icon-basics

- Icons Cleanup
  - What exactly is done? {#process}
  - Naming convention
  - Validation
  - Clean up and optimisation
  - Palette
    - Hardcoded palette
    - Monotone icons
      - Do not set fill! {#fill}
    - Mixing currentColor and custom colors
  - Format
    - Custom type
  - Icon sets


```
title: "Getting Started: Icons Cleanup"
types:
  IconifyJSON: "/docs/types/iconify-json.md"
  IconifyIcon: "/docs/types/iconify-icon.md"
functions:
  getIconData: "/docs/libraries/utils/get-icon-data.md"
  parseIconSet: "/docs/libraries/utils/parse-icon-set.md"
  defaultIconProps: "/docs/libraries/utils/default-icon-props.md"
```

Icons Cleanup

One of goals of Iconify project is to make icons easy to use.

This is not as simple as running icons through popular optimisation tools. Icon parsing in Iconify is very strict and process is much more invasive.

Icons are modified, so they can be used the same way regardless of icon set.

What exactly is done? {#process}

Each icon is:

- Validated (see below).
- Cleaned up and optimised (see below).
- Palette is fixed to either contain `[prop]currentColor` or hardcoded palette.
- If needed, icons are renamed to follow Iconify naming convention (see below).

Naming convention

To make icons easy to use regardless of their source, all icons follow the same naming convention.

Allowed characters in icon names are `[str]a-z`, numbers and `[str]-`. Hyphen cannot be used at start or end of name, 2 hyphens in a row `[str]--` are not allowed.

Examples of valid icon names: `[str]home-outline`, `[str]1st-place-medal`, `[str]camera-with-flash`.

Validation

When icons are imported and validated, icons that contain any of the following are not allowed:

- Scripts and event listeners. They can be harmful or badly coded, thus they do not belong in public icons.
- Raster images. They don't scale, so they do not belong in vector icons.
- External resources. Icon should not rely on something hosted elsewhere.
- Text. Texts are rendered differently in different browsers and operating systems. Icons should look identical to all visitors. Convert text to shapes.

Additionally, everything that does not affect icon rendering is removed. Many bad editors leave a lot of junk code in generated SVG, all of that is removed.

This is done during import process for open source icon sets using Iconify Tools. Custom icon sets might not follow the same guidelines.

Clean up and optimisation

Clean up and optimisation process removes all unnecessary code, making icon as small as possible, without breaking icon.

For more details about icon cleanup, see icons clean up process.

Palette

Palette is always checked and fixed if necessary.

What exactly is done depends on type of icon.

There are 2 types of icons:

- Icons with hardcoded palette, such as emojis. Colors cannot be changed.
- Monotone icons. They have only one color that can be changed.

What is the difference?

Hardcoded palette

Colors in icons with hardcoded palette cannot be changed.

Examples:

```
include: common/icon-types-palette
```

In CSS these icons are used as background images.

Palette in such icons is kept as is.

Monotone icons

All monotone icons are modified to use `[prop]currentColor` for color.

This makes it easy to change icon color by changing text color in CSS.

Examples:

```
include: common/icon-types-monotone
```

Hover samples above to see color change.

Why not `[prop]fill`? Main reason is because many icons use `[prop]stroke` for color. Using `[prop]fill` to set color is bad practice because it limits your icons to only icons that use `[prop]fill`. It also prevents icons from being used as mask images because background color cannot be set to fill color.

In CSS these icons are used as mask images, see using SVG in CSS for details.

Do not set fill! {#fill}

Some icon sets recommend changing color by changing `[prop]fill`, but you should never do that with Iconify!

This is a very bad practice because not all icons use fill. Many icons use `[prop]stroke`, but you cannot change `[prop]stroke` because it would add stroke to icons that do not have `[prop]stroke`.

To avoid all that mess, Iconify changes color in monotone icons to `[prop]currentColor`, so you can change icon color by changing text color, regardless of how icon is structured.

Mixing currentColor and custom colors

While it is possible to mix monotone and hardcoded palette, it is a very bad idea. Icon with mix of `[prop]currentColor` and hardcoded colors is unusable because:

- It cannot be used in CSS. If it is used as a background image, `[prop]currentColor` becomes black. If it is used as a mask image, custom palette disappears.
- It works only with a specific color scheme, usually designed only for white background.

Therefore, icons with mixed palette are not allowed in Iconify open source icon sets repository.

Format

Icons are stored converted to `[type]IconifyIcon` format, then stored in icon sets in `[type]IconifyJSON` format.

Data is stored in easy to use JSON files.

Custom type

Why is icon stored in custom `[type]IconifyIcon` type instead of SVG?

Format splits icon attributes for `[tag]svg` element and content. This it makes it easy to alter SVG without parsing XML: resize, change `[prop]viewBox`, render as components without parsing JSX, append/prepend shapes to icon, etc.

Icon sets

Multiple icons are stored in an icon set. See icon sets documentation for details.

□□□ #2
□ □□□ □□□ 2026-10-31 14:23:57 UTC
□ □□□ □□□ 2026-10-31 14:23:58 UTC