



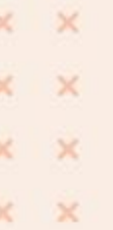
Everything you should know about handling Images in Drupal

Let's talk about getting most of the things right related to images in Drupal.



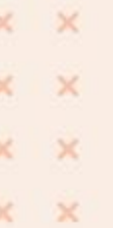
About me

- Gaurav Kapoor (Gurgaon, HR, India)
- Senior Drupal Developer at [Axelerant](#)
- [gaurav.kapoor](#) on Drupal.org
- Active contributor and maintainer of various projects.
- Blogger at [gktwlab.com](#) and [drupalabc.xyz](#)



Contents

- Image upload and server configuration.
- Image upload UX improvement.
- Manual image compression before upload
- Image storage metadata in Drupal
- Accessibility and Images
- Image Field vs Media Images
- Image Styles
- Responsive images in Drupal
- Image optimisation and CLI compression tools
- Image storage on Amazon S3 and DAMs.
- Next Generation Image Formats



Why knowing everything is so important?

- Avoid all possible bugs and issues related to image configurations and handling.
- Don't let images affect performance or accessibility.
- Major rework can be avoided if all constraints are known during discovery.
- Do not give editorial team a chance to hate Drupal.
- Architect websites in a better way.
- Keep human effort for gathering, uploading and managing data minimum.

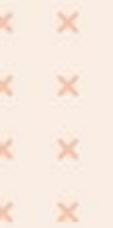


Image upload and server configurations

- Correct Drupal configurations (Size in pixels, image extension types, size in MB, storage type etc.)
- PHP configuration (Settings like `upload_max_filesize`, `post_max_size`)
- Apache/Nginx configuration (Settings like `client_max_body_size`, `request_size`)
- CDN/Reverse Proxy (Generally very high but still better to be sure about it.)
- These setting should be part of your deployment configuration/notes.
- Completely avoid bug tickets related to these kind of errors. Better to handle all these as early as possible.

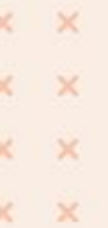
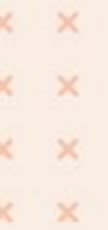


Image upload UX improvement

- Enhance editorial experience by providing features like cropping, adjusting focal point and compressing. Easy add-ons for your clients/users.
- Drupal module - [Image Widget Crop](#) - Provides option to crop images during upload.
- Drupal module - [Focal Point](#) - Specify the most important part of image.
- Drupal module - [Dropzone JS](#) - Drag n drop file upload instead of traditional upload button.
- Custom plugins to integrate any PHP/JS library.



Manual image compression before uploading

- Equally important as automated image compression.
- Very large size images (getting into MBs) should be compressed.
- Editors/Content creators should understand the importance and should also be made aware of impact of big size images on web applications.
- Use services like [TinyPNG](#), [Compress Now](#) and [Compress JPEG](#).
- JS libraries also available to automate the process during image upload.

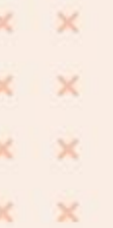


Image metadata storage in Drupal

- Database tables - 'file_managed', 'file_usage' and {'node__field_image'}.
- Important to understand storage to debug images related issues.
- Helpful in migration to/from Drupal and automated testing.
- Useful when switching between hosting providers.



Accessibility and Images

- All the images in your website should have an [appropriate alternate text](#). Official resources are helpful for developers, content creators and quality assurance engineers.
- Alt text field should be marked as required in all the forms with option of uploading images.
- Make sure images in the migrated content or any content getting generated programmatically have alternate text set.
- Leverage AI services provided by various cloud platforms to generate alternate texts. Drupal modules - [Drupal Rekognition](#), [Recognition API](#), and [Automatic Alternative Text](#) can help in automating the alternate text, description and caption generation.



Image Field vs Media in Drupal


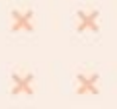
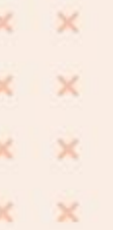
- Media entity was introduced in Drupal core in version 8.7.
- Editors can easily add Images, Audio, Video, Documents, and various other type of media content. 
- Use Media Image in case you want to provide functionality of images reusability and easy management. 
- Use Image field for simple purposes like gathering image data from anonymous or non-admin users.
- In case, both type exist in your application, make sure proper help texts are added for users to differentiate easily between the two type of fields.
- Media images functionality can be further enhanced by using modules such as [Entity Browser](#), [Entity Embed](#) and various other modules.



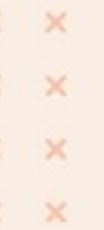
Image Styles in Drupal

- Most useful out of the box feature provided by Drupal related to images.
- Purpose isn't limited to generating images of different resolutions. The size of images in KB is also reduced.
- Should be preferred over using CSS.
- Drupal module - [Image Style Warmer](#) can be used to generate various presets when image is uploaded.



Responsive Images and picture tag in Drupal

- Very unpleasant to see images affecting website layout in mobile or tablets.
- HTML solution is using `<picture>` tag or `img srcset`.
- Core provided 'Responsive Image' module can be enabled and configured to use picture tag easily in Drupal.



Retina Images in Drupal

- Devices with Retina displays are very common nowadays. Your website should serve high quality images to look good on a device with retina display.
- Images with pixel size 2x have to be uploaded by the editors/content creators.
- Image style is the feature that can be leveraged to provide this functionality.
- Core module 'Responsive Images' can be cleverly used to for having retina images.
- Drupal module - [Retina images](#) automatically generates image styles with 2x dimensions. (Currently, only has development release for D8/D9)

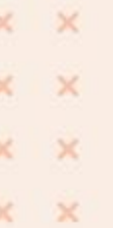
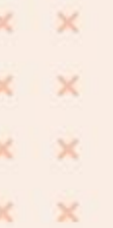


Image Optimization and CLI Compression Tools


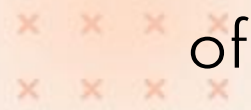
- **Image Quality:** Drupal core feature for lossless compression of the images. It is usually set between 60%-75% and relies on PHP-GD library. Contributed module - [ImageMagick](#) can be used to perform compression using 'Imagemagik' PHP library.
- **Image Compression Pipeline:** Setup a pipeline of image compression steps using the [Image Optimize](#). Provides easy way to integrate with various CLI image compression tools and paid services such as TinyPNG.
- **Lazy Loading:** Delay loading of image till the time user scrolls to the part where image is displayed. Contributed modules [Lazy-Load](#) and [Blazy](#) can be used to implement lazy loading.
- **Expiry Headers:** Set an 'expired header' for the image data being served from your server as that helps browsers cache the images and not request them again and again.
- **CSS Sprites, SVGs and Base64 encoding** are some other ways to save bandwidth required for serving images.

Image storage on Amazon S3 and serving via CDNs in Drupal

- Storing images on different server can help in reducing request to the server/VM running your Drupal codebase.
- Cloud storage dedicated to images is very cheap and has security benefits as well.
- Drupal module - [S3 File System](#) can be used to introduce file data storage in Drupal.
- Different modules are available for integrating with all popular cloud services and CDNs.
- CDN's like Fastly even provide features of optimising images further.



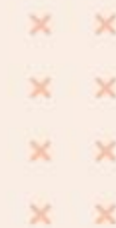
Integration with Digital Asset Management (DAM)

- It is a very standard practice of storing all the digital assets such as Images, Videos, Audio Files, Documents, and other graphical content in software dedicated to easy media management. 
 - By using centralized storage, organizations are able to use a single digital asset on multiple applications as well as get the flexibility of modifying media content without doing any changes in applications linked to their DAM software.
 - You'll very easily be able to find a contributed module or integration notes for most of the famous DAM software with Drupal such as [Bynder](#), [Vimeo](#), and [Acquia DAM](#).
 - By using a DAM, you get all the advantages of serving images from a different server, backup of assets in case of failure as well as managing utilities. 
-

Next Generation Image Formats

- Most of the images which are available on the internet right now mostly have an image format 'JPEG', 'PNG' or 'GIF'. 'SVG's have also become popular in some last year because of their advantages and lossless nature.
- Images with next-generation formats such as WebP, JPEG2000, AVIF, and HEIC are now being actively used to display better-looking pictures of less size, that load very quickly and thus enhance the overall web surfing experience.
- AVIF is newer than WebP and has better compression without compromising on image quality. It is recommended to allow support for these in your project even if it is not being used right now as these will definitely get popular in the future.
- Most probably this will be supported out-of-box Drupal core in the upcoming releases, you can check module [WebP](#) to provide support for WebP format images.

Questions?



See you in Prague !!

- “How to encourage your team members to contribute to open source?”
- Location: Room D7
- When: Wednesday, 21 September, 2022 - 17:40 to 18:00

