BRO Academy

Manual

Images

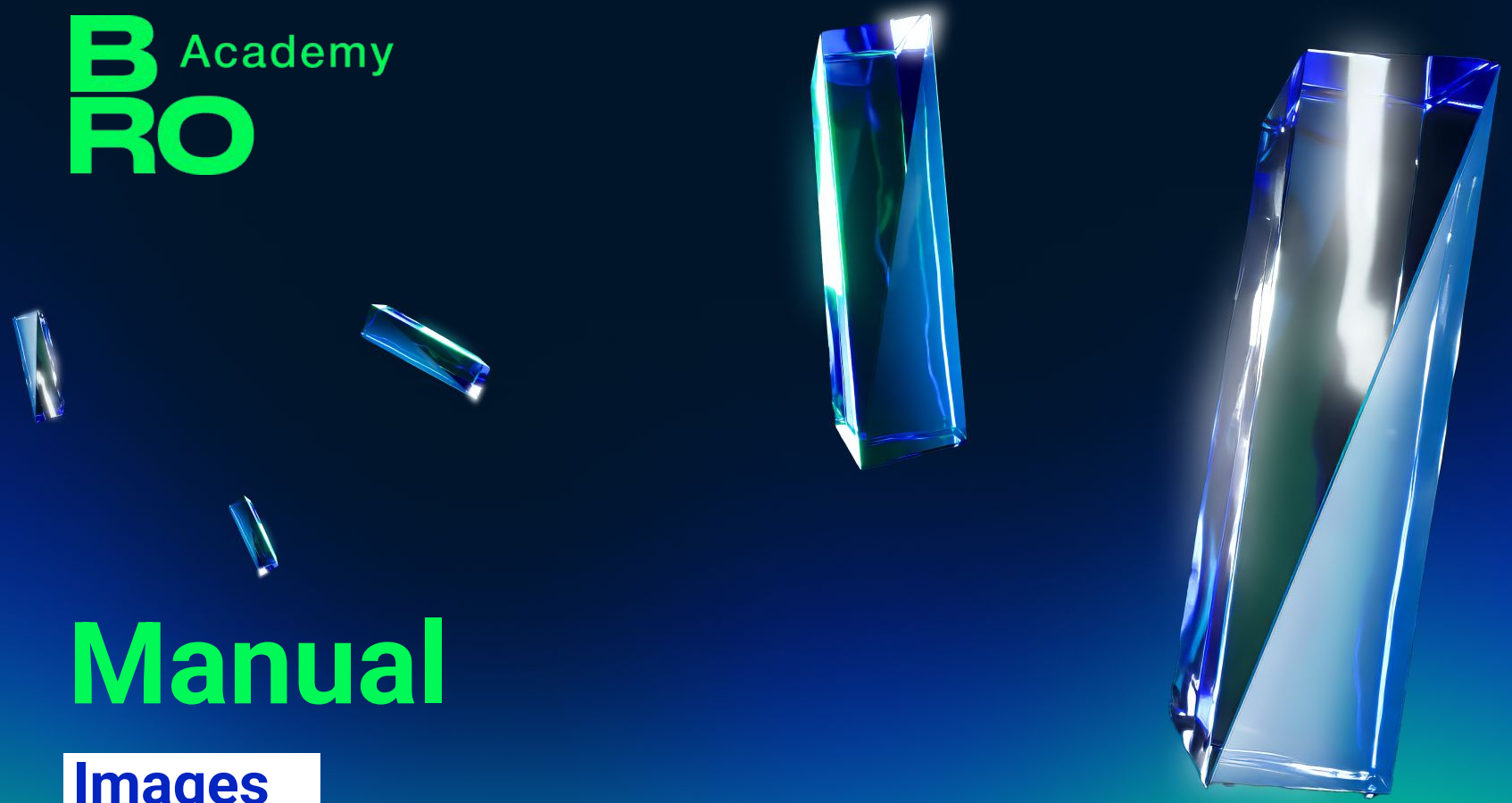


Image formats

raster images are composed of pixels, which are tiny dots of color that make up the image. Each pixel has a fixed position and color, giving raster images a fixed resolution. This means that if you try to scale up a raster image, it can become pixelated or blurry because you're essentially stretching the pixels.

raster images

Common raster image formats include:

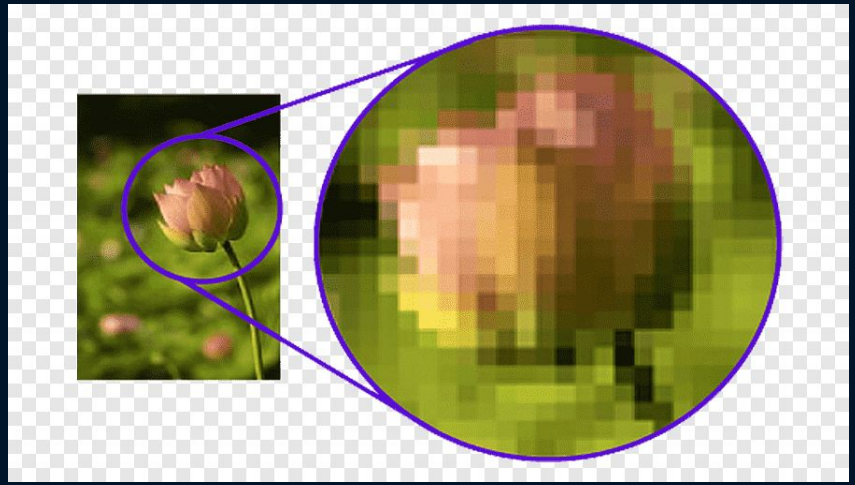
- **JPEG:** This format is great for photographs and images with lots of colors and details. It uses lossy compression, which means it reduces file size by discarding some data, which can lead to a slight loss in quality.
- **PNG:** Ideal for images that require transparency and for graphics with sharp edges, like logos. PNG uses lossless compression, so the image quality remains intact.
- **GIF:** Best for simple animations and images with limited colors. GIF supports transparency but only in a binary (fully transparent or fully opaque) way.
- **WEBP:** A modern format that provides superior compression and quality compared to JPEG and PNG. It supports both lossy and lossless compression.
- **AVIF:** Another modern format that offers even better compression rates and image quality than WEBP, making it great for high-quality images at smaller file sizes.

	JPEG **J**oint **P**hotographic **E**xperts **G**roup	**PNG** **P**ortable **N**etwork **G**raphics	**GIF** **G**raphics **I**nterchange **F**ormat	**WEBP**	**AVIF** **AV**1 Still **I**mage File **F**ormat
Lossless compression	Compression with quality loss			Better compression, retains quality	Excellent compression, retains quality
Size	Relatively small file size	Large file size		Weighs 30-40% less than JPEG and PNG	Weighs 10-20% less than WEBP
Detalisation	Common format for photos	Retains detail and color contrast		loss and blurring of fine details with strong compression	Suitable for large images, smaller images are better as WEBP
Text	hard to read text			may look blurry	
Supports transparency					
Colors		16 million colors	Limited colors (256)	Suitable for bright and colorful photos as it can display more pixels	Best choice for video, animations, and images with transparency
Animation					
Browser support	100%	100%	100%	96%	75%

Use raster

Images formats

photos	**jpeg, webp, avif**
images with transparency	**png, webp, avif**
images with gradient	**webp, avif**
large images	**webp, avif**



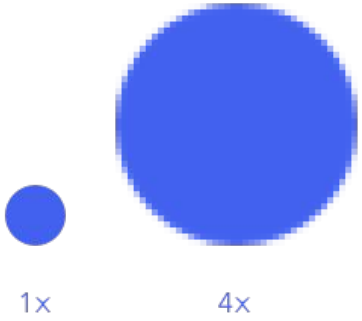
vector images are created using mathematical formulas and geometric shapes, such as lines and curves. Unlike raster images, vectors don't have a fixed resolution. This allows them to be scaled infinitely without losing quality, making them perfect for logos, icons, and any graphics that need to be resized frequently.

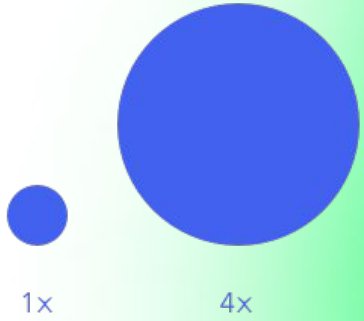
vector images

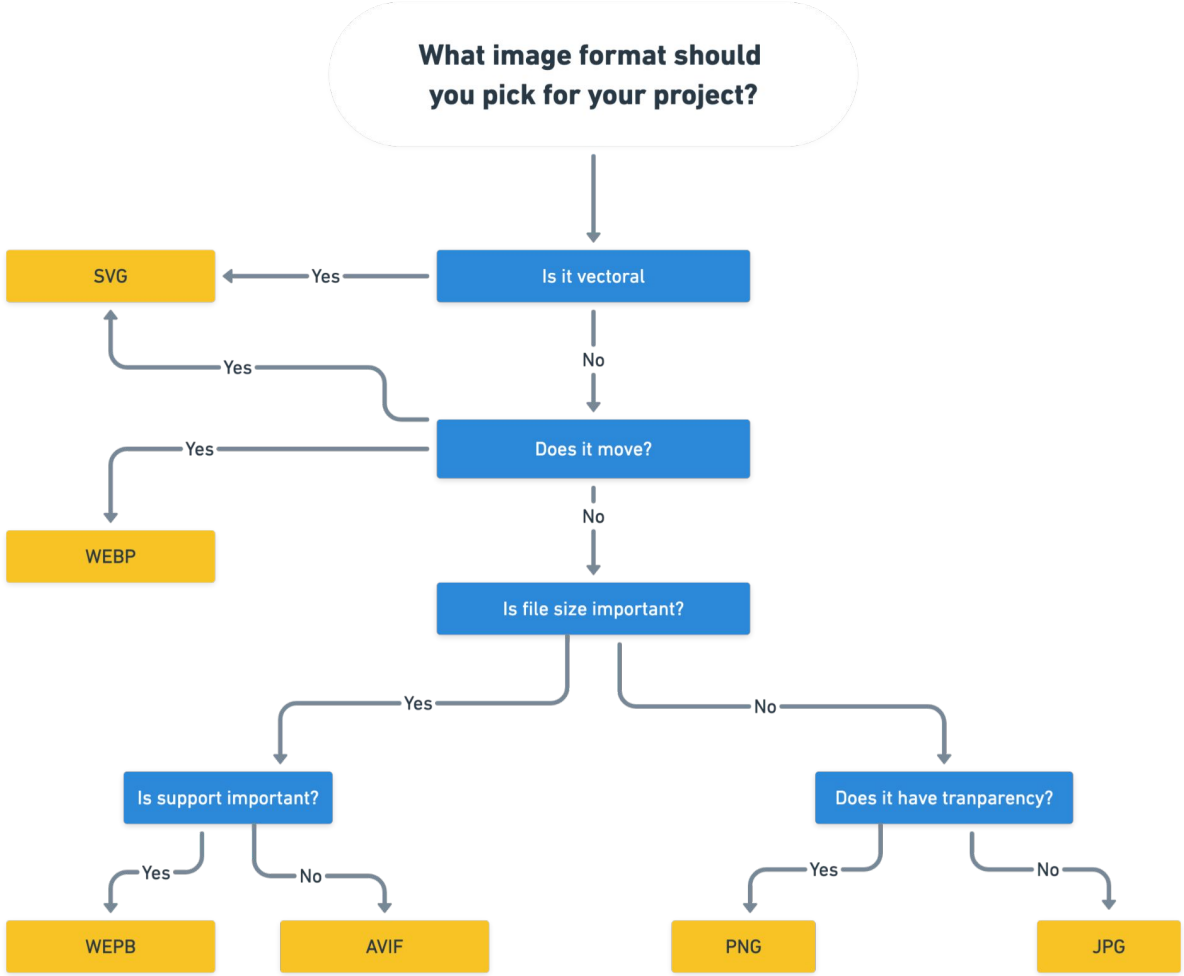
SVG: The most common vector image format used on the web. SVG files are lightweight and can be animated and styled with CSS, offering great flexibility and high-quality graphics.

VS

Raster	Vector
Consist of pixels	Consist of mathematical formulas, shapes, and lines
Large file size	Very small file size
Scales with quality loss	Looks the same at any size
Can only be converted to other raster formats	Can be converted to raster images
Cannot be modified easily	Can be easily modified (e.g., splitting into components, animating)
Ideal for photographs	Ideal for icons
Uses RGB color space	



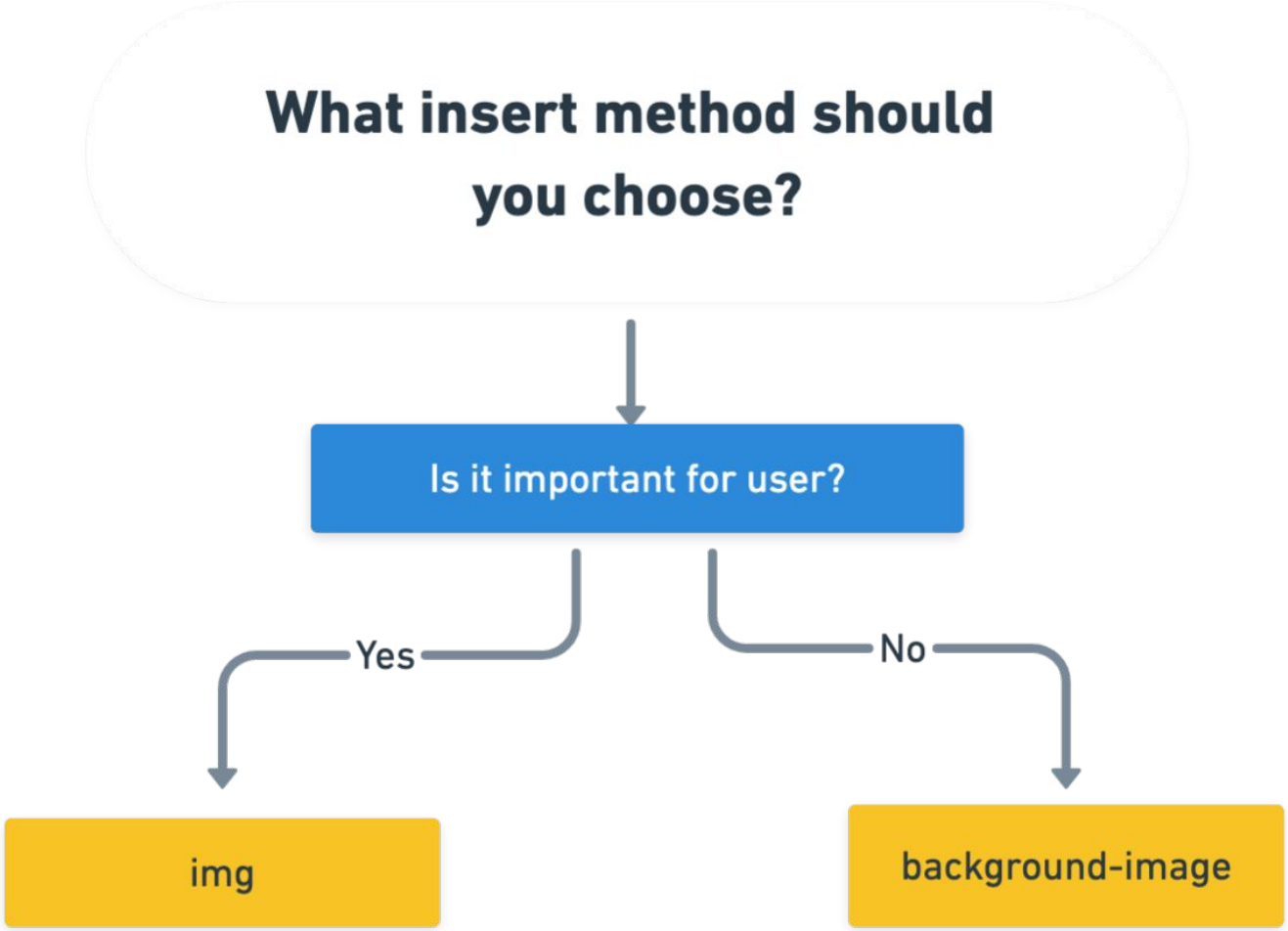



What image format should you pick for your project?
Is it vectoral
Yes
SVG
No
Does it move?
Yes
Yes
WEBP
No
Is file size important?
Yes
No
Is support important?
Does it have tranparency?
Yes
No
Yes
No
WEPB
AVIF
PNG
JPG

Choosing the Insertion Method

Content Images –  в HTML
– Essential for user understanding
– Important for SEO
– Take up space in the document flow

Decorative Images –  в CSS
– Non-essential for user understanding
– Used for decoration
– Not important for SEO
– Often have other elements overlaid


What insert method should you choose?
Is it important for user?
Yes
No
img
background-image

<img> tag

img

```
<img src="path.jpg" width="100" height="100" alt="cute corgi">
```

File naming:

- Use only English words
- No spaces
- No capital letters
- Reflect the location or subject of the image

blue triangle.jpg
BigPicture.jpg

cute-kitten.jpg
feature-block-1.jpg

Attributes:

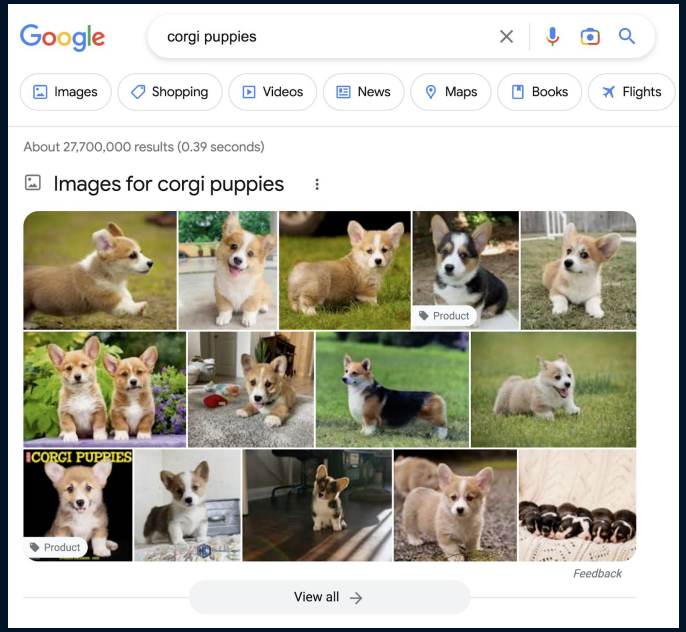
- `width/height:` Specifies width/height of the image
- Used for displaying images before styles load, in reading mode, and for accessibility

Importance of the alt Attribute

```
<img src="path.jpg" alt="cute corgi puppy on green field">
```

- Screen readers
- Search engines
- Broken images

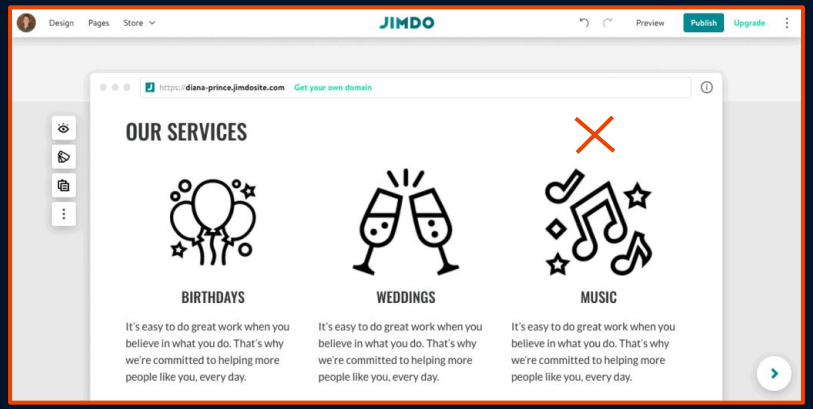
Digital image that loaded correctly	Digital image that did not load correctly and does not have alt text	Digital image that did not load correctly and **does** have alt text
		St. Isidore of Seville



When alt is not needed

```
<img src="path.jpg" alt="" >
```

- Decorative and stock images
- Icons
- When the image caption matches the `alt` text



When is the alt Attribute Not Needed?

Decorative and Stock Images:

- For purely decorative images that do not add any meaningful content to the page, you can use an empty alt attribute. This tells screen readers to ignore the image, preventing unnecessary descriptions that do not contribute to the user's understanding.
- Example: Background patterns, decorative borders, or purely aesthetic elements.

Icons:

- Icons used purely for decoration, such as those from a set of icons to enhance the visual layout, do not need alt text. These could be icons for social media links or small graphical elements that do not convey additional information.
- Example: Social media icons where the link text itself already describes the purpose.

When the Image Caption Matches the alt Text:

- If the image is accompanied by a caption that provides the same information as what you would put in the alt attribute, it is better to use an empty alt attribute to avoid redundancy.
- Example: An image with a caption directly below it that describes the image content.

- [x] Man wearing a helmet riding a bicycle
- [x] bicycle
- [x] Image of someone riding a bicycle

- [x] Lay's Classic Potato Chips, 1.5 Ounce
- [x] Potato Chips
- [x] buy potato chips online

- [x] Free SEO mini course for beginners by reliablesoft academy.
- [x] download now free
- [x] register to get free course

Good alt Practices

- Summarize what’s in the image
- Describe as accurately as possible
- Avoid stating "image", "photo", "picture"
- Use the context of the surrounding content
- Avoid repeating text that is in the content
- Don’t use very long descriptions, use longdesc for detailed descriptions
- Write in the site's language

<picture> tag

`<picture>` tag allows you to specify multiple image sources for different scenarios, such as different screen sizes or resolutions. It's useful for responsive design.

Use cases:

- Media queries
- Format
- Width and Pixel Density descriptor
- Cropping

<picture>

Cropping

If you have images, but need some adjustments with their sizes based on the screen, width and height can be implemented to crop images.

For example, narrow and tall images for mobile, but wide and short images for desktop browsers.

```
<picture>
  <source
    srcset="large.jpg"
    media="(min-width: 960px)"
    width="1280" height="600"
  />
  <source
    srcset="medium.png"
    media="(min-width: 500px)"
    width="800" height="400"
  />
  <img
    src="small.png"
    alt="Image Description"
    width="500" height="400"
  />
</picture>
```

Media Queries

The different images for different screen sizes (with media)

```
<picture>
  <source srcset="large.jpg" media="(min-width: 960px)" />
  <source srcset="medium.png" media="(min-width: 600px)" />
  <img src="small.png" alt="Image Description" />
</picture>
```

Format

The same image in different formats.
The browser will pick the first format it recognizes and ignore the rest. Therefore, it might be a good idea to put high-quality images upfront and low-quality images at the end.

```
<picture>
  <source srcset="image.avif" type="image/avif" />
  <source srcset="image.webp" type="image/webp" />
  <img src="image.png" alt="Image Description" />
</picture>
```

Width and Pixel Density descriptor

The different images for different screen sizes

What we are telling in the below is that we want the browser to change the image automatically depending on the screen size. These values, "500w, 860w, and 1200w" represent the actual size or more specifically width of the image. The difference is that you provide values in w (width viewport) instead of px (pixels) units.

Alternatively, a pixel density descriptor can be used as well. It defines the pixel density of the image. The value, represented in x, such as 2x displays high-resolution images on high-density (DPI) screens.

```
<picture>
  <source srcset="small.png 500w" />
  <source srcset="medium.png 860w" />
  <source srcset="large.jpg 1200w" />
  <img src="small.png" alt="Image Description"
/>
</picture>
```

srcset attribute is an HTML image attribute that specifies the list of images to use in different browser situations. The browser will pick the most optimal image version, based on the screen size and resolution. This concept is known as responsive images.

srcset attribute

- Here srcset tells a browser to load 480-pixel wide image version if the viewport width is less than 600 pixels. If the viewport width is more than 600 pixels, the browser will load a larger image, 800-pixel wide.

```
<img
      srcset="ahrefs-image-480w.jpg 480w,
      ahrefs-image-800w.jpg 800w"
      sizes="(max-width: 600px) 480px, 800px"
      src="ahrefs-image-800w.jpg"
      alt="Cool ahrefs illustration">
```

- Alternatively, you can set the pixel density (device pixels per CSS pixel), as a condition instead of size:

```
<img
      srcset="ahrefs-image-1x.jpg 1x,
      ahrefs-image-2x.jpg 2x"
      src="ahrefs-image-1x.jpg" alt="Another cool
ahrefs illustration">
```

Image optimization

Why Optimize Images

conversion (%)

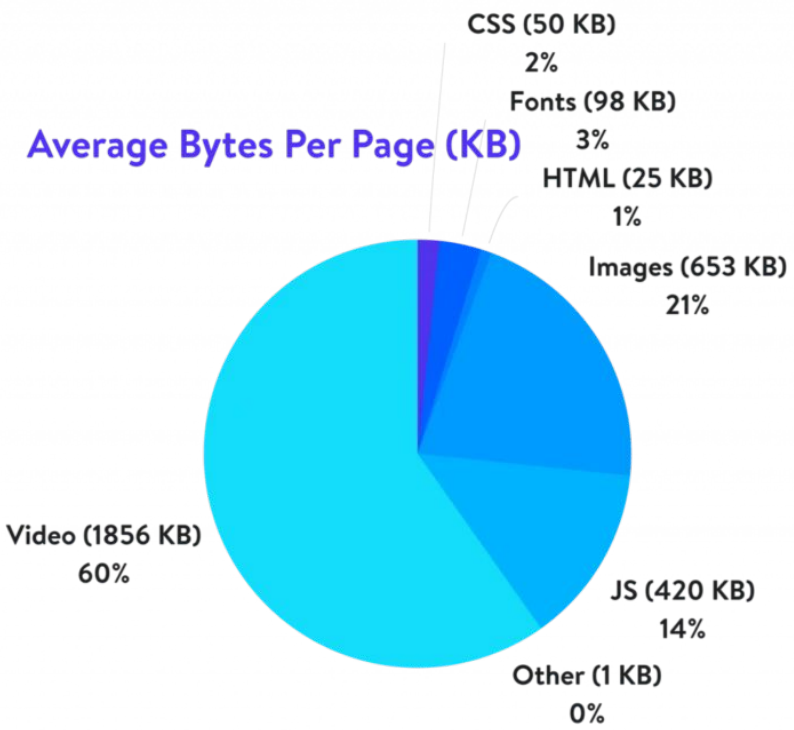
loading time (sec)

- Improves website loading speed
- Saves traffic and server resources
- Enhances website ranking in search results
- Increases conversion rates

Optimization

Methods

- Defer offscreen images
- Compress file size
- Crop to exact size
- Change format
- Use CSS styling instead of image merging
- Decomposition
- Using patterns



Optimization methods

Defer offscreen images

```
<img src="corgi.jpg" width="100" height="100" alt="cute corgi"
load="lazy" decoding="async">
```

`load`:

- `eager` tells the browser to load the image as soon as the `<img>` element is processed
- `lazy` tells the browser to load the image shortly before it will appear in the viewport

`decoding`:

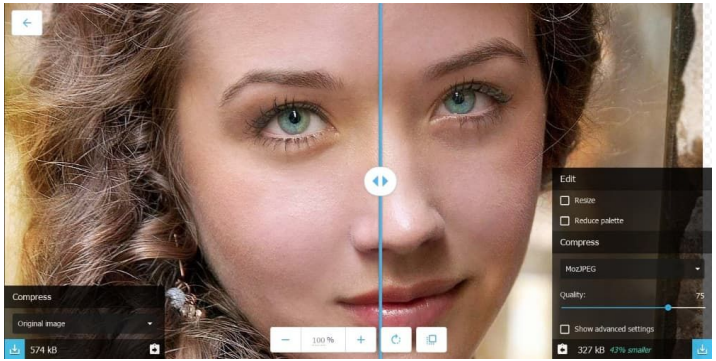
- `async` tells the browser to decode the image asynchronously and allow other content to be rendered before this completes
- `sync` tells the browser to decode the image synchronously for atomic presentation with other content.
- `auto` tells the browser to decide how to decode

Optimization methods

Reduce File Size

For example

- **squoosh.app**
- **SVGOMG**



- **Compression:** Use tools like TinyPNG, ImageOptim, or Photoshop to compress your images without significant loss of quality. This reduces the file size, making your site faster to load.

Optimization methods

Decomposition

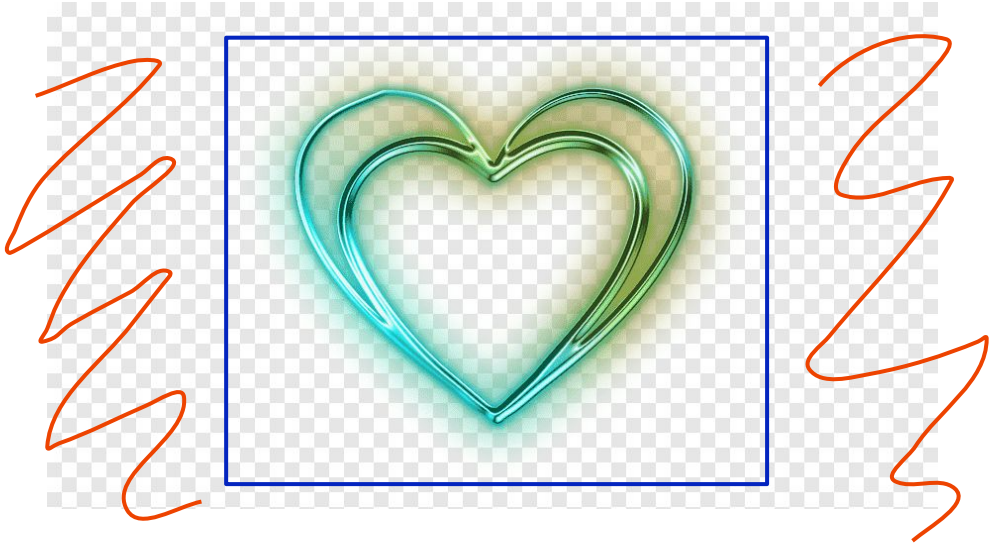
- **Split Images:** For large and complex images, consider splitting them into smaller parts. Load only the necessary parts initially and lazy-load the rest as needed. This technique can improve initial load times and overall performance.

Optimization methods

Crop to exact size

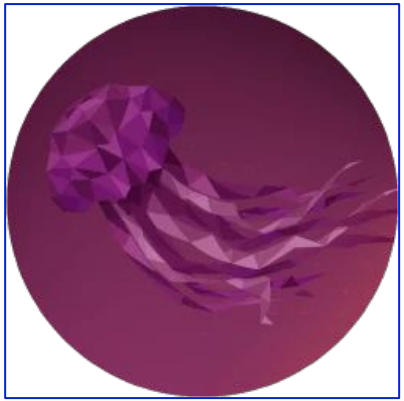
No empty areas or unnecessary transparency

- **Crop Images:** Crop your images to the exact dimensions needed for your design. Avoid using oversized images and scaling them down with CSS, as this can waste bandwidth and slow down your page.

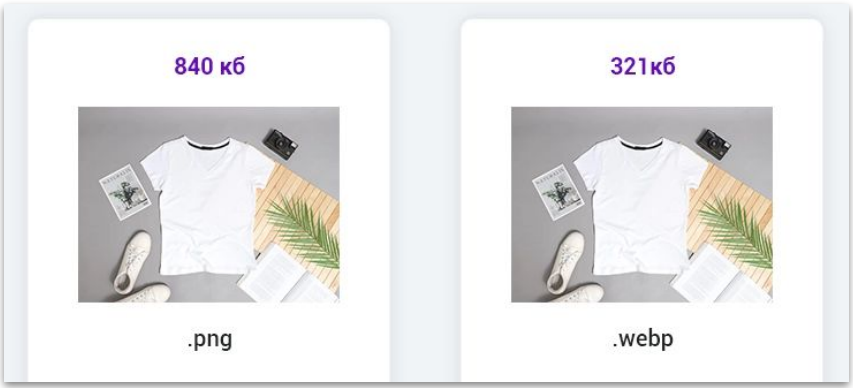


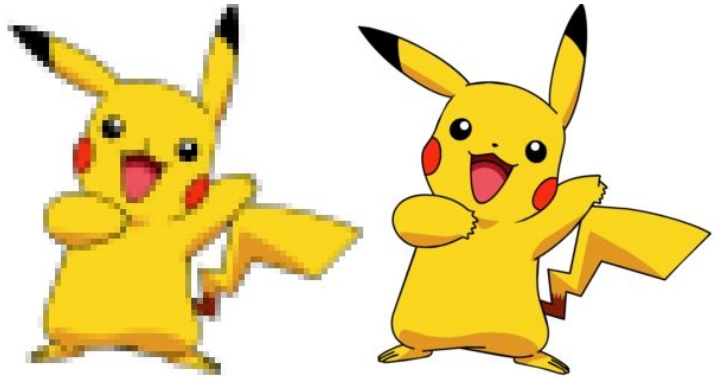
Optimization methods

Change format



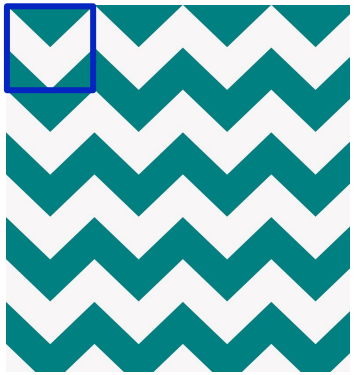
- **Choose a more modern format:** Consider using modern image formats like WebP and AVIF, which offer better compression rates and quality compared to traditional formats like JPEG and PNG.
- **Choose the Right Format:** Select the appropriate format for your image. For example, use JPEG for photographs and SVG for images with transparency or sharp edges.



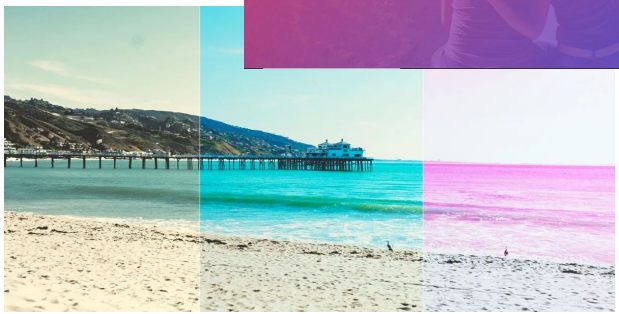


Optimization Methods

- **CSS Effects:** Use CSS for effects like gradients, shadows, and borders instead of creating these effects in an image editor. This reduces the need for additional images and decreases file size.



- **Repeating Patterns:** Use CSS to create repeating patterns or textures instead of using large images. CSS patterns are more lightweight and can be scaled easily without losing quality.



- **SVG Patterns:** SVG is also great for creating scalable and lightweight patterns. They can be defined once and reused throughout your design.

Export from Figma

Fill

Image 100 %

Stroke

Effects

Export

1x JPG

Export image 8

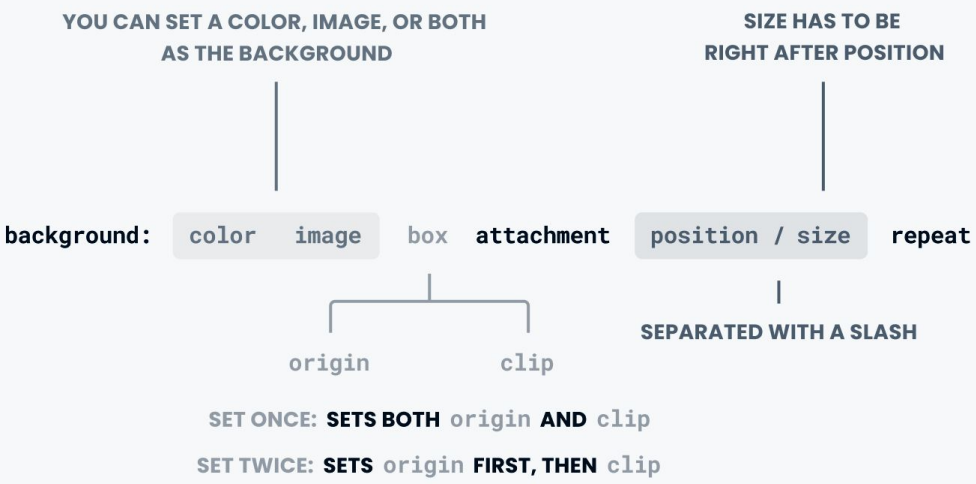
Preview

scale factor

image format

Export button

image preview



CSS BACKGROUND

background-color / image

background-color

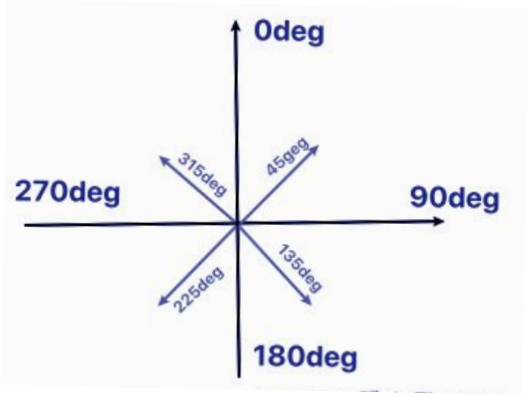
background-image

backgrounds-color for background-image

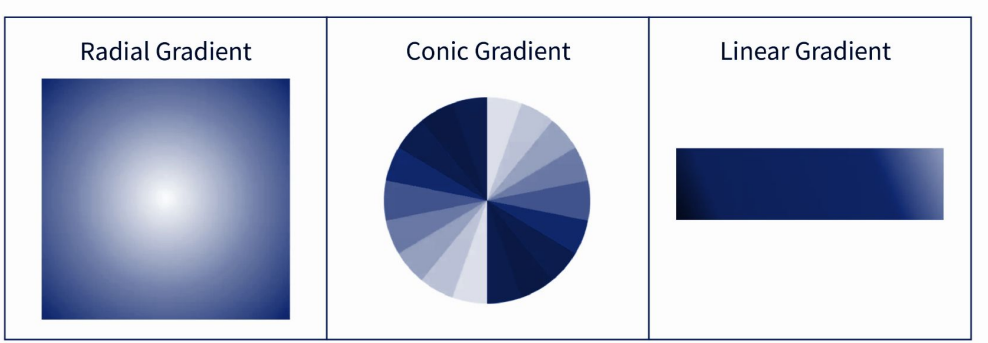
- Fallback for Missing Images
- Improved Loading Experience
- Enhancing Readability
- Visual Consistency
- Accessibility
- Design Control

background gradients

background-image: *-gradient(angle or direction , list of colors with transition points **)**



– The angle is specified in degrees – **deg**
– The direction is specified using words: **to left, right, top, bottom**



background linear gradient

Linear gradients create a smooth transition between two or more colors along a straight line. This can be used to add depth, visual interest, and a modern look to your web design.

linear-gradient(angle or direction , list of colors with transition points**)**

```
background-image:
     linear-gradient(
          90deg,          (to right)
          rgba(2,0,36,1) 0%,
          rgba(9,9,121,1) 35%,
          rgba(0,212,255,1) 100%
);
```

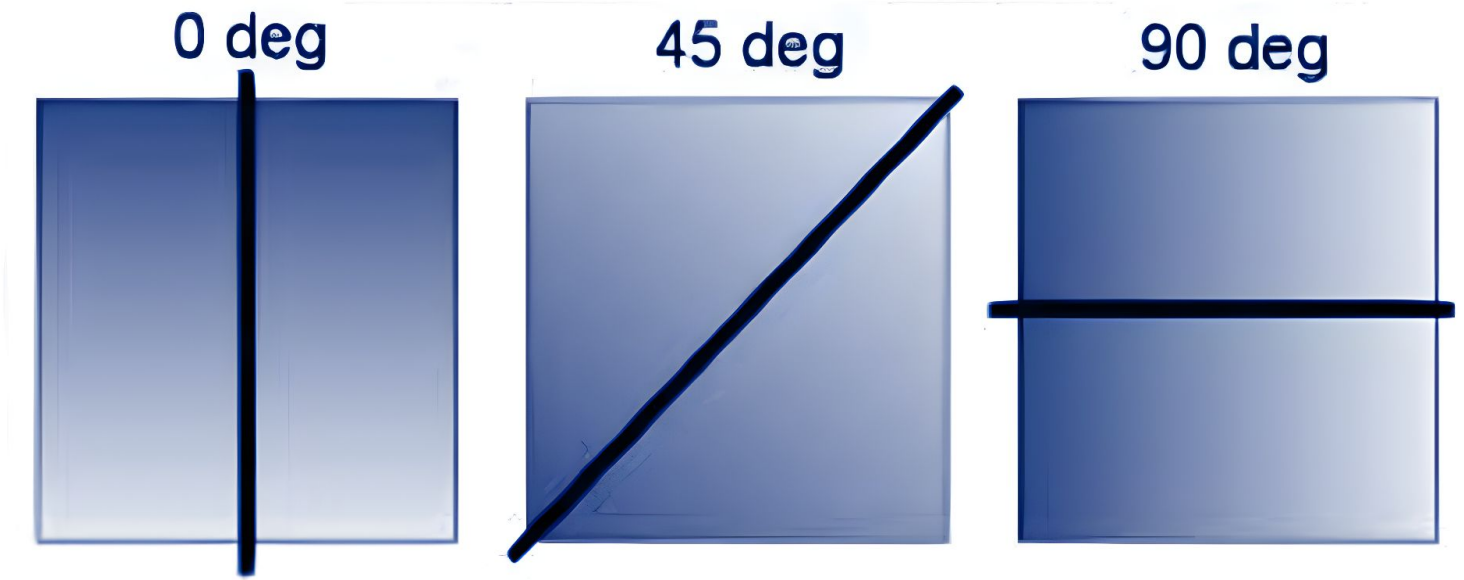
Angle or Direction:

- Angles are defined in degrees (**deg**).
- Directions can be specified using keywords like **to left, to right, to top, to bottom**.

List of Colors with Transition Points:

- You can specify multiple colors for the gradient.
- Optional transition points can be set using percentages to control where each color starts and ends.

background linear-gradient

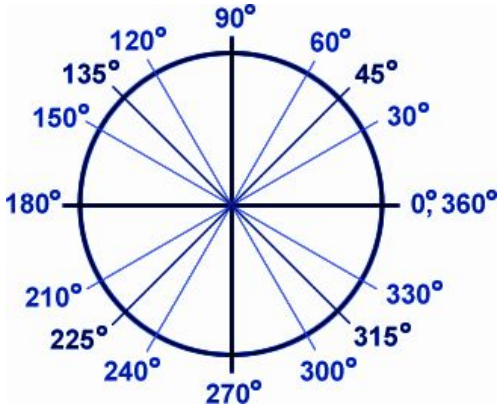


to bottom

to top right

to right

linear-gradient direction



45deg
to top right

135deg
to bottom right

225deg
to bottom left

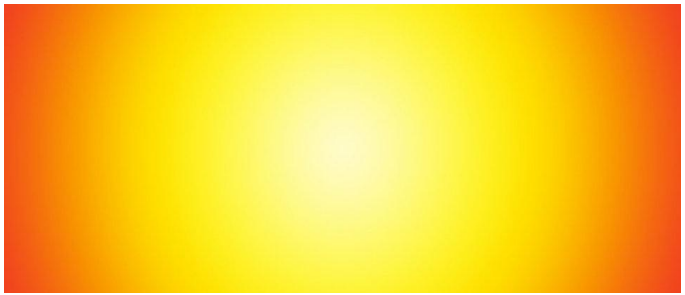
315deg
to bottom left

background radial-gradient

create a smooth transition between colors radiating outward from a central point. This type of gradient is perfect for creating spotlight or sunburst effects in your web design.

radial-gradient(shape or size direction, colors with transition points **)**

```
background-image:
     radial-gradient(
          circle at center,
          rgba(63,94,251,1) 0%,
          rgba(252,70,107,1) 100%
);
```



Shape or Size:

- Common forms are **circle or ellipse**.
- Sizes can be defined as **closest-side, farthest-side, closest-corner, or farthest-corner**.

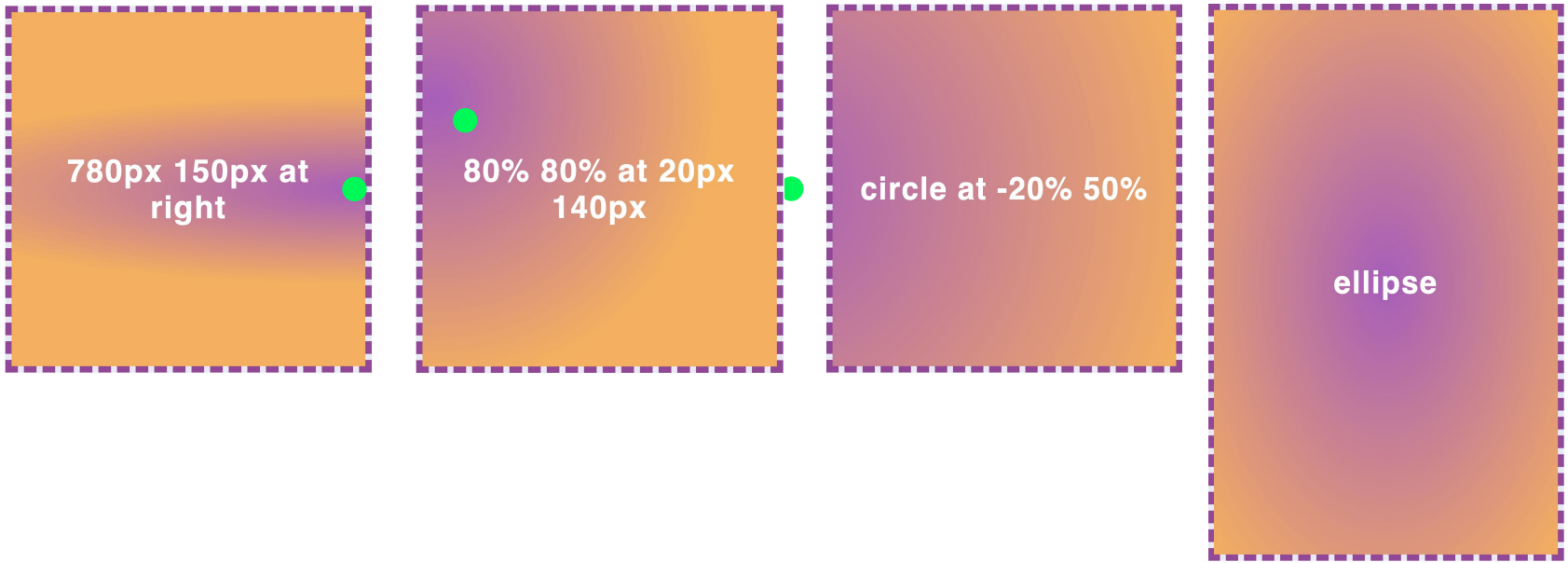
Direction:

- Specifies the position of the gradient's origin within the element.
- Common positions include **at center, at top, at bottom**, etc.

Colors with Transition Points:

- Defines the colors used in the gradient and where they transition from one to the next.
- Transition points are defined using percentages.

background radial gradient



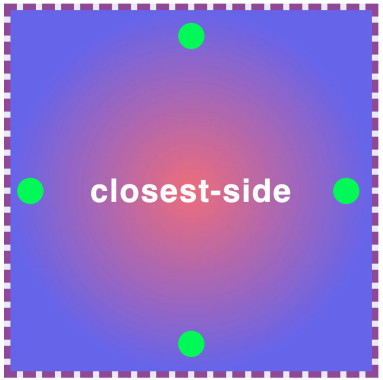
background radial gradient <size>

allows you to create gradients that radiate from a central point. The size of the gradient can be controlled to determine how it expands within the element

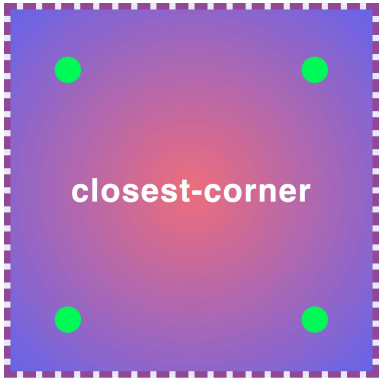
```
radial-gradient( circle closest-side at top, #9c27b0 0%, #ff9800 50%)
```

The gradient expands to the farthest side of the block. The transition extends until it reaches the edge farthest from the center.

The gradient expands to the farthest corner of the block. The transition extends until it reaches the farthest corner.



The gradient expands to the closest side of the block. The transition stops when it reaches the edge closest to the center.



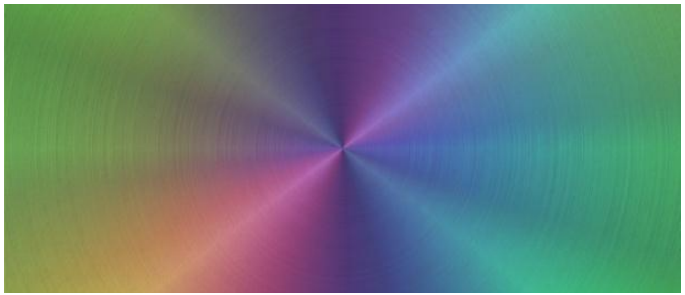
The gradient expands to the closest corner of the block. The transition stops when it reaches the closest corner.

background conic-gradient

create a gradient effect that rotates around a central point, much like the hands of a clock. This type of gradient is perfect for creating pie charts, color wheels, or unique background effects.

conic-gradient(angle and position, colors with transition points **)**

```
background-image:
     conic-gradient(
          from 90deg at 50% 50%,
          #A100FFFF 0%,
          #71C4FFFF 100%
);
```



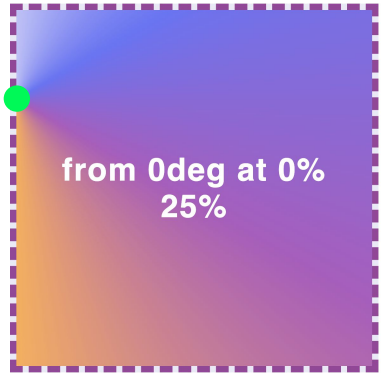
Angle and Position:

- Specifies the starting angle and the position of the gradient's center within the element.
- Angles are defined in degrees (**deg**).
- Positions can be defined using percentages or keywords like **at center, at top left**, etc.

Colors with Transition Points:

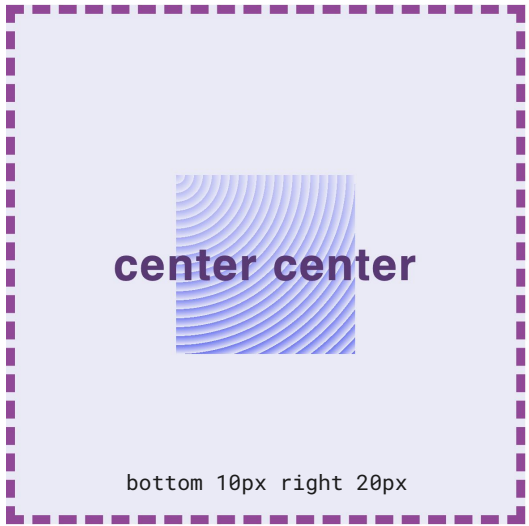
- Defines the colors used in the gradient and where they transition from one to the next.
- Transition points are defined using percentages.

background conic-gradient

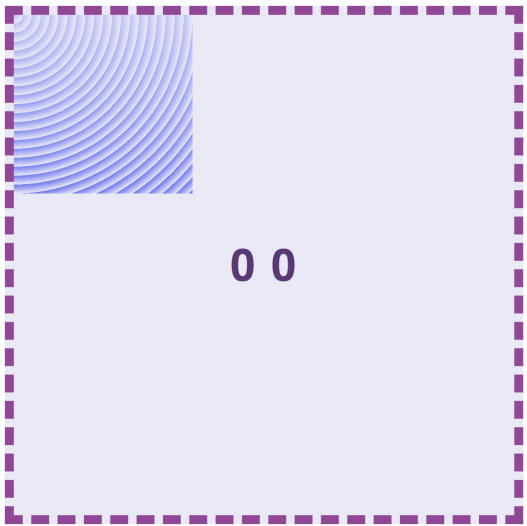


background-position

sets the initial position for each background image



keywords: **center, left, right, top, bottom**

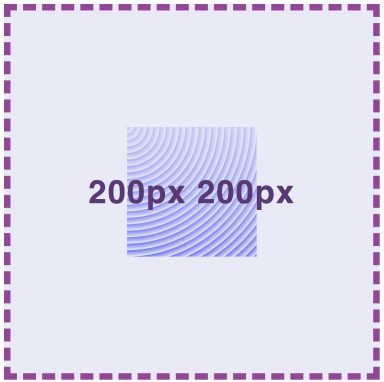


coordinates in **px**

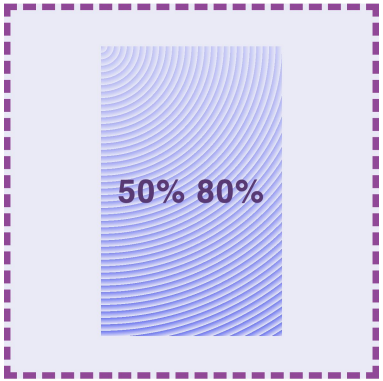
coordinates in **%**

background-size

sets the size of the element's background image



sizes in **px**



sizes in **%**

Scales the image as large as possible within its container without cropping or stretching the image.

Scales the image to the smallest possible size to fill the container, leaving no empty space.

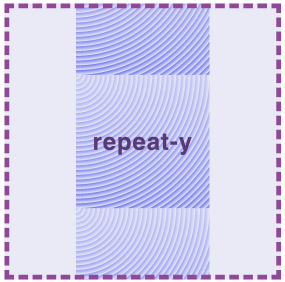
background-repeat

sets how background images are repeated. A background image can be repeated along the horizontal and vertical axes, or not repeated at all.

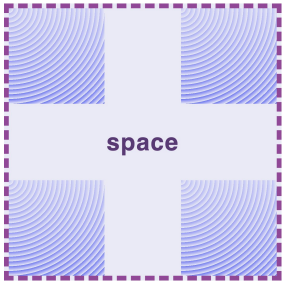
no repeat

repeat horizontally and vertically

repeat only horizontally



repeat only vertically



The image is repeated as much as possible without clipping.

As the allowed space increases in size, the repeated images will stretch (leaving no gaps) until there is room for another one to be added.

background-clip

determines how the background (color or image) is clipped, or where the background starts and stops. It controls whether the background extends under the border, padding, and content or just within specific areas.

The background extends to the edge of the padding, but not under the border. This is the default value.

The background extends only to the edge of the content, not into the padding or border.

The background extends to the outer edge of the border.

background-origin

determines the positioning area of a background image. It specifies whether the background image is placed relative to the border, padding, or content box of an element.

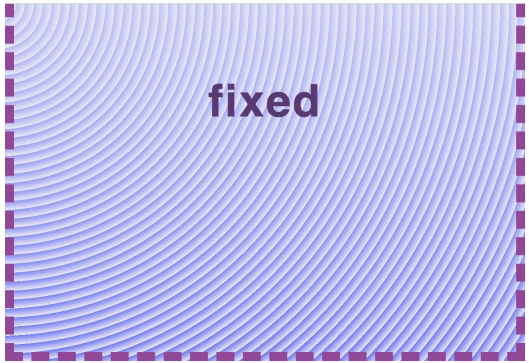
The background image starts from the edge of the padding. This is the default value.

The background image starts from the edge of the content. The image is positioned inside the padding area.

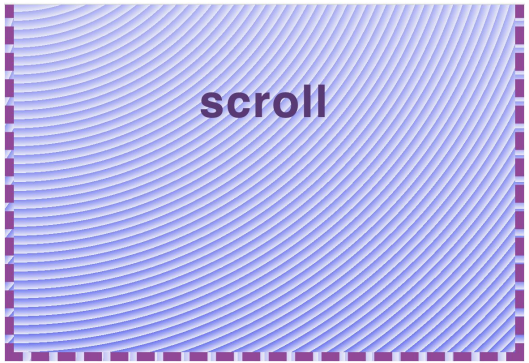
The background image starts from the outer edge of the border. The image extends beneath the border.

background-attachment

allows you to control whether a background image is fixed or scrolls with the rest of the content. This property can create interesting visual effects and improve the user experience on your website.



The background image is fixed in place and does not move when the page is scrolled.
This creates a parallax effect where the background stays stationary, giving the illusion of depth.



The background image scrolls with the rest of the content.
This is the default behavior where the background moves as you scroll down the page.

The background image scrolls along with the element's content if the element itself has a scrollbar.
This is useful for elements with overflow that can scroll independently of the page.

Multi backgrounds

```
background:
      url(cat-icon.png) bottom right / 20% auto no-repeat,
      linear-gradient(
         105deg,
         rgba(25, 25, 29, 0.5) 39%,
         rgba(185, 6, 27, 1) 96%
       ) center center / cover no-repeat,
      url(cat-photo.jpg) center / cover no-repeat;
```

```
background-image:
      url(cat-icon.png),
      linear-gradient( 105deg,rgba(25, 25, 29, 0.5) 39%,
rgba(185, 6, 27, 1) 96% ),
      url(cat-photo.jpg);
background-position: bottom right, center;
background-size: 20% auto, cover;
background-repeat: no-repeat;
```

Image set

Image set

Pixel destiny:

```
background-image: image-set(
  "small-image.png" 1x,
  "medium-image.png" 2x,
  "large-image.png" 3x
);
```

Image type:

```
background-image: image-set(
  "small-image.avif" type("image/avif"),
  "medium-image.png" type("image/webp")
);
```

CSS for <img>

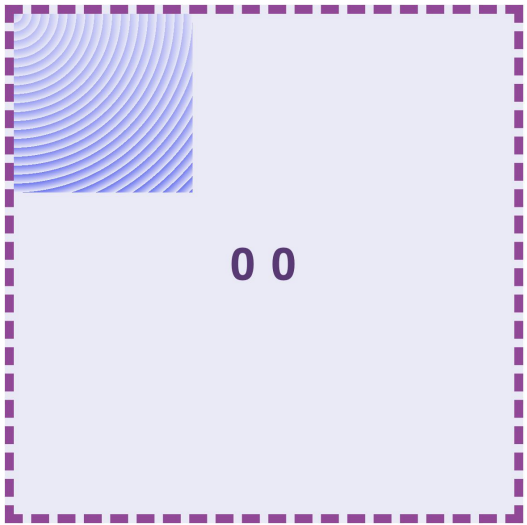
object-position

it’s like `background-position`

```
object-position: center;
```

specifies the alignment of the selected replaced element's contents within the element's box. Areas of the box which aren't covered by the replaced element's object will show the element's background.

keywords: **`center`, `left`, `right`, `top`, `bottom`**



coordinates in **px**

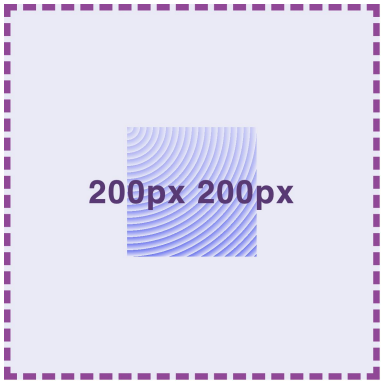
coordinates in **%**

object-fit

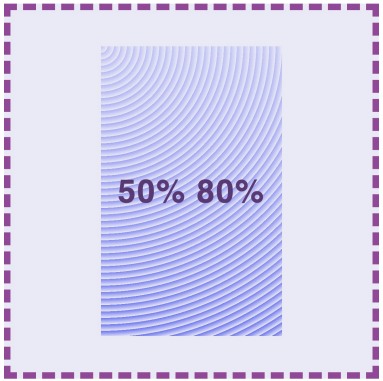
it's like `background-size`

`object-position: contain;`

sets how the content of a replaced element, such as an `<img>` or `<video>`, should be resized to fit its container.



sizes in **px**



sizes in **%**

Scales the image as large as possible within its container without cropping or stretching the image.

Scales the image to the smallest possible size to fill the container, leaving no empty space.

aspect-ratio

```
aspect-ratio: 16 / 9;
aspect-ratio: 16 / 9;
aspect-ratio: 1;
```

property allows you to define the desired width-to-height ratio of an element's box. This means that even if the parent container or viewport size changes, the browser will adjust the element's dimensions to maintain the specified width-to-height ratio. The specified aspect ratio is used in the calculation of auto sizes and some other layout functions.