Martin Domes

Tvorba WWW stránek pro úplné začátečníky

Názorná příručka krok za krokem
Od základů HTML a CSS po hotové stránky
Praktická cvičení, osvědčené postupy
Tipy a řešení nejčastějších problémů

CPRESS

Martin Domes

Tvorba WWW stránek pro úplné začátečníky

Computer Press
Brno
2012

Tvorba WWW stránek pro úplné začátečníky

Martin Domes

Obálka: Martin Sodomka
Odpovědný redaktor: Hana Kostovičová
Technický redaktor: Jiří Matoušek

Objednávky knih:
http://knihy.cpress.cz
www.albatrosmedia.cz
eshop@albatrosmedia.cz
bezplatná linka 800 555 513
ISBN 978-80-251-2160-3

Vydalo nakladatelství Computer Press v Brně roku 2012 ve společnosti Albatros Media a. s. se sídlem Na Pankráci 30, Praha 4. Číslo publikace 16 022.

Dotisk 1. vydání.

ALBATROS MEDIA a.s.

Obsah

Úvodem

Kdo se dnes neprezentuje na Internetu, jako by snad ani neexistoval. Vytvořit si vlastní stránky přitom není nic složitého. Stačí k tomu trocha cviku. Když chcete vytvořit osobní či firemní prezentaci, blog, fotogalerii nebo jiné WWW stránky, ale nevíte jak nato, tato kniha je ta pravá.

Tvorba WWW stránek je velkou zábavou a stejně tak k tomu přistupuje i tato kniha. Snaží se být praktická a zároveň se nezabývat tím, co čtenář nutně nepotřebuje. Výklad postupuje od úplných základů a s každou kapitolou se budete postupně zdokonalovat. Začnete texty na stránce a skončíte hotovou stránkou – včetně jejího grafického ztvárnění.

Všechna cvičení a příklady popisované v knize jsou dostupné ke stažení v přehledné podobě. Jak tyto příklady používat, popisuje podkapitola *Cvičení a příklady ke knize*.

Přeji vám hodně zábavy při tvorbě WWW stránek a spoustu spokojených uživatelů.

Komu je kniha určena

- *Pokud jste WWW stránky nikdy netvořili a nevíte, co to jsou jazyky HTML a CSS,* a rádi byste podnikli první krůčky, tato publikace vás jimi nenásilnou a trpělivou formou provede. Naučíte se vše od úplných základů, jako je tvorba textů a odkazů či vkládání obrázků do stránky.
- *WWW stránky jste zatím tvořili jen v různých editorech (např. Dreamweaver či FrontPage)*, ale láká vás naučit se používat značkovací jazyk HTML a kaskádové styly (CSS). V knize se mimo jiné na praktických ukázkách dozvíte, jak tyto jazyky fungují a jak se jejich kód tvoří, abyste dosáhli kýženého výsledku či efektu.
- *WWW stránky jste již dříve tvořili pomocí značkovacího jazyka HTML*, ale chcete se naučit tvořit stránky moderním způsobem pomocí kaskádových stylů (CSS). Naučíte se vše od úplných základů CSS – jak fungují a jak prvky tohoto jazyka používat, abyste vytvořili vzhled celé WWW stránky.
- *Znáte základy HTML i CSS a víte, jak vytvořit WWW stránku*, ale hledáte praktickou učebnici, díky níž své vědomosti zdokonalíte. V knize naleznete řadu v praxi využitelných návodů a postupů, díky nimž vytvoříte jak části stránek, tak celé layouty nebo třeba různé typy navigační nabídky.

S knihou se přístupnou formou naučíte všemu, co je třeba znát, abyste mohli sami vytvořit svoje vlastní WWW stránky od prvních textů po konečné grafické zpracování jednotlivých částí i celé stránky.

Forma výkladu

Kniha popisuje vše podstatné a důležité bez zabíhání do složitých detailů. Jednotlivé návody postupují krok za krokem a jsou doplněny informacemi o možných úskalích a jejich okamžitém řešení. To vše doprovází vizuální propojení textu a obrázků.

Výklad nevyžaduje žádné předchozí znalosti. Postupuje od úplných základů a informace v každé kapitole doplňuje. Všechny postupy si čtenář vyzkouší v praktických cvičeních.

Kniha používá několik speciálních odstavců, které doplňují text dalšími informacemi:

Poznámka – jde o rozšiřující informaci na okraj.

Tip – doplňuje výklad o zajímavou a prospěšnou informaci.

Důležité – informace, kterou byste si měli zapamatovat.

Řešení problému – v případě, že se v nějakém postupu může vyskytnout problém, informace v tomto odstavci vysvětlí, jak jej řešit.

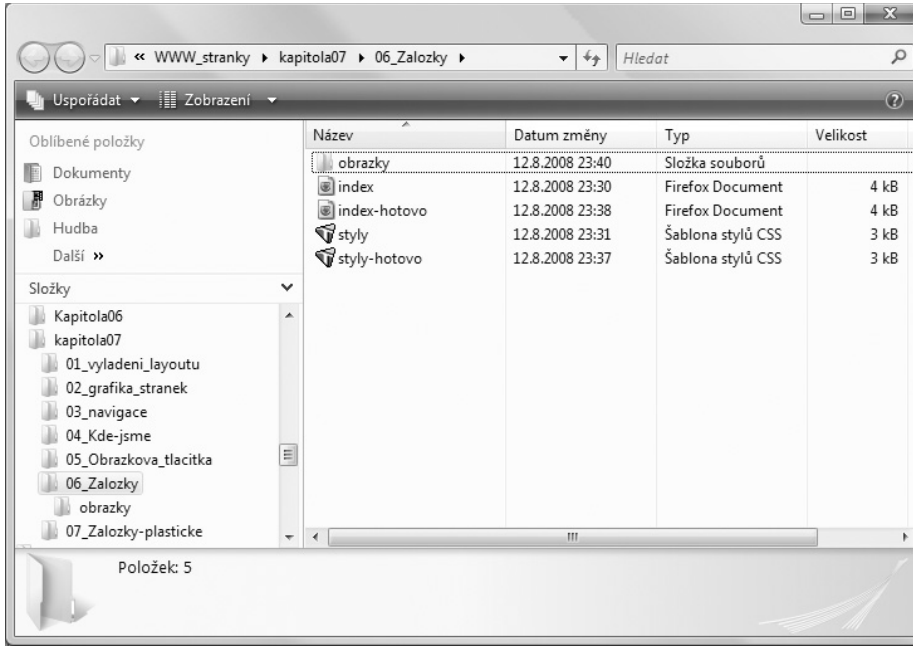
Cvičení a příklady ke knize

Celý balík všech cvičení z knihy si čtenáři mohou stáhnout z adresy *http://knihy.cpress.cz/k1622* – najdete je na záložce **Soubory ke stažení**. Celý balík kódů stáhněte a uložte do počítače. Následně balík rozbalte pomocí některého archivačního programu (např. WinRAR – viz *www.winrar.cz*).

Po rozbalení se na disku vytvoří složka *WWW_stranky*, v níž naleznete soubory cvičení uložené v jednotlivých složkách podle kapitol. K přehledu a zkoušení cvičení slouží přehledné rozhraní (obrázek Ú.1), které spustíte poklepáním na soubor *spustit.html* v hlavní složce. Dále se řiďte pokyny rozhraní.

Některá cvičení v kapitole 7 počítají s tím, že je čtenář bude provádět spolu s autorem knihy, a proto poskytují startovací dokumenty. V takovém případě ve složce najdete jak startovací, tak finální dokument. Finální dokument má v názvu slovo *hotovo* – například *index-hotovo.html* (obrázek Ú.2).

Obrázek Ú.1. Rozhraní ke cvičením



Obrázek Ú.2. Některé složky obsahují jak startovací, tak finální dokumenty

Kapitola 1
Než se pustíme do práce

Než se začneme zabývat samotnou tvorbou webových stránek od úplných základů, je třeba se na chvíli zastavit a najít odpovědi na několik základních otázek. Jestliže jsme úplnými začátečníky, získáme tak základní povědomí o tom, jak se vlastně tvoří webové stránky a co se skrývá pod jejich povrchem. Určitě nás také bude zajímat, v jakém programu internetové stránky vytvářet. A internetové stránky se určitě vytváří k tomu, aby se prohlížely – a k prohlížení je nutný prohlížeč internetových stránek. Jenomže ten neexistuje jen jeden, ale trh s prohlížeči je poměrně široký.

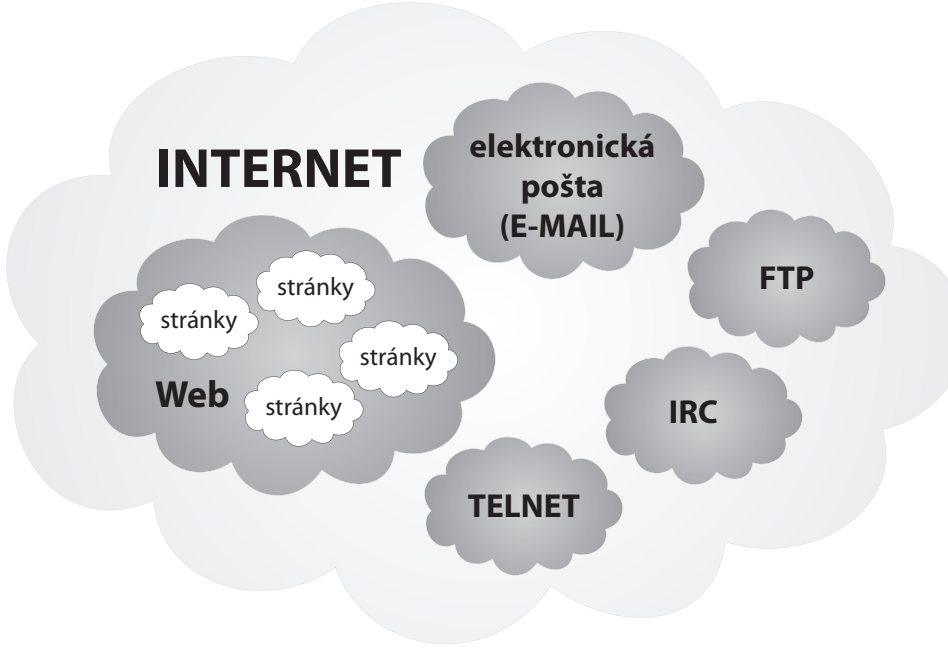
Proč se vlastně prohlížeči zabývat? Napsat internetové stránky pro jeden prohlížeč není příliš složité, ale vytvořit je tak, aby se shodně zobrazovaly ve všech hlavních prohlížečích, je už výzva, které se musí každý správný tvůrce WWW stránek (říkáme webdesignér) postavit čelem. Jak je to s prohlížeči si ale povíme až později.

Internet, web, WWW – co to vlastně je?

Ještě než uděláme krok dále, bude dobré se zastavit u pojmů, jichž budeme v této knize používat. A protože v nich uživatelé, a nejen uživatelé, často chybují, my si je vysvětlíte, abychom si dobře rozuměli. Nečekejte ovšem žádnou nudnou historii Internetu. Vše pěkně stručně a jasně bez zabíhání do detailů:

- Pod pojmem **Internet** si možná nic fyzického nepředstavíme, prostě zapneme počítač a ono to tam je. Internet je ovšem hmotná technologie. Jde o celosvětovou síť propojených počítačů, jež spolu mohou vzájemně komunikovat. Náš počítač se k síti Internet připojuje přes poskytovatele internetového připojení, respektive přes jeho přístupový bod. Internet je tedy technologie, jež umožňuje propojit počítače mezi sebou.
- **Web** je jedna ze služeb Internetu, díky níž můžeme na svých monitorech vidět textový a obrazový obsah Internetu. Tento obsah je uložen ve speciálních počítačích, tzv. *serverech*, k nimž se připojujeme ve svém prohlížeči pomocí *WWW adresy* (např. *www.cpress.cz*). A proč WWW?
- **WWW** je internetový protokol, jenž našemu počítači říká, že budeme využívat webovou službu Internetu. WWW je totiž jednoduše zkratka *World Wide Web* (česky celosvětová síť). Díky adrese WWW budeme schopni zobrazit dokumenty uložené

na Internetu (rozuměj na serverech) – pokud jde o text kombinovaný s obrázky, říkáme jim stránky. Je to totiž stejné jako v knize – Internetem můžeme také listovat podobně jako v knize.



Obrázek 1.1. Co se skrývá v Internetu

A co bylo cílem tohoto výkladu? Jde o to, že tyto pojmy se často vzájemně zaměňují. My se budeme zabývat internetovými stránkami neboli webovými stránkami neboli WWW stránkami – všechna tato slovní spojení mají stejný význam, a i když je zaměňujeme, máme na mysli stále totéž. Více podobně zaměřených a vzájemně pomocí jedné navigační nabídky provázaných stránek tvoří tzv. *webové sídlo*, které má konkrétní *WWW adresu* (např. *www.cpress.cz*). Tomuto webovému sídlu zkráceně říkáme web. Pokud tedy mluvíme o webu, nemíníme tím službu Internetu, díky níž se zobrazují jednotlivé stránky, ale mluvíme o webových stránkách jednoho člověka, jedné firmy nebo zpravodajském portálu.

Vypadá to složitě? Určitě ne. V této knize se budeme věnovat tvorbě WWW stránek, které budou dohromady tvořit onen web – například naše osobní stránky, stránky klubu nebo firemní stránky. Dozvíte se zde všechny nutné základy.

Jak se tvoří internetové stránky

Když máme jasno, o čem se budeme v této knize bavit, napadne vás jistě otázka, jak se takové internetové stránky tvoří. Pokud máte představu, že budeme jen klepat myší a sklá-

dat stránky jako skládačku, pak jste na špatné adrese. Tato kniha se nevěnuje ničemu takovému, ale odhalí vám, jak prakticky tvořit internetové stránky psaním kódu, díky němuž se samotné stránky mohou zobrazit, říkáme vykreslit, v prohlížeči.

Pokud nám je povědomé slovo programování, máme základní představu o tom, čemu se budeme v této knize věnovat. Jenomže o programování v pravém slova smyslu nepůjde. Jak to? WWW stránky se totiž v základu vytváří tzv. *značkováním*.

Nepředstavujme si ovšem nic záludného, opravdu budeme psát kód stránek tak, že budeme psát *značky* (anglicky *tag*). Značkování se ve skutečnosti neříká, webdesignéři totiž používají mnohem vznešenější výraz – *kódování*. Zní to lépe, že?

Pomocí značek řekneme prohlížeči: tady bude odstavec a tady nadpis. To samotné by ale nestačilo. Prohlížeči také musíme říct, jaký typ písma má použít k jeho vykreslení, jakou velikost, barvu a tak dále. K tomu už nebudeme používat značky, ale tzv. *styly*. Styly pak definujeme pomocí *šablony*, v níž napíšeme, že odstavec má mít takovou barvu, takové písmo a takovou velikost textu. Vytvoříme tak styl pro daný odstavec.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="cs" lang="cs">
<head>
  <meta http-equiv="content-type" content="text/html; charset=windows-1250" />
  <title>Upravujeme rámečky</title>
</head>
<body>
<table>
  <tr>
    <th class="doleva bily-ramecek">Jméno</th>
    <th class="bily-ramecek">Funkce</th>
    <th>E-mail</th>
  </tr>
  <tr class="zmena-pozadi">
    <td>Petr Novák</td>
    <td class="nastred">generální ředitel</td>
    <td class="nastred">novak@nase-spolecnost.cz</td>
  </tr>
  <tr>
    <td>Jarmila Gabrielová</td>
    <td class="nastred">provozní vedoucí</td>
    <td class="nastred">gabrielova@nase-spolecnost.cz</td>
  </tr>
  <tr class="zmena-pozadi">
    <td>Ingrid Brouzdalová</td>
    <td class="nastred">hlavní účetní</td>
    <td class="nastred">brouzdalova@nase-spolecnost.cz</td>
  </tr>
</table>
</body>
</html>
```

```
body {
  font-family: Arial, Verdana, sans-serif;
  font-size: small;
  width: 600px;
  margin: 10px auto;
}
table, td, th {
  border: 1px solid #005496;
  border-collapse: collapse; /* zhroucený model rámečku */
}
table {
  width: 100%;
}
td, th {
  padding: 3px 5px;
}
th {
  background: #005496;
  color: #FFF;
}
.nastred {
  text-align: center;
}
.doleva {
  text-align: left;
}
.zmena-pozadi {
  background: #d1d1d1;
}
.bily-ramecek {
  border-right-color: #FFF;
```

Obrázek 1.2. Vlevo kód se značkami tvořící WWW stránky, vpravo šablona se styly pro tyto značky

A to je celé? Ano i ne. Značky a styly jsou základní kameny tvorby WWW stránek. K tomu všemu ale budeme potřebovat ještě grafický editor, v němž si třeba připravíme obrázky, fotografie a další grafiku, kterou budeme chtít v našich WWW stránkách použít. Toho se v knize též dotkneme, ale nyní tuto záležitost odsuňme stranou.

Co to je WWW stránka

Nadpis této části podkapitoly je vlastně také otázkou. Víme, že máme nějaké značky a někam zapisujeme nějaké styly a díky nim se pak v prohlížeči zobrazí WWW stránka. Ale co je to ta samotná stránka a kam se zapisují tyto značky a styly? To je ona otázka a zde je odpověď.

WWW stránka není nic jiného než obyčejný textový dokument. Ovšem nikoliv textový dokument Wordu, ten totiž text formátuje, ale textový dokument s čistým neformátovaným textem (bez tučností, kurzivy či různých velikostí písma). Takový čistý textový dokument můžeme vytvořit třeba v Poznámkovém bloku, jenž je součástí Windows.

Shrnuto a podtrženo: WWW stránka je textový soubor, do něhož zapisujeme značky, popisujeme styly a vkládáme textový obsah stránky. Obrázky a další grafika jsou uloženy ve svých vlastních souborech (například fotografie) a z našeho textového souboru s kódem na ně pouze pomocí speciální značky odkazujeme (prohlížeči říkáme: na tomto místě zobraz tento soubor s obrázkem, který je uložený tady).

Co jsou jazyky HTML a XHTML

Od poměrně obecné definice toho, čím se tvoří WWW stránky, přejdeme ke konkrétním technologiím, jež ke stavbě stránek použijeme. Nejdříve jsme si řekli něco málo o značkách a o tom podivném značkování, jemuž vlastně říkáme kódování. Také jsme trochu naťukli, že budeme tvořit jakýsi kód se značkami, jímž řekneme prohlížeči, že tohle je odstavec a tohle nadpis. Tyto značky definuje tzv. *značkovací jazyk* – konkrétně HTML.

Poznámka: HTML je zkratkou pro Hypertext Markup Language, česky hypertextový značkovací jazyk. Je to jazyk, jenž se skládá ze značek a jehož smyslem je umožnit zobrazení textového a obrazového obsahu stránek a propojit jednotlivé stránky mezi sebou pomocí tzv. odkazů. O tom až dále.

Jazyk HTML byl vytvořen k tomu, aby dal obsahu WWW stránky (tedy jejím textům a dalším prvkům) smysl. Pomocí něj jednoduše označíme text, který má být nadpisem, běžným odstavcem nebo bodovým výčtem a podobně. Taktéž můžeme označit část textu v odstavci a říct mu: na tato slova kladu důraz.

Platí, že pomocí jazyka HTML vytváříme logickou (významovou) strukturu dokumentu – podobně jako v této knize máme nadpisy, odstavce textu, speciální odstavce a další textové a jiné prvky. HTML ovšem nevytváří vzhled textu, neříká prohlížeči nic o velikosti textu, typu písma či jeho barvě. Prohlížeči pomocí různých značek jazyka HTML pouze řekneme: tohle je nadpis a tohle odstavec.

```
<h1>Nadpis uzavřený do značek h1</h1>
<p>Text odstavce uzavřený do značek p.</p>
```

Obrázek 1.3. Značky jazyka (X)HTML dávají obsahu WWW stránky smysl

Pořád se věnujeme HTML, ale co onen jazyk XHTML? Co ten s tím má společného? Je to jednoduché, XHTML je moderním bratrem HTML. Vychází z něj a v mnohém ho zjednodušuje. Cílem XHTML bylo vytvořit jazyk vycházející z HTML, ale s přísnějšími pravidly. Nebojme se ale, tato pravidla jsou pro nás přínosem.

XHTML vznikl jako odpověď na praktiky webdesignérů prováděné s jazykem HTML. Začal se používat také k určení vzhledu částí stránky (pomocí značek se pak definoval tře-

ba typ písma či jeho velikost). Jak jsme si ale řekli, HTML nikdy k určení vzhledu čehokoliv sloužit neměl. Proto vznikl jazyk XHTML s přísnějšími pravidly a s cílem jasně vymezit rozdíl mezi strukturou WWW stránky (značkami) a jeho vzhledem (styly, jež se vytváří v jiném než značkovacím jazyce).

Poznámka: XHTML je zkratkou Extensible Hypertext Markup Language, česky rozšiřitelný hypertextový značkovací jazyk. Bere si to lepší z HTML (značky a podobně) a kombinuje to s přísnými pravidly značkovacího jazyka XML (Extensible Markup Language, česky rozšiřitelný značkovací jazyk; je jazykem určeným pro značkování dat k dalšímu, například databázovému, zpracování). XHTML je navržen tak, aby jej bylo možné použít pouze k dodání smyslu (významu) částem WWW stránky, nikoli k určení jejich vzhledu.

V této knize se budeme věnovat jazyku XHTML, ale protože v základu (rozuměj značkami) jsou tyto jazyky stejné, můžeme je bez problémů zaměňovat. Pokud tedy mluvíme o značkách jazyka HTML, mluvíme také o značkách jazyka XHTML. Pokud budeme v této knize hovořit o pravidlech a syntaxi zápisu značek, budeme vždy hovořit o pravidlech jazyka XHTML (mnoho z nich, především těch základních, ovšem bude platit beze změn i pro jazyk HTML). Nedělejme tedy nyní mezi oběma jazyky žádný rozdíl, protože pro nás zatím neexistuje.

Co umí a k čemu slouží jazyk CSS

Možná nám není zkratka CSS povědomá, ale je více než pravděpodobné, že jsme ve spojitosti s tvorbou WWW stránek zaznamenali slovní spojení *kaskádové styly*. Ano, kaskádové styly je český překlad zkratky CSS. Jazyk CSS není značkovacím jazykem, ale jazykem *stylovacím*. Jeho úkolem je dodat styl, tedy vzhled, částem WWW stránky, které jsme označili pomocí značek jazyka HTML (XHTML). Prohlížeči tak řekneme: obsahu této značky přiděl tento typ písma, tuto velikost a barvu textu. A prohlížeč to udělá, neboť tak jako rozumí jazyku HTML, tak rozumí i jazyku CSS.

Poznámka: CSS je zkratkou Cascading Style Sheets, česky šablony kaskádových stylů (zkráceně kaskádové styly). Byl vyvinut později než HTML s cílem vymýtit z jazyka HTML jakékoli prezentační, tedy vzhledové prvky. CSS samotný poskytuje vše podstatné pro určení vhledu kterékoli části WWW stránky.

Zastavme se jen velmi krátce u slova kaskádové v názvu kaskádových stylů. Ne, nebudeme si nyní vysvětlovat nic složitějšího než to, že toto slovo má velmi silný význam v rámci celého názvu. Jazyk CSS totiž pracuje kaskádovitě a dokáže jednotlivé vlastnosti (například typ písma) přelévat z jedné značky na další. Nyní nám bude stačit vědět, že je kaskáda opravdu důležitá a že se jí budeme věnovat dále v knize ve chvíli, kdy na to přijde čas.

Podstatou tedy zůstává to, co jsme si řekli už u stručného popisu jazyka HTML. CSS slouží k přidělení vzhledu, zatímco jazyk (X)HTML slouží pouze k určení významu částí webové stránky. Dodržování tohoto striktního rozdělení vede k přehlednému kódu, s nímž je radost pracovat. Na jednom místě totiž budeme mít textový obsah stránky a na jiném, oddě-

leném, místě budeme mít seznam vzhledů (stylů) těchto částí stránky. Toho se budeme při tvorbě WWW stránek v této knize držet.

```
h1 {
  font-family: ""Times New Roman";
}
p {
  font-family: Arial;
}
```

Obrázek 1.4. Kaskádové styly dodávají vzhled obsahu WWW stránek označenému pomocí značek

Jak je to s internetovými prohlížeči

Jestliže se právě chystáte přeskočit tuto kapitolu, zadržte! Téma prohlížečů je v případě WWW stránek extrémně důležité a nelze ho podceňovat, a to ani (a zvláště tehdy) pokud jste úplnými začátečníky. Pokud bychom totiž vytvářeli WWW stránky pouze pro jeden prohlížeč, ať už by to byl například nejrozšířenější Internet Explorer, přijdeme, například u firemních stránek, pravděpodobně o část klientů s jiným prohlížečem. A věřte mi, je jich hodně.

Standardy, standardy a zase standardy

Neděsme se. Nebudeme si zde vykládat o žádných technických standardech, to by bylo příliš nudné. Jsou ovšem podstatou problému s prohlížeči. Jazyky HTML, XHTML a CSS vychází z obecně přijatých standardů, které definuje společnost W3C. V ní pracuje tým odborníků, kteří určují standardy, podle nichž mají jednotlivé jazyky chápat samotné prohlížeče. Je třeba říct, že tahle značka určuje odstavec a že vlastnost určující velikost textu se jmenuje takto. Tento hotový a přijatý standard pak převezmou programátoři webového prohlížeče a vloží jej do něj. Výsledkem je, že prohlížeč jazykům (X)HTML a CSS rozumí.

Poznámka: W3C je zkratkou Word Wide Web Consortium. Jde o společnost založenou v 90. letech minulého století za účelem standardizace prostředí Internetu. Jejím cílem je zajistit společný vývoj technologií Internetu a jejich správné přijetí internetovými prohlížeči. Proto společnost na vývoji významně spolupracuje také s tvůrci internetových prohlížečů.

Aktuálními platnými standardy jsou tyto verze jazyků:

- HTML 4.01
- XHTML 1.0
- CSS 2

Tip: Jestliže jste zdatní v angličtině, podrobné informace k jednotlivým specifikacím najdete na následujících adresách: HTML na *www.w3.org/TR/html401/*, XHTML na *www.w3.org/TR/xhtml1/* a CSS na *www.w3.org/TR/CSS2/*.

A v čem je tedy onen zakopaný pes? Problém je v tom, že ne každý prohlížeč přijaté standardy vykládá správně. Tento problém platí zvláště pro starší prohlížeče, bohužel mimo jiné také pro stále rozšířený Internet Explorer 6 (je součástí Windows XP). Každý prohlížeč na trhu tak implementuje (zahrnuje) specifikace zmíněných standardů trochu odlišně. Výsledkem pak může být jiné, nebo dokonce chybné vykreslení (zobrazení) WWW stránky v různých prohlížečích.

Jaké jsou podíly prohlížečů na trhu, zobrazuje následující tabulka:

prohlížeč	podíl
Internet Explorer 7	32 %
Internet Explorer 6	27 %
Internet Explorer 5.5 a 5	<1 %
Mozilla Firefox	34 %
Opera	5 %

Zdroj: Toplist, 2.6.2008

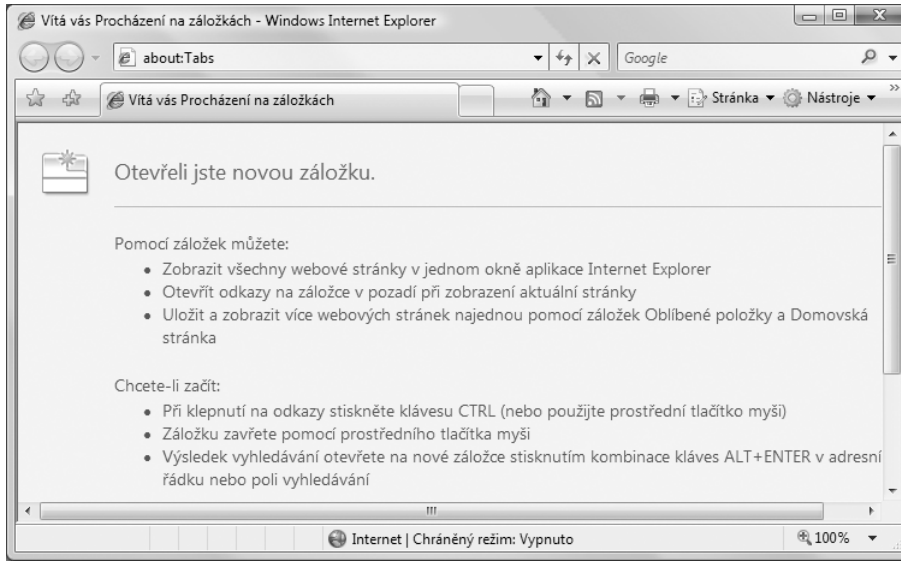
Podíl prohlížeče Internet Explorer je zhruba dvoutřetinový oproti třetinovému podílu Firefoxu. Světové statistiky udávají podíly méně dramatické: asi 73 % pro všechny verze Internet Exploreru a asi 19 % pro Firefox. Skutečný podíl Firefoxu bude tedy o několik procent nižší, ale to už není podstatné. Podívejme se na to, jak jednotlivé prohlížeče (a jejich verze) dodržují standardy.

Internet Explorer

Tento nejrozšířenější prohlížeč, jenž je součástí operačních systémů Windows, je dlouhodobě kritizován ze nedodržování standardů společnosti W3C. Problémem byla vždy především specifikace CSS 2. Chybově se v tomto ohledu chovají především pětkové verze prohlížeče, ale ani verze 6 se nemá čím chlubit. Spoustu chyb opravila až sedmá verze, která přišla s Windows Vista a kterou uživatelé Windows XP mohli získat s aktualizacemi nebo s třetím servisním balíčkem. Sedmá verze Internet Exploreru zobrazuje stránky dle standardu CSS obstojně.

Vzhledem k tomu, že je v současnosti stále velký počet uživatelů šesté verze, je třeba vytvářené stránky kontrolovat také v tomto prohlížeči. Tento prohlížeč však v žádném případě není vhodný jako hlavní vývojový prohlížeč, v němž budete vytvořené stránky průběžně kontrolovat na prvním místě. V mnoha ohledech se totiž jeho názor na CSS liší, byť drobně, od standardu společnosti W3C.

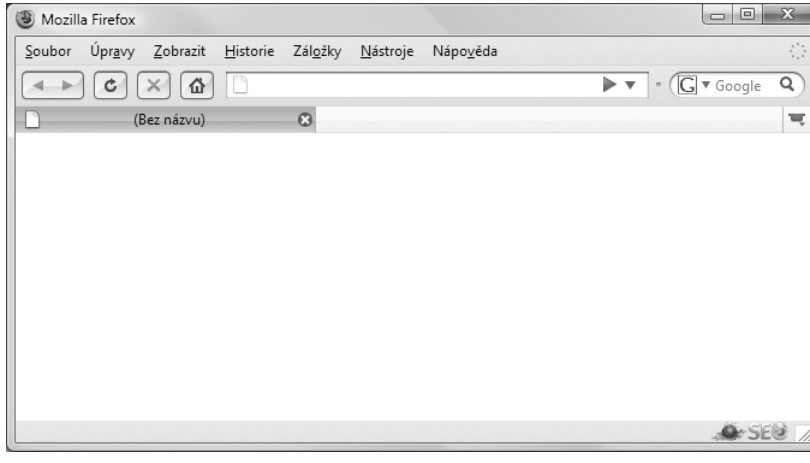
Tip: Pokud chcete mít na svém počítači více verzí Internet Exploreru, stáhněte si jednoduchý instalátor z adresy *www.tredosoft.com/Multiple_IE*. Pokud budete chtít na Windows Vista nainstalovat starší verzi Internet Exploreru, pak budete muset sáhnout po simulátoru virtuálního počítače, programu Virtual Server (ke stažení na adrese technet.microsoft.com/cs-cz/bb738033.aspx) od společnosti Microsoft a následně spustit virtuální počítač s Windows XP (obraz systému lze stáhnout na adrese *www.microsoft.com/downloads*, kde do vyhledávacího pole napište Internet Explorer Application Compatibility VPC Image). Jiným způsobem se k šesté verzi Internet Exploreru na Windows Vista nedostanete.



Obrázek 1.5. Prohlížeč Internet Explorer 7

Mozilla Firefox

Tento velmi oblíbený internetový prohlížeč podporuje CSS velmi dobře a striktně podle standardů. Prakticky platí, že pokud se vám stránky ve Firefoxu zobrazí, jak jste zamýšleli, pak jste vše provedli správně. Firefox je vhodný jako hlavní vývojový prohlížeč, takže při tvorbě stránek je průběžně prohlížejte právě v tomto prohlížeči.



Obrázek 1.6. Prohlížeč Mozilla Firefox 3 (s nestandardním vzhledem)

Firefox prošel vývojem několika verzí, nicméně od verze 1.6 obsahuje funkci automatické aktualizace, takže není třeba stránky prohlížet v jiné než aktuálně nejnovější stabilní ver-

zi, neboť je pravděpodobné, že u všech uživatelů v poměrně krátké době po vydání novější verze proběhne aktualizace.

V době psaní této publikace byla nejnovější verze 3.0.

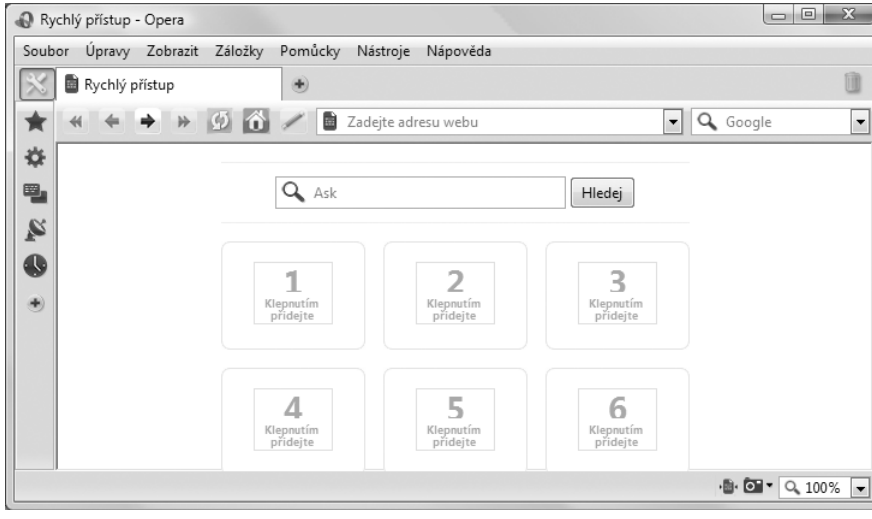
Tip: Prohlížeč Mozilla Firefox si můžete stáhnout na adrese firefox.czilla.cz. Protože se prohlížeč průběžně aktualizuje, není třeba jej přeinstalovávat.

Opera

Opera je prohlížečem s poměrně malým zastoupením, ale s o to věrnější základnou svých uživatelů. Je mu přisuzována vlastnost nejrychlejšího prohlížeče, což je možná uživatelsky zajímavé, ale z pozice webdesignéra to není nejpodstatnější. Podstatné je, že Opera je v přijímání webových standardů společnosti W3C ze všech prohlížečů nejdále. Specifikaci CSS 2 zvládá přesně a téměř bez jakýchkoli odchylek. Pokud tedy budete chtít Operu zvolit jako svůj hlavní prohlížeč, jen do toho. Problémem by mohla být snad jen menší základna uživatelů – proto obvykle dostane přednost Firefox.

V době psaní této publikace byla na světě verze 9.5. V době, kdy knihu čtete vy, už bude na světě jistě novější verze.

Tip: Prohlížeč Opera můžete stáhnout na adrese *www.opera.com/download/*. Českou podporu prohlížeči najdete na adrese *www.operacesky.net*.



Obrázek 1.7. Prohlížeč Opera 9.5

Další prohlížeče

Prohlížečů je samozřejmě podstatně více, než jsme si popsali, ale proč se jimi zatěžovat, když mají poměrně malý podíl. Zároveň platí, že další vyspělé prohlížeče často vychází

z jádra, které používá pro vykreslování WWW stránek například prohlížeč Firefox (jádro se nazývá Gecko). Pokud se tedy stránky zobrazí správně ve Firefoxu, znamená to, že se zobrazují správně i v podobných prohlížečích.

V této knize se dalším prohlížečům věnovat nebudeme, neboť to nebude pro naši práci podstatné. Vystačíme si se třemi u nás nejpoužívanějšími a vzájemně odlišnými prohlížeči.

V čem budeme tvořit WWW stránky

Jak už jsme si řekli dříve v této kapitole, tato kniha se věnuje tvorbě WWW stránek psaním kódu, nikoli skládáním částí grafiky. I to lze, ale k tomuto účelu jsou zde jiné knihy. My se budeme učit tvořit WWW stránky způsobem, jenž nám zajistí maximální kontrolu nad výsledkem našeho úsilí.

K tvorbě WWW stránek existují dva typy programů, jež nám dobře poslouží:

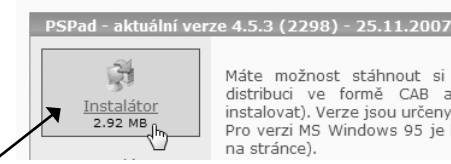
- *Editory WYSIWYG* – tato tajemná zkratka znamená *What You See IS What You Get*, česky *co vidíš, to dostaneš*. Takovéto editory fungují v grafickém módu a umožňují tvorbu WWW stránek jednoduše výběrem a stavěním prvků v grafickém rozhraní. Ihned tak vidíme, co se bude na webové stránce zobrazovat. Výhodou je, že nemusíme znát kódování, které jsme si popisovali. Nevyžaduje se tedy znalost jazyků HTML a CSS. Nevýhodou je, že tento kód na základě našich instrukcí tvoří onen editor. To ovšem znamená, že nad kódem máme jen velmi malou kontrolu, a proto tyto editory v této knize nedoporučuji. Mezi nejznámější editory tohoto typu patří Adobe Dreamweaver, Microsoft FrontPage a jeho novější bratr Microsoft Expression Web.
- *Strukturní editory* – to jsou vlastně obyčejné textové editory, v nichž píšeme kód, ale poskytují nástroje, díky nimž je tvorba kódu snadnější a automatizovaná. Jsou velmi vhodným doplňkem a velmi dobře je při tvorbě WWW stránek využijeme. Výsledek kódu si pak musíme zkontrolovat v internetovém prohlížeči. Jako strukturní editor může fungovat i Adobe DreamWeaver, který mimo grafického WYSIWYG režimu obsahuje také čistě strukturní režim, v němž vidíme a píšeme pouze kód. Nevýhodou je, že tento program je placený. V této knize si budeme doporučovat neplacený software. Nejlepším a ve světě známým českým editorem je PSPad. Zvolit ale můžeme samozřejmě jakýkoli jiný editor, který najdeme a stáhneme například na adrese *www.gigamania.cz*.

Výklad v této knize je naprosto nezávislý na jakémkoli editoru, který si zvolíme, je to tedy pouze na nás. WWW stránky lze zcela běžně tvořit také v obyčejném textovém editoru, jako je třeba Poznámkový blok, jenž je součástí Windows.

Důležité: Kód nikdy nepište v textových procesorech jako je Word, neboť soubor s WWW stránkou vyžaduje uložení jako čistý text bez formátování.

Instalace editoru PSPad

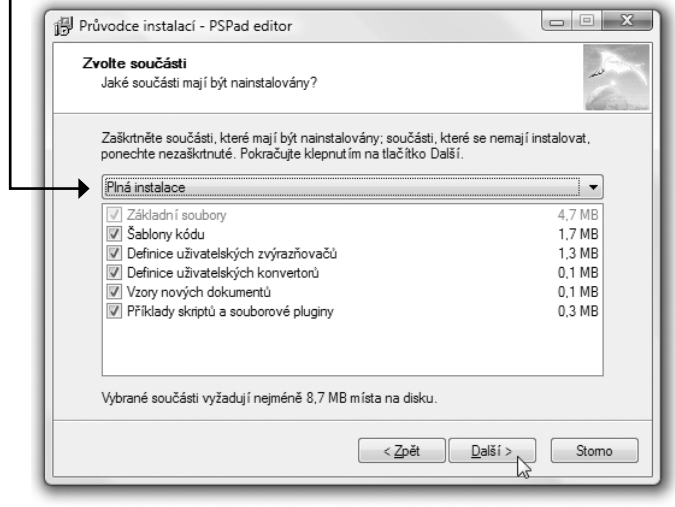
Podívejme se nyní na to, kde stáhnout PSPad a jak jej nainstalovat do svého počítače. Svou pouť v tomto případě začneme na adrese *www.pspad.cz*. Na stránce následujeme odkaz v levé navigační nabídce s názvem **Stažení PSPadu** a na načtené stránce klepneme na odkaz Instalátor, pomocí něhož stáhneme PSPad do svého počítače.



Obrázek 1.8. Stažení PSPadu

Po stažení spusťme samotnou instalaci:

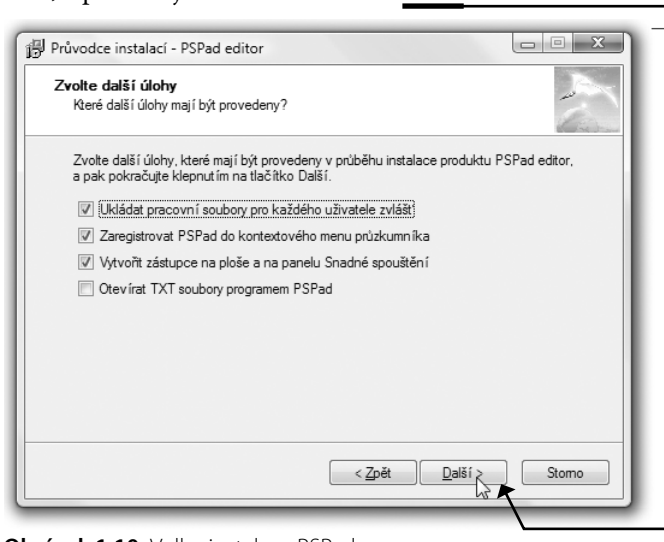
1. V prvním okně instalátoru klepněme na tlačítko **Další.**
2. Zatrhneme volbu, v níž souhlasíme s licenčními podmínkami, a pokračujeme pomocí tlačítka **Další**.
3. Pokud budeme chtít PSPad nainstalovat do specifického umístění, použijeme tlačítko **Procházet**. Instalovat je ale možné do výchozího adresáře. Pokračujeme stiskem tlačítka **Další.**
4. V dalším okně můžeme v roletce zvolit typ instalace. Lze nechat zvolenou plnou instalaci nebo pomocí volby **Uživatelská** postupně zatrhnout součásti PSPadu, které chceme nainstalovat. Pokračujeme tlačítkem **Další**.



Obrázek 1.9. Výběr součástí k instalaci PSPadu

5. V dalším okně zvolíme název, pod nímž se bude PSPad zobrazovat na ploše a v nabídce Start. Pokračujme tlačítkem **Další.**

6. V posledním okně před instalací je možné zatrhnutím či zrušením zatržení volit způsob fungování PSPadu. Ponechme volby ve výchozím nastavení, jak ukazuje obrázek 1.10, a pokračujme stiskem tlačítka **Další**.

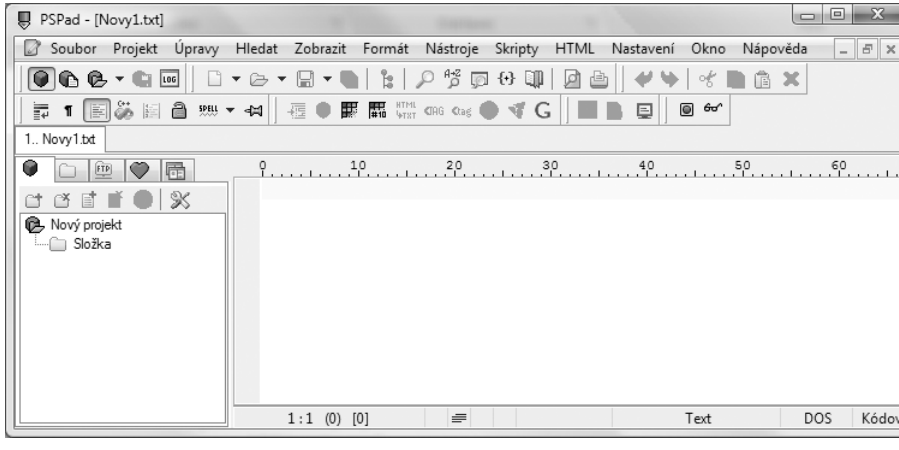


Obrázek 1.10. Volby instalace PSPadu

7. Editor se nainstaluje, jakmile stiskneme tlačítko **Instalovat**. Po instalaci stiskneme tlačítko **Dokončit**.

Prostředí PSPadu

Po dokončení instalace PSPad spusťme, abychom se seznámili s jeho prostředím a ovládáním. Jak zobrazuje obrázek 1.11, okno PSPadu se v základu neliší od jiných programů. Najdeme zde jak textové nabídky s nejrůznějšími základními i pokročilými nástroji, tak



Obrázek 1.11. Okno PSPadu

panel nástrojů s ikonami, jež nám nejdůležitější nástroje okamžitě zpřístupňují. Pokud všemu hned nerozumíme, není třeba propadat panice, neboť na všechno přijde čas. Co budeme v PSPadu či jiném editoru používat, je jen na nás. Tato kniha se nebude věnovat jeho ovládání, neboť věřím, že na to přijdete sami. Nicméně čas od času si ukážeme, jak tu kterou věc udělat jednodušeji pomocí nástrojů PSPadu.

Nejpodstatnější částí je samotné obsahové okno, nyní s bílou nepopsanou plochou. Právě sem budeme zapisovat kód, jímž budeme tvořit WWW stránku. Zkusme sem cokoliv napsat, abychom si program trochu osahali. Jak vidíme, do okna můžeme psát stejně jako v jakémkoli jiném textovém editoru.

V levém panelu nalezneme rychle přístupné záložky s následujícím obsahem:

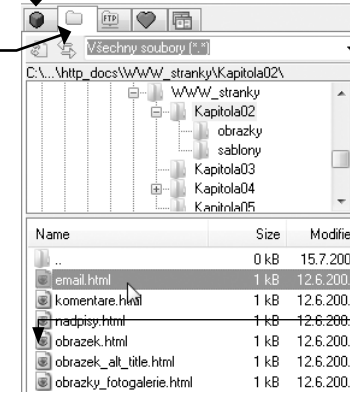
- První záložka zleva obsahuje naše projekty. Klepnutím pravým tlačítkem na **Nový projekt** a volbou **Nastavení projektu** v místní nabídce můžeme zvolit výchozí složku, do níž se budou všechny soubory zvoleného projektu ukládat. Tyto projekty ale využívat nemusíme, hodí se zvláště v případě, kdy vytváříme více webů.

Tip: Součástí balíku kódů je i celý projekt se všemi cvičeními k této knize v PSPadu. Lze ho otevřít jednoduše: v nabídce **Projekt** klepněte na příkaz **Otevřít projekt**, najděte složku s kódy a v hlavní složce otevřete projekt PSPadu, jehož soubor se jmenuje ***WWW stránky PÚZ.ppr.*** Kódy všech cvičení tak budete mít hned po ruce.

- Druhá záložka obsahuje adresářovou strukturu našeho počítače. Zde lze otevřít složku, do níž ukládáme soubory s našimi WWW stránkami. Tuto složku si můžeme vytvořit kdekoliv v počítači pomocí Průzkumníka Windows. Na záložce pak složku pouze nalistujeme a poklepáním na soubor WWW stránky jej otevřeme do PSPadu.

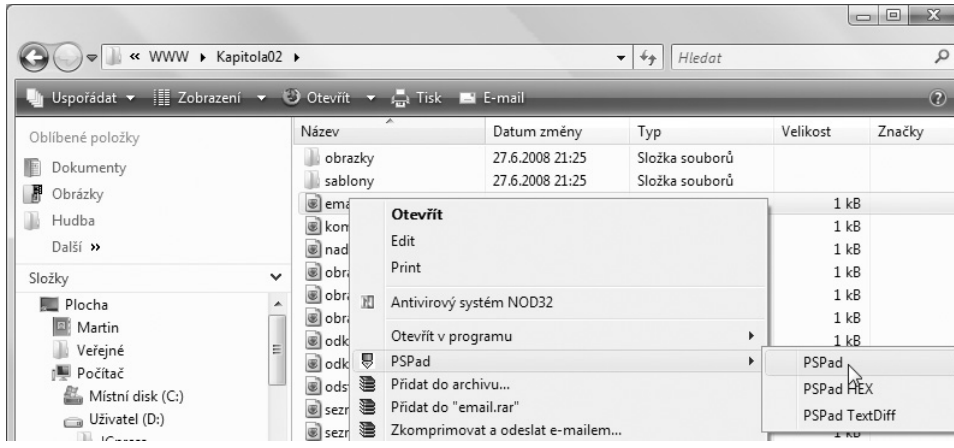
Vybraný dokument s WWW stránkou lze otevřít do PSPadu také tak, že v Průzkumníku Windows:

1. Pravým tlačítkem klepneme na soubor WWW stránky.
2. V místní pokračujeme na nabídku **PSPad.**
3. A zde klepneme na příkaz **PSPad**.



Obrázek 1.12. Rychlý přístup k jakékoli složce a souborům v levém panelu PSPadu

Řešení problému: Jestliže v místní nabídce souboru nemáte příkaz PSPad, pak jste při instalaci zrušili zatržení volby Zaregistrovat PSPad do kontextového menu průzkumníka. Nevadí, neboť to můžete napravit. V PSPadu otevřete nabídku **Nastavení** a zvolte příkaz **Nastavení programu**. V levé nabídce zvolte skupinu Integrace do systému a zatrhněte volbu Integrovat do kontextové nabídky průzkumníka.



Obrázek 1.13. Otevíráme soubor s WWW stránkou z Průzkumníka Windows do PSPadu

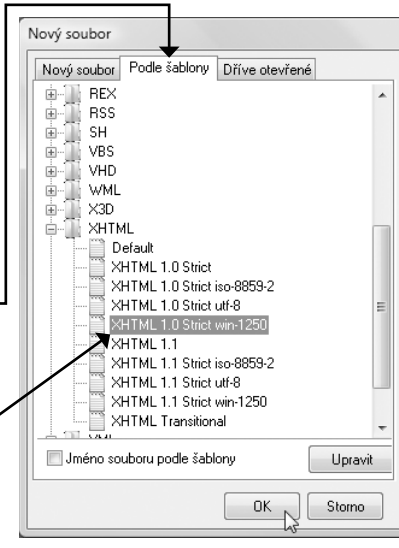
Jak sami brzo zjistíme, hlavní výhodou PSPadu je to, že dokáže barevně odlišit části kódu, a proto se v něm budeme snadněji orientovat. Kód sice můžeme psát v Poznámkovém bloku, ale tam o přehlednosti nemůže být řeč.

Práce s dokumenty v PSPadu

Protože je PSPad velmi vhodným pomocníkem při psaní kódu, je dobré jej využít také k zakládání nových dokumentů a k práci s více dokumenty v programu. Mezi nimi lze přepínat, vyhledávat a podobně.

Založení nového dokumentu s WWW stránkou:

1. V nabídce **Soubor** klepneme na příkaz Nový.
2. V dialogu **Nový soubor** zvolme záložku **Podle šablony**.
3. Najdeme kategorii **XHTML** a klepneme na ni.
4. Klepnutím vyberme šablonu **XHTML 1.0 Strict win-1250** a stiskneme **OK**.



Obrázek 1.14. Zakládáme novou WWW stránku ze šablony

Poznámka: Význam jednotlivých zkratek v šabloně je následující: XHTML 1.0 Strict je verze standardu a win-1250 je kódování znaků pro střední Evropu, tedy pro český jazyk (zároveň jde o kódování Windows).

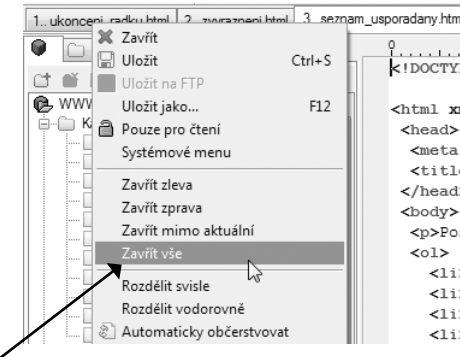
Tip: Nabídku šablon pro založení nového dokumentu můžete též rychle vyvolat, pokud poklepete na panel se záložkami.

Přepínání mezi otevřenými dokumenty. Nezabývejme se nyní tím, co se objeví v nově založeném dokumentu jazyka XHTML, ale zkusme si založit více nových dokumentů. Jak si můžeme všimnout, skládají se do záložek. Klepnutím na záložku se do dokumentu přepneme.

Uložení dokumentu. Jestliže budeme chtít práci v jakémkoli otevřeném dokumentu uložit, je nejrychlejší cestou klávesová zkratka Ctrl + S. Použít můžeme také ikonu diskety na panelu nástrojů.

Zavření dokumentu. Pokud budeme chtít jeden nebo více dokumentů zobrazených na záložkách zavřít, pak postupujeme takto:

1. Na záložku dokumentu klepneme pravým tlačítkem.
2. V místní nabídce zvolíme příkaz podle toho, co chceme provést:
   - **Zavřít** – zavře aktuálně otevřený dokument.
   - **Zavřít mimo aktuální** – zavře všechny dokumenty mimo aktuálně zobrazeného.
   - **Zavřít vše** – zavře všechny otevřené dokumenty.



Obrázek 1.15. Zavíráme otevřené dokumenty

Kapitola 2

Začínáme tvořit WWW stránku pomocí HTML

Začínáme, držme si klobouky. Během této kapitoly už budeme mít první výsledky své práce na monitoru. Pravda, nebude to možná nic moc, ale první WWW stránku vytvoříme se vším všudy, co má taková stránka mít. Především je třeba říct, že se nebudeme zabývat vzhledem textu na stránce ani jinými prvky stránky, ale provedeme pouze svoje první značkování, tedy označení různých textů na stránce pomocí značek, čímž jim přidělíme skutečný význam ve formě odstavců, nadpisů, odrážkových seznamů a podobně. A co víc – vytvoříme také odkazy, díky nimž propojíme více stránek dohromady, a také se naučíme do textu vkládat obrázky.

Začněme tím, že si spustíme PSPad nebo jiný editor, v němž založíme nový dokument pro naši WWW stránku. Zatím se nebudeme zabývat tím, co nám PSPad vytvořil za kód. Vzápětí si vysvětlíme, co to jsou značky jazyka (X)HTML a jak s nimi zacházet.

Poznámka: Založení dokumentu v PSPadu jsem popisoval na straně 28.

Co to jsou značky a kam vkládat obsah WWW stránky

Jakmile máme vytvořený dokument XHTML v PSPadu nebo jiném editoru, vysvětlíme si, co vlastně vidíme. Protože každý můžeme mít obsah dokumentu trochu odlišný, vysvětlíme si vše velmi univerzálně. Podrobněji o struktuře dokumentu jazyka (X)HTML až na straně 56.

Značky, kam se podíváš

Jak už jsme si v kapitole 1 pověděli, dokument jazyka (X)HTML se skládá především ze značek. Značka označuje část obsahu dokumentu, čímž mu dává význam. Říkáme tím prohlížeči například: toto je odstavec. Značka se zapisuje *párově*, existují tedy dvě varianty:

- *Značka otevírací* – říkáme prohlížeči, kde označená oblast začíná.
- *Značka uzavírací* – říkáme prohlížeči, že tady označená oblast končí.

Obecný zápis (místo slova značka bude třeba použít název značky) vypadá následovně:

```
<značka>
  Obsah značky
</značka>
```

Jak vidíme, značku uzavíráme do ostrých závorek (napíšeme ji pomocí kláves vpravo od klávesy s písmenem M), přičemž uzavírací značku doplňujeme lomítkem před jejím názvem. Vše, co je uzavřeno mezi obě varianty značky, je obsahem značky. Aby to nebylo tak jednoduché, existují také značky *nepárové*. Jejich zápis vypadá následovně:

```
<značka />
```

Nepárová značka se také zapisuje do ostrých závorek, ale za názvem značky uvnitř závorek následuje prázdná mezera jednoho znaku a lomítko. Jak si můžeme všimnout a jak nám asi došlo, nepárová značka žádný obsah nemá, neboť jej není kam uzavřít. Účel těchto značek je zkrátka jiný a poznáme ho dále v této kapitole.

Poznámka: Prázdnou mezeru za značkou nevyžadují nové prohlížeče, těm velmi starým by ale takový zápis dělal potíže, neboť neznají specifikaci jazyka XHTML. Přesto se ustálilo mezeru stále u nepárové značky zapisovat.

Vnořování značek a správný zápis

Řekněme, že budeme chtít pomocí značky označit odstavec textu, ale v tom odstavci dále budeme chtít označit ještě nějaké slovo, například proto, abychom ho zvýraznili oproti běžnému textu odstavce. Pak mluvíme o tzv. *vnořování* značek do sebe. Podívejme se, jak by to vypadalo:

```
<značka1>
Veškerý tento text je obsahem značky1, ale do ní je možné vnořit další značku.
<značka2>Tento text</značka2> je obsahem jak značky1, tak značky2.
</značka1>
```

Jak je vidět na uvedeném obecném příkladu, `značka2` je vnořená do `značky1` a obsah `značky2` je také obsahem `značky1`. I když to vypadá banálně, ve vnořování můžeme velmi snadno udělat chybu, a to když první značku uzavřeme dříve než vnořenou druhou značku. Vypadalo by to takto:

```
<značka1>Tento text patří do značky 1, <značka2>ale také do značky2. </značka1>
A tento text by měl patřit už jen značce2, ale ve skutečnosti to je chyba.</značka2>
```

Výše uvedený příklad prakticky ukazuje tzv. *křížení značek*. Takovýto zápis je zcela chybný a pravděpodobně povede ke zcela špatnému vykreslení webové stránky v prohlížeči. Samozřejmě záleží na okolnostech. Pamatujme si tedy, že značky je třeba uzavírat v opačném pořadí, než je otevíráme:

otevření značky1 → otevření značky2 → uzavření značky2 → uzavření značky1

Kam vkládat obsah WWW stránky

Dříve než se pustíme do praktické tvorby WWW stránky, musíme si povědět o tom, kam textový a další obsah, který se zobrazí v prohlížeči, vlastně v dokumentu jazyka XHTML

patří. Jestliže jsme si založili novou stránku v některém editoru, pak máme již několik nám nyní nesrozumitelných značek v dokumentu vložených.

Dokument HTML je uzavřen v elementu `html`, tedy mezi párové značky `<html>` a `</html>`. Před tímto elementem a v něm se vyskytují další značky, jimž zatím nerozumíme a ani se jimi nyní zabývat nemusíme – strukturu dokumentu si popíšeme až na straně 56. Nás bude nyní zajímat především to, kam zapisovat obsah a značky v rámci výuky v této kapitole.

Pamatujme si, že vše, co se má zobrazit v okně prohlížeče, patří mezi značky `<body>` a `</body>`. Více naznačuje obrázek 2.1, jenž zobrazuje dokument založený dle šablony popsané v kapitole 1 (strana 28) v PSPadu.

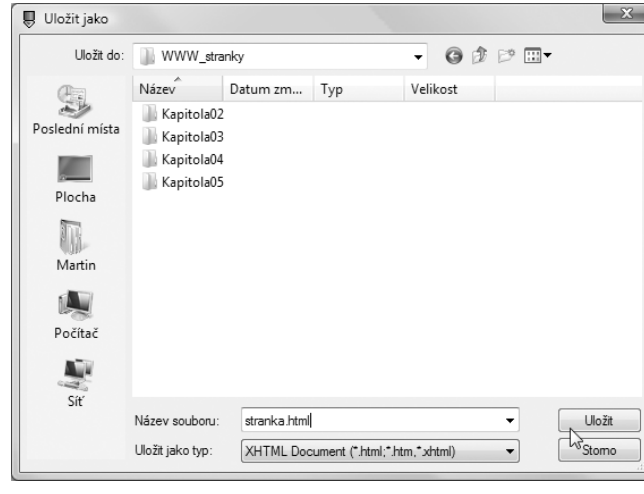
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="cs" lang="cs">
 <head>
  <meta http-equiv="content-type" content="text/html; charset=windows-1250" />
  <title></title>
 </head>
 <body>

  Tělo dokumentu - sem patří obsah zobrazený na monitoru

 </body>
</html>
```

Obrázek 2.1. Dokument založený v PSPadu – obsah vkládáme do značky body

Řešení problému: Jestliže stránku nevytváříte v PSPadu, ale třeba v Poznámkovém bloku, pak vytvořte stránku tak, že do prázdného textového dokumentu vložíte otevírací značku `<html>`, za ni otevírací značku `<body>`, poté uzavírací značku `</body>` a za ni uzavírací značku `</html>`. Dokument uložte s koncovkou html – například stranka.html – jako čistý text.

Značka `body`, česky tělo, označuje hlavní část dokumentu XHTML, opravdu tedy jeho tělo. Vše, co sem zapíšeme, se zobrazí v prohlížeči. Postupujme tedy takto:

1. Do elementu `body` napíšeme nějaký text.
2. Dokument uložíme (například **Soubor → Uložit**).
3. Zvolme svou složku a název souboru zapište

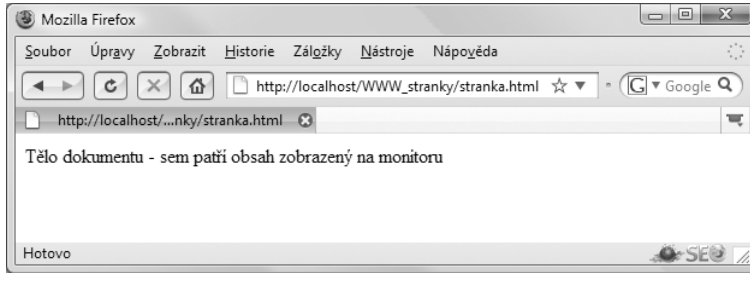


Obrázek 2.2. Ukládáme dokument HTML

včetně správné koncovky. Dokumenty jazyka HTML mají koncovky *html*. Název souboru by tedy mohl vypadat takto: *stranka.html*.

4. Stiskneme tlačítko **Uložit**.

Jakmile máme stránku uloženou v souboru, poklepejme na ni třeba v Průzkumníkovi Windows či v nějakém souborovém manažeru (např. Total Commanderu), a stránka se otevře ve výchozím internetovém prohlížeči. Pokud ji budeme chtít otevřít v jiném prohlížeči, pak budeme muset na soubor klepnout pravým tlačítkem a v místní nabídce zvolit příkaz Otevřít v programu. Výsledek si můžeme prohlédnout na obrázku 2.3.



Obrázek 2.3. Vše, co zapíšeme do značky body, se zobrazí v prohlížeči

Když jsme dokončili tuto tak trochu teoretickou pasáž, vrhněme se už raději na skutečné značky a tvorbu naší první webové stránky.

První texty na stránce

První text na stránce už vlastně máme za sebou. Ale tento text jsme neumístili do žádné značky, která by prohlížeči sdělila cokoliv o jeho významu. Prohlížeč neví, jestli má jít o nadpis, odstavec či něco jiného. S tím je třeba něco udělat.

Odstavce textu a nadpisy

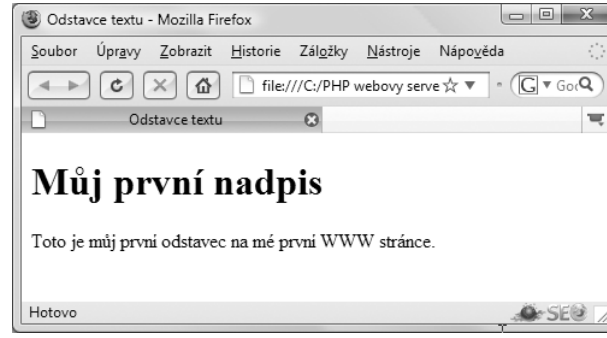
Začněme tím, že na stránce vytvoříme nadpis a pod ním odstavec textu. Postup bude následující (vysvětlení bude následovat):

1. Mezi značky `<body> </body>` vložme značky `<h1> </h1>`.
2. Mezi tyto značky napišme náš nadpis.
3. Za značkou `</h1>` stiskněme **Enter** a zapišme značky pro odstavec `<p> </p>`.
4. Mezi tyto značky zapišme text odstavce.

Celý kód může vypadat třeba takto:

```
<body>
  <h1>Můj první nadpis</h1>
  <p>Toto je můj první odstavec na mé první WWW stránce.</p>
</body>
```

Dokument uložme a spusťme v prohlížeči. Obrázek 2.4 zachycuje výsledek. Bylo to jednoduché, že?



Obrázek 2.4. Náš první nadpis a odstavec textu

A nyní slibované vysvětlení. Každý prvek v dokumentu HTML se nazývá element. A elementem odstavce je p (zkratka anglického slova paragraph, česky odstavec). Element `p` je určený k označení odstavce a má k dispozici dvě párové značky – `<p>` a `</p>`. Vše mezi těmito značkami je obsah elementu `p` a s tímto obsahem pracuje prohlížeč. Prohlížeč ví, že jde o odstavec textu, neboť text jsme umístili do elementu `p`.

Důležité: Běžně se pojem značka a element zaměňuje a vlastně o moc nejde, takže jestli budete chtít tyto pojmy zaměnit, nevadí. Hlavně když si budeme rozumět.

A nyní k nadpisu. Jazyk HTML nabízí 6 úrovní nadpisů. A proč? I v této knize totiž máme nadpis kapitoly, poté nižší úroveň – nadpis podkapitoly, a ještě nižší úroveň nadpisu – nadpis části podkapitoly. Obecný zápis elementu nadpis zní:

`hX` kde `h` je element nadpisu a `X` je číslo od 1 do 6

Element `h` (anglicky *headline*, česky *nadpis*) tedy může ve svých šesti úrovních vypadat takto: `h1`, `h2`, `h3`, `h4`, `h5` nebo `h6`. Podívejme se na praktický příklad:

```
<body>
<h1>Nadpis 1. úrovně</h1>
<p>Zcela běžný odstavec textu pod
   nadpisem první úrovně.</p>
<h2>Nadpis 2. úrovně</h2>
<p>Zcela běžný odstavec textu pod
   nadpisem druhé úrovně.</p>
<h3>Nadpis 3. úrovně</h3>
<p>Zcela běžný odstavec textu pod
   nadpisem třetí úrovně.</p>
...
</body>
```

Kód by mohl pokračovat dalšími úrovněmi. Jak vše vypadá v prohlížeči, ukazuje obrázek 2.5.

Obrázek 2.5. Nadpisy více úrovní jazyka HTML

Po pohledu na obrázek 2.5 vás možná napadne otázka – tedy napadne vás, pokud jste přečetli kapitolu 1, v níž jsme se bavili o tom, že jazyk HTML neslouží k určení vzhledu textu ani jiného obsahu stránky. Ano, to je pravda. Nicméně prohlížeče v sobě mají zabudované jisté základní styly jednotlivých elementů jazyka HTML, takže i bez toho, abychom jim pomocí jazyka CSS jakýkoli styl přidělovali, vykreslí tyto elementy se základním vzhledem.

Nutno podotknout, že v každém prohlížeči je tento základní vzhled trochu odlišný. My se nyní vzhledem nezabývejme, neboť k tomu slouží CSS a k němu se dostaneme později.

Zvýraznění textu v odstavci

Dříve v této kapitole, když jsme si ukazovali nebezpečí kříženého zápisu značek, jsme si ukazovali tuto modelovou situaci na příkladu, kdy chceme zdůraznit nějaké slovo nebo slova v odstavci. Nyní se podíváme na elementy, díky nimž můžete slovům přidat nějaký význam:

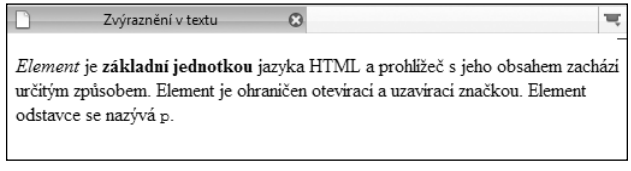
- `strong` (česky *silný*) – používá se k vyzvednutí důležitých částí věty
- `em` (zkratka z anglického *emphasize*, česky *zvýraznění*) – slouží ke zvýraznění částí textu
- `cite` (česky *citovat*) – element určený k vyznačení citací v textu
- `code` (česky *kód*) – element k označení výpisu kódu
- `ins` (zkratka anglického *inserted*, česky *vloženo*) – označuje vložený text
- `del` (zkratka anglického *deleted*, česky *vymazáno*) – označuje smazaný a neaktuální text

Existují ještě další elementy, ale jejich znalost není důležitá. Nejvíce se používají elementy `strong` a `em`, a to ke zvýraznění částí textu ve větách. V případě blogů, kdy zpětně upravujeme své příspěvky, ale chceme uživatelům ponechat i původní text, se budou hodit i elementy `ins` a `del`.

Důležité: Přestože prohlížeč ve výchozím stylu elementy strong a em formátuje tučně a kurzivou, neznamená to, že byste měli tyto elementy používat k formátování. HTML slouží k dodání významu. Jejich použitím tedy říkáme prohlížeči: na toto slovo kladu důraz, nikoliv: toto slovo naformátuj tučně.

Podívejme se na praktickou ukázku toho, jak tyto elementy použít a jak se budou ve výchozím stavu zobrazovat v prohlížeči. I když jejich zobrazení později upravíme pomocí CSS, získáme tak alespoň základní přehled:

```
<p><em>Element</em> je <strong>základní jednotkou</strong> jazyka HTML
   a prohlížeč s jeho obsahem zachází určitým způsobem. Element je
   ohraničen otevírací a uzavírací značkou. Element odstavce se nazývá
   <code>p</code>.</p>
```



Obrázek 2.6. Prohlížeč ve výchozím stavu přiděluje elementům základní styl – zde elementy pro zvýraznění textu

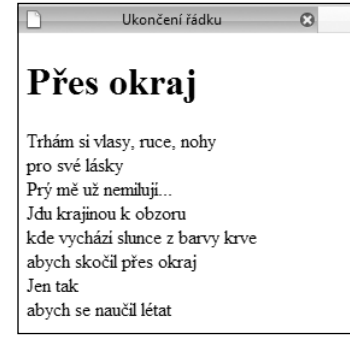
Řešení problému: Jestliže máte v PSPadu problém s příliš dlouhým řádkem, mám pro vás řešení. V nabídce **Zobrazení** klepněte na příkaz **Zalomení řádků**. Text se zalomí pod svislé vodicí čáry, kterou můžete posunout na místo, které vám vyhovuje, klepnutím na pravítko nad dokumentem. PSPad automaticky zalomí řádek – ale pozor, jen vizuálně. Pokud budete chtít udělat mezeru klávesou Enter, budete muset tuto klávesu použít.

Ukončení řádku

Někdy může nastat situace, kdy bude třeba použít element pro ukončení řádku. Tímto elementem je `br` a je označen nepárovou značkou `<br />`. Co je jeho cílem? Podobně jako na psacím stroji provede přejezd vozíku na konec řádku a skočí na řádek následující. Text následující tento element pak začne na následujícím řádku. Něco takového se může například hodit, pokud budete chtít na svých stránkách publikovat báseň.

Podívejme se, jak vypadá element `br` v praxi:

```
<body>
<h1>Přes okraj</h1>
<p>Trhám si vlasy, ruce, nohy<br />
   pro své lásky<br />
   Prý mě už nemilují...<br />
   Jdu krajinou k obzoru<br />
   kde vychází slunce z barvy krve<br />
   abych skočil přes okraj<br />
   Jen tak<br />
   abych se naučil létat<br />
</p>
</body>
```



Obrázek 2.7. Ukončení řádku se může v některých situacích hodit

Jak ukazuje obrázek 2.7, může se hodit i ukončení řádku odstavce. Element `br` bychom však neměli používat k vytváření mezer mezi odstavci či jinde. K vytváření okrajů elementu (například spodního okraje pod odstavcem) slouží vlastnosti jazyka CSS, a tomu se budeme věnovat dále v knize.

Tvoříme seznamy

Seznamy, ač se to na první pohled možná nebude zdát, jsou velmi silnou a používanou zbraní jazyků HTML, a zvláště XHTML. Dokážeme pomocí nich strukturovat text do odrážkových a číslovaných seznamů. Tyto seznamy nazýváme dle jejich povahy:

- *Neuspořádané seznamy* – jejich položky mají vzájemný vztah, ale nezáleží na jejich pořadí v seznamu – proto neuspořádaný.
- *Uspořádané seznamy* – jejich položky mají vzájemný vztah a záleží na pořadí, v němž se zobrazují.

Jak už text naznačil, budeme se bavit o seznamu jako takovém a dále o jeho položkách. Půjde tedy o dva elementy, které jsou do sebe vnořené. Podíváme se na ně nyní prakticky.

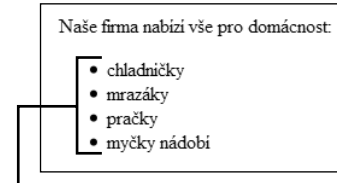
Neuspořádané (odrážkové) seznamy

Neuspořádané seznamy se skvěle hodí pro výčty, jako je třeba seznam položek nákupu, nabídka služeb firmy či seznam doporučené literatury. Takové neuspořádané seznamy můžeme vidět i v této knize – jejich položky vždy začínají odrážkou. Pro neuspořádaný seznam nabízí HTML element `ul` (zkratka anglického *unordered list*, česky *neuspořádaný seznam*) s párovými značkami `<ul>` a `</ul>`:

```
<ul>
Sem patří neuspořádaný seznam
</ul>
```

Položky takového seznamu definuje druhý element, jenž je společný pro oba typy seznamu a nazývá se `li` (zkratka anglického *list item*, česky *položka seznamu*) – používá párové značky `<li>` a `</li>`. Podívejme se nyní na to, jak takový seznam i s položkami vypadá:

```
<p>Naše firma nabízí vše pro domácnost:</p>
<ul>
  <li>chladničky</li>
  <li>mrazáky</li>
  <li>pračky</li>
  <li>myčky nádobí</li>
</ul>
```



Obrázek 2.8. Neuspořádaný seznam vytvořený v HTML

Výsledek v prohlížeči ukazuje obrázek 2.8. Jak si můžeme všimnout, jednotlivé položky se uspořádaly do seznamu a označily se odrážkami.

Uspořádané (číslované) seznamy

Představme si, že chceme na svých stránkách uvést nějaký postup nebo recept v rámci krokového postupu. Každý krok pak má být logicky označen číslem. Je samozřejmé, že bychom mohli vytvořit každý krok pomocí elementu `p` pro odstavec, ale logicky bychom tím prohlížeči řekli pouze to, že na stránce se nachází několik za sebou jdoucích odstavců bez jakékoli spojitosti. To ale nechceme. My chceme pomocí kódu HTML dodat textu význam krokového postupu. Pak zvolíme uspořádaný seznam.

Uspořádaný seznam definuje v HTML element `ol` (zkratka anglického *ordered list*, česky *uspořádaný seznam*) a jeho položky nám už známý element `li`. Návod vytvořený pomocí uspořádaného seznamu by mohl vypadat následovně:

```
<p>Postup je následující:</p>
<ol>
  <li>Do hrnce nalijte 1 litr vody.</li>
  <li>Vodu přiveďte k varu.</li>
  <li>Vsypte obsah sáčku a zamíchejte.</li>
  <li>Za občasného míchání vařte 10 minut.</li>
</ol>
```

Postup je následující:

1. Do hrnce nalijte 1 litr vody.
2. Vodu přiveďte k varu.
3. Vsypte obsah sáčku a zamíchejte.
4. Za občasného míchání vařte 10 minut.

Obrázek 2.9. Uspořádaný seznam vytvořený v HTML

Výsledek v prohlížeči zachycuje obrázek 2.9. Všimněme si, že prohlížeč použil pro označení položek uspořádaného seznamu číslice a jednotlivé položky čísluje vždy s přičtením jedničky.

Poznámka: Pomocí CSS později můžete změnit jak podobu odrážek neuspořádaného seznamu, tak značení položek uspořádaného seznamu. Místo arabských číslic tak použijete třeba římské číslice nebo písmena.

Víceúrovňové seznamy

Možná, nebo pravděpodobně, nás již při cvičení se seznamy napadlo, jestli bychom mohli vnořit více seznamů do sebe a vytvořit tak více úrovní seznamu. Vypadat by to mohlo například takto:

- Text
  - Nadpisy
  - Odstavce
  - Zvýraznění textu
- Seznamy
  - Neuspořádané
  - Uspořádané
  - Víceúrovňové

Ano, toto je víceúrovňový seznam a HTML se dokáže vypořádat i s tímto požadavkem. Podobně by mohl vypadat i seznam uspořádaný. Počtem úrovní seznamu přitom nejsme nijak limitování. Snad jen přehledností, přece jen seznam s 20 úrovněmi už není to pravé ořechové a hádám, že bychom se tak po šesté až sedmé úrovni ztratili. Podívejme se ale, jakým postupem výše uvedený víceúrovňový seznam vytvoříme.

Postupujme následovně:

1. Vytvořme běžný seznam o dvou položkách.

```
<ul>
  <li>Text</li>
  <li>Seznamy</li>
</ul>
```

2. Za textem první položky seznamu (v našem případě za slovem `Text`) stiskneme **Enter.**
3. Do tohoto místa (za slovo `Text` a před uzavírací značku `</li>`) zapíšeme seznam ve druhé úrovni.

```
<ul>
  <li>Text
      <ul>
         <li>Nadpisy</li>
         <li>Odstavce</li>
         <li>Zvýraznění textu</li>
      </ul>
  </li>
  <li>Seznamy</li>
</ul>
```

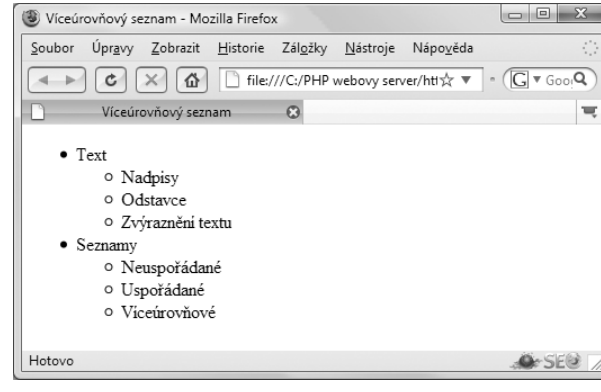
4. A totéž provedeme za textem druhé položky seznamu v první úrovni, tedy za slovem `Seznamy`.

   Hotový kód bude vypadat následovně:

```
<ul>
  <li>Text
      <ul>
         <li>Nadpisy</li>
         <li>Odstavce</li>
         <li>Zvýraznění textu</li>
      </ul>
  </li>
  <li>Seznamy
      <ul>
         <li>Neuspořádané</li>
         <li>Uspořádané</li>
         <li>Víceúrovňové</li>
      </ul>
  </li>
</ul>
```

Tip: Všimněte si, jak vhodné odsazení některých řádků kódu celý kód zpřehledňuje. Obecně se při kódování toto odsazení používá, aby si webdesignér snadno uvědomoval strukturu vnořených elementů. Odsazení lze udělat pomocí mezerníku či klávesy **Tab**.

Výsledek našeho víceúrovňového seznamu zachycuje obrázek 2.9. V jednotlivých úrovních bychom mohli samozřejmě pokračovat stejným způsobem. Každý nový seznam uzavřený mezi značky `<ul>` a `</ul>` (nebo `ol`) patří do položky původního seznamu – tedy mezi značky `<li>` a `</li>`. Všimněme si, že prohlížeč odlišuje jednotlivé úrovně jinou odrážkou. Podobu odrážky v jednotlivých úrovních sami snadno ovlivníme pomocí CSS (o tom dále v knize).

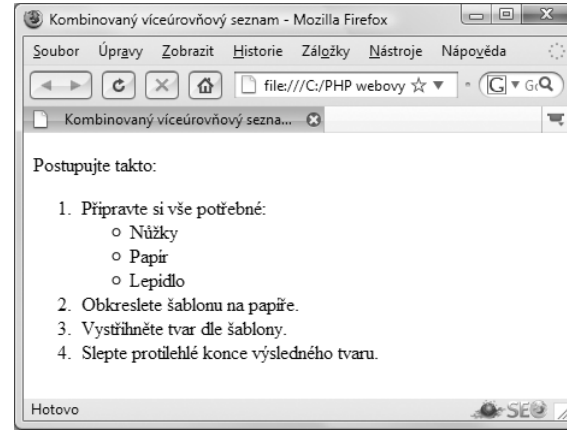


Obrázek 2.10. Víceúrovňový neuspořádaný seznam v HTML

Stejným způsobem můžeme vytvořit víceúrovňový uspořádaný seznam. Co je ale možné, je také jejich kombinace. Postup je zcela stejný, jen pro uspořádaný seznam použijeme známý element `ol`. Kombinovaný víceúrovňový seznam by mohl vypadat třeba takto:

```
<p>Postupujte takto:</p>
<ol>
  <li>Připravte si vše potřebné:
      <ul>
         <li>Nůžky</li>
         <li>Papír</li>
         <li>Lepidlo</li>
      </ul>
  </li>
  <li>Obkreslete šablonu na papíře.</li>
  <li>Vystřihněte tvar dle šablony.</li>
  <li>Slepte protilehlé konce výsledného tvaru.</li>
</ol>
```

Výsledek v prohlížeči zaznamenává obrázek 2.11. Všimněme si toho, že prohlížeč pokračuje správně v číslování, i když jsme do kroku 1 vložili neuspořádaný seznam.



Obrázek 2.11. Kombinovaný víceúrovňový seznam v HTML

Používáme odkazy

Odkazy jsou vlastně tím nejdůležitějším, co nám internetové stránky nabízí. Díky nim se můžeme dostat na jinou stránku téhož nebo zcela jiného webu na Internetu. Takový odkaz ale může vést též k nějakému textovému dokumentu, k hudební nahrávce či videu. V našem případě bude odkaz také velmi důležitý, neboť z odkazů se bude skládat navigační nabídka našeho webu.

My se budeme potýkat se dvěma typy odkazů, jejichž význam si také vysvětlíme a naučíme se je především používat:

- *Odkazy hypertextové* – vedou na jinou internetovou stránku (na jiný dokument HTML) či na jakýkoli jiný soubor (textový dokument, skladbu v MP3 atd.).
- *Odkazy jmenné* – označují konkrétní místo na WWW stránce, jež může být cílem hypertextového odkazu. O tomto typu odkazu dále v této podkapitole.

Bez hypertextových odkazů by se Internet neobešel – ostatně, jak jsme si asi všimli, slovo hypertextový je součástí také názvu HTML (hypertextový značkovací jazyk). Hlavním účelem hypertextových odkazů je umožnit přechod k jinému textu (rozuměj na jinou WWW stránku) klepnutím na nějaký text, případně obrázek. Hypertextové odkazy všichni dobře známe – obvykle jde o podtržený text na stránce, případně o grafické tlačítko.

K čemu jsou parametry

Pro odkaz se v jazyce HTML používá element `a` (zkratka anglického *anchor*, česky *kotva*), v dokumentu jej zaznačíme pomocí značek `<a>` a `</a>`. Do elementu `a` patří obvykle text nebo jiné elementy – třeba obrázek (o tom dále). Samotná značka `a` však není schopná přenést nás na jiné místo na Internetu. Potřebuje pomocníka, jemuž říkáme v jazyce HTML parametr.

Parametr zapisujeme následujícím způsobem:

```
<značka parametr="hodnota">obsah elementu</značka>
```

Jak si můžeme všimnout, parametr se zapisuje do otevírací značky, následuje jej rovnítko a hodnota vložená do uvozovek. Tak jako jazyk HTML definuje elementy s jejich značkami, tak definuje také parametry a jejich hodnoty. Každý parametr má své jméno, lze ho použít jen u některých elementů a může nabývat jen určitých hodnot. O všech nutných parametrech a jejich hodnotách se dozvíme dále v této knize.

Důležité: Přestože jazyk HTML umožňuje hodnotu parametru zapisovat bez uvozovek (pokud se skládá z jedné hodnoty), pravidla jazyka XHTML, jichž se držíme v této knize, to neumožňují. Hodnoty je tedy vždy třeba psát do uvozovek.

Adresa URL

Parametrem, který budeme potřebovat k určení cíle odkazu, je `href`, a jeho hodnotou je URL (anglicky *Unified Resource Locator*, česky *jednotný lokátor zdroje*) – obecně říkáme adresa URL. I když nám možná výraz URL moc neříká, známe jej. Adresou URL je třeba WWW adresa – například *www.cpress.cz*. Adresa URL může mít dvě podoby:

- *Absolutní adresa* – plná WWW adresa dokumentu na Internetu, například ***http://www.mojestranky.cz/stranka.html***. Skládá se z protokolu (http), WWW adresy a za lomítkem z názvu souboru. Za lomítko před název souboru může samozřejmě přijít ještě nějaká složka – pak by adresa vypadala takto: ***http://www.mojestranky.cz/adresar/stranka.html***.
- *Relativní adresa* – cesta k dokumentu v rámci jednoho webového sídla (webu). Například URL ve tvaru ***../adresar/stranka.html*** říká: „Běž o jednu úroveň v adresářové struktuře výš, pak do složky **adresar** a zde je cílem dokument ***stranka.html***.“ Relativní adresa se vždy odvíjí od umístění dokumentu, z něhož odkazujeme.

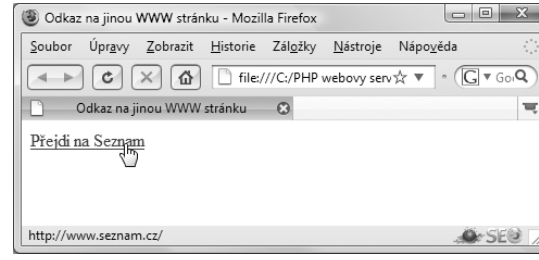
Tvoříme první odkazy

Jak už jsme si řekli, odkaz definuje element `a` a cíl odkazu parametr `href` s hodnotou ve formě adresy URL. A nyní konečně k dlouho očekávanému okamžiku. V našem testovacím dokumentu (X)HTML zapíšeme do těla WWW stránky (za značku `<body>`) náš první odkaz. Může vypadat třeba takto:

```
<a href="http://www.seznam.cz">Přejdi na Seznam</a>
```

Důležité: Absolutní adresa URL musí obsahovat také platný zápis protokolu HTTP – tedy `http://`. URL nelze začít obyčejnou WWW adresou.

Uložme náš dokument a vyzkoušejme odkaz v praxi. Na stránce se ve výchozím stavu text odkazu zobrazí podtržený a modře. Klepněme na něj a prohlížeč načte novou WWW stránku z adresy, kterou jsme určili jako hodnotu parametru `href`.



Obrázek 2.12. Odkaz na WWW stránce

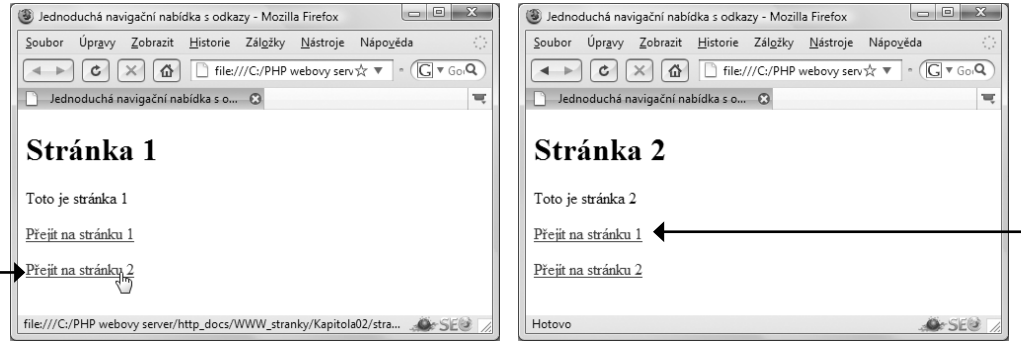
Na začátku jsem se ovšem zmiňoval o tom, že odkazy používáme zvláště k vytvoření navigační nabídky našeho webu, aby se návštěvník mohl snadno přepínat mezi jednotlivými stránkami. A to je pravda. Proto si ještě ukážeme, jak vytvořit odkazy, které propojí vzájemně více stránek. K tomu se nám bude hodit relativní URL.

Vytvořme nyní dva dokumenty jazyka (X)HTML, které uložme do jedné složky, například pod názvy *stranka1.html* a *stranka2.html*. Tělo toho prvního by mohlo vypadat následovně (nezapomeňte na element `html`, pokud stránku netvoříte v PSPadu):

```
<body>
    <h1>Stránka 1</h1>
    <p>Toto je stránka 1</p>
    <p><a href="stranka1.html">Přejít na stránku 1</a></p>
    <p><a href="stranka2.html">Přejít na stránku 2</a></p>
</body>
```

Druhý dokument vytvořme podobně, pouze zmínku o stránce 1 změňme na stránku 2. A pak postupujme následovně:

1. V prohlížeči otevřeme dokument stranka1.html.
2. Klepneme na odkaz **Přejdi na stránku 2**.
3. Prohlížeč otevře a zobrazí dokument stranka2.html.
4. Klepneme na odkaz **Přejdi na stránku 1**.
5. A prohlížeč zobrazí zase první stránku.



Obrázek 2.13. Pomocí hypertextových odkazů jsme vytvořili primitivní navigační nabídku, jež provazuje dva dokumenty HTML mezi sebou

Právě jsme vytvořili velmi jednoduchou navigační nabídku, jíž jsme provázali dva dokumenty (X)HTML mezi sebou. Jak si můžeme všimnout, protože jsou oba soubory uložené v jedné složce, použili jsme jako hodnotu parametru `href` pouze jejich název. Pokud by se vyskytoval druhý soubor v jiné složce, pak bychom museli před název souboru vepsat i relativní cestu k druhému dokumentu.

- Pokud by byl dokument stranka2.html umístěn ještě ve složce ***adresar***, pak bychom do parametru `href` napsali hodnotu `adresar/stranka2.html` (= vejdi do složky ***adresar*** a zde otevři dokument ***stranka2.html***).
- V dokumentu stranka2.html bychom museli do parametru `href` odkazu na dokument `stranka1.html` zapsat cestu `../stranka1.html` (= jdi do složky o úroveň výše a zde otevři dokument ***stranka1.html***).

Poznámka: Lomítko se používá k oddělení úrovní stromové struktury složek a dvě za sebou následující tečky znamenají „přejdi do složky o úroveň výše". Pokud byste chtěli přejít do složky o dvě úrovně výše, pak byste museli použít cestu `../../stranka.html`. Naopak, pokud byste chtěli přejít do vnořené složky v jiné vnořené složce, pak byste museli použít cestu `adresar1/adresar2/stranka.html`. A tak dále.

Je třeba si především pamatovat, že relativní URL vede ze složky, v níž je umístěn dokument (X)HTML, ze kterého odkazujeme (= v němž je umístěn odkaz). Odkazy jsou pro tvorbu WWW stránek skutečně zcela klíčové, proto je nezbytné, abychom si výše popsaný princip vyzkoušeli a abychom jej pochopili.

Odkazujeme na další soubory na Internetu

Jak jsme si uvedli dříve, odkaz nemusí vést pouze na jinou internetovou stránku, tedy dokument HTML, ale jeho cílem může být libovolný soubor umístěný na Internetu. Stačí k němu znát cestu nebo WWW adresu. A k čemu to vše využít?

- Můžeme tak na Internetu zveřejnit dokument Wordu či dokument ve formátu PDF.
- Lze odkazovat na hudební skladby či videa.
- Můžeme odkazovat na jakýkoli jiný soubor.

Takové odkazy mohou vypadat například takto:

```
<a href="dokument.pdf">Dokument v PDF</a>
<a href="skladba.mp3">Skladba v MP3</a>
```

Co se stane, pokud je cílem webová stránka, už víme. Prohlížeč ji načte a zobrazí. Ale co se stane, jestli je cílem odkazu například textový dokument nebo hudební nahrávka? Je to prosté, nastat může jeden ze dvou scénářů, a to v tomto pořadí:

1. Prohlížeč se podle přípony souboru pokusí najít výchozí program určený k otevření tohoto typu souboru (například přípona DOC znamená, že se otevře ve Wordu; přípona MP3 zase, že se otevře v hudebním přehrávači). A soubor se otevře ve výchozím programu.

2. Pokud k typu souboru v počítači není přiřazený žádný výchozí program, prohlížeč otevře dialog, v němž nabídne stažení souboru a volbu ručního výběru programu, v němž se má soubor otevřít.

Důležité: Jestli se soubor otevře ve správném programu, nelze zajistit, neboť záleží na nastavení počítače uživatele, jenž na odkaz klepne.

E-mailová adresa jako odkaz

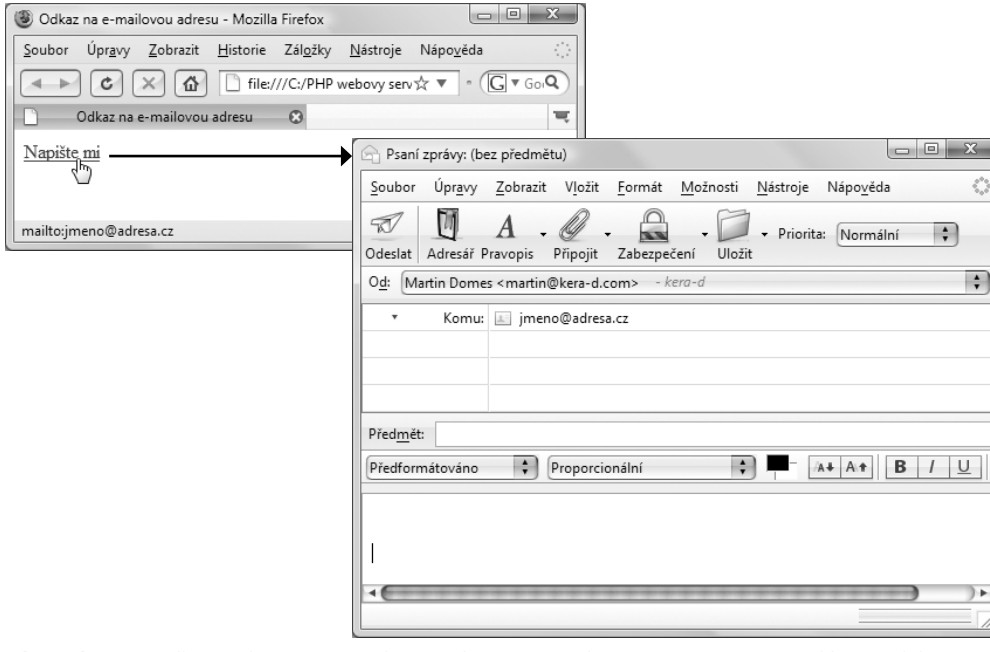
Ano, jako cíl odkazu může sloužit také e-mailová adresa se známým zavináčem. Můžeme tak snadno vytvořit odkaz, díky němuž nám budou moci uživatelé stránek rovnou napsat e-mail, aniž by museli znát naši e-mailovou adresu.

Odkaz směřující na e-mailovou adresu je trochu odlišný od běžných odkazů, ale ne tak moc, abychom hned nepochopili, v čem je onen trik. Odkaz může vypadat takto:

```
<a href="mailto:jmeno@adresa.cz">Napište mi</a>
```

Jak si můžeme všimnout, před e-mailovou adresou se objevuje záhadný výraz `mailto` – tento výraz je třeba vždy předřadit zápisu samotné e-mailové adresy, neboť tím říkáme prohlížeči, že cílem odkazu je právě e-mail.

No a co se stane, jestliže na takový odkaz klepneme? Prohlížeč vyšle požadavek do počítače uživatele, aby se otevřelo okno nové e-mailové zprávy v klientovi elektronické pošty s vyplněnou e-mailovou adresou. Vše ukazuje obrázek 2.14.



Obrázek 2.14. Odkaz vedoucí na e-mailovou adresu spustí okno pro novou zprávu v klientu elektronické pošty

Důležité: Aby takový odkaz fungoval, uživatel musí mít v počítači nainstalovaný klient elektronické pošty (např. Outlook, Outlook Express, Windows Mail nebo jakýkoli jiný). Pokud jej nemá, odkaz fungovat nebude. Také není nikde zaručeno, že i když uživatel klient v počítači nainstalovaný má, že jej také používá. Proto nikdy nepoužívejte texty jako **Napište mi**, jak jsem uvedl výše, ale vždy sem napište celou e-mailovou adresu, aby si ji lidé mohli zkopírovat.

Odkazujeme na jiné místo na stejné stránce

Představme si dlouhou, opravdu dlouhou internetovou stránku, například s nějakým příběhem zapsaným do několika kapitol. To vše v jediném dokumentu (X)HTML. Pokud budeme vše číst najednou, je vše v pořádku. Jestliže si ale dnes chceme přečíst jen první kapitolu a příště číst druhou, museli bychom se před čtením druhé kapitoly posunout dokumentem dolů až k jejímu začátku. Díky jmenným odkazům to však není nutné.

Na začátku této podkapitoly věnované odkazům jsem se zmínil o tom, že název elementu `a` je zkratkou anglického slova *anchor*, česky *kotva*. Kotva slouží k označení místa v dokumentu (X)HTML, k němuž se můžeme posunout pomocí klasického hypertextového odkazu. Této kotvě jinak také říkáme *jmenný odkaz* – to protože pojmenovává konkrétní místo v dokumentu a díky tomuto jménu na něj můžeme odkazovat. Pojďme se na to ale raději podívat prakticky.

Vytvořme nějaký hodně dlouhý dokument, třeba s nějakou opravdovou povídkou v několika kapitolách (nebo použijte ukázku kódu ze zdrojových kódů k této knize, viz strana 12). Použít ale můžeme jakýkoli text s jakýmikoli nadpisy, které budou označovat začátek kapitoly. Hlavně ať je dokument dlouhý alespoň přes několik stránek. To vše samozřejmě umístíme do elementu `body` a celý takový dokument (X)HTML uložíme. Podobu takového dokumentu naznačuje následující výpis kódu:

```
<body>
  <h1>Kapitola 1</h1>
  <p>Text ...</p>
  <h1>Kapitola 2</h1>
  <p>Text ...</p>
  ...
</body>
```

Nyní, jak už jsme si říkali, bude třeba pojmenovat části dokumentu, abychom na ně mohli později z úvodu stránky odkazovat. Pojmenujeme proto každý nadpis kapitoly pomocí jmenného odkazu. Použijeme element `a` pro odkaz, ale místo parametru `href` (protože nechceme udat cíl odkazu) použijeme parametr `name` (jímž pojmenujeme odkaz). Takto:

```
<h1><a name="kapitola1">Kapitola 1</a></h1>
<p>...
<h1><a name="kapitola2">Kapitola 2</a></h1>
<p>...
```

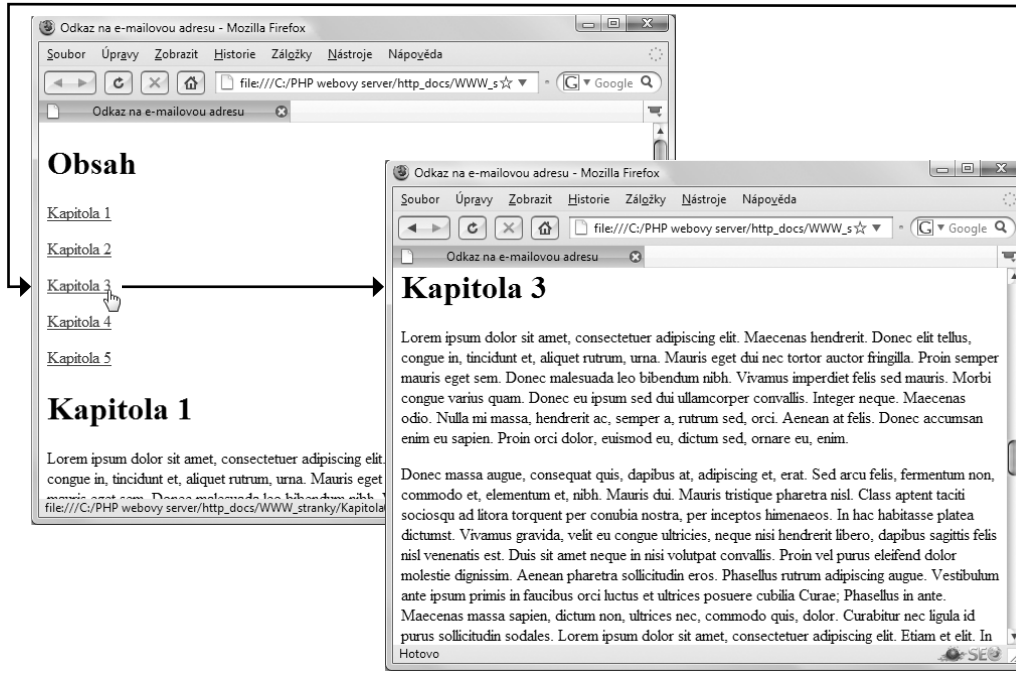
A nyní budeme chtít vytvořit na začátku dokumentu (před kapitolou 1) nabídku odkazů směřujících na jednotlivé kapitoly = na jednotlivé jmenné odkazy – v našem případě ji

můžeme nazvat Obsah. Použijeme opět element `a` pro odkaz a parametr `href` pro určení cíle odkazu. Jen jeho hodnota bude nyní vypadat odlišně. Podívejme se, jak bude náš kód obsahu vypadat:

```
<h1>Obsah</h1>
<p><a href="#kapitola1">Kapitola 1</a></p>
<p><a href="#kapitola2">Kapitola 2</a></p>
...
```

Jak si můžeme všimnout, hodnota parametru `href` se skládá ze znaku mřížky a názvu jmenného odkazu (tedy oné kotvy). Když nyní na odkaz klepneme, řekněme na odkaz na kapitolu 3, přesuneme se v prohlížeči na stránce dolů na kapitolu 3. Prohlížeč místo označené jako jmenný odkaz zobrazí zcela nahoře.

Řešení problému: Pokud nemáte k dispozici dostatečně dlouhý text, pak k vyzkoušení fungování jmenných odkazů zmenšete výšku okna prohlížeče.



Obrázek 2.15. Odkazy mohou směřovat i na jiné místo v témže dokumentu

Vkládáme obrázky do WWW stránky

Co by dnes byl Internet bez obrázků, ilustrací či fotografií, že? Navíc vhodný obrázek dokáže někdy textovou informaci zastoupit nebo minimálně vhodně doplnit. Obrázky používáme na webových stránkách dvojím způsobem. Nyní se podíváme na ten první:

- *Obrázky vložené pomocí kódu jazyka (X)HTML* do těla stránky slouží jako doplněk textu, k němuž se bezpodmínečně vztahují. Takové obrázky se nepoužívají k obyčejné dekoraci bez skutečného významu.
- *Obrázky na pozadí, obrázky tlačítek a jiná grafika* použitá jako dekorace se do stránky vkládá pomocí kaskádových stylů (CSS). Tomu se budeme věnovat později v kapitole 4.

V této podkapitole se tedy zaměříme na vložení obrázků či fotografií, které doplní text webové stránky. Může jít například o naši fotografii, pokud budeme tvořit osobní stránky, či fotografie obchodu, pokud budeme hovořit o firemní webové prezentaci.

Jaký obrázek lze použít na Internetu

Obrázky či fotografie můžeme uložit v mnoha různých formátech. Je tedy na místě si zodpovědět otázku, které z těchto formátů lze použít na Internetu. Problém není ani tak v tom, co zvládne Internet, ale spíše v tom, jaké obrázky dokáže zobrazit náš prohlížeč. A věřte mi, není jich mnoho.

Nebudeme se zde zatěžovat složitou teorií ani mnoha pojmy, pouze si stručně vyjmenujeme a popíšeme hlavní formáty, v nichž budeme obrázky ukládat v grafickém editoru. Pokud budeme v této knize hovořit o obrázcích, fotografiích či grafice, pak budeme mít na mysli soubory obrázků v následujících formátech:

- *GIF* – obrázky v tomto formátu se obvykle používají pro různá tlačítka a doplňkovou grafiku s malým počtem barev. Nabízí i jednoduché animace na základě skládání více obrázků do jednoho a také průhlednost.
- *PNG* – jde o podobný formát jako GIF, ale umožňuje použití více barev, proto je vhodný i pro menší fotografie. Nabízí lepší průhlednost než GIF, ale není použitelný pro animace.
- *JPEG* – tento formát se nejčastěji používá pro ukládání fotografií, jež chceme použít na Internetu. Pracuje s velkým počtem barev, neumí však průhlednost a animace.

A proč vlastně používat různé formáty při ukládání obrázků pro jejich následné použití na WWW stránce? Taková fotografie, kterou si stáhneme z fotoaparátu do počítače, je velmi objemná – jenomže na internetové stránce musíme dávat pozor na to, aby všechna doprovodná grafika měla co nejmenší velikost. K tomu právě slouží výše uvedené formáty, které používají při uložení obrázků tzv. kompresi. Pomocí ní zmenší velikost souboru obrázku na co nejmenší hodnotu. Postup uložení obrázku do některého z těchto formátů si ukážeme později.

Poznámka: Komprese znamená zhuštění dat tak, aby konečný soubor měl co nejmenší velikost a z Internetu se do počítače při prohlížení stránek stáhl co nejrychleji. Komprese může být ztrátová a bezztrátová. Ztrátová komprese znamená, že při uložení obrázku se části obrázku vypustí, a to na základě znalosti nedokonalosti našeho oka. Algoritmus vypouští části obrázku tak, abychom to ve výsledku nepoznali. Bezztrátová komprese používá pro zmenšení velikosti zhuštění dat souboru obrázku, nic z něj tedy nevypouští. Například u formátu GIF se menší velikosti dosahuje též ruční redukcí počtu barev zahrnutých do obrázku. Pouze však tak, aby to oko nepoznalo.