JaJuMa

Ultimate Image Optimizer

02/2023

Table of Contents

1. Introduction

Image optimization is key for performance optimization as well as User Experience and also SEO. One way to improve page load times is reducing the amount of data that needs to be transferred. Hence, with images usually making up for the biggest share of data to be downloaded for a webpage, it is crucial to choose the best file format and compression to reduce image sizes as much as possible and also lazy load images to avoid the page load and rendering being slowed down by images not visible to the visitor.
Of course, images should also still show in the best quality possible, even when compressed and also by using the visitors displays capabilities to the max.

Magento by default supports .jpg + .png images as file formats. Each of these can be optimized by applying lossless as well lossy compression.

However, with newer image file formats AVIF + WebP, size of images can be reduced even more with its superior compression and quality characteristics compared to their older JPEG and PNG counterparts.

JaJuMa Ultimate Image Optimizer Extension adds AVIF + WebP Support to new as well as existing Magento Stores. There is no need to change existing images. Also new images can be uploaded as usual as JPG or PNG, the extension will create and handle AVIF + WebP variants automatically.

For further details as well as comparisons between .jpg /.png and .avif/.webp regarding file size and quality, please see:

https://www.jajuma.de/en/jajuma-develop/extensions/ultimate-image-optimizer-extension-for-magento-2

Additionally, our extension allows you to add lazy loading, native and/or JS-based, for your optimized images and also high-resolution/Retina images can be automatically created for product images and added as 2x and/or 3x version for each image format, .png/.jpg, .avif and .webp

In total, our extension allows you to automatically create up to 9 versions of an image and serve the most suitable version regarding size and quality to your customer, depending on browser and display capabilities.

2. System Requirements & Installation

For the extension to work correctly, at least one of the supported conversion tools for WebP and AVIF conversion must be available on your server.
See sections → 2.3 - WebP Conversion Tools & 2.4 - AVIF Conversion Tools for further details.

For installing the extension, follow installation process as with any Magento Extension.

2.1. Extension Installation via composer

For installing the extension via composer, follow installation process as with any Magento Extension from Magento Marketplace: https://devdocs.magento.com/extensions/install/

2.2. Manual Extension Installation via FTP

For manual installation by FTP, please follow these steps:

Before Installing

1. We recommend you to duplicate your live store on a staging/test site and try installation on your staging/test site before deploying to your live store
2. Backup Magento files and the store database

Please Note: It's very important to backup all themes and extensions in Magento before installation, especially when you are working on a live server.
We strongly recommend you to do not skip this step.

Upload the Extension

1. Log into your hosting space via a FTP client (e. g. FileZilla, WinSCP, cuteFtp)
2. Create Folder: <magentoroot>/app/code/Jajuma/ImageOptimizerUltimate
3. Unzip extension package and upload files into: <magentoroot>/app/code/Jajuma/ImageOptimizerUltimate
4. Enter and run the following commands at the command line:

```
php bin/magento setup:upgrade
php bin/magento setup:static-content:deploy
```

2.3. WebP Conversion Tools

This Extension supports 3 different WebP conversion tools:

1. GD
2. cwebp +
3. Imagemagick

For using WebP images, at least one of these conversion tools must be available on your server.
GD should work out of-the-box on any Magento installation.
But the command line tools cwebp + Imagemagick might need additional installation or configuration steps to run correctly on your server.

Note:

In case of any issues with these tools or in case you are unsure how to get these tools up and running, please ask your hosting provider for assistance.

2.3.1. Check WebP Conversion Tools

Check from Backend

If you have the extension already installed you can check if the conversion tools are working on your server using the WebP Conversion Test Tool from Backend.
→ 4.4 - Test & Preview AVIF/WebP Conversion

Check from CLI

If you want to check which WebP conversion tools are available on your server from CLI / without having the extension installed, please follow these steps:

2.3.1.1. Check GD WebP support from CLI

Run following command on your CLI:

```
php -i | grep -E 'WebP Support|GD'
```

If GD is
- not supported / installed on your server: This will give you an empty result.
- supported / installed on your server: Output should be something like this:

```
GD Support => enabled
GD Version => bundled (2.1.0 compatible)
WebP Support => enabled
```

Note:

The last line of this output must be showing enabled.

2.3.1.2. Check cwebp from CLI

Run following command on your CLI see if cwebp is available as global command:

```
cwebp
```

If cwebp is
- not supported / installed on your server: This will give you an you an error message.
- supported / installed on your server: Output should be the man page for cwebp.

In case cwebp is not available as global command on your server, you can try these steps:

- Add cwebp to you server

a) Upload official cwebp lib from here https://developers.google.com/speed/webp/docs/precompiled
b) This lib is also bundled with our extension.
If you have the extension already installed you will find it in this path:
*[magentoroot]/app/code/Jajuma/*ImageOptimizerUltimate*/bin/cwebp or*
[magentoroot]/vendor/jajuma/module-imageoptimizerultimate/bin/cwebp

- Make sure the lib is executable

- Run the following command on your CLI:

```
</path/to/cwebp/lib/on/your_server>/cwebp
```

If cwebp
- can run on your server using this method: Output should be the man page for cwebp.
- cannot run on your server using this method: This will give you an error message / note about missing dependencies.

Please Note: For cwebp to work correctly with the extension:

➔ cwebp and all dependencies need to be installed correctly on your system
➔ cwebp binary needs to be executable (by webuser)

Note:

If needed, please check with your hosting provider to get cwepb up and running and to ensure these requirements are met.

2.3.1.3. Check Imagemagick WebP support from CLI

Run following command on your CLI to see if ImageMagick is available as global command:

```
convert
```

If Imagemagick is
- not supported / installed on your server: This will give you an error message.
- supported / installed on your server: Output should be the man page for Imagemagick.

If Imagemagick is available from any other custom command.

Note:

If needed, please check with your hosting provider to get Imagemagick up and running.

2.4. AVIF Conversion Tools

This Extension supports 2 different AVIF conversion tools:

1. cavif +
2. Imagemagick

For using AVIF images, at least one of these conversion tools must be available on your server.

cavif is included with the extension as bundled binary.
It should work on all servers, but the binary **needs to be executable by your webuser.**

Imagemagick might need additional installation or update to **at least version 7.0.10-25 or newer** to work correctly on your server.

> **Note:**
>
> **In case of any issues with these tools or in case you are unsure how to get these tools up and running, please ask your hosting provider for assistance.**

2.4.1. Check AVIF Conversion Tools

Check from Backend

If you have the extension already installed you can check if the conversion tools are working on your server using the AVIF Conversion Test Tool from Backend.
→ 4.4 - Test & Preview AVIF/WebP Conversion

Check from CLI

If you want to check which AVIF conversion tools on your server from CLI, please follow these steps:

2.4.1.1. Check cavif from CLI

Run following command on your CLI to see if AVIF conversion tool that is bundled with the extension as binary is working on your server*:*

[magentoroot]/app/code/Jajuma/ImageOptimizerUltimate/bin/./cavif
or
[magentoroot]/vendor/jajuma/module-imageoptimizerultimate/./bin/cavif

---

If cavif is

- not working on your server:
  This will give you an you an error message
  “command not found”.
- Working on your server:
  This will ask you to specify an image for conversion
  “Please specify image paths to convert”

For testing a manual image conversion to AVIF from your CLI using bundled binary, you can use these commands (Just replace <input.jpg> with image (path+) name on your server):

</path/to/cavif/lib/on/your_server>/./cavif <input.jpg>
or with conversion quality set as value between 0 and 100:
</path/to/cavif/lib/on/your_server>/./cavif --quality 60 <input.jpg>

Note:

For cavif to work correctly with the extension, cavif binary needs to be executable (by webuser)

2.4.1.2. Check Imagemagick AVIF support from CLI

Run following command on your CLI:

```
convert -v | grep 'Version'
```

If Imagemagick is
- not supported / installed on your server: This will give you an error message.
- supported / installed on your server: Output will show you the currently installed version similar to this:

```
Version: ImageMagick 7.0.10-55 Q16 x86_64 2021-01-04
```

Note:

Your ImageMagick needs to be at least version 7.0.10-25 or newer

(You can also try running this command: *convert -list format*
to get a list of all image formats supported by your Imagemagick.
But beware, this might give you false info. AVIF conversion might still work, even when not included in list of supported formats!)

For testing a manual image conversion to AVIF from your CLI using Imagemagick, you can use this command
(Just replace <input.jpg> with image (path+) name on your server):

```
convert -verbose <input.jpg> output.avif
```

Note:

If needed, please check with your hosting provider to get Imagemagick up and running with at least version 7.0.10-25 or newer.

3. Extension Scope

JaJuMa Ultimate Image Optimizer Extension supports conversion of .jpg and .png images into .avif and .webp files based on configuration as described below using one the supported conversion tools:

For WebP:

- GD
- cwebp (included as bundled binary)
- Imagemagick

For AVIF:

- cavif (bundled binary)
- Imagemagick (version 7.0.10-25 or newer)

All images and background images found in markup on all pages on your site, as well as images in Fotorama Media Gallery on Product Page are converted and delivered as AVIF/WebP images to visitors.
Using HTML5 <picture>-tag and browser AVIF/WebP support detection it is ensured that .avif / .webp files are only served to browsers that can show this image type.
For visitors using a web browser without .avif and/or .webp support, the extension automatically fall back to delivering the next best file type.

Additionally, the extension supports

- native lazy loading by optionally adding the *loading="lazy"* tag to converted images
- script based lazy loading by optionally lazy loading any image already converted to .webp
- excluding certain images from lazy loading via blacklist (e. g. images above-the-fold)

and

- adding High-Resolution images (double and / or triple resolution) for product images for better quality on so called “Retina”-displays with higher pixel density for AVIF, WebP as well as JPG/PNG images.
- Browsers will automatically download the correct resolution depending on display/monitor dpi value.

The extension integrates with Magento Media Management.
Means, .avif, .webp and High-Res version image files are created for the same size variants as for .jpg/.png files (for example thumbnails and small grid images as well as original size).
The extension comes with a separate Media Cache for optimized images (AVIF, WebP + High-Resolution images).
AVIF, WebP + High-Resolution images generated and stored in this Cache can be cleared at any time, e. g. to apply new conversion configs / quality level.

JaJuMa Ultimate Image Optimizer Extension is fully FPC (Full Page Cache) compatible.

As any Extension on Magento Marketplace, JaJuMa Ultimate Image Optimizer Extension is developed to work with Magento Default. While trying as far as feasible to avoid conflicts with, or breaking other custom functions, we cannot guarantee so for every imaginable customization out there.

Specifically, in case you have other customizations regarding how images are delivered to your customers (such as lazy loading, image swap on hover, images served from CDN...), the extension might still work fine, but it is also possible there might be conflicts preventing the extension from working as expected.
It totally depends on how your custom implementation is designed to work.

However, besides supporting 3 different conversion tools for WebP conversion and 2 different conversion tools for AVIF conversion, we also added some configuration options trying to have the extension work in as many scenarios as possible
(See → 4.5 - Advanced AVIF/WebP Conversion Configuration).

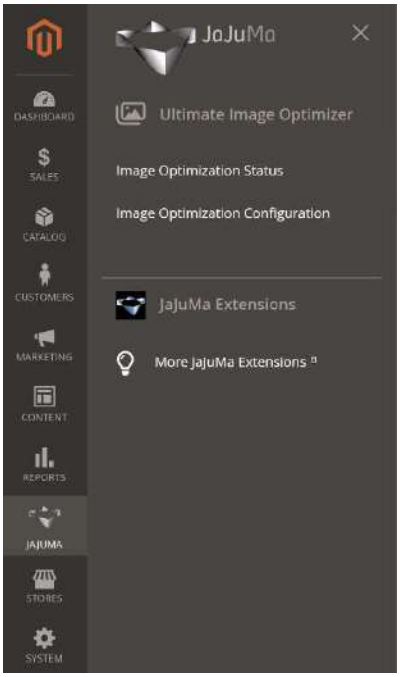
4. Configuration

In Magento Backend see

JaJuMa -> Ultimate Image Optimizer -> Image Optimization Status

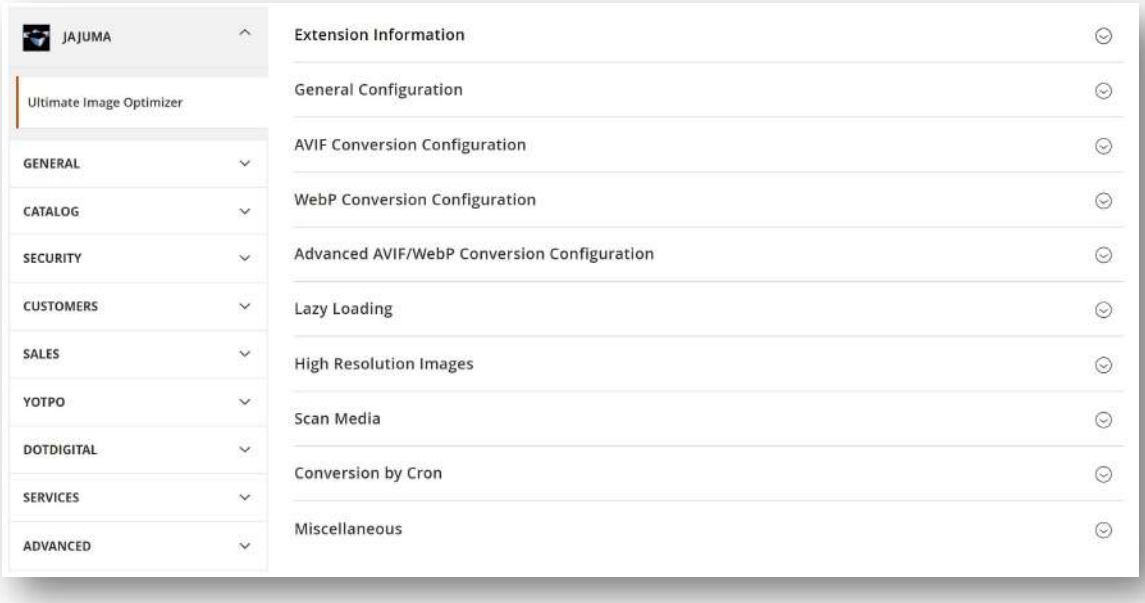
To see current image optimization status
(for details see section 6)

Or
JaJuMa -> Ultimate Image Optimizer -> Image Optimization Configuration

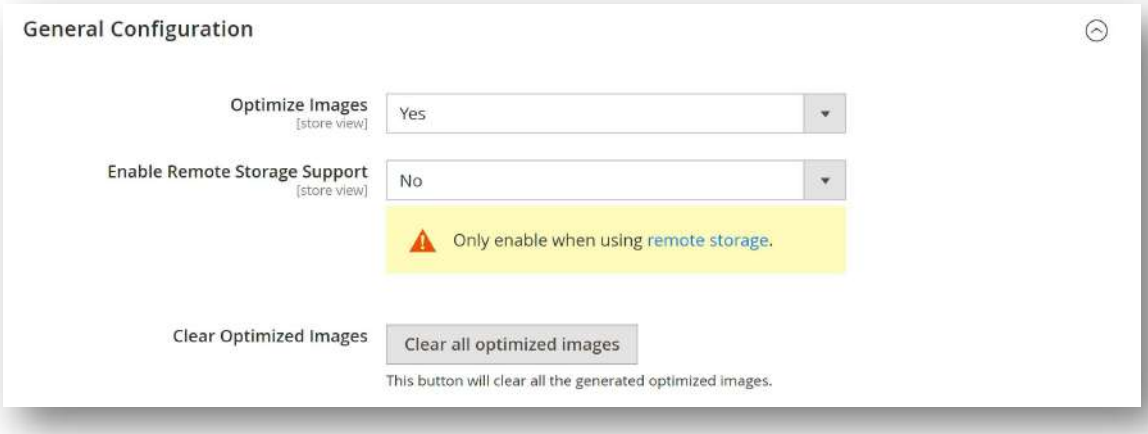


For JaJuMa Ultimate Image Optimizer configuration with 9 sections:

- General Configuration
- AVIF Conversion Configuration
- WebP Conversion Configuration
- Advanced AVIF/WebP Conversion Configuration
- Lazy Loading
- High Resolution Images
- Scan Media
- Conversion by Cron
- Miscellaneous



4.1. General Configurations



4.1.1. Optimize Images

To enable Ultimate Image Optimizer, select from Drop Down:

- Yes → Enable Extension
- No → Disable Extension

If enabled, the extension will convert existing images to WebP and AVIF, apply Lazy Loading / High-Resolution images as configured – see below for details.

4.1.2. Enable Remote Storage

To enable Remote Storage support , select from Drop Down:

- Yes → Enable Remote Storage support
- No → Disable Remote Storage support

For more details on using Remote Storage with Magento, please see: https://experienceleague.adobe.com/docs/commerce-operations/configuration-guide/storage/remote-storage/remote-storage-aws-s3.html.

4.1.3. Clear Optimized Images Cache

Click this button to clear all previously created images.
E. g. in case you want to change conversion settings to have AVIF/WebP images with higher or lower quality.
New WebP/High-Resolution images will be created automatically on the fly with following page requests, via Cron or manually by CLI command.
New AVIF images will be created automatically via Cron or manually by CLI command.
Note: Your original images will never be deleted.

Note:

When flushing Optimized Images Cache, please do not forget to also flush your Full Page Cache if applicable.

4.2. WebP Conversion Configurations

Please see below for configurations regarding conversion depending on conversion tool you want to use.
For the conversion to WebP to work, you need to make sure the conversion tool selected is setup and installed correctly on your server.
See below and section → 2.3 - WebP Conversion Tools for further details.
Please Note: For testing different conversion settings you may use Conversion Test Tool from Backend. → 4.4 - Test & Preview AVIF/WebP Conversion
For changing conversion tool / parameters also for previously created WebP images, just clear WebP Image Cache. This will delete generated WebP files.
By this, new image files will be created applying your new conversion configuration.
→ 4.1.3 - Clear Optimized Images Cache

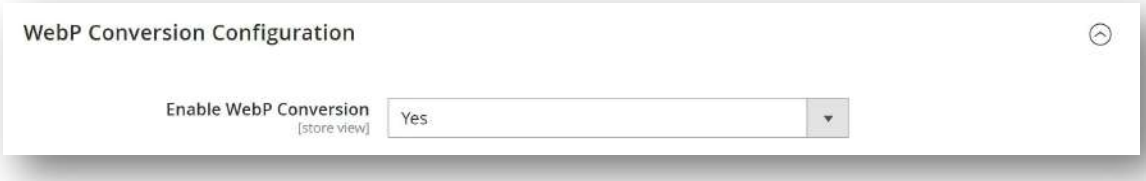
Note:

WebP images are created on-the-fly when an image is requested from frontend for the first time or by Cron / CLI. For having WebP images created by cron, please make sure to configure and enable Scan Media & Conversion by Cron – see sections 4.8 & 4.9.

Note:

Quality config values are not the same for different conversion tools.
E. g. Quality / compression factor "70" will deliver different results with cwebp compared to results with Imagemagick.
Please use the conversion test tool to find the most suitable quality value for each conversion tool used.

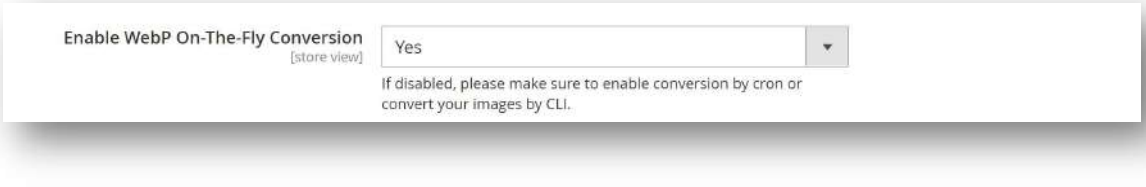
4.2.1. Enable WebP Conversion



For using WebP optimized images, please select from Drop Down:

- Yes → WebP images will be used if converted already
- No → WebP images will not be used

4.2.2. Enable On-The-Fly Conversion for WebP Images

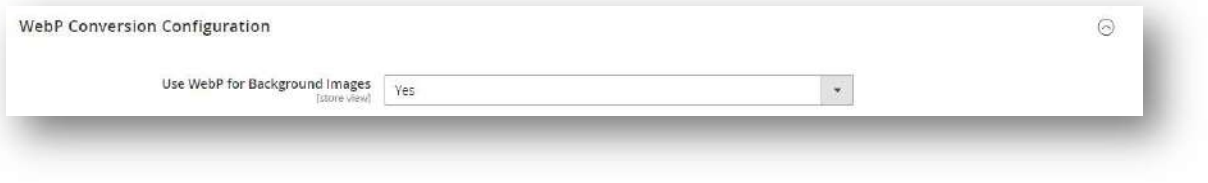


WebP images can be converted and added “On-The-Fly”, which means when they are requested from frontend for the first time.
For using On-The-Fly conversion for WebP images, please select from Drop Down:

- Yes → WebP images will be created on the fly
- No → WebP images will not be created on the fly

Note:
With On-The-Fly conversion disabled, please make sure to use conversion cron job (See sections 4.8 & 4.9) or convert your images via CLI (See section 5).

4.2.3. Use WebP for Background Images



For using WebP optimized images for inline background images,
please select from Drop Down:

- Yes → WebP versions for inline background images will be added
- No → WebP versions for inline background images will not be added

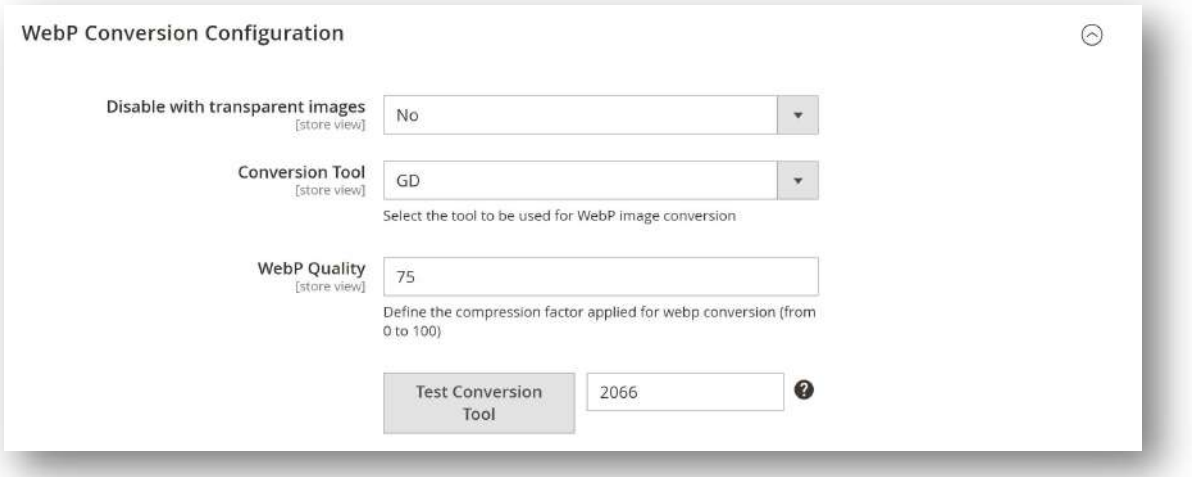
4.2.4. Disable WebP Conversion with transparent images

In case you see increased file sizes or quality issues with images that have transparency, you can disable WebP to be used for these images by selecting from Drop Down:

- Yes → Images with transparency will be excluded
- No → Images with transparency will be included

However, in most cases it should be fine to keep this as “No”.

4.2.5. Config WebP Conversion Using GD



For using GD as conversion tool, you just need to

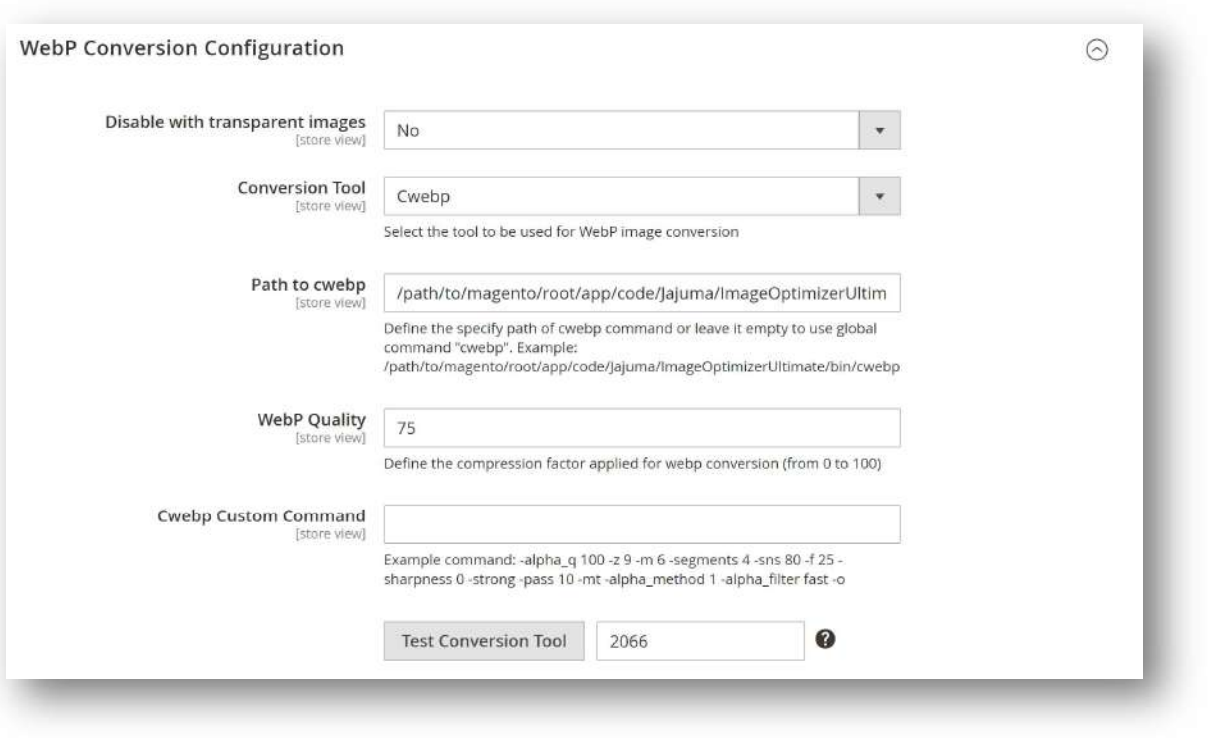
1. **Choose "GD" from Conversion Tool Drop-Down**

2. **WebP Quality:** Define the compression factor to be applied**.**
   (GD does not provide any further conversion options.)

3. **Test WebP Conversion using GD**

→ See 4.4 - Test & Preview AVIF/WebP Conversion

4.2.6. Config WebP Conversion Using cwebp



For using “Cwebp” as conversion tool, you just need to

1. **Choose “Cwebp” from Conversion Tool Drop-Down**
2. **Path to cwebp**

   Configure correct path to working cwebp executable.

   Please note: For cwebp to work correctly

   ➔ cwebp and all dependencies need to be installed correctly on your system
   ➔ cwebp binary needs to be executable (by webuser)

   Option 1

   The cwebp binary is included with the Extension.
   In order to use the bundled binary please use following path:

   *[magentoroot]/app/code/Jajuma/*ImageOptimizerUltimate*/bin/./cwebp or [magentoroot]/vendor/jajuma/module-imageoptimizerultimate/bin/./cwebp*

   Please double-check the binary is executable by your web-user and make sure all required dependencies are installed on your system.

Option 2

You can also use any cwebp installation already present on your system or download from https://developers.google.com/speed/webp/download and install in any other path.

Just add the path to the executable binary.

Option 3

In case cwebp is executable via global command as “cwebp”,
you can just leave this configuration empty.

5. **Conversion Parameters:**

Base Mode

WebP Quality:
Define the compression factor to be applied.

Conversion will be performed using following pre-configured command:

```
-q <WebP Quality Config> -alpha_q 100 -z 9 -m 6 -segments 4 -sns 80 -f 25 -sharpness 0 -strong -pass 10 -mt -alpha_method 1 -alpha_filter fast
```

Advanced Mode

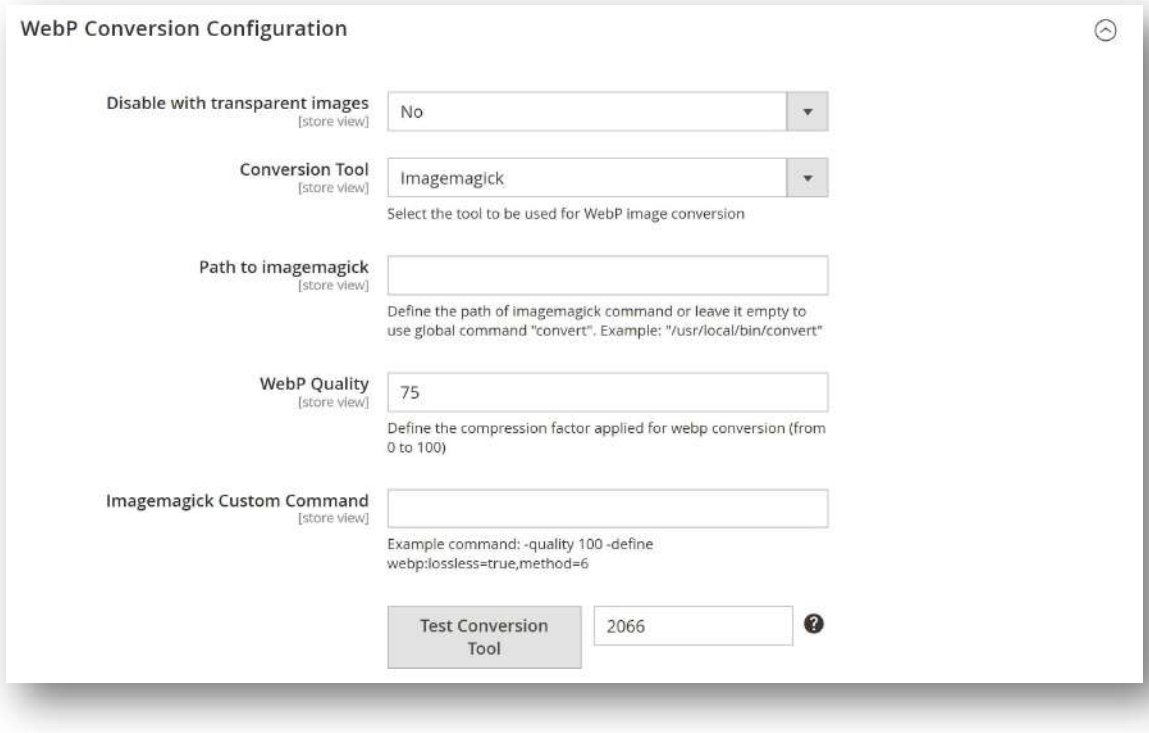
Cwebp Custom Command:
Define any custom cwebp command with parameters that suit your needs.
For a detailed description of parameters available, please see
https://developers.google.com/speed/webp/docs/cwebp

Note: Leave this configuration empty to use Base Mode

5. **Test WebP Conversion using cwebp**

→ See 4.4 - Test & Preview AVIF/WebP Conversion

4.2.7. Config WebP Conversion Using Imagemagick



For using “Imagemagick” as conversion tool, you just need to

1. **Choose “Imagemagick” from Conversion Tool Drop-Down**

2. **Configure correct path to Imagemagick executable**

   Option 1

   In case Imagemagick is available through global command "*convert*", you can just leave this configuration empty.

   Option 2

   Or define the custom path of imagemagick command on your system.

   Example: */usr/local/bin/convert*

3. **Conversion Parameters:**

Base Mode

WebP Quality:
Define the compression factor to be applied.
Conversion will be performed using following pre-configured command:

```
-quality <WebP Quality Config> -define
webp:lossless=false,method=6,segments=4,sns-strength=80,auto-
filter=true,filter-sharpness=0,filter-strength=25,filter-type=1,alpha-
compression=1,alpha-filtering=fast,alpha-quality=100
```

Advanced Mode

Imagemagick Custom Command:
Define any custom Imagemagick command with parameters that suit your needs
For a detailed description of parameters available, please see https://www.imagemagick.org/script/webp.php

Note: Leave this configuration empty to use Base Mode

4. **Test WebP Conversion using Imagemagick**

→ See 4.4 - Test & Preview AVIF/WebP Conversion

4.3. AVIF Conversion Configurations

Note:

For using AVIF images, please enable Scan Media & Conversion by Cron and make sure the cron jobs are running correctly – see sections 4.7 & 4.8.
AVIF will only be used for an image after the image has been converted to AVIF via Cron / CLI or Backend.

Note:

Encoding of AVIF is pretty slow by design of the image codec.
So please be patient for AVIF images to show up in Frontend and for conversion to complete. This process takes some time and slow conversion does not mean the extension is not working!

Note:

Due to the heavy processing needed for AVIF conversion, the conversion is running multi-threaded when using cavif. This means, the more CPU and resources your server can provide, the faster the conversion will be.
However, to avoid your server becomes overloaded and your site being slowed down by conversion, please make sure to
- limit number of threads used for conversion
- run conversion with low priority and
- have a server load limit configured
(See sections 4.3.1 & 4.8.)

Note:

Quality config values are not the same for different conversion tools.
E. g. Quality / compression factor "70" will deliver different results with cavif compared to results with Imagemagick.
Please use the conversion test tool to find the most suitable quality value for each conversion tool used.

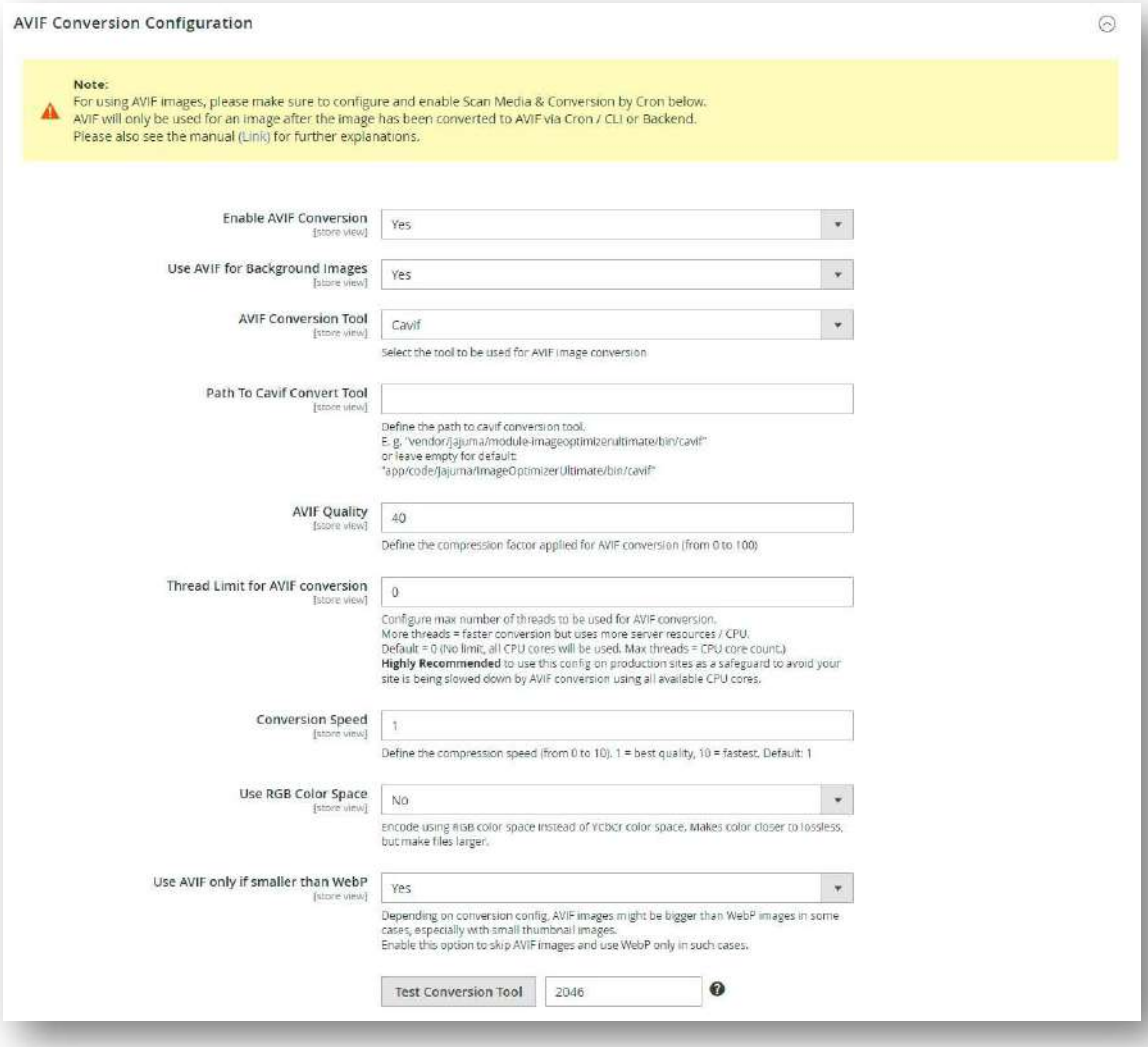
Please see below for configurations regarding conversion depending on conversion tool you want to use.

For the conversion to AVIF to work, you need to make sure the conversion tool selected is setup and installed correctly on your server.
See below and section → 2.4 - AVIF Conversion Tools for further details.

For testing different conversion settings, you may use Conversion Test Tool from Backend.
→ 4.4 - Test & Preview AVIF/WebP Conversion

For changing conversion tool / parameters also for previously created AVIF images, just clear Optimized Images Cache. This will delete all generated image files.
By this, new image files will be created applying your new conversion configuration.
→ 4.1.3 - Clear Optimized Images Cache

4.3.1. Config AVIF Conversion Using cavif



For using “cavif” as conversion tool, you just need to configure:

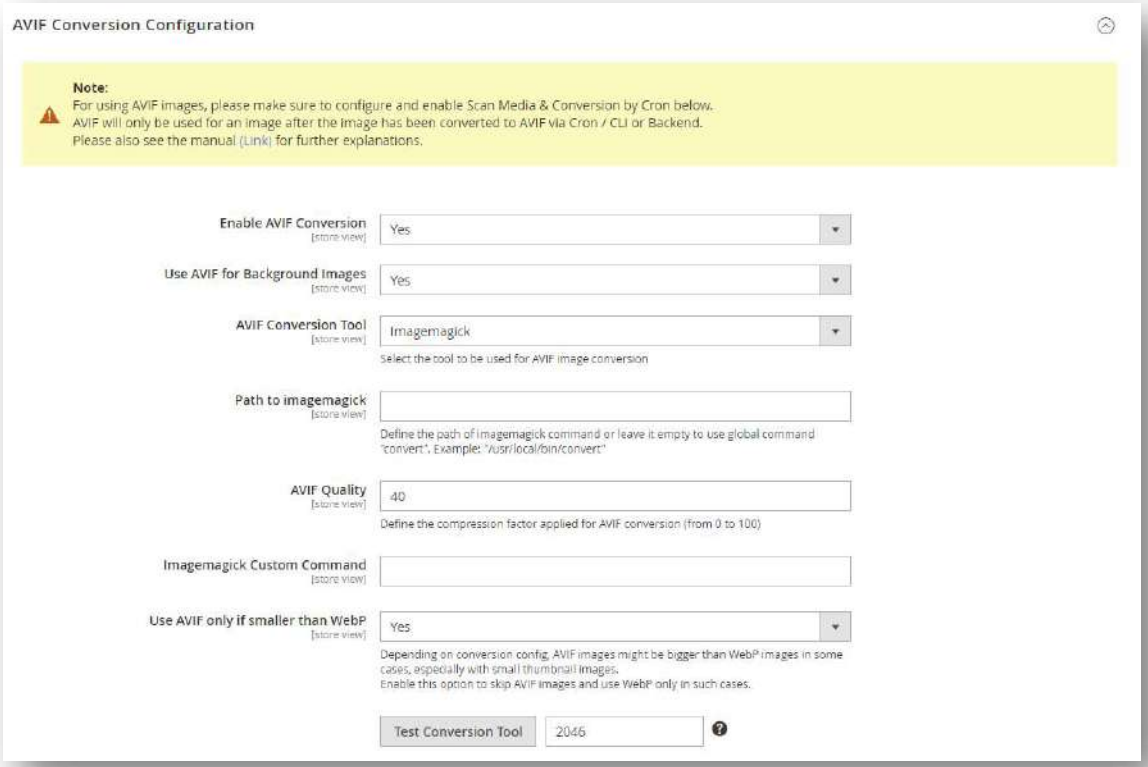
1. **Enable AVIF Conversion:**
   Choose “Yes” to enable AVIF Conversion
2. **Use AVIF for Background Images**
   Add AVIF optimized versions for inline background images
   (Also see 7 - Background Images)
3. **AVIF Conversion Tool:**
   Choose “Cavif”
4. **Path To Cavif Convert Tool:**
   Define the path to cavif conversion tool.
   E. g. "vendor/jajuma/module-imageoptimizerultimate/bin/cavif"

or leave empty for default:
"app/code/Jajuma/ImageOptimizerUltimate/bin/cavif"

5. **AVIF Quality:**
   Define the compression factor applied for AVIF conversion as a value from 0 to 100
6. **Thread Limit for AVIF conversion**
   The cavif conversion tool supports multi-threaded conversion to speed up the optimization process.
   In order to avoid all of your CPU resources are used for image optimization, you can configure a max number of threads to be used for AVIF conversion.
   More threads = faster conversion but uses more server resources / CPU.
   Default = 0 (No limit, all CPU cores will be used. Max threads = CPU core count.)
   Highly Recommended to use this config on production sites as a safeguard to avoid your site is being slowed down by AVIF conversion using all available CPU cores.
7. **Conversion Speed:**
   Encoding speed between 1 (best quality result, but slowest) and 10 (fastest, but lowest quality result). For best results, using value 1 and patience is recommended. Encoding of AVIF is pretty slow by design of the image codec.
6. **Use RGB Color Space:**
   Encode using RGB color space instead of YCbCr color space. Makes color closer to lossless, but make files larger.
7. **Use AVIF only if smaller than WebP**
   Depending on conversion config, AVIF images might be bigger than WebP images in some cases, especially with small thumbnail images.
   Enable this option to skip AVIF images and use WebP only in such cases.
6. **Test AVIF Conversion using cavif**

→ See 4.4 - Test & Preview AVIF/WebP Conversion

4.3.2. Config AVIF Conversion Using Imagemagick



For using “Imagemagick” as conversion tool, you just need to

1. **Enable AVIF Conversion:**
   Choose “Yes” to enable AVIF Conversion
2. **Use AVIF for Background Images**
   Add AVIF optimized versions for inline background images
   (Also see 7 - Background Images)
3. **AVIF Conversion Tool:**
   Choose “Imagemagick”
4. **Path to imagemagick:**
   Define the path of imagemagick command or leave it empty to use global command "convert". Example: "/usr/local/bin/convert"
5. **AVIF Quality:**
   Define the compression factor applied for AVIF conversion as a value from 0 to 100
6. **Imagemagick Custom Command**
   Please leave empty for now. At this time, Imagemagick does not support advanced conversion paramters.

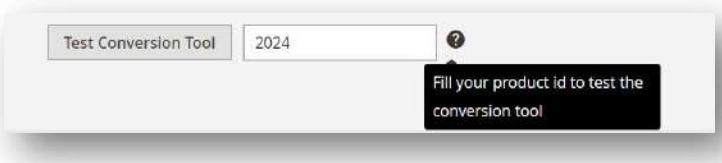
7. **Use AVIF only if smaller than WebP**
   Depending on conversion config, AVIF images might be bigger than WebP images in some cases, especially with small thumbnail images.
   Enable this option to skip AVIF images and use WebP only in such cases.
5. **Test AVIF Conversion using Imagemagick**

→ See 4.4 - Test & Preview AVIF/WebP Conversion

4.4. Test & Preview AVIF/WebP Conversion

For testing the AVIF/WebP conversion and preview the conversion result please follow these steps:

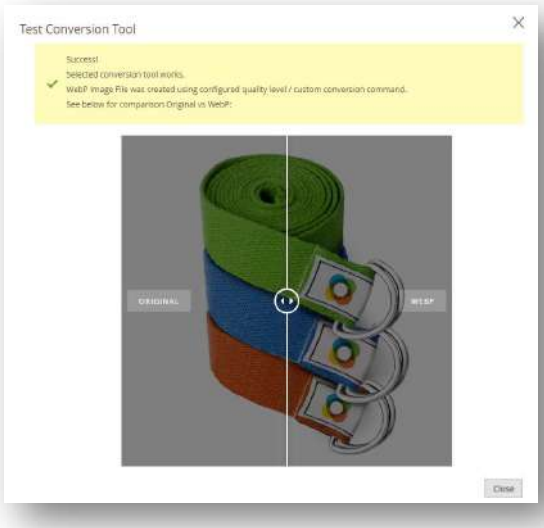
1. Complete Conversion Configuration as described in sections 4.2 & 4.3
2. Input the Product ID (not SKU) into the text field next to "**Test Conversion Tool**" - Button

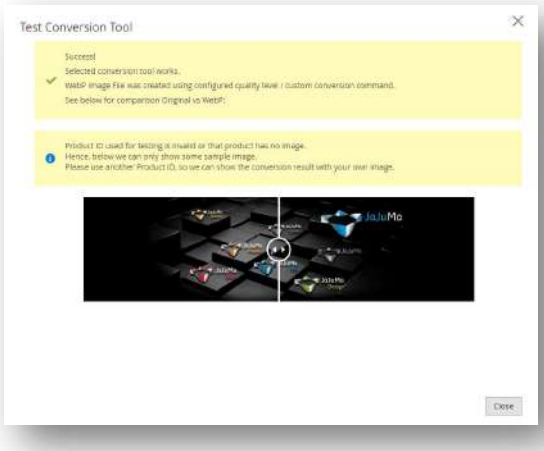


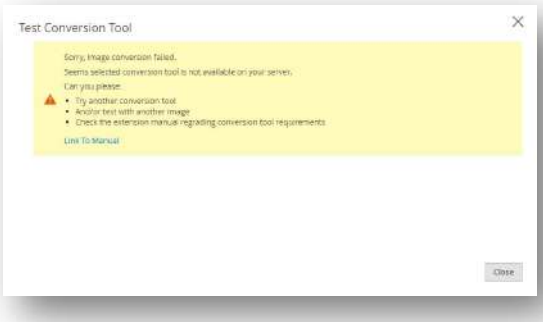
3. Click "**Test Conversion Tool**" - Button
4. See PopUp with conversion result:

4.1. **Success:** In case the conversion is successful:
You will see the result as a side-by-side comparison between original image (left) and created AVIF/WebP image (right).
You can move the vertical slider to reveal more of the original image or the AVIF/WebP image as you like, to get the perfect impression regarding quality.

4.2. **Fail:** In case the conversion fails the PopUp will give you some hint on what went wrong and how to fix it.




Test Conversion Tool
Success!
Selected conversion tool works.
See below for comparison Original vs WebP!
Close


Test Conversion Tool
Sorry, image conversion failed.
Seems selected conversion tool is not available on your server.
Can you please
Try another conversion tool
Anchor test with another image
Check the extension manual regarding conversion tool requirements
Link To Manual
Close


Test Conversion Tool
Success!
Selected conversion tool works.
See below for comparison Original vs WebP!
JaJuMa
Uuups...
something went wrong.
Please use a Browser
that supports WebP.
(e. g. Chrome, Opera, Firefox v.65+...)
Otherwise you can not
see conversion result.
Close

4.5. Advanced AVIF/WebP Conversion Configuration

Note:

For most sites following configs are not needed.
We recommend to use only if you understand what you are doing.

4.5.1. Conversion Blacklist

With Blacklist config it is possible to exclude certain images on your site from WebP conversion. You can add any string or RegEx as comma separated list. Any <img>-tag matching with any of the given strings/RegEx will be excluded from conversion.

Example 1: Exclude Logo named *logo.png*

logo\.jpg

Example 2: Exclude Logo named *logo.png* AND all images through WYSIWYG editor

logo\.jpg,wysiwyg

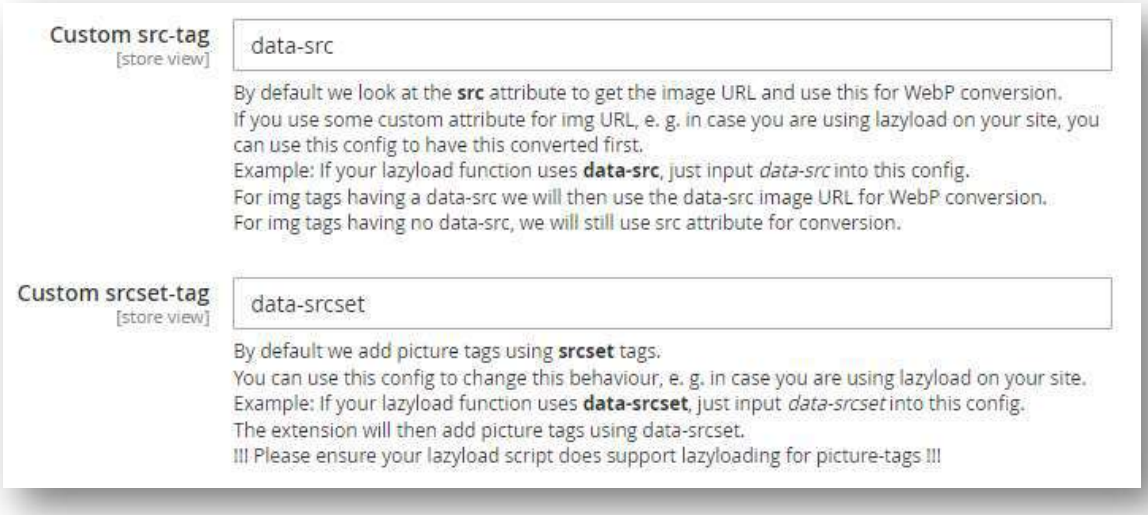
Example 3: Exclude images with a certain css class, e. g. product-image-photo

product-image-photo

Note:

img-tags with data-nowebp="true" or data-noavif="true" and images in media/captcha folder are excluded from conversion by default.

4.5.2. Custom src-/srcset-tags



With this config it is possible to make this extension work with some custom cases where images are loaded via some lazyloading function.

By default we look at the **src** attribute to get the image URL and use this for WebP conversion.
If you use some custom attribute for img URL, e. g. in case you are using lazyload on your site, you can use this config to have this converted first.
Example: If your lazyload function uses **data-src**, just input *data-src* into this config.
For img tags having a data-src we will then use the data-src image URL for WebP conversion.
For img tags having no data-src, we will still use src attribute for conversion.

By default, we add picture tags using **srcset** tags.
You can use this config to change this behaviour, e. g. in case you are using lazyload on your site.
Example: If your lazyload function uses **data-srcset**, just input data-srcset into this config.
The extension will then add picture tags using data-srcset.

Note:

!!! Please ensure your lazy load script does support lazy loading for picture-tags !!!
Or use lazy loading added by this extension instead.

4.6. Lazy Loading

JaJuMa Ultimate Image Optimizer supports Native Lazy Loading as well as JS based Lazy Loading. Please see below how to use and configure both options:

4.6.1. Native Lazy Loading

To enable Native Lazy Loading, select from Drop Down “Enable Native Lazy Loading”:

- Yes → Enable Native Lazy Loading
- No → Disable Native Lazy Loading

If enabled, the extension will add the *loading="lazy"* tag to converted images.

4.6.1.1. Native Lazy Loading Blacklist

With Native Lazy Loading Blacklist config it is possible to exclude certain images from native lazy loading. This is recommended e. g. for above-the-fold images.
You can add any string or RegEx as comma separated list.
Any <img>-tag matching with any of the given strings/RegEx will be excluded from native lazy loading.

Example 1: Exclude Logo named *logo.png*

```
logo\.jpg
```

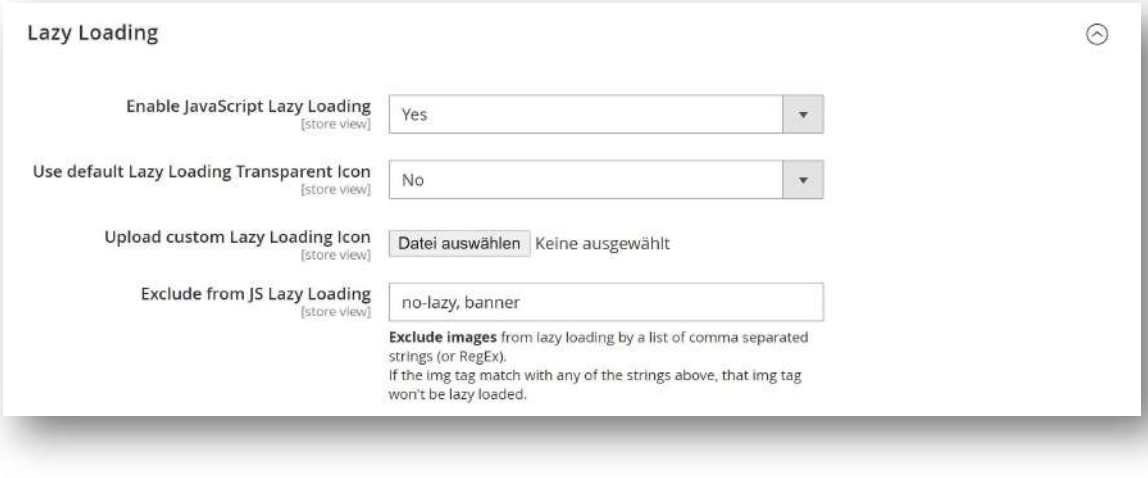
Example 2: Exclude Logo named *logo.png* AND all images through WYSIWYG editor

```
logo\.jpg,wysiwyg
```

Example 3: Exclude images with a certain css class, e. g. banner-image

```
banner-image
```

4.6.2. JavaScript Lazy Loading



To enable JavaScript Lazy Loading, select from Drop Down “Enable JavaScript Lazy Loading”:

- Yes → Enable JavaScript Lazy Loading
- No → Disable JavaScript Lazy Loading

If enabled, the extension will add the *data-src / data-srcset* attributes for converted images.

4.6.2.1. Configure Lazy Loading Icon

If “Use default Lazy Loading Transparent Icon” = Yes:
A transparent / invisible icon will be used for lazy loading.

If “Use default Lazy Loading Transparent Icon” = No:
A custom Lazy Loading Icon can be uploaded.

4.6.2.2. Exclude from JS Lazy Loading

With Exclude from JS Lazy Loading config it is possible to exclude certain images from JS lazy loading. This is recommended e. g. for above-the-fold images.
You can add any string or RegEx as comma separated list.
Any <img>-tag matching with any of the given strings/RegEx will be excluded from JS lazy loading.

Example 1: Exclude Logo named *logo.png*

```
logo\.jpg
```

Example 2: Exclude Logo named *logo.png* AND all images through WYSIWYG editor

```
logo\.jpg,wysiwyg
```

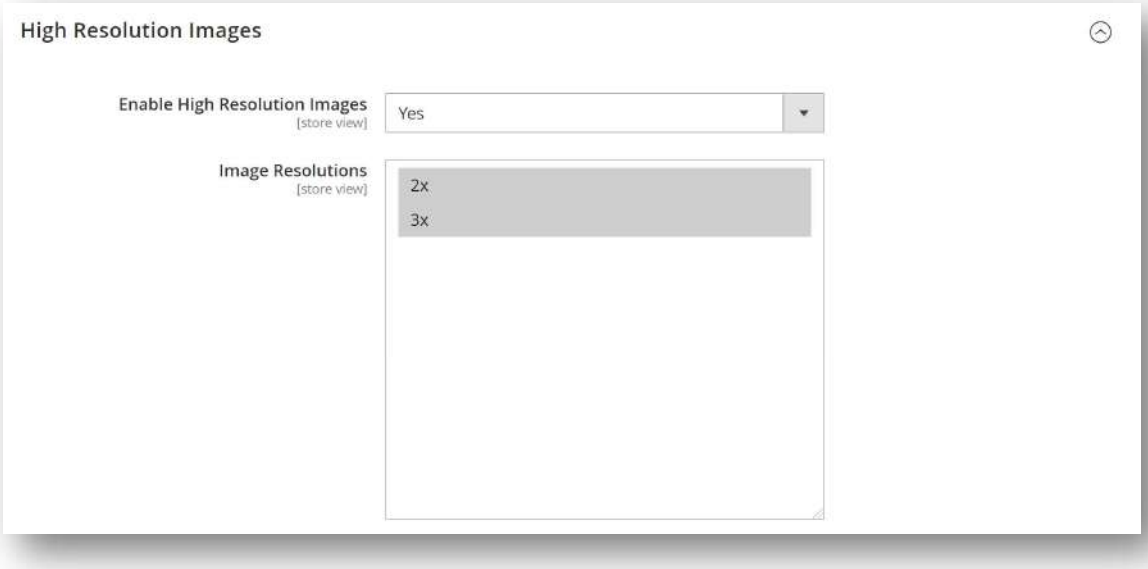
Example 3: Exclude images with a certain css class, e. g. banner-image

```
banner-image
```

4.7. High-Resolution Images

Note:

!!! High-Resolution images are only added for product images and not to Fotorama Media Gallery !!!.
Please also make sure to upload product images in a high enough resolution.
Your original/uploaded image has to be to at least 2x or 3x as big as your product images in product grids.
E. g. if your category grid is showing products with 250 x 250 px,
the original image needs to be at least:
500 x 500 px for adding 2x images and
750 x 750 px for adding 3x images



To enable High-Resolution Images, select from Drop Down "**Enable High-Resolution Images**":

- Yes → Enable High-Resolution Images
- No → Disable High-Resolution Images

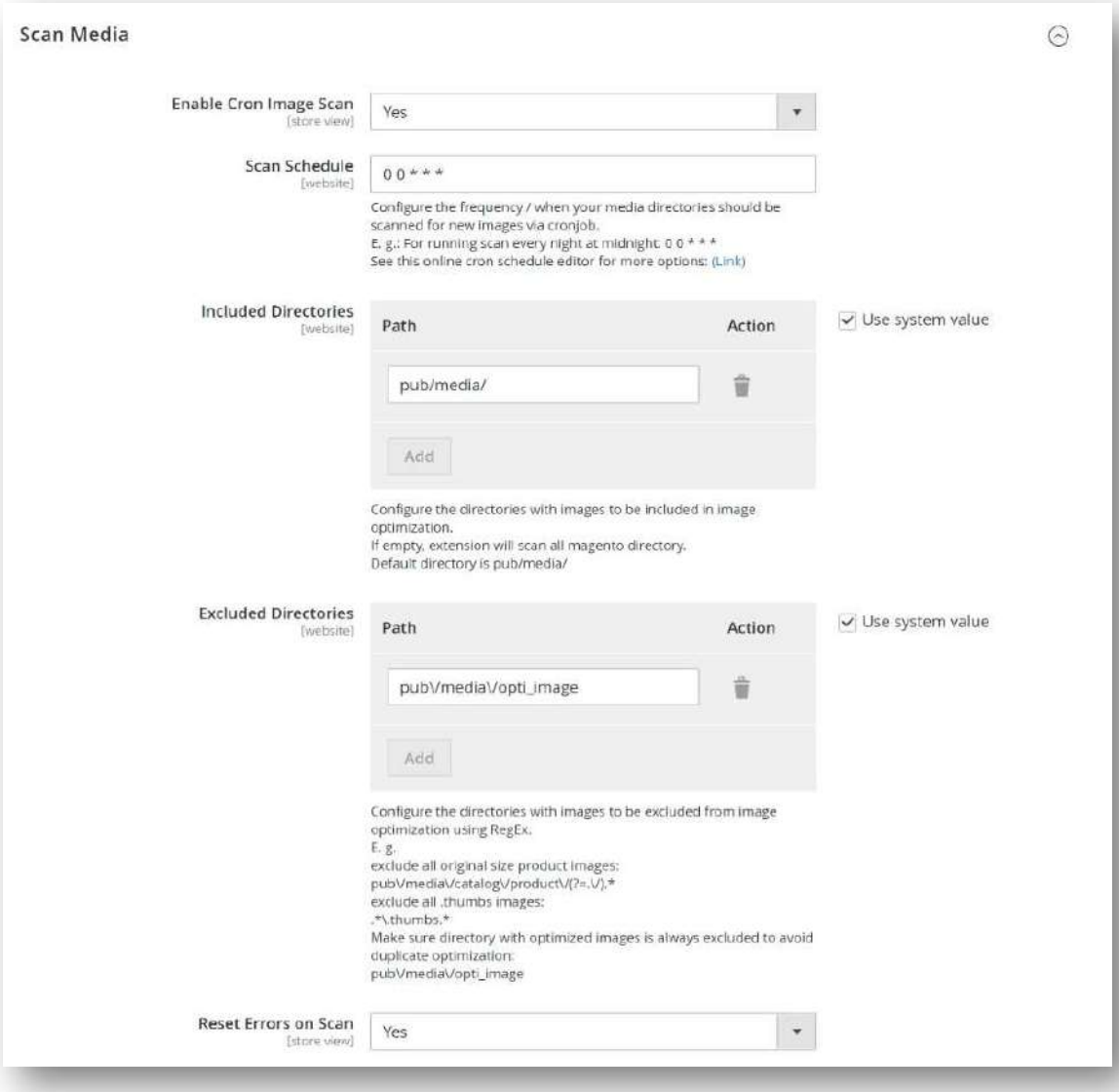
If enabled, the extension will add High-Resolution Images for both, WebP as well as original JPG/PNG images.

Please use the **Image Resolution** Multi-Select input to select if

- 2x (= double resolution) OR
- 3x (= triple resolution) OR
- 2x AND 3x

versions should be added to your product images.

4.8. Scan Media Cron Job



The Scan Media function is for discovering all your images that should be optimized as well as optimized image versions already created by the extension.
The data collected by Scan Media is used as base for conversion cron job (see 4.9 - Conversion Cron Job) to find images pending conversions/optimizations and for Image Optimization Status info (see 6 - Image Optimization Status).

Please make sure Scan Media is run frequently for the optimization process to work with up-to-date data including new images and Image Optimization Status showing accurate data.

Scan Media can be run as cron job by schedule or manually from Image Optimization Status screen (see 6 - Image Optimization Status).

Please see below how to configure the Scan Media function.

Enable Image Scan:

To enable Scan Media by cron schedule, select from Drop Down “Enable Image Scan”:

- Yes → Scan Media will be executed follow the cron config below
- No → Scan Media will not be executed by cron

Scan Schedule:

Configure time and frequency when Scan Media process should be executed as cron expression. E. g. Once a day at midnight:

```
0 0 * * *
```

(See e. g. this online cron editor for more options)

Included Directories:

Configure the directories with images to be included in image optimization.
If empty, extension will scan all Magento directory.
Default directory is

```
pub/media/
```

Excluded Directories:

Configure the directories with images to be excluded from image optimization using RegEx.

E. g. exclude all original size product images:

```
pub\/media\/catalog\/product\/(?=.\/).*
```

exclude all .thumbs images:

```
.*\.thumbs.*
```

Make sure directory with optimized images is always excluded to avoid duplicate optimization:

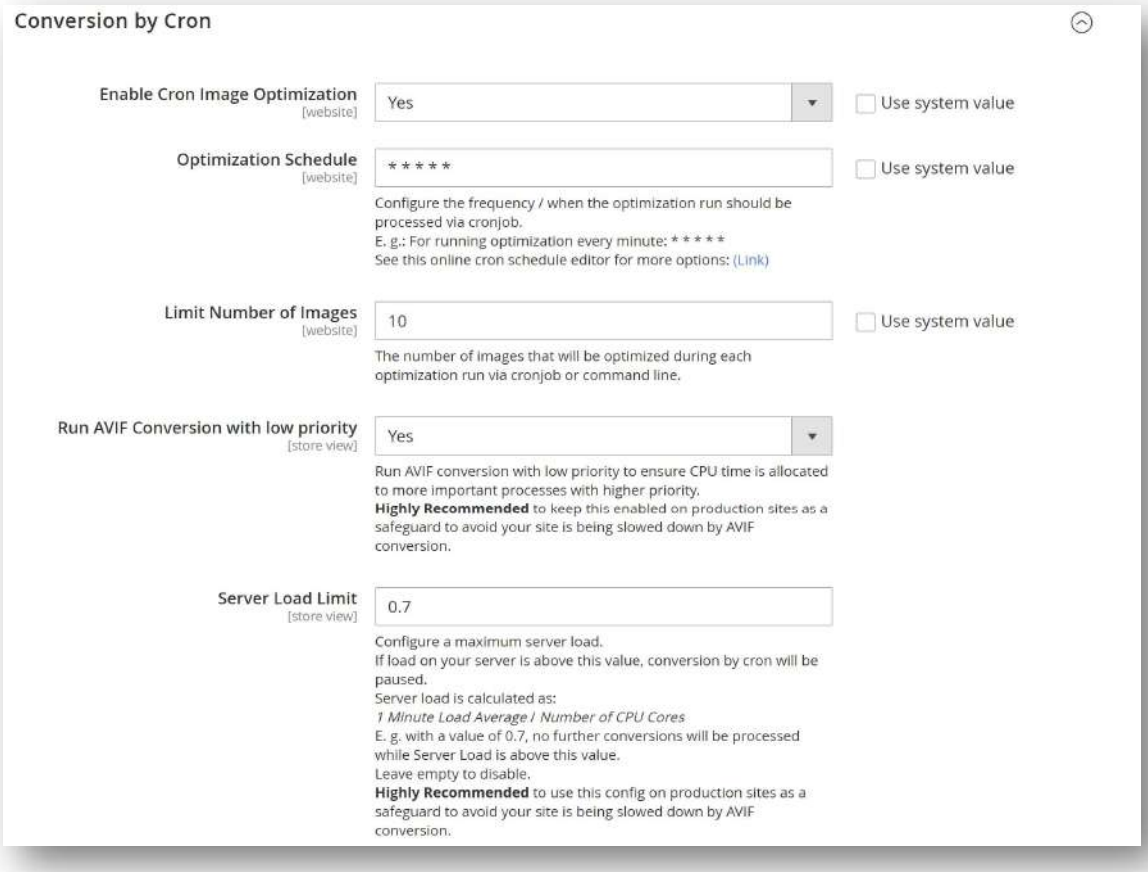
```
pub\/media\/opti_image
```

Reset Errors on Scan:

To enable Reset Errors on Scan Media, select from Drop Down “Reset Errors on Scan”:

- Yes → Image optimization status “Error” will be reset to “Pending” on Scan. Conversion Cron Job will try to convert these images again.
- No → Image optimization status “Error” will not be reset to “Pending” on Scan. Conversion Cron Job will not try to convert these images again.

4.9. Conversion Cron Job



The Conversion Cron Job checks for pending image optimizations based on data collected by 4.8 - Scan Media Cron Job and processes any conversion/optimization found missing.

The number of pending images processed with each Conversion Cron Job run can be configured. Conversion cron job can be run as cron job by schedule or manually from Image Optimization Status screen (see 6 - Image Optimization Status).
Please see below how to configure the Conversion Cron Job.

Enable Conversion Cron Job:

To enable Conversion by cron schedule, select from Drop Down “Enable Cron Image Optimization”:

- Yes → Scan Media will be executed follow the cron config below
- No → Scan Media will not be executed by cron

Optimization Schedule:

Configure time and frequency when Conversion process should be executed as cron expression. E. g. every minute:

```
* * * * *
```

(See e. g. this online cron editor for more options)

Limit Number of Images:

The number of images that will be optimized during each optimization run via cronjob (or as default for conversion via command line).

Run AVIF Conversion with low priority:

Select “Yes”, to run AVIF conversion with low priority to ensure CPU time is allocated to more important processes with higher priority if needed.

Select “No”, to run AVIF conversion with normal priority.

Note:

It is highly recommended to keep this enabled on production sites as a safeguard to avoid your site is being slowed down by AVIF conversion.
(This option only works if supported by your Hosting.
If not, conversion will still work when disabled)

Server Load Limit:

Configured value defines a maximum server load for processing further conversions/optimizations.
If load on your server is above this value, conversion/optimization by cron (and also CLI) will be paused.

Server load is calculated as:

```
1 Minute Load Average / Number of CPU Cores
```

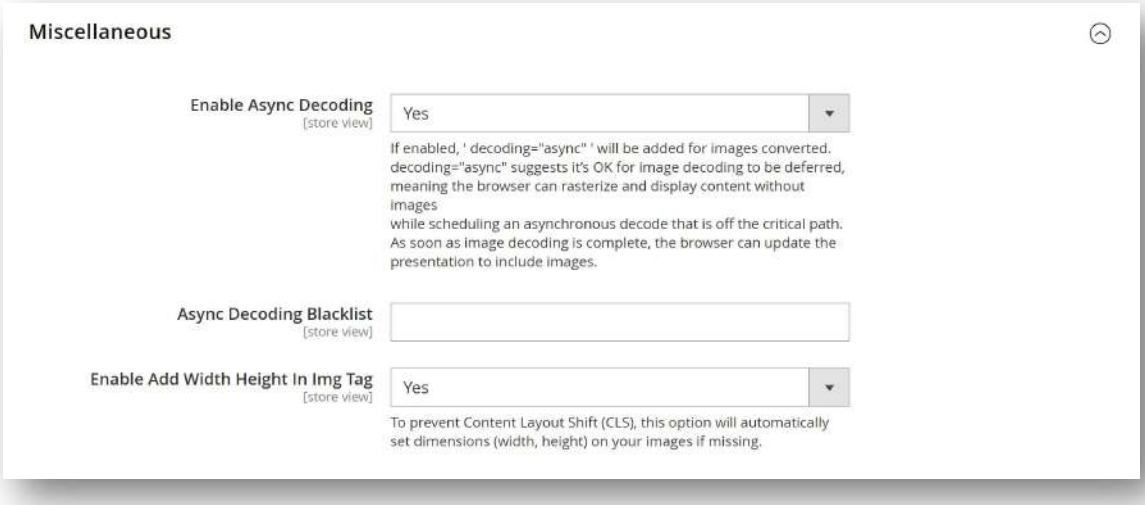
E. g. with a value of 0.7, no further conversions/optimizations will be started while Server Load is above this value.
Or in other words:
With a value of 0.7, further conversions/optimizations will only be started if your server had at least ~30% free resources during the last minute.

Leave empty to disable.

Note:

It is highly recommended to use this config on production sites as a safeguard to avoid your site is being slowed down by AVIF conversion.
(This option only works if supported by your Hosting.
If not, conversion will still work, but without limit)

4.10. Miscellaneous



4.10.1. Async Image Decoding

To enable async image decoding, select from Drop Down “Enable Async Decoding”:

- Yes → Enable Async Image Decoding
- No → Disable Async Image Decoding

If enabled, ' decoding="async" ' will be added for images converted.
decoding="async" suggests it’s OK for image decoding to be deferred, meaning the browser can rasterize and display content without images while scheduling an asynchronous decode that is off the critical path.

As soon as image decoding is complete, the browser can update the presentation to include images.

Async Decoding Blacklist:

With Async Decoding Blacklist config it is possible to exclude certain images from async decoding. This is recommended e. g. for above-the-fold images and other non-lazy loaded images.
You can add any string or RegEx as comma separated list.
Any <img>-tag matching with any of the given strings/RegEx will not be decoded async.

Example 1: Exclude Logo named *logo.png*

```
logo\.jpg
```

Example 2: Exclude Logo named *logo.png* AND all images through WYSIWYG editor

```
logo\.jpg,wysiwyg
```

Example 3: Exclude images with a certain css class, e. g. banner-image

```
banner-image
```

4.10.2. Add Width/Height In Img Tag / Auto Set Image Dimensions

To prevent Content Layout Shift (CLS), every image should have dimensions set (width & height). In case some images on your site are missing this, you can add dimensions automatically by using this function.

To enable set missing image dimensions automatically, select from Drop Down “Add Width Height In Img Tag”:

- Yes → Enable set missing image dimensions
- No → Disable set missing image dimensions

5. Command Line Interface / CLI

The media scan as well as optimization process for pending image conversions/optimizations can be also started by command line interface (CLI).

For running the media scan from CLI, please run following command:

```
php bin/magento jajuma:imageopti:scan
```

For processing pending AVIF/WebP conversions or pending High-Res/Retina versions, please run following commands from your command line in your Magento base directory.

For running optimization with limit of pending images as configured in Conversion Cron Job configuration:

```
php bin/magento jajuma:imageopti:run
```

For running optimization for a certain number of pending images, e. g. 100 images:

```
php bin/magento jajuma:imageopti:run --limit 100
```

For running optimization for all pending images:

```
php bin/magento jajuma:imageopti:run --limit all
```

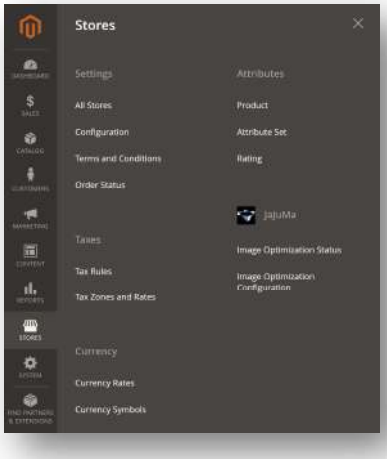
Note:

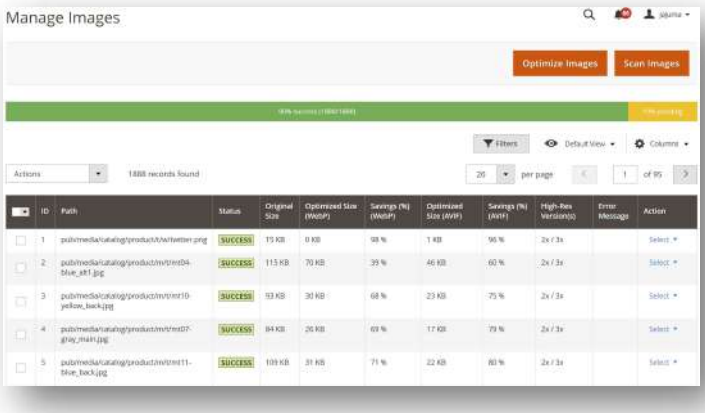
Optimize Images and Conversion by Cron have to be enabled for using command line interface.

6. Image Optimization Status

For the Image Optimization Status as well as some image optimization actions, see in Magento Backend

Stores -> JaJuMa -> Image Optimization Status





6.1. Manage Images Actions

Image Optimization Status screen provides access to several image optimization actions from backend:

Optimize Images:

By clicking this button, you can start one conversion run with number of images limit as configured for conversion cron job.
Conversion progress and result will be shown in a popup.

Scan Images:

By clicking this button, you can run the scan media function for discovering all your images that should be optimized as well as optimized image versions already created by the extension.
After completing the scan, info shown in Image Optimization Status grid will be updated and collected data will be used as base for conversion cron job.

Convert Now:

The "*Convert Now*" action via "*Select*" in Action column in Image Optimization Status grid allows you to process conversion/optimization for a certain image.

Reset Image Optimization:

The "*Reset Image Optimization*" action via Action Drop-Down or "*Select*" in Action column in Image Optimization Status grid allows you to reset optimization for a certain image from Image Optimization Status. If the source image still exists, it will be optimized again via next conversion cron job run or manual run.

6.2. Image Optimization Status Grid

The Image Optimization Status Grid gives an overview over all images found during last Media Scan and their optimization status with options to search/filter and following information:

- **Path:**
  Path to original images found by Media Scan that is used as source for optimized versions
- **Original Size:**
  Size of original image in kilobyte as reference for calculating the savings with optimized versions
- **Status:** Optimization Status
  - **Success:** Image has been optimized
  - **Pending:** Image has not been converted/optimized yet
  - **Error:** An error happened when trying to process the conversion/optimization
- **Optimized Size (WebP):**
  Size of .webp version for original image in kilobyte as reference for calculating the savings with .webp version
- **Savings (%) (WebP):**
  Savings in file size of .webp version compared to original image in percent.
- **Optimized Size (AVIF):**
  Size of .avif version for original image in kilobyte as reference for calculating the savings with .avif version
- **Savings (%) (AVIF):**
  Savings in file size of .avif version compared to original image in percent.
- **High-Res Version(s):**
  High-Resolution versions created for original image.

7. Background Images

The extension supports optimizing background images by adding AVIF and/or WebP versions for background images found in page source similar to these:

```
style="background-image: url(...);"
```

or

```
<style... >
 .css-class {
     background-image: url(...);
 }
</style>
```

To enable background image optimization, please see sections
4.2 + 4.3

Please note: CSS background images in css files are not converted automatically. However, JaJuMa Ultimate Image Optimizer extension enables you to manually use .avif/.webp images as CSS backgrounds on your site easily.

7.1. WebP CSS Background Images

In browsers that do not support.webp, our extension will automatically add a CSS class ".no-webp" to the <body> tag, e. g. like this:

```
<body class="cms-home cms-index-index page-layout-1column no-webp">
```

This enables you to use browser compatible .webp images as backgrounds.
If you have a background image added by CSS on your site like this:

```
.your .css .class {
    background: url(../path/to/background.png);
}
```

You can simply change it to this to use your manually converted .webp image as background and keep your .png/.jpg as fallback for old browsers:

```
.your .css .class {
    background: url(../path/to/background.webp);
}
body.no-webp .your .css .class {
    background: url(../path/to/background.png);
}
```

7.2. AVIF CSS Background Images

In browsers that do not support.webp, our extension will automatically add a CSS class ".avif" to the <html> tag, e. g. like this:

```
<html class="... avif ...">
```

This enables you to use browser compatible .avif images as backgrounds.
If you have a background image added by CSS on your site like this:

```
.your .css .class {
    background: url(../path/to/background.png);
}
```

You can simply change it to this to use your manually converted .avif image as background and keep your .png/.jpg as fallback for old browsers:

```
.your .css .class {
    background: url(../path/to/background.webp);
}
html.avif .your .css .class {
    background: url(../path/to/background.png);
}
```

8. FAQ & Further Recommendations

8.1. General

Conversion results, in terms of .avif/webp file size, saving compared to .jpg/.png as well as quality, highly depend on your existing images. Hence it is not possible to give a "one size fits all" recommendation or provide a configuration that delivers 'perfect' results for every use case.

With the preconfigured conversion parameters in Base Mode we tried our best to provide you with a configuration that provides good results out-of-the-box.
In case the results with this configuration are not satisfying to you, please feel free to play around with the different parameters in Advanced Mode to find a configuration that better suits your needs. It's as easy as saving your custom command and deleting Magento Media Cache to see the results.

Note:

Quality config values are not the same for different conversion tools.
E. g. Quality / compression factor "70" will deliver different results with cwebp compared to results with Imagemagick.
Please use the conversion test tool to find the most suitable quality value for each conversion tool used.

This table maps quality settings for JPEG to the respective AVIF and WebP quality settings:

JPEG quality	50	60	70	80
AVIF quality	48	51	56	64
WebP quality	55	64	72	82

(source)

Note: Please use these values as a rule of thumb only. Results will differ depending on conversion tool used as well as your images.

8.2. WebP

8.2.1. Which WebP Conversion Tool Should I Use?

We recommend to use cwebp, the official conversion tool developed by Google.
It is the best choice regarding quality, conversion speed and also provides the most conversion parameters.
However, the price for this is the “challenge” to get cwebp up and running on your server, which might not be possible in every case, e. g. with some shared hosts.

In that case we recommend to use GD, it should work out-of-the-box on any Magento System without further setup requirements and is easy to use due to having just one option to be configured. But in most cases GD still provides pretty good results.

Imagemagick is also pretty good regarding conversion speed, however it is said to produce slightly less good quality than the other 2 tools.

8.2.2. What “WebP Quality” Should I Use?

This value defines the compression factor applied for conversion - Default is 75.

In case of lossy compression (default), a small value produces a smaller file with lower quality. Best quality is achieved by using a value of 100.

In case of lossless compression (specified by the -lossless option), a small factor enables faster compression speed, but produces a larger file. Maximum compression is achieved by using a value of 100.

8.2.3. How About All These Other Paramters?

Please see Official Documentation for

cwebp: https://developers.google.com/speed/webp/docs/cwebp

Imagemagick: https://www.imagemagick.org/script/webp.php

8.3. AVIF

Please be aware: AVIF is a pretty new and cutting-edge technology. The standard behind this image format is not yet finalized.

We tested the conversion with the tools supported by our extension carefully and successfully on different environments. However, unexpected issues with AVIF conversion may still happen.

But with many Big Players working on developing and backing this new image format, a wider adoption and robust support and improved conversion tools for this new image format can be expected soon.

8.3.1. AVIF Conversion Is Very Slow – What Can I Do?

Yes, the AVIF conversion requires heavy processing to achieve the small file sizes, which makes it slow. The conversion is already processed multi-threaded when using cavif as conversion tool, which means it will use as much CPU time as available on your server already. There is nothing else we can do about this at the moment, except being patient or provide processing power to speed up the conversion process.

Note:
Please be aware that this multi-threaded conversion may affect the performance and availability of your site if not used carefully.
The extension provides configurations to control the resources used for AVIF conversion by limiting the number of threads used for AVIF conversion, by running the conversion with low priority and to pause the conversion when the server load is too.
Make sure to use these configurations as appropriate on your server.

8.3.2. Will This Extension Be Compatible With Other Browsers When AVIF Support Is Added?

Yes, our extension is designed to let each Browser download the most suitable version of an image regarding format and resolution as supported by Browser/Display used.
This will also work when Browsers add support for AVIF.

9. Support

Please feel free to contact JaJuMa support team via support@jajuma.de.
In case any additional information is required. We'd be more than happy to assist in setting up the extension.