Tworzenie Stron Internetowych

odcinek 8

CSS – jednostki

Jednostki miary stosuje się w deklaracjach dotyczących np. wielkość czcionki lub rozmiarów marginesów. Zapis składa się z znaku "**+**" (domyślny) lub "**–**", liczby oraz jednostki. Miedzy liczną a jednostką nie można wstawiać spacji. Po wartości **0** jednostka jest opcjonalna. Przykład: +1cm.

Jednostki względne:

- **em** - wysokość aktualnej czcionki
- **vw, vh** – procent szerokości/wysokości okna przeglądarki (1vw = 1%)
- **vmin**, **vmax** - względem 1% mniejszego/większego wymiaru okna przeglądarki
- **px** - piksele ekranowe (zależne od urządzenia wyświetlającego)
- **%** - procent rozmiaru elementu nadrzędnego

Względne jednostki długości stosujemy, gdy chcemy się odwołać do rozmiaru innego elementu (np. wysokości aktualnie używanej czcionki).

Jednostki bezwzględne:

- **in** - cal (1in = 2.54cm)
- **cm** – centymetr
- **mm** – milimetr
- **pt** - punkt (1pt = 1/72in) - często używane przy definiowaniu rozmiaru czcionki
- **pc** - pika (1pc = 12pt)

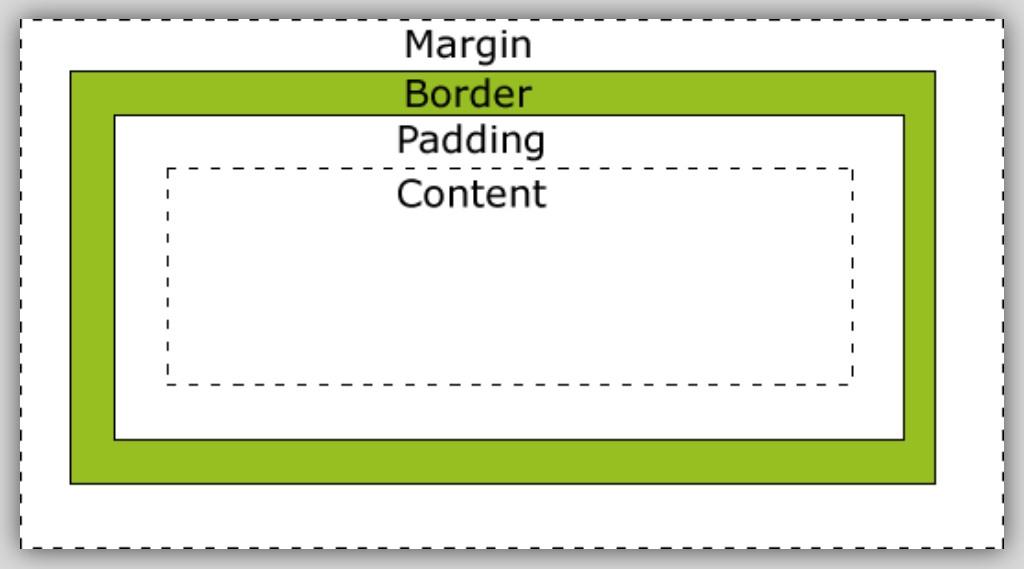
Jednostek bezwzględne używamy wtedy, gdy chcemy, aby wybrany element zajmował zawsze taki sam obszar wydruku lub ekranie (niezależnie od rozdzielczości i wielkości monitora).

CSS – model pudełkowy

Każdy element generuje w dokumencie prostokątny obszar zwany pudełkiem. Pudełko składa się z:

- **content** – właściwa zawartość pudełka (np. test, obraz)
- **padding** – margines wewnętrzny (odstęp)
- **border** – obramowanie
- **margin** – margines (zewnętrzny)

Dzięki temu można każdy element otoczyć ramką i określić przestrzeń pomiędzy elementami.



Uwagi:

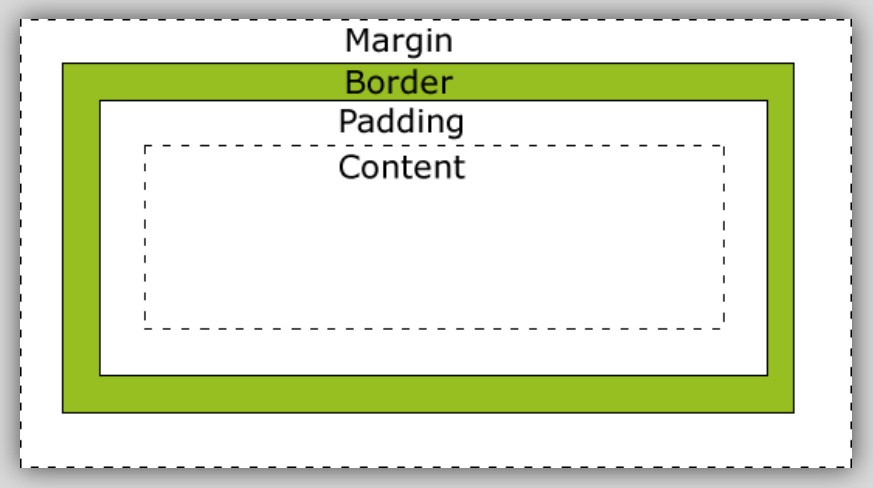
- Ostatnie trzy składniki są opcjonalne, tzn. mogą mieć wartość zero. Rozmiar każdego z czterech boków tych składników można zdać inny.
- Obwód zewnętrzny każdego z czterech obszarów nazywamy krawędzią (**edge**).
- Rozmiary elementu (cechy **width** oraz **height**) odnoszą się do samej **zawartości (content)** i… *cdn. za chwilę.*
- Tło elementu jest określone dla wszystkich obszarów z wyjątkiem marginesów zewnętrznych, które zawsze są przezroczyste.

CSS – model pudełkowy

Rozmiary elementu (cechy **width** oraz **height**) odnoszą się do samej **zawartości (content)** i nie wpływają na pozostałe obszary pudełka. Na całkowity rozmiar pudełka składa się więc rozmiar wszystkich jego składników:

Całkowita szerokość = width + lewy padding + prawy padding + lewy border + prawy border + lewy margin + prawy margin

Całkowita wysokość = height + górny padding + dolny padding + górny border + dolny border + górny margin + dolny margin



Przykład:

```
background-color: powderblue;
width:250px;
padding:10px;          /* każdy z czterech boków po 10px */
border:5px solid gray; /* każdy z czterech boków po 5px (cechy łączone, można je rozdzielić) */
margin:10px;           /* każdy z czterech boków po 10px */
```

Całkowita szerokość elementu wynosi:

250px + 2×10px (padding) + 2×5px (border) + 2×10px (margin) = 300px

CSS – model pudełkowy

Outline (kontur/obrys) to linia otaczająca element na zewnątrz obramowania umożliwiająca jego uwypuklenie. Obrys i obramowanie nie są tym samym.

Nie można oddzielnie regulować cech boków obrysu (wszystkie będą takie same). Obrys nie zabiera miejsca (jest tworzony na elemencie), więc rozmiar outline nie wpływa na rozmiar całkowity elementu, ale może nachodzić na sąsiednie elementy.

```
outline: kolor styl szerokość
```

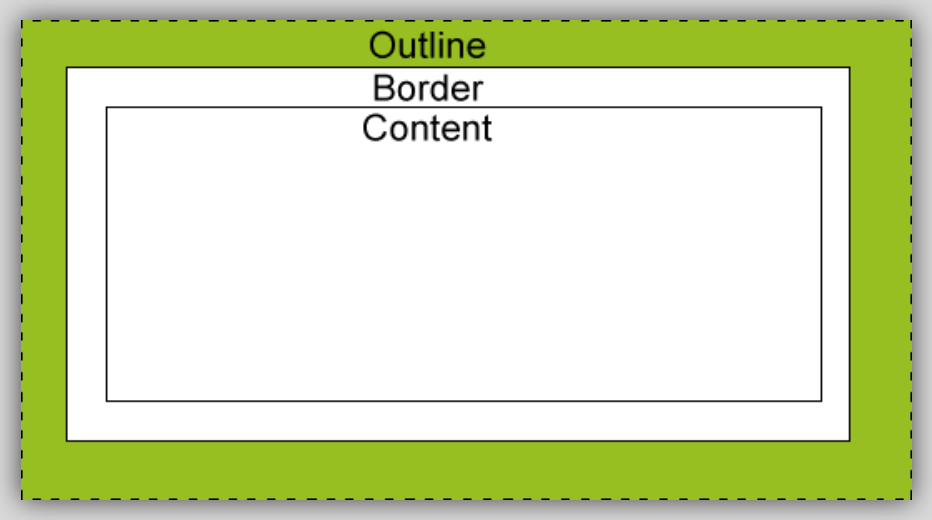
przykład:

```
p {border: 1px solid black;
   outline: green dotted 3px;}
```

lub po rozbiciu na osobne cechy:

```
outline-style: dotted;
outline-color: green;
outline-width: 3px;
```

musi być podana



CSS – podstawowe cechy

Margines

```
div {margin: 10px}
```

Powyżej deklaracja dla wszystkich stron marginesu. Można podać osobno dla każdej strony lub tylko dla wybranych z nich:

```
div {magrin-top: 5vh}
div {magrin: 5px 10px 15px 10px}
```

górny prawy dolny lewy

Odstęp

```
td {padding: 5px}
```

Powyżej deklaracja dla wszystkich stron odstępu. Można podać osobno dla każdej strony lub tylko dla wybranych z nich (podobnie jak dla margin)

CSS – podstawowe cechy

Obramowanie

Obramowanie elementu jest nadawane cechą border. Cecha obejmuje szerokość, styl i kolor obramowania. Możliwe jest odniesienie stylu do każdego boku osobno.

Przykłady:

```
p {border: 5px solid red;}
```
łączone cechy obramowania: grubość, styl, kolor

```
p {border-top-style: dotted;
   border-right-style: solid;
   border-bottom-style: dotted;
   border-left-style: solid;}
```
tylko styl obramowania, rozdzielony na boki

```
p {border-bottom: 5px dashed red;
   background-color: lightgrey;}
```
tylko łączony styl dla jednego boku
plus tło

```
p {border: 2px solid red;
   border-radius: 5px;}
```
obramowanie…
… z zaokrągleniem rogów

Możliwe jest też użycie obrazu jako elementu tworzącego ramkę (border-image).

CSS – podstawowe cechy

Wysokość i szerokość

Podanie tych cech dla elementu blokowego zapobiega zajęciu przez niego całej szerokości okna lub elementu, w którym się znajduje.

```
div {height: 200px; width: 50%;}
```

Zadana szerokość może być jednak większa od szerokości okna. Wtedy…

Określenie maksymalnej szerokości elementu: max-width

Jeśli okno przeglądarki ma szerokość mniejszą niż zadana maksymalna, element będzie zwężony. Cecha **width** nie umożliwia takiego dopasowania – przeglądarka w tym przypadku dodaje pasek przewijania.

Przykład:

```
div {max-width: 500px;
     height: 300px;
     background-color: lightyellow;}
```

CSS – podstawowe cechy

Nadanie tła: background

Tło elementu może być kolorem (jednolity, gradient) lub obrazem. Następujące cechy umożliwiają nadanie tła:

- **background-color** – tło wypełnione kolorem, np.: body {background-color: lightblue;}
- **background-image** – tło obrazkowe, np.: background-image: url("obraz.gif");
- **background-repeat** – powtarzanie tła w kierunku x lub y lub w ogóle, np.: background-repeat: repeat-x; lub background-repeat: no-repeat;
- **background-attachment** – możliwość zablokowania obrazu tła przy przewijaniu zawartości strony: background-attachment: fixed;
- **background-position** – położenie obrazu tła, np.: background-position: left top;
- **background: linear-gradient** – tło kolorowe z gradientem, np. background: linear-gradient(lightblue, gray)

Uwaga: obraz tła nie może utrudniać czytania tekstu

Cechy tła można podać kodem skróconym:

background: color image repeat attachment position

Poza użyciem nazw kolorów, w CSS kolory można zadawać w następujących modelach: RGB, HEX, HSL, RGBA i HSLA (to odnosi się nie tylko do kolorów tła).

CSS – położenie

Cechy opisujące położenie w CSS umożliwiają umieszczenie elementów HTML w wybranym miejscu, wsunięcie elementu za inny element lub wskazanie co ma się stać z niemieszczącą się zawartością elementu.

Metody pozycjonowania:

Statyczne (static) – domyślna, elementy są ułożone tak jak są wpisane w HTML

Względne (relative) – przesuwa element względem położenia domyślnego o zadaną wartość. Miejsce zajmowane w położeniu domyślnym nie zostaje zwolnione.

```
h1 {position:relative; left:-20px}
```

Ustalone (fixed) – położenie elementów jest zadane względem okna przeglądarki (viewport). Nie podlegają przesunięciom nawet przy przewijaniu strony. Miejsce domyślne zostaje zwolnione.

```
p {position:fixed; top:30px; right:5px}
```

Bezwzględne (absolute) – ustala pozycję elementu względem najbliższego elementu-przodka, który ma pozycjonowanie inne niż statyczne. Jeśli brak takiego elementu, to punktem odniesienia jest <body>. Elementy pozycjonowane w ten sposób nie wpływają na położenie innych elementów.

```
img {position:absolute; left:100px; top:150px}
```

Przyczepne (sticky) – pozycjonowanie elementu zależy od położenia suwaka. Po przewinięciu strony i wysunięciu miejsca z elementem, przyczepia się on do viewport jak w position:fixed.

```
div{position: -webkit-sticky; /* Safari */  position: sticky;    top: 0;}
```

Pozycjonowanie ustalane jest poprzez zadanie cech left, right, top i bottom oraz przypisanie im odpowiedniej wartości.

CSS – położenie

Nakładanie elementów (przesłanianie)

Elementy pozycjonowane inaczej niż statycznie mogą nakładać się na inne elementy. Porządkowanie nakładania umożliwia cecha **z-index**, która podaje kolejność umieszczenia w kierunku przód-tył. Elementy o większym z-index znajdują się przed tymi o mniejszym. Z-index może przyjmować wartości dodatnie i ujemne (liczby całkowite).

```
img {position:absolute; left:0px; top:0px; z-index:-1}
```

Przyleganie

Cecha **float** pozwala przesunąć element na lewo lub prawo, co umożliwia innym elementom blokowym ustawienie się obok. Przesunięcie jest wykonywane maksymalnie w lewo (left) lub w prawo (right). Elementy następujące po przesuniętym dostawią się do niego. Cecha ta jest użyteczna np. przy budowie szablonu strony.

```
img {float:right}
div {float:left}
```

Jeśli następne elementy nie mają dosuwać się do poprzedzającego je elementu z cechą float, należy tę cechę wyłączyć. Wyłączenie może dotyczyć strony lewej (left), prawej (right) lub obu (both).

```
p {clear:both}
```

CSS – położenie

Ustawienie w poziomie

Istnieje kilka sposobów na ustawienie elementów **blokowych** HTML w kierunku poziomym.

- **Wyśrodkowanie** elementów można uzyskać wykorzystując cechę **margin**. Nadając jej wartość **auto** dla marginesu lewego i prawego wskazujemy, że wolna przestrzeń powinna być równo podzielona pomiędzy lewą i prawą stronę. Uwaga: element musi być węższy niż 100%.

```
div.center
{margin-left:auto;
margin-right:auto;
width:70%;
background-color:#b0e0e6}
```

- Ustawienie elementów **do lewej lub prawej strony** można uzyskać wykorzystując cechę **float** i nadając jej wartość **left** lub **right**.

```
div.right
{float:right;
width:300px;
background-color:#b0e0e6}
```

Ustawienie w pionie

Cecha **vertical-align** pozwala przesuwać elementy w pionie. Przyjmuje ona następujące wartości: baseline, middle, top, bottom, text-top, text-bottom, super, sub lub podanie jednostek miary.

```
span{vertical-align: 3mm},    span{vertical-align: -10%},    span{vertical-align: sub}
```

CSS – wyświetlanie

Przepełnienie elementu

W przypadku, gdy zawartość elementu nie mieści się w zadanym rozmiarze, to szerokość elementu (width) zostaje zachowana, a wysokość (height) jest powiększana, o ile height nie jest określone.

Jeśli height jest zadane, to można, używając cechy **overflow**, określić się co ma się stać z niemieszczącą się zawartością. Dostępne są następujące opcje:

- **visible** – zawartość wystaje poza obszar elementu
- **hidden** – zawartość niemieszcząca się jest niewidoczna
- **scroll** – zawartość jest ograniczona do obszaru elementu, ale jest dostępna w całości, bo do elementu dodawane są paski przewijania
- **auto** – działa podobnie jak scroll, ale suwaki dodawane są tylko, jeśli potrzebna.

```
div {width: 150px; height: 150px; overflow: scroll; border: 1px dotted black}
```

Chechy **overflow-x** oraz **overflow-y** pozwalają na określenie działania w przypadku niemieszczącej się zawartości osobno w kierunku poziomym i pionowym.

CSS – wyświetlanie

Wyświetlanie elementu

Cechy **display**, **visibility** i **opacity** pozwalają sterować wyświetlaniem elementów.

- **display** – pozwala wyświetlać elementy blokowe tak jakby były liniowymi (inline), a liniowe jak blokowe (block). **Uwaga:** to nie zmienia ich charakteru, a jedynie sposób wyświetlania.

  ```
  li {display:inline}
  span {display:block}
  ```

  Wartość list-item pozwala wyświetlić element tak jak element listy nieuporządkowanej.

  ```
  p {display:list-item}
  ```

  Użycie wartości none skutkuje niewyświetleniem elementu, a miejsce po nim zajmują inne elementy

  ```
  h3 {display:none}
  ```

  Obrazy są wstawiane liniowym elementem <img>. Aby uzyskać ich wyśrodkowanie, należy skorzystać z cechy **display** z wartością pozwalającą wyświetlić elementy liniowe tak, jakby były blokowymi.

  ```
  img {display: block;
       margin-left: auto;
       margin-right: auto;
       width: 50%;}
  ```

- cecha **visibility** z wartością hidden również powoduje niewyświetlenie elementu, ale w przeciwieństwie do display:none puste miejsce niewyświetlonego elementu zostaje zachowane.

  ```
  h3 {visibility:hidden}
  ```

  Domyślna wartość tej cechy (visible) powoduje wyświetlenie elementu.

CSS – wyświetlanie

Wyświetlanie elementu

- **opacity** – pozwala ustawić przezroczystość elementu (obrazka, tła); cecha ta przyjmuje wartość od 0.0 (całkowicie przezroczysty) do 1.0 (całkowicie nieprzezroczysty)

```
img
  {opacity:0.4;
  filter:alpha(opacity=40)}  /* dla IE8 i wer. wcześniejszych */
```

wartość tej cechy może być zmieniana przy interakcji z użytkownikiem, np.:

```
img
  {opacity:0.4;
  filter:alpha(opacity=40)}
img:hover
  {opacity:1.0;
  filter:alpha(opacity=100)}
```

Zastosowanie opacity do tła bloku z tekstem powoduje (niestety) transparentność tego tekstu (efekt dziedziczenia). Można ten problem rozwiązać stosując zapis kolorów przez RGBA (red, green, blue, alpha). Alpha określa przezroczystość.

```
div {background: rgba(76, 175, 80, 0.3)}
```

CSS – wyświetlanie

Wyświetlanie elementu

- **inline-block** – elementy z taką cechą zachowują się jak liniowe, ale nadal mogą mieć cechy width i height. Pozwala to np. utworzyć siatkę obszarów (kolumny, wiersze), które zapełniają szerokość okna przeglądarki, dostosowując automatycznie liczbę kolumn do tej szerokości.

  Przykład:

```
div.grid {display: inline-block;
width: 150px;
height: 75px;
margin: 10px;
border: 3px solid #73AD21;}
```

  Do tego przykładu, w kodzie HTML trzeba umieścić kilka bloków div, np.:

```
<div class="grid">jakaś zawartość bloku 1</div>
<div class="grid">jakaś zawartość bloku 2</div>
<div class="grid">jakaś zawartość bloku 3</div>
<div class="grid">jakaś zawartość bloku 4</div>
…
<div class="grid">jakaś zawartość bloku N</div>
```

CSS3 – nowości (wybrane)

Cecha **border** wzbogacona o:

- **border-radius** – zaokrąglenia rogów ramki

```
div
{border: 2px solid;
border-radius: 25px}
```

- **box-shadow** – imitacja cienia pod ramką

```
div
{width: 300px;
height: 100px;
background-color: blue;
box-shadow: 10px 10px 15px #888888}
```

x, y wysunięcia cienia, rozmycie cienia, jego kolor

Cień można również ustawić za tekstem korzystając z cechy **text-shadow**

```
h4
{text-shadow: 5px 5px 5px #FF0000}
```

x, y wysunięcia cienia, rozmycie cienia, jego kolor

Efekty cienia można dodawać, np.:

```
h1
{text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF}
```

CSS3 – nowości (wybrane)

Inne ciekawe cechy:

- **transform** – cecha pozwalająca przesuwać, obracać, zniekształcać elementy.

  Dla przykładu: obrót o N stopni (+ - w prawo, – - w lewo) – transform:rotate(*N*deg)

```
div
{ transform: rotate(25deg);
-webkit-transform: rotate(25deg);      z prefiksem dla Chrome i Safari
-moz-transform: rotate(25deg);         z prefiksem dla Firefoxa
-o-transform: rotate(25deg);           z prefiksem dla Opery
-ms-transform: rotate(25deg);          z prefiksem dla IE9
width: 200px; height: 100px; background-color: #a3acb8}
```

  Prefiks dodawany jest dla poprawnego działania kodu w starszych wersjach przeglądarek.

  Więcej o **transform** można przeczytać na www.w3schools.com

CSS3 – nowości (wybrane)

Inne ciekawe cechy:

- **transition** – stopniowa zmiana wyglądu elementu (rozmiar, kolor, obramowanie, …). Przykład:

1. Blok kodu (definicja stanu początkowego i sposobu zmiany).

   Blok określa pierwotny rozmiar pojemnika div oraz sposób zmiany rozmiaru (tu szerokość, width) i czas jego trwania (2s). Klasa .zmiana wskazuje, który div ma podlegać tej zmianie.

```
div.zmiana
{ width: 400px;
  height: 100px;
  background-color: #d1d1d1;
  border:  5px solid black;
  transition: width 2s;
  -webkit-transition: width 2s;      z prefiksem dla Chrome i Safari
  -o-transition: width 2s}           z prefiksem dla Opery
```

2. Blok kodu (stan końcowy).

   Blok definiuje szerokość pojemnika po umieszczeniu nad nim kursora myszy (pseudoklasa :hover).

```
div.zmiana:hover
{ width: 300px;
  height: 100px;
  background-color:  #4fc1ff;
  border:  5px solid red}
```

Więcej o **transition** można przeczytać na www.w3schools.com

CSS3 – nowości (wybrane)

Inne ciekawe cechy:

- **@keyframes**, **animation** – tworzenie animacji, np.:

1. Definicja obiektu i animacji

```
div.pasek
{ width:100px;
height:50px;
background-color:green;
animation: pulsowanie 3s infinite alternate;
-webkit-animation: pulsowanie 3s infinite alternate; }
```

2. Definicja klatek

```
@keyframes pulsowanie
{ from {width:100px;} to {width:300px;} }
@-webkit-keyframes pulsowanie
{ from {width:100px;} to {width:300px;} }
```

Więcej o **animation** można przeczytać na www.w3schools.com

Uwaga: przez użyciem nowych cech CSS, należy sprawdzić czy i które wersje przeglądarek je poprawnie interpretują.