Elizabeth Castro, Bruce Hyslop

Překlad světového bestselleru

HTML5 a CSS3

Názorný průvodce tvorbou WWW stránek

Od úplných základů po pokročilé techniky

Postupy krok za krokem

Nové prvky HTML5 a CSS3

Tvorba mobilních webů

Ke stažení zdrojové kódy příkladů z knihy

Elizabeth Castro
Bruce Hyslop

HTML5 a CSS3
Názorný průvodce tvorbou WWW stránek

Computer Press
Brno
2012

HTML5 a CSS3
Názorný průvodce tvorbou WWW stránek

Elizabeth Castro, Bruce Hyslop

Překlad: Ondřej Baše, Kristýna Baše
Obálka: Martin Sodomka
Odpovědný redaktor: Martin Herodek
Technický redaktor: Jiří Matoušek

Objednávky knih:
http://knihy.cpress.cz
www.albatrosmedia.cz
eshop@albatrosmedia.cz
bezplatná linka 800 555 513
ISBN 978-80-251-3733-8

Vydalo nakladatelství Computer Press v Brně roku 2012 ve společnosti Albatros Media a. s. se sídlem Na Pankráci 30, Praha 4. Číslo publikace 16 598.

1. vydání

ALBATROS MEDIA a.s.

Stručný obsah

Obsah

Úvod

Ať už začínáte pronikat do světa tvorby webových stránek, nebo už jste nějaké vlastní vytvořili a chcete se jen přesvědčit, že vaše znalosti jsou aktuální, ocitli jste se ve velmi vzrušující době pro tuto problematiku.

Za posledních několik let se razantně zlepšil způsob psaní kódu a stylů pro webové stránky, dále se zlepšily webové prohlížeče, v nichž tyto stránky prohlížíme, a také se zlepšila zařízení, na kterých tyto webové prohlížeče provozujeme. Dříve jsme mohli prohlížet stránky pouze z desktopových počítačů a notebooků, ale nyní si můžeme vzít celý web s sebou na cesty – na chytrých telefonech, tabletech, výše zmíněných noteboocích a spoustě jiných zařízení.

Tak by to samozřejmě mělo být, protože web vždy sliboval zánik jakýchkoliv hranic. Jeho síla tkví v tom, že umožňuje sdílet a načítat informace zcela volně ve městech, ve vesnicích a kdekoliv jinde, a to z jakéhokoliv zařízení s připojením na Internet. Web se tedy neustále rozrůstá, jelikož technologie si proklestily cestu k dříve nedostupným komunitám.

Další vynikající vlastností webu je, že kdokoliv si může vytvořit a spustit své vlastní webové stránky. Tato kniha vám ukáže, jak postupovat. Je ideální volbou pro začátečníky bez znalosti jazyků HTML a CSS, kteří chtějí začít vyvíjet webové stránky. Tato kniha vás provede procesem tvorby webových stránek krok za krokem pomocí jednoduchých instrukcí. Tato kniha může být také užitečnou příručkou – v obsahu nebo rejstříku si můžete vyhledat pouze vybraná témata, o nichž se chcete dozvědět více informací.

Stručně o jazycích HTML a CSS

Za úspěchem webu stojí jednoduchý textový značkovací jazyk, který se lze snadno naučit a podporují jej téměř všechna zařízení – jazyk HTML. Každá webová stránka se zakládá alespoň na minimálním množství kódu jazyka HTML; jinak by to nebyla webová stránka.

Jak se postupně v této knize naučíte, jazyk HTML je vhodný pro definování významu obsahu stránek, zatímco jazyk CSS určuje, jak tento obsah vypadá. Jak stránky jazyka HTML, tak soubory jazyka CSS (šablony stylů) jsou textové soubory, díky čemuž je lze opravdu snadno upravovat. Ukázkové fragmenty kódu jazyka HTML a CSS uvidíte v části „Z čeho se tato kniha skládá“ ke konci tohoto úvodu.

Do studia základní stránky HTML se pustíte hned na začátku kapitoly 1, „Stavební kameny

webových stránek." Měnit vzhled svých webových stránek se naučíte v kapitole 7, „Stavební kameny kaskádových stylů." V části „Co se v této knize naučíte" najdete stručný přehled všech kapitol a v nich pokrytých témat.

Co to je HTML5

Je užitečné dozvědět se několik skutečností o původu jazyka HTML, abyste lépe pochopili jazyk HTML5. Jazyk HTML vznikl na počátku 90. let minulého století v podobě stručného dokumentu popisujícího několik elementů používaných pro tvorbu webových stránek. Spousta těchto elementů popisovala různé části webové stránky; například záhlaví, odstavce a seznamy. Číslo verze jazyka HTML se postupně zvyšovalo, jak se tento jazyk rozrůstal o nové elementy a přizpůsoboval se novým potřebám. Nejnovější verzí tohoto jazyka je HTML5.

Jazyk HTML5 se přirozeně vyvinul ze starších verzí jazyka HTML, přičemž se snaží reflektovat potřeby současných i budoucích webových stránek. Tento jazyk zdědil velkou část vlastností svých předchůdců. To znamená, že pokud už jste vyvíjeli stránky v jazyce HTML před příchodem verze HTML5, znáte již podstatnou část jazyka HTML5. Dalším důsledkem je, že převážná část funkcí jazyka HTML5 funguje rovněž ve starších webových prohlížečích. Zpětná kompatibilita byla klíčovou vlastností návrhu jazyka HTML5 (více informací se můžete dozvědět na adrese *http://www.w3.org/TR/html-design-principles/*).

Jazyk HTML5 samozřejmě přináší také spoustu nových funkcí. Některé z nich jsou velmi jednoduché; kupříkladu doplňkové elementy pro popis obsahu:

- `article` (článek),
- `section` (část),
- `figure` (obrázek),
- a další.

Jiné funkce jsou složitější a pomáhají s tvorbou propracovaných webových aplikací. Musíte opravdu důkladně proniknout do vývoje webových stránek, než budete moct přejít ke složitějším funkcím jazyka HTML5. Jazyk HTML5 také nově zavádí podporu přehrávání zvukových souborů a videosouborů přímo v moderních webových prohlížečích, a to bez nutnosti instalace doplňků. Této problematice se tato kniha bude rovněž věnovat.

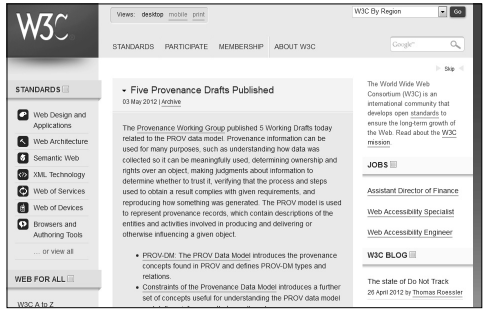
Co to je CSS3

První verze jazyka CSS se objevily až několik let po vzniku jazyka HTML, přičemž oficiálně se tento jazyk prosadil v roce 1996. Vztah mezi jazykem CSS3 a jeho staršími verzemi je analogický ke vztahu jazyka HTML5 s jeho předchůdci – jazyk CSS3 je přirozeným rozšířením svých starších verzí.

Jazyk CSS3 je mnohem mocnější než jeho předchůdci. Zavádí totiž několik vizuálních efektů, zaoblené rohy a přechody. Podrobnější informace o tom, co se v této knize dozvíte o jazyku CSS3, najdete v části „Co se v této knize naučíte."

Webové standardy a specifikace

Možná vás zajímá, kdo vytvořil jazyky HTML a CSS a dále pokračuje v jejich vývoji. Vývojem webových standardů se zabývá společenství W3C (World Wide Web Consortium), vedené Timem Berners-Leem, jenž vynalezl web a jazyk HTML. **Specifikace** jsou dokumenty, které definují parametry jazyků; například jazyků HTML a CSS. Veškerou aktivitu společenství můžete sledovat na adrese *http://www.w3.org/* (viz obrázek Ú.1).



Obrázek Ú.1. Webové stránky společenství W3C jsou hlavním zdrojem specifikací webových standardů

Z nejrůznějších důvodů vyvíjí specifikaci jazyka HTML5 jiná společnost, a to skupina WHATWG (se svými oficiálními webovými stránkami na adrese *http://www.whatwg.org/*). Společenství W3C zapojuje výsledky práce skupiny WHATWG do oficiální pracovní verze své specifikace.

Jestliže se řídíme standardy, můžeme postavit své webové stránky na sadě dohodnutých pravidel. Webové prohlížeče (kupříkladu prohlížeč Chrome, Firefox, Internet Explorer, Opera nebo Safari) potom můžou zobrazovat webové stránky s ohledem na tato pravidla. Webové prohlížeče implementují standardy vcelku dobře, ačkoliv některé starší verze prohlížečů s nimi měli poměrně problémy, a to zejména prohlížeč Internet Explorer 6.

Specifikace od společenství W3C procházejí několika fázemi vývoje, než je lze považovat za finální. V tomto okamžiku je společenství W3C přejmenovává na **doporučení** (*http://www.w3.org/2005/10/Process-20051014/tr*).

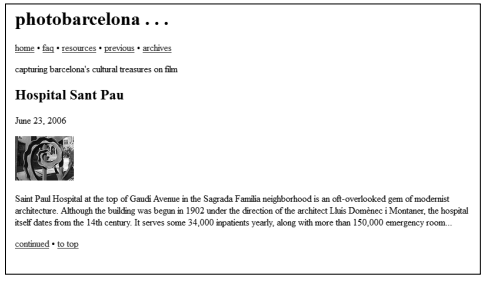
Společenství W3C stále neprohlásilo určité části specifikací jazyků HTML5 a CSS3 za finální, ale to neznamená, že je nemůžeme používat. Dokončení celého standardizačního procesu trvá nějaký čas (řádově roky). Webové prohlížeče implementují různé části specifikace mnohem dříve, než se z ní stane doporučení, jelikož skutečná implementace pomáhá utvářet samotnou specifikaci při jejím vývoji. Proto webové prohlížeče obsahují převážnou část funkcí ze specifikací jazyků HTML5 a CSS3, ačkoliv z nich ještě nejsou doporučení.

Funkce zapisované v této knize jsou již pevně zakořeněné v příslušných specifikacích, takže riziko, že se změní, než se z těchto specifikací stanou doporučení, je minimální. Vývojáři už používají spoustu funkcí jazyků HTML5 a CSS3 poměrně dlouho, takže vy můžete také.

Postupné vylepšování – praxí osvědčený postup

Úvod jsme zahájili povídáním o univerzálnosti webu – o převládajícím názoru, že informace na webu by měly být dostupné všem. **Postupné vylepšování** (**progressive enhancement**) nám pomáhá vytvářet webové stránky s ohledem na univerzálnost. Jedná se o přístup k tvorbě webových stránek, který vynalezl Steve Champeon v roce 2003 (*http://en.wikipedia.org/wiki/Progressive_enhancement*).

Princip tohoto přístupu je jednoduchý, ale o to více účinný – začínáme vytvářet obsah v jazyce HTML a chování tak, aby byly naše webové stránky přístupné všem návštěvníkům (jako na obrázku Ú.2). Na tuto stránku posléze aplikujeme design pomocí jazyka CSS (viz obrázek Ú.3) a přidáme dodatečné funkce kódu jazyka JavaScript. Obojí obvykle doplňujeme prostřednictvím externích souborů (jak na to, si ukážeme později).



Obrázek Ú.2. Základní stránka HTML bez stylů jazyka CSS. Stránka sice nevypadá příliš dobře, ale informace jsou dostupné, a to je důležité. Dokonce i prohlížeče z období blízkého vzniku webu před 20 lety dokážou zobrazit tuto stránku. Stejně tak ji umí zobrazit nejstarší mobilní telefony s webovými prohlížeči. Kromě toho se v této stránce vyznají skvěle také nástroje pro předčítání textu (někdy označované termínem screen reader)

Výsledkem je, že uživatelům se zařízeními a prohlížeči schopnými zobrazovat pouze základní stránky nachystáme zjednodušený uživatelský prožitek, zatímco ostatní uživatelé si užijí naše webové stránky plnými doušky. Uživatelský prožitek z našich webových stránek totiž nemusí být srovnatelný pro každého, dokud je přístupný jejich obsah. Základní myšlenou za přístupem postupného vylepšování je, že každý by měl být vítězem.

Obrázek Ú.3. Stejná stránka jako na předchozím obrázku, avšak otevřená v prohlížeči s podporou jazyka CSS. Obsahuje naprosto stejné informace, ale prezentuje je jinak. Uživatelé se schopnějšími zařízeními a prohlížeči získají lepší uživatelský prožitek při návštěvě této stránky

V této knize se naučíte vyvíjet webové stránky s ohledem na postupné vylepšování, přestože tento fakt tato kniha neustále nezmiňuje. Jedná se totiž o přirozený důsledek praxí osvědčených postupů v ní uvedených.

V kapitolách 12, „Kaskádové styly pro rozličná zařízení" a 14, „Vylepšení v jazyce CSS3," se ale budete zabývat postupným vylepšováním přímo. Můžete do nich však nakouknout už teď, jestliže vás zajímá, jak můžou být principy postupného vylepšování nápomocné při stavbě webových stránek, které se umí přizpůsobovat velikosti obrazovky daného zařízení a dovednostem webového prohlížeče, nebo jak starší webové stránky zobrazí zjednodušenou verzi vašich webových stránek, zatímco moderní webové prohlížeče zobrazí variantu doplněnou o efekty jazyka CSS3.

Postupné vylepšování je klíčovým přístupem, kterým by se měl při tvorbě webových stránek řídit opravdu každý.

Je tato kniha pro vás vhodná?

Tato kniha nepředpokládá žádné předchozí znalosti tvorby webových stránek. V tomto ohledu je tudíž určena pro naprosté začátečníky. Naučíte se jazyky HTML a CSS od úplných základů. V průběhu studia se dozvíte rovněž, jaké funkce jsou nové ve verzích HTML5 a CSS3, přičemž hlavní důraz tato kniha klade na ty z nich, které návrháři a vývojáři používají v každodenní praxi.

Ale dokonce i tehdy, když už znáte jazyky HTML a CSS, můžete se v této knize něčemu přiučit. Zejména v případě, že se nechcete ztratit v posledních novinkách z jazyků HTML5 a CSS3 a v praxi osvědčených postupech.

Co se v této knize naučíte

Krátce řečeno – toto vydání knihy je velmi rozsáhlou revizí.

Kapitoly jsou v této knize uspořádané následovně:

- V kapitolách 1 až 6 a v kapitolách 15 až 18 se budeme zabývat principy vytváření stránek HTML a celou škálou dostupných elementů jazyka HTML, přičemž si ukážeme kdy a jak je používat.
- V kapitolách 7 až 14 pronikneme do jazyka CSS. Začneme tvorbou prvního pravidla stylu a skončíme uplatněním vylepšených vizuálních efektů s pomocí jazyka CSS3.
- V kapitole 19 si ukážeme, jak přidat hotový kód jazyka JavaScript do svých stránek.
- V kapitole 20 si popíšeme, jak testovat a ladit své stránky předtím, než je zveřejníme na webu.
- V kapitole 21 si vysvětlíme, jak si sehnat své vlastní doménové jméno a následně publikovat stránky na webu pro širokou veřejnost.

U výše zmíněných témat nezapomeneme ani na následující oblasti:

- Vytváření, ukládání a úpravu souborů s kódem jazyka HTML a CSS.
- Co to znamená psát sémantické dokumenty HTML a proč je to tak důležité.
- Jak oddělovat obsah stránky (doména jazyka HTML) od jeho vzhledu (revír jazyka CSS), což je hlavní aspekt strategie postupného vylepšování.
- Uspořádávání našeho obsahu smysluplným způsobem prostřednictvím již zažitých elementů jazyka HTML i těch nových v jazyce HTML5.
- Vylepšování přístupnosti webových stránek pomocí rolí ARIA a řady jiných dobrých programátorských postupů.
- Přidávání obrázků do stránek a jejich optimalizace pro web.
- Odkazování se z jedné stránky na druhou nebo z jedné části stránky na jinou část.
- Stylování textu (velikost, barva, tučné písmo, kurzíva apod.); nastavování barvy a obrázku na pozadí; implementace flexibilního vícesloupcového rozvržení, které se umí přizpůsobovat různým rozměrům obrazovky.
- Používání nových selektorů jazyka CSS3, které umožňují aplikovat naše styly mnohem více způsoby než u předchozích verzí tohoto jazyka.
- Jaké jsou možnosti pro obsloužení uživatelů na mobilních zařízeních.
- Tvorba univerzálních webových stránek pro všechny uživatele. Ať už používají mobilní telefon, tablet, notebook, desktopový počítač nebo jiné zařízení s podporou webu. Toho dosáhneme pomocí principů **responzivního web designu**, z nichž některé se opírají o **dotazy na médium** (anglicky **media queries**) jazyka CSS3.
- Doplnění webových stránek o vlastní webová písma pomocí pravidla `@font-face`.
- Používání efektů jazyka CSS3, jako jsou průhlednost, průhlednost pozadí založená na alfa kanálu, barevné přechody, zaoblené rohy, vržené stíny, stíny uvnitř elementů, stíny textu a více obrázků na pozadí.
- Vytváření formulářů žádajících vstupní data od uživatelů, a to včetně některých nových typů vstupních polí z jazyka HTML5.
- Začleňování multimédií do našich stránek s použitím elementů `audio` a `video` jazyka HTML5.
- A ještě mnohem více.

Všechna tato témata jsou doprovázena spoustou ukázkových zdrojových kódů, které demonstrují, jak tyto funkce implementovat na základě praxí osvědčených postupů.

Co se v této knize nenaučíte

Ačkoliv má tato kniha spoustu nových stránek oproti svému dřívějšímu vydání, jakmile přijde řeč na jazyky HTML a CSS, lze mluvit téměř donekonečna, proto bohužel některá témata chybí.

Až na několik výjimek schází témata, která byste zřídka využili v praxi, mohou se stále měnit, chybí jim rozsáhlejší podpora ve webových prohlížečích, vyžadují znalosti jazyka JavaScript, nebo jsou příliš pokročilá.

K těmto tématům se řadí kupříkladu:

- Elementy `details`, `summary`, `menu`, `command` a `keygen` jazyka HTML5.
- Element `canvas`, s nímž je možné kreslit (a dokonce vytvářet hry) v jazyce JavaScript.
- Aplikační rozhraní jazyka HTML5 a další pokročilé funkce, které vyžadují znalost jazyka JavaScript, nebo se nevztahují přímo k novým sémantickým elementům jazyka HTML5.
- **Sprajty** jazyka CSS. Tato technika spočívá ve skládání více obrázků do jediného obrázku, což je velmi užitečné pro snížení počtu prostředků načítaných do našich webových stránek. Více informací najdete například na adrese *http://blog.proteus.cz/optimalizace-rychlosti-webu-snizeni-poctu-http-pozadavku*.
- Nahrazování textu obrázkem (**image replacement**) v jazyce CSS. Vývojáři kombinují často tuto techniku se sprajty. Více informací o této technice je k dispozici na adrese *http://css-tricks.com/css-image-replacement/*.
- Transformace, animace a přechody jazyka CSS3.
- Nové moduly rozvržení jazyka CSS3.

Z čeho se tato kniha skládá

V téměř každé části této knihy najdete praktický příklad v podobě výpisu zdrojového kódu, jenž ukazuje, jak byste probírané téma použili v praxi (viz výpisy Ú.1 a Ú.2). Výpisy zdrojového kódu budou obvykle doprovázeny nasnímanými obrazovkami, které demonstrují, jak budou výsledky programování vypadat, když si je prohlédnete ve webovém prohlížeči (viz obrázek Ú.4).

Většina nasnímaných obrazovek pochází z poslední verze prohlížeče Firefox dostupné v době psaní této knihy. To však není doporučení, abyste upřednostňovali prohlížeč Firefox před jinými webovými prohlížeči. Ukázkové příklady budou vypadat velmi podobně v posledních verzích prohlížečů Chrome, Internet Explorer, Opera a Safari. Jak se dozvíte v kapitole 20, „Testování a ladění webových stránek," své webové stránky byste měli testovat v široké škále webových prohlížečů, než je vypustíte na web, protože nemáte vůbec žádnou jistotu, jaké webové prohlížeče budou vaši návštěvnici používat.

Kód a nasnímané obrazovky doplňují také popisy použitých elementů jazyka HTML nebo vlastností jazyka CSS, abyste věděli, co se v nich nachází a lépe jim porozuměli.

V mnoha případech vám zajisté postačí přečíst si popisy a ukázkové kódy, abyste mohli začít používat příslušné funkce jazyků HTML a CSS. Pokud byste však potřebovali vodítko, jak je používat, vždy budou k dispozici podrobné návody.

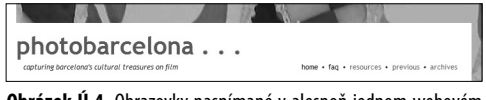
Na mnoho místech se rovněž setkáte s tipy nabízejícími dodatečné informace, praxí osvědčenými postupy, odkazy na související části knihy, odkazy na příbuzné zdroje apod.

Výpis Ú.1. Výpis kódu jazyka HTML najdete na spoustě stránek. Relevantní části kódu jsou zvýrazněné. Tři tečky reprezentují dodatečný kód nebo obsah, který chybí kvůli zachování stručnosti. Chybějící části kódu se obvykle zobrazují v jiném výpisu

```
...
<body>
<header role="banner">
  ...
    <nav role="navigation">
      <ul class="nav">
        <li><a href="/" class="current">
          home</a></li>
        <li><a href="/faq/">faq</a></li>
        <li><a href="/resources/">
          resources</a></li>
        <li><a href="/previous/">
          previous</a></li>
        <li><a href="/archives/">
          archives</a></li>
      </ul>
    </nav>
  ...
</header>
...
</body>
</html>
```

Výpis Ú.2. Pokud je kód jazyka CSS pro příklad důležitý, zobrazuje se ve svém samostatném výpisu, přičemž podstatné části jsou opět zvýrazněné

```
/* Navigace na webových stránkách */
.nav li {
  float: left;
  font-size: .75em;
    /* zmenšuje odrážky */
}

.nav li a {
  font-size: 1.5em;
}

.nav li:first-child {
  list-style: none;
  padding-left: 0;
}
```



Obrázek Ú.4. Obrazovky nasnímané v alespoň jednom webovém prohlížeči demonstrují, jak daný kód ovlivňuje vzhled dané stránky

Konvence používané v této knize

Tato kniha používá následující konvence:

- Text nebo kód, který byste měli nahradit vlastní hodnotou, se vypisuje kurzívou. Většina těchto zástupných výrazů se objevuje v podrobných návodech. Například ve větě: „Případně napište `#rrggbb`, přičemž výraz `rrggbb` představuje šestnáctkovou reprezentaci barvy."
- Zdrojový kód, který byste měli napsat, dále pak kód jazyka HTML a CSS, se píše ve větě `tímto neproporcionálním písmem`.
- První výskyt termínu se vypisuje **tučným písmem**.

Zpětná vazba od čtenářů

Nakladatelství a vydavatelství Computer Press, které pro vás tuto knihu přeložilo, stojí o zpětnou vazbu a bude na vaše podněty a dotazy reagovat. Můžete se obrátit na následující adresy:

Computer Press
Albatros Media a.s., pobočka Brno
IBC
Příkop 4
602 00 Brno

nebo

sefredaktor.pc@albatrosmedia.cz

Computer Press neposkytuje rady ani jakýkoli servis pro aplikace třetích stran. Pokud budete mít dotaz k programu, obraťte se prosím na jeho tvůrce.

Zdrojové kódy ke knize

Z adresy *http://knihy.cpress.cz/K2036* si po klepnutí na odkaz Soubory ke stažení můžete přímo stáhnout archiv s ukázkovými kódy.

Errata

Přestože jsme udělali maximum pro to, abychom zajistili přesnost a správnost obsahu, chybám se úplně vyhnout nelze. Pokud v některé z našich knih najdete chybu, ať už chybu v textu nebo v kódu, budeme rádi, pokud nám ji oznámíte. Ostatní uživatele tak můžete ušetřit frustrace a pomoci nám zlepšit následující vydání této knihy.

Veškerá existující errata zobrazíte na adrese *http://knihy.cpress.cz/K2036* po klepnutí na odkaz Soubory ke stažení.

Stavební kameny webových stránek

1

Přestože webové stránky jsou čím dál složitější, jejich základní struktura zůstává pozoruhodně jednoduchá. První věcí, kterou bychom si měli uvědomit, je, že není možné vytvořit webovou stránku bez jazyka HTML. Jak se naučíme později, kód jazyka HTML uchovává obsah stránky a popisuje jeho význam. Webové prohlížeče posléze zobrazují tento obsah obalený do značek jazyka HTML uživatelům.

Webová stránka se v podstatě skládá ze tří komponent:

- **Textový obsah**: prostý text zobrazovaný na stránce za tím účelem, abychom informovali návštěvníky o svých plánech, rodinné dovolené, produktech nebo jiném tématu, na něž se naše stránka zaměřuje.
- **Odkazy na jiné soubory**: slouží k nahrávání obrázkových, zvukových a videosouborů; dále se odkazují na jiné stránky HTML a další prostředky, a stejně tak na šablony stylů (ovlivňující vzhled stránky) a soubory s kódem jazyka JavaScript (ovlivňující chování stránky).
- **Značky**: značky jazyka HTML, s nimiž popisujeme náš textový obsah a definujeme odkazy. Písmeno M v názvu HTML totiž zkracuje slovo **markup**, neboli v češtině **značkovací**.

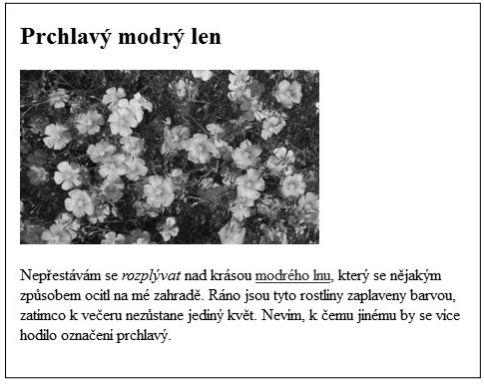
Měli bychom však poznamenat, že všechny tyto komponenty se skládají výhradně z textu. To znamená, že webové stránky ukládáme ve formátu prostého textu a můžeme je prohlížet prakticky v každém webovém prohlížeči na libovolné platformě. Je jedno, zda se jedná o desktopový počítač, mobilní telefon, tablet nebo něco jiného. To zaručuje univerzálnost webu. Stránka může sice vypadat na jednom zařízení jinak než na druhém, ale to je v pořádku. Důležitým prvním krokem je učinit obsah přístupný všem uživatelům, což jazyk HTML umožňuje.

Kromě těchto tří hlavních komponent obsahuje webová stránka ještě kód jazyka HTML s informacemi o ní samotné. Většinu z těchto informací uživatelé nevidí, jsou totiž určené spíše pro webové prohlížeče a vyhledávací roboty. Patří mezi ně kupříkladu primární jazyk obsahu (čeština, angličtina apod.), znaková sada textu (nejrozšířenější je znaková sada UTF-8) atd.

V této kapitole si projdeme tvorbu základní stránky HTML, přičemž si ukážeme několik praxí osvědčených postupů a vysvětlíme si každou z výše popsaných tří komponent.

Základní stránka HTML

Prohlédněte si základní stránku HTML, abyste získali představu, co bude následovat v této kapitole a v následujících kapitolách. Obrázek 1.1 ukazuje, jak webový prohlížeč přibližně zobrazuje kód jazyka HTML z výpisu 1.1. Dozvíte se několik základních poznatků o tomto kódu, ale nezoufejte, jestliže ho nepochopíte celý. Tento výpis pouze demonstruje, jak vypadá kód jazyka HTML. Na naučení jazyka HTML máte celý zbytek této knihy.



Obrázek 1.1. Typický výchozí vzhled naší stránky. Třebaže tuto stránku zobrazil prohlížeč Firefox, v ostatních prohlížečích by vypadala podobně

Výpis 1.1. Kód základní stránky HTML. Značky jazyka HTML jsou zvýrazněné, aby je bylo možné odlišit od textového obsahu stránky. Jak je patrné z obrázku 1.1, kód jazyka HTML, jenž obklopuje textový obsah, se v prohlížeči nezobrazuje. Jak ale zjistíme později, tento kód je nezbytný, jelikož popisuje význam obsahu. Spoustu značek jazyka HTML jsme zalomili na samostatné řádky – to není nutné, ale na druhou stranu to nijak neovlivňuje vzhled stránky

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="utf-8" />
  <title>Modrý len (Linum lewisii)
  </title>
</head>
<body>
  <article>
    <h1>Prchlavý modrý len</h1>
    <img src="modrylen.jpg"
      width="300" height="175"
      alt="Modrý len (Linum lewisii)" />

    <p>Nepřestávám se <em>rozplývat</em>
      nad krásou <a href="http://
      en.wikipedia.org/wiki/Linum_lewisii"
      rel="external"
      title="Více informací o modrém lnu">
      modrého lnu</a>, který se nějakým
      způsobem ocitl na mé zahradě. Ráno
      jsou tyto rostliny zaplaveny
      barvou, zatímco k večeru nezůstane
      jediný květ. Nevím, k čemu jinému
      by se více hodilo označení prchlavý.
    </p>
  </article>
</body>
</html>
```

Není těžké uhádnout, co se v tomto kódu odehrává, a to především v části `body`. Podívejme se ale nejprve na část před elementem `body`.

Všechno, co se nachází před počáteční značkou `<body>`, jsou informace určené prohlížečům a vyhledávacím robotům (viz výpis 1.2), jak jsme si říkali dříve. Každou stránku zahajujeme deklarací `DOCTYPE`, kterou sdělujeme webovému prohlížeči, jakou verzi jazyka HTML používáme.

Výpis 1.2. Text uvnitř elementu `title` je jedinou částí záhlaví dokumentu HTML, kterou uživatel uvidí. Zbylé informace jsou určené pouze pro webové prohlížeče a vyhledávací roboty

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="utf-8" />
  <title>Modrý len (Linum lewisii)
  </title>
</head>
```

Vždy bychom měli definovat typ dokumentu jazyka HTML5, což je `<!DOCTYPE html>`. Na velikosti písmen nezáleží, ale obvykle se používají velká písmena ve slově `DOCTYPE`. Na definici typu dokumentu bychom ale neměli zapomínat. (Více informací se nachází

v části „Vylepšená definice typu dokumentu u jazyka HTML5“ v kapitole 3, „Základní struktura dokumentu HTML.“)

Údaje mezi značkou `<!DOCTYPE html>` a koncovou značkou `</head>` jsou uživateli skryté, až na jednu výjimku – text mezi značkami `<title>` a `</title>`, v tomto případě `Modrý len (Linum lewisii)`, se zobrazuje v záhlaví okna webového prohlížeče nebo jako název panelu (viz obrázek 1.2). Kromě toho používají webové prohlížeče tento text pro názvy záložek a oblíbených položek, a také se jedná o užitečnou informaci pro vyhledávací roboty. V kapitole 3, „Základní struktura dokumentu HTML“ si vysvětlíme, co dělají ostatní části záhlaví dokumentu.

Obsah stránky ale uživatelé vidí, a ten se nachází mezi počáteční značkou `<body>` a koncovou značkou `</body>`. Poslední značkou je koncová značka `</html>`, se kterou ukončujeme naši stránku (viz výpis 1.3).

Výpis 1.3. Obsah stránky se nachází mezi počáteční a koncovou značkou elementu `body`. Na konci dokumentu lze najít koncovou značku `</html>`

```
<!DOCTYPE html>
<html lang="cs">
... záhlaví dokumentu ...
<body>
  <article>
    <h1>Prchlavý modrý len</h1>

    <img src="modrylen.jpg"
      width="300" height="175"
      alt="Modrý len (Linum lewisii)" />

    <p>Nepřestávám se <em>rozplývat</em>
      nad krásou <a href="http://
      en.wikipedia.org/wiki/Linum_lewisii"
      rel="external"
      title="Více informací o modrém lnu">
      modrého lnu</a>, který se nějakým
      způsobem ocitl na mé zahradě.
      Ráno jsou tyto rostliny zaplaveny
      barvou, zatímco k večeru nezůstane
      jediný květ. Nevím, k čemu jinému
      by se více hodilo označení prchlavý.
    </p>
  </article>
</body>
</html>
```

Odsazování zdrojového kódu nemá vůbec žádný vliv na validitu kódu jazyka HTML. Rovněž neovlivňuje, jak se obsah zobrazuje ve webovém prohlížeči. Jedinou výjimkou, u které to neplatí, je obsah elementu `pre`, o němž se dozvíme více v kapitole 4, „Text.“ Je však na rozhodnutí každého vývojáře, zda bude odsazovat kód vnořený do svého rodičovského elementu, aby bylo celé hierarchické uspořádání zřetelné na první pohled při čtení kódu. O rodičích a potomcích si povíme více později v této kapitole. Taktéž si podrobněji rozebereme standardní způsob zobrazování stránky ve webovém prohlížeči.

Nejprve si ale vysvětleme, co to znamená psát sémantický kód jazyka HTML a proč se jedná o základ efektivních webových stránek.

Sémantický kód jazyka HTML

Za jazykem HTML stojí promyšlený systém vkládání informací o obsahu do textového dokumentu. Tyto informace nazýváme **značky** a popisujeme s nimi **význam** obsahu, neboli jeho **sémantiku**. S několika příklady už jsme se setkali u naší základní stránky HTML; kupříkladu s elementem `p`, který označuje odstavec.

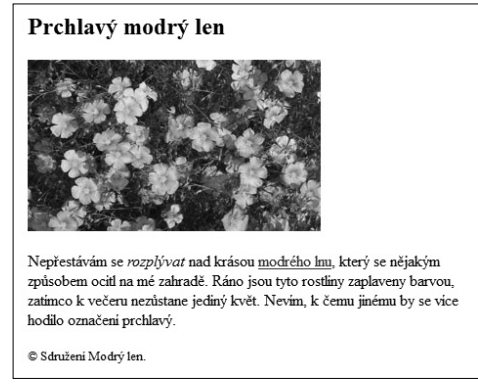
Jazykem HTML nespecifikujeme, jak by měl obsah vypadat v prohlížeči; to je úkolem jazyka CSS (kaskádových stylů). Jazyk HTML5 klade na tuto skutečnost mnohem větší důraz než starší verze tohoto jazyka. Je přímo zakořeněná v jádru tohoto jazyka.

Možná se divíte, proč je některý text na základní stránce HTML (viz výpis 1.4) větší, tučný nebo se zobrazuje kurzívou (viz obrázek 1.2).

To je vynikající otázka. Důvodem je, že každý webový prohlížeč má svou vestavěnou šablonu stylů jazyka CSS, která určuje, jak by se měly jednotlivé elementy jazyka HTML zobrazovat, dokud nenapíšeme vlastní styly, kterými tyto výchozí přepíšeme. Výchozí vzhled elementů se mírně liší v jednotlivých prohlížečích, ale ve výsledku je všude velmi podobný. Co je však důležitější, je to, že struktura a význam obsahu definovaného jazykem HTML se nemění.

Výpis 1.4. Obsah naší základní stránky obohacený o druhý odstavec na konci. Pomocí elementů jazyka HTML nediktujeme, jak by se měl obsah zobrazovat, ale co znamená. Vestavěná šablona stylů webového prohlížeče určuje, jak by měl vypadat

```
...
<body>
  <article>
    <h1>Prchlavý modrý len</h1>

    <img src="modrylen.jpg"
      width="300" height="175"
      alt="Modrý len (Linum lewisii)" />

    <p>Nepřestávám se <em>rozplývat</em>
      nad krásou <a href="http://
      en.wikipedia.org/wiki/Linum_lewisii"
      rel="external"
      title="Více informací o modrém lnu">
      modrého lnu</a>, který se nějakým
      způsobem ocitl na mé zahradě.
      Ráno jsou tyto rostliny zaplaveny
      barvou, zatímco k večeru nezůstane
      jediný květ. Nevím, k čemu jinému
      by se více hodilo označení prchlavý.
    </p>

    <p><small>&copy; Sdružení Modrý len.
      </small></p>
  </article>
</body>
</html>
```



Obrázek 1.2. Výchozí šablona stylů webového prohlížeče zobrazuje nadpisy (elementy `h1` až `h6`) jinak než normální text, dále text uvnitř elementu `em` zobrazuje kurzívou a odkazy zobrazuje jinou barvou a podtržené. Některé elementy začínají na samostatném řádku (například element `h1` nebo `p`), zatímco jiné obklopuje další obsah (kupříkladu element `a` nebo `em`). Tento příklad obsahuje druhý odstavec (s oznámením o copyrightu), aby bylo zřejmé, že každý odstavec začíná na novém řádku. Ve vlastních šablonách stylů můžeme přepsat libovolné z těchto pravidel

Blokové a řádkové elementy a jazyk HTML5

Jak je patrné, některé elementy jazyka HTML (například elementy `article`, `h1` a `p`) začínají na samostatném řádku stejně jako odstavce v této knize, zatímco jiné se zobrazují na stejném řádku (kupříkladu element `a` nebo `em`) jako jiný obsah (viz obrázek 1.2). Ještě jednou si zdůrazníme – toto je výsledkem výchozí šablony stylů prohlížeče, a ne samotných elementů jazyka HTML. Než vznikl jazyk HTML5, elementy se tradičně dělily na **blokové** (začínají na samostatném řádku) a **řádkové** elementy (pokračují na stejném řádku). V jazyce HTML5 se od těchto termínů upouští, protože spojují elementy s jejich prezentací, ale to není cílem jazyka HTML.

Jednoduše řečeno – elementy označované dříve jako řádkové se v jazyce HTML5 řadí do kategorie **formulační obsah**. Do této kate-

gorie spadá text a elementy, které ho označují, a to především na úrovni odstavce. V kapitole 4, „Text," se zaměříme výhradně na formulační obsah. Úplný seznam elementů z této kategorie je možné najít na adrese *http://dev.w3.org/html5/spec-author-view/content-models.html#phrasing-content-0*.

Původní blokové elementy nyní také spadají do nových kategorií na základě jejich významu. Velkou část těchto elementů tvoří hlavní strukturní bloky a nadpisy našeho obsahu. V kapitole 3, „Základní struktura dokumentu HTML," se naučíte více o elementech z kategorií **rozdělující obsah** a **nadpisový obsah**.

I přes tyto změny webové prohlížeče nezměnily výchozí pravidla zobrazování pro tyto elementy, protože ani nemusely. Stále totiž nechceme kupříkladu, aby dva odstavce (elementy `p`) navazovaly jeden na druhý, nebo aby element `em` (lehce zdůrazněný text) zalomil větu a zobrazil se na samostatném řádku.

To znamená, že nadpisy, odstavce a strukturní elementy (například element `article`) začínají typicky na novém řádku, zatímco formulační obsah se zobrazuje na stejném řádku jako jeho obklopující obsah. Ačkoliv v jazyce HTML5 nepoužíváme termíny blokový a řádkový element, je užitečné vědět, jaký je jejich význam. Ve většině návodů se s nimi můžeme setkat, protože byly zakořeněné v terminologii jazyka HTML velmi dlouhou dobu před příchodem verze HTML5. V této knize na ně rovněž občas narazíme, protože s jejich pomocí snadno poznáme, jestli element standardně obsazuje samostatný řádek, nebo sdílí řádek s dalším obsahem.

Podrobné informace o jazyku CSS najdete až v pozdějších kapitolách, ale prozatím vězte, že šablona stylů je obyčejný textový soubor, proto ho můžete vytvářet ve stejném textovém editoru jako svou stránku jazyka HTML.

Důraz na sémantiku v jazyce HTML5

Jazyk HTML5 klade důraz na sémantiku, přičemž veškerou změnu vzhledu nechává na jazyk CSS. U starších verzí jazyka HTML tomu tak ale vždy nebylo.

Správné prostředky pro stylování stránek neexistovaly v raných fázích webu; jazyk HTML už byl několik let na světě, když byl oficiálně zveřejněn jazyk CSS1 v roce 1996. Aby si s tímto nedostatkem jazyk HTML nějak poradil, obsahoval několik prezentačních elementů, jejichž účelem bylo umožnit základní stylování textu – psát text tučným písmem, kurzívou nebo měnit jeho velikost.

Tyto elementy sloužily v této době svému účelu, ale vývojáři je začali opouštět, když se objevily nové praxí osvědčené postupy pro vývoj webových stránek. Jejich hlavní myšlenka spočívala (a stále spočívá) v tom, že jazyk HTML by měl jen popisovat obsah, a ne ho zobrazovat.

Prezentační elementy jazyka HTML byly v přímém rozporu s touto myšlenkou, proto je jazyk HTML 4 označil jako zastaralé a autorům stránek doporučoval stylovat své stránky pomocí jazyka CSS.

Jazyk HTML5 jde ještě dál – zcela odstranil prezentační elementy a předefinoval některé ostatní, aby byly pouze sémantické, a nediktovaly, jak by se měl obsah zobrazovat.

Ukázkovým příkladem je element `small`. Tento element původně sloužil pro zobrazení textu menším písmem, než je běžná velikost. V jazyce HTML5 ale označuje poznámku tištěnou drobný písmem, jako je kupříkladu zákonné vzdání se odpovědnosti. S pomocí jazyka CSS bychom z něj mohli udělat největší text na stránce, kdybychom chtěli, ale to nic nemění na jeho významu.

Historický protějšek elementu `small`, element `big`, v jazyce HTML5 neexistuje. Existuje spousta dalších příkladů, s nimiž se budeme setkávat v průběhu této knihy.

Jazyk HTML5 zavádí rovněž nové elementy. Například element `header`, `footer`, `nav`, `article`, `section` a spoustu dalších, které obohacují významovou stránku obsahu. Tyto elementy si popíšeme později.

Ať už používáme element, který v jazyce HTML existoval odjakživa, nebo úplně nejnovější element, hlavní cíl zůstává stejný – měli bychom si vybrat element, který nejlépe vystihuje význam obsahu, aniž bychom se starali o jeho vzhled.

Význam naší základní stránky HTML

Když už víme, co je hlavním cílem jazyka HTML, podívejme se podrobněji na myšlenkový proces stojící v pozadí při označování našeho ukázkového obsahu. Jak zjistíme, psát sémantický kód jazyka HTML není vůbec složité. Jakmile se seznámíme s dostupnými elementy, jedná se spíše o používání zdravého selského rozumu. Zopakujme si, jak vypadá element `body` naší základní stránky, na němž si ukážeme několik nejčastěji používaných elementů jazyka HTML (viz výpis 1.5).

Výpis 1.5. Element `body` naší základní stránky obsahuje elementy `article`, `h1`, `img`, `p`, `em` a `a`, s nimiž popisujeme význam obsahu. Veškerý tento obsah je vnořený do elementu `article`.

```
<body>
  <article>
    <h1>Prchlavý modrý len</h1>

    <img src="modrylen.jpg"
      width="300" height="175"
      alt="Modrý len (Linum lewisii)" />

    <p>Nepřestávám se <em>rozplývat</em>
      nad krásou <a href="http://
      en.wikipedia.org/wiki/Linum_lewisii"
      rel="external"
      title="Více informací o modrém lnu">
      modrého lnu</a>, který se nějakým
      způsobem ocitl na mé zahradě.
      Ráno jsou tyto rostliny zaplaveny
      barvou, zatímco k večeru nezůstane
      jediný květ. Nevím, k čemu jinému
      by se více hodilo označení prchlavý.
    </p>

    <p><small>&copy; Sdružení Modrý len.
      </small></p>
  </article>
</body>
```

Všechen obsah se nachází uvnitř elementu `article`. Elementem `article` definujeme výrazný kus obsahu. Jedná se o vhodnou volbu pro obalení obsahu naší základní stránky, ale ne nezbytně všech stránek, které napíšeme. Více informací o tom, kdy používat element `article`, se dozvíme v kapitole 3, „Základní struktura dokumentu HTML."

Prvním elementem, jenž se nachází uvnitř elementu `article`, je nadpis (viz výpis 1.6). Jazyk HTML nabízí šest úrovní nadpisů označovaných elementy `h1` až `h6`, přičemž element `h1` představuje nadpis první úrovně. Element `h2` je podnadpis elementu `h1`, dále pak element `h3` je podnadpisem elementu `h2` atd. Jedná se o podobný princip strukturování nadpisů jako u běžných dokumentů, které píšeme v některém textovém procesoru.

Výpis 1.6. Nadpisy jsou důležité elementy pro shrnutí obsahu stránky. S nadpisy jsou stránky přístupnější uživatelům a nástrojům pro předčítání textu, a navíc vyhledávací roboti na základě nich určují zaměření stránky

```
<h1>Prchlavý modrý len</h1>
```

Každá stránka by měla mít minimálně jeden nadpis první úrovně, takže element `h1` byla jasná volba. Nadpisy `h1` až `h6` si popíšeme v kapitole 3, „Základní struktura dokumentu HTML."

Následuje obrázek (viz výpis 1.7). Element `img` je jednoznačnou volbou pro označování obrázků, proto nebyly žádné pochyby o tom, jaký element zvolit. Atribut `alt` obsahuje text, který se zobrazuje, když není obrázek k dispozici nebo prohlížíme stránku v textovém prohlížeči. Obrázkům se podíváme na zoubek v kapitole 5, „Obrázky."

Výpis 1.7. Obrázek lze do stránky přidat snadno. Atributem `alt` specifikujeme, že se zobrazí text „Modrý len (Linum lewisii)," když nebude daný obrázek k dispozici

```
<img src="modrylen.jpg" width="300"
  height="175"
  alt="Modrý len (Linum lewisii)" />
```

Odstavec označujeme elementem `p` (viz výpis 1.8). Stejně jako v tištěných materiálech se odstavec skládá z jedné nebo několika vět. Kdybychom museli do naší stránky doplnit další odstavec, jednoduše bychom zapsali druhý element `p` za ten první.

Výpis 1.8. Element `p` může obsahovat další elementy, s nimiž definujeme význam frází uvnitř odstavce. Typickým příkladem jsou elementy `em` a `a`

```
<p>Nepřestávám se <em>rozplývat</em>
  nad krásou <a href="http://
  en.wikipedia.org/wiki/Linum_lewisii"
  rel="external"
  title="Více informací o modrém lnu">
  modrého lnu</a>, který se nějakým
  způsobem ocitl na mé zahradě.
  Ráno jsou tyto rostliny zaplaveny
  barvou, zatímco k večeru nezůstane
  jediný květ. Nevím, k čemu jinému
  by se více hodilo označení prchlavý.
</p>
```

Náš odstavec obsahuje dva elementy, se kterými popisujeme význam částí textu – element `em` a element `a` (viz výpis 1.8). Jedná se o ukázkové elementy z kategorie formulačního obsahu, z nichž většina vyzdvihuje sémantiku textu uvnitř odstavce. O elementech `em`, `a` a `p` si ještě povíme v kapitole 4, „Text."

Elementem `em` označujeme text, na nějž chceme klást lehký důraz. V tomto případě zdůrazňuje nadšení z krásy květin (viz výpis 1.8). Přestože element `em` určuje pouze význam textu, většina prohlížečů zobrazuje takový text kurzívou.

Nakonec definujeme na naší základní stránce odkaz na jinou stránku prostřednictvím elementu `a`, což je nejdůležitější element z celého jazyka HTML, protože dělá web webem. Spojuje totiž jednu stránku s další stránkou nebo prostředkem, a také odkazuje z jedné části stránky na jinou část stránky (ať už stejné, nebo jiné). V našem příkladu označujeme text „modrého lnu" jako odkaz na stránku encyklopedie Wikipedia (viz výpis 1.9).

Výpis 1.9. Tímto elementem `a` se odkazujeme na stránku o modrém lnu v encyklopedii Wikipedia. Volitelným atributem `rel` ještě více upřesňujeme význam tím, že říkáme, že tento odkaz ukazuje na externí webové stránky. Volitelným atributem `title` rovněž vylepšujeme sémantiku tohoto elementu pomocí informace o související stránce. Jeho text se v prohlížeči objevuje, jakmile uživatel přesune ukazatel myši nad tento odkaz

```
<a href="http://en.wikipedia.org/wiki/
Linum_lewisii" rel="external"
  title="Více informací o modrém lnu">
  modrého lnu</a>
```

To bylo jednoduché, že? Jakmile známe dostupné elementy jazyka HTML, vybrat ty správné pro náš obsah je obvykle opravdu snadné. Občas narazíme na nějaký obsah, který lze označit více způsoby, ale to je v pořádku. Ne vždy je možné rozeznat správnou a špatnou cestu okamžitě, ale většinou tomu tak je.

Jazyk HTML5 se samozřejmě nesnaží popsat všechny myslitelné typy obsahu, protože takový jazyk by byl neohrabaný. Místo toho zaujímá praktický postoj a definuje jen ty elementy, které se hodí pro většinu případů.

Jazyk HTML je z velké části sympatický z toho důvodu, že jeho základům se může naučit

opravdu každý, vytvořit nějaké vlastní stránky a dále své znalosti rozvíjet. Ačkoliv existuje přibližně 100 elementů jazyka HTML, nenechte se tímto číslem vystrašit. Pouze hrstku z nich budete běžně používat, zatímco ostatní jsou užitečné v méně obvyklých situacích. Už jste se naučili o několika běžných elementech, takže jste na správné cestě.

Proč je sémantika důležitá

Jelikož už víte, že sémantika je v jazyce HTML důležitá, zbývá se jen dozvědět **proč**.

Zde jsou některé nejdůležitější důvody. O některých z nich jsme se již bavili:

- Lepší přístupnost a schopnost spolupráce (obsah je dostupný pomocným technologiím, které využívají handicapovaní návštěvníci, dále pak ve webových prohlížečích na desktopových počítačích, mobilních telefonech, tabletech a jiných zařízeních).
- Lepší optimalizace pro vyhledávače (SEO).
- Menší zdrojový kód a rychlejší zobrazování stránek (tedy alespoň většinou).
- Snadnější údržba zdrojového kódu a lehčí stylování.

Neznáte-li termín **přístupnost**, jedná se o postup tvorby obsahu tak, aby byl přístupný všem uživatelům (více informací najdete na adrese *http://www.w3.org/standards/webdesign/accessibility*). Tim Berners-Lee (vynálezce webu) pronesl slavný výrok: „Síla webu spočívá v jeho univerzálnosti. Přístup pro každého, bez ohledu na jeho postižení, je jeho základním aspektem."

Jakékoliv zařízení s webovým prohlížečem umí zobrazovat stránky HTML, protože se jedná o prostý text. Způsoby, jakými může uživatel přistupovat k obsahu, se však můžou lišit. Běžný uživatel například vidí obsah, ale zrakově postižení uživatelé si můžou zvětšit velikost písma nebo si nechat předčítat obsah v nějakém k tomu určeném nástroji. Předčítače textu někdy předčítají také typ elementu jazyka HTML, aby se mohl posluchač připravit na to, co bude následovat. Uživatel se tak kupříkladu dozví, že předčítač narazil na seznam ještě předtím, než začne předčítat jeho položky. Stejně tak se uživatel dozví, že předčítač narazil na odkaz, aby se mohl rozhodnout, jestli chce přejít na jeho cílovou adresu.

Uživatelé používající předčítače textu můžou procházet stránku různými způsoby – například můžou skákat z jednoho nadpisu na druhý pomocí klávesnice. Díky tomu poznají zajímavá témata, o nichž by se chtěli dozvědět více, a přitom nemusí poslouchat obsah celé stránky od začátku až do konce.

Už tedy víme, že dobrá sémantika znamená propastný rozdíl ve srozumitelnosti obsahu pro handicapované uživatele.

Rovněž se může zlepšit optimalizace pro vyhledávače (neboli umístění naší stránky ve výsledcích vyhledávačů, obvykle označovaná termínem SEO), protože vyhledávací roboti upřednostňují ty části obsahu, které jsou označené určitým způsobem. Podle nadpisů kupříkladu rozpoznávají hlavní témata naší stránky, a tak můžou snadněji indexovat obsah stránky.

V této knize se dozvíte, proč je kód s dobrou sémantikou efektivnější, udržovatelnější a lze snadněji měnit jeho vzhled.

Elementy, atributy a hodnoty

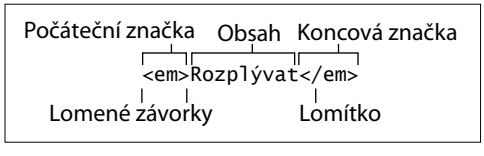
Jelikož už jsme se seznámili s jazykem HTML, popišme si, z čeho se jeho kód skládá.

Jazyk HTML se skládá ze tří hlavních komponent – z **elementů**, **atributů** a **hodnot**. Příklady těchto komponent jste již viděli u své základní stránky.

Elementy

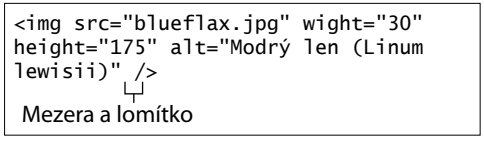
Element je část obsahu webové stránky označovaná drobnými popisky (kterým říkáme **značky**): „Toto je nadpis, ta věc támhle je odstavec a tato skupina prvků tvoří nabídku." Několik elementů jsme si ukázali v předchozí části této kapitoly. Některé elementy obsahují jeden nebo více atributů, s nimiž dále popisujeme účel nebo obsah (je-li nějaký) elementu.

Elementy můžou obsahovat text a další elementy, nebo můžou být prázdné. Neprázdný element se skládá z **počáteční značky** (tvořené názvem a atributy elementu uzavřenými mezi znaky < a >), obsahu a z **koncové značky** (tvořené lomítkem a názvem elementu uzavřenými mezi znaky < a >). Pokud mají značky svou počáteční a koncovou variantu, říká se jim **párové značky**. Prohlédněte si obrázek 1.3.



Obrázek 1.3. Zde je typický element jazyka HTML. Počáteční a koncová značka obklopuje text, který tento element popisuje. V tomto případě zvýrazňujeme slovo „rozplývat" pomocí elementu `em`. Bývá zvykem zapisovat značky malými písmeny

Prázdný element označený **nepárovou značkou** vypadá jako kombinace počáteční a koncové značky, přičemž začíná znakem <, za nímž následuje název elementu, atributy a jejich hodnoty, volitelná mezera, volitelné lomítko a povinný koncový znak > (viz obrázek 1.4).



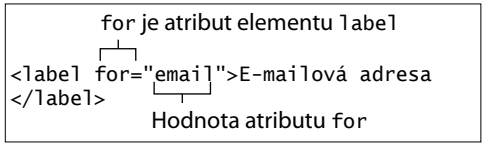
Obrázek 1.4. Prázdné elementy (jako je zde zobrazený element `img`) neobalují žádný textový obsah (text atributu `alt` je součástí daného elementu, ale není jím obklopený). Tento element má jedinou nepárovou značku, která ho otevírá i zavírá. Mezera a lomítko na jeho konci jsou volitelné, ale je dobrým zvykem je zapisovat. Poslední znak > je však povinný

Mezera a lomítko na konci prázdného elementu jsou v jazyce HTML5 nepovinné. Pokud jste však programovali v jazyce XHTML, pravděpodobně jste si na tento zápis zvykli. Ve zdrojových kódech této knihy na něj narazíte taktéž, ale pokud se jej rozhodnete vynechat, na výsledek to nebude mít žádný vliv. Ať už si zvolíte jakoukoliv možnost, držte se jí.

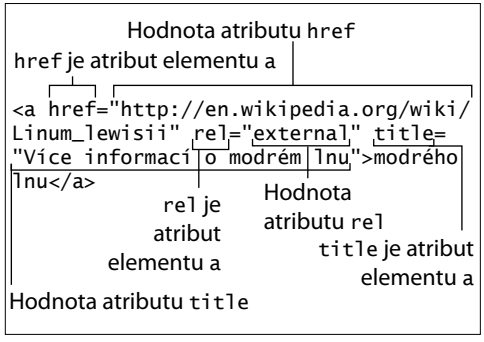
Běžně se názvy elementů zapisují malými písmeny, ačkoliv jazyk HTML5 umožňuje psát rovněž velká písmena. V dnešní době už je velmi vzácné najít někoho, kdo píše názvy elementů velkými písmeny, takže pokud v sobě nechcete nutně probudit rebela, raději byste to neměli dělat. Jedná se o velmi staromódní postup.

Atributy a hodnoty

Atributy obsahují informace o obsahu dokumentu (viz obrázek 1.5 a obrázek 1.6). V jazyce HTML5 není nutné uzavírat hodnoty atributů do uvozovek, ale opět se jedná o již zavedený zvyk, takže je to vhodné. A stejně jako názvy elementů byste měli psát i názvy atributů malými písmeny.



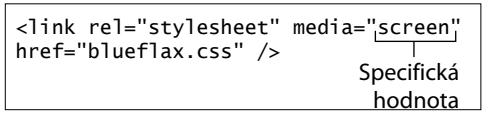
Obrázek 1.5. Element `label` (přidělující textový popisek formulářovému poli) s jedinou dvojicí atributu a hodnoty. Atributy se vždy nacházejí uvnitř počáteční značky. Běžně se jejich hodnoty uzavírají do uvozovek



Obrázek 1.6. Některé elementy (například následující element `a`) můžou mít více atributů. Každý atribut může mít svou vlastní hodnotu. Na pořadí atributů nezáleží a všechny dvojice atributu a hodnoty jsou od sebe oddělené mezerou

Ačkoliv se o přípustných hodnotách většiny atributů dozvíte postupně v této knize, podívejte se alespoň na typy hodnot, s nimiž se budete setkávat.

Některé atributy přijímají libovolnou hodnotu, zatímco jiné jsou omezenější. Atributy obvykle přijímají hodnoty výčtového typu a předdefinované hodnoty. Jinými slovy – musíme si zvolit hodnotu z nějakého standardního seznamu možností (viz obrázek 1.7). Hodnoty výčtového typu bychom měli psát výhradně velkými písmeny.



Obrázek 1.7. Některé atributy můžou mít pouze specifické hodnoty. Kupříkladu atributu `media` elementu `link` lze nastavit hodnotu `all`, `screen` nebo `print` apod., ale nemůžeme mu přiřadit jakoukoliv hodnotu jako atributu `title`

Spousta atributů přijímá číselnou hodnotu, a to zejména atributy popisující velikost a délku. Číselné hodnoty uvádíme vždy bez jednotky. Číselné hodnoty totiž udávají počet pixelů (například šířka a výška obrázku nebo videa).

Některé atributy, jako jsou atributy `href` a `src`, se odkazují na jiné soubory, a proto přijímají hodnoty ve formě adresy URL (Uniform Resource Locator), což je jedinečná adresa prostředku na Internetu. Na adresy URL se zaměříme v části „Adresy URL“ této kapitoly.

Rodičovské a dceřiné elementy

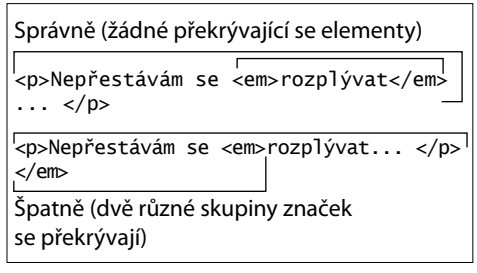
Jestliže jeden element obsahuje jiný element, považujeme ho za rodičovský element, který obaluje dceřiný element. Všechny elementy obsažené v tomto dceřiném elementu nazýváme potomky našeho vnějšího rodičovského elementu (viz výpis 1.10). Ve skutečnosti můžeme vytvořit strom webové stránky, který ukazuje vztahy mezi jednotlivými elementy stránky a jednoznačně identifikuje každý element.

Výpis 1.10. Náš element `article` je rodičovským elementem elementů `h1`, `img` a `p`. Elementy `h1`, `img` a `p` jsou dceřinými elementy (a také potomky) elementu `article`. Element `p` je rodičovským elementem pro elementy `em` a `a`. Elementy `em` a `a` jsou dceřinými elementy elementu `p`, a také potomky (ale ne dceřinými elementy) elementu `article`. Element `article` je předek elementů `em` a `a`

```
<article>
  <h1>Prchlavý modrý len</h1>
  <img src="modrylen.jpg"... />
  <p>Nepřestávám se <em>rozplývat</em>
  ... <a...>modrého lnu</a>...</p>
</article>
```

Tento strom podobající se genealogickému stromu rodinných vztahů je základní vlastností kódu jazyka HTML. Usnadňuje jak stylování elementů (na nějž se poprvé podíváme v kapitole 7, „Stavební kameny kaskádových stylů“), tak aplikování funkcí jazyka JavaScript na ně.

Měli bychom poznamenat, že když elementy obsahují jiné elementy, všechny vnitřní elementy by měly být správně vnořené; tzn. zcela obsažené uvnitř svého rodičovského elementu. Kdykoliv použijeme koncovou značku, měla by odpovídat poslední příslušné neuzavřené počáteční značce. Jinými slovy – když nejprve napíšeme počáteční značku elementu 1 a potom počáteční značku elementu 2, musíme napsat koncovou značku elementu 2 a za ní až koncovou značku elementu 1 (viz obrázek 1.8).



Obrázek 1.8. Elementy musí být správně vnořené. Pokud otevřeme element `p` a následně otevřeme element `em`, musíme uzavřít element `em` před uzavřením elementu `p`.

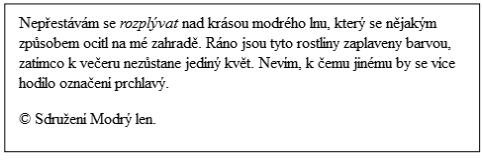
Textový obsah webové stránky

Textový obsah nacházející se uvnitř elementů je pravděpodobně nejzákladnější složkou webové stránky. Pokud jste někdy používali některý textový procesor, už jste psali nějaký text. Text stránky HTML se však liší v několika důležitých ohledech.

Zaprvé – když webový prohlížeč zobrazuje stránku HTML, spojuje nadbytečné mezery a tabulátory do jediné mezery a se zalomením řádku nakládá tak, že ho zobrazí buď jako mezeru, nebo ho zcela ignoruje (viz výpis 1.11 a obrázek 1.9).

Výpis 1.11. Textovým obsahem stránky (níže zvýrazněný) nazýváme v podstatě vše mimo značky a jejich atributy. V tomto kódu oddělujeme každý řádek nejméně jedním zalomením řádku a některá slova oddělujeme více mezerami (abychom si demonstrovali spojování zalomení řádků a mezer). Dále obsahuje speciální výraz s ampersandem (`&copy;`), kterému se říká znaková entita jazyka HTML a v tomto případě odpovídá znaku copyrightu. Tímto zápisem dosahujeme toho, že se příslušný znak zobrazí správně bez ohledu na použitou znakovou sadu

```
<p>Nepřestávám se <em>rozplývat</em> nad
krásou modrého lnu,  který se nějakým
způsobem ocitl na mé zahradě.

Ráno jsou tyto rostliny  zaplaveny
barvou, zatímco k večeru nezůstane
jediný květ.

Nevím, k čemu jinému by se více hodilo
označení prchlavý.</p>

<p>&copy; Sdružení Modrý len.</p>
```



Obrázek 1.9. Webový prohlížeč ignoruje nadbytečné mezery a zalomení řádků a nahrazuje danou znakovou entitu odpovídajícím symbolem (©)

Zadruhé – jazyk HTML býval omezený pouze na znaky sady ASCII, a to zejména písmena, číslice a několik běžných symbolů. Znaky s diakritickými znaménky a spoustu standardních symbolů museli vývojáři zapisovat výhradně pomocí znakových entit; kupříkladu `&eacute;` pro znak é nebo `&copy;` pro znak ©. Kompletní seznam entit je k dispozici na adrese *http://www.elizabethcastro.com/html/extras/entities.html*.

Znaková sada Unicode vyřešila většinu problémů se speciálními znaky. Běžnou praxí je nastavovat svým stránkám kódování UTF-8 (kódování UTF-8 je typ zápisu znaků ve znakové sadě Unicode; obvykle se ale pro jedno-

duchost označuje jako znaková sada UTF-8, a stejně tak tomu bude v této knize; tuto znakovou sadu používáme i u naší základní stránky, jak je patrné na výpisu 1.12) a pod stejným kódováním ukládat také soubory HTML. Rozhodně byste měli dělat to samé.

Výpis 1.12. Znakovou sadu dokumentu definujeme hned za počáteční značkou elementu `head`, a to prostřednictvím atributu `charset` elementu `meta`

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="utf-8" />
  <title>Modrý len (Linum lewisii)
  </title>
</head>
<body>
...
</body>
</html>
```

Jelikož znaková sada Unicode obsahuje všechny myslitelné i nemyslitelné znaky a vznikla jako rozšíření nedostačující sady ASCII, dokumenty s touto sadou lze zobrazit ve všech současných prohlížečích a editorech, ale nikoliv v těch obzvláště starých. Webové prohlížeče, jež neumí pracovat se znakovou sadou Unicode, zobrazují pouze ty znaky, které spadají do znakové sady ASCII, zatímco ostatní prohlížeče zobrazují také znaky specifické pro sadu Unicode. Přesto je i v dnešní době obvyklé zapisovat určité znaky pomocí znakových entit; typickým příkladem je právě entita `&copy;`.

Odkazy, obrázky a jiný netextový obsah

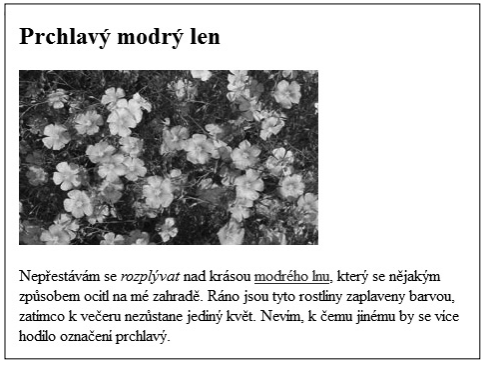
Webová stránka samozřejmě ožívá, když ji doplníme odkazy na jiné stránky, obrázky, videi, hudbou, animacemi atd. Tyto externí soubory samozřejmě neukládáme přímo do souboru HTML, ale ve stránce specifikujeme pouze jejich adresu (viz výpis 1.13). Protože takový odkaz je textový, soubor HTML zůstává dostupný téměř všude.

Výpis 1.13. V našem základním dokumentu HTML máme odkaz na obrázek s názvem *modrylen.jpg*. Webový prohlížeč požádá server o tento obrázek, nahraje jej a zobrazí, a to vše v průběhu načítání zbytku stránky. Naše stránka obsahuje také odkaz na jinou stránku s informacemi o modrém lnu

```
...
<article>
  <h1>Prchlavý modrý len</h1>

  <img src="modrylen.jpg"
    width="300" height="175"
    alt="Modrý len (Linum lewisii)" />

  <p>Nepřestávám se <em>rozplývat</em>
    nad krásou <a href="http://
    en.wikipedia.org/wiki/Linum_lewisii"
    rel="external"
    title="Více informací o modrém lnu">
    modrého lnu</a>, který se nějakým
    způsobem ocitl na mé zahradě.
    Ráno jsou tyto rostliny zaplaveny
    barvou, zatímco k večeru nezůstane
    jediný květ. Nevím, k čemu jinému
    by se více hodilo označení prchlavý.
  </p>
</article>
...
```

Webové prohlížeče (až na textové webové prohlížeče) si umí poradit s odkazy a obrázky jednoduše (viz obrázek 1.10). Neví si však rady s některými dalšími typy souborů. Jestliže se odkážeme na soubor, který webový prohlížeč návštěvníka nezná, pokusí se tento prohlížeč vyhledat zásuvný modul nebo nějakou vhodnou aplikaci v počítači návštěvníka, která by tento soubor otevřela.

Můžeme také prohlížeči sdělit, jak by měl daný obsah zobrazit v daném zásuvném modulu, nebo jak by si měl návštěvník příslušný zásuvný modul stáhnout, pokud ho na svém počítači nemá.



Obrázek 1.10. Na obrázky a jiný textový obsah se odkazujeme z webové stránky, přičemž prohlížeč je zobrazuje společně s textem

Veškeré požadavky na stahování a instalaci zásuvných modulů kazí uživatelský prožitek z našich webových stránek za předpokladu, že na nich uživatelé vůbec zůstanou. Zásuvné moduly rovněž můžou způsobovat výkonnostní problémy, protože nejsou přímou součástí webového prohlížeče.

Například zásuvný modul Flash je nejrozšířenějším modulem už spoustu let. Většina z vás pravděpodobně při přehrávání online videa ve Flashi někdy zaregistrovala zpomalení počítače, nebo dokonce pád webového prohlížeče.

Jazyk HTML5 se snaží zbavit těchto problémů nativní podporou přehrávání hudebních souborů a videí pomocí elementů `audio` a `video`. Bohužel vývojáři webových prohlížečů se nemohli shodnout, jaké formáty podporovat, takže zásuvné moduly prozatím nemůžeme úplně odstřihnout, ale určitě se jedná o zárodek nové funkčnosti webových prohlížečů.

Jak pracovat s obrázky, se naučíte v kapitole 5, „Obrázky.“ V kapitole 17, „Audio, video a jiná multimédia“ se podrobněji podíváte na to, jestli už jsou zásuvné moduly překonané.

Jména souborů

Webová stránka se podobá jakémukoliv jinému textovému dokumentu, takže má své jméno souboru, podle něhož ji poznáme, a také ji poznají návštěvníci a jejich webové prohlížeče. Když přidělujeme jména souborů svým webovým stránkám, měli bychom dbát na několik rad, které nám pomůžou lépe uspořádat své soubory. Díky tomu návštěvníci snadněji vyhledají stránky, webové prohlížeče je zobrazí správně, a navíc budou stránky optimalizované pro vyhledávače (viz obrázky 1.11 a 1.12).

Pište malá písmena v názvech souborů

Protože jméno souboru, které si zvolíme, rozhoduje o tom, co musí uživatel napsat do panelu adresy, aby se k dané stránce dostal, můžeme ho ušetřit překlepů a bolestí hlavy tak, že použijeme pouze malá písmena ve jméně. Je to také užitečné, když vytváříme odkazy na naše stránky ručně. Jestliže mají všechny soubory malá písmena ve jménech, máme o jednu starost méně.

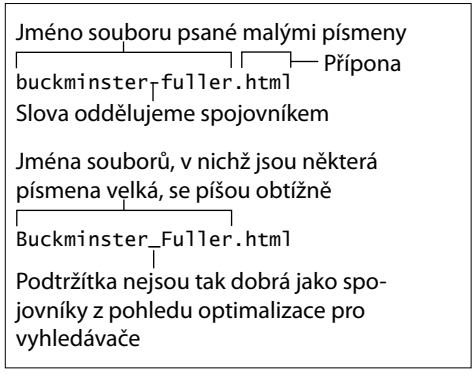
Slova oddělujte spojovníkem

Názvy souborů by nikdy neměly obsahovat mezery. Místo nich je lepší používat spojovníky; kupříkladu stejně jako ve jménech *historie-spolecnosti.html* a *me-oblibene-filmy.html*. Příležitostně můžete narazit na webové stránky, které oddělují slova v názvech souborů podtržítkem (_), ale to rozhodně nelze doporučit, jelikož vyhledávací roboti upřednostňují spojovníky.

Používejte správnou příponu

Základní postup, jakým webový prohlížeč pozná, že by měl s textovým dokumentem pracovat jako s webovou stránkou, je ten, že se podívá na jeho příponu. Přestože je možné používat příponu *.htm*, tato přípona je považována za zastaralou, proto mnohem častěji narazíme na příponu *.html*. Kdyby měla stránka jinou příponu, například *.txt*, webový prohlížeč by s ní zacházel jako s textovým souborem, takže by zobrazil její zdrojový kód uživateli.

Tip: Operační systém Max OS, ani operační systém Windows, standardně neodhalují skutečnou příponu dokumentu. Pokud to bude nutné, změňte nastavení složky, abyste viděli přípony.



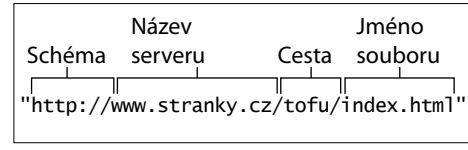
Obrázek 1.11. Jména souborů píšeme malými písmeny, slova v nich oddělujeme spojovníkem a na konec jména doplňujeme příponu *.html*. Kombinování velkých a malých písmen komplikuje návštěvníkům život při psaní adresy a hledání naší stránky

Obrázek 1.12. Jména adresářů bychom měli taktéž psát malými písmeny. Důvodem je zachování konzistence. Pokud nepoužíváme velká písmena, návštěvníci nemusí ztrácet čas přemýšlením nad otázkami typu: „Bylo tam malé písmeno b, nebo velké písmeno B?"

Adresy URL

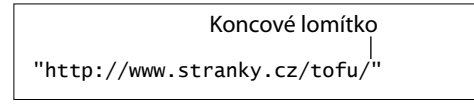
Adresám stránek říkáme adresy URL (Uniform Resource Locator). Adresa URL obsahuje informaci o tom, kde se soubor nachází a co by s ním měl webový prohlížeč dělat. Každý soubor na Internetu má svou jedinečnou adresu URL.

První část adresy URL se nazývá **schéma**. Schéma sděluje webovému prohlížeči, jak by měl se souborem zacházet. Nejčastějším schématem je schéma *http*, kterým označujeme prostředky dostupné protokolem HTTP (Hypertext Transfer Protocol). Tímto protokolem přistupujeme k webovým stránkám (viz obrázek 1.13).



Obrázek 1.13. Naše základní adresa URL obsahuje schéma, název serveru, cestu a jméno souboru

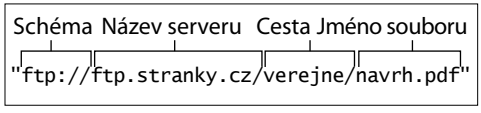
Druhou část adresy URL tvoří název serveru, na kterém je daný soubor uložený, za nímž následuje cesta k souboru a jméno samotného souboru. Na konci adresy URL může občas chybět jméno souboru, přičemž taková adresa může, ale také nemusí končit lomítkem (viz obrázek 1.14). V tomto případě se adresa URL odkazuje na výchozí soubor v adresáři, jenž se nachází v cestě na posledním místě. Tímto souborem je typicky soubor *index.html*.



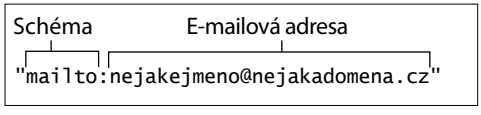
Obrázek 1.14. Adresa URL s lomítkem na konci bez jména souboru ukazuje na výchozí soubor v posledním adresáři (v tomto případě v adresáři *tofu*). Nejobvyklejším jménem výchozího souboru je jméno *index.html*. Tato adresa URL ukazuje tedy na stejný soubor jako adresa z předchozího příkladu

Další běžná schémata jsou:

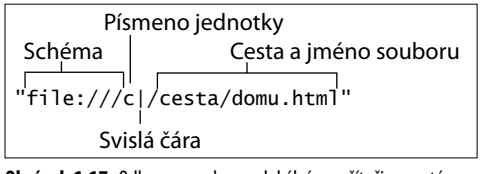
- *https* pro zabezpečené webové stránky,
- *ftp* (File Transfer Protocol) pro stahování souborů (viz obrázek 1.15),
- *mailto* pro označování e-mailových adres (viz obrázek 1.16),
- *file* pro přístup k souborům na lokálním pevném disku nebo lokální síti (pravděpodobně nebudete používat schéma *file* příliš často, pokud vůbec; viz obrázek 1.17).



Obrázek 1.15. Jakmile uživatel klepne na odkaz s touto cílovou adresou URL, webový prohlížeč zahájí přenos souboru *navrh.pdf* protokolem FTP



Obrázek 1.16. Adresa URL označující e-mailovou adresu obsahuje schéma *mailto* následované dvojtečkou bez dvou lomítek a dále zapisujeme samotnou adresu



Obrázek 1.17. Odkaz na soubor na lokálním počítači se systémem Windows pomocí schématu *file*. Uživatelé systému Mac OS by měli používat zápis *file://pevnydisk/cesta/jmenosouboru*. Svislá čára (|) není nutná (tento zápis někdy funguje i v systému Windows)

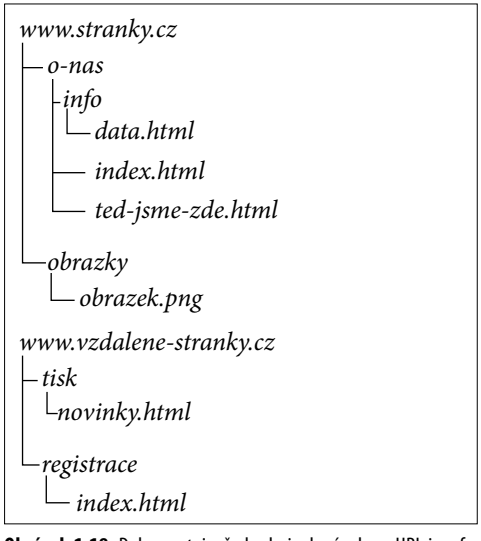
Za schématem obyčejně následuje dvojtečka a dvě lomítka. Výjimkou jsou schémata *mailto* a *news*, za nimiž následuje pouze dvojtečka.

Povšimněte si, že za schématem *file* píšete dvojtečku a tři lomítka. Je tomu tak z toho důvodu, že za hostitelský počítač, který u jiných schémat patří mezi druhé a třetí lomítko, se považuje lokální počítač. Názvy schémat pište vždy malými písmeny.

Z těchto schémat budete nejčastěji potřebovat schémata *http* a *mailto*. Ostatní slouží pro speciální případy.

Absolutní adresy URL

Adresy URL můžou být buď absolutní, nebo relativní. **Absolutní adresa URL** zobrazuje celou cestu k souboru, a to včetně schématu, jména serveru, kompletní cesty a samotného jména souboru (viz obrázek 1.18). Absolutní adresu URL lze přirovnat ke kompletní adrese, se jménem a příjmením, ulicí s číslem domu, obcí, PSČ a státem. Bez ohledu na to, odkud dopis posíláme, doručovací společnost bude schopná najít příjemce. Analogicky u adres URL můžeme říct, že absolutní adresa URL popisuje umístění odkazovaného souboru zcela přesně, takže nezáleží na tom, odkud se odkazujeme; tj. zda se odkaz nachází ve stránce na našem serveru, nebo na úplně jiném serveru.



Obrázek 1.18. Dokument, jenž obsahuje dané adresy URL je referenčním bodem pro relativní adresy URL. Jinými slovy – relativní adresy URL se vztahují k umístění tohoto souboru na serveru. Absolutní adresy URL fungují bez ohledu na to, kde se nacházejí, protože naprosto přesně ukazují na cílový prostředek

Když se na soubor odkazuje z externího serveru, vždy bychom měli používat absolutní adresu URL. Absolutní adresy URL musíme používat také pro soubory na serveru FTP a téměř vždy, když nepoužíváme protokol HTTP.

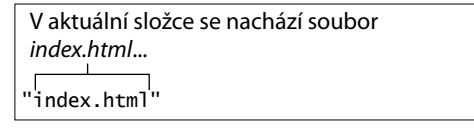
V tabulce 1.1 si popíšeme způsoby přístupu k nejrůznějším souborům ze souboru *ted-jsme-zde.html*., a to jak ze stejného serveru (*stranky.cz*), tak ze vzdáleného serveru (*vzdalene-stranky.cz*). Na nich si ukážeme, jaké jsou rozdíly mezi relativními a absolutními adresami URL.

Relativní adresy URL

Když budete někomu sdělovat, kde má dům váš soused, pravděpodobně mu nebudete dávat kompletní adresu, ale řeknete prostě: „Sousedovi patří třetí dům směrem dolů po pravé straně." Jedná se o relativní adresu. To znamená, že záleží na tom, kde se právě nacházíte. Kdybyste měli stejnou informaci v jiném městě, nikdy byste tohoto souseda nenašli.

Stejným způsobem popisuje umístění požadovaného souboru relativní adresa, přičemž jako referenční bod slouží umístění souboru, v němž tuto adresu použijeme. Můžeme tudíž definovat adresu URL takto: „stránka *xyz*, která se nachází ve stejné složce jako tato stránka."

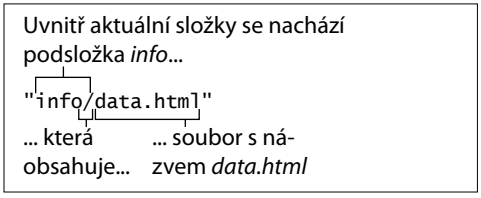
Relativní adresou URL pro soubor, jenž se nachází ve stejné složce jako aktuální stránka (stránka, co obsahuje tuto adresu), je jméno souboru s příponou (viz obrázek 1.19). Kdybychom chtěli napsat adresu URL pro soubor v podsložce aktuální složky, jednoduše bychom napsali název této podsložky, posléze lomítko a nakonec jméno požadovaného souboru s příponou (viz obrázek 1.20).



Obrázek 1.19. Relativní adresa URL pro soubor ve stejné složce (viz obrázek 1.18). Je nutné zadávat jen jméno souboru s příponou, a nemusíme před ně psát *http://www.stranky.cz/about/* (složka, v níž jsou uložené oba soubory)

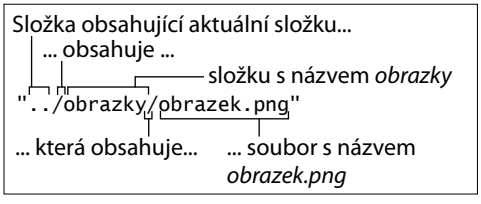
Tabulka 1.1. Absolutní versus relativní adresy URL

Jméno souboru	Absolutní adresa URL (lze použít všude)	Relativní adresa URL (funguje pouze relativně vzhledem k umístění souboru ted-jsme-zde.html)
index.html	*http://www.stranky.cz/o-nas/index.html*	*index.html*
data.html	*http://www.stranky.cz/o-nas/info/data.html*	*/info/data.html*
obrazek.png	*http://www.stranky.cz/obrazky/obrazek.png*	*../obrazky/obrazek.png*
novinky.html	*http://www.vzdalene-stranky.cz/tisk/novinky.html*	(musíme použít absolutní adresu URL)
index.html	*http://www.vzdalene-stranky.cz/registrace/*	(musíme použít absolutní adresu URL)



Obrázek 1.20. Na soubor (v tomto případě soubor *data.html*; viz obrázek 1.18) uvnitř podsložky aktuální složky se odkazujeme tak, že napíšeme jméno podsložky, za něj lomítko a jméno souboru s příponou

Kdybychom se chtěli odkázat na soubor uvnitř rodičovské složky, napsali bychom dvě tečky a lomítko před jméno souboru (viz obrázek 1.21). Předponu tvořenou dvěma tečkami s lomítkem můžeme opakovat, a tak se odkázat na libovolný soubor na lokálním serveru.



Obrázek 1.21. Tento soubor, který je možné spatřit na obrázku 1.18, se nachází ve složce (*obrazky*), jež sousedí s aktuální složkou (*o-nas*). Obě tyto složky jsou podsložkami kořenové složky serveru. V tomto případě se pomocí dvou teček s lomítkem vrátíme o jednu úroveň výše a potom napíšeme název dané podsložky následovaný lomítkem a jménem souboru s příponou (v praxi bychom nejspíše použili popisnější název souboru než *obrazek.png*)

Jestliže jsou naše soubory uložené na webovém serveru, můžeme se vyhnout těžkopádným relativním adresám URL typu *.../.../obrazky/rodina/dovolena.jpg*. Jednoduše skočíme do kořenové složky serveru a odtud budeme pokračovat až k cílovému souboru. Postačí nám přidat lomítko na začátek naší adresy. Relativní adresu URL vzhledem ke kořenové složce serveru bychom v tomto případě definovali takto: */obrazky/rodina/dovolena.jpg* (za předpokladu, že se složka *obrazky* nachází přímo v kořenové složce serveru). Jak už jsme si řekli, tento zápis funguje pouze na webovém serveru; ať už se jedná o server našeho poskytovatele hostingových služeb nebo server běžící na našem lokálním počítači (nejoblíbenějším serverem je webový server Apache).

Pokud nevyvíjíte své webové stránky na lokálním webovém serveru, pravděpodobně budete používat relativní adresy URL. Výjimku samozřejmě uděláte jen tehdy, když se budete chtít odkázat na soubory na serveru někoho jiného. Relativní adresy URL usnadňují přesun z lokálního systému souborů na server. Pokud struktura složek a adresářů zůstane stejná, nemusíte měnit žádné cesty, a přesto budou vaše odkazy fungovat.

Co byste si měli odnést

Základní znalosti jazyka HTML a některé praxí osvědčené postupy tvoří pevný základ pro stavbu efektivních webových stránek. Z této kapitoly byste si měli odnést:

- Webová stránka se skládá ze tří hlavních komponent – z textového obsahu, odkazů na jiné soubory a značkovacího kódu.
- Značkovací kód jazyka HTML tvoří elementy, atributy a jejich hodnoty.
- Běžně se značky jazyka HTML zapisují malými písmeny (s výjimkou značky `DOCTYPE`), hodnoty atributů byste měli psát do uvozovek a uzavírat prázdné (nepárové) elementy mezerou a lomítkem.
- Na začátek svých dokumentů HTML vkládejte deklaraci `DOCTYPE`:
  ```
  <!DOCTYPE html>
  ```
- Obsah stránky patří do elementu `body`. Informace určené prohlížečům a vyhledávacím robotům byste měli vkládat převážně do elementu `head`.
- Označujte svůj obsah sémantickými značkami jazyka HTML a nezabývejte se tím, jak by měl obsah vypadat.

- Sémantický kód jazyka HTML vylepšuje přístupnost a díky němu mohou být vaše webové stránky efektivnější, udržovatelnější a snadněji upravíte jejich vzhled.
- Jazyk CSS řídí prezentaci obsahu stránky HTML.
- Každý webový prohlížeč má svou šablonu stylů, která určuje výchozí vzhled vašeho dokumentu HTML. Pravidla obsažená v této šabloně stylů můžete přepsat svými vlastními pravidly stylů.
- Jména souborů a složek pište výhradně malými písmeny, přičemž slova oddělujte spojovníkem, a ne mezerou nebo podtržítkem.

V následující kapitole se dozvíte, jak pracovat se soubory svých webových stránek.

Práce se soubory webových stránek

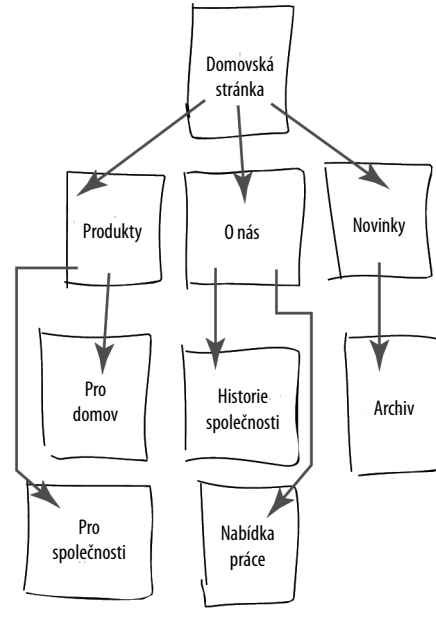
2

Než začnete psát své elementy a atributy v jazyce HTML, měli byste vědět, jak vytvářet své soubory, do nichž budete tento kód psát. V této kapitole se naučíte vytvářet, upravovat a ukládat soubory svých webových stránek. Rovněž se dozvíte, jak naplánovat své webové stránky a uspořádat obsah.

Pokud už se nemůžete dočkat samotného vývoje a víte, jak vytvářet soubory, můžete přeskočit ke kapitole 3, „Základní struktura dokumentu HTML."

Plánování webových stránek

Ačkoliv se můžeme vrhnout do vývoje webových stránek po hlavě, je dobrý nápad pozastavit se, zamyslet se a naplánovat své webové stránky (viz obrázek 2.1). Tímto způsobem si stanovíme směr vývoje a později nebude muset tolik přestavovat.



Obrázek 2.1. Nakreslíme si strukturu webových stránek na papír a popřemýšlíme, co by jednotlivé stránky měly obsahovat. To nám pomůže navrhnout správné uspořádání

Webové stránky plánujeme následovně:

- Zamyslíme se nad tím, proč vlastně vytváříme tyto webové stránky. Čeho chceme dosáhnout?
- Kdo budou naši návštěvníci? Jak bychom jim mohli přizpůsobit obsah?
- Kolik stránek potřebujeme? Jaké uspořádání upřednostňujeme? Chceme, aby návštěvníci procházeli obsah v určitém pořadí, nebo budou moct jednoduše prozkoumat jakoukoliv část obsahu?
- Nakreslíme si návrh našich webových stránek na papír.
- Vymyslíme jednoduchá výstižná jména pro naše stránky, obrázky a jiné externí soubory (více informací lze najít v části „Jména souborů" v kapitole 1, „Stavební kameny webových stránek").

Tip: Nepřežeňte to s fází plánování. V určité chvíli musíte také začít psát obsah a zdrojový kód.

Tip: Pokud web moc neznáte, zkuste po něm chvíli jen tak brouzdat, abyste zjistili, jaké máte možnosti. Možná byste měli začít s webovými stránkami své konkurence.

Tip: Bývá běžné (ale nikoliv nezbytné) navrhnout strukturu složek svých webových stránek tak, jak jste si uspořádali kategorie na papíře (viz obrázek 2.1). Více informací najdete v části „Uspořádání souborů."

Tip: V článku „A Checklist for Content Work" od Erin Kissaneové (*http://www.alistapart.com/articles/a-checklist-for-content-work/*) získáte představu, jak můžete postupovat při tvorbě obsahu svých webových stránek. Jedná se o ukázku z její knihy, ve které pojednává o strategii tvorby obsahu.

Tip: Kniha „The Principles of Beautiful Web Design" od Jasona Beaird (vydána nakladatelstvím SitePoint roku 2010) by vás mohla zaujmout, pokud nejste návrháři nebo teprve začínáte v oblasti návrhu, a přitom vás zajímá, jak navrhnout atraktivní a efektivní webové stránky.

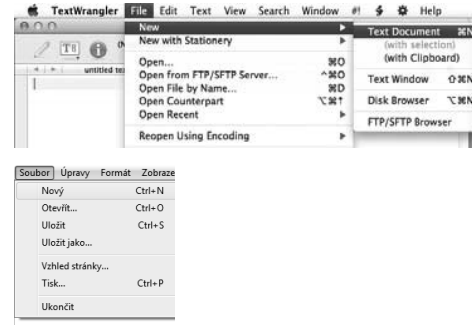
Vytvoření nové webové stránky

K tvorbě webové stránky nepotřebujeme žádné speciální nástroje. Můžeme používat jakýkoliv textový editor, a to dokonce program Poznámkový blok (viz obrázky 2.2 a 2.4), jenž je součástí operačního systému Windows, nebo program TextWrangler (viz obrázky 2.2 a 2.3), který lze stáhnout zdarma pro operační systém OS X (*http://www.barebones.com/products/textwrangler/*).

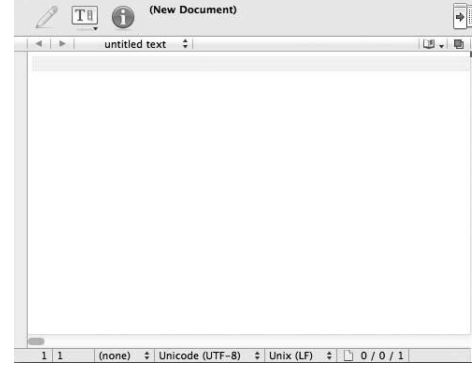
Poznámka: Operační systém OS X nabízí editor TextEdit, ale v některých verzích tohoto systému obsahuje chybu, která ztěžuje práci se soubory HTML.

Novou webovou stránku vytvoříme takto:

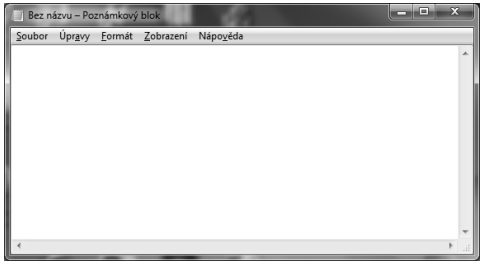
1. Otevřeme si libovolný textový editor.
2. Vybereme položku **Soubor/Nový**, čímž vytvoříme nový prázdný dokument (viz obrázek 2.2).
3. Začneme vytvářet obsah stránky HTML, jak popisuje tato kniha počínaje kapitolou 3, „Základní struktura dokumentu HTML."
4. Uložíme náš soubor. Jak postupovat, si předvedeme v části „Ukládáme naši webovou stránku."



Obrázek 2.2. Otevřeme si svůj textový editor. Napíšeme kód jazyka HTML do prázdného dokumentu, který se objeví, nebo vybereme položku **Soubor/Nový**. Přesný název položky nabídky se může mírně lišit. U nástroje TextWrangler se jedná o položku **File/New/Text Document**. Nahoře se zobrazuje snímek z textového editoru TextWrangler, zatímco dole lze spatřit snímek nástroje Poznámkový blok



Obrázek 2.3. V operačním systému OS X může editor TextWrangler napsat kód jazyka HTML místo nás. Níže si uvedeme tipy na editory pro tento operační systém, které mají dokonalejší funkce pro psaní kódu



Obrázek 2.4. Poznámkový blok je základní program operačního systému Windows umožňující vytvářet stránky HTML. K dispozici je také řada jiných editorů, jak si ukážeme v následujících tipech

Tip: Existuje spousta textových editorů pro operační systémy Windows a OS X, které jsou vhodné pro psaní kódu jazyka HTML (a jazyka CSS). Mezi jejich běžné funkce patří automatické dokončování kódu a nápověda, což umožňuje psát kód přesněji a rychleji. Rovněž zvýrazňují zdrojový kód, abychom snadněji rozeznali elementy jazyka HTML od textového obsahu. Takových funkcí, z nichž nástroj Poznámkový blok nenabízí ani jednu, poskytují celou řadu. K dispozici jsou jak editory zdarma, tak ty placené, které si většinou zaslouží investici, a navíc je možné si je před koupí vyzkoušet.

Tip: Mezi oblíbené editory pro systém OS X patří BBEdit (*http://www.barebones.com/products/bbedit/*), Coda (*http://www.panic.com/coda/*), Espresso (*http://macrabbit.com/espresso/*), Sublime Text (*http://www.sublimetext.com/*) a TextMate (*http://macromates.com/*). Editor TextWrangler se běžně označuje jako „BBEdit Lite." Nejoblíbenějším z těchto editorů je TextMate. Editor SublimeText je také k dispozici na operační systém Windows, stejně tak E Text Editor (*http://www.e-texteditor.com/*), Notepad++ (*http://notepad-plus-plus.org/*) a spousta dalších. Vyhledáte-li vyhledávačem výraz „editor HTML" najdete další editory.

Tip: Pokud používáte některý z výše uvedených editorů, princip vytváření nového souboru zůstává stejný. Pro editaci existující stránky vyberte položku **Soubor/Otevřít** ve svém zvoleném textovém editoru a označte soubor (více informací je možné najít v části „Editace webových stránek"). Ve zbytku této knihy se dozvíte, jak přidat vlastní kód jazyků HTML a CSS, aby stránka vypadala tak, jak potřebujete.

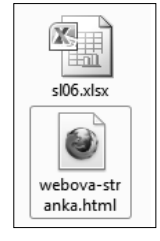
Tip: Nepoužívejte textové procesory (například aplikaci Word od společnosti Microsoft) pro tvorbu stránek HTML. Můžou totiž do vašich souborů přidat nechtěné znaky a kód přestane fungovat.

Ukládáme naši webovou stránku

Webové stránky vytváříme s pomocí textového editoru, ale jsou určené pro zobrazení mnoha webovými prohlížeči na mnoha platformách. Aby byly dostupné všude, musíme své webové stránky uložit ve formátu prostého textu; tzn. bez zbytečného formátování, které můžou přidávat textové procesory.

Aby webové prohlížeče (a servery) rozpoznaly webové stránky a věděly, že mají interpretovat jejich kód, soubory webových stránek musí mít příponu *.html* (případně *.htm*). Tímto přístupem rovněž na první pohled odlišíme prosté textové soubory od webových stránek. Ačkoliv je možné používat obě přípony, lepším řešením je přípona *.html*.

Ikona webových stránek odpovídá výchozímu webovému prohlížeči našeho operačního systému, a ne editoru, v němž jsme tyto stránky napsali (viz obrázek 2.5). Když poklepáme na soubor webové stránky, otevře se ve webovém prohlížeči, ale nikoliv v textovém editoru. To je skvělé pro testování webových stránek v prohlížeči, ale přináší to krok navíc při editaci webových stránek (více informací se nachází v části „Editace webových stránek").

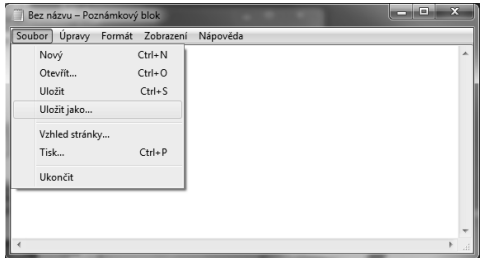


Obrázek 2.5. Sešit aplikace Excel má příponu *.xlsx* a lze ho rozpoznat na základě ikony této aplikace (nahoře). Pokud na něj poklepáme, otevře se v aplikaci Excel. Soubor webové stránky má příponu *.html* nebo *.htm* a ikonu výchozího webového prohlížeče (v tomto případě prohlížeče Firefox) nezávisle na tom, v jakém textovém editoru ho vytvoříme. Jestliže na něho poklepáme, otevře se výchozí webový prohlížeč (ne textový editor)

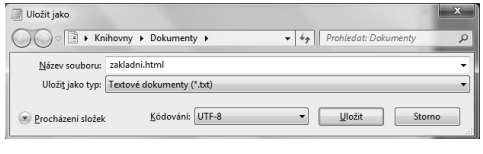
Pro shrnutí – když ukládáme soubor webové stránky, musíme jej uložit ve formátu prostého textu a přiřadit mu příponu *.html* nebo *.htm*.

Webovou stránku uložíme následovně:

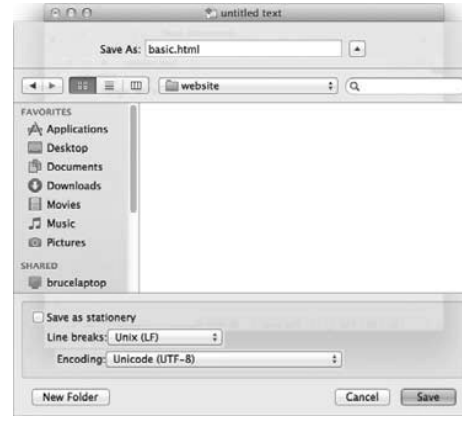
1. Jakmile vytvoříme webovou stránku, vybereme ve svém textovém editoru položku **Soubor/Uložit jako** (viz obrázek 2.6).
2. Objeví se dialogové okno, v němž musíme nastavit formát prostého textu (název se může lišit program od programu).
3. Našemu textovému dokumentu přidělíme příponu *.html* (lepší varianta) nebo *.htm*. Toto je velmi důležitý krok.
4. Vybereme složku, do které chceme uložit naši webovou stránku.
5. Klepneme na tlačítko **Uložit** (viz obrázky 2.7 a 2.8).



Obrázek 2.6. V textovém editoru vybereme položku **Soubor/Uložit jako**.



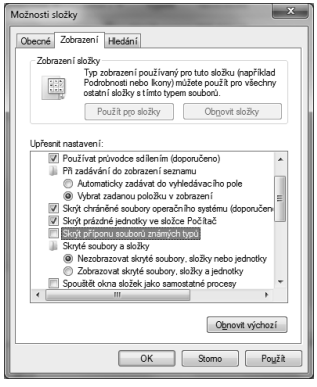
Obrázek 2.7. V nástroji Poznámkový blok si pojmenujeme náš soubor a přidáme mu příponu *.html* nebo *.html*, vybereme položku **Textové dokumenty (*.txt)** z rozevíracího seznamu **Uložit jako typ** a před klepnutím na tlačítko **Uložit** se ujistíme, že jsme vybrali kódování UTF-8. V různých textových editorech se můžou tyto možnosti nastavení lišit, ale budou velmi podobné



Obrázek 2.8. V nástroji TextWrangler pojmenujeme náš soubor a vybereme umístění, kam bychom ho chtěli uložit. Tento nástroj standardně vybírá kódování UTF-8, ale můžeme si vybrat také alternativní kódování.

Tip: Nezáleží na tom, jestli používáte příponu *.html* nebo *.htm*, třebaže přípona *.html* je vhodnější. Ať už si vyberete jakkoliv, držte se své volby, protože když budete používat jedinou příponu, snadněji si zapamatujete adresy URL.

Tip: Některé textové editory v operačním systému Windows můžou přidávat svou vlastní výchozí příponu k vašemu jménu souboru, přestože jste již za jméno doplnili příponu *.html* nebo *.htm* (to by se nemělo týkat editorů určených přímo pro editaci stránek HTML). Váš soubor pojmenovaný kupříkladu *webova-stranka.html.txt* by se nezobrazoval ve webovém prohlížeči správně. Situace je o to horší, že operační systém Windows obvykle skrývá přípony souborů, takže problém není možné na první pohled rozpoznat. Existují dvě řešení. Prvním z nich je uzavřít jméno souboru do uvozovek při jeho prvním ukládání. To by mělo zabránit přidání nadbytečné přípony. Druhým řešením je sdělit systému Windows, aby zobrazoval přípony souborů (viz obrázek 2.9). Díky tomu byste měli postřehnout nadbytečnou příponu a odstranit ji ze jména souboru.



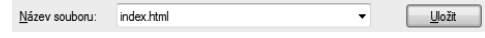
Obrázek 2.9. V nástroji Průzkumník Windows vybereme položku **Nástroje/Možnosti složky**, čímž zobrazíme dialogové okno, které se může mírně lišit v závislosti na verzi systému Windows. Klepneme na kartu **Zobrazení** a ve zde nalezeném seznamu vyhledáme položku **Skrýt příponu souborů známých typů**. Ujistíme se, že příslušné pole je odškrtnuté, aby se zobrazovaly přípony souborů

Tip: Jakmile zvolíte formát prostého textu, váš soubor se obvykle uloží s výchozí znakovou sadou operačního systému. Kdybyste chtěli jinou znakovou sadu, musíte si ji vybrat. Nejlepší volbou je znaková sada UTF-8. Pokud váš textový editor nabízí možnost typu **UTF-8 BOM** nebo něco podobného, zvolte ji. V opačném případě vyberte možnost **UTF-8**. V některých případech nepřidává textový editor značku BOM pro sadu UTF-8, přestože tuto skutečnost přímo nezmiňuje. Pokud se chcete dozvědět o značce BOM nějaké informace, najdete je na adrese *http://cs.wikipedia.org/wiki/UTF-8#BOM*.

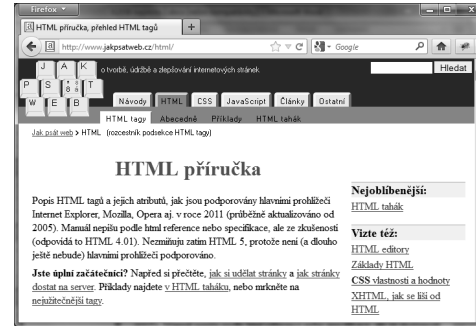
Obrázek 2.10. Spousta textových editorů nám dovoluje vybrat si znakovou sadu, abychom mohli používat písmena a symboly téměř ze všech světových jazyků. Ve většině případů je nejlepší možností znaková sada UTF-8. Měli bychom upřednostňovat možnost **UTF-8 no BOM**, pokud ji náš editor nabízí. V opačném případě se musíme spokojit s možností **UTF-8**. Některé editory mají tuto možnost jako výchozí (například zde zobrazený editor TextWrangler)

Definujeme výchozí a domovskou stránku

Většina webových serverů poskytuje systém pro rozpoznávání výchozí stránky v každé složce, a to na základě jména souboru. Ve většině případů se výchozí stránkou stává soubor *index.html* (viz obrázek 2.11). Pokud tento soubor neexistuje, webové servery obvykle hledají alternativu v podobě souboru *index.htm* nebo *default.htm*. Jestliže náš návštěvník zadá do panelu adresy adresu URL s názvem složky, ale bez jména souboru, zobrazí se výchozí soubor (viz obrázek 2.12).



Obrázek 2.11. Pokud chceme označit soubor jako výchozí stránku dané složky, uložíme ho pod jménem *index.html*



Obrázek 2.12. Když návštěvníci píší cestu ke složce, ale neuvedou jméno souboru, zobrazí se výchozí stránka. V tomto příkladu jsme napsali adresu *http://www.jakpsatweb.cz/html/*. Kdybychom napsali adresu *http://www.jakpsatweb.cz/html/index.html*, načetla by se stejná stránka

Výchozí stránka (tradičně *index.html*) nacházející se v kořenové složce našich webových stránek se nazývá **domovská stránka**. Tato stránka se zobrazí, když uživatel uvede jen naši doménu bez jakýchkoliv dalších informací: *http://www.nase-domena.cz*.

Stejným způsobem můžeme vytvořit výchozí stránku pro každou složku svých webových stránek. Kupříkladu výchozí stránka pro složky */produkty/* a */o-nas/* se rovněž může jmenovat *index.html*, ale každá z nich bude ve své vlastní složce. Návštěvníci obvykle přistupují k těmto částem z domovské stránky nebo prostřednictvím nabídky, kterou zobrazíme na všech stránkách.

Specifikace výchozí stránky složky nebo domovské stránky

Uložíme si svůj soubor pod názvem *index.html* (návod lze najít v části „Ukládáme naši webovou stránku“) do požadované složky.

Pokud soubor *index.html* nefunguje jako výchozí stránka na serveru, na němž máte uložené své webové stránky, a přitom jste postupovali podle rad z kapitoly 21, „Publikování stránek na webu,“ konzultujte tento problém se svým poskytovatelem hostingových služeb.

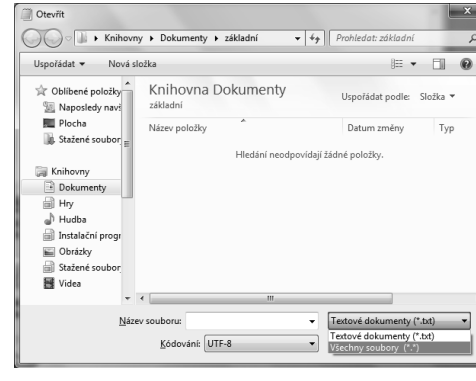
Tip: Jestliže nemáte výchozí stránku uvnitř každé složky, některé servery můžou zobrazovat kompletní obsah složky, což můžete chtít ukazovat návštěvníkům, ale také nemusíte. Pokud chcete ochránit tento obsah před zvědavými návštěvníky, vytvořte výchozí stránku v každé složce, nebo změňte konfiguraci serveru tak, aby skrýval seznam souborů a podsložek. Skrývání obsahu je vhodné zejména pro složky s prostředky; například složky s obrázky, médii, šablonami stylů a skripty jazyka JavaScriptu. Můžete se zeptat svého poskytovatele hostingových služeb, jak byste měli postupovat.

Editace webových stránek

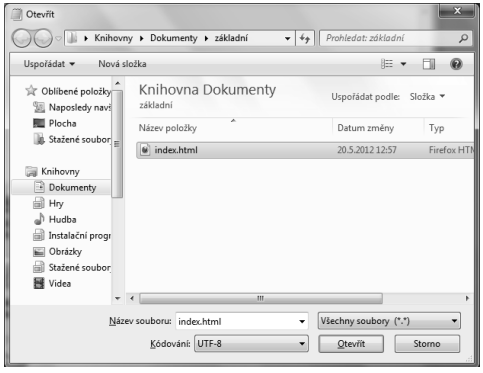
Protože webovou stránku nejčastěji prohlížíme webovým prohlížečem, když na ni poklepáme, otevře se webový prohlížeč zobrazující tuto stránku. Když chceme editovat webovou stránku, musíme ji ručně otevřít v textovém editoru.

Webové stránky editujeme tímto způsobem:

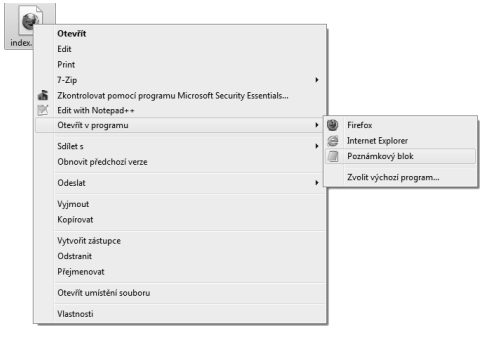
1. Otevřeme si textový editor.
2. Vybereme položku **Soubor/Otevřít**.
3. Vyhledáme složku, která obsahuje požadovaný soubor.
4. Pokud se náš soubor v příslušné složce nezobrazuje, zkusíme vybrat položku **Všechny soubory** (nebo podobnou položku; viz obrázky 2.13 a 2.14). Skutečné jméno této položky a její umístění se může lišit program od programu a platforma od platformy.
5. Klepneme na tlačítko **Otevřít**. Tím jsme přichystali náš soubor k editaci.



Obrázek 2.13. Některé textové editory v operačním systému Windows (kupříkladu Poznámkový blok) nevidí soubory HTML automaticky. Musíme vybrat položku **Všechny soubory**, abychom zobrazili soubory se všemi příponami



Obrázek 2.14. Až zobrazíme soubory se všemi příponami, můžeme si zvolit požadovaný soubor a otevřít ho klepnutím na tlačítko **Otevřít**



Obrázek 2.15. V operačním systému Windows můžeme rovněž klepnout pravým tlačítkem na ikonu dokumentu a z kontextové nabídky vybrat položku **Otevřít v programu** a v ní příslušný textový editor. V operačním systému OS X postupujeme stejně

Tip: Když upravíme již dříve uložený dokument, můžeme změny uložit jednoduše klepnutím na položku **Soubor/Uložit**, aniž bychom se museli už starat o formát souboru (popisovaný v části „Ukládáme naši webovou stránku").

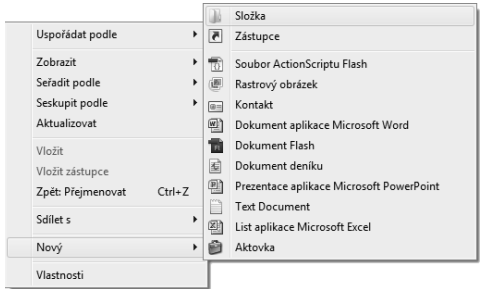
Uspořádání souborů

Předtím než budeme mít spoustu souborů, měli bychom se rozmyslet, kam je budeme vkládat. Běžné (nikoliv však nutné) je vytvářet složku pro každou hlavní část webových stránek. Tento postup umožňuje seskupovat související stránky HTML dohromady.

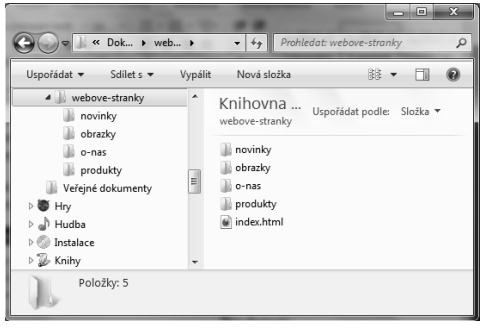
Své soubory uspořádáme následujícím způsobem:

1. Vytvoříme si ústřední složku, do níž budeme ukládat veškerý materiál dostupný na našich webových stránkách. V operačním systému OS X vybereme položku **Soubor/Nová složka** (viz obrázek 2.16). V operačním systému Windows klepneme pravým tlačítkem myši na plochu nebo v nástroji Průzkumník Windows a následně vybereme položku **Nový/Složka** z kontextové nabídky (viz obrázek 2.17). Novou složku pojmenujeme.
2. Vytvoříme podsložky s ohledem na strukturu svých webových stránek (viz obrázky 2.16 a 2.18). Můžeme se kupříkladu rozhodnout vytvořit samostatnou složku pro každou část webových stránek, a to včetně případných podsložek.
3. Dobrým zvykem je vytvářet složku pro obrázky uvnitř kořenové složky webových stránek a potom do ní případně přidat podsložky, abychom své obrázky rozdělili podle sekcí nebo jiných kritérií. Alternativní přístup spočívá v tvorbě složky pojmenované například prostredky, kterou rozdělíme na podsložky s názvy kupříkladu obrazky, videa, sablony-stylu apod. (Více informací o šablonách stylů je k dispozici v kapitole 7, „Stavební kameny kaskádových stylů.")

Obrázek 2.16. V systému OS X vybereme položku **New Folder** a pojmenujeme si svou složku. Potom vytvoříme samostatnou složku pro každou část webových stránek



Obrázek 2.17. V operačním systému Windows klepneme pravým tlačítkem myši na plochu nebo v nástroji Průzkumník Windows a potom vybereme položku **Nový/Složka** z kontextové nabídky



Obrázek 2.18. Bude-li to žádoucí, můžeme rozdělit novou složku na podsložky

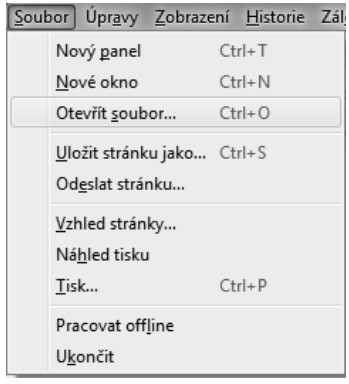
Tip: Používejte krátká popisná jména pro své soubory a složky. Slova v názvech oddělujte nejlépe spojovníky, a ne mezerami. Všechna písmena by měla být malá, aby bylo snadnější zapisovat vaše adresy URL. Více informací o správném pojmenování souborů najdete v části „Jména souborů" v kapitole 1, „Stavební kameny webových stránek."

Prohlížení stránek v prohlížeči

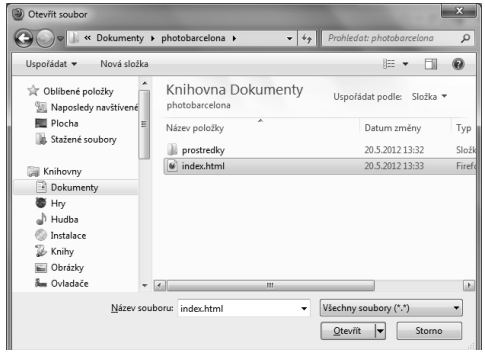
Jakmile vytvoříte stránku, určitě se budete chtít podívat, jak vypadá ve webovém prohlížeči. Jelikož nemůžete předvídat, jaký webový prohlížeč budou používat vaši návštěvníci, měli byste si prohlédnout svou stránku v několika webových prohlížečích. Ne všechny prohlížeče totiž zobrazují stránky stejně.

Stránku si zobrazíme ve webovém prohlížeči takto:

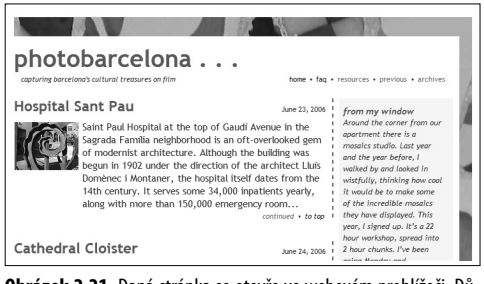
1. Otevřeme webový prohlížeč.
2. V závislosti na spuštěném webovém prohlížeči vybereme položku **Soubor/Otevřít**, **Soubor/Otevřít soubor** nebo **Soubor/Otevřít stránku** (viz obrázek 2.19).
3. Objeví se dialogové okno, v němž vyhledáme příslušnou složku na svém počítači, označíme požadovanou stránku a klepneme na tlačítko **Otevřít** (viz obrázek 2.20). Vybraná stránka se zobrazí ve webovém prohlížeči (viz obrázek 2.21) stejným způsobem, jako bychom ji publikovali na webový server (podrobněji se na téma publikování zaměříme v kapitole 21, „Publikování stránek na webu"). Tento postup se může lišit prohlížeč od prohlížeče.



Obrázek 2.19. Z nabídky prohlížeče (v tomto případě prohlížeče Firefox) vybereme položku **Soubor/Otevřít soubor**. V prohlížeči Internet Explorer plní stejnou funkci položka **Soubor/Otevřít**



Obrázek 2.20. Označíme soubor, jejž chceme otevřít, a klepneme na tlačítko Otevřít



Obrázek 2.21. Daná stránka se otevře ve webovém prohlížeči. Důkladně ji zkontrolujeme, abychom se přesvědčili, že vše postupuje podle našeho plánu

Tip: Většinou postačí poklepat na ikonu stránky, a tím ji otevřeme ve webovém prohlížeči. Pokud už máme webový prohlížeč otevřený, můžeme také přesunout ikonu stránky do okna prohlížeče. To je častokrát nejjednodušší způsob otevírání stránky ve webovém prohlížeči, když si na tento způsob zvykneme.

Tip: Některé moderní webové prohlížeče nemají položku **Soubor/Otevřít** ve své nabídce. V tomto případě můžeme zkusit výše popsanou metodu přesouvání ikony.

Tip: Pokud se naše webová stránka neobjevuje v dialogovém okně pro otevření souboru, měli bychom se přesvědčit, zda jsme ji uložili v textovém formátu a přidělili jí příponu *.htm* nebo *.html* (více informací lze najít v části „Ukládání naší webové stránky").

Tip: Nemusíme zavírat dokument v textovém editoru, než ho zobrazíme ve webovém prohlížeči, ale samozřejmě ho musíme uložit. Když změníme stránku v textovém editoru potom, co jsme ji otevřeli ve webovém prohlížeči, uložíme stránku v editoru znovu a ve webovém prohlížeči klepneme na tlačítko pro obnovení stránky. Tímto způsobem se změny projeví rovněž ve webovém prohlížeči. Mohli bychom také postupovat podle návodu na zobrazení stránky ve webovém prohlížeči, ale to by bylo zbytečně zdlouhavé.

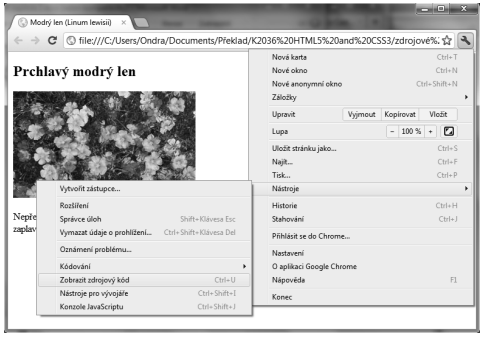
Tip: Naši návštěvníci neuvidí obsah webových stránek, dokud je nepublikujeme na webový server (o tom si povíme v kapitole 21, „Publikování stránek na webu").

Nechte se inspirovat od jiných

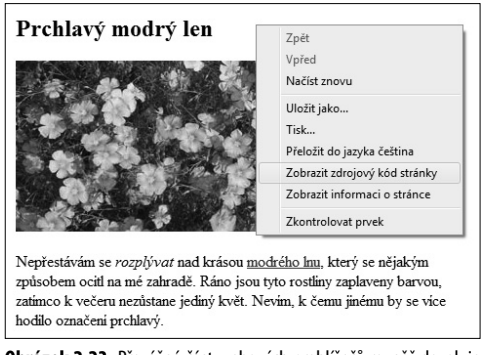
Nejjednodušším způsobem, jak se zdokonalit v jazyce HTML, je prohlédnout si stránky, které navrhli a vytvořili jiní vývojáři. Kód jazyka HTML je možné snadno prohlížet a učit se z něj. Neměli bychom však zapomínat, že textový obsah, grafické prvky, hudba, videa, šablony stylů a jiné externí soubory obvykle podléhají autorskému právu. Hlavní princip spočívá v tom, že se necháme cizími stránkami pouze inspirovat, ale potom vytvoříme své vlastní se svým vlastním obsahem.

Prohlížení zdrojového kódu stránek jiných vývojářů:

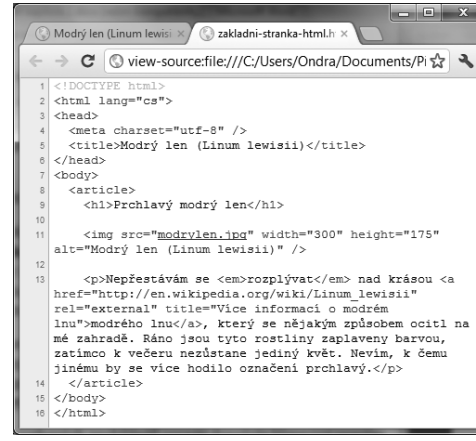
1. Otevřeme webovou stránku v libovolném prohlížeči.
2. Vybereme položku **Zobrazit zdrojový kód** (nebo podobnou položku u příslušného prohlížeče; viz obrázky 2.22 a 2.23). Po klepnutí na ni se objeví zdrojový kód jazyka HTML (viz obrázek 2.24).
3. Případně si můžeme stránku uložit pro pozdější studium.



Obrázek 2.22. Všechny webové prohlížeče na desktopových počítačích mají v nabídce položku, která nám umožňuje prohlížet si zdrojový kód aktuální stránky. Přesný název této položky závisí na konkrétním prohlížeči (například v prohlížeči Chrome se jmenuje **Nástroje/Zobrazit zdrojový kód**)



Obrázek 2.23. Převážná část webových prohlížečů rovněž dovoluje klepnout pravým tlačítkem myši do stránky, následkem čehož zobrazí kontextovou nabídku s položkou **Zobrazit zdrojový kód stránky** (nebo podobnou). To je mnohdy nejjednodušší způsob prohlížení zdrojového kódu, protože může být poměrně obtížné hledat tuto položku ve struktuře hlavní nabídky



Obrázek 2.24. Moderní webové prohlížeče zobrazují zdrojový kód na nové kartě (jako na tomto obrázku) nebo v novém okně, kdežto starší prohlížeče můžou otevřít nastavený textový editor. Barevné zvýraznění odlišuje obsah od elementů, atributů a jejich hodnot. Tato funkce se označuje termínem **zvýrazňování syntaxe**. Vlevo uvedená čísla řádků nejsou součástí daného kódu jazyka HTML, a ne všechny webové prohlížeče je zobrazují. Jedná se pouze o pomocné informace, které prohlížeč Chrome zobrazuje na své kartě se zdrojovým kódem stránky

Prohlížení kódu jiných vývojářů pomocí vývojářských nástrojů

Alternativně můžeme kód stránky prohlížet s pomocí vývojářských nástrojů webového prohlížeče. Každý webový prohlížeč má své vlastní nástroje, ale všechny mají některé společné funkce.

Tyto nástroje nabízejí interaktivnější zkoumání zdrojového kódu. Můžeme zkoumat kód jazyka HTML a CSS pro konkrétní části webové stránky, editovat ho přímo v prohlížeči a okamžitě pozorovat výsledky změn. Navíc můžeme tyto nástroje používat na jakýchkoliv stránkách, ne jen na těch našich. Změny v nich provedené jsou dočasné; tyto nástroje totiž nijak nezapisují do skutečných souborů načtené stránky. To je užitečné pro učení, protože můžeme zjistit, jak vývojáři dosáhli určitého efektu nebo si jen tak hrát s kódem a sle-

dovat, co se stane, aniž bychom se museli bát, že něco pokazíme.

Více informací o vývojářských nástrojích v moderních i starších prohlížečích si uvedeme v kapitole 20, „Testování a ladění webových stránek."

Tip: Neexistuje žádné pravidlo, které by diktovalo, kdo smí umístit webové stránky na web. To je právě na webu úžasné – jedná se o otevřené médium s malými vstupními bariérami. Na webu nepublikují jen experti, ale také naprostí nováčci. Na tuto skutečnost byste neměli zapomínat, když studujete kód jiných webových stránek. Pokud nějaký kód nevypadá pěkně, neměli byste se jím řídit jen proto, že jeho autor publikoval své webové stránky na webu, a vy ne. Můžete narazit na spoustu stránek, z nichž lze pochytit dobré postupy, ale také na celou řadu těch, které (slušně řečeno) nejsou ideální. Nezapomínejte tedy kriticky hodnotit a občas nahlédněte do této knihy nebo jiných zdrojů, pokud si nebudete jistí vhodností určité techniky.

Tip: Z okna (karty) se zdrojovým kódem můžete kód zkopírovat a vložit ho do svého textového editoru. Potom už jenom uložíte daný soubor.

Tip: Zdrojový kód webové stránky, a to dokonce včetně dostupných prostředků, můžete uložit ve většině webových prohlížečů výběrem položky **Soubor/Uložit jako** (nebo **Soubor/Uložit stránku jako**). Webový prohlížeč ale může v tomto případě přepsat některé části zdrojového kódu, takže nebude odpovídat originálu zcela přesně jako při postupu podle předchozího tipu.

Tip: Jak prohlížet kód jazyka CSS dané webové stránky se dozvíte v kapitole 8, „Práce s kaskádovými styly."

Základní struktura dokumentu HTML

3

V této kapitole se seznámíte s elementy jazyka HTML, které potřebujete pro tvorbu základu a uspořádání svých dokumentů. Jedná se o základní obalující elementy pro váš obsah.

V této kapitole se naučíte o:

- základní kostře webové stránky,
- koncepci dokumentu HTML5,
- elementech `h1` až `h6`, `hgroup`, `header`, `nav`, `article`, `section`, `aside`, `footer` a `div` (většina z nich jsou nové elementy jazyka HTML5),
- jak můžou role specifikace ARIA vylepšit přístupnost stránky,
- uplatnění atributů `class` a `id`,
- aplikování atributu `title` na elementy,
- přidávání komentářů do zdrojového kódu.

Čistá a konzistentní struktura tvoří nejen dobrý sémantický základ naší stránky, ale také usnadňuje aplikování kaskádových stylů. Styly jazyka CSS si popíšeme v kapitole 7, „Stavební kameny kaskádových stylů."

Pokud jste to dosud neučinili, měli byste si před pokračováním přečíst kapitolu 1, „Stavební kameny webových stránek." Tato kapitola ukazuje základní stránku jazyka HTML a vysvětluje některé základní koncepce. Jelikož nyní se prvně dostáváte k webové stránce, v této kapitole se bude opakovat několik informací z první kapitoly.

Základní kostra webové stránky

Každý dokument HTML by měl obsahovat následující komponenty (viz výpis 3.1):

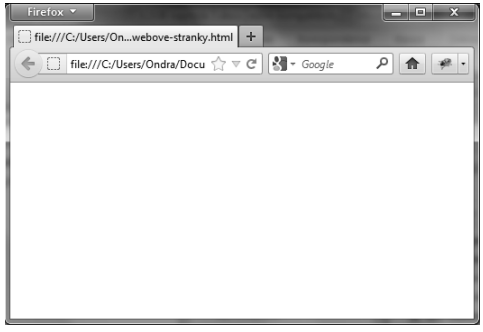
- definici typu dokumentu – `DOCTYPE`,
- element `html` s atributem `lang`, který je sice volitelný, ale doporučovaný,
- element `head`,
- specifikaci znakové sady v elementu `meta`,
- element `title` (jeho obsah doplníme za chvíli),
- element `body`.

Výpis 3.1. Zde je základní kostra pro všechny stránky HTML. Na odsazování příliš nezáleží, ale struktura je zásadní. V tomto příkladu volíme češtinu jako výchozí jazyk, a to nastavením hodnoty `cs` atributu `lang`. Znakovou sadu nastavujeme UTF-8

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8" />
  <title></title>
</head>
<body>

</body>
</html>
```

Na výše uvedeném výpisu se nachází ekvivalent prázdného listu papíru v jazyce HTML.

Tato stránka HTML totiž nemá žádný obsah uvnitř elementu body (viz obrázek 3.1).



Obrázek 3.1. Zobrazení minimální kostry dokumentu HTML ve webovém prohlížeči Firefox. Jak je patrné, není zde nic k vidění. Brzy však začneme přidávat obsah

Než budeme moct přidávat obsah a další informace, musíme tedy postavit základ naší stránky.

Základ stránky HTML5 postavíme takto:

1. Zápisem značky `<!DOCTYPE html>` prohlašujeme naši stránku za dokument jazyka HTML5.
2. Napíšeme značku `<html lang="kod-jazyka">`, čímž zahájíme samotný dokument HTML, přičemž výrazem *kod-jazyka* rozumíme kód výchozího jazyka, v němž budeme psát obsah naší stránky. Můžeme napsat kupříkladu značku `<html lang="cs">` pro češtinu, značku `<html lang="en">` pro angličtinu apod. Kompletní seznam dostupných kódů jazyka je k dispozici na adrese *http://www.w3schools.com/tags/ref_language_codes.asp*.
3. Zápisem značky `<head>` uvozujeme záhlaví dokumentu.
4. Pomocí značky `<meta charset="UTF-8" />` definujeme znakovou sadu UTF-8 pro náš dokument. Kdybychom chtěli, mohli bychom název této znakové sady napsat malými písmeny – `utf-8`. Rovněž mezera a lomítko na konci jsou volitelné, takže výraz `<meta charset="UTF-8">` by fungoval stejně. Samozřejmě můžeme zvolit jinou znakovou sadu, ale sada UTF-8 je nejuniverzálnější, takže jen zřídka budeme potřebovat jinou.
5. Napíšeme dvojici značek `<title></title>`. Mezi tyto značky později napíšeme název naší stránky. Název přidáme v části „Vytváříme název."
6. Zápisem značky `</head>` ukončujeme záhlaví dokumentu.
7. Tělo dokumentu uvozujeme značkou `<body>`. Za tuto značku patří obsah naší stránky.
8. Ponecháme několik prázdných řádků pro obsah, který budeme vytvářet ve zbytku této knihy.
9. Napíšeme značku `</body>`, čímž ukončíme tělo dokumentu.
10. Zápisem značky `</html>` ukončíme celý dokument HTML.

To bylo poměrně velké množství kroků. Protože však budou všechny vaše stránky začínat tímto způsobem, mohli byste použít jedinou stránku HTML jako vzor pro ostatní stránky, abyste se ušetřili psaní. Většina editorů kódu ve skutečnosti umožňuje nastavit si počáteční kód pro všechny nové stránky, což velmi ulehčuje práci. Toto nastavení byste měli hledat v možnostech nastavení nebo předvolbách svého editoru, pokud to tam nenajdete, tak nápověda daného editoru by mohla být užitečná.

Dvě části stránky – element head a element body

Jen pro rekapitulaci kapitoly 1, „Stavební kameny webových stránek" – stránky HTML se skládají ze dvou částí – z elementu `body` a elementu `head` (viz výpis 3.1). Element `DOCTYPE` na začátku každé stránky je jakýmsi úvodem.

V záhlaví dokumentu (v elementu `head`) definujeme název naší stránky, vkládáme do něj informace o naší stránce určené především vyhledávacím robotům, nahráváme zde šablony stylů, a občas také soubory s kódem jazyka JavaScript (ačkoliv z důvodu zlepšení efektivity je občas výhodnější načítat tyto soubory před koncovou značkou `</body>` na konci naší stránky). S příklady se budeme setkávat v průběhu této knihy. Obsah elementu `head` návštěvníci našich stránek nevidí; jedinou výjimkou je element `title`, jejž si popíšeme za okamžik.

Do elementu `body` vkládáme obsah naší stránky – text, obrázky, formuláře, audio, video a další interaktivní obsah. Jinými slovy – informace, které vidí naši návštěvníci. S elementy určenými pro obsah dokumentu HTML se setkáme ještě v mnoha kapitolách této knihy, ale na některé z nich se zaměříme už v této kapitole.

Tip: Specifikací výše uvedeného typu dokumentu `DOCTYPE` pro jazyk HTML5 si zajišťujeme, že webové prohlížeče zobrazí naši stránku správně. Kromě toho – nástroje pro validaci jazyka HTML posuzují správnost kódu na základě této specifikace typu. O těchto nástrojích pojednává kapitola 20, „Testování a ladění webových stránek."

Tip: Při specifikaci typu dokumentu není velikost písmen důležitá. Mohli bychom tudíž zapsat značku `<!doctype html>`, ale běžnější je zapisovat tuto značku jako ve výpisu 3.1 – `<!DOCTYPE html>`.

Tip: Element `html`, jenž následuje za specifikací typu dokumentu, musí obalovat všechny další elementy naší stránky (viz výpis 3.1).

Tip: Ujistěte se, že váš editor ukládá soubory se znakovou sadou UTF-8, aby znaková sada odpovídala zde definované sadě v elementu `meta`. Nebo obráceně – pokud definujete jinou znakovou sadu, uložte v ní i své soubory. Všechny editory nepoužívají znakovou sadu UTF-8 standardně, ale většina z nich alespoň umožňuje zvolit si požadovanou znakovou sadu (více informací najdete v kapitole 2, „Práce se soubory webových stránek"). Bez správné znakové sady uvidíte občas divné znaky ve svém obsahu, a to zejména místo písmen s diakritikou.

Tip: Nemusíme odsazovat zdrojový kód, který píšeme do elementu `head` (viz výpis 3.1). Odsazování kódu má však tu výhodu, že hned poznáme, kde element `head` začíná, co se nachází uvnitř něj a kde končí. Není žádnou výjimkou, že záhlaví dokumentu na některých stránkách výrazně naroste.

Vylepšená definice typu dokumentu u jazyka HTML5

Jelikož je jazyk HTML5 na světě, je o tolik jednodušší začínat s tvorbou webové stránky. Specifikace `DOCTYPE` je v jazyce HTML5 příjemně malá, a to především tehdy, když ji srovnáme se svými předchůdci.

Když byly na výsluní jazyky HTML 4 a XHTML 1.0, existovala spousta různých typů specifikací `DOCTYPE`, z nichž jsme si museli vybrat, přičemž jsme nevolili jen verzi jazyka, ale také jeho režim (Transitional, Strict atd.). To mělo jediný výsledek – definici typu dokumentu jsme museli odněkud zkopírovat, protože zapamatovat si ji bylo nemožné.

Takto kupříkladu vypadá definice typu dokumentu XHTML 1.0 v režimu Strict:

```
<!DOCTYPE html PUBLIC
"-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/ TR/xhtml1/DTD/
xhtml1-strict.dtd">
```

Naštěstí všechny webové prohlížeče (staré i nové) rozumí specifikaci `DOCTYPE` jazyka HTML5, takže ji můžeme používat pro všechny stránky a zapomenout, že kdy existovaly jiné. Jedinou výjimkou by mohlo být, kdybychom měli upravovat starší webové stránky, ale jejich vlastník by nám nedovolil změnit specifikaci `DOCTYPE` na verzi jazyka HTML5.

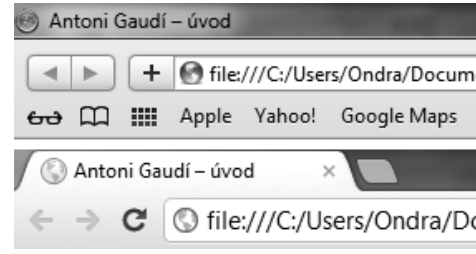
Vytváříme název

Základní kostra dokumentu HTML z předchozí části této kapitoly obsahovala prázdnou dvojici značek `<title></title>`, ale jen do chvíle, než se budeme bavit o elementu `title`. Nyní ta chvíle nastala.

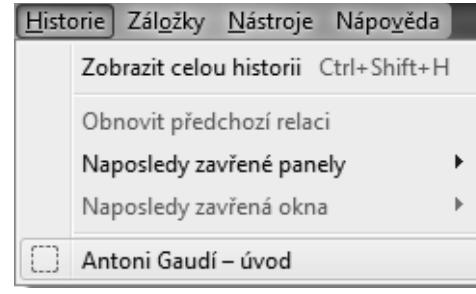
Každá stránka HTML musí mít svůj element `title`. Jinými slovy – musí mít svůj název, a ten by měl být krátký, výstižný a jedinečný pro jednotlivé stránky (viz výpis 3.2). V některých webových prohlížečích se název stránky objevuje v záhlaví okna, zatímco v jiných prohlížečích se zobrazuje jako název **karty** (nebo také **panelu**, tyto termíny lze zaměňovat). Dříve bylo obvyklejší, že se název zobrazoval v záhlaví okna prohlížeče, ale trendem moderních prohlížečů je zobrazovat ho jako název karty (viz obrázek 3.2). Název stránky se navíc návštěvníkovi zobrazuje v jeho výpisu historie a v jeho záložkách (viz obrázek 3.3).

Výpis 3.2. Element `title` musíme vložit do části `head`, a to za element `meta`, s nímž definujeme znakovou sadu naší stránky

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8" />
  <title>Antoni Gaudí – úvod</title>
</head>
<body>

</body>
</html>
```



Obrázek 3.2. V některých webových prohlížečích (například v prohlížeči Safari zobrazeném nahoře) se název stránky objevuje v záhlaví okna, zatímco v jiných prohlížečích (kupříkladu dole zobrazený prohlížeč Chrome) se zobrazuje jako název karty



Obrázek 3.3. Název stránky se rovněž objevuje v panelu historie, v seznamu oblíbených položek a v seznamu záložek

Pravděpodobně nejdůležitější ale je, že název stránky používají vyhledávače (Google, Yahoo!, Seznam atd.) při rozpoznávání obsahu stránky, a většinou se objeví i jako odkaz ve výsledcích vyhledávání (viz obrázek 3.4).

Obrázek 3.4. Pravděpodobně nejdůležitější na názvu stránky je to, že se zobrazuje jako text odkazu ve výsledcích vyhledávání vyhledávačů. Jeho znění má také vliv na umístění ve výsledcích vyhledávání. Zde můžete vidět, jak tento název stránky zobrazuje vyhledávač Google (obrázek však ukazuje název stránky pro příklad z původní nelokalizované knihy – „Antoni Gaudí - Introduction")

Krátce řečeno – jedinečný název stránky bychom měli volit proto, abychom zlepšili její pozici ve výsledcích vyhledávání, a také, aby měli naši návštěvníci lepší přehled o jejím obsahu.

Hlubší průzkum názvů stránek

Spousta vývojářů, a to dokonce i zkušených, věnuje malou pozornost elementu `title`. Jednoduše mu přidělí název svých webových stránek napříč všemi stránkami HTML, nebo dokonce hůře – ponechají v něm text, který do něj vložil jejich editor zdrojového kódu.

Vyhledávače používají různé algoritmy, s nimiž určují pořadí stránky ve výsledcích vyhledávání. Většinou má v těchto algoritmech velmi důležitou roli obsah elementu `title`. Vyhledávací roboti hledají v tomto elementu informace o tom, na co se daná stránka zaměřuje, a také zaznamenají obsah této stránky pro vyhledávání souvisejícího textu. Efektivní text elementu `title` by se měl skládat z několika slov, které souvisí s obsahem dané stránky.

Praxí osvědčeným postupem je volit text elementu `title` tak, aby shrnoval obsah dokumentu, což je velmi užitečná informace pro nástroje pro předčítání textu a vyhledávací roboty. Až v druhé řadě v něm můžeme uvést název našich webových stránek, což ale není povinné. Často se můžeme setkat s názvem webových stránek na začátku elementu `title`, ale lepším řešením je umístit jedinečný popis pro danou stránku na začátek.

Je vhodné omezit text elementu `title` na 60 znaků včetně mezer, protože vyhledávače obvykle zkracují délku tohoto textu ve svých výsledcích právě na tento počet znaků. Webové prohlížeče zkracují tento text na různé množství znaků, ale ne více než 60. Na kartách webových prohlížečů se zobrazuje ještě méně znaků, jelikož je na nich k dispozici méně volného místa.

Název stránky definujeme následovně:

1. Umístíme kurzor mezi značky `<title>` a `</title>` v záhlaví dokumentu.
2. Zadáme název naší webové stránky.

Tip: Element `title` je povinný.

Tip: Element `title` nemůže obsahovat žádné formátovací elementy, obrázky, odkazy na jiné stránky a jiné elementy jazyka HTML.

Tip: Některé editory zdrojového kódu předem vyplňují element `title` vlastním výchozím textem. Na to bychom si měli dávat pozor a nahradit výchozí text svým vlastním.

Tip: Pokud chceme v textu elementu `title` používat speciální znaky (kupříkladu písmena s diakritikou), měli bychom se ujistit, že jsme nastavili správnou znakovou sadu (což by se znakovou sadou UTF-8 neměl být problém), nebo je zapsat pomocí znakových entit (kompletní seznam je k dispozici na adrese *http://www.elizabethcastro.com/html/extras/entities.html*). Aby se speciální znaky uložily správně, nesmíme zapomenout nastavit svůj editor tak, aby ukládal stránky v odpovídající znakové sadě (více informací lze najít v kapitole 2, „Práce se soubory webových stránek").

Tvorba nadpisů

Jazyk HTML poskytuje šest úrovní nadpisů, s nimiž můžeme specifikovat hierarchickou strukturu informací na svých stránkách. Každý nadpis můžeme označit jedním z elementů v rozmezí `h1`–`h6`, přičemž element `h1` reprezentuje nadpis první úrovně, element `h2` představuje nadpis druhé úrovně (podnadpis nadpisu `h1`) atd. Nadpisy jsou jedny z nejdůležitějších elementů webové stránky.

Nadpisy `h1`–`h6` je možné srovnat s nadpisy jiných typů dokumentů, jako jsou například novinové články, produktové manuály apod. Při psaní takových dokumentů rozlišujeme hlavní část obsahu s nadpisem a také spoustu podnadpisů (a podnadpisů těchto podnadpisů atd.). Společně tyto nadpisy tvoří osnovu dokumentu. To rovněž platí o webových

stránkách (viz výpis 3.3). Více si řekneme v následující části této kapitoly.

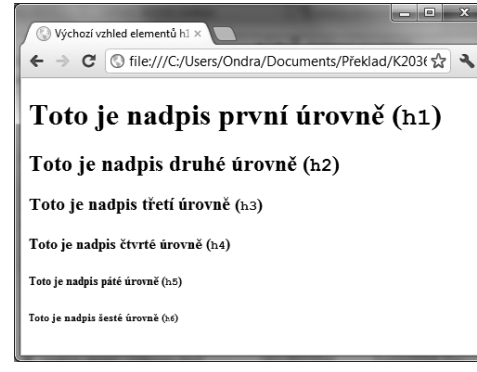
Výpis 3.3. Pomocí nadpisů definujeme strukturu dokumentu. V tomto příkladu jsou dva elementy `h2` s texty „La casa Milà" a „La Sagrada Família," což jsou podnadpisy elementu `h1` s textem „Antoni Gaudí." Hodnotou `es` jejich atributu `lang` sdělujeme prohlížeči, že jejich text je napsaný ve španělštině. Kdybychom nadpis „La Sagrada Família" označili elementem `h3`, byl by tento nadpis podnadpisem nadpisu „La casa Milà," a také podnadpisem nadpisu „Antoni Gaudí." Prázdný řádek mezi jednotlivými nadpisy není nutné psát a nemá žádný vliv na zobrazení stránky. Kdybychom měli nyní psát další obsah stránky (odstavce, obrázky, videa atd.), příslušné části by patřily za tyto nadpisy. Příklady si ukážeme za malou chvíli

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8" />
  <title>Antoni Gaudí – úvod</title>
</head>
<body>

  <h1>Antoni Gaudí</h1>

  <h2 lang="es">La Casa Milà</h2>

  <h2 lang="es">La Sagrada Família</h2>

</body>
</html>
```

Obsah webové stránky můžeme uspořádat prostřednictvím nadpisů takto:

1. Do elementu `body` našeho dokumentu HTML zapíšeme značku <h*n*>, přičemž *n* je číslo od 1 do 6 představující úroveň nadpisu. Nadpis `h1` je nejdůležitější – jedná se o nadpis první úrovně, zatímco nadpis `h6` je nejméně důležitý.
2. Napíšeme obsah nadpisu.
3. Doplníme koncovou značku </h*n*>, kde *n* je stejné číslo jako to, co jsme použili v prvním kroku.

Tip: Hlavním cílem nadpisů `h1`–`h6` je definovat osnovu naší stránky. Webový prohlížeč zobrazuje běžně nadpisy stále menším a menším písmem, jak postupujeme od nadpisu `h1` až k nadpisu `h6` (viz obrázek 3.5). Úrovně nadpisů bychom však měli volit výhradně na základě hierarchického uspořádání obsahu, a ne podle toho, jak velké písmo chceme nastavit. Tím učiníme naši stránku více sémantickou, což vylepší jak optimalizaci SEO, tak přístupnost. Vzhled nadpisů si můžeme přizpůsobit, jak potřebujeme – měnit písmo, velikost, barvu apod. Jak toho dosáhnout s pomocí jazyka CSS, si předvedeme v kapitole 10, „Formátování textu pomocí stylů."



Obrázek 3.5. Webový prohlížeč zobrazuje standardně všechny nadpisy tučným písmem, přičemž element `h1` je větší než element `h2`, a ten je větší než element `h3` atd. Jak však víme, vzhled není důležitý pro rozhodování, jakou úroveň nadpisu použít. Vzhled můžeme kdykoliv změnit pomocí jazyka CSS

Tip: Vyhledávače kladou na nadpisy velkou váhu, a to zejména na nadpisy `h1`. To ale neznamená, že bychom měli svou stránku přehltit těmito nadpisy, protože na to jsou vyhledávače moc chytré. Kromě toho nástroje pro předčítání textu umožňují uživatelům navigovat v obsahu stránky pomocí nadpisů. Díky nim můžou totiž používat klávesnici k rychlému přístupu k obsahu, který je zajímá, aniž by museli poslouchat celý obsah stránky. Existuje spousta dalších důvodů, proč zavádět logické hierarchické uspořádání pomocí nadpisů.

Tip: Úrovně nadpisů bychom měli specifikovat konzistentním způsobem v rámci celých webových stránek.

Tip: Nadpisu můžeme přidělit atribut `id`, pokud se na něj chceme přímo odkazovat odkazem (více informací je k dispozici v kapitole 6, „Odkazy").

Tip: Ve výpisu 3.3 jsme přidělili elementům h2 atribut lang, abychom prohlížeč upozornili na to, že jejich obsah je napsaný v jiném jazyce (ve španělštině reprezentované zkratkou es), než je výchozí jazyk celé stránky (čeština) definovaný značkou <html lang="cs">.

Struktura dokumentu v jazyce HTML5

V předchozí části jsme se dozvěděli, jak s elementy h1–h6 definovat osnovu stránky HTML. V této části se zaměříme důkladněji na to, jak elementy zavedené jazykem HTML5 můžou přispět k uspořádání dokumentu.

Víme tedy, že každý dokument HTML má jakousi svou osnovu vymezenou elementy nadpisů, a ta se podobá osnově v běžných tištěných dokumentech. Struktura dokumentu se běžně nikde nezobrazuje, ale stejně jako všechny sémantické elementy je důležitá pro vyhledávací roboty a předčítače textu. Tyto nástroje ji používají k nahlédnutí do obsahu stránky a jeho prezentaci uživatelům.

Ve starších verzích jazyka HTML, jsme mohli strukturu dokumentu popisovat výhradně pomocí elementů h1–h6. Jazyk HTML5 však zavádí čtyři elementy spadající do kategorie **rozdělující obsah** – article, aside, nav a section. Tyto elementy definují různé části dokumentu, a rovněž vymezují oblast obsahu, která se vztahuje k elementům h1–h6.

To znamená, že každý rozdělující element může mít svou hierarchickou strukturu elementů h1–h6, což je velký posun oproti dřívějším verzím jazyka HTML. Nejenže je v pořádku, když stránka obsahuje více než jeden element h1, ale přímo to doporučuje specifikace jazyka HTML5. Nepospíchejme však – brzy si ukážeme, proč je dobré držet počet nadpisů h1 na uzdě.

Všechny právě uvedené elementy ovlivňují strukturu dokumentu. Porovnejme si dvě shodné struktury, ať zjistíme, jak tyto elementy fungují. U obou případů bychom si ale měli představit, že za každým nadpisem následuje několik odstavců a další obsah příslušné části.

V první struktuře, která je zcela validní v jazyce HTML5 a znají ji i lidé se zkušenostmi se staršími verzemi jazyka HTML, používáme pouze nadpisy (viz výpis 3.4).

V její druhé verzi používáme jak nadpisy, tak elementy section jazyka HTML5, a to včetně vnořených elementů section (viz výpis 3.5).

Poznámka: Odsazování zdrojového kódu nás momentálně nezajímá a nijak neovlivňuje strukturu dokumentu, ale na druhou stranu velmi vypovídá o tom, jaké elementy k sobě patří.

Výpis 3.4. První verze struktury dokumentu

```
...
<body>
  <h1>Uživatelská příručka produktu</h1>
  <h2>Nastavení</h2>
  <h2>Základní funkce</h2>
  <h3>Přehrávání videa</h3>
  <h2>Pokročilé funkce</h2>
</body>
</html>
```

Výpis 3.5. Druhá verze struktury dokumentu (v podstatě stejná struktura jako v první verzi, ale se smysluplnějšími elementy)

```
...
<body>
  <h1>Uživatelská příručka produktu</h1>
  <section>
    <h1>Nastavení</h1>
  </section>

  <section>
    <h1>Základní funkce</h1>
    <section>
      <h1>Přehrávání videa</h1>
    </section>
  </section>
```

```
  <section>
    <h1>Pokročilé funkce</h1>
  </section>
</body>
</html>
```

Před chvíli jste se dozvěděli, že webové prohlížeče nezobrazují strukturu dokumentu. Můžete si však prohlédnout nástroj HTML 5 Outliner od Geoffreyho Sneddona (k dispozici na adrese *http://gsnedders.html5.org/outliner/*), což je jednoduchý, ale skvělý nástroj pro vizuální prezentaci struktury dokumentu. Vyzkoušíte-li si tento nástroj na kódech z výpisů 3.4 a 3.5, zjistíte, že přestože se oba liší, osnova dokumentu zůstává stejná:

1. Uživatelská příručka produktu
    1. Nastavení
    2. Základní funkce
        1. Přehrávání videa
    3. Pokročilé funkce

Jak je patrné, každý element `section` se stává podsekcí svého nejbližšího elementu `h1`–`h6` nebo rodičovského rozdělujícího elementu (což je v tomto případě také element `section`). Stejně se chovají všechny čtyři před chvílí zmíněné rozdělující elementy (`article`, `aside`, `nav` a `section`), a to i když je kombinujeme dohromady.

Pro srovnání – kdybychom z druhé verze naší struktury (viz výpis 3.5) vypustili všechny elementy `section`, výsledná struktura (řekněme třetí verze; viz výpis 3.6) by vypadala jinak.

Výpis 3.6. Třetí verze struktury dokumentu (jiná než první a druhá verze)

```
...
<body>
  <h1>Uživatelská příručka produktu</h1>
  <h1>Nastavení</h1>
  <h1>Základní funkce</h1>
  <h1>Přehrávání videa</h1>
  <h1>Pokročilé funkce</h1>
</body>
</html>
```

Konkrétně se liší v tom, že každý nadpis je stejně důležitý (všechny značíme jako elementy `h1`), což znamená, že zde neexistují žádné podnadpisy:

1. Uživatelská příručka produktu
2. Nastavení
3. Základní funkce
4. Přehrávání videa
5. Pokročilé funkce

Obě významově shodující se struktury jsou validní v jazyce HTML5, ale druhá z nich je lepší, protože elementy `section` poskytují více sémantiky. V praxi bychom obalili obsah druhé verze do elementu `article`, protože je to vhodnější pro náš příklad. Výsledná struktura by se mírně lišila, jak ukazuje výpis 3.7.

Výpis 3.7. Stejná struktura s dokonalejší sémantikou

```
<body>
  <article>
    <h1>Uživatelská příručka produktu</h1>
    <section>
      <h1>Nastavení</h1>
    </section>

    <section>
      <h1>Základní funkce</h1>
      <section>
        <h1>Přehrávání videa</h1>
      </section>
    </section>

    <section>
      <h1>Pokročilé funkce</h1>
    </section>
  <article>
</body>
</html>
```

Neměli bychom zapomínat, že za každým nadpisem by měl následovat související text, obrázky a jiný obsah, o němž se budeme do-

zvídat postupně v této knize. Tentokrát jsme však tento obsah opomněli, aby vynikly nadpisy a struktura dokumentu.

V dnešním ekosystému dělejte, co můžete

Na chvíli se zastavme – náš zdrojový kód potřebuje ještě jednu úpravu. Před okamžikem jsme si řekli, že počet elementů `h1` bychom měli držet na uzdě. Ačkoliv může každý rozdělující element obsahovat svou hierarchickou strukturu nadpisů počínaje elementem `h1`, není nutné takto postupovat. Ve skutečnosti je mnohem lepší zahájit každou část stránky odpovídající úrovní nadpisu – například elementem `h2`, pokud jeho rodičovská část začínala elementem `h1`.

Důvod si popíšeme nyní.

Web se neustále vyvíjí. Nové specifikace (jako například specifikace jazyka HTML5) se mění každý den, dokud nejsou prohlášeny za finální, což může trvat roky a u jazyka HTML5 se tomu tak ještě nestalo. Stále vznikají nové a nové verze webových prohlížečů. Také vznikají nové verze nástrojů pro předčítání textu a jiných pomocných technologií. Nic z toho však nevzniká v přesném souladu.

Místo toho jednotlivé webové prohlížeče získávají nové dovednosti postupně (což je většinou dobré), ale nemusí se jednat o stejné dovednosti jako u konkurence (to není už tak dobré), a rozhodně je nezískávají ve stejnou dobu jako u konkurence. To samé platí pro nástroje pro předčítání textu. Třebaže moderní webové prohlížeče podporují spoustu funkcí jazyka HTML5, žádný z nich neposkytoval v době psaní této knihy strukturu dokumentu HTML5 nástrojům pro předčítání textu, a tím pádem ji tyto nástroje netlumočily uživatelům.

Krátce řečeno – nástroje pro předčítání textu a jiné pomocné technologie prozatím nerozlišují mezi elementem `h1` umístěným uvnitř elementu `body`, a tím, jenž se nachází uvnitř elementu `article`, `aside`, `nav` nebo `section`. Všechny tyto elementy `h1` tudíž považují za nadpisy nejvyšší úrovně. Bruce Lawson, uznávaný webový nadšenec, informoval o této skutečnosti ve svém článku na adrese *http://www.brucelawson.co.uk/2009/headings-in-html-5-and-accessibility/*. Při čtení tohoto článku však dávejte pozor, protože některé informace v něm obsažené už můžou být zastaralé.

Uživatelé nástrojů pro předčítání textu samozřejmě nemůžou čekat, až si web vyřeší své problémy. Stále budou používat nadpisy jako osnovu pro navigování uvnitř stránky. Smysluplné hierarchické uspořádání nadpisů by jim mělo usnadnit život a uživatelský prožitek z vašich webových stránek.

Dokud se ekosystém trochu nevyčistí, měli byste používat nadpisy `h1`–`h6`, které explicitně označují hierarchickou strukturu dokumentu, jako by v dokumentu žádné rozdělující elementy nebyly. Spousta odborníků v oboru tento postup doporučuje. Prohlédněte si výpis 3.8.

Výpis 3.8. Pátá verze struktury dokumentu (doporučovaný přístup)

```
...
<body>
  <article>
    <h1>Uživatelská příručka produktu</h1>
    <section>
      <h2>Nastavení</h2>
    </section>

    <section>
      <h2>Základní funkce</h2>
      <section>
        <h3>Přehrávání videa</h3>
      </section>
    </section>

    <section>
      <h2>Pokročilé funkce</h2>
```

```
    </section>
  <article>
</body>
</html>
```

Co dříve bývaly elementy `h1` uvnitř první úrovně elementů `section`, jsou nyní elementy `h2`. Nadpis „Přehrávání videa," jenž se nachází uvnitř elementu `section` na druhé úrovni, je nyní elementem `h3`, a ne elementem `h1`. Struktura dokumentu se vůbec nezměnila – změnily se jen úrovně nadpisů.

V tomto příkladu jsme si ukázali pouze elementy `h1` až `h3`. Můžeme však používat také elementy `h4` až `h6`, pokud to obsah našich stránek vyžaduje. Například podnadpis nadpisu „Přehrávání videa" bychom označili jako element `h4` atd.

Nezapomínejte, že toto doporučení se týká všech rozdělujících elementů (`article`, `aside`, `nav` a `section`), nejen těch, které jsme si představili v tomto příkladu.

Shrnutí

Pokud jste něčemu nerozuměli, měli byste si tuto část, týkající se struktury dokumentu HTML5, přečíst ještě jednou. Není to tak náročné, jak se může zdát. Zkuste si vytvořit několik testovacích stránek a porovnat výsledky nástroje HTML 5 Outliner, abyste lépe pochopili, jak struktura dokumentu funguje. Během práce na svém projektu můžete tento nástroj používat rovněž. V prvé řadě validujte své stránky na adrese *http://validator.w3.org/*, na níž zjistíte, jestli vaše stránky obsahují nějaké chyby. Více informací najdete v kapitole 20, „Testování a ladění webových stránek."

Tip: Nenabývejte dojmu, že musíte vždy použít element `article` a uvnitř něj elementy `section`. Předchozí příklad pouze demonstroval jeden způsob použití těchto elementů. Ve skutečnosti byste mohli tento obsah označit celou řadou jiných způsobů a stále by se jednalo o validní kód jazyka HTML5. O elementech `article` a `section` se dozvíte více později v této kapitole, a uvidíte, že existuje několik způsobů jejich uplatnění v závislosti na obsahu stránky.

Jak prostředky pro definování struktury dokumentu v jazyce HTML5 pomáhají uspořádat obsah

V této části kapitoly jsme si řekli, že každá část dokumentu specifikovaná elementem `article`, `aside`, `nav` nebo `section` může mít svou osnovu – počínaje nadpisem `h1` až po nadpis `h6`.

To přináší ohromnou flexibilitu do tvorby nadpisů, ale také to má ještě jednu nepříliš zřejmou výhodu – náš obsah se může objevit na jiné stránce, nebo dokonce na jiných serverech, aniž by narušil osnovu svého rodičovského dokumentu. Navíc jeho struktura zůstává netknutá.

V dnešní době si mezi sebou sdílí obsah spousta webových stránek. Můžeme narazit na webové stránky, které sjednocují novinky z celého světa, blogy s vlastními kanály RSS, kanály sociální sítě Twitter atd. Jak zjistíme za chvíli, element `article` tvoří samostatnou jednotku, kterou je možné začlenit do jiné stránky.

Kupříkladu bychom mohli zobrazit následující článek na jiných webových stránkách:

```
...
<h2>Novinky ze světa</h2>

<article>
  <h1>Místní teenageři upřednostňují
    gramofonové desky před
    kompaktními disky</h1>

  <p>Místní teenagerka nahradila
    své digitální nahrávky
    analogovými. Tvrdí
    o sobě, že jde s dobou.</p>

  <h2>Posedlost prvním albem</h2>
  ...
</article>
...
```

Výstup nástroje HTML 5 Outliner by vypadal takto:

1. Novinky ze světa
    1. Místní teenageři upřednostňují gramofonové desky před kompaktními disky
        1. Posedlost prvním albem

Přestože nadpis „Místní teenageři...“ je nadpisem první úrovně, jedná se o podnadpis prvního elementu `h2`, protože se nachází uvnitř článku pod tímto nadpisem. Nadpis „Posedlost prvním albem“ není na stejné úrovni jako nadpis „Místní teenageři...,“ ale je jeho podnadpisem a současně je pod-podnadpisem prvního nadpisu „Zprávy ze světa.“

Ať nastavíme nadpisu „Zprávy ze světa“ jakoukoliv úroveň, osnova zůstane stejná. To samé platí o nadpisech „Místní teenageři...“ a „Posedlost prvním albem,“ dokud bude úroveň prvního z těchto dvou nadpisů větší než úroveň druhého.

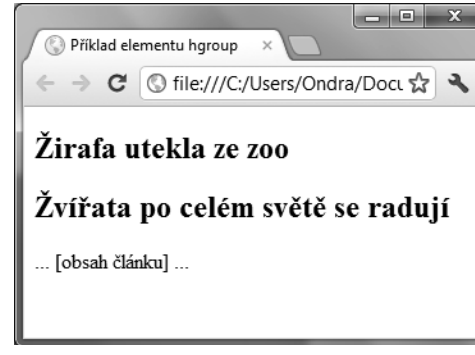
Seskupování nadpisů

Někdy se stává, že nadpis má spoustu podnadpisů. Tyto nadpisy můžeme seskupit dohromady pomocí elementu `hgroup`, jímž obalujeme související nadpisy (viz výpis 3.9). Skupina nadpisů musí obsahovat minimálně dva nadpisy `h1` až `h6`, avšak žádné jiné elementy.

Výpis 3.9. Seskupujeme dva související nadpisy. V tomto příkladu máme element `h2`, jenž je podnadpisem hlavního nadpisu článku. Jelikož jsme oba nadpisy označili jako skupinu, zobrazí se v osnově pouze nejvyšší nadpis „Žirafa utekla ze zoo.“ Oba nadpisy se však zobrazí ve webovém prohlížeči zcela normálně (viz obrázek 3.6). Kdybychom dovnitř elementu `hgroup` zapsali další element `h1`, dopadlo by to s ním stejně jako se zmíněným elementem `h2` – nezobrazil by se v osnově. Jelikož se tento element `h2` nezobrazuje v osnově, jako další nadpis uvnitř článku bychom mohli zvolit element `h2` (místo elementu `h3`), a na ten bychom nahlíželi jako na podnadpis elementu `h1` s textem „Žirafa utekla ze zoo“

```
...
<body>

<article>
  <hgroup>
    <h1>Žirafa utekla ze zoo</h1>
    <h2>Zvířata po celém světě se radují
    </h2>
  </hgroup>

  <p>... [obsah článku] ...</p>
</article>

</body>
</html>
```

V osnově dokumentu se zobrazuje pouze první výskyt nadpisu nejvyšší úrovně ve skupině `hgroup`. To může být další rozhodující faktor při rozhodování, kdy použít element `hgroup`. Všechny nadpisy skupiny `hgroup` se však zobrazují ve webovém prohlížeči (viz obrázek 3.6).



Obrázek 3.6. Oba nadpisy se zobrazují ve webovém prohlížeči, jako by vůbec nebyly součástí skupiny `hgroup`

Více nadpisů seskupíme následujícím způsobem:

1. Napíšeme značku `<hgroup>`.
2. Napíšeme značku `<hn>`, přičemž *n* je číslo od 1 do 6, a to v závislosti na důležitosti nadpisu, který chceme vytvářet.
3. Napíšeme obsah nadpisu.
4. Napíšeme značku `</hn>`, kde *n* je stejné číslo jako v druhém kroku.

5. Zopakujeme kroky 2 až 4 pro všechny nadpisy, které by měly být součástí skupiny `hgroup`. Úroveň nadpisu pro všechny následující nadpisy by měla vzrůstat (například od elementu `h1` k elementu `h2` atd.).
6. Napíšeme značku `</hgroup>`.

Tip: Nepoužívejte skupinu `hgroup` jen pro jeden nadpis. Měla by obsahovat přinejmenším dva.

Tip: Jak už jsme si řekli, v osnově dokumentu se zobrazuje pouze první instance nadpisu nejvyšší úrovně. Pořadí nadpisů je vedlejší. Kdyby tedy skupina `hgroup` obsahovala element `h3` následovaný elementem `h2`, element `h2` by se zobrazil v osnově. Samozřejmě běžně budeme řadit nadpisy podle jejich úrovně, aby méně důležité (kupříkladu element `h3`) nepředbíhaly ty důležitější (například element `h3`). Občas můžeme narazit na nějakou výjimku.

Běžné komponenty stránek

Bezpochyby jste navštívili desítky webových stránek, které byly uspořádané jako ty, které lze spatřit na obrázku 3.7. Když si odmyslíte jejich obsah, je možné rozpoznat čtyři hlavní komponenty – záhlaví s nabídkou, článek v hlavní oblasti s obsahem, postranní panel se souvisejícími informacemi a zápatí (viz obrázek 3.8).

Obrázek 3.7. Obvyklé rozvržení s hlavní navigací nahoře, obsahem vlevo, postranním panelem vpravo a zápatím dole. Aby stránka takhle vypadala, musíme použít jazyk CSS

Obrázek 3.8. Typy informací, které lze běžně najít na stránce. Jedná se pouze o jeden způsob rozvržení, přestože velmi typický

Stránku nemůžete nastylovat a uspořádat jako na obrázcích 3.7 a 3.8 bez pomoci jazyka CSS. S jazykem CSS se seznámíte v kapitole 7, „Stavební kameny kaskádových stylů," formátovat text a nastavovat barvy s jeho pomocí se naučíte na začátku kapitoly 10, „Formátování textu pomocí stylů, a konečně – vícesloupcové rozvržení vytvoříte v kapitole 11, „Rozvržení pomocí stylů."

Význam jednotlivých komponent stránky je však stejný bez ohledu na její rozvržení. Tyto komponenty prozkoumáme ve zbytku této kapitoly. Budeme-li postupovat po stránce odshora dolů, zjistíme, jak používat elementy

header, nav, article, section, aside a footer a následně také, jak používat element div jako obecný obalující element pro dodatečné stylování a jiné účely. S výjimkou elementu div neexistoval žádný z uvedených elementů až do verze HTML5. Některé z nich už jsme používali v předchozích ukázkových kódech.

Když se budete učit o těchto elementech, nezaměřujte se na to, kde se zobrazují v ukázkových rozvrženích, ale raději na jejich význam.

Dále v této kapitole rovněž narazíme na jiné elementy; například na element ul (neuspořádaný seznam) nebo element a (odkaz). Do detailu si je však vysvětlíme až v následujících kapitolách,

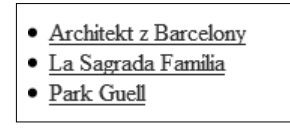
Vytváříme záhlaví

Jestliže máme na naší stránce část s úvodním textem nebo navigací, měli bychom ji označit elementem header, jenž definuje záhlaví.

Stránka může obsahovat libovolné množství elementů header, přičemž jejich význam se může lišit v závislosti na jejich kontextu. Například element header v horní části stránky může reprezentovat záhlaví celé stránky (viz výpis 3.10). Záhlaví stránky většinou obsahuje logo, hlavní nabídku (viz obrázek 3.9), další odkazy a vyhledávací formulář. Toto je bezpochyby nejčastější způsob použití elementu header, ale není jediný.

Výpis 3.10. Tento element header reprezentuje záhlaví stránky. Obsahuje seznam odkazů uvnitř elementu nav, jenž sděluje, že se jedná o hlavní nabídku na stránce. Ve výpise 3.11 přidělíme tomuto elementu header volitelný atribut role s hodnotou banner za účelem vylepšení přístupnosti

```
...
<body>
<header>
  <nav>
    <ul>
      <li><a href="#gaudi">
        Architekt z Barcelony</a></li>
      <li><a href="#sagrada-familia">
        La Sagrada Família</a></li>
      <li><a href="#park-guell">
        Park Guell</a></li>
    </ul>
  </nav>
</header>
</body>
</html>
```



Obrázek 3.9. Hlavní element header obsahující navigaci

S elementem header můžeme označovat taktéž úvodní obsah a navigaci hlouběji uvnitř stránky. Příkladem může být obsah části (viz výpis 3.11).

Element header je jedním ze čtyř elementů z kategorie rozdělující obsah, o nichž jsme se již bavili. To znamená, že jakýkoliv element h1–h6 uvnitř elementu header má jako svůj kontext právě tento element, a ne celou stránku, alespoň co se týká osnovy. Element header tudíž obvykle obsahuje své nadpisy (elementy h1–h6; případně včetně elementu hgroup), ale není to nutné. Kupříkladu můžeme pozorovat nadpisy ve výpisu 3.11, ale ne ve výpisu 3.10.

Záhlaví vytvoříme následovně:

1. Umístíme kurzor dovnitř elementu, v němž chceme vytvořit záhlaví.
2. Napíšeme značku <header>.
3. Zapíšeme obsah záhlaví, což může být obsah různého typu označený elementy jazyka HTML, které si popíšeme ve zbytku této knihy. Záhlaví může kupříkladu obsahovat elementy h1–h6, logo nebo sérii log, navigaci, vyhledávací formulář apod.
4. Napíšeme značku </header>.