Základy CSS

Co se v modulu dozvíte?

- jak se jednotlivým HTML prvkům určí vzhled
- jaké způsoby slouží k propojení HTML se styly
- jak zapisovat definice stylů
- jak definice CSS využívá stromovou strukturu dokumentu
- co rozhoduje, která z konfliktních definic se použije

Oddělení HTML a CSS

Pomocí HTML určujeme strukturu stránky, tj. stanovujeme postavení jednotlivých prvků v dokumentu (např. hlavní nadpis, odstavec, buňka tabulky). Jak jsme si ukázali v prvním modulu, je výhodné tuto **strukturu dokumentu důsledně oddělovat od vzhledu**, protože pak snadno můžeme změnit vzhled celého webu, aniž bychom měnili jednotlivé soubory HTML. Vzhled můžeme chápat jako další vrstvu, kterou dáváme na vrstvu HTML.

Pro určení vzhledu jednotlivých prvků webové stránky používáme CSS (Cascading Style Sheets, kaskádové styly), což je specifický jazyk, pomocí kterého lze nastavit umístění prvku na stránce, jeho vzdálenost od ostatních prvků, barvy, pozadí a další vlastnosti.

Příklad: V HTML dokumentu máme dva prvky, nadpisy a odstavce.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Moje první stránka</title>
</head>
<body>
    <h1>První nadpis</h1>
    <p>Text odstavce</p>
    <h1>Další nadpis</h1>
    <p>Text druhého odstavce</p>
</body>
</html>
```

Pomocí kaskádových stylů stanovíme vlastnosti nadpisu. Tyto vlastnosti určujeme jednou a projeví se na všech nadpisech první úrovně (`h1`):

```
h1 {
    color: red;
    background-color: black;
}
```

Propojení stylů s HTML dokumentem

Zbývá určit, že právě tento styl nadpisů se použije v konkrétním HTML dokumentu. Pro toto propojení existují tři způsoby:

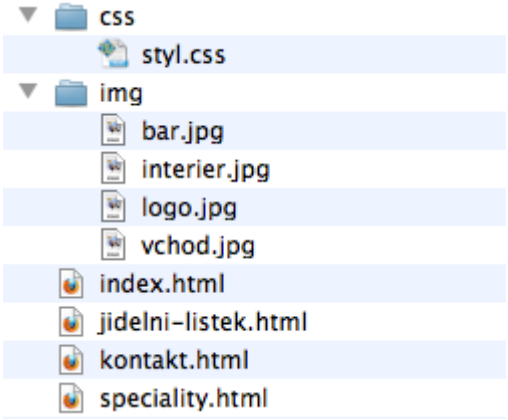
1. elementem `link` v hlavičce dokumentu, který odkazuje na samostatný soubor CSS
2. uvedením stylů jednotlivých prvků v HTML dokumentu v hlavičce (`head`) mezi tagy `<style>` a `</style>`
3. uvedením stylu přímo u konkrétního elementu v atributu `style`

První způsob: `link` v hlavičce dokumentu - externí soubor CSS

V předchozím modulu jsme si ukázali propojení použitím elementu `link` v hlavičce HTML dokumentu, tj. mezi tagy `<head>` a `</head>`:

```
...
<head>
    <meta charset="UTF-8">
    <title>Moje první stránka</title>
    <link rel="stylesheet" href="styl.css" type="text/css" />
</head>
...
```

Tento způsob je nejčastější a nejvýhodnější. Většinou totiž budete chtít, aby se stejný prvek v rámci celého webu zobrazoval stejně, tj. aby např. nadpisy byly vždy stejnou barvou, používaly stejný řez písma, stejnou velikost atd. Vytvoříte tedy jeden (nebo více) CSS souborů, na které se pak odkážete ze všech HTML dokumentů, které na webu máte.



Na příkladu výpisu kompletního adresáře jednoduchého webu vidíte, že soubor se styly se jmenuje `styl.css` (název může být libovolný, přípona `.css`) a je umístěn v adresáři `css`. Ve **všech HTML dokumentech** (`index.html`, `jidelni-listek.html`, `kontakt.html`, `speciality.html`) tedy bude v hlavičce stejný odkaz na soubor s definicí stylů:

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

Je zřejmé, jak je tento přístup výhodný: budete-li chtít změnit formátování jakéhokoli prvku HTML, nezáleží na tom, kolikrát se tento prvek vyskytuje ve všech dokumentech. Změníte pouze jeden prvek v souboru CSS.

Druhý způsob: definice stylu v hlavičce HTML dokumentu

Druhý způsob je vázán na konkrétní HTML dokument, kde je třeba do hlavičky přidat element `style`:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Moje první stránka</title>
    <style type="text/css">
    h1 {
        color: red;
        background-color: black;
    }
    </style>
</head>
<body>
    <h1>První nadpis</h1>
    <p>Text odstavce</p>
    <h1>Další nadpis</h1>
    <p>Text druhého odstavce</p>
</body>
</html>
```

Tento způsob je výhodný, pokud

- chcete nějaký vzhled otestovat v rámci jedné stránky (nemusíte mít otevřeno více souborů)
- potřebujete, aby uvedená vlastnost byla "důležitější" než standardně definovaná
- na stránce se vyskytuje prvek, který jinde na webu není, a potřebujete ho naformátovat

Třetí způsob: v atributu `style` u konkrétního HTML tagu

Třetí způsob je nejspecifičtější, protože vlastnosti (styly) určujete pro jeden konkrétní prvek. Tento způsob je lépe nepoužívat, protože se jeho užitím připravujete o zásadní výhodu, kterou lze získat oddělením struktury dokumentu (HTML) od jeho vzhledu (CSS), protože vzhled je definován u konkrétního prvku.

Příklad:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Moje první stránka</title>
</head>
<body>
    <h1 style="color: red; background-color: black;">První nadpis</h1>
    <p>Text odstavce</p>
    <h1>Další nadpis</h1>
    <p>Text druhého odstavce</p>
</body>
</html>
```

První nadpis tak bude formátován podle stylů (červená barva písma, černá barva pozadí), druhý nadpis už formátován pomocí CSS nebude, převezme tedy vzhled, který dostane od prohlížeče (tzv. defaultní styl).

Pokud to nebudou vyžadovat zvláštní podmínky, budeme vždy používat první způsob, tj. oddělení stylů do samostatného souboru.

Způsob zápisu CSS

Obecně lze zápis CSS zobrazit takto:

```
selektor {
    vlastnost: hodnota;
    vlastnost: hodnota;
}
selektor {
    vlastnost: hodnota;
    vlastnost: hodnota;
}
```

Příklad s použitím elementu

```
h3 {
    margin:0 0 1em 0;
    padding:0;
    font-size:115%;
    line-height:1.2em;
    color:#2f2f2f;
}
p {
    font-family: Georgia,"Times New Roman",Times,serif;
    color: #333;
    font-weight: normal;
}
```

Nastavení vlastností pro více elementů najednou

V následujícím příkladu nastavíme pro všechny nadpisy několik totožných vlastností a pak rozlišíme jednotlivé nadpisy velikostí písma:

```
h1, h2, h3 {
    font-family: Georgia,"Times New Roman",Times,serif;
    color: #333;
    font-weight: normal;
}
h1 {
    font-size: 220%;
}
h2 {
    font-size: 180%;
}
h3 {
    font-size: 150%;
}
```

Univerzální selektor

Pro nastavení jedné vlastnosti naprosto všem prvkům na stránce lze využít tzv. univerzální selektor - hvězdičku (*). Ten v podstatě nahrazuje zápis, kde bychom vyjmenovali všechny prvky. Místo zápisu

```
h1, h2, h3, h4, h5, h6, p, li .....a mnoho dalších..... {
    font-weight: normal;
}
```

použijeme univerzální selektor:

```
* {
    font-weight: normal;
}
```

Třída

Vlastnosti třídy, tedy podrobnější rozlišení konkrétního elementu, definujeme jako `.nazevtridy` nebo `element.nazevtridy`. Např. můžeme odlišit chybové hlášení v odstavci - standardní odstavec se zobrazuje nějakým způsobem, ale pokud k tagu `p` dopíšeme atribut `class` s hodnotou `error` (název si můžeme určit libovolný při dodržení jednoduchých pravidel pro pojmenování tříd), bude formátován jinak. Na příkladu je použitá jiná barva textu, barva pozadí, obrázek, který se u odstavce zobrazí, a mezery kolem prvku.

V HTML dokumentu tedy bude uvedeno:

```
...
<p class="error">Text chybové hlášky</p>
...
```

A formátování chybové zprávy pomocí CSS:

```
p.error {
    color:#ff1800;
    padding:1px 0;
    background-image:url(../img/error.gif);
    background-repeat:no-repeat;
    background-position:0 0.2em;
    padding-left:20px;
}
```

Více tříd

Pro jeden element je možné definovat více tříd. V HTML je zapíšeme s mezerou. Pozor: v uvedeném příkladu jsou použité nevhodné názvy tříd. Název třídy by neměl zohledňovat zobrazení, ale význam, který prvek nese (např. menu, chyba, banner atd.).

```
...
<p class="modrytext zlutepozadi">Text</p>
...
```

Formátovat můžeme každý prvek zvlášť:

```
p.modrytext {
    color:blue;
}
p.zlutepozadi {
    background-color:yellow;
}
```

nebo oba najednou:

```
p.modrytext.zlutepozadi {
    color:blue;
    background-color:yellow;
}
```

Identifikátor

Stejným způsobem můžeme formátovat element s identifikátorem.

V HTML vložíme prvek, který bude sloužit jako menu stránky. Bude to uzavřeno do elementu `div` s identifikátorem `menu`. To bude obsahovat nečíslovaný seznam s odkazy.

```
...
<ul id="menu">
    <li><a href="index.html">Hlavní stránka</a></li>
    <li><a href="jidelni-listek.html">Jídelní lístek</a></li>
    <li><a href="kontakt.html">Kontakt</a></li>
</ul>
...
```

A formátování chybové zprávy pomocí CSS:

```
#menu {
    float: left;
}
```

Totéž by bylo možné zapsat i s určením elementu, ale při použití identifikátoru je to jedno (v jedné HTML stránce může být jeden identifikátor použitý pouze jednou)

```
ul#menu {
    float: left;
}
```

Názvy tříd a identifikátorů

Názvy tříd i identifikátorů **musejí začínat písmenem**. Můžete v nich používat i další znaky, podtržítko (_) nebo pomlčku (-). Názvy tříd zohledňují velikost písmen, `menu` a `MENU` budou dvě odlišné třídy. Doporučení zní používat pro názvy tříd a identifikátorů malá písmena.

- příklady **správných názvů tříd a identifikátorů**: menu, varovani, zahlavi3, a2345678, banner2, prvni-seznam, prvni_seznam
- příklady **chybných názvů**: 1menu, _varovani, -seznam3

Pseudotřídy

Názvy tříd si můžeme definovat podle vlastního uvážení. Kromě nich ale ještě existují tzv. **pseudotřídy**, které jsou již předdefinované. Od názvu elementu se oddělují dvojtečkou. Nejčastěji se používají pro formátování různých stavů odkazu (nenavštívený - link, navštívený - visited, při přejetí myší - hover, při kliknutí, tj. při stisknutém levém tlačítku myši - active).

Příklad: Nastavíme různé barvy pro různé stavy odkazů

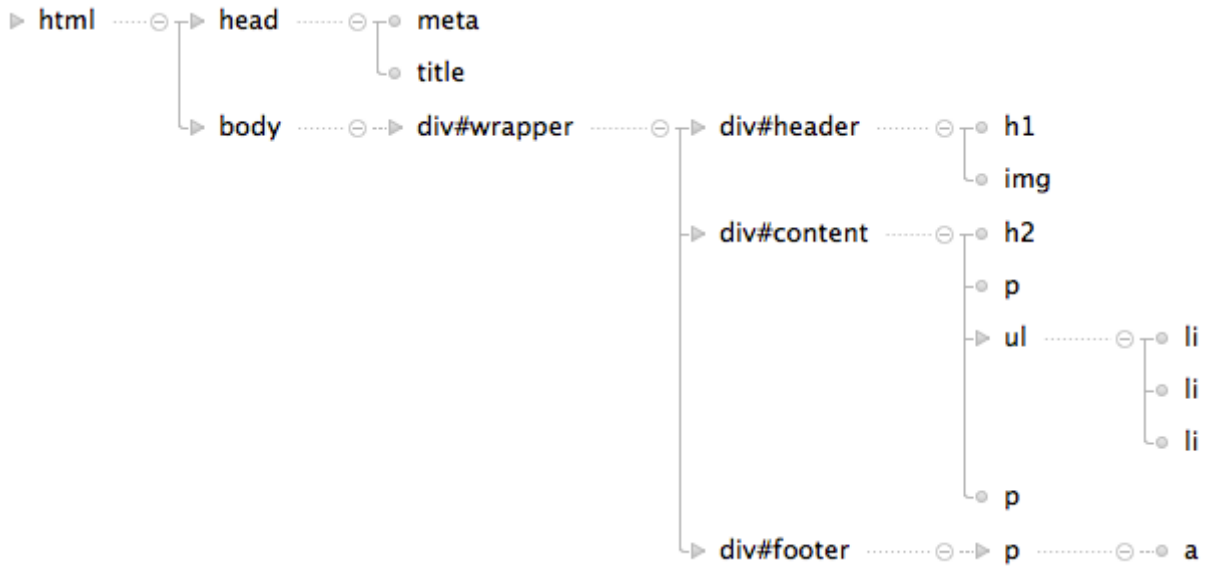
```
a:link {
    color: #A60000;
```

```
}
a:visited {
    color: #881A1A;
}
a:hover {
    color: #580909;
}
a:active {
    color: #074707;
}
```

Pro pseudotřídy u odkazů je nutné dodržet toto pořadí: link, visited, hover, active.

Stromová struktura dokumentu a kaskáda

Z předchozího modulu víme, že dokument je možné překreslit do stromové struktury.



Kaskádové znamená, že styly využívají tuto stromovou strukturu a když nadefinujeme nějakou vlastnost pro určitý prvek, např. písmo pro element `body`, tuto vlastnost zdědí i všechny prvky pod prvkem `body`. Není tedy nutné pro každý prvek definovat všechny vlastnosti. Je naopak vhodné postupovat od větších bloků k jednotlivým prvkům. Na vnořené úrovni je možné pak vlastnost přepsat jinou hodnotou.

Příklad: nastavíme téměř černou barvu písma pro celý dokument (element `body`), pak přebarvíme jinak prvky, které jsou v hlavičce a patičce (budou šedivé).

```
body {
    color: #333;
}
#header, #footer {
    color: #999;
}
```

Kaskády můžeme s výhodou využít, když chceme definovat určitý prvek pouze v rámci některého celku, například odstavec (`p`) budeme chtít zobrazit menším písmem, bude-li součástí patičky dokumentu (`div#footer`). V takovém případě uvedeme v CSS nadřazený prvek před definovaným prvkem a oddělíme je mezerou:

```
#footer p {
    font-size: 80%;
}
```

Toto zařazení do dokumentu jde i řetězit, tj. můžeme např. zacílit položku seznamu (`li`), která je v nečíslovaném seznamu (`ul`) zařazeném v hlavní části dokumentu (`div#content`).

```
#content ul li {
    margin-left: 20px;
}
```

Čím je ale řetězec delší, tím víc práce mají s interpretací prohlížeče, snažte se tedy určení prvku uvádět co nejjednodušší. Proto **nebudeme používat** zbytečně složitý zápis, který vidíte v následujícím příkladu:

```
#wrapper #content ul li {}
```

Nejbližší vnořený prvek zacílíme pomocí `>`. Například chceme-li zacílit na první úroveň seznamu, použijeme tento zápis:

```
#ul.menu > li {
    margin-left: 20px;
}
```

Podobně jako vnořené prvky můžeme formátovat i prvky sousední. Například chceme-li formátovat první odstavec (`p`) za nadpisem druhé úrovně (`h2`) jinak než ostatní odstavce, můžeme to nastavit v CSS takto:

```
h2 + p {
    font-size: 120%;
}
```

Jak funguje kaskáda

Pro fungování CSS platí několik základních pravidel. Jejich neznalost často působí problémy při návrhu CSS, protože se např. prvek chová jinak než jsme zamýšleli (a přitom třeba jen dědí nějakou vlastnost od nadřazeného prvku) nebo se nezobrazuje podle vlastností, které jsme mu definovali (a přitom můžeme mít jinde v CSS nastavenou definici vlastnosti, která je "silnější").

Dědičnost

Vnořený element přejímá většinu vlastností nadřazeného elementu. Když nastavíme pro `body` barvu písma, všechny prvky v `body` budou mít tutéž barvu písma. Pozor, že ne všechny vlastnosti se takto dědí: nedědí se vlastnosti spojené s box modelem, tj. ohraničení a okraje.

K dědičnosti je dobré znát pojmy, kterými vztahy mezi prvky popisujeme. Tyto pojmy vycházejí z popisu rodiny.

- **Předek (ancestor)** je jakýkoli nadřazený prvek. Element `html` bude předkem úplně všech dalších prvků, protože každá HTML stránka je uzavřena vždy mezi `<html>` a `</html>`.
- **Potomek** (descendent) je opakem předka - jakýkoli vnořený prvek. V zápisu výše je oddělen od svého předka mezerou, např. `#footer p` znamená jakýkoli odstavec, který má jako předka prvek s identifikátorem `footer` (v našem případě je to `<div id="footer">`)
- **Rodič (parent)** je nejbližší předek prvku. Např. u dokumentu se strukturou na výše uvedeném obrázku má prvek `h1` rodiče `div#header`. Mezi předky pak patří i `div#wrapper`, `body` a `html`.
- Dítě (child) je opakem rodiče - nejbližší vnořený prvek. V našem příkladu je `li` je dítětem `ul`. Dítě zacílíme pomocí znaku "je větší než" (`>`).
- **Sourozenec (sibling)** je prvek na stejném místě v hierarchii dokumentu, tj. třeba prvky `li` jsou si navzájem sourozenci.

Konfliktní definice vlastností

Konflikty jsou poměrně časté, protože když využíváme pro zacílení prvku na stránce vztahy mezi prkvy, snadno vícekrát "trefíme" stejný prvek.

Příklad:

```
<div id="novinky">
    <div class="zprava">
        <h2>Nadpis zprávy</h2>
        <p>Text zprávy</p>
        <a href="zprava99.html">více</a>
    </div>
</div>
```

Zde můžeme využívat např. tyto zápisy:

```
#novinky p {}
p {}
.zprava p {}
```

Všechny tři příklady směrují k jedinému odstavci. Pokud nastavíme na více místech v CSS témuž prvku různé hodnoty jedné vlastnosti, např. barvu textu, existuje poměrně jednoduchý mechanismus, jak zjistit, která z deklarací vyhraje. Existují dvě pravidla:

- specifičnost a
- pořadí

Specifičnost znamená, že čím specifičtěji určíme, o jaký prvek jde, tím je tato definice "silnější". Pořadí je:

1. styl přiřazený přímo k prvku v atributu `style` (př. `<p style="color:green"`)
2. identifikátor
3. třída
4. element

Takže když seřadíme podle důležitosti uvedený příklad (první = nejdůležitější), dostaneme:

```
#novinky p {}
.zprava p {}
p {}
```

Nerozhodne-li specifičnost, nastupuje pořadí. To znamená, že pokud je prvek nadefinován v CSS dvakrát, rozhoduje pozdější definice. V následujícím příkladu bude text odstavce červený.

```
#novinky p {
    color: green;
}
#novinky p {
    color: red;
}
```

Tyto materiály vznikly v rámci projektu CZ.2.17/3.1.00/34129
Rozvoj oboru Multimédia v ekonomické praxi pro lepší uplatnění absolventů v praxi

Evropský sociální fond - Praha & EU: Investujeme do vaší budoucnosti