Google PageSpeed Optimize Manual

Getting Started

Welcome to the **Google PageSpeed Optimizer Documentation**. Whether you are a new or an advanced user, you can find some useful information here.

First of all, we recommend you check the following link:

- How to install extension

How to install the extension

1. Backup your store's database and web directory.
2. Login to your server's SSH console and navigate to the root directory of the Magento 2 store.
3. Copy the installation instructions from the page My Downloadable Products into the SSH console and press ENTER.
4. Run the command below to enable the extension:

```
php -f bin/magento module:enable Mirasvit_Core Mirasvit_Optimize Mirasv
```

5. Run the command below to install the extension:

```
php -f bin/magento setup:upgrade
```

6. Run the command below to clean the cache:

```
php -f bin/magento cache:clean
```

7. Deploy static view files:

```
rm -rf pub/static/frontend/*; rm -rf pub/static/backend/*; rm -rf var/v
php -f bin/magento setup:static-content:deploy
```

8. Make sure that the native Magento cronjob is configured and working correctly. The extension performs all tasks in cron.

9. The module Mirasvit_OptimizeImage requires server-side software to work properly. You can find information about required packages and their installation here.

Google PageSpeed Optimizer settings

The settings of the Google PageSpeed Optimizer is grouped into the following sections:

- **Insight** - section for checking the pagespeed of your website.
- **JavaScript Optimization** - settings for JavaScript optimization.
- **CSS Optimization** - settings for CSS optimization.
- **Image Optimization** - settings for images optimization.
- **HTML Optimization** - settings for HTML optimization.
- **Other Optimization** - additional optimization settings.

Insight

This configuration section was designed for checking the page speed rate of your website for desktop and mobile versions.
The **Run GoogleSpeed Test** is used to run a quick speed test for your website's home page.

- **Configurations**
  - **URLs for monitoring** - specify your website's URLs that need to be monitored when determining page speed performance, placing each URL in a new line.

JavaScript Optimization

- **Enable JavaScript optimization** - enable/disable JavaScript optimization for your website.
  Run `bin/magento setup:static-content:deploy` after saving the configurations.
- **Defer YouTube videos** - toggles the lazy loading of YouTube videos.
  If enabled, the extension will replace the YouTube videos with placeholder images linked to the original video on YouTube. Clicking the link will load the video player. This option reduces the amount of resources needed for the initial page load.
- **Merge JavaScript files** - enable/disable merge JS files. This config depends on the default Magento "Merge JavaScript files" config that is available only in developer mode (Stores - Configuration - Advanced - Developer - JavaScript Settings).
  If enabled, the extension will merge JavaScript files into one file to reduce the number of HTTP requests made by a browser during the page loading process.
  JavaScript bundling will not work when this option is enabled.
- **Minify JavaScript** - enable/disable minification JS files. This config depends on the default Magento "Minify JavaScript files" config that is available only in developer mode (Stores - Configuration - Advanced - Developer - JavaScript Settings).
  If enabled, the extension combines JavaScript files into bundles when required for particular page types, excluding unnecessary bundle files from the page depending on the page type.
  Run `bin/magento setup:static-content:deploy` after saving the configurations.
- **Move JavaScript To Page Bottom** - enable/disable moving JS files to the bottom of the page. **How it works:** Before the HTML page is sent to the browser, our module checks for all external JS resources (<script> tags) in the HTML code. Then the extension moves all identified resources to the bottom of the page, except for those who match any of the exceptions from the **Ignore URL List** config.
  Clear the Magento cache after this option has changed.

- **Ignore URL List** - the list of pages (URL patterns) where JS will not move. Or a list of JS files (patterns for SCRIPT tags) that should not be moved.
  For example:

  - to exclude pages like /category/page/*, the pattern should be: category/page

    - to exclude JS files, there should be an added pattern for the JS file name (if the JS file was added as an external resource, with the "src" attribute inside the SCRIPT tag), or part of the content inside the SCRIPT tag (if JS code was added directly to the page)

  Each pattern on the new line.

  Note

  If default Magento JS merging is enabled, some JS files will not be excluded from moving to the bottom of the page if merged into a single file.
  Before adding patterns for JS file names, please check to see if these files are present on the page.
  You can do this by searching in the source of the page. To open the source of your page in the browser, please press Ctrl + u (Windows) or ? + Option + u (Mac).
- **Lazyload iframes** - loads the content embedded within iframes only if the iframes are visible in the browser's viewport.
- **Iframe lazyload exceptions** - Add the list of iframes excluded from lazy-loading here. Each pattern has to start from the new line.

  Note

  This field is available only if the **Lazyload iframes** option is enabled

CSS Optimization

- **Merge CSS Files** - enable/disable merge CSS files. This config depends on the default Magento "Merge CSS files" config that is available only in developer mode (Stores - Configuration - Advanced - Developer - CSS Settings).\ If enabled, the extension combines additional CSS files into a single file to reduce the number of HTTP requests.
  Run `bin/magento setup:static-content:deploy` after saving the configurations.
- **Minify CSS Files** - enable/disable minification CSS files. This config depends on the default Magento "Minify CSS files" config that is available only in developer mode (Stores - Configuration - Advanced - Developer - CSS Settings).
  If enabled, the extension reduces the size of CSS files.
  Run `bin/magento setup:static-content:deploy` after saving the configurations.
- **Move CSS To Page Bottom** - enable/disable moving CSS files to the bottom of the page. **How it works:** Before the HTML page is sent to the browser, our module checks for all external CSS resources (<link> tags) in the HTML code. Then the extension moves all identified resources to the bottom of the page, except for those who match any of the exceptions from the "Do not move CSS that contains" config.\
- **Do not move CSS that contains** - list of patterns for the page's link tags, which should not be moved to the bottom of the page. (Available only if the **Move CSS To Page Bottom** option is enabled).
- **Preload CSS exception** - if enabled, the extension will add link tags for CSS resources that match any exception from the **Do not move CSS that contains** field, with the rel="preload" attribute to force the browser to load those resources before the DOM is displayed in the browser.
  (Available only if the **Move CSS To Page Bottom** option is enabled).
- **Add styles to HTML from CSS resourses** - the list of CSS resources that are added directly to the HTML document.
  Each resource has to start from the new line.
- **Defer Google Fonts** - enable or disable a loading defer of Google Fonts. If enabled, the page will display the default fonts and then apply Google Fonts after they have been loaded.
- **Additional CSS Styles** - additional CSS styles that are added directly to the HTML document.

Note

This field is functionally identical to the **Additional CSS Styles** field in **Stores** -> **Configuration** -> **Mirasvit Extensions** -> **Developer** -> **CSS Settings**.

Image Optimization

The extension uses a set of tools in optimizing or converting images. Images get optimized in the pub/media folder.
To check the status of tools, please run the command: `bin/magento mirasvit:optimize-image:validate`

Note

Suppose you receive a message about certain libraries that have not yet been installed. In that case, you can install them by running the command: `sudo yum install <library_name>` (Centos) or `sudo apt-get install <library_name>` (Ubuntu).
They can be installed all together, for example, by running: `sudo apt-get install jpegoptim optipng gifsicle webp imagemagick`
After this, you can wait for a while (until optimization has been completed by cron) or else run commands for the optimization:
`bin/magento mirasvit:optimize-image:optimize` - Run images optimization process
`bin/magento mirasvit:optimize-image:webp` - Create a copy of the images in .webp format
The images optimization process doesn't require any additional commands in the crontab on your server. It runs on Magento's default crontab.
Should you receive a message stating that a package is not available at any point during the package installation, you can upload the relevant package directly to the server and perform a manual installation as a workaround.
Please refer to these articles for manual installation instructions:

- webp installation (source file)
- jpegoptim installation
- gifsicle source file (the installation process works the same as with the other packages)

- **Image Quality level** - choose a percentage level of the picture compression (100% - without quality loss, 70% - acceptable to display on the site). This is applicable only for JPEG and GIF, and only compresses the original image (each new thread of compression will delete the previous one automatically). Compression can be done by cron or CLI command.
- **Image optimization strategy** - divided in two options: **Filesystem** - scans 1000 files per cronjob in the pub/media folder; **Webpages** - pictures are added to the optimization queue from the visited pages. There may be fewer pictures optimized in the queue, but it features the most important pictures from the site instead of everything in a row. Optimization is processed by cron.
- **Use WebP images** - enable or disable, using webp image types instead of default image types. Clear the Magento cache after this option has been changed.
- **Image Lazy Load** - allows the loading of images only if they appear in the browser's viewport. Affects images added with IMG tags and background images added through CSS classes (including background images added in the page builder). Does not work with background images added inside the "style" attribute.

- **Enable Lazy Load** - enable or disable lazy loading of images.
  Clear the Magento cache after this option has been changed.
- **Preload first N images** - the number of images which should be fully loaded when the page is still loading.
- **Preload first N background images** - the number of background images which should be fully loaded when the page is still loading. Only affects backgound images added from the page builder.
- **Add 'fetchpriority="high"' attribute to preloaded images** - If enabled, the extension will add attribute `fetchpriority="high"` to images, preloaded according to the config **Preload first N images**.
- **Do not use a lazy load for images** - list of images for which lazyload should not be used. Each pattern is in the new line.
- **Lazyload background images by CSS classes** - The list of CSS classes, related to background images that should be lazyloaded.
  Each name in the new line.

  Note

  Styles set for the class **should not** contain instructions for the size of the container to prevent possible layout shift.
  Examples:
  Good \

```
.some-class-image {
    backgroud-image: url(...);
}
.some-class-size {
    height: 300px;
}
```

  Bad \

```
.some-class {
    backgroud-image: url(...);
    height: 300px;
}
```

- **Responsive Images** - allows you to generate responsive images for mobile and desktop site versions.
  Resized images will automatically replace the original images once you flush the cache.
  WebP images for resized images will be generated by cron.
  The resized images can be loaded lazily as well.
  - **Generate responsive images** - generates resized images.

    Note

    Please make sure to save the configuration by clicking on **Save Config** before clicking this button.
  - **Cleanup responsive images** - removes all the previously-generated resized images, even if they have been removed from the configuration.
  - **Config** - configure the resized images.
    - **File name** - the file name pattern for the images that should be resized.
    - **Desktop width** - the width of the resized image on the desktop site version. If empty, the original image will be used.
    - **Mobile width** - the width of the resized image on the mobile site version. If empty, resized images will not be generated for this image.

Note
The module changes the height of resized images proportionally to their width.

HTML Optimization

- **Minify HTML** - enable or disable HTML document minification. If enabled, the extension reduces the size of the loading HTML file.
  Run `bin/magento setup:static-content:deploy` after saving the configurations.
- **Preload recourses** - preloads key resources.
  - To preload fonts that are hosted on your own site, add the font path starting with the slash after the domain extension, including query parameters.
    Example:
    `/pub/static/version1600338479/frontend/Magento/luma/en_US/fonts/Luma-Icons.woff2`
    The static content version will be resolved automatically.
  - To preconnect to resources with 3rd party origins, add the base URL of the origin with the `preconnect::` suffix.
    This will instruct the browser to connect to the specified origin at the early stage of the page loading, which can decrease the time needed to fetch resources from that origin.
    Example: `preconnect::https://maxcdn.bootstrapcdn.com/`

Other Optimization

- **Sign Static Files** - if enabled, browsers will update cached versions of static files once they are updated on the server. This prevents the issue with browsers serving up old static resources from arising.
- **Asynchronous indexing** - helps you avoid a decrease in store performance when intensive sales on a storefront occur at the same time that Magento is performing intensive order processing. When this option is enabled, orders will be placed in temporary storage and moved in bulk to the Order Management grid without any collisions.
- **Use Flat Catalog Category, Use Flat Catalog Product** - when enabled, a flat catalog will create new tables on the fly, where each row contains all the necessary data about a product or category. These options help shorten queries to the database to get product or category data.

  Note

  Magento does not recommend the usage of a flat catalog as mentioned in their documentation.

Google PageSpeed Optimizer settings for Magento Cloud

Since Magento Cloud projects have writing restrictions, to configure the extension in regards to the static content, the extension settings should be placed inside the **[root_store_path]/app/etc/config.php** file.

The config, most typical for the majority of Magento stores should look like this:

```
'system' => [
    'default' => [
        'dev' => [
            'template' => [
                'minify_html' => 1   // minify HTML
            ],
            'js' => [
                'merge_files' => 0,  // merge JS files
                'minify_files' => 1, // minify JS files
            ],
            'css' => [
                'merge_css_files' => 1, // merge CSS files
                'minify_files' => 1,    // minify CSS files
            ]
        ]
    ],
    'mst_optimize' => [
        'optimize_js' => [
            'enabled' => 1,   // enable JS optimizations, also enables advanced
            'minify_js' => 1, // minify JS files
            'merge_js' => 0   // merge JS files
        ],
        'optimize_css' => [
            'merge_css' => 1,  // merge CSS files
            'minify_css' => 1  // minify CSS files
        ],
        'optimize_html' => [
            'minify_html' => 1 // minify HTML
        ]
    ],
]
```

Where the `default/dev` part of the `system` config array is the default Magento settings and the `mst_optimize` part is our extension settings.

This config should be placed inside the first-level array in the **[root_store_path]/app/etc/config.php** file.

As our extension extends from some of Magento's default functionality, the default settings and settings of our extension should be changed accordingly.
For example, `default/dev/template/minify_html` should be changed together with `mst_optimize/optimize_html/minify_html`.

In the above config, comments are added to explain which configs are related and should be changed correspondingly.
Remove these comments before adding the config to the **[root_store_path]/app/etc/config.php** file.

Note

If the file **[root_store_path]/app/etc/config.php** already contains some of these settings, then these settings should be changed according to what optimization settings should be enabled/disabled in the store.
Settings should be changed in both the `default/dev` and the `mst_optimize` section of the configuration array.

All other extension settings can be safely changed from the admin panel of the store.

Command Line Interface

Usage: `php -f bin/magento [options]`

- Run a Google PageSpeed test for the given page:

  ```
  bin/magento mirasvit:optimize-insight:pagespeed https://example.com/
  ```

- Validate the software required for image optimization:

  ```
  bin/magento mirasvit:optimize-image:validate
  ```

- Run the image optimization process and generate webp images:

  ```
  bin/magento mirasvit:optimize-image:optimize
  ```

  This command has next options:

  - `--image` - optimize images according to the **Image Quality level** which is set in the configurations
  - `--webp` - generate webp images with the compression according to the **Image Quality level**

  Note

  Both options are optional. If the command executed without any option the extension will consider both options enabled - images will be optimized and webp images will be generated (if enabled in the configurations)

  The extension also runs image optimizations and webp images generation processes on a cron basis

- Run the restore command to restore original images:

  ```
  bin/magento mirasvit:optimize-image:restore
  ```

  Note
  The extension will remove all optimized and webp images

How to upgrade extension

To upgrade the extension, follow these steps:

1. Backup your store's database and web directory.
2. Login to your server's SSH console and navigate to the root directory of the Magento 2 store.
3. Run the command below to update the current extension with all dependencies:

   ```
   composer require mirasvit/module-optimize:* --update-with-dependencies
   ```

Note

In some cases, the command above is not applicable; neither is it possible to update just the current module, nor need to upgrade all the Mirasvit modules in a bundle. In this case, the command above will be of no effect.

Run instead `composer update mirasvit/*` command. It will update all the Mirasvit modules installed in your store.

4. Run the command below to enable the extension:

```
php -f bin/magento module:enable Mirasvit_Core Mirasvit_Optimize Mirasv
```

5. Run the command below to install updates:

```
php -f bin/magento setup:upgrade
```

6. Run the command below to clean the cache:

```
php -f bin/magento cache:clean
```

7. Deploy static view files:

```
rm -rf pub/static/frontend/*; rm -rf pub/static/backend/*; rm -rf var/v
php -f bin/magento setup:static-content:deploy
```

Disabling the Extension

Temporarily Disable

To temporarily disable the extension please follow the following steps:

1. Login to your server's SSH console and navigate to the root directory of the Magento 2 store.
2. Run the command below to disable the extension:

```
php -f bin/magento module:disable Mirasvit_Optimize Mirasvit_ImageLazyL
```

3. Log in to the Magento backend and refresh the store's cache (if enabled).

Removing the Extension

To uninstall the extension, please follow these steps:

1. Login to your server's SSH console and navigate to the root directory of the Magento 2 store.
2. Run the command below to remove the extension:

```
composer remove mirasvit/module-optimize
```

3. Log in to the Magento backend to refresh the store cache (if enabled).

Change Log

2.1.5

(2024-11-25)

Improvements

- Image Lazy Loading improved

2.1.4

(2024-11-19)

Improvements

- Built-in exceptions for WebP images removed

2.1.3

(2024-10-29)

Fixed

- Prevent reloading lazyloaded iframes after they scrolled out from the viewport

---

2.1.2

(2024-10-28)

Fixed

- Fixed the issue with lazyloading iframes (since 2.1.0)

2.1.1

(2024-10-21)

Fixed

- Fixed the issue with error 'Undefined variable' in image optimization cron

---

2.1.0

(2024-10-21)

Improvements

- Lazyload for background images by CSS classes
- Lazyload for background images added in the page builder
- Browsers' native lazyload
- JavaScript lazyload library removed
- Replace background images with WebP images

2.0.18

(2024-10-11)

Fixed

- Fixed the conflict with lazyload from Rokanthemes_PageBuilder

---

2.0.17

(2024-10-08)

Improvements

- Ability to set high fetch priority for preloaded images (lazyload feature)

2.0.16

(2024-09-16)

Improvements

- Cleanup generated images after original images were deleted (Product, Category, CMS, Catalog Images cache)

2.0.15

(2024-08-19)

Fixed

- Fixed the issue with image optimization cron and old images

---

2.0.14

(2024-04-22)

Improvements

- Performance (DB queries) optimized

2.0.13

(2024-03-25)

Fixed

- Fixed the issue with WebP for images with the same path but different file extensions

---

2.0.12

(2024-02-20)

Improvements

- Reset webp path in the database if webp file was removed from the filesystem

---

2.0.11

(2024-01-24)

Improvements

- Replace WEBP images in source tags

2.0.10

(2024-01-23)

Fixed

- Fixed the issue with preloading fonts

2.0.9

(2023-12-20)

Improvements

- Improved WebP image compatibility with product galleries and swatches

---

2.0.8

(2023-12-07)

Improvements

- Improved replacing swatch options images with WEBP

2.0.7

(2023-10-23)

Fixed

- Fixed the issue with generating WEBP images after optimized images have been generated

---

2.0.6

(2023-10-05)

Fixed

- Fixed the issue with notification messages disappear on frontend after module's ajax call

---

2.0.5

(2023-09-05)

Fixed

- Fixed the issue with error on configurable product page when the product has many configurable options (WEBP images)

2.0.4

(2023-08-16)

Fixed

- Fixed the issue with review block in Breeze theme caused by WebP feature

---

2.0.3

(2023-07-28)

Fixed

- Do not scan iopt dir for images to optimize

2.0.2

(2023-07-25)

Fixed

- Fixed the issue with error when generating WebP images for images with file extension in uppercase

---

2.0.1

(2023-07-24)

Fixed

- Fixed the issue with image optimization cron tasks running simultaneously
- Optimize image with relative urls

2.0.0

(2023-07-14)

Improvements

- Implemented a brand-new approach to image optimization.

Notice

- The extension can optimize images and generate WebP images not only from the pub/media folder but any internal image available in the store (theme, media, static files) when the Webpages image scan strategy is used.
- Significantly reducing resource sizes.
- Since the usage of browsers that do not support WebP images continuously decreasing and their usage is now ~1% worldwide the extension dropped support of such browsers.
- Old commands for image optimization and WebP images are now merged into one command with corresponding options (flags). For more information refer to the user manual of the extension.
- Separate cron tasks for optimizing images and generating WebP images now merged into one cron task.
- Configuration option "Use WebP images on product pages" is removed. Original images will be replaced automatically with WebP images on product pages when the Use WebP images setting is enabled and corresponding WebP images are generated.

- The command bin/magento mirasvit:optimize-image:reset now removes both optimized and WebP images.

1.5.1

(2023-06-06)

Fixed

- Webp images (HYVA)

1.5.0

(2023-04-20)

Improvements

- Migrate to declarative schema
- JS files collecting for JS bundles improved

1.4.0

(2023-03-16)

Fixed

- PHP8.1 compatibility

1.3.23

(2023-03-06)

Fixed

- Fixed the issue with Move JS/CSS features applied to AJAX responses (requests without proper headers)

1.3.22

(2023-03-02)

Improvements

- WebP images support for Swissup Breeze themes' product gallery widget

---

1.3.21

(2023-01-20)

Fixed

- Fixed the issue with errors during image optimization process (quote in filename)

1.3.20

(2022-11-15)

Fixed

- Fixed the issue with the error 'Undefined index: [data-role=swatch-options]'

1.3.19

(2022-08-18)

Improvements

- WebP for swatch images in product gallery

Fixed

- Console command return value

1.3.18

(2022-06-09)

Fixed

- Fixed the issue with memory limit error when LazyLoad enabled (since 1.3.17)

1.3.17

(2022-06-08)

Fixed

- Fixed a few issues related to errors detected by validator.w3.org

---

1.3.16

(2022-05-27)

Fixed

- Fixed the issue with webp images for video preview on product page
- Fixed the issue with frontend page builder

1.3.15

(2022-01-21)

Fixed

- PHP 8.1 compatibility

1.3.14

(2021-12-02)

Improvements

- Ability to disable JavaScript bundling on a page basis

1.3.13

(2021-10-18)

Fixed

- Issue with move JS to bottom and long scripts

1.3.12

(2021-10-13)

Fixed

- Possible gaps in the markup after enabling "Move JS to Page Bottom"

1.3.11

(2021-09-07)

Fixed

- Fixed the issue with webp images on product pages when tags included in fotorama config

Improvements

- Support for notorama product gallery widget

1.3.10

(2021-07-27)

Fixed

- Fixed the issue with "Move JS to Page Bottom" and long scripts (empty pages)

1.3.9

(2021-07-15)

Fixed

- fixed the issue with resized image can be smaller that original image with original image width in responsive images config

---

1.3.8

(2021-07-06)

Fixed

- Preload only configured CSS exceptions

1.3.7

(2021-06-29)

Improvements

- Proper work of 'Use Webp for Fotorama' feature with cache

1.3.6

(2021-06-22)

Fixed

- Fixed the issue with the error when webp image removed

1.3.5

(2021-06-16)

Improvements

- Support for Catalog media URL format 'Image optimization based on query parameters' (added in Magento 2.4.2)

1.3.4

(2021-06-14)

Fixed

- Remove from DB images that removed from the filesystem
- Update image status if webp image removed from the filesystem

---

1.3.3

(2021-06-11)

Fixed

- Fixed the issue with webp images when picture-source-img construction is present in the layout
- Correct type for preloaded CSS

Improvements

- Additional configs

1.3.2

(2021-05-20)

Fixed

- Fixed possible issue with not using webp images on product pages

---

1.3.1

(2021-05-13)

Fixed

- Fixed the issue with the error in logs (Undefined index: lighthouseResult)

1.3.0

(2021-05-05)

Fixed

- Fixed the conflict with 3rd-party fotorama-based gallery widgets (Notice: Undefined index: mage/gallery/gallery)

1.2.9

(2021-04-30)

Features

- Preload CSS exceptions

1.2.8

(2021-04-29)

Fixed

- Fixed the issue with product pages not being cached (affects from 1.2.1)

1.2.7

(2021-04-26)

Improvements

- Small improvement of optimizations regarding fonts

1.2.6

(2021-04-21)

Improvements

- Small optimization improvements

1.2.5

(2021-04-08)

Fixed

- Issue with error 'Uncaught Error: Call to a member function getWebpPath() on boolean'

1.2.4

(2021-04-06)

Fixed

- Fixed the issue with changing products color on category pages

---

1.2.3

(2021-03-19)

Fixed

- CSS move to bottom (prevent move for canonical tags)
- Issue with processing (loading) CSS for inline
- Issue with preload fonts (custom domain for static content)

1.2.2

(2021-03-12)

Fixed

- The error "Notice: Undefined index: HTTP_ACCEPT"

1.2.1

(2021-03-12)

Features

- Ability to use webp images in gallery widget on product pages

Improvements

- Ability to generate/delete resized images from CLI
- Responsive images generation improved

1.2.0

(2021-03-11)

Fixed

- Fixed the issue with images in search autocomplete popup (WYOMIND)
- Fixed the issue with preload fonts
- Fixed the issue with applying optimizations in feeds
- Minor fixes

1.1.9

(2021-02-04)

Improvements

- Score check improved [#124]()

Fixed

- Fixed the issue with lazyload iframes
- Fixed the issue with inline CSS from resources feature

1.1.6

(2021-01-15)

Fixed

- Fixed the issue with error related to the responsive images functionality [#114]()

1.1.5

(2021-01-14)

Features

- Responsive Images ()

Improvements

- Ability to exclude webp image from lazyload by matching class of the original image [#110]()
- Minify JS exceptions [#111]()

1.1.4

(2020-12-24)

Improvements

- Webp quality depends on the Image Quality Level config [#106]()

Fixed

- Fixed the issue with inline styles from sources without protocol [#107]()

1.1.3

(2020-12-16)

Improvements

- GIF to WEBP conversion [#102]()
- Ability to add styles from CSS resourses directly to the HTML document [#104]()

Fixed

- Fixed the issue with empty exceptions for image lazyload [#103]()

1.1.2

(2020-12-15)

Improvements

- Ability to pre-connect third-party origins [#100]()

1.1.1

(2020-12-11)

Improvements

- Database performance improved [#98]()

1.1.0

(2020-12-09)

Fixed

- Fixed issue with errors during webp conversion due to braces in the name of the file [#96]()
- Fixed the issue with the admin panel not accessible after disabling JS optimizations [#95]()

1.0.25

(2020-12-07)

Improvements

- Database performance (indexes) [#93]()

1.0.24

(2020-11-25)

Improvements

- Ability to change some optimization configs in the scope of stores [#91]()
- Add debug.css only to debug pages [#90]()

1.0.23

(2020-11-19)

Improvements

- Image optimization strategy for webp conversion [#88]()

1.0.22

(2020-11-18)

Features

- Lazyload for offscreen iframes ([#86]())

1.0.21

(2020-11-12)

Improvements

- Additional CSS ([#83]())

Fixed

- Fixed the issue with image source in single quotes - webpages strategy ([#84]())

1.0.20

(2020-11-09)

Improvements

- Lazyload ([#80]())

1.0.19

(2020-11-03)

Improvements

- Improved image optimization strategy ([#78]())
- Debug mode for OptimizeImage ([#75]())

1.0.18

(2020-10-19)

Fixed

- Fixed issue with adding preload fonts to ajax response ([#72]())

1.0.17

(2020-10-06)

Improvements

- Minor code changes

1.0.16

(2020-10-01)

Fixed

- Compatibility with PHP 7.4

1.0.14

(2020-09-16)

Fixed

- Fixed error on pages with a few YouTube videos ([#64]())

1.0.13

(2020-09-15)

Improvements

- Validation for the ability to optimize images ([#62]())

1.0.12

(2020-09-10)

Improvement

- Conflicts validation

1.0.11

(2020-09-07)

Fixed

- Fixed issue with warnings during png images compression.

1.0.10

(2020-09-04)

Improvement

- Images compression ([#50]())

1.0.9

(2020-09-01)

Improvements

- Treshold for bundle/track requsts ([#46]())

Fixed

- Fixed issue with not all images displayed in some sliders ([#47]())

1.0.8

(2020-08-27)

Feature

- Lazy load for YouTube videos

Improvements

- Merge JS files option in the configurations of the extension

---

1.0.7

(2020-07-29)

Improvements

- Support of Magento 2.4

Fixed

- issue with CSS exceptions ([#37]())

1.0.6

(2020-06-17)

Fixed

- Issue with lazy load for images with a path in single quotes

---

1.0.5

(2020-06-02)

Fixed

- Issue with Magento 2.1.* compatibility

Improvements

- Disabling images conversion to webp when "Use WebP Images" is disabled

---

1.0.4

(2020-04-23)

Fixed

- Not all images visible with webp and lazyload
- Issue with webp conversion (Unsupported color conversion request)

1.0.3

(2020-03-10)

Improvements

- PageSpeed Optimizer in admin menu
- Move lazy-load exceptions to configuration

1.0.2

(2020-03-05)

Improvements

- Ability to exclude JS files from movement to the bottom of page
- Lazy-load (server transparent image placeholder in the original image size)
- Request validation (post, ajax, checkout etc)

1.0.1

(2020-02-10)

Improvements

- Additional exceptions (configurable in admin) for lazy-load

Fixed

- Issue with css defer
- Small config issues

1.0.0

(2020-02-03)

Improvements

- Additional exceptions for lazy-load

0.0.10

(2020-01-24)

Improvements

- Lazy load for webp images
- Ability to exclude specified URLs from move_js

Fixed

- Possible issues with measure google pagespeed score

0.0.9

(2020-01-02)

Improvements

- WebP images settings
- WebP Picture tag
- Added the command for validate required image optimization software

---

0.0.8

(2019-12-26)

Improvements

- WebP support

0.0.7

(2019-12-25)

Fixed

- Issue with saving the settings

0.0.6

(2019-12-24)

Improvements

- Lazy Images & Defer google fonts

0.0.5

(2019-12-23)

Fixed

- Replace 1px empty image with transparent

0.0.4

(2019-12-23)

Improvements

- Image optimization
- Add css option to all css files: font-display:swap
- Exception URL list for Move CSS option
- Deffer CSS

0.0.3

(2019-12-18)

Improvements

- Image optimization statistic
- JS minification
- PHPMD
- Debug option for lazy load images

- Optimize images queue

Fixed

- Issue with LazyLoad (Firefox)

0.0.2

(2019-12-16)

Improvements

- Image optimization module

0.0.1

(2019-12-11)

Improvements

- Initial release