

STYLOWANIE BARDZIEJ

Piotr Martyniak

DaftCode



CSS RESET I NORMALIZE

```
<link rel="stylesheet" type="text/css" href="reset.css">
<!-- lub: -->
<link rel="stylesheet" type="text/css" href="normalize.css">

<link rel="stylesheet" type="text/css" href="styles.css">
```

RESET

```
http://meyerweb.com/eric/tools/css/reset/
```CSS
html, body, div, span, applet, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, /* ... */ footer, header, hgroup, menu, nav, output, ruby, section, summary,
time, mark, audio, video {
 margin: 0;
 padding: 0;
 border: 0;
 font-size: 100%;
 font: inherit;
 vertical-align: baseline;
}
/* HTML5 display-role reset for older browsers */
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
 display: block;
}
body { line-height: 1; }
ol, ul { list-style: none; }
blockquote, q { quotes: none; }
blockquote:before, blockquote:after,
q:before, q:after { content: ''; content: none; }
table { border-collapse: collapse; border-spacing: 0; }
```

# NORMALIZE

<https://necolas.github.io/normalize.css/>

# ROZSzerzenia CSS



# ZMIENNE

```
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

body {
 font: 100% $font-stack;
 color: $primary-color;
}
```

# ZAGNIEŻDŻANIE (NESTING)

```
nav {
 background: purple;

 ul {
 margin: 0;
 padding: 0;
 list-style: none;
 }

 li { display: inline-block; }
}

//zbuduje się do:

nav { background: purple; }
nav ul {
 margin: 0;
 padding: 0;
 list-style: none;
}

nav li { display: inline-block; }
```

# PARENT SELECTOR

```
#main {
 color: black;

 a {
 font-weight: bold;

 &:hover {
 color: red;
 }
 }
}
```

a w efekcie:

```
#main { color: black; }

.main a { font-weight: bold; }

.main a:hover { color: red; }
```

# IMPORTY

## \_reset.scss

```
html, body, ul, ol {
 margin: 0;
 padding: 0;
}
```

## base.scss

```
@import 'reset';

body {
 font: 100% Helvetica, sans-serif;
 background-color: #efefef;
}
```

# MIXINY

```
@mixin gridTile($w: 200px, $h: 400px) {
 box-shadow: 5px 10px #888888;
 background: white;
 width: $w;
 height: $h;
}

.box {
 @include gridTile(300px, 300px);
}
```

```
@mixin apply-to-ie6-only {
 * html {
 @content;
 }
}
@include apply-to-ie6-only {
 #logo {
 background-image: url(/logo.gif);
 }
}

// wynik:
* html #logo {
 background-image: url(/logo.gif);
}
```

# @EXTEND - ROZSZERZANIE SELEKTORÓW

```
// placeholder selector: %
%message-shared {
 border: 1px solid #ccc;
 padding: 10px;
 color: #333;
}

.message {
 @extend %message-shared;
}

.success {
 @extend %message-shared;
 border-color: green;
}

.error {
 @extend %message-shared;
 border-color: red;
}

.warning {
 @extend %message-shared;
 border-color: yellow;
}
```

```
/* wynik */

.message, .success, .error, .warning {
 border: 1px solid #ccc;
 padding: 10px;
 color: #333;
}

.success {
 border-color: green;
}

.error {
 border-color: red;
}

.warning {
 border-color: yellow;
}
```

# @EXTEND

```
// zwykła klasa
.icon {
 font-family: icofont;
}

.success-icon {
 @extend .icon;
 color: green;
}

.error-icon {
 @extend .icon;
 color: red;
}

.warning-icon {
 @extend .icon;
 color: yellow;
}

.somewhere-else {
 .icon {
 font-size: 20px;
 }
}
```

```
/* skompiluje się pięknie do: */

.icon,
.message-icon,
.success-icon,
.error-icon,
.warning-icon {
 font-family: icofont;
}

/* ale zrobi nam też taki ambaras */

.somewhere-else .icon,
.somewhere-else .message-icon,
.somewhere-else .success-icon,
.somewhere-else .error-icon,
.somewhere-else .warning-icon {
 font-family: icofont;
}
```

# FUNKCJE

```
$grid-width: 40px;
$gutter-width: 10px;

@function grid-width($n) {
 @return $n * $grid-width + ($n - 1) * $gutter-width;
}

.sidebar { width: grid-width(5); }
```

becomes:

```
.sidebar {
 width: 240px;
}
```

# SKŁADNIE

## SCSS vs. SASS

```
nav {
 ul {
 margin: 0;
 padding: 0;
 list-style: none;
 }

 li { display: inline-block; }

 a {
 display: block;
 padding: 6px 12px;
 text-decoration: none;
 }

 @mixin transform($property) {
 -webkit-transform: $property;
 -ms-transform: $property;
 transform: $property;
 }

.box { @include transform(rotate(30deg)); }
```

```
nav
ul
 margin: 0
 padding: 0
 list-style: none

li
 display: inline-block

a
 display: block
 padding: 6px 12px
 text-decoration: none

=transform($property)
 -webkit-transform: $property
 -ms-transform: $property
 transform: $property

.box
 +transform(rotate(30deg))
```

# SASS + WEBPACK

```
> npm i -s sass-loader css-loader style-loader mini-css-extract-plugin postcss-loader postcss-preset-env
```

## webpack.config.js

```
const MiniCssExtractPlugin = require("mini-css-extract-plugin");

const isProduction = process.env.NODE_ENV === 'production';

module.exports = {
 //...
 module: {
 rules: [
 {
 test: /\.s(a|c)ss$/,
 use: [
 isProduction
 ? MiniCssExtractPlugin.loader
 : { loader: 'style-loader', options: { sourceMap: true } },
 { loader: 'css-loader', options: { sourceMap: isProduction } },
 { loader: 'postcss-loader', options: { sourceMap: isProduction } },
 { loader: 'sass-loader', options: { sourceMap: isProduction } }
]
 }
],
 plugins: [
 new MiniCssExtractPlugin({
 // Options similar to the same options in webpackOptions.output; optional
 filename: "[name].css",
 chunkFilename: "[id].css"
 })
]
 };
};
```

## postcss.config.js

```
module.exports = {
 plugins: {
 'postcss-preset-env': {}
 }
}
```

## index.js

```
@import 'main.sass'
document.body.innerHTML = '<h1>Hello!</h1> <p>I am styled.</p>'
```



# FRAMEWORKI CSS

- Bootstrap
- Foundation
- Semantic UI
- Pure.css
- UIkit
- Skeleton
- Susy

# **DOBRE I ZŁE PRAKTYKI**

# CO POWINNO BYĆ OCZYWISTE...

- !important
- <p style="..."
- .color-red
- #id-selectors

# ZAGNIEŻDŻANIE VS. DOBRE NAZEWNICTWO

```
// tak, kaskadowość jest potężna, ale...
.Select {
 &.has-value.is-clearable.Select--single {
 > .Select-control .Select-value {
 .icon {
 content: "\e92b";
 color: $rainbow;
 //...
 }
 }
 }
}

.evil-page {
 .Select {
 &.has-value.is-clearable.Select--single {
 > .Select-control .Select-value {
 .icon {
 color: $abyss;
 }
 }
 }
 }
}
```

```
// lepiej trafnie opisywać rzeczywistość

.select-icon {
 content: "\e92b";
 color: $rainbow;
 //...
}

.evil-select-icon {
 color: $abyss;
}
```



# WSZYSCY KOCHAJĄ Z-INDEX...

```
.drop-down-menu-content {
 display: absolute;
 z-index: 999; //more than anything else
}

.popup {
 display: fixed;
 z-index: 9999; //even morer
}

.select-on-popup {
 z-index: 99999; //kill me please
}

//and then:

.enter-advert-overlay {
 z-index:
```

# SASS na ratunek 🐍

```
///
//global z-indexes:

$zindex-dropdown: 100;
$zindex-sticky: 120;
$zindex-fixed: 130;
$zindex-modal-backdrop: 140;
$zindex-modal: 150;
$zindex-popover: 160;
$zindex-tooltip: 170;
```

# NAZEWNICTWO

```
//nieładnie
$color-gray: #eee;
$color-gray-second: #ccc;
$color-gray-third: #666;

//ładniej
$color-gainsboro: #DCDCDC;
$color-lightgray: #D3D3D3;
$color-silver: #C0C0C0;
$color-darkgray: #A9A9A9;
$color-dimgray: #696969;

//$color-gray-#{percent}
$color-gray-90: #e6e6e6;

//context-relevant - przepięknie
$color-gray-shadow: #eee;
```

```
//brzydko
$color-red: #F00;

//niebrzydko
$color-brand: #F00;
$color-accent: #F0F;
$color-positive: #0F0;
$color-negative: #F55;

//akceptowalnie
$color-black: #000;
$color-white: #fff;
```

**@EXTEND**

tylko z  
%placeholderami

# JAK ŻYĆ?!

- 7-1 pattern

```
sass/
|
|– abstracts/ # vars, mixins, functions, etc.
|– base/ # fonts, helpers, resets
|– components/
|– layout/ # footer, header, grid, etc.
|– pages/
|– themes/
|– vendors/
`– main.scss # Main Sass file
```

- Boilerplates:  
<https://github.com/HugoGiraudel/sass-boilerplate/>
- Linting: sass-lint, scss-lint, style-lint
- Methodologies

# METODOLOGIE

- OOCSS
- SMACSS
- Atomic
- BEM

# OOCSS

object oriented css

- Rozdział struktury od skórki
- Rozdział kontenerów od treści

Object == 'recurring visual pattern'

# SMACSS

## Scalable and Modular Architecture for CSS

- Base - podstawy
- Layout - ...layout? 🤔
- Module - komponenty
- State - stany
- Theme - skórka

# ATOMIC CSS

każdy styl -> nowa klasa

# BEM

## BLOCK

Niezależny, atomowy obiekt, który ma sens sam w sobie.  
Komponent.

nagłówek,  
avatar, menu,  
checkbox,  
modal

## ELEMENT

Element bloku.  
Kawałek HTMLa który występuje tylko w bloku, jest jego integralną częścią

pozycja  
menu/listy,  
etykietka  
checkbox'a, tytuł  
w nagłówku, 'x' w  
modalu

## MODIFIER

Flaga informująca o stanie, wersji, kontekście bloku (lub elementu)

disabled,  
highlighted,  
checked,  
open, size  
big, color  
yellow

`.block__element--modifier`

`.block__element`

`.block--modifier`

```
.user-list { //block
 padding: 5px;
}

&__header { //element
 @include h3-font;
}

&__item { //element
 padding: 0 5px;

 &--highlighted { //modifier elementu
 font-size: 2rem;
 background: $color-list-highlight;
 }
}

&--horizontal { //modifier bloku
 display: flex;

 .user-list {
 &__item { //element w zmodyfikowanym bloku
 padding: 5px 0;
 }
 }
}
```

# BLOKI TO KOMPONENTY

```
<div class='user-card'> <!-- user-card -->
 <div class='user-card__header'>
 <div class='card-header'> <!-- card-header -->
 <div class='card-header__title'>My profile</div>
 <div class='card-header__actions'>
 <div class='card-header__button button button--hide'> - </div> <!-- button -->
 <div class='card-header__button button button--expand'> + </div>
 </div>
 </div>
 </div>
 <div class='user-card__content'>
 things!
 </div>
</div>
```

```
sass/
|
|-- abstracts/
|-- base/
|-- components/
 |
 |-- button.scss
 |-- card-header.scss
 `-- user-card.scss
```

## Założenia

- Kod dzielimy na Komponenty i mapujemy na Bloki
- Każdy blok to nowy plik
- Ograniczamy zagnieżdżanie SASSa
- Niewiele generycznych klas i stylów
- Nie ma stylów kontekstowych

# POWSZECHNE BŁĘDY

## "Zagnieżdzanie nazw" elementów

```
.list__item__title // NOPE!
```

```
.list__item //👍
.list__title
```

## Kaskadowość

```
.user-card {
 &__header {
 padding: 0 5px;

 .delete-button {
 color: $color-navy;
 }
 }
}
```

raczej:

```
.delete-button {
 &--inline {
 color: $color-navy
 }
}
```

```
.user-card {
 &__button {
 color: $color-navy;
 }
}
```

# BŁĘDY...

```
.user-list {
 padding: 5px;

 &__header {
 @include h3-font;
 }
}
```

```
.product-list {
 color: $color-navy;

 &__header {
 @include h3-font;
 }
}
```

# BŁĘDY...

```
.polaroid {
 //... default styles
 //--- modified versions:

 &--elevated {
 box-shadow: $card-shadow;
 background: $color-white;
 }

 .polaroid {
 &__image {
 box-shadow: none;
 }
 &__caption {
 border: 1px solid $color-navy;
 }
 &__title {
 font-size: $font-size-h4;
 }
 &__description {
 color: $color-default-text
 }
 }
}

//and so on:

&--horizontal {
 //some styles

 .polaroid {
 &__image { //...more styles
 &__caption { //...
 &__title { //...
 }
 }

 &--tagged {
 //some styles

 .polaroid {
 &__image { //...more styles
 &__caption { //...
 &__title { //...
 }
 }
 }
}
```

# WIĘCEJ BŁĘDÓW... 😬

```
<!-- bad!!! -->
<div class="button--hide"> - </div>
<div class="button--expand"> + </div>
```

```
<!-- good :) -->
<div class="button button--hide"> - </div>
<div class="button button--expand"> + </div>
```

nie chce się pisać?

```
npm i -s bem-modifiers
```

# CSS-IN-JS

"Inline css" buduje się do:

```
<p style="color: white; background-color: black;">inline style!</p>
```

To nie jest CSS-IN-JS!

...to jest:

```
<style>
.hash136s21 {
 background-color: black;
 color: white;
}
</style>

<p class="hash136s21">Hello CSS-in-JS</p>
```

## Piszemy go w JS!

```
const styles = {
 wrapper: {
 background: 'black'
 },
 title: {
 color: 'white'
 }
};

applyStyles(myComponent, styles); // zmyślony przykład
```

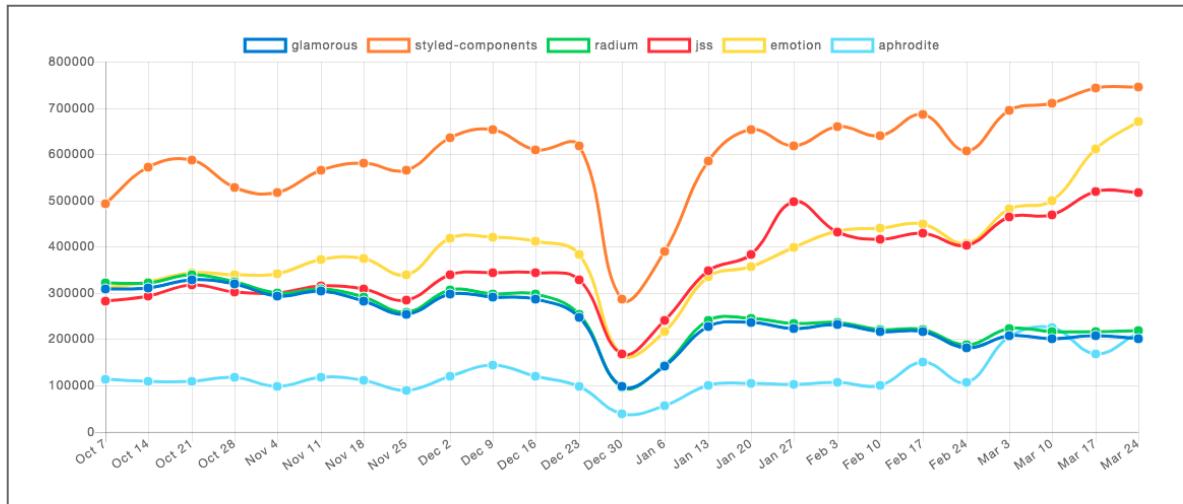
## 👍 ZALETY 👍

- Komponentowy
- Pełne wykorzystanie JS
- 100% wyizolowane style
- Code sharing
- Dołączane tylko obecnie potrzebne style
- Brak nieużywanych reguł
- Unit testy dla CSS! (jak już ktoś bardzo chce...)
- Łatwy setup (z webpackiem)

## 👎 WADY 👎

- Jeszcze jedna abstrakcja do wyuczenia.
- Słabe wykorzystanie cache przeglądarki (przy SSR).
- Kolejne zależności.
- Brak korzyści z BEMowskiego myślenia o komponentach.
- Tylko dla SPA / SSR 🤔
- Pożenione z Frameworkami JS 🤔

# CO MAMY DO WYBORU?



# ...A KONFIGURACJA?

...

...

```
npm i --s @emotion/core
```

Zrobione! 🍻

# ZADANIE DOMOWE

1. Rozszerzyć naszą konfigurację webpacka tak, aby:
  - kompilował SASSa
  - załączał style do projektu
  - automatycznie dołączał prefixy przeglądarki do stylów (postcss)
2. Wdrożyć projekt
  - Zaimplementować przedstawiony layout (HTML + SASS)
  - Zastosować metodologię BEM
  - RWD nie jest wymagane, ale będzie dodatkowo punktowane

Pracę domową wykonaj na podstawie rozwiązania z lekcji pierwszej:

**[https://github.com/daftcode/daftacademy-frontend\\_levelup-spring2019/tree/master/prace\\_domowe/rozwi%C4%85zanie](https://github.com/daftcode/daftacademy-frontend_levelup-spring2019/tree/master/prace_domowe/rozwi%C4%85zanie)** lub wykorzystaj własne rozwiązanie

d'inks

[Beer](#) [Coffee / Tea](#) [Cocktail](#) [Cocoa](#) [Milk / Float / Shake](#) [Shot](#) [Punch / Party Drink](#) [Homemade Liqueur](#)

Bijou

Negroni

Mojito

Texas  
Rattlesnake

Egg Nogg

Zinger

Say hello! - [hi@dinks.com](mailto:hi@dinks.com)

Follow us

Muchas gracias!

[https://m8ms.github.io/stylowanie\\_bardziej/](https://m8ms.github.io/stylowanie_bardziej/)