

WWW navigátor prostorami FAI

WWW navigator of FAI rooms

Ondřej Šubrt

Bakalářská práce
2007



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav aplikované informatiky
akademický rok: 2006/2007

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Ondřej ŠUBRT**
Studijní program: **B 3902 Inženýrská informatika**
Studijní obor: **Informační technologie**

Téma práce: **WWW navigátor prostorami FAI**

Zásady pro vypracování:

1. Provedte literární rešerši týkající se tvorby www stránek a s programem souvisejících jazyků, tj. HTML, SQL a PHP.
2. Uvedte popis software ve kterém vytvoříte www navigátor.
3. Vytvořte vlastní www navigátor vybranými prostorami FAI v areálu na Jižních Svazích.
4. Uvedte popis ovládání vámi vytvořeného navigátoru.
5. Umístěte navigátor na samostatné CD jako přílohu práce.

Rozsah práce:

Rozsah příloh:

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

Seznam odborné literatury:

1.Castagnetto J. a kol (2004). Programujeme PHP. Computer Press Brno 2004, ISBN 80-7226-310-2.

2.Staniček P. a kol. (2006). CSS - hotová řešení. Computer Press, ISBN 80-251-1031-1.

3.Dellwing I. (2002). Příručka tvůrce webu HTML 4. Grada, ISBN 80-247-0297-5.

4.Schlossnagle G. (2004). Pokročilé programování v PHP5, Zoner Press, ISBN 80-86815-14-5.

5.Bulger, B., Greenspan, J., Wall, D. (2004): MySQL/PHP Databases Applications. Willey Publishing, USA, ISBN 0-7645-4963-4.

Vedoucí bakalářské práce: **Ing. Karel Perůtka**
Ústav řízení procesů

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

Termín odevzdání bakalářské práce: **24. května 2007**

Ve Zlíně dne 13. února 2007


prof. Ing. Vladimír Vašek, CSc.
děkan




doc. Ing. Ivan Zelinka, Ph.D.
ředitel ústavu

ABSTRAKT

Hlavní částí této bakalářské práce je jednoduchý WWW navigátor prostorami Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně. Klade si za cíl sloužit jako výpomoc při orientaci v těchto prostorech nově příchozím studentům a ostatním návštěvníkům, případně k získávání informací o jednotlivých místnostech areálu U5 na Jižních Svazích. Text práce tvoří dvě části. V první části je stručný popis HTML, PHP a SQL jazyků spolu se stručným popisem software Macromedia Dreamweaver 8. Druhou částí je stručný popis této aplikace a dokumentace jeho ovládání.

Klíčová slova: Internet, PHP, HTML, CSS, SQL, Macromedia, Dreamweaver, Prohlížeč

ABSTRACT

The WWW navigator of the rooms of Faculty of Applied Informatics, Tomas Bata University in Zlin, is the main part of the bachelor thesis. This project is intended to help new students or visitors easily find way to the room in the university campus, which they want to see. Moreover, it enables to get short info about each room of U5 campus in the database. The campus is located at Jizni Svahy. The text part of the thesis is divided in two main subparts. First one provides the short introduction into HTML, PHP, and SQL languages together with the short description of Macromedia Dreamweaver 8 software. Second main subpart consists of short description of the newly created application together with its documentation.

Key words: Internet, PHP, HTML, CSS, SQL, Macromedia Dreamweaver, Browser

Poděkování patří zejména vedoucímu této bakalářské práce Ing. Karlu Perůtkovi za podporu a odbornou pomoc a jeho odborné rady při vytváření tohoto projektu. Dále bych chtěl poděkovat Ing. Tomáši Dulíkovi za prohloubení mých znalostí ve tvorbě www stránek. A v neposlední řadě děkuji Pavlu Polínkovi ml. za odbornou výpomoc při fotografování prostorů Fakulty aplikované informatiky.

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, 20. 05. 2007

.....
Ondřej Šubrt

OBSAH

ÚVOD	7
I TEORETICKÁ ČÁST	9
1 HTML (HYPERTEXT MARKUP LANGUAGE)	10
1.1 PROSTŘEDKY PRO PSANÍ (TVORBU) STRÁNEK.....	10
1.2 PROHLÍZEČ (BROWSER).....	10
1.3 ZÁKLADNÍ KOSTRA DOKUMENTU	11
1.4 ELEMENTY A ZNAČKY	12
1.5 ODKAZY (HYPERLINKY).....	18
1.6 SEZNAMY.....	18
1.7 OBRÁZKY.....	19
2 PHP PERSONAL HOME PAGE ANEB PHP: HYPERTEXT PREPROCESSOR	23
3 SQL STRUCTURED QUERY LANGUAGE	34
4 MACROMEDIA DREAMWEAVER 8	45
II PRAKTICKÁ ČÁST	58
5 WWW NAVIGÁTOR PROSTORAMI FAI	59
5.1 <i>ÚVOD</i>	59
5.2 <i>DATABÁZE</i>	59
5.3 <i>PHP A HTML</i>	61
5.4 <i>POPIS JEDNOTLIVÝCH ČÁSTÍ APLIKACE:</i>	62
ZÁVĚR	76
ZÁVĚR V ANGLIČTINĚ	77
SEZNAM POUŽITÉ LITERATURY	78
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	79
SEZNAM OBRÁZKŮ	80
SEZNAM PŘÍLOH	82

ÚVOD

Internet, který je znám dnes, se zrodil počátkem 90. let. Jeho historie je však mnohem složitější. První předchůdce Internetu byl vytvořen v roce 1969 institucí Advanced Research Project Agency (ARPA) pod záštitou Ministerstva obrany USA. Síť byla nazvána ARPANET. Zpočátku byla tvořena pouhými čtyřmi počítači. V roce 1972 k ní bylo připojeno 50 výzkumných a vojenských center. Později byla rozdělena na dvě sítě: Arpanet a Milnet (armádní síť). V roce 1981 přibyla síť Bitnet (využita k propojení amerických vysokých a středních škol). Velkým problémem však byla komunikace mnoha různých platformách. Proto probíhal v této oblasti intenzivní výzkum a jeho výsledkem (v r. 1983) byl protokol TCP/IP (Transmission Control Protocol / Internet Protocol), který je s menšími úpravami používán dodnes. Ačkoliv v polovině 80. let existovalo několik sítí, stále ještě nebyly předmětem zájmu veřejnosti, protože nebyly volně přístupné. V roce 1986 byla vytvořena síť NSFNET (National Science Foundation Network). Původně byla určená pro propojení pěti superpočítačů. Později se však toto řešení ukázalo natolik výhodné, že v roce 1990 byla síť Arpanet zrušena a nahrazena právě sítí NSFNET. V roce 1991 nad ní byla vytvořena nová síť NREN (National Research and Education Network).

Vznik dnešního Internetu je datován do 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 "síť okolo světa"). Obrovskou výhodou Internetu je, že při prohlížení WWW stránek se není třeba starat o to, zda informace jsou na počítači v České republice, Kanadě, nebo třeba v Austrálii.

V roce 1994 tedy slaví Internet 25. výročí. Roku 1995 se služba WWW dostává na první místo v počtu přenesených dat. Objevuje se první oficiální špionáž, namířená proti nelegálním výrobcům mobilních telefonů a dalšího elektronického zařízení, vedená americkou tajnou službou prostřednictvím Internetu. Rok 1996 přináší První veletrh Internetové technologie. V současné době se Internet u nás stává stále více součástí běžného života. A to hlavně prostřednictvím elektronické pošty. Dnes je již zcela běžnou záležitostí uvádět na své vizitce též adresu elektronické pošty, vlastní stránky (homepage) nebo společností, které přístup k Internetu nabízejí.

Tato bakalářská práce popisuje tvorbu dynamických internetových stránek s podporou databázových prostředků. Jejich hlavním cílem je pomoci nově příchozím studentům se alespoň částečně zorientovat v areálu U5 Univerzity Tomáše Bati ve Zlíně na Jižních Svazích. Hlavním výstupem práce je interaktivní webová aplikace sloužící jako navigátor prostorami výše zmíněného areálu.

Tato práce se dělí do dvou částí, teoretické a praktické. V teoretické části se stručně pojednává o HTML, PHP a SQL jazycích, protože tyto byly použity pro realizaci praktické části. Poslední kapitolou teoretické části práce je popis software Macromedia Dreamweaver 8, jelikož v něm byl projekt vytvářen. V druhé části práce je popis vytvořeného WWW navigátoru prostorami FAI pracovně označeného jako CLASS HUNTER.

I. TEORETICKÁ ČÁST

1 HTML (HYPERTEXT MARKUP LANGUAGE)

1.1 Prostředky pro psaní (tvorbu) stránek

Prostředky pro tvorbu www stránek v jazyce označovaném jako HTML lze obecně rozdělit následujícím způsobem:

- textový editor (např. Notepad)
- značkovací (HTML) editor
- textový procesor, který umí uložit formát HTML (např. Microsoft Word2000)
- speciální WYSIWYG prostředek (např. Microsoft FrontPage)

Výše uvedené dělení je pouze orientační. Jeho volba závisí na konkrétních znalostech uživatele, respektive zda je úplný začátečník nebo pokročilý uživatel. V případě, že se jedná o začátečníka, je velmi pravděpodobné, že použije nějaký WYSIWYG editor nebo program označovaný jako textový procesor.

1.2 Prohlížeč (browser)

Soubor zdrojového kódu v jazyce HTML představující konkrétní stránku je tvořen jednotlivými příkazy a bývá uložen počítači označovaném jako server. Aby tento soubor byl zobrazen v uživatelsky příjemné podobě na počítači uživatele, tj. klienta, je k tomu zapotřebí speciální software, který označujeme jako prohlížeč, anglicky browser. Nejpoužívanější prohlížeče jsou následující

- Microsoft Internet Explorer
- Mozilla Firefox
- Netscape Communicator

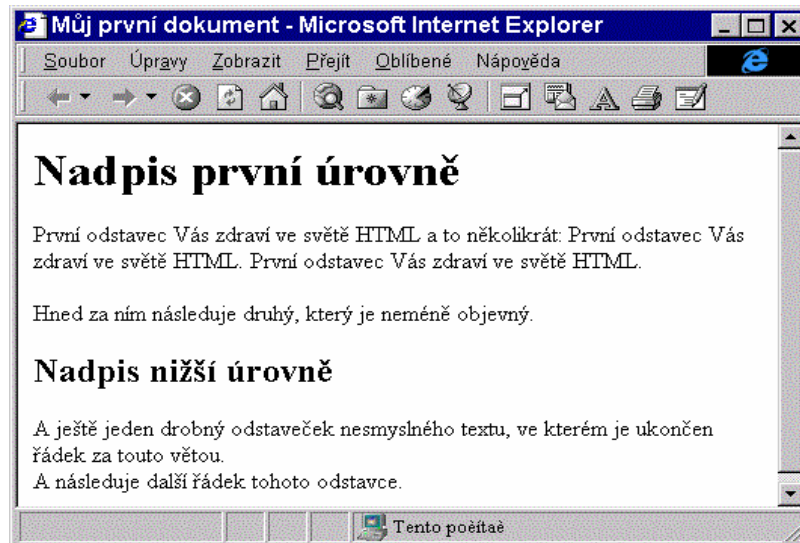
1.3 Základní kostra dokumentu

Vytváření webových stránek má svá pevně daná pravidla. Základní kostru HTML dokumentu tvoří

```
<HTML>
<HEAD>
<TITLE>Titulek</TITLE>
    ... ostatní prvky hlavičky (záhlaví) ...
    (Obecné informace o dokumentu, skripty)
</HEAD>
<BODY>
    ... tělo dokumentu...
    (vlastní viditelná část)
</BODY>
</HTML>
```

Příklad zápisu:

```
<HTML>
<HEAD>
<TITLE>Můj první dokument</TITLE>
</HEAD>
<BODY>
<H1>Nadpis první úrovně</H1>
<P>První odstavec Vás zdraví ve světě HTML a to několikrát: První odstavec Vás zdraví ve světě HTML. První odstavec Vás zdraví ve světě HTML.</P>
<P>
Hned za ním následuje druhý, který je neméně objevný.</P>
<H2>Nadpis nižší úrovně</H2>
<P>A ještě jeden drobný odstavček nesmyslného textu, ve kterém je ukončen řádek za touto větou.<BR>
A následuje další řádek tohoto odstavce.
</P>
</BODY>
</HTML>
```



Obrázek 1 - Reprezentace HTML dokumentu v prohlížeči

1.4 Elementy a značky

Kromě základní kostry dokumentu je třeba se zmínit o tzv. elementech ze kterých je WWW stránka složena.

Elementy - části dokumentu vyznačené *značkami (tagy)*. Lze je chápat jako objekty, obsahující řadu *vlastností (atributů)*. Definují podobu a obsah informace zobrazené prohlížečem.

Zápis značky:

- Název značky je ohraničen znaky < >
- Nezáleží na velikosti písmen
- Většina značek tvoří pár (kontejner): <značka> </značka>
- Značky lze vnořovat
- Značka má atributy (vlastnosti) a události

Příklad zápisu:

Element (prvek): <ZNAČKA> OBSAH </ZNAČKA>

Elementy a jejich vlastnosti:

- **Inline elementy** - jsou součástí textu na stránce, nemají okolo sebe zalomení (**STRONG, SPAN, B**)
- **Blokové elementy** - před i za nimi je zalomená řádka (např. **H1, P, DIV**)
- **Nahrazované elementy** - jsou nahrazené nějakým obsahem, pro jejich zařazení do stránky jsou důležité jejich rozměry (např. **IMG**)
- **Odkazy** - (**A**) – aktivní element, reagující automaticky na událost kliknutí myši – požadavek na zobrazení jiné stránky

Vlastnosti (atributy) určují chování (vzhled) jednotlivých elementů.

Příklad zápisu: `atribut = "hodnota"` nebo `atribut = ' hodnota'`

Elementy vyznačující strukturu dokumentu

- **HTML** - vymezení dokumentu
- **HEAD** - hlavička dokumentu
- **TITLE** - titulek dokumentu
- **BODY** - tělo dokumentu

Komentáře (poznámky) se zapisují ve tvaru

Příklad zápisu: `<!-- komentář -->`

Komentáře se používají k zajištění snadné orientace ve zdrojovém textu, příp. k zamezení interpretace části zdrojového textu stránky.

Prostý text

Mezery a konce řádků se interpretují jako jediná mezera. Text může obsahovat také speciální znaky, které jsou do zdrojového textu zapisují pomocí zástupných sekvencí znaků.

Speciální znaky – symboly:

číselná entita	znaková entita	znak	popis
<code>&#60;</code>	<code>&lt;</code>	<	menší než
<code>&#62;</code>	<code>&gt;</code>	>	větší než
<code>&nbsp;</code>			pevná mezera
<code>&#38;</code>	<code>&amp;</code>	&	ampersand
<code>&#34;</code>	<code>&quot;</code>	“	uvozovky
<code>&#176;</code>	<code>&deg;</code>	°	stupeň

Příklad zápisu:

```
<P align="center">Je-li <B><I>a &lt; b </i></B>
<BR>
```

potom

```
<B><I>b = 1&nbsp;000&nbsp;000 </i></B>.
</P>
```

Je-li **a < b**,

potom **b = 1 000 000**

Značky pro formátování textu a jejich atributy

HTML je značkovací jazyk. Obsahuje značky pro formátování textu. Můžeme je rozdělit na značky párové a značky nepárové. Párovými značkami jsou

např. `<HTML></HTML>` a jako příklad nepárové `<hr>`, `
`. Pro představu je zde uvedeno několik párových a nepárových značek s vysvětlivkou, příkladem použití a jejich atributy, které ovlivňují vzhled a polohu formátovaného textu stránky.

<P> **odstavec** - text uvnitř značek je zalomen a je před i za odstavec vložena mezera výšky řádku. Jeho jediným atributem je **align**.

align - je způsob zarovnání odstavce (**left**, **center**, **right**)

<H1>,<H4> **nadpis** - text uvnitř značek je zalomen a zvýrazněn. Značka má 6 úrovní.

Formátování je možno ještě upravit pomocí atributu **align**.

**
** **zalomení řádku** – v místě této značky je text odřádkován. Jeho jediným atributem je **clear**.

clear - např.- vynechání místa až skončí obrázky (**left**, **right**, **all**, **none**)

<HR> **vodorovná čára** - zalomí text a vloží vodorovnou čáru. Pomocí atributů je u tohoto tagu možné měnit zarovnání, tloušťku čáry, šířku a barvu.

align - způsob zarovnání (**left**, **center**, **right**)

size - tloušťka čáry v pixelech

width - šířka v pixelech nebo v procentech

color - barva čáry



Obrázek 2 - Formátování textu (nadpisy)

- <PRE>** **předformátovaný text** – využívá se pokud je třeba do stránky vložit předformátovaný zdrojový text například v jiném jazyce.
- <DIV>** **logický blok textu** - pro vyznačení související části textu se společným formátováním. Tag **<div>** nemá vlastní zobrazení.
- ** **fragment textu** - vymezuje části textu se společnými vlastnostmi, lze použít uvnitř odstavců a nadpisů a stejně jako v předchozím případě nemá vlastní zobrazení.

Písmo

Vzhled písma na webových stránkách určují styly, a to styly logické a fyzické. Logické styly určují význam textu a jeho zvýraznění. Prostředky ke zvýraznění volí prohlížeč. Fyzické styly ovlivňují vzhled textu a prostředky pro zvýraznění volí autor.

Logické styly písma:

	základní zvýraznění (obvykle kurzívou)
	silné zvýraznění (obvykle tučně)
<BIG>	velké písmo
<SMALL>	malé písmo

Fyzické styly písma:

	tučné
<I>	kurzíva
<U>	podtržené
<TT>	s pevnou šířkou znaku
<SUB>	dolní index
<SUP>	horní index
	definice velikosti, barvy a fontu písma, obsahuje tyto atributy:
size	velikost 1 až 7
color	barva
face	typ písma

Příklad zápisu:

```
<BODY background="URL" bgColor = "red" text = "blue">  
<FONT size = "velikost" color = "barva" align = "center">text</FONT>  
</BODY>
```

1.5 Odkazy (hyperlinky)

Odkaz je zvýrazněná část stránky, která odkazuje buď na jiný dokument, nebo na jiné místo v tomtéž dokumentu. Odkazem může být text, který je většinou zobrazen jinou barvou a je podtržen, nebo může jako odkaz sloužit také obrázek. Při vytváření odkazu je třeba určit kam bude odkaz ukazovat (URL zdroje), a dále text, který bude odkaz označovat ve stránce.

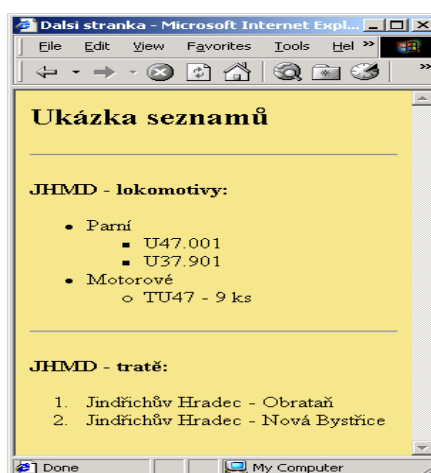
Nejdůležitějším vlastností tagu je atribut **href**. Ten udává kam bude odkaz ukazovat. V rámci jednoho dokumentu se lze pohybovat pomocí takzvané kotvy, která ukazuje na návěští umístěné v tomtéž dokumentu.

Příklad zápisu:

```
<A href="http://www.fce.vutbr.cz">FAST</A>  
<A href="#Kapitola2">Do kapitoly2</A>  
<A name="Kapitola2">Zde začíná 2.kapitola</A>
```

1.6 Seznamy

Na stránku je též možné umístit odrážkové seznamy, k čemuž slouží tag ****, jehož atribut **type** určuje vzhled odrážky (**disc**, **square**, **circle**), tag **** vytváří číslovaný seznam a tag ****, který určuje umístění odrážky. Seznamy je možné do sebe libovolně vnořovat.



Obrázek 3 - Seznamy UL, OL a LI

1.7 Obrázky

Obrázky se do stránky umisťují pomocí tagu ****. Jeho nejdůležitějším atributem je **src**, což je vlastnost určující cestu k obrázku. Dalšími atributy jsou např. **alt** popisující obrázek textem, **align** a další uvedené níže.

Příklad zápisu:

```
<IMG src = "URL obrázku" další nepovinné atributy >
```

Vlastnosti obrázků:

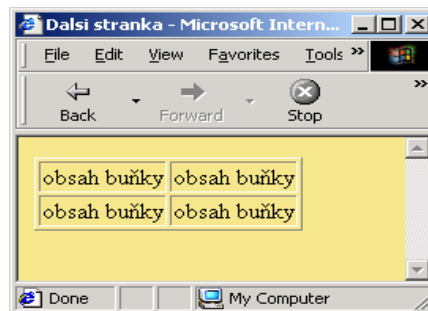
- **src** URL obrázku, většinou relativní
- **alt** popis obrázku; není-li, použije se název souboru
- **align** způsob zarovnání
 - **top**
 - **left**
 - **middle**
 - **right**
- **bottom**
- **width** šířka obrázku
- **height** a výška obrázku
- **border** šířka rámečku kolem obrázku (implicitně 0, u odkazů 1)
- **hspace** mezera vedle obrázku
- **vspace** mezera nad a pod obrázkem

1.8 Tabulky

Tabulka je párový tag, obsahuje nadpis a řádky do kterých se zapisují jednotlivé buňky, obsahuje spoustu atributů.

Základní elementy pro tvorbu tabulek:

```
<TABLE>
<TR>
<TD>obsah buňky </TD>
<TD>obsah buňky </TD>
</TR>
<TR>
<TD>obsah buňky </TD>
<TD>obsah buňky </TD>
</TR>
</TABLE>
```



Obrázek 4 - Ukázka tabulky

1.9 Formuláře

Formuláře slouží k získání informací od uživatele nebo k předání zpracovaných informací uživateli. Jsou buďto odesílány ke zpracování na server nebo zpracovávány skripty prováděnými přímo prohlížečem na straně klienta.

Ukázka zápisu:

```
<FORM>
  prvky formuláře
</FORM>
```

Atributy prvku FORM:

```
name = "jméno"
action = "url obslužného programu"
method = "GET/POST"
```

Další prvky formuláře:

```
<TEXTAREA>text </TEXTAREA>
```

atributy: **name**, **rows**, **cols**, **readonly**

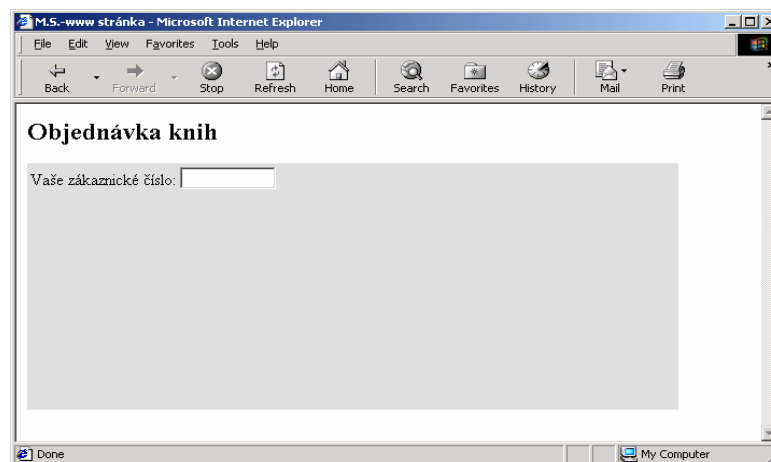
```
<SELECT> atributy: name, size, multiple
```

```
<OPTION> položka</OPTION> atributy: selected, value
```

```
</SELECT>
```

Příklad zápisu:

```
<H2>Objednávka knih </H2>  
<TABLE bgColor="#DDDDDD" width="90%" height="80%">  
<TR><TD valign="top">  
<FORM action="form.asp" method="POST" name="F">  
Vaše zákaznické číslo:  
<INPUT type="text" name="zc" size="9">  
</FORM>  
</TD></TR></TABLE>
```



Obrázek 5 - HTML formuláře

1.10 Kaskádové styly - CSS

Kaskádové styly jsou obecným rozšířením HTML. Pomocí atributů je možno upravit vzhled budoucí stránky nakonfigurovat s přesností na pixel. Stránka se skládá z vlastního zdrojového textu a definice stylů.

Příklad zápisu:

Definice stylů

```
<PRVEK style=" ... " >      Příklad: <P style="color:blue">
<STYLE type="text/css">    obvykle v části HEAD
</STYLE>                   dokumentu
<LINK rel="StyleSheet" href="soubor.css" type="text/css">
```

Styly jsou definované v samostatném textovém souboru s příponou *css*.

Deklarace stylu

```
.Selektor {
vlastnost1:hodnota;
    vlastnost2:hodnota;
    ...
    vlastnost3:hodnota
}
```

2 PHP PERSONAL HOME PAGE ANEB PHP: HYPERTEXT PREPROCESSOR

2.1 Úvod a prostředky pro psaní PHP skriptů

PHP je programovací jazyk umožňující vytváření dynamických webových stránek. Byl vytvořen v roce 1994 Rasmem Lerdorfem. Je to skriptovací jazyk, který se provádí na straně serveru. Je na serveru závislý, protože na něm běží jeho interpreter, který skripty provádí. Samotné PHP skripty se zapisují přímo do kódu HTML stránky (nejčastěji se ukládají příponou **.php*).

PHP běží na webserveru proto je třeba mít webserver nainstalovaný na počítači (tzv. localhost např. Apache, Xampp nebo jiný), nebo uploadovat své stránky přímo na reálný server. Skripty PHP lze psát v mnoha tomu určených editorech např. TSW webcoder, Macromedia Dreamweaver, NuSphere editor, ale také přímo v Poznámkovém bloku. Ovšem až na notepad jsou všechny uvedené editory licencované. Na internetu je však možno nalézt mnoho „freeware“ editorů, které však nejsou tak propracované.

2.2 Kostra PHP skriptu

Soubor je tvořen kousky HTML a PHP skriptů. HTML kód se posílá přímo browseru, část v PHP módu se interpretuje na straně serveru a do uživatelova prohlížeče se posílá pouze výstup skriptu, to je to, co skript vypisuje na standardní výstup (např. příkazem **echo**).

Začátek PHP skriptu začíná značkou pro opuštění HTML a končí značkou pro návrat do HTML:

```
<?php ?> nebo <? ?> nebo <script language="php"> </script>
```

Příklad zápisu:

```
<?php  
echo ("toto je výpis na standardní výstup");  
?>
```

2.3 Příkazy a proměnné

Příkazy PHP musí být od sebe odděleny. Jsou odděleny pomocí středníků nebo tagů. Přímo do skriptů lze také psát poznámky. Text mezi `//` a koncem aktuálního řádku je považován za poznámku, stejně tak jako text mezi `/*` a `*/`.

Při psaní proměnných je nutné rozlišovat malá a velká písmena, ale při psaní příkazů lze používat malá i velká písmena libovolně. Proměnné nemusí být deklarovány předem a k jejich zápisu se používá znak `$` a za ním následuje název proměnné. Proměnné slouží k uchování hodnot pouze v rámci skriptu. V proměnné lze uchovat hodnotu jakéhokoli typu. PHP si samo určí datový typ hodnoty.

Příklad zápisu:

```
$a=1; $b=1.0; $c= $a+$b;
```

Globální proměnné

Od verze 4.1.0 poskytuje PHP sadu předdefinovaných polí obsahujících proměnné WWW serveru (pokud to jde), prostředí a uživatelského vstupu. Tato nová pole mají tu zvláštnost, že jsou automaticky globální – tedy např. automaticky dostupné v každém kontextu. Z tohoto důvodu jsou často známa jako „autoglobální“ nebo „superglobální“ (v PHP neexistuje mechanismus pro uživatelskou definici superglobálních proměnných). Superglobální proměnné jsou vypsány níže.

`$GLOBALS` - obsahuje odkaz na každou proměnnou, která je momentálně dostupná v globálním kontextu skriptu, klíči tohoto pole jsou názvy globálních proměnných

`$_SERVER` - proměnné nastavované WWW serveru nebo jinak přímo spjaté s prováděcím prostředím aktuálního skriptu

`$_GET` - proměnné poskytované skriptu přes HTTP GET

`$_POST` - proměnné poskytované skriptu přes HTTP POST

`$_COOKIE` - proměnné poskytované skriptu přes HTTP cookies

`$_FILES` - proměnné poskytované skriptu přes HTTP post uploady souborů

`$_ENV` - proměnné poskytované skriptu z prostředí

\$_REQUEST - proměnné poskytované skriptu přes libovolný vstupní mechanismus a kterým proto nelze důvěřovat

\$_SESSION - proměnné, které jsou momentálně registrovány v aktuální relaci skriptu

Příkazy, podmínky a cykly

Příkazy se vždy musí nacházet uvnitř PHP značek. Jsou oddělené středníkem, resp. každý příkaz PHP je středníkem ukončen. Jde prakticky o stejnou strukturu jako v jazyce C.

Podmíněné příkazy:

if – slouží k podmíněnému provedení příkazu, pokud je splněna určitá podmínka, ta se musí zadat jako výraz, který vrací logickou hodnotu.

```
if ($y!=0) $vysledek=$x/$y;
```

if-else – příkaz uvedený za **else** se provede v případě nesplnění podmínky.

```
if ($ x > $ y) echo „Správně“;  
else echo „Špatně“;
```

if-elseif-else – příkaz uvedený za **elseif** se provede v případě, že není splněna podmínka pro **if** a zároveň je splněna podmínka za **elseif**.

```
if($x<$y) echo („x je menší než y“);  
elseif($x==$y) echo „x rovná se y“;  
else echo „x je větší než y“;
```

switch – na základě hodnoty jednoho výrazu se provede jedna z několika větví skriptu. Nejprve je vyhodnocen výraz, poté jsou postupně procházeny hodnoty uvedené za klíčovým slovem **case**, dokud se nenalezne hodnota shodná s hodnotou výrazu, následně jsou vykonávány příkazy dokud nenarazí na **break** nebo konec příkazu **switch**.

Příklad zápisu:

```
switch ($i) {  
case 0: print "i se rovná 0"; break;  
case 1: print "i se rovná 1"; break;  
case 2: print "i se rovná 2"; break;  
default: print "i se rovná 2"; break;  
}
```

Cykly

while – provádí zadaný příkaz dokud platí určitá podmínka, patří mezi nejjednodušší příkazy.

```
$i = 0;  
while($i < 10){  
echo "$i<BR>";  
  $i++;  
}
```

do-while – je příkaz podobný **while**, avšak podmínka je až na konci cyklu, příkazy v tomto cyklu jsou vykonány aspoň jednou oproti cyklu **while**, kde nemusí být provedeny ani jednou.

```
$i = 0;  
do{  
echo "$i"; $i++;  
}while ($i < 10);
```

for – před začátkem provádění cyklu je vyhodnocen **výraz1**, poté **výraz2** – pokud je jeho výsledek druhého výrazu vyhodnocen jako **true**, provede se tělo cyklu, na konci se vyhodnotí **výraz3**. Tělo cyklu se provádí tak dlouho, dokud má **výraz2** hodnotu **true**.

```
for ($i = 1; $i <= 10; ++$i)  
echo („$i“);
```

Existují též příkazy, kterými lze ovlivnit co a kdy se v cyklu vykoná. Jsou to následující:

break – okamžité ukončení provádění cyklu

continue – okamžitě přeskočí příkazy ve zbývající části těla cyklu a začne provádět další iteraci

PHP dále obsahuje příkazy, které slouží k načtení externích skriptů.

require – slouží k načtení skriptu ze souboru

include – stejný jako **require**, rozdíl je v tom, že jeden příkaz lze volat několikrát, např. uvnitř cyklu.

2.4 Datové typy

PHP, stejně jako jakýkoli jiný programovací jazyk, dokáže zpracovávat nejrůznější data. Protože se ale s různými daty pracuje různě (např. při práci s reálným číslem se používají jiné funkce, než při zpracování textového řetězce), existují v PHP tzv. *datové typy*.

PHP rozlišuje celkem 8 jednoduchých typů: **boolean** (pro hodnoty PRAVDA - NEPRAVDA), **integer** (celá čísla), **float** (čísla s plovoucí desetinnou čárkou), **string** (textové řetězce), **array** (pole), **object**, **resource** a **NULL**.

PHP se od řady programovacích jazyků (např. Pascal nebo C) liší tím, že není třeba typ proměnné nijak nastavovat, nebo se o něj vůbec zajímat. PHP jej většinou určí samo z kontextu, ve kterém je daná proměnná použita (a z informací, které v ní jsou uloženy).

Boolean - slouží k uložení hodnot **true** nebo **false** (pravda nebo nepravda). **False** lze vyjádřit číslem 0 (případně prázdným řetězcem), **true** pak jakýmkoli jiným nenulovým číslem nebo neprázdným řetězcem.

Integer - celé číslo v rozsahu od -2 147 483 648 do 2 147 483 647

Double - desetinné číslo v rozsahu od -1,7x10308 do 1,7x10308

String - hodnotou je znakový řetězec

Array - proměnná, která obsahuje více hodnot. Hodnota se vyvolává pomocí indexu prvku pole

Index - indexy se zapisují do hranatých závorek za název proměnné

Object – hodnotou tohoto datového typu je objekt

2.5 Řídící struktury (SMARTY)

Stejně jako v samotné aplikaci, i při prezentaci údajů a dat je nutné často využívat podmínek a cyklů. Zjednodušená podoba těchto řídicích příkazů je k dispozici i v systému. Při vytváření šablon lze použít podmínky i cykly typu "**for**" a "**foreach**". Základním formátem alternativní syntaxe je záměna otvírací závorky za dvojtečku (**:**) a uzavírací závorky za **endif;**, **endwhile;**, **endfor;**, **endforeach;**, resp. **endswitch;**.

Příklad zápisu:

```
<?php if ($a == 5): ?> A se rovná 5 <?php endif; ?>
```

2.6 Funkce

Funkce jsou v podstatě krátké části skriptů, které je možné opakovaně používat, do funkce může být vložen jakýkoli platný PHP kód.

Klíčové slovo **function** ukazuje, že jde o deklaraci funkce. Je výhodnější psát název funkce bez diakritiky. Předem definované funkce nikdy nemají český název, pokud náhodou funkce bude mít shodný název s některou již definovanou, skript by mohl fungovat chybně (**return**, **close**, **open**). Kulaté závorky „**()**“ definují, zda se jedná o funkci s parametrem, nebo bez (prázdné závorky-bez parametru). Složené závorky „**{ }**“ ohraničují deklaraci a obsah funkce.

Příklad zápisu:

```
function vypocet()
{
    $GLOBALS["x"] = 10;
    $y = 20;
}
$x = 100;
$y = 200;
vypocet();
$vysledek = $x + $y;
echo $vysledek;
```

Funkce bez parametru

Nemá uvnitř závorek žádný argument (proměnnou):

```
function vypis(){
echo("ahoj");
}
```

Funkce s parametrem

```
function vypis($jmeno){
echo($jmeno);
}
vypis(dobrou noc);
```

Funkce vracející hodnotu

```
function vrat($hodnota){
return $hodnota*2;
}
echo(vrat(20)+"<br>");
```

2.7 Objektově orientované programování v PHP

Objektově orientované programování (OOP) lze chápat jako snahu psát software z principu reálného světa, tedy mapování skutečných objektů z reálného světa do programového kódu a poskytuje abstrakci pro překlenutí mezery mezi reálnými objekty a kódem.

Klíčové vlastností OOP:

- **Zapouzdření (encapsulation)**
- **Dědičnost (inheritance)**
- **Polymorfismus (polymorphism)**

Zapouzdření

Pod tímto pojmem si lze představit zabalení vlastností a operací do jednoho balíčku, který se nazývá „objekt“. To znamená, že k příslušným vlastnostem a operacím se programátor dostane pouze pomocí rozhraní daného objektu. Pokud je to možné, jediný přístup k vlastnostem objektu by měl být přes operace (tzn. navenek jsou vidět jenom jeho operace, nikoli jeho vlastnosti).

Dědičnost

Tato vlastnost je využita tehdy, když jsou zpracovávány dva a více objektů, které mají určité společné vlastnosti nebo operace a je zbytečné, aby se jejich implementace opakovala. V tomto případě se vytvoří tzv. „rodičovská třída“, která bude obsahovat společné rysy pro její „potomky“. Potomci budou dědit veškeré vlastnosti a operace své rodičovské třídy a budou implementovat nové, více specifické vlastnosti a operace pro daný typ třídy. Jinak řečeno dědičnost je schopnost spojit různé typy pod jeden společný. Např. **Ženy** a **Muži** mají společné rozhraní **Člověk**.

V tomto případě není třeba deklarovat vlastnosti věk a výška ani operaci **zjistivěk()** v třídách **Muž** a **Žena**, protože dědí tyto vlastnosti a operaci od rodiče.

Polymorfismus

Polymorfismus neboli mnohotvárnost navazuje na dědičnost a je to vlastnost jak lze nabývat různých implementací na základě více tříd pomocí jediného názvu operace nebo atributu. Polymorfismem jsou tedy získány objekty o různých implementacích, které jsou sdružené pod jeden společný typ (mají společnou nadřazenou “rodičovskou” třídu).

Třída

Třída je speciální typ, může obsahovat data i funkce. Tvoří se pomocí příkazu **class**, kde se deklarují členské proměnné a členské funkce.

```
class < název třídy >
{
< členské proměnné >
< členské funkce >}

```

Členské proměnné

Členské proměnné jsou lokální proměnné, do kterých lze ukládat data a pracovat s nimi lze pomocí členských funkcí. Členská proměnná se většinou deklaruje na začátku třídy s klíčovým slovem **var**. Pokud je třeba deklarovat více proměnných, oddělují se čárkami.

Členské funkce

Členské funkce umožňují práci s členskými proměnnými, databázemi, soubory, atd. Za klíčové slovo **\$this** se po vytvoření instance automaticky doplní název instance třídy. Konstruktor je členská funkce, která má stejný název jako název třídy. Pomocí konstruktoru lze uvnitř třídy vykonat určitý kód ihned po vytvoření její instance. V PHP nelze použít více konstruktorů.

Instance

Pokud je nadefinovaná třída lze vytvářet její instance (objekty). Objekt je proměnná, jejímž typem je určitá třída. Pomocí této proměnné se pak přistupuje k datům objektu a k jeho funkcím. Nová instance třídy se vytváří se pomocí klíčového slova **new**.

```
$a = new Prvni();
```

Příklad zápisu:

```
class Kruznice {
var $x,$y,$polomer;
function Kruznice($x=0, $y=0, $polomer=1) {
$this->x=$x;
$this->y=$y;
$this->polomer=$polomer;
}
function obsah() { return 3.14*$this->polomer * $this->polomer;}
}

$k=new Kruznice();
$obsah=$k->obsah();
echo "Obsah k=$obsah<BR>";

$k1=new Kruznice(10,10,20);
$obsah=$k1->obsah();
echo "Obsah k1=$obsah<P>";
```

Reference (odkazy)

Reference je prostředek, jak v PHP přistupovat k témuž obsahu proměnné pod různými jmény, jsou to vlastně aliasy v tabulce symbolů. Je rozdíl mezi názvem proměnné a jejím obsahem, takže stejný obsah může mít různé názvy. Odkazy zajišťují, aby dvě proměnné odkazovaly na tentýž obsah. **\$a** i **\$b** ukazují na stejnou proměnnou a přesto to nejsou ukazatele:

```
$a =& $b
$bar =& new fooclass();
$foo =& find_var ($bar);
```

Nepoužití operátoru **&** by způsobilo zkopírování objektu. Použije-li se **\$this**, bude se pracovat s aktuální instancí třídy a přiřazení bez operátoru **&** zkopíruje instanci (např. objektu) a **\$this** bude pracovat s touto kopií.

```
function foo (&$var) { $var++; }
$a=5; foo ($a);
```


Vysvětlení

- nastaví do **\$a** hodnotu **6** to proto, že ve funkci **foo** proměnná **\$var** odkazuje tentýž obsah jako **\$a**.
- vrácení odkazem je užitečné, když je třeba použít funkci k nalezení proměnné, která by měla být odkazu přiřazena

```
function &find_var ($param)
{ ...nějaký kód... return $found_var; }
$foo =& find_var ($bar);
$foo->x = 2;
```

- v tomto příkladu by bylo funkcí **find_var** nastaveno vlastnictví téhož objektu, nikoli jeho kopie, jak by se stalo bez použití syntaxe bez odkazů
- narozdíl od předávání parametru, zde se musí **&** použít na obou místech k indikaci, že se vrací odkaz a nikoli kopie jako obvykle, a k indikaci přiřazení reference do **\$foo** namísto běžného přiřazení (hodnoty)
- odnastavení reference –přeruší se tak vazba mezi názvem proměnné a jejím obsahem
- to však neznamená, že by obsah byl zničen

```
$a = 1; $b =& $a; unset ($a);
```

- neodnastaví **\$b**, nýbrž pouze **\$a**

3 SQL STRUCTURED QUERY LANGUAGE

SQL patří do kategorie tzv. **deklarativních programovacích jazyků** - kód jazyka SQL není psán v žádném samostatném programu, ale je vkládán do jiného programovacího jazyka, který je již procedurální.

Se samotným jazykem SQL lze pracovat pouze v případě, že je pomocí terminálu připojen na SQL server a na příkazový řádek se zadávají přímo příkazy jazyka SQL.

3.1 Části SQL

SQL se skládá z několika částí:

- Jazyk **DDL** - Data Definition Language - jedná se o jazyk pro vytváření databázových schémat a katalogů
- Jazyk **SDL** - Storage Definition Language - definuje způsob ukládání tabulek
- Jazyk **VDL** - View Definition Language, určuje vytváření pohledů (pohled si lze představit jako virtuální tabulku složenou z různých jiných tabulek)
- Jazyk **DML** - Data Manipulation Language - obsahuje základní příkazy **INSERT**, **UPDATE**, **DELETE** a nejpoužívanější příkaz **SELECT**.

3.2 Návrh tabulky

Základem každé databáze jsou tabulky, které popisují nějakou entitu. Tabulka se skládá ze sloupců a polí, které se nazývají atributy a volí se takové vlastnosti, které nás o dané entitě zajímají.

Podle pravidel v MySQL se k pojmenování sloupců používají alfanumerické znaky a podtržítka.

Název sloupce	Příklad
id_cislo	128
jmeno	Jan
prijmeni	Jánský
firma	ABC
email	jansky@abc.cz
datum_registrace	2004-02-04

Obrázek 6 - Návrh tabulky

3.3 Datové typy sloupců

V MySQL existují 3 hlavní typy jednotlivých datových polí. Jsou to textové, číselné a časové datové typy. V rámci hlavních datových typů pak existuje řada specifikací :

Číselné datové typy

Název typu	Paměťový prostor (B)	Interval	Bez znaménka
TINYINT	1	-128 do 127	0 – 255
SMALLINT	2	-23768 do 32767	0 – 56535
MEDIUMINT	3	-8388608 do 8388607	0 – 16777215
INT	4	-2147483648 do 2147483647	0 – 4294967295
BIGINT	8	-9223372036854775808 do 922337203685477 5807	0 – 184467440737095 50615
FLOAT(M,D)	4	<u>Liší se podle použitých hodnot.</u>	-
DOUBLE(M,D)	8	<u>Liší se podle použitých hodnot.</u>	-
DECIMAL(M,D)	<u>Hodnota M + 2B</u>	<u>Liší se podle použitých hodnot.</u>	-

Obrázek 7 - číselné datové typy v SQL

Textové datové typy

Název datového typu	Maximální velikost	Přidělená paměť
CHAR(X)	255 B	X B
VARCHAR(X)	255 B	X+1 B
TINYTEXT	255 B	X+1 B
TINYBLOB	255 B	X+2 B
TEXT	65535 B	X+2 B
BLOB	65535 B	X+2 B
MEDIUMTEXT	1,6 MB	X+3 B
MEDIUMBLOB	1,6 MB	X+3 B
LONGTEXT	4,2 GB	X+4 B
LOB	4,2 GB	X+4 B

Obrázek 8 - Textové datové typy v SQL

Datum a čas

Typ	Velikost	Popis
DATE	3 bajty	datum ve formátu YYYY-MM-DD
DATETIME	8 bajtů	datum ve formátu YYYY-MM-DD HH:MM:SS
TIME	3 bajty	čas ve formátu HH:MM:SS
TIMESTAMP	4 bajty	ve formátu YYYYMMDDHHMMSS (hodnoty končí v roce 2037)
ENUM	1 nebo 2 bajty	výčet jenž označuje, že sloupec může nabýt jen jednu z povolených hodnot
SET	1 – 8 bajtů	výčet jenž označuje, že sloupec může nabýt více povolených hodnot

Obrázek 9 - datum a čas v SQL

3.4 Integritní omezení

Do tabulek většinou data vkládají koncoví uživatelé, proto lze na úrovni SQL serveru zajistit některé požadavky, které jsou na jednotlivé položky kladeny. Např.: Lze zajistit, aby některá z položek byla vždy vyplněna, což znamená, že server nedovolí uložit prázdný záznam do tabulky.

Možná integritní omezení:

NULL a **NOT NULL** – určují, zda je zadání hodnoty do příslušného pole povinné.

PRIMARY KEY – primární klíč je pole, jehož hodnota jedinečným způsobem identifikuje každý záznam tabulky.

UNIQUE – omezující podmínka jedinečnosti vynucuje integritu entity klíčem, který není primární. Zajišťuje, aby do sloupců s omezující podmínkou jedinečnosti nebyly vloženy duplicitní hodnoty.

UNSIGNED – označuje neznaménkový číselný typ.

3.5 Základy práce s klientem my SQL

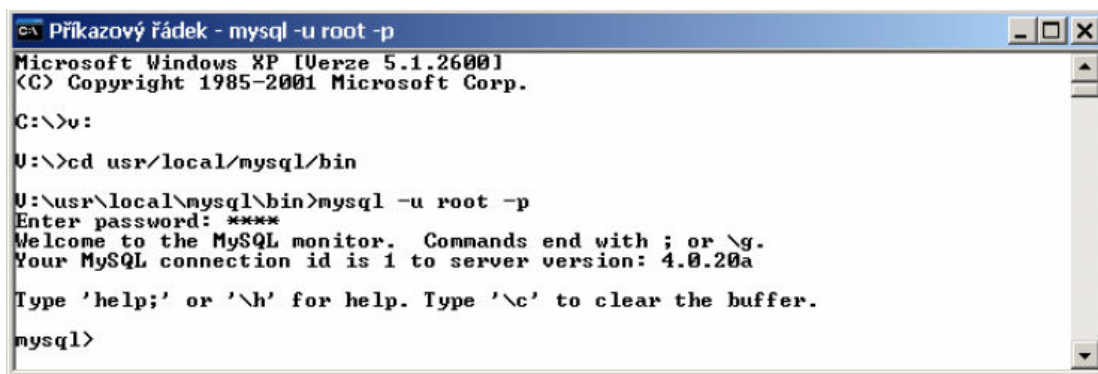
Jedním ze způsobů komunikace se serverem MySQL je klient MySQL. Klientskou aplikaci lze spustit z příkazového řádku příkazem **mysql**, který může mít několik argumentů. Takto lze komunikovat s MySQL serverem a upravovat data na něm. Všechny použité příkazy musí být ukončeny středníkem. Aby bylo možné se připojit k serveru a pracovat s daty na něm, je třeba, aby byl spuštěn démon MySQL.

Argumenty :

(-u) – uživatelské jméno

(-d) – heslo

(-h) – název hostitele



```
C:\> Příkazový řádek - mysql -u root -p
Microsoft Windows XP [Verze 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\> u:
U:\> cd usr/local/mysql/bin
U:\usr\local\mysql\bin> mysql -u root -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 4.0.20a
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

Obrázek 10 - Klient SQL

3.6 Tvorba databáze

Výpis seznamu existujících databází na serveru MySQL:

SHOW databases;

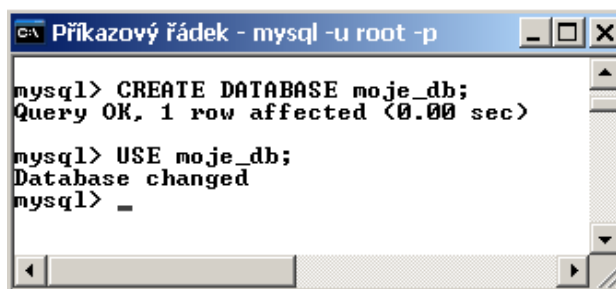
Vytvoření nové databáze

CREATE databases;

Výběr databáze, kterou chceme používat: - **moje_db**

USE test;

Vytvoření databáze



```
C:\> Příkazový řádek - mysql -u root -p
mysql> CREATE DATABASE moje_db;
Query OK, 1 row affected (0.00 sec)

mysql> USE moje_db;
Database changed
mysql> _
```

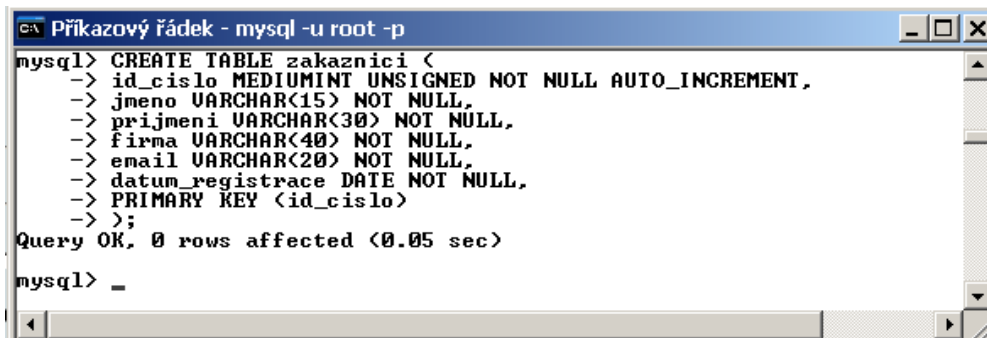
Obrázek 11 - Vytvoření databáze

3.7 Tvorba tabulek v SQL

K vytvoření tabulky je zapotřebí příkazu **CREATE**. Za něj se zapíše název tabulky oddělený mezerou. Do kulatých závorek se zapisují názvy sloupců, jejich typy a integritní omezení.

Příklad zápisu:

```
CREATE TABLE název_tabulky (  
jméno_sloupece1 TYP [integritní omezení],  
jméno_sloupece2 TYP [integritní omezení],  
...);
```



```
mysql> CREATE TABLE zakaznici (  
-> id_cislo MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT,  
-> jmeno VARCHAR(15) NOT NULL,  
-> prijmeni VARCHAR(30) NOT NULL,  
-> firma VARCHAR(40) NOT NULL,  
-> email VARCHAR(20) NOT NULL,  
-> datum_registrace DATE NOT NULL,  
-> PRIMARY KEY (id_cislo)  
-> );  
Query OK, 0 rows affected (0.05 sec)  
mysql> _
```

Obrázek 12 - Vytvoření tabulky v SQL

Ověření existence vytvořené tabulky:

```
SHOW TABLES;  
SHOW COLUMNS FROM název_tabulky;
```

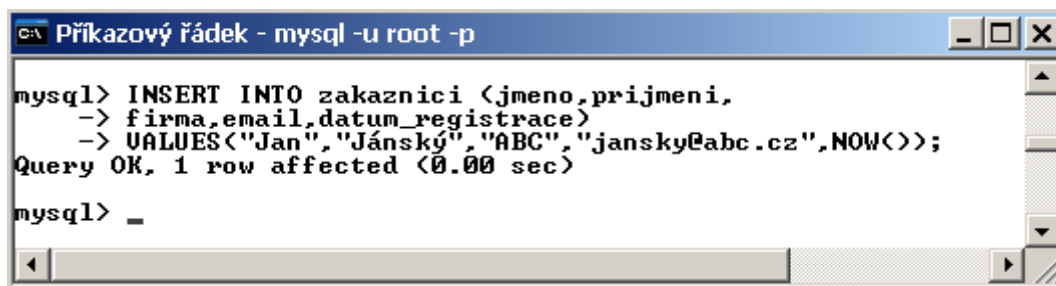
3.8 Vkládání záznamů

K účelu vkládání záznamů slouží příkaz **INSERT**. Tento příkaz má dvě varianty, které se liší v možnostech přidávání dat do tabulky. V prvním případě lze vybrat, které sloupce budou zaplněny novými hodnotami a v druhém případě se zapisují hodnoty všech sloupců postupně, což znamená, že počet vkládaných hodnot musí odpovídat počtu sloupců.

```
INSERT INTO název_tabulky (sloupec1, sloupec4, ...)  
VALUES (hodnota1, hodnota4, ...);
```

nebo

```
INSERT INTO tabulka VALUES(hodnota1, hodnota2, ...);
```



```

C:\> Příkladový řádek - mysql -u root -p

mysql> INSERT INTO zakaznici (jmeno,prijmeni,
-> firma,email,datum_registrace)
-> VALUES("Jan","Jánský","ABC","jansky@abc.cz",NOW());
Query OK, 1 row affected (0.00 sec)

mysql> _

```

Obrázek 13 - Vkládání hodnot do tabulky

System MySQL také umožňuje vkládání několika záznamů najednou.

```

INSERT INTO název_tabulky (sloupec1, sloupec4, ...)
VALUES (hodnota11, hodnota41, ...),
(hodnota12, hodnota42, ...),
(hodnota13, hodnota43, ...);

```

3.9 Zobrazení dat v tabulce

Pro zobrazení dat v tabulce se používá základní výběrový příkaz **SELECT** a zapisuje se v následujícím tvaru. Hvězdička znamená, že se zobrazí všechny údaje v tabulce.

```
SELECT * FROM název_tabulky;
```

```

mysql> SELECT * FROM zakaznici;
+-----+-----+-----+-----+-----+-----+
| id_cislo | jmeno | prijmeni | firma | email | datum_registrace |
+-----+-----+-----+-----+-----+-----+
| 1 | Jan | Jánský | ABC | jansky@abc.cz | 2005-02-16 |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

```

Obrázek 14 - Zobrazení dat v tabulce

3.10 Manipulace s daty

Odstraňování dat

Pro účely odstraňování dat se používá příkaz **DELETE**:

```
DELETE FROM název_tabulky WHERE sloupec = 'hodnota';
```

Je-li třeba zajistit větší bezpečnost před nechtěným odstraněním záznamů, může se použít klauzule **LIMIT**, která omezí počet odstraňovaných záznamů:

```
DELETE FROM název_tabulky WHERE sloupec = 'hodnota'LIMIT 1;
```


3.11 Změna struktury

Základním příkazem, který se používá pro jakoukoliv manipulaci s existující tabulkou je **ALTER TABLE**.

```
ALTER TABLE název_tabulky vlastní příkaz pro aktualizaci;
```

Přidání sloupců do tabulky

V případě, že v tabulce již nějaká data jsou, lze použít následující příkaz:

```
ALTER TABLE název_tabulky  
ADD COLUMN jméno_sloupce typ_sloupce [integritní omezení];
```

Přejmenování sloupce

Existují dvě možnosti jak přejmenovat sloupec. První možnost je rychlejší a využije se v ní již známý příkaz **ALTER TABLE**.

```
ALTER TABLE název_tabulky CHANGE COLUMN staré_jméno_sloupce nové_jméno_sloupce  
typ_sloupce [integritní omezení];
```

Druhá možnost je obecnější a je třeba provést ji v několika krocích. Nejprve vytvořit nový sloupec, zkopírovat hodnoty ze starého sloupce do nového a smazat starý sloupec :

```
ALTER TABLE název_tabulky  
ADD COLUMN jméno_sloupce typ_sloupce [integritní omezení];  
  
UPDATE název_tabulky  
SET staré_jméno_sloupce = nové_jméno_sloupce;  
  
ALTER TABLE název_tabulky  
DROP staré_jméno_sloupce;
```

Přejmenování tabulky

Změna názvu tabulky je možná v MySQL následovně:

```
ALTER TABLE název_tabulky  
RENAME AS nový_název_tabulky;
```

Odstranění tabulky

Odstranění tabulky se provede příkazem **DROP TABLE**. Pokud se provede tento příkaz zničí se všechna data, která byla v mazané tabulce uložena.

```
DROP TABLE název_tabulky;
```

Odstranění databáze

V MySQL lze odstranit celou databázi. Provede se to příkazem **DROP DATABASE**. Pokud je proveden příkaz **DROP DATABASE**, budou odstraněny všechny tabulky a data, která byla v databázi uložena.

```
DROP DATABASE název_databáze;
```

3.12 Vazby mezi tabulkami

Vazby mezi tabulkami lze rozdělit do následujících tří skupin.

1 : 1

1 : N

M : N

Reprezentace vazby 1:1

Vztah 1:1 je nejjednodušší, a zároveň nejvíce neobvyklý vztah, kdy každému prvku jednoho objektu náleží právě jeden prvek druhého objektu.

Reprezentace vazby 1:N

Tato vazba je nejobvyklejším vztahem. Jeden prvek jednoho objektu je možné spojit s více prvky druhého objektu pomocí takzvaného cizího klíče. Cizí klíč je zvláštní druh intergritního omezení. Hodnotu samotné položky vůbec nezobrazuje, ale zato popisuje vazbu mezi tabulkami. Provázané sloupce musí být stejného typu. Následující řádky ukazují strukturu tabulek:

Struktura tabulek:**ZAMESTNANEC :**

ID_ZAM	JMENO	PRIJMENI	ROD_CISLO	ODDELENI	FUNKCE
1	Jan	Nový	751015/2352	10	127
2	Petr	Novák	780401/4421	15	121
3	Ivan	Nová•ek	650906/1566	10	156
4	Pavel	Novotný	740205/3566	20	127

ODDELENI :

ID_ODD	NAZEV	ID_FN	FUNKCE	PLAT
10	sklad	121	vedoucí	21500
15	centrála	127	technik	15000
20	po•íta•ový sál	156	správce	17500

FUNKCE :**Vytvoření tabulek ve vztahu 1: N a jejich provázání**

```
CREATE TABLE oddeleni (id_odd INTEGER,
nazev VARCHAR(20),
PRIMARY KEY (id_odd));
CREATE TABLE funkce
(id_fn INTEGER,
funkce VARCHAR(10),
plat FLOAT(10) DEFAULT 8000,
PRIMARY KEY (id_fn));
CREATE TABLE zamestnanec
(id_zam INTEGER,
jmeno VARCHAR(10),
prijmeni VARCHAR(20),
rod_cislo VARCHAR(11) NOT NULL,
oddeleni INTEGER,
funkce INTEGER,
PRIMARY KEY (id_zam),
FOREIGN KEY (oddeleni) REFERENCES oddeleni (id_odd),
FOREIGN KEY (funkce) REFERENCES funkce (id_fn));
```

Reprezentace vazby M:N

Vazba M:N spočívá v tom, že více prvků jednoho objektu je možné spojit s více prvky objektu druhého. Struktura tabulek následuje.

Struktura tabulek:**ZAMESTNANEC :**

ID_ZAM	JMENO	PRIJMENI	ROD_CISLO
1	Jan	Nový	751015/2352
2	Petr	Novák	780401/4421
3	Ivan	Nová•ek	650906/1566
4	Pavel	Novotný	740205/3566

FUNKCE :**VYKON_FUNKCE :**

ID_FUN	NAZEV	CISLO_ZAM	CISLO_FUN
121	vedoucí	1	127
127	technik	2	121
156	správce	3	156

Vytvoření tabulek ve vztahu M: N a jejich provázání

```
CREATE TABLE zamestnanec (id_zam INTEGER,
jmeno VARCHAR(10),
prijmeni VARCHAR(20),
rod_cislo VARCHAR(11) NOT NULL,
PRIMARY KEY (id_zam));

CREATE TABLE funkce
(id_fn INTEGER,
nazev VARCHAR(10),
PRIMARY KEY (id_fn));

CREATE TABLE vykon_funkce
(cislo_zam INTEGER,
cislo_fn INTEGER,
PRIMARY KEY (cislo_zam, cislo_fn),
FOREIGN KEY (cislo_zam) REFERENCES zamestnanec (id_zam),
FOREIGN KEY (cislo_fn) REFERENCES funkce (id_fn));
```

4 MACROMEDIA DREAMWEAVER 8



Obrázek 15 - Macromedia Dreamweaver box

4.1 Úvod

Macromedia Dreamweaver 8 je profesionálním prostředím pro vývoj webových aplikací. Poskytuje přehlednou kombinaci grafických nástrojů a vývojového prostředí pro editaci kódu. Umožňuje využívat mnoho nástrojů pro tvorbu webu od HTML, přes PHP, Java Script až k ASP.NET C#.

V této verzi Dreamweaveru se Macromedia zaměřila především na zlepšení práce se zdrojovým kódem a kaskádovými styly. Práce s CSS ve starších verzích se většinou prováděla pomocí externích programů (např. TopStyle) ale s nástupem Dreamweaveru MX2004 se práce s CSS natolik “zpříjemnila”, že již nebylo nutno externí programy používat, a ve verzi 8 se přívětivost používání CSS ještě zlepšila.

Pro práci s CSS je nyní k dispozici opět přepracovaná paletka *CSS Styles*. V rámci ní lze přepínat mezi nastavením pro aktuálně vybraný element stránky či mezi všemi objekty stránky. V případě, že je třeba se podívat na vlastnosti vybraného elementu, Dreamweaver detekuje odvozené vlastnosti, a tak vedle přehledu atributů, které jsou přímo v definici dané třídy, zobrazí i seznam těch, které jsou děděny. Pokud je dokument upravován přímo, paletka si ohlíká, zda element danou vlastnost podporuje.

Dreamweaver obsahuje vylepšené zobrazování finálního vzhledu stránky, takže často není nutno spouštět vytvářené stránky v prohlížeči. Kvůli nutnosti interakce ale preview úplně dokonalé není.

4.2 Uvítací okno

Po spuštění aplikace Dreamweaver 8 se objeví okno se třemi nabídkami, a to :

- a) Otevřít naposledy měněné dokumenty
- b) Vytvořit nový dokument
- c) Vytvořit nový dokument z předlohy



Obrázek 16 - Uvítací okno Dreamweaveru

Otevření naposledy měněných dokumentů

Tato nabídka umožňuje jednoduchý přístup ke zrovna rozpracovaným (naposledy měněným) dokumentům, případně je možno použít položku *Open* ve spodní části okna.

Vytvoření nového dokumentu

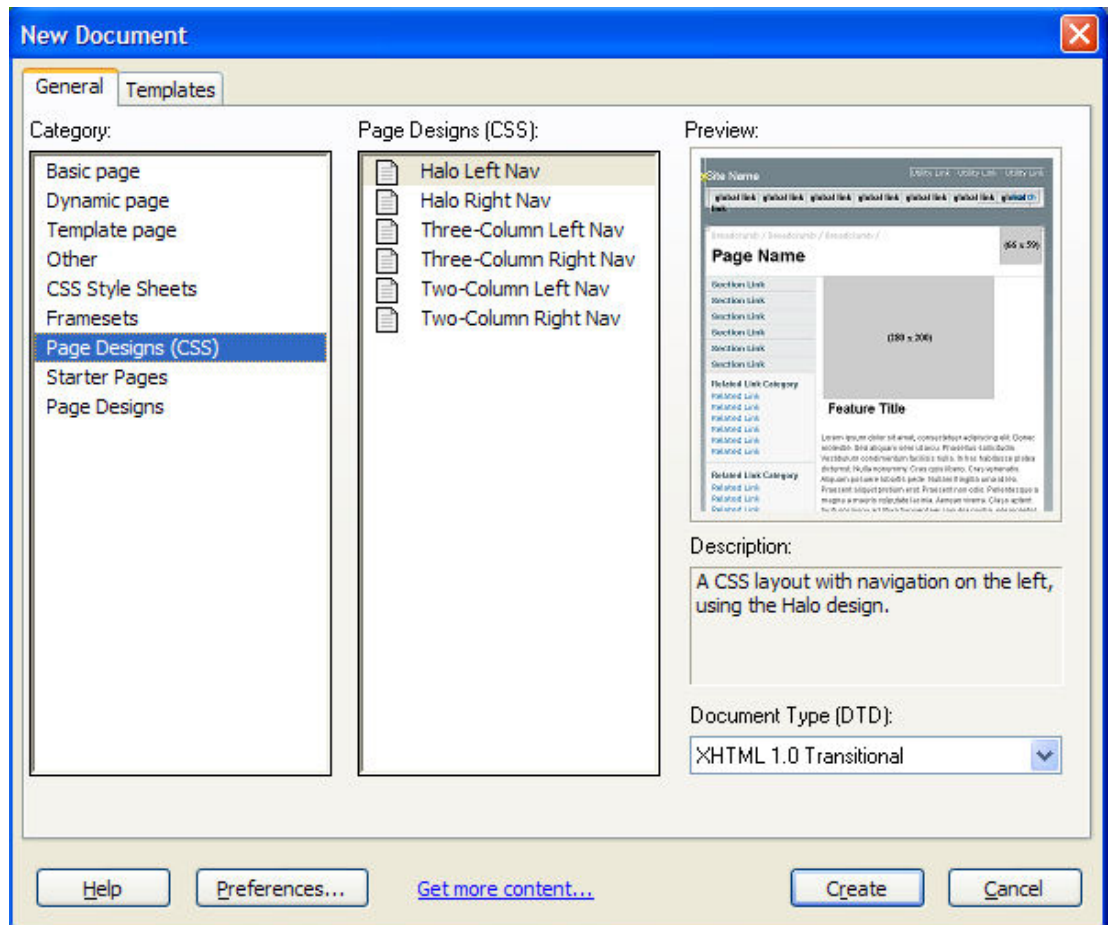
Tato část uvítacího okna dává uživateli na výběr jaký typ dokumentu může vytvořit. Nabídka je následující:

HTML	JavaScript
ColdFusion	XML
PHP	XSLT
ASP VBScript	CSS
ASP.NET C#	DreamweaverSite

Upřesnění výběru a další možnosti se nacházejí v nabídce *more*.

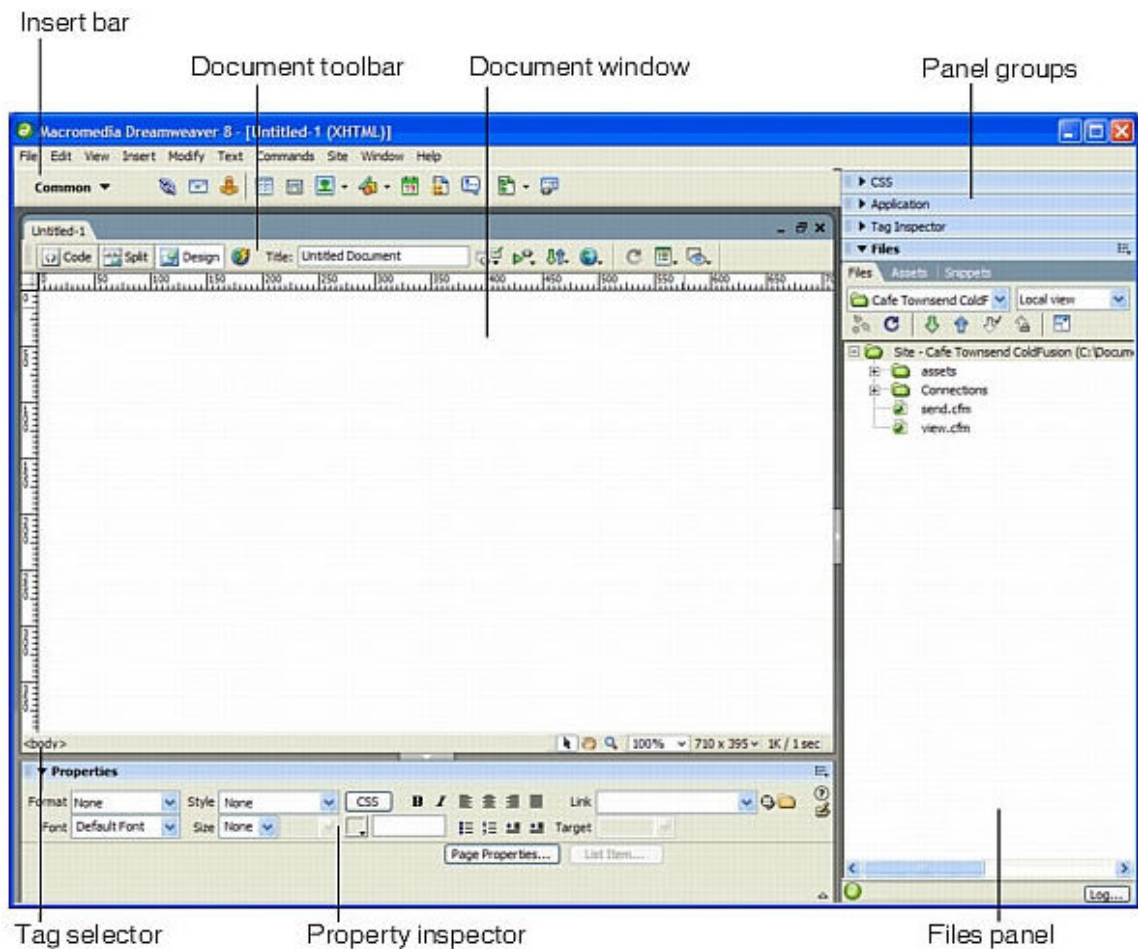
Vytvoření nového dokumentu z Předlohy

Macromedia Dreamweaver 8 umožňuje vytváření dokumentů předloh, ke kterým se dá jednoduše přistupovat hned z uvítacího okna. Obsahuje mnoho různých druhů designu stránek pomocí CSS stylů, i bez nich, mnoho možností rozložení stránek atd.



Obrázek 17 - Výběr dokumentu z předlohy

4.3 Okno editoru

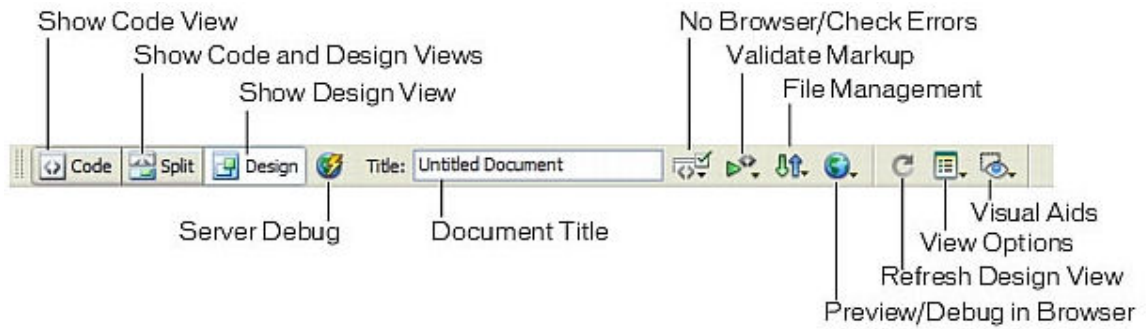


Obrázek 18 - Okno editoru

Okno editoru je rozděleno do několika částí. Jsou to *insert bar* (vkládací lišta), *document toolbar* (nástrojová lišta dokumentu), *document window* (okno dokumentu), *panel groups* (skupiny panelů), *tag selector* (výběr tagů), *property inspector* (inspektor vlastností) a *files panel* (panel souborů).

Insert bar (vkládací lišta)

Vkládací lišta obsahuje tlačítka pro tvorbu a vkládání objektů, jako například tabulek, vrstev a obrázků. Tlačítka jsou organizována do několika kategorií, které lze přepínat v levé části vkládací lišty. Po spuštění Dreamweaveru otevře se kategorie která byla naposledy otevřena.



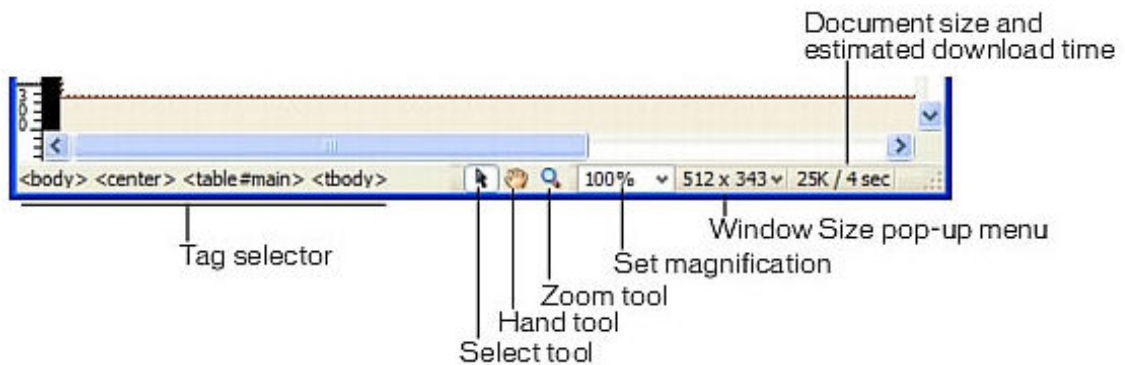
Obrázek 20 - Vkládací lišta

Dále jsou zde tlačítka:

- **Server Debug** Zobrazí zprávu, která pomáhá ladit stránku, obsahuje také chyby na stránce pokud stránka nějaké obsahuje.
- **Document Title** Jméno stránky zobrazené na hlavním panelu okna prohlížeče.
- **No Browser/Check Errors** Kompatibilita mezi prohlížeči.
- **Validate Markup** Ověří správnost stránky nebo vybraného tagu.
- **File Management** Otevře výsuvnou nabídku správy souborů.
- **Preview/Debug in Browser** Zobrazí výsuvnou nabídku správy souborů.
- **Refresh Design View** Obnoví náhledové zobrazení po úpravách v kódu.
- **View Options** Zobrazí vysouvací lištu nastavení aktuální stránky.
- **Visual Aids** Zobrazí názorné pomůcky (designová pomoc).

The status bar (stavový řádek)

Stavový řádek ve spodní části okna dokumentu poskytuje rozšiřující informace o právě vytvářeném dokumentu.



Obrázek 21 - Stavový řádek

- **Tag selector** Ukazuje hierarchii tagů v aktuální oblasti.
- **Hand tool** Nástroj ruka.
- **Select tool** Nástroj výběr.
- **Zoom tool** Nástroj lupa (až 6400x)
- **Window Size pop-up menu** (pouze v *Design view*) Umožňuje zvolit velikost stránky
- **Coding toolbar(Nástrojová lišta Code)**

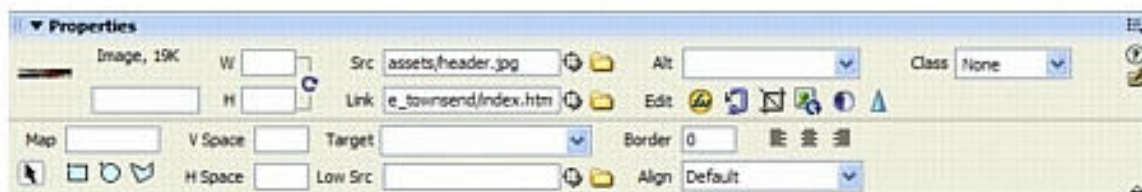
Nástrojová lišta *code* obsahuje tlačítka, která umožňují jednoduchou práci s kódem. Například sbalování a rozbalování vybrané části kódu, vysvícení chybného kódu, přidávání, nebo odebrání poznámek. Lišta *code* je viditelná pouze v kódovém zobrazení a zobrazuje se v levé části okna dokumentu.

```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
5 <title>uprava dat</title>
6 <link href="style/info.css" rel="stylesheet" type="text/css" />
7 </head>
8
9 <body>
10 <table align="center" width="80%" bgcolor="#CCCCCC">
11 <tr>
12 <td colspan="2" align="center" bgcolor="#CCCCCC"><font color="#FFFFFF" size="+5">VYHLEDÁNÍ TRÍDY</font></td>
13 </tr>
14 </table>
15
16 <br />
17
18 <table bgcolor="#CCCCCC" align="center" width="60%" border="0" >
19 <tr>
20 <td align="center">
21
22 Zadejte číslo třídy u které chcete změnit parametry:
23 <form action="zmena.php" method="post">
24 Pavilon:<br />
25 <input type="text" name="pavilon" value="pavilon..." /><br />
26 Trída:<br />
```

Obrázek 22 - Nástrojová lišta CODE

Property Inspector (Inspektor vlastností)

Inspektor vlastností umožňuje kontrolovat a editovat parametry aktuálně zvoleného elementu stránky, jako je například text nebo vložený objekt. Pro každý element jsou vlastnosti jiné, což znamená, že se pro každý element zobrazí jiné vlastnosti. U obrázků jsou to například síla okraje, barva okraje, šířka, výška.

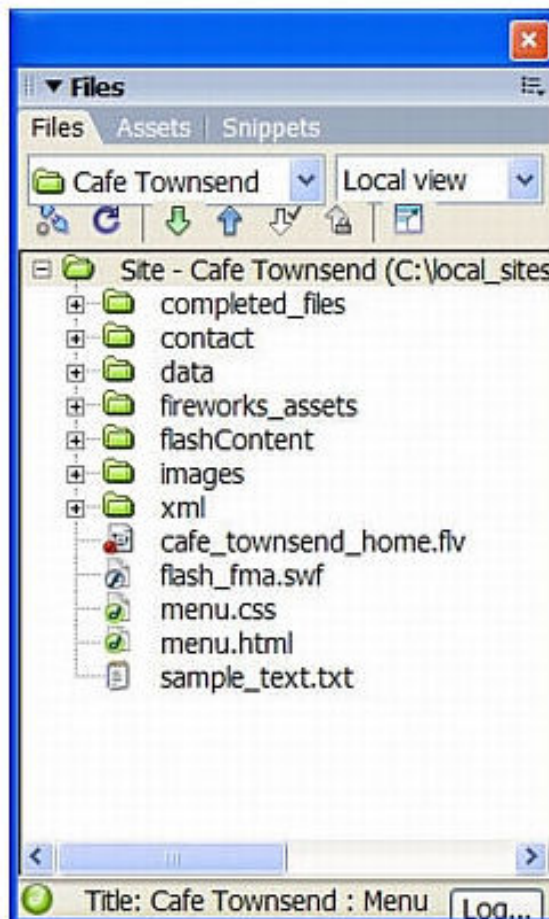


Obrázek 23 - Inspektor vlastností

Inspektor vlastností se nachází ve spodní části okna dokumentu, ale je možné jej přesunout například do horní části, nebo kamkoli na pracovní plochu.

File panel (Panel souborů)

Tento panel slouží k prohlížení a správě souborů v síti.



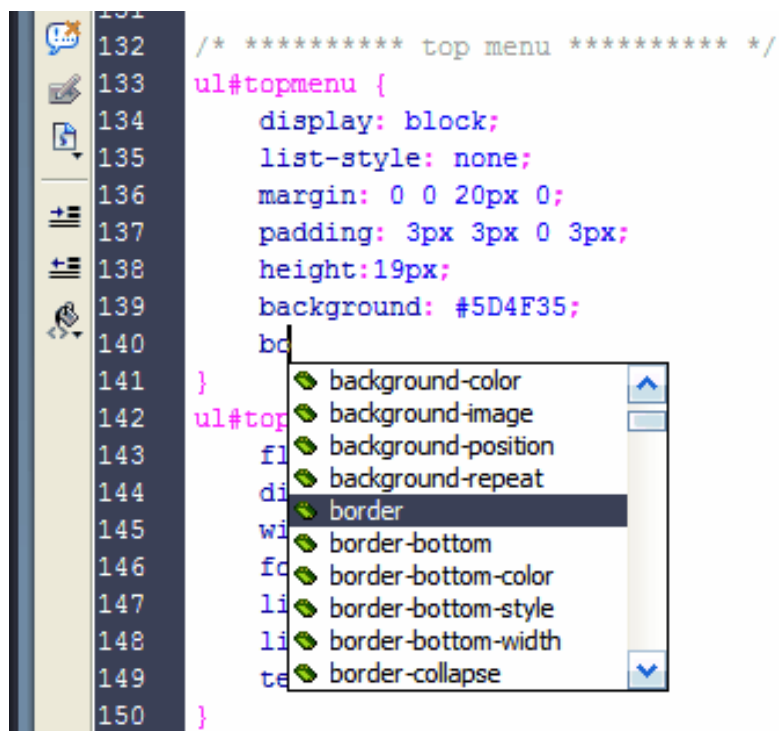
Obrázek 24 - Panel souborů

Při prohlížení sítí, souborů nebo složek v panelu souborů lze měnit velikost prohlížečské oblasti, popřípadě je možné ji sbalit, nebo rozbalit.

CSS editor

Základem tohoto editoru je WYSIWYG paletka pro CSS, která obsahuje CSS definice aktuálně zvoleného tagu, a to nejen ty přímo definované, ale včetně vlastností zděděných dle aktuálního kontextu tagu ve struktuře dokumentu od jeho „rodičů“ (což je mimořádně užitečné, je dokonce přesně vidět i ze kterých tříd jsou dané CSS vlastnosti poděděny!)

Pokud je dvakrát kliknuto na CSS paletku, zobrazí se zdrojový CSS soubor. Tento je následně možné jakkoliv upravovat, a to za pomoci editoru se zabudovanou kontextovou nápovědou a doplňováním.



Obrázek 25 - Automatické doplňování příkazů

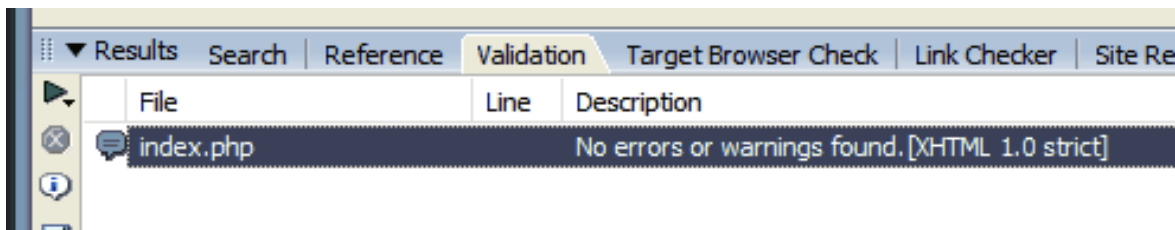
Dreamweaver 8 jde do všech detailů, pokud se v CSS souboru napíše vlastnost `color`, je ihned nabídnuta paletka pro výběr barvy:



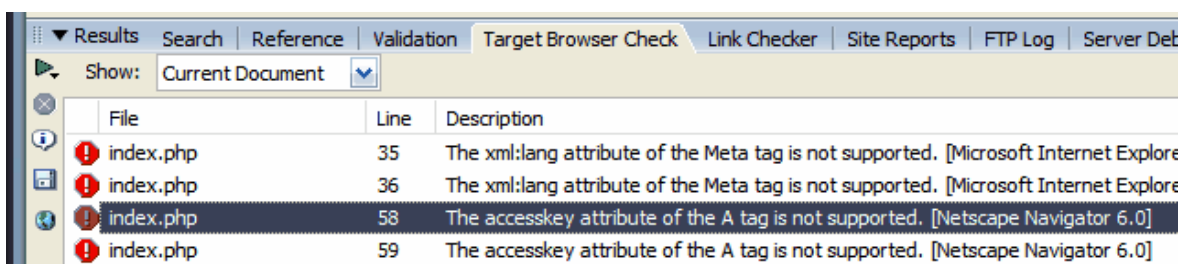
Obrázek 26 - Paletka barev

4.4 Další funkce

Funkce Dreamweaveru 8 zde zdaleka nekončí, při editaci libovolné stránky se bude mimořádně hodit i integrovaná kontrola validity stránky dle W3C norem, kontrola stránky na vlastnosti v ní obsažené vzhledem k uživatelsky definovaným browserům (je tak ihned vidět, které vlastnosti nebudou podporovány třeba v Netscape 6), kontrola validity odkazů na stránce a další funkce:

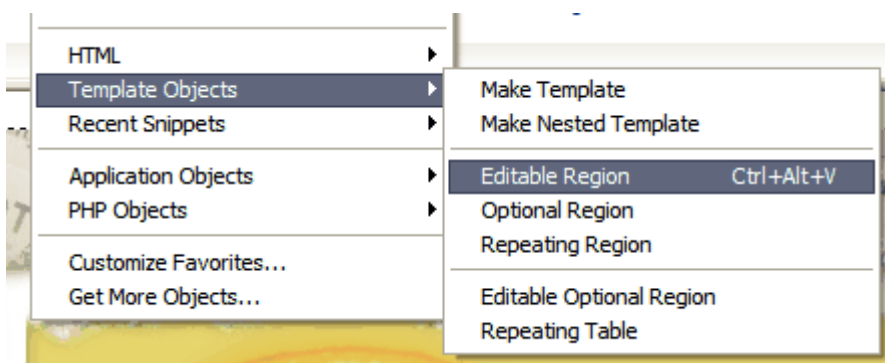


Obrázek 27 - Validace podle norem V3C



Obrázek 28 - Správnost reprezentace v prohlížečích

Dreamweaver 8 také umožňuje tvorbu šablon (templates) s oblastí, která je editována, a následně při úpravě zdrojové šablony jsou upravovány i veškeré stránky od této šablony odvozené, což je výborná funkce zvláště pro statické weby bez podpory PHP:



Obrázek 29 - Tvorba šablon

4.5 Závěr

Dreamweaver 8 je na trhu jeden z nejlepších nástrojů, který je možné si pro tvorbu XHTML, CSS, JS, PHP pořídit. Dreamweaver 8 ve srovnání s předchozí verzí MX 2004 možná nenabízí takovou řadu nových a zásadních funkcí, které by člověka ohromily, ale spíše je jaksi celý „*dokončen a dotažen k dokonalosti*“. Vše v něm funguje přesně tak jak má a jak se očekává, a člověk je s tímto produktem skutečně podstatně rychlejší.

II. PRAKTICKÁ ČÁST

5 WWW NAVIGÁTOR PROSTORAMI FAI

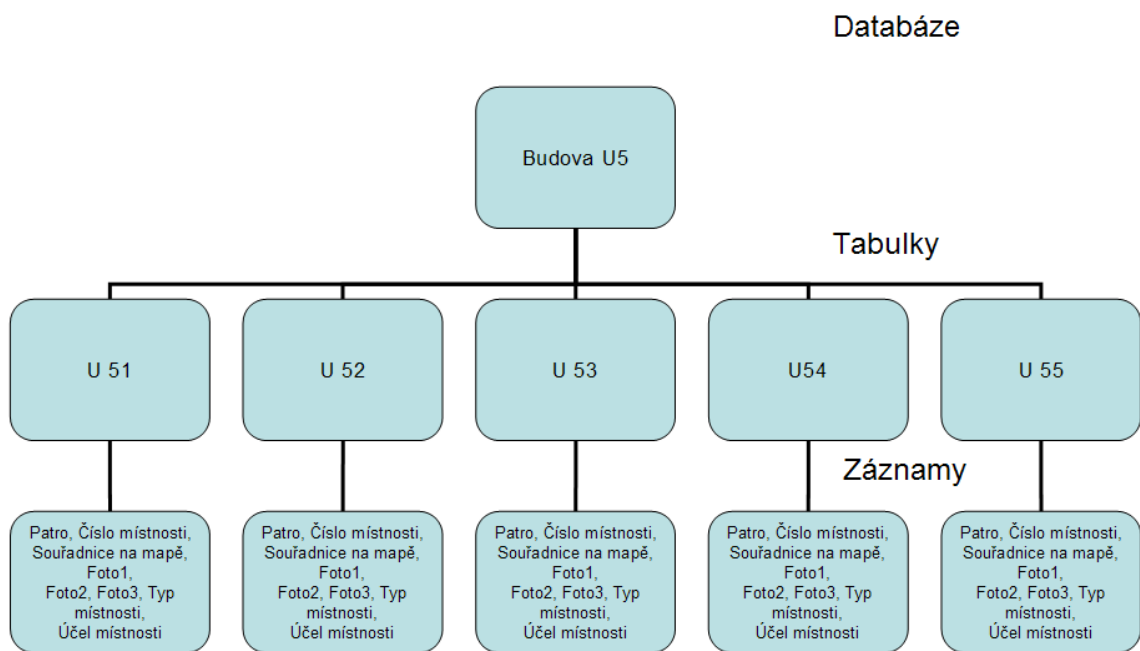
5.1 Úvod

WWW navigátor prostorami Fakulty aplikované informatiky pracovně nazvaný CLASS HUNTER je webová aplikace sloužící k orientaci prostorami areálu U5 Univerzity Tomáše Bati ve Zlíně. K tvorbě této aplikace bylo využito HTML jazyka, CSS stylů, dále jazyka PHP pomocí kterého je aplikace spojena s SQL databází.

5.2 Databáze

K vytvoření tohoto projektu bylo třeba navrhnout databázi, do které se budou ukládat informace o jednotlivých místnostech budovy. Databáze obsahuje 5 tabulek, z nichž každá reprezentuje jeden pavilon budovy. Každá z těchto tabulek obsahuje informace o poloze místnosti v budově (tj. ve kterém leží patře a číselné označení této místnosti.), dále jsou zde uložena data o poloze místnosti na mapě (souřadnice).

Následují tři fotografie, dvě zachycují přístupové chodby k jednotlivým místnostem a jedna zobrazuje dveře místnosti. V každé z tabulek jsou také uložena data o obsazení místnosti, pokud se jedná o kancelář, popřípadě informaci o tom, kterému ústavu místnost patří. Jedná-li se o učebnu, jsou zde uloženy informace o účelu vybavení této místnosti. A jako poslední položka tabulky je účel této místnosti.



Obrázek 30 - Struktura databáze

Jednotlivé sloupce tabulky mají různé datové typy. Sloupce s údaji o patře (**level**) a číselném označení místnosti (**class**) jsou typu **char** a jsou omezeny na 1 a 2 znaky, sloupec se souřadnicemi (**coordinate**) je typu **varchar** s omezením na 20 znaků. Foto1,2 a 3 jsou typu **blob**, a sloupce s daty o typu, obsazení a účelu místnosti mají datový typ **text**.

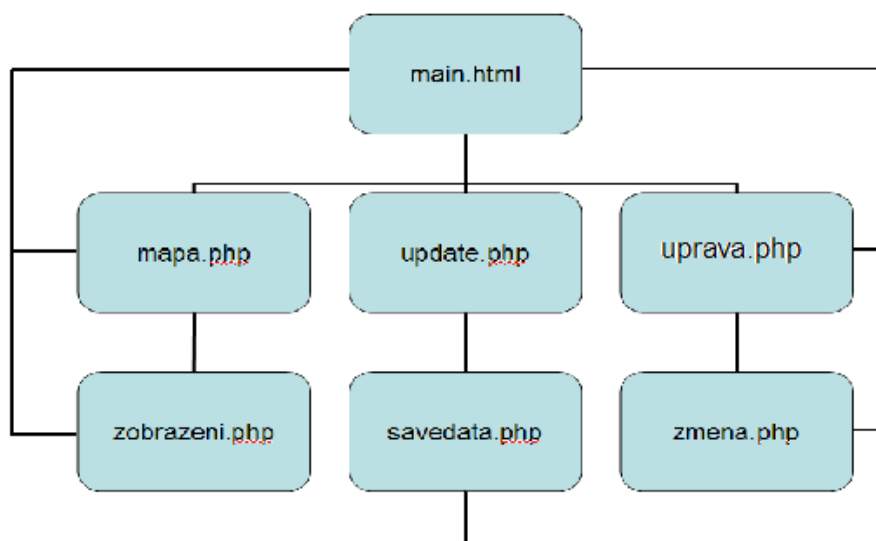
Příklad uložených dat(detail údajů o místnosti):

Sloupec	Typ	Funkce	Nulový	Hodnota
ID	int(11)	<input type="text"/>	<input type="checkbox"/>	6
level	char(1)	<input type="text"/>	<input type="checkbox"/>	8
class	char(2)	<input type="text"/>	<input type="checkbox"/>	12
coordinate	varchar(20)	<input type="text"/>	<input type="checkbox"/>	345,298
photo1	blob	Binární	<input type="checkbox"/>	Binární - nepravujte (0 bajtů) H:\škola\bc\sl ProcházeL (Maximální velikost: 65 536bajtů)
photo2	blob	Binární	<input type="checkbox"/>	Binární - nepravujte (0 bajtů) H:\škola\bc\sl ProcházeL (Maximální velikost: 65 536bajtů)
photo3	blob	Binární	<input type="checkbox"/>	Binární - nepravujte (0 bajtů) H:\škola\bc\sl ProcházeL (Maximální velikost: 65 536bajtů)
type	text	<input type="text"/>	<input type="checkbox"/>	Kancelář
obsazeni	text	<input type="text"/>	<input type="checkbox"/>	Ústav elektrotechniky a měření Ing. Stanislav Goňa Ph.D. Ing. Zdeněk Malánik

Obrázek 31 - Detail údajů o místnosti

5.3 PHP a HTML

Celá aplikace je tvořena sedmi PHP skripty a dvěmi HTML stránkami. PHP skripty zajišťují dynamičnost aplikace, jako propojení s databází, ukládání do databáze a změnu dat v databázi. Následuje mapa nejdůležitějších stránek (informační stránky jsou vynechány):

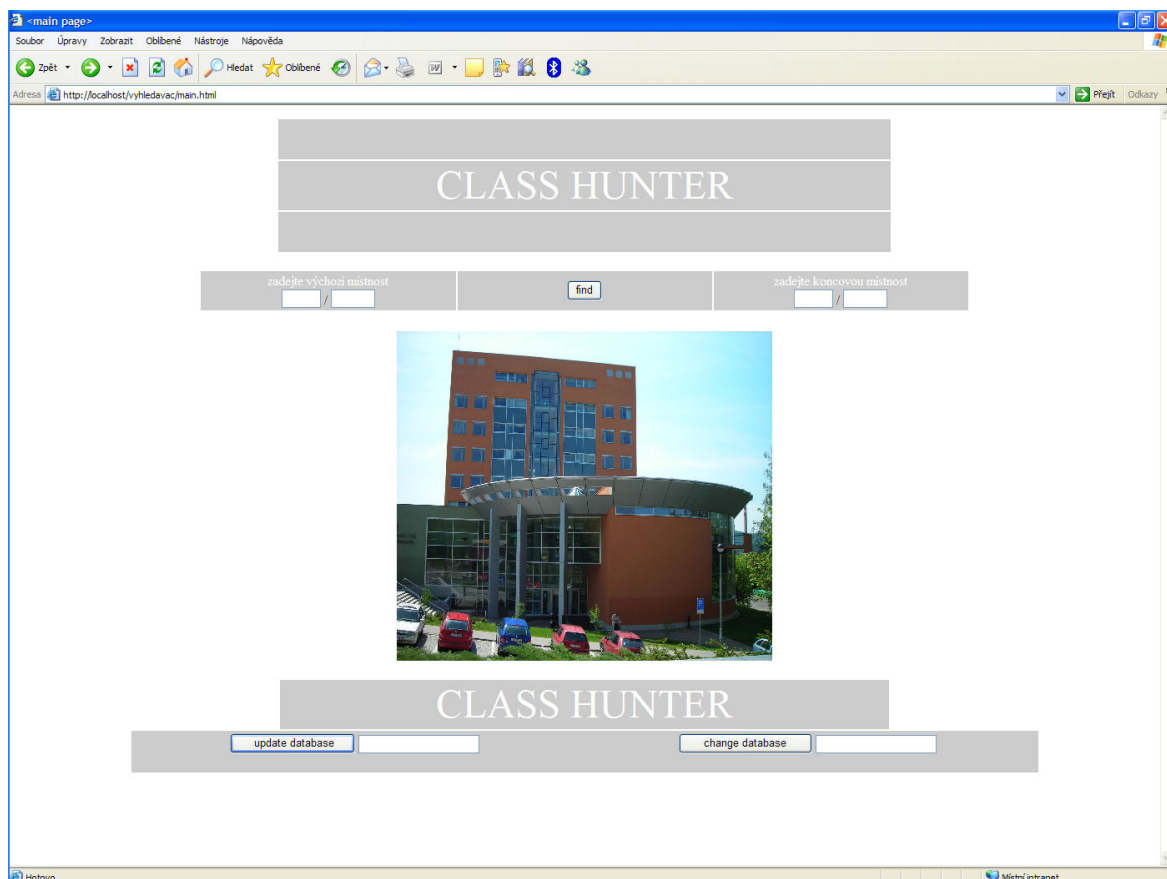


Obrázek 32 - Struktura stránek

5.4 Popis jednotlivých částí aplikace:

main.html

Stránka *main.html* je hlavní stránkou aplikace. Odtud je možné přistupovat ke všem třem hlavním funkcím projektu, tj. k funkci vyhledání místností (tlačítko *FIND*), přidávání dat do databáze (tl. *UPDATE DATABASE*) a úpravě dat v databázi (*CHANGE DATABASE*).



Obrázek 33 - main.html

Stránka je zapsána v čistém HTML kódu a je složena z několika tabulek, které jsou přesně umístěny ve stránce. Jsou to například tabulka pro hlavní nadpis, tabulka s formuláři pro zadání počáteční a koncové místnosti, tabulka s fotografií školy a tabulka s odkazovými tlačítky.

Příklad zápisu kódu pro vytvoření tabulky s formuláři:

```

<form action="mapa.php" method="post">
<table width="68%" border="0" cellspacing="2" cellpadding="2" align="center">
<!--vychozi mistnost----->
<tr>
<td class="gray" align="center" width="33%">
<font color="#FFFFFF">zadejte výchozí místnost</font>
<br />
<input name="startpav" maxlength="2" size="2" />
/
<input name="startcl" maxlength="3" size="3">
</td>
<!--odeslat----->
<td align="center" class="gray" width="33%">
<input type="submit" name="add" value=" find " /></td>
<!--koncova mistnost----->
<td class="gray" align="center" width="33%">
<font color="#FFFFFF">zadejte koncovou místnost</font>
<br />
<input name="endpav" maxlength="2" size="2"/>
/
<input name="endcl" maxlength="3" size="3">
</td>
</tr>
</table>
</form>

```

Ovládání stránky main.html

Ovládání této stránky je velmi jednoduché. Do formulářových prvků v levé horní části musí být zapsáno číselné označení místnosti, ve které se uživatel právě nachází, a to ve formátu *pavilon/číslo místnosti* a do formuláře v pravé horní části musí být zapsáno označení místnosti, do které se chce uživatel dostat.

Po stisknutí tlačítka *FIND* je uživatel přesměrován na stránku *mapa.php*, kde se zobrazí umístění tříd na jednoduché mapě celé budovy.

Pokud je uživatel oprávněn používat i ostatní funkce aplikace (zná přístupové heslo), může využívat i dalších možností jako jsou úprava a přidávání dat do databáze. Po zadání

správného hesla a stisknutí příslušného odkazového tlačítka je uživateli umožněn přístup k dalším funkcím.

mapa.php

Tento PHP skript je nejrozsáhlejším skriptem celého projektu. Zajišťuje zobrazení mapy, popřípadě dvou map příslušných poschodí budovy a vyznačení polohy vchodu do místnosti pomocí červeného čtverce, který současně slouží jako hypertextový odkaz na stránku s informacemi o aktuálně zvolené místnosti. Pro představu jsou zde následně uvedeny některé části kódu stránky a také reprezentace zmíněné stránky v prohlížeči.

Stránka *mapa.php* je tvořena částmi HTML kódu a částmi PHP skriptu. Zobrazení jedné či dvou map je založeno na datech získaných z předchozí stránky (*main.html*), které jsou posílány metodou post do skriptu pomocí formulářových prvků. Data získaná z formulářů jsou uložena v proměnných pod stejným jménem jako je hodnota **name** formulářového prvku **input** v předchozí stránce s tím rozdílem, že jméno proměnné musí začínat speciálním znakem **\$**.

Po přístupu na stránku se PHP skript okamžitě připojí k databázi, odkud získá informace o umístění místností na mapě, čili jejich souřadnice. Po té skript zjistí zda místnost, ve které se uživatel nachází a místnost koncová leží ve stejném pavilonu a na stejném patře. Pokud ano, je zobrazena pouze jedna mapa a na ní dva odkazové ukazatele ve formě červených čtverců. Pokud ne, zobrazí se mapy dvě a na každé z nich se podle souřadnic z databáze zobrazí poloha místnosti.

Ukázky kódu stránky

```
<div align='center' style = 'background-image:url(images/first.png); background-repeat:no-repeat; width:600px; height:562px; position:absolute; top:20%; left:28%'>
<div style='position:absolutetop:<?php $coords = explode(',',$radek['coordinate']);
print($coords[0]);?>px; left:<?php $coords = explode(',',$radek['coordinate']);
print($coords[1]);?>px;'>
<a href='zobrazeni.php?pavilon=<?php echo($startpav);?>&patro=<?php
echo(level($startcl));?>&trida=<?php echo(room($startcl));?>'
style='border:none'><img src='images/rect.bmp' border='0'/></a></div>
```


Tato část kódu zobrazí mapu budovy, která je umístěna ve stránce, určí její velikost a vykreslí červený odkazový čtverec. Dále posílá metodou **get** (přenášené informace jsou zobrazeny v adresovém řádku prohlížeče) informace o místnosti do dalšího skriptu aplikace (*zobrazeni.php*).

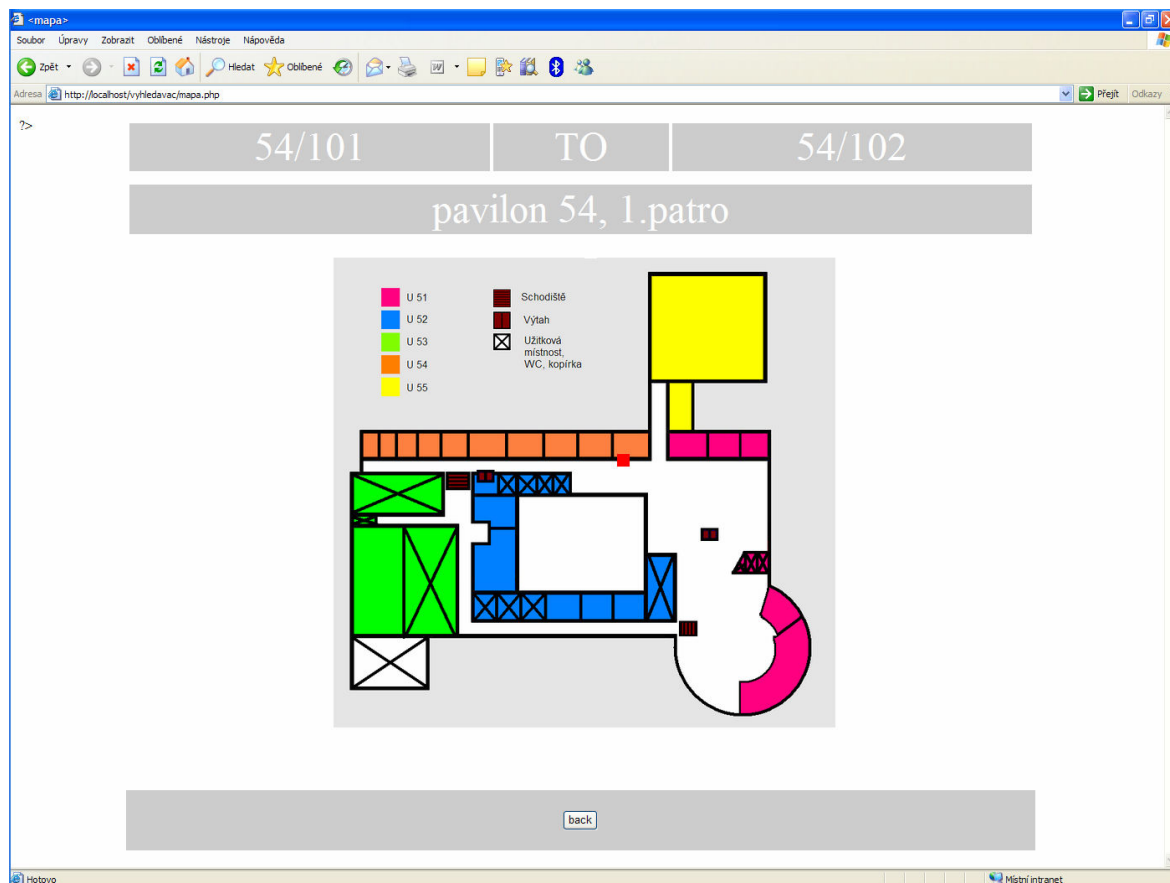
```
<?php $query = "SELECT coordinate FROM u".$sendpav." WHERE level=".level($sendcl)." and class
=".room($sendcl);

if(!$help = mysql_query($query,$link)) echo("NEJDE TO!");
$radek = mysql_fetch_array($help);?>
```

Tento útržek kódu definuje dotaz pro výběr dat z databáze. Obsahuje některé funkce používané ve více skriptech, které jsou nadefinované v souboru *common.php*, jež je pro skripty společný a vkládá se do zdrojového kódu pomocí klíčového slova **include**. Zapisuje se:

```
include 'common.php';
```

Pro tento skript a vlastně pro celý projekt bylo třeba vytvořit jednoduchou mapu budovy. Mapa je přehledně barevně rozčleněna do jednotlivých pavilonů. Jednodušeji a výstižněji by se dalo říci, že místnosti náležící jednotlivým pavilonům jsou barevně zvýrazněny. Každá mapa je také doplněna o přehlednou legendu ve formě barevných čtverců s popisem ke kterému pavilonu která barva náleží. Legenda dále obsahuje několik dalších značek pro schodiště nebo výtahy.



Obrázek 34 - mapa.php

zobrazeni.php

Účelem tohoto PHP skriptu je zobrazit informace o místnosti která byla zvolena. Skript se nejdříve připojí k databázi, potom porovná přijatá data z předchozího skriptu (tj. číslo místnosti a pavilon, což jsou nejdůležitější údaje pro vyhledávání v databázi). A nakonec zobrazí požadované informace.

Těmito informacemi je myšleno, že se v horní části stránky zobrazí fotografie okolí místností a jejich dveří, dále informace o obsazení místnosti a účel místnosti.

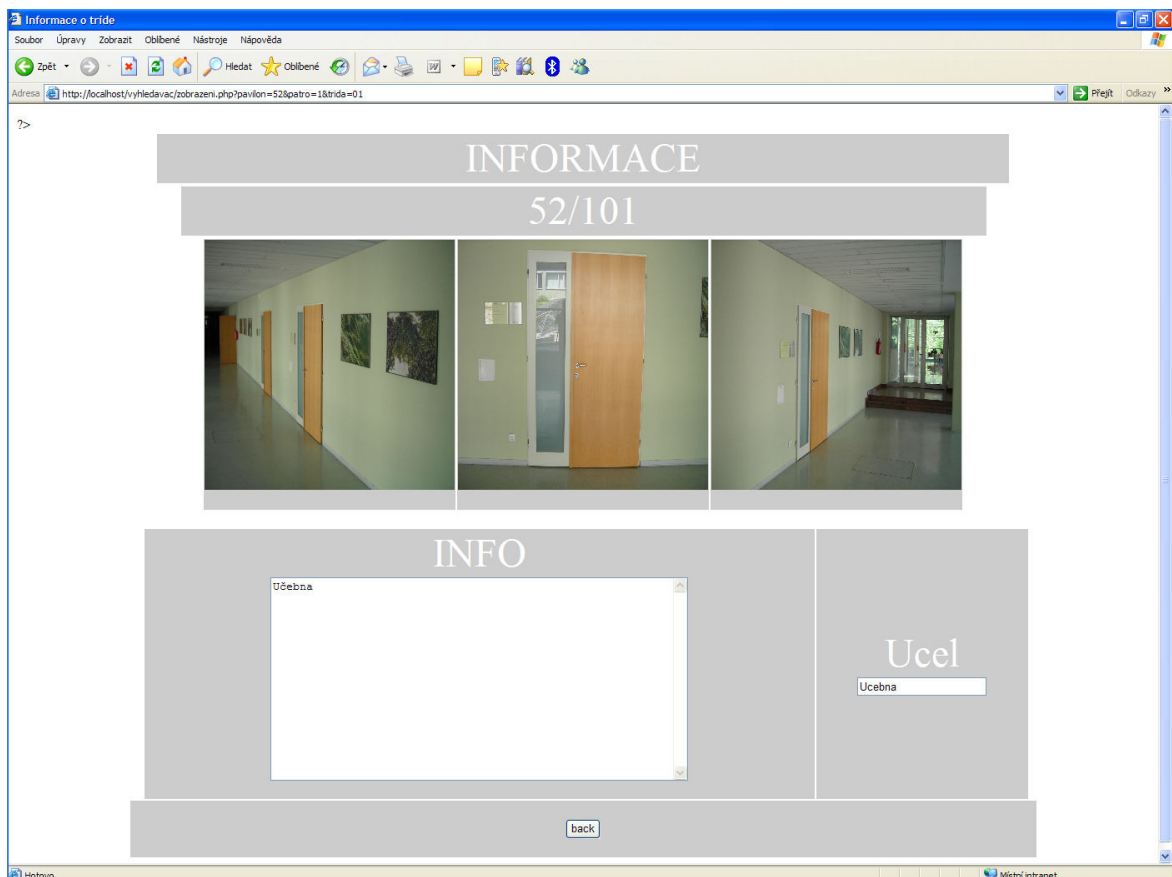
Formulářové prvky na této stránce jsou zajištěny proti přepisování pomocí atributu **readonly** v tazích `<input>` a `<textarea>`, což znamená, že z těchto prvků je možno pouze číst. Následuje ukázka kódu:

```

<table width="80%" border="0" cellspacing="2" cellpadding="2" align="center">
<tr>
<td colspan="2" align="center" bgcolor="#CCCCCC"><font color="#FFFFFF"
size="+5">INFO</font><br />
<textarea name="obsazeni" readonly="readonly" rows="15" cols="60" ><?php
echo($radek["obsazeni"]);?></textarea><br /><br /></td>
<td colspan="2" align="center" bgcolor="#CCCCCC"><font color="#FFFFFF"
size="+5">Ucel</font><br />
<input name="type" readonly="readonly" value="<?php echo($radek["type"]);?>" align="middle" >
<br />
</td>
<td colspan="2" align="center" bgcolor="#CCCCCC"><font color="#FFFFFF"
size="+5">Ucel</font><br />
<input name="type" readonly="readonly" value="<?php echo($radek["type"]);?>" align="middle" >
<br />
</td>
</tr>
</table>

```

Zpět na hlavní stránku je možné se dostat pomocí tlačítka *BACK*.



Obrázek 35 - zobrazeni.php

update.php

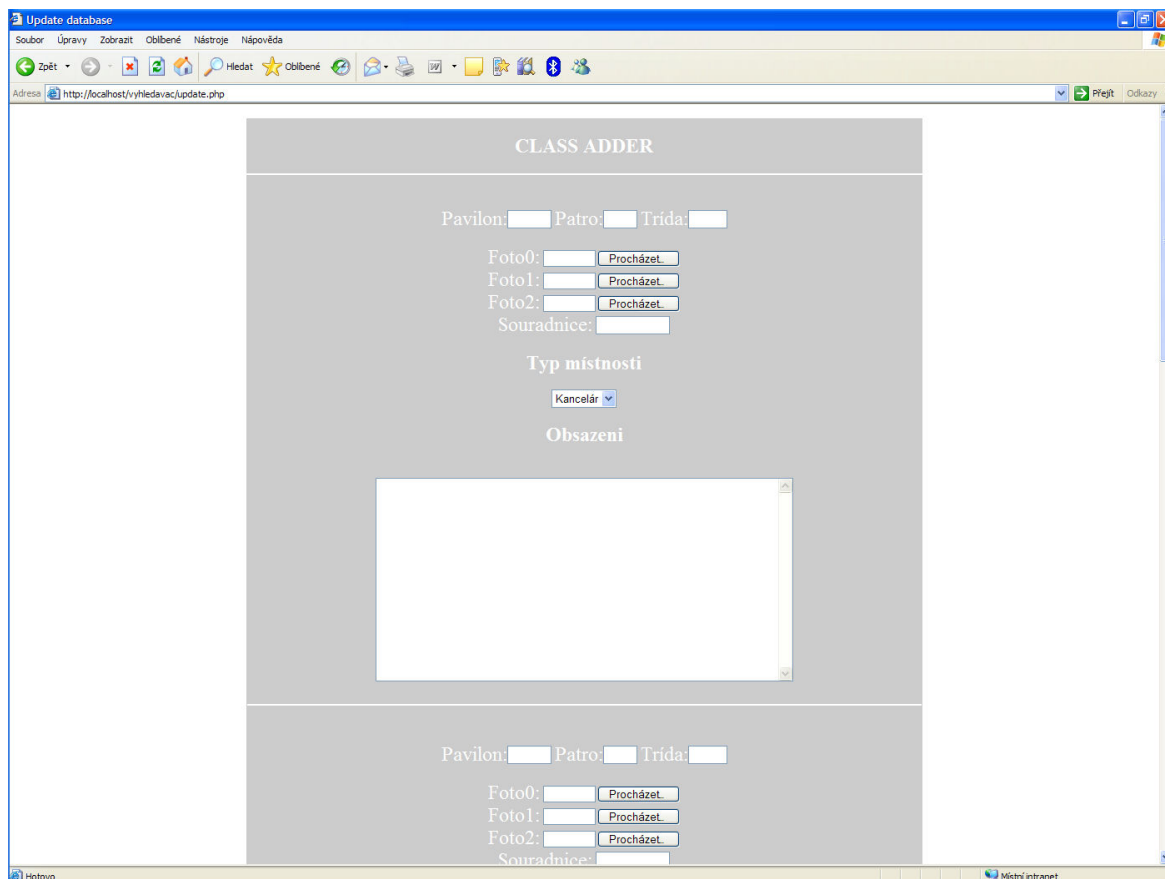
Funkcí skriptu *update.php* je získání a odeslání záznamů do další stránky . Skript v závislosti na předchozí HTML stránce (*main.html*) vygeneruje určitý počet záznamových oken. Jinými slovy, po zadání hesla a počtu přidávaných záznamů zobrazí tabulku s formulářovými vstupními prvky typů `<input>`, `<textarea>` a `<select>`, do nichž uživatel zapíše údaje o zadávané místnosti, a to číslo pavilonu, patro ve kterém se místnost nachází, její číselné označení, dále nabízí možnost přidání až tří fotografií, poté údaje o typu místnosti z rolovacího menu a informace o obsazení místnosti popřípadě dalších důležitých dat (ústav, informace o vybavení).

Přidávání dat je velmi jednoduché, stačí v každém jednotlivém okně zadat údaje do zřetelně popsaných vstupních prvků. Po stisknutí tlačítka *add* jsou informace odeslány do dalšího skriptu (*savedata.php*), který je uloží do databáze.

Co se týče zápisu kódu, obsahuje tento skript jednu výjimku, a to hodnoty atributu **name** ve formulářových elementech stránky. Je zde zapsáno jméno prvku následováno hranatými závorkami `[]`, což umožní přidávat tolik záznamů, kolik je zadáno ve stránce *main.html*. Ze všech stejnorodých údajů je totiž vytvořeno pole, k jehož jednotlivým hodnotám se v následujícím skriptu přistupuje pomocí indexů.

Ukázky kódu stránky

```
<?php if($kolik == 0){$kolik = 5;}
for($i = 1;$i < $kolik;$i++){?> <br /><br />
<font color="#FFFFFF" size="+2">Pavilon:</font><input type="text" align="bottom" name="hall[]"
maxlength="3" size="3" />
<font color="#FFFFFF" size="+2">Patro:</font><input type="text" align="bottom" name="level[]"
maxlength="1" size="1" />
<font color="#FFFFFF" size="+2">Trída:</font><input type="text" align="bottom" name="class[]"
maxlength="2" size="2" /> <br /><br />
<font color="#FFFFFF" size="+2">Foto0:</font> <input type="file" align="bottom" name="photo[]"
size="5" /><br />
<font color="#FFFFFF" size="+2">Foto1:</font> <input type="file" align="bottom" name="photo1[]"
size="5"/><br />
<font color="#FFFFFF" size="+2">Foto2:</font> <input type="file" align="bottom" name="photo2[]"
size="5"/><br />
<font color="#FFFFFF" size="+2">Souradnice:</font> <input type="text" align="bottom"
name="coords[]" size="9" maxlength="9" /> <br /><br />
<font color="#FFFFFF"><h2>Typ místnosti</h2></font>
<select name="type[]">
<option>Kancelár</option>
<option>Ucebna</option>
<option>Jiné</option>
</select> <br /><br />
<font color="#FFFFFF"><h2>Obsazeni</h2></font><br />
<textarea name="obsazeni[]" rows="15" cols="60"></textarea><br /><br />
<hr color="#FFFFFF" />
<?php }?> <br />
<input type="submit" value=" add " />
</form>
</td>
</tr>
</table>
```



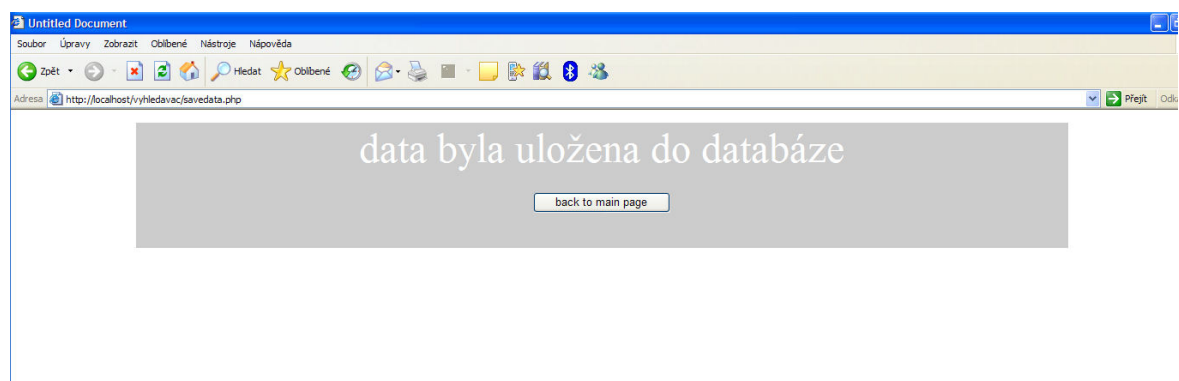
Obrázek 36 - update.php

savedata.php

Po přístoupení na tuto stránku se PHP skript okamžitě připojí k databázi a uloží obdržená data do databáze. Současně s přenášenými daty skript také obdrží proměnnou **\$kolik**, která nese informaci o počtu uložených záznamů v poli. Ukládání do databáze se provádí jednoduchým cyklem, kde proměnná určuje horní hranici počtu ukládaných dat.

Ukázky kódu stránky

```
<?phpfor($i = 0; $i < $kolik;$i++){  
$lev = $level[$i];  
$cla = $class[$i];  
$pho = $photo[$i];  
$pho1 = $photo1[$i];  
$pho2 = $photo2[$i];  
$typ = $type[$i];  
$obs = $obsazeni[$i];  
$coo = $coords[$i];  
$query = "INSERT INTO u".$hall[$i]." (level,class,photo,photo1,photo2,type,obsazeni,coordinate)  
VALUES ('$lev','$cla','$pho','$pho1','$pho2','$typ','$obs','$coo')";  
if(!mysql_query($query,$link)){echo("");}  
}?>
```

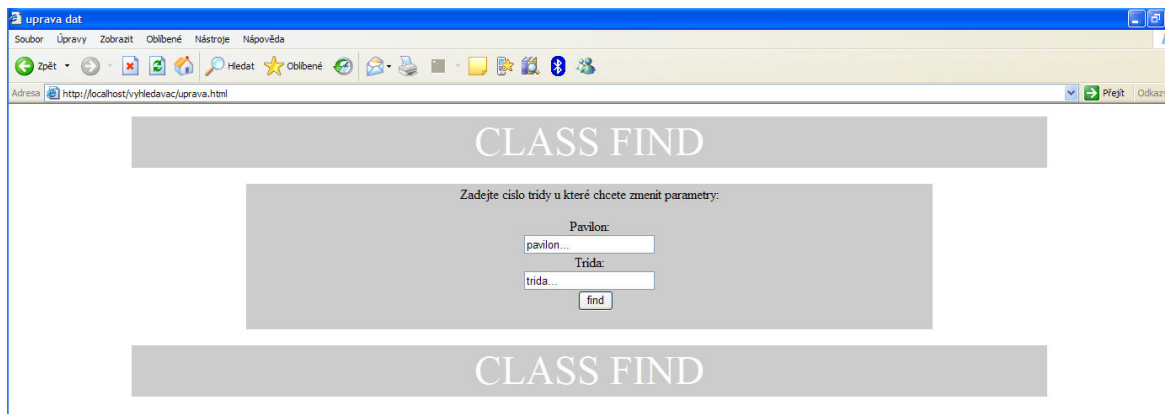


Obrázek 37 - savedata.php

Po uložení všech dat se zobrazí šedý rámeček s informací o uložení do databáze a tlačítkem pro návrat na hlavní stránku.

uprava.php

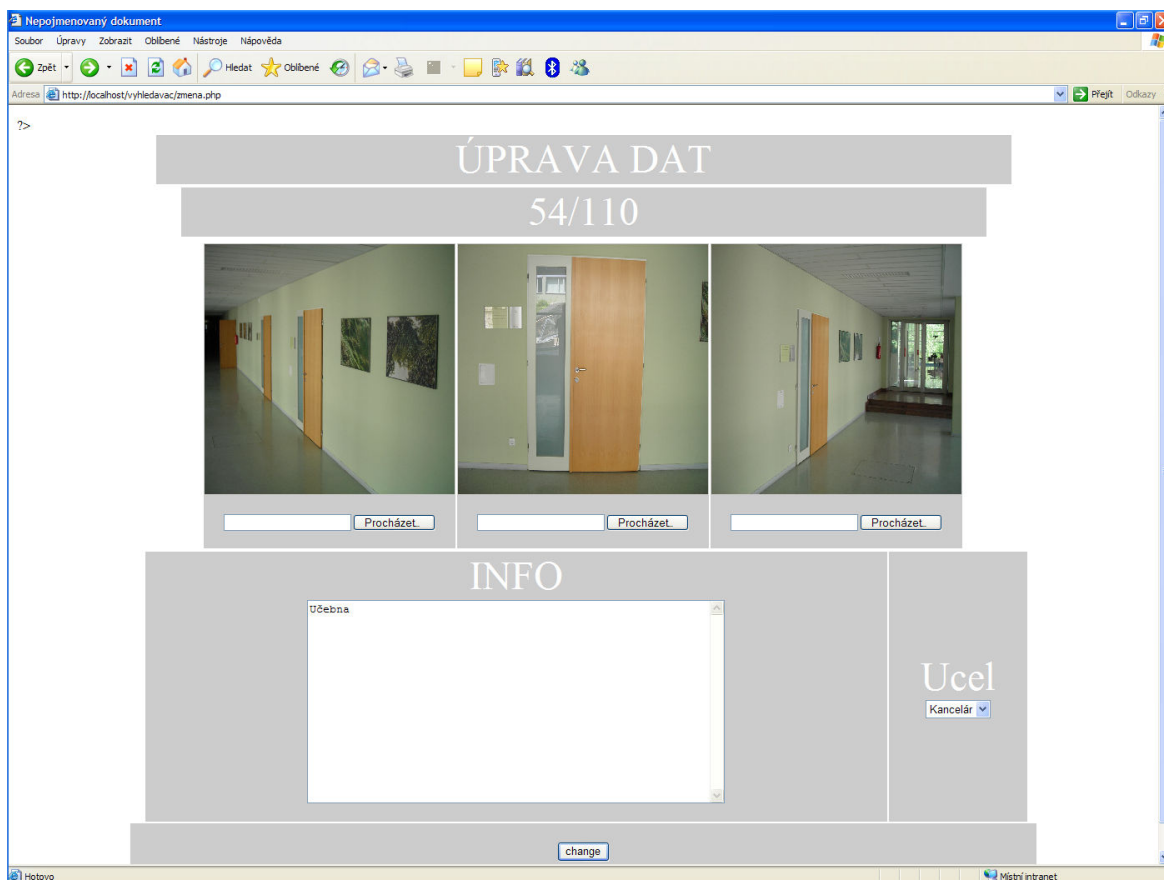
Pomocí této stránky se do následujícího skriptu odešlou potřebné údaje (pavilon, patro a číslo třídy) do následujícího skriptu. Tato stránka slouží pouze k zadání těchto informací.



Obrázek 38 - uprava.php

zmena.php

Tato stránka nabízí možnost upravovat stávající záznamy v databázi. Po připojení k databázi a výběru zvoleného záznamu se zobrazí informace o místnosti s možností tyto upravovat. Je možné změnit fotografie, informace o obsazení i účelu místnosti prostým nahrazením textu nebo fotografií. Pokud ovšem zůstanou některé formuláře nevyplněny, záznamy týkající se těchto dat nebudou v databázi nahrazeny a v databázi zůstanou informace nezměněny, což zajišťuje následující skript (*upraveno.php*). Jak již bylo předesláno, je úprava údajů velmi jednoduchá a intuitivní.



Obrázek 39 - zmena.php

upraveno.php

Tento skript upraví změněné záznamy v databázi a to tak, že pokud jsou údaje změněny, skript tyto informace o místnosti nahradí, pokud nejsou změněny (formuláře zůstanou prázdné) ponechá skript tato data beze změny. Tohoto je dosaženo jednoduchou podmínkou, viz dále.

Po úspěšném upravení záznamů v databázi je zobrazen šedý rámeček s informací o úspěšné změně dat a tlačítkem pro návrat zpět na hlavní stránku aplikace (*main.html*).

Ve skriptu je vložen společný skript *common.php* obsahující funkci **vetsi()**, která zajišťuje přidání nuly před čísla místností menší než 10.

Ukázka kódu stránky

```
include 'common.php';
$query = "SELECT photo,photo1,photo2,type,obsazeni FROM u".$pavilon." WHERE level
=".$level." and class =".vetsi($class);
$help = mysql_query($query);

$radek = mysql_fetch_array($help);
if($photo <> ""){
$query = "UPDATE u".$pavilon."SET photo ='$photo' WHERE level ='$level' and class
=".$vetsi($class);
$help = mysql_query($query,$link);
}
if($photo1 <> ""){
$query = "UPDATE u".$pavilon."SET photo1 ='$photo1' WHERE level ='$level' and class
=".$vetsi($class);
$help = mysql_query($query,$link);
}
```

common.php

Tento společný skript obsahuje tři funkce pro práci s řetězci. A to funkci **level()**, která vyjme z řetězce označujícího místnost pouze číslo patra, ve kterém se místnost nachází, funkci **room()** vybírající číslo místnosti z tohoto řetězce a funkci **vetsi()** popsanou již dříve.

Tento skript nemá žádnou grafickou reprezentaci. Slouží pouze jako úložiště funkcí používaných ve více skriptech projektu.

Ukázky kódu stránky

```
<?php
function level($a){
$b = str_split($a);
return $b[0];
}??>
<?php
function room($a){
$b = str_split($a);
return $b[1].$b[2];
}
?>
```

```
<?php
function vetsi($a){
If($a < 10){
return "0".$a;}
else{
return $a;}
}
?>
```

ZÁVĚR

Internet je v současnosti nejvíce používaným komunikačním nástrojem, který představuje rozsáhlou databanku informací ze všech oblastí. Pro je snahou každé instituce mít co nejlépe vypracované WWW stránky. Jedná-li se o rozsáhlý areál, ve kterém je instituce umístěna, pak průvodce tímto areálem může být zejména pro nově příchozí výhodou.

Za tímto účelem byl podle zadání práce vytvořen webový navigátor prostorami Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně, který má pomoci při orientaci v areálu U5 na Jižních Svazích. V rámci tohoto navigátoru bylo vytvořeno jednoduché uživatelsky přístupné prostředí pro vyhledávání jednotlivých místností.

Součástí navigátoru je přehledná mapa každého patra budovy, která obsahuje jednoduchou legendu. Každý pavilon na mapě je vyznačen jinou barvou. Na jednotlivé místnosti je po zadání označení místnosti na hlavní stránce aplikace ukázáno odkazovým červeným čtvercem. Tento čtverec sloužící jako hypertextový odkaz je ve stránce umístěn na přesnou pozici, jejíž souřadnice jsou uloženy v databázi.

Dále dle zadání tento navigátor obsahuje jednoduchou a přehlednou editační stránku, kde je možné upravovat každý záznam v tabulce databáze. Byla též naprogramována stránka, která umožňuje přidávání záznamů do databáze.

Přestože pro uživatele, který používá počítač, je ovládání této aplikace velmi jednoduché a intuitivní, je v praktické části této práce vypracován stručný manuál pro ovládání těchto stránek. Tento manuál byl měl usnadnit práci s navigátorem úplným začátečníkům v práci s počítačem.

ZÁVĚR V ANGLIČTINĚ

The Internet is presently the most widely used communication tool. It consists of large databank of information from all areas. Therefore it is one of the priorities to have preferably original and user friendly WWW pages for every organization. If the organization is located in large area, the guide of this area might be an advantage for those new visitors.

For that purpose, the web navigator of rooms of Faculty of Applied Informatics, Tomas Bata University in Zlin, was created. It should help to find the given room in the campus denoted as U5 located in town district called Jizni Svahy. The navigator offers easy-to-understand user friendly environment for the room searching.

In the navigator, there is general map of every floor with a short legend. Each pavilion is in different color in the map. When the room is entered to the box, there is pointed on it by red hyperlink square. This square is de facto the hyperlink and it is located in the exactly given position. The coordinates of the position are saved in the database.

Moreover, the navigator includes easy-to-understand and overview editing WWW page. It is possible to modify each entry in the database table. In addition, the WWW page used for adding new entries in the database was created.

Even if it is very easy to work with this navigator for users who are familiar in work on PC, there was created a short manual of it. The manual is a part of the practical part of the thesis. This manual should help the beginners in work on PC with the navigator.

SEZNAM POUŽITÉ LITERATURY

Bibliografie

- [1] Castagnetto J. a kol (2004). Programujeme PHP. Computer Press Brno 2004, ISBN 80-7226-310-2.
- [2] Staníček P. a kol. (2006). CSS - hotová řešení. Computer Press, ISBN 80-251-1031-1
- [3] Dellwing I. (2002). Příručka tvůrce webu HTML 4. Grada, ISBN 80-247-0297-5.
- [4] Schlossnagle G. (2004). Pokročilé programování v PHP5, Zoner Press, ISBN 80-86815-14-5.
- [5] Bulger, B., Greenspan, J., Wall, D. (2004): MySQL/PHP Databases Applications. Willey Publishing, USA, ISBN 0-7645-4963-4.

Webové zdroje

- [6] *Tvorba-webu.cz* [online]. c2003-2006 [cit. 2007-04-15]. Dostupný z WWW: <<http://www.tvorba-webu.cz/php/funkce.php>>.
- [7] *Jak vložit stejný kód do více stránek* [online]. c2004 [cit. 2007-04-15]. Dostupný z WWW: <<http://www.webtvorba.cz/php/jak-vlozit-kod.html>>.
- [8] ONDRA, Marek. *Práce s obrázky v databázi MySQL* [online]. 24.10.2002 [cit. 2007-04-15]. Dostupný z WWW: <<http://interval.cz/clanky/prace-s-obrazky-v-databazi-mysql/>>.
- [9] ŠITYCH , Petr. *Dreamweaver 8* [online]. 2007 [cit. 2007-04-16]. Dostupný z WWW: <<http://www.cgg.cvut.cz/~apg/apg-referaty05/ch11s02.html>>.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

HTML Hypertext Markup language (Hypertextový značkovací jazyk)

PHP Personál Home Page (Osobní Domovská stránka)

SQL Structured Query Language (Strukturovaný Dotazový Jazyk)

CSS Cascade Stylesheet (Kaskádové styly – rozšíření HTML)

SEZNAM OBRÁZKŮ

Obrázek 1 - Reprezentace HTML dokumentu v prohlížeči	12
Obrázek 2 - Formátování textu (nadpisy)	16
Obrázek 3 - Seznamy UL, OL a LI	18
Obrázek 4 - Ukázka tabulky.....	20
Obrázek 5 - HTML formuláře.....	21
Obrázek 6 - Návrh tabulky.....	35
Obrázek 7 - číselné datové typy v SQL.....	35
Obrázek 8 - Textové datové typy v SQL	36
Obrázek 9 - datum a čas v SQL	36
Obrázek 10 - Klient SQL.....	38
Obrázek 11 - Vytvoření databáze.....	38
Obrázek 12 - Vytvoření tabulky v SQL	39
Obrázek 13 - Vkládání hodnot do tabulky	40
Obrázek 14 - Zobrazení dat v tabulce	40
Obrázek 15 - Macromedia Dreamweaver box.....	45
Obrázek 16 - Uvítací okno Dreamweaveru.....	46
Obrázek 17 - Výběr dokumentu z předlohy.....	48
Obrázek 18 - Okno editoru	49
Obrázek 19 - Vkládací lišta.....	50
Obrázek 20 - Vkládací lišta.....	51
Obrázek 21 - Stavový řádek.....	52
Obrázek 22 - Nástrojová lišta CODE.....	53
Obrázek 23 - Inspektor vlastností	53
Obrázek 24 - Panel souborů.....	54

Obrázek 25 - Automatické doplňování příkazů	55
Obrázek 26 - Paletka barev	55
Obrázek 27 - Validace podle norem V3C	56
Obrázek 28 - Správnost reprezentace v prohlížečích.....	56
Obrázek 29 - Tvorba šablon.....	56
Obrázek 30 - Struktura databáze.....	60
Obrázek 31 - Detail údajů o místnosti	61
Obrázek 32 - Struktura stránek	61
Obrázek 33 - main.html	62
Obrázek 34 - mapa.php	66
Obrázek 35 - zobrazeni.php	67
Obrázek 36 - update.php	70
Obrázek 37 - savedata.php	71
Obrázek 38 - uprava.php.....	72
Obrázek 39 - zmena.php	73

SEZNAM PŘÍLOH

Příloha I – WWW stránky a databáze na CD - ROM

