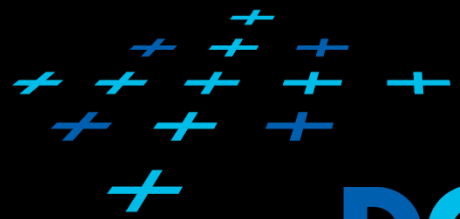


DCGI

KATEDRA POČÍTAČOVÉ GRAFIKY A INTERAKCE

Zend Framework

Frameworky

- K čemu slouží:
  - jednoduchý vývoj webových aplikací
  - spousta práce je už hotová
  - využití práce jiných lidí
- Klady:
  - rychlý vývoj
  - velká komunita
  - dobrá dokumentace
- Zápory:
  - nedělám si věci posvém
  - někdy kanon na vrabce

Přehled frameworků (http://www.phpframeworks.com/)

- Zend Framework
- Nette
- Prado
- CakePHP
- Akelos
- …
- …

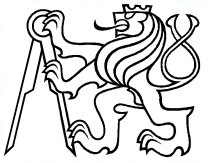
Co typicky řeší

- OOP
- MVC
- Přístup k DB (abstrakce)
- Obsluha formulářů
- ORM
- Template
- Ajax
- a spoustu dalších věcí

Dnešní téma: Zend Framework

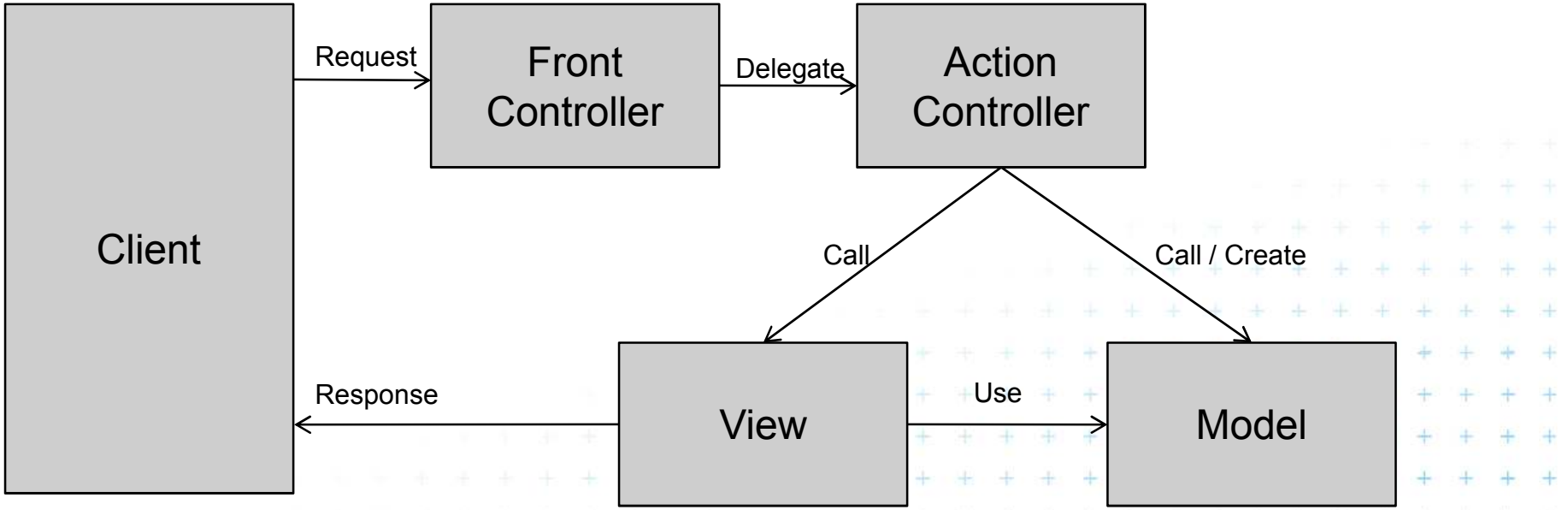
- Modulární
- Čistě PHP
- PHP 5
- Zdarma (New BSD licence = dělej si s tím co chceš)
- Velmi populární
- Podporují ho silní partneři (Zend, IBM, OmniTI)

- framework.zend.com

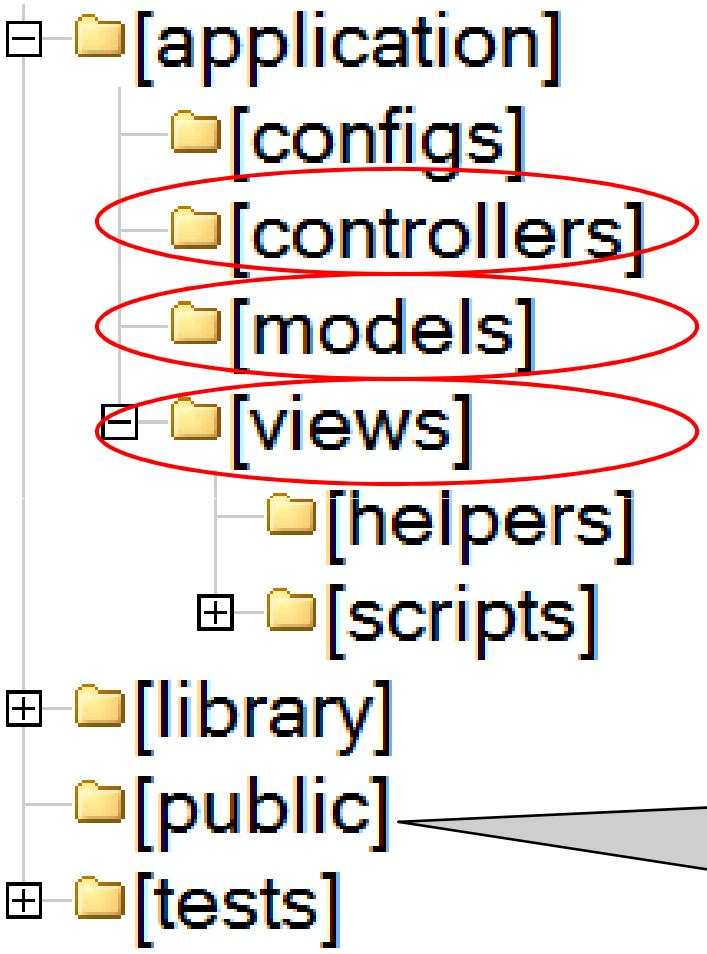


Základní principy

- Adresářová struktura odpovídá jmenným prostorům => dobrá orientace v názvech tříd
- Implementace Front Controller



Adresářová struktura




```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} -s [OR]
RewriteCond %{REQUEST_FILENAME} -l [OR]
RewriteCond %{REQUEST_FILENAME} -d
RewriteRule ^.*$ - [NC,L]
RewriteRule ^.*$ index.php [NC,L]
```

Root webu
pozor, zde musí být fungovat mod_rewrite .htaccess například

Zde je také soubor index.php

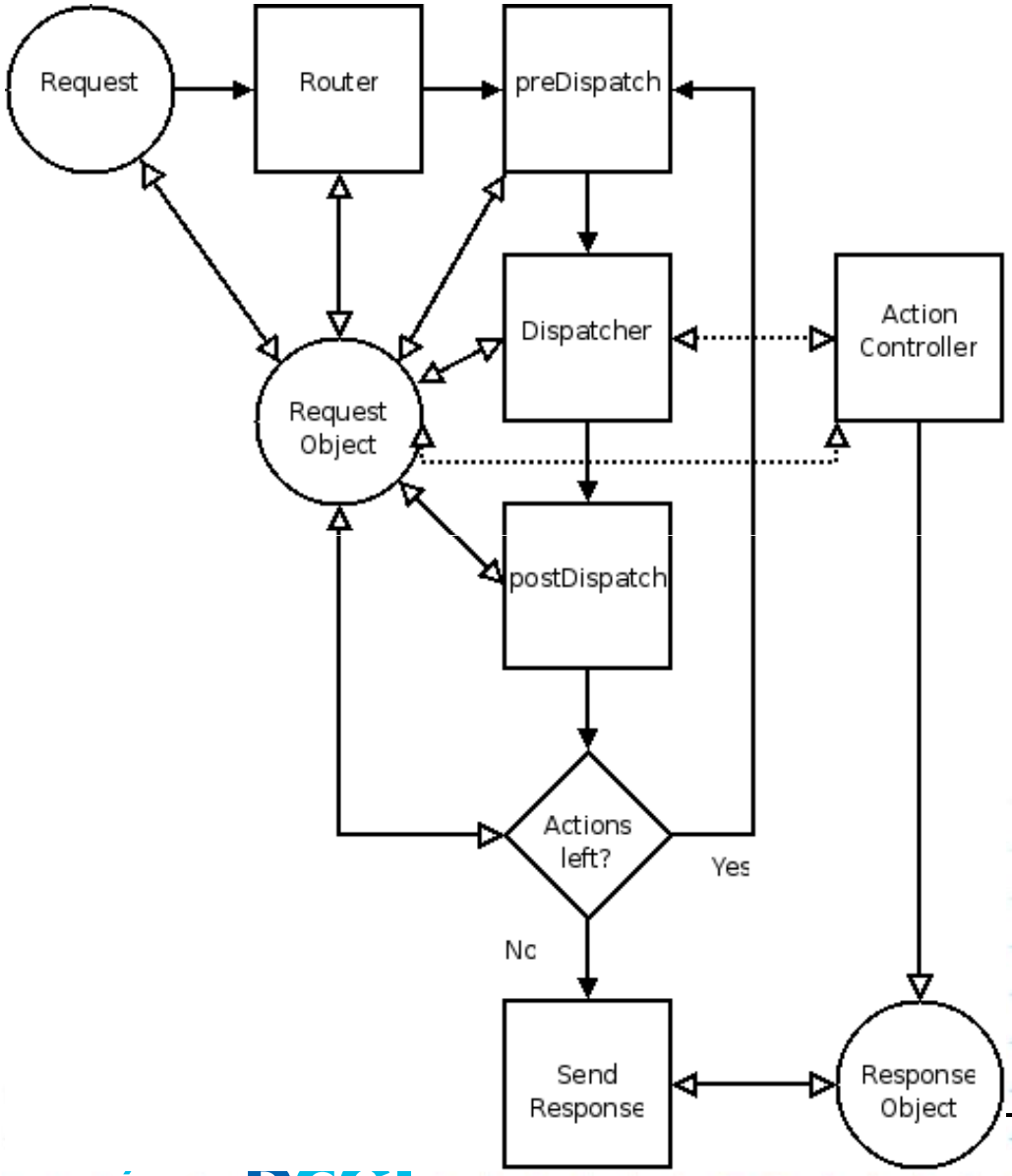
Bootstrap

- Přes tento soubor jde kompletně vše

```
/** Zend_Application */
require_once 'Zend/Application.php';

// Create application, bootstrap, and run
$application = new Zend_Application(
    APPLICATION_ENV,
    APPLICATION_PATH . '/configs/application.ini'
);
$application->bootstrap()
            ->run();
```

Front Controller

Zpracování dotazu a implementace MVC



Request – je dále reprezentován Request objektem

Router – přeloží URL na posloupnost akcí, které se mají provést

Dispatcher – na základě znalosti jména controlleru a akcí volá daný controller

Action Controller – pracuje s Request a Response objekty a může volat další akci

Zdroj: http://framework.zend.com/manual/en/zend.controller.basics.html

Request Object

- Instance třídy Zend_Controller_Request_Http
- Metoda getParam(jmeno) reprezentuje superglobální proměnné v následujícím přepisovacím pořadí:
  - GET, POST, COOKIE, SERVER, ENV
- Metoda setParam(jmeno, hodnota) nastavuje uživatelem definované parametry
- Pokud si potřebujeme sáhnout na konkrétní GET nebo POST proměnné, můžeme přes getPost(jmeno) a getQuery(jmeno)

Routování

- URL mapování na volání controlleru a jeho akce s nějakými parametry
- Lze definovat vlastní pravidla
  - obrovská volnost, viz http://framework.zend.com/manual/en/zend.controller.router.html
- Default chování přepisuje URL takto:
  - controller/action/var1/value1/var2/value2

Dispatcher

- Rozesílá Request objekt těm kontrolerům, které byly identifikovány v routovací fázi
- Každý kontroler může mít několik akcí
  - např. zobraz, ulož, …
- Na začátku nastaví příznak u Request objektu informující o tom, že byl *dispatched*.
  - Pokud někdo v pre a post a dispatch fázi znovunastaví tento příznak, bude Request znovu „dispatchnut“. Tím lze nastavovat posloupnost Requestů, které se provedou.
  - metoda _forward() posílá dotaz do nové akce
    - tím mohu implementovat například posloupnost akcí „ulož“ a „zobraz“

Action Controller

- Třída rozšiřující Zend_Controller_Action
- jmémo metody je vázáno na jméno akce z URL
- metoda init() může dělat inicializaci pro všechny akce
- metody preDispatch() a postDispatch() jsou volány před a po akci.
  - v preDispatch() můžeme například akci přeskočit pomocí _forward() pokud nejsme přihlášeni
- Automatická injekce view se stejným názvem jako je action. View je dostupné pod $this->view

Controller - View

- View je zavolána po volání metody render
  - string render(string $action = null,
    string $name = null,
    bool $noController = false);
  - pokud neřekneme jinak, je voláno view daného kontroleru a dané akce
- Mapování:

  views/scripts/[controller]/[action].phtml
  - pozor: znaky . a _ jsou převedeny na znak –

Controller – forward a redirect

- Metody pro předání na jinou action
- _forward předá v rámci řetězce dispatch
- _redirect odešle hlavičku s kódem typicky 302

Response objekt

- normálně ho nemusíme používat
- controller do něj zapisuje
- my můžeme využívat např. metod
  - setHeader, setRedirect, getHeaders, cleanHeaders ,…

Zend_Form

- Objektová podpora pro tvorbu formulářů
- Formulář je reprezentován objektem, který dědí z třídy Zend_Form
- Každá formulářová položka má svojí třídu
  - Zend_Form_Element_Text,
  - Zend_Form_Element_Textarea
  - Zend_Form_Element_...
- Položky mají API odpovídající jejich funkčnosti

```php
class Forms_Address extends Zend_Form {

    public function init() {
        $this->setMethod('post');

        $nameElement = new Zend_Form_Element_Text("jmeno");

        $nameElement->setLabel('jmeno');
        $nameElement->setValue('jmeno');

        $this->addElement($nameElement);
    }
}
```

Kontroler využívá formulář

- AddressController, newAction

```
public function newAction() {
        $request = $this->getRequest();
        $form = new Forms_Address();

        // prirazeni vlastnosti k view
        $this->view->form = $form;

        if ($request->isPost()) {
            if ($form->isValid($request->getPost())) {
                // udelame něco s daty a podekujeme
                return $this->_helper->redirector('thankyou');
            }
        }
    }
```

Zavolá akci thankyou

View

- V nejjednodušším případě necháme formulář renderovat jeho vlastními schopnotmi

view

```
<?php

echo $this->form ;

?>
```


Validace

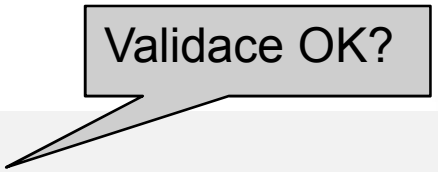
- Ověření správnosti vstupních dat
- Řada existujících validátorů
- alnum, alpha, barcode, between, callback, ccnum, date, digits, …, regex, …
- Možnost napsat validátor vlastní

Validace

- ve formuláři přidám k danému elementu validátor nebo podmínku existence hodnoty

```
// pouziti validatoru
$nameElement->setRequired();
$emailElement->addValidator(new Zend_Validate_EmailAddress());
```

- v kontroleru se ptám, zda vyplněný formulář je validní




```
if ($request->isPost()) {
            if ($form->isValid($request->getPost())) {
              return $this->_helper->redirector('thankyou');
        }
}
```

Psaní vlastních validátorů

- validátor je třída rozšiřující třídu Zend_Validate_Abstract
- metoda isValid($value, $context=null) vrací true/false podle situace
- V poli $_messageTemplates uchovává seznam chybových hlášek
- Metodou _error($klic) přidáme chybovou hlášku do pole výsledných chyb.

Implementace validátoru

library/myvalidators/Validname.php

```
class Myvalidator_Validname extends Zend_Validate_Abstract  {

        protected $_messageTemplates = array(
        "wrong_name" => "Jméno '%value%' není ani Martin ani
Petr."
    );

         public function isValid($value, $context = null){
               $this->_setValue($value);
               if ($value !== "Martin" && $value !== "Petr") {

                     $this->_error("wrong_name");
                     return false;
               } else {
                     return true;
               }
return false;
         }
}
```

Použití vlastního validátoru ve formuláři

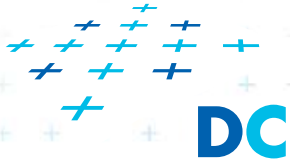
```
$nameElement->addPrefixPath("Myvalidator",
"myvalidators/", 'validate');

$nameElement->addValidator('validname');
```

Prefix namespace

Cesta ke třídě

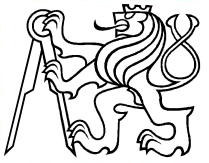
Jedná se o validátor

Použití validátoru na konkrétním elementu



i18n = internationalization

- Psaní kódu tak, aby bylo možné ho přeložit do jiných jazyků.
- Několik variant obvykle ve tvaru klíč – hodnota
- V Zend podpora konfiguračních souborů ve tvaru
  - Array, CSV, Gettext, Ini, Tbx, Tmx, Qt, Xliff, XmlTm, …
- Každá jazyková varianta je identifikovaná pomocí zkratky locale
  - en, cs, de, …
  - locale může mít i podtyp např. de_AT, de_DE, de_CH



Překladače

- Zend_Translate

Použití:

1. nastavíme překladové slovníky, v tomto případě typ array

```
$translate = new Zend_Translate('array',
'../library/languages/addressform_cs.php','cs');
$translate->addTranslation(
'../library/languages/addressform_en.php','en');
$translate->setLocale('en');
```

2. jazykově závislé hlášky nahradíme konstrukcí

```
$nameElement->setLabel($translate->_('name'));
```

…Zend je ještě mnohem více, nastuduj si sám(a)

DĚKUJI ZA POZORNOST