

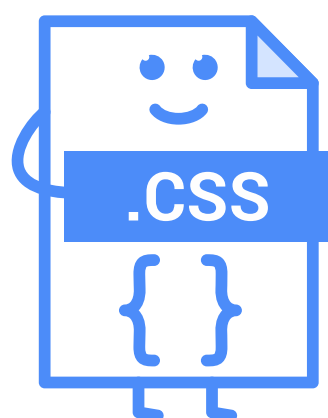
# Tworzenie stron internetowych w HTML5 i CSS3

Przedmiot prowadzony w ramach kursu  
UI/UX web designer

**Wprowadzenie do CSS**

Mateusz Janowski, 2024

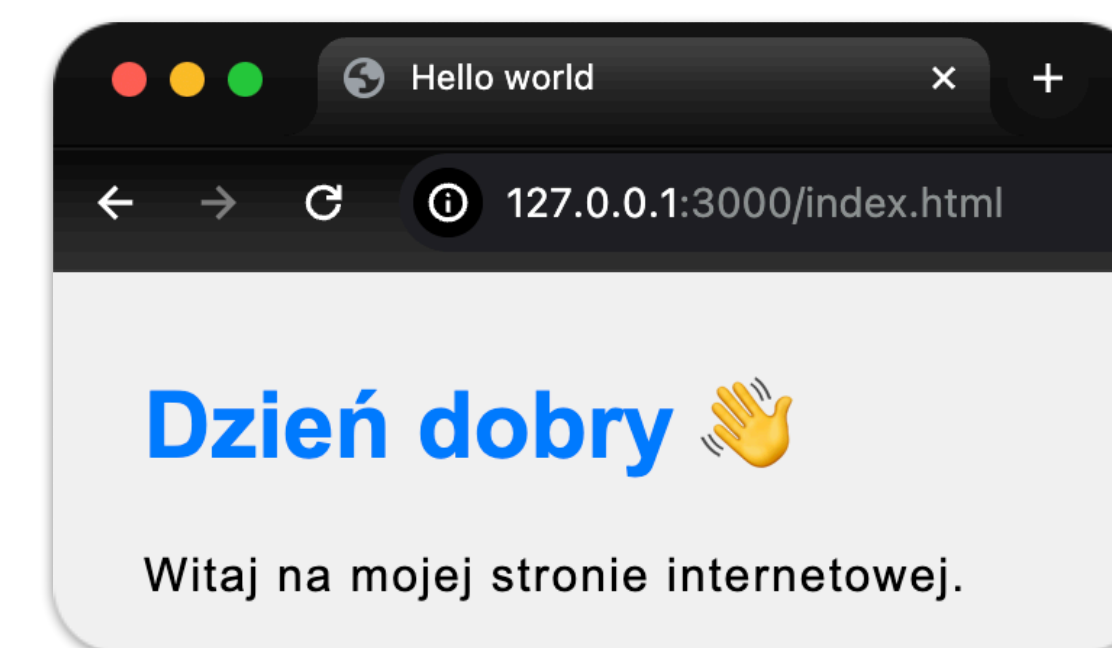
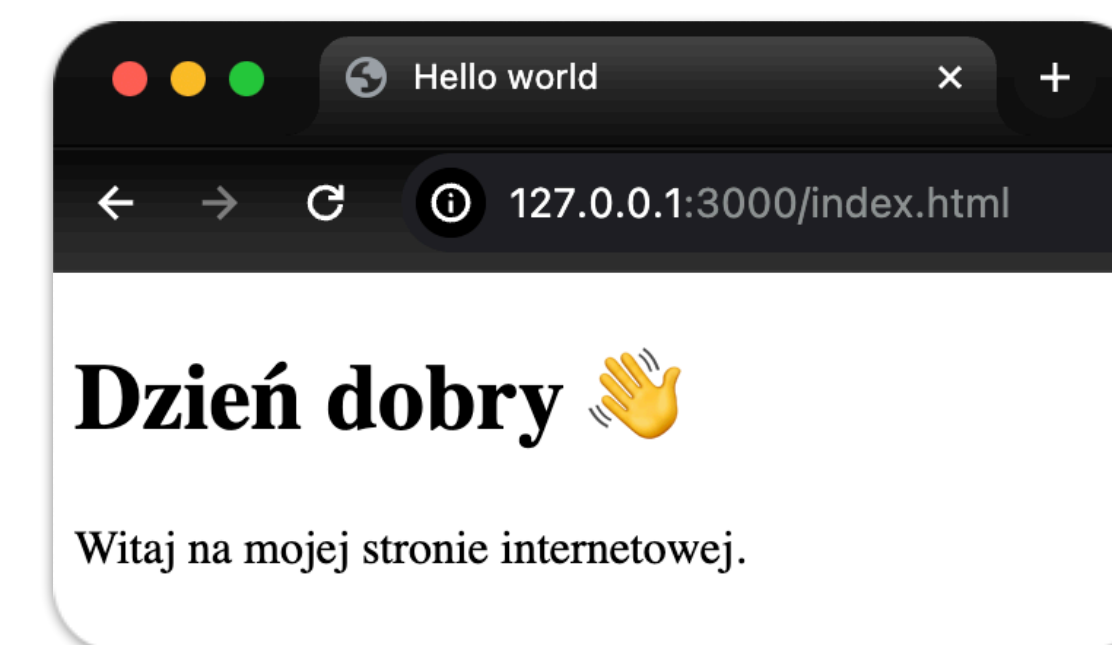
# Co to jest CSS



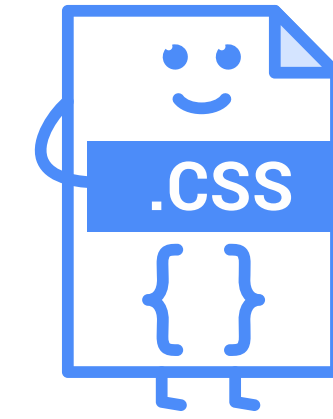
**CSS (Cascading Style Sheets)** - jest językiem arkuszy stylów używanym do definiowania wyglądu i formatowania stron internetowych tworzonych za pomocą HTML.

CSS składa się z niezliczonych właściwości, które są wykorzystywane do formatowania zawartości: właściwości dotyczące czcionek, tekstu, odstępów, układu itp.

```
1  body {  
2    font-family: Arial, sans-serif;  
3    margin: 32px;  
4    background-color: #f0f0f0;  
5  }  
6  
7  h1 {  
8    color: #007bff;  
9  }  
10  
11  p {  
12    letter-spacing: 1px;  
13  }  
14
```



# Budowa reguły CSS



Dokument CSS,  
znany również jako arkusz stylów, jest zbiorem reguł CSS

```
body {  
  margin: 32px;  
  background-color: #f0f0f0;  
}
```

## Selektor

wskazuje, który element lub elementy na stronie zostaną sformatowane przy użyciu określonych stylów.

## Deklaracja

jest to pojedynczy zestaw instrukcji, zawarty w nawiasach klamrowych {}, deklaracja może składać się z kilku par właściwość: wartość.

## Właściwość

jest to typ stylu, który chcesz zastosować do wybranego elementu, np. color, font-size, margin. Właściwości określają, jakie aspekty wyglądu elementu zostaną zmienione.

## Wartość

określa, jak należy zastosować wybraną właściwość, np. red dla color, 12px dla font-size. Wartości mogą przyjmować różne formy, takie jak nazwy kolorów, jednostki (px, em, % itp.), liczby czy słowa kluczowe.

# Sposoby nadawania stylu



## Inline, atrybut style

```
<!DOCTYPE html>
<html lang="pl">
  <head>
    <title>Hello world</title>
  </head>
  <body>
    <h1 style="color: #007bff;">
      Dzień dobry 🙌
    </h1>
  </body>
</html>
```



## Znacznik style

```
<!DOCTYPE html>
<html lang="pl">
  <head>
    <title>Hello world</title>
    <style>
      h1 {
        color: #007bff;
      }
    </style>
  </head>
  <body>
    <h1>Dzień dobry 🙌</h1>
  </body>
</html>
```



## Linkowanie pliku CSS

```
<!DOCTYPE html>
<html lang="pl">
  <head>
    <title>Hello world</title>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <h1>Dzień dobry 🙌</h1>
  </body>
</html>
```



+

```
h1 {
  color: #007bff;
}
```



# Kolory w CSS - podstawowe modele

## Nazwane kolory

Nazwy kolorów określone w specyfikacji CSS.

`dodgerblue`

## Model RGB

RGB oznacza czerwony (Red), zielony (Green) i niebieski (Blue)

`rgb(30, 143, 255)`

## Model RGB HEX

HEX, hexadecimal (szesnastkowy), to format zapisu oparty na modelu RGB

`#1e8fff`

# Kolory w CSS - zaawansowane modele

## Model HWB

HWB oznacza odcień (Hue), biel (Whiteness) i czern (Blackness).

**hwb**(210 12% 0%)

## Model HSL

HSL oznacza odcień (Hue), nasycenie (Saturation) i jasność (Lightness)

**hsl**(210, 100%, 56%)

## Model LCH

LCH oznacza jasność (Lightness), chromę (Chroma) i odcień (Hue):

**lch**(29.5, 87.4%, 301%)

## Model LAB

LAB oznacza jasność (Lightness), (A) odcień od zielonego do czerwonego, (B) odcień od niebieskiego do żółtego.

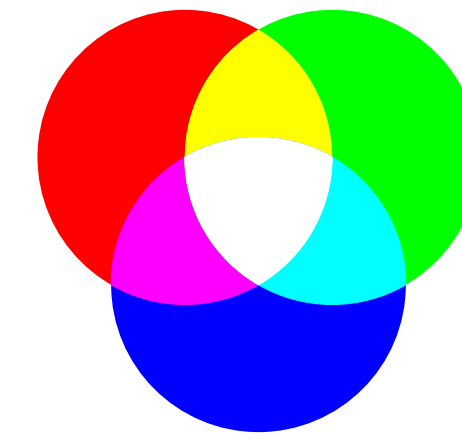
**lab**(58, 1, -31)



# Nazwane kolory

AliceBlue	AntiqueWhite	Aqua	Aquamarine	Azure	Beige	Bisque	Black
BlanchedAlmond	Blue	BlueViolet	Brown	BurlyWood	CadetBlue	Chartreuse	Chocolate
Coral	CornflowerBlue	Cornsilk	Crimson	Cyan	DarkBlue	DarkCyan	DarkGoldenRod
DarkGray	DarkGrey	DarkGreen	DarkKhaki	DarkMagenta	DarkOliveGreen	DarkOrange	DarkOrchid
DarkRed	DarkSalmon	DarkSeaGreen	DarkSlateBlue	DarkSlateGray	DarkSlateGrey	DarkTurquoise	DarkViolet
DeepPink	DeepSkyBlue	DimGray	DimGrey	DodgerBlue	FireBrick	FloralWhite	ForestGreen
Fuchsia	Gainsboro	GhostWhite	Gold	GoldenRod	Gray	Grey	Green
GreenYellow	HoneyDew	HotPink	IndianRed	Indigo	Ivory	Khaki	Lavender
LavenderBlush	LawnGreen	LemonChiffon	LightBlue	LightCoral	LightCyan	LightGoldenRodYellow	LightGray
LightGrey	LightGreen	LightPink	LightSalmon	LightSeaGreen	LightSkyBlue	LightSlateGray	LightSlateGrey
LightSteelBlue	LightYellow	Lime	LimeGreen	Linen	Magenta	Maroon	MediumAquaMarine
MediumBlue	MediumOrchid	MediumPurple	MediumSeaGreen	MediumSlateBlue	MediumSpringGreen	MediumTurquoise	MediumVioletRed
MidnightBlue	MintCream	MistyRose	Moccasin	NavajoWhite	Navy	OldLace	Olive
OliveDrab	Orange	OrangeRed	Orchid	PaleGoldenRod	PaleGreen	PaleTurquoise	PaleVioletRed
PapayaWhip	PeachPuff	Peru	Pink	Plum	PowderBlue	Purple	RebeccaPurple
Red	RosyBrown	RoyalBlue	SaddleBrown	Salmon	SandyBrown	SeaGreen	SeaShell
Sienna	Silver	SkyBlue	SlateBlue	SlateGray	SlateGrey	Snow	SpringGreen
SteelBlue	Tan	Teal	Thistle	Tomato	Turquoise	Violet	Wheat
White	WhiteSmoke	Yellow	YellowGreen				

# Model RGB



`rgb(30, 143, 255)`

`#1e8fff`

`rgb(30, 143, 255, 0.5)`

`#1e8fff80`

`rgb(R, G, B, A)`

Format RGB reprezentuje kolory poprzez połączenie czerwieni (R), zieleni (G) i niebieskiego (B).

Każdy parametr określany w zakresie 0-255.

Parametr alpha (A) odpowiada za przezroczystość. Przyjmuje wartości od 0 do 1. Wartość A jest opcjonalna i jej domyślna wartość wynosi 1 (nieprzezroczysty).

`#rrggbbaa`

Format HEX wykorzystuje sześciocyfrowy kod szesnastkowy do reprezentacji kolorów, gdzie pierwsza para cyfr określa intensywność czerwieni (R), druga para zieleni (G), a trzecia niebieskiego (B).

Każda para może przyjąć wartości od 00 do FF, co odpowiada zakresowi 0-255.

Parametr aa (Alpha), przyjmuje wartości od 00 do FF. Parametr Alpha jest opcjonalny, a jego domyślna wartość to FF (nieprzezroczysty).



# Selektory podstawowe

## Selektor identyfikatora

Wybiera unikalny element na podstawie atrybutu id.

```
#unique {  
  font-size: 20px;  
}
```

## Selektor pseudoklas

Wybiera elementy na podstawie ich stanu.

```
a:hover {  
  text-decoration: underline;  
}
```

## Selektor klasy

Wybiera elementy na podstawie atrybutu class.

```
.error {  
  color: red;  
}
```

## Selektor typu

Wybiera elementy na podstawie nazwy tagu.

```
div {  
  color: black;  
}
```

# Zaawansowane selektory

## Selektor ogólnego rodzeństwa

Wybiera wszystkie elementy tego poziomu, następujące po elemencie.

```
h1 ~ p {  
  color: green;  
}
```

## Selektor potomków

Wybiera elementy będące potomkami elementu.

```
.article-content p {  
  line-height: 1.6;  
}
```

## Selektor bezpośrednich potomków

Wybiera bezpośrednie potomstwo elementu.

```
section > ul > li {  
  margin-bottom: 10px;  
}
```

## Selektor sąsiadującego rodzeństwa

Wybiera elementy bezpośrednio następujące po innym elemencie.

```
h2 + p {  
  font-size: small;  
}
```

# Zaawansowane selektory c.d.

## Selektor atrybutów

Wybiera elementy na podstawie wartości atrybutu.

```
input[type="text"] {  
  border: 1px solid grey;  
}
```

## Selektor uniwersalny

Stosowane do wybrania wszystkich elementów na stronie.

```
* {  
  box-sizing: border-box;  
}
```

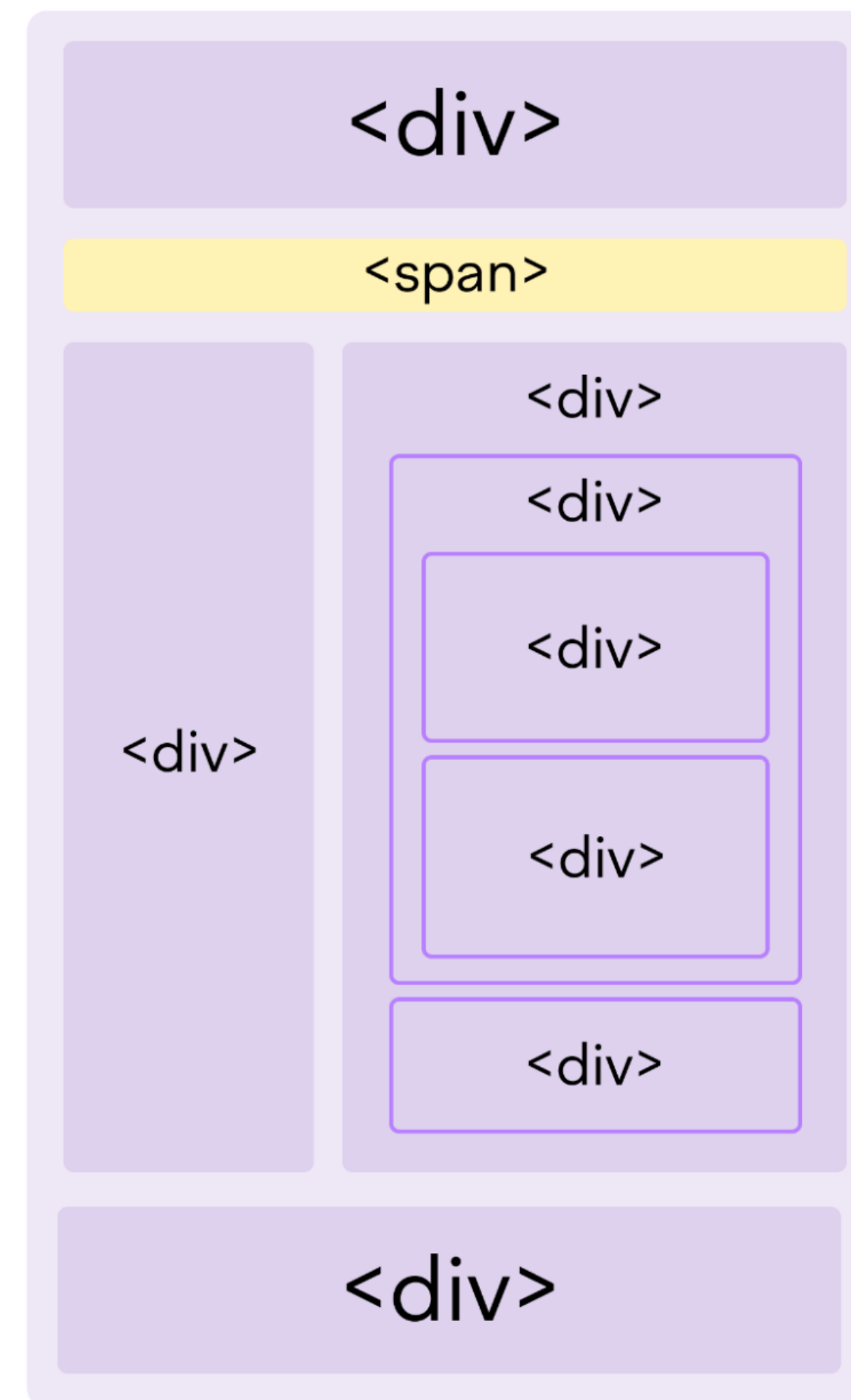
## Selektor pseudoelementów

Pozwala na stylizowanie określonych części elementu.

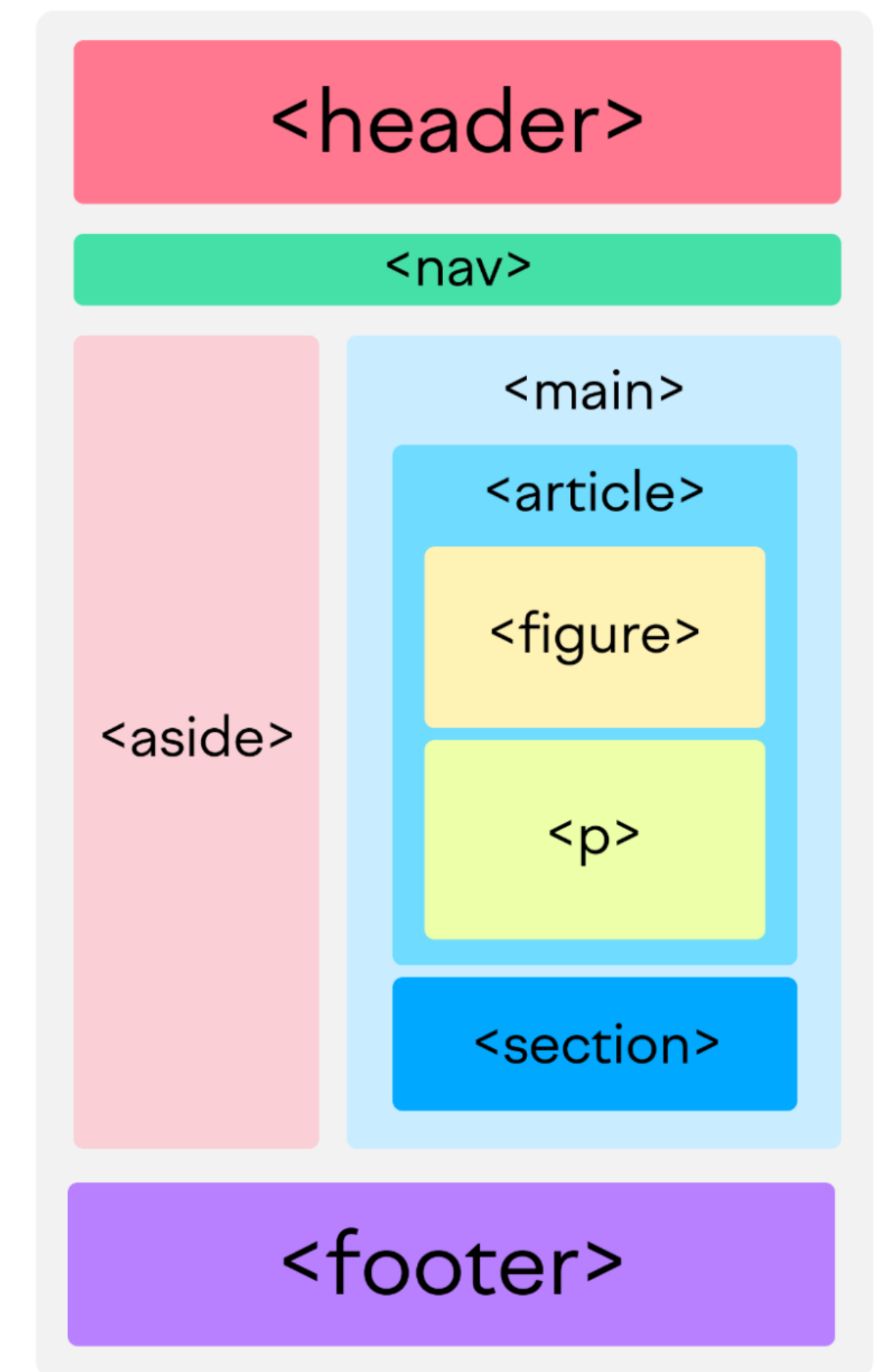
```
p::first-line {  
  font-weight: bold;  
}
```

# HTML Semantyczny

## Non-Semantic HTML



## Semantic HTML



HTML semantyczny to sposób pisania kodu HTML, w którym znaczniki opisują znaczenie treści, a nie tylko jej wygląd.

Oznacza to, że używamy znaczników, które informują przeglądarkę i wyszukiwarki o rodzaju zawartości, np. nagłówka, artykułu, sekcji, czy obrazka.

Zalety stosowania HTML semantycznego:

- Dostępność: Ułatwia osobom niepełnosprawnym korzystanie z witryny za pomocą czytników ekranu.
- Czytelność: Ułatwia programistom i osobom edytującym kod zrozumienie struktury i treści witryny.
- Optimalizacja SEO: Pomaga wyszukiwarkom zrozumieć zawartość witryny i poprawić jej pozycję w wynikach wyszukiwania.

# Box model

## Zawartość

Główna treść elementu (tekst, obrazy).

## Obramowanie

Linia dookoła zawartości, border.

## Wypełnienie

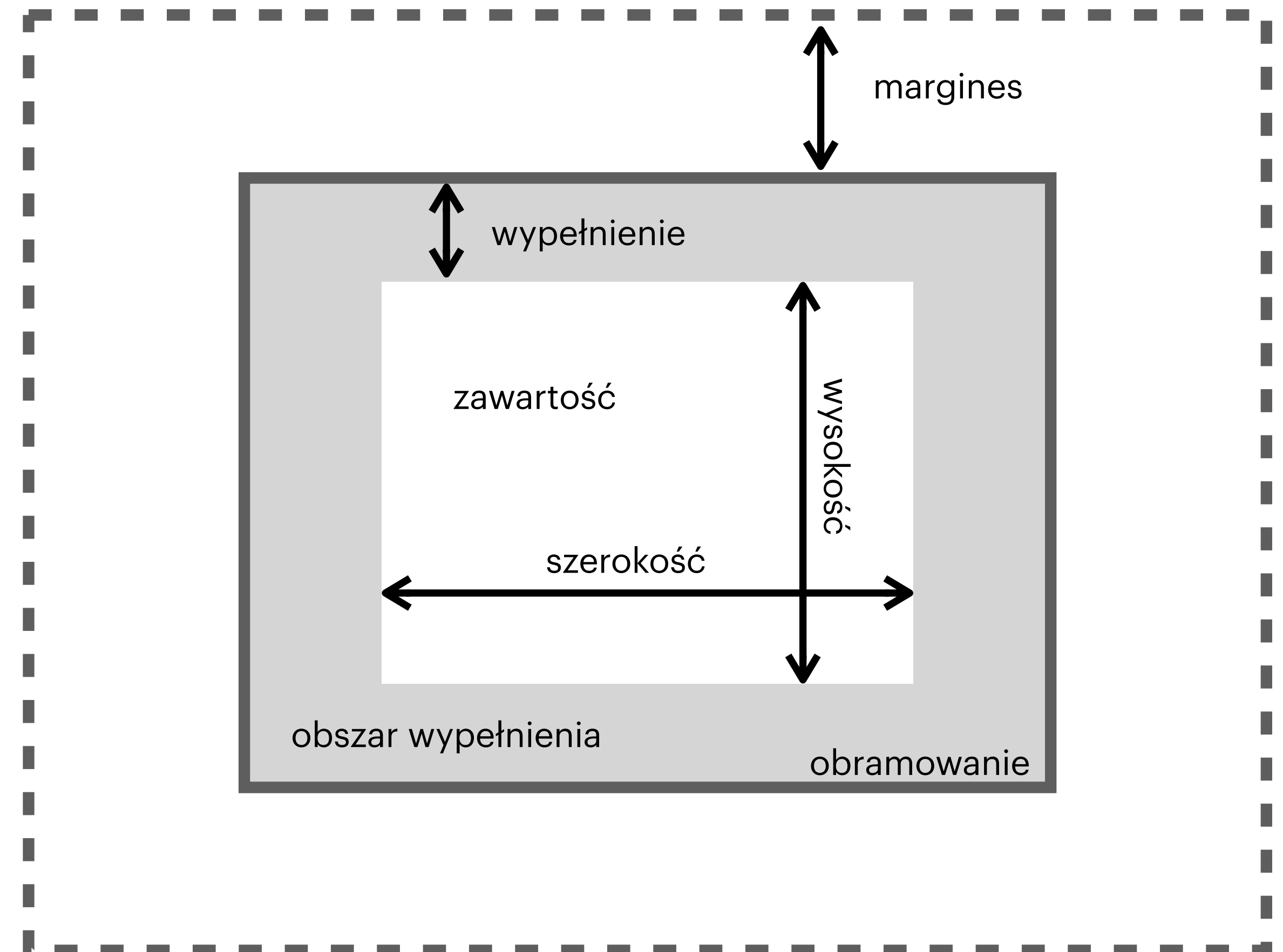
Przestrzeń między zawartością a obramowaniem, padding.

## Margines

Przestrzeń między elementami, margin.

## Obszar wypełnienia

Wypełniany kolorem tła lub obrazem, background-color, background-image.



# Właściwość box-sizing

## **content-box** (domyślne):

- Szerokość i wysokość dotyczą tylko zawartości.
- Całkowite wymiary = zawartość + padding + border.

## **border-box:**

- Szerokość i wysokość obejmują zawartość, padding i border.
- Całkowite wymiary pozostają jak zdefiniowano: 390px x 100px.

Klasa box-one, box-sizing: content-box (wartość domyślna)

Klasa box-two, box-sizing: border-box

```
1  div {
2    width: 390px;
3    height: 100px;
4    margin: 32px;
5    padding: 32px;
6    border: 10px solid black;
7  }
8
9  .box-one {
10   box-sizing: content-box;
11   background: lightyellow;
12 }
13
14 .box-two {
15   box-sizing: border-box;
16   background-color: lightblue;
17 }
18
```



# Właściwość display

**Właściwość display** określa sposób, w jaki dany element jest wyświetlany na stronie internetowej.

- **display: block;** sprawia, że element zajmuje całą dostępną szerokość kontenera, niezależnie od jego rzeczywistej szerokości. Bloki są wyświetlane jeden pod drugim. Typowe przykłady elementów blokowych to `<div>`, `<p>` i `<h1>`.
- **display: inline;** pozwala elementom na układanie się jeden obok drugiego, bez przejmowania całej dostępnej szerokości. Ich szerokość jest równa zawartości. Elementy takie nie mogą mieć ustawionej wysokości ani szerokości. Typowymi elementami inline są `<span>`, `<a>` oraz `<img>`.
- **display: inline-block;** łączy w sobie cechy block i inline. Elementy z tą właściwością mogą być ustawione jeden obok drugiego (jak inline), ale jednocześnie pozwalają na zdefiniowanie ich wysokości i szerokości (jak block).

**Block**

**Inline**

**Inline 2**

**Inline-Block**

**Inline-Block 2**

```
1  .block {
2    display: block;
3    background-color: lightblue;
4    margin: 10px 0;
5    padding: 20px;
6  }
7
8  .inline {
9    display: inline;
10   background-color: lightcoral;
11   margin: 10px;
12   padding: 20px;
13 }
14
15 .inline-block {
16   display: inline-block;
17   height: 64px;
18   background-color: lightgreen;
19   margin: 10px;
20   padding: 20px;
21 }
22
```

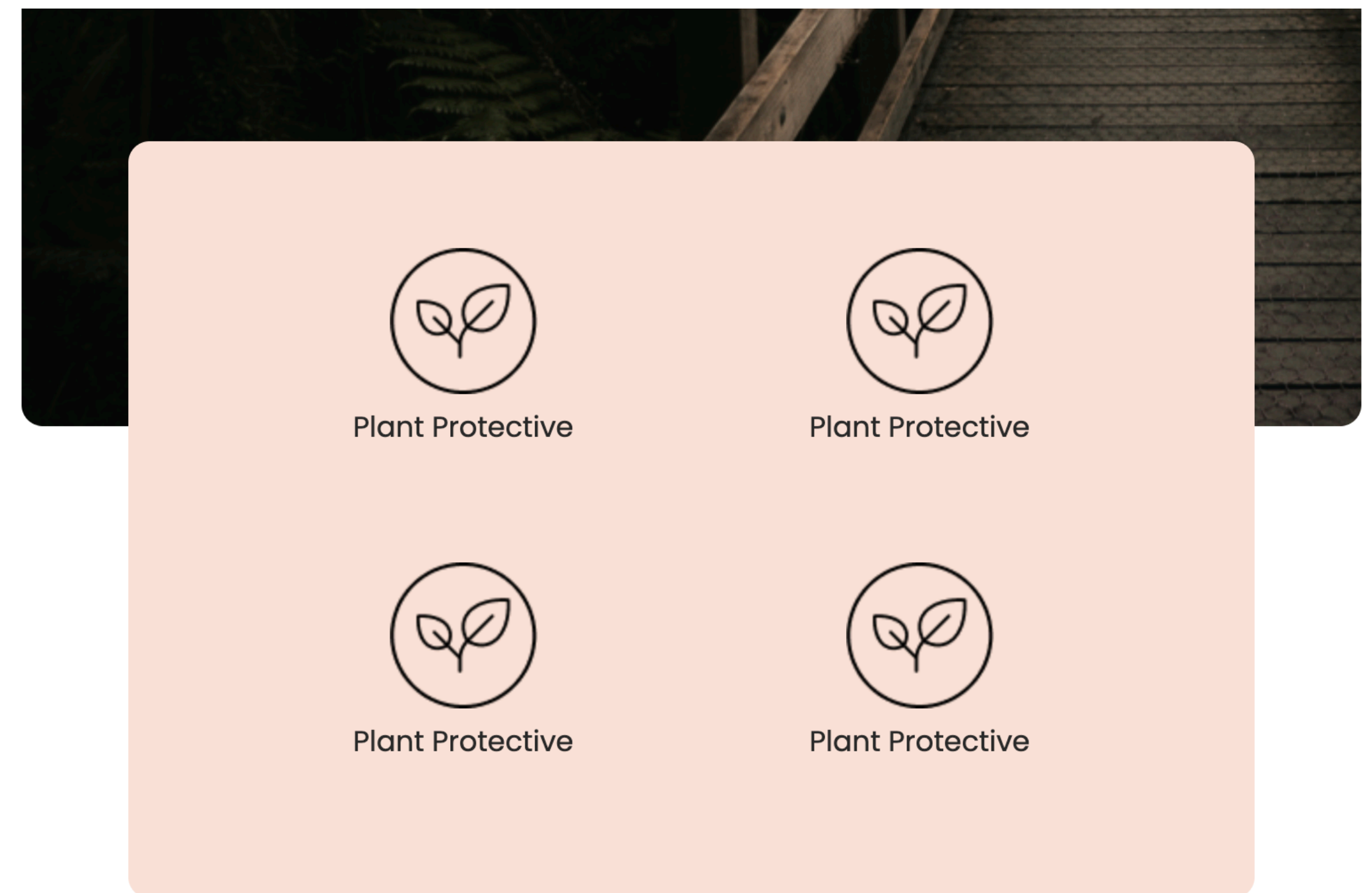
# Flexbox - układ 1 wymiarowy

Flexbox to jednowymiarowy model tworzenia układu elementów.

Główna idea polega na tym, że puste przestrzenie wewnątrz kontenera mogą być automatycznie dzielone przez jego elementy potomne.

Flexbox ułatwia automatyczne wyrównywanie elementów względem siebie wewnątrz nadrzędnego kontenera, zarówno w poziomie, jak i w pionie.

Rozwiązuje on także powszechne problemy, takie jak centrowanie w pionie oraz tworzenie kolumn o równych wysokościach.

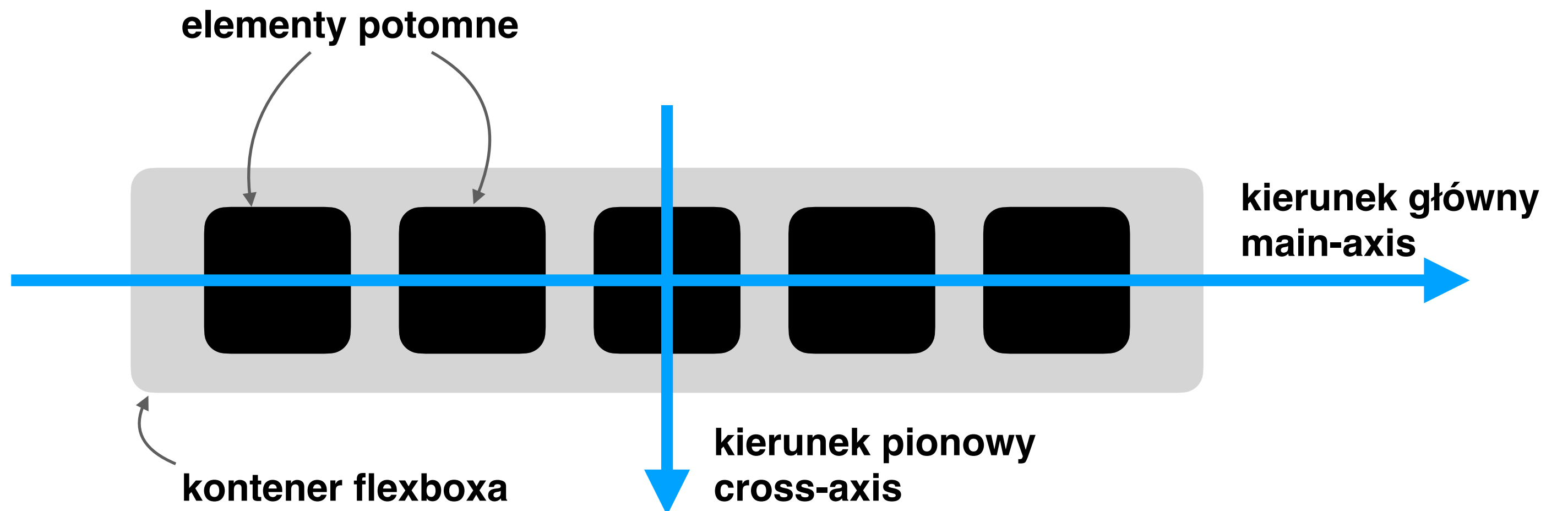


# Jak działa FLEXBOX

Kontener flexbox-a posiada domyślnie jeden kierunek główny (main axis) - domyślnie poziomy. Elementy potomne są rozmieszczane wzdłuż tego kierunku.

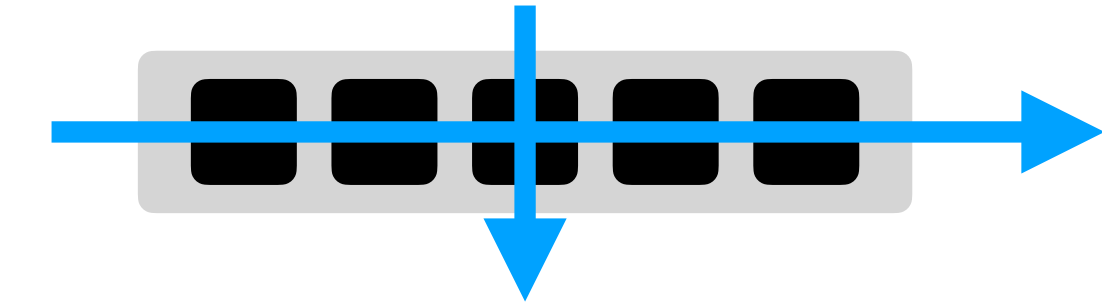
Można ustawić flexbox-owi kierunek pionowy (cross axis).

Wolne miejsce w kontenerze jest dzielone między elementami potomnymi zgodnie z ich ustawieniami flex.



```
nav {  
  display: flex;  
}
```

# Flexbox - właściwości css



## Kontener Flexbox

**gap:** Określa odstęp między elementami.

**justify-content:** Wyrównuje elementy wzdłuż osi głównej (w poziomie).  
flex-start | flex-end | center | space-between | space-around | space-evenly

**align-items:** Wyrównuje elementy wzdłuż osi poprzecznej (w pionie).  
stretch | flex-start | flex-end | center | baseline

**flex-direction:** Określa główną oś.  
row | row-reverse | column | column-reverse

**flex-wrap:** Pozwala elementom przenosić się do nowego wiersza.  
nowrap | wrap | wrap-reverse

**align-content:** Dotyczy wieloliniowego układu elementów.  
stretch | flex-start | flex-end | center | space-between | space-around

## Elementy potomne

**align-self:** Pozwala zastąpić zachowanie align-items dla potomków.  
auto | stretch | flex-start | flex-end | center | baseline

**flex-grow:** Określa zdolność elementu do rozrostu.  
0 (nie rośnie), (wartość większa od 0 oznacza zdolność do wzrostu)

**flex-shrink:** Określa zdolność elementu do kurczenia się.  
1 (kurczy się), (wartość 0 oznacza brak kurczenia się)

**flex-basis:** Definiuje domyślną szerokość elementu, zamiast width.  
auto | width

**flex:** Zalecany skrót do ustawienia flex-grow, flex-shrink i flex-basis.  
0 1 auto

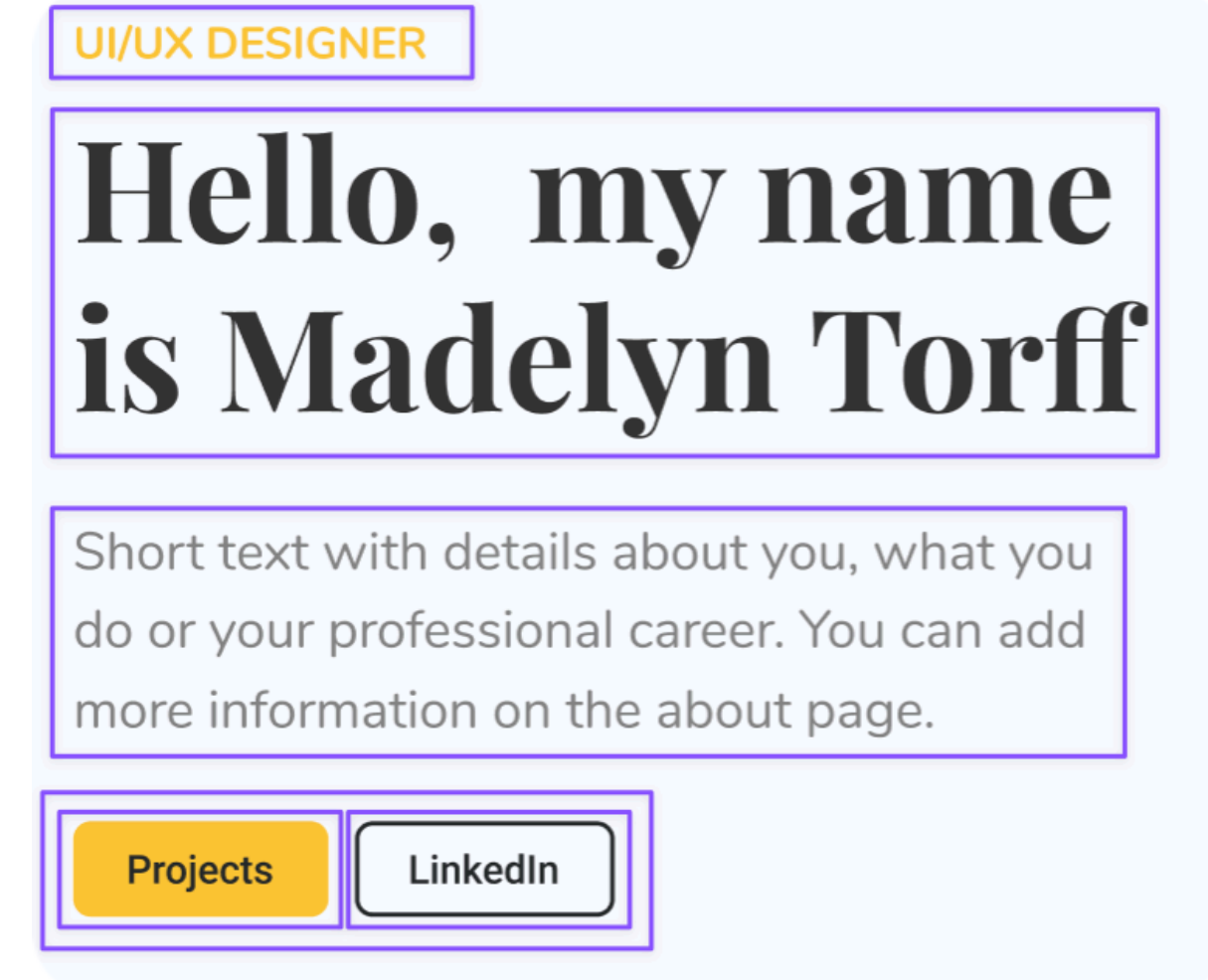
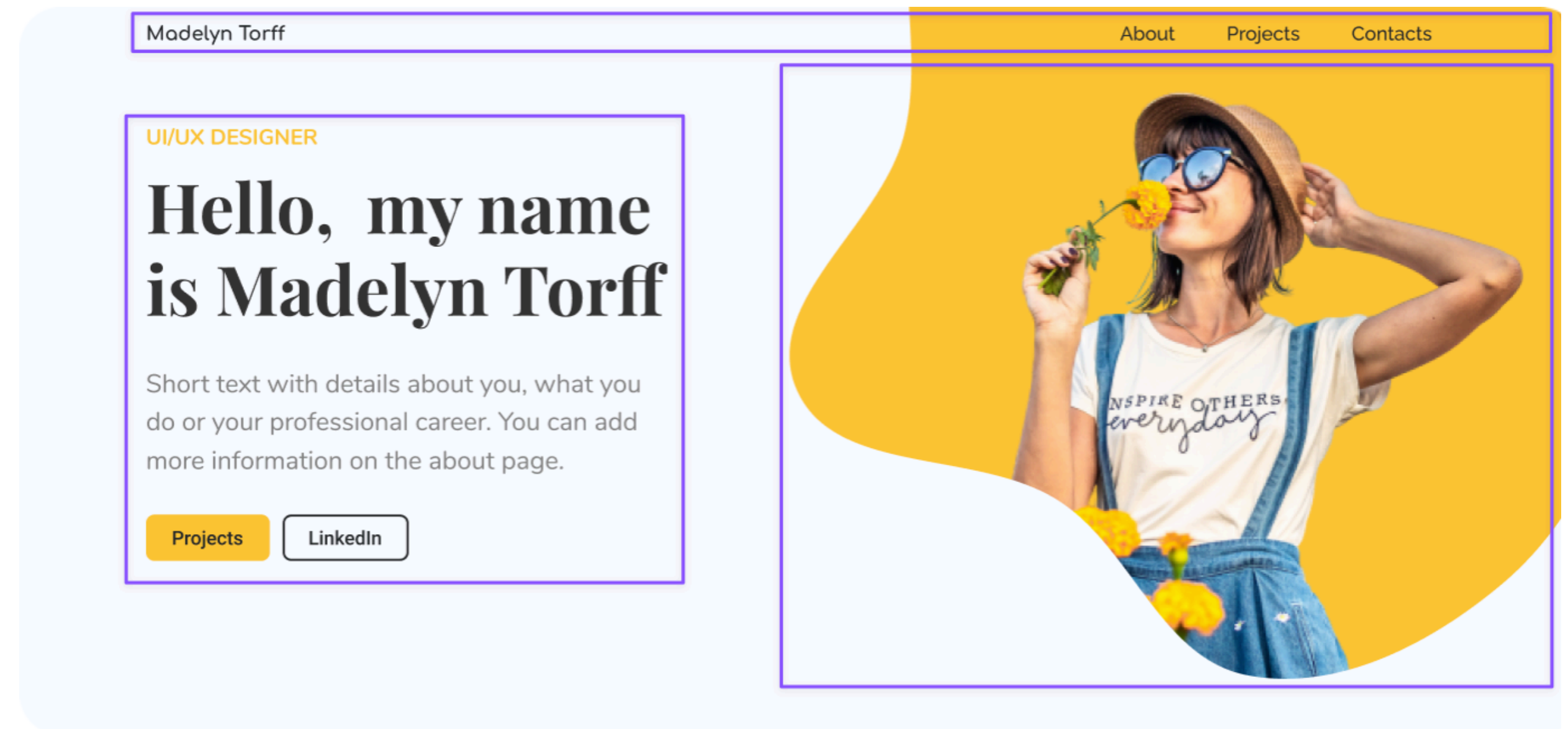
**order:** Kontroluje kolejność elementów.  
0 (domyślnie), (-1 czyni element pierwszym, 1 ostatnim)



# Layout

Layout to ogólne pojęcie określające sposób rozmieszczenia elementów na stronie internetowej.

Może obejmować zarówno **układ strony (page layout)**, jak i **układ komponentów (component layout)**.



# Projektowanie stron responsywnych



## DESKTOP FIRST

Polega na tworzeniu witryn internetowych z myślą o komputerach stacjonarnych, a następnie dostosowywaniu ich do ekranów urządzeń mobilnych.



## MOBILE FIRST

Projektowanie mobile first oznacza tworzenie witryn internetowych z myślą o urządzeniach mobilnych, a następnie dostosowywaniu ich do większych ekranów.



# Responsywność

## Jednostki responsywne (Responsive Units)

Korzystanie z jednostek **rem** zamiast **px** dla określania długości, szerokości, rozmiarów czcionek. Pozwala na łatwe skalowanie całego układu automatycznie po zmianie czcionki bazowej.

```
html {  
  font-size: 16px; /* Czcionka bazowa */  
}  
.container {  
  width: 40rem;  
  padding: 1rem;  
}
```

## Płynne układy (Fluid Layouts)

Umożliwia dostosowanie się układu strony internetowej do aktualnej szerokości widoku lub wysokości widoku.

Zaleca się korzystanie z jednostek % (lub **vh** / **vw**) zamiast **px**.

```
.element {  
  width: 50vw; /* Zajmuje 50% szerokości viewportu */  
  max-width: 500px; /* Maksymalna szerokość elementu */  
}
```

Responsywność stron internetowych to kluczowy aspekt współczesnego projektowania, zapewniający, że witryna wygląda i funkcjonuje dobrze na różnych urządzeniach, od komputerów stacjonarnych po smartfony.

# Responsywność c.d.

## Elastyczne obrazy (Flexible Images)

Domyślnie, obrazy nie skalują się automatycznie przy zmianie wymiarów widoku. Zalecane jest korzystanie z % dla wymiarów obrazów, wraz z właściwością **max-width**.

```
img {  
    max-width: 100%; /* Obraz nie przekroczy szerokości kontenera */  
    height: auto; /* Zachowuje proporcje obrazu */  
}
```

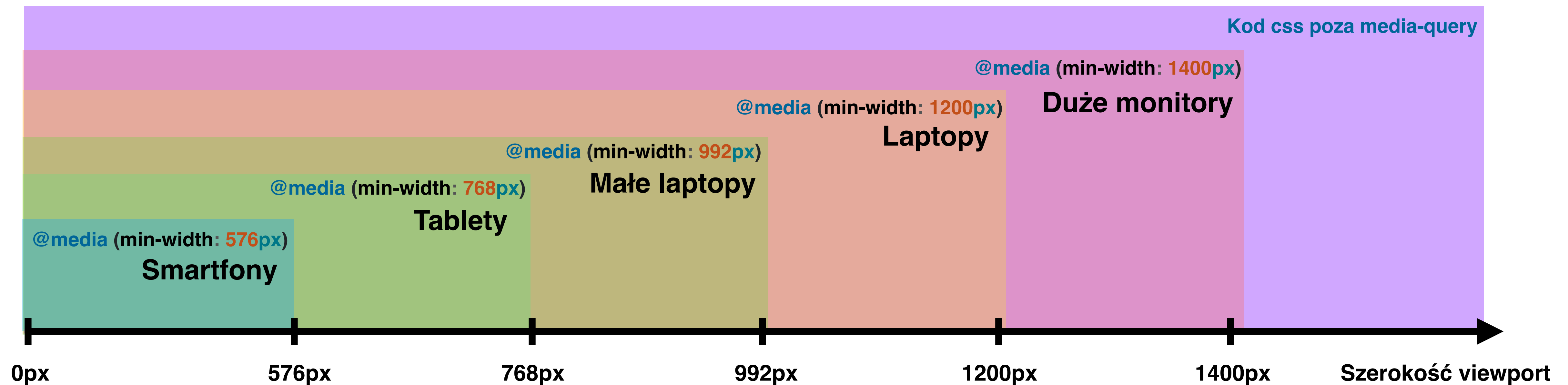
## Media query

Umożliwiają zmianę stylu CSS na określone szerokości wysokości widoku (**breakpoint**).

```
@media (max-width: 576px) {  
    .container {  
        flex-direction: column;  
    }  
}
```

# Media Query

Czy szerokość “**viewport**” jest mniejsza niż “**breakpoint**”?



# Jednostki w CSS

## Jednostki bezwzględne

Są stałe i nie zmieniają swojej wartości niezależnie od innych elementów na stronie.

Najczęściej używane jednostki bezwzględne to:

- **px** (piksele) - najbardziej podstawowa jednostka, odpowiada pojedynczemu pikselowi na ekranie.
- **pt** (punkty) - tradycyjna jednostka używana w druku, 1pt = 1/72 cala.
- **cm** (centymetry), **mm** (milimetry), **in** (cale) - jednostki długości oparte na wymiarach fizycznych.

## Jednostki względne

Ich wartość jest względna w stosunku do innych wartości, co jest elementem niezbędnym w responsywnym projektowaniu stron.

Przykłady jednostek względnych:

- **em** - względem rozmiaru czcionki rodzica. 1em = rozmiar czcionki elementu nadrzędnego.
- **rem** - względem rozmiaru czcionki elementu `<html>`. 1rem = rozmiar czcionki korzenia dokumentu.
- **vw/vh** - procentowa wartość szerokości/wysokości viewportu. 1vw = 1% szerokości viewportu, 1vh = 1% wysokości viewportu.
- **%** - procentowa wartość względem wielkości elementu nadrzędnego.

# Przejścia i animacje CSS

## Przejścia

- Umożliwiają proste zmiany stanu elementów, takie jak zmiana koloru, skali czy pozycji.
- Aktywowane przez akcję użytkownika, np. najechanie kursorem (:hover)

```
button {  
  background-color: blue;  
  color: white;  
  padding: 10px 20px;  
  border: none;  
  cursor: pointer;  
  transition: background-color 0.5s ease;  
}
```

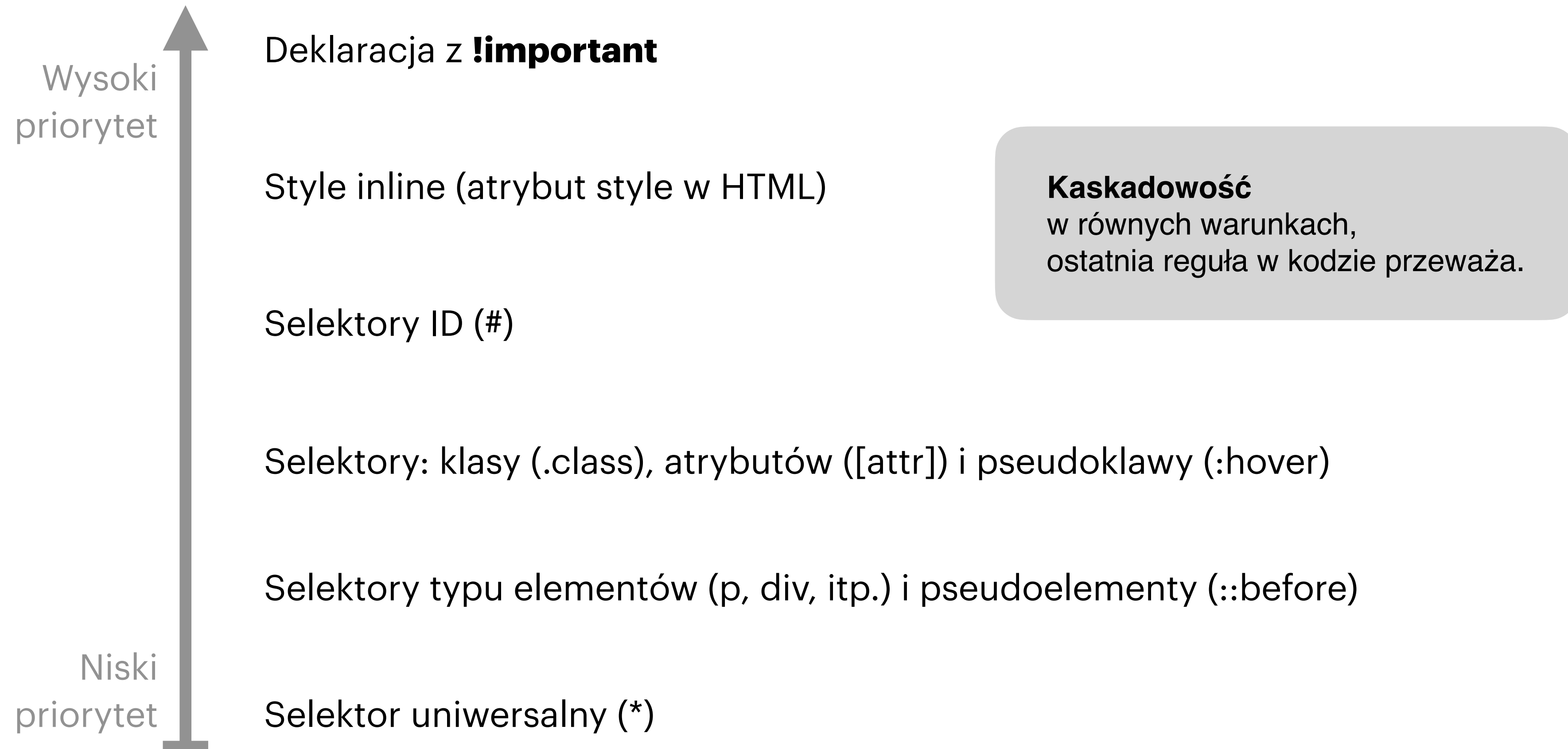
```
button:hover {  
  background-color: red;  
}
```

## Animacje

- Pozwalają na tworzenie złożonych serii ruchów z użyciem etapów (@keyframes).
- Mogą automatycznie uruchamiać się bez interakcji użytkownika i być zapętlane.

```
0% {  
  transform: scale(1);  
}  
50% {  
  transform: scale(1.1);  
}  
100% {  
  transform: scale(1);  
}  
}  
  
.animacja {  
  width: 100px;  
  height: 100px;  
  background-color: purple;  
  animation: pulse 2s infinite;  
}
```

# Konflikty reguł CSS





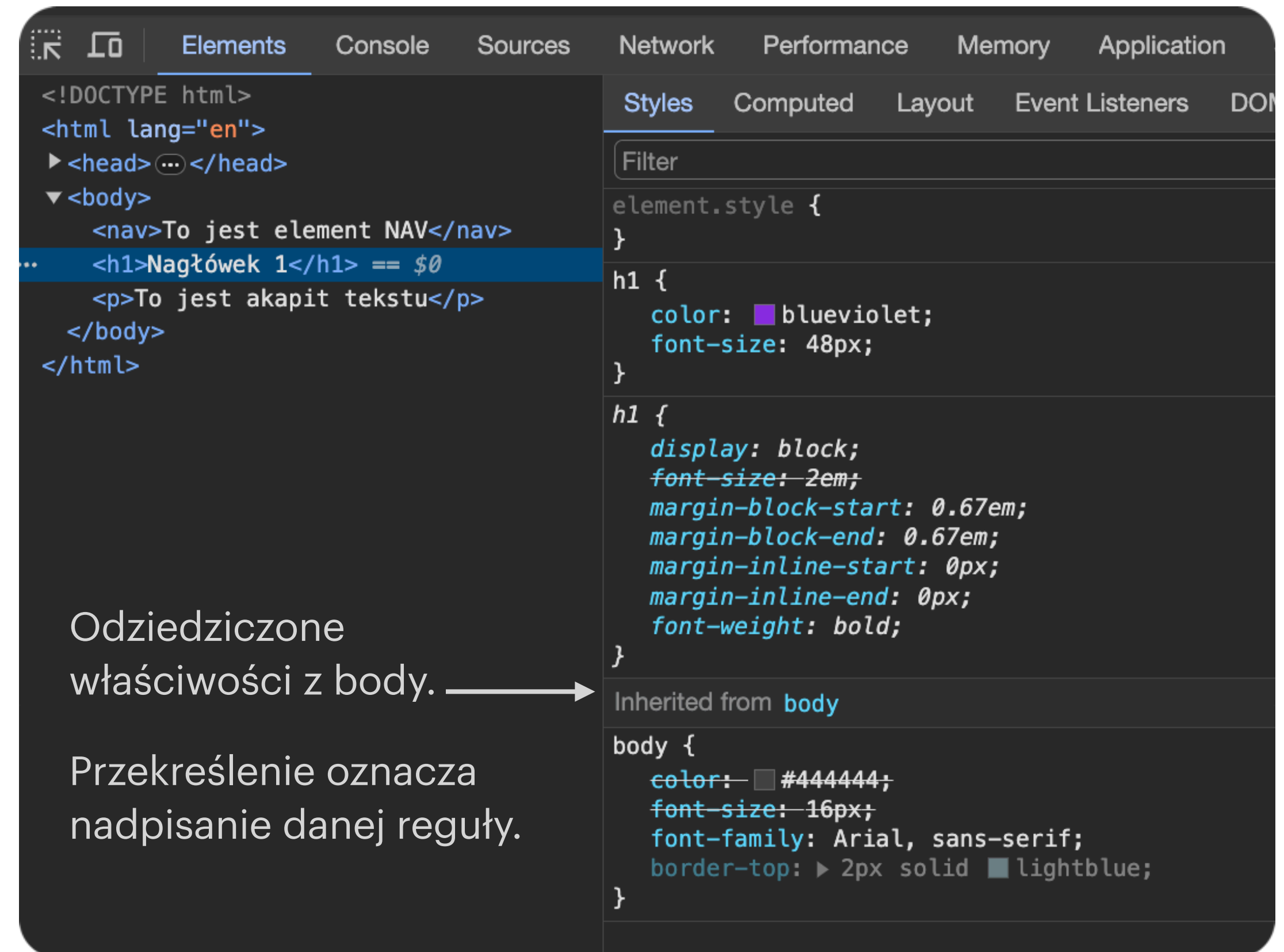
# Dziedziczenie reguł

Dziedziczenie w CSS to mechanizm, przez który elementy potomne mogą dziedziczyć właściwości CSS od swoich elementów rodzicielskich.

Nie wszystkie właściwości CSS są dziedziczone przez elementy potomne.

Głównie dziedziczone są te związane z tekstem, takie jak **color**, **font-family**, czy **font-size**.

Przykładowo, właściwości dotyczące układu i pozycjonowania, jak **margin**, **border**, czy **padding**, nie są dziedziczone przez elementy potomne.

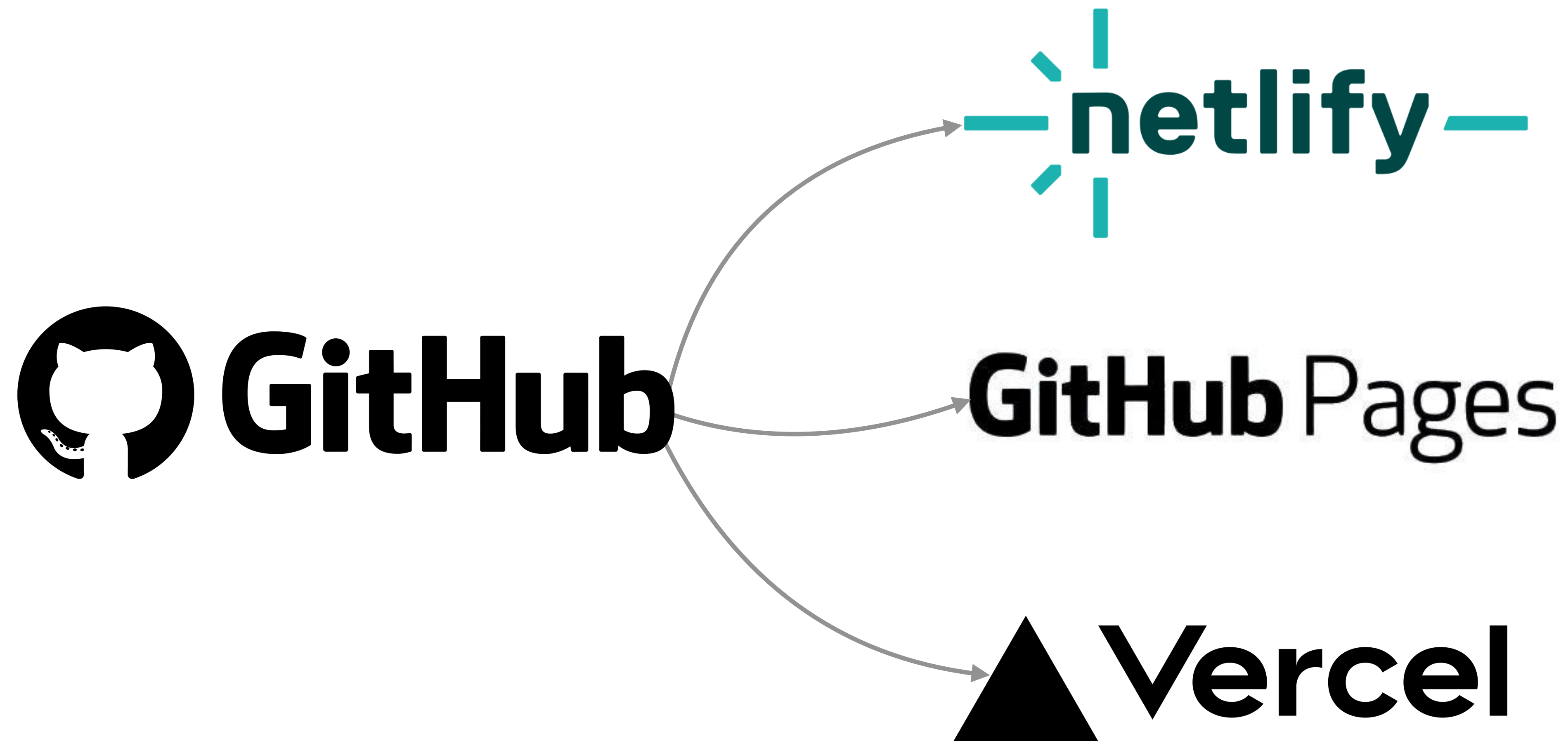


The screenshot shows a web browser's developer tools interface. On the left, the 'Elements' panel displays an HTML document structure. The root is `<html lang="en">`, containing a `<head>` and a `<body>`. Inside the `<body>`, there is a `<nav>` element with the text 'To jest element NAV', followed by an `<h1>` element with the text 'Nagłówek 1', and finally a `<p>` element with the text 'To jest akapit tekstu'. The `<h1>` element is selected, and its style is shown in the 'Styles' panel on the right. The 'Styles' panel shows the 'element.style' block, which is empty. Below it, the 'h1' block is shown with the following properties: `color: blueviolet;` and `font-size: 48px;`. Below the 'h1' block, the 'body' block is shown with the following properties: `color: #444444;`, `font-size: 16px;`, `font-family: Arial, sans-serif;`, and `border-top: 2px solid lightblue;`. An arrow points from the text 'Odziedziczone właściwości z body.' to the 'body' block in the 'Styles' panel. Another arrow points from the text 'Przekreślenie oznacza nadpisanie danej reguły.' to the 'h1' block in the 'Styles' panel.

Odziedziczone właściwości z body. →

Przekreślenie oznacza nadpisanie danej reguły.

# Platformy do publikacji i testowania stron www



# Dziękuję

Mateusz Janowski, 2024