

POSUDEK OPONENTA DIPLOMOVÉ PRÁCE

Jméno diplomanta: Bc. Jiří Staniševský

Název práce: Komponentový framework pro vývoj webových aplikací

Hledisko 1: náročnost práce

Náročnost tématu na teoretické znalosti	☒ vysoká	☐ průměrná	☐ nižší
Náročnost tématu na praktické zkušenosti	☒ vysoká	☐ průměrná	☐ nižší
Náročnost tématu na rozsah a úroveň podkladových materiálů	☒ vysoká	☐ průměrná	☐ nižší

Hledisko 2: splnění věcných požadavků

Stupeň splnění cíle práce	☐ vynikající	☒ solidní	☐ vyhovující	☐ nevyhovující
Hloubka provedené analýzy ve vztahu k tématu	☐ vynikající	☐ solidní	☒ vyhovující	☐ nevyhovující
Aktuálnost uvedených údajů	☐ vynikající	☐ solidní	☒ vyhovující	☐ nevyhovující
Adekvátnost použitých metod (pramenů)	☐ vynikající	☐ solidní	☒ vyhovující	☐ nevyhovující
Vlastní přínos k řešené problematice	☐ vynikající	☐ solidní	☐ vyhovující	☐ nevyhovující
Stupeň realizace navrhovaných řešení (programy ap.)	☐ vynikající	☒ solidní	☐ vyhovující	☐ nevyhovující
Využitelnost výsledků práce v praxi (teorii)	☐ vysoká	☐ průměrná	☐ nižší	

Hledisko 3: formální náležitosti

Logická stavba práce	☐ vynikající	☐ solidní	☒ vyhovující	☐ nevyhovující
Formální bezchybnost textu	☐ vynikající	☒ solidní	☐ vyhovující	☐ nevyhovující
Grafická úprava práce	☒ vynikající	☐ solidní	☐ vyhovující	☐ nevyhovující
Adekvátnost a srozumitelnost závěrů	☐ vynikající	☒ solidní	☐ vyhovující	☐ nevyhovující
Práce s literaturou (citace)	☒ vynikající	☐ solidní	☐ vyhovující	☐ nevyhovující

Připomínky a otázky vyžadující doplnění:¹

Proč není vhodné soubory v PHP ukončovat ?>

Jak budete řešit SEO Url a jak by jste řešil custom stránku pro chybu 404, 500 apod.?

¹ Podle potřeby rozepište na dalších stranách.

Práci k doporučení obhajobě.

Navržená výsledná známka: dobře

V Praze

dne 20.5.2011

Jméno: Ing.Ladislav Prskavec

Podpis:

Poznámky k hodnocení:

Autor si vzal náročné téma, které opravdu není jednoduché a vyžaduje velké množství praktických zkušeností, které autorovi evidentně chybí. PHP je vcelku sice jednoduchý jazyk, ale pracovat s ním dobře už tak jednoduché není.

Za klad práce považuji celkem dobře zpracované formuláře ve frameworku a oddělené views pro jednotlivé zařízení a vlastní kód přiložený k práci fungoval bez problémů. Celková úroveň tiskového zpracování práce a citované literatury je velmi dobrá.

Protože na vývoj frameworků je názorů mnoho, budu zde reprezentovat ten svůj opřený o svoji 10 let trvající praxi jako PHP programátor. Od 30.6.2009 je k dispozici PHP 5.3 a framework by dnes měl být psaný v něm. Mnoho výhod jazyka je dostupných až ve verzi 5.3 (namespaces, late static bindings, native closures, apod.). Obzvláště je to znát pokud píšete dost unit testy, které mi přijdou při vývoji dnes nezbytné.

V teoretické části mi chybí kvalitnější rozbor frameworků s ohledem na softwarové metriky, které umožňují zajímavý pohled na hodnocení. Velmi dobře se dá použít nástroj PHP Depend, který má implementovány například Average Hierarchy Height, Average Number of Derived Classes, Afferent Coupling, Cyclomatic Complexity Number, NPath Complexity atd. Sam jsem před pár lety hodnotil těmito metrikami několik frameworků (<http://blog.prskavec.net/2009/03/pdepend-a-php-frameworky/>).

Osobně bych hodnocení frameworku spíše viděl na této bázi:

- srozumitelnost kódu
- pokrytí testu (phpunit)
- kvalita api dokumentace
- kvalita uživatelské dokumentace
- komunita (irc, google groups apod.)
- možnost přispívání do kódu, opravy chyb (například github.com pro dost projektů velmi usnadňuje tyto možnosti)

Prakticky si myslím, že pro rozsah práce je realizovatelný jen nějaký microframework jako je například Silex (<http://silex-project.org/>), který vychází z komponent Symfony2 a nesnaží se realizovat všechno, inspirace frameworkem Sinatra z Ruby je v něm zřejmá.

Ztrácet čas realizací databázové vrstvy na bázi Active Record mi přijde opravdu zbytečné. Použitelných implementací je dost. Ale pro vlastní realizaci bych řešil spíše na bázi DataMapper. Chyběl mi ve frameworku řešený autoloading (lazy load), konfigurace mimo framework pro aplikaci, jednoduše editovatelná například v xml nebo ini souboru. Autor vůbec neřešil routing, custom error pages a seo url, které se obvykle řeší přes mod_rewrite. Tato metoda se mu nelíbí, ale žádnou alternativu nám nedává a jeho řešení tyto základní funkce neumí.

Sekce o cache je taky velmi povrchní chybí úrovně řešení a rozebrání frontend cache (varnish, squid), aplikační cache (memcache) a vlastní cache PHP (APC, eAccelerator). Autor se zabývá jen poslední z nich a dnes, když je APC přímo v jádru už se moc ostatní alternativy nepoužívají. Horizontální a vertikální škálování aplikací může být delší práci, ale není na škodu se v teoretické části věnovat.