Tiffany B. Brown, Kerry Butters, Sandeep Panda

sitepoint

Html5

OKAMŽITĚ

computer press

OVLÁDNĚTE HTML5 ZA VÍKEND

Tiffany B. Brown, Kerry Butters, Sandeep Panda

HTML5 Okamžitě

Computer Press
Brno
2014

HTML5 Okamžitě

Tiffany B. Brown, Kerry Butters, Sandeep Panda

Překlad: Ondřej Baše
Odpovědný redaktor: Martin Herodek
Technický redaktor: Jiří Matoušek

Authorized Czech translation of the English edition of Jump Start HTML5, ISBN 9780980285826

Objednávky knih:
http://knihy.cpress.cz
www.albatrosmedia.cz
eshop@albatrosmedia.cz
bezplatná linka 800 555 513
ISBN 978-80-251-4296-7

Vydalo nakladatelství Computer Press v Brně ve společnosti Albatros Media a. s. se sídlem Na Pankráci 30, Praha 4. Číslo publikace 18 618.

1. vydání

ALBATROS MEDIA a.s.

Obsah

O autorech

Tiffany B. Brown

Tiffany B. Brown je webová vývojářka na volné noze a spisovatelka odborné literatury, která žije v Los Angeles. Ve webovém odvětví pracuje již více než deset let. Než založila svou vlastní konzultantskou společnost Webinista, pracovala v týmu Developer Relations & Tools společnosti Opera Software. Nyní poskytuje služby v oblasti webového vývoje a souvisejícího poradentství pro agentury a menší týmy návrhářů.

Kerry Butters

Kerry Butters[1] je spisovatelka odborné literatury pocházející z Anglie. Díky svým technickým i spisovatelským znalostem píše o nejrůznějších odborných tématech – včetně webdesignu a technické podpory společností. Vede rovněž agenturu markITwrite[2], jež se zaměřuje na tvorbu digitálního obsahu. Kromě toho ráda poznává nové technologie a celkově vzato je opravdovým geekem.

Sandeep Panda

Sandeep Panda je webový vývojář a spisovatel, který se zajímá především o jazyky JavaScript a HTML5. Má více než čtyři roky zkušeností s programováním pro web. Rád zkouší nové technologie a učí se nové věci. Když zrovna neprogramuje, obvykle hraje počítačové hry nebo poslouchá hudbu.

1 *https://plus.google.com/u/0/+KerryButters/posts*

2 *http://www.markitwrite.com/*

Úvod

Jazyk HTML (HyperText Markup Language; česky hypertextový značkovací jazyk) je dominantním jazykem na webu. Kdykoliv si prohlížíte nějakou webovou stránku ve svém webovém prohlížeči nebo s ní pracujete, je velmi pravděpodobné, že se jedná o dokument jazyka HTML. Tento jazyk původně vznikl, aby bylo možné formátovat a sdílet vědecké materiály. V současné době ho používají téměř všechny typy dokumentů a také slouží pro tvorbu vizuálně zajímavých rozhraní pro webové aplikace.

S příchodem verze HTML5 se z „obyčejného" značkovacího jazyka HTML stalo plnohodnotné **rozhraní API** (**aplikační programové rozhraní**) pro vývoj webových aplikací. V této knize si uvedeme řadu historických údajů o vývoji jazyka HTML a také si popíšeme jeho nové funkce.

Jazyk HTML5 vylepšuje elementy starších verzí tohoto jazyka. S jeho novými typy vstupních polí můžeme vytvářet komplexní formuláře, aniž bychom k tomu potřebovali jazyk JavaScript. Například můžeme používat ovládací prvek jezdec jednoduše tak, že zapíšeme do dokumentu značku `<input type=range>`. K tomu navíc přibyly další typy vstupních polí, jakými jsou kupříkladu `email` nebo `url`, které nám zajistí validaci na straně klienta. S pomocí elementů `audio` a `video` můžeme vkládat zvukové a videosoubory do našich dokumentů. Oba tyto elementy nabízejí aplikační rozhraní, pomocí něhož můžeme vytvářet multimediální přehrávače nebo zajímavé vizuální efekty. To všechno můžeme dělat i bez podpory jakéhokoliv zásuvného modulu ve webovém prohlížeči.

V jazyce HTML5 můžeme dokonce kreslit, a to díky elementu `canvas` a podpoře formátu SVG (Scalable Vector Graphics; česky škálovatelná vektorová grafika). Element `canvas` reprezentuje rozhraní API pro kreslení bitmapové grafiky a je určené pro tvorbu 2D a 3D obrázků, grafů a her. Naproti tomu SVG je formát pro vektorovou grafiku, s nímž je možné opětovně používat, měnit velikost a jinak upravovat obrázky, které se budou zobrazovat na mnoha zařízeních s různými rozměry obrazovek.

Největší změnou specifikace HTML5 ale je to, že přináší rozhraní API, která jsou součástí objektového modelu dokumentu jazyka HTML, ale neexistují pro ně žádné odpovídající značky, které bychom mohli zapsat do dokumentu. Jedná se o rozhraní API určená výhradně pro model DOM, s nimiž můžeme pracovat v kódu jazyka JavaScript – sdílet a odebírat data nebo vytvářet aplikace, které budou umět určit, kde se nachází jejich uživatelé. Rozhraní Web Workers napodobuje procesy s více vlákny a úlohy na pozadí v jazyce JavaScript. Rozhraní Geolocation API umí určovat pozici. Díky podpoře komunikace mezi dokumenty můžeme odesílat data mezi dokumenty, nebo dokonce mezi doménami, aniž bychom museli zveřejňovat model DOM. Události spouštěné serverem a rozhraní WebSockets umožňují uskutečňovat komunikaci mezi klientem a serverem v (téměř) reálném čase.

Po přečtení této knihy budete mít základní znalosti o výše uvedených funkcích a budete na nejlepší cestě k tvorbě úžasných webových stránek a aplikací v jazyce HTML5.

Komu je tato kniha určena

Třebaže je tato kniha určená pro vývojáře, kteří začínají pracovat s jazykem HTML5, není obsahově úplná. Z tohoto důvodu předpokládá alespoň základní znalost jazyka HTML. Pokud s webovým vývojem začínáte, zkuste si nejprve přečíst knihu „Build Your Own Website Using HTML and CSS“[3] od nakladatelství SitePoint.

Jak budete postupovat touto knihou, narazíte na několik pokročilejších témat – například na nejrůznější rozhraní API a aplikace určené pro použití offline. Pro bezproblémové pochopení těchto částí knihy byste měli znát jazyk HTML a mít alespoň základní znalost jazyka JavaScript a objektového modelu dokumentu (modelu DOM). Není nutné znát jazyk JavaScript důkladně. Měli byste alespoň znát obsluhu událostí, datové typy, cykly (`while` apod.) a podmíněné příkazy (`if-else` apod.). Příklady v této knize budou však jednoduché a vysvětlené řádek po řádku. Pokud neznáte jazyk JavaScript, zkuste si přečíst knihu „JavaScript Okamžitě“[4] od nakladatelství Computer Press. Webové stránky Mozilla Developer Network[5] také poskytují skvělé materiály o jazyku JavaScript a modelu DOM.

Použité konvence

V této knize narazíte na řadu typografických konvencí a různých stylů rozvržení, které budou zvýrazňovat různé typy informací. Prohlédněte si proto následující konvence.

Ukázky zdrojového kódu

Zdrojový kód bude napsán neproporcionálním písmem; například takto:

```
<h1>Skvělý letní den</h1>
<p>Byl krásný letní den, který přímo vybízel na procházku parkem. Ptáčci
zpívali a všechny děti byly zpátky ve škole.</p>
```

Pokud se daný zdrojový kód nachází v archivu zdrojových kódů k této knize, příslušné jméno souboru se objeví nad tímto výpisem:

priklad.css

```
.zapati {
  background-color: #CCC;
  border-top: 1px solid #333;
}
```

3 *http://www.sitepoint.com/store/build-your-own-website-the-right-way-using-html-css-3rd-edition/*

4 *http://knihy.cpress.cz/javascript-okamzite.html*

5 *https://developer.mozilla.org/en-US/*

Pokud bude výpis zobrazovat jen část obsahu souboru, bude označen slovem *úryvek*:

priklad.css (úryvek)

```
border-top: 1px solid #333;
```

Když budeme k současnému příkladu přidávat další zdrojový kód, takový kód se bude zobrazovat tučně:

```
function animuj() {
  var nova_promenna = 'Ahoj';
}
```

Místo již napsaného zdrojového kódu bude výpis obvykle obsahovat tři tečky:

```
function animuj() {
  ...
  return nova_promenna;
}
```

Tipy, poznámky a varování

Tip: Haló, vy tam!

Tipy vám poskytují malé cenné rady.

Poznámka: Ehm, promiňte...

Poznámky jsou doplňující informace, které se týkají tématu, ale nejsou kriticky důležité.

Upozornění: Nezapomeňte vždy...

... věnovat pozornost těmto sdělením.

Nezbytné nástroje

K vývoji dokumentů v jazyce HTML5 budete potřebovat pouze textový editor pro psaní a webový prohlížeč pro prohlížení své práce. Nepoužívejte ale textové procesory (například aplikaci Word apod.). Tyto programy jsou určené pro psaní dokumentů, ale ne pro programování. K programování budete potřebovat editor, který umí číst a zapisovat prostý text.

Na operačním systému Windows vyzkoušejte kupříkladu volně dostupný textový editor Notepad++[6]. Pokud používáte operační systém Mac OS X, zkuste program TextWrangler[7] od společnosti Bare Bones Software. Program Brackets[8] představuje alternativu jak pro uživatele

6 *http://notepad-plus-plus.org/*

7 *http://www.barebones.com/products/textwrangler/*

8 *http://brackets.io/*

operačního systému Windows, tak pro uživatele operačního systému Mac OS X. Jestliže používáte operační systém Linux, vyzkoušejte si editor gEdit, který je součástí distribuce Ubuntu, nebo případně použijte volně dostupný editor Bluefish[9]. K dispozici jsou samozřejmě rovněž placené programy, které mnohdy poskytují více funkcí než volně dostupné editory.

Neobejdete se bez webového prohlížeče s podporou jazyka HTML5, pokud vám mají příklady z této knihy fungovat. Ujistěte se, že používáte nejnovější verzi prohlížeče Google Chrome, Firefox, Internet Explorer, Safari nebo Opera. Výchozím prohlížečem operačního systému Windows je prohlížeč Internet Explorer, zatímco u operačního systému Mac OS X se jedná o prohlížeč Safari. Ostatní prohlížeče si můžete stáhnout z webových stránek jejich výrobců.

Pro závěrečné kapitoly této knihy budete potřebovat také webový server. Volně dostupnými webovými servery pro operační systémy Windows, Mac OS X a Linux jsou Apache HTTP Server[10], Nginx[11] nebo Lighttpd[12]. V případě, že pracujete v operačním systému Mac OS X, můžete si zkusit nainstalovat balík MAMP[13], který obsahuje databázový systém MySQL, server Apache a interpreter jazyka PHP. Uživatelé operačního systému Windows můžou vyzkoušet balík WAMP[14] nebo XAMP[15], což jsou alternativy pro tento operační systém.

Jelikož je kniha černobílá, případné zmínky o barvách nemusí být na obrázcích patrné. Pokud však příslušnou část kódu implementujete v praxi, uvidíte na obrazovce monitoru odpovídající podobu barevně.

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.

9 *http://bluefish.openoffice.nl/index.html*
10 *http://httpd.apache.org/*
11 *http://wiki.nginx.org/Main*
12 *http://www.lighttpd.net/*
13 *http://www.mamp.info/en/*
14 *http://www.wampserver.com/en/*
15 *http://www.apachefriends.org/index.html*

Zdrojové kódy ke knize

Z adresy *http://knihy.cpress.cz/K2165* 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/K2165* po klepnutí na odkaz Soubory ke stažení.

KAPITOLA 1

Co je HTML5?

V této kapitole:
- Stručná historie jazyka HTML5
- Co HTML5 není
- Pár slov o specifikaci HTML5

Jednoduchá odpověď zní: „Poslední verze jazyka HTML.“ To nám ale příliš neřekne. HTML5 konkrétně:

- definuje parsovací algoritmus pro generování konzistentního stromu DOM (objektového modelu dokumentu), a to dokonce z nejednoznačného nebo hůře označkovaného dokumentu,
- přidává nové elementy pro podporu multimédií a webových aplikací,
- upravuje sémantický význam mnoha elementů jazyka HTML.

S jazykem HTML5 můžeme vkládat audio- a videosoubory do dokumentů HTML. Můžeme používat jazyk SVG. Můžeme vytvářet robustní formuláře s nativní kontrolou chyb. Můžeme vytvářet hry, grafy a animace prostřednictvím elementu `canvas`. Dokumenty spolu můžou komunikovat pomocí výměny zpráv. Jinými slovy: HTML5 je spíše celá **aplikační platforma** než prostý značkovací jazyk.

Stručná historie jazyka HTML5

Podrobný příběh, který stojí za vznikem jazyka HTML5, je příliš dlouhý, než abychom si ho mohli říct v této knize celý. Několik základních informací ale může přispět k lepšímu pochopení, jak tento jazyk vznikl.

Jazyk HTML má své kořeny v jazyku SGML (Standard General Markup Language). Jazyk SGML si lze představit jako sadu pravidel pro tvorbu značkovacích jazyků.

Jazyk HTML tudíž vznikl jako aplikace jazyka SGML, a to na počátku 90. let 20. století. Reprezentoval jednotný způsob definice struktury hypertextových dokumentů. Termín **hypertext** označuje „text, který obsahuje odkazy na jiné texty“ – tj. text, který není lineární[16].

16 *http://www.w3.org/WhatIs.html*

Tím, že popisujeme strukturu dokumentu, oddělujeme jeho obsah od vzhledu – tzn. od toho, jak ho prezentujeme koncovému uživateli. Díky tomu ho můžeme snadněji sdílet a upravovat. Šíření našeho dokumentu na Internetu napomáhá protokol HTTP (Hypertext Transfer Protocol).

Počátky jazyka HTML

První verze jazyka HTML přinesla jednoduchý zápis založený na označení struktury dokumentu značkami – jednalo se pouze o základní strukturu dokumentu. Použití odstavce (elementu `p`) nebo položky seznamu (elementu `li`) nevyžadovalo zápis koncové značky. Rané verze[17] tohoto jazyka dokonce neobsahovaly elementy `img` a `table`. Podpora obrázků se objevila až ve verzi 1.2.[18]

Gramatika jazyka HTML se lehce pozměnila ve verzi 2.0.[19] Koncové značky elementů `p` a `li` se staly obvyklé. Největší změny se však udály mezi verzemi 2.0 a 3.2.

Ve verzi 3.2 jsme mohli začít měnit písmo pomocí elementu `font`. Mohli jsme doplňovat interaktivitu prostřednictvím elementu `applet` a appletů napsaných v jazyce Java. Tabulková data bylo možné strukturovat pomocí elementů `table`, `tr` a `td`. Pravděpodobně nejvýraznější změnou této verze ale bylo zavedení šablon stylů.

K založení převážné části webových stránek ale došlo až s příchodem verze HTML 4. Od této verze jsme mohli začít sdělovat webovému prohlížeči, jak by měl parsovat naše dokumenty, a to tak, že jsme nastavili typ dokumentu. Verze 4 nabízela tři typy dokumentů:

- Přechodová verze (Transitional), která dovolovala používat jak zastaralé elementy z jazyka HTML 3.2, tak nové elementy jazyka HTML 4.
- Striktní verze (Strict), v níž jsme mohli používat jen elementy jazyka HTML 4.
- Verze s rámy (Frameset), jež umožňuje vkládat do dokumentu další dokumenty pomocí elementů `frame`.

Specifikace jazyka HTML od verze 1 do verze 4 ale neříkala jasně, jak by měly prohlížeče parsovat kód jazyka HTML.

Konsorcium W3C přestalo pracovat na specifikaci HTML 4 v roce 1998 a místo toho se začalo věnovat náhradě tohoto jazyka – jazyku XHTML.

Výlet do světa jazyka XHTML

Jazyk XHTML 1.0[20] vznikl díky přeformulování specifikace HTML 4 do jazyka XML. Jazyk XML (eXtensible Markup Language) je zjednodušenou verzí metajazyka SGML – nabízí striktnější pravidla pro zápis a parsování značkovaného kódu.

17 *http://info.cern.ch/hypertext/WWW/MarkUp/MarkUp.html*

18 *http://www.w3.org/MarkUp/draft-ietf-iiir-html-01.txt*

19 *http://www.w3.org/MarkUp/html-spec/*

20 *http://www.w3.org/TR/xhtml1/*

Jazyk XHTML kupříkladu požadoval zápis značek malými písmeny, kdežto jazyk HTML povoloval malá písmena, velká písmena i kombinaci obou. Jazyk XHTML vyžadoval koncové značky pro všechny párové elementy, kterými jsou například elementy `p` nebo `li`. Prázdné (nepárové) elementy (kupříkladu `br` nebo `img`) musely končit posloupností znaků `/>`. Bylo nutné uzavírat všechny hodnoty atributů do uvozovek a správně kódovat ampersandy do entit. Všechny stránky musely být typu MIME `application/xml+xhtml`.

Jazyk XHTML nás naučil, jak psát kvalitnější kód. Scházela mu ale bezchybná podpora napříč webovými prohlížeči.

Specifikace XForms[21] byla považována za nástupce formulářů v jazyce HTML. Představila elementy `upload` a `range` pro pestřejší interakci s našimi webovými stránkami.

Tato specifikace si však nezískala příliš velkou pozornost. Koneckonců – proč vlastně zavádět speciální jazyk pro tvorbu webových formulářů? Proč raději nevylepšit jazyk HTML?

Bitva o světovou DOMinanci

Společnost Netscape vydala v roce 1996 webový prohlížeč Netscape Navigator 2.0, který podporoval dvě nové technologie – jazyk JavaScript a model DOM (Document Object Model). Obvykle o nich mluvíme, jako by se jednalo o jedinou věc, ale není tomu tak – model DOM je rozhraní API pro práci s dokumenty HTML. Jazyk JavaScript je naproti tomu oblíbený jazyk pro interakci s tímto rozhraním API.

Rozhraní DOM od společnosti Netscape Navigator proměnilo každý element na stránce HTML na objekt, který je možné vytvořit, přesunout, upravit nebo smazat z kódu skriptovacího jazyka. Díky tomu bylo možné začít obohacovat webové stránky o další interaktivní prvky a animace – přestože by si návštěvníci počkali na jejich stažení pěkně dlouho, když měli modemy o rychlosti 14,4 Kb/s.

Model DOM představoval tak význačný přínos pro web, že ostatní webové prohlížeče se ho snažily také co nejrychleji implementovat. Jejich implementace se ale mírně lišila. Například prohlížeč Netscape Navigator používal objekt `document.layers` pro přístup k celým skupinám uzlů dokumentu HTML. Prohlížeč Internet Explorer od společnosti Microsoft přišel s objektem `document.all`. Weboví vývojáři pak trávili roky nad tím, aby se jejich kód choval v obou těchto prohlížečích stejně. Prohlížeče Opera a prohlížeče s jádrem WebKit si vzaly příklad z prohlížeče Internet Explorer – začaly používat objekt `document.all`.

Z modelu DOM ve verzi 0 se stal standard s jasnou specifikací až s příchodem specifikace Document Object Model (DOM) Level 1.[22] Místo objektů `document.layers` a `document.all` jsme mohli nově používat metody `document.getElementById()` a `document.getElementsByTagName()`. Všechny moderní webové prohlížeče podporují model DOM.

21 *http://www.w3.org/TR/2007/REC-xforms-20071029/*

22 *http://www.w3.org/TR/REC-DOM-Level-1/*

Applety a zásuvné moduly

Do všeho tohoto zmatku – vývoje jazyka HTML, modelu DOM a přechodu na jazyk XHTML – se přidaly rovněž applety a zásuvné moduly webových prohlížečů. Budiž jim k dobru, že doplňovaly funkčnost, které nebylo možné dosáhnout s jazykem HTML. Například zásuvné moduly RealPlayer a QuickTime od společnosti Apple obohatily web o audio- a videosoubory. Díky appletům jazyka Java jsme mohli spouštět tabulkový procesor přímo ve webovém prohlížeči. Se zásuvnými moduly Flash od společnosti Macromedia (nyní společnosti Adobe) a Shockwave jsme mohli dělat všechny výše uvedené aktivity, a navíc vytvářet animace.

Applety a zásuvné moduly měly ale tři hlavní nevýhody:

1. Uživatelé, kteří neměli nainstalovaný požadovaný zásuvný modul (nebo applet), neviděli příslušný obsah.
2. Applety a zásuvné moduly často otevíraly řadu bezpečnostních děr.
3. Obvykle se jednalo o komerční produkty, za jejichž používání museli vývojáři platit poplatky.

Aby toho nebylo málo, zásuvné moduly a applety většinou způsobovaly pády svého hostitelského prostředí – webového prohlížeče.

Na webu se bylo možné často setkávat s následující situací:

- Webové prohlížeče nezacházely s kódem jazyka HTML stejným způsobem.
- Nové značkovací jazyky přinesly několik výhod oproti jazyku HTML, ale vše bylo vykoupené dodatečnými režijními náklady na implementaci.
- Zásuvné moduly a applety nabízely dodatečnou funkčnost, ale představovaly bezpečnostní riziko, měly negativní vliv na stabilitu webových prohlížečů a vyžadovaly od vývojářů platbu poplatků.

Všechny tyto problémy řeší jazyk HTML5:

- Obsahuje funkce a gramatická pravidla zavedená jazyky XHTML a XForms.
- Téměř eliminuje nutnost používat zásuvné moduly a s nimi spojené problémy se stabilitou a bezpečností.

Co HTML5 není

Připouštím, že v této knize jsem zvolil poněkud puristický přístup. Z HTML5 se stalo populární slovo (buzzword) pro označení „všeho, co můžeme nově dělat ve webovém prohlížeči, ale co jsme nemohli dělat dříve". V této knize se však pod ním skrývají pouze nové elementy jazyka HTML a různá rozhraní API modelu DOM.

Nebudeme si podrobně popisovat nové funkce specifikace CSS Level 3. Na druhou stranu se budeme zabývat koncepcí, která se standardně skrývá pod obecným termínem **JavaScript**, ale

přesněji řečeno se jedná o rozhraní DOM HTML API. Budeme si také popisovat vektorovou grafiku ve formátu SVG, ale jen ve spojení s kódem jazyka HTML.

Tato kniha by měla být stručným úvodem do jazyka HTML5. Z tohoto důvodu se nebudeme detailně zabývat pokročilými tématy. Získáme ale přehled, co tento jazyk přináší nového oproti svým starším verzím.

Pár slov o specifikaci HTML5

Specifikaci HTML5 vydává jak skupina WHATWG (Web Hypertext Application Technology Working Group), tak konsorcium W3C (World Wide Web Consortium). Tyto dvě skupiny spolupracovaly na tvorbě této specifikace řadu let, přičemž vznikala jednotná specifikace, na kterou dohlížel jediný editor. V roce 2011 se však rozešly. Z tohoto důvodu existují dvě verze (třebaže velmi podobné) specifikace HTML5. Každá má svého vlastního editora.

Verze specifikace[23] od skupiny WHATWG je „živoucí" dokument. Neustále v něm přibývají nové funkce, mění se a výjimečně mizí, a to obvykle na základě mnoha diskuzí v komunitě okolo této specifikace. Tato specifikace je mnohem dynamičtější – podléhá mnoha změnám.

Konsorcium W3C zvolilo jiný přístup. Editoři této specifikace se také baví s komunitou o změnách, ale každý dokument musí projít několika fázemi – od fáze „Working Draft" přes fázi „Candidate Recommendation" až po finální fázi „W3C Recommendation". Specifikace W3C tedy mají své verze. Společná verze specifikace HTML5 z roku 2011 se stala výchozí specifikací[24] konsorcia W3C. Následné revize jsou součástí specifikace HTML 5.1.[25]

Obě specifikace se samozřejmě vzájemně liší – některé rozdíly jsou nepatrné, ale jiné jsou vcelku markantní. Rozdíly však nejsou dobře popsané. Jelikož tato kniha se nezabývá podrobnostmi specifikace HTML5, tyto rozdíly budeme převážně ignorovat.

23 *http://www.whatwg.org/specs/web-apps/current-work/multipage/*

24 *http://www.w3.org/TR/html5/*

25 *http://www.w3.org/html/wg/drafts/html/master/Overview.html*

KAPITOLA 2

Základy: Anatomie HTML5

V této kapitole:
- Náš první dokument HTML5
- Dva režimy syntaxe jazyka HTML5

Každý dokument HTML se skládá z elementů a elementy jsou reprezentované značkami. Značky jsou posloupnosti znaků, které vyznačují, kde element začíná a končí.

Všechny značky začínají levou ostrou závorkou (<) a končí pravou ostrou závorkou (>). Každý element má **počáteční značku** (které se někdy říká **otevírací značka**), jež začíná znakem <, za nímž následuje název elementu (nebo jeho zkratka). Za názvem elementu můžou následovat atributy a jejich hodnoty. Atributu můžeme přidělit hodnotu pomocí znaku =. Některé atributy můžou být prázdné (bez hodnoty). Hodnotou prázdného atributu je pravdivostní hodnota `true`. Ukažme si příklad elementu `input`:

```
<input type="text" name="jmeno" disabled>
```

V tomto případě jsme použili atributy `type`, `name` a `disabled`. Prvním dvěma jsme přiřadili explicitní hodnoty, ale atribut `disabled` je prázdný. Některé elementy můžou mít prázdné atributy, což jsou obvykle atributy, které by běžně přijímaly hodnoty `true` nebo `false`. Hodnotou takového atributu je hodnota `true` nebo `false` jednoduše podle toho, jestli je přítomný samotný název atributu, nebo naopak schází, a to bez ohledu na jeho hodnotu. Jinými slovy – výrazy `disabled="true"` a `disabled="false"` by oba zakázaly dané vstupní pole, protože je přítomný název `disabled`.

Většina elementů má také uzavírací značku. Uzavírací značka začíná znakem <, za nímž následuje lomítko (/). Až potom zapisujeme název elementu a pravou ostrou závorku (>). Některé elementy jsou ale **prázdné elementy**. Tyto elementy nemůžou mít obsah a nemají uzavírací značku. Výše uvedený element `input` je příkladem prázdného elementu.

Protože jsme si vysvětlili, co to jsou značky, podíváme se blíže na dokument jazyka HTML5.

Náš první dokument HTML5

Otevřete si svůj oblíbený textový editor a napište do něho níže uvedený kód. Výsledek uložte do souboru *ahoj.html*.

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

Právě jste napsali svůj první dokument HTML5. Ačkoliv není příliš zajímavý, demonstruje základy jazyka HTML5.

Na prvním řádku se nachází povinná deklarace typu dokumentu `<!DOCTYPE html>`. Tímto způsobem sdělujeme webovému prohlížeči, že pracuje s dokumentem jazyka HTML5. Kdybychom ji vynechali, riskujeme, že prohlížeč zpracuje náš dokument špatně. Proč? Protože umí přepínat `DOCTYPE`.

Přepínání `DOCTYPE` znamená, že prohlížeče parsují a zobrazují dokumenty rozdílnými způsoby – na základě jejich deklarace typu dokumentu (pokud narazí na hlavičku odpovědi `Content-type: text/html`). Většina prohlížečů implementuje vlastní způsob přepínání `DOCTYPE`, aby mohly zobrazovat dokumenty napsané s nestandardními funkcemi prohlížečů nebo pro zastaralé specifikace.

Například specifikace HTML 4.01 a XHTML 1.0 poskytují spoustu režimů (striktní, přechodové a s rámy), které lze nastavovat deklarací `DOCTYPE`. Například striktní režim jazyka HTML 4.01 bychom nastavili pomocí následující deklarace:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
  "http://www.w3.org/TR/html4/strict.dtd">
```

Přechodové deklarace `DOCTYPE` spouštějí **quirks režim**. Prohlížeče v tomto režimu parsují dokumenty odlišně – podle svých vlastních chyb a odchylek od webových standardů.

Striktní deklarace `DOCTYPE` spouštějí **standardní režim**. Všechny prohlížeče v tomto režimu parsují dokumenty na základně odsouhlasených specifikací jazyka HTML a CSS.

Chybějící deklarace `DOCTYPE` ale také spouští quirks režim. Z tohoto důvodu definovala specifikace HTML5 nejkratší možnou deklaraci `DOCTYPE`. V této specifikaci lze najít vysvětlení:

> „Deklarace DOCTYPE jsou z historických důvodů nezbytné. Když je vynecháme, prohlížeče můžou zapnout režim, který není kompatibilní s některými standardy. Díky deklaraci DOCTYPE se budou prohlížeče řídit odpovídajícími standardy."

Zápisem typu dokumentu jazyka HTML5 (`<!DOCTYPE html>`) spouštíme standardní režim, a to dokonce i ve starších prohlížečích, které neznají jazyk HTML5.

Dva režimy syntaxe jazyka HTML5

Jazyk HTML5 nabízí dva režimy syntaxe: po vzoru jazyka HTML a po vzoru jazyka XML. Rozdíl spočívá v tom, zda vygenerujeme dokument s hlavičkou `Content-type: text/html` nebo s hlavičkou `Content-type: application/xml+xhtml`.

Pokud je dokument typu `text/html`, měla by pro něj platit níže uvedená pravidla:

- Počáteční značky nejsou povinné pro všechny elementy.
- Koncové značky nejsou povinné pro všechny elementy.
- Pouze prázdné elementy lze ukončovat řetězcem `/>`. Jedná se například o elementy `br`, `img` a `link`.
- Nezáleží na velikosti písmen ve značkách a atributech.
- Hodnoty atributů nemusíme vkládat do uvozovek.
- Některé typy atributů nemusí mít hodnotu (například atribut `checked` nebo `disabled`).
- Speciální znaky a entity není nutné kódovat.
- Dokument HTML5 musí obsahovat správný `DOCTYPE`.

Syntaxe jazyka HTML

Ukažme si další dokument HTML5.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset=utf-8>
  <title>Ahoj</title>
  <!--
    Toto je ukázkový komentář.
    Níže uvedené řádky předvádějí, jak vkládat kaskádové styly.
  -->
  <link rel=stylesheet href=styly.css type=text/css>
  <style>
    body {
      background: aliceblue;
    }
  </style>
</head>
<body>
  <p>
    <img src=kvetina.jpg alt=Květina>
    Není to nádherná květina?
  <p>
    Ano, je to nádherná květina. Jaký je to druh?
  <script src=foo.js></script>
```

```
</body>
</html>
```

Na prvním řádku se opět nachází deklarace `DOCTYPE`. Podobně jako u jiných značek jazyka HTML5 u ní není důležitá velikost písmen. Pokud neradi posouváte ruku na tlačítko *Shift*, můžete psát jen `<!doctype html>`. Jestliže máte v oblibě klávesu `Caps Lock`, můžete psát `<!DOCTYPE HTML>`.

Následuje element `head`. Tento element většinou obsahuje informace o samotném dokumentu; například jeho název nebo znakovou sadu. V předchozím kódu obsahuje element `meta` pro definici znakové sady. Volba znakové sady je volitelná, ale vždy můžete nastavit jen jedinou, přičemž doporučená je znaková sada UTF-8.[26]

Poznámka: Používejte znakovou sadu UTF-8

Ujistěte se, že váš textový editor ukládá soubory ve znakové sadě UTF-8 bez BOM a umí používat unixové/linuxové konce řádků.

Náš element `head` rovněž obsahuje název dokumentu (`<title>Ahoj</title>`). V převážné většině webových prohlížečů se text mezi značkami elementu `title` zobrazuje jako popisek okna nebo karty pro aktuální stránku.

Komentáře v jazyce HTML jsou části textu, které se nezobrazují ve webovém prohlížeči. Lze je číst jen ve zdrojovém kódu – díky nim si zapisujeme poznámky pro sebe nebo pro naše spolupracovníky přímo do dokumentu. Některé programy pro generování kódu jazyka HTML umí rovněž vkládat komentáře. Komentáře se můžou objevit kdekoliv v dokumentu HTML. Každý z nich musí začínat posloupností `<!--` a končit posloupností `-->`.

Záhlaví dokumentu může rovněž obsahovat elementy `link`, které odkazují na externí prostředky, jak jsme si ukázali. K prostředkům patří šablony stylů, ikony webových stránek nebo vlákna RSS. Prostřednictvím atributu `rel` definujeme vztah mezi naším dokumentem a odkazovaným prostředkem. V tomto případě načítáme šablonu stylů – jinak řečeno soubor CSS. CSS je jazyk, s nímž můžeme popsat, jak by měl dokument vypadat.

Kromě toho můžeme kaskádové styly vkládat pomocí elementu `style` (s počáteční značkou `<style>` a koncovou značkou `</style>`). Externí šablonu stylů načítanou elementem `link` lze však vkládat do více stránek.

Mimochodem – elementy `meta` a `link` jsou příkladem prázdných elementů, které můžeme ukončovat také posloupností znaků `/>`. Kupříkladu značka `<meta charset=utf-8>` by mohla vypadat takto: `<meta charset=utf-8 />`; není to však nutné.

26 *http://www.w3.org/International/questions/qa-choosing-encodings*

Uvozovky okolo hodnot atributů v jazyce HTML5

Ve výše uvedeném příkladu jsme nezapisovali hodnoty atributů do uvozovek. V souboru *ahoj.html* jsme naopak uvozovky používali. Oba zápisy jsou v jazyce HTML5 platné – navíc můžeme používat jak dvojité uvozovky (`"`), tak jednoduché uvozovky (`'`).

Buďte však opatrní, pokud používáte hodnoty atributů bez uvozovek. Jednoslovné hodnoty nemusí mít uvozovky. Seznam hodnot oddělených mezerami už ale musí být uzavřen do uvozovek. Kdyby tomu tak nebylo, webový prohlížeč by považoval jen první uvedenou hodnotu za hodnotu atributu, zatímco zbylé hodnoty by považoval za prázdné atributy. Prohlédněte si následující řádek kódu:

```
<code class=php zvyraznisyntaxi><?php echo 'Ahoj!'; ?></code>
```

Jelikož jsme nezapsali hodnoty atributu `class` mezi uvozovky, webový prohlížeč je interpretuje takto:

```
<code class="php" zvyraznisyntaxi><?php echo 'Ahoj!'; ?></code>
```

Názvem třídy je tudíž jen slovo `php`, k němuž jsme nechtěně vytvořili prázdný atribut `zvyrazni syntaxi`. Aby se obě hodnoty staly názvy třídy, museli bychom přepsat deklaraci atributu následovně: `class="php zvyraznisyntaxi"` (nebo `class='php zvyraznisyntaxi'`).

Osekaný dokument HTML5

Na základě pravidel jazyka HTML (která platí i ve verzi HTML 4) nemusíme psát u některých elementů počáteční nebo koncové značky. Tyto elementy jsou implicitní. I když je do našeho zdrojového kódu nezapíšeme, webový prohlížeč se chová, jako by se tak stalo. Teoreticky bychom mohli přepsat náš ukázkový soubor *ahoj.html* následujícím způsobem:

```
<!DOCTYPE html>
<head>
  <meta charset=utf-8>
  <title>Ahoj</title>
  <p>Ahoj
```

Když webový prohlížeč sestavuje **strom uzlů**, automaticky doplní chybějící element `body`.

To, že můžete vypustit koncové značky, neznamená, že byste to měli dělat. Webový prohlížeč musí vygenerovat model DOM v každém případě. Koncové značky snižují riziko, že prohlížeč interpretuje strukturu dokumentu špatně. Párové značky navíc velmi usnadňují hledání a opravu chyb – zejména jestliže používáte textový editor, jenž umí zvýrazňovat syntaxi. Pokud pracujete v rozsáhlém týmu nebo v redakčním systému, používání počátečních a koncových značek zvyšuje pravděpodobnost, že váš kód jazyka HTML bude spolupracovat s kódem od někoho jiného. Ve zbytku této knihy proto budete narážet jak na počáteční, tak na koncové značky, přestože budou volitelné.

Poznámka: Počáteční a koncové značky

Pokud chcete přesně zjistit, které elementy vyžadují počáteční a koncové značky, prostudujte si příručku „HTML: The Markup Language (an HTML language reference)“[27] od konsorcia W3C. Tyto informace jsou k dispozici i na webových stránkách Web Platform Docs.[28]

„XHTML5“: syntaxe ve stylu jazyka XML

Kód jazyka HTML5 je možné psát i podle přísnějších pravidel podobných jazyku XML. V kapitole 1 jsme si řekli, že jazyk XHTML 1.0 vznikl přeformulováním jazyka HTML 4 do podoby jazyka XML 1.0. Pro striktnější variantu jazyka HTML5 se někdy nepřesně používá označení „XHTML5“. XHTML5 lze chápat jako jazyk HTML5 zapsaný na základě syntaktických pravidel jazyka XML, přičemž příslušné dokumenty se obvykle vydávají s hlavičkou `Content-type: application/xml+xhtml`.

Pro jazyk „XHTML5“ platí následující pravidla:

- Všechny elementy musí mít počáteční značku.
- Neprázdné elementy musí mít koncovou značku (například elementy `p` nebo `li`).
- Všechny prázdné elementy by měly být ukončené posloupností znaků `/>`.
- U názvů značek a atributů záleží na velikosti písmen – obvykle se vše zapisuje malými písmeny.
- Hodnoty atributů musí být uzavřené do uvozovek.
- Prázdné atributy jsou zakázané (místo pouhého výrazu `checked` musíme použít definici `checked="checked"` nebo `checked="true"`).
- Speciální znaky je nutné zapisovat do znakových entit.

Počáteční značka `<html>` musí obsahovat atributu `xmlns`. Kdybychom přepsali výše uvedený dokument do syntaxe jazyka XML, vypadal by přibližně takto:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta charset="utf-8" />
  <title>Ahoj</title>
</head>
<body>
  <p>
    <img src="kvetina.jpg" alt="Květina" />
    Není to nádherná květina?
  </p>
  <script src="foo.js />
</body>
</html>
```

27 *http://www.w3.org/TR/html-markup/*

28 *http://docs.webplatform.org/wiki/Main_Page*

Nyní jsme přidali jmenný prostor XML pomocí atributu `xmlns`, čímž jsme prohlížeči naznačili, že se budeme držet striktnější syntaxe. Lomítkem uzavíráme dokonce i prázdné elementy `meta` a `img`. Podle pravidel jazyka XML musíme uzavřít všechny elementy buď koncovou značkou, nebo mezerou, lomítkem a pravou ostrou závorkou (`/>`).

V předchozím příkladu jsme také uzavřeli element `script` přímo. Mohli bychom použít i běžnou koncovou značku `</script>`. Element `script` je poněkud zvláštní. Můžeme s ním vkládat kód JavaScriptu přímo do dokumentu tak, že ho zapíšeme mezi počáteční značku `<script>` a koncovou značku `</script>`. V takovém případě musíme uvést koncovou značku.

Můžeme se ale také pouze odkazovat na externí soubory s kódem, a to prostřednictvím elementu `script` s atributem `src`. Pokud to tak provedeme na běžné stránce typu `text/html`, musíme použít uzavírací značku `</script>`. Jestliže ale používáme stránku typu `application/xml+xhtml`, můžeme napsat jen počáteční značku uzavřenou znaky `/>`.

Nesmíme však zapomínat na to, že aby prohlížeč uplatnil pravidla jazyka XML, musíme dokument odeslat ze serveru v odpovědi s hlavičkou `Content-type: application/xml+xhtml`. V tomto případě prohlížeče poznají, že se jedná o dokument XHTML5, i když bude scházet deklarace `DOCTYPE`.

Poznámka: Konfigurace serveru

Aby váš webový server odesílal hlavičku `Content-type: application/xml+xhtml`, musíte ho tak nastavit. Pokud si platíte za hostingové služby, je velmi pravděpodobné, že váš poskytovatel tuto hlavičku už nastavil pro soubory s příponou `.xhtml`. V takovém případě by vám stačilo jen přejmenovat soubor *dokument.html* na *dokument.xhtml*. Pokud to nebude fungovat, zkuste se obrátit na svého poskytovatele hostingových služeb nebo na dokumentaci svého webového serveru.

Jak je patrné, syntaktická pravidla jazyka XML jsou příliš pedantská. Je mnohem jednodušší používat typ MIME `text/html` a jeho volnější syntaxi jazyka HTML.

KAPITOLA 3

Základy: Strukturování dokumentů

V této kapitole:
- Element article
- Skládáme kousky dohromady
- Element section
- Další elementy dokumentu

Jazyk HTML5 přináší několik elementů, které umožňují rozbíjet dokument na více obsahových částí, přičemž tyto části můžou být buď závislé, nebo nezávislé. Tyto elementy sémanticky obohacují náš kód a usnadňují přenos našich dokumentů na různá zařízení.

Představíme si je na fiktivních příbězích z fiktivních novinových webových stránek „HTML5 noviny", které lze vidět na obrázku 3.1.

Obrázek 3.1. HTML5 noviny

Naše stránka začíná záhlavím s hlavním navigačním panelem. Ve starších verzích jazyka HTML bychom tento obsah označili nějak takto:

```
<div id="zahlavi">
  <h1>HTML5 <i>noviny</i></h1>
  <h2>Všechny novinky, které stojí za to sdílet</h2>
  <ul id="nav">
    <li><a href="#">Ze světa</a></li>
    <li><a href="#">Z domova</a></li>
    <li><a href="#">Z velkoměsta</a></li>
    <li><a href="#">Sport</a></li>
    <li><a href="#">Umění &amp; zábava</a></li>
  </ul>
</div>
```

Naše stránka končí zápatím. V jazyce HTML4 bychom ho definovali přibližně následujícím způsobem:

```
<div id="zapati">
  <ul>
    <li><a href="#">Kontaktujte nás</a></li>
    <li><a href="#">Pravidla používání</a></li>
    <li><a href="#">Ochrana soukromí</a></li>
  </ul>
  <p>Bez copyrightu HTML5 noviny, 2013.</p>
</div>
```

Jazyk HTML5 ale nabízí elementy přesně pro tyto účely – elementy `header`, `nav` a `footer`.

Element `header` označuje záhlaví části daného dokumentu. Elementem `footer` označujeme naopak zápatí části dokumentu. Důležité je zde slovo **části** – neříkáme přímo **dokumentu** nebo **stránky**. Některé elementy nazýváme **rozdělující elementy**. Ty rozdělují dokument na více částí (sekcí). Jedním z nich je kupříkladu nový element `section`. Dalšími rozdělujícími elementy jsou elementy `body`, `article`, `aside` a `nav`. A teď přichází to hlavní – každý rozdělující element může mít své vlastní záhlaví a zápatí. Ačkoliv se to může zdát zmatené, plyne z toho především poznatek, že dokument může obsahovat více elementů záhlaví a zápatí.

```
<header>
  <h1>HTML5 <i>noviny</i></h1>
  <h2>Všechny novinky, které stojí za to sdílet</h2>
  <nav>
    <ul>
      <li><a href="#">Ze světa</a></li>
      <li><a href="#">Z domova</a></li>
      <li><a href="#">Z velkoměsta</a></li>
      <li><a href="#">Sport</a></li>
      <li><a href="#">Umění &amp; zábava</a></li>
    </ul>
```

```
  </nav>
</header>
```

Právě jsme zabalili hlavní záhlaví a navigaci stránky do elementu `header`. Kromě toho jsme odstranili identifikátor `nav` a použili místo něho element `nav`. Ještě nám zbývá označit zápatí stránky elementem `footer` jazyka HTML5.

```
<footer>
  <ul>
    <li><a href="#">Kontaktujte nás</a></li>
    <li><a href="#">Pravidla používání</a></li>
    <li><a href="#">Ochrana soukromí</a></li>
  </ul>
  <p>Bez copyrightu HTML5 noviny, 2013.</p>
</footer>
```

V tomto případě jsme jednoduše zaměnili element `div` s identifikátorem `zapati` za element `footer`. Tím jsme dokončili základní kostru našeho dokumentu, takže pojďme přidat svaly.

Element article

Specifikace jazyka HTML5 říká, že element `article`:

> „Reprezentuje kompletní nebo samostatný obsahový prvek v dokumentu, stránce, aplikaci nebo na webových stránkách – takový obsah je možné distribuovat nezávisle nebo používat opětovně (například na více stránkách)."

Časopisecké články a příspěvky blogů jsou jednoznačnými kandidáty na označení elementem `article`. Lze si ho představit také u komentářů blogu. Tímto elementem je možné označit jakýkoliv obsahový prvek, který lze použít na více místech.

Pro samotný článek na naší stránce tudíž nepoužijeme element `div` s identifikátorem `clanek`, ale element `article`:

```
<article>
  <h1>Mladý vůdce kuřat říká, že hrozí pád meteoritů</h1>

  <p class="o-autorovi">
    <b class="reporter">Jan Liška</b>
    <i class="povolani">novinář</i>
  </p>

  <div class="postranni-obsah">
    <h2>O Pavlu Hajném</h2>

    <dl>
```

```
      <dt>Věk</dt>
      <dd>32</dd>

      <dt>Povolání</dt>
      <dd>Generální ředitel Národní organizace kuřat</dd>

      <dt>Vzdělání</dt>
      <dd>Bc., Farmářská univerzita, obor kuřata</dd>
      <dd>Ing., Univerzita v Kvokálkově</dd>
    </dl>

    <p>
      Pavel začal pracovat v Národní organizaci kuřat v roce 2002.
      Postupně se dostával v hierarchii nahoru – od tajemníka přes
      zástupce ředitele až po generálního ředitele, za kterého si jej
      zvolili samotní zaměstnanci společnosti.
    </p>

    <p>
      Národní organizace kuřat je skupina advokátů, kteří se zaměřují na
      zachování zdravého životního prostředí pro kuřata.
    </p>
  </div>

  <p>
    PRAHA -- Pavel Hajný, mladý ředitel Národní organizace kuřat,
    prohlásil, že během několika týdnů dojde k pádu meteoritů. Pavlovi
    kritici ale říkají, že žaludy představují větší hrozbu.
  </p>

  <p>
    Phasellus viverra faucibus arcu ullamcorper sodales. Curabitur
    tincidunt est in imperdiet ultrices. Sed dignissim felis a neque
    dignissim, nec cursus sapien egestas.
  </p>
  ...

  <div id="o-clanku">
    <p class="kontakt-na-reportera">Reportéra Jana Lišku můžete
      kontaktovat na adrese jan.liska@html5noviny.cz.</p>
    <p class="prispevatel">K tomuto článku přispěl novinář Karel
      Krůta.</p>
    <p class="datum-vydani">Vydáno:
      <time datetime="2013-07-11">11. 7. 2013</time></p>
  </div>
</article>
```

Element `article` je příkladem rozdělujícího elementu, jenž může obsahovat záhlaví a zápatí. Pokud se zamyslíme nad předchozím kódem – element `div` s identifikátorem `o-clanku` bychom mohli považovat za zápatí našeho elementu `article`. Co kdybychom tedy zaměnili element `div` za element `footer`?

```
<footer id="o-clanku">
  <p class="kontakt-na-reportera">Reportéra Jana Lišku můžete kontaktovat
    na adrese jan.liska@html5noviny.cz.</p>
  <p class="prispevatel">K tomuto článku přispěl novinář Karel Krůta.</p>
  <p class="datum-vydani">Vydáno:
    <time datetime="2013-07-11">11. 7. 2013</time></p>
</footer>
```

Tentokrát jsme však ponechali atribut `id`. Díky němu budeme moct rozlišit toto zápatí od jiných zápatí na stránce za účelem stylování nebo skriptování.

Element `aside` reprezentuje ekvivalent pro postranní sloupec v novinách nebo časopise. Označuje obsah, který se vztahuje k hlavnímu článku, ale nelze ho použít samostatně. V našem kódu jazyka HTML4 jsme použili element `div` s třídou `postranni-obsah`. Element `aside` nabízí význam a kontext. Změňme proto element `div` na element `aside`.

```
<aside>
  <h2>O Pavlu Hajném</h2>

  <dl>
    <dt>Věk</dt>
    <dd>32</dd>

    <dt>Povolání</dt>
    <dd>Generální ředitel Národní organizace kuřat</dd>

    <dt>Vzdělání</dt>
    <dd>Bc., Farmářská univerzita, obor kuřata</dd>
    <dd>Ing., Univerzita v Kvokálkově</dd>
  </dl>

  <p>
    Pavel začal pracovat v Národní organizaci kuřat v roce 2002. Postupně
    se dostával v hierarchii nahoru – od tajemníka přes zástupce
    ředitele až po generálního ředitele, za kterého si jej zvolili
    samotní zaměstnanci společnosti.
  </p>
  <p>
    Národní organizace kuřat je skupina advokátů, kteří se zaměřují na
    zachování zdravého životního prostředí pro kuřata.
  </p>
</aside>
```

Skládáme kousky dohromady

Náš hotový dokument jazyka HTML5 by měl vypadat takto:

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8">
  <title>HTML5 noviny</title>
</head>
<body>
  <header>
    <h1>HTML5 <i>noviny</i></h1>
    <h2>Všechny novinky, které stojí za to sdílet</h2>
    <nav>
      <ul>
        <li><a href="#">Ze světa</a></li>
        <li><a href="#">Z domova</a></li>
        <li><a href="#">Z velkoměsta</a></li>
        <li><a href="#">Sport</a></li>
        <li><a href="#">Umění &amp; zábava</a></li>
      </ul>
    </nav>
  </header>

  <article>
    <h1>Mladý vůdce kuřat říká, že hrozí pád meteoritů</h1>

    <p class="o-autorovi">
      <b class="reporter">Jan Liška</b>
      <i class="povolani">novinář</i>
    </p>

    <aside>
      <h2>O Pavlu Hajném</h2>

      <dl>
        <dt>Věk</dt>
        <dd>32</dd>

        <dt>Povolání</dt>
        <dd>Generální ředitel Národní organizace kuřat</dd>

        <dt>Vzdělání</dt>
        <dd>Bc., Farmářská univerzita, obor kuřata</dd>
        <dd>Ing., Univerzita v Kvokálkově</dd>
```

```
    </dl>

    <p>
      Pavel začal pracovat v Národní organizaci kuřat v roce 2002.
      Postupně se dostával v hierarchii nahoru – od tajemníka přes
      zástupce ředitele až po generálního ředitele, za kterého si jej
      zvolili samotní zaměstnanci společnosti.
    </p>

    <p>
      Národní organizace kuřat je skupina advokátů, kteří se zaměřují na
      zachování zdravého životního prostředí pro kuřata.
    </p>
  </aside>

  <p>
    PRAHA -- Pavel Hajný, mladý ředitel Národní organizace kuřat,
    prohlásil, že během několika týdnů dojde k pádu meteoritů. Pavlovi
    kritici ale říkají, že žaludy představují větší hrozbu.
  </p>

  <p>
    Phasellus viverra faucibus arcu ullamcorper sodales. Curabitur
    tincidunt est in imperdiet ultrices. Sed dignissim felis a neque
    dignissim, nec cursus sapien egestas.
  </p>
  ...

  <footer id="o-clanku">
    <p class="kontakt-na-reportera">Reportéra Jana Lišku můžete
      kontaktovat na adrese jan.liska@html5noviny.cz.</p>
    <p class="prispevatel">K tomuto článku přispěl novinář Karel
      Krůta.</p>
    <p class="datum-vydani">Vydáno:
      <time datetime="2013-07-11">11. 7. 2013</time></p>
  </footer>
</article>

<footer>
  <ul>
    <li><a href="#">Kontaktujte nás</a></li>
    <li><a href="#">Pravidla používání</a></li>
    <li><a href="#">Ochrana soukromí</a></li>
  </ul>
  <p>Bez copyrightu HTML5 noviny, 2013.</p>
</footer>
```

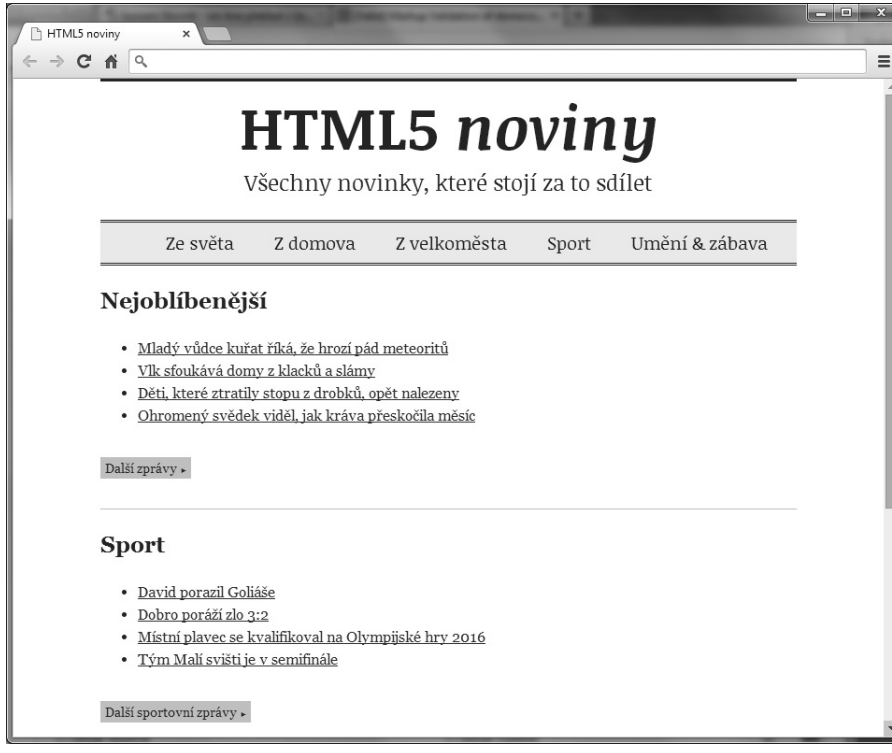
```
</body>
</html>
```

Účelem těchto nových elementů je zavést standardní způsob, jak popisovat strukturu dokumentu. Jazyk HTML je v srdci jazykem pro výměnu dokumentů. Díky novým strukturním elementům lze stejný dokumentu vydat jako webovou stránku nebo jako článek pro elektronické čtečky knih, aniž by bylo nutné procházet změť elementů `div` a atributů `id`.

Element section

Jazyk HTML5 rovněž zavádí element `section`, s nímž lze definovat segmenty dokumentu – záhlaví, zápatí, navigaci, článek nebo postranní obsah. Je specifičtější než element `div`, ale obecnější než element `article`.

Označme domovskou stránku HTML5 novin (viz obrázek 3.2) pomocí elementů `section`.



Obrázek 3.2. Domovská stránka HTML5 novin

Na domovské stránce se nachází tři kategorie článků – Nejoblíbenější, Sport a Podnikání. V jazyce HTML 4 je element `div` jasnou volbou pro označování sekcí, protože se jedná o jedinou volbu. V jazyce HTML5 je však k dispozici popisnější element `section`.

```
<section id="nejoblibenejsi">
  <h1>Nejoblíbenější</h1>
  <ul>
    <li><a href="#">Mladý vůdce kuřat říká, že hrozí pád meteoritů</a></li>
    <li><a href="#">Vlk sfoukává domy z klacků a slámy</a></li>
    <li><a href="#">Děti, které ztratily stopu z drobků, opět
      nalezeny</a></li>
    <li><a href="#">Ohromený svědek viděl, jak kráva přeskočila
      měsíc</a></li>
  </ul>

  <footer>
    <p><a href="#" class="odkaz-dalsi">Další zprávy</a></p>
  </footer>
</section>

<section id="sport">
  <h1>Sport</h1>
  <ul>
    <li><a href="#">David porazil Goliáše</a></li>
    <li><a href="#">Dobro poráží zlo 3:2</a></li>
    <li><a href="#">Místní plavec se kvalifikoval na Olympijské hry
      2016</a></li>
    <li><a href="#">Tým Malí svišti je v semifinále</a></li>
  </ul>

  <footer>
    <p><a href="#" class="odkaz-dalsi">Další sportovní zprávy</a></p>
  </footer>
</section>

<section id="podnikani">
  <h1>Podnikání</h1>
  <ul>
    <li><a href="#">VelkáVěc, a.s., otevírá 3 000 pracovních pozic</a></li>
    <li><a href="#">Kurz amerického dolaru dohání euro</a></li>
    <li><a href="#">ObříAerolinie a LeteckýObr se spojují</a></li>
    <li><a href="#">SvětováNadvláda je tak velká, že nemůže
      padnout</a></li>
  </ul>

  <footer>
    <p><a href="#" class="odkaz-dalsi">Další zprávy o podnikání</a></p>
  </footer>
</section>
```

Element div vs section

Přestože nám rozdělující elementy přinášejí lepší sémantiku, můžou být také zdrojem zmatení. Každý vývojář si často klade otázku: „Můžu zde použít element `div`?“

Odpověď zní: „Ano.“ Koneckonců – element `div` je platným elementem jazyka HTML5. Tento element nemá ale téměř žádnou sémantickou hodnotu. Nedozvíme se nic o tom, jaký typ obsahu v něm najdeme – proto bychom ho měli používat jen tehdy, když nenajdeme vhodnější sémantickou alternativu. Například můžeme potřebovat dodatečný obalující element pro stylování. Nebo můžeme seskupit několik francouzských vět v českém dokumentu. Uzavření těchto vět mezi značky `<div lang="fr">` a `</div>` je rozhodně vhodné. Ve většině ostatních případů je lepší použít element `section`, `header`, `nav`, `footer`, `article` nebo `aside`.

Další elementy dokumentu

Nové rozdělující elementy nejsou jediné nové elementy jazyka HTML5, které souvisejí s dokumentem. V této kapitole si ještě popíšeme následující elementy:

- Elementy `figure` a `figcaption` pro definici obrázků, grafů a diagramů.
- Element `main`, jenž explicitně definuje hlavní část obsahu stránky.

Elementy figure a figcaption

Kdybychom měli v dokumentu doprovodný graf nebo diagram, v jazyce HTML 4 bychom ho mohli označit elementy `div` a `p` jako v níže uvedeném kódu:

```
<div class="graf" id="ilustrace1">
  <img src="graf.jpg" alt="Cena čokolády od roku 2008">
  <p class="popisek">Ilustrace 1: Cena čokolády vzrostla od roku 2008
    o 200%.</p>
</div>
```

To je samozřejmě přijatelné řešení. Co kdybychom ale chtěli zobrazit tento dokument HTML v elektronické čtečce, která se ho pokouší zobrazit v knižním stylu Značka `<div class="graf">` neříká nic o tom, co daný obsah představuje a jak by měl být zobrazen.

V tomto případě bychom měli použít element `figure`. Ten nám umožňuje označovat grafy a jejich popisky tak, aby se staly nezávislé na toku dokumentu. Upravíme proto výše uvedený kód následujícím způsobem:

```
<figure class="graf" id="ilustrace1">
  <img src="graf.jpg" alt="Cena čokolády od roku 2008">
  <figcaption>Ilustrace 1: Cena čokolády vzrostla od roku 2008
    o 200 %.</figcaption>
</figure>
```

Element main

Element `main` je jedním z nejnovějších elementů jazyka HTML5. Není součástí specifikace HTML5 z roku 2011, kterou vydalo konsorcium W3C, ale objevuje se až ve specifikaci HTML 5.1.

Není nijak překvapivé, že prostřednictvím elementu `main` označujeme hlavní část dokumentu nebo aplikace. „Hlavní částí" dokumentu myslíme obsah, který:

- je jedinečný pro daný dokument nebo aplikaci,
- neobsahuje elementy, které jsou součástí více stránek (například záhlaví nebo zápatí).

V době psaní této knihy podporovaly tento element jen prohlížeče Chrome 26+, Firefox 21+ a Opera 15+. Příští verze prohlížečů Internet Explorer a Safari jej měly začít podporovat rovněž.

Kde lze element `main` používat? Některé webové prohlížeče (kupříkladu verze prohlížečů Safari a Firefox pro operační systém Android) nabízejí „čtenářský režim", jenž odstraňuje z dokumentu záhlaví, zápatí a reklamy. Tyto prohlížeče používají v současné době heuristickou analýzu (odborný odhad), aby určily hlavní obsah dokumentu. Elementem `main` jasně říkáme, na jakou část stránky by se měly zaměřit. Webové prohlížeče můžou tento element používat pro vylepšení přístupnosti – například přeskočí navigaci pro nevidomé uživatele.

Prohlédněme si ještě jednou náš novinový článek – tentokrát do něj přidáme element `main`.

```
<!DOCTYPE html>
<html lang="cs">
<head>
  <meta charset="UTF-8">
  <title>HTML5 noviny</title>
  <link rel="stylesheet" href="s.css" media="screen">
</head>
<body>
<div id="obal">
  <header>
    <h1>HTML5 <i>noviny</i></h1>
    <h2>Všechny novinky, které stojí za to sdílet</h2>
    <nav>
      <ul>
        <li><a href="#">Ze světa</a></li>
        <li><a href="#">Z domova</a></li>
        <li><a href="#">Z velkoměsta</a></li>
        <li><a href="#">Sport</a></li>
        <li><a href="#">Umění &amp;amp; zábava</a></li>
      </ul>
    </nav>
  </header>

  <main>
```

```
<article>
  <h1>Mladý vůdce kuřat říká, že hrozí pád meteoritů</h1>

  <p class="o-autorovi">
    <b class="reporter">Jan Liška</b>
    <i class="povolani">novinář</i>
  </p>

  <aside>
    <h2>O Pavlu Hajném</h2>

    <dl>
      <dt>Věk</dt>
      <dd>32</dd>

      <dt>Povolání</dt>
      <dd>Generální ředitel Národní organizace kuřat</dd>

      <dt>Vzdělání</dt>
      <dd>Bc., Farmářská univerzita, obor kuřata</dd>
      <dd>Ing., Univerzita v Kvokálkově</dd>
    </dl>

    <p>
      Pavel začal pracovat v Národní organizaci kuřat v roce 2002.
      Postupně se dostával v hierarchii nahoru – od tajemníka přes
      zástupce ředitele až po generálního ředitele, za kterého si
      jej zvolili samotní zaměstnanci společnosti.
    </p>

    <p>
      Národní organizace kuřat je skupina advokátů, kteří se zaměřují
      na zachování zdravého životního prostředí pro kuřata.
    </p>
  </aside>

  <p>
    PRAHA -- Pavel Hajný, mladý ředitel Národní organizace kuřat,
    prohlásil, že během několika týdnů dojde k pádu meteoritů.
    Pavlovi kritici ale říkají, že žaludy představují větší hrozbu.
  </p>

  <p>
    Phasellus viverra faucibus arcu ullamcorper sodales. Curabitur
    tincidunt est in imperdiet ultrices. Sed dignissim felis a neque
    dignissim, nec cursus sapien egestas.
```

```
      </p>
      ...
      <footer id="o-clanku">
        <p class="kontakt-na-reportera">Reportéra Jana Lišku můžete
          kontaktovat na adrese jan.liska@html5noviny.cz.</p>
        <p class="prispevatel">K tomuto článku přispěl novinář Karel
          Krůta.</p>
        <p class="datum-vydani">Vydáno:
          <time datetime="2013-07-11">11. 7. 2013</time></p>
      </footer>
    </article>
  </main>

  <footer>
    <ul>
      <li><a href="#">Kontaktujte nás</a></li>
      <li><a href="#">Pravidla používání</a></li>
      <li><a href="#">Ochrana soukromí</a></li>
    </ul>
    <p>Bez copyrightu HTML5 noviny, 2013.</p>
  </footer>
</div>
</body>
</html>
```

Nyní máme dokument, jenž je přístupnější a snadněji čitelný pro různé webové prohlížeče, které se ho můžou pokoušet zobrazit.

KAPITOLA 4

Základy: Formuláře

V této kapitole:
- Zakládáme formulář v jazyce HTML5
- Element input
- Sběr jmen
- Povinná formulářová pole
- Nahrávání souborů na server
- Element datalist

Formuláře jazyka HTML5 ušly pořádný kus cesty kupředu oproti starším verzím tohoto jazyka. Máme k dispozici celou řadu nových stavů a typů vstupních polí (například typy `range` a `url`), atributy pro definici povinnosti hodnoty nebo určitého formátu, a navíc můžeme také používat nové rozhraní API, které nám umožňuje omezovat a validovat vstupní data. V neposlední řadě máme k dispozici nová formulářová pole, kterým je kupříkladu element `datalist`, s nímž lze vytvářet ohromující uživatelská rozhraní bez rozsáhlých knihoven jazyka JavaScript a zásuvných modulů.

Bohužel ne každý prohlížeč podporuje všechny tyto funkce. Prozatím musíme stále používat knihovny JavaScriptu a **polyfilly**[29] (kód, který doplňuje chybějící funkčnost webového prohlížeče) jako alternativu.

Nejlépe pochopíme formuláře jazyka HTML5 tak, že si jeden vytvoříme. Pokusíme se vytvořit formulář, jenž bude sbírat nápady pro webové stránky HTML5 novin, které jsme založili v předchozí kapitole. S tímto formulářem bychom měli načítat níže uvedené informace:

- jméno,
- místo pobytu (obec/město),
- e-mailovou adresu,
- telefonní číslo,
- adresu URL, kde se můžeme dozvědět více (pokud existuje),
- samotný nápad na příběh.

Přinejmenším budeme po uživateli požadovat, aby zadal své křestní jméno, e-mailovou adresu a nápad na příběh.

29 *http://www.vzhurudolu.cz/prirucka/polyfill*

Zakládáme formulář v jazyce HTML5

Nový formulář založíme pomocí otevírací značky elementu `form`. Protože formulář chceme odesílat ke zpracování skriptu na straně serveru, musíme zahrnout dva atributy:

- `action` – adresa URL daného skriptu,
- `method` – metoda požadavku protokolu HTTP; někdy je jí metoda GET, ale většinou se jedná o metodu POST.

Jelikož očekáváme delší zprávu, upřednostníme metodu POST nad metodou GET. Metoda GET je vhodnější pro krátké dvojice klíč-hodnota, na které narazíme například u vyhledávacích polí. U formulářů lze používat i další metody protokolu HTTP – metody PUT, HEAD a DELETE, ale obvykle si vystačíme s metodami GET a POST.

Formulářová data mají výchozí typ `application/x-www-form-urlencoded`. Typ přenášeného obsahu můžeme ale také explicitně nastavit atributem `enctype`.

Kupříkladu bychom mohli přidělit atributu `enctype` hodnotu `multipart/form-data`:

```
<form action="/script" method="post" enctype="multipart/form-data">
```

Obě tyto hodnoty se hodí pro odesílání textových dat. Kdybychom však chtěli nahrávat na server soubory, potřebovali bychom právě hodnotu `multipart/form-data`.

Element input

Element `input` je nejčastěji používaným elementem pro tvorbu formulářových prvků. Můžeme u něj specifikovat níže uvedené atributy:

- `name` – název pole,
- `type` – typ zobrazovaného vstupního pole,
- `id` – jedinečný identifikátor pole,
- `value` – slouží k nastavení výchozí hodnoty pole.

Aby z našeho formuláře odcházela nějaká data, vystačili bychom jen s atributem `name`. Z hodnoty každého atributu `name` se stane klíč (název pole) pro hodnotu v našem skriptu na straně serveru. Ve většině případů ale budeme nastavovat rovněž atribut `type`.

Atributu `type` je možné přiřadit celou řadu hodnot – většinu z nich si popíšeme v této kapitole. Jednotlivé hodnoty tohoto atributu odpovídají různým druhům prvků uživatelského rozhraní a nesou s sebou jiné sady validačních omezení. Nejobecnější hodnotou (a současně výchozí hodnotou) atributu type u elementu `input` je hodnota `text`.

Sběr jmen

Jména lidí a názvy míst jsou obvykle kombinace alfanumerických znaků, mezer a interpunkce. Z tohoto důvodu budeme používat element `input` typu `text`. Přidejme formulářová pole pro jméno a město pobytu pisatele dopisu. Jelikož poskytnutí jména vyžadujeme, doplníme také atribut `required`.

```
<p>
  <label for="jmeno">Vaše jméno:</label>
  <input type="text" name="jmeno" id="jmeno" required>
</p>

<p>
  <label for="mesto">Místo pobytu:</label>
  <input type="text" name="mesto" id="mesto">
</p>
```

Poznámka: Atributy id a name

Hodnota atributu **`id`** nemusí být stejná jako hodnota atributu **`name`**.

Popisky polí

Před chvíli jsme použili dosud neznámý element `label`. Tento element ve formuláři jazyka HTML funguje podobně jako popisek formulářového pole na papíře. Informuje uživatele o tom, co by měl zadat do daného pole. Abychom mohli přidělit popisek formulářovému poli, měli bychom definovat atribut `for` se stejnou hodnotou, jakou má atribut `id` tohoto pole. Případně můžeme umístit dané pole přímo do elementu `label`.

```
<label>Vaše jméno:
  <input type="text" name="jmeno" id="jmeno" required>
</label>
```

Atributy `for` a `id` nám však nabízejí více flexibility při rozvrhování stránky.

Proč nenapíšeme jen samotný text? Proč ho vkládáme do elementu `label`? Elementy `label` zlepšují přístupnost webových stránek pro uživatele se zrakovými obtížemi a nevidomé uživatele. Software pro přečítání textu využívá popisky, aby těmto uživatelům pomohl vyplnit formulář. Jedná se tedy o funkci, která podporuje přístupnost a je přímou součástí jazyka HTML.

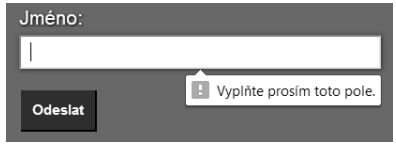
Povinná formulářová pole

Nativní validace formulářů je skvělým vylepšením jazyka HTML5 oproti jeho starším verzím. Pokud přidáme k poli atribut `required`, žádáme webový prohlížeč, aby před odesláním formuláře na server ověřoval, jestli uživatel vyplnil toto pole.

Poznámka: Prázdné atributy

Atribut `required` je příkladem prázdného atributu. Jeho přítomnost nebo absence rozhoduje o tom, zda je pole povinné, nebo není.

Jestliže je pole `jmeno` prázdné v době odesílání formuláře, prohlížeče Chrome, Firefox, Opera Internet Explorer 10+ zabrání jeho odeslání a zobrazí uživateli výstražnou zprávu (viz obrázek 4.1). Přitom nemusíme vůbec přistupovat k modelu DOM.



Obrázek 4.1. Odeslání formuláře způsobilo vznik chybové zprávy v prohlížeči Chrome

Povšimněte si, že mezi vyjmenovanými prohlížeči není prohlížeč Safari. Safari ve verzi 6.0.5 (a starší) neposkytuje žádnou viditelnou zpětnou odezvu uživateli. Podporuje validaci jazyka HTML5, ale potřebujete také skriptování modelu DOM a jazyk CSS, abyste mohli uživatele upozornit na chybu. Jak na to, se dozvíte v části věnované validačnímu rozhraní API.

Měníme vzhled povinných polí

Povinná a volitelná pole můžeme od sebe vzhledově odlišit pomocí kaskádových stylů. Povinná pole můžeme vybírat prostřednictvím dvou selektorů jazyka CSS:

1. selektoru atributu `[required]`,
2. pseudotřídy `:required`.

Pseudotřída `:required` patří do čtvrté úrovně[30] selektorů jazyka CSS, ale k dispozici je v nejnovějších verzích všech moderních webových prohlížečů. Lze používat také pseudotřídu `:optional`, která vyhledává volitelná vstupní pole.

V případě, že bychom museli zachovat funkčnost i ve starších webových prohlížečích, použili bychom selektor atributu. Kdybychom kupříkladu chtěli vykreslit červený rámeček široký 1 pixel okolo povinných polí, přidali bychom do naší šablony stylů níže uvedené pravidlo:

```
input[required], input:required {
  border: 1px solid #c00;
}
```

E-mailové adresy, telefonní čísla a adresy URL

Své přispěvatele budeme chtít informovat, že jsme přijali jejich námět. Proto doplníme do našeho formuláře pole pro e-mailovou adresu a telefonní číslo. Navíc se pokusíme od uživatele získat adresu URL, na níž bychom se mohli dozvědět více informací, takže přidáme ještě pole pro adresu URL.

30 *http://www.w3.org/TR/selectors4/*