Jihočeská univerzita v Českých Budějovicích

Pedagogická fakulta

Katedra fyziky

Využití prostředí PHP pro tvorbu odborného výukového textu

Bakalářská práce

Vedoucí práce: RNDr. Vítězslav Straňák

Vypracoval: Miroslav Vacikar

Anotace

Jedním z programovacích jazyků, který splňuje požadavky v oblasti vytváření webových aplikací je nástroj Personal Home Page (PHP). V bakalářské práci je popsáno, jaké základní možnosti programovací jazyk PHP nabízí. Je průvodcem od jeho výběru, přes zvolení instalace, vysvětlení základních pojmů a představení základních příkazů. V rámci praktické části bakalářské práce byla vypracována zcela nová, nestandardní webová aplikace, která umožňuje on-line změnu odborného textu, uveřejněného na www stránce. Vlastní programování nebylo pouze o řešení problémů s jednotlivými příkazy a zdrojovým kódem, ale především o hledání řešení v obecné rovině - hledání nápadu. Hlavními cíly bylo vytvoření aplikace, která by, mimo své praktičnosti, pracovala bez použití speciálních nástrojů a kladla by pouze minimální nároky na obsluhu, což je úspěšně splněno.

Abstract

Personal Home Page (PHP) is software which meets the requirements from area of web-application and programming. The basic description of the program and its possibility of applications are described in the bachelor thesis, too. The thesis brings the guide from choice of the program, selection of installation, explanation of basic conception and basic commands. In the frame of practical part of the work new and unique web application was developed for on-line editing of scientific text located at web-page. The main idea of the work is not introduced by solving of particular problems with PHP commands and codes but looking for a general solution (searching for idea). The main aim was developing of application, which is basically simply and can work without special tools and low service requirement. This point of the work was successfully fulfil.

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě - elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách.

V Protivíně 20. dubna 2007 Miroslav Vacikar

podpis

Rád bych na tomto místě poděkoval RNDr. Vítězslavu Straňákovi za aktivní přístup, pomoc a odborné vedení při vypracování mé bakalářské práce.

Obsah

1. ÚVOD

V dnešní moderní době internet zasahuje čím dál více do našeho běžného života a přichází s ním do styku stále více lidí. S rozvojem rychlého a většinou i snadného připojení domácností se zdá, že jeho dalšímu rozvoji již nic nebrání. Dnes již nejde pouze o získávání informací, internet toho může nabídnout mnohem více – ať se již jedná o sdílení souborů, poslech hudby, sledování filmů či přímo televizních stanic, velkého rozmachu se dočkalo i IP telefonování, snadnější přístup na státní úřady a podobně.

S rozvojem internetu se rozvijí i jeho podoba. Stále záleží na funkčnosti jednotlivých stránek a uživatelském pohodlí, ale více záleží i na jeho vlastní podobě – a mnoho firem či různých podnikatelských subjektů neváhá vynaložit značné prostředky či úsilí právě za své webové stránky. Do popředí se dostávají pojmy jako internetová bezpečnost, webové prohlížeče a v neposlední řadě i aplikace, které slouží k vytváření WWW stránek.

V současné době je stále patrnější trend používání poměrně silných programovacích jazyků, podporovaných možností rychlého přenosu dat a výrazného zvětšení výpočetní rychlosti počítačů. Tyto mají mnohé výhody, které převažují nad nevýhodami. Jednou z diskutabilních nevýhod může být náročnost kladená na uživatele, neboť tvorba WWW stránek pomocí těchto aplikací je jistě složitější, než například používání Microsoft Publisher, který bývá součástí kancelářského balíku Microsoft Office.

Jedním z programovacích jazyků, určených především pro tvorbu internetových aplikací, je i PHP. Zkratka PHP je na internetu poměrně častá, zcela jistě se dá prohlásit i za populární. PHP – neboli Personal Home Page (Tools) patří do skupiny programů, které se spouštějí na serverech. Dnešním požadavkem je, že server musí pružně reagovat na požadavky uživatelů a zpřístupňovat informace, které se v čase mění. Vše se odehrává na webovém úložišti (kde jsou uloženy data stránek), kde se skript nejprve provede, poté odešle výsledek prohlížeči (znamená to, že nejprve spočítá kolik je 10-5 a následně prohlížeči odešle pouze číslo 5).

Tato bakalářská práce pojednává o vývoji hypertextových stránek s využitím programovacího jazyka PHP (v úvodu práce je i stručná historie internetu z pohledu jednotlivých jazyků na vytváření WWW stránek).

Bude popsán vývoj programovacího jazyka společně s jeho historií, popis instalace včetně instalace webového serveru Apache, vytváření zajímavých hypertextových stránek, výhled do budoucnosti tohoto jazyka a nedílnou součástí této práce bude i praktická ukázka webové aplikace. Smyslem této práce není popsat všechny příkazy PHP, tyto jsou jistě kvalitněji popsány v knižních publikacích, spíše by tato práce měla být návodem a průvodcem pro začátečníky, kteří se snaží získat více informací o tomto jazyku a případně jej touží i vyzkoušet.

V rámci praktické části bakalářské práce byla vypracována zcela nová, nestandardní webová aplikace, která umožňuje on-line změnu odborného textu, uveřejněného na www stránce. Vzhledem k tomu, že byla zhotovena na půdě Pedagogické fakulty Jihočeské univerzity České Budějovice, byla ukázkově provedena na fyzice plazmatu. Tento obor se rychle rozvijí a je intenzivně studován, bohužel v ČR není dostatek aktuální odborné literatury. Překladem odborné literatury z cizího jazyka (v našem případě Wissenspeicher) dochází k nepřesnostem a chybám. Vytvořená webová aplikace umožní opravu překladu – tato oprava probíhá on-line (možnost zapojení více pracovníků), bez použití speciálních aplikací a jen s malými nároky na obsluhu.

2. HISTORIE INTERNETU

V roce 1990, kdy byla služba World-Wide Web poprvé spuštěna na půdě výzkumného centra CERN, jsme si vystačili s pouhými třemi technologiemi.

První z nich byl jazyk HTML (HyperText Markup Language), který sloužil k zápisu stránek. HTML je dodnes (spolu s XML a následovníkem HTML - XHTML) ústřední technologií Webu, existuje ve verzi 4.01 (je stejný s jazykem XHTML 1.1), nicméně je stále zpětně kompatibilní s původní jednoduchou verzí HTML.

Druhou nezbytnou technologií je přenosový protokol HTTP (HyperText Transfer Protocol), který zajišťuje přenos HTML-stránek z WWW-serveru (místa, kde jsou stránky uložené) do prohlížeče. Původní verze HTTP 0.9 byla velmi jednoduchá. V důsledku zvýšených požadavků postupně vznikly nové verze HTTP 1.0 a 1.1 HTTP 1.1, které se dnes stávají standardem a podporují je všechny nejvýznamnější WWW-servery a prohlížeče. Dnes se k HTTP přidává další protokol SSL, který je shodný s protokolem HTTP 1.1, ale odesílaná data šifruje.

Třetí technologií nezbytnou pro implementování služby WWW jsou URL (Uniform Resource Locator). Každý objekt přístupný na Webu má svojí jedinečnou URL-adresu, která slouží k vytváření odkazů na daný objekt.

Z dnešního pohledu spojení těchto tří technologií nenabízí mnoho - umožňuje pouze prohlížení elektronických dokumentům, které jsou provázány systémem odkazů. Jak se tedy ubíral vývoj dál k dnešní podobě Webu, který je interaktivní a reaguje na požadavky uživatele?

První inovací byla možnost automatického generování stránek, které obsahují informace proměnlivé v čase. HTML-stránka je soubor uložený na disku WWW-serveru, který má své URL. Nic však nebrání tomu, aby URL ukazovalo na nějaký spustitelný soubor (program), který vygeneruje HTML - stránku. Tato stránka pak může obsahovat aktuální informace. Spustitelný soubor je vyvoláván WWW – serverem, proto bylo zapotřebí rozhraní, které by definovalo způsob spuštění programu a předávání dat mezi WWW-serverem a programem. Toto rozhraní se jmenuje CGI (Common Gateway InterFace). Programům, které generují HTML-stránky, se proto často říká CGI-skripty.

CGI-skripty měly jednu nevýhodu - byly pomalé. Proto se na trhu objevily nové technologie, které byly rychlejší. Těmi technologiemi byly v roce 1996-97 **SSJS** a **ASP**.

SSJS (Server Side JavaScript), dříve LiveWire byla serverová podoba původně jen klientského JavaScriptu. Jsou velice podobné CGI-Skriptům, ale jsou zde dva základní rozdíly:

- jsou psány rovnou do HTML - stránky (<server></server>)
- jsou mnohem rychlejší a jednodušší. Technologii SSJS vyvíjí autor JavaScriptu (tedy klientského, slavnějšího) Netscape.

Posléze Microsoft vyvinul a uvedl na trh **ASP** (Active Server Pages). ASP jsou obdobou SSJS. Jako programovací jazyk je možno využít VBScript nebo JScript, což je jazyk od firmy Microsoft, který je podobný JavaScriptu. Systémy nejsou kompatibilní (např. ASP používá jiné značky (<%…%>) a další). Kromě VBScriptu a JScriptu je možno v ASP používat další jazyky, které dodávají jiné firmy (Perl, REXX, Python).

SSJS i ASP mají jednu velkou, společnou nevýhodu - jsou to komerční produkty, které nejsou levné a jejich použití je navíc svázáno s použitím WWW serveru dané firmy. ASP navíc funguje pouze na platformě (operačním systému) Windows – v případě rozhodnutí přechodu na platformu Linux se musí celá aplikace znovu naprogramovat v jiném jazyce.

Všechny tyto a mnohé další nedostatky odstraňuje systém PHP, který v současné době přitahuje mnoho pozornosti. Princip použití PHP je obdobný jako u SSJS a ASP. Na rozdíl od nich je však šířen celý produkt jako freeware. Nejen z tohoto důvodu byl využit při tvorbě praktické části bakalářské práce.

3. ZAČÁTKY PHP

Když v roce 1995 začal Lerdofr Rasmus pracovat na vývoji nástroje **PHP/FI** (Person Homepage Tools/Form Interpreter), nemohl v žádném případě předpokládat, že tento krok nakonec vyústí ve vznik jazyka PHP v takové podobě, jak je znám dnes. První verze PHP/FI byla kolekcí skriptů v jazyce Perl – byl to vlastně jazyk podobný jazyku Perl, jenž byl schopen zpracovávat odesílání dat z formulářů. Novému nástroji však chybělo mnoho základních užitečných jazykových konstrukcí.

V roce 1997 byl nástroj PHP/FI přepracován na **PHP/FI 2**. V té době se o vývoj staral pouze Rasmus. Nové verze si po jejím uvolnění v listopadu téhož roku všimli Gutmann Andi a Suraski Zeev, kteří pátrali po jazyce, jenž by jim usnadnil vývoj řešení elektronického obchodování. Jednou z nejzajímavějších vlastností byl způsob, jímž byl implementován cyklus „while“. Ručně napsaná lexikální čtečka procházela skript, a když narazila na klíčové slovo „while“, zapsala si jeho pozici v souboru do paměti. Na konci cyklu vyhledala čtečka uloženou pozici a celý cyklus byl znovu načten, vykonán.

Zeev a Andi se rozhodli skriptovací jazyk zcela přepracovat. Spojili se s Rasmusem a společně připravili verzi **PHP 3.** S novou verzí vznikl i nový název – **Personal Hypertext Preprocesor.** Autoři chtěli zdůraznit, že PHP je nyní jiným produktem a není určen pouze pro osobní potřebu, navrhli a implementovali nové rozšíření API. Toto nové rozhraní API umožňovalo snadnou podporu dalších rozšíření. Ta se dala využít například ke spojení s databází, ke kontrole pravopisu a k dalším procesům.

Nicméně ani tato verze nebyla finální a vývoj pokračoval. Na konci roku 1998 Zeev a Andi dospěli k názoru, že by bylo možné napsat skriptovací jazyk mnohem lépe. Skriptovací jazyk PHP 3 zpracovával skript během čtení, verze **PHP 4** už přišla s novým přístupem – „nejprve překlad, potom zpracování“. Skriptovací stroj nepřekládá do strojového kódu, ale do kódu bytového, jenž je pak zpracován pomocí stroje **Zend Engine** (**Ze**ev a A**nd**i). Stroj Zend se stal srdcem prostředí PHP 4. Díky novému způsobu zpracování skriptů se výkon jazyka značně zvýšil, aniž by byla narušena zpětná kompatibilita s jazykem PHP 3. Mezi další zdokonalení patří vylepšené rozhraní API s vyšším výkonem, dále vrstva serverové abstrakce, díky níž lze PHP provozovat na většině oblíbených webových serverů, a mnoho dalších.

I poté se začaly množit požadavky na obecnější objektově orientovaný přístup. Nakonec Andi přišel s myšlenkou přepracování objektově orientované části skriptovacího stroje Zend a byla zahájena diskuze o budoucnosti PHP. Základní funkce **PHP 5** zůstává stejná. Jazyk byl však proti zveřejněnému dokumentu rozšířen o mnoho dalších funkcí, jiné funkce byly změněny či odstraněny.

Od verze PHP 5 se očekává další rozšíření podílu jazyka PHP na poli webového vývoje. Tato verze přináší podle některých zdrojů revoluční změny, ale obsahuje rovněž mnoho funkcí, jež PHP činí rozhodující platformu vývoje webu. V současnosti (říjen 2006) je poslední verzí PHP 5.1.6 (www.php.net).

3.1 Instalace PHP a Web Serveru Apache

PHP je program volně šiřitelný, umístěný např. na webové adrese www.php.net. Rovněž bývá šířen jako součást standardních instalací operačního systému Linux (včetně nejpoužívanějších RedHat či Suse Linux). Další velkou výhodou je instalace nejen na PC s operačním systémem Linux, ale i na PC s operačním systémem Windows (od verze Windows 95) bez ohledu na to, zda se jedná o serverovou či klientskou instalaci operačního systému. Dokonce je možné stránky vyvíjet na počítači s operačním systémem Windows a stránky provozovat na serveru s instalovaným operačním systémem Linux. Změna operačního systému si však žádá drobné úpravy zdrojového kódu (například jiný sklon lomítka). Kvalitu programovacího jazyka dosvědčuje i skutečnost, že verze fungující pod Windows XP je plně funkční i pod novým operačním systémem Windows Vista – a to bez jakékoliv aktualizace.

Instalace PHP není nejjednodušší (pokud si nechceme PHP instalovat, můžeme využít služeb řady firem nabízecích umístění skriptů na serverech, které PHP podporují – tzv. webhosting).

3.1.1 Instalace Web Serveru Apache

Před vlastní instalací jazyku PHP je nutné mít nainstalovaný webový server. K dispozici je více možností, nejvíce obvyklé jsou:

- Internet Information Server – tento je u serverové verze operačního sytému Windows standardně nainstalovaný, u klientské verze je nutné jej doinstalovat z instalačního CD Windows (Start → Ovládací programy → Přidat nebo odebrat programy → Přidat nebo odebrat součásti systému → Internetová informační služba).
- Web Server Apache – volně dostupná verze (freeware), ke stažení například na adrese www.apache.org (poslední verze k datu říjen 2006 je Apache 2.2.3) – údajně je v současnosti nejpoužívanějším a nejlepším serverem. Na uvedené adrese je možné najít dokumentaci, fórum k danému tématu a další podporu programu. Mezi jeho výhody patří snadná dostupnost, velká konfigurovatelnost a výkon.

V našem případě byl zvolen Web Server Apache, především pro zjištění, jakých výsledků bude při tvorbě www stránek dosaženo použitím freewareových aplikací. Vlastní instalace je popsána v manuálu staženého souboru.

3.1.2 Výběr instalace PHP

Pokud jsme nezískali instalaci jako součást většího instalačního balíku, je nejsnazší cestou si potřebnou verzi stáhnout přímo z internetu. Například na adrese www.php.net v sekci download najdeme několik distribucí (podle použité platformy či instalované verze), rovněž se zde nachází dokumentace k jazyku PHP a další užitečné soubory.

Instalace se řídí dle připojeného manuálu, kde je podrobně popsána. Vlastní instalace je komplikovaná a vytváří mnoho prostoru pro chyby, které se později obtížně hledají. Z tohoto důvodu bylo přistoupeno k jiné možnosti nainstalování tohoto produktu (potažmo Web Serveru Apache). V literatuře je často uváděn nástroj **PHPTriad** (freeware) a je ke stažení například z adresy www.studna.cz (verze PHPTriad 2.2). Obsahuje kompletní sadu nástrojů k instalaci prostředí pro vývoj a aplikaci PHP na systémech Windows. Instaluje a nastaví PHP, Apache a MySQL. Jeho vývoj byl ale ukončen v roce 2002 a je tedy značně zastaralý. Jako novější nástroj

se jeví produkt **PHP Home**, který měl být nástupcem PHPTriad. Je rovněž distribuován jako freeware (http://sourceforge.net). Ani tento nástroj však není aktuální, poslední verze PHP Home 2.3.4. je z 29. 7. 2004.

Velkým překvapením byl nález české verze produktu s názvem **Internetový server Light** (viz: http://www.slunecnice.cz/product/Intranetovy-server/ autor Miroslav Ponkrác) a aktuální verze (říjen 2006) - **Complex Web Server-1.2.10.** Tento produkt obsahuje PHP 5.1.1., Web Server Apache 2.0.55 a MySQL 4.1. Instalace je bezproblémová, s popisem a možnostmi výběru. Produkt je doporučen pro instalaci na platformu Windows, instalace kompletních nástrojů, funkční, šířen jako freeware.

3.2 Psaní PHP skriptů

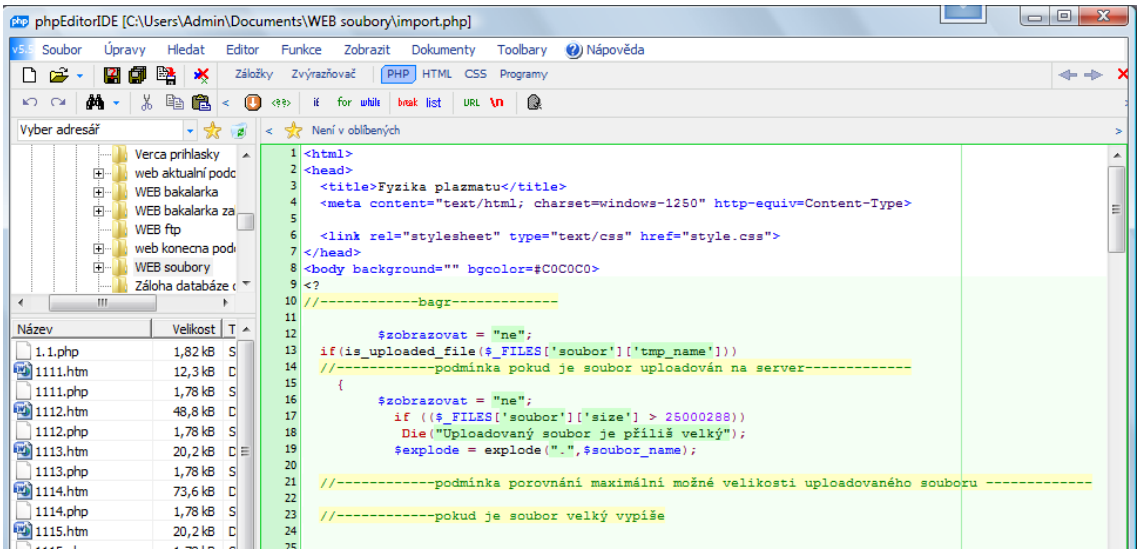
Nebude zde rozebírán programovací jazyk jako takový, ale jeho vytváření a editace. Moderní HTML editory většinou práci s PHP aktivně nepodporují (pokud je podporují, nejsou obecně doporučovány). PHP stránky nelze psát v programu FrontPage, protože všechny <?vsuvky?> odstraní (od verze 2003 funkční, přesto není považován za vhodný).

Doporučené editory jsou například **phpEditor** (viz obr. č. 1), **PSPad**, **UltraEdit**, **Vim** či **HomeSite** se zvýrazňováním syntaxe, což psaní významně usnadňuje. Mezi další patří například **PHPEd** – toto je profesionální produkt.

Vytváření web stránek pomocí jazyku PHP je v této práci bráno z pohledu běžného uživatele, tedy používány přednostně aplikace, které jsou přístupné zdarma – freeware (toto PHP 5 a Web Server Apache splňuje). Rovněž vývoj na platformě nabízí verzi pro každého – Linux i Windows. I když nejsme nikým nuceni používat placené nástroje pro úpravu skriptů, právě tyto nabízejí větší uživatelské pohodlí, např. zvýrazňuje syntaxi PHP a i jinak je práce s PHP v těchto editorech výrazně jednoduší. Většímu rozšíření profesionálních nástrojů však stojí v cestě jejich cena (na druhou stranu je nutno uznat, že vycházejí časté aktualizace těchto produktů).

Další možností je např. integrované vývojové prostředí Zend Studio (ZDE – Zend Development Envirinment), zahrnuje editor, ladící program a správce projektů.

Pro prohlížení PHP souborů také nestačí normální „dvojklik“, ale prohlížení přes prohlížeč s adresou začínající http://, např. http://localhost/cesta/jmenosouboru.php atd.



Obr. č. 1 Ukázka prostředí programu phpEditor

V horní části programového prostředí phpEditor se nachází lišty s ovládacími prvky programu. V levé části lze zadat adresář a konkrétní webová stránka, která je následně otevřena v pravé části programového prostředí. Zde se zobrazí zdrojový kód, kdy zobrazení kódu se řídí pravidly, které je možno měnit (zvýraznění syntaxe kódu, písmo, barva, kódování apod.).

3.3 Komentáře

Ačkoliv jsou často opomíjeny, představují důležitou část zdrojového skriptu a rozhodně by v něm měly být zahrnuty. Komentář (obr. č. 2) slouží nejen k tomu, aby se ve vytvořeném skriptu dokázal orientovat i někdo cizí, ale i pro orientaci autora. Některým uživatelům, kteří se nezabývají programováním soustavně, může činit problém jejich vlastní skript, ke kterému se vrátili po delším čase. Rovněž nemusí celý skript pocházet od jediného autora, pak není důvod proč se opět nespolehnout na části programů, které jsou již funkčně ověřené jen proto, že v dané chvíli nepochopíme význam.

Zápis komentáře:

- Způsob ve stylu jazyka C:

```
/* To je komentář ve stylu jazyka C
 * který pokračuje na dalších řádcích
 * až do ukončující značky uvedené na následujícím řádku
 */
```

- Způsob ve stylu jazyka C++:

```
// Toto je jednořádkový komentář ve stylu jazyka C++.
```

- Způsob ve stylu příkazového prostředí:

```
# To je jednořádkový komentář ve stylu příkazového prostředí.
```

```
<?
function overheslo ($link, $uzivatel, $heslo)
{
  /*
    funkce ověří heslo proti datům v tabulce hesel
    vyžaduje:
    $link - existující spojení na databázi
    $uzivatel - id ověřovaného uživatele
    $heslo - nešifrovaná podoba hesla
    vrací:
    TRUE v případě, že daný uživatel má dané heslo,
    FALSE ve všech ostatních případech
  */
  $vysledek=mysql_query("select * from hesla where uzivatel=$uzivatel and heslo='".$heslo."'",
$link);
  return (boolean) mysql_num_rows($vysledek);
}
?>
```

Obr. č.2 Ukázka víceřádkového komentáře kódu

3.4 Proměnné, datové typy a konstanty

3.4.1 Proměnné

Proměnná je místo v paměti počítače, která odkazuje na pojmenovanou hodnotu. Například, pokud si budeme chtít zapamatovat na určitou dobu, po kterou budeme provádět určitý kus kódu, nějakou hodnotu, uložíme ji do proměnné a tuto hodnotu libovolně pojmenujeme. Název by měl být pokud možno výstižný, jednoznačně identifikující jeho poslání.

Důležité je si zapamatovat, že každá proměnná, respektive její název musí začínat znakem „$". Proměnné není nutné v PHP předem deklarovat, vytvoří se v okamžiku

použití ve skriptu. Pokud proměnné nepřiřadíme hodnotu, obsahuje hodnotu odpovídající prázdnému řetězci.

- Příklady platných názvů proměnných (musí začínat písmeny A-Z, a-z či podtržítkem _) :

```
$a123
$_Abc
$ABC
```

- Příklady neplatných názvů proměnných:

```
$123
$*ABC
```

- Jak bylo řečeno, není potřeba proměnné ani jejich typy deklarovat s předstihem:

```
$PI=3.14
$poloměr = 5
$obvod = PI  * 2 * $poloměr; // obvod =  * d
```

Z ukázky je patrné, že proměnné nejsou deklarovány před místem svého prvního použití. Rovněž je zcela ignorována skutečnost, že proměnná `$PI` je reálné číslo, zatímco proměnná `$poloměr` je číslo celé. Přesto program funguje a vrací správné výsledky.

3.4.2 Základní datové typy

PHP obsahuje tři základní druhy datových proměnných a několik odvozených proměnných. Proměnná je deklarována tak, že se před identifikátor proměnné přidá znak „$“ **.**

Celá čísla (integers) jsou celá čísla, jejichž rozsah odpovídá rozsahu -2.147.483.648 až +2.147.483.648

Double - typ double reprezentuje desetinná čísla, zpravidla uložená do 8 bytů. K dispozici máme 15 platných číslic. Můžeme však používat matematickou knihovnu

BC (Binary Code), kterou PHP přímo podporuje. Čísla je možné zadávat také v exponenciálním tvaru.

Řetězce (string) jsou v jazyce PHP posloupnostmi znaků, jež jsou vždy interně zakončeny prázdným znakem (null). Na rozdíl od jiných jazyků se jazyk PHP nespoléhá při výpočtu délky řetězce na prázdný znak, ale pamatuje si délku řetězce interně. To umožňuje například dynamické vytvoření obrázku a jeho odeslání do klientského prohlížeče. Při tvorbě řetězových hodnot ve zdrojovém kódu můžeme použít uvozovky (běžně nazýváme dvojité uvozovky) či apostrofy (někdy označované jako jednoduché uvozovky).

Běžné uvozovky mají ještě další funkci spojenou s určitou formou zápisu proměnných a výrazů, jež do nich lze vkládat.

Znak	Význam
\n	Nový řádek.
\t	Tabulátor.
*	Uvozovka.
\\"	Zpětné lomítko.
\0	Prázdný znak.
\r	Přesun na nový řádek.
\$	Řídící znak (znak změny) $, který v tomto případě není považován za počáteční písmeno proměnné, ale za znak „$".
\{oktalové číslo}	Znak zastoupený oktalovým číslem (například řetězec \70 zastupuje znak 8.
\x{hexadecimální číslo}	Znak zastoupený hexadecimálním číslem (například řetězec \0x30 zastupuje znak 2).

Tabulka č. 1 Význam jednotlivých znaků [1]

Pro názornost:

```
"Výsledek je $výsledek \n";
"Prvek pole s indexem $i obsahuje $pole [ $i ] ."
```

Spojování řetězců – Jazyk PHP používá pro spojování řetězců operátor . (tečka). Díky němu mechanismus vyhodnocující kód nejprve převede hodnoty na obou stranách tečky (tzv. operandy) na řetězce a poté je sloučí.

Apostrofy můžeme používat k ohraničení řetězců, avšak neumožňují využít řídící znaky a dynamické nahrazování proměnných, protože překladač netestuje obsah řetězce na výskyt sekvencí escape, ani na obsah proměnných.

Sekvence	Popis
\'	Apostrof.
\\	Zpětné lomítko – používá se v případech, kdy je potřeba v řetězci použít zpětné lomítko následované apostrofem.

Tabulka č. 2 Řídící sekvence [1]

Tabulka č. 2 obsahuje jediné dvě řídící sekvence podporované v řetězcích ohraničených apostrofem.

Logické hodnoty byly poprvé použity v jazyce PHP 4. Logická proměnná může obsahovat pouze hodnotu true (pravda) nebo false (nepravda).

Prázdné hodnoty (datový typ NULL – prázdná hodnota) mají pouze jednu hodnotu – NULL. Ty vyjadřuje pouze chybějící nebo neznámá data a užitečná je zejména při odlišení prázdných řetězců a prázdných hodnot v databázích.

Prostředky (resources) jsou zvláštním datovým typem zastupujícím rozšíření jazyka PHP. Patří sem například databázové dotazy, otevření souboru, připojení k databázi a mnoho dalších externích typů.

Pole (array) je v jazyce PHP kolekcí dvojic, jež se skládají z klíče a hodnoty. To znamená, že pole mapuje klíče na hodnoty. Indexy pole mohou být celá čísla či řetězce. Hodnotami pole mohou být hodnoty libovolných typů – pochopitelně včetně polí.

Funkce array - pole lze deklarovat pomocí funkce array( ), jež má obvykle tuto podobu:

```
Array ( [ klíč => ] hodnota, [ klíč => ] hodnota, … );
```

Uvedení klíče je nepovinné a není-li klíč zadán, je automaticky použita hodnota o 1 vyšší, než u předchozího prvku (první prvek má automaticky klíč s hodnotou 0). V jedné deklaraci můžeme mísit prvky s klíčem i bez klíče.

Funkce each vrací aktuální prvek (dvojici klíč/hodnota) a přesouvá vnitřní ukazatel pole na další prvek. Jakmile dosáhne konce pole, vrací logickou hodnotu false. Dvojice klíč/hodnota je vrácena jako pole obsahující čtyři prvky: prvek 0 a „klíč“, obsahující klíč, dále prvky 1 a „hodnota“. Poslední prvek obsahuje hodnotu získaného prvku.

K práci s poli a jejich procházení můžeme použít i další funkce, např. current a next (tyto jsou zastaralé a neměli bychom je používat), dále třeba funkce array_walk( ).

3.4.3 Konstanty

V jazyce PHP můžeme pro jednoduché hodnoty definovat názvy. Ty se nazývají konstanty. Jak z názvu samotného vyplývá, hodnota konstanty je po prvotní definici neměnná. Názvy konstant se řídí stejnými zásadami jako názvy proměnných s jediným rozdílem – při jejich uvádění se nepoužívá uvozující znak dolaru ($). Je běžné (platí i v tomto programovacím jazyku), že se názvy konstant píší výhradně velkými písmeny. Sice můžeme používat v názvech malá i velká písmena, ale obecně toto doporučené není. Při odkazech na konstanty pak v kódu nebudeme vyžadovat, aby jejich názvy obsahovaly správnou kombinaci malých a velkých písmen. Na rozdíl od proměnných jsou jednou definované konstanty přístupné v globálním měřítku. Konstanty nemůžeme opětovně deklarovat v nové funkci ani v novém souboru PHP.

Příklad definice konstanty:

```
define (“NÁZEV_KONSTANTY”, hodnota [, rozlišování_velikosti_písmen]);
```

kdy

- “NÁZEV_KONSTANTY” je řetězec,

- proměnná hodnota je platným výrazem v jazyce PHP kromě polí a objektů
- výraz rozlišování_velikosti_písmen je logický (vrací tedy hodnotu true či false) a je nepovinný. Implicitně obsahuje hodnotu true.

3.4.4 Operátory

Jazyk PHP obsahuje tři typy operátorů: unární, binární a ternání.

Unární operátor pracuje s jedním operandem – např. operátory negace. Tyto se objevují před negovaným operátorem, viz tabulka č. 3.

Operátor	Název	Hodnota
!	Logická negace	Hodnota true, je-li výraz vyhodnocen jako nepravdivý. Hodnota false, je-li výraz vyhodnocen jako pravdivý.
~	Bitová negace	U číselných operací je výsledkem bitová negace bitové podoby vyhodnocované hodnoty. U řetězců je výsledkem řetězec stejné délky, v němž je každý znak bitovou negací znaku se stejným indexem ve vyhodnocovaném řetězci.

Tabulka č. 3 Unární operátory [1]

Binární operátor se používá v operacích vyžadujících dva operandy. V jazyce PHP lze binární operace provádět pouze s dvěma operandy stejného typu. Jsou-li typu různého, zajistí skriptovací stroj automatickou konverzi jednoho z nich. Postupuje přitom podle pravidel uvedených v následující tabulce č. 4 (není-li určeno jinak).

Typ prvního operandu	Typ druhého operandu	Provedená konverze.
Celé číslo	Reálné číslo	Celočíselný operand je převeden na reálné číslo.
Celé číslo	Řetězec	Řetězec je převeden na číslo. Pokud je výsledkem reálné číslo, je na reálné číslo převeden i celočíselný operand.
Reálné číslo	Řetězec	Řetězec je převeden na reálné číslo.

Tabulka č. 4 Pravidla automatické konverze binárního operandu [1]

Binární operátory (vyjma operátoru spojení) pracují pouze s číselnými operandy. Jsou-li oba operandy řetězci, logickými hodnotami či prostředky, jsou automaticky převedeny na odpovídající číselné hodnoty. K požadovanému výpočtu dojde poté – zásady převodu uvedeny v tabulce č. 5.

Operátor	Název	Hodnota
+	Sčítání	Součet dvou operandů.
-	Odečítání	Rozdíl dvou operandů.
*	Násobení	Součin dvou operandů.
/	Dělení	Podíl dvou operandů.
%	Modul	Oba operandy jsou převedeny na celá čísla. Výsledek je zbytek z podílu dvou operandů.

Tabulka č. 5 Zásady automatického převodu na číselné hodnoty [1]

Ternání operátor (jediný) – je ?: (otazník, dvojtečka).

Jeho zápis vypadá takto:

```
$a=1;
$zpráva = isset( $a )? 'Proměnná $a existuje.': 'Proměnná $a neexistuje.';
print $zpráva;
```

Operátor vyhodnotí podmínku pravdivosti a zjistí, zda vrací hodnotu true či false. Je-li výsledkem vyhodnocení hodnota true, je vykonána první část, v opačném případě je vykonána část druhá.

V našem příkladě se objeví vypsaná zpráva:

Proměnná $a existuje.

V tabulce č. 6 je uveden seznam logických výrazů a operátorů.

Operátor	Příklad	Popis
==	$a==$b;	Rovnost.
!=	$a ! $b; $a <> $b;	Nerovnost.
>	$a > $b;	Proměnná ‘a’ je větší než proměnná ‘b’.
<	$a < $b;	Proměnná ‘a’ je menší než proměnná ‘b’.
>=	$a >= $b;	Proměnná ‘a’ je větší nebo rovna proměnné ‘b’.
<=	$a <= $b;	Proměnná ‘a’ je menší nebo rovna proměnné ‘b’.
&&	($a>1 && $b<10);	Logický součin.
||	($a>1 || $b<1);	Logický součet.

Tabulka č. 6 [1]

3.5 Jazykové konstrukce

Jazyk PHP zvládá mnoho nejběžnějších konstrukcí dostupných i v jiných programovacích jazycích, např. v jazyce C. Patří sem příkazy if, else, for, switch, while, do..while, break a continue.

Jakmile budeme chtít zařídit, aby naše stránky byly interaktivní (dynamické – chovají se podle aktuální situace), neobejdeme se bez příkazů sloužících k větvení programu. Tyto programy porovnávají hodnoty proměnných a podle výsledku porovnání zvolí jednu ze dvou či více možností, jak dále pokračovat ve vykonávání kódu.

3.6 Příkazy pro větvení programu

3.6.1 Příkaz if

Zápis příkazu: if (výraz) příkaz;

Pokud potřebujeme otestovat, zda proměnná obsahuje nějakou určitou hodnotu či naopak zda je od této hodnoty odlišná, použijeme tento příkaz. Jeho doslovný význam je „když“. Vícenásobné rozhodování o pokračování vykonávání kódu obstarává příkaz **else**, který je možné použít v jednom příkazu i vícekrát (vztahuje se vždy k nejbližšímu if). Příkaz se provede, když podmínka za if není splněna.

Pro názornost jeden příklad:

```
<?php
$strJmeno = "Mira";
if ($strJmeno == "Mira"):
    echo "Ahoj Miro!" ;
elseif ($strJmeno == "Petr"):
   echo "Ahoj Petre!";
else:
    echo "A kdo jsi ty?";
endif;
?>
```

Dvojice složených závorek stejně jako dvojtečky dovolují zařadit do kódu pro kterýkoli případ (tedy za podmínku if, else či elseif) i libovolně dlouhou sérii různých příkazů, které ale musí být oddělení středníkem. Příkaz **endif** je nutné použít, abychom jasně definovali konec testovaného bloku. Příkazu je možné se vyhnout za použití složených závorek. Měli bychom si také všimnout toho, že testovaná podmínka se v PHP vždy uzavírá do závorek. Další důležitou věcí je, že každý samostatný příkaz se musí oddělit středníkem.

3.6.2 Příkaz switch

Občas budeme potřebovat, aby se dle vyhodnocení logického výrazu (podmínky) dalo lépe vybrat z několika větví kódu, než umožňuje příkaz if (else, elseif). Místo

nepřehledného opakování příkazu elseif autoři jazyka PHP zařadili příkaz přepínače switch neboli příkaz pro mnohonásobné větvení programu.

Příkaz funguje tak, že se vyhodnotí výraz, poté jsou postupně popořadě procházeny hodnoty uvedené za klíčovým slovem case, dokud se nenalezne hodnota shodná s hodnotou výrazu. Je potřeba si zapamatovat, že procházení mezi jednotlivými větvemi příkazu a jejich vyhodnocení trvá, dokud systém nenalezne jeho ukončení nebo nenarazí na příkaz **break**.

Příklad:

```
<?php
$a ~ o;
switch ($a) {
        case o:
                print "a je rovno O<br>";
        case 1:
                print "a je rovno 1<br>";
        case 2:
                print "a je rovno 2";
}
?>
```

Pokud proměnná $a v uvedeném miniskriptu bude mít hodnotu 0, zobrazí se všechny tři hlášení. Pokud proměnná $a bude mít hodnotu 1, zobrazí se poslední dvě hlášení atd. Tento miniskript demonstruje fakt, že pokud je splněna podmínka, všechny následující příkazy budou provedeny bez ohledu na splnění či nesplnění podmínky dané větve.

3.7 Příkazy cyklu

Cyklus neboli smyčka, představuje sekvenci příkazů, kterou program vykonává opakovaně, tedy cyklicky. Aby toto opakování mělo smysl, měly by příkazy uvnitř cyklu provádět něco, co je potřeba provést mnohokrát – tedy například získat od uživatele řadu hodnot či provést určitou pomocnou operaci s každým prvkem pole podobně. Každý cyklus musí být ukončen, a proto je programu sděleno, kolikrát se má tělo cyklu opakovat či jeho opakování vázáno na splnění určité podmínky, která bude

záviset na proměnné, jejíž hodnota se po každém průchodu cyklem bude měnit, až jednou podmínce pro zavedení cyklu nevyhoví.

3.7.1 Příkaz while

Zápis příkazu: while (výraz) příkaz;

Cyklus typu while je nejméně náročný, na druhou stranu má nejmenší komfort, ale i tak nabízí velkou oblast použití. Na počátku každé interakce je vyhodnocen pravdivostní výraz. Je-li vyhodnocen jako pravdivý, bude vykonán blok kódu vnořený do cyklu. Je-li výraz vyhodnocen jako nepravdivý (vrací hodnotu false), bude kód ve vnořeném bloku vynechán.

Někdy je potřeba ukončit průchod cyklem ještě dříve, než bude vykonán celý cyklus. Z tohoto důvodu jazyk PHP obsahuje příkazy na řízení cyklů: **break** a **continue**. Vyskytuje-li se příkaz break samostatně, je ukončen cyklus, v němž je příkaz umístěn. Příkazu break můžeme předat argument v podobě hodnoty, která uvádí, z kolika nadřazených cyklů se má vyskočit:

```
break n;
```

Příkaz break 1 má stejný význam jako samostatný příkaz break. Místo argumentu „n“ můžeme dosadit jakýkoliv platný výraz.

V některých případech můžeme chtít zastavit zpracovávání kódu uvnitř cyklu po určitém příkazu, aby nastal návrat na počátek cyklu a byl spuštěn nový průchod cyklem. V této situaci se používá příkaz continue, který slouží k ukončení zbytku aktuálního opakování a k přechodu na následující opakování. Příkaz continue, stejně jako příkaz break, akceptuje nepovinný číselný argument, který říká, kolik úrovní cyklu se má naráz přeskočit (bez jeho použití bude ukončen aktuální cyklus).

3.7.2 Příkaz do…while

Zápis: do { příkazy, } while (výraz) ;

Tento příkaz je velmi podobný příkazu předcházejícímu. Na rozdíl od příkazu while se však testování podmínky provádí až po průchodu cyklem, což znamená, že cyklus proběhne minimálně jednou. Průběh cyklem je ukončen, pokud podmínka není splněna.

3.7.3 Příkaz for

Zápis: for (výraz_start; výraz_stop; výraz_inter) příkaz;

Cyklus for je jedním z nejsložitějších cyklů v PHP. První výraz je vyhodnocen jednou, bezpodmínečně na začátku cyklu. Na začátku každého opakování musí být vyhodnocen pravdivostní výraz (výraz_stop). Má-li hodnotu true, cyklus pokračuje a zpracovává se kód uvnitř cyklu. V opačném případě je cyklus ukončen. Na konci každého opakování se vyhodnotí výraz (výraz_inter), poté nastává obrátka cyklu.

4. VYUŽITÍ PHP

Po vysvětlení základních věci a příkazů je v následující části věnován prostor praktické činnosti, jako je práce s textovými soubory, nahrávání souborů na server, rozesílání emailových zpráv, apod. Je však mnoho dalších velmi významných oblastí, které v této práci nejsou zahrnuty (například grafika, propojení s databází). Jejich vysvětlení by znamenalo komplexnější přístup k jednotlivým příkazům a jejich použití.

4.1 Práce se soubory

Práce se soubory - to je hlavní přednost PHP. Je možné ukládat, upravit a třeba publikovat na serveru přímo z textových souborů. V praktické části bakalářské práce je práce se soubory využívána a je tedy nutné naučit se používat funkce PHP pro jejich vytvoření, čtení a modifikaci.

4.1.1 Vytvoření nového souboru, otevření existujícího souboru

Ať již potřebujeme vytvořit soubor nový či otevřít stávající pro editaci, použijeme funkci **FOpen** (odvozeno od File Open). Krom souborů na vlastním disku můžeme pracovat i se soubory pomocí adresy URL, tedy pokud máme příslušná oprávnění.

Zápis příkazu:

```
fopen (nazev_souboru, mode [, pouzit_include_patch)
```

První parametr funkce není potřeba komentovat. Parametr mode určuje, jakým způsobem se má soubor otevřít, jakým způsobem s ním hodláme pracovat – přehled parametrů v následující tabulce č. 7:

Parametr	Popis
r	Otevře soubor pouze pro čtení a umístí ukazatel na začátek souboru.
r+	Otevře soubor pro zápis i čtení a ukazatel přemístí na začátek souboru, soubor již musí existovat.
w	Otevře soubor pro zápis nových dat a jeho původní obsah ruší. Pokud soubor daného jména doposud neexistuje, funkce jej vytvoří. Z toho plyne, že tento parametr se použije pro vytvoření nových souborů.
w+	Obdobně jako předchozí hodnota parametru s tím rozdílem, že soubor lze také číst.
a	Otevře soubor pro zápis nových dat a umístí ukazatel na konec souboru. To znamená, že nová data budou připojena ke starým. Pokud soubor zadaného jména neexistuje, bude vytvořen.
a+	Obdoba předchozí hodnoty parametru s tím rozdílem, že soubor lze číst.

Tabulka č. 7 Přehled parametrů funkce

Příklad použití:

```
<?php
$file = fopen (“/home/mira/file.txt”, “r”);
$file = fopen (http://www.mira.cz/, “r”);
$file = fopen (ftp://user:mirda@seznam.cz/”, “w”);
?>
```

4.1.2 Ověření existence souboru

Než začneme otevírat pro editaci již existující soubory, je vhodné se nejprve o jejich existenci přesvědčit. K tomuto slouží funkce **File_Exist**, která své uplatnění najde i tehdy, kdy vytváříme soubor nový a potřebujeme se pouze ujistit, že zvolené pojmenování souboru již nebylo použito.

Příklad zápisu:

```
<?php
if (file_exists(soubor.txt)):
        $file = fopen (“soubor.txt“, “r”);
else:
        echo “soubor Neexistuje”;
endif;
?>
```

4.1.3 Ukončení práce s otevřeným souborem

Pokud je určitý objekt otevírán, je třeba jej po ukončení práce rovněž uzavřít. Výjimkou není ani práce se soubory a proto veškeré soubory otevřené funkcí **FOpen** je nutné následně uzavřít pomocí funkcí **FClose**. Tato funkce je opakem k funkci FOpen, tedy otevřený datový soubor uzavře a ukončí možnost s ním pracovat.

Příklad zápisu:

```
<?php
$file = fopen ("soubor.txt", "r");
fclose($file) ;
?>
```

4.1.4 Kopírování souboru

Jedná se o funkci **Copy**, která slouží ke kopírování zdrojového souboru jinam či pod jiným jménem. Pokud kopírování proběhne v pořádku, vrátí funkce hodnotu true (pravda), v opačném případě false (nepravda).

Příklad zápisu:

```
<?php
if (!copy($file, $file.'.bak')) {
        print ("Soubor $file se nepovedlo zkopírovat<br>\n");
}?>
```

U tohoto příkazu bychom si měli dát pozor na to, že existující soubor se shodným jménem a umístěním bude bez upozornění nahrazen. Je proto důrazně doporučeno používat funkci funkce **File_Exist.**

4.1.5 Smazání souboru

Funkce **UnLink** smaže soubor zadaného jména z disku (pokud to umožňuje nastavení přístupových práv).

Příklad zápisu:

```
<?php
if (unlink("test.txt")) echo "Soubor byl smazán";
?>
```

4.1.6 Struktura adresáře

Podobně jako Průzkumník Windows dovede zobrazit obsah adresáře (složky), dovede to i PHP a to pomocí metody **opendir**, která otevře složku a **readdir**, která

složku přečte. Zdrojový kód, který vypíše na každý řádek jeden soubor (nebo složku) aktuálního adresáře:

```
$adresar = opendir(".");
while ($soubor = readdir($adresar)){
echo ($soubor."<br />");
```

Parametr metody opendir může být nahrazen URL. Mezi další možnosti práce s textem patří například zjištění velikosti souboru, čtení dat z otevřeného souboru, čas poslední úpravy souboru atd.

4.1.7 Nahrání souborů na web (Upload)

Smyslem této oblasti je umožnit uživatelům nahrání souborů uložených na jejich počítači na webový server (využívá se to například pro různá fotoalba apod.). Nejjednodušším způsobem je uložení pomocí skriptu PHP za využití standardních funkcí jazyka PHP.

Pro vlastní práci je potřeba vytvořit formulář pro specifikaci souboru nahrávaného na server – tento můžeme vytvořit například pomocí klasického jazyka HTML.

Po jeho vytvoření přejdeme ke skriptu pro zpracování souboru:

```
----- začátek skriptu upload.php -----
if ($the_file<> "none"):
  echo "Původní jméno souboru: ".$the_file_name."<br>";
  echo "Dočasné jméno souboru: ".$the_file."<br>";
  echo "Velikost souboru v bytech: ".$the_file_size."<br>";
  echo "Typ souboru: "$the_file_type."<br>";
  $cíl = "C:\\wwwroot\\doc\\test.txt";
  if (copy($the_file,$cíl)) echo "Soubor $the_file_name byl úspěšně nahrán na
server";
endif;
?>
<form ENCTYPE="multipart/form-data" action="test.php" method="post">
<P>Vyberte soubor:
<br><INPUT NAME="the_file" Type="file" SIZE=35><br>
<input type="submit" Value="Upload">
</form>
---- konec skriptu uplod.php-----
```

Tímto by měl být soubor umístěn na server. V jiných skriptovacích jazycích to bývá zpravidla složitější. Proměnná $the_file se v tomto skriptu netestuje na prázdnou hodnotu, ale na hodnotu „none“. V případě, že soubor není vybrán, otestuje hodnota INPUT prvku typu FILE zvláštní hodnotu „none“, takže na tento rys bychom při psaní kódů neměli zapomínat.

4.1.8 Emailové zprávy

Využívání emailových zpráv je v dnešní době důležité. Z tohoto důvodu se při registraci k novým stránkám vyplňuje emailová adresa nového návštěvníka. Mezi výhody rozesílání emailů patří například informace o nové registraci, nových službách webu či například upozornění na odpověď na inzerát, popřípadě na reakci na fóru. Mezi negativní stránky registrací s povinou kolonkou emailová adresa je spam. Pakliže je správně nastavena adresa serveru SMTP v konfiguračním souboru php.ini (při automatické instalaci ji rovněž zadáváme), je odesílání emailů skriptem PHP rychlé. Postačí k tomu jediná funkce s názvem **mail**. Jejími parametry jsou: emailová adresa příjemce, předmět zprávy, obsah zprávy (nepovinnými znaky jsou dodatečné emailové hlavičky a příkazy). V praktické části bakalářské práce je funkce mail využito pro kontakt správce webové aplikace.

Příklad zápisu:

```
<?php
if  (mail("mira@seznam.cz, "První email"", "Odeslal jsem první email")):
        echo "Zpráva byla odeslána":
else:
        echo "Zpráva nebyla odeslána";
endif;
?>
```

Ve skriptu je využito toho, že mail je funkcí a vrací hodnotu true v případě úspěšného odeslání, v opačném případě hodnotu false. Lze tak snadno zpracovat úspěšné pokusy či naopak. Uvedený skript je možné i dále upravovat:

```
<?php
$to             =  "mira@seznam.cz";
```

```
$subjekt      = "První email";
$message      = "Odeslal jsem první email";
$headers      = "From: Miroslav Vacikar <mira@seznam.cz>\ n";
if  (@mail($to, $subjekt, $headers)):
echo "Zpráva byla odeslána":
else:
echo "Zpráva nebyla odeslána";
endif;
?>
```

Toto by měl být dostatečně propracovaný skript na odesílání emailu. Je možné věnovat pozornost i dalším hlavičkám, které jsou k dispozici. Mezi ty nejdůležitější například patří:

"X-Priority: 1\n";

Toto definuje výši priority emailové zprávy. V tomto případě je email stanoven jako urgentní.

4.2 Budoucnost jazyka PHP

PHP není v roce 2006 jako skriptovací jazyk na svém počátku – spíše na svém vzestupu. Má pro to i řadu předpokladů – je poskytován zdarma, dokáže spolupracovat s oblíbeným Web Serverem Apache, který je rovněž poskytován zdarma, je možné jej instalovat na obě nejrozšířenější platformy (Linux, Windows) a navzájem je mezi sebou kombinovat. Jeho dalšímu rozvoji stojí v cestě jeho vlastní vývoj (na odborných fórech se objevují ohlasy na chyby v nových verzích). Jeho šíření neprospívá i ukončení podpory většiny automatických instalátorů, které mohou hlavně v začátcích některým uživatelům pomoci.

Oproti tomu jazyku PHP velkou naději dává i spousta různých nadšenců a programátorů po celém světě. Překvapením je, kolik je českých webových stránek, zabývajících se programováním v jazyce PHP. Na těchto stránkách se nachází mnoho návodů, rad i postupů. Nacházejí se zde základy i pokročilejší postupy, je zde zpřístupněno i mnoho skriptů, které jsou již funkčně vyzkoušené a ověřené. Každý zájemce se tak má možnost nechat pomalinku vést od jednodušších skriptů po ty

složitější, na fórech probrat své problémy a společně s jinými uživateli se je naučit překonat.

Největším konkurentem, který by mohl PHP překonat, je jazyk **ASP** (Active Server Pages), jehož popis je uveden na začátku této práce. Ale i tento má své nevýhody – velké náklady na pořízení, složitější syntaxe, vázanost na konkrétní operační systém.

5. TVORBA WEBOVÉ APLIKACE

Cílem praktické části práce bylo vytvořit web, na kterém by byly uveřejněny informace, které zadavatel považuje za důležité. Uvedené by bylo možné stvořit i v jiném programovacím jazyku. Pro volbu jazyka PHP hovoří jeho dynamičnost, snadná možnost poupravovat již zveřejněný text – pokud je totiž načítán ze souboru ve formátu .doc, stačí tento pozměnit, což byl hlavní záměr. Vytvořená webová aplikace umožní změnu obsahu webových stránek – tato oprava probíhá on-line (možnost zapojení více pracovníků), bez použití speciálních aplikací a jen s malými nároky na obsluhu. Případné rozšíření webové aplikace nebude již tolik pracné, jako vytvoření webové aplikace samotné.

Při vytváření webové aplikace je v první řadě potřeba brát zřetel na požadované funkce. V tomto případě je vycházeno z následujících požadavků:

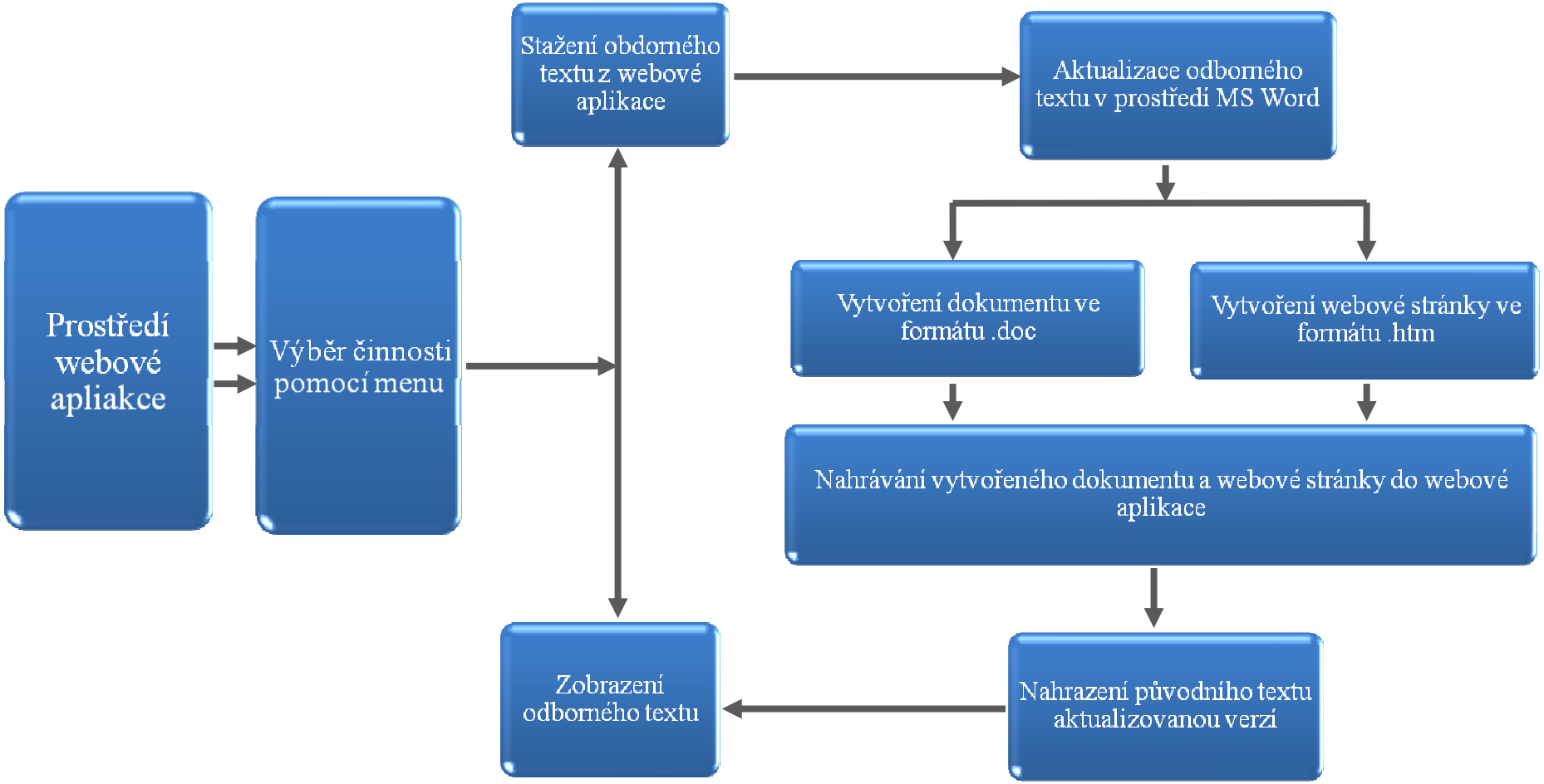
- Možnost měnit odborné texty, které jsou umístěny na www stránce (změna odborného textu, vzorců, obrázků apod.)
- Snadný přístup (pomocí internetových prohlížečů)
- Co nejmenší nároky na odbornost uživatelů
- Použití nástrojů, které využívá zpravidla každý běžný uživatel PC (bez použití speciálních nástrojů)

Vše uvedené vytvořená webová aplikace splňuje:

- Přístup pomocí internetového prohlížeče (Internet Explorer, Firefox, Mozila)
- Znalosti kladené na uživatele jsou minimální (ukládání souboru ve formátu .doc a .htm, práce s programem MS Word)
- Není nutná instalace jakýkoliv dalších programů

V zásadě jsem se pokusil při vytváření aplikace používat takové programy, které jsou zpravidla na běžném PC nainstalovány (operační systém Windows, MS Word) či jsou dostupné zdarma (server Apache či Complex Web Server, phpEditor). V následujících kapitolách je popsána funkce webové aplikace a její ovládání.

5.1. Schéma funkce webové aplikace



5.2 Popis zdrojového kódu webové aplikace

On-line změna odborného textu probíhá v několika krocích, kdy jednotlivé části skriptů budou v této části popsány. Nejprve je nutné si z webové aplikace stáhnout odborný text, který je uložen ve formátu .doc. To je zajištěno následujícím skriptem:

5.2.1 Stažení odborného textu ve formátu .doc

V první části je pomocí příkazu „chdir“ zajištěno zobrazení adresáře „upload“. Následně je z tohoto adresáře pomocí příkazu „readdir“ postupně vypisován (za pomoci podmínky „while“) jeho obsah, který je načten do jednotlivých řádků tabulky. V závěru skriptu je určena barva jednotlivých řádků, kdy sudé a liché jsou vždy stejné. Vypsané názvy dokumentů jsou hypertextovým odkazem, který umožní jejich vlastní stažení.

```
<?
/*chdir("upload");*/
$dir = opendir("upload");
$radek="tabliche";
while ($file = readdir($dir))
{ if ($file <> "." && $file <> "..")
  { $pole[] = $file;
  }
}
sort ($pole);
reset ($pole);
 //--------------načtení názvů souborů do pole-------------
foreach ($pole as $polozka)  {
//---------------určení barev řádků tabulky---------------
  if ($radek == "tabliche")  {
    $radek = "tabsude";
  }else{
    $radek = "tabliche";
  }
  if (!ereg("slozka",$polozka))
     {
  echo("<tr><td class=\"$radek\"><a
href=\"upload/$polozka\">$polozka</a></td></tr>");
//--------------odkaz na cestu k vybranému souboru ze
seznamu--       }
}
?>
```

Stažený odborný text lze upravit dle vlastních požadavků – v programovém prostředí MS Word. Po provedení úprav se dokument uloží ve formátu .doc a jako webová stránka ve formátu .htm.

5.2.2 Nahrávání vytvořeného dokumentu a webové stránky do webové aplikace

V předchozím kroku došlo k vytvoření dokumentu, webové stránky a složky, potřebné pro správné zobrazení webové stránky. Tyto dokumenty je nutné uložit zpět do webové aplikace – tento proces je rozdělen do několika kroků:

V první části se kontroluje velikost souboru, který je ukládán do webové aplikace. Tento krok je nutný, aby nedošlo k přeplnění datového prostoru, určeného celé webové aplikaci, jedním souborem.

```
        $zobrazovat = "ne";
  if(is_uploaded_file($_FILES['soubor']['tmp_name']))
  //------------podmínka pokud je soubor uploadován na
server-------
    {
        $zobrazovat = "ne";
          if (($_FILES['soubor']['size'] > 25000288))
           Die("Uploadovaný soubor je příliš velký");
          $explode = explode(".",$soubor_name);
```

Příkaz „Die“ (v překladu smrt) zajistí, že v případě naplnění podmínky je celý PHP skript ukončen. Dojde pouze k vypsání textu, který byl vypsán jako parametr tohoto příkazu. Tento příkaz se používá v případech, kdy nastane ve skriptu chyba a další vykonávání PHP skriptu nemá význam (např. výpis souboru, který neexistuje apod.).

V následujícím kroku dojde pomocí příkazu „FOpen“ (File Open) k otevření webové stránky ve formátu .htm, kterou ukládáme do webové aplikace. Celá webová stránka je následně procházena a za pomoci příkazu „EReg“ je kontrolována přítomnost vyjmenovaných znaků (např. „$“). Kontrola znaků a jejich následné smazání je prováděno, aby nedošlo k zaměnění výrazů, které jsou používány jako příkazy apod. (např. uváděný znak „$“ je označením proměnné).

```
$soubor = FOpen("upload/$name","r+");
while ($data = fgets($soubor,5000))
{
//  echo "<p>Zpracovávam řádek č.$radek: <br>";
//  $radek++;
     if((EReg('src="',$data)) || (EReg ('SRC="',$data)))
         {
//        echo "---jo----";
        $pouvozovkach = explode("\"",$data);
         $data="";
        foreach ($pouvozovkach as $cast)
             {
            if(EReg ("^.*\..*$",$cast))
             {
            if(!EReg ("^.*\\\.*\..*$",$cast))
             {
            if(!EReg ("^.*/.*\..*$",$cast))
             {
          $cast = $sname."/".$cast;
```

Do webové aplikace se ukládají aktuální podoby souborů, které mají stejný název jako soubory již zde uložené. Z tohoto důvodu je před uložením souboru zkontrolováno, zda soubor pod stejným názvem existuje. Podmínkou „if“ je testována existence souboru v adresáři „upload“, kdy v kladném případě je za pomoci příkazu „unlink“ vymazán. Tento krok zajistí možnost uložení souboru do adresáře pod názvem, s jakým je zde již jiný soubor uložen.

```
if(file_exists("upload/$name"))
           {
           unlink("upload/$name");
             }
                copy($_FILES['soubor']['tmp_name'],
"upload/$name");
echo "Soubor $name byl úspěšně vložen<br>";
```

Další část zdrojového kódu tvoří formuláře, pomocí nichž určujeme cestu k jednotlivým souborům, které jsou ukládány do webové aplikace. Zobrazený zdrojový kód ukazuje část formuláře, pomocí něhož se ukládají pomocné soubory pro správné zobrazení webové stránky ve formátu .htm.

```
settype($j,"string");
echo("  <tr>");
echo("    <td align=\"center\" class=\"menu\"
colspan=\"2\">Nalistování souboru".$j." z disku:<input
```

```
type=\"file\" name=\"addsoubor".$j."\"
class=\"modretlacitko\"></td>");
echo("  </tr>");
}
echo("  <tr>");
echo("    <td colspan=\"2\" align=\"center\"
class=\"menu\"><input type=\"submit\" name=\"akce\"
value=\"Odešli soubory na server\"
class=\"cervenetlacitko\"></td>");
echo("  </tr>");
echo("<input type=\"hidden\" name=\"pocet\" value=\"$pocet\">");
echo("<input type=\"hidden\" name=\"sname\" value=\"$sname\">");
echo ("</form>");
echo ("</table>");
```

Tímto způsobem se do webové aplikace uloží veškeré soubory, potřebné pro změnu odborného textu. Prvním z nich je dokument ve formátu .doc, který je posléze možné z webové aplikace uložit zpět do PC a provádět další změny. Jeho aktuálnost (opakované ukládání do webové aplikace) zajišťuje, aby odborný text v tomto dokumentu odpovídal odbornému textu zobrazovanému na webové stránce.

Shora popsaným způsobem dojde také k nahrazení webové stránky ve formátu .htm jinou, aktuální verzí. Je však potřeba, aby byla zachována struktura celé webové stránky, která je zobrazena v internetovém prohlížeči (tzn. menu, ovládací prvky, úvodní lišty apod.). Pro tento případ má skriptovací jazyk PHP jednoduchý příkaz „include“. Pomocí tohoto příkazu lze webovou stránku, která se v internetovém prohlížeči zobrazuje jako celek, rozdělit na několik nezávislých částí.

Příkaz „include“ slouží k tomu, aby bylo možné rozdělit jednu stránku na více souborů. V následující ukázce zdrojového kódu jsou do jedné webové stránky vloženy dvě jiné. Jednou z nich je menu, pomocí něhož celou webovou aplikaci ovládáme. Druhou samostatnou webovou stránkou je pak webová stránka 1111.htm, kde je umístěn vlastní odborný text. Tímto jednoduchým způsobem můžeme měnit webovou stránku s odborným textem, aniž by to mělo vliv na zbytek zobrazené webové stránky.

```
<td height="20px" class="menu">
    <?
    include("tree.html");
    ?>
  </tr>
  </tr>
</table>
<td valign="top">
```

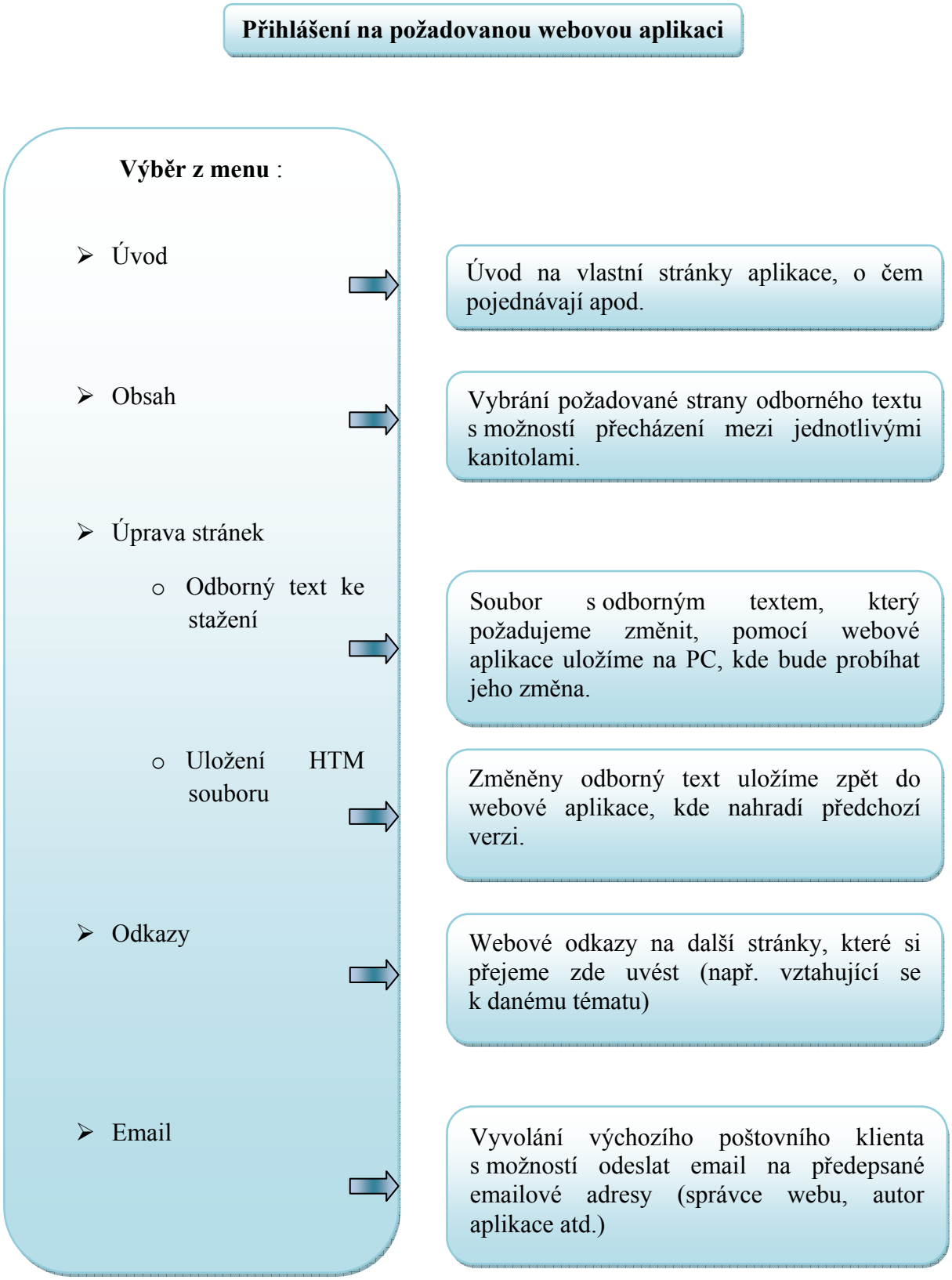
```
       <table border="5" width="910px">
        <tr><td align="center" valign="top"
class="intable"></td></tr>
        </table>
        <table>
        <br><br>
        <td>
<?
    include("1111.htm");
?>
```

Použití příkazu „include“ bývá mnohem větší, jelikož je celek (webová stránka zobrazená v internetovém prohlížeči) složen z několika samostatných webových stránek. V takovém projektu se pak je možné lépe orientovat, případně každou část může vytvářet i jiná osoba.

Další praktické použití příkazu „include“ je pro názornost uvedeno na příkladu:

Při vytváření webových stránek je potřeba na jejich konec vložit vždy stejný prvek – například adresu. Při změně těchto prvků (adresy) je nutné na každé webové stránce provést změnu na aktuální údaje. Pokud uvedená adresa bude vložena pomocí příkazu „include“, stačí, když uskutečnit změnu ve vloženém souboru a změna se projeví na všech stránkách, kde je použit uvedený prvek (adresa).

5.3 Schéma ovládání webové aplikace

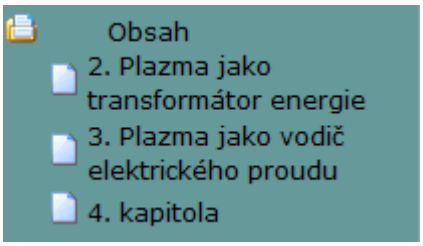


5.4 Popis některých částí ovládání webové aplikace

5.4.1 Obsah

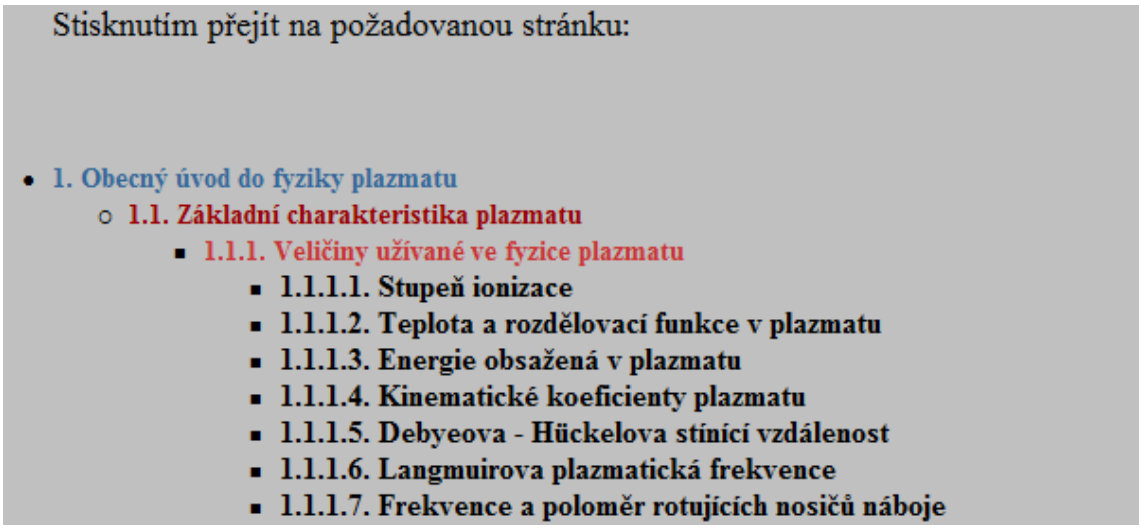
Tento hypertextový odkaz má dvě funkce:

První z nich se projeví „kliknutím“ na ikonu (obr. č. 3) - poté se otevře obsah pouze v menu – po jednotlivých kapitolách (toto je zajištěno Java skriptem, není zde rozebírán, neboť není předmětem této práce).



Obr. č. 3 Rozbalení obsahu v menu

Druhá funkce se projeví po „kliknutí“ přímo na hypertextový odkaz „Obsah“ (obr. č. 4) – v pravé části webové stránky se zobrazí textový obsah jednotlivých stránek, kdy pojmenování každé stránky je zároveň hypertextovým odkazem na danou stranu odborného textu. Číselné označení poté souhlasí s číselným označením souborů ve formátu .doc, ve kterém je uložen odborný text dané stránky.



Obr. č. 4 Rozbalení obsahu s hypertextovými odkazy na jednotlivé strany odb. textu

Formátování textu v obsahu je zajištěno pomocí HTML jazyka, aby bylo zajištěno logické členění na jednotlivé části – viz obr. č. 5.

```
<big>
<ul>
<lh> Stisknutím přejít na požadovanou stránku:</lh>
<br> <br> <br> <br>
<li> <h4>1. Obecný úvod do fyziky plazmatu</h4>
        <ul>
        <li><a href="1.1.php"><h3>1.1. Základní charakteristika plazmatu</h3></a>
              <ul>
              <li> <h5>1.1.1. Veličiny užívané ve fyzice plazmatu</h5>
                    <ul>
                    <li><a href="upload/1111.php"><h6> 1.1.1.1. Stupeň ionizace</h6> </a>
                    <li><a href="upload/1112.php"><h6> 1.1.1.2. Teplota a rozdělovací funkce v plazmatu</h6> </a>
                    <li><a href="upload/1113.php"><h6> 1.1.1.3. Energie obsažená v plazmatu</h6> </a>
                    <li><a href="upload/1114.php"><h6>  1.1.1.4. Kinematické koeficienty plazmatu</h6> </a>
                    <li><a href="upload/1115.php"><h6> 1.1.1.5. Debyeova - Hückelova stínící vzdálenost</h6> </a>
                    <li><a href="upload/1116.php"><h6> 1.1.1.6. Langmuirova plazmatická frekvence</h6> </a>
                    <li><a href="upload/1117.php"><h6> 1.1.1.7. Frekvence a poloměr rotujících nosičů náboje</h6> </a>
                    </ul>
               <li> <h5>1.1.2. Plazmatické záření</h5>
```

Obr. č. 5 Formátování textu v obsahu

5.4.2 Design stránek

Tato sekce slouží k vlastní úpravě odborného textu - tímto je myšlena změna uveřejněných informací, jednotlivé menu editovat nelze.

Vlastní změna odborného textu je zajištěna následujícím způsobem:

- Uživatel si vybere stranu odborného textu, kterou požaduje změnit (obr. č. 6)

1.1.1.5. Debyeova – Hückelova stínící vzdálenost

Průběh elektrického potenciálu v okolí nabitých částic ve vakuu se výrazně odlišuje od průběhu potenciálu v plazmatu. Tento rozdíl je způsoben velkou pohyblivostí nosičů náboje, což má za následek vytváření záporně nabitých zón kolem kladného náboje a naopak. Vzniku těchto zón může v některých případech bránit tepelný pohyb částic. Pro výpočet průběhu potenciálu $V(r)$ plazmatu kolem jednoho iontu mohou být přímo použity výsledky z teorie silného elektrolytu (Debye – Hückel 1923). Coulombův potenciál zasahuje do oblasti stíněné potenciálem prostorového náboje V_R, proto platí $V = V_C + V_R$. S ohledem na Boltzmanovu rovnováhu v prostorové nábojové zóně lze pro izotermické plazma ($Z = 1$), v přiblížení $e_o U < kT$, psát :

$$V(r) = \frac{1}{4\pi\varepsilon_0}.\frac{\varepsilon_0}{r}\exp\left(-\frac{r}{\lambda_D}\right) \qquad 1.16.$$

$$\frac{df_i}{dt} = \frac{\partial f_i}{\partial t} = \frac{\partial f_i}{\partial r} = \frac{\partial f_i}{\partial c} = B\, f_i \qquad 1.17.$$

Obr. č. 6 – Strana odborného textu, kde požadujeme provést změny

- V menu „Úprava stránek“ v položce „Odborný text ke stažení“ si stáhne požadovanou stranu odborného textu (uložena ve formátu .doc) – jejich číselné pojmenování odpovídá číselnému pojmenování, které je uvedeno v obsahu (obr. č. 7)

1112.doc
1115.doc
1117.doc
1122.doc

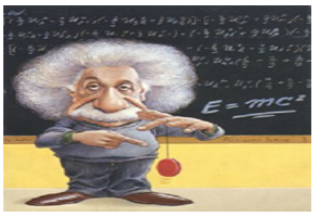
Obr. č. 7 Hypertextový seznam stránek odborného textu

- V uloženém odborném textu – dokumentu, provede požadované změny (obr. č. 8)

1.1.1.5. Debyeova – Hückelova stínící vzdálenost

Průběh elektrického potenciálu v okolí nabitých částic ve vakuu se výrazně odlišuje od průběhu potenciálu v plazmatu. Tento rozdíl je způsoben velkou pohyblivostí nosičů náboje, což má za následek vytváření záporně nabitých zón kolem kladného náboje a naopak. Vzniku těchto zón může v některých případech bránit tepelný pohyb částic. Pro výpočet průběhu potenciálu $V(r)$ plazmatu kolem jednoho iontu mohou být přímo použity výsledky z teorie silného elektrolytu (Debye – Hückel 1923). Coulombův potenciál zasahuje do oblasti stíněné potenciálem prostorového náboje V_R, proto platí $V = V_C + V_R$. S ohledem na Boltzmanovu rovnováhu v prostorové nábojové zóně lze pro izotermické plazma ($Z = 1$), v přiblížení $e_o U < kT$, psát :

jakékoliv změny.....



Obr. č. 8 Změny provedené v odborném textu – dokument ve formátu .doc

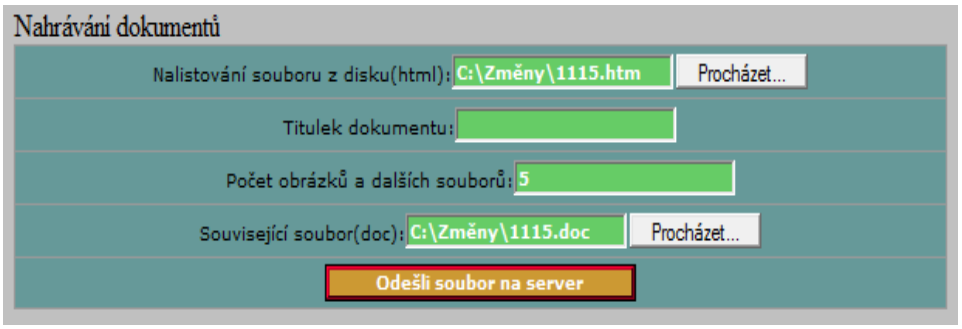
- Dokument uloží na PC, kde provádí změny – a to ve formátu .doc a dále ve formátu webové stránky. Tímto mu v daném umístění vzniknou dva dokumenty (první ve formátu .doc, druhý ve formátu .htm) a dále složka s obrázky a jinými soubory, které jsou nutné pro správné zobrazení webové stránky

➢ V menu „Úprava stránek“ v položce „Uložení HTM souboru“ pomocí hypertextového odkazu vyvoláme uploadovací okno, ve kterém je nutno zadat (obr. č. 9) :

- Umístění dokumentu ve formátu .htm
- Volitelně titulek dokumentu

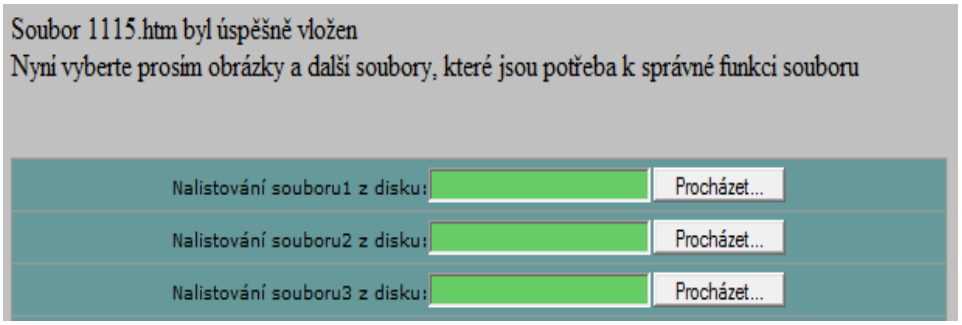
- Volitelně počet obrázků a dalších souborů (pokud se měnila pouze textová část, není to nutné, rovněž při změně obrázků na začátku dokumentu není nutné znovu uploadovat všechny obrázky)

- Související dokument ve formátu.doc (abychom při příští opravě měli zajištěnou aktuální podobu odborného textu dané stránky



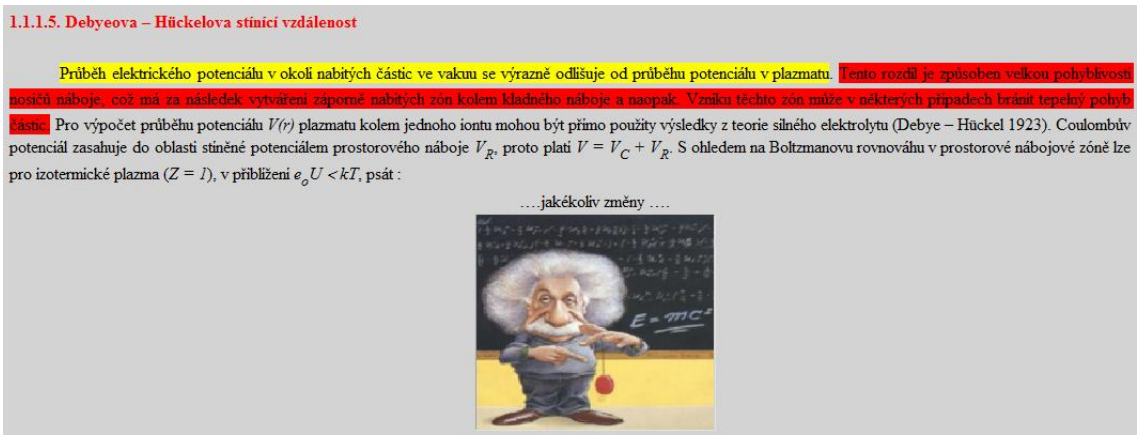
Obr. č. 9 Zobrazení uploadovacího formuláře s vyplněnými údaji

➢ V případě, že jsme zadali určitý počet obrázků a dalších souborů, objeví se další uploadovací okno, jehož pomocí zadáme umístění jednotlivých souborů (počet uploadovacích řádek odpovídá zadanému počtu obrázků a dalších souborů), viz obr. č. 10.



Obr. č. 10 Uploadovací okno na obrázky a další soubory

➢ Po dokončení tohoto postupu je odborný text požadované strany změněn, viz obr. č. 11.



Obr. č. 11 Požadované změny v odborném textu

Z jednotlivých obrázků je patrné, webová aplikace umožňuje velkou škálu změn – doplnění textu, změnu barvy písma a jeho pozadí, změna obrázků apod. Rovněž ovládání aplikace je přehledné, výstižně popsáno a vede uživatele jednotlivými kroky, aby se předešlo případným pochybením.

6. ZÁVĚR

V bakalářské práci je popsáno, jaké základní možnosti programovací jazyk PHP nabízí. Měla by být průvodcem od jeho výběru, přes zvolení instalace, vysvětlení základních pojmů a představení základních příkazů. Čtenáři by neměla přinést pouhou znalost příkazů jako takových, spíše by měla být ukázkou nevšedního použití příkazů a tím i návodem na vytváření takových skriptů, které nejsou běžně dostupné.

V rámci praktické části bakalářské práce byla vypracována zcela nová, nestandardní webová aplikace, která umožňuje on-line změnu odborného textu, uveřejněného na webové stránce. Tato aplikace má široké použití především v odborné oblasti, kde se na jednom konkrétním projektu podílí více osob (práce jednotlivce by postrádala v tomto případě smysl). Mezi hlavní přednost, krom samotné praktičnosti, je snadné ovládání a tím i možnost velkého rozšíření, neboť své praktické uplatnění si jistě najde. Nalezené technické řešení dle mého názoru plně vypovídá o některých možnostech programovacího jazyku PHP.

Pro mne osobně byla práce velkým přínosem, před jejím započetím jsem měl pouze omezenou představu, jakým způsobem jazyk PHP pracuje. Jsem rovněž rád, že mi bylo umožněno podílet se na projektu, který není pouze teoretického rázu. Vytvořená webová aplikace, kromě svého praktického použití, může být některou ze svých částí vzorem pro další projekty, které nejsou zcela běžné. Věřím, že mi nasbírané zkušenosti budou do budoucna k užitku a výhodou jak v soukromém životě, tak v profesi.

7. SEZNAM POUŽITÉ LITERATURY

[1] GUTMANS, A., BAKEN, S.S., RETHANS, D. Mistrovství v PHP 5. 1. vyd. Brno: Nakladatelství CP Books, a.s., 2005.
ISBN 80-251-0799-X.

[2] MACH, J. PHP pro úplné začátečníky. 2. rozšíř. vyd. Brno: Nakladatelství CP Books, a.s., 2005.
ISBN 80-7226-834-1.

[3] Milda, M. HTML pro začátečníky. 1. vyd. České Budějovice: Nakladatelství KOPP, 2000.
ISBN 80-7232-052-1.

[4] Kosek, J. HTML Tvorba dokonalých WWW stránek. 1. vyd. Praha: Grada Publishing, 1999.
ISBN 80-7169-608-0.

[5] Kadavý, D. CorelDRAW 12. 1. vyd. Brno: Nakladatelství CP Books, a.s., 2005.
ISBN 80-251-0559-8.

Sekundární údaje z prospektů a propagačních materiálů jednotlivých počítačových programů.

Internetové stránky:

URL: www.jaknaweb.com

URL: www.jakpsatweb.cz

URL: www.zive.cz

URL: www1.lf1.cuni.cz

URL: www.plavacek.net

URL: http://programujte.com

URL: www.mrkev.info

URL: www.linuxsoft.cz

URL: www.pcsvet.cz

URL: www.gifet.net

URL: http://wellstyled.com

URL: www.kosek.cz

URL: www.pvtnet.cz

URL: www.php.net

URL: www.php.cz

URL: www.phpbuilder.com

URL: http://java.tatousek.cz

URL: www.javascript.com