# Projektowanie Aplikacji Internetowych
# Wykład 4

Zaawansowane techniki CSS

**Mateusz Pawełkiewicz**

## 1. Preprocesory CSS

Preprocesory CSS to narzędzia, które rozszerzają możliwości standardowego CSS, wprowadzając dodatkowe funkcje takie jak zmienne, zagnieżdżanie selektorów, funkcje, mixiny i wiele innych. Ułatwiają one pisanie kodu CSS, czyniąc go bardziej modularnym, czytelnym i łatwiejszym w utrzymaniu.

## 1.1 SASS/SCSS

### 1.1.1 Czym jest SASS/SCSS

- **SASS (Syntactically Awesome Style Sheets)** to jeden z najpopularniejszych preprocesorów CSS.
- **SCSS** to nowsza, bardziej przyjazna składnia SASS, która jest kompatybilna ze składnią CSS3.

### 1.1.2 Zalety korzystania z SASS/SCSS

- **Zmienne**: Umożliwiają przechowywanie wartości, takich jak kolory, rozmiary czcionek, które można ponownie wykorzystać w całym arkuszu stylów.

```
$primary-color: #3498db;

body {
  background-color: $primary-color;
}
```

- **Zagnieżdżanie selektorów**: Pozwala na zagnieżdżanie reguł CSS w sposób hierarchiczny, co zwiększa czytelność kodu.

```
nav {
  ul {
    list-style: none;
    li {
      display: inline-block;
    }
  }
}
```

- **Mixiny**: Definiowanie fragmentów kodu, które można wielokrotnie wykorzystać.

```
@mixin flex-center {
  display: flex;
  justify-content: center;
  align-items: center;
}

.container {
  @include flex-center;
}
```

- **Dziedziczenie (Extend)**: Pozwala jednemu selektorowi dziedziczyć style innego.

```
%button-styles {
  padding: 10px 20px;
  border: none;
  cursor: pointer;
}

.primary-button {
  @extend %button-styles;
  background-color: blue;
  color: white;
}

.secondary-button {
  @extend %button-styles;
  background-color: gray;
  color: black;
}
```

### 1.1.3 Instalacja i kompilacja SASS/SCSS

- **Instalacja poprzez npm**:

```
npm install -g sass
```

- **Kompilacja plików SCSS do CSS**:

```
sass styles.scss styles.css
```

- **Automatyczna kompilacja z obserwacją zmian**:

```
sass --watch styles.scss:styles.css
```

## 1.2 Less

### 1.2.1 Czym jest Less

- **Less (Leaner Style Sheets)** to inny popularny preprocesor CSS.
- Składnia Less jest podobna do SCSS i również rozszerza możliwości standardowego CSS.

### 1.2.2 Funkcje Less

- **Zmienne**:

```
@primary-color: #3498db;

body {
  background-color: @primary-color;
}
```

- **Zagnieżdżanie**:

```
nav {
  ul {
    list-style: none;
    li {
      display: inline-block;
    }
  }
}
```

- **Mixiny**:

```
.flex-center() {
  display: flex;
  justify-content: center;
  align-items: center;
}

.container {
  .flex-center();
}
```

- **Operacje matematyczne**:

```
@base-font-size: 16px;

h1 {
  font-size: @base-font-size * 2;
}
```

### 1.2.3 Instalacja i kompilacja Less

- **Instalacja poprzez npm**:

```
npm install -g less
```

- **Kompilacja plików Less do CSS**:

```
lessc styles.less styles.css
```

---

## 2. Metodologie organizacji kodu CSS

Organizacja kodu CSS jest kluczowa dla utrzymania dużych projektów. Dobre praktyki i metodologie pomagają w tworzeniu struktury, która jest czytelna, łatwa w nawigacji i skalowalna.

## 2.1 BEM (Block Element Modifier)

### 2.1.1 Czym jest BEM

- **BEM** to akronim od **Block**, **Element**, **Modifier**.
- Jest to metodologia nazewnictwa klas CSS, która ułatwia tworzenie złożonych interfejsów użytkownika.

### 2.1.2 Struktura BEM

- **Block (Blok)**: Niezależny komponent, np. menu, header, footer.

```
<div class="menu">
  ...
</div>
```

- **Element (Element)**: Część bloku, która pełni określoną funkcję, np. menu__item, menu__link.

```
<ul class="menu__list">
  <li class="menu__item">
    <a href="#" class="menu__link">Home</a>
  </li>
</ul>
```

- **Modifier (Modyfikator)**: Wariant bloku lub elementu, który zmienia jego wygląd lub zachowanie, np. menu__item--active.

```
<li class="menu__item menu__item--active">
  <a href="#" class="menu__link">Home</a>
</li>
```

### 2.1.3 Zalety BEM

- **Czytelność**: Jasno określa relacje między elementami.
- **Uniknięcie konfliktów**: Unikalne nazewnictwo minimalizuje ryzyko nadpisywania stylów.
- **Skalowalność**: Ułatwia rozbudowę kodu w miarę rozwoju projektu.

### 2.1.4 Przykład zastosowania

CSS:

```
/* Blok */
.menu {
  background-color: #333;
}

/* Elementy */
.menu__list {
  list-style: none;
  display: flex;
}
```

```
.menu__item {
  margin-right: 20px;
}

.menu__link {
  color: white;
  text-decoration: none;
}

/* Modyfikator */
.menu__item--active .menu__link {
  font-weight: bold;
}
```

HTML:

```
<nav class="menu">
  <ul class="menu__list">
    <li class="menu__item menu__item--active">
      <a href="#" class="menu__link">Home</a>
    </li>
    <li class="menu__item">
      <a href="#" class="menu__link">About</a>
    </li>
  </ul>
</nav>
```

## 2.2 OOCSS (Object-Oriented CSS)

### 2.2.1 Czym jest OOCSS

- **OOCSS (Object-Oriented CSS)** to podejście do pisania CSS oparte na zasadach programowania obiektowego.
- Polega na tworzeniu modularnych, wielokrotnego użytku komponentów.

### 2.2.2 Zasady OOCSS

- **Oddzielanie struktury od skóry**: Rozdzielenie stylów dotyczących struktury (np. układ) od stylów wizualnych (np. kolory, tła).

```
/* Struktura */
.box {
  padding: 20px;
  border: 1px solid #ccc;
}

/* Skóra */
.theme-light {
  background-color: #fff;
  color: #000;
}

.theme-dark {
```

```
  background-color: #000;
  color: #fff;
}
```

- **Oddzielanie kontenera od zawartości**: Stylowanie elementów niezależnie od ich umiejscowienia w strukturze DOM.

### 2.2.3 Zalety OOCSS

- **Reużywalność kodu**: Komponenty mogą być wykorzystywane w różnych miejscach.
- **Łatwiejsza konserwacja**: Zmiany w jednym miejscu wpływają na wszystkie wystąpienia komponentu.

---

## 3. Frameworki CSS

Frameworki CSS to biblioteki gotowych stylów i komponentów, które przyspieszają proces tworzenia interfejsów użytkownika. Ułatwiają tworzenie responsywnych i spójnych wizualnie stron internetowych.

## 3.1 Bootstrap

### 3.1.1 Czym jest Bootstrap

- **Bootstrap** to najpopularniejszy framework CSS stworzony przez Twittera.
- Oferuje gotowe komponenty UI, takie jak siatka responsywna, przyciski, formularze, nawigacje, modale i wiele innych.

### 3.1.2 Kluczowe funkcje Bootstrap

- **Siatka responsywna**: System 12-kolumnowej siatki umożliwiający tworzenie responsywnych układów.

```
<div class="container">
  <div class="row">
    <div class="col-md-6">Kolumna 1</div>
    <div class="col-md-6">Kolumna 2</div>
  </div>
</div>
```

- **Komponenty UI**: Przyciski, karty, alerty, paski postępu, itp.

```
<button class="btn btn-primary">Kliknij mnie</button>
```

- **Wtyczki JavaScript**: Interaktywne komponenty takie jak modale, karuzele, dropdowny.

### 3.1.3 Zalety korzystania z Bootstrap

- **Szybkość prototypowania**: Możliwość szybkiego tworzenia interfejsów bez pisania dużej ilości własnego kodu CSS.
- **Spójność**: Jednolity wygląd i zachowanie komponentów.
- **Społeczność**: Duża społeczność i bogata dokumentacja.

## 3.2 Tailwind CSS

### 3.2.1 Czym jest Tailwind CSS

- **Tailwind CSS** to framework CSS oparty na narzędziach (utility-first), który dostarcza niskopoziomowe klasy narzędziowe.

### 3.2.2 Kluczowe cechy Tailwind CSS

- **Klasy narzędziowe**: Zamiast gotowych komponentów, Tailwind oferuje klasy do stylizowania poszczególnych właściwości.

```
<div class="bg-blue-500 text-white font-bold py-2 px-4 rounded">
  Przycisk
</div>
```

- **Konfigurowalność**: Możliwość dostosowania frameworka do własnych potrzeb poprzez plik konfiguracyjny.
- **Bez konieczności nadpisywania stylów**: Ponieważ nie ma domyślnych stylów komponentów, unikamy konfliktów stylów.

### 3.2.3 Zalety korzystania z Tailwind CSS

- **Elastyczność**: Pełna kontrola nad wyglądem elementów.
- **Mały rozmiar produkcyjny**: Dzięki narzędziom do usuwania nieużywanych klas (np. PurgeCSS), rozmiar plików CSS jest minimalny.
- **Szybkość tworzenia interfejsów**: Przyspiesza proces stylizacji poprzez użycie klas bezpośrednio w znacznikach HTML.

---

# 4. Nowe funkcje w CSS (CSS Variables, Custom Properties)

Nowoczesny CSS wprowadza szereg funkcji, które zwiększają jego możliwości i elastyczność. Jedną z nich są zmienne CSS, znane również jako właściwości niestandardowe.

## 4.1 CSS Variables (Właściwości niestandardowe)

### 4.1.1 Czym są zmienne CSS

- **Zmienne CSS** pozwalają na przechowywanie wartości, które można ponownie wykorzystać w arkuszu stylów.

- Deklarowane są z użyciem prefiksu -- i są dostępne w zakresie selektora, w którym zostały zdefiniowane.

### 4.1.2 Deklaracja i użycie zmiennych

- **Deklaracja**:

```
:root {
  --primary-color: #3498db;
  --font-size-base: 16px;
}
```

- **Użycie**:

```
body {
  color: var(--primary-color);
  font-size: var(--font-size-base);
}
```

### 4.1.3 Zalety korzystania z zmiennych CSS

- **Dynamiczność**: Możliwość zmiany wartości zmiennych w czasie rzeczywistym za pomocą JavaScript.
- **Dziedziczenie**: Zmienne mogą być nadpisywane w zasięgu potomnych selektorów.
- **Organizacja kodu**: Ułatwiają zarządzanie powtarzającymi się wartościami.

### 4.1.4 Przykład zastosowania

- **Tematy kolorystyczne**:

```
.theme-light {
  --background-color: #fff;
  --text-color: #000;
}

.theme-dark {
  --background-color: #000;
  --text-color: #fff;
}

body {
  background-color: var(--background-color);
  color: var(--text-color);
}
```

## 4.2 Funkcje niestandardowe

### 4.2.1 Funkcje calc(), min(), max(), clamp()

- **calc()**: Umożliwia wykonywanie obliczeń matematycznych.

```
.container {
  width: calc(100% - 50px);
}
```

- **min()**: Wybiera najmniejszą wartość z podanych.

```
.element {
  width: min(50%, 300px);
}
```

- **max()**: Wybiera największą wartość z podanych.

```
.element {
  width: max(50%, 200px);
}
```

- **clamp()**: Ustawia wartość w określonym przedziale.

```
.text {
  font-size: clamp(1rem, 2.5vw, 2rem);
}
```

### 4.2.2 Zalety korzystania z funkcji

- **Responsywność**: Ułatwiają tworzenie elementów, które dostosowują się do różnych rozmiarów ekranu.
- **Elastyczność**: Pozwalają na precyzyjne kontrolowanie wartości CSS.