Belbinův test online

## Technická dokumentace

# Popis architektury

Aplikace je napsaná v jazyce PHP a funguje na principu MVC (mode-view-controller). Základní funkčnost zajišťuje PHP framework **zEngine**. K chodu aplikace je potřeba mySQL databáze.

## Požadavky na běh aplikace

* Webový server Apache 2 nebo adekvátní alternativa
  + mod\_rewrite
  + PHP 7
* mySQL 5
* moderní prohlížeč s podporou HTML5 a CSS3 (Firefox, Chrome, ...)

## Popis běhu aplikace využívající zEngine

Když webový server obdrží požadavek na webovou stránku, je tento požadavek pomocí mod\_rewrite přesměrován na soubor index.php, původní požadovaná cesta je vložena do proměnné **path**, kterou zEngine použije pro určení toho, jaký controller a view použít.

Celý požadavek je poté zpracován v metodě **run()** ve třídě **zEngine**, která je srdcem celé aplikace. Tato třída je velice minimalistická, její jediný úkol je načíst použité moduly a poté ve správném pořadí zavolat klíčové metody z modulu **core**, který jako jediný je pro běh aplikace nezbytný.

Modul **core** se nejprve pokusí najít vhodný controller v adresáři aplikace a pokud jej nenalezne, použije controller z adresáře zEngine. Pokud není controller nalezen ani v jednom adresáři, dojde k přesměrování na výchozí adresu pro nenalezenou stránku (lze nastavit v konfiguračním souboru pro modul **core**).

Controllery, stejně jako view, jsou ve skutečnosti tři – master, main a page – a vykonávají se v tomto pořadí. Ve výchozím stavu zEngine předpokládá, že název view je identický s názvem controlleru, ale toto lze v controlleru změnit zavoláním metody **setPageTemplate**. V controlleru je potřeba provést potřebnou aplikační logiku – načíst data z databáze apod. Pomocí metody **setData** posílá controller do view potřebná data.

View se také vykonávájí v pořadí master, main, page. Úkolem view je vygenerovat html kód, který je poslán prohlížeči. Data poslaná z controlleru si view načte pomocí metody **getData**.

V módu debug (lze nastavit v konfiguračním souboru pro modul **core**) se všechny chyby vypisují do prohlížeče, v opačném případě se chyba pouze zapíše do souboru a prohlížeč je přesměrován na obecnou stránku „Chyba“ (lze rovněž nastavit).

### Controller

Controllery jsou uloženy v adresářích s názvem controllers a jejich název má vždy tvar **<controllername>.c.php**. Tyto adresáře jsou vždy tři (**master**, **main** a **page**) na celkem čtyřech místech. Samotný framework má jeden adresář controllers pro veřejnou část a jeden pro administrátorskou část. Samotná aplikace má také jeden adresář controllers pro veřejnou část a jeden pro administrátorskou část. Framework na základě adresy rozhodne, které controllery budou vykonány. Administrátorská část je odlišena specifickým začátkem adresy (ve výchozím stavu všechny stránky administrátorské části začínají **/admin/**). Při výběru controlleru se framework nejprve podívá, zda požadovaný controller je k dispozici v aplikaci, až poté, zda existuje ve frameworku.

*Příklad výběru controlleru pro stránku (****page****):*

Základní adresa aplikace je [http://belbin.zavadil.eu](http://belbin.zavadil.eu/), uživatel přejde na adresu <http://belbin.zavadil.eu/prihlaseni>. Název page controlleru, který se zEngine pokusí nalézt, bude **prihlaseni**. Nejprve se podívá do adresáře **belbin/app/controllers/page**, zda tam nejde soubor **prihlaseni.c.php.** Pokud jej tam nenalezne, podívá se do adresáře **zEngine/app/controllers**.

Controllery se volají v pořadí **master**, **main** a nakonec **page**.

Controllery nemusí nutně existovat. Některé stránky nevyžadují žádnou specifickou serverovou logiku a controller nemají. Framework v takovém případě žádný controller nespustí a rovnou přejde k renderování view.

PHP kód controlleru lze začít psát přímo dovnitř souboru, není třeba definovat žádné třídy nebo metody. Proměnná **$this** použitá uvnitř controlleru ukazuje na modul **core**.

### View

Views jsou uloženy v adresářích s názvem views a jejich název má vždy tvar **<viewname>.v.php**. Tyto adresáře jsou vždy tři (**master**, **main** a **page**) na celkem čtyřech místech. Samotný framework má jeden adresář views pro veřejnou část a jeden pro administrátorskou část. Samotná aplikace má také jeden adresář views pro veřejnou část a jeden pro administrátorskou část.

Framework na základě adresy rozhodne, která view budou renderovány. Administrátorská část je odlišena specifickým začátkem adresy (ve výchozím stavu všechny stránky administrátorské části začínají **/admin/**). Při výběru view se framework nejprve podívá, zda požadovaný view je k dispozici v aplikaci, až poté, zda existuje ve frameworku.

Jako první se začne renderovat **master** view. Někde uvnitř tohoto view se předpokládá (není to však nutnost), že bude zavolána metoda **$this→renderMainView()**, která vyrenderuje **main** view. Uvniř **main** view bude, opět volitelně, vyrenderováno **page** view pomocí metody **$this→renderPageView().**

HTML kód controlleru lze začít psát přímo dovnitř souboru, není třeba definovat žádné třídy nebo metody. Proměnná **$this** použitá uvnitř view ukazuje na modul **core**.

## Popis souborového systému

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **zEngine** | | | | Soubory frameworku. Tento adresář musí být připojen ke každé aplikaci využívající zEngine. |
|  | **src** | | | Zdrojové kódy frameworku. |
|  |  | **app** | | Zdrojové kódy v PHP. |
|  |  |  | **admin** | Administrační část. Základ pro administraci webů. |
|  |  |  | **controllers** | Kontrolery pro MVC. |
|  |  |  | **jobs** | Samostatně spustitelné úlohy pro správu. Spouští se např. pomocí cron. |
|  |  |  | **models** | Modely pro MVC. |
|  |  |  | **resources** | Výchozí CSS styly a skripty v javascriptu. |
|  |  |  | **views** | Views pro MVC. |
|  |  | **classes** | | Nejrůznější třídy v PHP usnadňující práci. |
|  |  | **modules** | | Moduly frameworku – základní stavební kameny aplikací. |
|  | **sql** | | | SQL skripty pro vytvoření databáze. |
|  | **tests** | | | Testy. |
| **belbin** | | | | Soubory samotné aplikace. |
|  | **prezentace** | | | Prezentace projektu vytvořená pro předmět PRE. |
|  | **www** | | | Soubory, které musí být nakopírovány na www server. |
|  |  | **app** | | Soubory aplikace. Tento adresář by z bezpečnostních důvodů neměl být přes prohlížeč přístupný. |
|  |  |  | **admin** | Administrační část. Přístup do této části webu vyžaduje přihlášení správce. |
|  |  |  | **config** | Konfigurační soubory. Veškerá konfigurace aplikace se provádí editací konfiguračních souborů v tomto adresáři. |
|  |  |  | **controllers** | Kontrolery pro MVC. |
|  |  |  | **lang** | Soubory s lokalizací (překladem) aplikace. |
|  |  |  | **models** | Modely pro MVC. |
|  |  |  | **views** | Views pro MVC. |
|  |  | **public** | | Tento adresář obsahuje zásadní soubor index.php. Tento adresář musí ve webovém serveru nastaven jako výchozí. |
|  | **zadani** | | | Zadání projektu ve formátu html. |

## Instalace

Tento návod předpokládá, že používáte server Apache, ale postup bude s drobnými obměnami funkční i pro jiné webové servery.

Nejprve zkopírujte všechny soubory na webový server. Pokud máte na serveru nainstalovaný GIT, tak nejsnažším způsobem je stáhnout si všechny soubory z GitHubu.

Začneme tím, že vytvoříme pro aplikaci adresář:

|  |
| --- |
| cd /var/www/ mkdir belbin |

Teď stáhneme soubory frameworku:

|  |
| --- |
| git clone <https://github.com/lotcz/zEngine> zEngine |

A teď soubory samotné aplikace:

|  |
| --- |
| git clone <https://github.com/lotcz/belbin> belbin |

Soubory můžete samozřejmě zkopírovat i jinak, např. pomocí FTP nebo SSH. Důležité je, aby adresář **zEngine** se soubory frameworku byl umístěn ve stejném adresáři jako adresáš **belbin** se soubory aplikace.

Dalším důležitým krokem je nastavení webového serveru, aby výchozím adresářem byl adresář **belbin/www/public**, ve kterém je soubor **index.php**. Nastavení serveru Apache můžete provést například následovně: v adresáři **/etc/apache2/sites-available** vytvořte soubor belbin.conf s následujícím obsahem:

|  |
| --- |
| <VirtualHost \*:80>  ServerName belbin.zavadil.eu  DocumentRoot /var/www/belbin/belbin/www/public </VirtualHost> |

**ServerName** musí být nastaven na skutečnou adresu, na které chcete aplikaci provozovat. **DocumentRoot** musí být celá systémová cesta do adresáře **public**.

Pro aktivaci stránek je ještě třeba webovému serveru říci o nové aplikaci a restartovat jej:

|  |
| --- |
| a2ensite belbin.conf service apache2 restart |

Pro správné fungování aplikace je důležitý zapnutý mod\_rewrite a správně napsané cesty v souboru **.htaccess**. Pro náš příklad, kdy web běží na adrese [http://belbin.zavadil.eu](http://belbin.zavadil.eu/), musí soubor .htaccess vypadat následovně:

|  |
| --- |
| <IfModule mod\_rewrite.c>  RewriteEngine On    RewriteCond %{REQUEST\_URI} ^/resources/ [NC]  RewriteCond %{REQUEST\_URI} !^/resources/$ [NC]  RewriteRule ^resources/(.\*) /resources.php?path=resources/$1 [L,QSA]    RewriteCond %{REQUEST\_URI} ^/jobs/ [NC]  RewriteCond %{REQUEST\_URI} !^/jobs/$ [NC]  RewriteRule ^jobs/(.\*) /jobs.php?path=jobs/$1 [L,QSA]    # Handle Front Page...  RewriteCond %{REQUEST\_FILENAME} !-d  RewriteCond %{REQUEST\_FILENAME} !-f  RewriteRule ^(.\*) /index.php?path=$1 [L,QSA]  </IfModule> |

V případě, že chcete aplikaci provozovat např. Na adrese <http://example.com/belbinuvtest>, budete muset upravit cesty v souboru **.htaccess**, aby vypadal následovně:

|  |
| --- |
| <IfModule mod\_rewrite.c>  RewriteEngine On    RewriteCond %{REQUEST\_URI} ^/belbinuvtest/resources/ [NC]  RewriteCond %{REQUEST\_URI} !^/belbinuvtest/resources/$ [NC]  RewriteRule ^resources/(.\*) /belbinuvtest/resources.php?path=resources/$1 [L,QSA]    RewriteCond %{REQUEST\_URI} ^/belbinuvtest/jobs/ [NC]  RewriteCond %{REQUEST\_URI} !^/belbinuvtest/jobs/$ [NC]  RewriteRule ^jobs/(.\*) /belbinuvtest/jobs.php?path=jobs/$1 [L,QSA]    # Handle Front Page...  RewriteCond %{REQUEST\_FILENAME} !-d  RewriteCond %{REQUEST\_FILENAME} !-f  RewriteRule ^(.\*) /belbinuvtest/index.php?path=$1 [L,QSA]  </IfModule> |

## Instalace databáze

Na Vašem databázovém serveru je potřeba vytvořit nový katalog, vytvořit tabulky frameworku, tabulky aplikace, a naplnit je daty. SQL skripty pro toto vše jsou součástí zdrojových kódů.

Soubor **zEngine/src/sql/zEngine.sql** vytvoří katalog a tabulky pro framework, spusťte jej na mySQL serveru jako první. Poté spusťte soubor **belbin/belbin.sql**, který vytvoří tabulky pro samotnou aplikaci. Nakonec spusťte soubor **belbin/data.sql**, který naplní databázi základními daty, především otázkami a odpověďmi Belbinova testu. Poslední soubor nemusíte spouštět, pokud chcete otázky a odpovědi vytvořit sami pomocí uživatelského rozhraní.

Na závěr ještě vytvořte v databázi administrátora, abyste se mohli poprvé přihlásit. Následující SQL query vytvoří uživatele **admin** s heslem **admin123**:

|  |
| --- |
| INSERT INTO `users` (`user\_login`, `user\_email`, `user\_password\_hash`, `user\_is\_superuser`, `user\_language\_id`) VALUES ( 'admin', 'mojemejly@centrum.cz', '$2y$10$PK4Q7SADwoQLvO3gbXz/dOBvOL6tHPthTXAoDdNs/uFcBbMOGz5ym', 1, 1); |

## Konfigurace aplikace

Poslední krok instalace je vytvoření konfiguračních souborů pro aplikaci. Soubory nemusíte vytvářet ručně, v adresáři **belbin/www/app/config/examples** jsou připraveny příklady konfiguračních souborů, které stačí jen trochu upravit. Zkopírujte tedy všechny soubory z adresáře **belbin/www/app/config/examples** o úroveň výše, tedy do adresáře **belbin/www/app/config/**.

Pro fungování aplikace je potřeba provést pouze dvě změny. Za prvé otevřete soubor **core.php** a změňte základní URL aplikace (měla by být stejná jako hodnota, kterou jste nastavili jako ServerName při konfiguraci web serveru). V tomto souboru je také dobré vypnout mód ladění, ale to až později - poté, co aplikaci úspěšně nastavíte.

|  |
| --- |
| <?php  return [  'site\_title' => 'Belbinův test týmových rolí',  'base\_url' => 'http://belbin.zavadil.eu',  'debug\_mode' => false,  'error\_page' => 'error.html',  ]; |

A teď ještě otevřete soubor **mysql.php** a nastavte údaje pro přístup do databáze:

|  |
| --- |
| <?php  return [  'db\_host' => 'localhost',  'db\_login' => 'dbuser',  'db\_password' => 'your\_db\_password',  'db\_name' => 'belbin'  ]; |

Gratuluji, teď je aplikace nainstalovaná a nakonfigurovaná.