

**EKONOMICKÁ UNIVERZITA V BRATISLAVE  
FAKULTA HOSPODÁRSKEJ INFORMATIKY**

Evidenčné číslo: 17300/B/2012/2480946202

**Inovácie v platforme Zend Framework 2.0**

**Bakalárska práca**

**2012**

**Svetlana Margetová**

**EKONOMICKÁ UNIVERZITA V BRATISLAVE  
FAKULTA HOSPODÁRSKEJ INFORMATIKY**

**INOVÁCIE V PLATFORME ZEND  
FRAMEWORK 2.0**

**Bakalárska práca**

**Študijný program:** Hospodárska informatika  
**Študijný odbor:** 6292 Hospodárska informatika  
**Školiace pracovisko:** Katedra aplikovanej informatiky  
**Vedúci záverečnej práce:** Ing. Kamil Krauspe

**Bratislava 2012**

**Svetlana Margetová**



Ekonomická univerzita v Bratislave  
Fakulta hospodárskej informatiky

---

## ZADANIE ZÁVEREČNEJ PRÁCE

**Meno a priezvisko študenta:** Svetlana Margetová  
**Študijný program:** Hospodárska informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)  
**Študijný odbor:** 9.2.10 Hospodárska informatika  
**Typ záverečnej práce:** Bakalárska záverečná práca  
**Jazyk záverečnej práce:** slovenský

**Názov:** Inovácie v platforme Zend Framework 2.0

**Anotácia:** Zend Framework je knižnicou voľne previazaných komponentov pre vývoj aplikácií na platforme PHP. Knižnicu možno využiť ako bázu aplikácie, pričom sa utilizuje architektúra podľa vzoru Model-View-Controller, resp. Model-View-Presenter. Práca analyzuje inovácie a zmeny v architektúre a návrhových vzoroch plánovanej knižnice Zend Framework 2.0 oproti súčasnej verzii 1.X. Zmeny vhodne ilustruje na krátkych príkladoch.

**Vedúci:** Ing. Kamil Krauspe  
**Katedra:** KAI FHI - Katedra aplikovanej informatiky FHI  
**Vedúci katedry:** doc. Ing. Gabriela Kristová, CSc.  
**Dátum zadania:** 12.09.2011

**Dátum schválenia:** 17.10.2011

doc. Ing. Gabriela Kristová, CSc.  
vedúci katedry

## **Čestné vyhlásenie**

**Čestne vyhlasujem, že záverečnú prácu som vypracovala samostatne a že som uviedla všetku použitú literatúru.**

**Dátum: 7. 5. 2012**

.....

## **Pod'akovanie**

*Touto cestou by som sa chcela úprimne pod'akovať Ing. Kamilovi Krauspemu, za rady a pripomienky pri spracovaní bakalárskej práce.*

## **ABSTRAKT**

MARGETOVÁ, Svetlana: *Inovácie v platforme Zend Framework 2.0.* – Ekonomická univerzita v Bratislave. Fakulta hospodárskej informatiky; Katedra aplikovanej informatiky. – Vedúci záverečnej práce: Ing. Kamil Krauspe, – Bratislava: FHI EU, 2012, 64 s.

Cieľom záverečnej práce je analýza známej PHP platformy Zend Frameworku 2.0, na základe ktorej je poskytnutý komplexný obraz vyzdvihujúci prínosy a možné nedostatky novej verzie. Dôležitým aspektom je poukázanie na význam PHP frameworkov, ktorých prínos je v dnešnej dobe nekompromisný a sú neoddeliteľnou súčasťou tvorby webových stránok a aplikácií. Jedným z najpopulárnejších PHP frameworkov v súčasnosti je aj Zend Framework.

Práca je rozdelená do 4 kapitol. Obsahuje 6 obrázkov, 11 tabuliek a 25 ukážkových PHP skriptov.

Prvá kapitola je venovaná súčasnému stavu problematiky týkajúcej sa webových frameworkov, spojená s objasnením základných pojmov. Stanovenie kritérií komparácie známych PHP frameworkov, s následným predstavením samotného Zend Frameworku.

V ďalšej časti je stručná charakteristika jednotlivých komponentov, ktoré sú podkladom komparácie, stanovenie objektívnych a subjektívnych kritérií vyhodnocovania analýzy. Záverečná kapitola sa zaoberá praktickým využitím nadobudnutých poznatkov v podobe komparácie vybraných komponentov. Pozornosť je sústredená najmä na opis nových komponentov a funkcií, ktoré Zend Framework 2.0 poskytuje.

Výsledkom riešenia danej problematiky je poukázanie na prínosy, ktoré Zend Framework 2.0 prináša, ako aj stanovenie metodického návrhu prechodu na novú verziu.

## **Kľúčové slová:**

PHP framework, Zend Framework, Zend Framework 2.0, webové stránky

## **ABSTRACT**

MARGETOVÁ, Svetlana: Innovations in the Zend Framework 2.0 platform – University of Economics in Bratislava. Faculty of Economic Informatics; Department of Applied Informatics. – Advisor of the bachelor thesis: Ing. Kamil Krauspe, – Bratislava: FHI EU, 2012, p. 64.

The aim of this final thesis is the analysis of the well-known PHP Zend Framework 2.0 platform, that gives a comprehensive picture of benefits and also possible weaknesses of the new version. Nowadays, it is important to highlight the importance of PHP frameworks, whose contribution is uncompromising and is an inseparable part of the process of website and application development. One of the most popular PHP frameworks of present days is the Zend Framework.

The thesis consists of four chapters. It contains 6 pictures, 11 tables and 25 PHP sample scripts.

The first chapter is devoted to the present problems related to the website frameworks connected with basic terms explanation. Criteria determination of the well-known PHP frameworks comparison with the introduction of the Zend Framework.

In the next chapter we start with a brief characteristic of the individual components, which are the basis of the comparison, and end with the determination of objective and subjective criteria of the evaluation of the analysis. The last chapter deals with the practical use of the acquired knowledge in the form of selected components comparison. Attention is devoted mainly to the description of new components and functions which Zend Framework 2.0 provides.

The result of the discussed issue is the assessment of the set main goal, emphasizing the contributions which the Zend Framework 2.0 brings, but also the methodological design of the switch to the new version.

### **Keywords:**

PHP Framework, Zend Framework, Zend Framework 2.0, websites

# O B S A H

<b>Zoznam skrípt .....</b>	<b>9</b>
<b>Zoznam tabuliek a ilustrácií .....</b>	<b>10</b>
<b>Zoznam skratiek a značiek.....</b>	<b>11</b>
<b>Slovník termínov .....</b>	<b>12</b>
<b>Úvod .....</b>	<b>13</b>
<b>1 Súčasný stav riešenej problematiky doma a v zahraničí.....</b>	<b>15</b>
1.1 Pojem framework.....	15
1.1.1 PHP frameworky.....	15
1.2 Kritériá komparácie a analýzy PHP frameworkov .....	16
1.2.1 Najznámejšie PHP frameworky.....	17
1.3 Zend Framework.....	19
1.3.1 Nevýhody ZF .....	20
1.4 História a budúci vývoj Zend Frameworku .....	21
1.4.1 Vývoj Zend Frameworku 2.0.....	22
1.4.2 Základná charakteristika Zend Frameworku 2.0 .....	23
1.4.3 Model View Controller architektúra.....	25
<b>2 Cieľ práce .....</b>	<b>27</b>
<b>3 Metodika práce a metódy skúmania .....</b>	<b>28</b>
<b>4 Vlastná práca.....</b>	<b>30</b>
4. 1 Nové komponenty.....	31
4.1.1 Autoloader .....	32
4.1.2 Event Manager.....	34
4.2 Dependency Injection .....	36
4.2.1 Prostredníctvom konštruktora.....	37
4.2.2 Prostredníctvom setter .....	38
4.2.3 Dependency Injection Contajner .....	38
4.3 Adresárová štruktúra.....	39
4.3.1 Nový modulový systém ZF2.....	41
4.4 Nová prepracovaná MVC architektúra .....	43
4.4.1 Model.....	44
4.4.2 View.....	45
4.4.3 Controller .....	48



4.5 Vyhodnotenie naplnenia cieľov .....	54
4.5.1 Vlastné odporúčanie prechodu na novú verziu.....	57
<b>Záver .....</b>	<b>58</b>
<b>Zoznam použitej literatúry .....</b>	<b>60</b>

## Zoznam skript

Skript 1	Definícia formuláru bez využitia konceptu menných priestorov.....	30
Skript 2	Definícia formuláru s využitím konceptu menných priestorov .....	30
Skript 3	Import triedy .....	31
Skript 4	Aliasy v menných priestoroch .....	31
Skript 5	StandardAutoloader v ZF2.....	33
Skript 6	ClassMapAutoloader v ZF2.....	33
Skript 7	Autoloader Factory v ZF2.....	34
Skript 8	Spustenie EventManager v ZF2.....	35
Skript 9	Volanie udalosti v ZF2 .....	35
Skript 10	Vytvorenie spojenia EventManager v ZF2 .....	36
Skript 11	DI - pripojenie prostredníctvom konštruktora .....	37
Skript 12	DI - získanie hodnoty premennej pomocou konštruktora.....	37
Skript 13	Definícia setter metódy .....	37
Skript 14	DI - volanie set metódy.....	38
Skript 15	DIC - Service Locator .....	38
Skript 16	ModuleManager v ZF2 .....	39
Skript 17	Definovanie ModuleManager v ZF2 .....	42
Skript 18	ModuleAutoloader v ZF2 .....	43
Skript 19	Zend View v ZF1 .....	43
Skript 20	Zend View v ZF2 .....	47
Skript 21	Dispatchable Request a Response objekty v ZF2 .....	47
Skript 22	Front Controller v ZF1 .....	49
Skript 23	Front Controller v ZF2.....	51
Skript 24	Action Controller v ZF1 .....	51
Skript 25	Action Controller v ZF2 .....	53

## **Zoznam tabuliek a ilustrácií**

Tab. 1:	Porovnanie ZF a CodeIgniter.....	18
Tab. 2:	Porovnanie ZF1 a ZF2 Autoloadingu .....	32
Tab. 3:	Porovnanie adresárovej štruktúry ZF1 a ZF2 .....	40
Tab. 4:	Table Data Gateway a Row Data Gateway v ZF2.....	45
Tab. 5:	Porovnanie modelu v ZF1 a ZF2 .....	45
Tab. 6:	Zend_View v ZF2.....	46
Tab. 7:	Porovnanie čiastkových komponentov Front_Controlleru v ZF1 a ZF2 .....	50
Tab. 8:	Porovnanie Front Controlleru v ZF1 a ZF2 .....	51
Tab. 9:	Porovnanie metód Action Controller v ZF1 a ZF2.....	52
Tab. 10:	Metódy RestfulController .....	54
Tab. 11:	Porovnanie rýchlosti ZF1 a ZF2 .....	56
Obr. 1:	Najpopulárnejšie PHP frameworky .....	17
Obr. 2:	Porovnanie rýchlosti PHP frameworkov .....	21
Obr. 3:	Základné komponenty.....	24
Obr. 4:	Interakcia MVC s používateľom .....	26
Obr. 5:	Princíp fungovania Controlleru .....	48
Obr. 6:	Čas spracovania žiadostí.....	55

## Zoznam skratiek a značiek

BSD	<b>B</b> erkeley <b>S</b> oftware <b>D</b> istribution, permissívna licencia, jedna z najpoužívanějších pre open source softvér
CSS	<b>C</b> ascading <b>S</b> tyle <b>S</b> heets, kaskádové štýly
DI	<b>D</b> ependency <b>I</b> njection, návrhový vzor
FLOSS	<b>F</b> ree/ <b>L</b> ibre/ <b>O</b> pen <b>S</b> ource <b>S</b> oftware, spoločný názov pre slobodný softvér a open source softvér
GIT	<b>G</b> NU <b>I</b> nteractive <b>T</b> ools, verzionovací systém pre správu súborov
HTML	<b>H</b> yper <b>T</b> ext <b>M</b> arkup <b>L</b> anguage, jazyk určený na vytváranie webových stránok a iných
HTTP	<b>H</b> ypertext <b>T</b> ransfer <b>P</b> rotocol, protokol pre prenos HTML dokumentov medzi servermi a klientmi služby WWW
IBM	<b>I</b> nternational <b>B</b> usiness <b>M</b> achines Corporation, svetová spoločnosť podnikajúca v odbore informačných technológií
IDE	<b>I</b> ntegrated <b>D</b> evelopment <b>E</b> nvironment, integrované vývojové prostredie
MVC	<b>M</b> odel <b>V</b> iew <b>C</b> ontroller, softvérová architektúra, ktorá rozdeľuje dátový model, užívateľské rozhranie a aplikačnú logiku do troch na sebe nezávislých komponentov
ORM	<b>O</b> bjektovo <b>R</b> elačné <b>M</b> apovanie, technológia prepojenia sveta relačných databáz a sveta objektov
PEAR	<b>P</b> HP <b>E</b> xtension and <b>A</b> pplication <b>R</b> epository, systém balíčkov rozširujúcich funkcie štandardného jazyka PHP
PHP	<b>P</b> ersonal <b>H</b> ome <b>P</b> age, open source skriptovací programovací jazyk
RSS	<b>R</b> esource <b>D</b> escription <b>F</b> ramework <b>S</b> ite <b>S</b> ummary
XHTML	<b>E</b> xtensible <b>H</b> ypertext <b>M</b> arkup <b>L</b> anguage, rozšíriteľný hypertextový značkový jazyk
ZF	<b>Z</b> end <b>F</b> ramework, open – source PHP framework používaný pre tvorbu webových aplikácií v jazyku PHP 5

## Slovník termínov

**Autoloader** – komponent, ktorý umožňuje automatické nahrávanie modelov a zdrojov.

**Controller** - reaguje na udalosti a zisťuje zmeny v modeli a pohľade.

**Dependency Injection** – nový návrhový vzor, ktorého cieľom je znížiť väzbu medzi jednotlivými zložkami kódu.

**Design by contract** - predstavuje zmenu časti frameworku. Vytvára alternatívnu implementáciu štandardných tried a zároveň zabezpečuje, že táto implementácia naďalej pracuje s triedami.

**Event Manager** - komponent umožňujúci elimináciu Singletonov.

**Framework** - je prostredie, v ktorom je organizovaná a napísaná ďalšia aplikácia. Je napísaný v tom istom programovacom jazyku ako aplikácia. Je to súbor knižníc a kódu usporiadaných tak, aby pokrývali čo najviac funkčných požiadaviek spoločných pre rôzne aplikácie.

**Open-source softvér** - počítačový softvér, ktorého zdrojový kód je prístupný pod takou licenciou, ktorá umožňuje študovanie, poprípade vkladanie zmien a vylepšení do zdrojového kódu do softvéru a umožňuje ďalšiu redistribúciu v modifikovanej alebo nezmenenej forme.

**Projekt Magento** – komplexný systém pre eshopy, pokrývajúci všetky činnosti potrebné pre efektívny predaj, správu zákazníkov, vyhodnocovanie úspešnosti predaja a ďalšie skupiny činností.

**Singleton** - návrhový vzor singleton zabezpečuje existenciu jednej inštancie danej triedy v pamäti a zároveň k nej vytvára globálny prístup pomocou statickej metódy.

**View** – (pohľad) prevádza dáta z modelu do podoby vhodnej k prezentácii užívateľovi.

**Zend Conf** - medzinárodná konferencia, na ktorej sú vývojármi prezentované teoretické a praktické poznatky o Zend Frameworku.

**Zend Studio** - je editor pre tvorbu PHP skriptov s integrovaným vývojovým prostredím pre profesionálnych vývojárov.

# Úvod

*Základnou podstatou ambicióznosti je len tieň nejakého sna*

**William Shakespeare**

Ľudia neustále vymýšľajú nové veci, sny a ambície ich vedú k novým technologickým pokrokom a objavom. Podobne to bolo aj u Andi Gutmansa, zakladateľa a vedúceho vývojára jazyka PHP, ktorého hlavnou ambíciou je poskytnúť prostriedok, zjednodušujúci vývoj aplikácií a tým prispieť k vytvoreniu plnohodnotnej informatickej spoločnosti.

Za posledných desať rokov sa PHP stalo uznávaným nástrojom, zabezpečujúcim vývoj webových stránok. V súčasnosti patrí medzi najpoužívanejšie jazyky na svete. V rámci štandardizácie riadenia a neustáleho rozvoja, postupne narastali aj požiadavky používateľov, kladené na webové aplikácie, ktoré sa stávajú čoraz viac dynamickými. Z tohto dôvodu sa s príchodom PHP 5, začali postupne rozširovať frameworky, ktoré poskytujú základné nástroje urýchľujúce vývoj, funkcie zabezpečujúce konzistenciu a overovanie dát, jednoduchý prístup k údajom v databázach a predstavujú bezpečnejší spôsob vytvárania aplikácií. Jedným z nich bol aj Zend Framework, ktorý sa stáva čoraz viac populárnejším a obľúbenejším, najmä vďaka flexibilita poskytujúcej možnosť oddelenia jednotlivých komponentov, čím sa filozoficky odlišuje od svojich konkurentov. Je zameraný na vybudovanie spoľahlivejších, bezpečnejších a moderných Web 2.0 aplikácií. Rýchla narastajúca obľúbenosť podnietila vývojárov k vývoju novej hlavnej verzie, vďaka ktorej bude Zend Framework naozaj robustnejším frameworkom, s plným využitím nových aspektov PHP 5.3, založených výlučne na objektovo orientovanom prístupe.

Ústredným cieľom bakalárskej práce je porovnanie novej verzie Zend Framework 2.0 so starším predchodcom, Zend Frameworkom 1.X, s poukázaním na nové možnosti využitia. Cieľom je poskytnúť komplexný obraz zobrazujúci hlavné rozdiely, prínosy a prípadné nedostatky v novej verzii.

Prvá kapitola je zameraná na objasnenie pojmu framework, pre lepšie pochopenie problematiky. Prehľad niektorých najpopulárnejších PHP frameworkov za posledný rok a porovnanie dvoch najznámejších pomocou kritérií, ktoré sú na dané frameworky kladené. Vysvetlenie pojmu Zend Framework cez stručnú históriu, popis výhod, vďaka ktorým predstihuje konkurenciu, ale aj nevýhod, ktoré sa snaží nová verzia odstrániť. Súčasťou teoretickej analýzy je aj popis základných komponentov, ktoré tvoria kostru každej aplikácie.

Hlavné zdroje údajov a metódy ich získavania sú opísané v tretej kapitole. Sú rozdelené na spracovanie teoretickej a praktickej časti. Zdrojom poznatkov, na základe ktorých bola vykonaná výsledná analýza, boli najmä informácie týkajúce sa základných komponentov. Metódy komparácie boli klasifikované na objektívne a subjektívne. Výber objektívnych kritérií sa uskutočnil na základe komponentov, ktoré predstavujú najväčšie zmeny a vylepšenia. Subjektívne kritériá sme stanovili podľa požiadaviek, ktoré chce firma Zend priniesť a dosiahnuť vývojom novej verzie.

Záverečná kapitola je kľúčovou, ktorá sa zaoberá už konkrétnou komparáciou na jednotlivých príkladoch, s poukázaním na hlavné rozdiely a možnosti použitia. Zároveň sú tu uvedené nové komponenty a funkcie. Objektívna komparácia analyzuje nové kompletne prepracované komponenty. Ich zavedením sú viditeľne jasné prínosy Zend Frameworku 2.0, ktoré sú tu vyhodnotené. Na strane druhej subjektívna analýza, kde sú vyhodnotené výsledky na základe sumarizácie, výsledné zhodnotenie naplnenia požiadaviek a očakávaní firmy Zend, kladených na Zend Framework 2.0. Rozhodujúcim faktorom výsledkov vyhodnotenia, boli najmä tri hľadiská – *flexibilita, výkon a dokumentácia*. Záver kapitoly je spojený s možným návrhom prechodu na novú verziu.

# 1 Súčasný stav riešenej problematiky doma a v zahraničí

Analýza súčasnej situácie týkajúca sa webových frameworkov, predstavuje neustály vývoj a zlepšovanie, čo je priamo spojené s rozvojom programovacích jazykov. Každý dostupný framework poskytuje iné funkcie, preto existujú určité požiadavky, na základe ktorých frameworky rozlišujeme. Je dôležité pochopiť, čo vlastne framework je, aby sme naplno využívali možnosti, ktoré poskytuje.

## 1.1 Pojem framework

Existuje mnoho definícií tohto pojmu, my sa zameriame na webové frameworky, ktoré priamo súvisia s tvorbou internetových stránok a aplikácií. Podľa Porebského a kol. (poznámka pod čiarou) webový aplikačný framework predstavuje súbor architektonicky usporiadaných zdrojových kódov, umožňujúci rýchly a bezpečný vývoj dynamických webových stránok a aplikácií.<sup>1</sup>

Webové frameworky sú napísané v rovnakom programovacom jazyku ako vyvíjaná aplikácia a predstavujú prostredie, v ktorom je organizovaná. Nejedná sa o samostatný program. Základ tvoria knižnice, ktoré pokrývajú veľkú časť funkčných požiadaviek spoločných pre väčšinu projektov, uplatnenie majú najmä pri tvorbe komplexnejších úloh.<sup>2</sup>

### 1.1.1 PHP frameworky

Vďaka popularizácii webových frameworkov sa rapídne rýchlo začali rozvíjať a rozširovať aj pre jazyk PHP. V súčasnosti ich je neprehľadné množstvo. Existujú malé a veľké, rýchle a pomalé, platené aj voľne šíriteľné.<sup>3</sup> Niektoré sú vysoko štruktúrované, ponúkajú aj nadštandardné postupy, kým iné poskytujú kompletnú sadu funkcií a vyžadujú si dodržiavanie striktných pravidiel kódovania. Každý z nich má svoje klady a zápory. Preto je veľmi dôležitý výber správneho frameworku, ktorý skutočne uľahčí prácu a zároveň umožňuje vytvoriť robustnejšiu aplikáciu.

---

<sup>1</sup>POREBSKI, B. – PRZYSKALSKI, K. – NOWAK, L. 2011. *Building PHP Applications with Symfony, CakePHP, and Zend Framework*. Indianapolis : Wrox, 2011. s. 40. ISBN 978-0-470-88734-9.

<sup>2</sup>GILMORE, J. W. 2009. *Easy PHP Websites with the Zend Framework*. Columbus : W.J. Gilmore, LLC, 2009. s. 129. ISBN 978-0470887349.

<sup>3</sup>BÖHMER, M. 2010. *Zend Framework : Programujeme webové aplikace v PHP*. Brno : CPress, 2010. s. 23. ISBN 978-80-251-2965-4.



## 1.2 Kritériá komparácie a analýzy PHP frameworkov

Na základe kľúčových kritérií a požiadaviek, ktoré sú kladené na PHP frameworky je možné vykonať ich komparáciu a vybrať ten najvhodnejší, vyhovujúci špecifickým potrebám a tým stanoviť akýsi štandard v programovaní našich aplikácií. Nižšie sú uvedené najdôležitejšie z nich, ktoré je nutné rozhodne zvážiť pri výbere:<sup>4</sup>

### ***Podpora***

Je vhodné, vybrať si framework s rozšírenou komunitou alebo vyvíjaný známou spoločnosťou, kde môžeme očakávať dlhodobú podporu a vývoj aj v budúcnosti.

### ***Flexibilita***

Predstavuje druhý najdôležitejší faktor. Na základe tohto kritéria existujú frameworky, ktoré si vyžadujú dodržiavanie prísnych konvencií kódovania a organizácie, kým iné poskytujú voľnosť a možnosť rozšíriteľnosti, bez priamej závislosti medzi jednotlivými komponentmi.<sup>5</sup>

### ***Výkon***

Zohráva významnú úlohu pri komparácii. Existujú frameworky, ktoré obsahujú mnoho knižníc a pokrývajú takmer všetky potreby, ktoré si vyžaduje tvorba bežnej aplikácie. Na druhej strane patria k tým pomalším, s väčším výpočtovým a hardvérovým zaťažením. Tie, ktoré obsahujú menšiu škálu knižníc, sú modulárnejšie a je jednoduché ich rozširovať o potrebné moduly. Tým pádom sú zvyčajne aj rýchlejšie.<sup>6</sup>

### ***Dokumentácia***

Je neoddeliteľnou súčasťou. Za dobre spracovanú dokumentáciu považujeme jasnú, aktuálnu a dokončenú, na základe ktorej bude krivka učenia čo najkratšia.<sup>7</sup>

---

<sup>4</sup>MCARTHUR, K. 2008. *Pro PHP Patterns, Frameworks, Testing and More*. 2008. United States of America : Apress, 2008. s. 230. ISBN 978-1-59059-819-1.

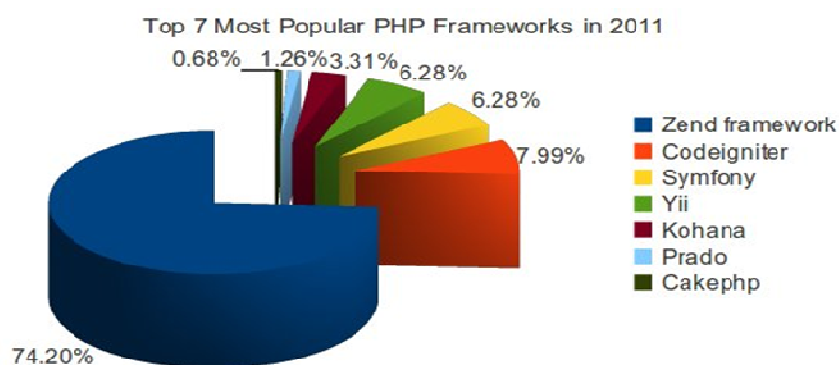
<sup>5</sup>MCARTHUR, K. 2008. *Pro PHP Patterns, Frameworks, Testing and More*. 2008. United States of America : Apress, 2008. s. 231. ISBN 978-1-59059-819-1.

<sup>6</sup>MCARTHUR, K. 2008. *Pro PHP Patterns, Frameworks, Testing and More*. 2008. United States of America : Apress, 2008. s. 230. ISBN 978-1-59059-819-1.

<sup>7</sup>MCARTHUR, K. 2008. *Pro PHP Patterns, Frameworks, Testing and More*. 2008. United States of America : Apress, 2008. s. 230. ISBN 978-1-59059-819-1.

### 1.2.1 Najznámejšie PHP frameworky

Zoznam siedmich najpopulárnejších PHP frameworkov za rok 2011 (obr. 1).



Obr. č. 1: Najpopulárnejšie PHP frameworky <sup>8</sup>

Dáta boli porovnávané pomocou nástroja Google Analytics. Na základe výsledkov je najpopulárnejším frameworkom Zend Framework (skrát. ZF) so 74,20%. Za ním nasleduje CodeIgniter. Na základe kritérií uvedených v kapitole 1.2, sme porovnali ZF na strane jednej a CodeIgniter, na strane druhej (tab. 1).

<sup>8</sup>2011. *Most Used PHP Framework-The Popular Top 7 List in year 2011*. [online]. 2011. [cit. 2012.04.20]. Dostupné na internete: <<http://www.php-developer.org/most-used-php-framework-the-popular-top-7-list-in-year-2011/>>.

Tab. č. 1: Porovnanie ZF a CodeIgniter

Kritériá	ZF	CodeIgniter
<b>Podpora</b>	Stojí za ním firma Zend, ktorá je aj za vývojom samotného PHP. Je distribuovaný pod BSD licenciou, ktorá ochraňuje autorské práva vlastníkov a je možné využívať ZF aj na komerčné účely. Taktiež má veľkú podporu komunity. <sup>9</sup>	Je vyvíjaný americkou spoločnosťou EllisLab, s rozsiahlou komunitou aj na Slovensku.
<b>Flexibilita</b>	Hlavným princípom je „použi podľa potreby“ (z angl. <i>use at will</i> ). Umožňuje používať knižnice spolu alebo osobitne. Taktiež je možné integrovať ho s inými knižnicami ako PEAR, Doctrine ORM alebo Smarty Template Library. Umožňuje vysokú úroveň riešení konštrukčných problémov a ich implementáciu, na základe vlastného výberu požadovaného komponentu. <sup>10</sup>	Poskytuje dobrý základ, ktorý je možné ľahko nasadiť v našich aplikáciách a vďaka dynamickému načítavaniu je veľmi rýchly. Nemusí však byť tou najlepšou voľbou pre vývoj rozsiahlych a zložitých aplikácií. Je vhodnejší pre malé a stredne veľké projekty. Je pomerne zložitý vytvoriť čiastkové aplikácie v kóde. <sup>11</sup>
<b>Dokumentácia</b>	Popisuje viac ako 60 komponentov, ktoré tvoria súčasť základnej distribúcie, vygenerovaná pomocou PHPDoc. Od prvej verzie je súčasťou vybavenia aj referenčná príručka, kde sú opísané všetky zložky a komponenty aplikácie. Okrem anglickej verzie, je možné ju nájsť aj v jazyku nemeckom, ruskom a francúzskom. V súčasnosti obsahuje popis niektorých komponentov aj v slovenskom jazyku. <sup>12</sup>	Prehľadná dokumentácia, ktorá je rozdelená na všeobecné témy a referencie tried. Obsahuje mnoho užitočných rád. Na hlavnej stránke sa nachádzajú dva video tutoriály, ktoré sú dobre spracované. <sup>13</sup>

<sup>9</sup> ALLEN, R., LO, N. 2009. *Zend Framework in Action*. United States of America : Manning Publications, 2009. s. 8. ISBN 978-1933988320.

<sup>10</sup> POPE, K. 2009. *Zend Framework 1.8 Web Application Development*. Birmingham : Packt Publishing, 2009. s. 3. ISBN 978-1-847194-22-0.

<sup>11</sup> POREBSKI, B. – PRZYKALSKI, K. – NOWAK, L. 2011. *Building PHP Applications with Symfony, CakePHP, and Zend Framework*. Indianapolis : Wrox, 2011. s. 1061. ISBN 978-0-470-88734-9.

<sup>12</sup> VASWANI, V. 2010, *Zend Framework, A Beginner's Guide*. United States of America : McGraw-Hill Osborne Media, 2010. s. 7. ISBN 978-0-07-163940-8.

Výber je individuálny podľa špecifických požiadaviek, ktoré si vyžaduje naša aplikácia. Žiadny framework nevyhovuje potrebám všetkých. Hlavnou výhodou ZF sú jeho široké možnosti použitia v mnohých odvetviach, je populárnym na celom svete o čom svedčia aj nasledujúce čísla. Za posledné roky zaznamenal desať miliónov stiahnutí z oficiálneho serveru, takmer šesť miliónov hľadání v Google a bol základom pre vytvorenie mnohých projektov. Stoja za ním známe firmy ako je IBM s podporou aj na Slovensku, Microsoft a Amazon.<sup>14</sup> Významným projektom je najmä projekt Magento, ktorý predstavuje najpopulárnejší e-shop na svete. Navyše, firma Zend poskytuje aj ďalšie vývojové nástroje, ktoré s ním úzko súvisia. Je možné využívať Zend Studio, komerčné IDE, ktoré optimalizuje vývojový proces a zvyšuje produktivitu. Druhým nástrojom je Zend Server, ktorý poskytuje príležitosť sledovania a diagnostikovania vývoja aplikácií.<sup>15</sup> Má množstvo prispievateľov, ktorý sa podieľajú na vývoji a zdokonaľovaní, denne odstraňujú nové chyby a prispievajú k rozvoju. Na Slovensku nemá veľmi početnú komunitu vývojárov, ale silné zastúpenie je v susednej Českej republike, kde sa pravidelne konajú známe tzv. „ZF Meetup“ stretnutia, na ktorých sa preberá súčasná problematika a vývoj. Momentálne ja hlavnou témou diskusií nová vyvíjaná verzia ZF 2.0, ktorá je jadrom celej práce.

### 1.3 Zend Framework

V súčasnosti patrí medzi najznámejšie platformy vo svete PHP. ZF je open – source knižnica umožňujúca tvorbu webových aplikácií v PHP. Je spustiteľný na ktoromkoľvek serveri s podporou minimálne PHP 5.1.4 a vyššou.<sup>16</sup> Zvyšuje produktivitu vývojárov. Na rozdiel od iných, je možné definovať ho aj za behu aplikácie. Poskytuje vysoko výkonnú MVC implementáciu, databázovú abstrakciu, jednoduché renderovanie HTML formulárov,

---

<sup>13</sup>DANEK, P. 2008. *Velký test PHP frameworků: Zend, Nette, PHP a RoR*. [online]. Praha : Internet Info, s.r.o., 2008. [cit. 2012.01.20]. Dostupné na internete: < <http://www.root.cz/clanky/velky-test-php-frameworku-2-dil/>>. ISSN 1212-8309.

<sup>14</sup>ZEND TECHNOLOGIES, Ltd. 2012. *Zend Framework by the Numbers*. [online]. Zend Technologies Ltd. 2012. [cit. 2012.01.20]. Dostupné na internete: < <http://zendframework.com/about/numbers>>.

<sup>15</sup>ALLEN, R., LO, N. 2009. *Zend Framework in Action*. United States of America : Manning Publications, 2009. s. 8. ISBN 978-1933988320.

<sup>16</sup>EVANS, C. 2008. *php|architect's Guide to Programming with Zend Framework*. United States of America : Marco Tabini & Associates, Inc., 2008. s. 19. ISBN 978-0-9738621-5-7.

overovanie, filtrovanie žiadostí a mnoho ďalších funkcií. Všetky tieto činnosti je možné zlúčiť pomocou jedného ľahko použiteľného objektovo orientovaného rozhrania.<sup>17</sup>

Má mnoho výhod, vďaka ktorým predstihuje konkurenciu. Podľa vedúceho projektu Matthew Weier O'Phinneya: „*Kladieme dôraz na Unit testing a dokumentáciu kódu*“.<sup>18</sup>

### *1.3.1 Nevýhody ZF*

Ako každý framework aj ZF má svoje nevýhody. Medzi hlavné patria slabá podpora pre prácu s databázovým modelom, čo znižuje jeho flexibilitu, nejasná dokumentácia, problémy s výkonom a veľké pamäťové nároky.

#### *Slabá podpora pre prácu s databázovým modelom*

Podľa významných predstaviteľov zastupujúcich ZF (poznámka pod čiarou) sú základnými problémami najmä neefektívny spôsob pripájania, obmedzenie rozšíriteľnosti a zložité získavanie schémy metadát konzistentným spôsobom. Výrazne znižujú flexibilitu a výkon ZF. Možnosťou eliminácie tohto nedostatku je využitie jednej z dostupných ORM knižníc, ako napríklad Doctrine.<sup>19</sup>

#### *Nejasná dokumentácia*

Aj keď je kompletná, pre začínajúcich vývojárov môže byť nejasná a zle čitateľná, z dôvodu veľkého množstva komponentov, čím sa krivka učenia zvyšuje. Neobsahuje príklady na tvorbu komplexnej aplikácie.

#### *Problémy s výkonom a veľké pamäťové nároky*

Na základe výsledkov testov, kde bol testovaný jednoduchý spôsob vypísania „Hello World“ (obr. 2), sme zobrazili celkové porovnanie výkonu jednotlivých najznámejších frameworkov.

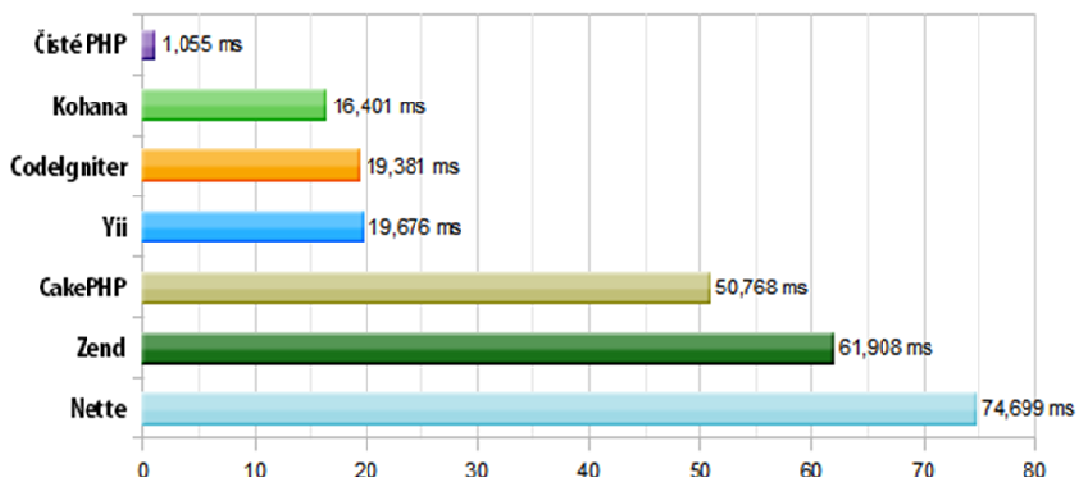
---

<sup>17</sup>ALLEN, R., LO, N. 2009. *Zend Framework in Action*. United States of America : Manning Publications, 2009. s. 9. ISBN 978-1933988320.

<sup>18</sup>DROZD, M. 2009. *PHP frameworky na konferencii Zend/PHP*. [online]. Bratislava : Bloog.sk, 2009. [cit. 2012.01.13]. Dostupné na internete: <<http://www.bloog.sk/?s=zend+framework>>.

<sup>19</sup>O'PHINNEY, W. M., SCHINDLER, R. 2010. *Introducing Zend Framework 2.0*. [online]. SlideShare Inc., 2010. [cit. 2012.01.20]. Dostupné na internete: <<http://www.slideshare.net/weierophinney/introducing-zend-framework-20>>.

Ako už na prvý pohľad vidno, ZF patrí k tým najpomalším. Priemerný čas zobrazenia dát je okolo jednej sekundy, čo je v porovnaní s konkurenciou pomerne veľké číslo.



Obr. č. 2: Porovnanie rýchlosti PHP frameworkov<sup>20</sup>

Hlavným cieľom novej vyvíjanej verzie Zend Framework 2.0 je práve odstrániť vyššie menované nedostatky.

## 1.4 História a budúci vývoj Zend Frameworku

ZF vynašla americko-izraelská spoločnosť Zend Technologies Ltd. V októbri 2005 na konferencii ZendConf vznikol prvý návrh tohto projektu, ako súčasť PHP Collaboration Project. Zakladateľmi sú Andi Gutmans a Zeev Suraski.<sup>21</sup> Okrem hlavného sponzora k vývoju prispeli aj ďalšie známe firmy, ako napríklad spoločnosti Google, Microsoft a Strikelron, ktoré ho obohatili o mnohé nové komponenty a významné funkcie.<sup>22</sup>

<sup>20</sup>KOPRDA, M. 2010. Porovnanie frameworkov: CakePHP vs. CodeIgniter vs. Kohana vs. Nette vs. Yii vs. Zend in *Zajtra.sk*. [online]. Bratislava : 2010. [cit. 2012.04.20]. Dostupné na internete: <<http://www.zajtra.sk/programovanie/87/porovnanie-frameworkov-cakephp-vs-codeigniter-vs-kohana-vs-nette-vs-yii-vs-zend>>.

<sup>21</sup>GILMORE, J. W. 2009. *Easy PHP Websites with the Zend Framework*. Columbus : W.J. Gilmore, LLC, 2009. s. 132. ISBN 0615303889.

<sup>22</sup>VASWANI, V. 2010. *Zend Framework, A Beginner's Guide*. United States of America : McGraw-Hill Osborne Media, 2010. s. 3. ISBN 978-0-07-163940-8.

Prvá verejná verzia s označením Pre Alpha Version 0.1.1. bola vydaná v marci 2006. Ešte nebola vhodná na nasadenie v praxi. Následne na to v júli vyšla verzia 1.0.0, ktorá už zahŕňala 35 základných súčastí a veľa užitočných funkcií ako je MVC architektúra, vrátane komponentov pre ukladanie do vyrovnávajúcej pamäte, riadenie prístupu do databázy, Lucenesearch engine, autentizácia, autorizácia, rozhranie webových služieb, RSS a lokalizácia.<sup>23</sup>

Samozrejme, žiadny softvér sa nezaobíde bez neustáleho vývoja a zdokonaľovania. Výnimkou nie je ani ZF, ktorý je priamo spojený s rozvojom jazyka PHP. V nadväznosti na prvú verziu bol vývojový proces pomerne rýchly, najmä vďaka početnej komunite. Súčasná verzia je ZF 1.11, ktorá obsahuje viac ako 65 komponentov.

Verzie ZF sú označované v tvare:<sup>24</sup>

- **X – číslo hlavnej verzie**
- **Y – číslo minor verzie**
- **Z – číslo mini verzie, ktorá opravuje chyby minor verzie**

#### 1.4.1 Vývoj Zend Frameworku 2.0

Po štyroch rokoch je ZF 2.0 prvou hlavnou verziou po verzii 1.0. Prvá vývojová vetva Beta 1 bola sprístupnená 6. augusta 2010, druhá bola vydaná v novembri 2010. V súčasnosti je poslednou aktívnou vetvou verzia Beta 3. Funkčne kompletná verzia bude prístupná v lete 2012. Vývoj verzie 2.0 prináša mnoho nových funkcií, rovnako ako zlepšenie už uceleného súboru základných funkcií. Často sú tieto zmeny spojené so zmenou celej architektúry.

Vývojový proces ZF 2.0 je spojený s prechodom zo SVN na GIT repozitár, ktorý predstavuje open - source distribuovaný verzionovací systém pre správu malých a stredne veľkých projektov. Ponúka lepšie pracovné postupy ako SVN, možnosť lepšieho vetvenia a zlučovania jednotlivých častí kódu rýchlym a efektívnym spôsobom. Obsahuje kompletnú

---

<sup>23</sup>POPE, K. 2009. *Zend Framework 1.8 Web Application Development*. Birmingham : Packt Publishing, 2009. s. 2. ISBN 978-1-847194-22-0.

<sup>24</sup>FELTON, D. 2010. *Zend Framework Version Lifecycle*. [online]. 2010. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV/Zend+Framework+Version+Lifecycle>>.

históriu, možnosť sledovania každej zmeny revízií, s nezávislým prístupom k sieti alebo centrálnemu serveru.<sup>25</sup>

Jednou z možností prechodu na novú verziu je refaktorovanie. Prestavuje často používaný nástroj slúžiaci k prepisu starých aplikácií. Umožňuje plynulý prechod, behom ktorého je možné používať staré aj nové časti aplikácie súčasne.<sup>26</sup>

### 1.4.2 Základná charakteristika Zend Frameworku 2.0

Súčasná verzia Zend Framework 2.0 Beta 3 je posledným míľnikom popredných svetových frameworkových platforiem. Plne podporuje PHP 5.3, s čím je spojených mnoho nových výhod a funkcií, vďaka ktorým je realizácia moderných návrhových vzorov jednoduchšia. Najväčším prínosom je možnosť využívania menných priestorov, ktoré umožňujú zapuzdrenie triedy a ostatných položiek PHP, čím zvyšujú znovupoužiteľnosť kódu. Ďalšími významnými funkciami PHP 5.3, ktoré plne ZF2 využíva sú:<sup>27</sup>

- **Closures, Lambdas, Late Static Binding** – predstavujú synonymá pre konštruktory a anonymné funkcie. Sú definované za chodu aplikácie, môžu byť priradené premennej alebo volané ako argument ďalších metód. Všetky *create\_function()* sú nahradené funkciou *closures*.
- **Invokables** – predstavujú novú metódu *\_invoke()*, ktorá volá objekty ako funkcie.

V súčasnosti zahŕňa aj podporu pre PHP 5.4.0, čo prináša ďalšie zásadné vylepšenia, ktorých cieľom je zvýšiť kvalitu kódu a poskytnúť efektívne a jednoduché pracovné postupy. Súčasťou PHP 5.4 je zvýšená podpora JavaScriptového objektového zápisu (skrát. JSON), podpora ázijských jazykov, bez nutnosti prekompilácie aplikácie a vylepšená možnosť uploadu súborov. Navyše obsahuje novú modifikovanú BSD licenciu s názvom FLOSS v spolupráci s CopyrightLicense.<sup>28</sup>

<sup>25</sup>O'PHINNEY, W. M. 2011. *RFC - Git or SVN*. [online]. 2011. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/pages/viewpage.action?pageId=20873259>>.

<sup>26</sup>O'PHINNEY, W. M. 2011. *Zend Framework 2.0 Roadmap*. [online]. 2011. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV2/Zend+Framework+2.0+Roadmap>>.

<sup>27</sup>O'PHINNEY, W. M. 2011. *Zend Framework 2.0 Roadmap*. [online]. 2011. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV2/Zend+Framework+2.0+Roadmap>>.

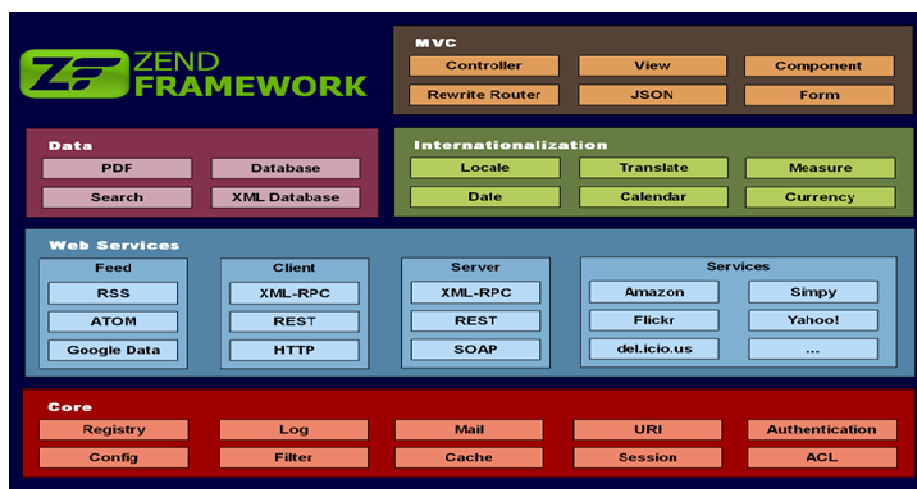
<sup>28</sup>KERNER, M. S. 2011. *PHP 5.4 and Zend Framework 2.0 Gearing up for Release*. [online]. QuinStreet Inc., 2011. [cit. 2012.02.02]. Dostupné na internete: <<http://www.developer.com/lang/php/php-5-4-and-zend-framework-2-0-gearing-up-for-release.html%20>>.



Matthew Weier O'Phinney, vedúci Zend Frameworku sa o ZF2 vyjadril: „Zend Framework 2.0 ponúka PHP vývojárom kompletný architektonický redizajn pre lepšiu, rozšíriteľnejšiu a flexibilnejšiu MVC framework. PHP vývojári už dlho poznajú široké možnosti Zend Frameworku a rozsiahlu knižnicu opakovane použiteľných kódov. Zend Framework 2.0 má jednoduchšie použitie, podporuje najlepšie PHP postupy a prenositeľnosť kódu bude cenným doplnkom akéhokoľvek projektu.”<sup>29</sup>

### Základné komponenty

ZF je možné rozdeliť do piatich základných modulov (obr. 3), ktoré majú uplatnenie vo všetkých projektoch. Poskytujú základné funkcie ako štandardizované zavádzanie aplikácie, načítanie tried, vykonávajú potrebnú konfiguráciu, filtrovanie a overovanie údajov, posielanie e-mailov a mnoho ďalších možností, ktoré pokrývajú všetky oblasti vývoja aplikácií. Každá časť sa skladá z niekoľkých zložiek a každý komponent obsahuje rad tried, čím vytvárajú komplexný systém.<sup>30</sup>



Obr. č. 3: Základné komponenty ZF<sup>31</sup>

<sup>29</sup>ZEND TECHNOLOGIES, Inc. 2012. Zend Framework 2.0 Beta 3 Release Gives Developers an Early Start on PHP App Development In MC PRESS. [online]. 2012, no. 3. [cit. 2012.02.02]. Dostupné na internete: < <http://www.mcpressonline.com/programming-languages/zend-framework-20-beta-3-release-gives-developers-an-early-start-on-php-app-development.html>>.

<sup>30</sup>BÖHMER, M. 2010. *Zend Framework : Programujeme webové aplikace v PHP*. Brno : CPress, 2010. s. 61. ISBN 978-80-251-2965-4.

<sup>31</sup>ŠALTYS, Ž. 2007. *Zend Framework pros and cons*. [online]. 2007. [cit. 2012. 01. 13]. Dostupné na internete: < <http://www.thedeveloperday.com/zend-framework-pros-and-cons/>>.

### 1.4.3 Model View Controller architektúra

Model View Controller (skrát. MVC), čo môžeme preložiť ako Model Pohľad Radič, je základným architektonickým vzorom, na ktorom je ZF založený. Podporuje pokročilé techniky a osvedčené postupy tvorby aplikácií, avšak každá implementácia sa líši v závislosti od konkrétnej aplikácie, čo môže byť v niektorých prípadoch problematické.<sup>32</sup>

Princíp spočíva v rozdelení zodpovednosti MVC komponentov do jasne, logicky definovaných skupín, ktoré rozdeľujú aplikáciu na tri samostatné časti.

#### ➤ **Model**

Reprezentuje dátovú vrstvu. Je dôležité pochopiť, že model nie je ekvivalentom databázy. Vo väčšine architektúr ukladá údaje v relačnej databáze, ako napríklad MySQL alebo Oracle. Môže však využívať aj textový súbor, RSS alebo indexy. Jeho hlavnou funkciou je čítanie a zapisovanie údajov.<sup>33</sup>

#### ➤ **View**

Reprezentuje prezentačnú vrstvu. Zobrazuje údaje, ktoré mu poskytne controller, no pozná iba ich typ a formát. Je úplne odizolovaný od hlavnej aplikačnej logiky. V konečnom dôsledku sa jedná o pole alebo objekt. Na zobrazenie využíva rôzne technológie, v mnohých prípadoch je realizovaný najčastejšie vo forme HTML stránky.<sup>34</sup>

#### ➤ **Controller**

Reprezentuje riadiacu vrstvu. Zabezpečuje prepojenie medzi modelom a view. Je zodpovedný za načítanie a overenie parametrov požiadavky, ukladanie a načítanie záznamov a výber vhodných zobrazení, rozhoduje o ďalšej realizácii programu a spravuje výnimky.<sup>35</sup>

---

<sup>32</sup>PADDILA, A. 2009. *Beginning Zend Framework*. United States of America : Apress, 2009. s. 73. ISBN 978-1-4302-1825-8.

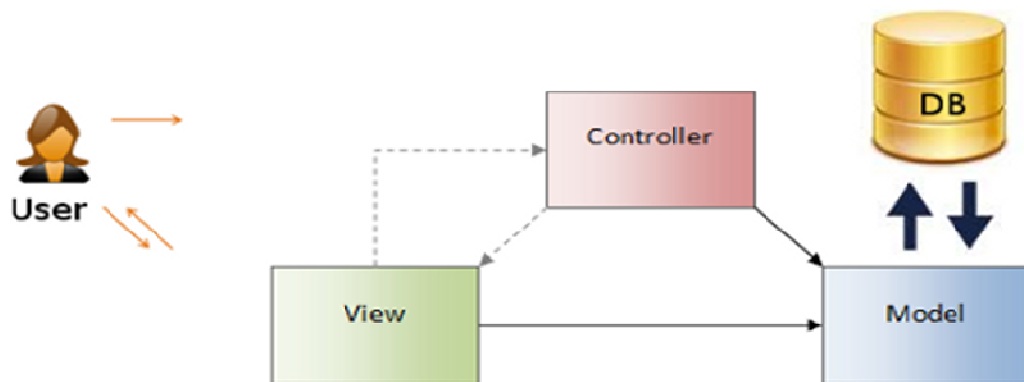
<sup>33</sup>BÖHMER, M. 2010. *Zend Framework : Programujeme webové aplikace v PHP*. Brno : CPress, 2010. s. 399. ISBN 978-80-251-2965-4.

<sup>34</sup>MCARTHUR, K. 2008. *Pro PHP Patterns, Frameworks, Testing and More*. United States of America : Apress, 2008. s. 227. ISBN 978-1-59059-819-1.

<sup>35</sup>BÖHMER, M. 2010. *Zend Framework : Programujeme webové aplikace v PHP*. Brno : CPress, 2010. s. 401. ISBN 978-80-251-2965-4.

### **Interakcia MVC s užívateľom**

Princíp fungovania MVC architektúry (obr. 4), ktorý je nasledovný. V prvom kroku sa uskutoční odoslanie HTTP požiadavky serveru. Controller ju prijme a zistí, ktorý model má byť v prípade potreby zavolaný. V niektorých prípadoch controller zobrazuje údaje zadané užívateľom a vyžaduje si aj ich uloženie.



**Obr. č. 4: Interakcia MVC s užívateľom<sup>36</sup>**

Controller zobrazí view potrebný na výpis údajov a následne ich odošle do prehliadača používateľa. Vzhľadom rôznorodosti spôsobov realizácie, môže nastať aj situácia, kedy view zobrazuje údaje od modelu, respektíve model informuje view o zmene údajov. Na záver vytvorí view výpis údajov, najčastejšie vo forme HTML stránky, môže však použiť aj iné spôsoby zobrazenia, ktoré controller odošle späť do prehliadača vo forme HTTP odpovede.

<sup>36</sup>BERNARD, B. 2009. *Úvod do architektury MVC*. [online]. Praha : Devel.cz Lab s.r.o. [cit. 2012.01.19]. Dostupné na internete: < <http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/>>. ISSN 1803-5620.

## 2 Cieľ práce

Hlavným cieľom bakalárskej práce je na základe teoretických východísk týkajúcich sa platformy Zend Framework, analýza jeho novej vyvíjanej verzie Zend Framework 2.0 v konkrétnych podmienkach a komparácia jednotlivých komponentov s poukázaním na hlavné rozdiely a prínosy, ktoré so sebou prináša.

V záujme naplnenia takto formulovaného hlavného cieľa je potrebné si stanoviť a splniť nasledovné parciálne ciele.

- Analyzujeme teoretické východiska skúmanej problematiky.
- Navrhujeme pokyny prechodu na novú verziu.
- Stanovíme subjektívne a objektívne kritériá komparácie.
- Overíme výsledky komparácie.

Analýzou teoretických východísk poskytneme základný obraz o PHP frameworkoch a Zend Frameworku, kde podrobnejšie identifikujeme aj novú verziu Zend Framework 2.0. a popíšeme základne komponenty, ktoré sú súčasťou oboch knižníc.

S novou verziou sú spojené rôzne možnosti prechodu na ZF 2.0, ktoré navrhujeme prostredníctvom rozličných metód a postupov.

Stanovením objektívnych a subjektívnych kritérií komparácie, vyhodnotíme výsledky práce, pomocou ktorých môžeme obe verzie analyzovať a dospieť k všeobecne platným výsledkom. Subjektívne kritériá určujeme na základe vlastných skúseností a snažíme sa dokázať, či boli ciele spoločnosti Zend, očakávané od ZF2 splnené, na základe vlastného uváženia a vychádzajúc z výsledkov analýzy.

Na záver overíme výsledky komparácie, ktoré vedú k identifikácii problémov Zend Frameworku 1.X a výsledné vyhodnotenie prínosov Zend Frameworku 2.0.

### 3 Metodika práce a metody skúmania

Na základe stanovených cieľov sme stanovili základné metódy, ktoré sme uplatnili pri vypracovaní bakalárskej práce.

V prvom rade sme sa zaoberali *teoretickými východiskami* problematiky frameworkov, kde sme sa zamerali predovšetkým na:

- získanie základných teoretických údajov o frameworkoch a samotnom Zend Frameworku,
- popis vývojového cyklu a s ním spojený návrh metodiky prechodu na novú verziu,
- stručné oboznámenie sa so Zend Frameworkom 2.0,
- popis vybraných komponentov.

Pre správne vypracovanie bolo nutné získať najmä aktuálne informácie týkajúce sa novej verzie. V práci sa nachádza analýza a popis momentálne najnovšej verzie ZF2.0 Beta 3. Je možné, že oficiálna verzia, ktorá vyjde na konci leta bude obsahovať niekoľko zmien, no funkcionality ostane nezmenená. Primárnymi zdrojmi informácií, ktoré tvoria podklad pre vypracovanie teoretickej časti, boli najmä rôzne knižné publikácie zahraničných autorov.

Pri vypracovaní a spracovaní *teoretických východísk* týkajúcich sa ZF2, boli informácie získavané najmä z internetových zdrojov, keďže nie je k dispozícii ešte žiadna odborná literatúra. Podkladom pri vypracovaní boli príspevky z oficiálnej stránky ZF, kde sa nachádzajú najnovšie informácie o ZF2, popis komponentov a zmien, ktoré sú pravidelne aktualizované a modifikované.

Pri spracovaní *vlastnej práce*, kde prebehla samotná komparácia a analýza nových komponentov, boli podkladom informácie získané z videozáznamov medzinárodných konferencií ZendConf, kde boli prezentované teoretické a praktické poznatky samotnými vývojármi ZF2. Významným zdrojom a materiálom pre pochopenie sú ukážkové vzorové aplikácie umiestnené v GIT repozitáre, kde sú uložené rôzne projekty prispievateľov do ZF, pravidelne opravované chyby, ktoré sú verejne prístupné.

Na základe takto získaných informácií boli hlavnými metódami spracovania údajov analýza a komparácia, ktoré prebehli vyhodnotením objektívnych a subjektívnych kritérií. Ich analyzovaním s použitím metódy indukcie a dedukcie boli zhodnotené základné prínosy a prípadné nedostatky.

V rámci *objektívnych kritérií*, ktoré sú všeobecným ukazovateľom komparácie ZF1 a ZF2, sme sa zamerali na analýzu a komparáciu najmä týchto oblastí:

- nové jazykové funkcie a prístupy implementácie,
- nové komponenty,
- nová modulárna architektúra,
- nová implementovaná a prepracovaná MVC vrstva.

*Subjektívne kritéria*, vyhodnocujeme na základe vlastného vyhodnotenia a analýzy, ktoré priamo súvisia s naplnením hlavných požiadaviek, ktoré sa očakávajú od ZF2:

- jasnejšia dokumentácia,
- flexibilita,
- výkon.

## 4 Vlastná práca

Hlavným prínosom je podpora PHP 5.3, s ktorou je spojená najmä možnosť využívania *menných priestorov*. Ďalšou hlavnou zmenou je princíp „*design by contract*“, čo môžeme voľne preložiť ako „*návrh podľa zmluvy*“, poskytujúci abstraktné rozhranie pre aplikácie.

### *Menné priestory*

Menné priestory predstavujú spôsob zoskupenia kódu, ktorý existuje vo viacerých súboroch. Predstavujú adresáre so súbormi, majú určitú štruktúru a hierarchicky organizujú kód. Umožňujú pomenovať dve triedy rovnakým názvom.<sup>37</sup>

Slúžia na zapuzdrenie jednotlivých položiek. Pri ich implementácii je nutné prepísať celý kód pôvodnej aplikácie, lebo sú deklarované v každom súbore. ZF obsahuje aj osobitné menné priestory pre jednotkové testy. K triedam, ktoré nie sú súčasťou menných priestorov prístupujeme pomocou aliasov. Ich použitím dôjde k nasledujúcej zmene:

ZF1 → Zend\_Foo

ZF2 → Zend\Foo

Pre lepšie znázornenie si zoberme príklad definície formulára:

```
class Application_Form_Contact extend Zend_Form
{
}
```

**Skript č. 1: Definícia formuláru bez využitia konceptu menných priestorov**

Vďaka menným priestorom je možné vytvoriť model, ktorý vyzerá nasledovne:

```
namespace Application\form;

class Contact extend\Zend\form Form
{
}
```

**Skript č. 2: Definícia formuláru s využitím konceptu menných priestorov**

<sup>37</sup>MCARTHUR, K. 2008. *Pro PHP Patterns, Frameworks, Testing and More*. United States of America : Apress, 2008. s.73. ISBN 978-1-59059-819-1.

Pristupovať k menným priestorom je možné:

1. **Import konkrétnej triedy s použitím menných priestorov** – umožňuje importovať menný priestor do iného súboru použitím kľúčového slova *use*.

```
use My\Db\Statement\Sqlsrv;  
$stmt = new Statement\Sqlsrv();
```

Skript č. 3: Import triedy

2. **Pristupovanie k jednotlivým členom triedy prostredníctvom aliasov** – umožňuje importovať dve menné priestory s rovnakým názvom a rôzne ich pomenovať.

```
use My\Db\Statement\Sqlsrv as DbStatement;  
  
use My\Db\Adapter\Sqlsrv as DbAdapter;  
$stmt = new DbStatement ();  
$adapter = new DbAdapter();
```

Skript č. 4: Aliasy v menných priestoroch

### *Design by contract*

Nový prístup „*design by contract*“, predstavuje zmenu časti frameworku. Všetky komponenty majú vlastné rozhranie. Vytvára alternatívnu implementáciu štandardných tried a zároveň zabezpečuje, že táto implementácia naďalej pracuje s triedami.<sup>38</sup>

## 4.1 Nové komponenty

ZF2 prináša rad nových komponentov, ktoré sa v ZF1.X nenachádzajú, ako aj sadu refaktorovaných (vysvetlivka pod čiarou)<sup>39</sup> komponentov.

Problémy s *výkonom* priamo rieši:

- nový refaktorovaný Autoloader.

*Flexibilitu*, pružnosť a vyššiu modularitu zabezpečujú:

- EventManager,
- Dependency Injection.

<sup>38</sup>O'PHINNEY, W. M. 2011. *Zend Framework 2.0 Roadmap*. [online]. 2011. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV2/Zend+Framework+2.0+Roadmap>>.

<sup>39</sup>Refaktoring je proces softvérového vývoja a údržby, pri ktorom dochádza k zlepšeniu štruktúry a návrhu existujúceho kódu.



### 4.1.1 Autoloader

Autoloading znamená automatické načítanie tried a ostatných zdrojov, ktoré neboli vložené pomocou PHP funkcie *require\_once*. ZF poskytuje dve základné typy Autoloaderov, ktoré sa od seba odlišujú (tab. 2).

**Tab. č. 2: Porovnanie automatického načítania súborov ZF1 a ZF2<sup>4041</sup>**

ZF1	ZF2
<b>Zend_Loader_Autoloader</b> zabezpečuje nahrávanie tried podľa konvencií z <code>include_path</code> . Funguje na základe PEAR noriem, kde každá trieda a súborový systém sú v pomere 1:1. Táto trieda implementuje rozhranie <i>Zend_Loader_Autoload_Interface</i> a využíva návrhový vzor Singleton, v súlade s konvenciami pomenovania. Existujú však systémy, kde nie je vzťah v pomere 1:1.	<b>Zend_Loader_Autoloader</b> je určený na správu menných priestorov knižnice, ktoré nedodržiavajú štandardné pokyny a nie sú v pomere 1:1 medzi názvom triedy a adresárovou štruktúrou. Poskytuje koncové podčiarknutia („_“) pre menné priestory, tak že Autoloader im zodpovedá.
<b>Zend_Loader_AutoloaderResource</b> umožňuje nahrávanie tried z rôznych aplikačných špecifických zdrojov a vlastné usporiadanie adresárov. Používa koncové podčiarknutia pri registrácii zdrojov.	<b>Zend_Loader_AutoloaderResource</b> predpokladá, že celý kód bude načítaný automaticky. Nie je nutné používať koncové podčiarknutia pri registrácii zdrojov.
Nahrávané sú pomocou <b>Zend_Application_Module_Autoloader</b> .	
<b>StandardAutoloader</b> – nastavuje absolútnu alebo relatívnu cestu volanému skriptu. Je volaný pomocou funkcie <i>require_once</i> .	<b>StandardAutoloader</b> načítava triedy podľa mena triedy a na základe toho nájde požadovaný súbor na disku. Nevyužíva už funkciu <i>require_once</i> .
<b>Využívajú dve stratégie:</b> <ol style="list-style-type: none"> <li>1. Vyhľadávanie pomocou konvencií <code>include_path</code>.</li> <li>2. Vyhľadávanie na základe prefixového systému.</li> </ol>	
Prefixový zápis sa používa pre nemenné triedy, ktorých adresáre sú oddelené podčiarkovníkmi.	Adresáre menných priestorov sú oddelené podľa jednotlivých menných priestorov.

<sup>40</sup>O'PHINNEY, W. M. 2011. *Proposal For Autoloading In ZF2*. [online]. 2011. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV2/Proposal+For+Autoloading+In+ZF2>>.

<sup>41</sup>O'PHINNEY, W. M., SCHINDLER, R. 2010. *Introducing Zend Framework 2.0*. [online]. SlideShare Inc. 2010. [cit. 2012.01.20]. Dostupné na internete: <<http://www.slideshare.net/weierophinney/introducing-zend-framework-20>>.

Príklad *Zend\Loader\StandardAutoloader* v ZF2.

```
require_once ZF2_PATH . '/Loader/StandardAutoloader.php';
$loader = new ZendLoaderStandardAutoloader();
$loader->registerPrefix('MyVendor', __DIR__ . '/MyVendor')
    ->registerNamespace('MyNamespace', __DIR__ . '/MyNamespace')
    ->setFallbackAutoloader(true);

$loader->register();
```

Skript č. 5: StandardAutoloader v ZF2

### *Zend\Loader\ClassMapAutoloader*

Trieda *ClassMapAutoloader* predstavuje vysoko výkonný spôsob automatického načítavania. Používa *ClassMap* triedy, ktoré sú jednoduché asociatívne polia. ZF poskytuje dva základné nástroje umožňujúce vytvorenie *ClassMap*.

1. **Nástroj automatického generovania ClassMaps** – ZF2 poskytuje nástroj príkazového riadku s názvom *classmap\_generator.php*. V predvolenom nastavení vytvára súbory s názvom „Classmap.php“, pomocou *\_\_DIR\_\_* zobrazujúceho prefix cesty.

```
$ cd your\library
$ php \path\to\classmap_generator.php -w
```

2. **Nástroj pre generovanie vlastnej ClassMap** - v predvolenom nastavení vytvorí súbor s názvom „autoload.php“, využíva menný priestor v aktuálnom adresári. Keď je vytvorený súbor *ClassMap* s názvom *autoload\_classmap.php* sa načítajú triedy, ktoré prislúchajú danému súboru.

```
require_once ZF2_PATH . '/Loader/ClassMapAutoloader.php';
$autoLoader = new ZendLoaderClassMapAutoloader(
    array(__DIR__ . '/autoload_classmap.php'));
$autoLoader->register();
```

Skript č. 6: ClassMapAutoloader v ZF2

### *Zend\Loader\AutoloaderFactory*

*AutoloaderFactory* zjednodušuje proces zavádzania a viaže sa priamo na *Zend\_Application*. Umožňuje načítať viacero *Autoloaderov* naraz. Kombinuje stratégie *ClassMapAutoloader* a *Standard Autoloader* (skript 7).

```

require_once ZF2_PATH . '/Loader/AutoloaderFactory.php';
ZendLoaderAutoloaderFactory::factory(array(
    'ZendLoaderClassMapAutoloader' => array(
        __DIR__ . '/../library/Zend/autoload_classmap.php',
    ),
    'ZendLoaderStandardAutoloader' => array(
        'prefixes' => array(
            'MyVendor' => __DIR__ . '/MyVendor',
        ),
        'namespaces' => array(
            'MyNamespace' => __DIR__ . '/MyNamespace',
        ),
        'fallback_autoloader' => true,
    ),
));

```

Skript č. 7: Autoloader Factory v ZF2

### *Prínosy nových spôsobov automatického nahrávania*<sup>42</sup>

- Použitím ClassMapAutoloader došlo k zlepšeniu výkonu o 25%.
- Použitím AutoloaderFactory došlo k zvýšeniu výkonu o 60%.
- Odstránenie funkcie require\_once umožňuje jednoduchšie použitie a konfiguráciu.

#### *4.1.2 Event Manager*

Event Manager, nový významný komponent, predstavuje kľúčovú súčasť ZF2, je jadrom systému MVC. Jeho najvýznamnejším prínosom je aspekt objektovo orientovaného programovania, čo umožňuje elimináciu Singletonov. Integruje návrhový vzor Observer a je založený na udalost'ami riadenej (z angl. *Event Driven*) architektúre. Rieši problémy s protokolovaním kódu, ukladá záznamy do vyrovnávajúcej pamäti bez nutnosti rozšírenia frameworku. Vykonáva zmeny v častiach kódu, bez porušenia globálnej štruktúry projektu. V súvislosti s EventManager sú dôležité tieto pojmy:<sup>43</sup>

- **EventManager** – objekt, ktorý agreguje listenery pre jednu alebo viac udalostí (z angl. *event*) a spúšťa ich,
- **Listener** – predstavuje spätné volanie, ktoré reaguje na udalosti,
- **Event action** – predstavuje akciu, ktorá spúšťa EventManager.

<sup>42</sup>ZIMUEL, E. 2012. *Zend Framework 2.0 quick start*. [online]. Moscow : SlideShare Inc. 2012. [cit. 2012.04.20]. Dostupné na internete: <<http://www.slideshare.net/e.zimuel/zend-framework-2-quick-start>>.

<sup>43</sup>O'PHINNEY, M. W. 2011. *Using the ZF2 EventManager*. [online]. 2011. [cit. 2012.02.23]. Dostupné na internete: <<http://mwop.net/blog/266-Using-the-ZF2-EventManager.html>>.

### Spustenie EventManagera

EventManager spustí udalosť v ClassMap triede, kde *\$events* predstavuje inštanciu EventManageru pomocou metódy *events()*. Spustí dve udalosti, ktoré poskytujú:<sup>44</sup>

1. Názov cieľovej triedy, ktorá spustí udalosť pomocou *\$this*.
2. Rad údajov, ktoré sú poskytované a prijímané listenermi.

```
use Zend\EventManager\EventManager;
use Zend\EventManager\Event;
class PhotoMapper
{
    public $events;

    public function events()
    {
        if (!$this->events) {
            $this->events = new EventManager(__CLASS__);
        }
    }
    return $events;
}
public function findById($id)
{
    $this->events()->trigger(__FUNCTION__ . '.pre', $this,
        array('id' => $id));

    $this->events()->trigger(__FUNCTION__ . '.post', $this,
        array('photo' => $photo));

    return $photo;
}
```

Skript č. 8: Spustenie EventManager v ZF2

### Volanie udalostí

Listener je PHP funkcia, ktorá vracia jeden argument. Je pripojený k EventManageru pomocou metódy *attach()*. *\$event* je parameter, ktorý je odovzdaný metóde listenera. Obsahuje tri používateľské metódy:

1. *getName()* – umožňuje pripojiť viacnásobné udalosti,
2. *getTarget()* – umožňuje spustiť požadovanú udalosť,
3. *getParams()* – umožňuje načítať odoslané parametre spustených udalostí.

```
$photoMapper->events()->attach('findById.pre', function(Event $event) {
    $message = "Trying to retrieve photo: " . $event->getParam('id');
    MyLogger::log($message); });
```

Skript č. 9: Volanie udalosti v ZF2

<sup>44</sup>ALLAN, R. 2012. *An introduction to ZendEventManager*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: < <http://akrabat.com/zend-framework-2/an-introduction-to-zendeventmanager/> >.

### Vytvorenie spojenia

Návratovou hodnotou metódy *trigger()* je súhrn všetkých vrátených výsledkov z každého listenera. Príklad ukladania do vyrovnávajúcej pamäte, kde je volaná metóda *trigger()*, ktorej návratovou hodnotou je *true*, ak výsledok vráti listenera, ktorý je inštanciou požadovanej triedy.

```
public function findById($id)
{
    $results = $this->events()->trigger(__FUNCTION__ . '.pre', $this,
        array('id' => $id),
        function ($result) {
            return ($result instanceof Photo) ? true : false;
        }
    );
    if ($results->stopped()) {
        $photo = $results->last();
        return $photo;
    }

    $this->events()->trigger(__FUNCTION__ . '.post', $this,
        array('photo' => $photo));
    return $photo;
}
```

Skript č. 10: Vytvorenie spojenia EntityManager v ZF2

## 4.2 Dependency Injection

Dependency Injection (skrát. DI) predstavuje nový návrhový vzor, ktorého cieľom je znížiť väzbu medzi jednotlivými zložkami kódu. Zaisťuje objektom kompozíciu. Nie je už nutné pracovať s objektmi pomocou jednej hierarchie založenej na dedičnosti, teraz reprezentujú určitú množinu objektov, kde ďalšie objekty vkladáme (z angl. *inject*) jeden do druhého. V ZF2 je tento prístup reprezentovaný komponentom *Zend\Di*. Dependency Injection priradzuje každému objektu inštanciu. Poskytuje podporu najmä pre prácu controlleru.

Existuje viacero spôsobov implementácie týchto objektov:<sup>45</sup>

- prostredníctvom konštruktora,
- prostredníctvom metódy setter,
- prostredníctvom aliasov, cez ktoré je vkladán externý objekt.

---

<sup>45</sup>BRADY, P. 2008. *The Zend Framework, Dependency Injection and Zend\_Di*. [online]. 2008. [cit. 2012.01.23]. Dostupné na internete: < <http://blog.astrumfutura.com/2008/02/the-zend-framework-dependency-injection-and-zend-di/>>.

### 4.2.1 Prostredníctvom konštruktora

Príklad zobrazuje kód, kde sme na začiatku vytvorili ako prvý DatabaseAdapter, lebo konštruktor je naň priamo pripojený.

```
namespace My;

class DatabaseAdapter
{
}

class UserTable
{
    protected $db;

    public function __construct (DatabaseAdapter $db)
    {
        $this->db = $db;
    }
}
```

**Skript č. 11: DI – pripojenie prostredníctvom konštruktora**

Komponent *Zend\Di* použije \$db parameter v konštruktoře UserTable a metódou get() vráti vždy rovnaký parameter. Keď bude volaná \$di->get() metóda, spracuje konštruktor UserTable.

```
$db = new My\DatabaseAdapter();
$userTable = new My\UserTable($db);
$di = new Zend\Di\Di();
$userTable = $di->get('My\UserTable');
```

**Skript č. 12: DI – získanie hodnoty premennej pomocou konštruktora**

Následne môžeme dostať interný objekt \$userTable. Volaním \$di->bude inštancia triedy automaticky zdieľaná.

Na vytvorenie novej inštancie je možné použiť \$di->newInstance().

```
$userTable = $di->get('My\UserTable');
```

**Skript č. 12: DI – návratová hodnota z konštruktora**

### 4.2.2 Prostredníctvom setter

Setter Injection je ďalším často používaným spôsobom nastavenia závislostí triedy. Na začiatku sme vytvorili definíciu triedy.

```
namespace My;

class DatabaseAdapter
{
    protected $dsn;
    public function __construct($dsn)
    {
        $this->dsn = $dsn;
    }
}

class UserTable
{
    protected $db = null;

    public function setDatabaseAdapter(DatabaseAdapter $db)
    {
        $this->db = $db;
    }
}
```

Skript č. 13: DI – definícia setter metódy

Objekt DI `setDatabaseAdapter()` je inštanciou `My\UserTable` objektu. Môže mať aj viac metód `set`, ktorými je DI volaný.

```
$di = new \Zend\Di\Di();
$di->configure(new \Zend\Di\Configuration(array(
    'definition' => array(
        'class' => array(
            'My\UserTable' => array(
                'setDatabaseAdapter' => array('required' => true)
            )
        )
    )
));
$userTable = $di-
>get('My\UserTable', array('dsn'=>'mysql:host=server2;dbname=somedb'));
```

Skript č. 14: DI – volanie set metódy

### 4.2.3 Dependency Injection Contajner

Dependency Injection Contajner (skrát DIC) predstavuje kontajner komponentov, ktorého úlohou je vytvárať požadované objekty, ktoré nie sú závislé na aplikácii aj v prípade, že pochádzajú z rôznych vrstiev aplikácie a riadi celý proces závislostí. Neobsahuje žiadne statické závislosti. Najvýznamnejšiu úlohu má *ServiceLocator* rozhranie.

### ***ServiceLocator rozhranie***

ServiceLocator je objekt, ktorý vytvára a získava požadované objekty ich vstreknutím do hľadaných závislostí. Dependency Injection Container obsahuje *Zend\Di\ServiceLocator\Generator* komponent, ktorý konfiguruje inštancie DI a vytvára ServiceLocator triedy. Metóda *getCodeGenerator()* vracia inštanciu *Zend\php\Codegenerator*, z ktorej môžeme napísať triedu s novým ServiceLocator.<sup>46</sup>

```
use Zend\Di\ServiceLocator\Generator;

$generator = new Generator($di);

$generator->setNamespace('Application')
    ->setContainerClass('Context');
$file = $generator->getCodeGenerator();
$file->setFilename(__DIR__ . '/../Application/Context.php');
$file->write();
```

**Skript č. 15: DIC - Service Locator**

## **4.3 Adresárová štruktúra**

ZF poskytuje rozličné spôsoby tvorby projektov. Jednou z možností je použitie skriptu *Zend\_Tool*, ktorý pomocou príkazového riadku automaticky vygeneruje celú stromovú adresárovú štruktúru projektu. Kvôli lepšej štruktúrovanosti je výhodné využiť moduly. Sú inicializované pomocou *Zend\_Application\_Resource\_Modules*.

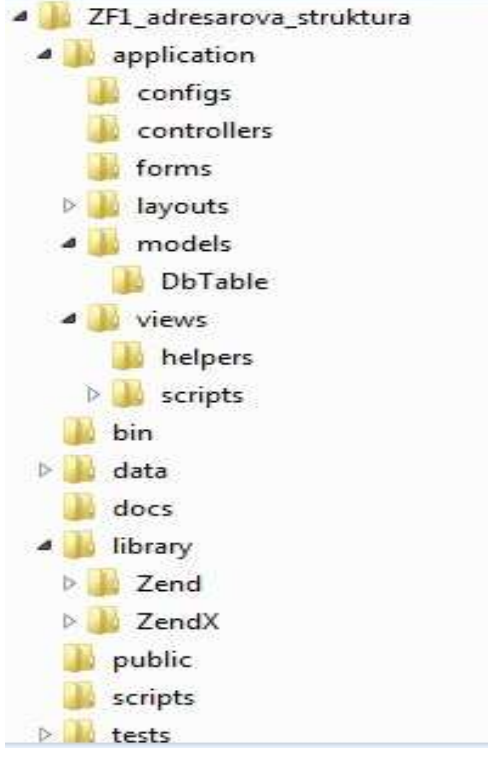
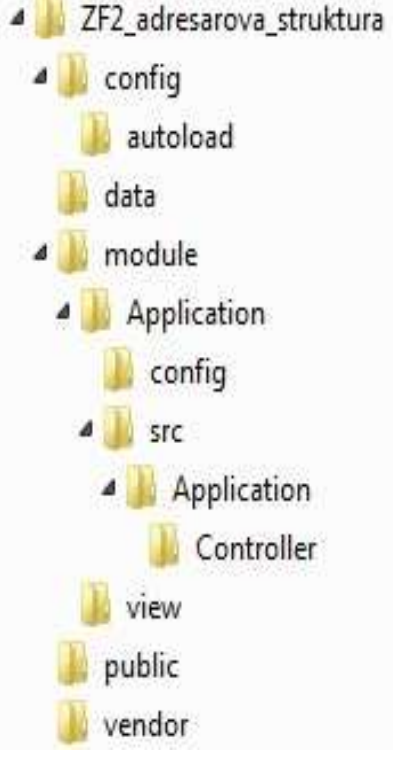
V novej verzii boli mnohé komponenty odstránené alebo premenované. ZF2 reprezentuje nový výkonný prístup založený na moduloch. Modul je kolekcia kódov, môže obsahovať obrázky, CSS a iné zdroje. Predstavujú spôsob, ako usporiadať kód do logických blokov. Princíp spočíva vo vytvorení vlastného modulu, ktorý obsahuje podadresáre so súbormi. Sú založené na logike MVC. Moduly predstavujú menné priestory.

Anatómia aplikácie obsahuje mnoho adresárov, čím je zaistená nezávislosť jednotlivých častí. Odporúčaná adresárová štruktúra v ZF1 a ZF2 a popis hlavných funkcií jednotlivých adresárov (tab. 3), kde každý komponent zohráva osobitnú úlohu a má rôzne účely použitia.

**Tab. č. 3: Porovnanie adresárovej štruktúry ZF1 a ZF2<sup>4748</sup>**

<sup>46</sup>ZEND TECHNOLOGIES, Inc. 2012. *Programmer's Reference Guide*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://packages.zendframework.com/docs/latest/manual/en/learning.di.generating-service-locators.html/>>.



 <p>ZF1_adresarova_struktura</p> <ul style="list-style-type: none"> <li>application <ul style="list-style-type: none"> <li>configs</li> <li>controllers</li> <li>forms</li> <li>layouts</li> <li>models <ul style="list-style-type: none"> <li>DbTable</li> </ul> </li> <li>views <ul style="list-style-type: none"> <li>helpers</li> <li>scripts</li> </ul> </li> </ul> </li> <li>bin</li> <li>data</li> <li>docs</li> <li>library <ul style="list-style-type: none"> <li>Zend</li> <li>ZendX</li> </ul> </li> <li>public</li> <li>scripts</li> <li>tests</li> </ul>	 <p>ZF2_adresarova_struktura</p> <ul style="list-style-type: none"> <li>config</li> <li>autoload</li> <li>data</li> <li>module <ul style="list-style-type: none"> <li>Application <ul style="list-style-type: none"> <li>config</li> <li>src <ul style="list-style-type: none"> <li>Application <ul style="list-style-type: none"> <li>Controller</li> </ul> </li> <li>view</li> </ul> </li> </ul> </li> </ul> </li> <li>public</li> <li>vendor</li> </ul>
<p><b>application</b> je hlavný adresár, v ktorom sú všetky aplikačné kódy a konfiguračné údaje vrátane:</p> <ul style="list-style-type: none"> <li>➤ <b>controllers</b> – nachádzajú sa v adresári application/controllers,</li> <li>➤ <b>models</b> – nachádzajú sa v adresári application/models,</li> <li>➤ <b>views</b> – nachádzajú sa v adresári application/view/scripts.</li> </ul>	<p>Hlavný koreňový adresár obsahuje organizačný adresár spolu s Module.php a podadresáre:</p> <ul style="list-style-type: none"> <li>➤ <b>config</b> – konfiguračné súbory a metadáta s názvom, číslom verzie a informáciami o autorovi,</li> <li>➤ <b>src</b> – obsahuje vlastný modul, ktorého súčasťou je Controller. Všetky triedy sú menné priestory, ktoré sú načítame pomocou ModuleManager z Module.php.,</li> <li>➤ <b>view</b> – pre view skripty.</li> </ul>

<sup>47</sup>ALLAN, R. 2012. *Modules in ZF2*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: < <http://akrabat.com/zend-framework-2/modules-in-zf2/>>.

<sup>48</sup>ZIMUEL, E. 2012. *Zend Framework 2.0 quick start*. [online]. Moscow : SlideShare Inc. 2012. [cit. 2012.04.20]. Dostupné na internete:<<http://www.slideshare.net/e.zimuel/zend-framework-2-quick-start>>.

<b>Data</b> slúži na ukladanie dočasných dát.	
<b>Docs</b> obsahuje dokumentáciu, ktorá je buď vygenerovaná automaticky alebo priamo napísaná v tomto adresári.	
<b>Library</b> obsahuje hlavnú knižnicu ZF, ako aj vlastné knižnice.	
<b>Public</b> predstavuje koreňový adresár aplikácie. Je postavený na verejne prístupných zdrojoch ako sú obrázky, CSS štýly a JavaScripty.	
<b>Skipts</b> spravuje všetky skriptá príkazového riadku a zabezpečuje konfiguráciu.	<b>Vendor</b> obsahuje podadresár iných dodávateľov modulov alebo knižníc.
<b>Tests</b> obsahuje všetky aplikačné testy, vrátane unit testov.	

#### 4.3.1 Nový modulový systém ZF2

Nový modulový systém sa skladá sa z týchto častí:

- **Module Manager** – *Zend\Module\Manager* zabezpečuje načítanie a konfiguráciu modulov v koreňovom adresári a spúšťa sled udalostí. V predvolenom nastavení systém očakáva, že každý modul predstavuje inštanciu triedy, ktorá volá tri metódy v rámci svojej modulevej triedy:

1. *getAutoloaderConfig()* – nastavuje modul automatického načítania,
2. *init()* – slúži na inicializáciu a pridanie udalostí do EventManageru,
3. *getConfig()* – riadi konfiguráciu modulov.

Volaním metódy *getAutoloaderConfig()*, načítame modul triedy. Návratovou hodnotou sú dva autoloader. ClassMap súbor vytvorený pomocou `bin\classmap_generator.php` (skript16).

```

<?php
namespace Simple;
use Zend\Module\Manager,
    Zend\Module\Consumer\AutoloaderProvider;
class Module implements AutoloaderProvider
{
    public function getAutoloaderConfig()
    {
        return array(
            'Zend\Loader\ClassMapAutoloader' => array(
                __DIR__ . '/autoload_classmap.php',
            ),
            'Zend\Loader\StandardAutoloader' => array(
                'namespaces' => array(
                    __NAMESPACE__ => __DIR__ . '/src/' . __NAMESPACE__,
                ),
            ),
        );
    }
    public function getConfig()

```

Skript č. 16: Module Manager v ZF2

Ak v predvolenom nastavení definujeme metódu *init()*, nahráme modul a vytvoríme inštanciu ModulManager. Tento kód volá EventManager. V bootstrap nastaví DI Locator, router, application a view objekty.

```

    public function init(Manager $moduleManager)
    {
        $events = StaticEventManager::getInstance();
        $events->
>attach('bootstrap', 'bootstrap', array($this, 'onBootstrap'));
    }

```

Skript č. 17: Definovanie Module Manager v ZF2

- **Modul Autoloader** – *Zend\Loader\ModuleAutoloader* predstavuje špecializovaný spôsob automatického nahrávania, zodpovedný za vyhľadávanie a načítanie modulov tried z rôznych zdrojov. Je založený na princípe automatického načítania. Nasledujúci príklad vyhľadá modul v miestnom adresári.

```

<?php
$appConfig = include __DIR__ . '/../configs/application.config.php';
$moduleLoader = new Zend\Loader\ModuleAutoloader(
    $appConfig->module_paths
); $moduleLoader->register();

```

Skript č. 18: Module Autoloader v ZF2

## 4.4 Nová prepracovaná MVC architektúra

Všetky vrstvy systému MVC sú založené na novom princípe modulov, udalosťami riadenej architektúre a prístupe „design by contract“, ktorý poskytuje rozhranie Dispatchable.

### *MVC v ZF 1*

Súčasná implementácia systému MVC je pomerne pomalá. Spájajú sa s ňou najmä nasledujúce problémy:

- Ako zaistiť závislosť controllerov?
- Ako efektívne využívať návrhové vzory Front Controller a ActionController?
- Ako zabezpečiť vyšší výkon?

### *MVC v ZF 2*

Nový prístup systému MVC, rieši tieto problémy a predstavuje pružný systém, ktorý umožňuje vytvoriť optimálnu, ľahko vytvoriteľnú modulárnu infraštruktúru. Jeho hlavnou úlohou je princíp explicitného mapovania URL ciest ku controllerom. Nová MVC architektúra je tvorená na vrchole týchto komponentov:<sup>49</sup>

- **Zend\DI** - konkrétne ServiceLocator rozhranie,
- **Zend\EventManager**- Zend\EventManager\EventDescription obsahuje Zend\Mvc\Mvc Event, ktorý udeľuje prístupové práva nasledujúcim objektom:
  1. Request objekty,
  2. Response objekty,
  3. Router objekty,
- **Zend\Http** - spracováva požiadavky a odpovede prostredníctvom rozhrania Zend\Stdlib\Dispatchable.

---

<sup>49</sup>ZEND TECHNOLOGIES, Inc. 2012. *Programmer's Reference Guide*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: < <http://packages.zendframework.com/docs/latest/manual/en/zend.mvc.html>>.

#### 4.4.1 Model

V ZF neexistuje komponent *Zend\_Model*. Implementácia častí modelu je tak rôznorodá a komplexná, že sme povinní vykonať ju sami. Predstavuje akúsi databázovú abstrakciu založenú na abstraktných triedach. Reprezentuje aplikačnú logiku aplikácie, jeho hlavnou úlohou je načítanie a ukladanie dát do databázy. Existuje viacero možností implementácie. Vytvorením modelovej triedy, ktorá pomocou *ClassMap* objektov načíta a uloží objekty do databázy, alebo použitím ORM, prípadne *Propel*.

##### ✦ *Zend\_DB* v ZF1

Najčastejší spôsob implementácie modelu v ZF je uskutočňovaný pomocou databázy, čo je reprezentované komponentom *Zend\_Db*. Najvýznamnejšiu úlohu majú komponenty *Zend\_Db\_Adapter* a *Zend\_Db\_Table*.

*Zend\_Db\_Adapter* poskytuje objektovo orientované rozhranie, ktoré zabezpečuje flexibilný, výkonný a predovšetkým jednoduchý spôsob načítavania, vkladania, aktualizácie a mazania dát z databázy. Je zodpovedný za úpravu akéhokoľvek kódu napísaného v *Zend\_Db*, preto je považovaný za jeho najdôležitejšiu súčasť. Inicializácia sa uskutočňuje pomocou statickej metódy *Zend\_db::factory()*, ktorej návratovou hodnotou je názov adaptéru.

Druhým hlavným komponentom je *Zend\_Db\_Table*, ktorý predstavuje implementáciu návrhového vzoru Table Data Gateway, kde každý potomok triedy predstavuje jednu tabuľku databázy. Trieda *Zend\_Db\_Table\_Row* implementuje návrhový vzor Table Data Row, kde každý potomok triedy predstavuje jeden riadok tabuľky.<sup>50</sup>

##### ✦ *Zend\_Db* v ZF2

Nový prístup predstavuje novú abstraktnú SQL vrstvu a upravené návrhové vzory Table Data Gateway a Row Data Gateway (tab. 4).

---

<sup>50</sup>BÖHMER, M. 2010. *Zend Framework : Programujeme webové aplikácie v PHP*. Brno : CPress, 2010. s. 139. ISBN 978-80-251-2965-4.

Tab. č. 4: Table Data Gateway a Row Data Gateway v ZF2<sup>51</sup>

Table Data Gateway	Row Data Gateway
V súčasnosti máme tabuľku operácií nachádzajúcu sa na dvoch miestach, a to <i>Zend_Db_Adapter</i> a <i>Zend_Db_Table</i> . Bola vytvorená flexibilnejšia Table Data Gateway implementácia. Prijíma <i>Zend_Db_Query</i> objekty a vykonáva funkcie <i>insert()</i> , <i>update()</i> , <i>delete()</i> , <i>select()</i> .	Obsahuje nový komponent <i>Zend\Db\Metadata</i> , ktorý spracováva schému databázy za účelom spracovania riadkových operácií. Využíva taktiež <i>Zend\Db\ResultSet</i> , ktorého úlohou je odovzdať všetky riadky tabuľky z objektu <i>RowGateway</i> . Tiež je spojený s možnosťou uloženia alebo zmazania údajov z trvalého miesta uloženia.
<b>Obe návrhové vzory je možné definovať aj za behu programu.</b>	

Tab. č. 5: Porovnanie modelu v ZF1 a v ZF2<sup>52</sup>

ZF1	ZF2
Zložité pripájanie k databáze.	Nový prístup nerieši problém modelovej oblasti.
Neposkytuje možnosť zdieľania dát medzi rôznymi triedami.	Využíva DI a DIC, ktoré umožňujú zdieľanie medzi triedami ich stretnutím do objektov.
Neposkytuje možnosť vytvorenia schémy metadát rôznymi spôsobmi.	Pomocou rozhrania Metadata môžeme vytvoriť schému metadát.

#### 4.4.2 View

Funkcie view sú v ZF implementované pomocou komponentov *Zend\_View* a *Zend\_Layout*.

##### ✦ *Zend\_View* v ZF1

*Zend\_View* poskytuje flexibilný systém zobrazovania údajov. Interakcia s užívateľom je zabezpečená pomocou front-end kódu, ktorý slúži na renderovanie logiky návrhu prostredníctvom XHTML, JavaScriptu, CSS a iných technológií.

Viacstupňové layouty sú tvorené komponentom *Zend\_Layout*, ktorý vytvára centrálny layout stránky.

<sup>51</sup>SCHINDLER, R. 2010. *Zend Db 2.0 Requirements*. [online]. 2010. [cit. 2012.02.23]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV2/Zend+Db+2.0+Requirements>>.

<sup>52</sup>SCHINDLER, R. 2012. *Zend\Db in ZF 2.0*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://static.zend.com/topics/ZendDb-2.pdf>>.

## ➤ *Zend\_View* v ZF 2

*Zend\_View* v ZF2 obsahuje nové čiastkové komponenty:<sup>53</sup>

- **Zend\View\Model** predstavuje komplexnú hierarchiu zobrazenia, slúži na reprezentáciu a agregáciu dát. Všetky dáta zviaže do jedného objektu *ViewModel*. *ViewModel* objekt vracia z controllera premenné, ktoré sú použité v rámci view skriptu spolu s metainformáciami na renderovanie view skriptu.
- **Zend\View\Resolver** poskytuje mená šablón, ktoré si vyžaduje objekt *Renderer*.
- **Zend\View\Renderer** generuje šablóny buď prostredníctvom premennej alebo pomocou modelov.
- **Zend\View\Strategy** využíva *EventManager* komponent pre výber stratégie renderovania.

Hlavné zmeny súvisiace s novým *Zend\_View* v ZF (tab. 6).

Tab. č. 6: *Zend\_View* v ZF2<sup>54</sup>

Zend_View	
Pôvodná <i>Zend_View</i> trieda bola presunutá do <i>Zend\View\Renderer\PhpRenderer</i> , s cieľom preniesť viac právomocí na helperov, čo výrazne zjednodušuje dizajn a poskytuje viac možností.	Nová trieda <i>Zend\View\View</i> umožňuje výber stratégií jednotlivých šablón na základe ľubovoľných kritérií a posiela výsledky do response objektu a request objektu pomocou DI.  Systém už automaticky zabezpečuje, aby boli premenné extrahované do view skriptov.

<sup>53</sup>ZEND TECHNOLOGIES, Inc. 2012. *Programmer's Reference Guide*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://packages.zendframework.com/docs/latest/manual/en/zend.view.html#zend.view.quick-start>>.

<sup>54</sup>O'PHINNEY, W. M. 2012. *View Layers, Database Abstraction, Configuration, Oh, My!* [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://mwop.net/blog/zf2-beta3.html>>.

## **Zend\***View*

Existujú dva spôsoby prístupovania k premenným vo view skriptoch. Jedným z nich je prostredníctvom *initView()* metódy v Action Controllery. Umožňuje spracovávať hodnoty premenných a view skriptov. Inicializáciou objektov view nastavíme cesty ku skriptom view, helperom a filtrom.

```
class FooController extends Zend_Controller_Action
{
    public function init()
    {
        $this-> initView();
    }
    public function listAction()
    {
        $this->view->assign('meno' , 'a');
        $this->view->zoznam = array(
            1=> 'ahoj',
            2=> 'dobry');
        $this ->render();
    }
}
```

**Skript č. 19: Zend View v ZF1**

## **Zend\***View\***View**

*Zend\***View\***View* využíva nový návrhový vzor ViewModel. Vytvára sa v Action Controllery a nastaví všetky premenné a názov šablóny vo ViewModely. Hlavný rozdiel spočíva v tom, že nepoužívame zápis v tvare *\$this->view*, hodnoty premenných nastavíme priamo pomocou *\$view*.

```
namespace Foo\Controller;
use Zend\Mvc\Controller\ActionController,
    Zend\View\Model\ViewModel;
class BarController extends ActionController
{
    public function doSomethingAction()
    {
        $view = new ViewModel(array(
            'message' => 'Hello world',
        ));
        $view->setTemplate('bar/do-something');
        return $view;
    }
}
```

**Skript č. 20: Zend View v ZF2**

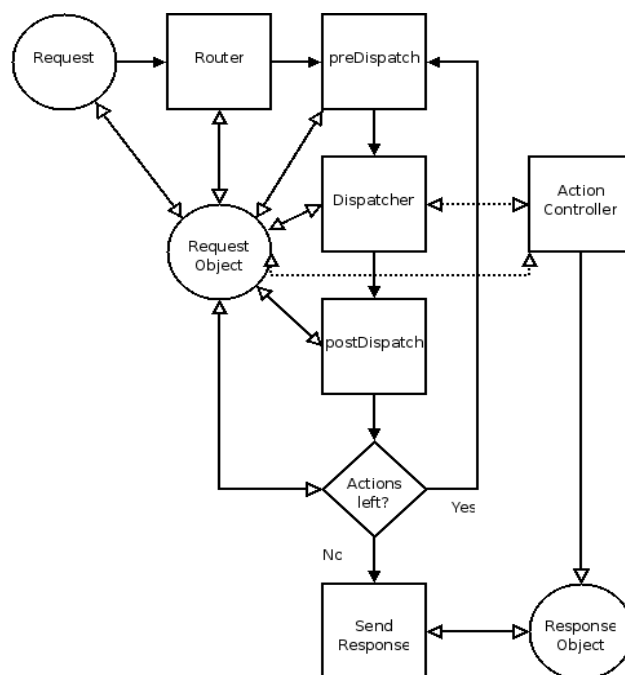


#### 4.4.3 Controller

Je jadrom MVC architektúry. Funkcie controlleru sú zabezpečované v ZF pomocou komponentu *Zend\_Controller*, ktorý poskytuje výkonnú a flexibilnú sadu pomocných prostriedkov, taktiež podporuje pluginy na všetkých úrovniach procesu.

##### **Princíp fungovania Controlleru**

Princíp fungovania controlleru (obr. 5) začína, keď Bootstrap aktivuje Front Controller, ktorý zodpovedá za zviazanie celej aplikácie dohromady. Spracovanú žiadosť odošle routeru, dispatcher nastaví response objekt zodpovedajúcemu modulu a následne sú všetky controllery a akcie odoslané späť Front Controlleru. Na záver Front Controller odošle požiadavky dispatcherovi, ktorý inicializuje modul, controller a požadované akcie.



Obr. č. 5: Princíp fungovania Controlleru<sup>55</sup>

Implementácia sa uskutočňuje prostredníctvom návrhových vzorov:

- Front Controller,
- Action Controller

<sup>55</sup>O'PHINEY, M. W. 2008. *Zendframeworkworkshop*. [online]. Slideshares. Inc. 2008. [cit. 2012.01.23]. Dostupné na internete: < <http://static.slideshare.net/swf/ssplayer2.swf?doc=20080613zendframeworkworkshop-1213725963433006-9/>>.

## Controller vrstva v ZF2

Základná štruktúra aplikácie pozostáva z diskretných request a response objektov. Nová controller vrstva je riadená prostredníctvom rozhrania *Dispatchable*. Dostupné ovládače controlleru sú riadené pomocou *Zend\Stdlib\Dispatchable*. Na základe tohto nového prístupu môžeme žiadosti a odpovede na jednotlivé požiadavky jednoducho agregovať pomocou metadát.

```
use Zend\Stdlib\Dispatchable,  
    Zend\Stdlib\RequestDescription as request,  
    Zend\Stdlib\ResponseDescription as response;  
class Foo implements Dispatchable  
{  
    public function dispatch  
    (Request $ request, response $ response = null)  
    {}  
}
```

**Skript č. 21: Dispatchable Request a Response objekty v ZF2**

Predmetom vykonávania *Zend\Stdlib\Dispatchable* sú nasledujúce tri abstraktné triedy:<sup>56</sup>

1. **Zend\MVC\Controller\FrontController**
2. **Zend\MVC\Controller\ActionController**
3. **Zend\MVC\Controller\RestfulController**

*Front Controller* ako aj *Action Controller* boli implementované s menšími zmenami, kým *Restful Controller* predstavuje novú abstraktnú triedu, s ktorou sme sa v ZF1 nestretli.

### ➤ *Front Controller v ZF1*

Front Controller reprezentuje jadro celého Zend\_Controlleru. V ZF je zastúpený triedou *Zend\_Controller\_Front*, ktorá ho implementuje. Jeho hlavnou úlohou je koordinovanie interakcií jednotlivých komponentov, zabezpečuje konfiguráciu modulov, riadiacich systémov a akcií, pričom je nutné mu poskytnúť informácie o umiestnení a organizácii controllerov.<sup>57</sup>

<sup>56</sup>ZEND TECHNOLOGIES, Inc. 2012. *Programmer's Reference Guide*. [online]. 2012. [cit. 2012.04.23].

Dostupné na internete: <<http://packages.zendframework.com/docs/latest/manual/en/zend.mvc.quick-start.html>>.

<sup>57</sup>COGGESHALL, J., TOCKER, M. 2009. *Zend Enterprise PHP Patterns*. United States Of Amerika : Apress , 2009. s. 5. ISBN 978-1-4302-1975-0.

## ✦ *Front Controller v ZF2*

Hlavným cieľom zmeny tejto časti je odstránenie Singletonov. Front Controller v ZF2 je založený na udalost'ami riadenom modeli s využitím EventManageru a prístupuje k objektom pomocou DI. Front Controller sa skladá z čiastkových komponentov, kde každý zohráva kľúčovú úlohu (tab. 7).

Tab. č. 7: Porovnanie čiastkových komponentov Front\_Controlleru v ZF1 a ZF2<sup>58</sup>

ZF1	ZF2
<b>Zend_Controller_Request</b> zapudzuje všetky požiadavky do request objektu. Štandardne sa používa <i>Zend_Controller_Request_Http</i> .	<b>Object request</b> poskytuje prístup k aktuálnemu prostrediu, rovnako ako <i>Zend_Controller_Request</i> . Primárnym rozdielom je zabudovaný prístup injekcie do rôznych „superglobal“ objektov.
<b>Zend_Controller_Response</b> zapudzuje všetky údaje, ktoré sú odosielané späť prehliadaču. Štandardne využíva <i>Zend_Controller_Response_Http</i> .	<b>Response object</b> – predstavuje kontajner pre odpovede na požiadavky, vrátane záhlavia. Primárnou metódou rozhrania je <i>sendOutput()</i> , ktorá sa používa na serializáciu rôznych predmetov smerujúcich do výstupu. Sú primárne realizované cez <i>Renderer</i> .
<b>Zend_Controller_Router</b> zodpovedá za smerovanie. Štandardný smerovač <i>Zend_Controller_Router_Rewrite</i> rozpozná modul, controller, akciu a parametre v URL adresách.	<b>Renderer objekt</b> – hodnotí odozvy a rozhoduje o ich ďalšej činnosti. Používa sa v spojení so <i>Zend_View</i> a <i>Zend_Layout</i> . <b>RouterMatch objekt</b> obsahuje integráciu <i>EventManagera</i> , je rýchlejší a flexibilnejší.
<b>Zend_Controller_Dispatcher</b> volá požadovaný controller.	<b>Dispatcher objekt</b> – určuje ako sa má odoslať žiadosť. Východiskom použitia je určiť modul, controller a akciu z objektu request, potom vytvoríme inštanciu, ktorou vyvoláme príslušnú triedu.

<sup>58</sup>O'PHINNEY, W. M. 2009. *Zend Framework 2.0 Roadmap*. [online]. 2009. [cit. 2012.01.20]. Dostupné na internete: < [http://framework.zend.com/wiki/display/ZFDEV2/Zend\\_Controller+2.0](http://framework.zend.com/wiki/display/ZFDEV2/Zend_Controller+2.0)>.

Tab. č. 8: Porovnanie Front Controller v ZF1 a ZF2<sup>59</sup>

ZF1	ZF2
Inštalácie objektov spravuje pomocou vzoru Singleton, čo väčšinou obmedzuje počet ich vytvorení. Na základe toho môže existovať iba jedna inštancia Front Controller. Metóda konštruktora je definovaná ako protected. Objekty získavame pomocou statickej metódy getInstance().	Objekty získavame prostredníctvom konštruktora DI, kde sa objekt vstrekuje priamo tam, kde je to potrebné.
Štart Front Controlleru pomocou metódy <i>dispatch()</i> , ak bol už nakonfigurovaný v Bootstrap súbore. Poskytuje nám potrebné inštalácie request a response objektov.	Volaním metódy <i>dispatch()</i> controller vytvorí tri udalosti: 1. <b>Routing</b> - definuje modul, controller a udalosť. 2. <b>Dispatching</b> – kontrola volaného modulu controllera, udalosti a volaných akcií. 3. <b>Response</b> – renderovanie view.

### Front Controller v ZF1

```
Zend_Controller_Front::getInstance() ->dispatch();
new Hello_Controller_Request(),
new Hello_Controller_Response()
);
Zend_Controller_Front::getInstance()->dispatch(
$request, $response
);
```

Skript č. 22: Front Controller v ZF1

### Front Controller v ZF2

```
class FrontController implements Dispatchable
{
    public function _construct(DependencyInjection $di)
    {
        $this ->di = $di;
    }
    public function dispatch(Request $request, Response $response = null)
    {
        $controller = $this ->di->get($controllerName);
        $result = $controller->dispatch($request, $response);
    }
}
```

Skript č. 23: Front Controller v ZF2

<sup>59</sup> ZEND TECHNOLOGIES, Inc. 2012. *Programmer's Reference Guide*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: < <http://framework.zend.com/manual/en/zend.controller.action.html>>.

### ➤ *Action controller v ZF1*

Action Controller vykonáva špeciálne úlohy nazývané akcie, ktoré sú mu poskytované priamo Front Controllerom. V ZF každý Action Controller reprezentuje potomka abstraktnej triedy *Zend\_Controller\_Action*, ktorá integruje view, prístup k dátam a poskytuje rôzne pomocné metódy.

### ➤ *Action Controller v ZF2*

Nový Action Controller je založený na podobnom princípe. Poskytuje niekoľko nových metód (tab. 9), kde sú opísané nové funkcie, ktoré prináša ZF2 pri práci s Action a Front Controllerom.

Tab. č. 9: Porovnanie metód Action Controller v ZF1 a ZF2<sup>6061</sup>

ZF1	ZF2
<b>_call()</b> – magická metóda, ktorá slúži k odchyteniu nedefinovaných akcií.	<b>_call()</b> je volaná iba v prípade zistenia mena helpera, ktorý vovylá.
<b>getRequest, getResponse</b> slúžia na získanie prístupu k request a response objektom.	<b>_invoke()</b> slúži na získanie prístupu k objektom.
<b>_getParam(), _getAllParams()</b> slúžia na získanie prístupu k parametrom danej požiadavky.	<b>setResult(), getResult()</b> slúžia na získanie parametra, ktorý je očakávaný v RouteMatch objekte, súčasťou MvcEvent.
<b>getFrontController()</b> slúži na získanie inštancie Front Controlleru.	<b>_get</b> – vracia priamo helper, čím je volanie helperov omnoho jednoduchšie.

### *Action Controller v ZF1*

```
class FooController extends Zend_Controller_Action{
    public function barAction()
    {
        $front = $this ->getFrontController();
        $request = $this ->getRequest();
        $response = $this ->getResponse();
        $root = $this ->getInvokeArg('root');
        $module = $this ->_getParam('controller');
        $action = $this ->_getParam('action');
    }
}
```

Skript č. 24: Action Controller v ZF1

<sup>60</sup>ZEND TECHNOLOGIES, Inc. 2012. *Programmer's Reference Guide*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: < <http://framework.zend.com/manual/en/zend.controller.action.html>>.

<sup>61</sup>O'PHINNEY, M. W. 2009. *Zend\_Controller 2.0 RoadMap*. [online]. 2009. [cit. 2012.02.23]. Dostupné na internete: < [http://framework.zend.com/wiki/display/ZFDEV2/Zend\\_Controller+2.0](http://framework.zend.com/wiki/display/ZFDEV2/Zend_Controller+2.0)>.

## Action Controller v ZF2

```
namespace Zend\Mvc\Controller;
abstract class ActionController implements Dispatchable {

    public function dispatch(Request $request, Response $response = null)
    {
        $this->request = $request;
        if (!$response) {
            $response = new HttpResponse();
        }
        $this->response = $response;

        $e = $this->getEvent();
        $e->setRequest($request)
            ->setResponse($response)
        $result = $this->events()->trigger(MvcEvent::EVENT_DISPATCH, $e,
function($test) {
    return ($test instanceof Response);
});
    }
```

Skript č. 25: Action Controller v ZF2

### ✦ RestfulController

RestfulController prináša nový spôsob odosielenia request a response objektov priamo controlleru vo chvíli, keď je volaná metóda `dispatch()`. Identifikuje HTTP metódy a na základe toho, voláme požadovanú metódu. RestfulController implementuje funkcie **Create**, **Read**, **Update**, **Delete** (skrát. **CRUD**), ktoré umožňujú vytvorenie, načítanie, zmenu a vymazanie dát, poskytuje ich Action Controlleru (tab. 10).

Tab. č. 10: Metódy Restful Controller<sup>62</sup>

<b>POST</b>	Volá metódu <i>create()</i> , ktorá prechádza hodnoty v <code>\$_POST</code> .
<b>GET</b>	Volá metódu <i>get()</i> , ktorá identifikuje „ID“ počas smerovania.
<b>PUT</b>	Volá metódu <i>update()</i> , ktorá vypisuje všetky údaje z „ID“.
<b>DELETE</b>	Volá metódu <i>delete()</i> , ktorá čaká na uzavretie „ID“ počas smerovania.

<sup>62</sup>ZEND TECHNOLOGIES, Inc. 2012. *Programmer's Reference Guide*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://packages.zendframework.com/docs/latest/manual/en/zend.mvc.controllers.html>>.

## 4.5 Vyhodnotenie naplnenia cieľov

Vyhodnotením prínosov ZF2 podľa troch kľúčových kritérií, ktoré boli uvedené v kapitole 3 a na základe analýzy výsledkov komparácie, sme dospeli k nasledujúcim výsledkom.

Z hľadiska *flexibility* najväčšiu zmenu predstavuje vrstva MVC, vďaka čomu je možné vytvárať aplikácie oveľa robustnejším, rýchlejším a flexibilnejším spôsobom. Tieto zmeny mali dopad najmä na View vrstvu, ktorá bola kompletne prepracovaná s cieľom pracovať samostatne, nezávisle na modely či controlleroch, s čím je spojená možnosť lepšej implementácie.

Rozšíriteľnosť a flexibilita je tiež spojená s odstráneným silným vnútorným závislostí, vytvorením nových rozhraní, prístupujúcich ku komponentom prostredníctvom abstraktných tried a elimináciou Singletonov. Vďaka týmto zmenám sú aplikácie jednoduchšie modularizované a jednotlivé komponenty sú ľahšie použiteľné. Pri tvorbe aplikácií nie je nutné sa zameriavať na všeobecné úlohy, ale na kľúčovú aplikačnú logiku. Významnú úlohu v tomto procese zohrávajú komponenty *Zend\Di*, ktoré odstraňujú vnútorné závislosti a *Zend\MVC* je založený na princípe abstraktného rozhrania *Dispatchable*, ktoré využíva nový princíp „design by contract“.

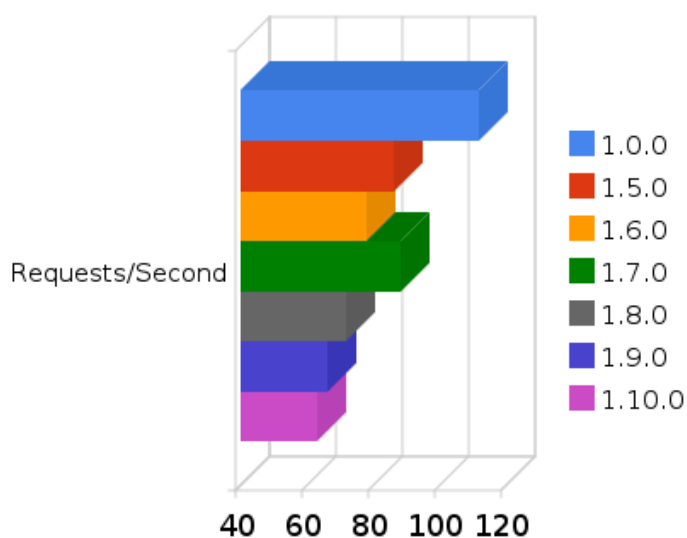
Údržba frameworku je jednoduchšia na základe využívania izolovaných tried, ktoré poskytuje nová modulárna architektúra a systém modulov, založené na princípe menných priestorov, ktoré poskytuje jazyk PHP 5.3.

Jediným nedostatkom, ktorý môžeme vytknúť, je skutočnosť, že hoci je modulárny, chýba mu niekoľko základných funkcií. Neobsahuje žiadny komponent alebo konfiguračný mechanizmus na riadenie závislostí medzi modelom a controllerom. Tiež *Zend\_Controller* neumožňuje načítanie súborov z modelu v rámci svojho modulu, rovnako ako mimo neho.

Krivka učenia a čas potrebný na naučenie je výrazne kratší, vďaka lepšie prepracovanej *dokumentácii*, na základe skúsenosti s používaním frameworku v predchádzajúcich verziách. Je výrazne jasnejšia, prehľadnejšia a ľahšie čitateľná. Obsahuje viac príkladov a tutoriálov, lepšie prepracovanú referenčnú príručku a konzistentnejšia API dokumentácia dodržiava

štandardný formát. Je tu podrobnejší popis jednotlivých komponentov pre lepšie pochopenie a s ním spojený popis možností použitia. Je vypracovaná vo viacerých jazykoch.

Cieľom bolo *zvýšenie výkonu* oproti predchádzajúcej verzii. S každou novou verziou sa výkon postupne však znižuje (obr. 6), kde je znázornený počet spracovaných požiadaviek za sekundu. Ich počet sa postupne znižoval.



Obr. č. 6: Čas spracovania žiadosti<sup>63</sup>

Porovnali sme ZF1, konkrétne verzie ZF1.11. a ZF2 (tab 11), pomocou porovnávajúceho nástroja Apache HTTP server, kde je vyhodnotený počet požiadaviek za sekundu a taktiež spracovanie 155 požiadaviek súbežne. Podľa výsledkov testov je ZF2 je 4 krát pomalší, než ZF1. Ale nemôžeme zabudnúť na to, že je stále vo verzii Beta, ktorá nie je dostatočne optimalizovaná. Obsahuje však ClassMap a nový spôsob automatického načítavania, v tomto smere poskytuje oveľa viac než ZF1. Prevažná časť programu je spustiteľná v súčasnej verzii v DIC, ktorý pracuje s mnohými knižnicami, čo znižuje výkon, ale momentálne je prioritou kladená najmä na zdokonaľovanie funkcií. Vo finálnej verzii budú niektoré komponenty presunuté, čo zoptimalizuje výkon.

<sup>63</sup>O'PHINNEY, W. M., SCHINDLER, R. 2010. *Introducing Zend Framework 2.0*. [online]. SlideShare Inc. 2010. [cit. 2012.01.20]. Dostupné na internete: < <http://www.slideshare.net/weierophinney/introducing-zend-framework-20>>.



Tab. č. 11: Porovnanie rýchlosti ZF1 a ZF2<sup>64</sup>

Podmienky	ZF1	ZF2
Čas v sekundách	1.197 s	11.091 s
Žiadosť/sek.	129.53	13.98
Čas vybavenia požiadaviek, súbežne(155)	7.720 ms	11091.008 ms

Benchmark results of ZF 1.11.11 WITH APC

Benchmark results of ZF 2 WITH APC

<sup>64</sup>IVALDI, P. 2012. *Benchmark of some popular Web frameworks*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: < <http://www.piprime.fr/1541/benchmark-of-some-popular-web-frameworks/>>.

#### *4.5.1 Vlastné odporúčanie prechodu na novú verziu*

Firma Zend poskytuje mnoho nástrojov, ktoré umožňujú prechod na novú verziu. Prechod z existujúcej PHP aplikácie môže byť niekedy skľučujúcou úlohou.

Je lepšie zvoliť si plynulý prechod s postupným oboznamovaním sa s novými funkciami a komponentmi. Z dôvodu zmeny celej architektúry založenej na systéme modulov a zavedením menných priestorov, je nutné prepísať celý pôvodný kód. Preto ak začíname teraz s novým projektom, je vhodné používať už nové funkcie a možnosti ZF2. Verzia Beta 3 nie je ešte vhodná pre produkčné nasadenie, firma Zend vydáva momentálne novú mini verziu ZF 1.12, kde budú mnohé funkcie „backportované“ zo ZF2. Príkladom je napríklad Autoloader a Event Manager, takže je možné sa s nimi zoznámiť bližšie a využívať ich výhody už teraz.

## Záver

*Bezradnosť a nespokojnosť sú prvými predpokladmi k pokroku.*

**Thomas Alva Edison, významný vedec**

Ukázali sme si, že Zend Framework je všeobecne uznávaným frameworkom vo svete PHP a silným nástrojom pre tvorbu webových aplikácií. Verzia ZF2 prináša mnohé vylepšenia, ktoré uľahčia našu prácu. Môžeme skonštatovať, že ZF2 predstavuje významný evolučný pokrok, prináša nové prístupy, nové funkcie a komponenty, ktoré sú súčasťou tejto distribúcie. Rozširujú funkcionality ZF, vďaka čomu je možné predstihnúť aj konkurenciu.

Naplnením hlavného cieľa bakalárskej práce bola vykonaná komparácia a analýza Zend Frameworku 2.0., so zameraním na odhalenie rozdielov, či spoločných znakov s verziou ZF 1.X, na základe ktorých je možné zhodnotiť celkové prínosy. V tejto práci bola opísaná naposledy vydaná Beta 3 verzia. Naplnenie hlavného cieľa bolo podmienené postupným napĺňaním čiastkových cieľov.

Úvod práce bol venovaný teoretickej analýze PHP frameworkov a Zend Frameworku. Boli prezentované základné kritériá, ktoré určujú výber správneho frameworku, využité na porovnanie Zend Frameworku s CodeIgniter. Výsledok komparácie poukázal na prednosti Zend Frameworku. Bližšie boli špecifikované nedostatky ZF, ktoré negatívne ovplyvňujú a znižujú jeho flexibilitu a výkon. V práci bolo prezentovaných niekoľko objektívnych kritérií vyhodnocovania komparácie. Rozhodujúcimi boli najmä nová modulárna MVC architektúra a nový systém, založený na moduloch. Pri vyhodnocovaní celkovej analýzy nových komponentov a funkcií sa celková flexibilita a možnosti zvýšili, vďaka možnosti plného využitia nových funkcií jazyka PHP 5.3 a PHP 5.4.

Subjektívna komparácia viedla k celkovým výsledkom, kde je zo vzájomného porovnania evidentné, že ZF2 je konzistentnejším, flexibilnejším a modulárnejším frameworkom. Problém s výkonom nebol vyriešený podľa očakávaní, ale aspoň zoptimalizovaný. ZF2 prináša lepšie prepracovanú dokumentáciu, je možné ľahšie a rýchlejšie jednotlivé nové komponenty pochopiť a aplikovať ich do projektov. Vzhľadom na fakt, že komparácia bola uskutočnená na ešte neoficiálnej Beta verzii, konečná hlavná verzia môže

priniesť ešte mnoho nových zmien. Okrem základnej funkcionality, ktorú sme ukázali v tejto práci, obsahuje množstvo ďalších veľmi silných a účinných nástrojov.

Dôležité je nepretržité sledovanie vývoja ZF, ktorý prechádza mnohými zmenami, čo nás posúva smerom k ďalšiemu rozvoju a zdokonaľovaniu. ZF2 Beta 3 poskytuje nové možnosti riešenia a spôsoby implementácie a môžeme sa tešiť na finálnu verziu, ktorá nás isto prekvapí mnohými inováciami a funkciami, umožní zodpovedne porovnať a vyhodnotiť celkové prínosy Zend Frameworku 2.0.

## Zoznam použitej literatúry

### Knihy / Monografie

1. **ALLEN, R., LO, N.** 2009. *Zend Framework in Action*. United States of America : Manning Publications, 2009. 199 s. ISBN 978-1933988320.
2. **BÖHMER, M.** 2010. *Zend Framework : Programujeme webové aplikace v PHP*. Brno : CPress, 2010. 416 s. ISBN 978-80-251-2965-4.
3. **COGGESHALL, J., TOCKER, M.** 2009. *Zend Enterprise PHP Patterns*. United States Of America : Apress, 2009. 282 s. ISBN 978-1-4302-1975-0.
4. **EVANS, C.** 2008. *php/architect's Guide to Programming with Zend Framework*. United States of America : Marco Tabini & Associates, Inc., 2008. 222 s. ISBN 978-0-9738621-5-7.
5. **GILMORE, J. W.** 2009. *Easy PHP Websites with the Zend Framework*. Columbus : W.J. Gilmore, LLC, 2009. 342 s. ISBN 0615303889.
6. **MCARTHUR, K.** 2008. *Pro PHP Patterns, Frameworks, Testing and More*. United States of America : Apress, 2008. 375 s. ISBN 978-1-59059-819-1.
7. **PADDILA, A.** 2009. *Beginning Zend Framework*. United States of America : Apress, 2009. 424 s. ISBN 978-1-4302-1825-8.
8. **POPE, K.** 2009. *Zend Framework 1.8 Web Application Development*. Birmingham : Packt Publishing, 2009. 379 s. ISBN 978-1-847194-22-0.

9. **POREBSKI, B. – PRZYSKALSKI, K. – NOWAK, L.** 2011. *Building PHP Applications with Symfony, CakePHP, and Zend Framework*. Indianapolis : Wrox, 2011. 1235 s. ISBN 978-0-470-88734-9.
10. **VASWANI, V.** 2010. *Zend Framework, A Beginner's Guide*. United States of America : McGraw-Hill Osborne Media, 2010. 465 s. ISBN 978-0-07-163940-8.

### Články v elektronických časopisoch a iné príspevky

1. **KOPRDA, M.** 2010. Porovnanie frameworkov: CakePHP vs. CodeIgniter vs. Kohana vs. Nette vs. Yii vs. Zend in *Zajtra.sk* [online]. Bratislava : 2010. [cit. 2012.04.20]. Dostupné na internete: <<http://www.zajtra.sk/programovanie/87/porovnanieframeworkov-cakephp-vs-codeigniter-vs-kohana-vs-nette-vs-yii-vs-zend>>.
2. **ZEND TECHNOLOGIES, Inc.** 2012. Zend Framework 2.0 Beta 3 Release Gives Developers an Early Start on PHP App Development In *MC PRESS* . [online]. 2012, no.3 [cit. 2012.02.02]. Dostupné na internete: <<http://www.mcpressonline.com/programming-languages/zend-framework-20-beta-3-release-gives-developers-an-early-start-on-php-app-development.html>>.

### Elektronické dokumenty – monografie

1. **ALLAN, R.** 2012. *An introduction to Zend\EventManager*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://akrabat.com/zend-framework-2/an-introduction-to-zendeventmanager/>>.
2. **BERNARD, B.** 2009. *Úvod do architektury MVC*. [online]. Praha : Devel.cz Lab s.r.o., 2009. [cit. 2012.01.19]. Dostupné na internete: <<http://www.zdrojak.cz/clanky/uvod-do-architektury-mvc/>>. ISSN 1803-5620.
3. **BRADY, P.** 2008. *The Zend Framework, Dependency Injection and Zend\_Di*. [online]. 2008. [cit. 2012.01.23]. Dostupné na internete: <<http://blog.astrumfutura.com/2008/02/the-zend-framework-dependency-injection-and-zend-di/>>.

4. **DANEK, P.** 2008. *Velký test PHP frameworků: Zend, Nette, PHP a RoR*. [online]. Praha : Internet Info, s.r.o., 2008. [cit. 2012.01.20 ] Dostupné na internete: <<http://www.root.cz/clanky/velky-test-php-frameworku-2-dil/>>. ISSN 1212-8309.
5. **DROZD, M.** 2009. *PHP frameworky na konferencii Zend/PHP*. [online]. Bratislava : Bloog.sk., 2009. [cit. 2012.01.13]. Dostupné na internete: <<http://www.bloog.sk/?s=zend+framework>>.
6. **FELTON, D.** 2010. *Zend Framework Version Lifecycle*. [online]. 2010. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV/Zend+Framework+Version+Lifecycle>>.
7. **IVALDI, P.** 2012. *Benchmark of some popular Web frameworks*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://www.piprime.fr/1541/benchmark-of-some-popular-web-frameworks/>>.
8. **KERNER, M. S.** 2011. *PHP 5.4 and Zend Framework 2.0 Gearing up for Release*. [online]. QuinStreet Inc., 2011. [cit. 2012.02.02]. Dostupné na internete: <http://www.developer.com/lang/php/php-5-4-and-zend-framework-2-0-gearing-up-for-release.html%20>>.
9. **O'PHINEY, W. M.** 2008. *Zendframeworkworkshop*. [online]. Slideshares. Inc. 2008. [cit. 2012.01.23]. Dostupné na internete: <<http://static.slideshare.net/swf/ssplayer2.swf?doc=20080613zendframeworkworkshop-1213725963433006-9/>>.
10. **O'PHINNEY, W. M.** 2009. *Zend\_Controller 2.0 RoadMap*. [online]. 2009. [cit. 2012.02.23]. Dostupné na internete: <[http://framework.zend.com/wiki/display/ZFDEV/2/Zend\\_Controller+2.0](http://framework.zend.com/wiki/display/ZFDEV/2/Zend_Controller+2.0)>.

11. **O'PHINNEY, W. M.** 2011. *Proposal For Autoloading In ZF2*. [online]. 2011. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV2/Proposal+For+Autoloading+In+ZF2>>.
12. **O'PHINNEY, W. M.** 2011. *Zend Framework 2.0 Roadmap*. [online]. 2011. [cit. 2012.01.20]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV2/Zend+Framework+2.0+Roadmap>>.
13. **O'PHINNEY, W. M., SCHINDLER, R.** 2010. *Introducing Zend Framework 2.0*. [online]. SlideShare Inc. 2010. [cit. 2012.01.20]. Dostupné na internete: <<http://www.slideshare.net/weierophinney/introducing-zend-framework-20>>.
14. **O'PHINNEY, W.M.** 2012. *View Layers, Database Abstraction, Configuration, Oh, My!* [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://mwop.net/blog/zf2-beta3.html>>.
15. **ŠALTYS, Ž.** 2007. *Zend Framework pros and cons*. [online]. 2007. [cit. 2012. 01. 13]. Dostupné na internete: <<http://www.thedeveloperday.com/zend-framework-pros-and-cons/>>.
16. **SCHINDLER, R.** 2010. *Zend Db 2.0 Requirements*. [online]. 2010. [cit. 2012.02.23]. Dostupné na internete: <<http://framework.zend.com/wiki/display/ZFDEV2/Zend+Db+2.0+Requirements>>.
17. **SCHINDLER, R.** 2012. *Zend\Db in ZF 2.0*. [online]. 2012. [cit. 2012.04.23]. Dostupné na internete: <<http://static.zend.com/topics/ZendDb-2.pdf>>.
18. **ZEND TECHNOLOGIES, Inc.** 2012. *Programmer's Reference Guide*. [online]. 2012.[cit. 2012.04.23]. Dostupné na internete: <<http://packages.zendframework.com/docs/latest/manual/en/zend.mvc.html>>.



19. **ZEND TECHNOLOGIES**, Ltd. 2012. *Zend Framework by the Numbers*. [online]. Zend Technologies Ltd. 2012. [cit. 2012.01.20]. Dostupné na internete: <<http://zendframework.com/about/numbers>>.
20. **ZIMUEL, E.** 2012. *Zend Framework 2.0 quick start*. [online]. Moscow : SlideShare Inc. 2012. [cit. 2012.04.20]. Dostupné na internete:<<http://www.slideshare.net/e.zimuel/zend-framework-2-quick-start>>.
21. 2011. *Most Used PHP Framework-The Popular Top 7 List in year 2011*. [online]. 2011. [cit. 2012.04.20]. Dostupné na internete: <<http://www.php-developer.org/most-used-php-framework-the-popular-top-7-list-in-year-2011/>>.