Webový slovník anglických pojmů z oblasti automatizace

Web dictionary of English terms from automatization area

Michal Brázdil

Bakalářská práce
2007

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav automatizace a řídicí techniky
akademický rok: 2006/2007

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Michal BRÁZDIL**
Studijní program:	**B 3902 Inženýrská informatika**
Studijní obor:	**Automatické řízení a informatika**
Téma práce:	**Webový slovník anglických pojmů z oblasti automatizace**

Zásady pro vypracování:

1. Zpracujte literární rešerši dané problematiky.
2. Nastudujte a popište základní principy tvorby webových stránek s využitím technologií HTML, CSS, PHP, SQL, atd.
3. Zvolte vhodný základ slovní zásoby z oblasti automatizace.
4. Vytvořte webový česko-anglický/anglicko-český slovník se stručným popisem vybraných výrazů a s možností administrace a prezentujte jej na Internetu.

Rozsah práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

[1] e-Automatizace: Anglicko-český slovník -- Informační portál z oblasti automatiza [online]. [cit. 1. února 2007]. Dostupné z WWW: http://www.e-automatizace.cz/slovnik.asp.
[2] IFAC keyword list of control terminology. [online]. [cit. 1. února 2007]. Dostupné z WWW: http://www1.elsevier.com/homepage/saf/ifac/site/IPV%20keywords.htm.
[3] Kosek, J.: Vše o WWW [online]. [cit. 1. února. 2007]. Dostupné na WWW: http://www.kosek.cz/.
[4] Prokop, R., Matušů, R., Prokopová, Z.: Teorie automatického řízení -- lineární spo dynamické systémy. Skriptum FAI UTB ve Zlíně, 2006.

Vedoucí bakalářské práce: **Ing. Radek Matušů**
Ústav automatizace a řídicí techniky

Datum zadání bakalářské práce: **13. února 2007**

Termín odevzdání bakalářské práce: **25. května 2007**

Ve Zlíně dne 13. února 2007

L.S.

prof. Ing. Vladimír Vašek, CSc.
děkan

prof. Ing. Vladimír Vašek, C
ředitel ústavu

ABSTRAKT

Cílem bakalářské práce je vytvoření anglicko-českého, česko-anglického internetového slovníku pojmů z oblasti automatizace. Ve stručnosti popisuje technologie, kterými byl internetový slovník vytvořen (XHTML, CSS, PHP a MySQL). V další části je stručný popis instalace a zprovoznění internetového slovníku na serveru, popis veřejné a administrační části s názornými ukázkami nastavení a funkcemi internetového slovníku.

Klíčová slova:

automatizace, www, web, slovník, xhtml, css, php, mysql

ABSTRACT

The main aim of the Bachelor work is creation of english-czech, czech-english internet dictionary from automation area. It briefly describes technologies used for internet dictionary development (XHTML, CSS, PHP and MySQL). The next part contains concise description of installation and launching internet dictionary on server, description of public and administration part with illustration of setting and functions of internet dictionary.

Keywords:

automatization, www, web, dictionary, xhtml, css, php, mysql

Tímto bych chtěl poděkovat vedoucímu mé bakalářské práce panu Ing. Radku Matušů za odborné vedení, cenné rady a podmětné připomínky udílené během vypracovávání této práce.

Prohlašuji, že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků, je-li to uvolněno na základě licenční smlouvy, budu uveden jako spoluautor.

Ve Zlíně …….

Podpis diplomanta

OBSAH

ÚVOD

Vše začalo na počátku 60. let, kdy se začali objevovat první myšlenky, převážně v USA, na vytvoření sítě, která by navzájem propojovala nejdůležitější vojenské, vládní a vědecko-výzkumné počítače.

První testovací síť byla instalována v roce 1968 v Národní výzkumné laboratoři ve Velké Británii. Ukázala se být velmi úspěšnou a proto rostl zájem o její připojení. Hlavním využitím této sítě byla elektronická pošta a bylo zajímavé, že kromě konferencí sloužících k výměně vědeckých informací se objevily i konference určené pro zábavu.

Vznik dnešního Internetu je datován od roku 1993, kdy Švýcar Tim Berners Lee vymyslel pro atomové fyziky ve švýcarském Bernu nový způsob výměny informací. Tento nový způsob, známý pod zkratkou WWW (World Wide Web, což se dá přeložit jako "světová pavučina"), byl startem závodu za zábavou a informacemi na Internetu. Obrovskou výhodou Internetu je, že při prohlížení WWW stránek se nemusíte starat o to, zda informace jsou na počítači v České republice, Kanadě, nebo třeba v Austrálii.

A jelikož se Internet těší neustále větší oblibě, jak v počtu uživatelů, kteří ho využívají, tak v nepřeberném moři informací, které se nacházejí na milionech serverech a stránkách je téměř zbytečné hledat informace jinde. Možná se najdou tací, kteří by nesouhlasili, ale hledat v knihách, které nejsou aktuální je zbytečné. Na internetu se dá najít mnohem více informací, které se aktualizují i několikrát za hodinu a to vše ještě k tomu zadarmo.

Proto jsem při tvorbě této bakalářské práce využil své zkušenosti z předchozích let na vytvoření internetového slovníku, který by se měl stát dalším zdrojem informací.

V teoretické části této práce jsou popsány technologie, kterými byl internetový slovník naprogramován. V praktické části je popsán postup instalace slovníku, popis veřejné části a administrace.

I. TEORETICKÁ ČÁST

1 HTML

1.1 Historie

První definici HTML vytvořil Tim Berners-Lee v roce 1991. Tato verze umožňovala vkládat do textu obrázky, hypertextové odkazy, vytvořit několik logických úrovní a několik druhů zvýraznění. Byla označena jako *HTML 0.9*. Požadavky uživatelů se zvyšovaly a producenti prohlížečů začali vytvářet HTML nové prvky. Tim Berners-Lee všechny používané prvky shrnul do standardu *HTML 2.0* (specifikace v RFC 1866), který již plně vyhovuje standardu SGML (ISO 8879, r. 1986). V roce 1996 vzniklo *HTML 3.0*. V roce 1997 *HTML 4.0*, o dva roky později opravná verze *HTML 4.01*, o rok později, tedy v roce 2000 vzniklo XHTML 1.0. Nyní již existuje specifikace XHTML 1.1 a ve vývojovém stádiu se nachází specifikace XHTML 2.0.

1.2 Co to je HTML?

HTML, jak už název napovídá, patří mezi značkovací jazyky, pomocí kterých definujete strukturu dokumentů - v tomto případě webových stránek. To, že jde o značkovací jazyk znamená, že pomocí tohoto jazyka nemůžeme dělat žádné výpočty ani grafická kouzla. HTML je tu z úplně jiného důvodu, a to kvůli tvorbě jasně a přesně strukturovaných dokumentů. Možná, že vyznačit v dokumentu jeho strukturu není nic obtížného, ale bohužel se na internetu stále objevují dokumenty, které nevyhovují standardům. Někdy to jsou jen drobné chyby, které nevyhovují přísnějším standardům (např. chybějící popisek obrázku, kódování stránky nebo samotná definice html dokumentu), jindy to jsou ale chyby, které brání správnému zobrazení stránek (např. chybně ukončená tabulka, chybějící párový tag, atd.).

1.3 Co musí HTML dokument obsahovat?

Každý dokument musí obsahovat několik základních značek (tagů), které dohromady tvoří kostru stránky vyhovující standardům. Validita stránek by měla být naším hlavním cílem, toho ale nedocílíme bez správné kostry dokumentu.

Jako úplně první informace by se ve zdrojovém kódu naší stránky, měla objevit informace o tom, jakou verzi HTML na svých stránkách používáme - neboli *specifikace typu*

dokumentu (DTD). Tato specifikace byla - a bohužel stále je - dost často tvůrci stránek opomíjena, ale přitom jde o zásadní část každého dokumentu. Podle tohoto prvního řádku se určí, v jakém režimu má pracovat validátor, kterým si budete kontrolovat syntaktickou bezchybnost svých stránek, i prohlížeče návštěvníků stránek. Většina současných prohlížečů pracuje ve dvou módech: standardním (stránka se zobrazuje podle specifikovaného typu dokumentu, aktivuje se při určení typu dokumentu i s URL adresou definice) a nestandardním (stránka se zobrazuje přibližně jako v Netscape 4, aktivuje se pokud není definován typ dokumentu nebo je definována verze starší než HTML 4). K čemu by byla stránka napsaná podle standardu, která by se nezobrazovala tak, jak má?

V této bakalářské práci byla použita asi nejběžnější používaná verze DTD - XHTML 1.0, která má tři specifikace:

XHTML 1.0 Strict

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

XHTML 1.0 Transitional

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

XHTML 1.0 Frameset

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

Typy dokumentů *Strict* a *Transitional* se liší jen tolerancí k chybám webmastera a vyžadováním CSS. Stránky psané v *XHTML 1.0 Transitional* jsou mezistupněm mezi starým *HTML 4.01* a *XHTML 1.0 Strict*. Ve verzi *Strict* byste si měli zvyknout na veškeré formátování pomocí CSS, pokud si osvojíte tento způsob práce bude pro vás později přechod na XHTML hlavně o změně první řádky určující typ dokumentu.

Typ dokumentu *Frameset* má tytéž vlastnosti, jako typ *Transitional*, ale je navíc obohacen o podporu rámů (slouží k definici rozložení rámů na stránce).

Po určení typu dokumentu můžeme už přejít k dalším tagům. V XHTML se všechny tagy (značky) zapisují do špičatých závorek <značka>. Značky se dělí na dvě hlavní skupiny

- párové
- nepárové

Párové značky musí mít svůj začátek, ve kterém jsou vypsány i atributy doplňující vlastnosti prvku, nějaký svůj obsah a samozřejmě konec - typickým příkladem je třeba odstavec . Konec platnosti určitého tagu se označuje pomocí stejného tagu, jako byl začáteční tag, jediný rozdíl je ten, že za otevírací závorku „<“ přidáte lomítko „/“. Nepárové značky naproti tomu nemají žádný obsah, který by ovlivňovaly - nejčastější nepárovou značkou je asi znak nové řádky
. V XHTML se nepárové tagy musí ukončit stejně jako párové tagy a to dvěma způsoby. Prvním způsobem je, že nepárový tag zapíšeme jako párový např.
 zapíšeme
</br>. Druhý způsob je kratší a asi nejpoužívanější a zapisuje se takto např.
 zapíšeme
.

Mezi nepárové tagy patří ještě např. <input />, <img />, <hr /> a další.

Každá stránka musí být uzavřena do párového tagu <html></html>, který "ohraničuje" celou stránku.

Dalším tagem, na nějž nesmíme zapomenout je opět párový tag označující hlavičku dokumentu, je to tag <head></head>. Sem se zapisují informace o dokumentu (použité kódování atd.), informace pro vyhledávače (klíčová slova atd.) a titulek dokumentu.

Kódovat stránky, nám zabezpečí správné zobrazení textu v HTML dokumentu, který se zobrazuje v internetovém prohlížeči. Nejběžnější kódování html dokumentů je v iso-8859-2, proto napíšeme mezi tagy <head></head> tento řádek:

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-2">
```

Druhým nejčasteji používáným kódováním je kódování pro windows, windows-1250.

Další důležitou značkou je <title></title>, do které se uzavírá nadpis stránky, který se má zobrazit v titulku okna prohlížeče nebo záložky. Pokud tuto značku nepoužijeme, objeví se v titulku adresa URL.

Tímto jsme skončili s nejdůležitějšími prvky v hlavičce dokumentu a můžeme ji proto uzavřít tagem </head> a posunout se v tvorbě dokumentu, tedy k samotnému tělu stránky. Tělo stránky je ta část, kterou vidíme ve svých prohlížečích jako tabulky, odstavce, seznamy, články atd. Tělo stránky se označuje zcela prozaicky tagem <body></body>, který následuje hned za tagem označujícím konec hlavičky </head>.

Pokud mezi tagy <body></body> vložíme nějaký text, tak se nám, kromě HTML tagů, zobrazí v prohlížeči. Pokud byste chtěli vytvořit stránku, kde se bude text dělit na několik odstavců využijeme k tomu tag .

Základ tedy už známe a můžeme si zkusit vytvořit svůj první dokument v jazyku XHTML. Vytvořený soubor ukládáme s příponou .html. Jako inspirace nám může posloužit třeba tento výpis:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
2">
  <title>Moje první stránka</title>
 </head>
 <body>
  <p>Ahoj světe!</p>
  <p>Já jsem webová stránka, kterou sis právě napsal.
  Moc toho neumím, ale přesto bys mě mohl mít rád - jsem přece TVOJE.
:)</p>
 <p>Tak už se loučím</p>
 </body>
</html>
```

1.4 Na co si tedy dát pozor?

V XHTML zmizely značky a atributy, které se starají jen o vzhled určitých prvků, např.: bgcolor="". Tyto atributy a značky byly v HTML 4.01 označeny jako zastaralé a v budoucnu nepodporované - tím budoucnem je právě XHTML.

Značky se nesmí křížit

Toto pravidlo platilo už v HTML, ale pokud jsme ho porušili, prohlížeč se s tím dokázal nějak srovnat. Nyní je to považováno za chybu.

Příklad špatného kódu:

```
<p>Nějaký text - část je <b>tučně</p></b>
```

A takto je to správně:

```
<p>Nějaký text - část je <b>tučně</b></p>
```

Značky a atributy malými písmeny

Značky a atributy se musí v XHTML psát malými písmeny. Ačkoli to není pro HTML 4.01 povinné, myslím, že je to dobrý zvyk, který se určitě vyplatí.

Všechny párové značky musí být ukončené

Tohle pravidlo je jednoduché a jasné. Ukončovací tag je povinný u všech párových elementů.

Příklad špatného kódu:

```
<p>první odstavec<p>druhý odstavec
```

A takto je to správně:

```
<p>první odstavec</p><p>druhý odstavec</p>
```

Hodnoty všech atributů musí být v uvozovkách

```
<div class=“text“ style=“border: 1px solid #000000“>
```

Atribut musí mít hodnotu

Pokud chceme použít atribut, který byl v HTML 4.01 bez hodnoty, musíte ji v XHTML uvést.

```
<option value="01" selected="selected">
```

Nepárové značky musí končit znaky „/>“

Bez dalšího zbytečného vysvětlování uvedu příklad:

, <input />, <img />, <hr /> …

2 CSS – KASKÁDOVÉ STYLY

2.1 Historie

CSS (Cascading Style Sheets) neboli kaskádové styly vznikly jako souhrn metod pro úpravu vzhledu stránek. První návrh normy byl zveřejněn v roce 1994, v roce 1997 byla pak vydána specifikace CSS 1, v roce 1998 specifikace CSS 2, nyní se pracuje na verzi specifikace CSS 3.

2.2 Co je to CSS?

HTML je značkovací jazyk, z toho vyplývá, že jednotlivé značky (tagy) mají vyznačovat význam jednotlivých částí textu. Přesto se v HTML v průběhu let objevilo několik atributů a elementů, které ovlivňují pouze grafický vzhled, a některé z původních tagů určených pro logické členění textu, je často účelově využíváno k dosažení určitých grafických efektů. Aby se předešlo těmto nešvarům, a bylo možné jednoduše oddělit strukturální formátování textu od grafického formátování, definovalo v roce 1997 koncorsium W3C soubor metod pro grafickou úpravu webových stránek. Tento soubor metod se jmenuje Cascading Style Sheets (CSS), česky "kaskádové styly".

Kaskádové styly umožňují definovat způsob zobrazení (velikost a druh písma, barvu, zarovnání, orámování, pozadí apod.) u každého html elementu. Tento předpis vzhledu není přímo součástí textu stránky, a díky tomu je zápis stránky přehlednější a dobře strukturovaný. Navíc styly umožňují definovat jednotný vzhled konkrétního elementu v celém dokumentu jedním zápisem tzn. nemusí se opakovat u každého elementu. Pokud styl uložíme do externího souboru, může ho využívat více stránek najednou. Definice vzhledu všech stránek je uložena na jednom místě. Při požadavku na změnu vzhledu stránek, stačí upravit styl na jednom místě, a změny se automaticky promítnou do všech stránek.

Jak je běžné ve světě webdesignu, i podpora kaskádových stylů v jednotlivých prohlížečích webových stránek je různá. Jednotlivé prohlížeče mají vlastní výklad či pravidla CSS. V současné době existují dvě verze kaskádových stylů – CSS 1 (http://www.w3.org/TR/REC-CSS1) a CSS 2 (http://www.w3.org/TR/REC-CSS2). Při vytváření této bakalářské práce bylo použito specifikace CSS 2.

2.3 Základní principy CSS

2.3.1 Vazba na (X)HTML dokument

Aby mohl být dokument zformátován prostřednictvím kaskádových stylů, musí být stylový předpis k dokumentu nějak připojen. V případě (X)HTML dokumentů k tomu lze použít několika různých prostředků s různými výhodami a nevýhodami a různou mírou podpory v klientech. Právě různá míra podpory navíc umožňuje „filtrovat“ stylové předpisy pro různé prohlížeče.

2.3.2 Základní metody připojení stylů k (X)HTML

Existují čtyři základní metody, jak připojit stylový předpis (stylové předpisy) k (X)HTML dokumentu:

- prvek <*link*>
- prvek <*style*>
- atribut *style* různých HTML prvků
- klientské skriptování (JavaScript) a DOM (Document Object Model)

Prvek <*link*>

Prvek <*link*> se používá pro připojení stylového předpisu uloženého v externím souboru. Jedná se o preferovaný způsob pro normální použití. Mezi hlavní výhody toho způsobu patří:

- pro celé webové místo nebo jeho sekci stačí jediný soubor se stylovým předpisem, což výrazně zjednodušuje jeho správu a usnadňuje dosažení jednotné grafické úpravy celého webového místa
- po prvním načtení spolu s prvním dokumentem daného webového místa, se stylový předpis cachuje na straně klienta a načítaní dalších dokumentů z téhož místa se tím podstatně urychlí

- vyčleněním stylového předpisu do externího souboru je důsledně naplněna idea oddělení formátování od struktury dokumentu, což má pozitivní význam pro přenositelnost a správu webového obsahu.

Na druhou stranu se však umístění stylového přepisu do samotného souboru vně dokumentu může ve specifických případech jevit jako nevýhodné. Je to např. tehdy, chceme-li dokument zachovat jako nedělitelný celek (např. pro odeslání elektronickou poštou), nebo když se celý dokument včetně některých částí stylového předpisu generuje dynamicky na straně serveru.

Samotný prvek se vkládá výhradně do hlavičky (<*head*>) a má tyto atributy:

- **href** – určuje cestu k externímu souboru se stylovým předpisem
- **type** – určuje typ obsahu (*content-type*) odkazovaného souboru a v tomto případě musí mít vždy hodnotu *text/css*
- **rel** – určuje typ vazby dokumentu k externímu zdroji. V tomto případě musí mít hodnotu *stylesheet* pro základní stylové předpisy, případně *alternative stylesheet* pro alternativní stylové předpisy.
- **media** – určuje médium (média), pro které je stylový předpis určen. Tento atribut není povinný a jeho hodnotou je řetězec obsahující jeden či více deskriptorů média
- **title** – přiřazuje textový titulek danému stylovému předpisu. Praktický význam má jen u alternativních stylových předpisů.

Zde jsou typické příklady použití prvku <*link*> pro připojení externího stylového předpisu:

```
<link rel=“stylesheet“ type=“text/css“ href=“style1.css“ />
<link rel=“stylesheet“ type=“text/css“ href=“style2.css“ media=“all“ />
<link rel=“stylesheet“ type=“text/css“ href=“style3.css“ media=“screen“ />
<link rel=“stylesheet“ type=“text/css“ href=“style4.css“ media=“screen, print“ />
<link rel=“stylesheet“ type=“text/css“ href=“style5.css“ media=“print“ />
<link rel=“stylesheet“ type=“text/css“ href=“style6.css“ title=“Styls vysokým kontrastem“ />
```

Nad prvním řádku se k dokumentu připojí stylový předpis uložený v souboru *style1.css*. Protože není specifikováno médium, použije se pro libovolné médium. Na druhé řádku se připojí stylový předpis *style2.css* určený explicitně pro média všech typů. Naproti tomu se stylový předpis *style3.css*, třetí řádek, připojí pouze pro médium typu *screen*, předpis *style4.css*, čtvrtý řádek, pro média *screen* a *print* a předpis *style5.css*, pátý řádek, pouze pro médium *print*.

V hlavičce (X)HTML dokumentu se může nacházet víc prvků <link> a k dokumentu tak může být tímto způsobem připojeno více stylových předpisů. V takovém případě klient použije postupně všechny stylové předpisy adekvátní danému médiu. Výjimku tvoří alternativní stylové předpisy označené hodnotou *alternative stylesheet* v atributu *rel*. Ty by měl klient uživateli vhodným způsobem nabídnout (např. formou menu, jehož jednotlivé položky jsou tvořeny obsahy atributu *title* prvku <*link*> pro všechny alternativní stylové předpisy) a použít je jen tehdy, pokud si je uživatel z nabídky zvolí. Alternativní stylové předpisy však zatím podporuje jen omezený počet minoritních prohlížečů (Mozilla, NN6+).

V souvislosti s ukládáním stylových předpisů do externího souboru je vhodné zmínit, že pokud je ve stylovém předpisu uvedeno relativní URL (např. cesta k obrázku na pozadí v hodnotě vlastnosti *background-image*), bude toto relativní URL vyhodnoceno vzhledem k umístění souboru se stylovým předpisem, nikoli vzhledem k umístění samotného dokumentu.

Prvek <style>

Stylový předpis může být vložen přímo do (X)HTML dokumentu prostřednictvím prvku <*style*>. Stejně jako <*link*> se vkládá do hlavičky (<*head*>) dokumentu a má tyto atributy:

- type – určuje typ obsahu (content-type) odkazovaného souboru v tomto případě musí mít vždy hodnotou text/css
- media – určuje médium (média), pro které je stylový předpis určen. Tento atribut není povinný a jeho hodnotou je řetězec obsahující jeden či více deskriptorů média (viz část „Typy médií“).

- title – přiřazuje textový titulek danému stylovému předpisu. Praktický význam má jen u alternativních stylových předpisů

Stylový předpis je pak obsahem prvku <*style*>, jak ukazuje následující příklad:

```
<style type="text/css">
   body {
      background: #000000;
      color: #eeeeee;
      font-family: Arial, Helwetica, sans-serif;
      font-size: 12px;
   }
   h1 {
      font-size: 20px;
      margin-top: 0px;
   }
   p {
      line-height: 14px;
      margin: 5px 0px 5px 0px;
   }
</style>
<style type="text/css" media="print">
   body {
      color: #000000;
      background: #ffffff;
      font-family: "Times New Roman", Times, serif;
      font-size: 10px;
   }
</style>
```

Tento způsob připojení stylového předpisu je již méně praktický a v novějších verzích XHTML (verze 2) možná již nebude možný. Neodděluje styly od dokumentu tak zřetelně, jako připojení externího stylového předpisu a z toho plynou hlavní nevýhody:

- stylový předpis zvětšuje celkový objem dokumentu a necachuje se, takže se načítá s každou novou stránkou znovu
- dokumenty se obtížněji udržují, neboť jedna změna stylu musí provést obvykle do více dokumentů téhož webového místa. Hrozí, že nedopatřením dojde k nežádoucí nekonzistenci stylu.

Tyto nevýhody je do určité míry možné kompenzovat pomocí pravidla *@import*, které umožňuje načtení stylového předpisu z externích souboru i prostřednictvím prvky <*style*>.

```
<style type="text/css">
   @import "styl.css";
</style>
```

Tento způsob se využívá poměrně často k ukrývání stylových předpisů před staršími prohlížeči, které pravidlo *@import* buď vůbec, nebo jednu z jeho možných syntaxí nepodporují.

Prvek <*style*> se ovšem hodí tehdy, když z nějakého důvodu chceme mít styly definované přímo v dokumentu, např. proto, že jsou pro daný dokument jedinečné (pak se většinou kombinuje <*link*> a <*style*>), nebo proto, že chcete usnadnit studium či testování zdrojového kódu dokumentu.

Atribut *style*

Styly lze v (X)HTML přiřadit i jednotlivým prvkům prostřednictvím atributu *style*. Tato metoda se nazývá vřazené stylové předpisy (inline style sheets) a nejlépe ji ilustruje několik příkladů:

```
<body>
   <h1 style="font-size: 150%; color: #ff0000">Nadpis dokumentu</h1>
   <p style="font-style: italic">
      Nadpis nad tímto odstavcem bude červeným písmem o velikosti 150%.
      Text tohoto odstavce bude vykreslen italikou (kurzívou).
   </p>
   <p style="border-bottom: 1px solid #00ff00">
```

```
      Tento odstavec bude mít pod sebou vykresleno orámování plnou
zelenou
      čárou.
   </p>
</body>
```

Vřazovat stylové předpisy přímo do kódu (X)HTML dokumentu je obecně nevhodné a v novějších verzích XHTML (verze 1.1) dokonce nepřípustné. Jedinou akceptovatelnou výhodou je snadné testování vzhledu jednotlivých prvků během tvorby dokumentu s tím, že po otestování se definice stylu přenese do externího stylového předpisu.

Klientské skriptování a DOM

Styly lze jednotlivým prvkům HTML a XML dokumentů přiřazovat též dynamicky, pomocí klientského skriptování a DOM (Document Object Model, objektový model dokumentu). Jedná se o velmi mocný způsob tvorby webových aplikací či nástroj přizpůsobení vzhledu dokumentu aktuálním podmínkám klienta (rozlišení obrazovky, resp. velikost okna prohlížeče, nastavení uživatelských preferencí atd.)

3 PHP

3.1 Co je to PHP?

PHP je v současnosti velmi rozšířená technologie umožňující snadné programování na straně serveru (server-side programming). Toho lze využít k tvorbě různých interaktivních webových stránek. Stručně lze říci, že skript napsaný v PHP je proveden na serveru podle zadaných kritérií a výsledek je odeslán volajícímu počítači stejným způsobem, jakým se odesílají běžné statické (XHTML) stránky. Jakmile je však stránka načtena klientem, pomocí PHP ji již není možné dále měnit.

PHP je programovací jazyk umožňující procedurální i objektově orientované programování. Znalost objektově orientovaného programování (OOP) tedy může být při práci v PHP výhodou, není však nutnou podmínkou. PHP také patří mezi jazyky, kde například není nutné předem definovat typ proměnných, navíc jakákoli proměnná může kdykoli změnit svůj typ. Jednoduše řečeno, co se týče psaní kódu, z PHP při psaní skriptů "sálá" určitá volnost a neomezenost. Na druhé straně záleží plně na programátorovi, jaký si bude v kódu udržovat pořádek.

3.2 Historie a současnost PHP

PHP urazilo v posledních několika málo letech dlouhou cestu. Růst v jeden z nejprominentnějších jazyků ovládajících web nebyl snadný.

3.2.1 PHP/FI

PHP je nástupcem staršího produktu, nazvaného PHP/FI. PHP/FI vytvořil Rasmus Lerdorf v roce 1995, na počátku jako jednoduchou sadu skriptů v jazyce Perl pro zpracování záznamů o přístupech k jeho webu. Tuto sadu nazval Personal Home Page Tools. Protože byla potřeba větší funkčnost, napsal Rasmus mnohem rozsáhlejší implementaci v C, která byla schopna komunikovat s databázemi a umožňovala uživatelům vyvíjet jednoduché dynamické aplikace pro web. Rasmus se rozhodl uvolnit zdrojový kód PHP/FI pro všechny, takže kdokoli ho může používat, stejně jako opravovat chyby a vylepšovat kód.

PHP/FI, což znamená Personal Home Page/Forms Interpreter, obsahovalo něco ze základní funkcionality PHP, jak ho známe dnes. Mělo proměnné perlovského typu, automatickou

interpretaci formulářových proměnných a syntaxi vloženou do HTML. Syntaxe samotná byla podobná jazyku Perl, přestože mnohem omezenější, jednodušší a v něčem nekonzistentní.

V roce 1997 se PHP/FI 2.0, druhá implementace psaná v C, stala kultovní záležitostí pro (odhadem) tisíce uživatelů po celém světě a přibližně 50 000 domén oznamujících nainstalované PHP/FI, což čítalo zhruba 1% všech domén na Internetu. I když do projektu začalo svými kusy kódu přispívat více lidí, stále to byl velký projekt jednoho muže.

PHP/FI 2.0 bylo oficiálně uvolněno až v listopadu 1997, poté, co strávilo většinu svého života v betaverzích. Krátce nato bylo následováno první alfaverzí PHP 3.0.

3.2.2 PHP 3

PHP 3.0 byla první verze, která se velmi blížila takovému PHP, jak ho známe dnes. Vytvořili ho Andi Gutmans a Zeev Suraski v roce 1997 jako kompletně přepsaný celek poté, co shledali PHP/FI 2.0 výrazně "poddimenzovaným" pro vývoj svých aplikací pro e-komerci. Ve snaze spolupracovat a zahájit budování nad existující uživatelskou základnou PHP/FI, rozhodli se Andi, Rasmus a Zeev pracovat společně a prohlásit PHP 3.0 za oficiálního nástupce PHP/FI 2.0 a vývoj PHP/FI 2.0 byl v podstatě zastaven.

Jednou z nejsilnějších zbraní PHP 3.0 byly jeho obrovské možnosti rozšíření. Kromě pevné infrastruktury pro mnoho různých databází, protokolů a API koncovým uživatelům, přilákaly především možnosti rozšíření PHP 3.0 také tucty vývojářů, kteří se připojili a vytvořili nové rozšiřující moduly. Toto byl nesporně klíč k obrovskému úspěchu PHP 3.0. Jiným klíčovým prvkem v PHP 3.0 byla podpora objektově orientované syntaxe a mnohem silnější a konzistentnější syntaxe jazyka.

Nový jazyk byl uvolněn pod novým názvem, který odstranil implikaci omezeného osobního použití, kterou neslo označení PHP/FI 2.0. Byl nazván pouze PHP, což je rekurzívní akronym - PHP: Hypertext Preprocessor.

Na konci roku 1998 vyrostlo PHP do rozsahu instalací v řádu (odhadem) desítek tisíc uživatelů a stovek tisíc webů. V době svého vrcholu bylo PHP 3.0 instalováno na přibližně 10 % všech WWW serverů na Internetu.

PHP 3.0 bylo oficiálně uvolněno v červnu 1998 poté, co strávilo cca 9 měsíců ve veřejném testování.

3.2.3 PHP 4

V zimě 1998, krátce po oficiálním uvolnění PHP 3.0, začali Andi Gutmans a Zeev Suraski pracovat na přepisu jádra PHP. Cílem návrhu bylo zlepšit modularitu kódové báze PHP a zvýšit výkon pro složité aplikace. Tyto aplikace byly schopny pracovat s PHP 3.0 (díky novým možnostem a podpoře široké škály databází a API od jiných tvůrců), ale PHP 3.0 nebylo navrženo pro efektivní práci tak náročných aplikací.

Nový engine, nazvaný **Zend Engine** (název byl sestaven z jejich křestních jmen - Zeev a Andi), úspěšně splnil cíle návrhu a byl uveden v polovině roku 1999. PHP 4.0, založené na tomto enginu a doplněné širokou škálou nových prvků, bylo oficiálně uvolněno v květnu 2000, necelé dva roky po svém předchůdci, PHP 3.0. K podstatně zvýšenému výkonu této verze přidává PHP 4.0 další klíčové prvky, jako je podpora pro mnoho různých WWW serverů, HTTP sessions, buffering výstupu, bezpečnější způsoby zpracování vstupů uživatele a mnoho nových jazykových konstruktů.

PHP 4 používaly a používají (odhadem) stovky tisíc vývojářů a nainstalované PHP hlásí několik milionů serverů - tedy přes 20 % domén na Internetu.

Vývojový tým PHP zahrnuje tucty vývojářů, stejně tak jako tucty dalších lidí, kteří pracují na projektech spojených s PHP, jako je PEAR a dokumentační projekt.

3.2.4 PHP 5

Na vývoji PHP 5 se začalo pracovat již v roce 2002. Základem této verze je zcela přepracovaný **Zend Engine 2**, který jednak přinesl vyšší výkon kritických PHP aplikací, jednak umožnil zakomponovat do PHP řadu pokročilejších programovacích struktur, čímž především otevřel cestu kvalitnější podpoře objektově orientovaného programování v PHP.

První betaverze PHP 5 byly veřejnosti k dispozici na jaře roku 2003 a první oficiální verze PHP 5 byla uvolněna 13. července 2004.

PHP 5 je do značné míry zpětně kompatibilní s PHP 4. V nové verzi jazyka byly především posíleny bezpečnostní mechanismy (což může vést k nefunkčnosti některých špatně postavených aplikací pro PHP 4) a uveden nový, podstatně kvalitnější objektový model, umožňující používat PHP jako skutečný objektově orientovaný jazyk.

3.3 Princip PHP

V době internetového pravěku byly všechny internetové stránky statické. Prostě tak, jak byla stránka napsána, tak byla odeslána do prohlížeče a tak byla také zobrazena. To pochopitelně časem přestávalo stačit, a proto byla vyvinuta celá řada technologií, které měly stránky rozpohybovat. Zhruba řečeno se dají tyto technologie rozdělit do dvou skupin, na "klientské" a "serverové".

"Klientské" technologie se spoléhají na jednoduchou věc: Spolu s HTML stránkou je prohlížeči odeslán i nějaký kus programového kódu a ten je ve vhodnou chvíli na "cílovém" počítači spuštěn. Vhodná chvíle může nastat například při kliknutí na tlačítko, při najetí myší na odkaz, při otevření okna prohlížeče a podobně. O spuštění klientského kódu se stará prohlížeč - a to může být nevýhoda. Prohlížeč totiž musí znát programovací jazyk, v němž je kód napsán. Příkladem technologií běžících na straně klienta je například Java script.

"Serverové" technologie jsou založeny na jiném principu. Když prohlížeč požaduje webovou stránku ze serveru, server tuto stránku nejprve sestaví a pak odešle. Servery mohou (a také to často dělají) sestavovat pokaždé jinou stránku v závislosti na tom, co přesně prohlížeč požaduje.

PHP JE TECHNOLOGIE BĚŽÍCÍ NA SERVERU. Typický PHP skript obsahuje jednak kusy normálního HTML kódu, a jednak kusy programového kódu. Když webový server obdrží požadavek na zpracování takového skriptu, vezme:

- kusy HTML kódu tak, jak jsou
- části PHP programového kódu provede
- výsledek zkombinuje a odešle prohlížeči

Tato filozofie fungování je nesmírně mocná. Server totiž může provést jednu nebo dokonce několik operací a výsledek poslat do prohlížeče jako obyčejnou HTML stránku.

4 MYSQL

4.1 Úvod

S databázemi se samozřejmě setkáváme denně. V nejširším slova smyslu je databáze i seznam věcí, které máme koupit k obědu, výpis z účtu nebo třebas seznam uskutečněných telefonních hovorů. V počítačovém slova smyslu se pod výrazem "databáze" obvykle rozumí software, který spravuje určitý "balík" dat a umožňuje uživatelům tento "balík" dat nějak rozumně měnit a spravovat.

Když se například vrátím k výpisu telefonních hovorů, bude situace následující:

- Někdo nebo něco (dejme tomu telefonní ústředna) bude zapisovat do databáze údaje o uskutečněných hovorech.
- Někdo nebo něco jiného bude ze zapsaných údajů nějaké vybírat (například program pro tisk výpisu bude vybírat podle telefonního čísla a období)
- Někdo další může chtít data upravovat (třebas reklamační oddělení může chtít změnit záznamy v případě oprávněné reklamace)

Z toho vidíme, že databáze není jen prosté shromaždiště, úložiště dat, ale že většinou slouží zároveň k jejich organizaci, třídění, prohledávání, seskupování a podobně. K typické dnešní databázi patří rovněž to, že zároveň chce s daty pracovat více uživatelů.

S tím souvisí jiná důležitá věc: S rozvojem sítí a zejména internetu se stalo obvyklé, že databáze mohou být přístupné vzdáleně. Taková databáze umístěná centrálně může poskytovat data mnoha víceméně nezávislým odběratelům. Z celého toho popisu bude pravděpodobně všem zřejmé, že databázové programy obecně zažívají zlaté časy a že databází bude existovat celá řada.

4.2 Dělení databází

Rozdělit databáze je kupodivu čím dál tím složitější, protože jednotlivá kritéria se v poslední době vzájemně překrývají a mnoho databází umí hodně podobných věcí. Nicméně, pokusím se alespoň nastínit, jak různě se dají databáze dělit.

- Objektové a relační - tam se databáze liší podle způsobu, jak ukládají data.

- Jednouživatelské a víceuživatelské - Podle toho, kolik uživatelů se k databázi může připojit. MySQL je víceuživatelská databáze.
- Souborové a systémové - liší se tím, zda použijí pro uložení dat jeden soubor (např. dbf), nebo zda je úložiště dat nějak zabudováno do systému. MySQL je systémová databáze.
- Podle licence a ceny - Kód databáze může být uzavřený nebo otevřený, šíření software může být svobodné nebo může podléhat nějakým podmínkám, databáze se může využívat bez poplatků nebo může jít o placené software. Licence MySQL je duální.
- Podle toho, kde běží - na jednoplatformní a multiplatformní. Jednoplatformní poběží jen na některém systému (třeba na Windows), multiplatformní na více systémech. MySQL je multiplatformní databáze.
- Podle velikosti, výkonu a vhodnosti nasazení - Databáze mívají limity ve velikosti, počtu současně přihlášených uživatelů, počtu současně probíhajících procesů a podobně. Je těžké někam zařadit MySQL. Obecně je totiž obtížné nějak rovnoprávně posoudit databáze. Diplomaticky můžeme říci, že MySQL je někde uprostřed pomyslného žebříčku vhodnosti nasazení a že v malých až středních projektech ji rozhodně použít můžete

Databáze lze samozřejmě dělit i podle celé řady dalších kritérií. Nejdůležitější přitom je, co všechno daná databáze "umí" s daty provádět.

4.3 Popularita MySQL

MySQL je velmi populární databáze. Podle mnoha zdrojů je to rovněž velmi rychlá databáze. Nemá však tolik funkcí a možností jako některé konkurenční databázové systémy. Vybrat si vhodnou databázi je tedy klasický kompromis mezi rychlostí softwaru a jeho schopnostmi. MySQL má samozřejmě své zastánce a odpůrce. Odpůrci například často tvrdí, že MySQL je tak rozšířená proto, že webhostingové společnosti ji často nabízejí pro hostované weby jako součást portfolia svých služeb. Kdyby se webhostingové společnosti rozhodly upřednostnit jiný software, popularita MySQL by podle jejich odpůrců podstatně klesla.

Naproti tomu zastánci MySQL tvrdí, že webhostingové společnosti nabízejí MySQL proto, že je dobrá, a kdyby existovala lepší databáze než MySQL, byla by nabízena. Zastánci MySQL prostě tvrdí, že trh si MySQL vybral - a to podle nich jasně poukazuje na její kvality.

Pravda asi bude někde uprostřed mezi těmito tvrzeními. Nemám na to vyhraněný názor, ale MySQL podle mého ještě nějakou dobu bude velmi rozšířená, takže se vyplatí ji používat. Abych byl ale objektivní, podívejme se ještě na to, jaké nejvážnější konkurenty MySQL ve světě databází má.

4.4 Konkurenční databáze

Slovem "konkurenční" zde myslím rivaly MySQL z hlediska možností, nikoli ceny. Jinými slovy, je to úvaha o softwaru, nikoli marketingu. Jelikož marketing databází se rozvíjí tak prudce jako celé toto odvětví, jen velmi dobře informovaní lidé by mohli předpovědět, jakým směrem se bude vývoj ubírat (a ještě to není jisté).

Asi největším soupeřem MySQL o pozornost uživatelů byla donedávna databáze PostgreSQL. PostgreSQL umí běžet na Windows NT (2000, XP) a na celé řadě UN*X platforem a má podstatně více možností pro uživatele než MySQL. Je však o něco pomalejší.

Podle výsledků nedávného průzkumu firmy Evans Data Corporation však v poslední době dobývá tento segment trhu poměrně výrazným způsobem databáze Firebird. To upřímě řečeno mnoho lidí překvapilo. Firebird je rovněž open-source databáze, může běžet na systémech Windows, Linux, Mac OS X, HP-UX a Solaris a má rovněž více možností než MySQL. Je také o něco pomalejší.

Troufnu si však tvrdit, že pokud někdy něco vytlačí MySQL z pozice, kterou teď má, bude to jedna z dvojice PostgreSQL, Firebird. Jiné vážné konkurenty MySQL podle mě v současné době nemá.

II. PRAKTICKÁ ČÁST

5 WEBOVÝ SLOVNÍK

Jak již bylo zmíněno v úvodu, dnešní Internet obsahuje nepřeberné množství informací všeho druhu avšak většinou jsou tyto informace neucelené a občas velice nepřehledně umístěny. Slovník je svým způsobem soubor přehledně uspořádaných a ucelených informací umístěných na jednom místě např. kniha. Ale jelikož žijeme v době počítačů, je zřejmě zbytečné se zabývat psaním knihový slovníků, které by sepsat trvalo i několik měsíců možná i let. Nabízí se zde řešení, vytvořit internetový slovník a do něj posupně přidávat jednotlivé pojmy. Navíc je tato volba nenákladná, a uživateli internetu určitě využívanější než knihy.

Před samotnou tvorbou bakalářské práce jsem provedl menší průzkum českého internetu, jak to vypadá s českými internetovými slovníky. Výsledek byl, že existuje spousta internetových anglicko-českých a česko-anglický slovníku, ale skoro vůbec jsem nenašel slovníky, které by byly nějakým způsobem zaměřeny na automatizaci. V této oblasti jsem nalezl jediné stránky, které se této problematice věnují, na adrese www.e-automatizace.cz, kde se nachází i slovník.

5.1 Použitý software

Pro tvorbu internetového slovníku bylo využito serverového skriptovacího jazyka PHP a relační databáze MySQL. Zvolil jsem tuto kombinaci z toho důvodu, že PHP a MySQL jsou nejpoužívanější technologie na světě a tudíž jejich využití je možné skoro všude a navíc je zdarma.

Celý systém byl naprogramován pomocí freewarového programu PSPad, který jsem použil pro psaní HTML a PHP kódu. Vyznačuje se hlavně podporou zvýrazňování syntaxe jednotlivých programovacích jazyků, což usnadňuje přehlednost psaného kódu. Všechny soubory slovníku jsou kódovány v ISO-8859-2.

Pro psaní kaskádových stylu, takzvaných css souborů, jsem využil opět freewarového programu Topstyle Lite, který mi přišel jako nejlepší volba.

Systém byl vyvíjen a testován v operačním systému MS Windows XP Professional SP2 a pro provozování webového serveru jsem použil freewarový balík Vertrigo Server verze 2.15, který jsem stáhnul z adresy http://vertrigo.sourceforge.net/.

Tento softwarový balík obsahoval:

- Apache 2.0.59
- PHP 5.2.1
- MySQL 5.0.27
- PhpMyAdmin 2.9.2
- Zend Optimizer 3.2.2

Pro reálný provoz na internetu jsem využil freehostingu na serveru www.ic.cz , kde jsem založil účet, abych mohl systém testovat i jinde. Momentálně se slovník nachází na adrese www.automatization.ic.cz

Webový slovník byl naprogramován tak, aby jej bylo možné nasadit na server s PHP 5 tak i s PHP 4. Při psaní HTML kódu a kaskádových stylů, byl brán zřetel na validitu kódu a dodržování pravidel konsorcia W3C.

Dále byl systém graficky maximálně optimalizován pro nejpoužívanější internetové prohlížeče jako jsou Internet Explorer, Firefox, Opera a Netscape.

5.2 Struktura databáze

Slovník pro svou činnost využívá nejrozšířenější a nejpoužívanější databázi MySQL. Pro správnou funkci slovníku musí obsahovat databáze tři tabulky:

- *setting*

- *user*

- *words*

Nyní si jednotlivé tabulky databáze popíšeme.

5.2.1 Tabulka *setting*

Tato tabulka slouží k nastavení systému jako je třeba název slovníku, klíčová slova pro vyhledávače, počet zobrazených výsledků na stránku atd. Hlavičku tabulky tvoří:

item	value

Tabulka 1: Struktura tabulky *setting*

Sloupec *item* s datovým typem *varchar(17)* slouží k ukládání názvu jednotlivých nastavení a může obsahovat maximálně 17 libovolných znaků.

Sloupec *value* s datovým typem *varchar(128)* obsahuje hodnoty k jednotlivým řádkům položky *item* a může obsahovat maximálně 128 libovolných znaků.

Příklad uložené položky v tabulce *setting*

item	value
keywords	automatizace, slovník, web …

Tabulka 2: Příklad uložené položky v tabulce *setting*

5.2.2 Tabulka *user*

V této tabulce je uložen vytvořený uživatelský účet pro přihlášení do administrace slovníku. Jelikož se jedná o tzv. single user system neboli jednouživatelský systém, kde existuje pouze jediný účet pro správu slovníku. Hlavičku tabulky tvoří:

id	username	password	realname

Tabulka 3: Struktura tabulky *user*

Sloupec *id* s datovým typem *int(4) primary key* slouží jako celočíselný identifikátor řádku a jeho atribut primary key slouží jako označení pro primární klíč, který musí být vždy unikátní a nesmí chybět. Primární klíč může být v jedné tabulce použit pouze jednou.

Sloupec *username* s datovým typem *varchar(16)* slouží k uložení uživatelského (přihlašovacího) jména. Může obsahovat až 16 libovolných znaků.

Sloupec *password* s datovým typem *varchar(32)* slouží k ukládání přihlašovacího hesla, které se zakóduje před uložením přes jednosměrovou šifru funkce *md5()*.

Sloupec *realname* s datovým typem *varchar(64)* slouží k uložení skutečného jména administrátora slovníku. Může obsahovat až 64 libovolných znaků.

5.2.3 Tabulka *words*

V této tabulce je uložen veškerý obsah slovníků, jde o tabulku, kde se ukládají jednotlivá slova. Hlavičku tabulky tvoří:

id	czech	english	description	link

Tabulka 4: Struktura tabulky *words*

Sloupec *id* s datovým typem *int(6) primary key auto_increment* slouží jako celočíselný identifikátor daného řádku, opět obsahuje atribut *primary key*, což vlastně označuje daný sloupec za primarní klíč a atribut *auto_increment* slouží k tomu, aby se čísla neopakovala, tudíž se stará o to, aby každý nově vložený záznam měl své jedinečné číslo. To lze zajistit v případě tohoto atributu navýšením posledního nejvyššího čísla *id* o jedničku.

Sloupec *czech* a *english*, oba s datovým typem *varchar(64)* slouží k uložení českého a anglického překladu. Každý z těchto sloupců může obsahovat až 64 libovolných znaků.

Sloupec *description* s datovým typem *text* slouží k popisu jednotlivých překládaných pojmů. Datový typ *text* je rozsáhlé datové pole o maximální délce 65 535 bajtů.

Sloupec *link* s datovým typem *binary(1)* slouží k tomu, aby informoval při výpisu výsledku hledání zda nalezený překlad obsahuje i popis. Informace o tom zda obsahuje či nikoliv je uložena v binární hodnotě 0 nebo 1.

5.3 Instalace

Před samotným začátkem instalace slovníku je potřeba zajistit hosting ať už placený nebo neplacený, popřípadě pokud chcete tento slovník jen otestovat tak si stáhnout nějaký softwarový balík na vytvoření web serveru u Vás na počítači. Při vyvíjení tohoto slovníku jsem využíval webového serveru Vertrigo. Až jsem měl systém hotový, přesunul jsem vytvořený slovník na freehosting, kde jsem provoz a funkčnost celého systému ještě doladil.

Pro reálný provoz slovníku jsem si zřídil freehostingový účet na serveru www.ic.cz. V této části bakalářské práce popíši instalaci slovníku na skutečný server. Slovník je momentálně nainstalovaný na adrese *www.automatization.ic.cz*

5.3.1 Přenesení systému na server

Pokud již máme zajištěný hosting na některém serveru, musí podporovat minimálně PHP 4 a databáze MySQL, můžeme slovník nakopírovat na server pomocí softwaru, který podporuje přenos souboru přes FTP. Já jsem k tomu použil plugin FireFTP, který jsem doinstaloval do internetového prohlížeče Firefox.

Pokud nakopírování slovníku proběhlo v pořádku, otevřeme okno prohlížeče a do něj zadáme adresu, kde jsem slovník nahráli, v mém případě napíšu www.automatization .ic.cz

Obrázek 1: Stav po nakopírování slovníku na server

Pokud se Vám v prohlížeči zobrazí něco jiného než na obrázku 1, zřejmě se bude jednat o chybu na serveru nebo jste spatně zkopírovali soubory slovníku.

Nyní nám systém zahlásil, že nenašel databázi, ke které by se mohl připojit. Proto nyní musíme vytvořit databázi a vytvořit v ní patřičnou strukturu tabulek pro správnou funkci slovníku.

5.3.2 Vytvoření struktury databáze

Ve většině případech lze databázi vytvořit přímo v administraci účtu na serveru, kde se nachází hosting a vytvoření je velmi jednoduché, někdy stačí jen pouhé kliknutí na tlačítko v administraci.

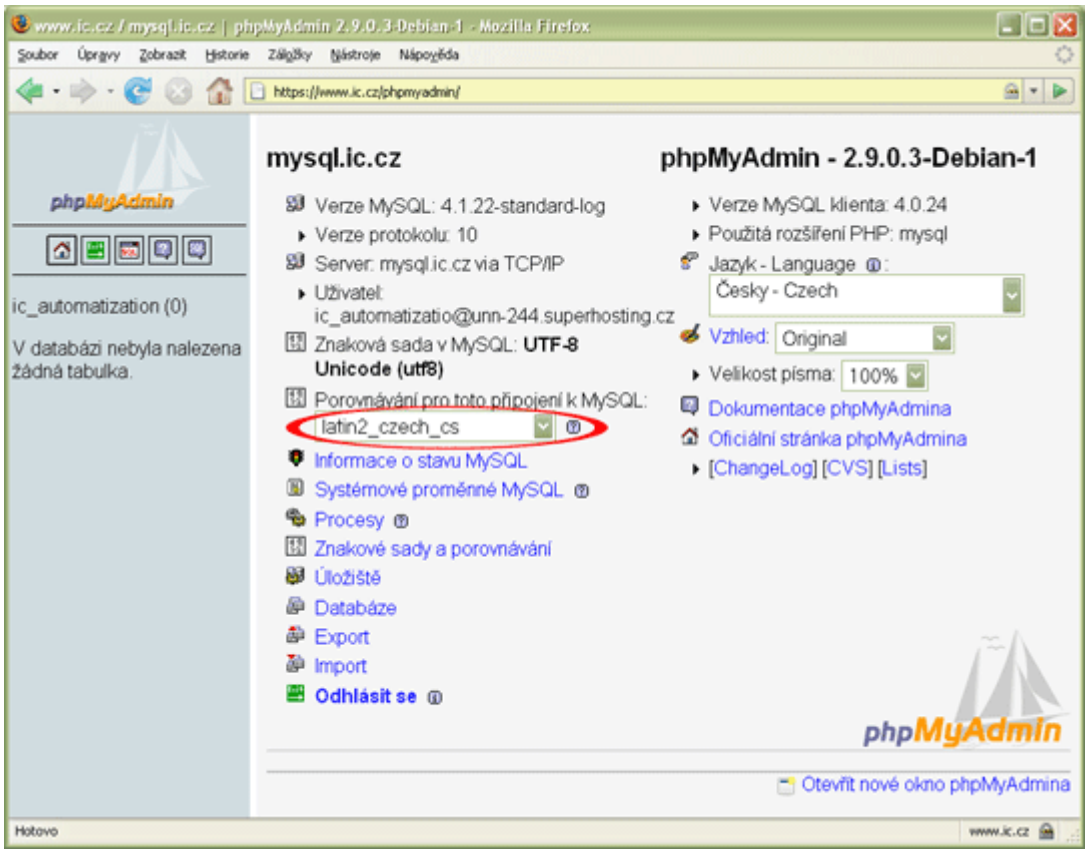
Pokud se nám podařilo databázi vytvořit, je důležité si někam poznačit název a heslo pro přístup do databáze, protože je budeme později potřebovat.

Skoro na každém web hosting se používá pro správu databází MySQL systém s názvem phpMyAdmin, který je vlastně vytvořen jako internetový systém pro správu databází přes webový prohlížeč.

Další postup instalace bude proto prováděn přes tento systém.

5.3.2.1 Přihlášení do phpMyAdmin

V první řadě se musíme nejprve přihlásit do systému phpMyAdmin. V našem případě to bude na adrese http://mysql.ic.cz. Po zadání adresy by se měl objevit přihlašovací formulář, do kterého vyplníte uživatelské jméno a heslo, které jsme si buď zadávali sami nebo nám bylo automaticky vygenerováno při vytváření databáze. Po úspěšném přihlášení by se mělo zobrazit toto:



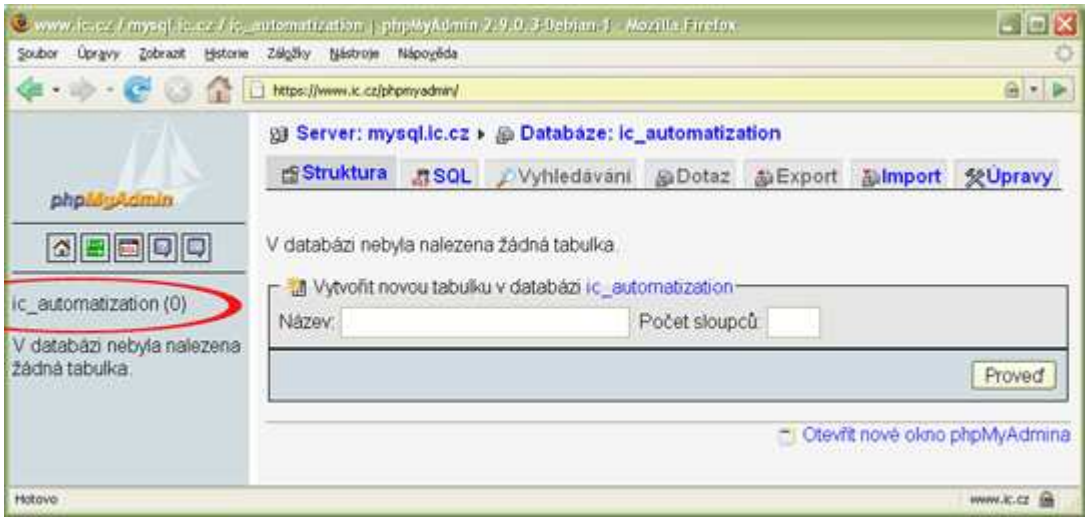
Obrázek 2: Úspěšné přihlášení do systému phpMyAmin

5.3.2.2 Nastavení a vytvoření základní struktury databáze

Po úspěšném přihlášení do systému je v první řadě třeba nastavit položku „Porovnávání pro toto připojení k MySQL“ na hodnotu „latin2_czech_cs“.

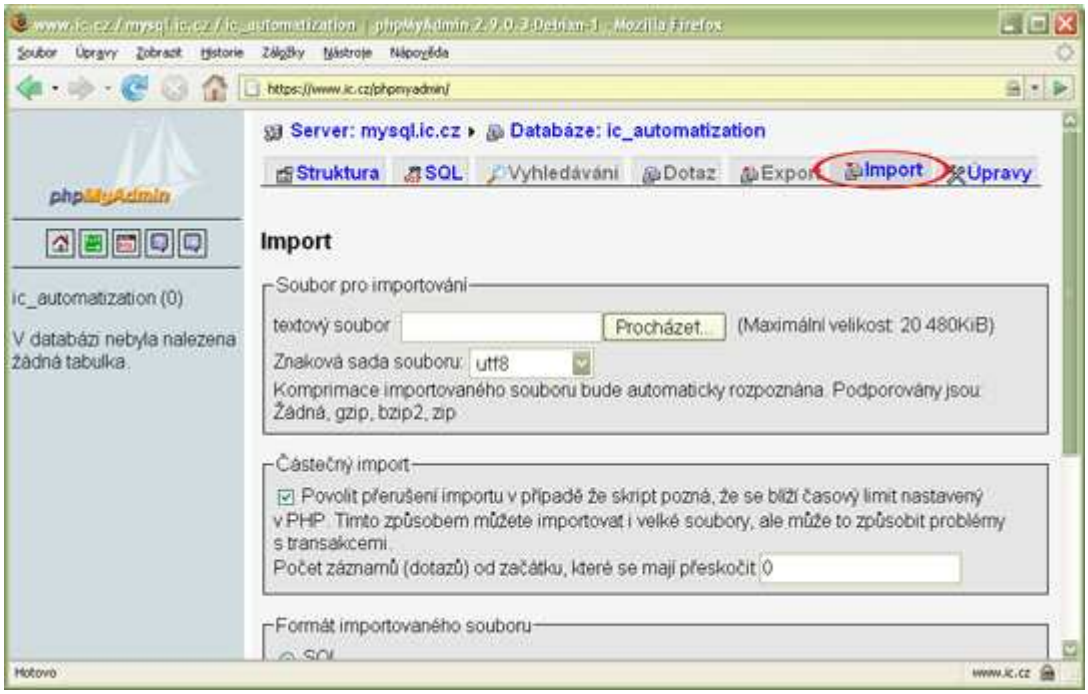
Nyní můžeme přistoupit k vytváření tabulek v databázi. Na přiloženém CD se nachází soubor *struktura.sql*, který obsahuje SQL příkazy pro vytvoření potřebných tabulek v databázi.

Klikneme v levém panelu na název databáze.



Obrázek 3: Otevření databáze

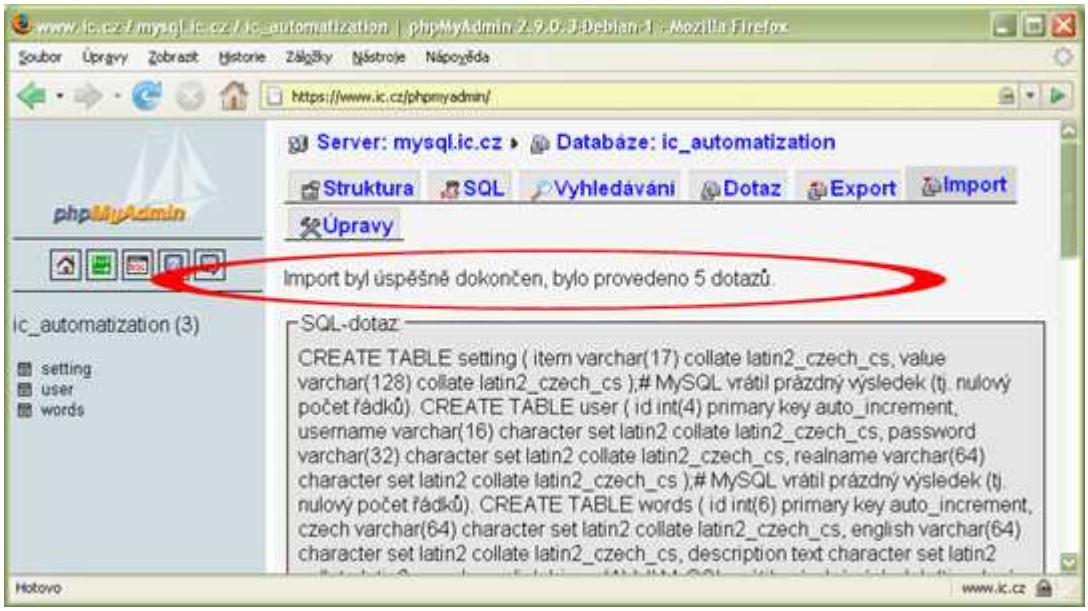
Nyní vidíme, že databáze je opravdu prázdná. Klikneme proto na záložku „Import“.



Obrázek 4: Obsah položky „Import“

Nyní klikneme na tlačítko „Procházet“ a vyhledáme soubor *struktura.sql*, dále nastavíme u položky „Znaková sada souboru“ na hodnotu „utf8“ a klikneme na tlačítko „Proveď“, které se nachází ve spodní části stránky.

Po úspěšném vykonání příkazů by měla databáze vypadat takto:



Obrázek 5: Úspěšné vykonání SQL příkazů

V levém panelu je vidět, že přibyly tři nové položky, jedná se o tři nové tabulky. Nyní je databáze připravena, ale zatím neobsahuje žádná slova, pouze nastavení.

Proto opět klikneme na záložku „Import“ a provedeme ten samý postup jako pro vytváření struktury, jen s tím rozdílem, že do cesty zadáme soubor *words.sql*.

5.3.3 Nastavení konfiguračního souboru *config.php*

Pokud předcházející kroky proběhly v pořádku, databáze je kompletně připravena a teď už jen stačí správně nastavit konfigurační soubor *config.php* na serveru, který se nachází ve složce *admin*. Tento konfigurační soubor slouží k nastavení připojení k databázi. V souboru je potřeba nastavit tyto položky:

```
$db_server = "mysql.ic.cz";
```

V této položce se nastavuje adresa SQL serveru

```
$db_username = "ic_automatizatio";
```

V této položce se nastavuje uživatelské jméno k SQL serveru

```
$db_password = "*****";
```

V této položce se nastavuje uživatelské heslo k SQL serveru

```
$db_name = "ic_automatization";
```

V této položce se nastavuje název databáze

Pro editaci konfiguračního souboru doporučuji editor s podporou kódování ISO-8852-9 např. PSPad.

Po nastavení prvních čtyř položek je nutné soubor uložit a přepsat soubor umístěný na serveru. Pokud jsme nastavili všechny položky dobře, měl by celý systém být provozu schopný. Můžeme to vyzkoušet tak, že do internetového prohlížeče zadáme adresu, kde jsme slovník nahráli. Měla by se nám zobrazit úvodní stránka slovníku, kde je informace o aktuálním počtu slov ve slovníku a úvodní text. Po instalaci je úvodní text nastaven „* * * Slovník byl úspěšně nainstalován * * *“.

Obrázek 6: Úspěšně nainstalovaný slovník

Upozornění: po nainstalování slovníku je nastaveno výchozí administrátorské jméno „root“ a heslo „toor“. Z bezpečnostních důvodů doporučuji co nejdříve po instalaci změnit alespoň heslo.

5.4 Popis jednotlivých částí slovníku

Celý slovník se skládá ze dvou částí. Veřejná a administrační. Nyní si jednotlivé části popíšeme.

5.4.1 Veřejná část

Tato část slovníku bude asi nejvyužívanější částí celého systému. Je to ta část, které bude sloužit veřejnosti k užívání slovníku. Na tuto část se dostaneme, zadáme-li do internetového prohlížeče adresu, kde jsme slovník nakopírovali, v našem případě *www.automatization.ic.cz*

Obrázek 7: Úvodní stránka slovníku

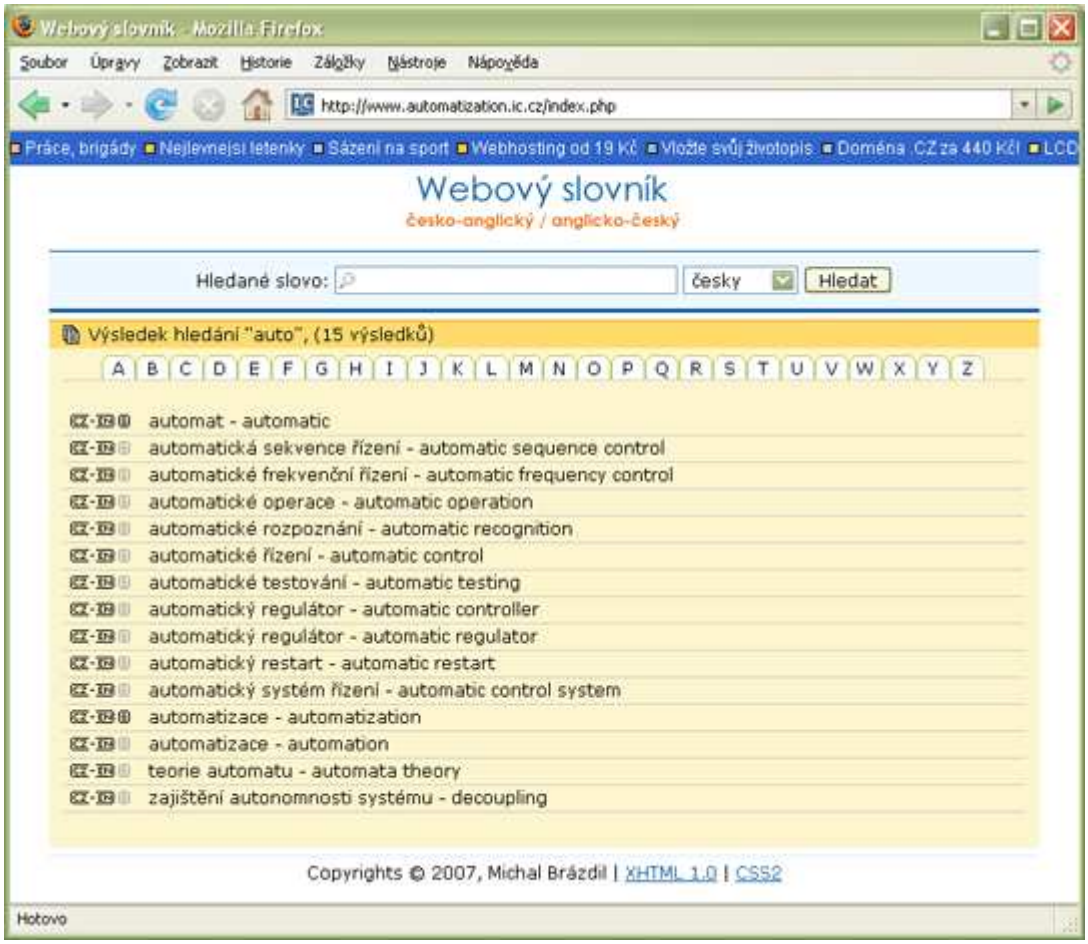
Jak je vidět na obrázku 7, je veřejná část rozdělena na dvě části.

První část slouží pro rychlému vyhledání zadaného výrazu, druhá část slouží k listování ve slovníku.

5.4.1.1 Rychlé vyhledávání

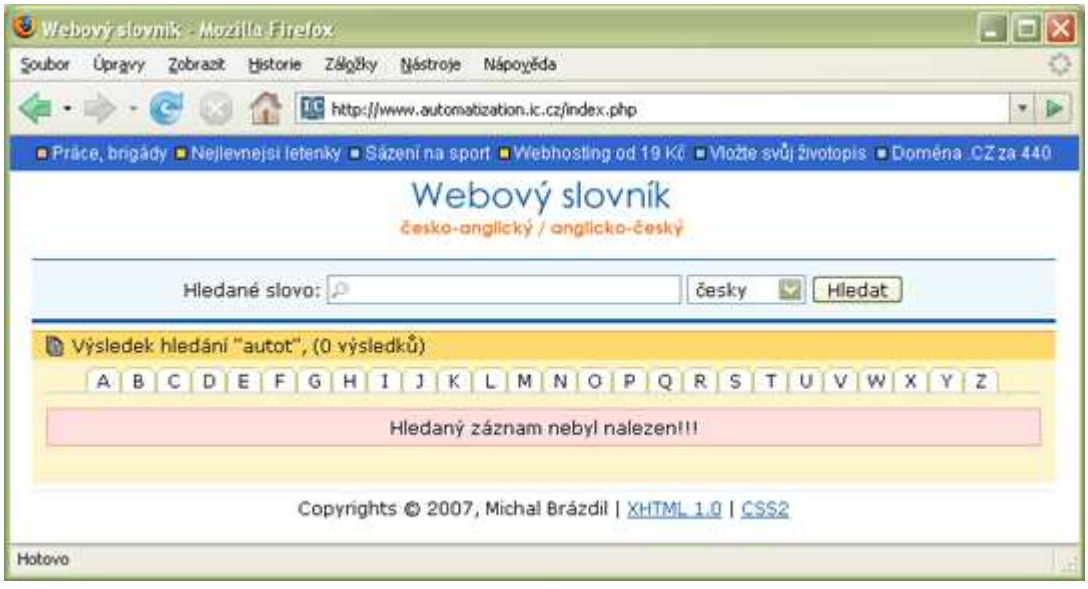
Tato část slovníku bude asi nejvyužívanější, protože nabízí jednoduché a hlavně rychlé vyhledání slov ve slovníku. Do pole „Hledané slovo“ zadáme hledaný výraz, nastavíme v jaké části slovníku chceme hledat, jestli v české či anglické, a klikneme na tlačítko „Hledat“. Pokud bude hledaný výraz ve slovníku, výsledek se nám zobrazí takto:

Příklad hledání slova „auto“



Obrázek 8: Příklad hledání slova „auto“

Pokud hledaný výraz nebude nalezen ve slovníku, zobrazí se chybová hláška.

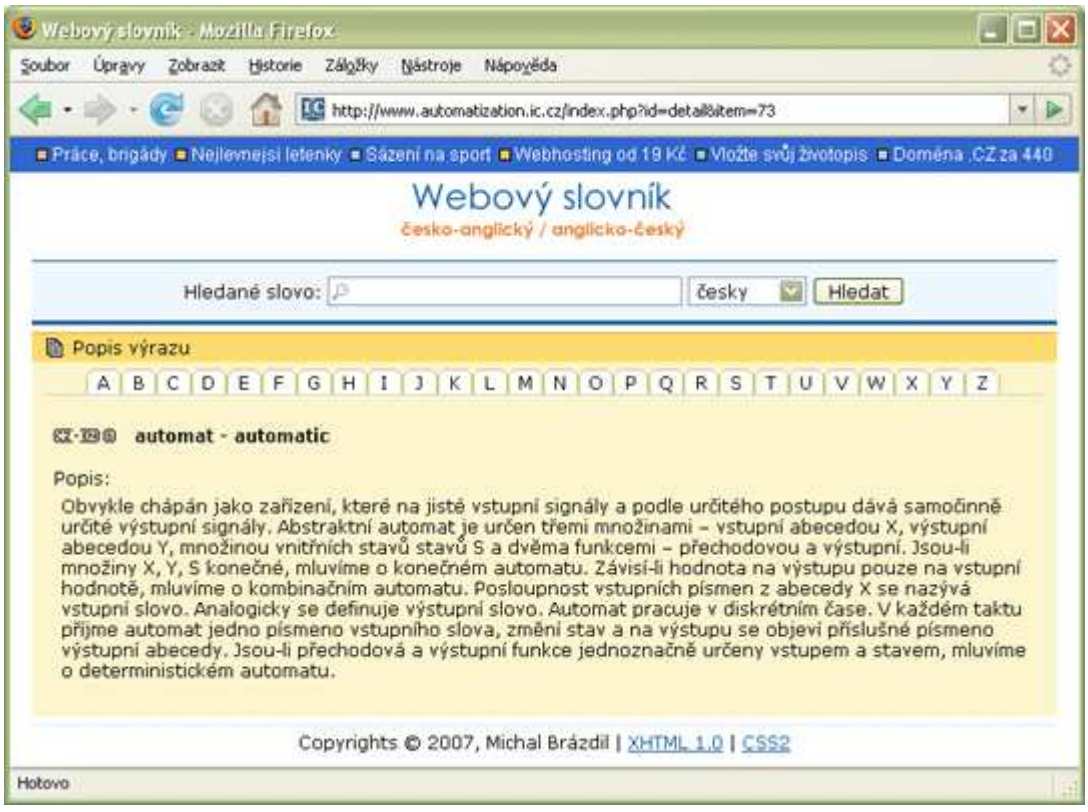


Obrázek 9: Hledaný záznam nebyl nalezen

Pří hledání slov se nejprve prohledají slova začínající zadaným výrazem a po té se hledají slova která obsahují zadaný výraz.

Pokud je výsledkem vyhledání seznam slov, které byly nalezeny, jsou u každého slova zobrazeny na levé části symbol popsaného listu papíru. Pokud je list šedý, znamená to, že daný výraz neobsahuje podrobný popis. V opačném případě je symbol černý a na dané slovo lze kliknout jako na další odkaz.

V případě kdy jsme zkoušeli hledat slovo „auto“ ve slovníku, bylo zobrazena slova začínající na „auto“ např. automat, automatické řízení atd., ale také slova např. teorie automatu, zajištění autonomnosti systému. Jak je dále vidět, většina nalezených slov nemá podrobný popis, proto mají na začátku světlý symbol. Na prvním řádku se, ale nachází slovo, které popis má a tudíž můžeme na něj kliknout.

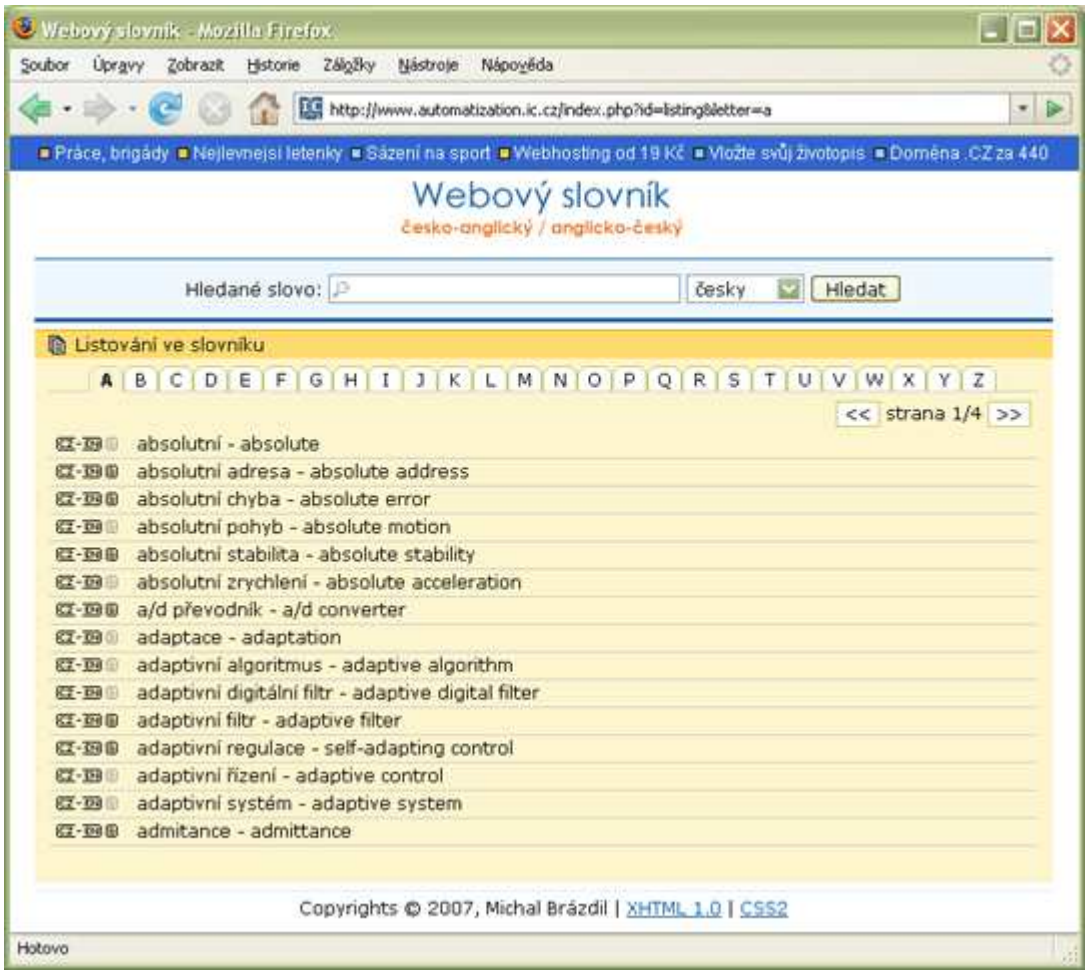


Obrázek 10: Podrobný popis slova „automat“

Jak je vidět na obrázku 10, po kliknutí se nám zobrazí podrobnosti o daném výrazu.

5.4.1.2 Listování ve slovníku

Listování ve slovníku nám slouží k procházení celého obsahu slovníku. Když nám nestačí výsledek hledání, můžeme hledaný výraz hledat ručně a postupně procházet celý slovník. Listování slovníkem je rozděleno do kategorií podle abecedy. Při procházení slovníku se nám zobrazují výsledky úplně stejně jako pří rychlém vyhledávání.

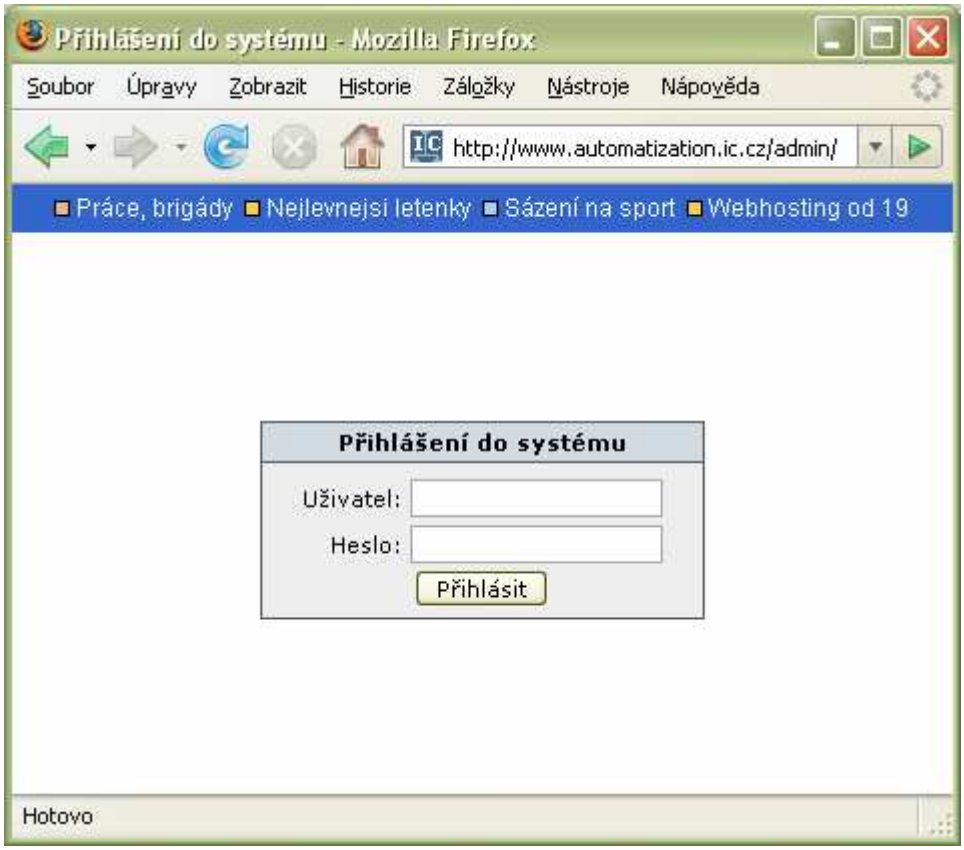


Obrázek 11: Listování ve slovníku

5.4.2 Administrační část

Administrační část byla ta nejtěžší část z celé této bakalářské práce. Slouží k nastavení a správě slov ve slovníku. Do administrace se přihlásíme tak, že zadáme adresu jako k veřejné časti a za tuto adresu přidáme */admin*. V našem případe *www.automatization.ic.cz/admin*

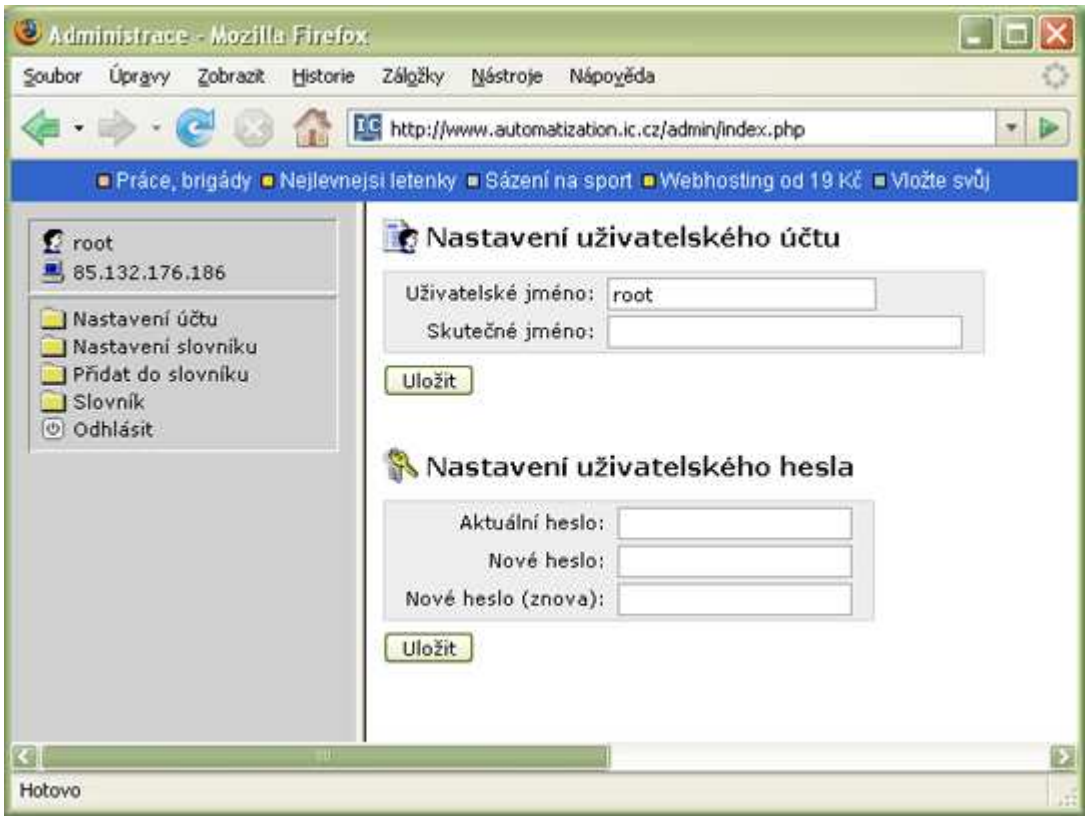
Po zadání této adresy by jsme se měli dostat na přihlašovací formulář do administrace.



Obrázek 12: Přihlašovací formulář do administrace

Pro přihlášení je nutné vyplnit přihlašovací jméno uživatele a uživatelské heslo. Po instalaci je defaultně nastaveno uživatelské jméno „root“ a heslo „toor“. *Proto hned po prvním přihlášení je z bezpečnostních důvodů lepší změnit alespoň heslo!*

Po správném zadání uživatelského jména a hesla by jsme sem měli dostat do administrace.



Obrázek 13: Úvodní stránka administrace, nastavení účtu

Administrace se skládá z levého a pravého panelu. V pravém panelu se zobrazují obsahy jednotlivých položek menu např. formuláře pro ukládání nastavení. V levém panelu se nachází informace o přihlášeném uživateli, jako je uživatelské jméno a IP adresa. Pod těmito informacemi se nachází menu administrace, které obsahuje celkem pět položek. Nyní si popíšeme každou část samostatně.

5.4.2.1 Nastavení účtu

Při každém přihlášení do administrace je uživatel automaticky přesměrován do nastavení účtu. V této části administrace může uživatel nastavovat uživatelské jméno, skutečné jméno a heslo pro přihlášení do administrace.

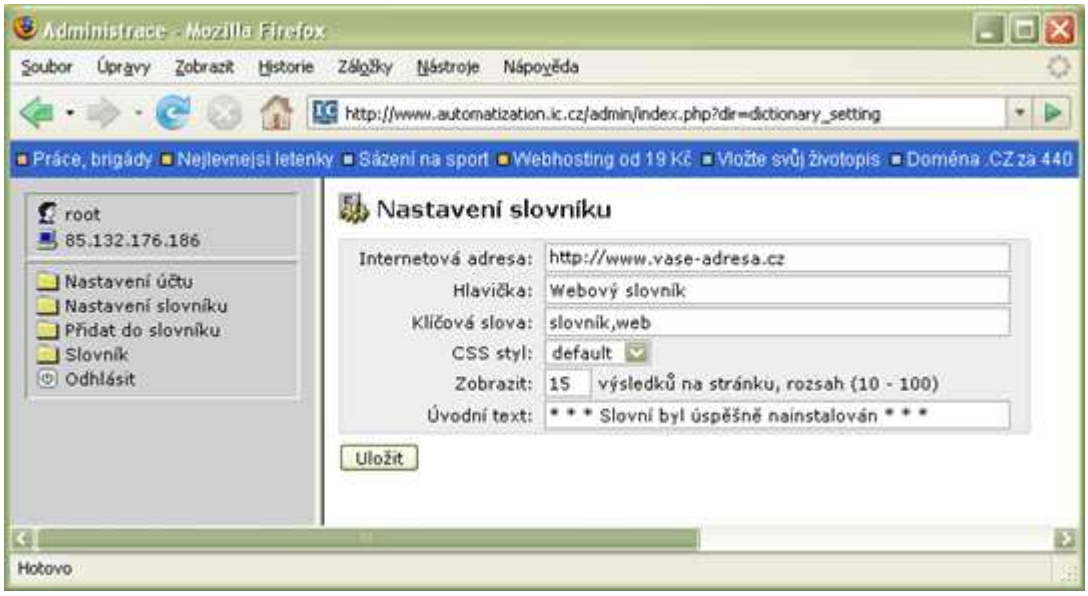
Pokud budeme chtít změnit uživatelské jméno, stačí jen kliknout na příslušné políčko a napsat nové uživatelské jméno. Pro uložení je potřeba kliknout na tlačítko „Uložit“. Pokud

bude uživatelské jméno prázdné nebo bude obsahovat zakázané znaky, budeme na to upozorněni ve spodní části formuláře chybovou hláškou.

Položka „Skutečné jméno“ je nepovinná položka, slouží jen jako doplněk k uživatelskému účtu, není nijak systémem využívána.

Nastavení uživatelského hesla se provádí tehdy pokud chceme změnit stávající heslo, např. po prvním přihlášení do systému. Pro změnu hesla je nutné zadat aktuální heslo a poté zadat dvakrát stejně nové heslo. Pokud jsme vše vyplnili dobře, můžeme kliknout na tlačítko „Uložit“. Pokud jsme zadali aktuální heslo dobře a nové hesla stejně systém stávající heslo nahradí novým. Změny se projeví až po odhlášení.

5.4.2.2 Nastavení slovníku



Obrázek 14: Administrace - Nastavení slovníku

V této části administrace se nastavuje veřejná část slovníku.

Internetová adresa – touto položkou nastavujeme internetovou adresu slovníku, které je použita pro validaci slovníku pomocí html a css validátoru. V našem případě „www.automatization.ic.cz“.

Hlavička – tato položka vypisuje zadaný text do lišty prohlížeče ve veřejné části slovníku. V našem případě „Webový slovník anglických pojmů z oblasti automatizace“.

Klíčová slova – tuto položku vypisujeme pouze jednotlivými slovy oddělené čárkami, které vystihují problematiku, kterou se slovník zabývá. Slova se nezobrazují přímo v prohlížeči, ale jsou začleněny do HTML kódu. V případě, že na slovník zabrouzdá bot třeba googlu uloží si odkaz pod těmito klíčovými slovy.

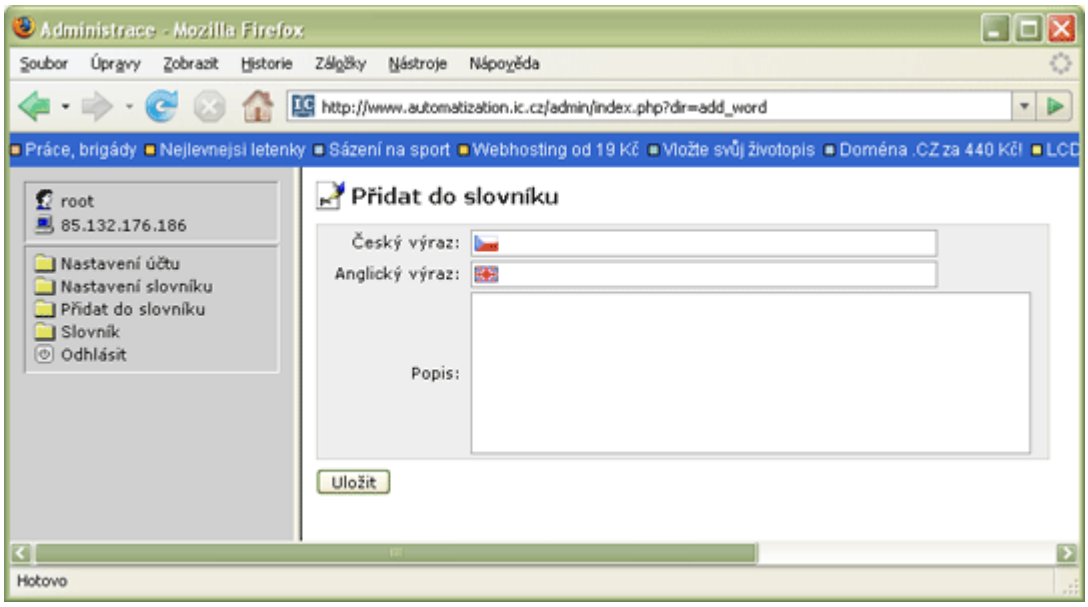
CSS styl – tato položka slouží k nastavení grafického designu veřejné části slovníku. Defaultně je nastavena tato položka na hodnotu „default“, ale můžeme ji změnit. V našem případě na „fai-utb“.

Zobrazení – tato položka se stará o počet vypsaných položek hledaní. Toto nastavení platí jak pro rychlé vyhledání tak i pro listování slovníkem. Jako výchozí hodnota je nastavena „15“.

Úvodní text – tato položka nastavuje na úvodní stránce slovníku text, který se zobrazuje hned pod textem, který informuje o počtu slov uložených v databázi. V našem případě můžeme nastavit „Slovník je tvořen částečným překladem keyword listu IFAC“.

Po nastavení všech položek můžeme nastavení uložit kliknutím na tlačítko „Uložit“. Nastavení se projeví okamžitě.

5.4.2.3 Přidat do slovníku

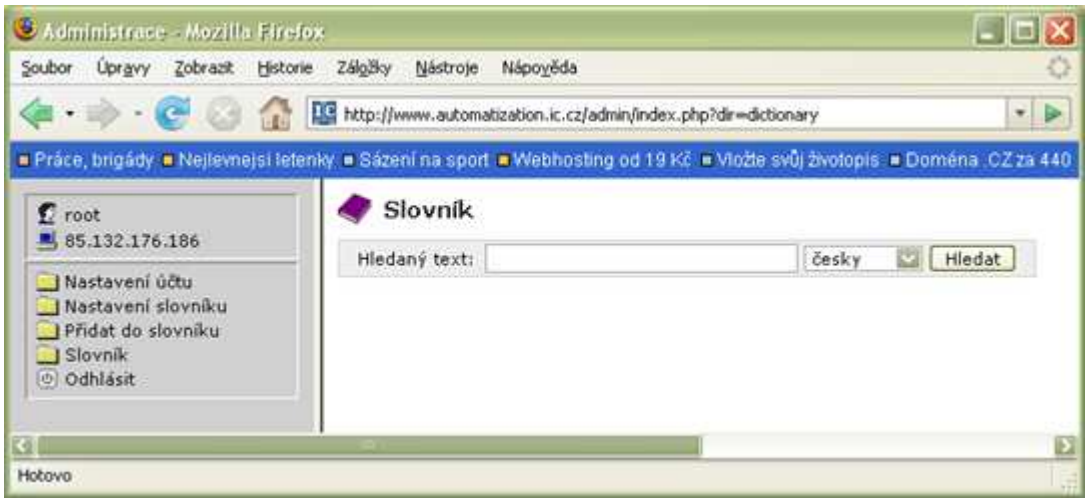


Obrázek 15: Administrace - Přidat do slovníku

Tato část administrace slouží k zadávání nových slov do slovníku. Položky „Český výraz“ a „Anglický výraz“ jsou povinné a proto při každém zadávaní nového slova do slovníku je požadován i jeho překlad. Nepovinná položka je „Popis“, do které se mohou psát podrobnosti nebo popis k zadávaným výrazům.

5.4.2.4 Slovník

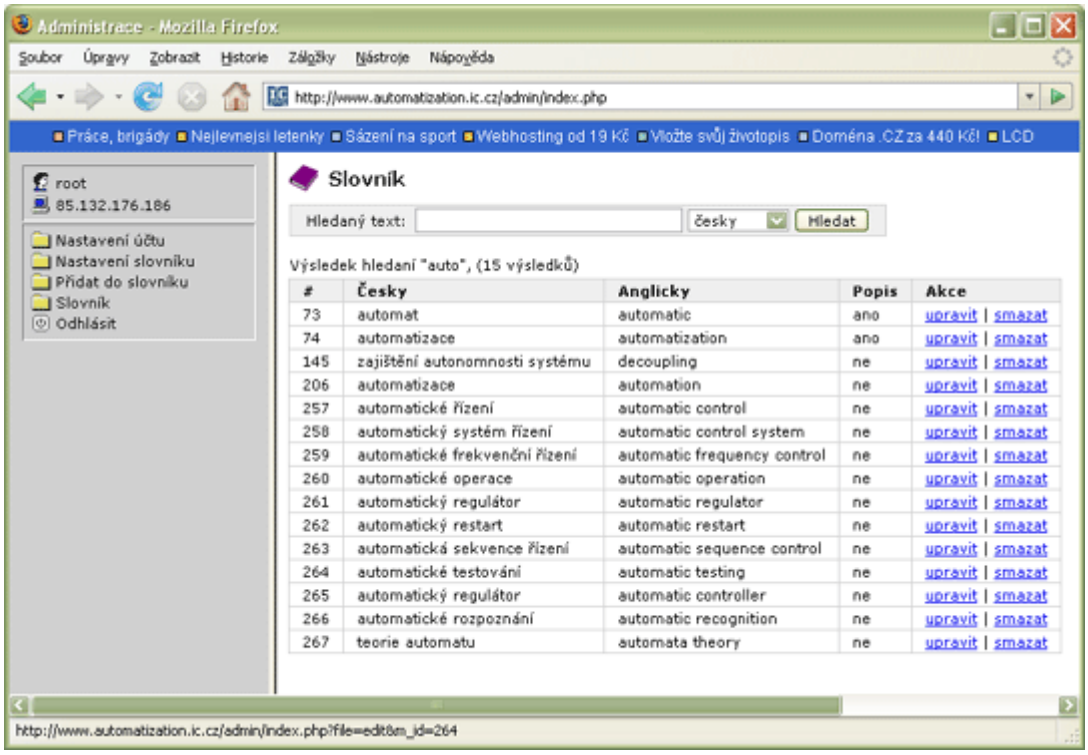
Tato část slovníku je asi nejdůležitější z celé administrace. Zde můžeme hledat slova podobně jako ve veřejné časti, jen s tím rozdílem, že zde můžeme slova upravovat nebo mazat. Po kliknutí na položku „Slovník“ se nám zobrazí následující stránka.



Obrázek 16: Administrace – Slovník

Když potřebujeme upravit nebo odstranit nějaké slovo ze slovníku, není nic jednoduššího než ho zadat do položky „Hledaný text“, nastavit zda chceme hledat v české či anglické části a pak kliknout na tlačítko „Hledat“. V administraci je počet vypsaných výsledků limitován na 50.

Pokud slovník obsahuje zadaný výraz, zobrazí se ve výsledku hledaní. Výsledky jsou seřazeny v tabulce, která obsahuje identifikační číslo, český překlad, anglický překlad, popis a akce. V každém řádku výpisu můžeme vidět zda obsahuje popis. Každý vypsaný záznam lze editovat nebo smazat.



Obrázek 17: Administrace – příklad vyhledání výrazu „auto“

Pokud chceme některý výraz upravit, stačí kliknout na odkaz „upravit“ v příslušném řádku. Zobrazí se nám stejný formulář jako pro přidávání slov do slovníku.

Pokud chceme některý výraz smazat, klikneme na položku smazat a záznam bude vymazán.

5.4.2.5 Odhlásit

K této položce asi nemá cenu mnoho psát, jak už název této položky napovídá, slouží k odhlášení z administrace. Po odhlášení se nám zobrazí přihlašovací formulář.

ZÁVĚR

Cílem bakalářské práce bylo vytvořit anglicko-český / česko-anglický internetový slovník a jeho naplnění pojmy z oblasti automatizace. Při vývoji slovníku byl kladen důraz na spolehlivost, funkčnost a jednoduchost.

V teoretické části této práce byly stručně popsány technologie, kterými byl slovník vytvořen. Slovník byl naprogramován pomocí serverového skriptovacího jazyka PHP, který byl optimalizován tak, aby slovník mohl byt nasazen na serverech, kde je nainstalovaná nejnovější verze PHP 5 tak i PHP 4. Pro ukládání dat slovníku byla použita relační databáze MySQL. Generovaný HTML kód slovníku ctí standardy webového konsorcia W3C. A pro naprogramovaní byl použit standard XHTML 1.0 Strict. Pomocí kaskádových stylů (CSS) byl navržen design, který podle použití slovníku lze přizpůsobit a tím změnit celý design veřejné části slovníku.

V úvodu praktické části je popsán software, který byl použit na vývoj slovníku, dále je podrobně popsána instalace slovníku a zprovoznění. V další části je popsána veřejná část slovníku, hledání a listování. Poslední část je věnována podrobnému popisu administrace a nastavení slovníku.

Při plnění databáze, jsem vycházel z keyword listu IFAC. Slovní byl navržen tak, aby jej bylo možné využít i na jiné účely, tudíž jej lze nasadit i jinde a s jiným zaměřením.

ZÁVĚR V ANGLIČTINĚ

The main aim of the Bachelor work was to create english-czech / czech-english internet dictionary and to fill it with notions from automation area. In dictionary progress it was put emphasis on reliability, functionality and simplicity.

In teoretical part of this work were shortly described technologies, which dictionary was created from. Dictionary was programmed via server scripting language PHP, which was optimised, in order to dictionary can be operated on server, where it is installed latest version PHP 5 or PHP 4. Relational database MySQL was used for saving dictionary data. Generated HTML code reveres standards of web consortium W3C. For programming it was used standard XHTML 1.0 Strict. The design was proposed with the assistance of cascade styles (CSS). Consequently, the visual aspect of whole public part of dictionary can be changed and adjusted according to its purpose.

The introduction of practical part is focused on description of software used for dictionary development. Next, it is described installation and launching of dictionary in detail. The further part takes notice of public part of dictionary, searching and listing. The last section pays attention to detailed description of administration and setting.

The database of terms is based on IFAC keyword list. Dictionary was designed so that it is possible to use it also for other purposes, i.e. it can be employed in different applications.

SEZNAM POUŽITÉ LITERATURY

[1] e-Automatizace: Anglicko-český slovník – Informační portál z oblasti automatizace [on-line]. [cit. 1. února 2007]. Dostupné z WWW: http://e-automatizace.cz/slovnik.asp

[2] IFAC keyword list of kontrol technology. [on-line]. [cit. 1. února 2007]. Dostupné z WWW: http://www1.elsevier.com/homepage/saf/ifac/site/IPV%20keywords.htm.

[3] Kosek, J.: Vše o WWW [on-line]. [cit. 1. února 2007]. Dostupné na WWW: http://www.kosek.cz

[4] Prokop, R., Matušů, R., Prokopová, Z.: Teorie automatického řízení – lineární spojité dynamické systémy. Skriptum FAI UTB ve Zlíně, 2006.

[5] Linux Software [on-line]. Dostupné na WWW: http://www.linuxsoft.cz

[6] Manuál PHP [on-line]. Dostupné na WWW: http://www.php.net

[7] Jiří Bráza: Učebnice základů jazyka PHP 4. GRADA 2002. ISBN 80-247-0442-0

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

W3C	World Wide Web Consortium
WWW	World Wide Web - celosvětová pavučina
HTML	HyperText Markup Language - značkovací jazyk pro hypertext
XHTML	E**x**tensible **H**yper**T**ext **M**arkup **L**anguage – rozšířitelný značkovací jazyk pro hypertext
PHP	Hypertext Preprocesor - hypertextový preprocesor
SQL	Structured Query Language - strukturovaný dotazovací jazyk
CSS	Cascading Style Sheets – jazyk pro formátování internetových stránek
IFAC	International Federation of Automatic Control
DOM	Dokument Object Model – objektový model dokumentu
DTD	Document Type Definition - Definice typu dokumentu
SGML	Standard Generalized Markup Language

SEZNAM OBRÁZKŮ

SEZNAM TABULEK