Maturitní témata 2020/21

Otázka č. 18 – Počítačové sítě a programování

CSS:

- Cascading Style Sheets (Kaskádové styly), jazyk pro popis způsobu zobrazení elementů na stránkách napsaných v jazycích HTML, XHTML nebo XML

- hlavním smyslem je umožnit návrhářům oddělit vzhled dokumentu od jeho struktury a obsahu, původně to měl umožnit už jazyk HTML ale v důsledku nedostatečných standardů se vyvinul jinak

- definice kaskádových stylů je z několika pravidel, každé pravidlo obsahuje selektor a blok deklarací, každý blok deklarací pak obsahuje deklarace oddělené středníky ; a každá deklarace obsahuje identifikátor vlastnosti, následuje dvojtečka : a hodnota vlastnosti

```
body {
  background-color: white;
  color: black;
  padding: 10px !important;
}
```

- výhody: rozsáhlejší možnosti formátování, jednodušší údržba webové prezentace, oddělení struktury a stylu, v kombinaci s Javascriptem se může různě vylepšovat vzhled stránek

- nevýhody: hlavní nevýhodou je ne vždy dostatečná podpora v prohlížečích → prohlížeče obsahují v implementaci CSS chyby a bývá obtížné výsledek zobrazit stejně

Selektory:

- CSS definuje mnoho různých selektorů, které obvykle můžeme kombinovat, mezi základní patří:

- body - tyto deklarace budou platit pro všechny výskyty elementu body
- body p - tyto deklarace budou platit pro všechny elementy p, které se nachází v elementu body, v jakékoliv hloubce
- body>div - tyto deklarace budou platit pro všechny elementy div, které jsou potomky elementu body
- .trida - tyto deklarace budou platit pro všechny elementy které mají v HTML nastavenou třídu trida, to se provádí pomocí HTML atributu class
- #id - tyto deklarace budou platit pro všechny elementy které mají nastavený identifikátor id
- :link, :visited, :hover a :active - tyto deklarace budou po řadě platit pro dosud nenavštívené odkazy, navštívené odkazy, část textu, na kterém stojí ukazatel myši a aktivní odkaz
- sel1, sel2, sel3 - selektory můžeme seskupovat pomocí čárek, následující deklarace budou platit pro všechny selektory

Pseudotřídy:

- zápis pseudotřídy začíná dvojtečkou, pseudotřídy se od tříd liší tím že určují nějaký stav prvku nebo jeho část, třídy naproti tomu jsou určeny tím jestli má prvek v zápisu atribut class s konkrétní hodnotou

- obecné pseudotřídy se dají použít u všech tagů, dělí se podle:

Chování uživatele:

:hover	prvek je přejížděný myší
:focus	na prvku je fokus, což většinou znamená, že je v něm kurzor
:focus-within	vztahuje se k prvkům, které obsahují prvek, na nichž je focus. Smysl to má u form:focus-within
:selection	část prvku vybraná uživatelem (zamodření myší)

Struktury HTML kódu:

:empty	prvek je prázdný, nemá žádné vnořené prvky
:root	kořenový prvek dokumentu, typicky tag <html>
:first-child	první potomek prvku
:last-child	poslední potomek prvku
div:first-of-type	vybere takový div, který je prvním divem svého rodiče
li:last-of-type	vybere takový li, který je posledním li svého rodiče
:only-child	vybere všechny prvky, které jsou jediným potomkem svého rodiče
p:only-of-type	vybere takový odstavec (tag p), který je jediný odstavec svého rodiče
:nth-child(n)	n-tý potomek, kde n je číslo v závorce
:nth-last-child(n)	n-tý poslední potomek, kde n je číslo v závorce (poslední je jednička)
:nth-of-type(n)	n-tý potomek daného typu (třeba li:nth-of-type(4) je čtvrtý tag svého rodiče)
:nth-last-of-type(n)	n-tý potomek daného typu svého rodiče
:not(selektor)	vybírá každý prvek, který neodpovídá selektoru v závorce
:lang(cs)	prvek má nastaven atribut lang začínající znaky cs. Lepší řešit to selektorem.

Výběr dle atributu:

- pokud je potřeba stylovat pomocí css elementy html a neexistuje jiné rozlišení elementů než např. jejich popis (title), můžeme použít výběr na základě atributů

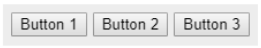
- tag[atribut], např. a[rel]{font-style: italic;} (všechny odkazy které mají nastaven atribut rel na nějakou hodnotu, budou kurzívou)

Výběr na základě hodnoty

HTML

```
<input type="button" value="Button 1">
<input type="button" value="Button 2">
<input type="button" value="Button 3">
```

Bez CSS

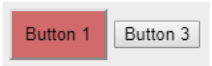


- nyní pomocí výběru na základě atributu hodnota (value) ostylujeme první input a druhý odstraníme

```
input[value="Button 1"] {
  background-color: #D46A6A;
  padding: 10px;
}

input[value="Button 2"] {
  display: none;
}
```

Výsledek



Výběr na základě popisu

Nyní ostylujeme elementy `<div>` dle jejich popisu (atribut title)

HTML

```
<div>Nějaký div</div>
<div title="Popis">Toto je popis.</div>
<div title="Vysvetleni">Toto je vysvětlení</div>
```

CSS

```
div[title="Popis"] {
  color: #116611;
  font-weight: 600;
}

div[title="Vysvetleni"] {
  color: #AA6C39;
}
```

Vysledek

Nějaký div
Toto je popis.
Toto je vysvětlení

Priorita pravidel:

1) Přednastavený vzhled -kaskádové styly respektují původní vzhled html elementů, pokud styly něco nemění tak to zůstane, např. když u tagu H2 změníme barvu tak se sice změní barva ale zůstane původní velikost písma (styly výslovně neupravují to co zůstává)

```
h2 {color: green}
```

2) Dědičnost - podřízený element přejímá formátování nadřazeného elementu, např. když se nastaví barva písma těla dokumentu na červenou tak budou červené vše (některé vlastnosti se nedědí např. border)

```
body {color: red}
```

3) Pozdější vyhrává - pokud se zapisuje vlastnost nějakého elementu vícekrát za sebou (třeba i různě), tak platí zápis který je uveden nejpozději, to se stává hlavně tak že něco deklaruji ve stylopisu stránky a pak je potřeba na určitém místě změnit

- pravidlo pozdější vyhrává neplatí v případě že je nějaký zápis uveden podrobnějším selektorem

```
<style>
p {color: red; font-style: italic}
</style>
<body>
<p style="color: blue">modrý odstavec kurzívou</p>
<p>normální červený odstavec kurzívou</p>
</body>
```

- pomocí !important se dá nastavit aby vlastnost nebyla později přepsána, např. h1 bude červený i když byl pokus změnit barvu

```
h1 {color: red !important}
h1 {color: blue}
```

4) Priorita vnitřního elementu - v případě konfliktu vyhrává ten element který je „vnitřnější“, je blíže formátovanému obsahu

```
<span style="color: red"><span style="color: green">obsah</span></span>,
```

→ obsah bude zelený

5) Vlastní nastavení - moderní prohlížeče dovolují uživateli používat při čtení stránek vlastní předlohu se styly

Boxing model:

- jakým způsobem můžeme tyto prvky umisťovat do stránky, každý prvek na stránce si můžeme představit jako obdélník s pevně danými rozměry - výška a šířka, prvek můžeme ohraničit (border), prvku můžeme také nastavit vyplnění (padding) a odsazení (margin)

- HTML elementům navíc můžeme přidat výšku a šířku pomocí vlastností height a width

- **Vyplnění (padding)** - když text přiléhá těsně k okraji prvku, je možné definovat vyplnění, které určuje vzdálenost obsahu od okraje prvku

```
.box-of-chocolates {
  padding: 30px;
  width: 200px;
  height: 100px;
  background-color: lightsalmon;
}
```

Krabička s čokoládou.

- **Ohraničení (border)** - ohraničení můžeme dát různé styly

```
.box-of-chocolates {
  padding: 30px;
  width: 200px;
  height: 100px;
  border: 2px dashed navy;
  background-color: lightsalmon;
}
```

Krabička s čokoládou.

- **Odsazení (margin)** - odsazení prvku je vzdálenost mezi okrajem (hranicí) elementu a dalším - sousedním prvkem

```css
.box-of-chocolates {
  padding: 30px;
  margin: 20px;
  width: 200px;
  height: 100px;
  border: 2px dashed navy;
  background-color: lightsalmon;
}
```

Krabička s čokoládou.

Krabička s čokoládou.