get-icons-css

- getIconsCSS()
  - Color
  - Usage
  - Options
  - Result
  - Color option
  - Selectors
    - Simple selector
    - Old browsers
    - Wide and thin icons
    - Custom selectors
      - Using only 1 option
    - Pseudo-selector
  - Example


```
title: getIconsCSS() in Iconify Utils
functions:
  getIconCSS: './get-icon-css.md'
  getIconsContentCSS: './get-icons-content-css.md'
types:
  IconifyIcon: '/docs/types/iconify-icon.md'
  IconifyJSON: '/docs/types/iconify-json.md'
```

getIconsCSS()

This function is part of Iconify Utils package.

Function `[func]getIconsCSS()` generates stylesheet for several icons from an icon set to render them as background or mask images.

It generates code multiple icons from an icon set, splitting common code and icon-specific code. To generate code for one icon without splitting code, see `[func]getIconCSS()`.

If instead of using icons as background or mask images, you want to use icons as content of pseudo-elements, see `[func]getIconsContentCSS()`.

To use icons in HTML, all you need to do is create any element, such as `[tag]span` with class names for an icon set and icon.

Color

Monotone icons are rendered as mask image with background color set to `[prop]currentColor`. That means icon will use the same color as text.

To change icon color, simply change text color.

```
include: common/css-demo
```

Icons with palette are rendered as background image.

You can also use `[prop]color` option to convert monotone icon to icon with palette. See `[str]Color option` section below.

Usage

Function has the following parameters:

- `[prop]iconSet`, `[type]IconifyJSON`. Icon set data.
- `[prop]names`, `[type]string[]`. Array of icon names.
- `[prop]options`. Options object, optional.

Function returns `[type]string` with stylesheet for icons.

Options

The `[prop]options` object has the following properties:

- `[prop]iconSelector`, `[type]string`. Selector for icon, defaults to `[str].icon--{prefix}--{name}`. Variable `[str]{prefix}` is replaced with icon set prefix, `[str]{name}` with icon name.
- `[prop]commonSelector`, `[type]string`. Common selector for icons, defaults to `[str].icon--{prefix}`. Set it to empty to disable common code (see one of examples below). Variable `[str]{prefix}` is replaced with icon set prefix.
- `[prop]overrideSelector`, `[type]string`. Selector that mixes `[prop]iconSelector` and `[prop]commonSelector` to generate icon specific style that overrides common style. See below. Default value is `[str].icon--{prefix}.icon--{prefix}--{name}`.
- `[prop]pseudoSelector`, `[type]boolean`. Set it to `true` if selector for icon is a pseudo-selector, such as `[str].icon--{prefix}--{name}::after`.
- `[prop]varName`, `[type]string`. Name for variable to use for icon, defaults to `[str]svg` for monotone icons, `null` for icons with palette. Set to `null` to disable.
- `[prop]forceSquare`, `[type]boolean`. Forces icon to have width of `[num]1em`.
- `[prop]color`: `[type]string`. Sets color for monotone icons. Also renders icons as background images.
- `[prop]mode`: `[str]mask` or `[str]background`. Forces icon to render as mask image or background image. If not set, mode will be detected from icon content: icons that contain `[prop]currentColor` will be rendered as mask image, other icons as background image.
- `[prop]format`. Stylesheet formatting option. Matches options used in Sass. Supported values: `[str]expanded`, `[str]compact`, `[str]compressed`.
- `[prop]rules`, `[type]Record<string, string>`. Extra rules to add to CSS.

Result

Example of generated stylesheet:

```
/* Common code is combined in one class that should be added to each icon */
.icon--tabler {
  display: inline-block;
  width: 1em;
  height: 1em;
  background-color: currentColor;
  -webkit-mask: no-repeat center / 100%;
  mask: no-repeat center / 100%;
  -webkit-mask-image: var(--svg);
  mask-image: var(--svg);
}

/* SVG for each icon. Class name should be combined with common class name used above */
.icon--tabler--code {
```

```
--svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cpath fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2' d='m7 8l-4 4l4 4m10-8l4 4l-4 4M14 4l-4 16'/%3E%3C/svg%3E");
}

.icon--tabler--crystal-ball {
--svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cg fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2'%3E%3Cpath d='M6.73 17.018a8 8 0 1 1 10.54 0'/%3E%3Cpath d='M5 19a2 2 0 0 0 2 2h10a2 2 0 0 0 2-2v-4H7a2 2 0 0 0-2 2zm6-12a3 3 0 0 0-3 3'/%3E%3C/g%3E%3C/svg%3E");
}

.icon--tabler--view-360 {
--svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cg fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2'%3E%3Ccircle cx='12' cy='12' r='9'/%3E%3Cellipse cx='12' cy='12' rx='4' ry='9'/%3E%3Cpath d='M3 12c0 2.21 4.03 4 9 4s9-1.79 9-4s-4.03-4-9-4s-9 1.79-9 4z'/%3E%3C/g%3E%3C/svg%3E");
}
```

That code can be used in HTML with any element, such as `[tag]span` with class names for both common code and icon specific code:

```
<span class="icon--tabler icon--tabler--code"></span>
```

Color option

```
include: libraries/utils/css-color
```

```
<span
class="icon--tabler icon--tabler--view-360"
style="color: var(--icon-color)"
></span>
```

Selectors

Simple selector

If you want to change selectors to something simple, like `[str].icon-home` for `[str]home` icon, use the following options:

```
{
  "iconSelector": ".icon-{name}",
  "commonSelector": ""
}
```

It will result in stylesheet like this:

```
.icon-code,
.icon-crystal-ball,
.icon-view-360 {
  display: inline-block;
  width: 1em;
  height: 1em;
  background-color: currentColor;
  -webkit-mask: no-repeat center / 100%;
  mask: no-repeat center / 100%;
  -webkit-mask-image: var(--svg);
  mask-image: var(--svg);
}

.icon-code {
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cpath fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2' d='m7 8l-4 4l4 4m10-8l4 4l-4 4M14 4l-4 16'/%3E%3C/svg%3E");
}

.icon-crystal-ball {
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cg fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2'%3E%3Cpath d='M6.73 17.018a8 8 0 1 1 10.54 0'/%3E%3Cpath d='M5 19a2 2 0 0 0 2 2h10a2 2 0 0 0 2-2v-4H7a2 2 0 0 0-2 2zm6-12a3 3 0 0 0-3 3'/%3E%3C/g%3E%3C/svg%3E");
}
```

```
.icon-view-360 {
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cg fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2'%3E%3Ccircle cx='12' cy='12' r='9'/%3E%3Cellipse cx='12' cy='12' rx='4' ry='9'/%3E%3Cpath d='M3 12c0 2.21 4.03 4 9 4s9-1.79 9-4s-4.03-4-9-4s-9 1.79-9 4z'/%3E%3C/g%3E%3C/svg%3E");
}
```

Old browsers

If you need to support ancient browsers that do not support variables, set `[prop]varName` to `null`:

```
{
  "varName": null
}
```

It will result in CSS like this:

```
.icon--tabler {
  display: inline-block;
  width: 1em;
  height: 1em;
  background-color: currentColor;
  -webkit-mask: no-repeat center / 100%;
  mask: no-repeat center / 100%;
}

.icon--tabler--code {
  -webkit-mask-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cpath fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2' d='m7 8l-4 4l4 4m10-8l4 4l-4 4M14 4l-4 16'/%3E%3C/svg%3E");
  mask-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cpath fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2' d='m7 8l-4 4l4 4m10-8l4 4l-4 4M14 4l-4 16'/%3E%3C/svg%3E");
}

.icon--tabler--crystal-ball {
  -webkit-mask-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg'
```

```
viewBox='0 0 24 24' width='24' height='24'%3E%3Cg fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2'%3E%3Cpath d='M6.73 17.018a8 8 0 1 1 10.54 0'/%3E%3Cpath d='M5 19a2 2 0 0 0 2 2h10a2 2 0 0 0 2-2v-4a2 2 0 0 0-2-2H7a2 2 0 0 0-2 2zm6-12a3 3 0 0 0-3 3'/%3E%3C/g%3E%3C/svg%3E");
  mask-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' width='24' height='24'%3E%3Cg fill='none' stroke='black' stroke-linecap='round' stroke-linejoin='round' stroke-width='2'%3E%3Cpath d='M6.73 17.018a8 8 0 1 1 10.54 0'/%3E%3Cpath d='M5 19a2 2 0 0 0 2 2h10a2 2 0 0 0 2-2v-4a2 2 0 0 0-2-2H7a2 2 0 0 0-2 2zm6-12a3 3 0 0 0-3 3'/%3E%3C/g%3E%3C/svg%3E");
}
```

With monotone icons it will cause duplication because some browsers still require prefix for masks.

Not needed for icons for palette because default value for icons with palette is `null` (see example below that uses icons with palette).

Wide and thin icons

Not all icons are square. Some icon sets use variable width for icons, this mostly applies to icon sets that were designed to be used as fonts.

By default, common code will have `[prop]width` set to `[num]1em` and it will be overridden by using selectors with higher specificity for each icon:

```
.icon--fa6-solid {
  display: inline-block;
  width: 1em;
  height: 1em;
  background-color: currentColor;
  -webkit-mask: no-repeat center / 100%;
  mask: no-repeat center / 100%;
  -webkit-mask-image: var(--svg);
  mask-image: var(--svg);
}

.icon--fa6-solid.icon--fa6-solid--angle-left {
  width: 0.63em;
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 320 512' width='320' height='512'%3E%3Cpath fill='black' d='M41.4 233.4c-12.5 12.5-12.5 32.8 0 45.3l160 160c12.5 12.5 32.8 12.5 45.3 0s12.5-32.8 0-45.3L109.3 256l137.3-137.4c12.5-12.5 12.5-32.8 0-45.3s-32.8-12.5-45.3 0l-160 160z'/%3E%3C/svg%3E");
```

```
}

.icon--fa6-solid.icon--fa6-solid--arrow-left {
  width: 0.88em;
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 448 512'
width='448' height='512'%3E%3Cpath fill='black' d='M9.4 233.4c-12.5 12.5-12.5 32.8 0 45.3l160
160c12.5 12.5 32.8 12.5 45.3 0s12.5-32.8 0-45.3L109.2 288H416c17.7 0 32-14.3 32-32s-14.3-32-
32-32H109.3l105.3-105.4c12.5-12.5 12.5-32.8 0-45.3s-32.8-12.5-45.3 0l-160
160z'/%3E%3C/svg%3E");
}
```

But what if you want icons to be square? Use `[prop]forceSquare` option:

```
{
  "forceSquare": true
}
```

```
.icon--fa6-solid {
  display: inline-block;
  width: 1em;
  height: 1em;
  background-color: currentColor;
  -webkit-mask: no-repeat center / 100%;
  mask: no-repeat center / 100%;
  -webkit-mask-image: var(--svg);
  mask-image: var(--svg);
}

.icon--fa6-solid--angle-left {
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 320 512'
width='320' height='512'%3E%3Cpath fill='black' d='M41.4 233.4c-12.5 12.5-12.5 32.8 0 45.3l160
160c12.5 12.5 32.8 12.5 45.3 0s12.5-32.8 0-45.3L109.3 256l137.3-137.4c12.5-12.5 12.5-32.8 0-
45.3s-32.8-12.5-45.3 0l-160 160z'/%3E%3C/svg%3E");
}

.icon--fa6-solid--arrow-left {
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 448 512'
width='448' height='512'%3E%3Cpath fill='black' d='M9.4 233.4c-12.5 12.5-12.5 32.8 0 45.3l160
160c12.5 12.5 32.8 12.5 45.3 0s12.5-32.8 0-45.3L109.2 288H416c17.7 0 32-14.3 32-32s-14.3-32-
32-32H109.3l105.3-105.4c12.5-12.5 12.5-32.8 0-45.3s-32.8-12.5-45.3 0l-160
```

```
160z'/%3E%3C/svg%3E");
}
```

Custom selectors

If you want to use custom selectors, such as `[str].iconify-{name}` for icons, specify 3 options:

- `[prop]iconSelector` with value for icons, where you can use `[str]{prefix}` for icon set prefix and `[str]{name}` for icon name.
- `[prop]commonSelector` with value for common code, where you can use `[str]{prefix}` for icon set prefix. You can set it to empty or same value as `[prop]iconSelector`.
- `[prop]overrideSelector` with value for mix of selectors. Why is it a separate option? In case if mix selector is different from simply combining common and icon selectors.

If you set only `[prop]iconSelector`, other 2 options will be ignored.

Examples:

Using only 1 option

```
{
"iconSelector": ".iconify-{name}"
}
```

Results in:

```
.iconify-alien,
.iconify-balance-scale {
display: inline-block;
width: 1em;
height: 1em;
background: no-repeat center / 100%;
}

.iconify-alien {
background-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0
0 32 32' width='32' height='32'%3E%3Cg fill='none'%3E%3Cpath fill='%2386D72F' d='M29.84
15.92C29.84 23.61 18.92 30 15.92 30S2 23.61 2 15.92S8.23 2 15.92 2c7.69 0 13.92 6.23 13.92
13.92Z'/%3E%3Cpath fill='%23212121' d='M13.368 24.225a1 1 0 0 1 1.407.143c.376.461.825.645
1.225.645c.4 0 .85-.184 1.225-.645a1 1 0 1 1 1.55 1.264c-.724.889-1.726 1.38-2.775 1.38c-1.05
0-2.05-.491-2.775-1.38a1 1 0 0 1 .143-1.407Z'/%3E%3Cpath fill='%23533566' d='M6.474
```

```
19.527c2.254 2.253 4.6 3.549 6.845 1.306c2.253-2.242 2.233-5.905-.01-8.147c-2.244-2.243-5.898-2.253-8.142 0c-2.243 2.253-.946 4.588 1.307 6.84Zm19.051 0c-2.255 2.253-4.602 3.549-6.847 1.306c-2.244-2.242-2.234-5.905.01-8.147c2.245-2.243 5.9-2.253 8.144 0c2.245 2.253.947 4.588-1.307 6.84Z'/%3E%3Cpath fill='%23fff' d='M12.62 15.62c.61-.61.47-1.73-.31-2.51c-.78-.77-1.9-.92-2.51-.31c-.61.61-.47 1.73.31 2.51c.78.78 1.9.92 2.51.31Zm13.28 0c.61-.61.47-1.73-.31-2.51c-.77-.77-1.9-.92-2.51-.31c-.61.61-.47 1.73.31 2.51c.78.78 1.9.92 2.51.31Z'/%3E%3C/g%3E%3C/svg%3E");
}

.iconify-balance-scale {
background-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 32 32' width='32' height='32'%3E%3Cg fill='none'%3E%3Cpath fill='%239B9B9B' d='M19.949 11.75h4.02a.749.749 0 1 0 0-1.5h-4.02c-.38 0-.73-.18-.95-.48l-2.37-3.25a.765.765 0 0 0-.705-.296a.747.747 0 0 0-.585.306l-2.37 3.24c-.22.3-.58.48-.95.48h-4.02c-.41 0-.75.34-.75.75s.34.75.75.75h4.02c.85 0 1.66-.41 2.16-1.09l1.809-2.473l1.801 2.463c.5.69 1.31 1.1 2.16 1.1Z'/%3E%3Cpath fill='%23D3D3D3' d='M15.249 4h1.5v24h-1.5V4Zm-2.21 16.73c.09.15.26.23.42.23c.09 0 .18-.02.24-.08a.5.5 0 0 0 .15-.69l-5.372-8.338A.494.494 0 0 0 8 11.5c-.16 0-.45.27-.45.27l-5.39 8.38c-.07.08-.11.19-.11.31c0 .28.22.5.5.5c.17 0 .33-.09.42-.23l4.53-7.034V20.8c0 .28.22.5.5.5s.5-.22.5-.5v-7.105l4.54 7.035Zm16 0c.09.15.26.23.42.23c.09 0 .18-.02.24-.08a.5.5 0 0 0 .15-.69l-5.372-8.338A.494.494 0 0 0 24 11.5c-.16 0-.359.155-.45.27l-5.39 8.38c-.07.08-.11.19-.11.31c0 .28.22.5.5.5c.17 0 .33-.09.42-.23l4.53-7.034V20.8c0 .28.22.5.5.5s.5-.22.5-.5v-7.105l4.54 7.035Z'/%3E%3Cpath fill='%23F9C23C' d='M17.59 3.59a1.59 1.59 0 1 1-3.18 0a1.59 1.59 0 0 1 3.18 0Zm-8 7.41a1.59 1.59 0 1 1-3.181 0a1.59 1.59 0 0 1 3.18 0Zm-7.571 9.64c-.09-.32.15-.64.49-.64h10.98c.34 0 .58.32.49.64c-.74 2.56-3.13 4.36-5.98 4.36c-2.85 0-5.24-1.8-5.98-4.36Zm16 0c-.09-.32.15-.64.49-.64h10.98c.34 0 .58.32.49.64c-.74 2.56-3.13 4.36-5.98 4.36c-2.85 0-5.24-1.8-5.98-4.36ZM23.4 30c.48 0 .71-.58.36-.91A11.216 11.216 0 0 0 16 26c-3.01 0-5.74 1.17-7.76 3.09c-.35.33-.12.91.36.91h14.8Zm.6-17.41a1.59 1.59 0 1 0-.001-3.18a1.59 1.59 0 0 0 0 3.18Z'/%3E%3C/g%3E%3C/svg%3E");
}
```

Pseudo-selector

This example shows why `[prop]overrideSelector` is an option and how to render icons with pseudo-selectors:

```
{
"iconSelector": ".icon--{prefix}--{name}::after",
"commonSelector": ".icon--{prefix}::after",
"overrideSelector": ".icon--{prefix}.icon--{prefix}--{name}::after",
```

```
  "pseudoSelector": true
}
```

```
.icon--fa6-solid::after {
  display: inline-block;
  width: 1em;
  height: 1em;
  content: '';
  background-color: currentColor;
  -webkit-mask: no-repeat center / 100%;
  mask: no-repeat center / 100%;
  -webkit-mask-image: var(--svg);
  mask-image: var(--svg);
}

.icon--fa6-solid.icon--fa6-solid--angle-left::after {
  width: 0.63em;
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 320 512'
width='320' height='512'%3E%3Cpath fill='black' d='M41.4 233.4c-12.5 12.5-12.5 32.8 0 45.3l160
160c12.5 12.5 32.8 12.5 45.3 0s12.5-32.8 0-45.3L109.3 256l137.3-137.4c12.5-12.5 12.5-32.8 0-
45.3s-32.8-12.5-45.3 0l-160 160z'/%3E%3C/svg%3E");
}

.icon--fa6-solid.icon--fa6-solid--arrow-left::after {
  width: 0.88em;
  --svg: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 448 512'
width='448' height='512'%3E%3Cpath fill='black' d='M9.4 233.4c-12.5 12.5-12.5 32.8 0 45.3l160
160c12.5 12.5 32.8 12.5 45.3 0s12.5-32.8 0-45.3L109.2 288H416c17.7 0 32-14.3 32-32s-14.3-32-
32-32H109.3l105.3-105.4c12.5-12.5 12.5-32.8 0-45.3s-32.8-12.5-45.3 0l-160
160z'/%3E%3C/svg%3E");
}
```

Example

Above examples show various options. This example shows actual code with options:

```
src: libraries/utils/get-icons-css.ts
title: 'generate-css.ts'
extra:
```

```
  - src: libraries/utils/get-icons-css.css
    title: 'Result:'
  - src: libraries/utils/get-icons-css.html
    title: 'Usage in HTML:'
```

□□□ #2
□ □□□ □□□ 2026-10-31 14:28:40 UTC
□ □□□ □□□ 2026-10-31 14:28:42 UTC