Handbook

HTML/CSS

LEARN 2CODE

O knihe

Predslov

Učebnica HTML/CSS je jednoduchý a ucelený návod venovaný začiatočníkom, ktorí sa chcú vzdelávať v oblasti tvorby webu. Nemusia mať žiadne predchádzajúce znalosti HTML alebo CSS. Treba mať na pamäti, že učebnica je napísaná podľa vlastných skúseností autora.

Autor

Autorom učebnice je Štefan Húska, ktorý sa venuje webom od roku 2003. Od roku 2008 sa špecializuje na vývoj webových aplikácií pomocou frameworku Ruby on Rails. Od roku 2011 pracuje na voľnej nohe. Dlhodobo spolupracoval s firmami Alice in Tokyo, Sun Marketing, Azet, Bonetics a ďalšími.

Pripomienky

Nápady, otázky alebo opravy smerujte na adresu info@learn2code.sk.

Verzie

Verzia	Vydanie	Popis zmien
1	7.3.2013	• Prvé vydanie
1.1	13.9.2013	• Čistenie dlhých textov • Úpravy v štruktúre
1.2	19.1.2014	• HTML 5
1.3	28.8.2014	• Úpravy v štruktúre • Doplnenie učebnice

Úvod do webových technológií

Vznik webu

Je dôležité nemýliť si vznik internetu a webu.

- Počiatky internetu siahajú už do 60-tych rokov.
- Samotný web, tak ako ho poznáme dnes, sa začal rodiť až neskôr, na konci roku 1990.

Tim Berners-Lee

Tim Berners-Lee zohral dôležitú úlohu pri vzniku webu.

O systéme podobnom WWW začal uvažovať Sir Tim Berners-Lee v roku 1980. Narodil sa v Londýne v roku 1955. V tom čase pracoval v CERN-e (Európska organizácia pre jadrový výskum).

V CERN-e v tej dobe pracovalo približne 10 000 zamestnancov. Každý z vedcov používal iný software a iný hardware. Väčšina komunikácie prebiehala pomocou emailu (ktorý existoval ešte pred vznikom WWW). Táto situácia spôsobovala, že výmena materiálov súvisiacich s ich prácou a projektami medzi kolegami, bola obtiažna.

Tim sa rozhodol, že vytvorí riešenie, ktoré spojí hardware a software do jedného kompatibilného celku (na úrovni komunikačných schopností). Projekt nazval **ENQUIRE**. Začal v roku 1980 a vývoj trval 6 mesiacov. Tim si vybral programovací jazyk Pascal.

ENQUIRE vyriešil problém s kompatibilitou sietí, diskových systémov, dátovými formátmi a znakovým kódovaním. Obsahoval už prvé znaky hypertextu. Funkcionalitou sa podobal skôr dnešnému systému WIKI ako webstránkam. Vývoj projektu sa ale neskôr zastavil.

Krátko na to sa v spolupráci s ďalšou osobou **Robert Cailliau** pustil do nového projektu WWW. Na konci roku 1990 zverejnili návrh, ako by Web mohol vyzerať.

Pôvodný návrh HyperTextu a WWW z 12. novembra 1990 nájdete na adrese http://www.w3.org/Proposal.html

4 piliere webu

Tim Berners-Lee je autorom štyroch základných pilierov webu:

1. jazyk HTML
2. protokol HTTP
3. webový server
4. webový prehliadač

Svet neporiadku

Problém č. 1: Benevolentnosť webových prehliadačov

Webový prehliadač, ktorý sa stará o vykreslenie HTML kódu maskuje a automaticky opravuje niektoré syntaktické chyby v zdrojovom kóde.

Na automatické zahladenie chýb sa ale nemôžeme stopercentne spoliehať - každý webový prehliadač má inú mieru tolerancie. Preto musí byť náš HTML kód vždy syntakticky správny.

Kontrola správnosti kódu musí prebiehať na našej strane, predtým než stránku publikujeme. Slúži nám na to nástroj nazývaný *validátor*, ktorý kontroluje správnosť HTML kódu. Validátor nájdeme na adrese http://validator.w3.org.

Autorom validátoru je organizácia W3C (World Wide Web Consorcium), ktorá sa okrem iného stará aj o štandardy HTML jazyka.

Problém č. 2: Rozdiely v prehliadačoch

Existuje viacero internetových prehliadačov:

- Google Chrome, Mozilla Firefox, Internet Explorer, Safari, Opera

Aj keď vo svojej podstate každý z prehliadačov plní rovnaký účel, môžu sa líšiť v dizajne alebo funkciách. Jedna súčasť prehliadačov je ale spoločná - **renderovacie jadro**.

Renderovacie jadro sa stará o to, že prijatý HTML kód spracuje do vizuálne príťažlivej podoby. Zdrojový kód na pozadí premení do grafickej podoby, ktorej rozumie bežný človek.

Tento prevod zdrojového kódu do grafickej podoby prebieha podľa istých pravidiel, ktoré určuje W3C. Faktom je, že každý prehliadač si tieto pravidlá vysvetľuje po svojom a vyskytujú sa tu malé odlišnosti.

Obzvlášť problematický prehliadač je Internet Explorer verzie 6. Príčinou je, že nedodržiava štandardy a obsahuje množstvo bugov. Verzia 6 je už ale výrazne na ústupe a používa ju minimum surferov.

Problém č. 3: Pomalý pokrok

Nové nápady a podnety na vylepšenie existujúcej technológie vznikajú každý deň. Preniesť ich ale do praxe, vzhľadom na to, že internet používa veľká časť populácie, je zodpovedná úloha. Vyžaduje si plánovanie, schvaľovanie a kompromisy.

Ekosystém, ktorý poháňa vývoj HTML a webových technológií vyzera v skratke nasledovne:

1. W3 **konzorcium**, zodpovedné za HTML jazyk, zverejní novú špecifikáciu (s novinkami alebo opravami). Vydanie takejto špecifikácie môžu sprevádzať roky diskusií.

2. Následne treba počkať, pokiaľ **vývojári** webových prehliadačov tieto zmeny premietnú do svojich produktov a vydajú nové verzie programov. To je druhý krok, ktorý môže trvať týždne až mesiace.

3. Potom sú na rade vývojari webových stránok. Nové postupy sa musia naučiť a vo forme kódu umiestniť do svojich webových stránok. Tento proces môže trvať mesiace až roky.
   Pozn.: Napríklad prechod od tabuľkového dizajnu ku CSS trval veľmi dlho.

4. A na koniec sú na ťahu **používatelia**. Tých treba presvedčiť, že si majú svoj prehliadač aktualizovať na najnovšiu verziu. Toto môže opäť trvať týždne až mesiace.
   Pozn.: S súčastnosti sa prehliadač Google Chrome dokáže aktualizovať úplne sám.

Verzie HTML

Prvá verzia HTML obsahovala 18 formátovacích značiek (tagov).

Každá verzia HTML jazyka sa snažila priniesť do špecifikácie nové značky. Napríklad tabuľky sa vyskytli až vo verzii 3.2.

V súčastnosti sa pripravuje HTML 5 a je stále vo vývoji, no dá sa používať už dnes.

Čo budeme a čo nebudeme potrebovať

Základné potreby

Povieme si aký software a služby budeme k tvorbe webstránok potrebovať. Vystačíme si s úplným minimom:

1. **Textový editor**
   Textový editor je program, v ktorom upravujeme zdrojový kód webstránky.

2. **Internetový prehliadač**
   Internetový prehliadač (alebo browser) je program, v ktorom si prezeráme webové stránky.

Cena

Textový editor aj webový prehliadač sú súčasťou operačného systému, sú teda **zadarmo**.

Ak by sme si chceli nainštalovať lepší editor a lepší prehliadač od tretích strán, nájdeme riešenia, ktoré sú tiež úplne zadarmo. V tomto smere teda žiadne prekážky nenájdeme.

Pokročilé potreby

Ďalšie potreby sú spojené s publikáciou webstránky na internet.

1. **Hosting**
   Hosting je služba, pomocou ktorej vieme naše stránky umiestniť na internet.

2. **Doména**
   Doména je jedinečný identifikátor počítača alebo počítačovej siete, ktorá je pripojená do internetu.

Príklad internetovej domény: www.google.sk, www.sme.sk.

Cena

Tieto položky nie sú veľmi nákladne, ak sa bavíme o malých alebo stredne veľkých weboch.

Hosting v nenáročnej podobe vie byť naozaj lacný, napríklad slovenská hostingová spoločnosť Websupport má ceny už od 50 centov za mesiac.

Freehosting môže byť aj zadarmo, ale býva často nepríjemne obmedzovaný - napríklad zobrazovaním reklám.

Náročné webové stránky samozrejme môžu mať náklady na prevádzku desiatky až stovky eur mesačne.

Doména sa registruje na 1 rok dopredu. Ceny slovenských domén sa pohybujú na hranici 15 eur za rok prevádzky. V prepočte na jeden mesiac teda 1,20 €.

Ak to zhrnieme, prevádzka webu vie byť v prvotnej fáze doslova za drobné - pod 2 € mesačne.

Bez čoho sa zaobídeme

Drahý software

Existuje množstvo predraženého softwaru, v podobe editorov s nadbytkom funkcií, ktoré nevyužijeme. Drahý software je v našom prípade zbytočný.

Znalosť programovania

Písať HTML kód ešte neznamená programovať. HTML je značkovací jazyk.

S programovaním sa určite stretneme, k vytvoreniu statických webových stránok ho však nebudeme potrebovať.

Pojmy

Webový prehliadač

Internetový prehliadač je program, v ktorom si prezeráme webové stránky. Plní teda zobrazovaciu funkciu.

Prehliadač plní svoju hlavnú funkciu v troch krokoch:

1. Pripojí sa na server cez HTTP protokol
2. Vyžiada si zdrojový kód na základe zadanej URL
3. Získaný zdrojový kód spracuje do prívetivej grafickej podoby

Najdôležitejšou súčasťou každého prehliadača je renderovacie jadro, ktoré sa stará o vykreslenie grafickej podoby stránky na základe zdrojového kódu.

Textový editor

Textový editor je program, v ktorom upravujeme zdrojový kód webstránky.

Všeobecne poznáme dva typy editorov: plain text a wysiwyg.

4. **Plain text editor** sa používa na priame písanie HTML kódu.
5. **WYSIWYG editor** zobrazuje text tak ako bude vyzerať na stránke. Nevyžaduje znalosť HTML a produkuje kód za nás.

   Pozn.: Nevýhodou WYSIWYG editoru je nedostatočná kvalita výsledného kódu, ktorá obsahuje veľa nepotrebných elementov.

Každý operačný systém (či už Windows, Linux, alebo Mac OS) má plain text editor predinštalovaný. Vo Windows je to napríklad Notepad (Poznámkový blok), ktorý bol štartovacím bodom pre nejedného začínajúceho kódera.

Pre pokročilé funkcie sa však oplatí zaobstarať si lepší editor. Asi najviac užitočnou funkciou, ktorú ponúkajú editory tretích strán je zvýrazňovanie syntaxe, prípadne automatické doplňanie tagov.

Zvýrazňovanie syntaxe je vlastnosť editoru, ktorá zobrazuje rôzne časti kódu v odlišných farbách. Táto funkcia uľahčuje čítanie kódu, a takisto pomôže včas odhaliť chyby syntaxe.

Automatické dopĺňanie je funkcia editora, ktorá sa pri písaní kódu snaží napovedať používateľovi slovo alebo html značku podľa prvých znakov. Takisto môže automaticky dopĺňať rôzne párove znaky, napríklad zátvorky. Môžu podporovať pokročilé funkcie ako napríklad verziovanie kódu pomocou GIT alebo SVN.

HTML súbor

HTML súbor je súbor s príponou .html. Takýto súbor obsahuje HTML značky. Vytvoríme ho jednoducho vytvorením prázdneho súboru a premenovaním jeho prípony.

Na publikovanie internetovej stránky na internet nám stačí iba jediný takýto súbor. Webové stránky sa ale často skladajú z desiatok až stoviek HTML súborov.

Webhosting

Webhosting je služba, ktorá umožňuje prevádzku webstránok. Hosting je v istom zmysle prenájom dátového priestoru a jeho softwaru na niektorom zo serverov.

Cena hostingu sa odvíja od parametrov ako diskový priestor, veľkosť RAM, počet databáz, veľkosť databázy, počet emailových schránok, ale aj podľa podporovaných programovacích jazykov alebo frameworkov (PHP, ASP.NET, Ruby on Rails).

Súčasťou služby býva zväčša možnosť pripojiť sa na server, napríklad cez protokol FTP. Pomocou FTP je možné kopírovať lokálne súbory na server a sprístupniť ich tak na internete.

Hostingové služby

Hostingových služieb poznáme niekoľko:

Free hosting

Hosting zadarmo, ale s obmedzeniami. Obyvkle malý diskový priestor, možnosť prevádzkovať web iba na doméne 3. rádu, a zobrazovanie reklám. A častokrát pri 30-dňovej neaktivite zvyknú celé konto zmazať.

Zdieľaný hosting

Najčastejší typ hostingu. Na jednom serveri je umiestnených niekoľko (stovky až tisíce) webstránok. Tieto weby zdieľajú napríklad RAM a CPU jedného servera. Takže výkon nie je fixný a môže kolísať.

Virtuálny hosting - VPS (virtuálny privátny server)

Softvérovo vytvorený server, ktorý sa správa a má vlastnosti ako plnohodnotný fyzický stroj.

Vyznačuje sa root prístupom na server. Je vhodný pre pokročilejších používateľov, vyžaduje konfiguráciu Linuxu a jeho správu.

Dedikovaný server

Prenájom celého serveru, ktorý si zákazník spravuje sám.

Serverhousing

Umiestnenie vlastného fyzického serveru do serverovne. Výhodou je klimatizovaná miestnosť (štandardne okolo 20 stupňov), zabezpečená a s garanciou pripojenia na internet.

Cloud hosting

Moderný hosting vhodný pre náročné weby. Hlavnou výhodou je škálovateľnosť výkonu podľa aktuálneho zaťaženia a náporu zo strany návštevníkov.

Platí sa väčšinou sa presne spotrebované prostriedky, či už je to výpočtový výkon, množstvo prenesených dát alebo iných parametrov.

Server

Server pomenúva dve veci: software aj hardware.

Server je **software** ktorý beží na serverovom stroji a obsluhuje požiadavky klientov (prehliadače). Odovzdáva vyžiadané stránky, spracúva emaily atď.

Pod pojmom server sa označuje aj samotný **hardware** (stroj/mašina), zapojený v sieti internet.

Vlastnosti

Server je zaujímavý v niekoľkých smeroch:

- **Server je zapnutý nonstop**
  Ak je server vypnutý, žiadna jeho služba nefunguje - webstránky sú offline, neprijíma emaily a pod. Pre nepretržitú dostupnosť jeho služieb musí byť stále zapnutý.
- **Má vysoký výkon**
  Hardware serveru musí čeliť veľkému náporu zo strany používateľov. Sú situácie, keď musí v jednej sekunde obslúžiť napríklad 200 rôznych požiadaviek.

- **Disponuje vysokorýchlostným pripojením na internet**
  Kvôli veľkému počtu požiadaviek musí mať rýchle pripojenie na internet.

Pýtate sa, čím sa líši server od bežného domáceho počítača? Veď napokon tieto znaky môže spĺňať takmer každý lepší domáci počítač.

Avšak...

Server vs. PC

- **Automatické zálohy**
  Asi prvým rozdielom je, že každý správny server zálohuje dáta :)

- **Bez monitora a klávesnice**
  Server nepotrebuje k svojej prevádzke žiadne periférie. Ani klávesnicu, ani myš. Nemajú ani zvukové karty alebo USB. Pre chod servera sú takéto zariadenia zbytočné.
  Interakcia so strojom prebieha výhradne po sieti.

- **Zabezpečenie proti výpadku**
  Servery bývajú kryté záložným zdrojom elektrickej eneregie - UPS.

IP adresa

IP adresa je unikátny číselný identifikátor počítača (konkrétne sieťovej karty) v sieti internet.

Aby bolo tvrdenie presné, IP adresa identifikuje každý počítač, ktorého uzol má priamy prístup na internet. Teda IP adresy vo vnútornej sieti môžu mať IP adresy nezávislé na zvyšnom internete (tzv. lokálne IP adresy).

IP adresy možno prirovnať k telefonným číslam.

Poznáme dve verzie protokolu IP - IPv4 a IPv6:

IPv4 (protokol IP verzie 4) - 1981

IP adresa je 32-bitové číslo, pre väčšiu zrozumiteľnosť sa ale zapisuje ako štyri 8-bitové čísla (čísla v rozsahu 0-255). Napríklad:

```
75.101.145.87
```

Tento spôsob identifikácie počítačov v sieti umožňuje viac než **4 miliardy kombinácii**.

Vzhľadom na súčasný počet počítačov zapojených v sieti, voľné IPv4 adresy sú takmer vyčerpané.

IPv6 - 1998

Nahrádza predchádzajúci štandard IPv4 z dôvodu nedostatku adresného priestoru.

Skladá sa z 32 hexadecimálnych číslic. Napríklad:

```
2001:db8:0:1234:0:567:8:1
```

Umožňuje 340 sextilionov kombinácií (10 na 36, alebo jednotka a 36 núl).

Napriek tomu, že IPv6 bolo vymyslené v roku 1998, do praxe sa uvádza až v týchto rokoch. Na konci roka 2012 tvoril dátový prenos cez IPv6 iba 1 % z celkového prenosu dát (zdroj).

Doména

Pre lepšiu prácu s internetovými adresami sa môže k IP adrese viazať meno, ktoré si vie človek ľahšie zapamätať.

Párovanie doménových mien a IP adries obstaráva DNS služba.

Opäť môžeme spomenúť analógiu s telefonnými číslami a telefónnym zoznamom. Aby sme si ich nemuseli pamätať, ukladáme si ich do telefónu a priraďujeme im mená.

Syntax zápisu domény

Doména sa skladá z troch častí.

1. Doména 1. radu (TLD)

   Môže označovať krajinu (dvojznakové označenie), napríklad .sk alebo .cz
   Alebo označuje kategóriu či organizáciu, napríklad .com, .org, .info

2. Doména 2. radu (SDL)

   Doménové mená, ktoré si môžeme slobodne zvoliť a registrovať - ak sú voľné v príslušnom bloku domény 1. rádu.

3. Doména 3. radu

Doména tretieho radu sa môže využiť na podsekcie portálov, napr. "sport", "kultura" v prípade sport.sme.sk, kultura.sme.sk.

V praxi sa tiež využiva na uľahčenie prístupu k službám, napríklad prístup k emailovej schránky cez mail.domena.sk.

HTML

HTML je značkovací jazyk, pomocou ktorého definujeme štruktúru obsahu webstránky. Na to aby sme zvládli túto tému, musíme sa naučiť:

- syntax jazyka
- dostupné HTML značky a ich účel
- zaužívané postupy a triky

Tag (značka) a syntax jazyka

Tag je základná jednotka HTML jazyka. Podieľa sa na štruktúre HTML dokumentu. Značka anglicky pomenúva funkciu, ktorú plní. Najčastejšie má názov vo forme skratky.

Zapisuje sa do "zobáčikových" zátvoriek. Napríklad:

```
<abbr>Telo značky</abbr>
```

Párové - nepárové tagy

Podľa formy, tagy rozdelujeme na:

1. **Párové tagy**

   Skladájú sa z počiatočnej a koncovej značky. Párové tagy majú obsah.

   Napríklad: `<h1>Nadpis</h1>`

2. **Nepárové tagy**

   Nenesú žiaden obsah, takže nemajú ukončovací tag.

   Ukončenie tagu prebieha priamo v počiatočnom tagu, lomkou na konci.

   Napríklad: `<hr/>`

Atribúty

Tag môže obsahovať ľubovoľný počet atribútov. Atribúty bližšie špecifikujú vlastnosti tagu.

Atribúty sa uvádzajú v počiatočnom tagu, formou zápisu `atribút=hodnota`.

```
<abbr title="Compact Disk">CD</abbr>
```

Pravidlá zápisu

Pre zápis HTML značiek si musíme zapamätať tri najdôležitejšie pravidlá:

- značky zapisujeme vždy malými písmenami
- prelínanie tagov je zakázané
- atribúty musia byť v úvodzovkách

Kategórie tagov

Podľa účelu rozdeľujeme tagy na nasledovné skupiny.

- Meta tagy
- Obrázky a objekty
- Formuláre
- Tabuľky
- Rámy

Meta tagy definujú vlastnosti stránky, napríklad titulok, kódovanie, alebo načítanie ďalších externých súborov (CSS, Javascript).

Podľa vlastostí rozdeľujeme tagy na blokové a inline (neblokové) elementy.

- Blokové elementy
- Inline elementy

Blokové elementy zabezpečia po svojom skončení zalomenie riadku. Naopak, **inline elementy** musia byť vo vnútri niektorého z blokových elementov a zalomenie riadku nezabezpečujú.

Štruktúra dokumentu

Každý HTML dokument sa musí skladať z týchto častí:

- definícia HTML verzie (doctype)
- wrapper pre HTML dokument (značka <html>)
- hlavička a titulok stránky (značky <head> a <title>)
- telo (značka body)

Základný HTML dokument vyzerá takto:

```
<!doctype html>
<html>
  <head>
    <title>Titulok stránky</title>
  </head>

  <body>
    Obsah stránky...
  </body>
</html>
```

V prvom riadku definujeme "**doctype**" - teda typ a verziu dokumentu. V našom prípade sa jedná o zápis pre dokument typu HTML a verzie 5 - HTML5.

Zvyšok stránky je zabalený v časti medzi značkami **<html> a </html>**. V tejto časti sa nachádza hlavička a telo dokumentu.

Hlavička (<head>) slúži na definíciu titulku stránky (<title>) a ďalších nastavení ako je kódovanie znakov, načítanie externých súborov CSS a pod.

Telo stránky (<body>) obsahuje samotný obsah, ktorý uvidí používateľ v prehliadači.

Prehľad najpoužívanejších tagov

Odstavec - p

Vytvára odstavec - paragraph. Je to jedna z najpoužívanejších html značiek.

```
<p>Blok textu...</p>
```

Nadpisy - h1..h6

Nadpisy definujú kapitoly. H1 označuje nadpis najvyššej - prvej úrovne, H2 podkapitolu, atď. Popisujú štruktúru a organizáciu dokumentu.

Väčšina prehliadačov zobrazuje nadpisy väčším a hrubým písmom. Tento vzhľad sa dá samozrejme upraviť pomocou CSS.

```
<h1>Nadpis 1. úrovne</h1>
<h2>Nadpis 2. úrovne</h2>
```

Číslovaný zoznam - ol

Číslovaný zoznam zobrazuje odrážky, štandardne s arabským číslovaním.

Pomocou atribútu 'type' je možné nastaviť zobrazovanie napríklad aj rímskych číslic. Úprava sa však odporúča robiť iba pomocou CSS (jazyk určený výhradne na definíciu štýlov, ktorý sa naučíme neskôr).

```
<ol>
  <li>Item 1</li>
  <li>Item 1</li>
  <li>Item 1</li>
</ol>
```

Nečíslovaný zoznam - ul

Nečíslovaný zoznam zobrazuje odrážky s grafickou ikonou - guličky, štvorčeky alebo krúžky.

V dizajnovaní sa tento element často využíva na vytvorenie navigácie, kde pomocou CSS možno ľahko docieliť žiadaný vzhľad.

```
<ul>
  <li>Item 1</li>
  <li>Item 1</li>
  <li>Item 1</li>
</ul>
```

Definícia - dl

Definičný list pozostáva z terminov a ich definícii. V párovej značke `dl` sa nachádza striedavo tag `dt` - definition term, a `dd` - definition of a term.

Prehliadač štandardne štýluje termín (dt) hrubým písmom a definíciu (dd) s miernym odsadením zľava.

```
<dl>
  <dt>HTML</dt>
  <dd>Značkovací jazyk na tvorbu webstránok</dd>
  <dt>JavaScript</dt>
  <dd>Skriptovací jazyk na výrobu interaktívnych prvkov</dd>
</dl>
```

Odkaz - a

Odkaz je interaktívny prvok, na ktorý je možné klikať. Pri kliknutí dochádza k presunu a zobrazeniu stránky, ktorá je uvedená v odkaze.

Najdôležitejší atribút je `href` - hypertext reference. Môže obsahovať URL alebo môže odkazovať aj na časť toho istého dokumentu.

```
<a href="kontakt.html">Kontaktujte nás</a>
```

Prehliadač zobrazuje odkazy štandardne ako modré a podčiarknuté, navštívené odkazy fialovou farbou.

Skratka - abbr

Značka pre zápis skratiek.

Atribút `title` a jeho hodnota vysvetľuje skratku v tele značky.

Pri ukázaní kurzorom myši na abbr tag, prehliadače štandardne zobrazia hodnotu atribútu `title` v bubline.

```
<abbr title="Compact Disc">CD</abbr>
```

Zdôraznenie - em

Slúži na zdôraznenie textu. Prehliadače zobrazujú em naklonenným písmom.

```
<em>Ochutnajte našu výbornú kuchyňu.</em>
```

Silné zdôraznenie - strong

Slúži na silné zdôraznenie textu. Prehliadače zobrazujú strong hrubým písmom.

```
<strong>Nezmeškajte túto príležitosť.</strong>
```

Zalomenie - br

Umožňuje zalomiť text na nový riadok.

Kedže zalomenia riadkov a iné "biele znaky" v zdrojovom kóde za sebou znamenajú na výstupe vždy len jednu medzeru, zalomenie pomocou `br` môže prísť vhod.

`br` by sa nemalo používať na členenie textu. Na tento účel sú určené odstavce. Členenie textu iba za pomoci `br` je častou chybou.

```
Prvý riadok<br/>Druhý riadok
```

Horizontálna čiara - hr

Vykreslí horizontálnu čiaru.

```
<hr/>
```

Obrázok - img

Umožňuje do dokumentu umiestniť obrázok.

Povinné atribúty sú src, ktorý určuje cestu k obrázku a atribút alt poskytujúci alternatívny text v prípade, že sa obrázok nepodarí zobraziť (Chrome a Safari alt nezobrazujú). Alternatívny text môže byť dôležitý aj z hľadiska čítačiek pre nevidiacich.

```
<img src="iphone.jpg" alt="iPhone 5"/>
```

div & span

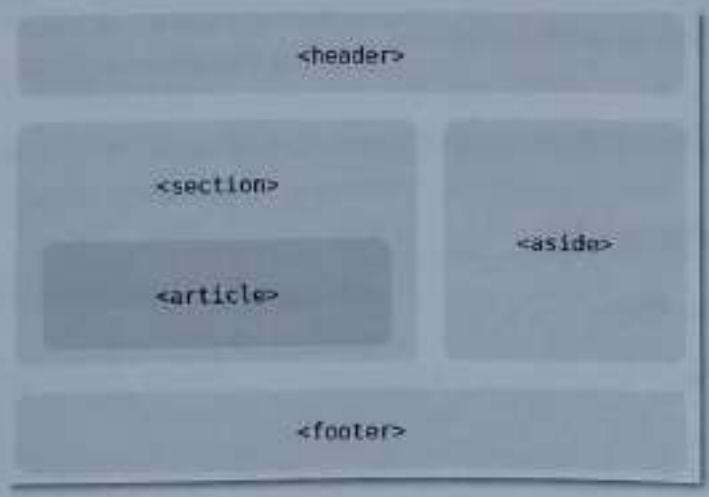
Tagy div a span nemajú žiadny sémantický význam. Používajú sa hlavne pri dizajnovaní stránok pomocou CSS.

Div je blokový element, span je inline element.

Štruktúra webstránky

Dlhé obdobie bola štruktúra webstránky vytváraná pomocou blokového elementu. Problémom bolo to, že tieto elementy neposkytovali žiadnu sémantickú hodnotu a mnohokrát bolo veľmi zložité určiť zámer použitia -ov. Našťastie s príchodom HTML5 boli uvedené nové elementy, ako napr. <header>, <nav>, <article>, <section>, <aside> a <footer>. Význam týchto značiek je poskytnúť lepšiu prehľadnosť štruktúry webstránky.

Jeden z možných príkladov HTML5 štruktúry webstránky môže vyzerať nasledovne:



Finálna špecifikácia HTML 5 a CSS 3 je stále v príprave, no používať sa dá už dnesx. Nové značky, vlastnosti a postupy boli vymyslené a vo väčšej miere sú implementované do súčasných prehliadačov. Preto ich používa čoraz viac vývojárov.

Čo prináša HTML 5

HTML 5 prináša mnoho vylepšení súčasného jazyka:

- nové sémantické značky
- nové formulárové značky a atribúty a ich validácie
- nové značky pre audio a video a ich ovládanie cez JavaScript
- geolokáciu - možnosť získať polohu zariadenia
- webstorage - offline úložisko pre dáta webu
- a ďalšie

Nový DOCTYPE

Ak chceme používať HTML 5, musíme používať nový zápis Doctype.

Doctype je deklaruje typ dokumentu. Hovorí o tom, akú verziu HTML má prehliadač zohľadniť pri vykreslovaní. Zapisuje sa na prvý riadok každého HTML dokumentu.

Nový zápis Doctype HTML 5 vyzerá nasledovne:

```
<!DOCTYPE html>
```

Nové sémantické značky

HTML 5 ponúka niekoľko nových značiek. Z toho sú najdôležitejšie tieto:

<article>	Definuje článok
<aside>	Definuje vedľajší obsah
<section>	Definuje sekciu v dokumente
<header>	Definuje hlavičku
<footer>	Definuje zápätie stránky
<nav>	Definuje navigáciu a jej odkazy
<mark>	Definuje zdôraznený text (highlighted)
<figure> a <figcaption>	Definuje akýkoľvek grafický prvok (obrázok, tabuľku, schému) a popis
<progress> a <meter>	Zobrazenie priebehu udalosti (progress), meranie hodnoty (meter)

<article>

Značka article reprezentuje kompletnú, samostatnú časť textu, ktorá vie byť publikovaná aj samostatne. Napríklad:

- blogpost
- článok v novinách
- komentár pod článkom

Značky article môžu byť do seba vnorené. Vnorený article znamená, že obsahovo súvisí s nadradenou značkou article (napríklad spomínané komentáre pod článkom).

Príklad značky article:

```
<article>
  <header>
    <h1>Novinky HTML 5</h1>
    <p>Publikované 28.9.2014</p>
  </header>
```

```
  <p>V tomto článku sa pozrieme na novinky v HTML a...</p>
</article>
```

Ukážka vnárania značky article na účel komentárov:

```
<article>
  <header>
    <h1>Novinky HTML 5</h1>
    <p>Publikované 28.9.2014</p>
  </header>

  <p>V tomto článku sa pozrieme na novinky v HTML a...</p>

  <section>
    <h1>Komentáre</h1>

    <article>
      <p>Super článok! Ďakujem.</p>
    </article>

    <article>
      <p>Kedy bude schválená špecifikácia?</p>
    </article>
  </section>
</article>
```

<section>

Značka section reprezentuje časť dokumentu v ktorej sú zoskupené viaceré celky.

Môžeme ich použiť napríklad na:

- oddelenie kapitol článku alebo knihy
- oddelenie častí hlavnej stránky ako predstavenie, novinky, kontaktné informácie

Príklad použitia značky section:

```
<article>
  <header>
    <h1>Recenzia Canon EOS 70D</h1>
```

```
    <p>Profesionálna zrkadlovka pod drobnohľadom.</p>
  </header>

  <p>V tomto teste sa pozrieme na funkcie tejto zrkadlovky.</p>

  <section>
    <h1>Parametre foťáku</h1>
    <p>...</p>
  </section>

  <section>
    <h1>Kvalita fotiek</h1>
    <p>...</p>
  </section>
</article>
```

<nav>

Značka nav reprezentuje časť dokumentu, ktorá obsahuje odkazy na ďalšie stránky alebo odkazy na časti toho istého dokumentu. Jedná sa o sekciu s navigáciou.

Ukážka použitia značky nav:

```
<body>
  <header>
    <h1>Super e-shop</h1>
    <nav>
      <h1>Menu</h1>
      <ul>
        <li><a href="produkty.html">Produkty</a></li>
        <li><a href="doprava.html">Doprava tovaru</a></li>
        <li><a href="kontakt.html">Kontakt</a></li>
      </ul>
    </nav>
  </header>

  <main>
    <article>
      ...
    </article>
```

```
  </main>

  <footer>
    <p>Copyright 2013</p>
  </footer>
</body>
```

Značka nav sa môže nachádzať v dokumente viackrát. Jedna môže slúžiť napríklad ako navigácia celého webu (podobne ako v príklade vyššie), druhá navigácia môže slúžiť ako navigácia častí obsiahleho článku.

Značka nav nemusí nutne obsahovať iba list odkazov. Môže sa jednať o súvislý text, podobne ako v tomto príklade:

```
<nav>
  <h1>Navigácia</h1>
  <p>
    Vitajte na mojej stránke. Nájdete tu články o <a href="#">psoch</a>,
    <a href="#">mačkách</a> a <a href="#">papagájoch</a>.
  </p>
  <p>
    V prípade otázok ma kontaktujte cez <a href="#">email</a>.
  </p>
</nav>
```

<aside>

Značka aside reprezentuje časť dokumentu, ktorá nie je hlavnou, ale vedľajšou časťou. Obsahovo však súvisí s hlavnou časťou stránky.

Značka sa dá využiť napríklad na tieto účely:

- typografické efekty pre citáty
- reklamné plochy
- skupiny <nav> elementov

Nasledujúci príklad ukazuje použitie značky aside v obsiahlej recenzii konkrétneho fotoaparátu. Neskúseným čitateľom vysvetľuje princíp zariadenia:

```
<aside>
  <p>
    Digitálna zrkadlovka je obdobou analógovej (filmovej) zrkadlovky,
    ale používa namiesto filmu digitálny senzor. Konštrukcia a činnosť
    fotoaparátu v porovnaní s filmovými zrkadlovkami zostala nezmená.
  </p>
</aside>
```

Značku možno taktiež použiť na citáty v článku:

```
<p>...</p>

<aside>
  <q>Podmienky v NHL sú nastavené tak, že sa hráč nemusí starať o nič
  iné len o hokej.</q>
</aside>

<p>...</p>
```

Značka aside sa tiež používa na bočnú stranu blogov, napríklad na umiestnenie archívu, spriatelených stránok alebo twitter feedu:

```
<body>
  <header>
    <h1>Môj blog</h1>
    <p>Píšem o všetkom čo ma napadne</p>
  </header>

  <aside>
    <nav>
      <h1>Spriatelené stránky</h1>
      <ul>
        <li><a href="#">Ferov blog</a>
      </ul>
    </nav>

    <nav>
      <h1>Archív</h1>
      <ol>
        <li><a href="#">Február</a>
```

```
        <li><a href="#">Január</a>
      </ol>
    </nav>
  </aside>

  <article>
    <h1>Môj článok</h1>
    <p>Lorem ipsum...</p>
  </article>
</body>
```

Pokročilé HTML značky

Tabuľky

Tabuľky prezentujú dáta v prehľadnej podobe. Tabuľky takisto vytvárame pomocou kódu. Na pomoc máme obsiahlu sadu HTML značiek.

Pre tabuľky máme k dispozícii 10 značiek:

table, tr, th, td, thead, tbody, tfoot, caption, col, colgroup

A dva atribúty:

rowspan, colspan

Pre základnú prácu s tabuľkami si ale vystačíme iba so štyrmi značkami:

table, tr, th, td

Vytvárame tabuľku

Tabuľka sa skladá z riadkov a stĺpcov, resp. s riadkov, v ktorých sú bunky. Ako prvé, vytvoríme obaľovaciu značku table.

```
<table>
</table>
```

Definujeme prvý riadok tabuľky. Použijeme značku tr (table row).

```
<table>
  <tr>
  </tr>
</table>
```

Do prvého riadku vložíme 2 bunky. Použijeme značku td (table data).

Pre účely ukážky ešte zapneme orámovanie buniek, aby sme štruktúru tabuľky videli lepšie. Učiníme tak atribútom border v table.

```
<table border="1">
  <tr>
    <td>Stefan</td>
    <td>Peter</td>
  </tr>
</table>
```

Naša prvá tabuľka je hotová.

Tabuľka s hlavičkou

Ukážeme si ďalší priklad, ktorý obsahuje hlavičku tabuľky.

Ako by sme napísali kód k takejto tabuľke?

```
| Meno   | Vek | Mesto      |
-------------------------
| Stefan | 31  | Bratislava |
| Jano   | 26  | Pezinok    |
```

V prvom rade vytvoríme obaľovaciu značku table:

```
<table border="1">
</table>
```

Ďalej definujeme hlavičku - prvý riadok s troma bunkami Meno, Vek, Mesto. Pre bunky použijeme tag th (table header cell).

```
<table border="1">
  <tr>
    <th>Meno</th>
    <th>Vek</th>
    <th>Mesto</th>
  </tr>
</table>
```

Ak si otvoríme html v prehliadači uvidíme prvý riadok buniek, ktoré tvoria hlavičku tabuľky.

Všimnime si formátovanie textu. Text vo vnútri značky th prehliadač vykresľuje štandardne hrubým písmom so zarovnaním na stred bunky.

Obdobným spôsobom definujeme dva riadky tabuľky, ktoré tvoria dáta. Použijeme html značku td (table data cell).

```
<tr>
  <td>Stefan</td>
  <td>31</td>
  <td>Bratislava</td>
</tr>
<tr>
  <td>Jano</td>
  <td>26</td>
  <td>Bratislava</td>
</tr>
```

Naša tabuľka je hotová.

Použitie thead, tbody a tfoot

Tieto tagy nie sú povinné, a nemajú vplyv na vzhľad, no je rozumné ich používať - definujú *štruktúru* tabuľky.

- `thead` - obaľuje skupinu riadkov (tr elementov), ktoré tvoria hlavičku
- `tbody` - obaľuje skupinu riadkov, ktoré obsahujú dáta
- `tfoot` - obaľuje skupinu riadkov, ktoré obsahuje pätičku

Ak si predstavíme skript, ktorý môže prípadný html dokument spracovávať, štruktúrovaná tabuľka mu značne uľahčí prácu. Bude vedieť, ktoré riadky tvoria iba dáta, ktoré riadky tvoria hlavičku.

Tieto tagy majú isté **pravidlá**. Tabuľka by mala obsahovať najviac jeden tag thead a najviac jeden tfoot. Môže však obsahovať ľubovoľný počet tagov tbody.

Príklad použitia thead, tbody a tfood

Predstavme si tabuľku, ktorá znázorňuje úspory:

```
| Mesiac | Uspory |
+--------+--------+
| Jan    | 200 €  |
| Feb    | 150 €  |
| Spolu  | 350 €  |
```

Aj keď má tabuľka spolu štyri riadky, jej dáta sú tvorené iba dvoma riadkami v strede. Prvý riadok je hlavička a posledný, sumárny riadok, tvorí pätičku.

Preto môžeme túto tabuľku v HTML zapísať takto:

```
<table>
  <thead>
    <tr>
      <th>Mesiac</th>
      <th>Uspory</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Jan</td>
      <td>200</td>
    </tr>
    <tr>
      <td>Feb</td>
      <td>150</td>
    </tr>
```

```
  </tbody>
  <tfoot>
    <tr>
      <td>Spolu</td>
      <td>350</td>
    </tr>
  </tfoot>
</table>
```

Zlučovanie buniek

HTML tabuľky nám umožňujú zlučovať bunky - či už v riadkoch, alebo v stĺpcoch. Na pomoc máme dva atribúty.

- colspan - zlučovanie buniek do stĺpcov
- rowspan - zlučovanie buniek do riadkov

Ukážeme si to na príklade s rozvrhom školských hodín:

```
    | 1. hod | 2. hod | 3. hod |
=============================
Po  | Fyzika |     Telesna     |
Ut  | Matika | Chemia | Fyzika |
```

Potrebujeme docieliť, aby sa bunka s telesnou výchovou "roztiahla" do dvoch stĺpcov. Použijeme colspan:

```
<table>
  ...
  <tr>
    <th>Po</th>
    <td>Fyzika</td>
    <td colspan="2">Telesna</td>
  </tr>
  ...
</table>
```

Chyba: Layout pomocou tabuliek

Ak vám práve napadlo, že pomocou tabuliek by sa dal vyrobiť dizajn stránok s hlavičkou, pätičkou a navigáciou naľavo, tak vedzte, že áno, dal. No zabudnite na to.

Pred niekoľkými rokmi bolo zaužívané vytvárať layout stránky pomocou tabuľky. Podpora jazyka CSS ešte nebola natoľko dokonalá a tabuľky boli akousi skratkou ako rozdeliť stránku na ľavú a pravú časť.

Použitie tabuliek na účel dizajnu je ale veľká chyba z hľadiska sémantiky (účelu). Účel tabuliek je zobrazovať dáta. Na vytvorenie layoutu stránky sa používa výhradne CSS.

Rámy

V súvislosti s rámami bývali k dispozícii dve možnosti.

- frameset
- iframe

Avšak, framesety boli kvôli problémom úplne vyradené zo špecifikácie. Preto ich používať nebudeme.

Iframe

Iframe je vnorený rám. Umožňuje zobraziť vo vnútri stránky inú stránku. Vnorený rám má nezávislý scrollbar. Je to párový tag, no do jeho obsahovej časti sa nič neumiestňuje.

```
<iframe src="page.html"></iframe>
```

Najdôležitejší atribút je src, ktorý určuje *source* - URL, ktorú má vnorený rám vykresliť.

Použitie v praxi: Embedovanie YouTube videí alebo reklamných plôch.

CSS

Čo je CSS

CSS je skratka pre Cascading Style Sheets - kaskádové štýly.

Je to jazyk na vizuálne formátovanie stránok. CSS popisuje vzhľad HTML dokumentov.

Hlavným **významom je oddelenie obsahu a štruktúry dokumentu od vzhľadu**.

Prvá špecifikácia

Autorom prvej špecifikácie CSS je Håkon Wium Lie (momenátlne CTO v Opera Software).

Verzie

- CSS1 (1996)
- CSS2 (1998)
- CSS2.1 (2011)
- CSS3

Význam a schopnosti CSS

Čo umožnuje CSS

Pomocou CSS máme pod kontrolou typografiu. Môžeme ovplyvňovať vlastnosti ako:

- farba, písmo, zarovnanie, veľkosť, orámovanie

Význam CSS

- **Oddeľuje obsah dokumentu od jeho vzhľadu**

  HTML značky neobsahujú definíciu vzhľadu.

- **Ľahšia údržba dizajnu**

Stačí zmeniť jeden riadok v CSS a ten ovplyvní sadu elementov. Túto výhodu uvidíme lepšie na príkladoch.

- **Úsporné vlastnosti**

  CSS je po správnosti uložené v externom súbore, takže sa načíta iba raz a pri prechádzaní ďalších stránok si prehliadač drží CSS v cache (vo vyrovnávacej pamäti). Tak sa redukuje množstvo prenesených dát po sieti a načítanie stránky je rýchlejšie.

Problémy webových prehliadačov

Trvalo veľa rokov, kým sa CSS dalo používať plnohodnotne - teda na kompletné dizajnovanie stránky, vrátane layoutu.

Webové prehliadače podporovali a podporujú CSS rôznou mierou. Poznáme dva typy problémov:

- Neúplna podpora vlastností
- Chyby v renderovaní

Dôsledkom toho bol nástup popularity CSS spomalený. Weboví dizajnéri, ktorí sa odhodlali používať najnovšiu technológiu, museli robiť ústupky a workaroundy (zvané CSS Hacky / CSS filtre).

CSS súbor

Súbor obsahuje CSS vlastnosti a nami zvolené hodnoty týchto vlastností. CSS súbor teda obsahuje sadu pravidiel, ktoré sa aplikujú na HTML značky.

Prípona súboru je .css.

Syntax

CSS kód má odlišnú syntax od jazyka HTML.

Pomocou CSS vždy definujeme súbor pravidiel pre html tag - či už globálne, alebo na jeho podskupinu podľa triedy (class) alebo identifikátora (ID).

```
selector {
  property: value;
}
```

- **Selektor** - určuje, pre ktoré elementy budú platiť pravidlá
- **Pravidlo** - určuje grafickú vlastnosť

Pravidlo sa skladá z vlastnosti a hodnoty. Viac pravidiel oddeľujeme bodkočiarkou.

Ukážka CSS kódu

Ak chceme určiť pravidlá pre odstavec, v kóde napíšeme:

```
p {}
```

`p` je v tomto príklade selektor a určuje, ktorý html tag nadobudne pravidlá. V tomto prípade odstavec. Kučeravé zátvorky obaľujú set pravidiel.

Zatiaľ sme nenapísali žiadne pravidlá. Určime pravidlo, ktoré definuje farbu písma:

```
p {
  color: red;
}
```

`color: red` je príklad pravidla. Pravidlo sa skladá z *vlastnosti* a *hodnoty*. V tomto prípade je vlastnosťou `color` a hodnotou `red`. Za pravidlom vždy píšeme dvojbodku.

Čo sa týka medzier a riadkovania, nezáleží na nich. Pomáhajú ale k lepšej čitateľnosti. Predchádzajúci kód je ekvivalentný tomu zápisu:

```
p { color: red; }
```

Za celým pravidlom vždy píšeme bodkočiarku. Bodkočiarka nám umožňuje oddeliť jednotlivé pravidlá, v prípade, že ich je viac.

Zápis dvoch pravidiel bude vyzerať takto:

```
p {
  color: red;
  font-size: 18px;
}
```

Teraz môžeme skúsiť aplikovať CSS kód na nasledujúci HTML kód:

```
<h1>Nadpis</h1>
<p>Prvý odstavec.</p>
<p>Druhý odstavec.</p>
```

Vo výsledku uvidíme každý odstavec červenou farbou s veľkosťou 18 pixelov. Nadpis H1 ostane neovplyvnený.

3 spôsoby pre vkladanie CSS do dokumentu

Štýl môžeme vkladať tromi spôsobmi.

1. Priamo v texte (inline zápis)

```
<p style="color: red">Cerveny text.</p>
```

Inline zápis pomocou atribútu `style` v kontrétnej html značke.

Tento zápis sa neodporúča. Je priamo v rozpore s významom CSS - odeliť obsah od dizajnu.

2. Použitie značky style

```
<style>
  p { color: red; }
</style>
```

CSS sa dá zapísať medzi tagy `<style>` a `</style>` (vo vútri tagu `<head>`), no tiež to nie je najvhodnejší spôsob. Prídeme tak napríklad o spomínanú výhodu cachovania (CSS kód tak bude zbytočne prenášaný po sieti pri každom zobrazení podstránky).

3. Použitie externého súboru

Použitie externého CSS súboru je **najlepší spôsob**. Výhodou je možnosť použiť jeden súbor na naštýlovanie niekoľkých stránok, resp. celého webu.

Nasledovný kód vložíme do súboru `syle.css`.

```
p { color: red; }
```

Následne zabezpečíme načítanie súboru style.css do html dokumentu pomocou meta tagu:

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

Triedy a indentifikátory

Triedy a identifikátory nám umožňujú väčšiu variabilitu v písaní pravidiel. Pomocou nich vytvárame set pravidiel, ktoré môžeme aplikovať na ľubovoľný tag.

CSS trieda (class)

```
.error { color: red; }
```

Pravidlo sa aplikuje na každý HTML element s triedou `error`.

CSS identifikátor (ID)

```
#notice { color: blue; }
```

Pravidlo sa aplikuje na každý HTML element s identifikátorom `notice`.

Príklad použitia CSS triedy

Predstavme si nasledujúci HTML kód:

```
<p>Normalny odstavec</p>
<p>Cerveny odstavec</p>
```

Ako docielime, že bude druhý odstavec červený? V prvom rade, html tagu `p`, pridáme atribút `class`.

```
<p>Normalny odstavec</p>
<p class="red">Cerveny odstavec</p>
```

V CSS následne vytvoríme pre túto triedu (class) pravidlo. Selektor pre triedu zapisujeme s bodkou na začiatku. Príklad:

```
.red { color: red; }
```

Druhý odstavec bude červený.

Príklad použitia CSS identifikátora

Podobne ako používame CSS triedy môžeme používať aj identifikátor. Identifikátor predstavuje ID elementu.

```
<p id="red">Cerveny odstavec</p>
```

CSS selektor sa zapisuje s mriežkou na začiatku:

```
#red { color: red; }
```

Pozn.: V tele dokumentu musí byť identifikátor unikátny. Teda nemôžeme dvom html značkám priradiť `id="red"`. ID využívajú hlavne skripty, kde je potrebné zabezpečiť odkaz na konkrétny element - a práve to zabezpečia unikátne označenia.

Pomenovanie tried a identifikátorov

Existujú isté pravidlá pre pomenovanie tried a identifikátorov: je povolené používať základné znaky, číslice a spojovník. Naopak:

- **Nepoužívať "_".** Namiesto toho používať spojovník, teda napríklad trieda s názvom .top-navigation.
- **Nepoužívať "ľščť".** Vyhýbajte sa diakritike a iným špeciálnym znakom. Nikdy nemáte istotu, že to bude fungovať.
- **Nezačínať číslovkou.** Názov triedy sa nemôže začínať číslovkou. Napríklad trieda s názvom .37-logo je nesprávna.

CSS vlastnosti

Tieto štyri vlastnosti sú pre prácu s CSS kľúčové:

Multi triedy	Aplikovanie viacero CSS tried na jeden element
Hromadná deklarácia	Definícia spoločných pravidiel pre viacero elementov naraz
Kontextová deklarácia	Definícia rôznych pravidiel pre rovnaký element na základe jeho umiestnenia v inom prvku (kontextu)
Kaskádovosť	Komplexné pravidlá pre aplikovanie vlastnosti na element, pokiaľ sú definované viackrát v rôznom kontexte

Multi triedy

Užitočnou vlastnosťou je, že HTML značke môžeme vymenovať viac tried.

Urobíme to tak, že do atribútu `class` napíšeme žiadané triedy oddelené medzerou.

```
.red { color: red; }
.underline { text-decoration: underline; }
<p class="red underline">
  Cerveny a podciarknuty.
</p>
```

Hromadná deklarácia

Takisto môžeme deklarovať pravidlá pre viac elementov naraz. Oddeľujeme ich čiarkou. Napríklad:

```
h1, h2, h3 { color: blue; }
```

Všetky nadpisy prvej, druhej a tretej úrovne budú modré.

Vyššie uvedený zápis je ekvivalentný zápisu:

```
h1 {color: blue;}
h2 {color: blue;}
h3 {color: blue;}
```

Hromadné deklarácie nám umožnia písať menej kódu. Oplatí sa ich preto využiť.

Kontextová deklaracia

Pravidlo môžeme deklarovať aj v nejakom kontexte. Napríklad majme HTML kód:

```
<h1>Nadpis s <em>kurzivou</em><h1>
<p>Odstavec s <em>kurzivou</em></p>
```

Všimnime si, že oba rodičovské elementy (h1 a p) obsahujú element .

Čo ak by sme chceli docieliť, aby bol `em` tag v značke <p> podčiarknutý, ale v nadpise <h1> nie? Môžeme postupovať dvomi spôsobmi.

Mohli by sme em elementom prideliť atribút `class` a vytvoriť preň pravidlo.

My však využijeme možnosť *kontextovej deklarácie*.

```
p em {
  text-decoration: underline;
}
```

Tento zápis určuje pravidlo podčiarknutia iba pre také `em`, ktoré je vo vnútri elementu `p`.

Kaskádovosť

Kaskádovosť je jedna z najdôležitejších vlastností CSS. Jeho logika vykreslovanie pozostáva z troch krokov v tomto poradí:

1. prednastavené vlastnosti
2. dedičnosť
3. kaskáda (posledný prebija)

Prednastavené vlastnosti

Každý element má svoje preddefinované hodnoty niektorých vlastností. Použijú sa vždy ak ich nezmeníme. V'daka tomu nemusíme znova a znova definovať CSS pre každý element.

Napríklad farba pozadia je štandardne biela, farba písma čierna, veľkosť písma 100%, resp. 1em a podobne.

Dedičnosť

Skôr než sa na element použijú prednastavené vlastnosti, prebehne pokus zdediť ich od rodičovského elementu.

Nie všetky vlastnosti sú ale dedičné. Obecne sa dedia iba vlastnosti popisujúce štýl textu (farba, písmo, veľkosť, výška riadku).

Na nasledovnom kóde si môžeme všimnúť ako funguje dedičnosť:

CSS:

```
p { color: red; border: 1px solid green; }
```

HTML:

```
<p>Odstavec s <strong>vnoreným prvkom</strong></p>
```

Vo výsledku bude odstavec červený, a to vrátane vnoreného prvku strong. Ten totiž zdedil vlastnosť color z rodičovského elementu p.
Vnorený prvok ale nebude mať orámovanie, pretože orámovanie nie je dedičná vlastnosť. Orámovanie bude mať iba prvok p, ktorý ho má definovaný priamo.

Kaskáda

Každý element môže mať definovanú jednu vlastnosť viackrát. Na to, akú hodnotu potom CSS uprednostní, existujú komplexné pravidlá.

Konkrétnosť selektoru

Dôležitá je konkrétnosť selektoru. Prednosť má pravidlo, ktoré popisuje cestu k prvku presnejšie.

Predstavme si tzv. bodový systém:

- Najobecnejší CSS selektor, v ktorom uvedieme samotný názov HTML tagu sa počíta za 1 bod.
- Konkrétnejší selektor, v ktorom uvedieme triedu (class), získa 10 bodov.

- Najlepšie ukazujú na prvok identifikátory (id), za tie sa počíta 100 bodov.
- Nakoniec sa použije definícia, ktorá získa najviac bodov.

```
p { color: red; }              /* 1 bod */
p strong { color: green; }     /* 1 + 1 = 2 body */
strong.blue { color: blue }    /* 1 + 10 = 11 bodov*/
#orange { color: orange; }     /* 100 bodov */
```

Prebíjanie

Pokiaľ máme niekoľko pravidiel s rovnakou váhou, použije sa to, ktoré je definované ako posledné.

```
p { color: blue; font-style: italic; }
p { color: green; }
```

Odstavec, pre ktorý sa aplikujú tieto 2 pravidlá, bude zelený (green), pretože neskoršie pravidlo to prvé prepíše. Navyše, odstavec bude písaný kurzívou (italic), pretože túto vlastnosť druhé pravidlo neovplyvňuje.

Pseudo-elementy a pseudo-triedy

Pseudo-elementy a psedo-triedy umožňujú abstraktný prístup k stromu HTML dokumentu.

Zápis pseudo-selektoru sa zapisuje s dvojbodkou na začiatku:

```
selector:pseudo-class { property: value; }
```

Pseudo-elementy

Pseudo-elementy vytvárajú abstrakciu nad štruktúrou dokumentu. Umožňujú pristupovať k elementom, ktoré v skutočnosti nie sú zapísané v HTML strome.

Príkladom môže byť :first-letter pomocou ktorého pristúpime k prvému znaku elementu. Môžeme tak napríklad definovať farbu prvého znaku odstavca.

```
p:first-letter { color: red; }
```

Pseudo-triedy

Pseudo-triedy ukazujú na nejakú interakciu s elementom (s výnimkou pseudotriedy napríklad :first-child, ktorá sa dá vydedukovať z HTML stromu).

Pseudo-triedy sa používajú na štýlovanie špeciálnych udalostí elementov.

Ako príklad si uvedieme odkaz a udalosť prejdenia kurzorom myši nad prvok (hover).

Takýto stav vieme odchytiť a napísať preň štýly. Napríklad, chceme aby sa odkaz pri ukázaní myšou zafarbil na červeno:

```
a:hover { color: red; }
```

Ďalšie selektory

Existujú ďalšie selektory, ktoré môžu byť v niektorých prípadoch užitočné.

Selektor *

Vyberie všetky prvky na stránke, resp. vo vnútri elementu v prípade kontextovej deklarácie.

```
* { color: red; }
```

Hviezdičkový selektor sa nám neskôr zíde pri dizajnovaní, s použitím jednoduchého triku získame tzv. debug-mód.

Selektor >

Vyberá prvok, ktorý je priamy potomok iného prvku.

```
ul > li
```

Selektor +

Vyberá prvok, ktorý nasleduje za iným elementom. Napríklad h1 + p naštýluje iba odstavec, ktorý je hneď za nadpisom (prvý výskyt tejto značky).

```
h1 + p
```

Selektor []

Vyberá prvok, ktorý má konkrétny atribút, napríklad a[title] vyberie každý odkaz ktorý má atribút title, s akoukoľvek hodnotou.

Tento selektor umožňuje vybrať aj elementy s atribútom, ktorý ma konkrétnu hodnotu.

```
input[type="text"]
```

Jednotky v CSS

Možností je veľa:

- px, mm, cm, in, pt, pc, em, ex, rem, %.

Používanie jednotiek v CSS môže pôsobiť trochu rozpačito, preto si radšej uvedieme iba tie najpoužívanejšie. Najpoužívanejšie jednotky sú:

- px, em, %.

Za každou číselnou hodnotou musí nasledovať jednotka.

```
  font-size: 1.2em
      width: 250px
line-height: 110%
```

V CSS je uvedenie samotného čísla bez jednotky nevalidné (Internet Explorer to ale zvykne chybne akceptovať a považuje číslo za hodnotu v pixeloch).

Box Model

Predtým, ako sa pustíme do box modelu, je dobré vysvetliť si, ako sú jednotlivé elementy na stránke zobrazované.

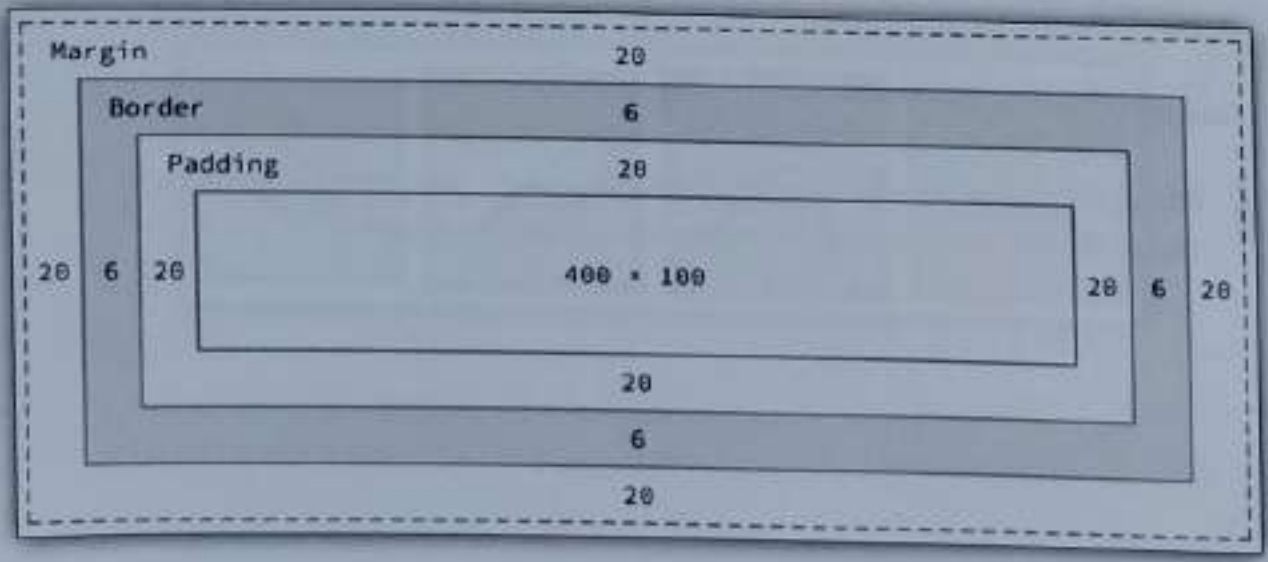
Display

To, ako sú jednotlivé elementy zobrazené (blokové, inline alebo iné), závisí od CSS pravidla *display*. Každý element má prednastavenú hodnotu pravidla display, samozrejme túto hodnotu môžeme prepísať. Najznámejšie hodnoty pravidla display sú: **block**, **inline**, **inline-block** a **none**. Hodnotu pravidla display nami zvoleného elementu môžeme zmeniť deklaráciou nového pravidla:

```
p {
   display: block;
}
```

Box model

Box model v CSS hovorí o tom, ako sú usporiadané jednotlivé atribúty width, height, padding, margin a border tak, aby spolu vytvorili objekt. Týmito atribútami sú vymedzené obdĺžnikové oblasti stránky (každá oblasť je tzv. box), ktoré ohraničujú výsledný obsah objektu. Príklad box modelu je na nasledujúcom obrázku:



CSS zápis pre predchádzajúci obrázok by vyzeral nasledovne:

```
div {
    width: 400px;
    height: 100px;
    padding: 20px;
    border: 6px solid #949599;
    margin: 20px;
}
```

Width & Height

Každý element má svoj šírku (*width*) a výšku (*height*). Blokové elementy majú štandardnú šírku nastavenú na 100% a tým využívajú celý horizontálny priestor, ktorý je k dispozícii.

Margin & Padding

Margin je CSS vlastnosť, ktorá určuje šírku vonkajšieho okraja prvku. Padding je vnútorný okraj. Pod paddingom sa vykresľuje pozadie, pod marginom nie. Margin, padding a border je možné zadať jednou hodnotou naraz pre všetky štyri strany, alebo vypíšeme hodnoty pre všetky strany. Strany sa zapisujú v poradí smeru hodinových ručičiek:

1. horná strana (padding-top)
2. pravá strana (padding-right)
3. dolná strana (padding-bottom)
4. ľavá strna (padding-left)

Pokiaľ nejaká hodnota na konci chýba, vezme sa do úvahy protiľahlá. Keď je v zápise len jedna hodnota, platí pre všetky štyri strany prvku.

Príklad zápisu paddingu môže vyzerať takto:

```
div {
    padding: 10px 20px 30px 40px;
}
```

Tento zápis je ekvivalentný tomuto:

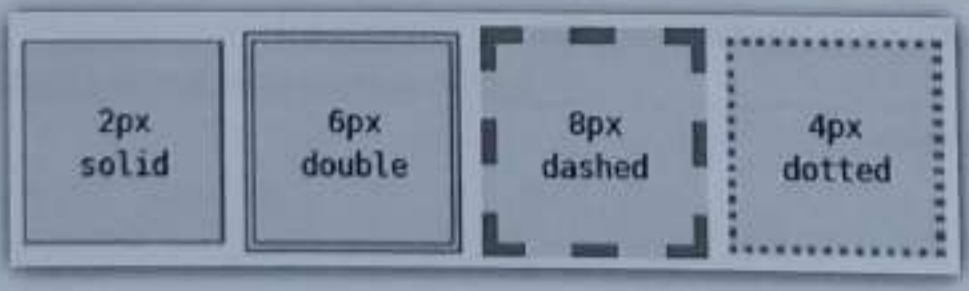
```
div {
    padding-top: 10px;
    padding-right: 20px;
    padding-bottom: 30px;
    padding-left: 40px;
}
```

Border

CSS vlastnosť border určuje vlastnosti rámčeku - ide o nasledovné vlastnosti: border-width, border-style a border-color (resp. len width, style a color). Kód pre 10 px široký neprerušovaný modrý rámček bude vyzerať nasledovne:

```
div {
    border: 10px solid blue;
}
```

Rámčeky môžu mať rôzne orámovanie, príklad uvádzame na nasledujúcom obrázku:

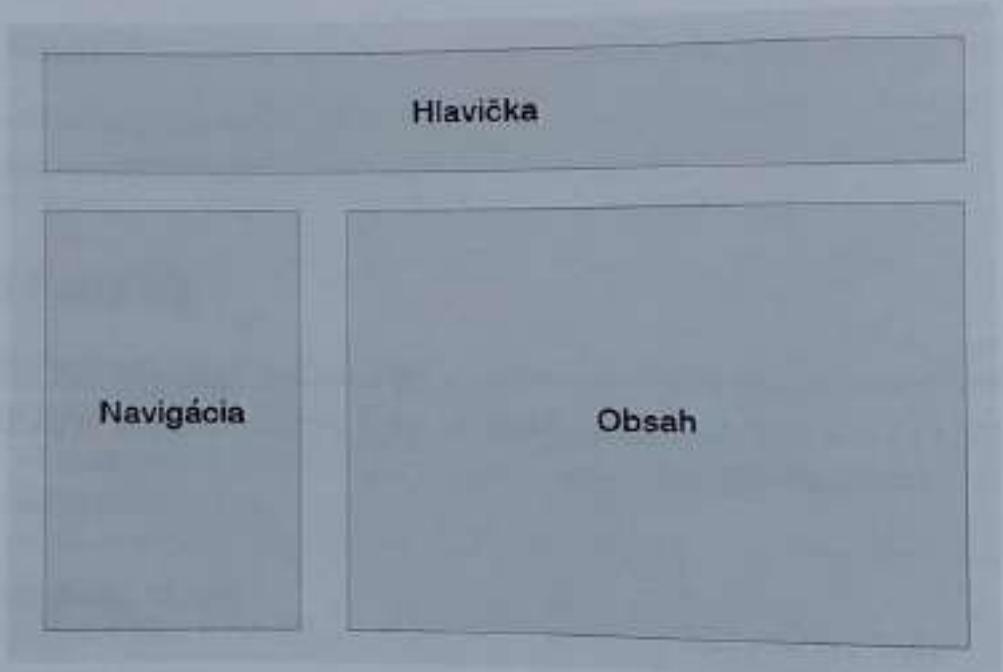


CSS layout

Častou úlohou kódera je vytvoriť základný layout - kostru webstránky.

Layout stránky najčastejšie pozostáva z týchto častí:

- hlavička s logom
- ľavá časť s navigáciou
- pravá časť s obsahom



Budeme potrebovať HTML kód, ktorý pozostáva z troch div elementov. Určíme im tiež príslušné CSS triedy "header", "navigation" a "content":

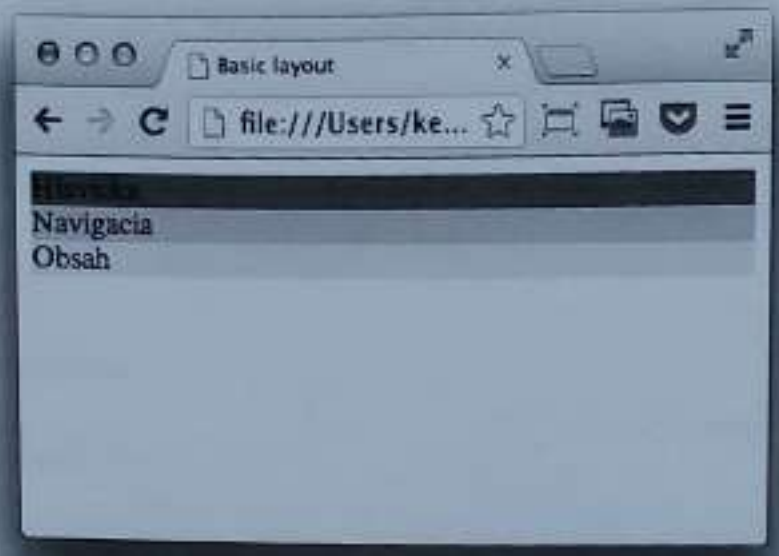
```
<!doctype html>
<html>
  <head>
    <title>Basic layout</title>
    <link rel="stylesheet" href="style.css"/>
  </head>

  <body>
```

```
    <div class="header">
      Hlavicka
    </div>

    <div class="navigation">
      Navigacia
    </div>

    <div class="content">
      Obsah
    </div>
  </body>
</html>
```

V hlavičke dokumentu si môžeme všimnúť načítanie externého súboru "style.css". CSS súbor vytvoríme a na začiatok nastavíme iba farby pozadia:

```
.header {
  background: red;
}
.navigation {
  background: lightgreen;
}
.content {
  background: yellow;
}
```

V prehliadači by sme mali vidieť takýto výsledok:

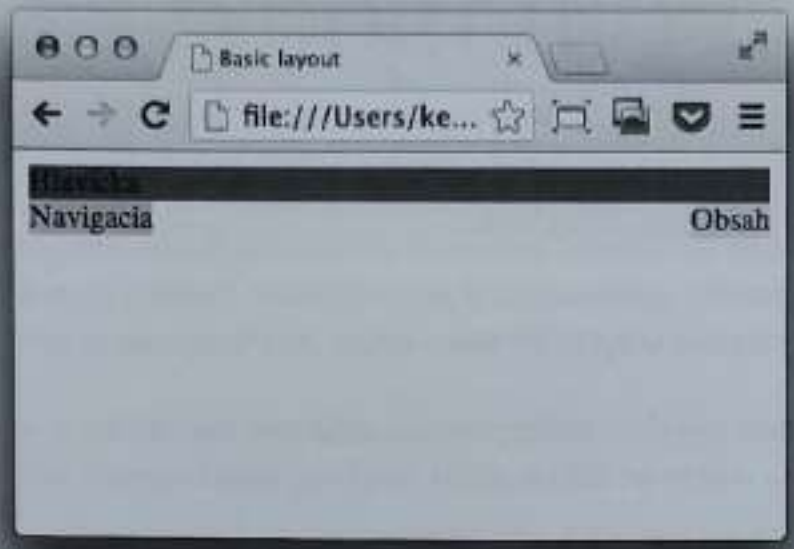


Ako môžeme vidieť, elementy sú umiestnené pod sebou, kedže je blokový element a zaberá plnú šírku rodičovského elementu.

Teraz elementom "navigation" a "content" nastavíme vlastnosť float:

```
.navigation {
  background: lightgreen;
  float: left;
}
.content {
  background: yellow;
  float: right;
}
```

Elementy sa teraz umiestnia doľava a doprava, ako môžeme vidieť na obrázku:

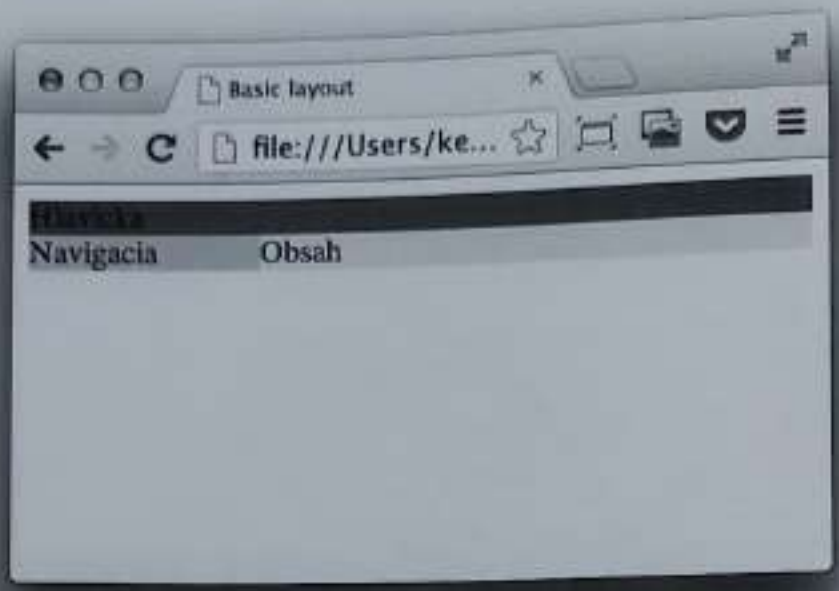


Ostáva nám vyriešiť problém so šírkami elementov, ktoré sa sa nastavujú nie podľa našich predstáv. Šírka elementov sa momentálne vypočítava automaticky.

Nastavme šírku elementov napríklad v percentách:

```
.navigation {
  background: lightgreen;
  float: left;
  width: 30%;
}
.content {
  background: yellow;
  float: right;
  width: 70%;
}
```

Týmto docielime žiadaný výsledok:


Basic layout
file:///Users/ke...
Hlavicka
Navigacia
Obsah

Práca s typografiou

V minulosti sme boli obmedzovaní malým počtom písiem, ktoré sme mohli použiť na našich webstránkach. Tieto typy písma boli väčšinou nainštalované fonty v počtačoch, takže sa s najväčšou pravdepodobnosťou na obrazovke vykresľovali správne. Ak písmo nebolo nainštalované na počítači, nezobrazilo sa. V súčasnosti sú možnosti podstatne lepšie a s možnosťou *embeddovať* font, máme oveľa väčší výber písiem ako v minulosti.

S týmito možnosťami ale zároveň prichádza nutnosť poznať možnosti webovej typografie a správne používanie rôznych fontov pomocou HTML a CSS na našich webstránkach.

Web safe fonts - bezpečné písma pre web

Komplatibilita tvorí veľkú časť webového designu. Ak chceme, aby prehliadač zobrazil určitý typ písma, musí byť v počítači nainštalovaný (web safe fonts) alebo odkazovaný pomocou CSS3 font-face.

Pozn.: Existujú aj rôzne techniky vykreslovania písma, ktoré sa spoliehajú na SVG, JavaScript (Cufón) alebo Flash (sIFR). Vo všeobecnosti sa vzhľadom na prístupnosť webovej stránky neodporúčajú.

Existuje sada písiem web safe fonts, ktorá je predvolene nainštalovaná na väčšine operačných systémoch.

Windows typ písma	Mac typ písma	Rodina písiem
Arial	Arial, Helvetica	sans-serif
Arial Black	Arial Black, Gadget	sans-serif
Comic Sans MS	Comic Sans MS	cursive
Courier New	Courier New	monospace
Georgia	Georgia	serif
Impact	Impact, Charcoal	sans-serif
Lucida Console	Monaco	monospace
Lucida Sans Unicode	Lucida Grade	sans-serif
Palatino Linotype	Palatino	serif
Tahoma	Geneva	sans-serif
Times New Roman	Times New Roman, Times	serif
Trebuchet MS	Trebuchet MS	sans-serif
Verdana	Verdana	sans-serif
Symbol	Symbol	-
Webdings	Webdings	-

Zdroj: http://www.ampsoft.net/webdesign-l/WindowsMacFonts.html

Externé fonty

Častokrát webdesignerom web safe fonts nestačia a tak musia siahnuť po externých fontoch. Medzi najznámejšie zdroje fontov patria napr.:

Google Fonts

Open source fonty

https://www.google.com/fonts

My fonts

Platené fonty

http://www.myfonts.com/

DaFont

Open source fonty

http://www.dafont.com/

Tip:

Ak potrebujete zistiť typ písma napr. z obrázku, vyskúšajte online službu http://www.myfonts.com/WhatTheFont/

Typ písma
Predstavuje určitý názov rodiny alebo kolekciu fontov.

Font
Predstauje jeden rez písma určitej kolekcie písiem.

Efekty pre text

Farbu a veľkosť písma môžeme zmeniť nasledovným zápisom:

```
p {
  color: red;
  font-size: 18px;
}
```

Okrem toho je možné meniť riadkovanie (*line-height*), okraje (*margins*), štýl fontu (*font-style*), ktorý môže byť normal, italic, oblique, inherit. Poznáme tiež *font-weight*, táto vlastnosť môže nadobúdať hodnoty bold, bolder, lighter, inherit. Tejto vlastnosti je možné zadať aj číselné hodnoty (100, 200, ..., 900).

Text zarovnávame pomocou vlastnosti *text-align*, ktorá má 5 hodnôt: left, right, center, justify a inherit. Text môžeme spraviť atraktívnejším použitím vlastnosti *text-decoration*, ktorá nadobúda hodnoty none, underline, overline, line-through a inherit. Príklad použitia:

```
p {
  text-align: center;
  text-decoration: underline;
}
```

CSS 3 prináša niekoľko efektov ktoré možno aplikovať na text.

Vlastnosť	Účel
text-shadow	Tieňovanie textu
word-wrap	Zalomenie textu aj na úkor rozdelenia slova

Text-shadow

Pomocou text-shadow si môžeme navoliť tieň textu:

```
h1 {
  text-shadow: 5px 5px 5px blue;
}
```

Vlastnosť text-shadow je využívaná pomerne vo veľkej miere, možno pomocou nej jednoducho dosiahnuť napríklad zaujímavý 3D efekt.

Word-wrap

Vlastnosť text-wrap sa nám zíde, ak potrebujeme zaručiť aby nám cez element nepretiekol text.

Hodnota break-word zaručí zalomenie dlhých slov v prípade, že sa do elementu nevmestí:

```
p {
  word-wrap: break-word;
}
```

Ľubovoľné písma

Pohľad do nedávnej minulosti

Pred vznikom CSS boli kóderi nútení používať iba tzv. bezpečné písma, teda písma ktoré sú nainštalované na väčšine klientskych počítačoch.

Tento problém sa riešil nepekným spôsobom - v grafickom editore sa vytvoril nadpis v písme podľa priania a ten sa následne uložil ako obrázok.

Nevýhoda tohto riešenia spočíva v tom, že nemožno zmeniť znenie textu bez opätovného uloženia obrázku. Tiež sa toto riešenie dalo použiť iba na nadpisy, kedže veľké bloky textu by nebolo možné spravovať. A po tretie, text ukrytý v obrázku Google bot neprečíta.

Kóderi pochopiteľne hľadali nové cesty a jedno obdobie boli v kurze JavaScript-ové knižnice, ktoré nahrádzali každý písaný znak za obrázkovy znak v danom písme osobitne. Prepísať text teda už nebol žiadny problém.

@font-face

CSS 3 tento problém rieši a prináša pravidlo @font-face. Pomocou tohto pravidla možno načítať súbor s písmom do dokumentu ako externý súbor (podobne ako CSS alebo JavaScript):

```
@font-face {
  font-family: myCustomFont;
  src: url(./font.woff);
  font-weight: bold;
}
```

Následne možno používať názov fontu ako hodnotu vlastnosti font-family:

```
h1 {
  font-family: myCustomFont;
}
```

Formuláre

HTML formuláre umožňujú posielať dáta od používateľa smerom na server.

V praxi sa s nimi stretávame dennodenne - vo vyhľadávači google alebo pri registráciach.

Formuláre disponujú niekoľkými prvkami - textovými poliami, výberovými zoznamami, tlačidlami a ďalšími.

Formulár - form

Formuláre, podobne ako tabuľky, pozostávajú zo setu HTML značiek. Všetky musia byť vo vnútri <form> a </form>.

Povinnými atribútami sú `action`, ktorý určuje kam formulár posiela dáta, a atribút `method`, ktorý definuje spôsob akými dáta posiela.

```
<form action="/contact/send_mail" method="post">
  ...
</form>
```

Input

Všeobecné vlastnosti

Input je najpoužívanejším HTML elementom formulárov. Je to nepárový tag.

Dôležité atribúty:

- Pomocou atribútu `type` určujeme, aký typ prvku chceme vykresliť.
- Atribút `name` slúži na identifikáciu políčok a získanie ich hodnôt pri spracovaní formuláru.

Máme k dispozícii niekoľko typov input značiek, ktoré určujeme pomocou atribútu `type`:

- krátky textový vstup
- políčko na heslo

- zaškrtávacie políčko
- prepínacie tlačidlo
- súbor
- skryté dáta
- tlačidlo na odoslanie

Textový vstup

```
<input type="text" name="first_name"/>
```

Umožňuje zadávať text krátkeho rozsahu, na jeden riadok.

Heslo

```
<input type="password" name="password"/>
```

Jeho účelom je možnosť zadať heslo. Zadané znaky sú totiž maskované, aby sa vstup nedal odsledovať "cez rameno". Väčšinou sú znaky maskované formou "bodiek" alebo hviezdičiek.

Využitím v praxi je napríklad registračný alebo prihlasovací formulár, alebo kdekoľvek, kde po používateľovi požadujeme zadanie hesla.

Zaškrtávacie políčko

```
<input type="checkbox" name="remember_me" value="true"/>
```

Zaškrtnutý checkbox odosiela hodnotu zadanú vo `value`, nezaškrtnutý neodosiela nič.

Do formuláru je možné začleniť viacero checkboxov s rovnakým `name`.

Prepínač

```
<input type="radio" name="gender" value="man"/> muž
<input type="radio" name="gender" value="woman"/> žena
```

Prepínač alebo "radio button" umožňuje vybrať jednu hodnotu spomedzi viacerých. Tie sa vkladajú s rovnakým `name` a rôznymi hodnotami. Odosiela sa iba hodnota vybraného prvku.

Súbor

```
<input type="file" name="avatar"/>
```

Zobrazí prvok, ktorý slúži na nahratie súboru. Spolupracuje s operačným systémom, pretože vyvolá zobrazenie okna na výber súboru z počítača.

Dlhodobým problémom je, že sa nedá preložiť textový popis tlačidla "Choose File" (Chrome) alebo "Browse" (IE).

Skryté dáta

```
<input type="hidden" name="country" value="SK"/>
```

Tento prvok je skrytý a používateľ s ním nemá možnosť interakcie. Úspešne odosiela jeho dáta.

Pozor, vyhnite sa akémukoľvek použitiu na účely odosielania citlivých dát. Nikdy ho nepoužívajte na prenos hesla.

Odosielacie tlačidlo

```
<input type="submit" value="Odoslat" />
```

Submit vykresľuje tlačidlo, ktoré odosiela formulár na ďalšie spracovanie. Vyvolá odoslanie zadaných dát v ostatných prvkoch formulára na adresu uvedenú v `action` značky `form`.

Štítok

```
<label for="name">Meno</label>
```

Štítok, alebo label, je z hľadiska použiteľnosti a pohodlia veľmi užitočný element.

Tvorí pár s akýmkoľvek inputom, previazaný je cez zhodný atribút `name`. Po kliknutí na label sa nastaví fokus do spárovaného inputu.

Textové pole

```
<textarea name="message"></textarea>
```

Textové pole dovolí vkladať dlhý text na viacej riadkov.

Výberový zoznam

```
<select name="country">
  <option value="sk">Slovensko</option>
  <option value="cs">Cesko</option>
</select>
```

Výberový zoznam, select, nazývaný aj drop-down list.

Umožňuje vybrať jednu, alebo aj viacero hodnôt z preddefinovaného listu.

Štandardne je možné vybrať iba jednu hodnotu, ale atribút `multiple='multiple'` umožní vyznačiť viacero možností naraz.

Ďalšie atribúty formulárových elementov

Rozmery textových polí

Ak by ste rozmýšlali ako spraviť dlhší input, aby ste videli viac textu, je k dispozícii atribút `size`. Defaultne je jeho hodnota 25 a určuje počet zobrazených znakov.

```
<input type="text" name="test" size="35"/>
```

Textarea a jej veľkosť sa určuje inak, má dva atribúty `cols` a `rows`. Pomocou nich sa určujú šírka a výška elementu v znakoch.

```
<textarea name="test" cols="50" rows="15"></textarea>
```

Cols a rows sú povinné atribúty.

Deaktivácia polí

Všetky vstupné polia a tlačidlá možno deaktivovať. Spravíme tak pomocou atribútu `disabled`.

```
<input type="text" name="test" disabled="disabled"/>
```

Pozor, hodnoty sa pri deaktivovaných poliach neodosielajú.

Read only

Obdobou deaktivácie polia je nastavenie read only. Hodnotu vo vnútri read only elementu nemožno meniť. Rozdielom oproti disabled vlastnosti je, že hodnota sa odosiela.

```
<input type="text" name="test" readonly="readonly"/>
```

Nové formulárové značky a atribúty

Pre formuláre HTML 5 poskytuje nové značky a atribúty:

<datalist>	Element s preddefinovanými hodnotami a automatickým dopĺňaním (podobné elementu <select>)

Takisto značka <input/> dostala nové možnosti v podobe týchto atribútov:

<input type="range" .../>	Posuvník pre výber čísla z daného intervalu.
<input type="search" .../>	Textové pole určené pre frázu na vyhladanie.
<input type="color" .../>	Výber farby pomocou systémového popup okna.
<input type="date" .../>	Výber dátumu pomocou datepicker-u.
<input type="time" .../>	Výber času.
<input type="number" .../>	Textové pole pre zadanie čísla.
<input type="email" .../>	Textové pole pre zadanie emailu.
<input type="tel" .../>	Textové pole pre zadanie telefónneho čísla.

Publikujeme web cez FTP

HTML súbory ktoré sme doteraz vytvorili, sú umiestnené na našom počítači - lokálne. To znamená, že nie sú prístupné zvonka. Ako sa dajú publikovať na internet?

Na to, aby sme ich mohli sprístupniť svetu, budeme potrebovať webhosting a vedieť pracovať s FTP protokolom.

Hosting

V krátkosti si pripomenieme čo je to hosting. Hosting je služba, ktorá umožňuje prevádzku webstránok. Je to prenájom diskového priestoru a príbuzných služieb.

Hľadáme freehosting

Aby sme náš web mohli umiestniť von, budeme potrebovať hosting. Pre naše účely využijeme free hosting.

Free hosting je bezplatný, ponúka zadarmo doménu 3. úrovne, avšak zobrazuje reklamy. Pre prvé pokusy nám to ale bude stačiť.

http://www.endora.cz

Po jednoduchej registrácii nám príde email s inštrukciami, ako sa pripojiť na server a ako naň nahrať súbory.

Tieto inštrukcie majú podobu prístupových práv. Príkladom môže byť:

Doména: http://tester1.8u.cz
Adresa administrace: http://webadmin.endora.cz
Hostitel FTP: srv8.endora.cz
Uživatelské jméno: tester1
Heslo: 123123

Pred tým, než sa pripojíme na server, objasníme vlastnosti FTP protokolu, cez ktorý sa na webhosting budeme pripájať.

FTP protokol

Čo je to FTP

FTP je skratka pre File Transfer Protokol. Je to protokol na prenos súborov medzi klientom (našim počítačom) a serverom.

Prvá špecifikácia vznikla v roku 1971. Odvtedy prešla mnohými revíziami a boli pridané mnohé bezpečnostné prvky.

Get a Put

Pri práci s FTP pracujeme s dvoma základnými typmi operácií:

- GET operácia, pri ktorej sťahujeme súbor zo serveru k nám na lokálny počítač - download.
- PUT operácia, pri ktorej posielame súbor na server - upload.

Nájdeme ho všade

FTP ako program je súčasťou všetkých známych operačných systémov - Windows, Mac OS aj Linux, hoci má podobu príkazového riadka a ide o konzolový nástroj.

Pre väčšie pohodlie existuje množstvo nástrojov s GUI - s grafickým používateľským rozhraním.

Ako písať texty pre web

Ako písať pre web. Rady z oblasti typografie, použiteľnosti, marketingu a SEO.

Ako ľudia čítajú

Pred tým, než si povieme, ako by sme mali písať texty pre webstránky, je dôležité vedieť, ako ich ľudia čítajú.

Najdôležitejší poznatok je, že ľudia nečítajú. Ľudia iba očami "skenujú" a hľadajú potrebné informácie. A to je potrebné zohľadniť pri písaní.

Ako písať aby si to všimli

Dodržiavaním nasledujúcich zásad zvýšite šance, že si čitateľ prečíta aspoň z každej časti stránky kúsok.

Štruktúra

Text musí mať výraznú štruktúru. Základom je správne používanie nadpisov h1, h2, h3. Nadpisy musia byť k veci a čo najkratšie.

Odstavce tiež štruktúrujte a rozdeľujte podľa myšlienok. 3-4 riadky pre jeden odstavec je maximum.

Zvýraznenie

Značky a nám umožňujú zvýrazniť dôležité slová. Netreba to však preháňať. Ak zvýrazníme mnoho slov, stratí sa v tom význam.

Zoznamy

Používanie číslovaného a nečíslovaného zoznamu (ol, ul) umožňuje tiež vyniknúť a ľudia si ich všimnú.

Krátko a k veci

Všeobecne platí, že kratšie bloky textu sú lepšie.

Typografia

Pomocou CSS môžete výrazne vylepšiť pôžitok z čítania. Obzvlášť pomáha zväčšiť výšku riadku (line-height) a dostatočné odstupy medzi blokmi textu (margin).

Postupy z oblasti žurnalistiky a marketingu

Obrátená pyramída

Tento výraz by mali poznať žurnalisti či novinári. Je dôverne známy z novín, "horúcich" správ.

Princíp spočíva v tom, že spomenieme vždy tie najpodstatnejšie informácie hneď na začiatku. V prvých dvoch vetách spomenime kto, čo, kedy, kde a prečo. A až následne sa rozoberajú menej podstatnejšie detaily.

Obrátená pyramída umožňuje čitateľovi prečítať si zopár viet a urobiť si komplexný obraz o téme. Následne môže preskočiť na ďalší článok a takto pokračovať až do konca. Určite poznáte ľudí, ktorí sú schopní prelistovať si noviny za 5 minút a pri tom získajú kompletný obraz o všetkom čo sa stalo.

5 otázok (Five Ws)

Ak píšete web hlavne preto, aby niečo predal - nech je to produkt, služba alebo čokoľvek, mali by ste poznať pravidlo piatich otázok.

Vraví sa, že človeka dokážete o niečom presvedčiť (v našom prípade, že chce váš produkt) tak, že dostane odpoveď na nasledujúcich 5 otázok (Five Ws):

1. kto (who)
2. čo (what)
3. kedy (when)
4. kde (where)
5. prečo (why)

Ak má vaša webstránka spĺňať účel, mala by nezáväzne na poradí, odpovedať na všetky tieto otázky.

AIDA

AIDA je skratka slov Attention, Interest, Desire, Action. Je to pojem z marketingu a účinne sa dá aplikovať do písania textov na web.

- **Attention** - upútať pozornosť
- **Interest** - vzbudiť záujem o ďalšie informácie a produkt
- **Desire** - vytvorenie túžby vlastniť produkt
- **Action** - výzva k akcii (napr. objednávkový formulár)

Čomu sa vyhnúť

- Vyhýbajte sa veľkým písmenam
- Vyhýbajte sa sekvenciám "!!!"
- Vyhýbajte sa superlatívom "Sme najlepší" a podobne. Čitateľ začne rozmýšľať nad pravdivosťou týchto tvrdení, a bude čítať pomalšie.

Responzívny dizajn

Čo je responzívny dizajn

Webstránky vytvorené pomocou responzívneho dizajnu prispôsobujú svoj layout na základe zobrazovacieho zariadenia.

Dnes je internetové browsovanie bežne prevádzkované na mnohých zariadeniach:

- desktop
- notebook/netbook
- tablet
- smartphone

Každé z rýchto zariadení má v súčastnosti dostatočný výkon na beh internetového prehliadača s pokročilým renderovacím jadrom. V jednej vlastnosti sa ale od seba líšia: **každé z nich má odlišnú veľkosť displeja**.

Pri čom desktop môže poskytnúť bez problémov displej s uhlopriečkou 27", veľkosť obrazovky na smartphonoch sa pohybuje okolo 4". Z tohto dôvodu sú v súčasnosti na tvorcov stránok kladené nároky, ktoré z tejto skutočnosti vyplývajú.

Kóderi majú k dispozícií spôsob ako tú istú stránku zobraziť inak, podľa veľkosti displeja. Cieľom je poskytnúť optimálnu úroveň čitateľnosti, prehľadnosti a použiteľnosti.

V nasledujúcom obrázku vidieť rovnakú stránku pri rôznej šírke zobrazovacieho zariadenia:

Responzívny dizajn sa niekedy označuje skratkou RWD (Responsive Web Design).

Evolúcia

RWD je ďalší krok v evolúcii webového dizajnu. Zásadnejšie zmeny v trende CSS môžeme zhrnúť v týchto etapách:

1. Tabuľkový layout	koniec 90-tych rokov	Zneužívanie značky <table> pre účely dizajnu. Dot-com boom produkoval tisíce takýchto stránok.
2. Beztabuľkový layout	približne od 2005	Začína sa šíriť osveta, že layout stránok má byť vytvorený výhradne pomocou CSS, tzv. tableless design.
3. Responzívny dizajn	2012 a 2013	V dôsledku čoraz väčšieho počtu mobilných zariadení je potrebné dizajn prispôsobovať pre odlišné veľkosti displejov.

Autor

Za autora RWD sa považuje **Ethan Marcotte**, ktorý v máji 2010 uverejnil článok, kde sa tento termín prvýkrát spomína a popisuje jeho princípy.

Pôvodný článok pojednávajúci o RWD nájdete na adrese:
http://alistapart.com/article/responsive-web-design

Hlavné aspekty RWD

Hlavné témy, ktoré sú dôležité pre správnu implementáciu responzívneho dizajnu sú:

- použitie metatagu viewport
- používanie CSS media query
- rozdiel medzi min-width a max-width
- responzívne obrázky
- responzívne videá
- testovanie responzívneho dizajnu

Metatag viewport

Meta značka viewport je veľmi dôležitá pre správne zobrazenie stránky v mobilných zariadeniach - v pomere 1:1.

Použitie nasledujúcej meta značky je predpokladom pre ďalšiu implementáciu responzívneho dizajnu.

Do hlavičky HTML dokumentu musíme vložiť tento zápis:

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

Nastavenie minimum-scale na hodnotu 1.0 zabezpečí, že sa stránka zobrazí v pomere 1:1, a teda naše vlastnosti určené pre responzívny dizajn sa nestratia (prehliadač sa nebude viac snažiť škálovať stránku pre menší displej).

Okrem toho, nastavenie width=device-width hovorí prehliadaču o tom, že šírka stránky je široká tak ako je široké zobrazovacie zariadenie.

Media queries

Pomocou media queries môžeme selektívne načítavať, alebo určovať CSS pravidlá pre rôzne zobrazovacie zariadenia.

Mnohým bude známy zápis media='screen' pri načítavaní externého css súboru:

```
<link rel='stylesheet' media='screen' href='./main.css'/>
```

Ďalším známym zápisom je media='print' na načítanie externého CSS súboru, ktorý bude použitý iba pri tlači:

```
<link rel='stylesheet' media='print' href='./print.css'/>
```

Nie každý však vie, že tento zápis má ďalšie možnosti zápisu, ktoré sa stali základom pre ideu responzívneho dizajnu.

Tak napríklad môžeme načítať CSS súbor určený pre 'screen', ale zároveň s rozlíšením najmenej 700px a najviac 900px:

```
<link rel='stylesheet' media='screen and (min-width: 700px) and (max-width:
900px)' href='css/medium.css' />
```

Takéto selektívne načítavanie CSS pravidiel sa dá uskutočniť aj priamo vo vnútri CSS súboru. Zapis vyzerá nasledovne:

```
@media screen {
  body {
    width: 75%;
  }
}

@media print {
  body {
    width: 100%;
  }
}
```

Samozrejme môžeme použiť zložitejší zápis, podobný tomu v príklade s načítaním externých CSS súborov:

```
@media screen and (min-width: 700px) and (max-width: 900px) {
  h1 {
    color: red;
  }
}
```

Takto definujeme červenú farbu nadpisu H1 iba pre 'screen' s rozlíšením najmenej 700px a najviac 900px.
Všimnime si kľúčové slovo "and", ktoré spája podmienky. Ak chceme použiť "or", musíme uviesť "," (čiarku). Dostupné je tiež kľúčové slovo "not".

Demo s použitím min-width

Predstavme si nasledujúci HTML kód súboru index.html, ktorý obsahuje jednoduchý nadpis a odstavec textu:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8"/>
    <link rel="stylesheet" href="style.css"/>
    <meta name="viewport" content="width=device-width, minimum-scale=1.0"/>
  </head>

  <body>
    <h1>Hi this is example of responsive design</h1>
    <p>Lorem ipsum...</p>
  </body>
</html>
```

Súbor index.html načítava CSS súbor, ktorý vyzerá nasledovne:

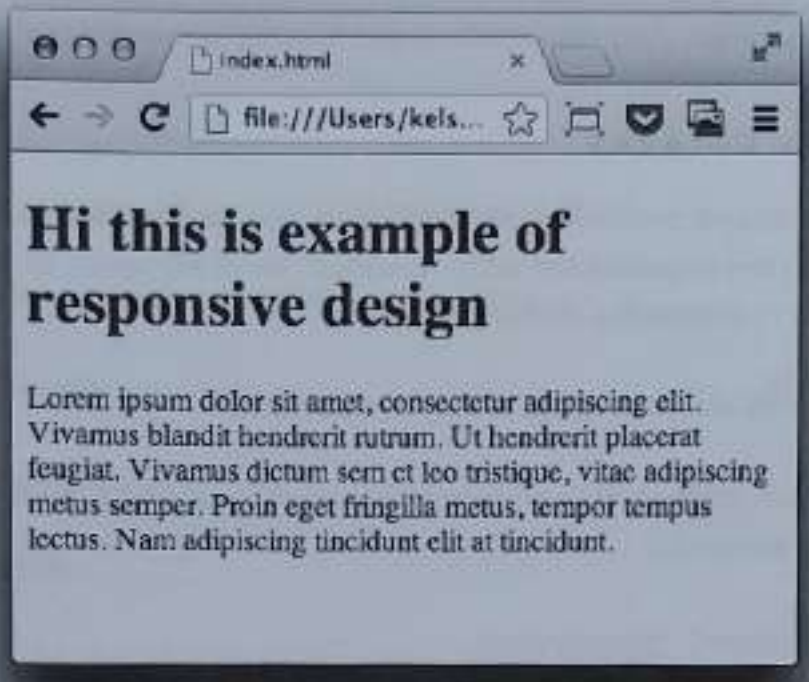
```
body {
  color: red;
}

@media (min-width: 480px) {
  body {
    color: blue;
  }
}
```

Tento súbor definuje dve veci:

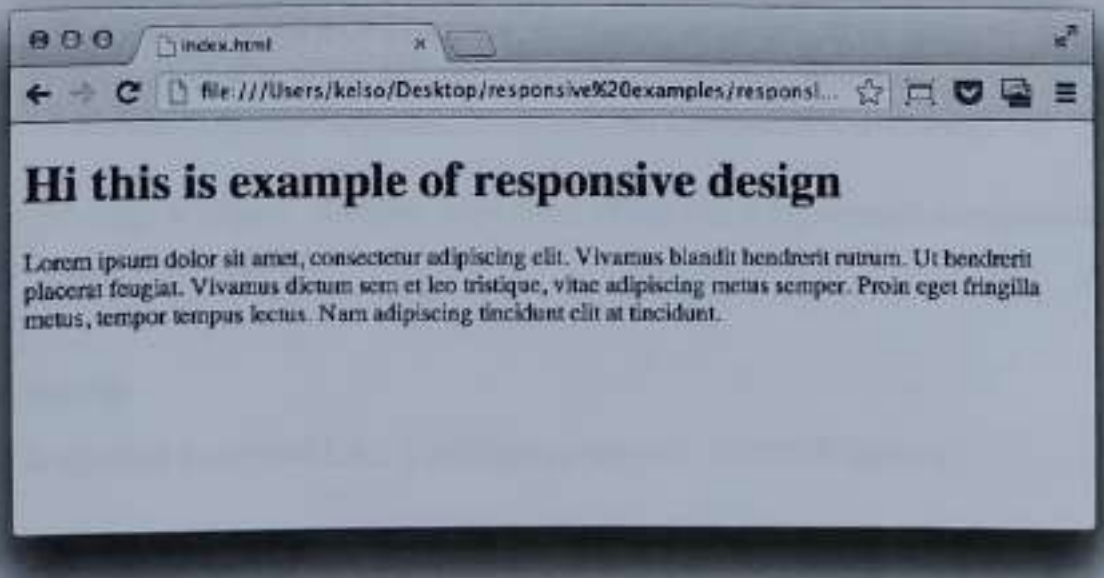
- farba písma v celom dokumente bude **červená**

- ak šírka okna (alebo šírka zobrazovacieho zariadenia) prekročí 480px, farba bude **modrá**

Pozrime sa, ako to vyzerá v praxi. V prvom prípade je okno menšie ako 480 px:



V druhom prípade je okno širšie ako 480px:



Týmto spôsobom sa nám otvárajú nekonečné možnosti, ako media queries využiť v prospech responzívneho dizajnu. Častým využitím je napríklad zmena šírky navigácie, alebo jej premiestnenie.

Niektoré menej podstatné informácie, môžeme vďaka media queries dokonca aj úplne skryť (napríklad na menších mobilných displejoch). Stačí nám na to jednoduché CSS pravidlo "display: none".

Responzívne obrázky

Pri responzívnom dizajne nesmieme zabudnúť na to, aby sa prispôsobovali šírke displeja aj obrázky. V opačnom prípade nám môžu "pretekať" von z elementu, ak je obrázok širší ako aktuálna šírka rodičovského elementu.

Existujú dva spôsoby ako zabezpečiť responzívne obrázky:

1. Poskytnúť **jeden** obrázok veľkých rozmerov a zmenšovať jeho šírku na základe rodičovského elementu.

   Výhoda: Jednoduchá implementácia.
   Nevýhoda: Klient sťahuje vždy ten istý súbor, čo môže byť v prípade malého displeja mrhanie prenášanými kilobajtami. Častokrát sa sťahuje obrázok veľkých rozmerov iba pre účel zobrazenia jeho zmenšenej podoby.

2. Poskytovať **niekoľko** verzií súborov obrázkov a na základe šírky elementu vyberať najvhodnejšiu verziu súboru, s ohľadom na prenášané kilobajty.

   Výhoda: Prenáša sa čo najmenší objem dát, načítanie stránky je teda rýchlejšie.
   Nevýhoda: Zložitejšia implementácia, často si treba pomôcť JavaScript-om.

Twitter Bootstrap 3

O frameworku

Twitter Bootstrap je v súčasnosti jeden z najpopulárnejších CSS frameworkov.

Čo dokáže

Vďaka CSS frameworku sa nemusíme zaoberať vecami ako:

- zarovnanie nosných elementov (alebo grid systém)
- základná typografia
- vzhľad formulárov
- responzívny dizajn

Všetky tieto a mnoho ďalších úloh za nás vyrieši Twitter Bootstrap.

Pre koho je určený

Používajú ho profesionáli ale takisto aj začiatočníci.

Profesionáli sa nemusia zdržovať CSS kódom, ktorý je často spoločný pre väčšinu stránok. Môžu tak svoj čas venovať hlavne veciam, ktoré sú pre danú stránku špecifické.

Začiatočníci, ktorí si ešte neosvojili pokročilé techniky CSS, môžu vďaka frameworku siahnuť po takmer hotových riešeniach - stavebných kameňoch webstránky.

Twitter bootstrap je určený všetkým, ktorí chcú ušetriť čas a zabezpečiť kompatibilitu vzhľadu medzi prehliadačmi.

Inštalácia

Inštalácia spočíva v načítaní CSS a JavaScript súborov do HTML stránky.

Najrýchlejším spôsobom ako si Twitter Bootstrap vyskúšať, je použitie CDN serverov. Potrebné súbory môžeme načítať priamo z ich serverov.

CSS súbor:

```
<link rel="stylesheet" href="http://netdna.bootstrapcdn.com/bootstrap/3.0.0/css/
bootstrap.min.css">
```

JS súbor:

```
<script src="http://netdna.bootstrapcdn.com/bootstrap/3.0.0/js/
bootstrap.min.js"></script>
```

Pre ďalšie použitie sa odporúča stiahnuť si súbory k sebe. Načítanie zo vzdialeného servera môže spôsobovať oneskorenia v načítaniach stránok.

CSS Vlastnosti

Tlačidlá

Začnime s niečím jednoduchým: tlačidlami. Pre vytvorenie tlačidla stačí kedykoľvek použiť CSS triedu "btn btn-default" a dostaneme naštýlované tlačidlo.


```
<a href="#" class="btn btn-default">Link</a>
```

Celkovo máme v k dispozícii varianty: default, primary, success, info, warning a danger.

Jednotlivé označenia v prvom rade reprezentujú účel, nie farbu. Napríklad "primary" slúži na označenie toho najdôležitejšieho tlačidla (v prípade, že sa ich na jednej stránke vyskytuje viacero). "Danger" predstavuje potencionálne nebezpečnú akciu. Používa sa na tlačidlá ako "Odstrániť", "Daktivovať" a pod.

Obrázky

Pre obrázky máme k dispozícii tri varianty:

- "img-rounded" pre zaoblené rohy
- "img-circle" pre kruh
- "img-thumbnail" pre náhľad fotky


```
<img src="..." alt="..." class="img-rounded">
```

Panely

Ďalším z prvkov, ktoré nás ušetria od písania CSS sú panely. K dispozícii máme štandardný "default" a takisto kontextové variácie (prvé tri z nich sú na obrázku).

Panel title

Panel content

Panel title

Panel content

Panel title

Panel content

Farebné odlíšenia opäť reprezentujú účel, nie samotnú farbu.

Progress bar

Bootstrap 3 poskytuje dokonca komponent na zobrazenie stavu priebehu udalosti - progress bar.

Použitie sa skladá zo súborov niekoľkých značiek, css tried a dátových atribútov:

```
<div class="progress">
  <div class="progress-bar" role="progressbar" aria-valuenow="60" aria-
```

```
valuemin="0" aria-valuemax="100" style="width: 60%;">
    <span class="sr-only">60% Complete</span>
  </div>
</div>
```

Vo výsledku dostane naštýlovaný progress bar:

V ponuke sú takisto všetky kontextové varianty:

Grid systém

Twitter Bootstrap 3 obsahuje funkcie pre grid systém, ktorý má priamo zabudovanú podporu pre responzívny dizajn.

Framework implementuje všetky druhy displejov a používa pri tom tieto označenia:

Označenie	Názov	Zariadenie a ich šírky
-	Extra small devices	smartfóny do 480px
sm	Small devices	tablety do 768px
md	Medium devices	počítače do 992px
lg	Large devices	veľké počítače, 1200px a viac

Grid pozostáva vždy z 12-tich stĺpcov.

Ak by sme chceli vytvoriť dvojstĺpcový layout, stačí použiť takýto HTML kód:

```
<div class="row">
  <div class="col-md-6">A</div>
  <div class="col-md-6">B</div>
</div>
```

Vytvorili sme rodičovský prvok div s css triedou "row" (riadok), do ktorého sme umiestnili div elementy predstavujúce stĺpce v danom riadku. CSS triedy stĺpcov predstavujú ich šírky.

Ukážme si teraz príklad responzívneho správania týchto stĺpcov.

Predchádzajúci kód vieme upraviť tak, aby sa síce zobrazoval v dvoch stĺpcoch, no na veľkých displejoch sa zobrazí až v troch stĺpcoch. Jediné čo potrebujeme spraviť je doplniť číslelné pomery s ďalším prefixom (lg):

```
<div class="row">
  <div class="col-md-6 col-md-4">A</div>
  <div class="col-md-6 col-md-4">B</div>
  <div class="col-md-6 col-md-4">C</div>
</div>
```

Správanie responzívneho dizajnu si môžeme jednoducho overiť zmenšovaním a zväčšovaním okna prehliadača. Pri vývoji je však dobré mať k dispozícii aj skutočné zariadenia s menšou uhlopriečkou displeja - teda smartphone a tablet.

Vysvetlenie množstva ďalších vlastností a komponentov frameworku Twitter Bootstrap sa nachádza v zrozumiteľnej oficiálnej príručke na http://getbootstrap.com.