

mask



- [Converting SVG to mask](#)
 - [How masks work](#)
 - [Usage](#)
 - [False result](#)
 - [Options](#)
 - [Solid and transparent](#)
 - [Custom](#)
- [Examples](#)
 - [Fix black/white](#)
 - [Converting as is](#)
 - [Multiple colors](#)
 - [Advanced example](#)

```
title: Convert SVG to mask
types:
  SVG: '../svg/index.md'
  IconSet: '../icon-set/index.md'
functions:
  #parseColors: './colors/parse.md'
```

Converting SVG to mask

This function is part of [icon manipulation functions](#) in [Iconify Tools](#).

Function `[func]convertSVGTOMask()` converts icon content to an alpha mask and masks a rectangle that has the same dimensions as icon's `[prop]viewBox`.

This function is intended to be used in two scenarios:

- When changing icon with multiple colors to a monotone icon, using different shades for different colors.
- When fixing badly designed icon, which has white shapes instead of transparent parts.

How masks work

Masks help in a situation when there are multiple shapes, some of which should be solid, some transparent. Image in a mask is treated as an alpha channel for shape that is being masked.

Color in shapes is treated as an alpha channel: white (255) has opacity of 1, black (0) has opacity of 0. Other colors are converted to alpha channel by combining their red, green and blue channels are dividing it by 255×3 .

For example, this icon has multiple shapes with different colors (using black background to make sure white shapes are visible):

Here is the same icon used as a mask for rectangle, rendered using various colors:

Code used to create this sample is available below in the examples section, in "Converting as is" demo.

Usage

Function has the following parameters:

- `[prop]svg`, `[type]SVG`. Icon instance.
- `[prop]options`, `[type]object`. Optional options.

Function returns `true` on success, `false` if icon was not converted to mask.

False result

When can function fail to convert icon to mask?

- If icon contains unexpected color(s). Make sure all colors are handled by `[prop]solid`, `[prop]transparent` and/or `[prop]custom` options.
- If there is nothing to mask (icon has one color) and `[prop]force` option is not enabled. Use `[func]parseColors` to change color in such icons.

Options

Options is an object with the following optional properties:

- `[prop]id`, `[type]string`. Mask id, default value is `[str]mask`.
- `[prop]force`, `[type]boolean`. If enabled, forces function to apply mask even if there is nothing to mask.
- `[prop]color`, `[type]string`. Color to fill mask, default value is `[str]#000`.

To tell function which colors should be treated as solid, transparent or semi-transparent, use the following options:

- `[prop]solid`, `[type]string | string[] | function`. Solid colors: string, array of strings or a callback.
- `[prop]transparent`, `[type]string | string[] | function`. Transparent colors: string, array of strings or a callback.
- `[prop]custom`, `[type]function`. Callback that should return mask color or opacity for a color found in icon.

By default, black color is considered to be solid, white color is considered to be transparent (opposite of how masks work, but exactly how most bugged icons are designed), all other colors found in an icon will cause function to fail.

When using `[prop]custom` option, you might want to disable default values for `[prop]solid` and `[prop]transparent` options if `[prop]custom` does not handle unknown colors. To do that, set them to empty arrays.

Solid and transparent

`[prop]solid` and `[prop]transparent` can be:

- `[type]string`, such as `[str]#fff`
- `[type]string[]`, such as `[js]("#fff", "black")`
- `[type]function`, which takes color string as parameter and returns `true` (if color should be solid/transparent) or `false` (if color should not be solid/transparent)

Color values and color passed to callback are always in lower case. So, if your icon uses `[str]#FFF`, check for `[str]#fff`.

Custom

`[prop]custom` property is used when both `[prop]solid` and `[prop]transparent` return `false`.

It is a function with 2 parameters:

- Color as `[type]string`, in lower case, such as `[str]#fff`.
- Color as `[type]Color` instance.

Function can return:

- Color as `[type]string`, such as `[str]#fff` for solid shapes, `[str]#000` for transparent shapes, any other valid color for semi-transparent shapes.
- Opacity as `[type]number` between 0 and 1. Function will convert opacity returned by callback to a grayscale color.

Default values for `[prop]solid` and `[prop]transparent` properties might interfere with `[prop]custom` property, so set them to either an empty array or a callback that always returns `false`.

Examples

The following examples show how to use function.

Fix black/white

This example fixes badly designed icons, which use black color for solid shapes, white color for transparent shapes.

Such icons are widespread in icon sets available in Figma community resources: designers use white background for pages and don't notice that shapes contain white colors because outside of an icon background is also white.

```
src: libraries/tools/icon/mask-white.ts
title: 'example.ts'
extra:
  - src: libraries/tools/icon/mask-white.svg
    title: 'Result:'
```

If instead of black color, icon is designed using another color, such as `[str]#1c274c`, add it to `[prop]solid` option in lower case:

```
convertSVGMask(svg, {
  solid: ['#000', 'black', '#1c274c']
});
```

Converting as is

This example masks an icon without changing any colors. It is used to create mask sample used at the beginning of this page:

```
src: libraries/tools/icon/mask-demo.ts
title: 'example.ts'
```

Demo of this icon is available above in "How masks work" section.

Multiple colors

Sometimes icons are designed with multiple colors. This function can be used to convert them to monotone icons, using different colors as solid, semi-transparent or transparent colors.

This can be done using `[prop]custom` option.

Example (using gray background for original icon to make sure both black and white shapes are visible):

The following code converts icon to monotone icon, rendering black shapes as solid, blue and white as different semi-transparent:

```
src: libraries/tools/icon/mask-colors.ts
title: 'example.ts'
```

Result:

Advanced example

How to parse an entire icon set and check for errors?

This can be done using functions `[func]forEach()` or `[func]forEachSync()` of `[type]IconSet` instance. Both functions are suitable, but because masks are applied synchronously, there is no reason to use async functions, so `[func]forEachSync()` is a better choice.

See [very long example for importing icon set from Figma](#) for code.

In that example, `[func]convertSVGToMask()` is used after all colors are parsed with `[func]parseColors()`. If, when parsing colors, white or semi-transparent colors are found, `[func]convertSVGToMask()` is called to convert icon to mask.

□□ #2

□ □□ □□ 2026-10-31 14:25:52 UTC

□ □□ □□ 2026-10-31 14:25:53 UTC