AllThingsSmitty / **css-protips** Public

A collection of tips to help take your CSS skills pro

CC0-1.0 license

21.7k stars **1.8k** forks

Star

Notifications

Code Issues 6 Pull requests 7 Actions Projects Wiki master

Go to file

AllThingsSmitty Added :empty pseudo-class tip ... 14 days ago 340

View code

README.md

Releases

No releases published

Packages

No packages published

Contributors 35

+ 24 contributors

CSS Protips

A collection of tips to help take your CSS skills pro.

> For other great lists check out @sindresorhus's curated list of awesome lists.

Table of Contents

Protips

1. Use a CSS Reset
2. Inherit `box-sizing`
3. Use `unset` Instead of Resetting All Properties
4. Use `:not()` to Apply/Unapply Borders on Navigation
5. Check If Font Is Installed Locally
6. Add `line-height` to `body`

Use a CSS Reset

CSS resets help enforce style consistency across different browsers with a clean slate for styling elements. You can use a CSS reset library like Normalize, *et al.*, or you can use a more simplified reset approach:

```css
*,
*::before,
*::after {
  box-sizing: border-box;
  margin: 0;
  padding: 0;
}
```

Now elements will be stripped of margins and padding, and `box-sizing` lets you manage layouts with the CSS box model.

Demo

Note: If you follow the Inherit `box-sizing` tip below you might opt to not include the `box-sizing` property in your CSS reset.

back to table of contents

Inherit `box-sizing`

Let `box-sizing` be inherited from `html` :

```css
html {
  box-sizing: border-box;
}

*,
*::before,
*::after {
  box-sizing: inherit;
}
```

This makes it easier to change `box-sizing` in plugins or other components that leverage other behavior.

Demo

back to table of contents

Use `unset` Instead of Resetting All Properties

When resetting an element's properties, it's not necessary to reset each individual property:

```css
button {
  background: none;
  border: none;
  color: inherit;
  font: inherit;
  outline: none;
  padding: 0;
}
```

You can specify all of an element's properties using the `all` shorthand. Setting the value to `unset` changes an element's properties to their initial values:

```css
button {
  all: unset;
}
```

Note: the `all` and `unset` shorthand isn't supported in IE11.

back to table of contents

Use `:not()` to Apply/Unapply Borders on Navigation

Instead of putting on the border...

```
/* add border */
.nav li {
  border-right: 1px solid #666;
}
```

...and then taking it off the last element...

```
/* remove border */
.nav li:last-child {
  border-right: none;
}
```

...use the `:not()` pseudo-class to only apply to the elements you want:

```
.nav li:not(:last-child) {
  border-right: 1px solid #666;
}
```

Here, the CSS selector is read as a human would describe it.

Demo

back to table of contents

Check If Font Is Installed Locally

You can check if a font is installed locally before fetching it remotely, which is a good performance tip, too.

```
@font-face {
  font-family: "Dank Mono";
  src:
    /* Full name */
    local("Dank Mono"),
```

```css
    /* Postscript name */
    local("Dank Mono"),
    /* Otherwise, download it! */
    url("//...a.server/fonts/DankMono.woff");
}

code {
  font-family: "Dank Mono", system-ui-monospace;
}
```

H/T to Adam Argyle for sharing this protip and demo.

back to table of contents

Add `line-height` to `body`

You don't need to add `line-height` to each `<p>`, `<h*>`, *et al*. separately. Instead, add it to `body`:

```css
body {
  line-height: 1.5;
}
```

This way textual elements can inherit from `body` easily.

Demo

back to table of contents

Set `:focus` for Form Elements

Sighted keyboard users rely on focus to determine where keyboard events go in the page. Make focus for form elements stand out and consistent then a browser's default implementation:

```css
a:focus,
button:focus,
input:focus,
select:focus,
textarea:focus {
  box-shadow: none;
  outline: #000 dotted 2px;
  outline-offset: .05em;
}
```

Demo

back to table of contents

Vertically-Center Anything

No, it's not black magic, you really can center elements vertically. You can do this with flexbox...

```css
html,
body {
  height: 100%;
  margin: 0;
}

body {
  -webkit-align-items: center;
  -ms-flex-align: center;
  align-items: center;
  display: -webkit-flex;
  display: flex;
}
```

...and also with CSS Grid:

```css
body {
  display: grid;
  height: 100vh;
  margin: 0;
  place-items: center center;
}
```

Want to center something else? Vertically, horizontally...anything, anytime, anywhere? CSS-Tricks has a nice write-up on doing all of that.

Note: Watch for some buggy behavior with flexbox in IE11.

Demo

back to table of contents

Comma-Separated Lists

Make list items look like a real, comma-separated list:

```css
ul > li:not(:last-child)::after {
  content: ",";
}
```

Use the `:not()` pseudo-class and no comma will be added to the last item.

Note: This tip may not be ideal for accessibility, specifically screen readers. And copy/paste from the browser doesn't work with CSS-generated content. Proceed with caution.

back to table of contents

Select Items Using Negative `nth-child`

Use negative `nth-child` in CSS to select items 1 through n.

```css
li {
  display: none;
}

/* select items 1 through 3 and display them */
li:nth-child(-n+3) {
  display: block;
}
```

Or, since you've already learned a little about using `:not()`, try:

```css
/* select all items except the first 3 and display them */
li:not(:nth-child(-n+3)) {
  display: block;
}
```

Demo

back to table of contents

Use SVG for Icons

There's no reason not to use SVG for icons:

```css
.logo {
  background: url("logo.svg");
}
```

SVG scales well for all resolution types and is supported in all browsers back to IE9. Ditch your .png, .jpg, or .gif-jif-whatev files.

Note: If you have SVG icon-only buttons for sighted users and the SVG fails to load, this will help maintain accessibility:

```css
.no-svg .icon-only::after {
  content: attr(aria-label);
}
```

back to table of contents

Use the "Lobotomized Owl" Selector

It may have a strange name but using the universal selector (`*`) with the adjacent sibling selector (`+`) can provide a powerful CSS capability:

```css
* + * {
  margin-top: 1.5em;
}
```

In this example, all elements in the flow of the document that follow other elements will receive `margin-top: 1.5em`.

For more on the "lobotomized owl" selector, read Heydon Pickering's post on *A List Apart*.

Demo

back to table of contents

Use `max-height` for Pure CSS Sliders

Implement CSS-only sliders using `max-height` with overflow hidden:

```css
.slider {
  max-height: 200px;
  overflow-y: hidden;
  width: 300px;
}

.slider:hover {
  max-height: 600px;
  overflow-y: scroll;
}
```

The element expands to the `max-height` value on hover and the slider displays as a result of the overflow.

back to table of contents

Equal-Width Table Cells

Tables can be a pain to work with. Try using `table-layout: fixed` to keep cells at equal width:

```css
.calendar {
  table-layout: fixed;
}
```

Pain-free table layouts.

Demo

back to table of contents

Get Rid of Margin Hacks With Flexbox

When working with column gutters you can get rid of `nth-`, `first-`, and `last-child` hacks by using flexbox's `space-between` property:

```css
.list {
  display: flex;
  justify-content: space-between;
}

.list .person {
  flex-basis: 23%;
}
```

Now column gutters always appear evenly-spaced.

back to table of contents

Use Attribute Selectors with Empty Links

Display links when the `<a>` element has no text value but the `href` attribute has a link:

```css
a[href^="http"]:empty::before {
  content: attr(href);
}
```

That's pretty convenient.

Demo

back to table of contents

Style "Default" Links

Add a style for "default" links:

```css
a[href]:not([class]) {
  color: #008000;
  text-decoration: underline;
}
```

Now links that are inserted via a CMS, which don't usually have a `class` attribute, will have a distinction without generically affecting the cascade.

back to table of contents

Intrinsic Ratio Boxes

To create a box with an intrinsic ratio, all you need to do is apply top or bottom padding to a div:

```css
.container {
  height: 0;
  padding-bottom: 20%;
  position: relative;
}

.container div {
  border: 2px dashed #ddd;
  height: 100%;
  left: 0;
  position: absolute;
  top: 0;
  width: 100%;
}
```

Using 20% for padding makes the height of the box equal to 20% of its width. No matter the width of the viewport, the child div will keep its aspect ratio (100% / 20% = 5:1).

Demo

back to table of contents

Style Broken Images

Make broken images more aesthetically-pleasing with a little bit of CSS:

```
img {
  display: block;
  font-family: sans-serif;
  font-weight: 300;
  height: auto;
  line-height: 2;
  position: relative;
  text-align: center;
  width: 100%;
}
```

Now add pseudo-elements rules to display a user message and URL reference of the broken image:

```
img::before {
  content: "We're sorry, the image below is broken :(";
  display: block;
  margin-bottom: 10px;
}

img::after {
  content: "(url: " attr(src) ")";
  display: block;
  font-size: 12px;
}
```

Learn more about styling for this pattern in Ire Aderinokun's original post.

back to table of contents

Use `rem` for Global Sizing; Use `em` for Local Sizing

After setting the base font size at the root (`html { font-size: 100%; }`), set the font size for textual elements to `em` :

```
h2 {
  font-size: 2em;
}

p {
  font-size: 1em;
}
```

Then set the font-size for modules to `rem` :

```
article {
  font-size: 1.25rem;
}

aside .module {
  font-size: .9rem;
}
```

Now each module becomes compartmentalized and easier to style, more maintainable, and flexible.

back to table of contents

Hide Autoplay Videos That Aren't Muted

This is a great trick for a custom user stylesheet. Avoid overloading a user with sound from a video that autoplays when the page is loaded. If the sound isn't muted, don't show the video:

```
video[autoplay]:not([muted]) {
  display: none;
}
```

Once again, we're taking advantage of using the `:not()` pseudo-class.

back to table of contents

Use `:root` for Flexible Type

The type font size in a responsive layout should be able to adjust with each viewport. You can calculate the font size based on the viewport height and width using `:root`:

```css
:root {
  font-size: calc(1vw + 1vh + .5vmin);
}
```

Now you can utilize the `root em` unit based on the value calculated by `:root`:

```css
body {
  font: 1rem/1.6 sans-serif;
}
```

Demo

back to table of contents

Set `font-size` on Form Elements for a Better Mobile Experience

To avoid mobile browsers (iOS Safari, *et al*.) from zooming in on HTML form elements when a `<select>` drop-down is tapped, add `font-size` to the selector rule:

```css
input[type="text"],
input[type="number"],
select,
textarea {
  font-size: 16px;
}
```

💃

back to table of contents

Use Pointer Events to Control Mouse Events

Pointer events allow you to specify how the mouse interacts with the element it's touching. To disable the default pointer event on a button, for instance:

```css
.button-disabled {
  opacity: .5;
  pointer-events: none;
}
```

It's that simple.

back to table of contents

Set `display: none` on Line Breaks Used as Spacing

As Harry Roberts pointed out, this can help prevent CMS users from using extra line breaks for spacing:

```
br + br {
  display: none;
}
```

back to table of contents

Use `:empty` to Hide Empty HTML Elements

If you have HTML elements that are empty, i.e., the content has yet to be set either by a CMS or dynamically injected (e.g., `<p class="error-message"></p>`) and it's creating unwanted space on your layout, use the `:empty` pseudo-class to hide the element on the layout.

```
:empty {
  display: none;
}
```

Note: Keep in mind that elements with whitespace aren't considered empty, e.g., `<p class="error-message"> </p>`.

back to table of contents

Support

Current versions of Chrome, Firefox, Safari, Opera, Edge, and IE11.

back to table of contents

Translations

Note: I've had less time available to maintain the growing list of translated tips; adding a new tip requires including it with over a dozen translations. For that reason, translated README files may not include all the tips listed on the main README file.

- 简体中文
- 正體中文
- Deutsch
- Español
- Français
- λληνικά
- ગુજરાતી
- Italiano
- 日本語
- 한국어
- Polskie
- Português do Brasil
- Português do Europe
- Русский
- Tiếng Việt

back to table of contents