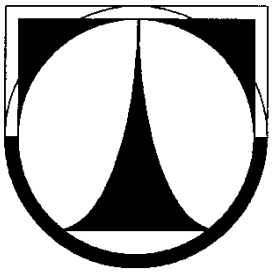
TECHNICKÁ UNIVERZITA V LIBERCI

FAKULTA MECHATRONIKY, INFORMATIKY A MEZIOBOROVÝCH STUDIÍ



Ústav nových technologií a aplikované informatiky

Studijní program

B2612 Elektrotechnika a informatika

Obor Informatika a Logistika

E-shop konfigurátor pro CMS Joomla

E-shop configuration tool for Joomla CMS

Bakalářská práce

Karel Bartůněk

Vedoucí bakalářské práce: Mgr. Jiří Vraný, Ph.D.

Počet stran: 38

Počet obrázků: 13

Počet příloh: 3

Dokončeno: květen 2011

Místo pro oficiální zadání BP

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce.

V …… dne ……..…… ……

podpis

Poděkování

Na tomto místě bych rád poděkoval Mgr. Jiřímu Vranému, Ph.D. za vedení mé bakalářské práce.

Dále bych rád poděkoval především rodičům a všem mým blízkým, kteří mě po celou dobu studia na FM TUL podporovali a byli mi po dobu studia pevnou oporou.

Téma

E-shop konfigurátor pro CMS Joomla

Anotace

Bakalářská práce se zabývá problematikou návrhu a vývoje komponent pro redakční systém Joomla!. Na úvod jsou popsány základní vlastnosti, možnosti a historie tohoto systému. Dále je pozornost věnována implementaci návrhového vzoru Model-View-Controller, využití technologií XHTML, PHP, MySQL a JavaScript.

Následně jsou tyto znalosti a technologie využity k vytvoření vlastní, uživatelsky přívětivé komponenty, která bude moci být nasazena v e-shopech založených na CMS Joomla, doplněných o komponentu VirtueMart poskytující funkce e-shopu a která slouží jako dynamický vyhledávač produktů v e-shopech.

Theme

E-shop configuration tool for Joomla CMS

Annotation

The Bachelor thesis deals with the issue of design and development of components for Joomla Content Management System. The introduction describes the basic features, possibilities and history of this system. Further attention is paid to the implementation of Model-View-Controller design pattern as well as to the use of XHTML, PHP, MySQL and JavaScript technologies.

Consequently, this knowledge and technologies are used for creation of an own, user-friendly component, which is to be employed in Joomla CMS-based e-shops, enhanced with VirtueMart component that provides e-shop functions and that serves as a dynamic search engine of products in e-shops.

Klíčová slova

Joomla, VirtueMart, komponenta, modul, e-shop.

Key words

Joomla, VirtueMart, component, module, e-shop.

Obsah

Seznam obrázků

Přehled použitých zkratek, symbolů

AJAX	Asynchronous JavaScript and XML a XML	Asynchronní JavaScript
CMS	Central Management System	Systém pro správu obsahu
CSS	Cascading Style Sheets	Kaskádové styly
GNU GPL	GNU General Public License	Všeobecná veřejná licence
HTML	HyperText Markup Language	
PHP	Hypertext Preprocessor	Hypertextový preprocesor
SEO	Search Engine Optimalization	Optim. pro vyhledávače

1 Úvod

Bakalářská práce se zabývá tvorbou komponenty a modulu pro jeden z nejvíce používaných systémů pro tvorbu a správu obsahu (CMS - Content Management System) profesionálních internetových či intranetových stránek – systém Joomla!.

Tento systém si získal velké množství příznivců a počet instalací díky své propracovanosti. Základní instalace obsahuje funkce jako SEO-friendly URL, caching, indexaci stránek, RSS, tisknutelné verze stránek, zobrazování novinek, blogy, diskusní fóra, hlasování, kalendář, vyhledávání v rámci webserveru a také například podporuje změny vzhledu. Pokud nám chybí nějaká specifická funkce, je možné doinstalovat či naprogramovat rozšiřující komponentu, která danou funkčnost doplní. Základní systém můžeme rozšířit například o fotogalerii, internetový obchod, možnost vícejazyčné verze webu a tak dále.

1.1 Cíle bakalářské práce

Cílem této bakalářské práce je vytvoření komponenty s modulem, díky kterým se rozšíří funkcionalita e-shopů založených na CMS Joomla! o možnost vyhledávání produktů na základě zvolených kategorií. S tím souvisí se seznámit s objektově orientovaným programováním komponent pro CMS Joomla!. Obecně bude popsán vývoj vlastní komponenty a modulu od návrhu, přes testování až po její samotnou implementaci. Podrobněji bude popsána má komponenta, která bude umožňovat dynamickou tvorbu šablon pro vyhledávání zboží v internetovém obchodě na základě zvolených parametrů a bude testována na reálném a zaběhnutém e-shopu s výpočetní technikou.

2 CMS Joomla!

2.1 Historie systému Joomla!

Samotná Joomla vznikla ze svého předchůdce – redakčního systému Mambo. Mambo bylo vyvíjeno v od roku 2000 a od té doby získalo mnoho ocenění jako je například "Best Open Source Solution" či "Best of Show - Total Industry Solution". V roce 2005 však vnitřní rozepře ve Výboru řízení Mamba způsobily odchod základu vývojářského týmu (tzv. Core Team) a tento tým následně začíná vystupovat pod názvem Open Source Matters. V září roku 2005 tedy vychází verze systému Joomla! 1.0 pod hlavičkou OSM. Od té doby Joomla! prošla mnoha fázemi vývoje, od verze 1.0 se za šest let vývoje dostala k verzi 1.7, která vyšla v červenci roku 2011.

2.2 Licence Joomla!

Systém je licencován pod GNU GPL, je šířen zdarma a práva vlastní společnost Open Source Matters. Licence GNU GPL nám umožňuje používat systém zdarma jak pro osobní účely, tak je možnost jej nasadit i v komerční sféře, můžeme jej dále upravovat, kopírovat či distribuovat a to i za poplatek. V případě, že upravený systém poskytneme k užívání dalšímu subjektu, je nutné uveřejnit jeho zdrojové kódy. Pokud upravujeme systém pouze pro vlastní použití, licence nám dává právo úpravy nezveřejňovat.

V této práci vznikne rozšíření systému – komponenta a modul. I přesto, že oba budou využívat vnitřní funkce frameworku Joomly, tak na takto vytvořené rozšíření se již licence Joomly nevztahuje a určit si ji může vývojář sám.

2.3 Požadavky systému a vytvořené aplikace

Joomla! je vytvořena skriptovacím programovacím jazykem PHP. Ke svému běhu tedy potřebuje podporu PHP, databázový systém MySQL a webový server Apache. Na uživatelské stanici je nutné mít nainstalovaný moderní internetový prohlížeč (např. Chrome, Opera, Firefox, IE apod.), povolený JavaScript, aplikace totiž využívá AJAX (Asynchronous JavaScript and XML) a funkce knihovny jQuery.

3 Použité technologie

3.1 XHTML

XHTML je značkovací jazyk pro tvorbu hypertextových dokumentů v prostředí WWW vyvinutý mezinárodním konsorciem W3C.

3.2 JavaScript

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk, používaný jako interpretovaný programovací jazyk pro WWW stránky.

3.3 MySQL

MySQL je multiplatformní relační databázový systém. Komunikace s ním probíhá pomocí jazyka SQL.

3.4 AJAX (Asynchronous JavaScript and XML)

Cílem bylo vytvořit interaktivní komponentu, proto byla zvolena tato technologie, díky které je možné měnit obsah stránky aniž by se musela celá znovu načítat. Umožňuje okamžitou komunikaci mezi prohlížečem a serverem a tak je docíleno průběžného obnovování výsledků vyhledávání mezi zbožím.

3.5 jQuery

jQuery je JavaScriptová knihovna (sada funkcí) uvolněná pod duální licencí MIT a GPL, která umožňuje a velice ulehčuje interakci mezi JavaScriptem a HTML. Má velké množství funkcí, umožní nám jednoduše pracovat s objekty DOM, manipulovat s CSS, vytvářet animace, reagovat na události atd. V této aplikaci je jQuery využito k práci s AJAX, bylo tím umožněno vytvořit posuvník s cenou díky utilitě Slider (obrázek), či vykreslit strom kategorií pomocí utility Treeview.

3.6 PHP

PHP je skriptovací programovací jazyk. Nejčastěji se používá při tvorbě internetových aplikací s technologiemi HTML, XHTML. Skripty jsou prováděny na straně serveru.

4 Rozšíření pro CMS Joomla!

Joomla! umožňuje rozšířit svou funkčnost pomocí třech typů rozšíření, které mají své specifické vlastnosti – komponenty, moduly, pluginy (v této aplikaci je využito komponenty pro správu šablon vyhledávání a modul pro výpis vyhledávacího formuláře).

4.1 Komponenta

Komponenta je základní typ rozšíření pro tento systém. Je rozdělena na dvě části, back-end a front-end. Back-end komponenty je přístupný v administraci redakčního systému a slouží administrátorovi například k vytváření, editaci obsahu webových stránek, či může plnit jakoukoliv jinou funkci, která pro ni byla naprogramována. Front-end se stará naopak o zobrazení dat na straně uživatele. Joomla byla navržena tak, že na každé stránce se může zobrazit pouze jedna komponenta.

4.2 Modul

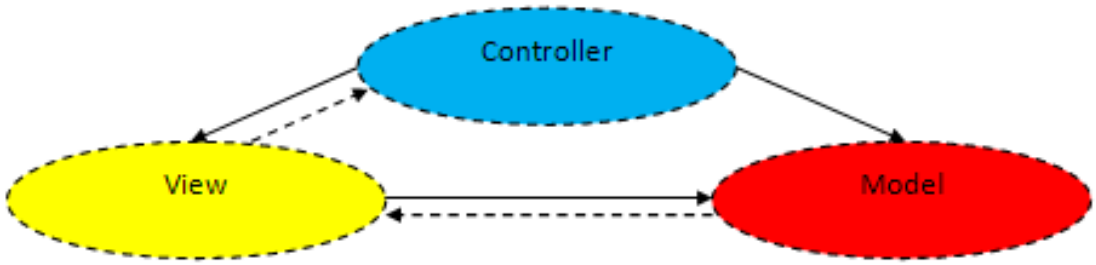
Oproti komponentám je jich možné zobrazit více na jedné stránce. Typickými moduly jsou například menu, ankety nebo také formuláře. Starají se o to, jak a kde se vypíší data vytvořená nějakou komponentou.

4.3 Plug-in

Plug-in se používá v případě, že potřebujeme vložit na stránku krátký kus zdrojového kódu. Často jsou plug-iny používány k formátování výstupu komponenty nebo modulu, například zvýrazňování klíčových slov.

5 Model-View-Controller

Model-View-Controller (zkráceně MVC) je softwarová architektura používaná ke strukturování zdrojového kódu aplikací, která rozděluje datový model, uživatelské rozhraní a řídící logiku do tří oddělených částí. Následně je možné upravovat každou část zvlášť s minimálním dopadem na části ostatní. Velkou výhodou využití tohoto vzoru je usnadnění testování výsledného software díky přehlednosti kódu. Využití této struktury je oceňováno hlavně v případě, kdy na určitém projektu pracuje více lidí. V Joomle je MVC vzor implementován použitím tří tříd: JModel, JView a JController.



Obrázek 1: Schéma softwarové architektury MVC

5.1 Model

Model je část komponenty, která zajišťuje přístup k datům a manipulaci s nimi. V této komponentě bude model například zajišťovat ukládání, editaci a mazání šablon pro vyhledávání.

5.2 View

View má na starost převod dat, která získá z modelu, do podoby vhodné k prezentaci pro uživatele. U webových aplikací se většinou jedná HTML stránku.

5.3 Controller

Controller je odpovědný za zpracování akcí uživatele, řídí tok událostí v programu.

6 Vývoj komponenty

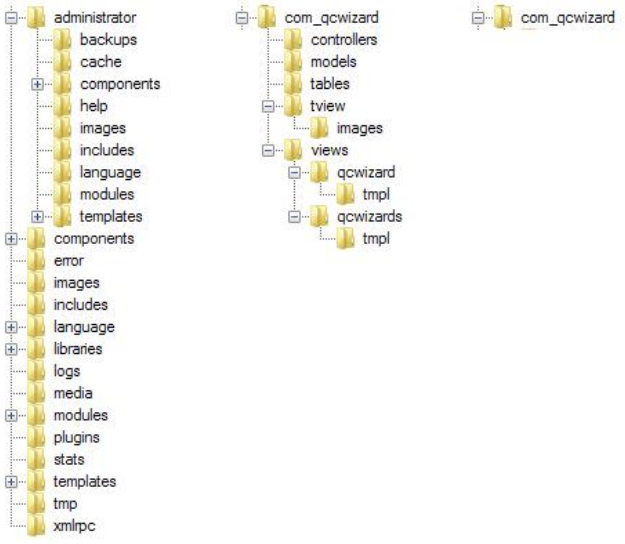
6.1 Požadavky komponenty

Při vývoji této komponenty a modulu bude využita Joomla! verze 1.5.20. Jelikož se jedná o komponentu, která má umožňovat vyhledávání zboží na základě zvolených parametrů, je nutné mít nainstalovánu komponentu VirtueMart, poskytující funkce e-shopu. Při vývoji komponenty byla zvolena ve verzi 1.1.5. Tyto verze Joomla! a VirtueMart jsem zvolil, protože na těchto verzích jsou e-shopy na internetu nejčastěji postaveny. V současné době je již dostupná sice verze Joomly 1.7, ale VirtueMart prozatím nemá stable verzi pro jinou než 1.5 verzi Joomly.

6.2 Adresářová struktura

Při vývoji komponenty se musíme řídit pevně daným schématem pojmenovávání jejích adresářů a souborů. Pro každou komponentu je vyžadováno unikátní jméno bez mezer. Pro tuto komponentu byl zvolen název QC Wizard. Celá komponenta je rozdělena do dvou adresářů, jejichž jméno je uvozeno prefixem com_, budeme tedy pracovat s adresáři *com_qcwizard*. První z těchto adresářů bude sloužit pro front-end část komponenty. Je umístěn ve složce *components*. Druhý adresář se nachází ve složce *administrator/components* a bude nám sloužit pro back-end. [1]

Na obrázku č.2 je uvedena adresářová struktura CMS Joomla!. První strom je root adresář celého webu, druhé dva jsou námi výše vytvořené adresáře pro back-end a front-end sekci komponenty.



Obrázek 2: Ukázka adresářové struktury

6.3 Souborový systém

Pro tuto komponentu byl vytvořen seznam souborů, který je uveden na listu Příloha B, z nichž každý má svou určitou funkci, která je v příloze stručně popsána.

Nepopsané soubory s názvem *view.html.php*, soubor *form.php* a *default.php* se starají o výpis dat, která zpracovává příslušný model, soubory *index.php* mají pouze bezpečnostní funkci (Kapitola 12.2).

Instalační balíček komponenty obsahuje také soubory ve složce */treeview* se skripty a styly pro vypsání dynamického stromu kategorií pomocí JavaScriptu.

6.4 Databáze

Systém Joomla! nejčastěji využívá databázový systém MySQL. Při vývoji této komponenty byl tedy též využit. Databáze je využívána pro uložení seznamu šablon a jejich vlastností. V databázi nainstalovaného systému Joomla! byly vytvořeny dvě tabulky. V první z nich (#__qc_templates) se ukládají jména šablon s jejich ID.

	Sloupec	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Extra	Akce
☐	id	int(11)			Ne		auto_increment	
☐	name	varchar(25)	utf8_general_ci		Ne			

Obrázek 3: Struktura tabulky #__qc_templates

Druhá tabulka *(#__qc_categories*) slouží k uložení informací o tom, jaké kategorie produktů souvisí s jakou šablonou pro vyhledávání. Sloupec *wizard_id* odkazuje na sloupec *id* z tabulky *#__qc_templates*, *category_id* odkazuje do tabulky komponenty VirtueMartu, kde jsou uloženy informace o kategoriích a sloupec *parent_id* odkazuje na identifikátor rodičovské kategorie.

	Sloupec	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Extra	Akce
☐	id	int(11)			Ne		auto_increment	
☐	wizard_id	int(11)			Ne			
☐	category_id	int(11)			Ne			
☐	parent_id	int(11)			Ne			

Obrázek 4: Tabulka #__qc_categories

Automatická instalace za nás vyřeší i vytvoření těchto dvou tabulek v databázi. Do instalačního balíčku musíme přidat textový soubor, který se bude jmenovat install.sql a jeho obsahem budou SQL příkazy pomocí nichž se tabulky vytvoří. Obsah souboru s příkazy, které vytvoří požadované tabulky je následující:

```
CREATE TABLE IF NOT EXISTS `my_qc_categories` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `wizard_id` int(11) NOT NULL COMMENT 'číslo průvodce',
  `category_id` int(11) NOT NULL COMMENT 'číslo kategorie, zařazené do
průvodce',
  `parent_id` int(11) NOT NaLL COMMENT 'číslo rodiče zařazené kategorie',
  PRIMARY KEY (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8;
```

Práci s databází v Joomla! nám velice usnadňuje její abstrakní databázová vrstva, která usnadňuje práci s daty. Jelikož Joomla! podporuje různé SQL databáze, použitím této třídy předejdeme mnoha problémům a náš kód se stane flexibilní a kompatibilní bez závislosti na databázi. Práci s databází začínáme tím, že si vytvoříme instanci objektu databáze. Následně můžeme začít provádět požadované dotazy na DB.

```
$db =& JFactory::getDBO(); // Vytvoření instance
$sql = "SELECT * FROM #__qc_categories;";  // Vytvoření SQL dotazu
$db->setQuery($sql); // Připraví dotaz na pozdější
```

V názvu databáze se objevuje symbol křížku #. Ten nám nahrazuje prefix jména databáze (ve verzi Joomla! 1.5 standardně jos_), který si můžeme zvolit při instalaci CMS.

Různé SQL databáze se liší v drobnostech, jako je používání rozličných druhů apostrofů. Například MySQL používá pro oddělení názvů sloupců zpětné apostrofy `` a pro hodnoty v dotazu používá jednoduché apostrofy ''. Abychom se vyhnuli chybám způsobeným touto syntaxí, naše databázová třída disponuje funkcemi (nameQuote(), quote()), které toto řeší za nás.

```
$sql = "SELECT * FROM " . $db->nameQuote('#__qc_categories') . " WHERE " .
$db->nameQuote('id') . " = " . $db->quote('1') . ";";
```

V případě, že již máme dotaz připraven, budeme ho chtít nějakým způsobem exekuovat. Pro spuštení dotazu, kdy nepotřebujeme získat žádná data a postačí nám jen informace o tom, zda příkaz v pořádku proběhl (TRUE/FALSE), použijeme funkci query().

```
$result = $db->query(); // Spustí dříve připravený SQL dotaz
```

Databázová třída nám nabízí několik možností, jakým způsobem získávat z DB data. [2]

- loadResult() – Získává jedinou hodnotu ze sloupce na základě SQL dotazu, často se využívá pro získání počtu záznamů COUNT
    - k výsledku přistupujeme pomocí

```
$count = $db->loadResult();
```

- loadRow() – Získává všechny hodnoty z jediného určitého řádku a vrací je jako indexované pole
    - k výsledku přistupujeme pomocí

```
$row = $db->loadRow();
$row['index'];
```

- loadAssoc() – Získává všechny hodnoty z jediného určitého řádku stejně jako loadRow(), ale vrací je jako asociované pole (key => value)
    - k výsledku přistupujeme pomocí

```
$row = $db->loadAssoc();
$row['name'];
```

- loadObject() - Získává všechny hodnoty z jediného určitého řádku stejně jako loadRow() a loadAssoc(), ale vrací je jako objekt
    - k výsledku přistupujeme pomocí

```
$row = $db->loadAssoc();
$row['name'];
```

- loadResultArray() – Získává indexované pole jednotlivých hodnot ze sloupce
    - k výsledku přistupujeme pomocí

```
$column= $db->loadResultArray();
$column['2'];
```

- loadResultArray($index) – Vrací indexované pole jednotlivých hodnot ze sloupce tabulky na základě indexu
    - k výsledku přistupujeme pomocí

```
$column= $db->loadResultArray(1);
$column['2'];
```

- loadRowList() – Vrací indexované pole indexovaných polí
  - k výsledku přistupujeme pomocí

```
$row = $db->loadRowList();
$row['0']['1'];
```

- loadAssocList() – Vrací indexované pole asociovaných polí
  - k výsledku přistupujeme pomocí

```
$row = $db->loadAssocList();
$row['1']['jmeno'];
```

- loadAssocList($key) – Vrací indexované pole asociovaných polí
  - k výsledku přistupujeme pomocí

```
$row = $db->loadAssocList('znacka');
$row['Mercedes']['SPZ'];
```

6.5 jQuery v administraci

Pro výpis stromu kategorií byla zvolena JavaScriptová knihovna jQuery. Nastal ale konflikt s jinou knihovnou, která je používána administrací Joomly – knihovnou Mootools. Proto je nutné pracovat s jQuery v nekonfliktním módu. To zajistíme vložením řádku:

```
jQuery.noConflict();
```

V případě, že bychom nepoužívali nekonfliktní mód, zdrojový kód pro inicializaci stromu kategorií pomocí jQuery by vypadal následovně:

```
$ (document).ready(function() {
    $ ("#browser").treeview({
        collapsed: true,
        animated: 250,
        persist: "cookie"
    });
});
```

Jediné, co je potřeba změnit při jeho použití, je záměna symbolu dolaru *$* za řetězec *jQuery*:

```
jQuery(document).ready(function() {
    jQuery("#browser").treeview({
        collapsed: true,
        animated: 250,
        persist: "cookie"
    });
});
```

6.6 Komentáře

Pro lepší orientaci a hlavně do budoucna k úpravám zdrojových kódů byly okomentovány některé důležité funkce následujícím způsobem:

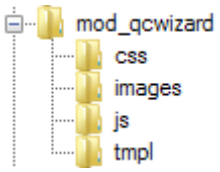
```
/**
 * Metoda ukládající záznam
 *
 * @access public
 * @return boolean True při úspěšném uložení
 */
```

7 Vývoj modulu

O vypsání vyhledávacího formuláře pro návštěvníka e-shopu se nám postará jednoduchý modul. Byl vytvořen nejzákladnější modul, který se skládá z konfiguračního souboru XML a z PHP souboru s kontrolerem.

7.1 Souborový systém

Při vývoji modulu se musíme řídit podobnými pravidly jako při vývoji komponenty. Jako u komponent platí, že hlavní adresář je ve tvaru *mod_jmenomodulu* (v tomto případě *mod_qcwizard*).



Obrázek 5: Struktura adresáře modulu

Souborový systém modulu se skládá ze souborů uvedených v části Příloha C.

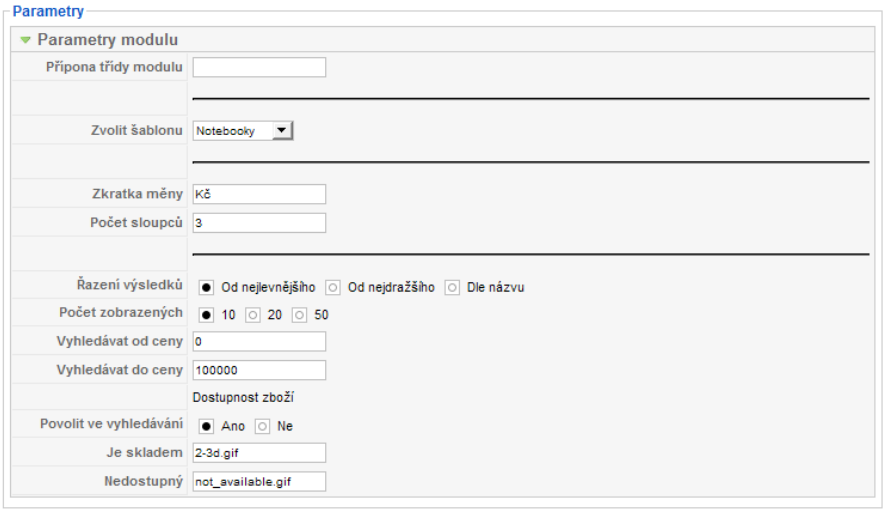
7.2 Nastavení modulu

O možnost změn chování tohoto modulu se stará konfigurační XML soubour. V tomto konfiguračním XML souboru jsou uloženy informace o modulu, které budou zobrazeny v administrátorském rozhraní a administrátor je bude moci editovat.

Tento modul má možnost nastavit si několik vlastností, které ovlivňují výsledky vyhledávání produktů v e-shopu. Obsah takového XML souboru nám představuje následující výňatek:

```
<params>
    <param type="radio" name="sort" label="SORT" default="a">
       <option value="a">Od nejlevnějšího</option>
       <option value="b">Od nejdražšího</option>
       <option value="c">Dle názvu</option>
    </param>
    <param type="radio" name="count" label="COUNT" default="20">
       <option value="10">10</option>
       <option value="20">20</option>
```

```
        <option value="50">50</option>
    </param>
    <param name="min" type="text" default="0" label="MIN" />
</params>
```



Obrázek 6: XML soubor s parametry a výstup v administraci

Všechny parametry jsou nyní napevno uloženy v XML, tak tu nyní vzniká problém, jak načítat seznam šablon vyhledávání, který vytváříme v komponentě. Joomla! však podporuje vkládání SQL dotazů do konfiguračního souboru a tak můžeme dynamicky načítat aktuální seznam šablon. Způsob vložení SQL dotazu do XML provádíme následovně:

```
<param
    name="parent" type="sql" default="id" label="SELECTCOM"
    description="SELECTCOM2" query="SELECT id, name FROM #__qc_templates"
    key_field="id" value_field="name"
/>
```

7.3 Multijazyčná komponenta

V případě, že bychom chtěli vytvářet multijazyčnou verzi modulu, je možné do balíku připojit soubory s danou lokalizací:

```
<languages>
    <!-- Any language files included with the module -->
    <language tag="cs-CZ">cs-CZ.mod_qcwizard.ini</language>
</languages>
```

Daný jazyk modulu se poté řídí nastavením jazyka celé administrace. Pokud není nalezena jazyková mutace modulu, která je použita v administraci, použije se defaultní. Každý řádek představuje jeden výraz a je složen z klíče a překladu. Níže je uvedeno několik řádků ze souboru s překladem na ukázku:

```
CURRENCY=Zkratka měny
COLUMNS=Počet sloupců
SORT=Řazení výsledků
MIN=Vyhledávat od ceny
MAX=Vyhledávat do ceny
```

7.4 Úprava <head>

Abychom mohli používat knihovnu jQuery a dodatečné kaskádové styly, je nutné vložit odkaz do sekce <head>. Samozřejmě by šlo vložit odkazy například do šablony, ale to by správce e-shopu musel zasahovat do zdrojového kódu a to je u tvorby rozšíření nechtěná komplikace. Proto Joomla! obsahuje možnost vložit tam tyto odkazy automaticky, poslouží nám k tomu třída JDocument.

7.5 JDocument

Třída JDocument nám pomůže s vkládáním CSS a JavaScriptových souborů, JavaScriptových kódů, též nám může pomoci s čtením či zapisováním metadat dané stránky. V dalších několika odstavcích budou představeny nejdůležitější a v této komponentě použité funkce této třídy. K instanci třídy přistupujeme pomocí příkazu:

```
$document =& JFactory::getDocument();
```

Nyní máme dostupné tyto funkce:

- `$document->addScript();` - vloží odkaz na JavaScriptový soubor
- `$document->addScriptDeclaration();` - vloží JavaScriptový kód
- `$document->addStyleSheet();` - umožňuje do hlavičky vložit odkaz na kaskádové styly
- `$document->addStyleDeclaration ();` - umožňuje do hlavičky přímo vložit CSS styly
- `$document->setMetaData();` - setMetaData nám umožňuje změnit metadata dané stránky
- `$document->getMetaData();` - tato funkce nám umožní metadata dané stránky přečíst

Následující ukázkový zdrojový kód inicializuje instanci dokumentu a postupně vkládá do výsledného HTML kódu CSS soubor, JavaScriptový soubor a nakonec přímou definici JavaScriptové funkce:

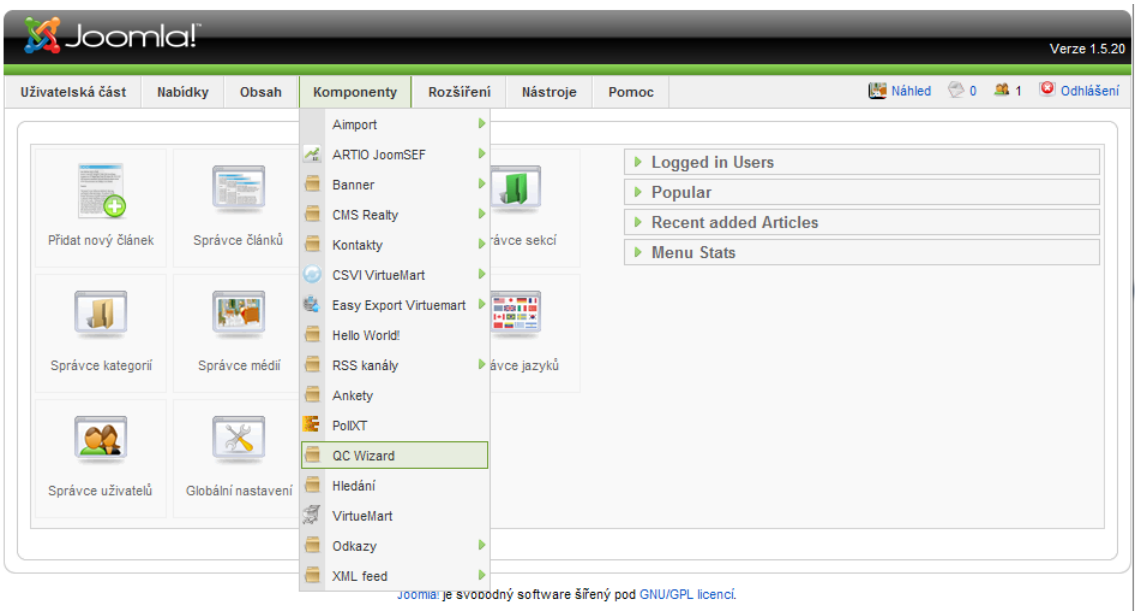
```
$document =& JFactory::getDocument();
$document->addStyleSheet("../jquery.treeview.css");
$document->addScript("../jquery.js");
$document->addScriptDeclaration('jQuery(document).ready(function() {
    jQuery("#browser").treeview({
        collapsed: true,
        animated: 250,
        persist: "cookie"
        });
    });
');
```

Výsledek předchozích PHP příkazů je HTML kód, který je odeslán uživateli do prohlížeče:

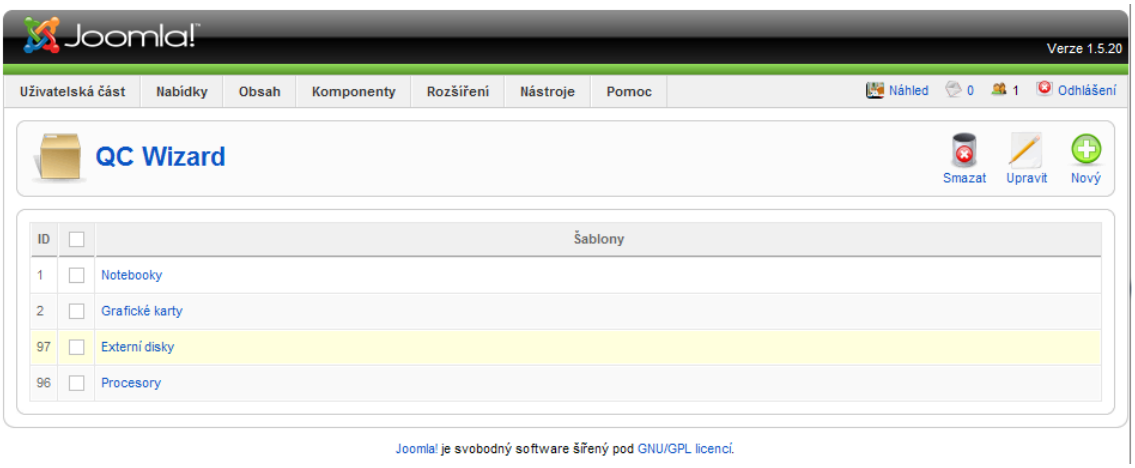
```
<link rel="stylesheet" href="../jquery.treeview.css" type="text/css" />
<script type="text/javascript" src="../jquery.js"></script>
<script type="text/javascript">
    jQuery(document).ready(function() {
        jQuery("#browser").treeview({
            collapsed: true,
            animated: 250,
            persist: "cookie"
        });
    });
</script>
```

8 Administrační část (back-end)

Administrační část systému Joomla! umožňuje správcům webu spravovat jeho obsah. Obsahuje řadu funkcí jako přidávání článků, tvorbu anket, správu produktů v e-shopu atd. V případě této komponenty se budou editovat šablony pro vyhledávání. Na obrázku č.7 je hlavní stránka administračního rozhraní CMS Joomla!. V horní části je hlavní menu, přes které se dá dostat ke všem funkcím systému. Naší komponentu najdeme v sekci Komponenty pod názvem QC Wizard.



Obrázek 7: Hlavní stránka administračního rozhraní Joomly s umístěním komponent

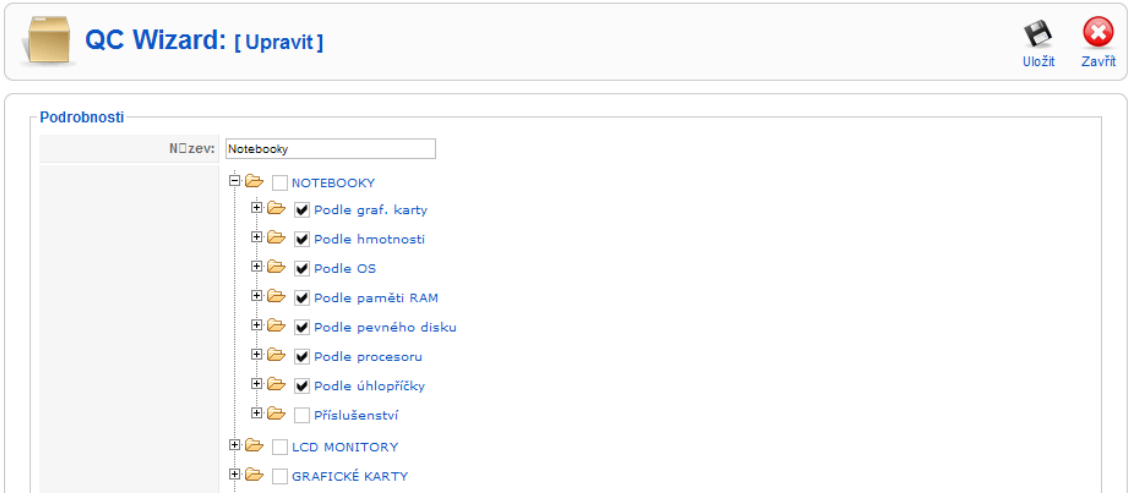


Obrázek 8: Hlavní stránka komponenty

Po kliknutí na položku QC Wizard se nám otevře editační prostředí pro šablony. V tabulce vidíme všechny již vytvořené šablony, pomocí tlačítek v horním pravém rohu

však můžeme přidat další, editovat stávající a nebo smazat jednu či více šablon najednou.

Jelikož byla komponenta vyvíjena a testována na e-shopu, který se zabývá prodejem výpočetní techniky, byly vytvořeny 4 testovací šablony pro vyhledávání (Notebooky, Grafické karty, Externí disky, Procesory).

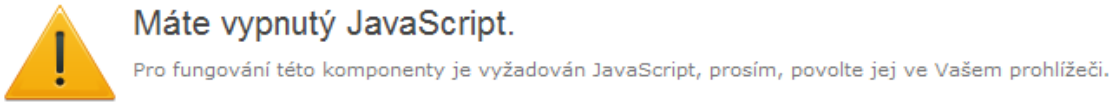


Obrázek 9: Úprava šablony

Na obrázku č. 9 je již editační formulář jediné kategorie. Zvolena byla kategorie Notebooky, kde se po jejím rozkliku vypíší její podkategorie, z kterých následně můžeme sestavit šablonu pro vyhledávání. Podle obrázku je nyní označeno sedm podkategorií, které se následně využijí jako filtr pro vyhledávání. Zde se musí počítat s rozumným označením kategorií. Tím je myšleno, že abychom dostali funkční filtr, nesmí být označeny v jenom filtru kategorie například z notebooků a zároveň z monitorů. To bychom logicky nemohli dostat žádný výsledek, jelikož tyto dva produkty nemají žádnou společnou kategorii.

9 Uživatelská část (front-end)

Uživateli této komponenty jsou mimo majitelů e-shopu hlavně jejich zákazníci. Ta jim dodává větší komfort při elektronickém nakupování. Je určena především pro ty zákazníky, kteří vědí, jaké má produkt splňovat parametry, jelikož komponenta umožňuje vyhledávání v nepřeberném množství produktů právě na základě různých parametrů. Všemu dodává ještě větší komfort okamžité zobrazení výsledků vyhledávání díky technologii AJAX. Jak bylo uvedeno v úvodu práce, je očekáván na uživatelské stanici moderní prohlížeč a zapnutý JavaScript. V případě jeho absence není uživateli umožněno vyhledávání a je zobrazena chybová hláška s upozorněním.



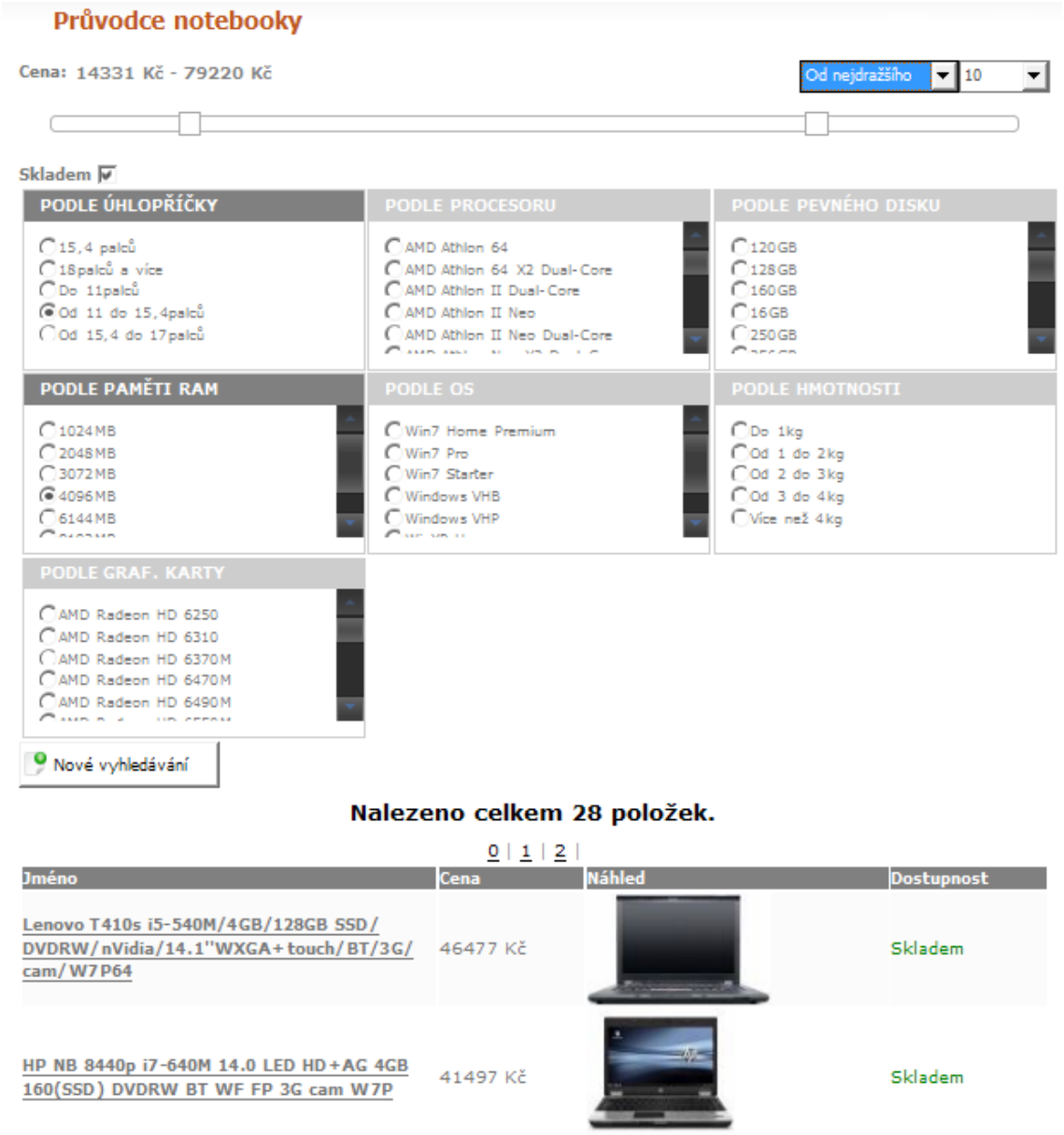
Obrázek 10: Chybová hláška

Pokud je splněna podmínka zapnutého JavaScriptu, nic nebrání vyhledávání zboží. Na následujícím obrázku č. 11 je snímek uživatelského rozhraní.

Seznam prvků uživatelského rozhraní:

- Uživatel si může pomocí posuvníku vybrat rozmezí ceny, v jaké se bude výsledný produkt vyskytovat. Posuvník je realizován pomocí jQuery utility jménem Slider.
- K určení, podle jaké logiky se budou vypisovat produkty, slouží HTML prvek Select. Produkty můžeme řadit od nejlevnějších, nejdražších, podle názvu.
- Počet zobrazovaných produktů ovlivníme HTML prvkem Select, který nabízí možnost 10, 20, 50 produktů na jednu stránku.
- V případě, že zákazník bude chtít nakupovat pouze z produktů, které má obchod skladem, může zakliknout možnost Skladem.

Dále se nám zobrazí kategorie a jejich prvky, podle kterých budeme vyhledávat. V případě zakliknutí nějakého prvku, se nám pro lepší přehled podbarví název kategorie. Pokud se zákazník rozmyslí a bude chtít začít nové vyhledávání, slouží k resetování formuláře tlačítko Nové vyhledávání. V případě, že by průběh SQL dotazu trval déle, je po dobu jeho trvání zobrazen grafický progressbar.



Obrázek 11: Uživatelské rozhraní (možnosti filtrování s aktuálním seznamem odpovídajících produktů)

9.1 Prezentace front-endu

Vyzkoušet možnosti vyhledávání je možné na e-shopu pcprovsechny.cz s reálnými produkty. V databázi se běžně vyskytuje kolem 10000 produktů rozdělených do několika kategorií. Formulář pro vyhledávání mezi notebooky (průměrně 420 produktů v DB) je následující: http://www.pcprovsechny.cz/konfigurator.

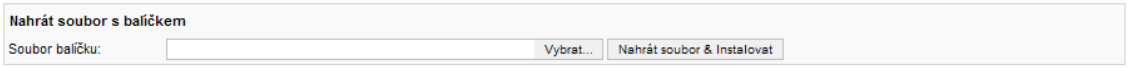
10 Instalace

10.1 Kde získat komponentu a modul

Pokud bude mít kdokoliv zájem si nainstalovat tuto komponentu s modulem, má možnost si je stáhnout ze serveru GitHub, využívající verzovací systém Git. Na tomto serveru bude vždy ta nejaktuálnější verze zdrojových kódů. Odkaz na repozitář komponenty s modulem je: https://github.com/karelbartunek/QC-Wizard.

10.2 Administrátor e-shopu

Aby nám Joomla! usnadnila práci při instalaci rozšíření, obsahuje instalačního průvodce. Pokud chceme nainstalovat modul, komponentu či třeba nový design webu, instalace nám zabere jen pár sekund. Vývojáři uvolňují svá díla zazipovaných souborech. V administraci Joomla! pak stačí v sekci Instalace vybrat daný ZIP soubor a kliknout na tlačítko Instalovat. Vše již zařídí Joomla! a my budeme moci začít používat dané rozšíření.



Obrázek 12: Instalace rozšíření

Jediné, co bude muset správce e-shopu udělat po instalaci komponenty a modulu vytvořeného v této bakalářské práci, je nakopírování souboru *qcwizard.php*, který se nachází v adresáři */administrator/components/com_qcwizard/tmpl* do adresáře */templates/system*. Bohužel nainstalování tohoto potřebného souboru do adresáře */templates/system* instalátor systému Joomla! neumí.

10.3 Vývojář

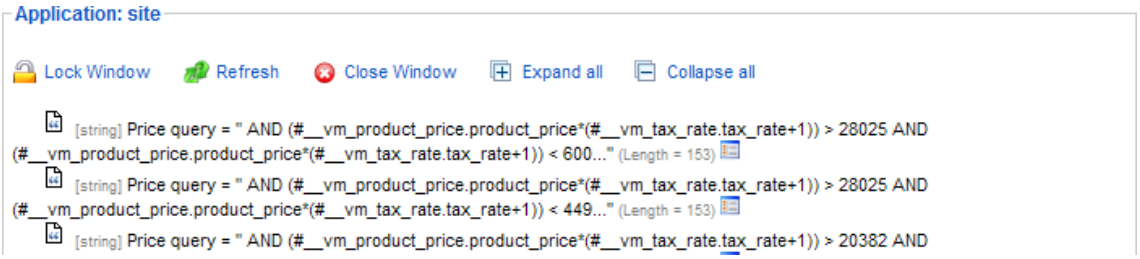
Aby administrátoři e-shopu mohli jednoduše na pár kliknutí nainstalovat dané rozšíření, vývojář musí dodržet jistá pravidla. Proto se pro instalaci vytváří balíček souborů, který je následně zazipován. Instalace se řídí podle souboru XML (viz. strana 33), který obsahuje všechna důležitá data. Platí při tom, že XML soubor v balíčku musí mít název ve tvaru (v tomto případě pro modul) *mod_qcwizard.xml*. XML soubor obsahuje všechny informace o rozšíření, jméno, licenci, informační text, seznam všech souborů v administrátorské i uživatelské části atd.

Ukázka z instalačního souboru modulu komponenty QC Wizard může vypadat takto:

```
<?xml version="1.0" encoding="utf-8"?>
<install type="module" version="1.5.0">
    <!-- Name of the Module -->
    <name>QC Wizard</name>
    <!-- Name of the Author -->
    <author>Karel Bartůněk</author>
    <!-- Version Date of the Module -->
    <creationDate>2011-05-22</creationDate>
    <!-- Module version number -->
    <version>1.0.0</version>
    <files>
        <filename module="mod_qcwizard">mod_qcwizard.php</filename>
        <filename>index.html</filename>
        <filename>helper.php</filename>
    </files>
    <languages>
        <!-- Any language files included with the module -->
        <language tag="cs-CZ">cs-CZ.mod_qcwizard.ini</language>
    </languages>
    </params>
</install>
```

11 Ladění kódu – Debug

V průběhu vývoje komponenty bylo nutné dbát na ošetření chyb, aby nebyla narušena základní funkčnost celého systému. Samotná základní instalace systému Joomla! obsahuje v administraci možnost zapnutí módu Debug. Po zapnutí tohoto módu budou v patičce stránky zobrazovány diagnostické informace, překlady jazyka, SQL dotazy a jejich chyby. Mód se vztahuje jak na uživatelskou sekci, tak na administrátorskou.



Obrázek 14: Komponenta J!Dump

Při vývoji nastaly situace, kdy bylo potřeba vypisovat různé proměnné a sledovat jejich hodnoty. Zpočátku stačilo využívání PHP funkcí `print_r()`, `dump()` či `var_dump()`. Aby se nám usnadnila práce s tímto vypisováním a sledováním aktuálních hodnot, byla zvolena pro instalaci komponenta, která by vše přehledně zobrazila. Zvolena byla komponenta a její plugin jménem J!Dump. Ta umožnila pomocí PHP funkce `dump()` vypisovat v dialogovém okně všechny důležité informace. Syntaxe pro vyvolání takové okna je velice jednoduchá, pouze PHP funkce s parametry.

```
dump($variable, "Variable Name");
```

Po ukončení vývoje je nutné dbát na to, aby všechny ladící funkce byly vypnuty a nezobrazovali se návštěvníkům webových stránek. Pro obyčejného návštěvníka je takový výpis nepotřebný až obtěžující, v horším případě je takový výpis proměnných zneužitelný a použitelný k odhalení slabých míst v systému.

12 Bezpečnost aplikace v Joomla!

Na bezpečnost je v systému Joomla! kladen velký důraz. Autoři, kteří se podílejí na tvorbě Joomla! vydávají nové a nové aktualizace systému, přičemž velice často řeší bezpečnostní otázky. Jelikož jsou zdrojové kódy CMS a mnoha komponent volně dostupné, tak mají útočníci zjednodušenou práci, protože vědí, jak co funguje. Je tedy nutné dodržovat určitá pravidla, abychom takovým lidem, kteří chtějí poškodit či jinak zneužít naše webové stránky, znepříjemnili až znemožnili jejich nekalou snahu. V následující podkapitole bude uveden příklad jedné z mnoha základních ochran před takovýmito praktikami, která byla v této práci použita.

12.1 Ochrana proti přímému přístupu k souboru (Direct Access)

Jednou z možných taktik hackerů může být zkoušení přímého přístupu k souborům. V CMS Joomla! by mohl vypadat přímý přístup k souborům této komponenty například takto:*www.domena/components/com_qcwizard/qcwizard.php*.

Tím by útočník docílil spuštění vstupního souboru komponenty v uživatelském či i administrátorském rozhraní. Abychom tomu předešli, vkládáme na úplný začátek každého PHP souboru řádek, který přímé spuštění nedovolí:

```
defined('_JEXEC') or die('Direct Access to this location is not allowed.');
```

Při spustění Joomla! pohlídá, zda byl soubor volán jádrem systému a pokud ne, pouze vypíše uvedenou chybovou hlášku.

12.2 Ochrana proti Directory Listing

V seznamu souborů v příloze B se nám vyskytuje několik souborů, které se jmenují *index.html*. Ty mají za úkol se zobrazit, pokud se uživatel pokouší vstoupit do adresáře, do kterého nechceme, aby se dostal. Některé servery jsou totiž nastaveny tak, že vypisují obsah adresáře, což může být do jisté míry bezpečnostní riziko, protože návštěvník by mohl otevřít jakýkoliv soubor umístěn v daném adresáři a číst i soubory, které chceme před veřejností skrýt. Použitím *index.html* tedy zvyšujeme bezpečnost systému. Pokud ale máme v adresářích opravdu důležitá data, která chceme před veřejností skrýt, doporučuje se nespoléhat na tento způsob ochrany a použít jako ochranu soubor *.htaccess*. Soubor *index.html* obsahuje jediný řádek HTML kódu:

```
<html><body bgcolor="#FFFFFF"></body></html>
```

13 Závěr

Díky této bakalářské práci vznikla univerzální komponenta s modulem, které je možné nainstalovat na jakýkoliv e-shop postavený na kombinacích systémů Joomla! + Virtuemart, jelikož byla dodržována pravidla softwarové architektury Model-View-Controller s použitím funkcí frameworku CMS Joomla!.

V bakalářské práci byl na úvod ve druhé kapitole stručně popsán použitý redakční systém Joomla! a jeho softwarové požadavky. Dále byly představeny technologie, které byly potřebné k vývoji komponenty a modulu. Podrobněji byly věnovány stránky vývoji komponenty v šesté kapitole. Byly zmíněny hlavní problémy a jejich řešení, které vedlo k úspěšnému dokončení práce. Vývoj modulu potřebného pro zobrazení dat uživateli e-shopu a který umožňuje administrátorovi měnit chování vyhledávacího formuláře byl popsán v kapitole sedm.

Následně bylo uvedeno, kde se dají vytvořené zdrojové kódy stáhnout a jak je nasadit do provozu na internetový obchod.

Pro dosažení dobrých praktických výsledků bylo vše během vývoje projednáváno s provozovateli internetového obchodu s výpočetní technikou pcprovsechny.cz. Komponenta s modulem je již úspěšně nasazena v tomto internetovém obchodě, zvyšuje komfort pro uživatele při nákupu a hlavně má svůj podíl na celkovém počtu konverzí (objednávek).

Vývoj této komponenty by měl a bude v budoucnu směřovat k jejímu přepracování pro nové verze systému Joomla! (1.7+) a VirtueMart 2 a rozšiřování funkcionality podle nových požadavků provozovatelů zmíněného internetového obchodu.

Zdroje informací

[1] LEBLANC Joseph. Learning Joomla! 1.5 Extension Development. Birmingham: Packt Publishing Ltd., 2007. ISBN 978-1-847191-30-4

[2] Joomla Framework API documentation [online]. URL: < http://docs.joomla.org/Framework >.

[3] Joomla component development [online]. URL: < http://docs.joomla.org/Component_Development >.

[4] Dokumentace k PHP [online]. URL: < http://www.php.net/docs.php >.

Příloha A - obsah přiloženého CD

Přiložené CD obsahuje adresáře s daty:

- Sources – Kompletní zdrojové kódy
- Archive – Instalační balíčky modulu a komponenty
- Document – Bakalářská práce ve formátu PDF

Příloha B – seznam souborů komponenty QC Wizard

- */administrator/components/com_qcwizard/controller.php* – základní controller
- */administrator/components/com_qcwizard/index.html*
- */administrator/components/com_qcwizard/install.sql* – soubor s MySQL příkazy, díky nimž se vytvoří potřebné tabulky v databázi
- */administrator/components/com_qcwizard/qcwizard.php* – vstupní bod komponenty
- */administrator/components/com_qcwizard/uninstall.sql* – soubor s MySQL příkazy, které zajistí odstranění dat a tabulek při odinstalování komponenty ze systému
- */administrator/components/com_qcwizard/controllers/index.html*
- */administrator/components/com_qcwizard/controllers/qcwizard.php*
- */administrator/components/com_qcwizard/models/index.html*
- */administrator/components/com_qcwizard/models/qcwizard.php* – model, který zajišťuje práci s aktuální šablonou
- */administrator/components/com_qcwizard/models/qcwizards.php* – tento model obsahuje funkce pro práci se seznamem šablon
- */administrator/components/com_qcwizard/tables/index.html*
- */administrator/components/com_qcwizard/tables/qcwizard.php* – obsahuje definici třídy, která představuje tabulku v databázi
- */administrator/components/com_qcwizard/views/qcwizard/index.html*
- */administrator/components/com_qcwizard/views/qcwizard/view.html.php*
- */administrator/components/com_qcwizard/views/qcwizard/tmpl/form.php*
- */administrator/components/com_qcwizard/views/qcwizard/tmpl/index.html*
- */administrator/components/com_qcwizard/views/qcwizards/index.html*
- */administrator/components/com_qcwizard/views/qcwizards/view.html.php*
- */administrator/components/com_qcwizard/views/qcwizards/tmpl/default.php*
- */administrator/components/com_qcwizard/views/qcwizards/tmpl/index.html*
- */components/com_qcwizard/qcwizard.php*

Příloha C – souborový systém modulu QC Wizard,

- */modules/mod_qcwizard/helper.php* – soubor s pomocnými funkcemi
- */modules/mod_qcwizard/index.html*
- */modules/mod_qcwizard/mod_qcwizard.php* – vstupní bod modulu
- */modules/mod_qcwizard/mod_qcwizard.xml* – konfigurační soubor
- */modules/mod_qcwizard/qcwizard.css* – soubor se styly
- */modules/mod_qcwizard/css/showLoading.css* – soubor se styly
- */modules/mod_qcwizard/js/jquery.cookie.js* – JavaScriptová knihovna
- */modules/mod_qcwizard/js/jquery.showLoading.min.js* – JavaScriptová knihovna
- */modules/mod_qcwizard/tmpl/default.php* – soubor s HTML
- */modules/mod_qcwizard/tmpl/index.html*