Université Sorbonne Paris Nord

IUT BOBIGNY

R213 : Développement Web

Jérôme Nobécourt

jerome.nobecourt@gmail.com

Sur le réseau pédagogique http://tomcat/Portail

PHP : Programmation Objet, Principe de conception d'une application en OO

Nom de la ressource	R213 - Développement Web			
Semestre	Semestre 2			
Compétence(s) ciblée(s)				
Comprendre	Concevoir	Exprimer	Développer	Entreprendre
-	-	-	X	-
Apprentissages critiques				
Niveau 1 de Comprendre	Niveau 1 de Concevoir	Niveau 1 de Exprimer	Niveau 1 de Développer	Niveau 1 de Entreprendre
			AC4103 : Générer des pages Web ou vues à partir de données structurées incluant des interactions simples AC4105 : Modéliser les données et les traitements d'une application Web AC4106 : Utiliser et adapter un modèle d'accès aux données	
SAÉ concernée(s)	SAE203 : Site Web et BDD			
Prérequis	R113 : Développement Web et R216 : Système d'information			
Descriptif détaillé	Objectif : Découvrir la programmation Web côté serveur et les interactions avec une base de données Introduction au modèle de conception MVC (focus sur les modèles et l'accès aux données) Modélisation des traitements et transmission de données d'une page à l'autre (méthode GET) - Découverte d'une librairie permettant l'accès à une base de données - Gestion des échanges de données client/serveur (méthode POST, gestion des formulaires et validation des données) - Compléments algorithmiques - Outils et méthodes de débogage			
Mots clés :	Développement Web, client/serveur, débogage, accès aux données,			
Heures de formation (dont TP)	20 heures (dont 10 heures TP)			

1. Utilisation de PHP et de mariaDB
2. On a déjà utilisé le MVC, mais juste au niveau du contrôleur, on ajoute maintenant le coté modèle qui s'occupe des données.
3. Le serveur gère les réponses aux questions du client
4. En PHP, la bibliothèque PDO
5. Utilisation des échanges entre le client et le serveur via HTTP

Implique savoir programmer en objet

Rappel : notion de type

Un type correspond à :

- l'ensemble des valeurs que peut prendre la variable
- l'ensemble des opérations utilisables
- une codification de l'information (signification du codage binaire)

Lorsque le contenu d'une variable correspond à :

- **l'information codée**, on parle de **type statique**. C'est le cas de tous les types de base (entier, booléen, caractère, réel, scalaire).
- **l'adresse mémoire** de la zone ou est stockée l'information, on parle de **type dynamique**. C'est le cas pour les objets, les chaînes de caractères et les tableaux.

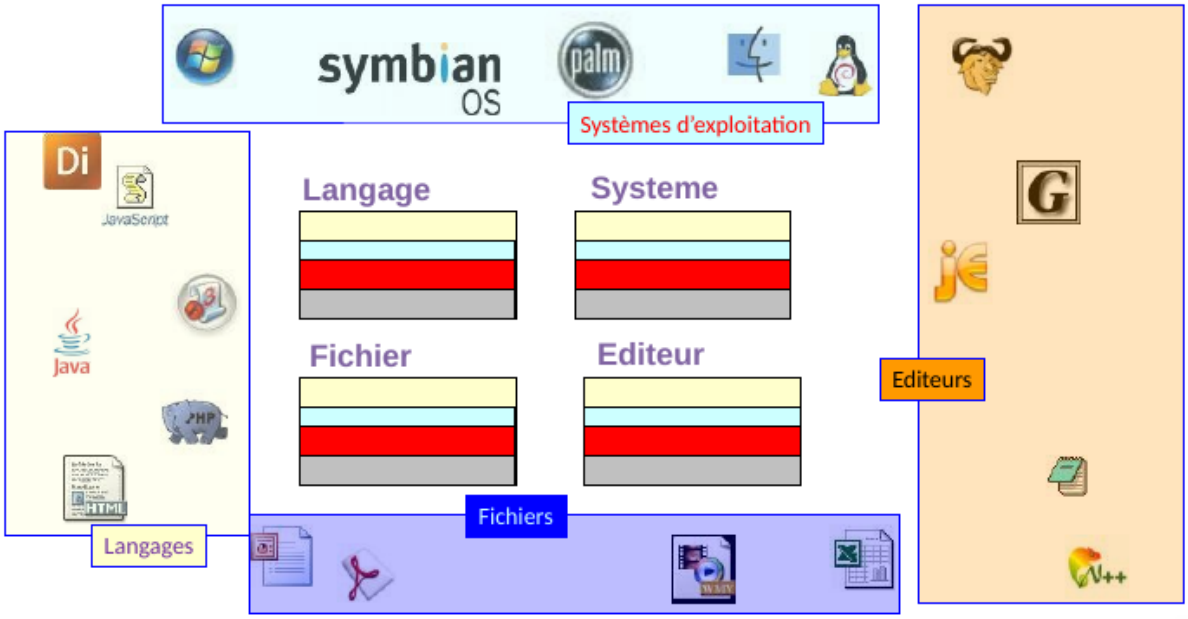
C’est quoi un objet ?

Voici des instances. Les objets sont ?

C'est quoi les classes ?

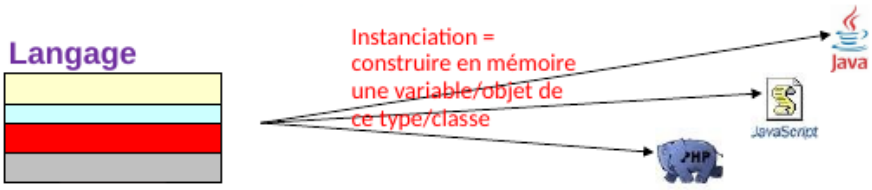
Des instances, des objets et des classes

Des instances, des objets et des classes

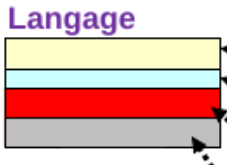


Des classes et des instances

- **Classe**
  - –Un patron
  - –La description de ce que sait faire la boîte noire
  - **–un type**
- **Instance**
  - –Un modèle réalisé avec le patron
  - –Une boîte noire
  - **–un objet**



Mémo : bonnes pratiques




```
<?php
class Langage
{
// zone jaune : déclaration d'une propriété
public $nomPropriete ;
// zone cyan : constructeur et destructeur
public function __construct() ;
public function __destruct() ;
// zone rouge : déclaration des accesseurs (get/set)
public function getNomPropriete()
{return $this→nomPropriete ;}
public function setNomPropriete($val)
{this→nomPropriete=$val ;}
// zone grise : déclaration des autres méthodes
public function __toString()
{ return "Je suis une instance de la classe".__CLASS__}
}
?>
```

Le fichier a pour nom Langage.class.php

Toutes les variables qui sont des propriétés de l'objet. Ex : nomLangage

get... pour renvoyer la variable
set... pour initialiser la variable

On appel ces méthodes des méthodes magiques
https://www.php.net/manual/fr/language.oop5.magic.php

Résumé: B.A.BA en POO

public : accès tout le monde à le droit
class : une classe ie un patron

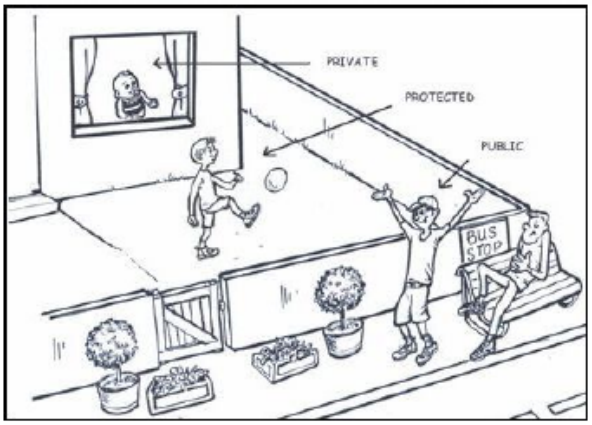
static : méthode ou variable de classe
=> pas de new pour créer une instance, il n'y a pas d'intance !
Instance (pas de mot clef static) : un exemple d'une classe, un objet en mémoire, une méthode de cet objet
=> new obligatoire pour créer l'objet

```
$unObjet= new Langage() ;
```

Le nom de la classe

Dépend de la déclaration du constructeur dans la classe.

Visibilité



- Un constructeur est toujours public.
- Par défaut en php l'accès est public.
- Il est préférable de ne pas déclarer des propriétés en public. On utilise les accesseurs (get/set) pour cela.

https://www.php.net/manual/fr/language.oop5.visibility.php

Mot-clé "final"

Le mot-clé `final` empêche les classes enfants de redéfinir une méthode ou constante en préfixant la définition avec `final`. Si la classe elle-même est définie comme finale, elle ne pourra pas être étendue.

Une classe qui fait rien

```php
<?php
class Rien
{
}

// Programme principal

$x=new Rien(); // appel constructeur par défaut
print_r($x); // par défaut __toString affiche le nom de la classe
?>_
```

Rien Object ( )

Deux classes, un héritage et une contrôleur pour les gérer.

```php
<?php
class Rien
{
    function __construct()
    {
        echo "Dans le constructeur de Rien<br/>";
    }
}
?>
```

```php
<?php
include ('Rien.class.php');
class RienOuPresque extends Rien
{
    function __construct()
    { parent::__construct();
      echo "fin de la construction RienOuPresque_class.php";
    }
}
?>
```

```php
<?php
require 'RienOuPresque.class.php';
echo "Héritage<br/>";
$y=new RienOuPresque();
print_r($y);
?>
```

__construct fonction magique, permet la construction de l'objet.
En php les constructeurs ne sont pas forcement appelés si on en défini au moins 1, il faut donc gérer l'appel du constructeur de Rien en premier, avant de faire celui de RienOuPresque
parent ::__construct() permet de le faire.

extends : hérite de..

include/require/include_once/require_once demande le chargement du fichier, sinon le code ne peut pas marcher.

```
Héritage
Dans le constructeur de Rien
fin de la construction RienOuPresque.class.phpRienOuPresque Object ( )
```

__autoload()

- Pour faire de la POO propre en PHP 5, il faut :
  - Définir une bibliothèque contenant la **fonction __autoload** qui se chargera de charger toutes les classes
  - Un fichier **.class.php** ne contient qu'UNE classe
  - Pour tester une classe, on fait un fichier **TestNomDeLaClasse.php** :
    - Avec le include de la bibliothèque autoload
    - Avec le programme principal de test
- Exemple :

autoload.inc.php

```php
<?php
if (defined("AUTOLOAD"))
{
    function __autoload($nomClasse)
    {
        $nomFichier=$nomClasse.".class.php";
        if (file_exists($nomFichier))
            // Charge la classe et stop le script si ne trouve pas
            require_once($nomFichier);
    }

}
?>
```

1

TroisFoisRien.class.php

```php
<?php
class TroisFoisRien extends PresqueRien
{
    private $val2=10;

    public function setVal2($valeur)
    {
        $this->val2=$valeur;
    }

    public function getVal2()
    {
        return $this->val2;
    }
}
?>
```

2

TestQuelqueChoseEnPlus.php

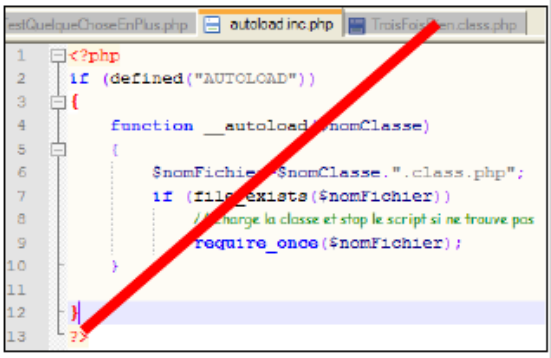
```php
<?php
define("AUTOLOAD",TRUE);
include ("autoload.inc.php");
$test=new QuelqueChoseEnPlus();
echo $test;
?>
```

3

```php
define("CHARGE_AUTOLOAD",true);
require_once("inc/poo.inc.php");
```

Dans le programme principal

- La fonction __autoload est obsolète.
- Il faut obligatoirement utiliser un gestionnaire d'événement **spl_autoload_register()** pour faire se travail à partir de la version 8
- Le code précédent ne change que pour autoload.inc.php



*poo.inc.php

```php
<?php

if (defined("CHARGE_AUTOLOAD"))
{ // Chargement de l'autoload

  set_autoload();
}
else
    die(""); // Arrêt en erreur, en mode silence


// fonction de chargement de classe en fonction de la version de php

function set_autoload()
{

   // cherche le numéro de version de PHP
   // Reprendre le code du manuel pour avoir PHP_VERSION_ID

   // PHP_VERSION_ID est disponible depuis PHP 5.2.7,
    // si votre version est antérieure, émulez-le.
    if (!defined('PHP_VERSION_ID')) {
       $version = explode('.',PHP_VERSION);

       define('PHP_VERSION_ID', ($version[0] * 10000 + $version[1] * 100 + $version[2]));
    }

    if (PHP_VERSION_ID>70100)
    {  // on passe par le gestionnaire d'évenement
       // sinon on aura l'erreur : Fatal error: __autoload() is no longer supported, use
spl_autoload_register() instead

        function my_autoloader($classname)
        {
            $filename = './class/' . $classname . '.class.php';
             if (file_exists($filename))
                include_once($filename);
                else
                  die("Erreur fichier inconnu : ".$filename);
        }

        spl_autoload_register('my_autoloader'); //PHP > 5

    }
```

Exemple d'une classe qui fait quelque chose : stockage d'un nombre

*Memoire.class.php

```php
<?
class Memoire
{
    private $aux;

    public function __construct($valDepart=0)
    {
        $this->setAux($valDepart);
    }

    public function setAux($nb)
    {
        $this->aux=$nb;
    }

    public function getAux()
    {
        return $this->aux;
    }

    public function inc()
    {
        $this->setAux($this->getAux()+1);
    }

    public function dec()
    {
        $this->setAux($this->getAux()-1);;
    }

}
?>
```

```php
$m=new Memoire();
$m->inc();$m->inc();
echo $m->getAux();
```

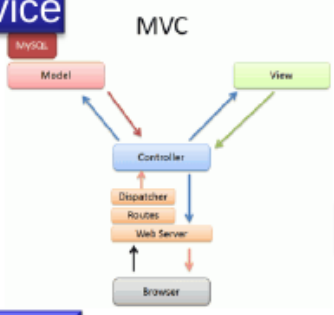
2

- Opérateur → permet d'accéder aux méthodes d'un objet et à ces propriétés
- **$this** est le pointeur sur l'objet courant

- Dans une architecture MVC, cette classe est un modèle. Elle permet d'encapsuler tout ce qui est nécessaire pour gérer le compteurs aux.

MVC : MODÈLE VUE CONTRÔLEUR sur le web

Vue service



Vue HTTP

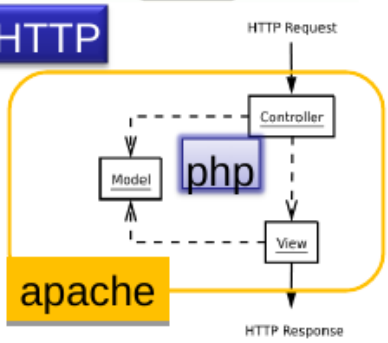
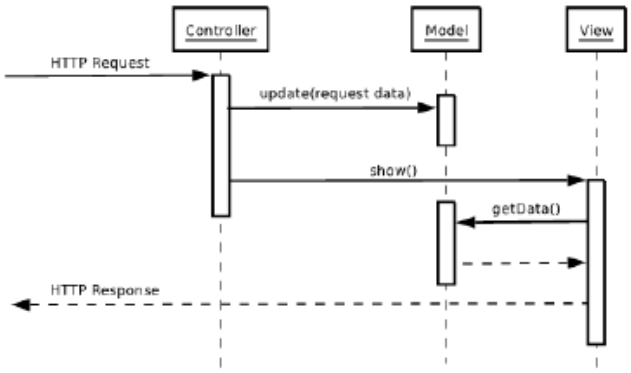


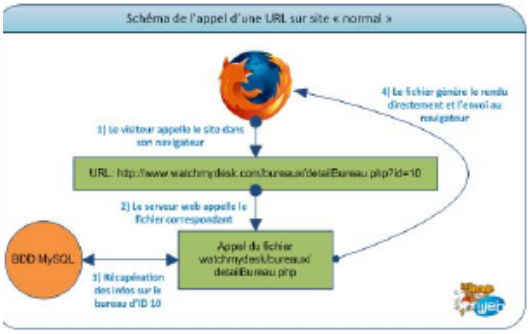
Diagramme des états : communication et messages



M V C en PHP

- M
  - » Le modèle est un ensemble de classes php de manipulation des données
  - » Encapsulation
- V
  - » La vue est un ensemble de classes php pour la présentation de composants html/css/js…
  - » Utilisation de __toString
- C
  - » Le contrôleur est généralement index.php
  - » Il sert aiguilleur aux traitements
  - » Il sert aussi de gendarme : vérification des flux

Organisation fichier=traitement



Encore utilisé pour
- Test et prototypage rapide d'une fonction
  - Pour voir comment elle marche
  - Pour apprendre à l'utiliser

C'est pas bien car :

Une modification dans le fichier
=> Il faut trouver ou est cette modification
=> Tout est mélangé : M=C=V
=> Bug/Répercutions à tous les niveaux

Pas réaliste à plusieurs
Pas réaliste dans le temps

Un code qui fait tout
= Si on a besoin de l'adapter, il faut le refaire

entièrement

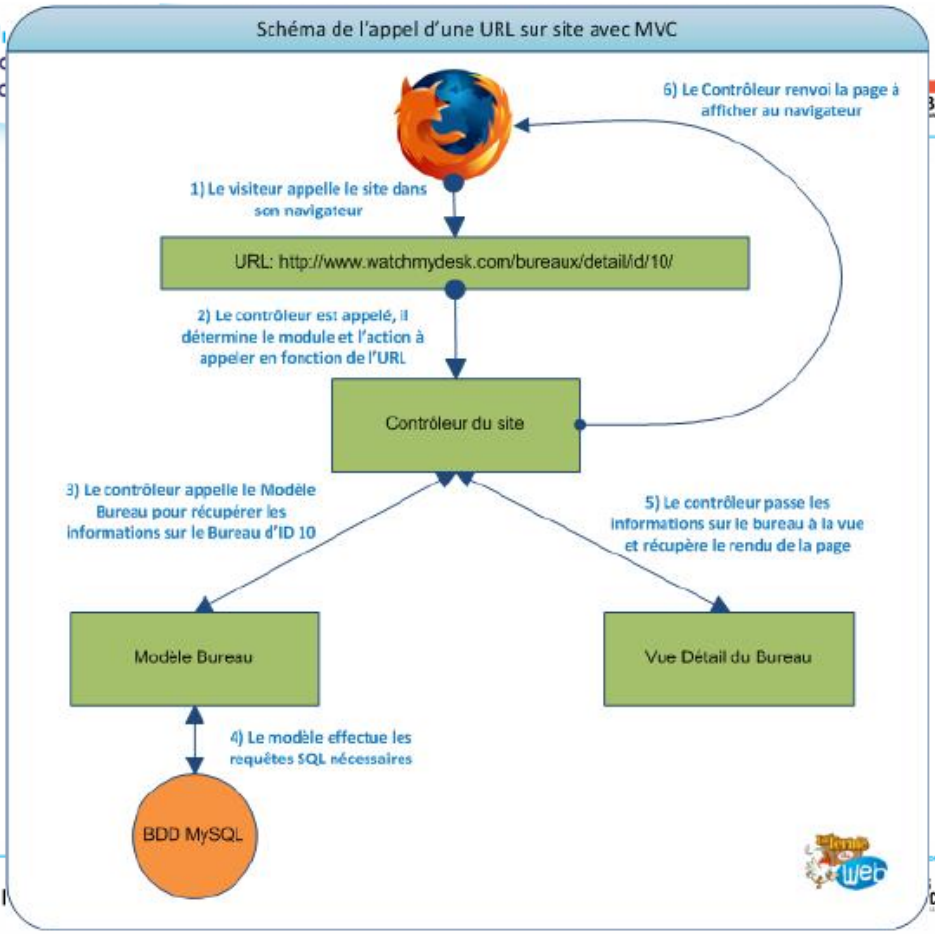
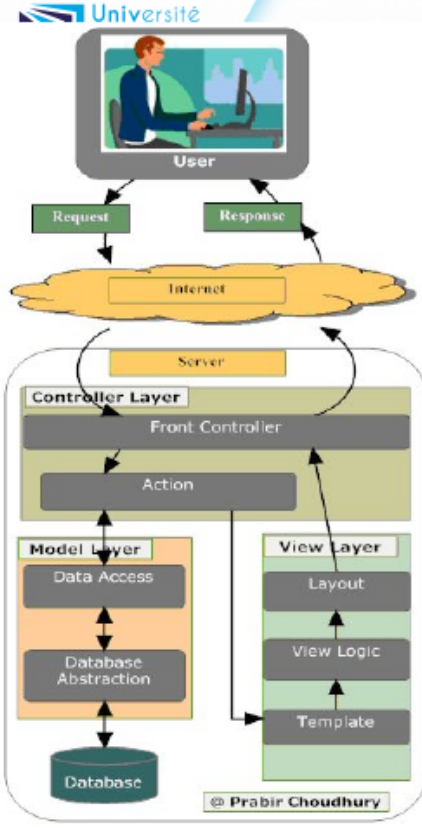
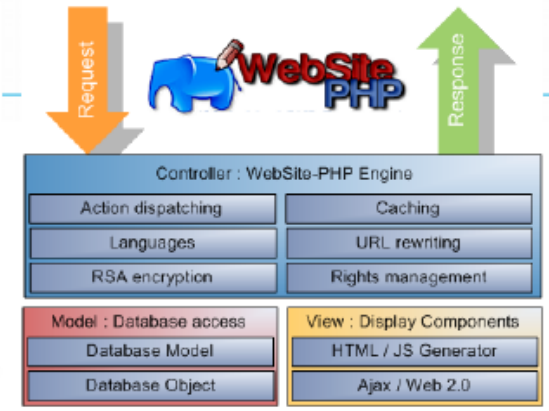


Schéma de l'appel d'une URL sur site avec MVC
6) Le Contrôleur renvoi la page à afficher au navigateur
1) Le visiteur appelle le site dans son navigateur
URL: http://www.watchmydesk.com/bureaux/detail/id/10/
2) Le contrôleur est appelé, il détermine le module et l'action à appeler en fonction de l'URL
Contrôleur du site
3) Le contrôleur appelle le Modèle Bureau pour récupérer les informations sur le Bureau d'ID 10
5) Le contrôleur passe les informations sur le bureau à la vue et récupère le rendu de la page
Modèle Bureau
Vue Détail du Bureau
4) Le modèle effectue les requêtes SQL nécessaires
BDD MySQL





La classe PDO

(PHP 5 >= 5.1.0, PHP 7, PHP 8, PECL pdo >= 0.1.0)

Introduction

Représente une connexion entre PHP et un serveur de base de données.

https://www.php.net/manual/fr/class.pdo.php

Etude de cas : le zoo MMI

Le Zoo MMI est constitué de 4 animaux,
Un animal dit par défaut « Je dis : bien! ». Il a aussi une couleur.
Le Chat est un animal du zoo. Il dit « Miaou ».
Le Chien est un autre un animal qu'on peut rencontrer. Il dit « Woof ».
Le Koala est le dernier animal du zoo, il dit « Bonne journée monsieur ».
Le zoo contient une liste de tous les animaux présents. Il a un chat blanc, un chien marron et un koala gris. Un dernier animal vient d'arriver mais la seule chose qu'on connaît dessus est qu'il est jaune.
Par défaut, on affiche la liste des animaux. Lorsqu'on clique sur un animal, on affiche sa couleur et le message « qu'il dit » puis de nouveau la liste.
On vous donne l'arborescence que doit respecter votre site.
Les modèles seront des classes. Vous utiliserez **spl_autoload_register** pour charger les classes.
index.php du répertoire racine vous permet d'avoir la liste des animaux et de les faire parler quand vous cliquez sur le lien dans la liste affichée.

```
nobecourt@ether:~/SRC/2021-2022/but1/R213/exemples$ tree
.
├── class
│   ├── Animal.class.php
│   ├── Chat.class.php
│   ├── Chien.class.php
│   ├── index.php
│   ├── Koala.class.php
│   └── Zoo.class.php
├── inc
│   ├── index.php
│   └── poo.inc.php
└── index.php

2 directories, 9 files
```

Animaux dans le zoo:

Chat
Chien
Koala
Animal

Le Koala Gris dit 'Bonne journée monsieur'

Animaux dans le zoo:

Chat
Chien
Koala
Animal