Arkusze reguł stylistycznych CSS

Beata Pańczyk

1

Plan wykładu

- Co to jest CSS
- Kaskadowość arkuszy CSS
- Sposoby osadzania CSS w kodzie HTML
- Box Model
- Deskryptory
- Klasy i wyjątki
- Formaty selektorów CSS
- Standardy W3C
- SEO

2

CSS

- Kaskadowe arkusze stylów (*CSS - Cascading Style Sheets*) są nieodłącznym towarzyszem współczesnej wersji języka HTML/XHTML.
- Język HTML/XHTML odpowiada za strukturę tworzonej witryny internetowej i poszczególnych dokumentów (ich stronę semantyczną)
- Język CSS odpowiada za wizualną prezentację stron internetowych w przeglądarkach

3

Co to jest CSS?

- arkusz reguł stylistycznych (ang. *cascading style sheet*) jest kolekcją reguł formatujących, które mogą odnosić się do wielu dokumentów HTML; spełniają funkcję wzorca, pozwalając na zapewnienie takiego samego wyglądu wszystkim wystąpieniom danego elementu;
- powszechnie zaakceptowanym i ustandaryzowanym mechanizmem specyfikacji stylów dla HTML jest CSS (standardy CSS1, CSS2 , CSS3)
- CSS umożliwia znacznie dokładniejszą kontrolę sposobu wyświetlania elementów aniżeli sam HTML
- nie wszystkie mechanizmy CSS3 są zaimplementowane w dostępnych obecnie przeglądarkach

4

CSS i HTML

- CSS umożliwia przypisanie określonego stylu dowolnemu elementowi dokumentu HTML (np. akapitowi, komórce tabeli)
- Przeniesienie opisów formatowania elementów do arkuszy stylów - dokument składa się z "czystych" znaczników HTML
- Zalety stosowania arkuszy w HTML: lepsza czytelność dokumentu i szybsze ładowanie się witryn
- Style pozwalają kontrolować np. rodzaj, rozmiar i kolor czcionki, wyrównanie akapitu, wielkość marginesów, głębokość wcięcia akapitu, kolor i grafikę tła, odstępy między elementami
- Definicję stylu umieszcza się w dokumencie HTML lub w zewnętrznym pliku

5

Rodzaje arkuszy stylów

- **Importowany** (*imported*) - zewnętrzny arkusz stylów, dołączony do innego arkusza
- **Zewnętrzny** (*external*) - plik tekstowy z rozszerzeniem .css zawierający definicje stylów (odwołania w dowolnym pliku HTML)
- **Osadzony** (*embedded*) - w nagłówku pliku HTML (działa tylko w danym dokumencie)
- **Wpisany** (*in-line*) - dołączony do jednego znacznika, określa sposób jego wyświetlania

6

Kaskadowość stylów

- Kaskadowość - hierarchia źródeł stylów
- Priorytety stylów w kolejności
  - wpisany
  - osadzony
  - zewnętrzny
  - importowany
- W przypadku konfliktu (element znajduje się w zasięgu więcej niż jednego stylu) - styl znajdujący się "bliżej" elementu znosi działanie stylu "odległego"

7

Dołączanie styli do HTML

- atrybut **<style>** elementów w sekcji **<body>** - styl wpisany dla danego znacznika HTML
  np. **<p style="color: blue">**
- reguły stylistyczne zagnieżdżone w dokumencie HTML za pomocą znacznika **<style>** w sekcji **<head>**
- reguły stylistyczne w oddzielnym pliku, wskazanym znacznikiem (najbardziej preferowany)
  <link rel="stylesheet" href="plik lub url" media="screen">
  (domyślnie media= "all" , dostępne też media= "print")
- reguły importowane

8

Atrybut STYLE elementów HTML – przykład 1

```
<html><body>
<div>
 Zwykły tekst
 <h1>Widok nagłówka H1
 </h1>
 <p>Widok tekstu akapitu </p>
</div>
</body></html>
```

```
<html><body>
<div>
 Zwykły tekst
 <h1 style="color:blue;">
    Widok nagłówka H1</h1>
 <p style="font-size:18pt;
    margin-left:20pt;">
    Widok tekstu akapitu </p>
</div></body></html>
```

Znacznik STYLE w sekcji HEAD – przykład 2


```
<html>
<head>
  <style type="text/css">
  <!--   p { font-size:18pt;
                margin-left:20pt; }
         h1 { color:blue; }        -->
  </style>
</head>
<body>
<div> Zwykły tekst
  <h1>Widok nagłówka h1</h1>
  <p> Widok tekstu akapitu</p>
  <h1> Widok kolejnego nagłówka h1 </h1>
  <p> Widok kolejnego tekstu akapitu</p>
</div>
</body></html>
```

10

CSS w pliku zewnętrznym – przykład 3

- Taki sam efekt jak w przykładzie 2 można uzyskać definiując plik zewnętrzny np. ***styl1.css*** a następnie umieścić odwołanie w odpowiednim dokumencie HTML (np. ***css-p3.html***)
- zawartość pliku ***styl1.css:***
  p { font-size:18pt; margin-left:20pt; }
  h1 { color:blue; }
- zawartość dokumentu ***css-p3.html*** :

```
<html>
    <head>
    <link rel="stylesheet" href="styl1.css"type="text/css">
    </head>
<body>
    ... zawartość jak w przykładzie 2
</body></html>
```

11

Importowanie styli

```
<!-- prolog na początku -->

<head>
<!-- style w sekcji head -->
 <style type="text/css">
   @import url(http://www.xyz.pl/style.css)
 </style>
</head>

<!-- dalsza część dokumentu -->
```

12

Reguły Stylistyczne CSS

- reguła CSS zawiera dwie części: **selektor** (np. h1) i **deskryptor** (np. "color:blue"); **deskryptor** ma dwie części: **właściwość** (np. color) i **wartość** (np. blue)
- **selektor** składa się z jednego lub więcej prostych selektorów połączonych **łącznikami** (spacja, >, +)

13

Definicja reguły CSS

- Składnia definicji

```
selektor1, selektor2, ..., selektorn
{
    właściwość1: wartość1;
    właściwość2: wartość2;
    ...
}
```

- **Właściwość** - parametr znacznika HTML, podlegający specyfikacji CSS (np. wielkość czcionki akapitu)

14

Uwagi do definicji stylu

- Przy deklarowaniu wartości parametru nie należy używać podwójnego cudzysłowu
- CSS jest bardzo wrażliwe na składnię - brak jednego średnika może spowodować zignorowanie całego arkusza

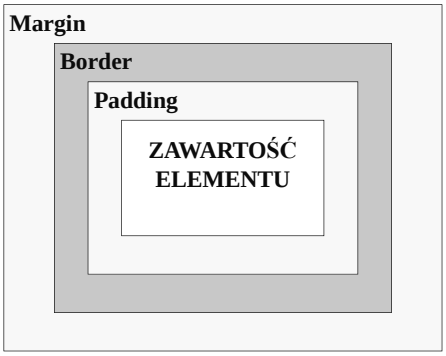
15

Box Model

- Standardowo szerokość i wysokość ustalane w CSS odnoszą się do wielkości **zawartości** danego elementu
- Całkowita wielkość elementu jest większa o **padding, obramowanie** (border) i niewidoczny **margines** (margin)
- W starszych przeglądarkach działanie box modelu było inne (***Quirks Mode*** - nazwa dla trybu zgodności z CSS w Internet Explorer 5 i starszych. Przeglądarki, aby były zgodne ze starymi stronami WWW muszą udawać błędy i niedoróbki ówczesnych przeglądarek)

16

Box model



17

Deskryptory CSS – marginesy

- Właściwości marginesów (obszaru pomiędzy obramowaniem elementu a elementem zewnętrznym): **margin-top, margin-right, margin-bottom, margin-left, margin**

body {margin:2em } /*wszystkie ustawione na 2 znaki*/
body {margin:1em 2em } /* top&bottom=1em, right&left=2em*/
body {margin:1em 2em 3em} /* top=1em, bottom=3em, right&left=2em*/
ostatnia z powyższych reguł jest równoważna:
body { margin-top:1em; margin-right:2em; margin-bottom:3em; margin-left:2em;}
em – oznacza rozmiar używanej czcionki

18

Deskryptory CSS - wypełnienie

- Własności wypełnienia (obszaru pomiędzy treścią elementu a ewentualnym obramowaniem) – właściwości: **padding-top, padding-right, padding-bottom, padding-left, padding**
- poniższa sekwencja ustawia padding w pionie (padding-top i padding-bottom) na 1em oraz padding w poziomie (padding-left i padding-right) na 2em

  h1{ background:white; padding:1em 2em}

19

Deskryptory CSS - obramowanie

- **szerokość krawędzi:** border-top-width, border-right-width, border-bottom-width, border-left-width i border-width; wartości: thin, medium, thick;
  np. **h1{border-width: thin}**
- **kolor krawędzi**: border-top-color, border-right-color, border-bottom-color, border-left-color i border-color;
  np. **h1{border-color: red}**
- **styl krawędzi**: border-top-style, border-right-style, border-bottom-style, border-left-style i border-style, wartości: none, dotted, dashed, solid, double, groove, ridge, inset, outset;
  np. **h1{border-style: solid dotted }**
  p { border: 1px solid black;
  border-left-color:red; }

20

Deskryptory CSS - czcionka

- Właściwość **font-family**; wartości: **serif, sans-serif, cursive, fantasy, monospace** i inne**;**
- Właściwość **font-style**; wartości: **normal, italic, oblique**
- Właściwość **font-variant**; wartości: **normal, small-caps**
- Właściwość **font-weight**; wartości: **normal, bold, bolder, lighter, 100, 200, ..., 900;**
- Właściwość **font-stretch** (rozstrzelenie czcionki); wartości: **ultra-condensed, extra-condensed, condensed, semi-condensed, normal, semi-expanded, expanded, extra-expanded, ultra-expanded;**
- Właściwość **font-size;** wartości: **xx-small, x-small, small, medium, large, x-large, xx-large, larger, smaller, bezwzględny rozmiar czcionki** (np. 1cm, 0.3in, 2mm, 12pt)**, względny rozmiar czcionki** (wyrażony w %)

21

Kolejność własności pisma

- Większość własności CSS przyjmuje wiele słów kluczowych, które można ustawiać w dowolnej kolejności i nie ma wymogu, żeby stosować wszystkie z nich
- Jednym z niewielu wyjątków jest font, który ma określony minimalny zestaw własności i musza one być ustawione w odpowiedniej kolejności
- Najprostsza możliwa deklaracja (rozmiar i rodzina):
  font: <font-size> <font-family>
  (do rozmiaru można doczepić wartość wysokości linii **line-height** po znaku */*)
- np.
  font:100% sans-serif;
  font:100%/2.5 Helvetica,sans-serif;
- Chcąc użyć innych własności muszą one być wpisane przed powyższymi np.:
  font:italic bold small-caps 200% Helvetica, arial,sans-serif;

22

Deskryptory CSS - kolory i tło

- kolor pierwszego planu: **color**;
  span {color: red}
  p {color: rgb(255,0,0) }
- tło: **background-color**, **background-image, background;**
  h1 { background-color:#F00 }
  body { background-image: url("tło.gif") }
  #nav { background: #FFCB2D url(bg.gif)
  repeat-x bottom left;}

23

Deskryptory CSS - właściwości tekstu

- wcięcie: **text-indent**, np. **p{ text-indent: 3em }**
- wyrównanie: **text-align**; wartości: **left, right, center, justify**;
  vertical-align; wartości: middle, bottom, top;
  np. **p{ text-align: center; vertical-align:bottom}**
- dekoracja: **text-decoration**; wartości: **none, underline, overline, line-through, blink**;
  np. **p{ text-decoration: blink }**
- cień: **text-shadow**; np. **p{ text-shadow: 0.2em 0.2em}**
- odstępy: **letter-spacing** i **word-spacing**;
  np. **h1{ letter-spacing: 0.1em; word-spacing: 1em }**
- wielkość liter: **text-transform; wartości: capitalize, uppercase, lowercase, none**

24

Klasy

- Właściwości obiektów można definiować w **klasie**
- Każdemu elementowi można przyporządkować odpowiednią klasę
- Klasy pozwalają definiować style tylko wybranych elementów HTML (a nie całości dokumentu)

25

Klasy

- Tworzenie klasy **(w pliku CSS):** **znacznik.NazwaKlasy {własność:wartość;}**
- Określenie klasy obejmującej wszystkie znaczniki **(w pliku CSS):** ***.NazwaKlasy {własność:wartość;}**
- Element **(w pliku HTML)** mający własności wybranej klasy musi mieć parametr: **class="NazwaKlasy"**

26

Przykład wykorzystania klas

```
<html>
<head>
  <style type="text/css">
    <!--   p.czerwony { color:red; }
           p.niebieski { color:blue; }
           *.zielony {color:green;}
    -->
    </style>
</head>
<body>
    <p  class="czerwony">Akapit czerwony</p>
    <p  class="niebieski">Akapit niebieski</p>
    <p> Akapit zwykły </p>
    <p  class="zielony">Akapit zielony</p>
    <h2 class="zielony">Nagłówek zielony</h2>
</body></html>
```

Akapit czerwony

Akapit niebieski

Akapit zwykły

Akapit zielony

Nagłówek zielony

27

Wyjątki

- Tworzenie wyjątku (w pliku CSS): **#nazwa_wyjątku {własność:wartość}**
- Definicja wyjątku dla wybranego dokumentu - nadanie znacznikowi parametru (w pliku HTML): **id="nazwa_wyjątku"**

28

Przykład dokumentu z wyjątkami


```
<html><head>
  <style type="text/css">
  <!--
     * {margin:0}
     p.czerwony { color:red; }
     p.niebieski { color:blue; }
     #zolty { color:yellow; }
     #blok {background:black}
     #tytul{background:gray;}
  --></style></head>
<body>
<div id="tytul">Przykład wyjątków</div>
<div id="blok">
    <p  class="czerwony">Akapit czerwony</p>
    <p  class="niebieski" id="zolty">Akapit
    klasy niebieski ale wyjątkowo żółty</p>
</div>    </body></html>
```

29

Formaty selektorów reguł CSS

*****	spełniony przez każdy element
E	spełniony przez każdy element <E> (np. body, p)
E1,E2	spełniony przez każdy element <E1> i <E2>
E F	spełniony przez element F leżący wewnątrz elementu E, np. Ten nagłówek jestbardzo ważny)
E * F	spełniony przez element F (może być zagnieżdżony wewnątrz innego elementu), wewnątrz elementu E, np. Ten nagłówek jest bardzo ważny
E > F	spełniony przez element F będący elementem potomnym elementu E, np. div ol>li p

30

Formaty selektorów reguł CSS

a:link	spełniony przez każdy link <a>, który nie został jeszcze odwiedzony
a:visited	spełniony przez każdy link <a>, który został już odwiedzony
E:active	spełniony przez każdy element E, który jest właśnie naciskany myszką
E:hover	spełniony przez każdy element E, nad którym właśnie przesuwa się myszka
E[atr="wart"]	spełniony przez te elementy E, które posiadają atrybut **atr** o wartości '*wart*'
E.klasa	spełniony przez te elementy E, które posiadają atrybut **class** o wartości '*klasa*'
***.klasa**	spełniony przez wszystkie elementy, które posiadają atrybut **class** o wartości **val**
.klasa	**j.w.**

31

Plik CSS - *styl2.css*

```
body          {background-color:#ffffb9}
table         {width:100%;background-color:#ffffff;}
*             {font-family:Sans-serif}
*.wiersz_n    {background-color:#9fff9f}
*.wiersz_p    {background-color:yellow}
*.wiersz_nag  {background-color:#00006f;
                color:white;  font-weight:bold;
                text-align:center}
*.tabela_obr  {background-color:#00006f;}
a:active      {color:green;  font-weight:bold;
                text-decoration:none}
a:hover       {color:red}
```

32

Przykład tabeli

```
<table class='tabela_obr'> <tbody> <!-- tabela zewnętrzna -->
<tr><td>
    <table>                    <!-- tabela wewnętrzna -->
    <tr class='wiersz_nag'> <td colspan='2'> Wyszukaj książkę</td>
    <tr class='wiersz_n'>   <td> Tytuł</td><td><input type='text' name='tyt' /></td>
    <tr class='wiersz_p'>   <td> Autor</td><td><input type='text' name='aut' /></td>
    <tr class='wiersz_n'>
          <td> Wydawca</td><td><input type='text' name='wyd' /></td>
    <tr class='wiersz_p'>
      <td> Gatunek</td><td><select name='gat'>
          <option value='1'> Historia </option>
          <option value='2'> Kryminał  </option>
          <option value='3' selected='selected'> Fantastyka</option></select></td>
    <tr class='wiersz_n'>
      <td colspan='2'><br /><input type='submit' value='Szukaj' /></td>
    </table>                   <!-- koniec tabeli wewnętrznej -->
</td></tr>
</tbody>
</table> <!-- koniec tabeli zewnętrznej -->
```

33

Tabela z CSS i bez CSS

34

Element DIV i SPAN

- **span** - uniwersalny element liniowy; nie ma konkretnego zastosowania - można użyć go w dowolnym celu dodając do niego np. style CSS np.
  Jakiś **** kolorowy **** tekst.
- **div** - uniwersalny element blokowy
  np.

```
<div> Jakiś tekst
      <div class='k2'>  Inny tekst
            <h3>Nagłówek tekstu</h3>
            Akapit tekstu dowolnego
      </div>
Jeszcze jeden tekst
</div>
```

35

Właściwość display

Definiuje sposób wyświetlania/interpretowania elementu

- `display:block` - przeglądarka zawsze wstawia znak konca linii przed i po elemencie (tekst elementu jest wyświetlany w odrębnym „bloku")
- `display:inline` (domyślnie)
- `display:none` - przeglądarka nie wyświetla takiego elementu (stosuje się do elementów, które mają być ukryte)
- właściwość `display` nie jest dziedziczona przez elementy potomne

36

Właściwość float i clear

- `float` i `clear` - kontrolują miejsce wyświetlania elementów `block` względem pozostałego tekstu dokumentu
- `float` - wyświetla zawartość elementu `block` obok tekstu pozostałych elementów - po lewej lub prawej stronie (pływające ramki); wartości: `left`, `right`, `none`
- `clear` z konkretnym elementem - zapobiega wyświetlaniu tego elementu obok elementu w ramce (z ustawioną własnością `float`); `wartości: none (domyślnie)`
  `left` - element zostanie wyświetlony powyżej a nie obok czy wokół poprzedzającego go elementu z ustawieniem `float:left`
  `right` - element zostanie wyświetlony poniżej poprzedzającego go elementu z ustawieniem `float:right`
  `both` - element zostanie wyświetlony poniżej poprzedzającego go elementu z ustawieniem `float:right` lub `float:left`

37

Przykład - wiersz.html

```
<html><head><link rel='stylesheet' href='styl.css' type='text/css'></head>
<body>   <h1> Wół minister</h1>
         <h2> Ignacy Krasicki</h2>
         <h3> 1779</h3>
   <p></p>          <!-- Pierwsza zwrotka -->
   <div> <span class='w1'>Kiedy wół był ministerm i rządził rozsądnie</span>
         <span>Szły, prawda, rzeczy z wolna ale szły porządnie.</span>
         ...
   </div>
   <p></p>          <!-- Druga zwrotka -->
   <div> <span class='w1'>Pan sie śmiał, śmiał minister, płakał lud ubogi.</span>
         <span>Kiedy więc coraz większe nastawały trwogi,</span>
   </div>
</body>
```

38

Przykład - styl.css

```
body   {font-size:12pt}
body, h1, h2, h3, p, div, span {display:block}
h3, div {margin-bottom:6mm}
span   {color:#00006f}
span.w1 {color:red}
H1     {font-weight:bold; background:#aaaaff}
H2     {font-style:italic}
p      {background-image:url(wol.jpeg);
        background-repeat:no-repeat;
        background-position:center;
        width:80px;
        height:55px;
        float:left}
```

39

Wiersz z CSS

40

Przykład – formularz z CSS

```
<form action="..." method="get">
 <fieldset> <!--Pierwsza grupa pól formularza -->
  <legend>Pola tekstowe</legend> <!-- Opis pierwszej grupy pól-->
   <label for="imie">Imie: </label> <!--Etykieta pola tekstowego-->
    <input type="text" id="imie"
          name="imie"/><br />
  <label for="nazwisko">Nazwisko: </label>
    <input type="text" id="nazwisko"
          name="nazwisko"/><br />
  Pole bez label: <input type="text"/><br />
</fieldset> <!-- koniec 1 grupy pól-->
```

41

Przykład – c.d.

```
<fieldset> <!--Druga grupa pól formularza -->
  <legend>Pola wyboru</legend> <!-- Opis drugiej grupy pól-->
  <select name="wybor">
   <optgroup label="Systemy operacyjne"> <!--Opcja 1  -->
    <!--Podopcje dla opcji 1:-->
    <option value="wind">System Windows</option>
    <option value="unix">System Unix</option>
   </optgroup> <!-- Koniec opcji 1 -->
   <optgroup label="Procesor"> <!--Opcja 2  -->
   <!--Podopcje dla opcji 2:-->
    <option value="i">Procesor Intel</option>
    <option value="a">Procesor Amd</option>
   </optgroup> <!--Koniec opcji 2  -->
  </select>
 </fieldset><!-- koniec 2 grupy pól-->
<p>  <input type="submit" value=" wyślij dane "/> </p>                </form>
```

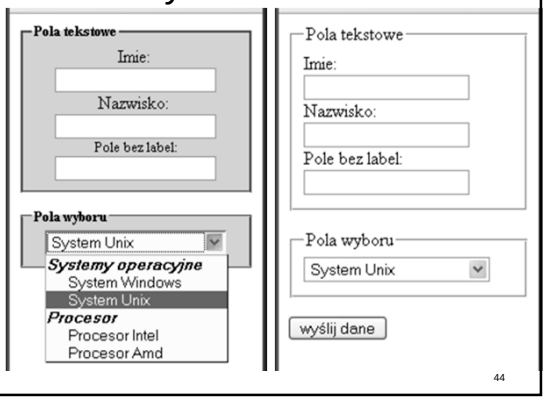
42

CSS dla formularza

```
body     {      font-size:small }
fieldset {      background:#a4ffa4;
                text-align:center;
                border-style:double;
                border-color:#007100 }
legend   {      font-weight:bold; }
label    {      font-size:120% }
select   {      background-color: #ffffb9}
p        {      text-align:center; }
```

43

Wynik z CSS i bez



44

Pozycjonowanie elementów strony

- Domyślnie wszystkie elementy mają pozycję statyczną (position:static), czyli układają się od lewej do prawej, od góry do dołu.
- Żeby uzyskać dwa elementy, jeden pod drugim, wystarczy nadać im display:block i nie potrzeba ich łączyć w żaden dodatkowy sposób. Odstęp między nimi reguluje margin i padding
- Poziomo obok siebie elementy statyczne można układać za pomocą display:inline, display:inline-block, display:table-cell oraz (nie do końca statyczne) float

45

Pozycjonowanie absolutne position:absolute

- Umieszcza obiekt w pozycji wskazanej przez właściwości top, right, bottom, left.
- Pozycja tak naprawdę nie jest absolutna, tylko jest względna do elementu zawierającego (ang. containing block), którym domyślnie jest <body>. Każdy element, który ma **position** inny, niż ***static*** staje się elementem zawierającym dla swoich potomków
- Obiekt pozycjonowany absolutnie jest wyjęty z biegu dokumentu (przestaje zajmować miejsce w swojej oryginalnej pozycji — można to nazwać tworzeniem warstwy)

46

Pozycjonowanie relatywne: position:relative

- Przesuwa obiekt o ofset ustalony we właściwościach top i left. Obiekt pozostaje w tym samym miejscu w biegu dokumentu. Zmienia się tylko miejsce, w którym jest rysowany. Przesunięcie rzadko kiedy jest przydatne i pozostawia się domyślne, zerowe, właściwości.
- praktyczne zastosowanie **position:relative** to łączenie elementów w biegu dokumentu z pozycjonowanymi absolutnie

47

Łącznie position:relative i position:absolute

- Wybranemu obiektowi (kontenerowi) nadaje się ***position:relative***. Jeśli nie poda się top, ani left, to to nie zmienia jego wyglądu w ogóle, natomiast czyni go **elementem zawierającym**. Dzięki temu elementy w nim zawarte, które mają ***position:absolute*** będą pozycjonowane względem jego krawędzi, a nie krawędzi <body> czy innego, nadrzędnego elementu zawierającego.
- Ponieważ elementy absolutnie pozycjonowane są wyjmowane z biegu dokumentu będą zasłaniały inne elementy, o ile nie zarezerwuje się pod nie odpowiednio dużo miejsca. Miejsce można zapewnić np. ustawiając padding na kontenerze.

```
#kontener {position:relative; padding-
  bottom:100px;}
#naDoleKontenera {position:absolute; bottom:0;
  left:0; height:100px;}
```

48

Pozycja nieruchoma position:fixed

- Działa jak **position:absolute**, z tym wyjątkiem, że zamiast pozycjonować względem elementu zawierającego, zawsze pozycjonuje względem okna przeglądarki.
- Daje to efekt przyczepienia elementu na stałe na ekranie. Przewijanie strony nie będzie miało wpływu na pozycję elementu i nie będzie możliwości zobaczenia fragmentu elementu, który nie mieści się na ekranie.

49

Centrowanie elementów w poziomie - tekst

- **Tekst wewnątrz bloku centruje się za pomocą text-align:center**. Właściwość ta działa tylko na elementy typu inline — cały blok może mieć inne położenie
- Elementy mające display:inline, display:inline-block, display:inline-table są traktowane jak tekst, więc też zostaną wycentrowane (m.in.: obrazki, elementy formularza)
  p.komunikat {text-align:center;}
- text-align **jest dziedziczony**, więc jeśli nie chcemy, żeby działał na potomków elementu, należy im nadać inną wartość tej właściwości

50

Statyczne elementy blokowe

- Najczęściej elementy centruje się za pomocą **margin:auto**.
- Aby ta właściwość zadziałała, element musi mieć ustaloną szerokość albo ustawiony display:table (który spowoduje automatyczne wyliczenie szerokości).
- np.:

```
body {
   width:80%;
   min-width:500px;
   max-width:60em;
   margin: 0 auto;
  }
```

- Wyśrodkowanie całej strony, która zajmie 80% szerokości okna, ale nie mniej niż 500px (bo np. umieszczone obrazki się nie zmieszczą), ani nie więcej niż 60em (bo tak bardzo rozciągnięty tekst jest słabo czytelny)

51

Wyśrodkowanie elementu w poziomie - kod HTML

```
<body>
 <div id="srodek">
        To jest test na div w srodku strony
        <div id="d1">
              A tu jeszcze jeden div1 zawarty w div
              srodkowym
        </div>
 </div>
<div id="stopka">&copy;BP</div>
</body>
```

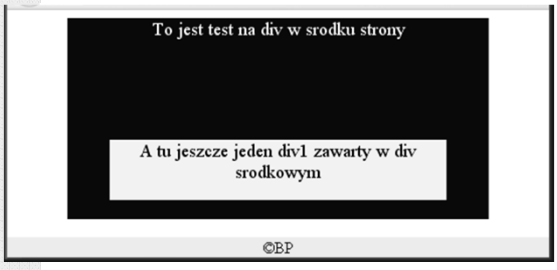
52

Wyśrodkowanie strony w poziomie - CSS

```
body {text-align:center; /*dla IE */ }

div#srodek {width:80%;height:200px;text-align:center;
    background:#000080;color:white;       font-size:large;
    margin:0 auto;
}

div#d1{background:#f8fb88; color:black; height:30%; margin:0
    10%;position:relative; top:50%;}

div#stopka {width:100%; height:1em; text-align:center;
    background:yellow; position:fixed; bottom:0; left:0; }
```

53

Wyśrodkowanie elementu w poziomie - wynik



54

Centrowanie w pionie

- Strony internetowe z założenia są "ciągłe" i mają zajmować **minimalną** niezbędną wysokość
- Również gdy się je drukuje na wielu kartkach, to nie jest jasne, gdzie jest środek (strony? kartki?)
- Z tego powodu centrowanie w pionie nie jest łatwe w CSS

55

Projektowanie stron WWW w oparciu o standardy W3C

Wprowadzenie

- zastosowanie CSS - nie oznacza automatycznego polepszenia projektu
- ważne - stosowanie stylów zgodnych z istniejącymi standardami
- należy:
  - **ograniczać objętość kodu**
  - **zwiększać dostępność stron**
  - **ułatwiać aktualizację i rozbudowę projektu**
- poprawienie elastyczności projektu - oddzielenie treści strony od jej wyglądu
- na poprawny i interesujący projekt internetowy składają się:
  - **komponent wizualny**
  - **elastyczna implementacja** (adoptowalna i łatwo dostępna)

57

Dostępność

- Zgodnie z Web Accessibility po polsku

  "*Dostępność w projektowaniu stron internetowych oznacza tworzenie takich dokumentów, które są czytelne i funkcjonalne dla wszystkich użytkowników niezależnie od ich fizycznych ograniczeń, sytuacji w jakiej się znajdują, a także używanego oprogramowania oraz sprzętu. Podobnie jak w przypadku usuwania architektonicznych barier dostępu jest to działanie na rzecz sytuacji, w której nie ma żadnych "użytkowników drugiej kategorii". Dostępność jest miarą łatwości w uzyskaniu dostępu, odczytaniu i zrozumieniu zawartości stron WWW.*"
- Pierwszy krok w kierunku dostępności - oddzielenie treści od prezentacji

58

Dostępność

- strona jest bardziej przystępna dla różnych urządzeń
- strona jest łatwiejsza w przeprowadzaniu zmian w wyglądzie
- pliki są "lżejsze" - szybsze ładowanie i mniej kodu na stronie, a mniej kodu na stronie - łatwiejsze kodowanie
- istnieje możliwość dostosowywania wyglądu przez użytkowników - przełączniki stylów
- kontrola kodu generuje uporządkowaną formę dla programów udźwiękawiających

59

Obecne standardy projektowania

- użycie zwięzłego i zrozumiałego kodu oraz kaskadowych arkuszy CSS
- można tworzyć interesujące projekty, które jednocześnie są elastyczne i przygotowane na różne ewentualności
- **optymalne** projekty powinny być **dostępne** zarówno dla przeglądarek graficznych jak i tekstowych, przy stosowaniu CSS lub bez, również dla przeglądarek mobilnych
- **elastyczne** projekty - elastycznie dopasowują się do tekstu każdej wielkości.

60

Rozmiar tekstu

- Często stosowany sposób definiowania wielkości tekstu - określenie podstawowej wielkości czcionki w jednostkach pikselowych poprzez zdefiniowanie wartości dla właściwości font-size w deklaracji stylu:

  body {font-size:11px;}
- Zalety określania tekstu w pikselach:
  - wielkość czcionki pozostaje bez zmian niezależnie od tego, w jakiej przeglądarce lub urządzeniu wyświetlany jest tekst
  - stosowanie wartości pikselowych stało się popularne właśnie ze względu na ich konsekwentną, przewidywalną wielkość

61

Elastyczna wielkość czcionki

- rozmiar czcionki w CSS można określić przy użyciu 7 słów kluczowych: xx-small, x-small, small, medium, large, x-large, xx-large np.:

  body {font-size:small;}
- słowo kluczowe definiuje wielkość tekstu według ustawionego w przeglądarce rozmiaru podstawowego. Współczynnik skalowania pomiędzy poszczególnymi słowami kluczowymi pozostaje stały, bez względu na to jaką wielkość podstawową ustalono
- problem - tekst może mieć trochę inną wielkość w zależności od przeglądarki, systemu operacyjnego oraz indywidualnych ustawień

62

Przykład css – skalowalny tekst

body {font-size:small}

h1 {font-size:150%}

.note {font-size:85%}

63

Skalowalna nawigacja

- prosta lista elementów

```
<ul id="nav">
  <li ><a href="\">Wstęp </a></li>
  <li ><a href="prezentacja.html">Oferta </a></li>
  <li ><a href="news.html">Nowości </a></li>
  <li ><a href="kontakt.html">Kontakt </a></li>
</ul>
```

- Wstęp
- Oferta
- Nowości
- Kontakt

64

Skalowalna nawigacja – arkusz CSS

```
#nav {    float:left;
          width:100%;margin:0; padding:10px 0 0 46px;
          list-style:none;
          background:#FFCB2D url(nav_bg.gif) repeat-x bottom left;}
#nav li { float:left;margin:0; padding:0;
          font-family:"Lucida Grande", sans-serif;
          font-size:85%}
#nav a {  float:left;display:block;margin:0 1px 0 0; padding:4px 8px;
          color:#333; text-decoration:none;
          border: 1px solid #9B8748;
          border-bottom:none;
          background:#F9E9A9 url(off_bg.gif) repeat-x top left;}
#nav a:hover {color:#333; background:#fff url(on_bg.gif)}
```

65

Skalowalna nawigacja - uwagi

- *float* dla elementów *li* - pozycje listy zostaną ułożone w linii poziomej
- *float* dla *ul* - rozciągnięcie tła listy i wyświetlenie go za wszystkimi jej pozycjami. Odnośniki w zakładkach będą wyświetlane jako elementy blokowe (*display:block*) . Elementy blokowe wyświetlane są domyślnie w osobnych liniach - korzystając z właściwości *float* ustawiono je w jednej linii
- dla elementu o identyfikatorze ***nav*** ustawienie:

  background:#FFCB2D url(nav_bg.gif) repeat-x bottom left;

  oznacza, że obrazek będzie powtarzany w poziomie i wyrównany do dolnej krawędzi. Obrazek bg.gif ma tylko 3 piksele wysokości dlatego zadeklarowane żółte tło (#FFCB2D) będzie widoczne w pozostałej części paska

66

Skalowalna nawigacja z CSS

on_bg.gif nav_bg.gif off_bg.gif

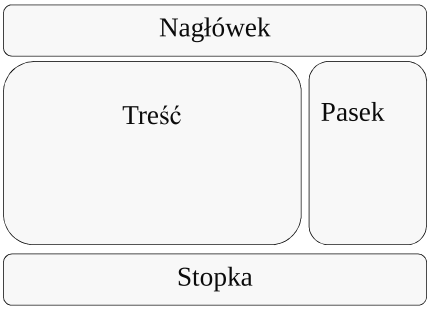
67

Nawigacja tekstowa - zalety

- nawigacja oparta na grafice - użytkownicy nie mogą dostosować wielkości tekstu do swoich potrzeb
- kod np. w postaci listy wypunktowanej zapewnia lepszą dostępność dla szeregu zakresu przeglądarek, urządzeń i programowania dodatkowego
- nawigacja tekstowa – prosta aktualizacja - nie trzeba tworzyć nowych rysunków przy wprowadzaniu każdej zmiany.
- pomysłowe rozmieszczenie obrazków tła może wprowadzić ciekawe efekty, nie pogarszając przy tym jego elastyczności.

68

Dwukolumnowy układ strony



69

Układ kolumnowy z elementami div

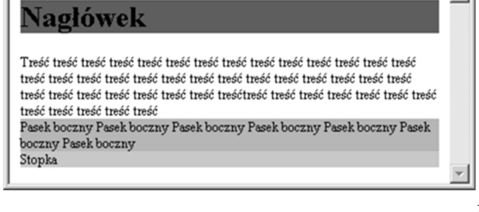
```
<div id="kontener">
    <div id="nag"><h1>Nagłówek</h1> </div>
    <div id="tresc">Treść</div>
    <div id="pasek">Pasek boczny</div>
    <div id="stopka">Stopka</div>
</div>
```

Porządek wyświetlania elementów w przeglądarce tekstowej lub bez CSS:
nagłówek, treść, pasek, stopka.

70

Reguły CSS – układ przed pozycjonowaniem

```
#nag    {background:#369; }
#pasek {background:#9c3; }
#stopka{background:#cc9; }
```



71

Pozycjonowanie elementów – elementy pływające (float)

#nag {background:#369; }
#tresc {**float:left; width:70%;** }
#pasek{**float:right; width:30%;**
background:#9c3; }
#stopka {background:#cc9;
height:10% }

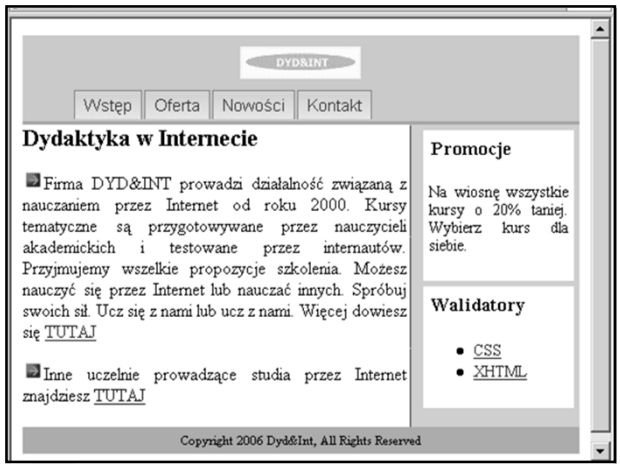
Stopka miesza się z elementami pływającymi, nie ma określonej szerokości i jej pole przejmuje szerokość całego kontenera

72

Stopka - modyfikacja w CSS

- Właściwość **clear:both;**
- stopka nie oblewa poprzedzających ją elementów (z żadnej strony) i jest umieszczona pod obydwoma kolumnami (niezależnie od ich długości)

73



Pasek boczny - elementy *<h3>*, *<p>* i *<ul>*

```
<div id="pasek">
  <div class="box">
    <h3> Promocje</h3>
    <p>Na wiosnę wszystkie kursy o 20% taniej. Wybierz kurs dla siebie.</p>
  </div>
  <div class="box">
    <h3>Walidatory</h3>
    <ul>
      <li><a href="http://validator.w3.org/">CSS</a></li>
      <li><a href="http://jigsaw.w3.org/css-validator">XHTML</a></li>
    </ul>
  </div>
</div>
```

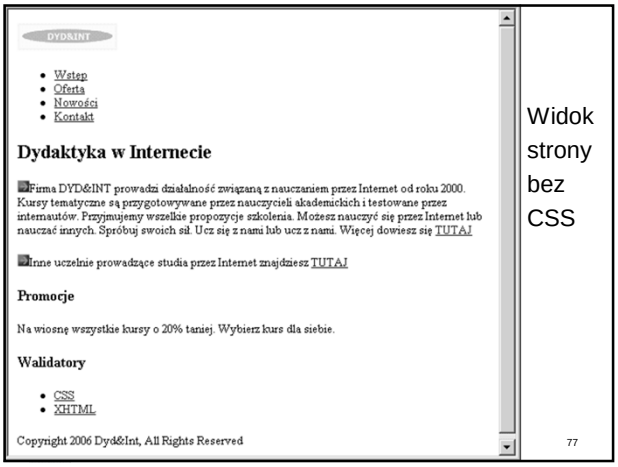
75

Modyfikacje w CSS

- klasa ***box*** i format elementu <***h3***> zagnieżdżonego w elemencie klasy ***box:***

```
.box   {    background:#fff;
            padding:5px;
            margin:5px;
            font-size:90%;}
.box h3 {   margin:0px;
            padding:2px;
            font-size:120%;}
```

76



Widok strony bez CSS

77

Układ trójkolumnowy

```
<div id="kontener">
   <div id="nag"><h1>Nagłówek</h1> </div>
   <div id="tresc">
        <div id="paseklewy">Pasek boczny lewy </div>
        <div id="pasektresci"> treść</div>
   </div>
   <div id="pasek">Pasek boczny </div>
   <div id="stopka">Stopka</div>
</div>
```

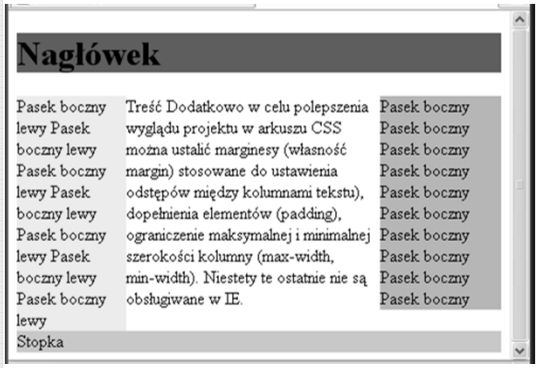
78

Przykładowy CSS

```
#nag   {background:#369;}
#tresc {float:left;width:75%; }
#paseklewy {float:left; width:30%;
        background:yellow;}
#pasektresci {float:right;width:70%;  }
#pasek {float:right; width:25%;
        background:#9c3; }
#stopka {background:#cc9;
        height:10%;clear:both}
```

79

Układ trójkolumnowy



80

Pozycjonowanie stron internetowych w wyszukiwarkach (SEO)

- **Optymalizacja dla wyszukiwarek internetowych** (ang. *Search engine optimization – SEO*; zwana także **pozycjonowaniem**, ang. *Web Positioning*) – działania zmierzające do osiągnięcia przez dany serwis internetowy jak najwyższej pozycji w wynikach organicznych wyszukiwarek internetowych dla wybranych słów i wyrażeń kluczowych
- W polskiej nomenklaturze branżowej - rozróżnienie między **pozycjonowaniem** (działania prowadzone poza docelową witryną np. Link building) a **optymalizacją** stron (działania prowadzone bezpośrednio na stronie internetowej - wpływające na jej budowę, treść i strukturę).

81

Optymalizacja treści

- dopasowanie treści w tagu tytułowym,
- umieszczenie lub rozmieszczenir słów kluczowych w istniejących tekstach lub tworzenie nowych (SEO copywriting),
- dobór adekwatnych nagłówków,
- odpowiednie zaaranżowanie treści menu i innych elementów, wchodzących w skład linkowania wewnętrznego serwisu,
- ustawienie tekstu alternatywnego dla elementów graficznych oraz innych obiektów (np. Flash).

Dawniej działania te ograniczały się do zmiany treści tagów meta Keywords i Description, których obecność na stronie dla większości współczesnych wyszukiwarek ma znaczenie marginalne.

82

Optymalizacja kodu i struktury

- dostosowanie strony do standardów W3C,
- oddzielenia warstwy logicznej struktury dokumentu od warstwy jego prezentacji (np. poprzez zastosowanie CSS),
- poprawa czasu ładowania strony,
- rozwiązanie problemu powielonej treści,
- zastosowanie przyjaznych adresów,
- umożliwienie głębokiego linkowania w przypadku animacji Flash oraz zapewnienie alternatywnej wersji dla przeglądarek bez wsparcia dla wspomnianej technologii (w tym robotów wyszukiwarek internetowych).

83

Nieetyczne metody pozycjonowania

- To metody takie, których nie akceptują twórcy wyszukiwarek lub nie są zgodne z zaleceniami dla tworzących strony gdyż "zaśmiecają" wyniki wyszukiwania.
- Np.:
  - duże nagromadzenie słów kluczowych,
  - słowa kluczowe niezgodne z treścią,
  - ukrywanie słów kluczowych, np. użycie tekstu w kolorze tła, umieszczenie na niewidocznym elemencie lub w miejscu zwykle nie oglądanym przez użytkownika,
  - **cloaking** (ang. *cloak* – płaszcz) – serwowanie robotom wyszukiwarek (rozpoznawanych np. po adresach lub tzw. *User-agent*) specjalnie spreparowanej wersji strony,
- wykrycie stosowania takich technik – zwykle powoduje całkowitą eliminację strony z wyników wyszukiwania.

84

Etyczne metody pozycjonowania

- Wymiana linków i banerów ze stronami o podobnej tematyce, umieszczanie stron w katalogach tematycznych
- Rezygnacja z nawigacji uzależnionej od dodatkowych technologii (JavaScript, Flash, ramki), użycie mapy strony
- Usunięcie błędów składniowych z kodu strony (walidacja)
- Optymalizacja kodu strony - usunięcie nadmiarowych znaczników HTML
- Dobre opisanie elementów strony, np. treści alternatywnej dla grafiki
- Czytelne opisywanie odnośników (unikanie opisów "tutaj","kliknij")
- Stosowanie "przyjaznych" adresów (np. http://example.com/sklep/drukarki/laserowe/ zamiast http://example.com/index.php?page=236&id=32)
- Wyróżnianie ważniejszych słów, stosowanie nagłówków
- Optymalizowanie treści i artykułów na stronie poprzez np. używanie słów kluczowych w mianowniku.
- Metody te będą zawsze akceptowane przez wyszukiwarki, a ich stosowanie nie spowoduje eliminacji strony z wyników wyszukiwania

85