Classes et librairies PHP

Code: php-libs

Originaux

url: **http://tecfa.unige.ch/guides/tie/html/php-libs/php-libs**

url: **http://tecfa.unige.ch/guides/tie/pdf/files/php-libs.pdf**

Auteurs et version

- Daniel K. Schneider - Vivian Synteta
- Version: 0.5 (modifié le 16/1/06)

Prérequis

Module technique précédent: php-intro

Module technique précédent: php-html

Autres modules

Module technique suppl.: act-php-mysql

Module technique suppl.: visu-gen

Abstract

- La notion de “librairie” et d’API
- La notion de classe dans le contexte de PHP 4
- Utilisation de librairies
- ... il s’agit ici d’une ***première version*** ! (il faut notamment compléter le ch. 4)

Objectifs

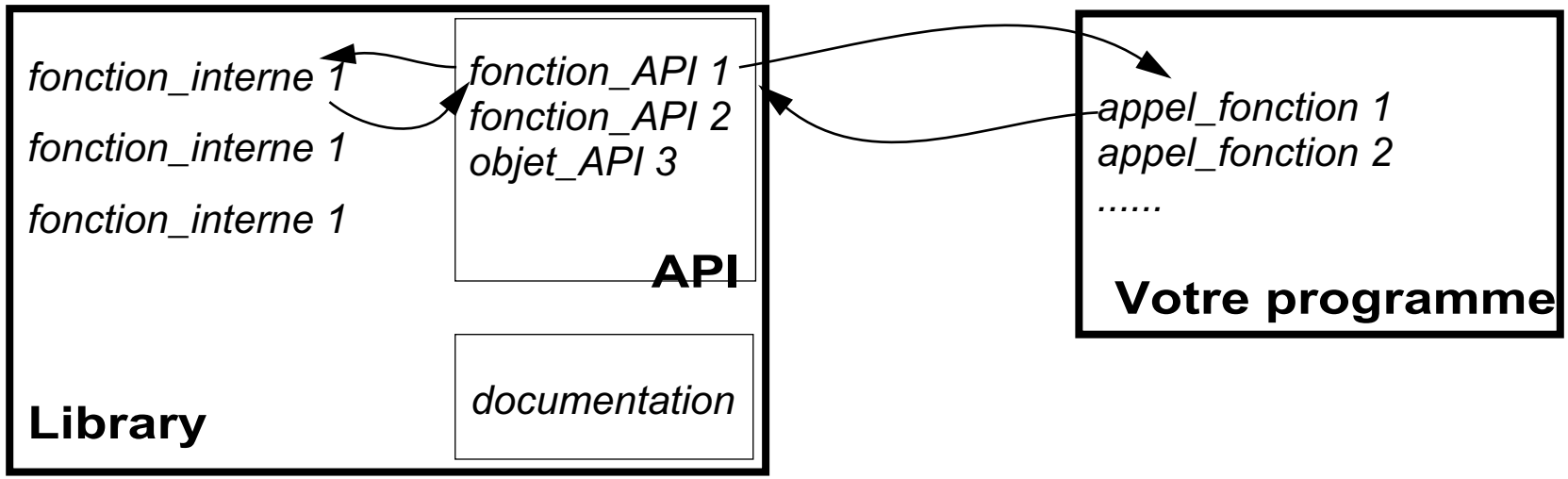
- Introduction à l’utilisation de code PhP “trouvé sur Internet”
- Introduction (douce et incomplète) à la programmation orientée objet

1. Table des matières détaillée

2. Classes et libraires PhP

2.1 Utilisation de librairies et d'API

- Une librairie est un programme permettant d'effectuer un certain nombre d'opérations. Autrement dit, une boite à outils que vous pouvez utiliser dans vos programmes (au lieu de tout programmer vous-même). Une librairie a une API.
- Un API (Application Programmer's Interface) est une liste de fonctions (ou procédures ou méthodes) et de variables (ou classes) documentées et que vous pouvez utiliser.
  - N'utilisez pas d'autres fonctions/variables que vous repérées dans le code, car elle risquent de ne plus marcher avec la prochaine mise à jour de la librairie.



A. Inclusion de librairies

- Normalement la documentation vous dit comment utiliser une librairie. Dans la doc de ADOdb (voir page 22) par exemple, on trouve le texte suivant:

  When running ADOdb, at least two files are loaded. First is adodb/adodb.inc.php, which contains all functions used by all database classes.
  `include('/path/to/set/here/adodb.inc.php');`
- Cela veut dire que pour utiliser adodb, il faut d'abord télécharger et installer la librairie quelque part et ensuite la "charger" dans votre fichier PhP avec la fonction PhP "include". Voici un exemple:

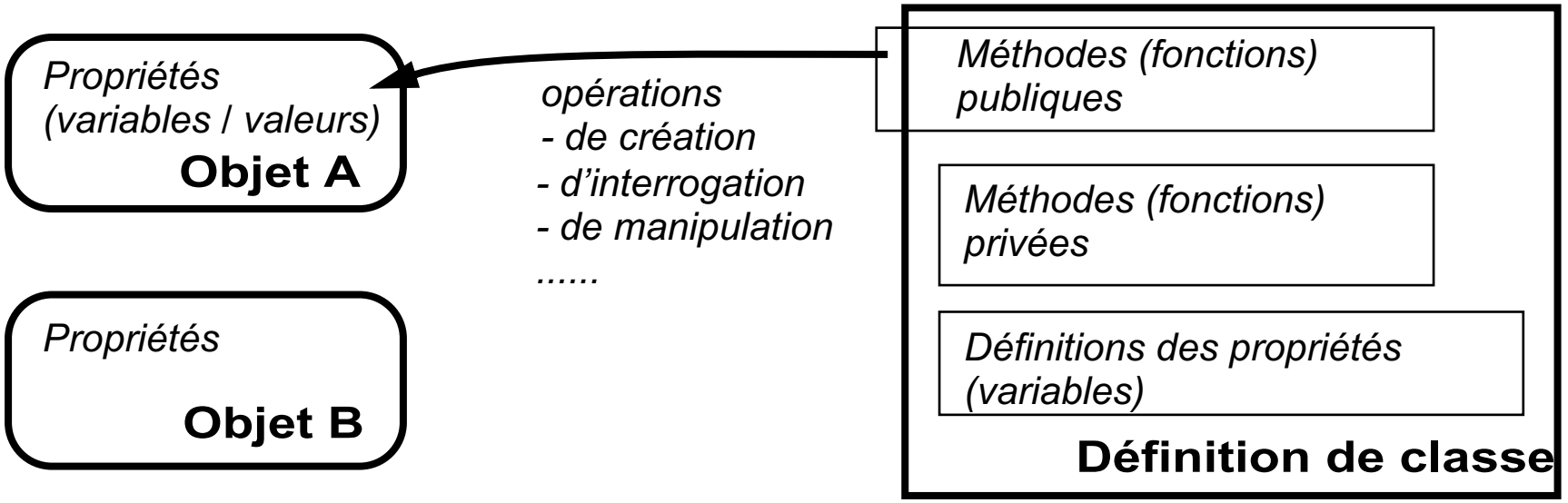
```
$adodb_path = "/web/lib/php/adodb";
include("$adodb_path/adodb.inc.php");
```

  - Note: On peut aussi installer des libraires "dans le système" et dans ce cas il aurait suffit de la chercher avec un simple `include("adodb/adodb.inc.php");`
- La fonction **`include`** (par défaut) charge un fichier depuis n'importe quel répertoire accessible à PHP, donc il faut indiquer un chemin relatif ou absolu du système !
- Variantes de la fonction **`include ()`**:
  - **`include_once()`** - évite d'inclure plus qu'une fois le même fichier lors d'une éxécution, variante souvent utilisée dans des systèmes larges (portails, etc.)
  - **`require()`** - arrête l'exécution de votre programme s'il y a erreur
  - **`require_once()`**

2.2 La programmation orientée objet

Définitions simplifiées (!!):

- Un objet est une structure informatique qui encapsule des informations sous forme de propriétés (variables internes) et que l'on peut manipuler avec des méthodes (fonctions) définies pour une classe d'objets.
- Une classe définit donc des variables et fonctions pour une classe d'objets.
- Comme pour les librairies, les classes définissent des fonctions ***publiques*** accessibles au programmeur-utilisateur (l'API de la classe), et des fonctions ***privées*** (internes) que l'utilisateur n'a pas besoin de connaître.
- Les ***objets*** contiennent les informations (pas les classes).



2.3 Un premier regard sur les objets en PhP 4

- Ici nous présentons un simple exemple (voir les explications 2 pages plus loin)
- Pour une classe "Students" nous définissons le "cahier des charges" suivant:
  - Elle définit un objet "étudiant" avec 2 propriétés: son nom (**`$name`**) et une liste d'une longueur aléatoire de notes (**`$scores`**)
  - On peut demander à un objet le ***nom de l'étudiant*** (fonction accesseur)
  - On peut ***ajouter des notes*** une par une avec une autre méthode
  - On peut demander à l'objet la note globale, c.a.d. de ***calculer la moyenne*** des notes.

Exemple 2-1: Une simple classe PHP

url: **http://tecfa.unige.ch/guides/php/examples/simple-objects/class-object.php**

```
class Student {

// propriétés de la classe
   var $name;
   var $scores;

// fonction "constructeur" pour créer l'objet (voir explication plus loin)
 function Student ($nom) { $this->name = $nom; }

// Fonction accesseur pour le nom de l'étudiant
 function getName () { return $this->name; }
```

```
// fonction accesseur pour les notes (sans tester s'il en a !)
  function getGrade () {
    $sum = 0;
    $n_scores = sizeof($this->scores);
    foreach ($this->scores as $score) {
      $sum += $score;
    }
    $grade = $sum / $n_scores;
    return $grade;
  }
 // accessor function pour insérer une note
  function addScore ($score) { $this->scores[] = $score; }

// end of Student class definition
}
```

Utilisation de la classe student:

```
// Création de 2 étudiants:
$nath = new Student("Nathalie Dupont");
$jule = new Student("Gilles Jules");

// Ajouter quelques notes à Nathalie:
$nath->addScore(4); $nath->addScore(5); $nath->addScore(3);
$nath->addScore(6);
// Imprimer sa moyenne:
echo "Grade obtained by ".$nath->getName(). ": " .$nath->getGrade();
```

2.4 Définition de la classe

- Toutes les définitions de propriétés (variables de la classe) et de méthodes (fonctions de la classe) sont faites à l'intérieur des { ... } [en tout cas on le conseille]

```
class Student{
....
}
```

- Pour créer un objet, aussi appelé instance on utilisera le mot clé "new" et un constructeur (voir E. "Constructeurs" [11])

```
new Student();
```

- Pour ne pas le "perdre" on peut le mettre dans une variable

```
$etudiant = new Student ();
```

A. Définition de propriétés (variables d'objets)

- On définit les propriétés (variables de la classe) avec le mot clé "var"
- Le nom de la variable commence par un $ suivi d'au moins une lettre

```
class Student{
  var $propriete_XXX;
  var $propriete_YYY
}
```

2.5 Définition de méthodes qui accèdent à une propriété

- On accède à une propriété d'un objet avec la construction $object_x->variable_xxx
  - La variable "$object_x" contient donc un objet
- "$this" veut dire qu'on demande à un objet de se référer à lui-même
- Voici un exemple simple qui retourne la valeur de la propriété "name":

```
function getName () {
  return $this->name;
}
```

Exemple d'utilisation:

```
$etudiant->getName();
```

Note: Il serait aussi possible d'accéder directement au nom avec **`$etudiant->name`**, mais ce type d'usage est fortement déconseillé, car il vaut mieux laisser une classe gérer sa méthode de stockage ! C'est le principe d'encapsulation, c.a.d le programmeur- utilisateur d'une classe devrait utiliser strictement les fonctions décrites dans l'API.

A. Définition de méthodes qui modifient une propriété

- Même principe que ci-dessus pour modifier une propriété
- Voici une simple méthode qui met à jour une propriété

```
function changeXXX ($valeur) {
  $this->propriete_XXX = $valeur
}
```

B. Constructeurs

- Un constructeur est une méthode qui crée une instance (objet) d'une classe. Cette méthode aura obligatoirement le même nom que la classe.
  - Note: En PHP on ne peut pas définir plusieurs constructeurs ayant des arguments différents (mais voir valeurs par défaut ci-dessous).
- Le constructeur suivant crée un objet "student" et définit son nom

```
function Student ($nom) {
  $this->name = $nom;
}
```

Exemple d'utilisation:

```
$etudiant = new Student ("David Taylor");
```

Le constructeur suivant crée un objet "student" avec une valeur par défault

```
function Student ($nom="Anonymous") {
  $this->name = $nom;
}
```

Exemple d'utilisation:

```
$etudiant = new Student ;  // un constructeur sans parenthèses utilise
                           // les valeurs par défaut !
```

C. Méthodes "ordinaires"

- Au-delà de ces exemples simples on utilisera les méthodes comme des fonctions "normales" avec la seule différence qu'on appelle toujours une méthode par le biais d'un objet.
- Au concepteur de décider lesquelles feront partie de l'API.
  - dans les "vraies langages" de programmation on peut définir des méthodes publiques et privés, mais pas en PhP 4.
- Voici une fonction qui calcule un score
  - (mais attention si l'étudiant n'a pas de scores, il va avoir une division / zéro et donc une erreur).

```
function getGrade () {
  $sum = 0;
  $n_scores = sizeof($this->scores);
  foreach ($this->scores as $score) {
    $sum += $score;
  }
  $grade = $sum / $n_scores;
  return $grade;
}
```

Exemple d'utilisation:

```
$etudiant->getGrade();
```

2.6 Utilisation d'une classe I

Exemple 2-2: Une simple classe PHP avec un include

url: **http://tecfa.unige.ch/guides/php/examples/simple-objects/student-class.phps**

url: **http://tecfa.unige.ch/guides/php/examples/simple-objects/student-objects.php**

url: **http://tecfa.unige.ch/guides/php/examples/simple-objects/**

- Cet exemple contient plus ou moins le même code.
  - Toutefois la classe est définie dans un fichier à part (student-class.php). Il s'agit de la même classe.
  - Elle est utilisé dans student-objects.php, fichier qui contient une 2ème couche d'abstraction, c.a.d. des fonctions pour utiliser la classe "students".

Voici des extraits de code du fichier student-objets.php

```
// inclusion de la classe
include ("student-class.php");
// On définit un array pour stocker les étudiants
$students=array();

// une fonction pour entrer des notes

function addScore ($id, $grade) {
  global $students;
  $students[$id]->addScore($grade);
 }
// une fonction pour afficher tous les résultats
function displayScores () {
  global $students;
  echo "<ol>";
```

```
  foreach ($students as $student) {
    echo "<li>Grade obtained by " . $student->getName() . ": "
         . $student->getGrade() . "</li>";    }
  echo "</ol>";
}
```

// Un constructeur pour créer des étudiants

```
function makeStudent ($id, $name) {
  global $students;
  $students[$id]  = new Student($id, $name);
}
```

// On crée des étudiants, ils sont mis dans le array indexé $students

```
makeStudent("nath", "Nathalie Dupont");
makeStudent("jules, "Gilles Jules");
makeStudent("steph", "Stephane Durand");
```

// on ajoute des notes ...

```
addScore("nath",4); addScore("nath",5); addScore("nath",3);
addScore("jules",2); addScore("jules",1); addScore("jules",3);
....
```

// Afficher les résultats

```
 displayScores ();
```

2.7 Utilisation d'une classe II

- Dans l'exemple précédant on avait une classe ("students") et quelques fonctions qui facilitent son usage (dans le fichier student-objects.php).
- Au fond, on peut aussi créer une classe pour gérer tous les étudiants qui suivent un cours, autrement dit inclure dans une nouvelle classe ces fonctions.

Exemple 2-3: Deux simples classes PHP

url: **http://tecfa.unige.ch/guides/php/examples/simple-objects/course-class.phps**

url: **http://tecfa.unige.ch/guides/php/examples/simple-objects/course-objects.php**

url: **http://tecfa.unige.ch/guides/php/examples/simple-objects/**

- La classe "Course" contient les méthodes publiques suivantes qu'un utilisateur pourra utiliser: **`Course()`**, **`makeStudent(), addScore(), et displayScores()`**
- Dans ce cas précis l'utilisateur n'est plus censé utiliser les fonctions de la classe "Students" qu'on a également repris dans le code.

```
$staf14 = new Course("Staf14 ..... ");
$staf14->makeStudent("nath", "Nathalie Morand");
$staf14->addScore("nath",4); $staf14->addScore("nath",5);
$staf14->addScores("jules",array(2,1,3,5));
$staf14->displayScores (); // Print the results
```

- Cet exemple montre bien qu'on peut avec une librairie simplifier la vie pour un programmeur d'application et donc pour un étudiant staf-14 ;)

3. Exemple d'installation / utilisation de la classe MiniPoll

- Cette librairie orientée objet permet d'organiser des petits sondages
- Elle nécessite l'accès à une base de données MySQL

  url: **http://www.free-midi.org/scripts/ (Site du développeur)**

  url: **http://www.phpclasses.org/browse/package/1453.html (un repositoire)**

Exemple 3-1: Demo de MiniPoll à TECFA

url: **http://tecfa.unige.ch/guides/php/examples/mini-poll/ (répertoire)**

url: **readme.txt (instructions de l'auteur qu'on va suivre)**

- Certaines librairies offrent une routine d'installation par le web.
  - Ici (et cela arrive souvent) ce n'est pas le cas et il faut faire l'installation manuellement de A-Z.
  - On va suivre plus ou moins les instructions de l'auteur

3.1 Installation des tables MySQL

- Dans le fichier readme.txt, l'auteur nous demande de créer 3 tables MySQL et il donne le code SQL.
- Il y a plusieurs façons de le faire (voir **mysql-intro.html**)
- Pour une petite application nous conseillons simplement de "plâtrer" les définitions SQL dans votre base de données à l'aide de phpMyAdmin. Concrètement il faut copier/coller le texte suivant:

```
# Table structure for table ‘poll_check‘

CREATE TABLE ‘poll_check‘ (
  ‘pollid‘ int(11) NOT NULL default ’0’,
  ‘ip‘ varchar(20) NOT NULL default ’’,
  ‘time‘ varchar(14) NOT NULL default ’’
) TYPE=MyISAM COMMENT=’’;

# --------------------------------------------------------

# Table structure for table ‘poll_data‘

CREATE TABLE ‘poll_data‘ (
  ‘pollid‘ int(11) NOT NULL default ’0’,
  ‘polltext‘ varchar(50) NOT NULL default ’’,
  ‘votecount‘ int(11) NOT NULL default ’0’,
  ‘voteid‘ int(11) NOT NULL default ’0’,
  ‘status‘ varchar(6) default NULL
) TYPE=MyISAM COMMENT=’’;

# --------------------------------------------------------

# Table structure for table ‘poll_desc‘

CREATE TABLE ‘poll_desc‘ (
  ‘pollid‘ int(11) NOT NULL default ’0’,
  ‘polltitle‘ varchar(100) NOT NULL default ’’,
  ‘timestamp‘ datetime NOT NULL default ’0000-00-00 00:00:00’,
  ‘votecount‘ mediumint(9) NOT NULL default ’0’,
  ‘STATUS‘ varchar(6) default NULL,
  PRIMARY KEY (‘pollid‘)
) TYPE=MyISAM COMMENT=’’;
```

3.2 Le fichier de configuration

- La plupart des librairies ont un fichier de configuration qui demande d'indiquer les paramètres suivants:
  - le nom de la base de données, utilisateur MySQL, mot de passe, le nom de la machine "hôte" ("localhost" si MySQL est sur la même machine que le serveur Web)
  - Souvent (mais pas ici !) il faut également indiquer où se trouvent les fichiers et quel URL vous allez utiliser pour accéder à votre dispositif.
- Editer le fichier de configuration "config.php"
  - Note: Dans le fichier readme.txt l'auteur ne dit pas quel fichier il faut éditer, mais il est souvent sous-entendu qu'un tel fichier s'appelle "config.php" ou encore "config.inc.php".
  - Note: Dans la version téléchargé de phpclasses.org la structure des répertoires est cassée (les fichiers sont à plat), et les instructions de l'auteur ne collent pas.
- Configuration retenu (que vous devez changer pour vous):

```
// config.php
$host = "localhost"; // db host
$user = "nobody"; // db username
$pass = ""; // db password
$db = "demo"; // db name
```

3.3 Installation / utilisation de la classe

- Selon l'auteur (point 3 de readme.txt), vous avez 2 possibilités: soit vous suivez les instructions, soit vous adaptez les fichiers exemples.
- Pour tester les classes, nous suggérons d'utiliser les fichiers exemples, et ensuite d'incorporer leur code dans vos propres pages *.php.
- En gros, l'utilisation de la classe est assez simple:

Pour afficher un "poll":

```
include_once ("includes/miniPoll.class.php");
include_once ("config.php");

$connection = mysql_connect ($host, $user, $pass) or die ("Unable to connect");
mysql_select_db ($db) or die ("Unable to select database");

$test = new miniPoll;

$test->pollForm();

@mysql_close($connection);
```

- Attention si vous changez les fichiers d'endroit, il faut:
  - (a) changer les instructions PHP "include_once (" ...")
  - (b) changer les lignes $this->results_page dans les classes. (voir points 6 et 7)

3.4 Sécuriser !

- Le fichier "test_poll_admin.php" est accessible à tout le monde. On suggère de le mettre dans un répertoire protégé par un mot de passe "serveur Web".
  - Sur un serveur "Apache", la façon la plus simple est de copier un fichier .htaccess que vous utilisez déjà qq. part ailleurs. (A Tecfa par exemple celui qui protège les corrections STAF-14).
  - Sinon, lire le manuel Apache ou chercher "htaccess tutorial" sur Google

4. Librairies / classes populaires

(chapitre à compléter un jour ....)

4.1 Comment trouver / comment choisir ?

- Il existe plusieurs repositoires de bonne qualité, en ce qui concerne les classes PhP (donc pas les applications larges), nous conseillons:

  url: **http://www.phpclasses.org/browse/**
- La plupart des repositoires offrent un "rating" (ou plusieurs): popularité, sondages sur la qualité, etc. En tenez compte !
- Dans la suite on présentera brièvement quelques grandes classes connues qui sont souvent utilisés également dans des applications plus larges comme des portails.

4.2 La classe phpHtmlLib

- phpHtmlLib est une grande librairie qui vous assiste à la fabrication de pages (X)Html, SVG, etc.

  url: **http://phphtmllib.newsblob.com/**

Exemple 4-1: SVG avec phpHtmlLib

url: **http://tecfa.unige.ch/guides/php/examples/svg-phphtmllib/**

4.3 La classe ADOdb

- ADOdb est une classe populaire dans le domaine des bases de données

  url: **http://php.weblogs.com/adodb**
- Elle permet d'écrire du code assez portable entre différentes types de bases de données SQL (MySQL, Oracle, Acess, etc.)
- Autrement dit, PHP malheureusement définit des fonctions très différentes suivant le type de base de données qu'on utilise, donc en choisissant de coder spécifiquement pour MySQL vous rendez votre code très peu portable.