Technologie internetowe

CZĘŚĆ 4

CSS
kaskadowe arkusze stylów

Wykład opracowany na podstawie materiałów z:
http://pl.wikipedia.org/
http://www.kurshtml.edu.pl/
http://www.htmlkurs.yoyo.pl/index.html

Dr inż. Arkadiusz Rzucidło

Czym jest CSS

- CSS (Cascading Style Sheets) Arkusze stylów kaskadowych są językiem zarządzania formatowania elementów dokumentu HTML
- Język ten nie istnieje samodzielnie i jest uzupełnieniem HTML'a
- Mimo iż CSS jest standardem niektóre przeglądarki mogą traktować go odmiennie
- Głównym celem powstania CSS jest rozdzielenie warstwy struktury dokumentu od warstwy prezentacji.
- Dzięki CSS wszystkie polecenia formatowania można umieścić w jednym miejscu i powiązać je z poszczególnymi elementami lub rodzajami elementów charakteryzujących się pewnymi wspólnymi cechami
- Dzięki CSS edycja dokumentu oraz wszelkie globalne modyfikacje są zdecydowanie łatwiejsze niż stosowane tradycyjnych rozwiązań

Dlaczego stosować ?

- Jest to efektywne narzędzie formatowania stron
- Zdeprecjonowane elementy formatowania będą powoli wycofywane z obsługi poszczególnych przeglądarek
- Poprawia się przejrzystość dokumentów HTML
- Pozwalają zarządzać całą serią dokumentów powiązanych w serwis lub cześć serwisu
- Oszczędność podczas pisania dokumentów HTML oraz głównie treści zamieszczanych na kartach serwisu
- Możliwość stosowanie lokalnych i globalnych jednostek oraz klas, kolorów, itp.
- Różne nowe sposoby formatowania – podwójne podkreślenia, przekreślenia treści, typy krawędzi, itp.
- Dodatkowe możliwości formatowania treści dokumentu (interlinia, odległość pomiędzy znakami, itp.)

Używanie stylów

- CSS nie istnieje samodzielnie więc musi zostać odpowiednio dołączony do dokumentów HTML
- Zasada jest prosta:
  - Dokument (X)HTML zawiera strukturę i treść strony wraz z oznaczeniami poszczególnych grup elementów
  - Wszelkie informacje o kształcie i wizualizacji grup i treści w dokumentach zawiera arkusz CSS
- Sposoby umieszczania stylów dla dokumentów HTML są różne
- Każdy ma swoje zalety i przydaje się w różnych sytuacjach
- Są wybierane w zależności od potrzeby
  - Lokalny
  - Blokowy
  - Wewnętrzny
  - Zewnętrzny

Styl lokalny

- Dotyczy ustawień związanych z konkretnym elementem

```
<selektor style="cecha: wartość; cecha2: wartość2...">...</selektor>
```

- Selektorem może być dowolny element
- Formatowanie nadaje się treści pomiędzy znacznikami selektora
- Tego rodzaju style – inline – ponieważ dotyczą jednej linii stosuje się lokalnie, gdzie zależy nam na stałym formatowaniu tego elementu bez względu na globalne ustawienia
- Stosuje się go dość rzadko z uwagi na elastyczność
- Przykład:

```
<p style=" color: red" > To jest przykładowy tekst w kolorze czerwonym</p>
```

Styl lokalny rozciągnięty

- Czasem lokalny styl potrzeba zastosować do większej liczby elementów niż jeden – wtedy stosuje się rozciąganie stylu

```
<span style="cecha: wartość; cecha2: wartość2...">...</span>
```

- Ustalanie parametrów stylu wygląda tak jak lokalnie
- Stosuje się znacznik …
- Przykład:

```
<span style=„color: red”>
 To jest normalna treść <b> a to jest pogrubiony fragment tekstu</b><p>
 Następnie nowy akapit</p></br> oraz nowa linijka pod akapitem
</span>
```

Wydzielone bloki treści

- Obiekty blokowe nadają się do formatowania większych treści oraz szablonów

```
<div style="cecha: wartość; cecha2: wartość2...">...</div>
```

- Elementy div nie posiadają zdeklarowanego formatowania dlatego za ich pomocą można tworzyć swobodnie dowolną strukturę prezentacyjną
- dokonuje zmian w linii natomiast w bloku
- Elementy mogą zwierać inne elementy blokowe oraz liniowe, natomiast elementy liniowe nie mogą zawierać blokowych
- Przykład:

```
<div style=„background-color: yellow”>
<span style=„color: red”>
 To jest normalna treść <b> a to jest pogrubiony fragment tekstu</b><p>
 Następnie nowy akapit</p></br> oraz nowa linijka pod akapitem
</span>
A to jest nornalny tekst poza formatowaniem
</div>
```

Wewnętrzne arkusze stylów

- Stosuje się gdy formatowanie ma dotyczyć jednej strony w całej jej zawartości

```
<head>
<style type="text/css">
/* <![CDATA[ */
selektor { cecha: wartość; cecha2: wartość2... }
selektor2 { cecha: wartość; cecha2: wartość2... }
/* ]]> */
</style>
</head>
```

- Kompozycję stylów wewnętrznych zawsze zawiera się w części nagłówka
- Jednemu selektorowi można określić kilka argumentów, odseparowanych od siebie średnikami. Należy zwrócić uwagę na nawiasy klamrowe, które rozpoczynają i kończą listę argumentów
- Komentarze w tej sekcji można wykonywać stosując zapis:

```
/* to jest linia komentarza ignorowana przez przeglądarkę */
```

Przykład

```
<style type="text/css"> /* <![CDATA[ */ h2 { color: green } /* ]]> */ </style>
```

- Po wpisaniu na stronie ***<h1> Nagłówek </h1>*** dostaniemy treść Nagłówek napisaną niebieską czcionką

Zewnętrzne arkusze stylów

- Najbardziej powszechna a jednocześnie najbardziej efektywna forma stosowania CSS.

```
<head> <link rel="Stylesheet" type="text/css" href="style.css" /> </head>
```

- Pozwala zastosować zdefiniowane style na wielu stronach jednocześnie
- Deklaruje się jedynie powiązany ze stroną (stronami) plik zawierający wszystkie definicje CSS
- Wpis w nagłówek powinny zwierać wszystkie strony serwisu, których formatowanie dotyczy
- W pojedynczym dokumencie XHTML można dołączyć dowolną liczbę dokumentów CSS
- Deklaracje konfliktowe rozwiązywane są w myśl reguły „później odczytane –> ważniejsze”
- Plik CSS jest zwykłym plikiem tekstowym ASCII o rozszerzeniu CSS
- Ciekawostka ! Niektóre przeglądarki pozwalają mimo stylów określonych dla strony zastosować indywidualne formatowania według plików użytkownika <źródło>

Przykład zewnętrznego pliku CSS

```
/* lista selektorów */
body
{
    font-family: "arial";
    background: #555;
    color: navy;
}

H1
{
    color: red;
}
.etykieta
{
    float: left;
    width: 300px;
    background: #000;
    padding: 5px 5px 5px 5px;
    color: #fff;
    text-align: right;
    font-size: 85%;
}
```

Style zewnętrzne alternatywne

- Stosuje się w celu indywidualizacji wyglądu strony stosuje się zestaw alternatywnych plików CSS

```
<head>
<link rel="Stylesheet" type="text/css" href="style.css" title="Nazwa domyślna" />
<link rel="Alternate stylesheet" type="text/css" href="style1.css" title="Nazwa 1" />
<link rel="Alternate stylesheet" type="text/css" href="style2.css" title="Nazwa 2" />
<link rel="Alternate stylesheet" type="text/css" href="style3.css" title="Nazwa 3" />
</head>
```

- Taki zapis pozwala wybrać użytkownikowi i ustawić alternatywny wygląd strony za pomocą przeglądarki
- Wystarczy tylko utworzyć kilka alternatywnych plików CSS

Kaskadowość stylów CSS

- Jest to określenie priorytetowości poszczególnych ustawień dla określenia formatowania każdego z elementów serwisu
- Dzięki kaskadowości unika się kolizji formatowania ponieważ te rozwiązują się samoistnie
- Jeśli kilka sposobów określa typ formatowania dla elementu obowiązują te stojące najbliżej w strukturze:
  - Styl lokalny (inline)
  - Rozciąganie stylu (SPAN)
  - Wydzielone bloki (DIV)
  - Wewnętrzny arkusz stylów
  - Import stylów do wewnętrznego arkusza
  - Zewnętrzny arkusz stylów
  - Import stylów do zewnętrznego arkusza
  - (Atrybuty prezentacyjne HTML - np. align="...", bgcolor="...", color="...", height="...", width="..." i inne)
- Rozwiązanie kaskadowości jest dość wygodne, jednak należy pamiętać i dobrze przemyśleć sposób formatowania i wykorzystywania metod prezentacji
- Porządek kaskady można wymusić za pomocą atrybutu `!important` stojącego zaraz po wartości cechy którą chcemy wymusić

Selektory elementów

- Dotyczą określania formatowania dla dowolnych elementów XHTML takich jak np.: <b></p><body></hr>, itp.
- Atrybuty formatowania w CSS stosuje się według reguł
- Reguła określa z selektora i deklaracji

```
/* Pierwsza reguła: */
p { color: red }
/* Druga reguła: */
p b { color: red; background-color: yellow }
```

- Deklaracja składa się z dowolnej liczby par `cecha: wartość;`
- Każdy element, grupa elementów ma określoną liczbę cech a każda cecha ma określoną wartość lub listę wartości, którą może posiadać
- Dziedziczenie stylów oznacza, że elementy „potomne” leżące niżej w hierarchii jeśli nie określimy inaczej dziedziczą cechy po elementach „przodkach”

Selektory typów

- Najprostszy selektor, który może prezentować dowolny znacznik XHTML
- To element występujący w kodzie źródłowym strony <p></hr> itp.
- Za jego pomocą ustala się formatowanie dla konkretnego znacznika XHTML
- Przykład:

```
h6 { color: red }
```

Selektory uniwersalne i inne

- Globalny selektor pozwalający ustalić parametry dla wszystkich znaczników w ramach strony bądź serwisu
- Oznacza się za jego pomocą wspólne cech jak np. ogólny rodzaj czcionki treści, jej kolor, itp.

```
*
{
Font-type: Arial;
Color: blue;
}
```

- Selektory potomka
- Selektory dziecka
- Selektory braci
- Ogólny selektor braci
- Grupowanie selektorów

Selektory atrybutów

- Selektor atrybutu pozwala nadać formatowanie elementowi któremu został nadany konkretny atrybut

```
selektor[atrybut] { cecha: wartość }
```

- Dla przykładu treść określana przez znacznik może zostać wyróżniona tekstem w kolorze zielonym jeśli np. w znaczniku zostanie użyty atrybut title=„…”
- Akcja wyróżnienia jest realizowana bez względu na treść jaka towarzyszy argumentowi title. Jeśli tylko argument istnieje w znaczniku, formatowanie następuje według określonej reguły
- Przykład:

```
p[title][lang] { color: red }
```

Selektory atrybutów

- … o określonej wartości:

```
p[title="To jest akapit"] { color: red }
```

- … zawierającego określone wyrazy:

```
p[title~="jest"] { color: red }
```

- … zawierającego łącznik:

```
p[title|="to"] { color: red }
```

- … o wartości rozpoczynającej się od

```
p[title^="to"] { color: red }
```

- … o wartości kończącej się na

```
p[title$="akapit"] { color: red }
```

- … zawierającego określony tekst

```
p[title*="jest aka"] { color: red }
```

- łączenie selektorów atrybutów

```
p[class][dir="ltr"][title~="akapit"][lang|="pl"] { color: red }
```

Selektory specjalne

- Klasy selektorów używane są dla zmiany tylko niektórych wybranych treści w zakresie konkretnego elementu.
- Przykład: wspólne formatowanie dotyczy wszystkich akapitów w serwisie ale tylko te określone mianem class=„ramka” są obramowane

```
selektor.klasa { cecha: wartość }
```

- Przypisanie jednemu selektorowi kilku klas – kilka stylów

```
<p class="klasa1 klasa2 klasa3...">...</p>
```

- Podzbiory klas – jakie klasy musi mieć element by formatowanie było egzekwowane

```
selektor.klasa1.klasa3 { cecha: wartość }
```

- Stosowanie klas jest bardzo użyteczne kiedy pewien rodzaj formatowania występuje kilka razy w serwisie (nie opłaca się stosować stylu inline)
- Reguła uniwersalnej klasy

```
.uniwersalna { color: red }
```

Selektory pseudoelementów

- HTML nie pozwala na odnoszenie się bezpośrednie do elementów takich jak jedna linia, pierwszy znak wyrazu itp.
- Potrzeba ich wyróżnienia nie jest efektywna za pomocą znaczników
- CSS posiada możliwość obsługi takich pseudoelementów bez wprowadzania dodatkowych znaczników
- Ta właściwość pozwala także generować specjalną treść np. przed lub po akapicie
- Pierwsza linia – wyróżnienie pierwszej linii:

```
p:first-line { color: red }
```

- Pierwsza litera – wyróżnienie tylko pierwsze litery:

```
p:first-letter { color: red; font-family: 'Times New Roman‘ ; font-size: 300%;}
```

- Przed i po

```
p:before { content: " (...) "; color: red }
p:after { content: " (…) "; color: red }
```

Selektory pseudoelementów

- Cudzysłowy – XHTML posiada specjalne znaczniki dla cudzysłowów <q>…</q> oraz <blockquote> … </blockquote>
- Specyfikacja nie zaleca dodawania cudzysłowów ponieważ powinna to zrobic przegladarka
- Można jednak to zmienić

```
q { quotes: '"' '"' "'" "'" }
q:before { content: open-quote }

q:after { content: close-quote  }
```

Lub

```
q:lang(pl) { quotes: "\201E" "\201D" "\00BB" "\00AB" }
q:before { content: open-quote }

q:after { content: close-quote }
```

Znak	Kod	Nazwa
"	"\0022"	Cudzysłów (podwójny znak cudzysłowu ASCII)
'	"\0027"	Apostrof (pojedynczy znak cudzysłowu ASCII)
‹	"\2039"	Pojedynczy lewy cudzysłów ostrokątny
›	"\203A"	Pojedynczy prawy cudzysłów ostrokątny
«	"\00AB"	Podwójny lewy cudzysłów ostrokątny
»	"\00BB"	Podwójny prawy cudzysłów ostrokątny
‘	"\2018"	Lewy pojedynczy cudzysłów (górne "6")
’	"\2019"	Prawy pojedynczy cudzysłów (górne "9")
“	"\201C"	Lewy podwójny cudzysłów (górne "66")
”	"\201D"	Prawy podwójny cudzysłów (górne "99")
„	"\201E"	Podwójny dolny cudzysłów (dolne "99")
‚	"\201A"	Pojedynczy dolny cudzysłów (dolne "9")

Selektory pseudoklas

- Pseudoklasy są specyficznym typem CSS i nie są one statycznie przypisane elementom HTML'a
- Elementy, których dotyczą mogą zyskiwać formatowania pseudoklas lub je tracić w zależności od wyników interakcji z użytkownikiem (np. podświetlenie elementu menu przy ruchu myszką)
- Pseudoklasy dynamiczne
  - Linki (link, visited)
  - Akcji użytkownika (active, hover, focus)
- Pseudoklasy etykiet (target)
- Pseudoklasy języka lang()

Selektory pseudoklas

- Pseudoklasy interfejsu użytkownika
  - enabled, disabled
  - checked
- Pseudokalsy strukuralne
  - root
  - nth-child(), nth-last-child(), nth-of-type(), nth-last-of-type()
  - first-child
  - last-child
  - only-child
  - first-of-type, last-of-type, only-of-type
  - Empty
- Pseudoklasa negacji not()

Selektory pseudoklas

- Odsyłacz podstawowy – definiuje jak powinien wyglądać odsyłacz

  `a:link { cecha: wartość }`

- Odsyłacz odwiedzony – definiuje format odsyłacza, który już został wybrany przez użytkownika

  `a:visited { cecha: wartość }`

- Wskazanie myszką – zmienia formatowanie w momencie pojawienia się wskaźnika myszki nad elementem

  `a:hover { cecha: wartość }`

- Aktywacja – pozwala zmienić format jeśli użytkownik uaktywni element np. wciśnie przycisk myszki i przytrzyma na elemencie

  `a:active { cecha: wartość }`

Selektory pseudoklas

- Zogniskowanie – zmienia formatowanie elementowi, który został wybrany przez użytkownika – np. wybrane pole formularza zmienia swoje tło na żółte

```
textarea:focus { color: white; background-color: black }
```

- Blokada – zmienia formatowania zablokowanych i odblokowanych pól formularza

```
input:disabled { border: 1px solid blue }
input:enabled { border: 1px solid red }
```

- Tylko do odczytu – zmienia formatowania pól formularza oznaczonych atrybutem tylko do odczytu

```
input:read-only { border: 1px solid blue }
input:read-write { border: 1px solid red }
```

- Zaznaczenie – jeśli zaznaczymy element to zmieniamy jego formatowanie – np. pozycję (odskakuje w bok)

```
input:checked, option:checked { margin-left: 30px }
```

- Inne w samouczku Kurs HTML

Konkluzje

- Lista wszystkich argumentów i wartości selektorów jest dostępna na wielu witrynach o projektowaniu i tworzeniu serwisów
  - http://www.kurshtml.edu.pl/css/wprowadzenie,tlo.html

- Zależą od typu selektora
- Lista jest dość długa, jednak wiele z atrybutów jest typu uniwersalnego
- Lista zastosowanych atrybutów nie musi być długa
- Efektywność i efektowność serwisu nie oznacza zastosowania wszystkich możliwych sposobów formatowania
- Jeden wybrany scenariusz pozwala na łatwiejszy odbiór serwisu i przekazywanej w nim treści
- Przesyt „możliwości” nie jest ubogaceniem serwisu i bardziej wpływa na dekoncentrację niż służy przekazywaniu informacji

Netografia

- Kurs HTML http://www.kurshtml.edu.pl

Pytania sprawdzające

- Czym jest CSS?
- Jakie są główne zalety CSS?
- Jak powinna wyglądać poprawnie skonstruowana witryna (podział strukturalny)?
- Na czym polega umieszczanie CSS w dokumentach HTML jako: lokalny, rozciągnięty, blokowy, wewnętrzny, zewnętrzny, alternatywny?
- Jakie są cechy poszczególnych sposobów zagnieżdżania CSS?
- Na czym polega kaskadowość stylów CSS?
- Czy można przerwać porządek kaskady stylów?
- Co charakteryzuje selektory: elementów, typów, uniwersalne, atrybutów, specjalne, pseudoelementów, pseudoklas?

Koniec

Temat następnego wykładu to:

MySQL