Interaction design

Tuur Vanhoutte

21 september 2020

Inhoudsopgave

1 CSS Variables

1.1 Wat?

- CSS custom properties for cascading variables.
- CSS Variables = Custom Properties.
- Relatief nieuwe manier om veelgebruikte values om te zetten naar Variables.
- To keep consistency, set global variables for everything except layout values.

1.2 Opbouw & Syntax

1.2.1 Custom property: defining the variable

Custom property: value

```
--my-cool-color: HotPink;
```

[language=CSS]

1.2.2 Cascading variable: applying the variable

Applying your custom property using the var() function

```
var(--my-cool-color)
```

[language=CSS]

1.3 Syntax

```
:root {
    --my-cool-color: HotPink;
}

p {
    color: var(--my-cool-color);
}

.foo {
    background-color: var(--my-cool-color);
}
```

[language=CSS]

1.3.1 CSS Variables Are Case Sensitive

```
:root {
    --foo: HotPink;
    --FOO: #BADA55;
}
```

```
p {
    color: var(--foo);
}
.foo {
    background-color: var(--FOO);
}
```

[language=CSS]

1.3.2 This is wrong

```
.test {
    --side: margin-top;
    var(--side): 20px
}
```

[language=CSS]

1.3.3 Use the calc() function to do math

```
:root {
    --whitespace: 20px;
    --whitespace-lg: var(--whitespace) * 2; /* won't work */
    --whitespace-lg: calc(var(--whitespace) * 2); /* correct */
}
```

[language=CSS]

1.3.4 Kan ook shorthand values bevatten

```
:root {
    --transition: all .1s ease-out;
}
a {
    transition: var(--transition);
}
```

[language=CSS]

1.3.5 Kan ook bestaan uit andere variables.

```
:root {
    --transition-property: all;
    --transition-duration: .1s;
    --transition-timing-function: ease-out;
    --transition: var(--transtion-property)
        var(--transition-duration)
        var(--transition-timing-function);
}

a {
```

```
    transition: var(--transition);
}
```

[language=CSS]

1.3.6 Default values

```
.c-button {
    border: 1px solid var(--button-color, HotPink);
    background-color: transparent;
}
.c-button:hover {
    background-color: var(--button-color, HotPink);
}
.c-button--beta {
    --button-color: #BADA55;
}
```

[language=CSS]

1.3.7 Default values with other variables

```
:root {
    --color-pink: HotPink;
    --color-badass: #BADA55;
}
.c-button {
    border: 1px solid var(--button-color, var(--color-pink));
}
.c-button:hover {
    background-color: var(--button-color, var(--color-pink));
}
.c-button--beta {
    --button-color: var(--color-badass);
```

[language=CSS]

1.4 Cascade

CSS Variables Are Subject to the Cascade and Inheritance rules

`https://codepen.io/simoncoudeville-nmct/pen/BaBMGPZ`

```
:root {
    --my-cool-color: HotPink;
}
/* Alle elementen binnen de root waar je --
my-cool-color variable toepast zullen de
value HotPink overerven. */

p {
color: var(--my-cool-color);
}

```

```
.foo {
/* Elke paragraaf in het element met
de class ".foo" krijgt de kleur #BADA55.*/

--my-cool-color: #BADA55;
}
```

[language=CSS]

https://codepen.io/simoncoudeville-nmct/pen/aboMBop

1.4.1 CSS variables can be made conditional with @media and other conditional rules

```
:root {
    --whitespace: 1em;
}
@media screen and (min-width: 768px) {
    :root {
        --whitespace: 2em;
    }
}
.c-card {
    padding: var(--whitespace);
}
```

[language=CSS]

https://codepen.io/simoncoudeville-nmct/pen/JjXaQwB

1.4.2 Ideal for dark themes

```
:root {
    --color: white;
    --background-color: black
}
@media (prefers-color-scheme: dark) {
    :root {
    --color: black;
    --background-color: white
    }
}
.html {
    background-color: var(--background-color);
    color: var(--color);
}
```

[language=CSS]

https://codepen.io/simoncoudeville-nmct/pen/eYOxbPp

1.5 Hoisting

= Accessing a Variable First and Declaring Later

```
body{
    background: var(--bg-fill);
}
:root{
    --bg-fill: green;
}
```

[language=CSS]

1.6 Scoped variables

Twee soorten:

- Global Scoped Variables
- Local Scoped Variables

1.6.1 Global scoped variables

```
:root {
    --global-color: black;
}
```

[language=CSS]

- :root is a CSS pseudo-class selector used to select the element that represents the root of the document.
- :root is hetzelfde als html maar is specifieker
- :root = html
- :root > html

`https://codepen.io/simoncoudeville-nmct/pen/OJLBKXq`

Global variables kan je overal hergebruiken en overschrijven. Ideaal dus voor values die veel hergebruikt worden:

- colors
- whitespace
- border-radius
- transitions
- ...

1.6.2 Local scoped variables

- Local scoped variables worden gedeclareerd binnen een specifieke selector.
- Local scoped variables hebben access tot global scoped variables.
- Ideaal voor components.

```css
.alert {
    --alert-color: #222;
    color: var(--alert-color);
    border-color: var(--alert-color);
}
:root {
    --global-fontSize: 16px;
}
.c-button {
    --button-fontSize: var(--global-fontSize);
    font-size: var(--button-fontSize);
}
```

[language=CSS]

1.7 Naming system

1.7.1 The two-level theming system

Systeem om global variables te gebruiken in local variables.

1.7.2 Global level

De hoofdreden om global variables te hebben is consistentie.

They are prefixed with the word global and follow this formula:

--global--concept--modifier--state--propertyCamelCase

- a concept is something like a spacer, main-title or text
- a state is something like hover, or expanded
- a modifier is something like sm, or lg
- and a propertyCamelCase is something like backgroundColor or fontSize

1.7.3 Local level

They follow this formula:

--block__element--modifier--state--propertyCamelCase

- The block__element--modifier selector name is something like alert__actions or alert–primary
- a state is something like hover or active
- The value of component scoped variables is always defined by a global variable.

```css
.c-alert {
    /* Component scoped variables are always defined by global variables */
    --c-alert--Padding: var(--global--spacer--md);
    --c-alert--primary--BackgroundColor: var(--global--primary-color);
    --c-alert__title--FontSize: var(--global--secondary-title--fontSize);
    /* --block--propertyCamelCase */
    padding: var(--c-alert--padding);
}

/* --block--state--propertyCamelCase */
```

```
.c-alert--primary {
    background-color: var(--c-alert--primary--backgroundColor);
}

/* --block__element--propertyCamelCase */
.c-alert__title {
    font-size: var(--c-alert__title--fontSize);
}
```

[language=CSS]

1.8 Herhalingsvragen

- Hoe worden CSS variables nog genoemd?
- Hoe worden CSS variables opgebouwd?
- Wat is CSS Hoisting?
- Wat is het verschil tussen global en local variables?

`https://codepen.io/simoncoudeville-nmct/pen/vYBbbXz?editors=1100`