



# Wstęp

# Czym jest Sass?

## Syntactically Awesome Style Sheets

Jest to nakładka na CSS, która pozwala wykorzystać nam wiele dodatkowych funkcjonalności takich jak podział na oddzielne pliki, funkcje, pętle, zmienne, mixiny i bardzo wiele innych dodatków.

Aby móc korzystać z dobrodziejstw tego języka, musimy użyć jakiegoś narzędzia, które zamieni kod SCSS na CSS.

**Do tego właśnie wykorzystamy Gulpa, którego przed chwilą skonfigurowaliśmy.**



# Preprocessing

## Czym jest preprocessing?

- **Sass** nie jest interpretowany przez przeglądarki, więc musi zostać przetworzony na CSS.
- Umożliwia to **preprocessing**, czyli przetwarzanie kodu źródłowego według określonych reguł na kod wyjściowy.
- Kodem źródłowym jest skrypt języka SCSS. Wynikiem – kod CSS.
- Domyślnie przetwarzanie **Sassa** zostało napisane w języku Ruby.
- Dzięki bibliotekom **LibSass** można kompilować **Sass** w innych językach: Javie, PHP, .NET itp.

# Sass jest preprocesorem CSS



# Wynikowe pliki CSS

## Wczytujemy pliki CSS

W plikach HTML odwołujemy się do wygenerowanych plików **CSS** – nie do plików **Sassa**.

## Dlaczego?

- Przy kompilowaniu plików **Sassa** wszystkie zmiany wprowadzone bezpośrednio w plikach **CSS** zostaną nadpisane wynikiem kompilacji z plików **Sassa**.
- Zmiany z plików **CSS** nie są przenoszone do plików **Sassa**.

## Ważne!

- Do HTML-a dodajemy pliki **CSS** a nie pliki **Sassa**.
- Nie modyfikujemy plików **CSS**.

# .sass – składnia

## Pliki z rozszerzeniem .sass

**Sass** może być zapisany na dwa sposoby (pliki z rozszerzeniem `.sass` oraz `.scss`). Przykład pierwszego z nich widzimy po prawej.

- W pierwotnej składni `.sass` nie używano nawiasów klamrowych oraz średników.
- Zagnieżdżenia tworzone były przez wcięcia w liniach kodu, **tzw. indented syntax** zaczerpnięty z języka **Haml**.

## Przykład pliku z rozszerzeniem .sass

```
nav
  ul
    margin: 0
    padding: 0
  li
    display: block
```

# .scss – składnia

## Pliki z rozszerzeniem .scss

- Z czasem (wersja 3) składnię pierwotną zastąpiono składnią **SCSS** ( **Sassy CSS**), tak aby była jak najbardziej zbliżona do **CSS**.
- Dzięki temu każdy poprawny kod **CSS** może być wstawiony do plików **Sassa**.
- W przypadku starej składni z kodu **CSS** trzeba było usuwać zbędne znaki.

## Przykład pliku z rozszerzeniem .scss

```
nav {  
  ul {  
    margin: 0;  
    padding: 0;  
  }  
  li {  
    display: block;  
  }  
}
```

# Stosujemy SCSS – Sassy CSS

## Składnia sass

```
nav
  ul
    margin: 0
    padding: 0
  li
    display: block
```

## Składnia scss

```
nav {
  ul {
    margin: 0;
    padding: 0;
  }
  li {
    display: block;
  }
}
```

Sass zawiera też domyślnie narzędzie **Sass-convert** pozwalające zawsze szybko przejść z jednej składni na drugą niemal całkowicie automatycznie, a nawet pomóc przekonwertować kod CSS na SCSS.





# Dlaczego warto pracować z Sassem?

# Dlaczego warto pracować z Sasseem?

- Początki CSS-a sięgają połowy lat 90. Nie przewidziano wtedy pewnych zastosowań deklaracji, które są używane obecnie. Na przykład `float` miał wtedy służyć do pozycjonowania obrazków względem tekstu.
- Struktura CSS-a jest płaska i prosta.
- Sass rozszerza język CSS o możliwości dostępne w językach wykorzystujących paradygmat programowania obiektowego.
- Takie podejście ułatwia tworzenie oraz utrzymanie kodu.

## Możliwości Sassa

1. Definiowanie zmiennych.
2. Zagnieżdżanie właściwości i selektorów.
3. `Mixins` – definiowanie grup deklaracji.
4. Pliki częściowe ( `partials` ) i importowanie ( `@import` ).
5. Dziedziczenie ( `@extend` ).
6. Operacje i operatory `+`, `-`, `*`, `/`, `%`.
7. Wbudowane funkcje np. manipulacja kolorami.
8. Definiowanie własnych funkcji.
9. Logika ( `@if`, `@for`, `@each`, `@while` ).