European Funds
Knowledge Education Development
Wrocław University of Science and Technology
European Union
European Social Fund

ZPR PWr – Zintegrowany Program Rozwoju Politechniki Wrocławskiej

Aplikacje webowe na platformę .NET

W03 – CSS3, krótki przegląd możliwości

Syllabus

- CSS3 – przegląd podstawowych możliwości
  - Elementy wierszowe `<span>` i blokowe `<div>`
  - Format stylu
  - Arkusze stylów
  - Zapis koloru
  - Style dla czcionek
  - Podstawowe selektory
  - Złożone selektory
  - Hierarchia stylów
  - Pozycjonowanie elementów
  - Wyświetlanie elementów
  - Przykład pseudoklasy i jej użycia
  - Przegląd innych możliwości CSS3
  - Emmet w VS Code

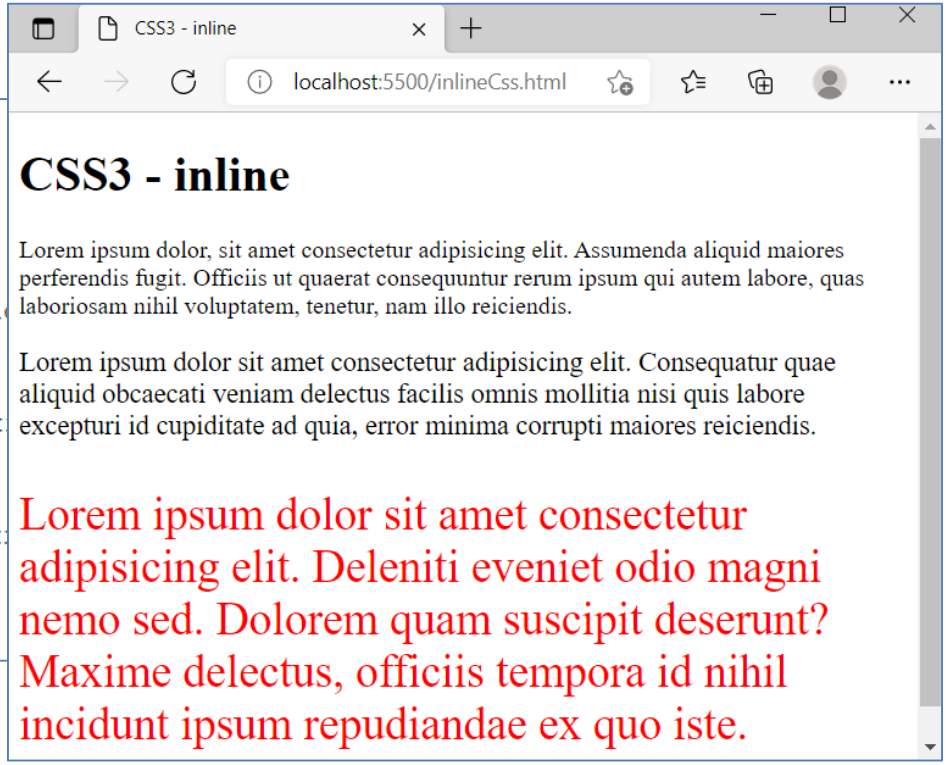
Elementy wierszowe `<span>` i blokowe `<div>`

- W ramach HTML istnieją elementy wierszowe typu `<strong>`, `<sup>`, `<em>`, `<img>` itd.
  - Mają nadane znaczenie (semantykę)
- Istnieje jeden *element **wierszowy*** `<span>` bez semantyki
- W ramach HTML istnieją elementy blokowe typu `<h1>`, `<p>`, `<section>` itd.
  - Mają nadane znaczenie (semantykę)
- Istnieje jeden *element **blokowy*** `<div>` bez semantyki.
- Praktycznie każdy element HTML może mieć nadaną **klasę** poprzez atrybut `class`.
  - Jest to po prostu identyfikator klasy
  - **NIE JEST** to klasa w rozumieniu programistycznym, znaczeniowo to raczej **zbiór, do którego należy dany element**
  - Element może należeć **do wielu klas**, wówczas (w cudzysłowiu) nazwy klas rozdzielamy spacją
- **Dodając** do elementów `<span>` i `<div>` **klasę** nadajemy im, w pewnym sensie, **znaczenie**.
- **Nazwa klasy** jest bardzo często używana w **selektorach CSS3**

Wstęp o stylach

- Prezentację wizualną (kolory, czcionki, położenie, zachowanie) elementów HTML można zmieniać.
- Służy do tego część atrybutów lub atrybut ogólny `style`.
  - jest to użycie **inline**.
  - Przykład: `inlineCss.html`

```
    <title>CSS3 - inline</title>
</head>
<body>
    <h1>CSS3 - inline</h1>
    <p>
        Lorem ipsum dolor, sit amet consectetur adipisi
    </p>
    <p style="font-size:large">
        Lorem ipsum dolor sit amet consectetur adipisic
    </p>
    <p style="font-size: 30px; color: red">
        Lorem ipsum dolor sit amet consectetur adipisic
    </p>
</body>
</html>
```



Format stylu

- Każdy element stylu składa się z pary:
  - Właściwości stylu CSS3
  - Wartości tej właściwości (nieraz nazywanej atrybutem stylu, lub wręcz atrybutem)
- Wartości te są rozdzielone dwukropkiem
- Kolejne pary stylu rozdzielone są średnikiem
  - Po ostatniej parze też może być średnik, co ułatwia edycję stylu.

```
Właściwość1: wartość1; Właściwość2: wartość2; … ; WłaściwośćN: wartośćN
```

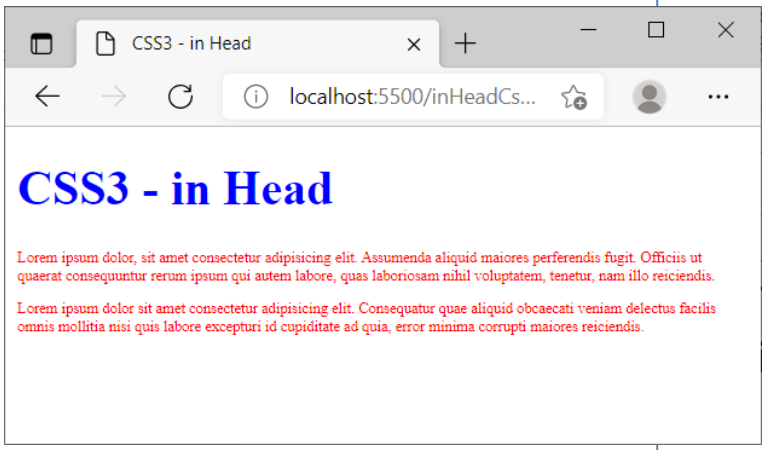
```
Właściwość1: wartość1;
Właściwość2: wartość2;
… ;
WłaściwośćN: wartośćN
```

```
Właściwość1: wartość1;
Właściwość2: wartość2;
… ;
WłaściwośćN: wartośćN;
```

Osadzone arkusze stylów CSS3 (w elemencie <head>)

- Zamiast sposobu inline można (powinno się) zebrać stylowania w pewien zestaw zasad – arkusz stylów.
- Arkusz stylów można określić w elemencie <head> bezpośrednio (jako **osadzone arkusze stylów**)
  - inHeadCss.html
  - Za pomocą elementu <style>

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>
        p {
            font-size: 10px;
            color:red
        }
        h1 {
            color:blue
        }
    </style>
    <title>CSS3 - in Head</title>
</head>
<body>
    <h1>CSS3 - in Head</h1>
    <p>
        Lorem ipsum dolor, sit ...
    </p>
    <p>
        Lorem ipsum dolor sit ...
    </p>
</body>
</html>
```

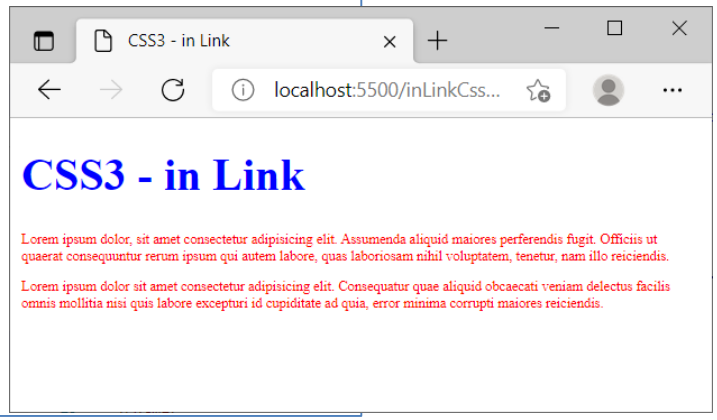


Zewnętrzne arkusze stylów CSS3

- Arkusz stylów może być też w innym pliku.
- Wówczas w elemencie `<head>` umieszcza się informację o arkuszu stylów za pomocą pustego elementu `<link>` z atrybutem `rel=„stylesheet”` oraz atrybutem `href` z linkiem do tego pliku.
  - `inLinkCss.html`
  - `inLink.css`
- Ta forma (**zewnętrzne arkusze stylów**) jest najbardziej wskazana.
- W osadzonych i zewnętrznych arkuszach stylów można umieszczać komentarze między /* a */.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="inLink.css">
    <title>CSS3 - in Link</title>
</head>
<body>
    <h1>CSS3 - in Link</h1>
    <p>
        Lorem ipsum dolor, sit amet ...
    </p>
    <p>
        Lorem ipsum dolor sit amet ...
    </p>
</body>
</html>
```

```css
p {
    font-size: 10px;
    color:red;
}
h1 {
    color:blue
}
```



Podstawowe właściwości w arkuszach stylów - kolor

- Kolory:
  - Właściwość `color` – kolor czcionki
  - Właściwość `background-color` – kolor tła

- Kolor można wyrazić jako:
  - Nazwę symboliczną koloru np. red, lightgreen, aqua
  - Kod szesnastkowy w formacie „`#RRGGBB`” np. `#AB081F`
  - Kod szesnastkowy w formacie „`#RGB`” np. `#A18`, co oznacza `#AA1188`
  - Krotki „`rgb(R,G,B)`” – każda z wartości w zakresie 0-255.
  - Krotki „`rgb(R%,G%,B%)`” – każda z wartości w zakresie 0-100.
  - Krotki „`hsl(H,S%,L%)`” – H – barwa 0-360 z koła barw, S - nasycenie 0-100, L – jasność 0-100.

Podstawowe właściwości w arkuszach stylów - czcionki

- Właściwość `font-weight` – grubość (waga) czcionki:
  - Predefiniowane wartości symboliczne absolutne: `bold`, `normal`,
  - Wartości liczbowe 0-1000, gdzie 400 to `normal`, a 700 to `bold`.
  - Wartości względne (dziedziczenie stylów): `bolder`, `lighter`
- Właściwość `font-size` – wielkość czcionki:
  - Predefiniowane wartości symboliczne absolutne: `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`.
  - Symboliczne wartości względne (dziedziczenie stylów): `smaller`, `larger`
  - Procentowe: `x%` wartości bieżącej czcionki
  - Wyrażone w wartościach dopuszczalnych **jednostek**

Jednostki używane w CSS3

- em – jednostka względna, 1em oznacza 100% wartości rodzica.
- px – piksele (monitora), nie nadają się do styli przygotowywanych dla wydruków
- pt – punkt (drukarski), teoretycznie miała uniezależnić od rozdzielczości ekranów, ale nadaje się głównie do styli dla wydruków, 1 in (cal) =72 pt
- pc – pica, 1/12 pt
- cm, mm, in – centymetr, milimetr, cal
- ex – wysokość czcionki (czyli małych liter a, e itp.)

Ogólne zalecenia:

- Ekran: em, px, %, sporadycznie ex (np. marginesy zależne od powiększenia czcionki)
- Druk: em, cm, mm, in, pt, pc, %, sporadycznie reszta.

Nowe jednostki:

- rem – bazowa wielkość czcionki (od początku dokumentu) do określania innych niż czcionka właściwości, np. marginesów
- vw – 1/100 szerokości okna
- vh – 1/100 wysokości okna

Podstawowe właściwości w arkuszach stylów - rodzina czcionki

- Właściwość `font-family` – ciąg rodzajów czcionek, które przeglądarka będzie starała się użyć, jeśli nie znajdzie pierwszej.
  - Kolejne rodzaje czcionek rozdzielone przecinkiem.
- Mogą to być bardzo konkretne typy, albo rodziny czcionek o pewnej właściwości
  - Najlepiej na końcu ciągu podać rodzinę czcionek.
- Rodziny czcionek i konkretne czcionki:
  - `Serif` – z ozdobnikami na końcach
    - `Times New Roman`, `Georgia`
  - `Sans-serif` – bez ozdobników
    - `Arial`, `Verdana`, `futura`
  - `Monospace` – o stałej szerokości znaków
    - `Courier new`, `fixedsys`
  - `Cursive` – pochyła
    - `script`
  - `Fantasy` – fantazyjna
    - `critter`

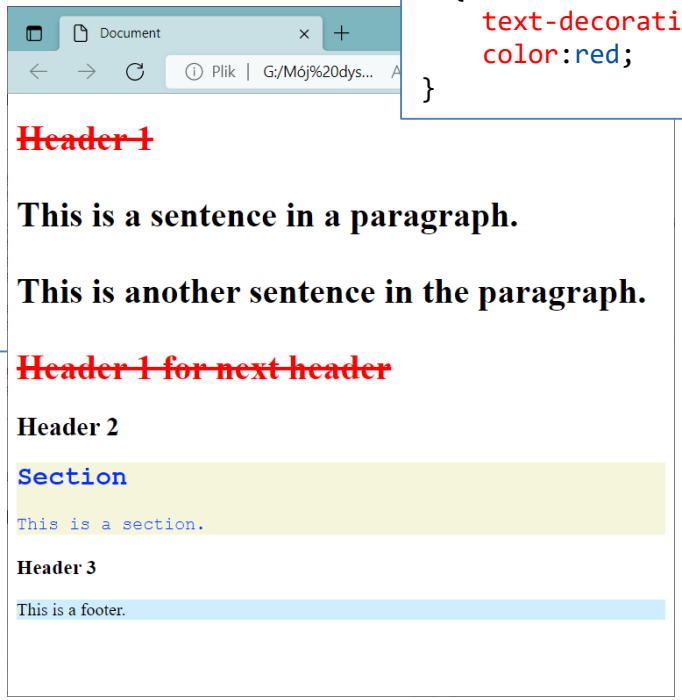
Dekoracja czcionki

- Właściwość `text-decoration` – narzuca dekoracje w tekście i jest skrótem do ustawienia w jednej właściwości trzech innych (w kolejności):
  - `text-decoration-line` (wymagane)
  - `text-decoration-color`
  - `text-decoration-style`
- Wartości dla `text-decoration-line`:
  - `underline` - podkreślenie
  - `overline` – linia ponad tekstem
  - `line-through` – przekreślenie na środku tekstu
  - `none` – bez linii (domyślnie)
- Wartości dla `text-decoration-color` to po prostu kolor dekoracji.
- Wartości `text-decoration-style` to:
  - `solid` – linia ciągła (domyślnie)
  - `double` – linia podwójna
  - `dotted` – linia kropkowana
  - `dashed` – linia kreskowana
  - `wavy` – linia falowana
- Inne możliwe wartości dla wszystkich powyższych właściwości to:
  - `initial` – powrót do ustawień domyślnych
  - `inherit` – dziedziczenie wartości z rodzica.
- Część tych wartości można łączyć np.:
  - `text-decoration: underline overline  wavy red;`
  - `text-decoration: underline line-through dashed blue;`

Przykład – czcionki (`fonts.html`, `fonts.css`)

```
    <link rel="stylesheet" href="fonts.css">
    <title>Document</title>
</head>
<body>
    <h1> Header 1</h1>
    <p>
        This is a sentence in a paragraph.
    </p>
    <p>
        This is another sentence in the paragraph.
    </p>
    <h1> Header 1 for next header</h1>
    <h2> Header 2</h2>
    <section>
        <h2>Section</h2>
        This is a section.
    </section>
    <h3> Header 3</h3>
    <footer>
        This is a footer.
    </footer>
</body>
</html>
```

```
p{
    font-size: 200%;
    font-weight: 800;
}
section{
    font-family: 'Courier New', Courier, monospace;
    background-color: beige;
    color:#0430FF;
}
footer{
    background-color:hsl(200,90%,90%)
}
h1{
    text-decoration: line-through;
    color:red;
}
```



Format arkusza stylów CSS3

- Jest to zestaw reguły następujących jedna po drugiej rozdzielonych białym znakiem
- Każda reguła to para:
  - Selektor CSS3
  - Styl opisany jak w sposobie *inline*, ale zawarty w nawiasy klamrowe

```
Selektor1 {
    własność1_1: wartość1_1;
    …
}
Selector2 {
    własność2_1: wartość2_1;
    …
}
…
```

```
p {
    font-size: 10px;
    color:red;
}
h1 {
    color:blue
}
```

Podstawowe selektory CSS3

- Nazwa elementu HTML:
  - Bez specjalnego znaku poprzedzającego
- Nazwa klasy
  - Poprzedzona kropką ‚.'
- Identyfikator
  - Poprzedzony znakiem hasz ‚#'
- Nazwa pseudoklasy, pseudoelementu
  - Poprzedzony znakiem dwukropka „:"
  - Dokładne wytłumaczenie na późniejszych slajdach
  - W przykładzie oznacza zdarzenie umieszczenia wskaźnika myszki nad elementem (tu całym dokumentem).
- Wszystkie elementy:
  - Znak mnożenia: ‚*'
- Przykłady:
  - `simpleSelectors.html`
  - `simpleSelectors.css`

```css
p {
    font-size: 15px;
}
.important{
    color:red;
}
#bottom{
    font-weight: 800;
}
:hover{
    background-color: yellow;
}
```


```html
    <link rel="stylesheet" href="simpleSelectors.css">
    <title>CSS3 - simple selectors</title>
</head>
<body>
    <h1 class="important">CSS3 - simple selectors</h1>
    <p class="important">
        Lorem ipsum dolor, sit ...
    </p>
    <p>
        Lorem ipsum dolor sit ...
    </p>
    <p id="bottom">
        Lorem ipsum dolor, sit ...
    </p>
</body>
</html>
```

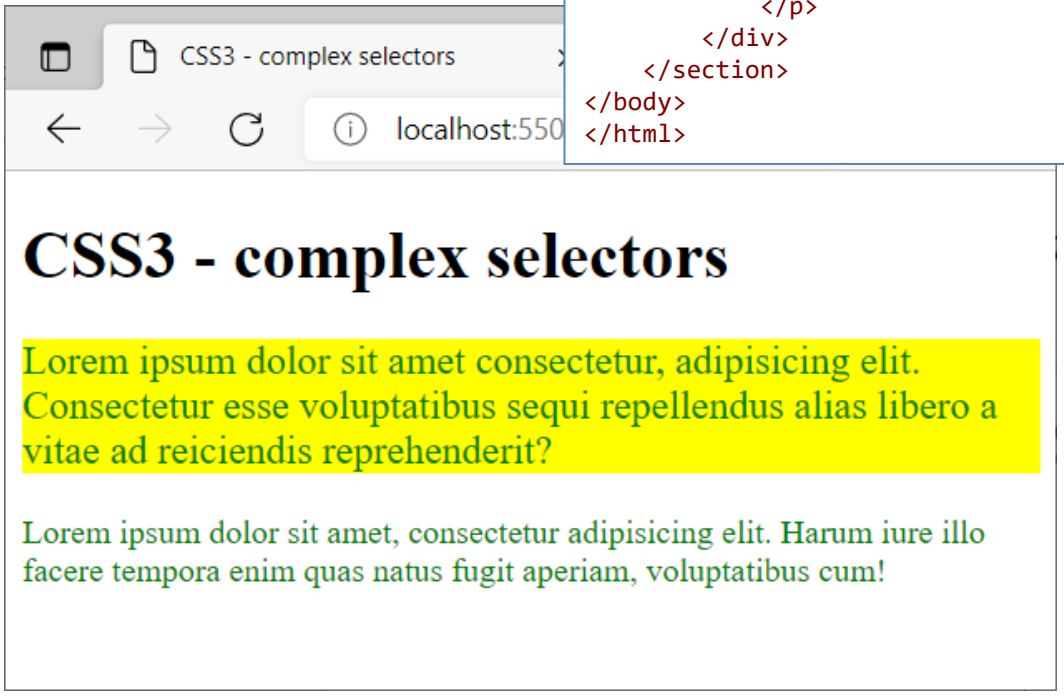
CSS3 – łączenie selektorów

- Siła CSS3 objawia się w tym, że selektor może być złączeniem innych selektorów. Łączenie to może się odbywać na wiele sposobów:
  - Poprzez przecinek „`,`”, co oznacza, że styl odnosi się z osobna do każdego z nich:
    - Np. `p, h1, strong {…}` oznacza, że styl zapisany w klamrach odnosi się do elementów `<p>`,`<h1>` i `<strong>`
  - Poprzez zapis bez przerw, co oznacza, że wszystkie selektory muszą być spełnione:
    - Np. `p.important.hard {…}` oznacza, że styl zapisany w klamrach odnosi się do elementów `<p>`, które muszą jednocześnie należeć do dwóch klas: `important` oraz `hard`.
  - Poprzez spację np. „`sel1 sel2`”, co oznacza, że odnosi się do elementów `sel2`, które są wewnątrz elementów `sel1` (gdziekolwiek wewnątrz).
  - Poprzez znak większości „`>`”, np. „`sel1 > sel2`”, co oznacza, że element `sel2` jest **bezpośrednio wewnątrz** elementu `sel1`.

Przykład złożonych selektorów – `complexSelectors(.html/.css)`

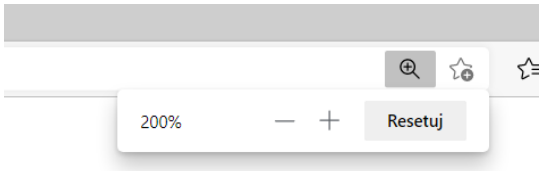
```css
section  p{
    color:green;
}
section > p{
    background-color: yellow;
}

p.important.hard{
    font-size: larger;
}
```

```html
    <link rel="stylesheet" href="complexSelectors.css">
    <title>CSS3 - complex selectors</title>
</head>
<body>
    <h1 class="important">CSS3 - complex selectors</h1>
    <section class="outer">
        <h2>Important hard</h2>
        <p class="important hard">
            Lorem ipsum dolor sit amet ...
        </p>
        <h2>Important medium</h2>
        <div class="inner">
            <p class="important medium">
                Lorem ipsum dolor sit amet ...
            </p>
        </div>
    </section>
</body>
</html>
```



Sprzeczne style

- Styl elementu HTML może być opisany przy pomocy wielu selektorów:
  - Jeśli są to różne właściwości, zastosowane zostaną wszystkie
  - Jeśli są to te same właściwości styli, następuje kolizja zwana **sprzecznymi stylami**.
- Dodatkowo style mogą być określone przez agenta użytkownika (przeglądarkę) lub samego użytkownika (w ustawieniach przeglądarki, ale też bezpośrednio np. poprzez powiększenie strony).



- Obowiązują ogólne reguły:
  - Style zdefiniowane przez użytkownika mają pierwszeństwo przed stylami zdefiniowanymi przez agenta użytkownika.
  - Style zdefiniowane przez autorów mają pierwszeństwo przed stylami zdefiniowanymi przez użytkownika.

Hierarchia stylów

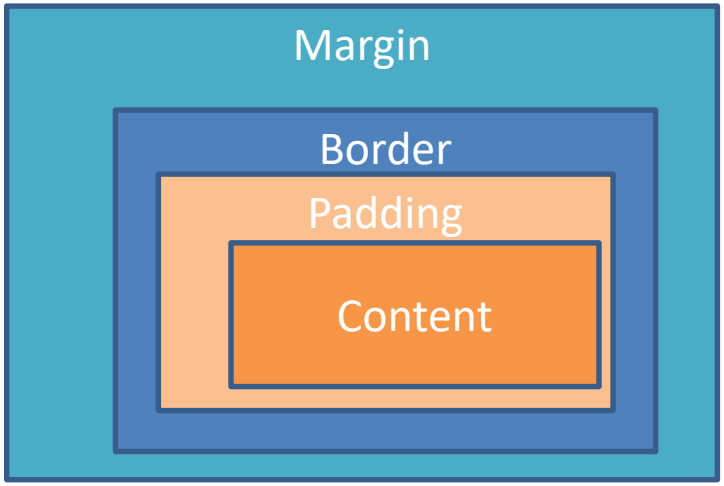
- Większość stylów zdefiniowanych dla elementów (rodziców) jest także **dziedziczonych** przez elementy zagnieżdżone (dzieci).
  - Stąd wyraz „kaskadowe” w CSS
  - Podejście jest logiczne, bo inaczej zmieniając np. `font-family` w `<body>` trzeba by potem we wszystkich elementach wewnętrznych powtarzać to ustawienie.

- Właściwości zdefiniowane dla dzieci (potomków) mają wyższą tzw. **swoistość** niż właściwości zdefiniowane dla rodziców (przodków).
- Konflikty są rozwiązywane na korzyść właściwości z **większą swoistością**, zatem style dziecka mają pierwszeństwo.
- Największą swoistość ma styl zapisany *inline*.
- Jeśli selektor jest bardziej szczegółowy, to ma większą swoistość.
- Jeśli dwa selektory są równoważne, to tekstowo ostatni analizowany ma większą swoistość.
  - Co pozwala używać zewnętrznych arkuszy stylów z wybranego frameworku, a potem dopisać własny arkusz z modyfikacjami stylów, które „nadpiszą” te z frameworku

Elementy wierszowe a blokowe

Standardowo (`position=static`):

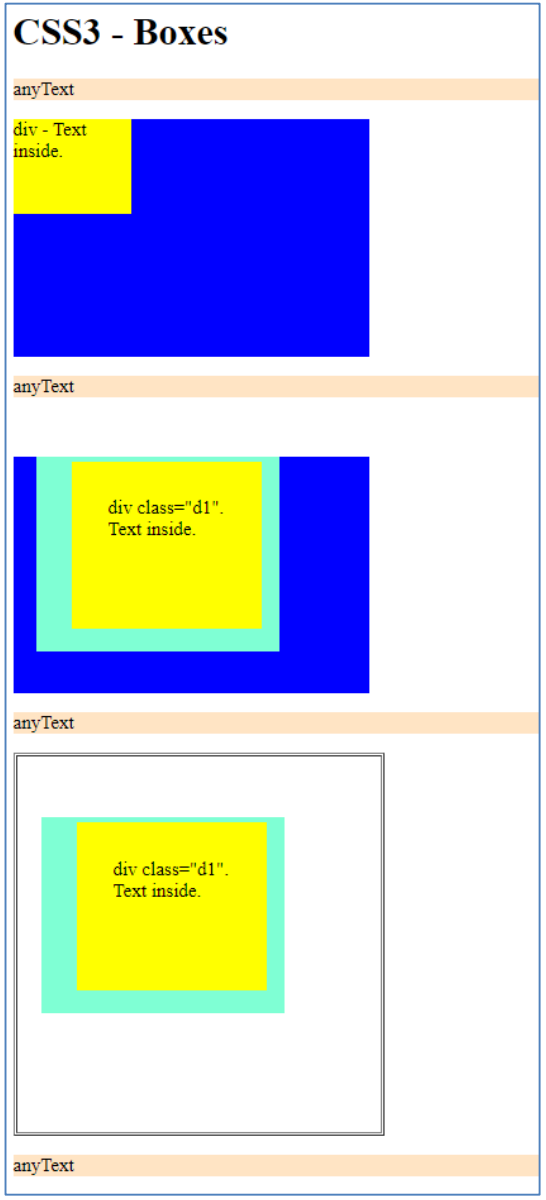
- Elementy wierszowe układane są po kolei liniami.
  - Wysokość linii wyznacza najwyższy element danej linii.
- Elementy blokowe zawsze są w nowej linii (zajmują całą jedną linię).
  - Można im dodać kolory, obrazki tła itp.:
    - Właściwość `background-image`
    - Właściwość `background-position`
    - Właściwość `background-repeat`
    - Właściwość `background-attachment`:
    - Właściwość `text-indent`
    - Właściwość `font-style`
  - Mają wirtualne pola wokół siebie:
    - `Padding` – odstęp od zawartości do ramki
    - `Border` - ramka
    - `Margin` – odstęp od ramki otaczającej do pozostałych
      - Różne działanie w zależności od miejsca elementu
- Szerokości wirtualnych pól (góra, dół, lewo, prawo) można ustawiać wspólnie lub niezależnie.



Przykład – pudełka – `elementBox (.html/.css)`

```css
p {
    background-color: bisque;
}
div {
    height: 80px;
    width: 100px;
    background-color:yellow;
}

.outer{
    height: 200px;
    width: 300px;
    background-color:blue;
}

.d1{
    margin:50px 80px 100px 20px;
    border-color: aquamarine;
    border-width: 5px 15px 20px 30px;
    border-style: solid;
    padding: 30px;
}
```

```html
    <link rel="stylesheet" href="elementBox.css">
    <title>CSS3 - Boxes</title>
</head>
<body>
<h1 class="important">CSS3 - Boxes</h1>
<p>anyText</p>
<div class="outer">
    <div>
        div - Text inside.
    </div>
</div>
<p>anyText</p>
<div class="outer">
    <div class="d1">
        div class="d1". Text inside.
    </div>
</div>
<p>anyText</p>
<table border="1">
<tbody>
<tr>
    <td>
        <div class="d1">
            div class="d1". Text inside.
        </div>
    </td>
</tr>
<tbody>
</table>
<p>anyText</p>
</body>
</html>
```

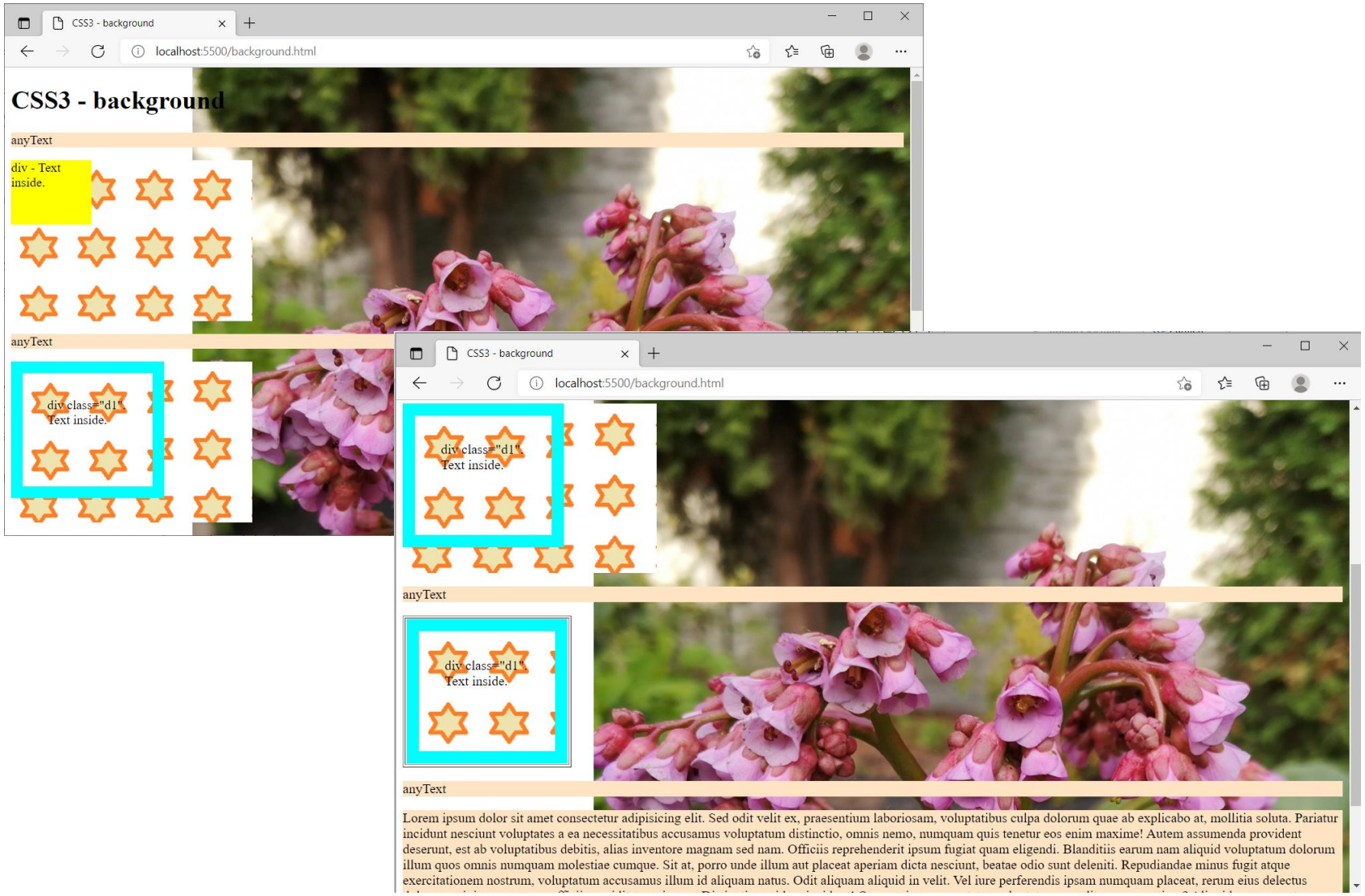


Przykład - tła – background(.html/.css) 1/2

```
    <link rel="stylesheet" href="backgorund.css">
    <title>CSS3 - background</title>
</head>
<body>
    <h1 class="important">CSS3 - background</h1>
    <p>anyText</p>
    <div class="outer">
        <div>
            div - Text inside.
        </div>
    </div>
    <p>anyText</p>
    <div class="outer">
        <div class="d1">
            div class="d1". Text inside.
        </div>
    </div>
    <p>anyText</p>
    <table border="1">
    <tbody>
    <tr>
        <td>
            <div class="d1">
                div class="d1". Text inside.
            </div>
        </td>
    </tr>
    </tbody>
    </table>
    <p>anyText</p>
    <p>
        Lorem ipsum dolor sit ...     </p>
</body>
</html>
```

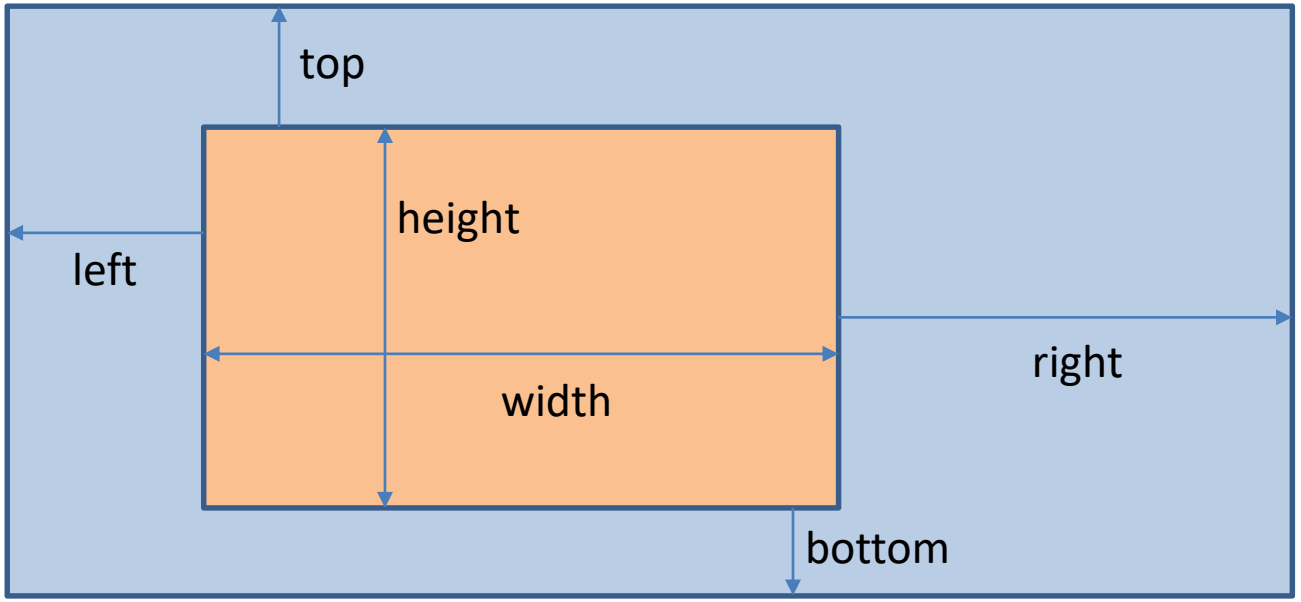
```
body{
    background-image: url("bergenia.jpg ");
    background-repeat: no-repeat;
    background-position: right top;
    background-attachment: fixed;
}
p {
    background-color: bisque;
}
div {
    height: 80px;
    width: 100px;
    background-color:yellow;
}

.outer{
    height: 200px;
    width: 300px;
    background-image: url("star.png");
}

.d1{
    border: 15px solid aqua;

    padding: 30px;
    background-image: url("star.png");
}
```

Przykład – tła 2/2



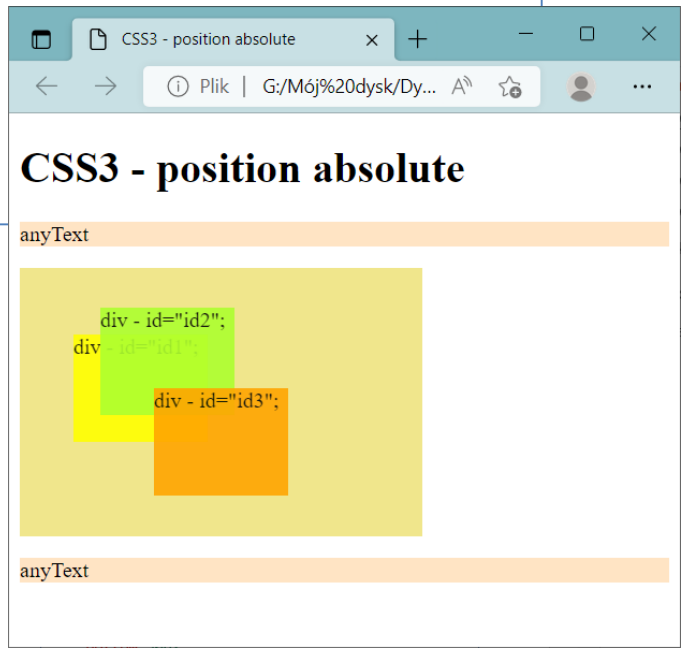
Pozycja absolutna elementu blokowego

- Właściwość `position=absolute`
  - Oznacza pozycję absolutną
  - Usuwa element ze standardowej kolejki elementów na stronie i umieszcza w odpowiedniej odległości od górnego, lewego, prawego lub dolnego brzegu jego elementu-przodka (rodzica).
  - Wielkość elementu można określić właściwościami CSS: `top`, `left`,`bottom`, `right` (znaczenie opisane wcześniej), `width`, `height`. Zbiór tych właściwości jest nadmiarowy, więc powinno się wybrać 4 z nich.
- Właściwość `opacity=x%` to przezroczystość elementu (x=100 – brak przezroczystości, x=0 – całkowita przezroczystość, niewidoczny)



Przykład – pudełka – elementBox (.html/.css)

```
    <link rel="stylesheet" href="absolute.css">
    <title>CSS3 - position absolute</title>
</head>
<body>
    <h1 class="important">CSS3 - position absolute</h1>
    <p>anyText</p>
    <div class="outer">
        <div id="id1">
            div - id="id1";
        </div>
        <div id="id2">
            div - id="id2";
        </div>
        <div id="id3">
            div - id="id3";
        </div>
    </div>
    <p>anyText</p>
</body>
</html>
```




```
p {
    background-color: bisque;
}
div {
    height: 80px;
    width: 100px;
}
.outer{
    height: 200px;
    width: 300px;
    background-color: khaki;
    position: relative; /* cannot be static */
}
.outer  div{
    position:absolute;
    opacity: 90%;
}
#id1{

    background-color: yellow;
    top:50px;
    left:40px;
}
#id2{
    background-color: greenyellow;
    top:30px;
    left:60px;
}
#id3{
    background-color: orange;
    bottom:30px;
    right:100px;
}
```

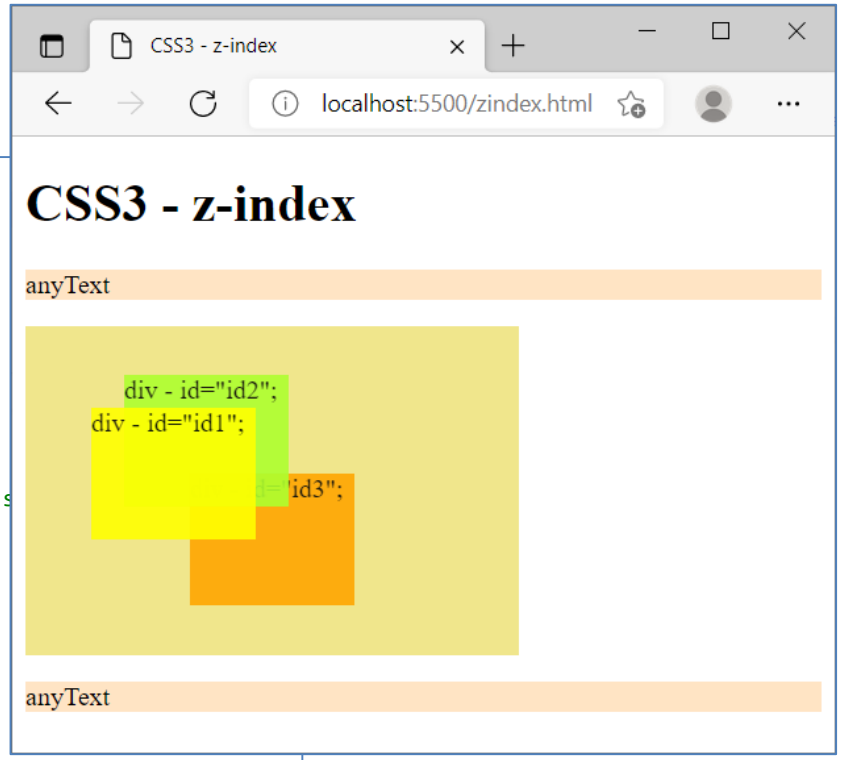
Warstwy

- W pozycjonowaniu absolutnym pojawia się problem zakrywania elementów. Standardowo kolejny element w tekście HTML będzie rysowany **na** poprzednich.
- Kolejność można zmienić ustawiając właściwość `z-index` na odpowiednią liczbę całkowitą.
- Element z wyższym `z-index` rysowany jest ponad elementami na niższym indeksie.

Przykład pozycjonowania absolute i z-index – `zindex (.html/.css)`

```
    <link rel="stylesheet" href="zindex.css"
type="text/css">
    <title>CSS3 - z-index</title>
</head>
<body>
    <h1 class="important">CSS3 - z-index</h1>
    <p>anyText</p>
    <div class="outer">
        <div id="id1">
            div - id="id1";
        </div>
        <div id="id2">
            div - id="id2";
        </div>
        <div id="id3">
            div - id="id3";
        </div>
    </div>
    <p>anyText</p>
</body>
</html>
```

```
p {
    background-color: bisque;
}
div {
    height: 80px;
    width: 100px;
}
.outer{
    height: 200px;
    width: 300px;
    background-color: khaki;
    position: relative; /* cannot be s
}
.outer  div{
    position:absolute;
    opacity: 90%;
}
#id1{
    z-index: 2;
    background-color: yellow;
    top:50px;
    left:40px;
}
#id2{
    z-index: 1;
    background-color: greenyellow;
    top:30px;
    left:60px;
}
#id3{
    background-color: orange;
    bottom:30px;
    right:100px;
}
```



Pozycjonowanie względne

- Pozycjonowanie względne to właściwość `position=relative`
- Element „rezerwuje” to samo miejsce, co zajmowały oryginalnie, ale jego obraz przesuwa się względem oryginalnej pozycji (używając dwóch właściwości spośród: `left`,`right`, `top`, `bottom`).
  - `left=15px` to przesunięcie **od lewego brzegu** wyjściowej pozycji o 15px, czyli przesunięcie bloku w prawo o 15px
  - Można używać liczb ujemnych lub przeciwstawnych właściwości, czyli `left=15px` jest równoważne `right=-15px`.
  - Domyślnie wszystkie te właściwości są równe 0.

Przykład dla pozycji względnej – relative (.html/.css)

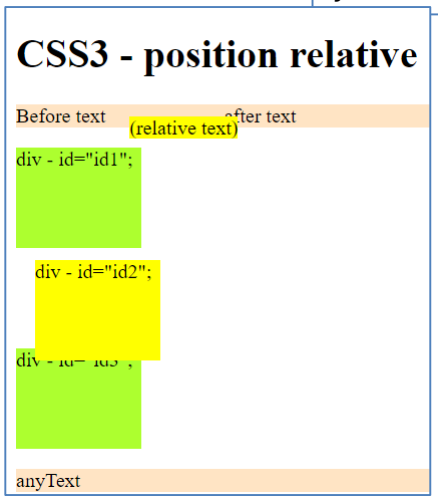
```
    <link rel="stylesheet" href="relative.css">
    <title>CSS3 - position relative</title>
</head>
<body>
    <h1 class="important">CSS3 - position relative</h1>
    <p>Before text <span class="moved">(relative text)</span> after text</p>
    <div id="id1">
        div - id="id1";
    </div>
    <div id="id2" class="moved">
        div - id="id2";
    </div>
    <div id="id3">
        div - id="id3";
    </div>
    <p>anyText</p>
</body>
</html>
```

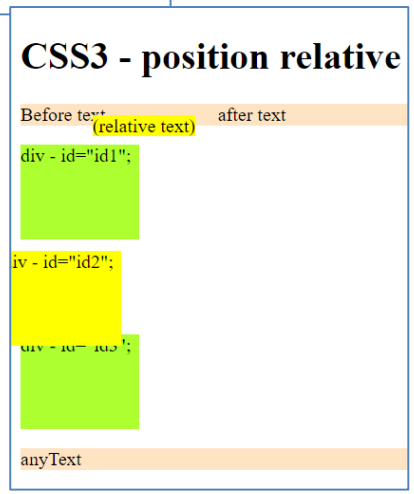
```
p {
    background-color: bisque;
}
div {
    height: 80px;
    width: 100px;
    background-color: greenyellow;
}
```

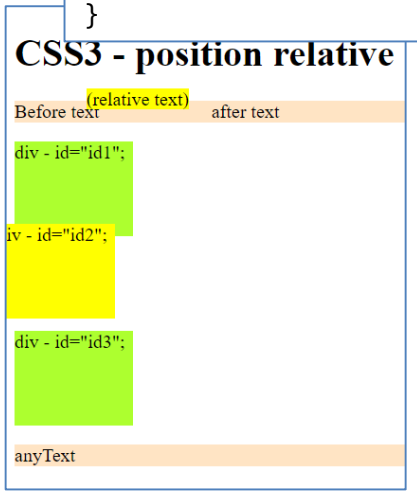
```
.moved{
    position: relative;
    background-color: yellow;
    top:10px;
    left:15px;
}
```

```
.moved{
    position: relative;
    background-color: yellow;
    top:10px;
    left:-15px;
}
```

```
.moved{
    position: relative;
    background-color: yellow;
    top:- 10px;
    left:-15px;
}
```







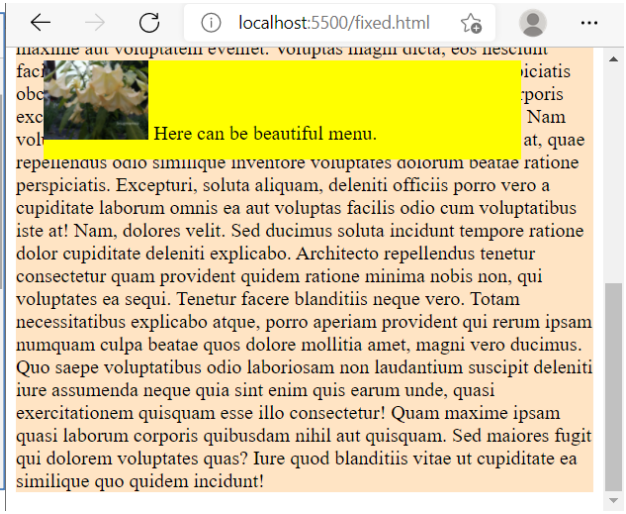
Pozycjonowanie ustalone

- Pozycjonowanie ustalone to właściwość `position=fixed`
  - Podobnie jak dla tła
- Pozycja element jest wyznaczana z ewentualnym użyciem 4 (lub mniej) z 6 (`top`, `left`,`bottom`, `right`, `width`, `height`) właściwości **względem viewportu** (części okna przeglądarki dla strony WWW)
  - Przesuwanie suwakami treści strony nie powoduje przesunięcia takiego elementu
  - Przydatne do zrobienia górnego/lewego menu lub dolnej stopki. Trzeba jednak pamiętać, aby na stronie było „pusto” w miejscu takiego menu.

Przykład – pozycja ustalona – `fixed (.html/.css)`

```
    <link rel="stylesheet" href="fixed.css">
    <title>CSS3 - position fixed</title>
</head>
<body>
    <h1>CSS3 - obscured title</h1>
    <div id="menu">
        <img src="brugmansia.jpg" alt="brugmansia flower">
        Here can be beautiful menu.
    </div>
    <h1>CSS3 - position fixed</h1>
    <p>
        Lorem ipsum dolor sit ...
    </p>
</body>
</html>
```

```
p {
    background-color: bisque;
}
#menu img{
    height: 80%;
}
#menu{
    position: fixed;
    background-color: yellow;
    top:10px;
    left:30px;
    height: 80px;
    width: 80%;
}
```

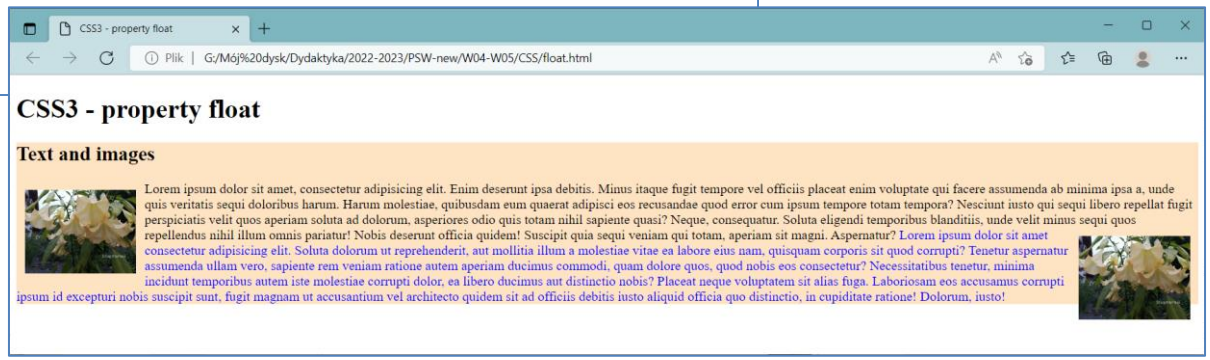


Elementy pływające

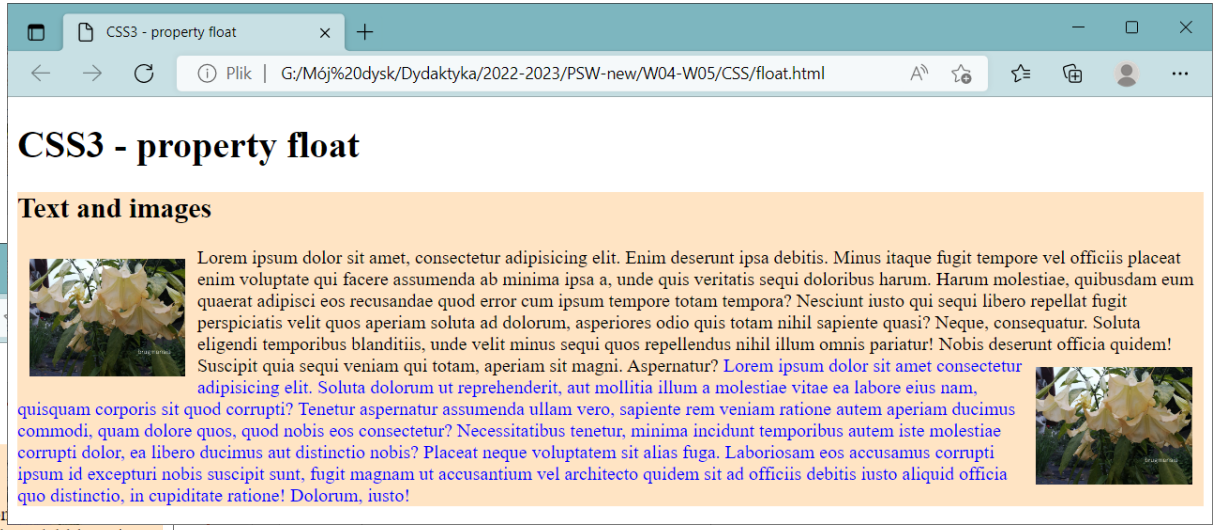
- Właściwość `float` pozwala dosunąć element do wybranego brzegu (`left`, `right`, `top`, `bottom`) elementu rodzica, a pozostałe elementy będą otaczać ten element.
- Przykład:
  - `float (.html/.css)`

```
    <link rel="stylesheet" href="float.css" type="text/css">
    <title>CSS3 - property float</title>
</head>
<body>
    <h1>CSS3 - property float</h1>
    <section>
        <h2>Text and images</h2>
        <img class="image-left" src="brugmansia.jpg" height="100"
alt="brugmansia flower">
        Lorem ipsum dolor sit …
        <img class="image-right" src="brugmansia.jpg" height="100"
alt="brugmansia flower">
        <span style="color:blue">
        Lorem ipsum dolor sit ...
        </span>
    </section>
</body>
</html>
```

```
section {
    background-color: bisque;
}
.image-left{
    float:left;
    text-align: center;
    margin:10px;
}
.image-right{
    float:right;;
    text-align: center;
    margin:10px;
}
```



Inne rozmiary okienka



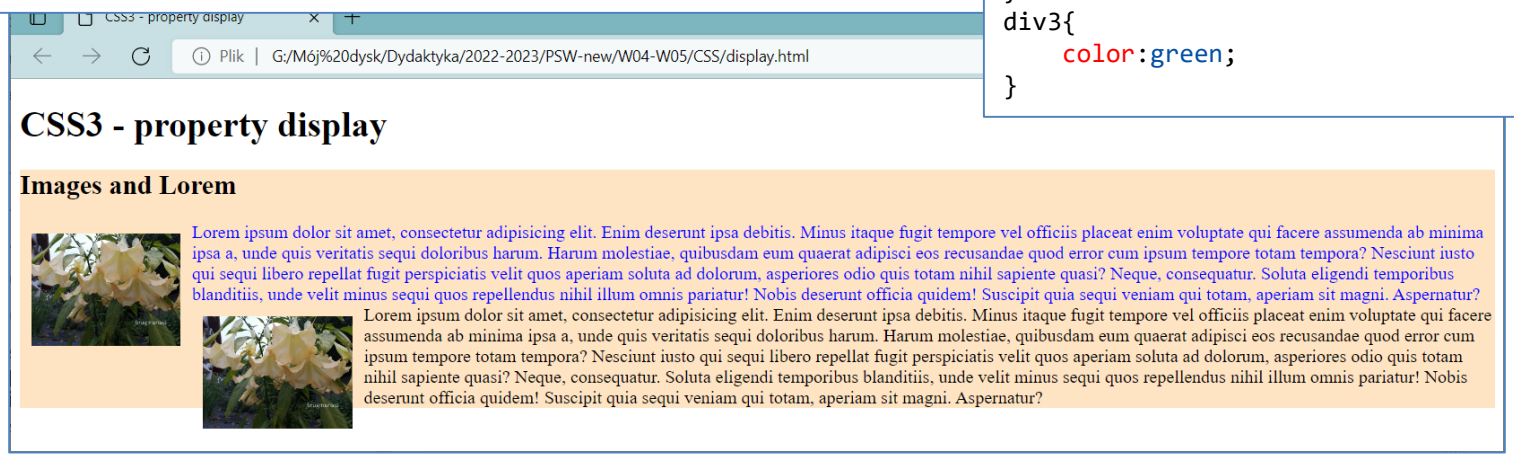
Właściwość `display`

- Właściwość display pozwala programiście decydować, czy element ma być wyświetlany jako element blokowy (`block`), element wierszowy (`inline`) lub czy w ogóle nie ma być renderowany (`none`).
  - Można zamienić domyślny sposób renderowania, co nie jest wskazane, raczej powinno się używać wartości `none` z wartością właściwą dla elementu.
  - Jeśli `display=none` to również wszystkie elementy potomne takiego elementu nie są widoczne
- Istnieje jeszcze kilkanaście wartości tej właściwości,np. `inline-block`, `run-in`, `table` czy `table-foot-group`.

Przykład display – `display (.html/.css)` 1/2

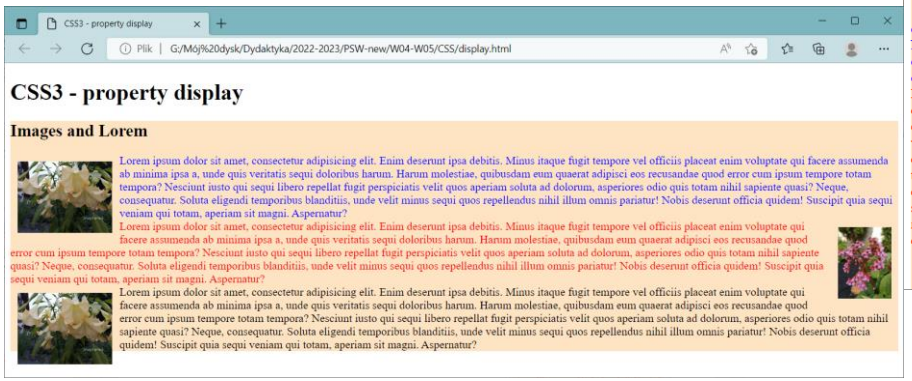
```
    <link rel="stylesheet" href="display.css" type="text/css">
    <title>CSS3 - property display</title>
</head>
<body>
    <h1>CSS3 - property display</h1>
    <section>
        <h2>Images and Lorem</h2>
        <div id="div1">
            <img class="image-left" src="brugmansia.jpg" height="100" alt="
            Lorem ipsum dolor sit ...
        </div>
        <div id="div2">
            <img class="image-right" src="bergenia.jpg" height="100" alt="b
            Lorem ipsum dolor sit ...
        </div>
        <div id="div3">
            <img class="image-left" src="brugmansia.jpg" height="100" alt="
            Lorem ipsum dolor sit …
        </div>
    </section>
</body>
</html>
```

```
section {
    background-color: bisque;
}
.image-left{
    float:left;
    margin:10px;
}
.image-right{
    float:right;;
    margin:10px;
}

#div1{
    color:blue;
}
#div2{
    display:none; /* display:block/none; */
    color:red;
}
div3{
    color:green;
}
```



Przykład display 2/2


```css
section {
    background-color: bisque;
}
.image-left{
    float:left;
    margin:10px;
}
.image-right{
    float:right;;
    margin:10px;
}

#div1{
    color:blue;
}
#div2{
    display:block; /* display:block/none; */
    color:red;
}
div3{
    color:green;
}
```



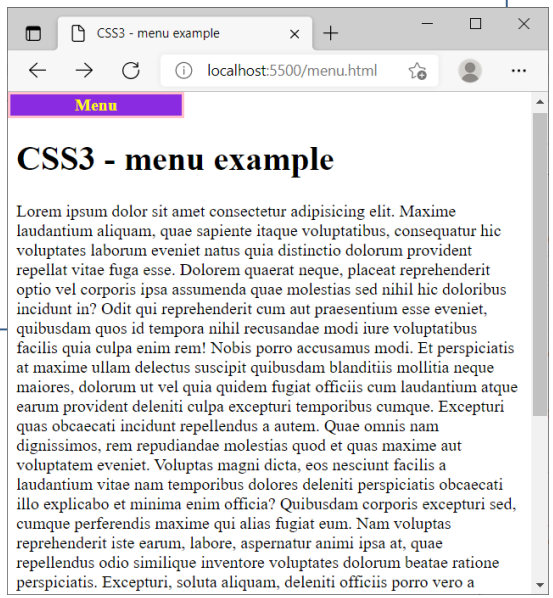
Pseudoklasa `hover`

- Pozwala na wykrycie zdarzenia polegającego na przesunięcie wskaźnika myszki w obszar wybranego elementu.
- Styl powiązany z takim **selektorem** pozwala na dynamiczną reakcję np. pokazanie ukrytych elementów itp.

- Przykład: stworzenie z elementu `<nav>` menu reagującego na pozycję myszki nad nim.
  - Element `<nav>` wstawiony zostanie w lewy górny róg i będzie miał ustalone wymiary (i inne graficzne właściwości).
  - Oprócz tekstu ”`Menu`” będzie zawierał listę (`<ul>`/`<li>`) linków do innych stron (`<a>`)
  - Lista `<ul>` wewnętrzna dla `<nav>` będzie ukryta (`display:none`), jednak po ustawieniu kursora myszki nad napis „`Menu`” (selector `nav:hover ul`) ma się pojawić.
  - Elementy `<li>` będą miały własny styl, ale ich kolor tła ma się zmienić po ustawieniu myszki na wybranym z nich (selector `nav ul li:hover`), aby użytkownik łatwiej widział, co ma zamiar wybrać.
  - Należy wyłączyć standardowe dekoratory tekstu dla linków `<a>`.
  - Za pomocą klasy `top-gap` zrobione zostanie miejsce odstępu dla tworzonego górnego menu.

Przykład – menu – `menu (.html/.css)` 1/2

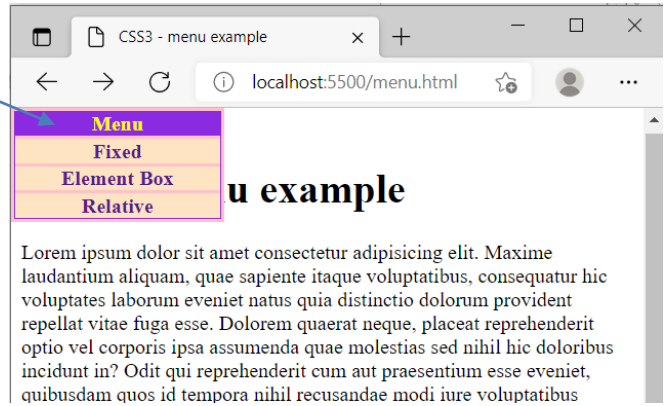
```
    <link rel="stylesheet" href="menu.css" type="text/css">
    <title>CSS3 - menu example</title>
</head>
<body>
    <div class="top-gap"></div>
    <h1>CSS3 - menu example</h1>
    <nav> Menu
        <ul>
            <li>
                <a href="fixed.html"> Fixed</a>
            </li>
            <li>
                <a href="elementBox.html"> Element Box</a>
            </li>
            <li>
                <a href="relative.html"> Relative </a>
            </li>
        </ul>
    </nav>


    <p>
        Lorem ipsum dolor
    </p>
</body>
</html>
```




```
.top-gap{
    height: 20px;
}
nav{
    position:fixed;
    left:0px;
    top:0px;
    font-weight: bold;
    color: yellow;
    border: 3px solid pink;
    text-align: center;
    width: 10em;
    background-color:blueviolet;
}
nav ul{
    display: none;
    list-style: none;
    margin: 0;
    padding: 0;
}
nav:hover ul{
    display: block;
}
nav ul li{
    border-top: 3px solid pink;
    background-color:bisque;
    width: 10em;
    color: black;
}
nav ul li:hover{
    background-color: aqua;
}
a{
    text-decoration: none;
}
```

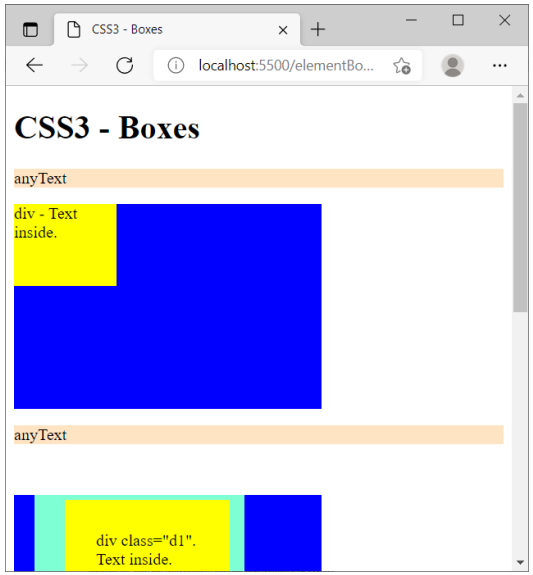
Przykład menu 2/2

Wskaźnik myszki



Wskaźnik myszki

Po kliknieciu myszką



Inne elementy CSS3

- Selektor następstwa elementów (na tym samym poziomie drzewa).
- Selektory z porównywaniem wartości **dowolnego atrybutu** elementu np. `input[type="url"]`.
- `position=sticky`
- Flexbox-y, czyli `display=flex`. Oraz **kilkanaście** innych wartości tej właściwości.
- Inne selektory pseudoklas:
  - `link`
  - `visited`
  - `active`
  - `focus`
  - `first-child`
  - `last-child`
  - `nth-child(n)`
  - itd..
- Selektory pseudoelementów (poprzedzane podwójnym dwukropkiem , **: :** '):
  - `first-line`
  - `first-letter`
  - `before`
  - `after`
- Właściwości typu `min_width`, `max-width`, itd.
- Właściwości zależne od przeglądarki zaczynające się `webkit-`, `moz-`
- Przekształcenia geometryczne elementów: przeskalowanie, obrót, odbicie, skośność
- Animacje opierające się na przekształceniach geometrycznych

- Narzędzie programistyczne w przeglądarce pozwala na dostęp do stylu elementu oraz jego zmianę (w lokalnej kopii strony).

Wiadomości związane z CSS3

- Część właściwości w ramach styli można osiągnąć poprzez atrybuty elementów:
  - Standard zakłada zasady, które z nich są ważniejsze, ale najlepiej używać tylko jednych z nich.
- CSS walidator
  - jigsaw.w3.org/css-validator/ pomaga w sprawdzeniu, czy kod CSS3 jest poprawny i będzie działał na przeglądarkach interpretujących go wg standardów.
- W CSS nie ma stałych ani operacji na liczbach, więc nie można napisać , że jedne elementy mają mieć długość x, a inne 2*x. Nie można też zagnieżdżać definicji styli
  - Rozwiązanie: Preprocesory CSS (kompilują kod w swoim języku na poprawny CSS
  - Less - https://lesscss.org/
  - Sass/Scss https://sass-lang.com/guide
  - VS Code posiada rozszerzenia do kompilacji tych języków
- https://www.w3.org/Style/CSS/

Emmet – VS Code

- https://docs.emmet.io/
- Składnia selektorów jest bardzo podobna do narzędzia Emmet, w którym pisząc znaczniki, atrybuty lub zawartość w jednej linii w odpowiednim formacie i kończąc klawiszem <tab>/<enter> można bardzo szybko wytworzyć szkielet fragmentu strony.
- Skróty emmet są również dostępne w pliku CSS
- Przykłady możliwości (poniższe screeny z tego adresu):
  - https://blog.greenroots.info/10-vs-code-emmet-tips-to-make-you-more-productive

```
ul>li.list>a.link
```

```
<ul>
    <li class="list">
        <a href="" class="link"></a>
    </li>
</ul>
```

```
ul>(li>a)*5
```

```
<ul>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
</ul>
```

Emmet - przykłady

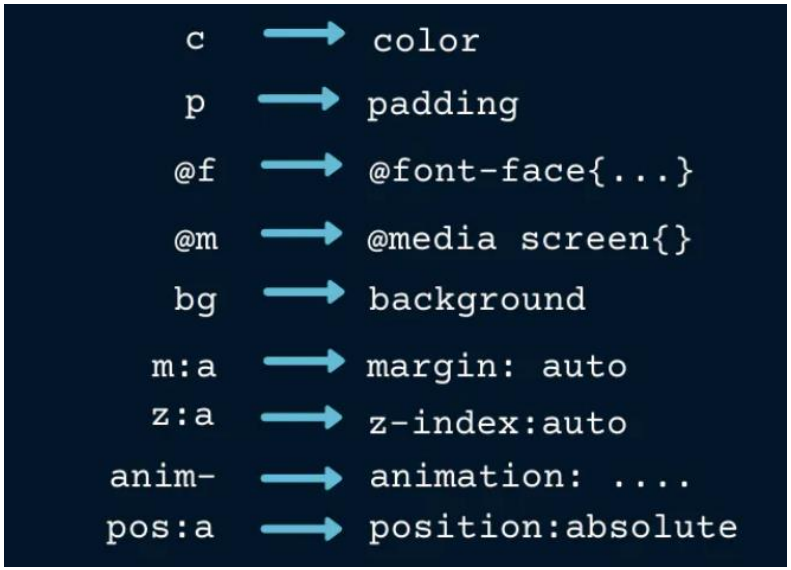
```
div>(header>ul>li*2>span.item)+section.content+(footer>(p>Lorem)*2)
```

```
<div>
    <header>
        <ul>
            <li><span class="item"></span></li>
            <li><span class="item"></span></li>
        </ul>
    </header>
    <section class="content"></section>
    <footer>
        <p>Lorem ipsum dolor ...</p>
        <p>Veritatis, molestiae corporis ...</p>
    </footer>
</div>
```

```
div>div>h3+span^div{I can climb up}
```

```
<div>
    <div>
        <h3></h3>
        <span></span>
    </div>
    <div>I can climb up</div>
</div>
```

Emmet w CSS



- Oficjalna strona nt. skrótów Emmet w VS Code:
  - https://code.visualstudio.com/docs/editor/emmet

CSS

DODATEK

Inne selektory

- Selektor elementów sąsiadujących bezpośrednio:
  - `h1 + p {…}`
  - Pierwszy element `<p>` umieszczony za elementem `<h1>`
- Ogólny selektor elementów sąsiadujących:
  - `h1 ~ p {…}`
  - Wszystkie elementy `<p>` sąsiadujące (posiadające tego samego rodzica) z elementem `<h1>`