Smile
OPEN SOURCE SOLUTIONS

Formation Zend Framework

Montpellier le 18 juin 2012

Formateur: Olivier Chabert

Smile
OPEN SOURCE SOLUTIONS

Plan

1 - Zend Framework & MVC Introduction

2 - Organisation d'un projet avec Zend Framework

3 - Contrôleurs, Vues et Layouts

4 - Modèles et base de données

5 - Les formulaires

6 - Authentification

7 - Registre et logs

Smile
OPEN SOURCE SOLUTIONS

Mise en pratique

Application de gestion de produits "forsale"

- Application de gestion de produits
  - doit permettre:
    - la création
    - l'édition
    - la suppression
    - le listing

  - Les produits possèdent un nom, une description et un prix

Smile
OPEN SOURCE SOLUTIONS

Zend Framework & MVC Introduction

Smile
OPEN SOURCE SOLUTIONS

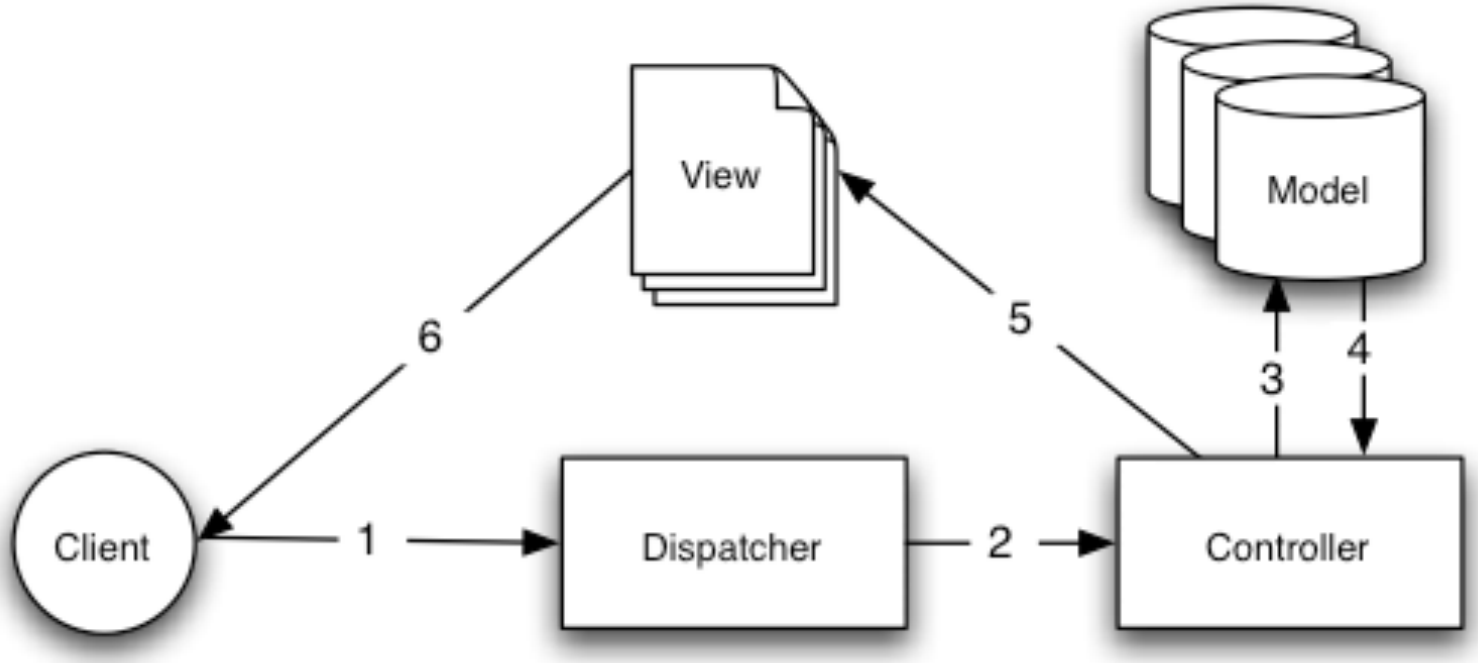
Introduction au Zend Framework (1/2)

- Framework:
  - bibliothèque générique de composants
  - guide l'architecture logiciel du projet
- Zend Framework créer en mars 2006 par Zend Technologie
- Arrivée du composant Zend_Application dans la version 1.8
- Avantages:
  - pas d'obligation d'utiliser le framework complet
  - on peut utiliser seulement certain composants
- Inconvénients:
  - plus de travail pour connaitre et relier les différents composants
  - moins de contraintes sur l'application des bonnes pratiques

Introduction au Zend Framework (2/2)

- Nécessite PHP 5.2.4 minimum
- Utilisation du mod_rewrite Apache pour les friendly URLs
  - meilleur référencement
  - URLs porteuses de sens pour l'utilisateur
- Framework de test: phpUnit 3.3.0
- Plus d'informations: http://framework.zend.com
- Documentation officelle: http://framework.zend.com/manual/en/

Le pattern d'architecture MVC avec Front Controller



Front Controller

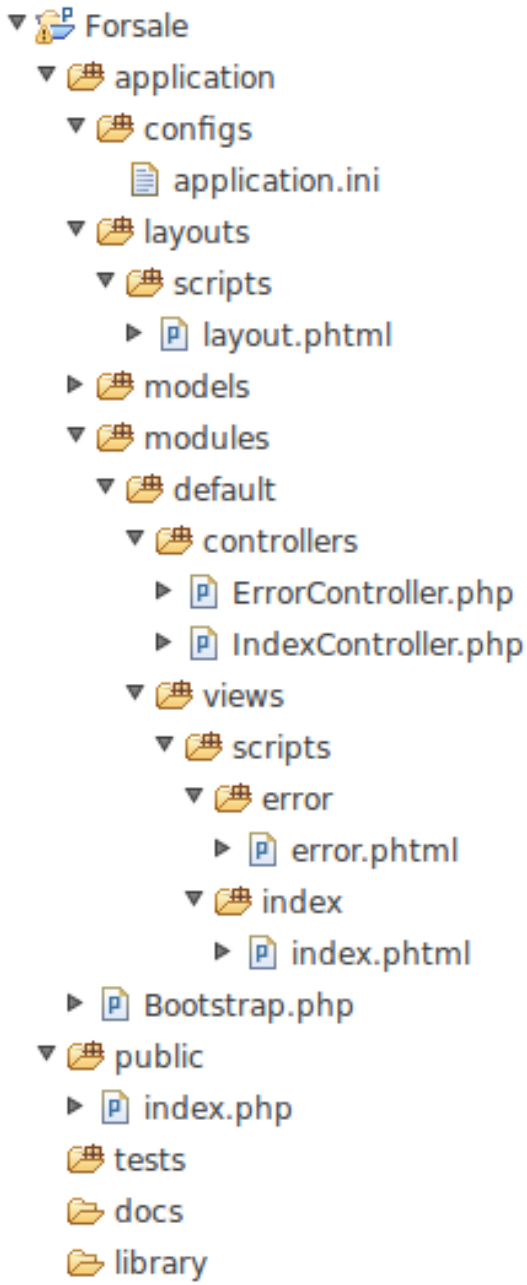
- Aussi appelé 'dispatcher'
- Point d'entrée unique de toutes les requêtes
- Aiguille la requête vers le contrôleur et l'action demandée

Smile
OPEN SOURCE SOLUTIONS

Organisation d'un projet avec Zend Framework

Smile
OPEN SOURCE SOLUTIONS

Architecture




```
Forsale:
  racine de l'application
application:
  MVC et configuration
library:
  Zend et autres bibliothèque
public:
 racine du site DocumentRoot
```

Smile
OPEN SOURCE SOLUTIONS

Configuration du VirtualHost

```
<VirtualHost *:80>
    DocumentRoot "/var/www/forsale/public"
    ServerName zend.lxc

    # This should be omitted in the production environment
    SetEnv APPLICATION_ENV development

    <Directory "/var/www/forsale/public">
        Options Indexes MultiViews FollowSymLinks
        AllowOverride None
        Order allow,deny
        Allow from all

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

    ErrorLog /var/log/apache2/forsale_error.log
    CustomLog /var/log/apache2/forsale_access.log combined

</VirtualHost>
```

Smile
OPEN SOURCE SOLUTIONS

Bootstrapping

```php
<?php
¶
// Define path to application directory
define('APPLICATION_PATH', __DIR__.'/../application');
¶
// Define application environment
define('APPLICATION_ENV', (getenv('APPLICATION_ENV') ? getenv('APPLICATION_ENV') : 'production'));
¶
// Ensure library/ is on include_path
set_include_path(implode(PATH_SEPARATOR, array(
    APPLICATION_PATH . '/../library',
    get_include_path(),
)));
¶
/** 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();
```

public/index.php => fichier d'entrée dans l'application

Initialise les variables, charge la configuration et lance l'application.

Smile OPEN SOURCE SOLUTIONS

Fichier de configuration

```
1 [production]¶
2 appnamespace·=·"Application"¶
3 bootstrap.path·=·APPLICATION_PATH·"/Bootstrap.php"¶
4 bootstrap.class·=·"Bootstrap"¶
5 resources.frontController.params.prefixDefaultModule·=·"1"¶
6 resources.frontController.moduleDirectory·=·APPLICATION_PATH·"/modules"¶
7 ¶
8 [development·:·production]¶
```

- Utilisation de fichier ini mais possibilité d'utiliser d'autres formats
- Création de configuration par environnement
- Notion d'héritage et de surchage par section
- Initialisation de ressources pour le fonctionnement de l'application

Smile
OPEN SOURCE SOLUTIONS

Contrôleurs, Vues et Layouts

Smile
OPEN SOURCE SOLUTIONS

Contrôleurs et Actions

```php
<?php
class Default_IndexController extends Zend_Controller_Action
{
    public function init()
    {
        /* Initialize action controller here */
    }

    public function indexAction()
    {
        // action body
    }
}
```

- Contrôleurs et Actions sont déterminés par l'URL
  `http://zend.lxc/<module>/<controller>/<action>`
- Emplacement
  `application/modules/<module>/controllers`
- Contrôleur par défaut
  `default/index/index`

Smile OPEN SOURCE SOLUTIONS

Communication controller-view

- Le contrôleur accède à l'instance Zend_View par: `$this->view`
- Par défaut une action est associée à une vue `application/views/scripts/<controller>/<action>.phtml`
- La vue est un fichier phtml (PHP + HTML) qui accède directement à l'instance Zend_View par: `$this`
- `Zend_View_Abstract` définie les méthodes magiques `__set` et `__get` pour la création automatique de membres

```
//controller
$this->view->title = 'le titre de ma page'

//view
<?php echo $this->title ?>
```

Smile OPEN SOURCE SOLUTIONS

Le ErrorHandler plugin

- Le ErrorHandler est un plugin du composant Controller
- Il est activé par défaut
- en cas d'erreur, appelle l'action 'error' sur le contrôleur 'error'
- Pour qu'il fonctionne correctement il faut implémenter:
  - `controller/ErrorController.php`
  - `views/scripts/error/error.phtml`

Smile
OPEN SOURCE SOLUTIONS

Le layout de la page

- `Zend_Layout` implémente le Two Step View pattern
- Permet d'obtenir un modèle d'écran commun à tout le site (ou à certaines pages du sites)
- Emplacement `application/layouts/scripts`
  Layout utilisé par défaut: `layout.phtml`
- Intégration de la vue "interne"

```
<?php echo $this->layout()->content ?>
```

- Les variables passées à la vue "interne" sont également accessibles depuis le layout

```
<?php echo $this->title ?>
```

- Activation dans le fichier de configuration

```
resources.layout.layoutPath = APPLICATION_PATH
"/layouts/scripts/"
```

Modèles et base de données

Smile
OPEN SOURCE SOLUTIONS

Configuration de la base de données

- La factory `Zend_DB` initialise l'accès à la base de données
- Il existe plusieurs adapter (mysql, sqlite, ...)
- Dans le fichier de configuration application.ini

```
[development]
resources.db.adapter = Pdo_Mysql
resources.db.params.host = "localhost"
resources.db.params.username = "user_forsale"
resources.db.params.password = "pass_forsale"
resources.db.params.dbname = "forsale"
```

- `Zend_Db` par défaut en "Lazy connexion" : connexion à la base réalisée seulement quand nécessaire
  - la factory ne fait que conserver des paramètres
  - en cas de problème de paramétrage, l'exception aura lieu lors de la première utilisation de la connexion

Création d'un modèle avec Zend_Db_Table

- `Zend_Db_Table` Implémente le design pattern Table Data Gateway
  - pas réellement un ORM (pas de réel objet PHP complet, pas de système de mapping, pas de cache, etc...)
  - plutôt une interface objet légère d'accès aux données d'une table

- Création d'un modèle pour communiquer avec la table 'product'

`application/models/DbTable/Product.php`

```
1 <?php¶
2 ¶
3 class Application_Model_DbTable_Product extends Zend_Db_Table_Abstract¶
4 {¶
5 ····protected $_name = 'product';¶
6 }¶
```

Smile OPEN SOURCE SOLUTIONS

Mise en pratique 1/3

- Afficher une liste de produits avec le nom et le prix sur la page d'accueil de notre application

```
$product = new Product();
$product->fetchAll();
```

- Un lien sur chacun des produits nous permet de voir la fiche détaillée de celui-ci

```
nouvelle action: /product/view

// génération d'une url dans une vue
$this->url(array(
    'controller' => 'product',
    'action' => 'view',
    'id' => $product->id
));

// réccupération d'un paramètre dans un controller
$this->getRequest()->getParam('id');
// chargement d'un produit par son id
$product->fetchRow('id=1'))
```

Smile OPEN SOURCE SOLUTIONS

Mise en pratique 2/3

- Contrôleur de la page d'accueil:

  application/modules/default/controllers/IndexController.php

```php
<?php

class Default_IndexController extends Zend_Controller_Action
{
    public function init()
    {
        /* Initialize action controller here */
    }

    public function indexAction()
    {
        $product = new Application_Model_DbTable_Product();
        $this->view->products = $product->fetchAll();
    }
}
```

- Vue associée:

  application/modules/default/views/scripts/index/index.phtml

```php
<h1>Catalog</h1>
<ul>
<?php foreach($this->products as $product): ?>
    <?php $url = $this->url(array('controller' => 'product', 'action' => 'view', 'id' => $product->id)) ?>
    <li>
        <a href="<?php echo $url ?>"><?php echo $product->name ?></a>: <?php echo $product->price ?>€
    </li>
<?php endforeach ?>
</ul>
```

Smile OPEN SOURCE SOLUTIONS

Mise en pratique 3/3

- Contrôleur de la fiche produit:

application/modules/default/controllers/ProductController.php

```php
<?php

class Default_ProductController extends Zend_Controller_Action
{
    public function viewAction()
    {
        $id = (int) $this->getRequest()->getParam('id');
        $product = new Application_Model_DbTable_Product();
        $this->view->product = $product->fetchRow('id='.$id);
    }
}
```

- Vue associee:

application/modules/default/views/scripts/product/view.phtml

```php
<h1><?php echo $this->product->name ?> - <?php echo  $this->product->price ?>€</h1>
<p><?php echo  $this->product->description ?></p>
<a href="<?php echo $this->url(array(), null, true) ?>">Liste des produits</a>
```

Smile
OPEN SOURCE SOLUTIONS

Les formulaires

Smile
OPEN SOURCE SOLUTIONS

Formulaire d'ajout - Objet Form 1/2

- Emplacement: `application/forms/Product.php`

```php
<?php

class Application_Form_Product extends Zend_Form
{
    public function init()
    {
        $this->setName('product');

        $id = new Zend_Form_Element_Hidden('id');

        $name = new Zend_Form_Element_Text('name');
        $name->setLabel('Product Name')
             ->setRequired(true)
             ->addFilter('StripTags')
             ->addFilter('StringTrim')
             ->addValidator('NotEmpty');

        $submit = new Zend_Form_Element_Submit('submit');

        $this->addElements(array($id, $name, $submit));
    }
}
```

Smile
OPEN SOURCE SOLUTIONS

Formulaire d'ajout - Objet Form 2/2

- Autres type d'éléments de formulaires:
  - Zend_Form_Element_Submit
  - Zend_Form_Element_Textarea
  - ... Cf Zend/Form/Element

- Autres validators:
  - Float
  - GreaterThan
  - Regex
  - ... Cf Zend/Validate

- Autres filtres:
  - HtmlEntities
  - PregReplace
  - ... Cf Zend/Filter

Formulaire d'ajout - Préparation et Affichage

- Dans l'action du contrôleur

```php
<?php
¶
class Admin_ProductController extends Zend_Controller_Action¶
{¶
    public function addAction()¶
    {¶
        $form = new Application_Form_Product();¶
        $form->submit->setLabel('Add');¶
        $this->view->form = $form;¶
    }¶
}¶
```

- Dans la vue

```php
<?php echo $this->form ?>¶
```

Smile
OPEN SOURCE SOLUTIONS

Formulaire d'ajout - Validation et enregistrement

- On poste le formulaire sur la même action

```
5     public function addAction()¶
6     {¶
7         $form = new Application_Form_Product();¶
8         $form->submit->setLabel('Add');¶
9         $this->view->form = $form;¶
10        ¶
11        if ($this->_request->isPost()) {¶
12            $formData = $this->_request->getPost();¶
13            if ($form->isValid($formData)) {¶
14                $product = new Application_Model_DbTable_Product();¶
15                $row = $product->createRow();¶
16                $row->name = $form->getValue('name');¶
17                $row->description = $form->getValue('description');¶
18                $row->price = $form->getValue('price');¶
19                $row->save();¶
20                $this->_redirect('/');¶
21            } else {¶
22                $form->populate($formData);¶
23            }¶
24        }¶
25    }¶
```

- En cas d'erreur, lors de l'enregistrement, une Exception est levée et la page d'erreur s'affiche

Mise en pratique - Formulaire d'édition

- Pas de POST, chargement des données depuis la base:

```
$row = $product->fetchRow('id='.$id);
$form->populate($row->toArray());
```

- Pour la sauvegarde de l'objet, comme le add sauf la création de la ligne remplacée par le chargement de celle de la base:

```
$id = (int)$form->getValue('id');
$row = $product->fetchRow('id='.$id);
```

Smile
OPEN SOURCE SOLUTIONS

Mise en pratique - Suppression du produit

- Ajout du lien
- Utilisation d'un formulaire de confirmation
- Suppression en POST
  - Code de suppression:

```
$product->delete('id='.$id);
```

Smile
OPEN SOURCE SOLUTIONS

Authentification

Smile
OPEN SOURCE SOLUTIONS

Authentification avec Zend_Auth 1/3

- Autoriser uniquement les utilisateurs authentifié à accéder aux fonctions d'éditions (add, edit, delete)

- Utilisation de Zend_Auth

```
$auth = Zend_Auth::getInstance();
$adapter = new Zend_Auth_Adapter_XXX(<params>);
$authResult = $auth->authenticate($adapter);
if ($authResult->isValid()) ...
```

Smile
OPEN SOURCE SOLUTIONS

Authentification avec Zend_Auth 2/3

- Exemple pour LDAP
  - Zend_Auth_Adapter_Ldap($options, $username,$password);
  - Avec options:

```
$options = array('smileldap' =>
    array('host' => 'ldap.lvl.intranet',
          'bindRequiresDn' => true,
          'baseDn' => 'ou=personnes,dc=smile,dc=fr',
          'accountFilterFormat' => 'uid=%s')
);
```

Authentification avec Zend_Auth 3/3

- Par défaut, persistance dans la session
  - namespace "Zend_Auth"

- Persistance Transparent avec la classe Zend_Auth

```
$auth = Zend_Auth::getInstance();
if ($auth->hasIdentity())
{
    $username = $auth->getIdentity();
}
```

- Logout

```
Zend_Auth::getInstance()->clearIdentity();
```

Smile
OPEN SOURCE SOLUTIONS

Registre et logs

Smile
OPEN SOURCE SOLUTIONS

Zend_Registry

Zend_Registry (remplace en plus propre le $GLOBALS)

- Enregistrement

```
Zend_Registry::set('environment',$env);
```

- Récupération

```
$env = Zend_Registry::get('environment');
```

Smile
OPEN SOURCE SOLUTIONS

Log avec Zend_Log

- Objet Zend_Log nécessite au moins un Zend_Log_Writer
  - objet responsable de l'écriture des lignes des log sur fichier, en base, etc...)
  - plusieurs writers possibles à la fois

```
$logWriter = new Zend_Log_Writer_Stream
('/path/to/logfile');
$logger = new Zend_Log();
$logger->addWriter($logWriter);
```

- Chaque message de log a une priorité (error, warning, info, cf Formation Bonnes Pratiques du Dev PHP)

```
$logger->log('Informational message'), Zend_Log::INFO);
```

- ou

```
$logger->info('Informational message');
```

Smile OPEN SOURCE SOLUTIONS