Toggles in CSS

Google is prototyping an unofficial CSS Toggles specification. The specification defines a way to associate a `toggleable value` with an element which developers can use to select an element, and declarative ways to set and modify this value on the element.

> **Warning**
>
> This is not a complete specification that is ready to use in production. There are some serious issues still remaining, including accessibility concerns.

The HTML is just a basic button.

We also use a `script` tag to load the CSS toggles polyfill from a CDN. In a more rigurous test I would download the polyfill and link to it from the page using it.

```
<button>TOGGLE ME</button></button>t src="https://unpkg.com/@oddbird/css-t
</script>
```

CSS toggles have three components.

toggle-root : Creates toggles on an element and controls how the toggles are updated when they are activated

toggle-trigger : Specifies that an element can be activated to change the value of one or more toggles

:toggle() : The pseudo-class matches if the element is in scope for a toggle with the name given as the value of the pseudo-class, and either the (1) toggle matches the provided value, or (2) the value is omitted and the toggle is in any active value

In the example below we use the properties as follows:

1. Define the `lightswitch` toggle in the html element

2. Use the button element to trigger the toggle
3. Specify what changes to make when the trigger is activated

```
/* 1 */
html {
  toggle-root: lightswitch;
}

/* 2 */
button {
  toggle-trigger: lightswitch;
  background-color: lightblue;
}

/* 3 */
html:toggle(lightswitch) button {
  background-color: rebeccapurple;
  color: white;
}
```

In The Future of CSS: CSS Toggles Bramus mentions additional things to consider:

- A Toggle Root can host more than one toggle
- A toggle can have more than 1 active state, so it's not only 0 and 1
- The states don't need to be numbers, but can also be a set of words
- The initial toggle state is 0 by default, but you can override it
- An element can be both the `toggle-root` and the `toggle-trigger`. In that case, use the toggle property
- The scope of a toggle value can be narrowed down to descendant elements only (instead of being visible to siblings and their descendants)
- `toggle-visibility` allows you to show/hide an element based on a toggle value. Think of details/summary and scenarios where you would rely on the checkbox checkbox hack. Benefit to using `toggle-visibility` is that the contents of the element are always available to the UA for in-page-find and similar tasks
- Toggles can be grouped using `toggle-group`. Useful for tab interfaces, where only 1 tab can be active at the same time

The example in this post is the simplest possible one. It relies on defaults and implicit assumptions. There is a lot to explore but those experiments may or may

not be covered by the polifill yet.