

PHP

Architectures avancées

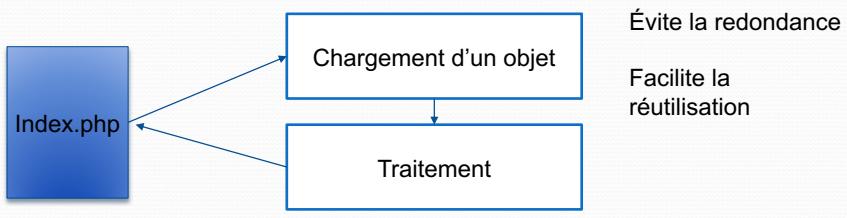


Resp: S. SALVA IUT 2ème année IUT d'aubiere

1

Vers des architectures plus évoluées

- Séparation du traitement des données
- Utilisation de plusieurs classes à partir du CGI initial
- Modèle Service to Workers
- Modèle vue/contrôleur



```
graph TD; A[Index.php] --> B[Chargement d'un objet]; B --> C[Traitement]
```

Évite la redondance
Facilite la réutilisation

2

Vers des architectures plus évoluées

Quels exemples de classes donneriez vous ?????

3

3

MVC de base et optimisation

4

4

Architecture via pattern MVC

Modèle vue contrôleur

- Séparer l'affichage du traitement
- Définition de rôles (qui fait l'affichage, etc.)
- Utilisation de plusieurs fichiers et classes (traitement, affichage, etc.)

Programmation extrêmement répandue, notamment en Web !

ex: Servlet/JSP (struts, spring) en Java

ex: php.mvc, symfony, zend, Yii, Code Igniter, etc. en php

5

5

Architecture via pattern MVC

Modèle vue contrôleur

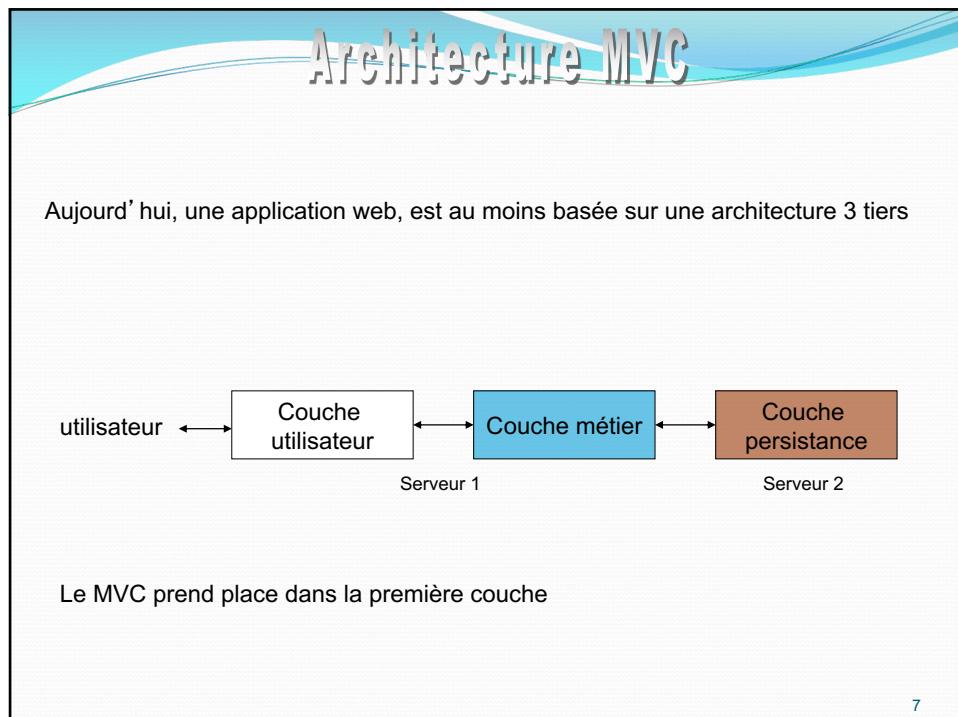
Plusieurs possibilités:

1. Développement d'un MVC complet qui est proche du problème traité (peu fréquent)
2. Utilisation d'un framework MVC
 - **Nécessité de comprendre**, de se former mais ensuite rapidité d'implantation, garantie qualité de certaines parties (DAL, etc.)

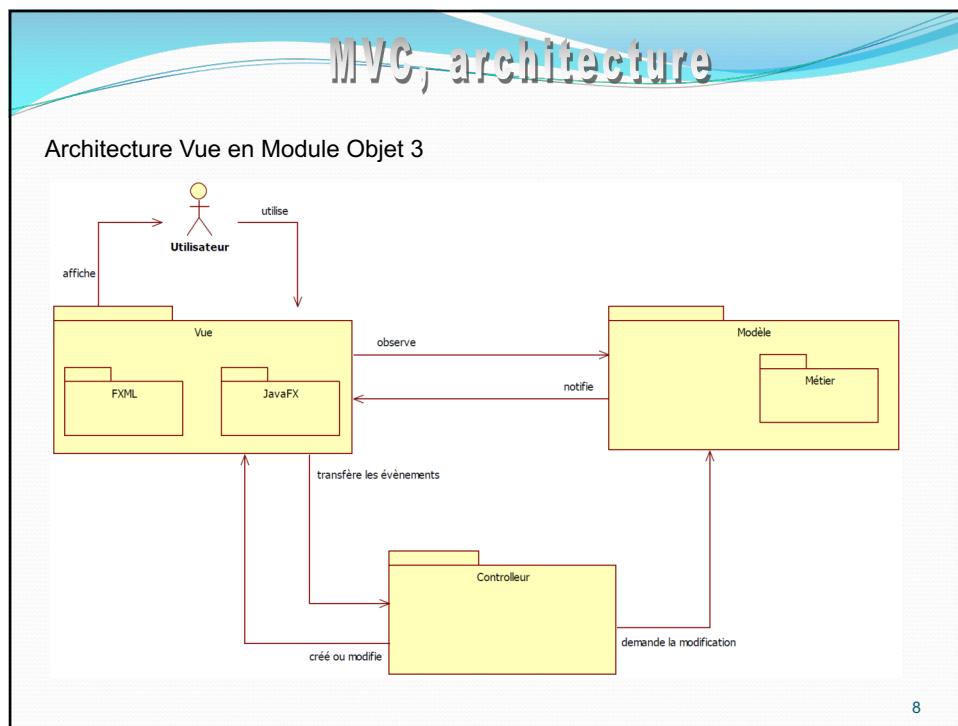
=>Il faut comprendre le MVC -> retour à 1. (Et pourquoi de ce cours)

6

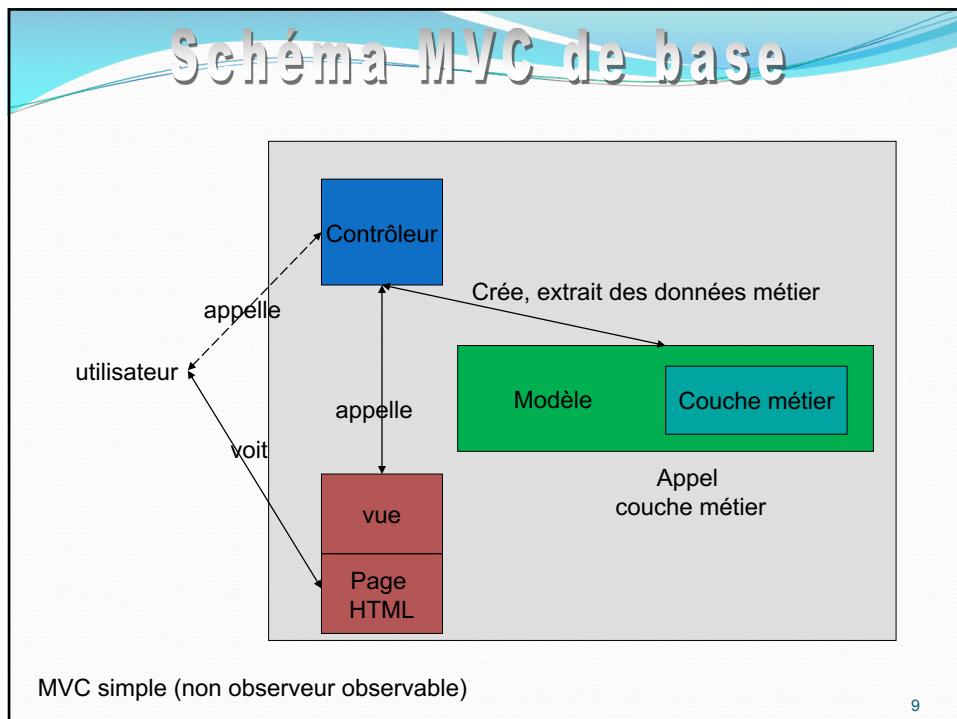
6



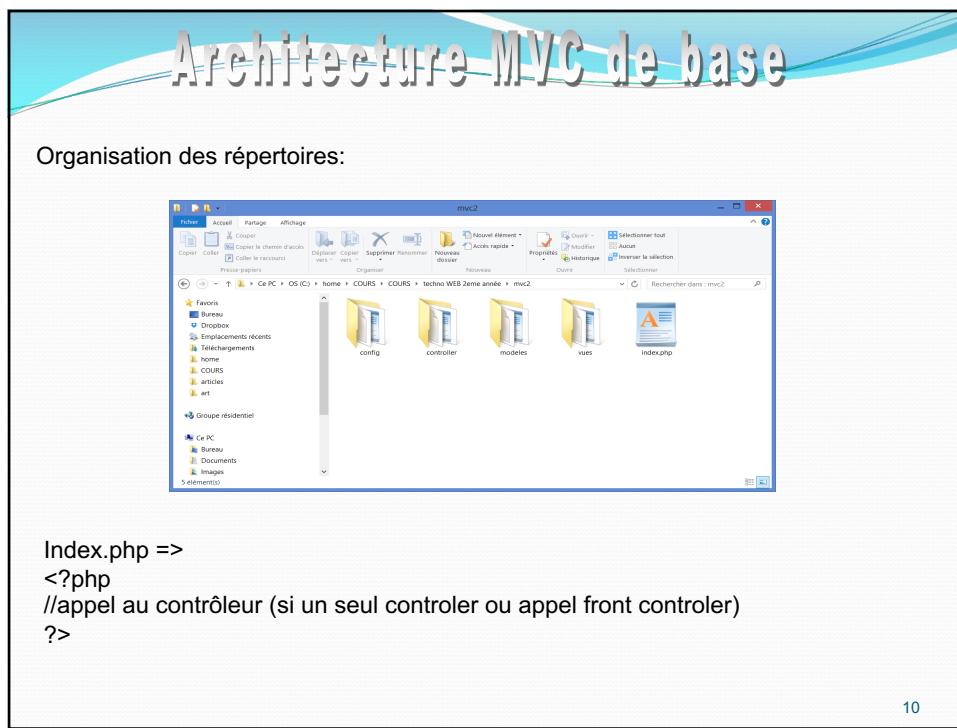
7



8



9



10

Contrôleur dans MVC

Contrôleur

C'est toujours le contrôleur qui est appelé

Son rôle:

1. Contrôler l'intégrité des données reçues (Validation)
(peut aussi être fait par modèle)
2. Lecture d'une **action**:
Action: donnée par la vue
=>indique le traitement à faire
3. Création d'un modèle, gestion des données
4. Redirection vers la vue en donnant les données
(=>page réponse ou une page d'erreur)

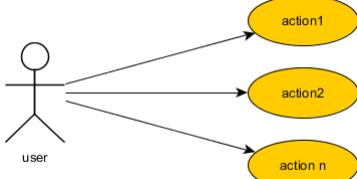
Contrôleur gère les erreurs (classiques + exceptions !!!)

11

11

Contrôleur dans MVC

Les **actions** ?



```

<?php
Class ControllerUser {
    Si Action1 -> méthode 1
    ...
    Si Action n -> méthode n
?>

```

Ex: connecter, ajouterLivre, supprimerLivre,
Action NULL (ou pas d'action ->en général appel page principale)

=> Toujours commencer par effectuer une analyse !!!

12

12

Contrôleur dans MVC

Les **actions** ? Comment les donner ?

Sur un lien, GET

```
<a href=http://...?action=etre_attentif>
```

Dans un formulaire GET ou POST

```
<form action=http://...?action=etre_attentif>
```

Ou

```
<input type=hidden name=action...>
```

13

13

Contrôleur dans MVC

```
<?php
try{
$action=$_GET['action'];
switch($action) {
    case action1:
        action1();
        break;
    ...
    //mauvaise action
    default:
        $dVueEreur[] = "Erreur d'appel
php";
        require
        (__DIR__.'../../vues/vuephp1.php');
        // ajout du code de la vue ici
        break;
} }
```

14

14

Contrôleur dans MVC

```
function action1() {
    //traitement
    $mdl = new modele();
    $mdl->traitement();

    $dVue = array (
        //données
    );
    //utilisation d'une vue pour afficher
    require ('vue/vue.php');
}
?>
```

Contrôleurs toujours identiques, pas besoin du pattern observeur/observable
(possible avec closure et classes anonymes...)

=> Facile à faire depuis une analyse !

15

15

Modèle et MVC

Modèle

Couche qui manipule les données
peut valider les données
peut être simple: retourne une chaîne ? Une liste de chaîne
Souvent fait appel à DAL pour extraire des collections d'objets
-> appel aux classes ***Gateway (cours PDO et DAL)

Modèle = classe, pas d'HTML ou autre

Session, cookies ? Idéalement gestion par un modèle

16

Modèles et MVC

Modèle

Remarques:

Parfois, il n'y a pas de modèle

Pas de SQL dans un modèle, ni de parseur, etc.

Une méthode d'un modèle peut faire plusieurs choses :
demander une collection d'objets depuis une BD, mettre
à jour une session, etc.

17

17

Vues et MVC

**Vue
(html+php)**

Vue : code html+php qui produit la page HTML (php est utilisé en moteur de template)

1. Récupère des variables produites par contrôleur ou modèle (tableaux, variables, etc.)
2. Crée la page avec ces données et echo et boucles

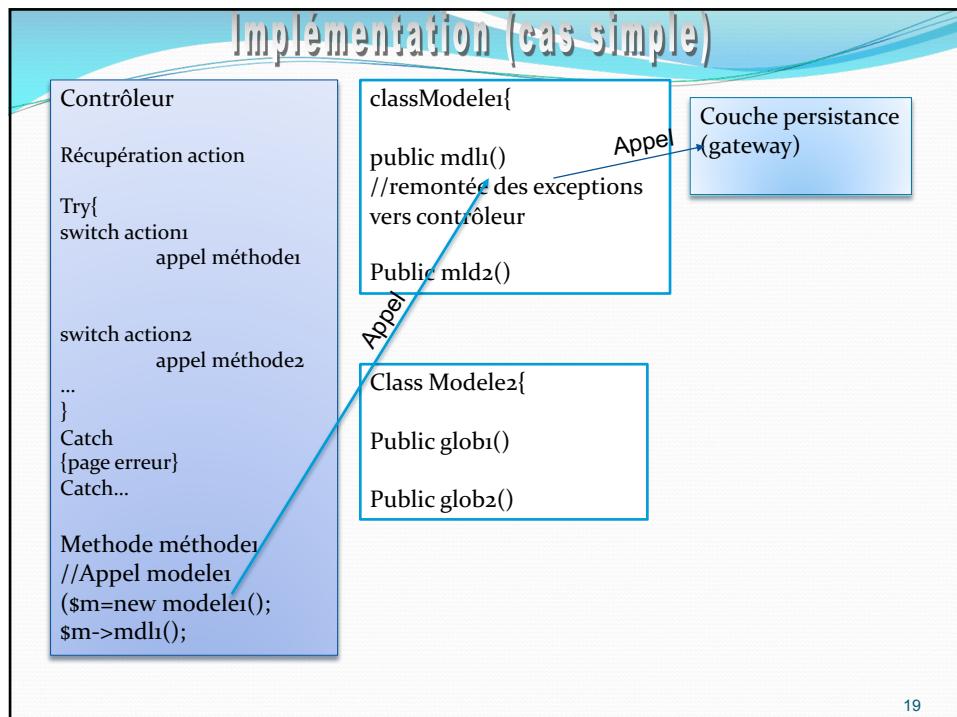
Attention: ne se connecte pas à la base ou ne modifie pas des données => c'est le modèle qui le fait !!!

Pas de : \$_GET, \$_POST, \$_SESSION, etc. car ces données ne sont pas validées !

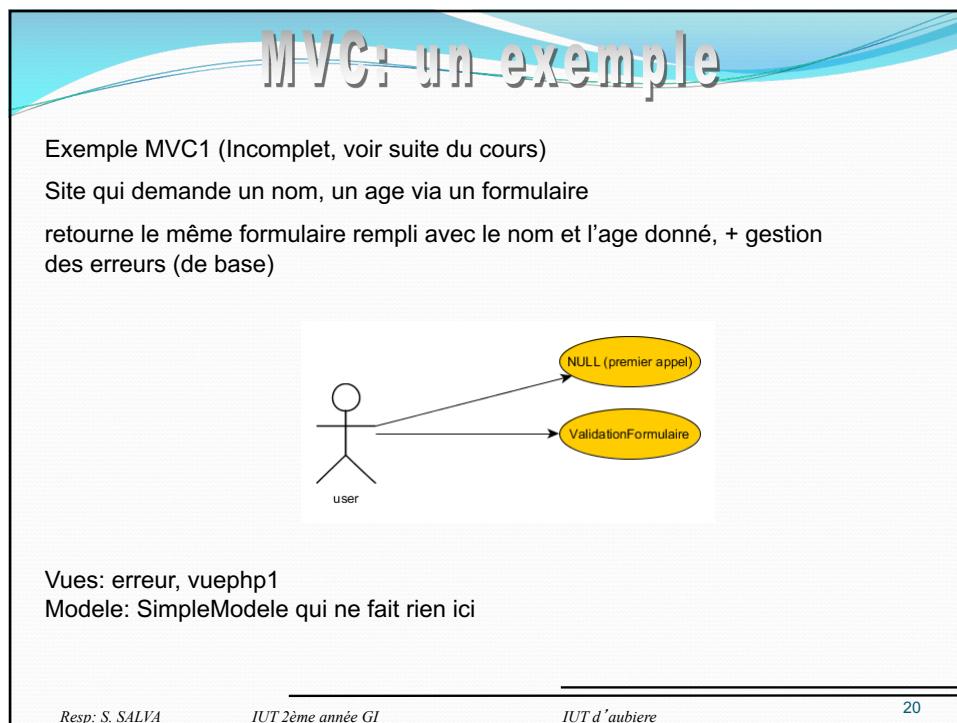
Les boutons (ou autres) de la vue font appels uniquement à des contrôleurs en leur donnant des paramètres et une action !

18

18



19



20

MVC: un exemple

Vuephp d'erreur (minimaliste)

```
<!DOCTYPE html>
<html lang="fr">
<body>

<h1>ERREUR !!!!</h1>

<?php
if (isset($dataVueErreur)) {

foreach ($dataVueErreur as $value){
    echo $value;
}
?>

</body> </html>
```

données obtenues par la variable \$VueErreur

Resp: S. SALVA IUT 2ème année GI IUT d'aubiere 21

21

MVC: un exemple

'vuephp1.php'

```
<!DOCTYPE html>
<html lang="fr">
...<?php
// on vérifie les données provenant du modèle
if (isset($dataVue)) //tableau contenant les informations à afficher
{?>
<center>

<?php

if (isset($dataVueErreur) && count($dataVueErreur)>0) {
    echo "<h2>ERREUR !!!!</h2>";
    foreach ($dataVueErreur as $value){
        echo $value;
    }
?>
<h2>Personne - formulaire</h2><hr>
<!-- affichage des données provenant du modèle, juste pour exemple ici-->
<?= $dataVue['data'] ?>
```

22

22

MVC: un exemple

```

<form method="post" >
<table> <tr>
<td>Nom</td>
<td><input name="txtNom" value=<?= $dataVue['nom'] ?>" type="text" size="20"></td>
</tr>
<tr><td>Age</td>
<td><input name="txtAge" value=<?= $dataVue['age'] ?>" type="text" size="3"></td>
</tr></table>
<table > <tr>
<td><input type="submit" value="Envoyer"></td>

<!-- action !!!!!!! -->
<input type="hidden" name="action" value="validationFormulaire">
</form></center>

<?php }
else {
print ("erreur !!<br>");
print ("utilisation anormale de la vuephp");
} ?>

</body> </html>

```

Donne ici l' action du contrôleur

Resp: S. SALVA *IUT 2ème année GI* *IUT d'aubiere* **23**

23

MVC: un exemple

Index.php ->contrôleur:

```

<?php
require_once(__DIR__.'/Validation.php');
require_once(__DIR__.'/../modeles/Simplemodel.php');
//chargement config
include_once(__DIR__.'/../config/Config.php');

//on initialise un tableau d'erreur
$dataVueEreur = array ();

try{
$action=$_REQUEST['action']; // on peut aussi utiliser $_POST, $_GET bien sûr

switch($action) {

```

24

24

MVC: un exemple

```
//pas d'action, on réinitialise 1er appel
case NULL:
    Reinit();
    break;
case "validationFormulaire":
    ValidationFormulaire();
    break;

//mauvaise action
default:
    $dataVueErreur[] = "Erreur d'appel php";
    require (__DIR__.'../../vues/VueErreur.php');
    break;
}
catch (Exception $e2){...}

//fin
exit(0);
```

On traite l' action dans une méthode à part

Appel vue erreur

Appel vue erreur

25

25

MVC: un exemple

```
function Reinit() {

$dataVue = array (
    'nom' => "",
    'age' => 0,
);

require (__DIR__.'../../vues/vuephp1.php');
}
```

Appel Vue formulaire

26

26

MVC: un exemple

```

function ValidationFormulaire() {
//si exception, ca remonte !!!
$nom=$_POST['txtNom']; // txtNom = nom du champ texte dans le formulaire
$age=$_POST['txtAge'];
Validation::val_form($nom,$age,$dataVueEreur);

$model = new Simplemodel();
$data=$model->get_data();

$dataVue = array (
    'nom' => $nom,
    'age' => $age,
    'data' => $data,
);
require (__DIR__.'/../vues/vuephp1.php');
}

?>

```

ICI
Validation simple
MAIS aussi
Filtrage éventuellement
Ici ou dans modèle

Appel Vue formulaire

27

27

MVC: un exemple

ValidationFormulaire=> valide les paramètres reçus

```

class Validation {

static function val_form($nom,$age,&$dataVueEreur) {

if (!isset($nom)||$nom=="") {
    $dataVueEreur[] ="pas de nom";
    throw new Exception('pas d'action');
}
if (!isset($age)||$age=="") {
    $dataVueEreur[] ="pas d'age ";
    throw new Exception('pas d'age');
}
}} ?>

Mais on peut aussi retourner un booleen

```

28

28

Optimisation de l'architecture précédente

Critique précédente architecture :

- Pas tout objet !
- Un require par classe...
- Des url (de vues) en dur dans le code

=> 2eme architecture

- -> utilisation du fichier de config (ou d'une classe !)
- Autoloader
- **Code MVC v2 dispo sur <http://berlin.iut.local/~progweb/> et <https://github.com/sasa27/Basic-PHP-MVC>**

29

29

Vers des architectures plus évoluées

Comment charger une classe ?

- Require
- **Mieux : autoloader**
- `spl_autoload_register` — Enregistre une fonction en tant qu'implémentation de `__autoload()`->charge des classes
 - <http://php.net/manual/fr/language.oop5.autoload.php>, que vous pouvez faire vous-même
 - *Autoloader PSR-0, PSR-4 si utilisation de Namespaces*, <http://www.php-fig.org/psr/0/fr/>, <http://www.php-fig.org/psr/4/>

30

30

Optimisation de l'architecture précédente

Index.php

```
<?php
//chargement config
require_once(__DIR__.'/config/config.php');

//chargement autocomplete pour autochargement des classes
require_once(__DIR__.'/config/Autoload.php');
Autoload::charger();

//chargement contrôleur
$cont = new controller();
?>
```

31

31

Optimisation de l'architecture précédente

config.php (exemple simple, il y a plein de façons de faire)

Avec des variables globales
 Variables récupérables dans des fonctions avec mot clé *global*
 Voir <http://php.net/manual/fr/language.variables.scope.php>

```
<?php
//préfixe
$rep=__DIR__.'../';

//BD
$base="votre_base";
$login="";
$mdp="";

//Vues
$vues['erreur']='vues/erreur.php';
$vues['vuephp1']='vues/vuephp1.php';
?>
```

32

32

Vers des architectures plus évoluées

Exemple simple de classe de Chargement (de classes)

```

class Autoload{
    private static $_instance = null;

    public static function charger(){
        if(null !== self::$_instance) {
            throw new RuntimeException(sprintf("%s is already
started', __CLASS__));
        }
        self::$_instance = new self();
        if(!spl_autoload_register(array(self::$_instance, '_autoload'),
false)) {
            throw RuntimeException(sprintf("%s : Could not start
the autoload', __CLASS__));
        }
        ....
    }
}

private static function _autoload($class){
    global $rep; // récupère var. globales
    $filename = $class.'.php';
    $dir
    =array('modeles','./','config','contrôleur');
    foreach ($dir as $d){
        $file=$rep.$d.$filename;
        //echo $file;
        if (file_exists($file))
        {
            include $file;
        }
    }
}

```

A chaque *new* -> on lance *_autoload()*

33

33

Optimisation de l'architecture précédente

Classe Controller

```

<?php

class controller {

    function __construct(){
        //code du contrôleur ici
        global $rep,$vues;

        // on démarre ou reprend la session
        session_start();

        ...
        try{
            $action=$_REQUEST['action'];
            switch($action) {
                //pas d'action, on réinitialise 1er
                //appel
            }
        }
        ...
    }

    ...
    catch (PDOException $e)
    {
        //si erreur BD, pas le cas ici
        $dataVueEreur[] = "Erreur
inattendue!!! ";
        require ($rep.$vues['erreur']);
    }

    catch (Exception $e2)
    {
        $dataVueEreur[] = "Erreur
inattendue!!! ";
        require ($rep.$vues['erreur']);
    }
    //fin constructeur

    function Reinit() {
        ...
        require ($rep.$vues['vuephp1']);
    }

    function ValidationFormulaire() {
        ...
    }
}

```

34

34

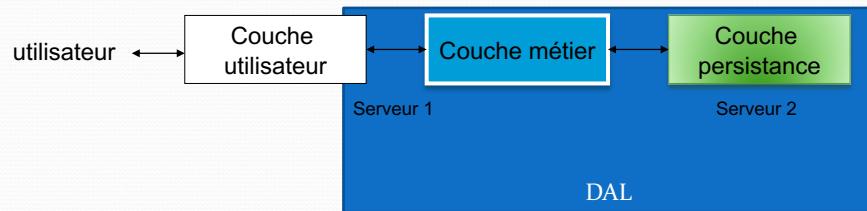
Lien avec Data access layer

35

35

Présentation DAL

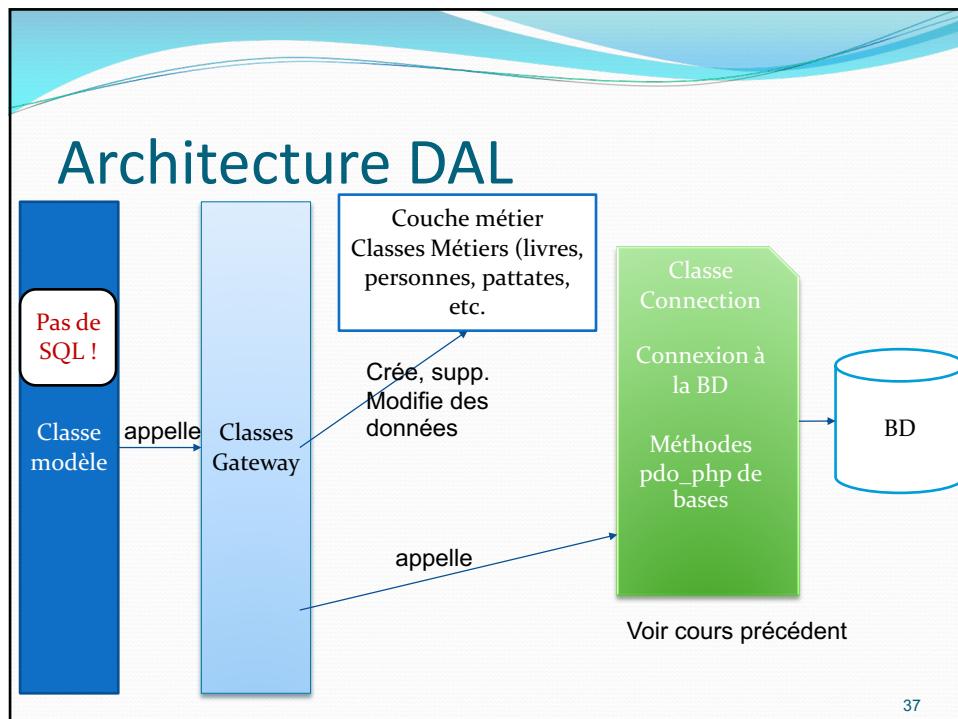
- Couche d'accès aux données (DAL)



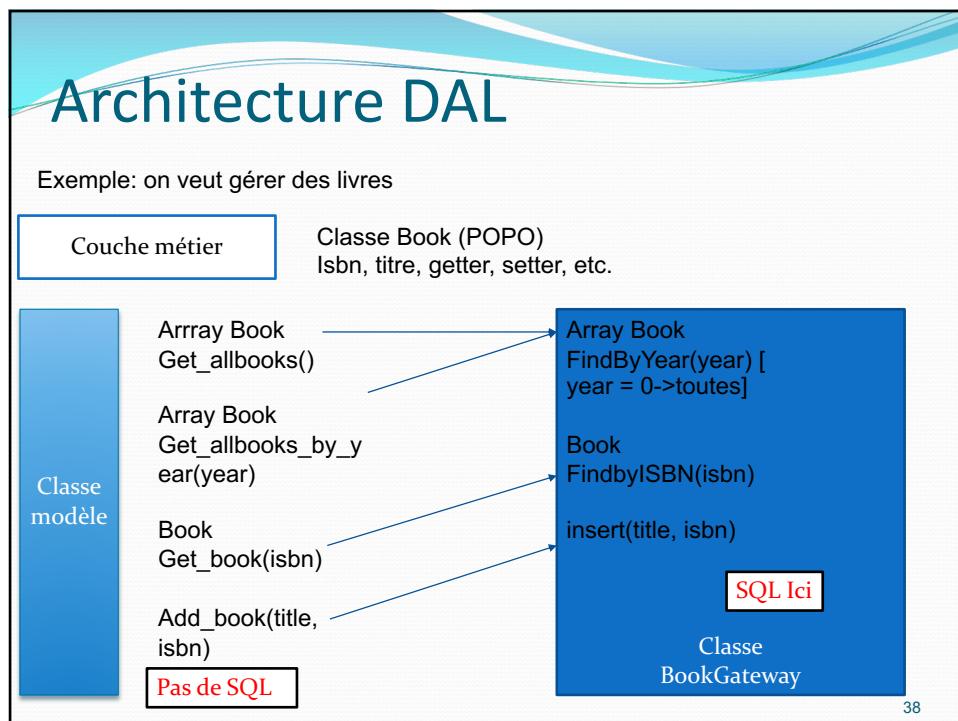
- Couche généralement composée de plusieurs classes
 - Permet de gérer les données à la sauce Objet
- on y trouve des patrons : singleton, fabrique, stratégie, etc.

36

36



37



38

Architecture DAL

Exemple: on veut gérer des livres

Classe modèle

```
Public function get_allbooks($year){  
  
    $b = new BookGateway(...);  
    $Tab_de_Books=$b->FindByYear($year);  
    // puis renvoyer vers le contrôleur (return) ou autre...  
}  
  
Ou avec méthodes statiques quand c'est possible
```

39

39

Architecture DAL

Exemple: on veut gérer des instances de Books

Classe
BookG
ateway

```
Public function FindByYear($year){  
    //préparation + execution requête sql  
    If $year>>0 {  
        $query=' select * from Books where year=:year';  
        $this->$con->executeQuery($query, array( ':year'  
=> array($year,PDO::PARAM_STR),  
            ));  
  
        $results=$this->con->getResults();  
        Foreach ($results as $row)  
            $Tab_de_Books[]=new  
Book($row['isbn'], $row['titre'], $row['year']);  
        Return $Tab_de_Books;  
    }  
    Else{...}
```

Création des
objets,
On retourne les
objets

40

40

Architecture DAL

Composée du code en PDO

Rôle:
préparer les requêtes, injecter les paramètres, exécuter la requête
=> Découpler l'accès bas niveau de la BD du reste du projet

Non rôle:
ne traite pas les erreurs
Pas d'affichage

!! Ne pas retourner d'objet statement (objet retourné par l'appel prepare)
!! Ne pas faire de fetchall (ou autre) or de cette classe
Chacun son rôle

41

41

Architecture DAL

Gestion des erreurs (PDOException)

Cas 1 : le plus simple

Pas de *try catch* dans la classe Connection
Les erreurs sont automatiquement remontées dans le Contrôleur

C'est le rôle du Contrôleur de récupérer les erreurs, de les gérer et d'appeler la bonne vue qui affiche les erreurs

Dans contrôleur

```
Try{ $action=... switch($action)
Catch (PDOException $e)
    {$dataVueEreur[] = $e->getMessage()
    require ($vue['vue']);}
```

42

42

Architecture DAL

Classe
Connection

Connexion à
la BD

Méthodes
pdo_php de
bases

Gestion des erreurs

Cas 2 :

Class Connection : pas de *try ...catch*

Les classes *Gateway récupèrent les exceptions PDO
retournent des exceptions non PDO

**Catch (PDOException \$e) {throw new Exception
('message');}**

=> découpler PDO du reste
=>le contrôleur n'est plus lié à PDO

ou ajoute des messages d'erreur dans tableau d'erreur
tableau d'erreur = var globale

43

43

Gestion des acteurs !

44

44

Des acteurs ...

- Plusieurs acteurs peuvent être utilisateurs d'une application Web:

- Visiteur



- Utilisateur



- Administrateur



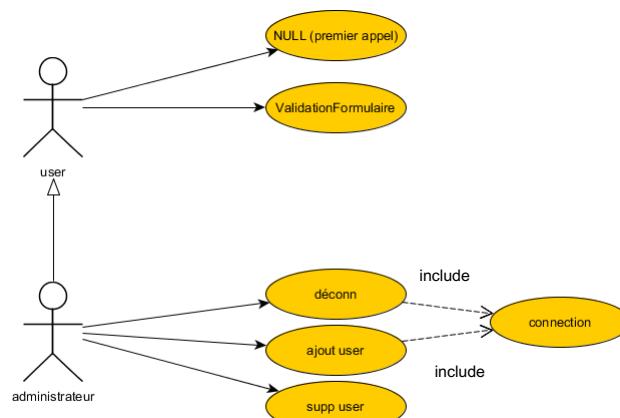
-

45

45

Des acteurs

- En UML, exemple:



46

46

Des acteurs

- En UML, exemple:

Contrôleur User
2 actions

Contrôleur Admin
4 actions

+:
on fait l'analyse, et on a l'architecture !
C'est clair, propre, maintenable, pas 50 questions à se poser

47

47

Des acteurs

- Comment lier les contrôleurs

Solution 1 Héritage
Dans chaque Vue, on fait appel à l'un ou l'autre
+: solution simple
-: mais complexe à mettre en œuvre car chaque lien ou formulaire doit faire appel au bon contrôleur

On perd en souplesse

Contrôleur User
2 actions

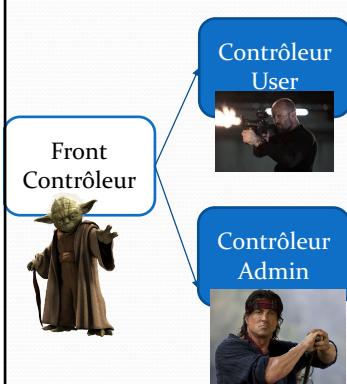
Contrôleur Admin
4 actions

48

48

Des acteurs

- Comment lier les contrôleurs



Solution 2 Front contrôleur (a.k.a. dispatcheur)

Rôle: choisit le bon contrôleur suivant les droits et l'action demandée

- : ajout d'un contrôleur supplémentaire
- +: toujours le même, partout
- +: gère les droits

49

49

Front controller

- C'est un patron d'architecture
- Wikipedia:

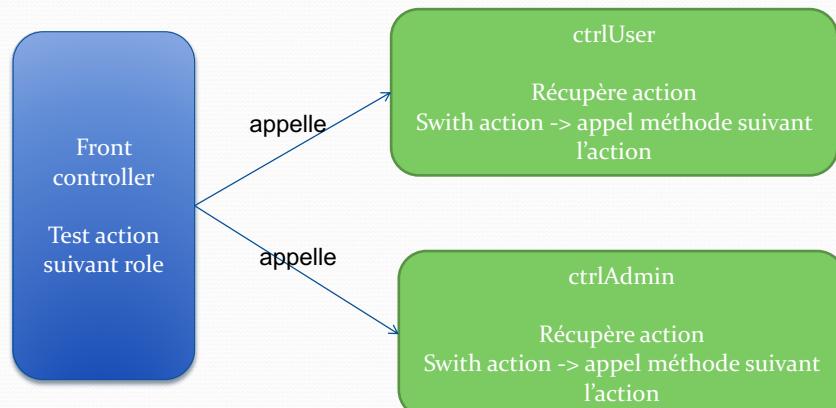
The **Front Controller Pattern** is a software [design pattern](#) listed in several pattern catalogs.

The pattern relates to the design of web applications. It "provides a centralized entry point for handling requests."

50

50

Front controller (du cours)

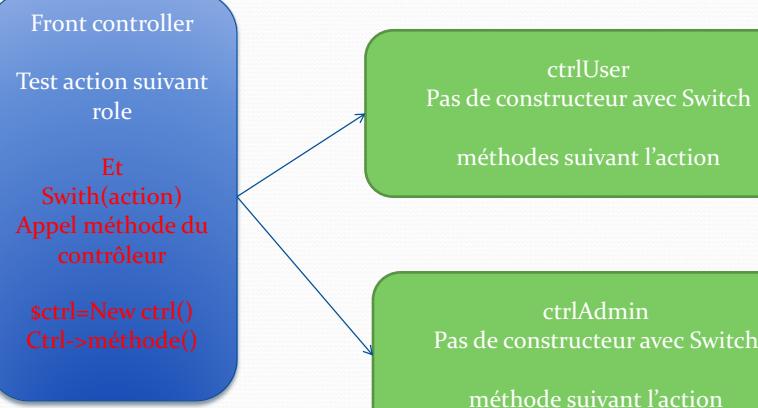


51

51

Front controller

Dans les Frameworks: c'est un peu différent



52

52

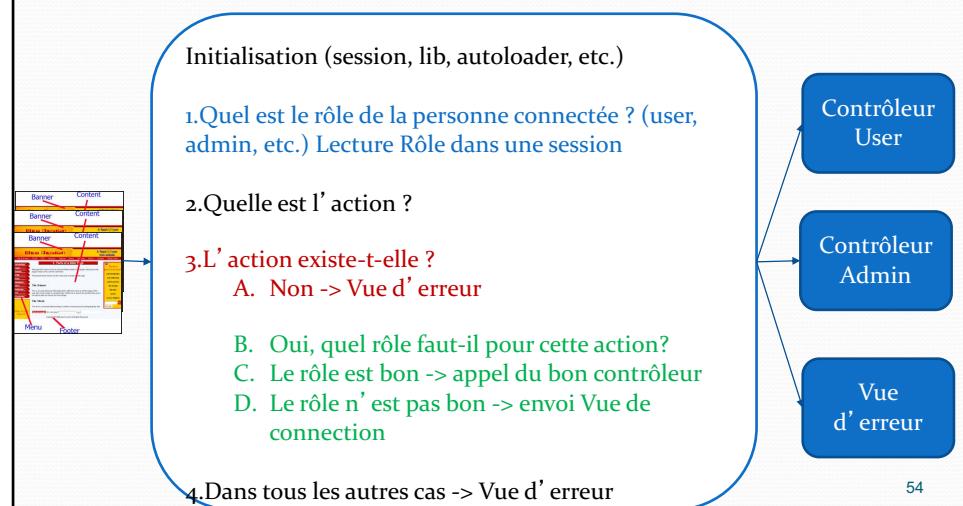
Front controller

- Front controller = classe appelée dans index.php (new FrontController());
- Reprenez toutes les vues
 - Chaque lien ou action doit faire appel à index.php
- Code principal du Front controller dans son constructeur

53

Front controller

- Idée générale de fonctionnement



54

54

Front controller

Quel est le rôle de la personne connectée ? (user, admin, etc.) Lecture du rôle dans une session

Exemple:

- Classe MdlAdmin
 - Méthode connection()
 - Méthode déconnexion()
 - Méthode isAdmin()

55

Front controller

Classe MdlAdmin

- Méthode connection(\$login, \$mdp)
 1. Nettoyage (\$login, \$mdp)
 2. Appel DAL pour vérifier si \$login+\$mdp dans la BD
 3. Ajout Rôle dans session


```
$_SESSION['role']='admin';
$_SESSION['login']=$login;
```
 - On peut sécuriser en cryptant données de session, etc.
 - <http://php.net/manual/fr/session.security.php>
 - Méthode déconnexion()
 - Méthode isAdmin()

56

Front controller

- Classe MdlAdmin

- Méthode déconnexion()

```
session_unset();
session_destroy();
$_SESSION = array();
```

- Méthode isAdmin()

```
{ //teste rôle dans la session, retourne instance d'objet ou booleen
if isset $_SESSION['login'] && isset $_SESSION['role'] {
    $login=Nettoyer::nettoyer_string($_SESSION['login']);
    $role=Nettoyer::nettoyer_string($_SESSION['role']);
    return new Admin($login,$role)
} else return null;
}
```

57

Front controller

- Algorithme simplifié (2 acteurs Admin, User):

```
$listeAction_Admin = array('deconnecter','supprimer', 'ajouter');
//appel modèle admin pour vérifier si utilisateur est connecté
Try{
    $admin = mdlAdmin.isAdmin(); (ici il suffit de tester si la session existe ou pas)
```

Récupération de l'action
Si action dans listeAction_Admin
 si \$admin=null
 require Page_Authentification // ou appel ctrlUser avec action connecter
 sinon
 new CtrlAdmin()

Sinon new CtrlUser()
//on peut aussi valider l'action ici au lieu de le faire dans Ctrl User
}
Catch {require Page_Erreur}

58

58

Front controller

- Algorithme simplifié (2 acteurs Admin, User):

Attention:

toujours tester si \$admin==null au début de CtrAdmin

Si null ->page d'erreur

appel du Ctrl Admin directement doit retourner une page d'erreur !!!

59

59

Front Controller

```

try{ $String_actor="";
$listeActions=array(
    'Admin' => array ('action1','action2'),
    'SuperAdmin' => array ('action3','action4'));

$action=$_REQUEST['action'];
//methode isGoodAction -> retourne " ou acteur

$string_actor=isGoodAction($action);
$mdlclass="Mdl".$string_actor;
$mdlActor= new $mdlclass();
$actor=$mdlActor->isActor();

if ($string_actor!=""){
    if ($actor==null)
        require('pageAuth'.$string_actor);
    else {
        $ctrlclass= "Ctrl".$string_actor;
        new $ctrlclass();
    }
}
else {new CtrlUser();}
}

catch(Exception $e) {require ('pageErreur');} catch(Error $e2) { $error=$e2->getMessage(); require ('pageErreur');}

```

60

60

Front controller

- Autres fonctionnalités du front controller:
 - Vérification de toutes les actions

61

61

Front controller

- Autres fonctionnalités du front (mode avancé)

URL rewriting, routing (Au lieu d'utiliser des paramètres GET/POST ou de formulaire):

 - Action dans l'URL ex: .../user1/action1/param1/param2
 - front controller découpe l'URL récupère l'action et envoie au Contrôleur qui doit faire l'action
 - -> routage !!!
 - <http://www.phpaddiction.com/tags/axial/url-routing-with-php-part-one/>
 - <https://github.com/bencrowder/router>

62

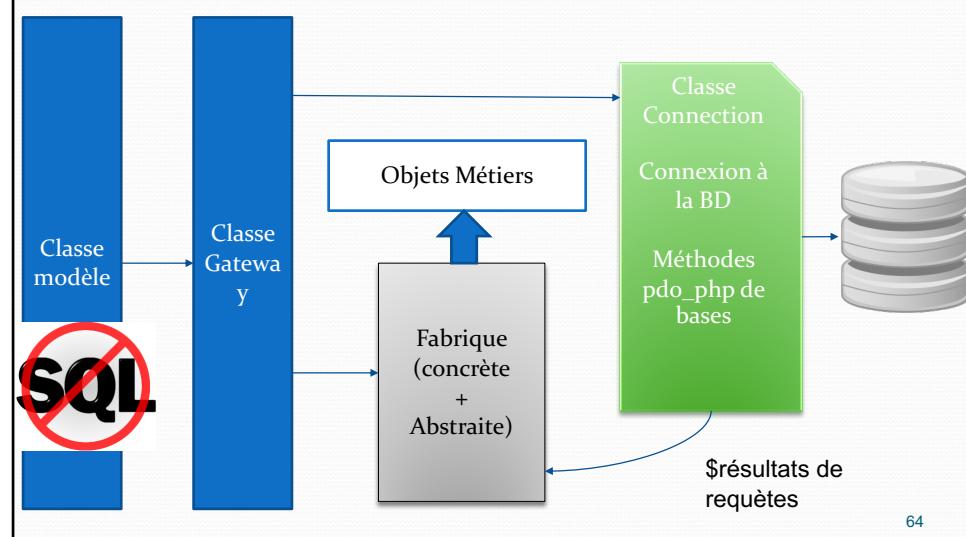
62

Pour aller (un peu) plus loin!

63

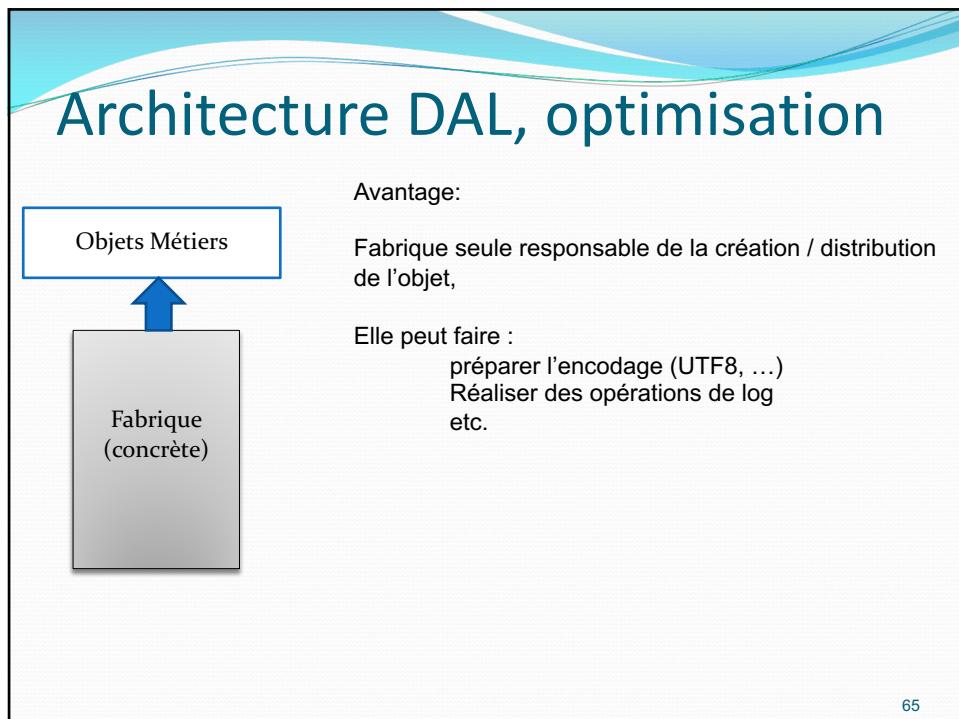
63

Architecture DAL, optimisation

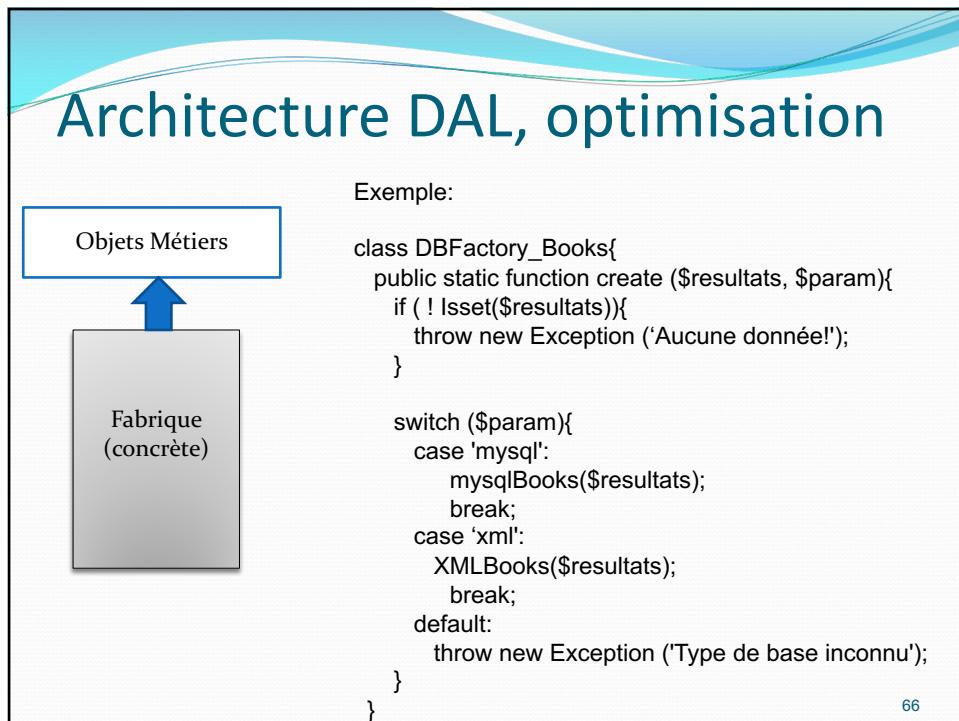


64

64



65



66

Architecture DAL, optimisation

Exemple:

```
Objets Métiers
↑
Fabrique (concrète)
```

```
mysqlBooks($resultats) {
    $Tab_Books=array(); $i=0;
    Foreach ($results as $row)
        $Tab_de_Books[]=$row=new Book($row['isbn'], $row['titre'], $row['year']);
    Return $Tab_Books;
}
```

67

67

Séparation HTML PHP

Moteur de template : séparer code HTML et PHP

Pourquoi ? Souvent 2 métiers (postes)

```

graph LR
    HTML[HTML] --> MT[Moteur template]
    PHP[PHP] --> MT
    MT --> VB[Variables boucles]
    MT --> V[variables]
  
```

Soit à la main (PHP est un moteur de template à la base)

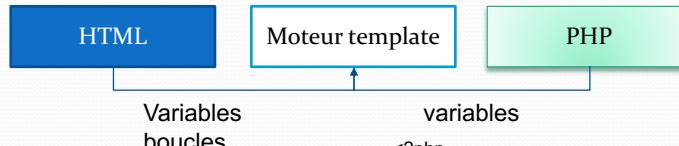
Soit outils
Exemple Moteur : twig (<https://twig.symfony.com/>)

Resp: S. SALVA IUT 2ème année GI IUT d'aubiere 68

68

Séparation HTML PHP

Exemples:



Template Index.html

```

<html><body>
<strong>{{ name }}</strong> <br />

{% if users|length > 0 %}<ul>
{% for user in users %}
    <li>{{ user.username }}</li>
{% endfor %}
{% endif %}

</body></html>
  
```

```

<?php
require_once __DIR__ . '/vendor/autoload.php';
$loader = new Twig_Loader_Filesystem('templates'); // Dossier contenant les templates
$twig = new Twig_Environment($loader, array( 'cache' => false ));

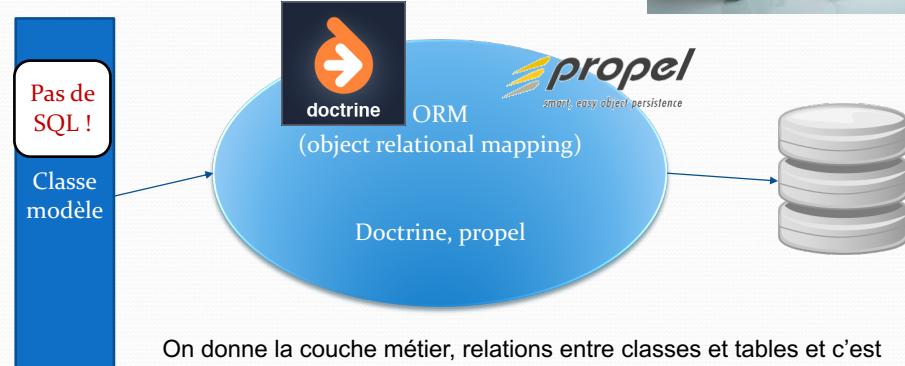
echo $twig->render('index.html', array(
    'name' => 'Salva',
    'users'=> array(
        '1'=> array(
            'username'=>'toto'),
        '2'=> array(
            'username'=>'sasa')
    )));
  
```

69

69

Architecture DAL, optimisation

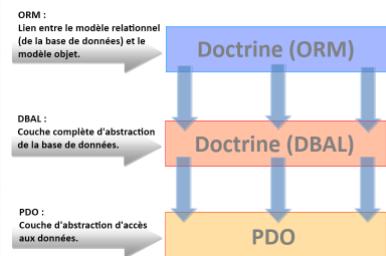
Faire une DAL au moins une fois pour comprendre c'est bien
Mais à chaque fois, c'est lourd !



70

70

Architecture DAL, optimisation



Tutoriel : <http://openclassrooms.com/courses/apprendre-a-utiliser-doctrine>

Résumé dans la suite:

71

71

Architecture DAL, optimisation



Doctrine 2:

Installation avec Composer

<https://www.doctrine-project.org/projects/doctrine-orm/en/2.6/tutorials/getting-started.html>

Configuration de EntityManager:

Classe qui donne accès à l'ORM, entités (classes métiers), persistance.

72

72

Architecture DAL, optimisation



Configuration de EntityManager:

```
<?php
// bootstrap.php
use Doctrine\ORM\Tools\Setup;
use Doctrine\ORM\EntityManager;

require_once "vendor/autoload.php";

// Create a simple "default" Doctrine ORM configuration for Annotations
$isDevMode = true;
$proxyDir = null;
$cache = null;
$useSimpleAnnotationReader = false;
$config = Setup::createAnnotationMetadataConfiguration(array(__DIR__."/src"),
$isDevMode, $proxyDir, $cache, $useSimpleAnnotationReader);
// or if you prefer yaml or XML
// $config = Setup::createXMLMetadataConfiguration(array(__DIR__."/config/xml"), $isDevMode);
// $config = Setup::createYAMLMetadataConfiguration(array(__DIR__."/config/yaml"), $isDevMode);

// database configuration parameters
$conn = array(
    'driver' => 'pdo_sqlite',
    'path' => __DIR__ . '/db.sqlite',
);

// obtaining the entity manager
$entityManager = EntityManager::create($conn, $config);
```

73

73

Architecture DAL, optimisation

Création d'entité avec annotations dans le code (ou fichier XML)

```
<?php
// src/Product.php

use Doctrine\ORM\Mapping as ORM;

/**
 * @ORM\Entity
 * @ORM\Table(name="products")
 */
class Product
{
    /**
     * @ORM\Id
     * @ORM\Column(type="integer")
     * @ORM\GeneratedValue
     */
    protected $id;

    /**
     * @ORM\Column(type="string")
     */
    protected $name;

    // .. (other code)
}
```

74

74

Architecture DAL, optimisation

Création de la BD associée

Taper en ligne de commande:
 vendor/bin/doctrine orm:schema-tool:create

Ou

vendor/bin/doctrine orm:schema-tool:update --force --dump-sql
 Pour mettre à jour

75

75

Architecture DAL, optimisation



Insertion en BD

```
$product = new Product();
$product->setName('toto');

$entityManager->persist($product);
$entityManager->flush();

echo "Created Product with ID " . $product->getId() . "\n";
```

Suppression

```
$entityManager->remove($product);
$entityManager->flush();
```

76

76

Architecture DAL, optimisation



Lecture en BD (tous)

```
$productRepository = $entityManager->getRepository('Product');
$products = $productRepository->findAll();

foreach ($products as $product) {
    echo sprintf("-%s\n", $product->getName());
}
```

77

77

Architecture DAL, optimisation



Lecture en BD (par id) puis modification

```
$id = $argv[1];
$newName = $argv[2];

$product = $entityManager->find('Product', $id);
if ($product === null) {
    echo "Product $id does not exist.\n";
    exit(1);
}
$product->setName($newName);
$entityManager->flush();
```

78

78

Architecture DAL, optimisation



Lecture en BD (par id) puis modification

```
$id = $argv[1];
$newName = $argv[2];

$product = $entityManager->find('Product', $id);
if ($product === null) {
    echo "Product $id does not exist.\n";
    exit(1);
}
$product->setName($newName);
$entityManager->flush();
```

79

79

Architecture DAL, optimisation



Lecture en BD avec condition simple (<https://www.doctrine-project.org/projects/doctrine-orm/en/2.6/reference/working-with-objects.html>)

```
// All users that are 20 years old
$users = $em->getRepository('MyProject\Domain\User')->findBy(array('age' => 20));

// All users that are 20 years old and have a surname of 'Miller'
$users = $em->getRepository('MyProject\Domain\User')->findBy(array('age' => 20,
'surname' => 'Miller'));

// A single user by its nickname
$user = $em->getRepository('MyProject\Domain\User')->findOneBy(array('nickname' => 'romanb'));
```

80

80

Architecture DAL, optimisation



Requêtes complexes (en DQL, req sur instance d'objet)

```
$dql = "SELECT b, e, r, p FROM Bug b JOIN b.engineer e ".
      "JOIN b.reporter r JOIN b.products p ORDER BY b.created DESC";
$query = $entityManager->createQuery($dql);
$bugs = $query->getArrayResult();
```

81

81

Ce qui n'a pas été vu

Frameworks

Simples:

code igniter (<https://ellislab.com/codeigniter>),
 Yii (<http://www.yiiframework.com/>)

Plus complexes:

zend (<http://framework.zend.com/>),
 symfony2 (<http://symfony.com/>)

Ces frameworks intègrent des ORM soit maison soit doctrine, propel

82

82

Ce qui n'a pas été vu

Oubliez les ORM, les bases de données ????????

Services Web (Rest, SOAP)
instances d'objets sur serveurs Web

génériques, utilisables sur Web, Mobile, desktop !

Marre du SQL ?
Faites du NoSQL
dépôt où l'on met de tout comme on veut
MongoDb, ElasticSearch

Marre de l'hébergement (d'ailleurs c'est quoi ?)
Faites du Cloud, des VMs,

Resp: S. SALVA

IUT 2ème année GI

IUT d'aubiere

83

83

Ce qui n'a pas été vu

Le vaste Monde du JS (Javascript), qui bouge tous les 6 mois

Tout est mis à la sauce JS ! (windows, mobile, jeux)

Jquery (à connaître, de base)

WinJs pour Windows 8, 10

createJs,phaser ->jeux en JS

BabylonJS->(opengl (webgl)) dans navigateur, façon easy

Aframe -> VR

NodeJs (application réseaux serveur en asynchrone) <http://>
envie de faire son serveur Web plus rapide qu'A



84

84

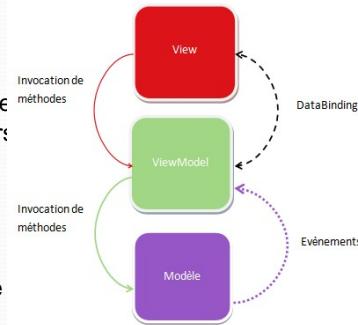
Ce qui n'a pas été vu

Angular, React
faire du MVVM (adieu MVC?)

- La vue est couplée aux données via du DataBinding et invoque les méthodes du ViewModel.

Le ViewModel invoque les méthodes du modèle.
contient la data spécifique à la gestion de l'écran et les méthodes de réponses aux interactions utilisateurs;
contient également une référence vers un ou des modèles.

- Le modèle contient la data et les méthodes de manipulation de cette dernière (calculs, appels de services, ...).



Resp: S. SALVA

IUT 2ème année GI

IUT d'aubiere

85

85

Ce qui n'a pas été vu

Faire des tests !

Test unitaires (de classes): phputil

Test d'intégration: selenium, CasperJs



86

86

D'autres patterns

Pattern Observer avec classes anonymes et **SplObserver**

Pattern CommandBus

But: séparer les aspects techniques lourds des contrôleurs
Ex: acheter un podcast -> demander login, utiliser carte de crédit, assigner podcast, etc.

Objet Command = classe métier contenant les données provenant d'une requête, puis exécuter par un Objet Commandhandler (découplage)

Intégré dans les frameworks laravel, symfony, etc.

Ex:

<http://php-and-symfony.matthiasnoback.nl/2015/01/a-wave-of-command-buses/>

Implémentation: simpleBus: <https://github.com/SimpleBus>