**Директива @each**

Директива @each - це цикл, який виконає тіло директиви для кожного значення зі списку. Це дозволяє скоротити кількість коду, який необхідно написати вручну:

@each <значення> in <список значень> {

*// Тіло директиви @each*

}

Наприклад, необхідно створити серію селекторів для елементів з фоновими зображеннями. В CSS це виглядає наступним чином. Занадто багато повторюваного коду:

main.css

.dog-icon {

 background-image: url('../images/dog.png');

}

.cat-icon {

 background-image: url('../images/cat.png');

}

.bird-icon {

 background-image: url('../images/bird.png');

}

.turtle-icon {

 background-image: url('../images/turtle.png');

}

Використовуючи директиву @each, можемо досягнути такого самого результату, записавши наступний Sass-код, який буде скомпільований в CSS з попереднього прикладу:

main.scss

@each $animal in dog, cat, bird, turtle {

 .#{$animal}-icon {

   background-image: url('../images/#{$animal}.png');

 }

}

Директива @each встановлює $animal у кожне зі значень списку тварин і для кожного створює код, записаний в тілі директиви - CSS-правило з селектором класу і набором властивостей:

* $animal - локальна змінна, в яку будуть послідовно присвоюватися значення зі списку тварин. Ім'я змінної може бути довільним. Ми перебираємо список тварин, тому назвали animal - тварина.
* dog, cat, bird, turtle - список тварин, кожен елемент якого розділений пробілом з комою.

\**ІНТЕРПОЛЯЦІЯ*: для того щоб підставити значення змінної $animal, необхідно використовувати інтерполяцію #{$ім'я\_змінної}, оскільки ми створюємо рядки. Без інтерполяції ім'я змінної не буде замінено на її значення під час компіляції.

**Арифметичні операції**

До того як в CSS з'явилася функція calc(), препроцесори були єдиним рішенням за потреби виконати арифметичні обчислення.

**5.1) Додавання і віднімання**[**​**](https://textbook.edu.goit.global/html-css-a7u5xv/v1/uk/docs/lesson-14/math#%D0%B4%D0%BE%D0%B4%D0%B0%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F-%D1%96-%D0%B2%D1%96%D0%B4%D0%BD%D1%96%D0%BC%D0%B0%D0%BD%D0%BD%D1%8F) - на відміну від функції calc(), в препроцесорі не можна змішувати типи одиниць. При спробі виконати додавання або віднімання несумісних типів, виникне помилка. Пікселі до пікселів, слони до слонів.

main.scss

.box {

 width: 960px + 10%; *// Помилка!*

 width: 960px + 30px; *// 990px*

 width: 960px + 30; *// 990px*

 width: 100% + 20%; *// 120%*

 width: 100% - 50px; *// Помилка!*

 width: 960px - 30px; *// 930px*

 width: 960px - 30; *// 930px*

 width: 100% - 20%; *// 80%*

}

Справа в тому, що препроцесор не знає наперед, скільки буде 100% або 5em у пікселях. Значення відносних одиниць в пікселях можна дізнатися тільки в момент рендеру HTML-сторінки. Тому для таких обчислень необхідно використовувати нативну функцію calc().

main.scss

.box {

 width: calc(100% - 20px);

 width: calc(5em + 20px);

}

**5.2) Множення**[**​**](https://textbook.edu.goit.global/html-css-a7u5xv/v1/uk/docs/lesson-14/math#%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F) виконується аналогічно функції calc() в CSS, за винятком того, що не можна множити несумісні типи. Також не можна множити слонів на слонів, можна тільки слонів на множник.

main.scss

.box {

 width: 50px \* 50px; *// Помилка!*

 width: 50% \* 5%; *// Помилка!*

 width: 50px \* 2; *// 100px*

 width: 10px \* 2 + 50px; *// 70px*

 width: 5 \* (5px + 15px); *// 100px*

}

**5.3) Ділення**[**​**](https://textbook.edu.goit.global/html-css-a7u5xv/v1/uk/docs/lesson-14/math#%D0%B4%D1%96%D0%BB%D0%B5%D0%BD%D0%BD%D1%8F) в препроцесорі виконується у трьох випадках.

main.scss

$value: 50px;

.box {

*// 1 - значення зберігається у змінній.*

 width: $value / 5; *// 10*

*// 2 - значення, взяті в круглі дужки.*

 width: (100px / 5); *// 20px*

*// 3 - значення використовується як частина іншого виразу.*

 width: 100px / 5px + 10px; *// 30px*

}

**5.4) Змінні в операціях**[**​**](https://textbook.edu.goit.global/html-css-a7u5xv/v1/uk/docs/lesson-14/math#%D0%B7%D0%BC%D1%96%D0%BD%D0%BD%D1%96-%D0%B2-%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D1%96%D1%8F%D1%85) - якщо в арифметичній операції використовується валідне значення змінної, не буде жодних проблем:

* SCSS

main.scss

$gridItemMargin: 20px;

.box {

 margin: $gridItemMargin \* 2;

}

* CSS

main.css

.box {

 margin: 40px;

}

Але у разі, якщо SASS-змінна використовується в функції calc(), під час компіляції її ім'я не замінюється на значення:

* SCSS

main.scss

$gridItemMargin: 20px;

.box {

 margin: calc($gridItemMargin \* 2);

}

* CSS

main.css

.box {

 margin: calc($gridItemMargin \* 2);

}

У таких випадках необхідно робити інтерполяцію значення змінної, використовуючи спеціальну конструкцію #{$ім'я\_змінної}:

* SCSS

main.scss

$gridItemMargin: 20px;

.box {

 margin: calc(#{$gridItemMargin} \* 2);

}

* CSS

main.css

.box {

 margin: calc(20px \* 2);

}