CSS

Softwarové inžinierstvo

MARTIN TIMOTHY TIMKO

3.9. 2017 – 5.9. 2017

1 ÚVOD

V minulých častiach sme uviedli jazyk HTML, ktorý slúži na definovanie sémantiky obsahu webovej stránky. Týmto sme popísali vývoj webovej stránky z hľadiska jej základnej výstavby. Prispôsobovanie tohto obsahu našim estetickým podmienkam je už plne v kompetencii kaskádových štýlov, CSS (Cascading Style Sheets) v podobe tzv. **prezentačnej časti webu**.

Kaskádové štýly alebo šablóny štýlov pozostávajú z jednoduchých súborov, toré sú nejakým spôsobom odkazované v HTML stránke, alebo ako šablóny štýlov zapísaných priamo do HTML dokumentu a v najmenej odporúčanej možnosti ako tzv. vnorené štýly. Každá HTML stránka môže odkazovať na niekoľko súborov kaskádových štýlov, alebo nemusí odkazovať na žiadny takýto súbor. Každý súbor obsahuje niekoľko pravidiel podľa ktorých definujeme konečný vzhľad webovej stránky, teda konkrétne v súbore kaskádového štýlu definujeme vlastnosti jednotlivých elementov. Jazyk CSS ponúka množstvo vlastností napr. pre formátovanie textu, veľkosť a farba písma, umiestňovanie elementov a podobne. Ak by sme to mali prirovnať z klasického imperatívneho objektového programovania, tak sa jedná o programovanie alebo skôr nastavovanie vlastností (properties) objektov, v našom prípade elementov. V zásade by malo vždy platiť, že sémantická časť webu v podobe HTML stránky by nemala zasahovať do prezentačnej časti v podobe CSS štýlov.

Najlepšie podporovanou verziou jazyka CSS kompatibilnou pre všetky prehliadače je verzia CSS2, pretože táto verzia je konečná verzia, pričom verzia CSS3 je ešte stále vo vývoji, ale v každom prípade verzia CSS2 slúži ako nosná verzia pre túto novú verziu. Aj napriek tomu už väčšina prehliadačov implementovala niekoľko špecifických vlastnosti jazyka CSS3, takže ich je možné používať. Na druhej strane v našom prípade to ani nebude až také potrebné, pretože našim cieľom bude výstavba komplexnej webovej stránky z hľadiska **správneho rozvrhnutia** a **základného formátovania** vzhľadu. Ostatné náležitosti, ako napr. textové efekty, obrázky, formuláre, audio/video a iné multimédia sú dobre vyhľadateľné na internete a z hľadiska podstaty HTML/CSS návrhu zaujímajú len okrajový význam.

Jednotlivé ukážky kódu bodú použité z knihy HTML and CSS od Elizabeth Castro a Bruce Hyslop.

Konkrétne obrázky a výstupy z jednotlivých stránok je možné získať od pôvodných autorov knihy na adrese: http://www.albatrosmedia.cz/tituly/13170172/html5-a-css3/ alebo tiež dostupné aj v anglickej verzii.

1.1 Základné pravidlá štýlu

Každá šablóna, kaskádový štýl sa skladá z nejakého predpisu, ktorý bližšie špecifikuje elementy danej HTML stránky. Týchto pravidiel existuje pomerne veľké množstvo, avšak najčastejšie používané pravidlá tento rozsah výrazne zmenšujú. V každom prípade je možné si dané pravidlá jednoducho vyhľadať na internete, napr.: w3schools/css.

Existujú v zásade tri spôsoby, akým môžme definovať šablóny:

- Externé šablóny štýlu: existuje súbor (textový), na ktorý odkazuje HTML stránka (doporučovaný spôsob)
- Šablóny štýlu zapísané v dokumente: existuje blok textu, ktorý je súčasťou HTML stránky
- Vnorené štýly: existuje jedna šablóna priamo v jendom elemente (najmenej doporučovaný spôsob)

2 EXTERNÉ ŠABLÓNY ŠTÝLOV

2.1 Externé šablóny štýlov

Predtým než začneme vkladať do HTML stránky nejaký CSS štýl je potrebné ho najskôr vytvoriť. Jedná sa o obyčajný textový súbor s príponou `*.css`. Takýto súbor vytvoríme nasledovne:

```
@charset "UTF-8";

/* Jednoduch  ablona styl  */

img {
  border: 4px solid red;
}
```

Deklarácia `@charset "UTF-8"` nie je nutné uvádzať, avšak vždy je vhodné uviesť kódovanie súboru a to platí aj pre HTML stránku. Komentár sa píše podobne ako v jazyku C, ktorý začína a končí lomítkom a hviezdičkou. Tento CSS štýl nám vytvorí červený okraj, plnej čiary s hrúbkou 4 pixely okolo každého elementu `img`. Tento súbor si pomenujeme napr. `"zakladni.css"`.

Vytvoríme si nejakú jednoduchú HTML stránku podobnú z minulých častí, napr.:

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8" />
  <title>El Palau de la M sica</title>
  <link rel="stylesheet" href="zakladni.css" />
</head>
<body>
<article>
  <h1>El Palau de la M sica</h1>

  <img src="obrazky/palau250.jpg" width="250" height="163" alt=
     "El Palau de la M sica" />
  <img src="obrazky/pokladna.jpg" width="87" height="163" alt="
     Pokladna" />

  <p>Miluju koncertn  s l <span lang="es">Palau de la M sica
     </span>. Je n dh rn  zdoben , k iklav  a
     p edstavuje v e , co je na modernismu kr sn . Je to
     tak  z zem  pro sbor <span lang="es">Orfe  Catal </
     span>, s nimi  jsem se nau il , jak  v hody  m
     Moscatell.</p>
</article>
</body>
</html>
```

Teda, v elemente `head` sme špecifikovali odkaz v podobe elementu `link` s atribútom `rela` `href`.

```
<link rel="stylesheet" href="zakladni.css" />
```

Samozrejme, predpokladáme, že HTML stránka a CSS súbor sa nachádzajú v rovnakom adresári, v opačnom prípade je potrebné bližšie špecifikovať relatívnu alebo absolútnu cestu kaskádového štýlu k súboru typu `css`.

V starších verziách HTML bolo nutné písať do elementu `link` atribút `html/css`, avšak v HTML5 to už nie je nutné. Šablóny štýlov môžme obmedzovať len pre určité typy výstupu, ktoré sa nastavujú atribútom `media`. Vždy môžme odkazovať na viac súborov CSS štýlov, ale ak

náhodou definujeme ten istý element v dvoch alebo viacerých súboroch súčasne, tak konečné nastavenie štýlu bude podliehať poslednému súboru definovaného štýlu pre daný element.

2.2 Šablóna štýlov v dokumente

Tento spôsob umožňuje definovať pravidlá štýlu priamo v dokumente HTML stránky. Vzhľadom k tomu, že tieto pravidlá musia byť v rámci HTML stránky definované v elemente `head`, tak dané pravidlá nemôžu byť uplatniteľné pre iné HTML stránky a teda sú viazané len pre HTML stránku, v ktorej sú definované. Teda z hľadiska prenostiteľnosti sa toto javí ako určitá nevýhoda. Avšak, ak si predstavíte predošlú situáciu z viacerými súbormi CSS štýlov, tak je možné definovať napr. jeden CSS štýl, ktorý bude platný pre niekoľko HTML stránok s tým, že každá HTML stránka bude obsahovať interne v elemente `head` niektoré úpravy platné lne pre danú stránku. Teda, tento spôsob ešte má relevantné použitie v praxi.

Za týmto účelom je možné písať priamo do elementu `head` element `style`, ktorým špecifikujeme daný CSS štýl ekvivalentný s CSS štýlom z externého súboru.

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8" />
  <title>El Palau de la M sica</title>
  <style>
  img {
    border: 4px solid red;
  }
  </style>
</head>
<body>
<article>
  <h1>El Palau de la M sica</h1>

  <img src="obrazky/palau250.jpg" width="250" height="163" alt=
     "El Palau de la M sica" />
  <img src="obrazky/pokladna.jpg" width="87" height="163" alt="
     Pokladna" />

  <p>Miluju koncertn  s l <span lang="es">Palau de la M sica
     </span>. Je n dh rn  zdoben , k iklav  a
     p edstavuje v e , co je na modernismu kr sn . Je to
     tak  z zem  pro sbor <span lang="es">Orfe  Catal </
     span>, s nimi  jsem se nau il , jak  v hody  m
     Moscatell.</p>
</article>
</body>
</html>
```

Uvedenie pomocou `style` by nemalo obsahovať deklaráciu `@charset`. Tiež je nutné tento spôsob obmedzovať, pretože písanie štýlov priamo do dokumentu pri častom načítavaní stránky môže zaťažovať server, čo sa môže prejaviť v rýchlosti načítavania.

2.3 Vnorené štýly

Tento spôsob je najhorší z uvedených spôsobov a preto pokiaľ možno sa treba tomuto spôsobu vyhýbať. Je vhodný pri dočasnom skúšaní, alebo v nejakým výnimočných situáciách, kde by prepísanie alebo nájdenie toho ktorého elementu malo za následok prepísanie väčšieho množstva niektorého CSS štýlu. Nevýhoda spočíva v tom, že daný CSS štýl je platný len pre jeden element,

v ktorom daný štýl uvádzame. Tento spôsob výrazne zneprehľadňuje samotný zdrojový kód HTML stránky a zasahujeme do sémantickej (HTML) a prezentačnej vrstvy (CSS).

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8" />
  <title>El Palau de la M sica</title>
</head>
<body>
<article>
  <h1>El Palau de la M sica</h1>

  <img src="obrazky/palau250.jpg" width="250" height="163" alt=
     "El Palau de la M sica" style="border: 4px solid red" />
  <img src="obrazky/pokladna.jpg" width="87" height="163" alt="
     Pokladna" />

  <p>Miluju koncertn  s l <span lang="es">Palau de la M sica
     </span>. Je n dh rn  zdoben , k iklav  a
     p edstavuje v e , co je na modernismu kr sn . Je to
     tak  z zem  pro sbor <span lang="es">Orfe  Catal </
     span>, s nimi  jsem se nau il , jak  v hody  m
     Moscatell.</p>
</article>
</body>
</html>
```

Teda, definovali sme v prvom elemente `img` atribút `style`. V takto definovanom elemente `style` môžme písať akékoľvek CSS štýly navzájom oddelené bodkočiarkou, pričom celý CSS štýl musí začínať a končiť úvodzovkami. Takýmto spôsobom dostaneme orámovanie obrázku len pri jednom z nich.

Ak sa nevieme rozhodnúť pre niektorý typ šablóny, tak ich môžme mať väčšie množstvo. Nie je to rovnaký spôsob ako mať viac šablón, ktoré sa vzájomne ovplyvňujú alebo dopĺňajú. Jedná sa o to, že jednu šablónu môžme používať pre akýsi základný štýl a inú šablónu pre nejakú špeciálny štýl. Takúto špeciálnu, alternatívnu šablónu označíme kľúčovým slovom **alternate**, napr.:

```
<link rel="alternate stylesheet" href="specialny.css" title="
   specialny" />
```

Ono je nutné uvádzať názov odkazu, pretože pod týmto názvom bude vyhľadateľná v menu prehliadača (napr. Firefox) v umiestnení: `View -> Page Style`. Obyčajne obsahuje stránka bez bližšej špecifikácie označenie `No Style` a `Basic Page Style`.

Kaskádový štýl zapísaný takýmto vnoreným štýlom má vždy prednosť pred všetkými ostatnými štýlmi, pričom je možné vykonať výnimku v deklarácii vlasnosti pridaním kľúčového slova **!important**. Avšak použitie tohto kľúčového slova sa neodporúča, pretože má absolútnu platnosť a núti vytvárať dlhšie selektory, aby ste prípadnú platnosť zrušili.

3 SELEKTORY

V podstate najdôležitejšou časťou CSS štýlov je práve selektory. Jedná sa o predpis pomocou ktorého definujeme jednotlivé elementy na HTML stránke. Ako bolo uvedené, CSS štýly je možné zapisovať rôznymi spôsobmi, avšak definovanie selektorov nezávisí od spôsobu zápisu konkrétneho CSS štýlu.

V selektoroch môžme používať 5 rôznych kritérií pre výber elementu:

- Názov elementu
- Kontext, v ktorom sa daný element nachádza
- Hodnota atribútu class alebo id daného elementu
- Pseudotrieda elementu alebo pseudoelementu
- Kritérium daného elementu pre určité atribúty a hodnoty

Selektormi definujeme elementy, pomocou ktorých nejakým spôsobom meníme vzhľad HTML stránky. Ono z tohto hľadiska je dosť dôležité správne napísať HTML stránku z hľadiska sémantiky, pretože na takto definované elementy sa priamo budeme odkazovať z CSS štýlov pomocou selektorov. Jednoduchšie selektory nám umožňujú meniť všetky elementy daného typu, avšak zložitejšími selektormi môžeme meniť len určité elementy podľa definovanej triedy, identifikátoru, kontextu, stavu a podobne.

```
p {
    color: blue;
}
```

Jednoduchý selektor pre element `p` s deklaráciou `color`. Takýmto spôsobom zmeníme **všetky** elementy danej HTML stránky, ktoré odkazujú na element `p`. Je to síce praktické, ale v prípade, ak potrebujeme zmeniť daný element `p` len pre niektoré vybrané elementy, tak si týmto spôsobom nepomôžeme. Za týmto účelom potrebujeme bližšie špecifikovať rozdiely v daných elementoch. Na tento účel slúži **trieda** alebo **identifikátor**.

```
p.hlavny {
    color: blue;
}

p.vedlajsi {
    color: green;
}
```

Týmto spôsobom odlíšime triedou dva rôzne farby pre ten istý element. V HTML kóde by to vyzeralo nasledovne:

```
<p class="hlavny">
modry text
</p>

<p class="vedlajsi"></p>
zeleny text
```

Presne to isté je možné definovať aj identifikátormi:

```
p#hlavny {
    color: blue;
}

p#vedlajsi {
    color: green;
}
```

A následne v HTML stránke to odlíšime takto:

```
<p id="hlavny">
modry text
</p>

<p id="vedlajsi"></p>
zeleny text
```

Obidve spôsoby sú ekvivalentné, ale s tým rozdielom, že triedami sme schopní špecifikovať všeobecnejší popis než identifikátormi. Množstvo profesionálnych web UI dizajnérov odporúča používať triedy namiesto identifikátorov, ale to neznamená, že návrh pomocou identifikátorov je úplne chybný. Samozrejme, označenia triedami a identifikátormi je možné vzájomne kombinovať a z toho dôvodu sa obidve spôsoby môžu vhodne dopĺňať.

Ak by sme chceli predchádzajúci príklad zovšeobecniť, tak by sme to mohli vykovať nasledovne:

Pre triedy:

```
.hlavny {
    color: blue;
}

.vedlajsi {
    color: green;
}
```

Pre identifikátory:

```
#hlavny {
    color: blue;
}

#vedlajsi {
    color: green;
}
```

Toto bude platiť pre **všetky** a zároveň **akékoľvek** elementy, ktoré budú obsahovať označenie danou triedou alebo identifikátorom.

Selektovanie na základe predkov, potomkov a rodičovských elementov je tiež pomerne častý spôsob. Máme napr. nasledujúci kód:

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8" />
  <title>Antoni Gaud       vod</title>
  <link rel="stylesheet" href="css/sousedni-sourozenecky-
     element.css" />
</head>
<body>

<article class="ohledne">
  <h1>Antoni Gaud </h1>

  <p>Neuv  iteln  stavby od Antoni Gaud ho  p  iv  d
     ka d m rokem do Barcelony miliony turist . </p>
  <p>Barcelona <a href="http://www.gaudi2002.bcn.es/english/"
     rel="external">oslavila v roce 2002 150.  v  r o    </a>
     narozen  Gaud ho. </p>

  <section class="projekt">
    <h2 lang="es">La Casa Mil </h2>
    <p>Gaud ho  pr ce byla  u  ite  n  . <span lang="es">La
       Casa Mil </span> je budova, v  n     bydl  <em>
       oby ejn  lid </em>.</p>
  </section>

  <section class="projekt">
    <h2 lang="es">La Sagrada Fam lia </h2>
    <p>Komplikovan  pojmenovan  a nedokon en  mistrovsk
       d lo , kostel Sagrada Fam lia , je v bec <em>
       nejnav t vovan j  </em> stavbou Barcelony. </p>
  </section>
</article>

</body>
</html>
```

Pre tento kód napíšeme selektor v tvare:

```
article.ohledne p {
   color: red;
}
```

Teda, hľadáme elementy `p`, ktoré sú potomkami elementu `article` s triedou `ohledne`. V rámci tohto generického špecifikovania sa vyskytujú aj pomerne zložité konštrukcie, avšak v praxi sa veľmi nepoužívajú, napr.:

```
.ohledne h1:first-child {
   color: red;
}
```

alebo

```
.ohledne p+p {
   color: red;
}
```

Týmto selektorom vyberáme napr. len tie elementy `h1`, ktoré sú prvými dcérskymi elementmi elementu s triedou `ohledne` alebo selektorom susedného súrodeneckého elementu vyberáme len tie elementy `p`, ktoré priamo nasledujú za susedným elementom `p`.

Dokonca je možné selektovať priamo atribúty v nejakom elemente. Napr.:

```
a[rel="external"] {
    color: red;
}
```

Týmto selektorom hľadáme ľubovoľný element `a`, ktorého hodnota atribútu `rel` obsahuje hodnotu `external`.

Takže takýmto spôsobom sme schopní efektívne selektovať elementy a atribúty.