I also tested the Save-Data flag.

On every provider, I tried to set up the CDN to work as automatically as possible in terms of format and quality selection, including setting WebP automatic

What is an image optimizer? | How to reduce image sizes

Image optimizers reduce image file sizes so that images are optimized for the Internet and can load quickly.

Copy article link

What is an image optimizer?

An image optimizer is a service, product, or library that makes image files smaller. Typically, an image optimizer will reduce the file size of an image by compressing and resizing it, ideally without compromising the quality of the image too much. This optimizes images for the web because they will take less time to load in a user's browser, increasing website speed and performance.

Why is it necessary to reduce image size?

All images that appear on a webpage need to be downloaded by the user's browser before they can be displayed. The larger an image is (in terms of file size, not dimensions), the longer it takes to download, and the more bandwidth it will will take up. If users are on a mobile device, large images will also use up a lot of their data as they download.

Therefore, keeping images small is crucial for website performance, and website performance is extremely important for SEO and for keeping users engaged and active on a website. Google prioritizes sites that load quickly, and users are more likely to bounce and less likely to convert if a webpage takes a long time to load.

How is image file size reduced?

The first step for reducing image size is to shrink its dimensions. The typical website will not need images that are 3,000-plus pixels wide, for example. (In fact, most desktop displays are 1,920 pixels wide or smaller.) Adjustments to the dimensions of an image should reduce file size without reducing quality.

Images can also be compressed. Image compressors (such as Photoshop's 'Save for Web' feature) can shrink JPEG files to a much lower resolution level, and the images will look essentially the same. However, images should still look professional, not pixelated. There's a tipping point where the resolution becomes so low that the accompanying performance gains are not worth it. Testing is important; images should appear professional on large monitors and small smartphone screens alike.

Are there other ways to optimize images besides resizing or compressing them?

The file format used for an image affects how large the file is. Most images for the web should be in JPEG format, not PNG or GIF. This is because it's easiest to adjust the quality (which affects the file size) with JPEG files. JPEG files are lossy, which means they lose visual information when they are compressed. As a result, compression can shrink JPEG files to a fraction of their original sizes, which is usually not possible with GIF and PNG files (both are lossless).

How does a CDN speed up images?

A CDN, or content delivery network, is a group of servers distributed around the world that store and deliver content, including images, to end users. CDN servers are optimized for speed, and they are located closer to end users than origin servers are, reducing latency and speeding up load times for images, video, and other content delivered over the Internet. Learn more about the Cloudflare CDN.

What is image SEO optimization?

Image search engine optimization and image optimization are separate, but related. Reducing image file size does help optimize images for search by reducing load times, and Google encourages developers to compress images when possible.

However, for an image to be truly optimized for search, developers should:

1. Give the image file a relevant, readable name, and include a relevant keyword if possible
2. Include a brief, descriptive image alt tag that is helpful for site visitors using screen readers and contains one or more keywords
3. Make images responsive and mobile-friendly
4. Caption images where doing so enhances the user experience and keeps users engaged
5. Include Open Graph and Twitter card images for social sharing