

Programowanie aplikacji internetowych

CSS

Arkusze stylów CSS

Kaskadowe Arkusze Stylów CSS (ang. *Cascading Style Sheets*) to język służący do formatowania dokumentów WWW, utworzonych w językach HTML oraz XHTML.

- CSS stworzono w celu odseparowania formy prezentacji dokumentu (CSS) od jego struktury i zawartości (HTML)
- Arkusz CSS składa się z listy dyrektyw (reguł) określających sposób prezentacji określonej cechy określonego elementu HTML/XHTML (np. kolor czcionki nagłówka poziomu 3, szerokość lewego wewnętrznego marginesu elementu listy, ...)
- Kaskada stylów polega na możliwości definiowania stylu w kilku miejscach i określeniu priorytetów poszczególnych arkuszy (na wypadek gdyby reguły były sprzeczne)
- Począwszy od CSS 2.1 można definiować odrębne style dla różnych mediów (ekran, drukarka, tv, ...)

Arkusze stylów CSS

Zalety CSS:

- Zwiększenie dostępności witryny
Oddzielenie formy od struktury i zróżnicowanie mediów, w tym dla urządzeń przenośnych, osób niewidzących, ...
- Ułatwienie modyfikacji formy i treści WWW,
Zmiana formy sprowadza się do zmiany CSS (nie HTML).
Zmiana treści nie wymaga formatowania HTML (jest CSS).
- Mniej skomplikowane dokumenty HTML
Dokumenty HTML zawierające tylko opis struktury są mniej skomplikowane – łatwiej zachować poprawność i czytelność dokumentów; Dokumenty o prostszej i bardziej przejrzystej strukturze są łatwiejsze w utrzymaniu i modyfikacji.
- Większe możliwości
CSS oferuje większe możliwości formatowania (a zwłaszcza pozycjonowania) niż HTML

Arkusze stylów CSS

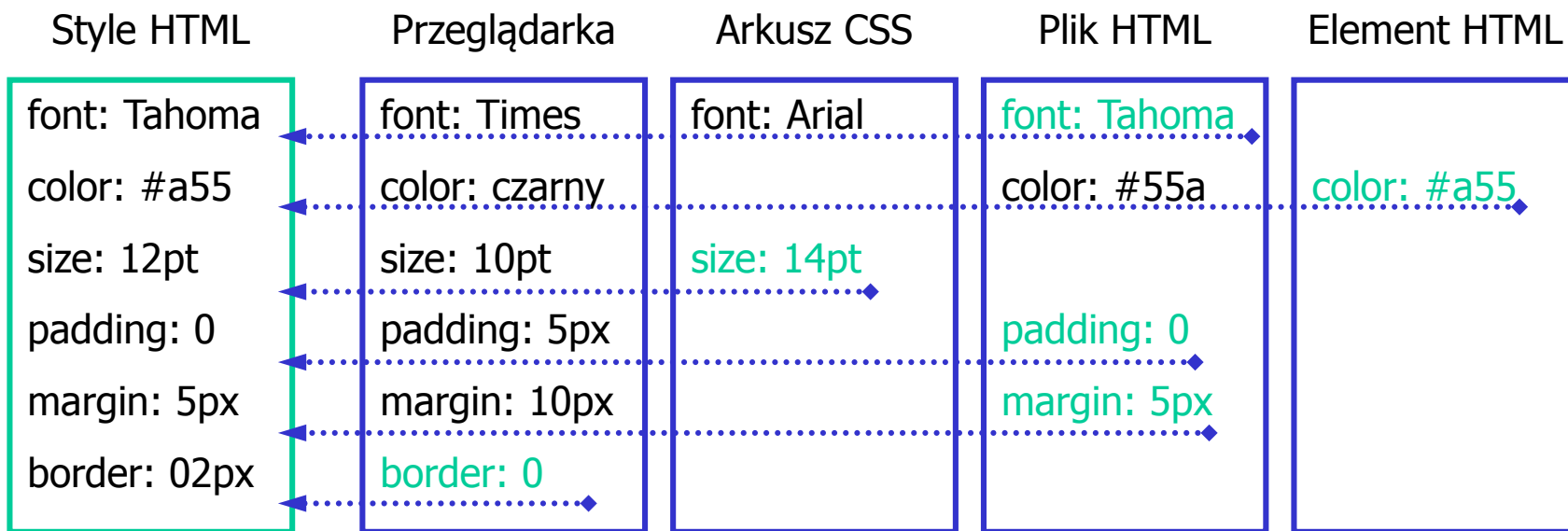
Wady CSS:

- Brak ścisłego związku z HTML/XHTML
Twórcy CSS zakładali, że CSS będzie służył także do innych celów – nie ma ścisłego związku między CSS a HTML (specyfikacja CSS nie mówi nic o HTML i *vice versa*)
- Niejednoznaczność
Oddzielenie CSS od HTML powoduje niejednoznaczności – różne przeglądarki różnie interpretują te same arkusze
- Problemy z implementacją
Żadna przeglądarka nie implementuje w pełni żadnej wersji CSS (obecnie główne przeglądarki mają >90% zgodności z CSS, braki dotyczą elementów mniej ważnych i rzadko stosowanych)

Kaskada stylów CSS

Kaskada stylów

- Styl każdego elementu jest efektem wypadkowym działania kaskady stylów: styl wbudowany przeglądarki może być modyfikowany przez dołączony arkusz CSS, element "style" nagłówka dokumentu HTML i atrybut "style" elementu.



Arkusze stylów CSS

Metody dołączania stylów:

- Element *link* w nagłówku HTML:

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

- Element *style* w nagłówku HTML:

```
<style>
<!--

-->
</style>
```

- Atrybut *style* elementu HTML

```
<??? style=" ... ">
```

Arkusze stylów CSS

Struktura CSS:

- Arkusz CSS składa się z listy deklaracji (statement).

Wyróżnia się dwa rodzaje deklaracji:

- @reguła (at-rule), np.:

```
@media print {  
    ...  
}
```

- zbiory reguł (rule set), np.:

```
h1 {  
    font-family: Arial, sans-serif;  
    font-size: 18pt;  
}
```

Arkusze stylów CSS

Struktura CSS:

- @reguła może mieć jedną z dwu postaci:

```
@reguła zawartość;
```

```
@reguła {  
    zawartość  
}
```

Zawartość zależy od samej reguły.

Najczęściej są stosowane:

- @media
- @import

Arkusze stylów CSS

Struktura CSS:

- Zbiór reguł ma postać:

```
selektor { właściwość: wartość }
```

```
selektor { właściwość: wartość }
```

Można łączyć reguły dla tego samego selektora w bloki:

```
selektor {  
    właściwość: wartość;  
    właściwość: wartość;  
}
```

Można też łączyć selektory w różny sposób:

```
S1, S2, S3 { cecha: wartość }
```

```
S1.clA + S2 > S3.clB { cecha: wartość }
```

Arkusze stylów CSS

Słowa kluczowe:

- Słowa kluczowe (należy pisać bez cudzysłówów!)
 - inherit – wartość właściwości ma być odziedziczona z jego elementu nadrzędnego ("rodzica"):

```
em { color: inherit; }
```

- initial – wartość domyślna

```
em { color: initial; }
```

- predefiniowane wartości, zależnie od cechy, np.:

```
font-style: normal | italic
```

```
font-weight: normal | bold | bolder | lighter
```

```
border-style: solid | dotted | dashed
```

Arkusze stylów CSS

Jednostki długości:

- Jednostki względne

- em - wysokość czcionki,
- ex - wysokość litery x,
- px - piksele (rozmiar px zależy od urządzenia)

```
h1 { margin: 2em; font-size: 1.2em; }
```

```
img.thumb { width: 75px; height: 100px; }
```

- Jednostki bezwzględne:

- cm - centymetry,
- mm - milimetry
- in - cale,
- pt - punkty (1/72 in)
- pc - pica (12 pt = 1/6 in)

Arkusze stylów CSS

Wartości procentowe:

- % - procent

```
div.banner { width: 75% }
```

```
td.lp { width: 10% }
```

Zasoby Internetu

- URI (postaci "protokół://domena.com/folder/plik.typ")
 - url("łańcuch") – ścieżka może być bezwzględna lub względna

```
p { background: url("http://www.d.pl/r.png") }
```

```
p { background: url("www.d.pl/r.png") }
```

```
p { background: url("./r.png") }
```

Arkusze stylów CSS

Kolory:

- Predefiniowane nazwy
(są to identyfikatory, nie należy stosować cudzysłowów)

```
p { color: teal; }
```

- Wartość RGB, RGBA, HSL, HSLA:

- #RRGGBB

- #RGB

- rgb (r, g, b)

- rgb (r%, g%, b%)

```
p { color: #00FFFF; }
```

```
p { color: #0FF; }
```

```
p { color: rgb(0, 255, 255); }
```

```
p { color: rgba(0, 255, 255, 0.75); }
```

```
p { color: rgb(0%, 100%, 100%); }
```

Arkusze stylów CSS

Dziedziczenie właściwości

- Element zawarty w innym elemencie dziedziczy właściwości tego elementu:

```
p { color: blue; }  
<p><em>Lorem</em> ipsum dolor sit amet
```

Lorem ipsum dolor sit amet

- Właściwości danego elementu przesłaniają właściwości odziedziczone:

```
p { color: blue; }  
em { color: teal; }  
<p><em>Lorem</em> ipsum dolor sit amet
```

Lorem ipsum dolor sit amet

Arkusze stylów CSS

Selektory CSS:

- * dowolny element
- E element E
- E, F elementy E oraz F
- E F element F zawarty w elemencie E
- E > F element F zawarty bezpośrednio w E
- E + F element F umieszczony bezpośrednio za E
- E[atr] element E z atrybutem atr (dowolnej wartości)
- E[atr="w"] element E z atrybutem atr o wartości w

Ostatni przypadek ma dwa specjalne warianty w HTML:

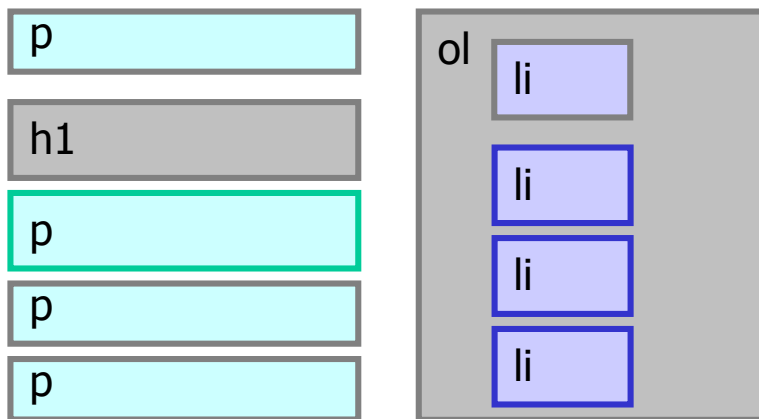
- E.cl element E z atrybutem class="cl"
- E#i element E z atrybutem id="i"

Arkusze stylów CSS

Selektory CSS:

- E + F – element F umieszczony bezpośrednio za elementem E

```
h1 + p { bordercolor: teal; }  
li + li { bordercolor: blue; }
```



Arkusze stylów CSS

Selektory CSS:

- `E[atr]` – element E z atrybutem atr (dowolnej wartości)

`img[alt]`

- `E[atr="w"]` – element E z atrybutem atr o wartości w

`td[colspan=2]`

- `E.cl` – element E z atrybutem `class="cl"`

nazwa klasy może wystąpić wielokrotnie w dokumencie HTML

`p.comment`

`<p class="comment">Lorem ipsum dolor sit amet`

- `E#i` – element E z atrybutem `id="i"`

identyfikator elementu musi być unikalny

`div#banner`

`<div id="banner">`

Arkusze stylów CSS

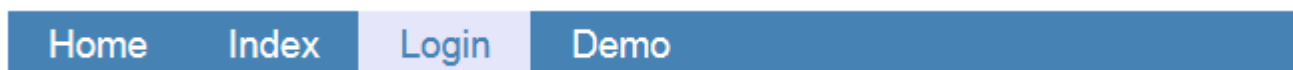
Pseudoklasy elementów aktywnych:

- E:active element (łączy) właśnie aktywowany
- E:hover element (łączy) wskazany, ale nie aktywowany
- E:focus element posiadający fokus

Pseudoklasy odnośników:

- a:link łączy jeszcze nie odwiedzone
- a:visited łączy odwiedzone
- a:hover łączy wskazane, ale nie aktywowane (nie kliknięte)
- a:active łączy właśnie aktywowany (w czasie kliknięcia)

pseudoklasy odnośników są wykorzystywane do przekształcenia listy odnośników w menu:



Arkusze stylów CSS

Pseudoklasy strukturalne:

- `E:first-child` element `e`, który jest pierwszy w innym elemencie (podobnie `:first-of-type`, `:last-child`, `:last-of-type`, `:nth-child`, ...)

```
div > p:first-child { ... }
```

Pseudoelementy:

- `E::first-letter` pierwsza litera
- `E::first-line` pierwsza linia
- `E::before` łańcuch znaków lub licznik przed elementem
- `E::after` łańcuch znaków lub licznik za elementem

```
p.new::before { content: "Nowość! " }
```

```
li:before { content: counter(...) }
```

Arkusze stylów CSS

Łączenie selektorów:

- W praktyce większość autorów wykorzystuje znacznie prostsze podejście, w dwóch krokach:
 1. formatowanie domyślne elementów (E)
 2. formatowanie specyficzne dla klas elementów (E.class) oraz ewentualnie pojedynczych elementów (E#id)

`div`

`div.menu`

`div#banner`

W uzasadnionych wypadkach można zastosować łączenie selektorów, bez przesadnych komplikacji i z użyciem klas:

`div.menu li`

`ol.contens em`

Media

Arkusze CSS mogą definiować różne sposoby formatowania, zależnie od medium prezentacji. Wyróżnia się media:

- all – wszystkie media (wartość domyślna)
- screen – ekran komputera, smartfona, ...
- print – drukarka

Media

Metody przypisania różnych stylów dla mediów:

- Element link z atrybutem media – w nagłówku HTML:

```
<link rel="stylesheet" type="text/css"
      href="main.css">
```

```
<link rel="stylesheet" type="text/css"
      href="print.css" media="print">
```

- Dyrektywa @import i/lub @media w pliku CSS:

```
@import url(./print.css) print;
```

```
@media print {
    /* styl dla drukarki */
}
```

Reguły znajdujące się poza zasięgiem dyrektywy @media domyślnie dotyczą wszystkich mediów.

RWD

RWD (Responsive Web Design) – technika projektowania stron WWW w taki sposób, aby ich układ dostosowywał się do rozmiaru okna przeglądarki

- Początkowo strony WWW były projektowane wyłącznie z myślą o ekranach komputerów → pierwsze urządzenia mobilne skalowały stronę tak, aby zmieściła się na ekranie w całości
- Z czasem zaczęły powstawać odrębne strony dla urządzeń mobilnych (domena.com/ → domena.com/m/);
Wada: trzeba zaprojektować dwa odrębne serwisy WWW
- Obecnie większość stron stosuje podejście RWD, najczęściej wykorzystuje się do tego framework CSS Bootstrap

RWD

Elementy RWD

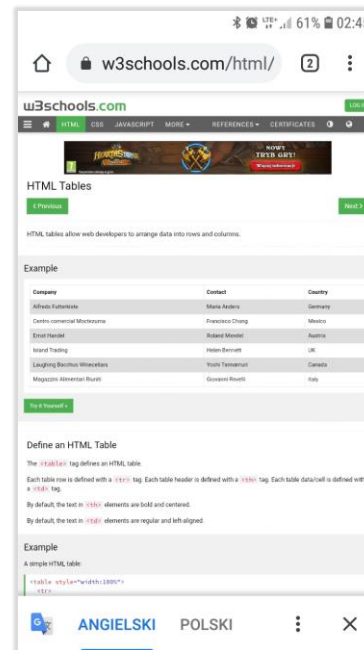
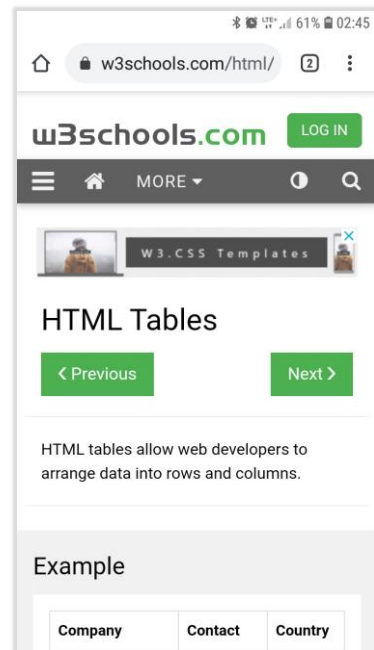
- Viewport
Dyrektywa wyłączająca skalowanie okna
- CSS Media Queries
Rozszerzenie reguł CSS @media o warunki,
np. minimalna/maksymalna szerokość okna, orientacja ekranu
(landscape/portrait)
- Techniki zmiany układu strony
 - elementy pływające
 - systemy grid
 - elementy flex
 - gotowe frameworki CSS, w tym Bootstrap

RWD

Vieport

- Dyrektywa wyłączająca skalowanie okna (HTML5):

```
<meta name="viewport"
      content="width=device-width,
      initial-scale=1.0">
```



RWD

CSS Media Queries

- Rozszerzenie reguł CSS @media o warunki, np. minimalna/maksymalna szerokość okna; Dotyczy tylko mediów: all, screen, print
- Składnia:

```
@media not|only mediatype  
      and (mediafeature and|or|not mediafeature)
```

- Przykłady

```
@media (min-width: 951px) {...}  
@media only screen and (max-width: 950px) {...}  
@media (orientation: landscape) {...}
```

RWD

CSS Media Queries

- ...
- Można również stosować oddzielne arkusze CSS dla różnych mediów i różnych warunków:

```
<link rel="stylesheet" href="style.css">
```

```
<link rel="stylesheet"  
      media="screen and (min-width: 951px)"  
      href="style-wide.css">
```

```
<link rel="stylesheet"  
      media="print"  
      href="style-print.css">
```

RWD

CSS Media Queries

- Zalecane jest podejście "Mobile first" – najpierw reguły jak dla smartfona, potem @media dla innych urządzeń

```
aside {  
    display: block;  
}
```

```
@media only screen and (min-width: 951px) {  
    aside {  
        float: left;  
        width: 200px;  
    }  
}
```

RWD

Techniki zmiany układu strony

- Elementy pływające
- Systemy gridowe
- Flexbox
- Gotowe frameworki CSS, w tym Bootstrap

Bootstrap jest systemem gridowym;

Definiuje 12 kolumn i 5 szerokości okna: xsm (<576 px),
sm (≥576 px) md (≥768 px) lg (≥992 px), xlg (≥1200 px)

Wersja 3 wykorzystuje elementy pływające, wersja 4 – flexbox

- <https://getbootstrap.com/docs/4.0/getting-started/introduction/>

RWD

Techniki zmiany układu strony

- CSS box-sizing

Domyślnie ma wartość content-box – rozmiar pudełka BEZ odstępu (padding) i ramki (border); Dla RWD lepsza jest wartość border-box – rozmiar obejmuje też odstęp i ramkę:

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

RWD

Techniki zmiany układu strony

- CSS display

Domyślny sposób wyświetlania elementów (jako liniowe lub blokowe) można zmienić przez właściwość display;

Do wyboru jest ponad dwadzieścia (!) wartości, w tym inline, block, inline-block, grid, flex oraz none:

```
div.rwd {  
    display: grid;  
    grid-template-columns: 200px auto;  
}
```

RWD

Techniki zmiany układu strony

- CSS display
 - inline-block – pośredni pomiędzy block a inline, element ma swój oddzielny prostokątny blok, ale kilka takich bloków może być wyświetlanych obok siebie, można również określić wysokość bloku i marginesy; często używany do menu:

```
nav li {  
    display: inline-block;  
}  
  
nav a {  
    display: inline-block;  
    padding: 5px 15px;  
}
```

RWD

Techniki zmiany układu strony

- CSS display
 - grid – element grid kontroluje sposób wyświetlania elementów umieszczanych wewnątrz niego, kontroluje ich szerokość i wysokość oraz odstępy między nimi;

```
div.container {  
  display: grid;  
  grid-template-columns: 200px auto;  
  gap: 20px;  
}
```

RWD

Techniki zmiany układu strony

- CSS display
 - grid – elementy umieszczone wewnątrz elementu grid mogą określić liczbę zajmowanych kolumn lub wierszy oraz swoje położenie

```
aside {  
    grid-area: 1 / 2;  
}  
  
main {  
    grid-area: 1 / 1;  
}
```

RWD

Techniki zmiany układu strony

- CSS display
 - none – element nie jest wyświetlany; przydatne do ukrycia elementów np. na małym ekranie lub podczas drukowania

```
@media print {  
    header, footer {  
        display: none  
    }  
}
```

RWD

Techniki zmiany układu strony

- Elementy pływające uzyskuje się przez właściwość float:

```
nav, main {  
    float: left;  
}
```

- Elementy pływające wyświetlane są jeden obok drugiego, zamiast jeden pod drugim, jak dla elementów blokowych; Aby kolejny element był wyświetlony "od nowej linii", powinien mieć w CSS clear: both;

```
footer {  
    clear: both;  
}
```