MÃ©mento du dÃ©veloppeur (API XOOPS)

CatÃ©gorie : Fiches techniques

PubliÃ© par Fooups le 16/04/2005

API de xoops

Une API (Application Programming Interface), que l'on peut traduire par Interface de Programmation d'Application, est une bibliothÃªque de classes et de fonctions Ã Â destination du dÃ©veloppeur.
L'application dÃ©veloppÃ©e - un module dans le cas de xoops - est ainsi rendue indÃ©pendante de diffÃ©rents paramÃªtres tels que la base de donnÃ©es, le navigateur ou l'OS du serveur. La programmation est Ã©galement facilitÃ©e puisqu'il devient inutile de coder ce qui l'est dÃ©jÃ Â dans cette API. Enfin une meilleure perennitÃ© de l'application - du module - est assurÃ©e, car en cas d'Ã©volution de Php p.ex. l'API intÃ©grera les modifications nÃ©cessaires, et en principe Ã©vitera d'avoir Ã Â modifier le module.
Ce guide a pour but de fournir une documentation sur les diffÃ©rentes classes et fonctions de Xoops, sous une forme synthÃ©tique.

Vous ne trouverez pas d'explications dÃ©taillÃ©es, mais l'essentiel Ã Â connaitre pour l'utilisation de ces fonctions ou mÃ©thodes de classe avec : ->un bref descriptif ->les paramÃªtres requis ou optionnels ->des exemples chaque fois que possible

Convention de notation

Les fonctions seront reprÃ©sentÃ©es de la maniÃªre suivante

getFirstChild (int $sel_id, [string $order = ''])

Avec entre paranthÃªses, le(s) paramÃªtre(s) Ã Â tranmettre, entre crochets lorsqu'ils sont optionnels chaque paramÃªtre Ã©tant prÃ©cÃ©dÃ© du type de donnÃ©e. *Ce guide sera complÃ©tÃ© progressivement et si vous souhaitez apporter votre contribution Ã Â sa rÃ©daction, vous serez les bienvenus. Il vous sera seulement demandÃ© de respecter la mise en forme pour des question d'homogÃ©nÃ©itÃ© des documents*.

Classes de Xoops

xoopsDB	AccÃªs Ã Â la base de donnÃ©es
xoopsList	DiffÃ©rentes listes disponibles (langues, pays, etc.)
xoopsMailer	Envoi de mail
xoopsMediaUploader	Upload de fichiers
xoopsModule	Gestion des modules
xoopsObject	Classe de base pour la crÃ©ation de classes dÃ©rivÃ©es
xoopsPageNav	Pour la crÃ©ation d'une navigation entre pages
xoopsTree	Gestion de catÃ©gories/sous-catÃ©gories
xoopsTpl	Classe dÃ©rivÃ©e de la classe Smarty pour l'affichage par templates
xoopsUser	Gestion des utilisateurs

Fichiers

functions.php	Fonctions utilisables par tout module
xoops_version.php	Fichier de dÃ©finition des caractÃ©ristiques du module

Formulaires

Classes de formulaire CrÃ©ation du formulaire
ElÃ©ments de formulaire (1) ElÃ©ments hidden, text, label, button, file, etc.

ElÃ©ments de formulaire (2) ElÃ©ments radio, checbox et select

Classe xoopsDB

Connaissances requises : tableaux (associatifs, indexÃ©s) et instructions mysql de Php.

prefix genId queryFromFile
query getInsertId freeRecordSet
fetchRow getRowsNum errno
fetchArray getAffectedRows error
fetchBoth quoteString

A utiliser pour toutes les instructions relatives aux bases de donnÃ©es.
Assure la connection Ã Â la base de donnÃ©es. Permet de bÃ©nÃ©ficier du mode debug sql.
N'a pas besoin d'Ãªtre instanciÃ©e (fait par le fichier common.php), sauf dans une classe.

Utilisation dans une fonction :

Si utilisation dans une fonction ne pas oublier la dÃ©claration globale :

```
function myfunction($param=0) {
    global $xoopsDB;   Utilisation dans une classe :
```

Dans ce cas, doit Ãªtre instanciÃ©e :

```
 class myClass extends XoopsObject
    {
    var $db;       // constructor
      function myClass($id=null)
      {
      $this->db =& Database::getInstance();
      $this->initVar("id", XOBJ_DTYPE_INT, null, false);
      // etc
```

---

prefix

Retourne le nom de la table passÃ©e en paramÃªtre avec le prefixe choisi pour les tables de xoops (xoops par dÃ©faut) prefix (string $tablename)

Exemple

$xoopsDB->prefix('matable') retournera 'xoops_matable'

---

query, queryF

Execution de la requÃªte $sql passÃ©e en paramÃªtre (chaine de caractÃ¨res).

prefix (string $sql, [int $limit = 0}, [int $start = 0])

$sql	string	requÃªte sql
$limit	int	nombre de lignes Ã Â retourner
$start	int	offset : dÃ©calage par rapport au premier enregistrement

Exemple

```
$sql = "SELECT cid, title, imgurl FROM ".$xoopsDB->prefix("mydownloads_cat")."  WHERE pid = 0 ORDER BY title";
$result=$xoopsDB->query($sql);
```

Remarque:

xoopsDB->query doit Ãªtre utilisÃ© par dÃ©faut, xoopsDB->queryF ne devant Ãªtre utilisÃ© que pour les requÃªtes INSERT ou UPDATE en cas de problÃªme.

---

fetchRow

fetchRow($result) Ã©quivaut Ã Â mysql_fetch_row($result)

Retourne une ligne du rÃ©sultat de la requÃªte sous forme de tableau indexÃ© numÃ©riquement.

Retourne false si pas (ou plus) de ligne.

Exemples

- lorsque la requÃªte ne retourne qu'une ligne (requÃªte SELECT COUNT(*) , requÃªte avec clause WHERE sur l'id,....)

```
list($count) = $xoopsDB->fetchRow($result);
// ou encore
list($id, $name, $phone) = $xoopsDB->fetchRow($result);
```

- lorsque les rÃ©sultats peuvent Ãªtre exploitÃ©s directement dans une boucle

```
while (list($id, $name) = $xoopsDB->fetchRow($result)) {
echo 'Le nom est '.$name.' et le numero id '.$id.'
';
}
```

---

fetchArray

fetchArray($result) Ã©quivaut Ã Â mysql_fetch_assoc($result)

Retourne une ligne du rÃ©sultat de la requÃªte sous forme d'un tableau associatif, la clef Ã©tant le nom du champ.

Retourne false si pas (ou plus) de ligne. Exemples

- lorsque la requÃªte ne retourne qu'une seule ligne

```
$myrow = $xoopsDB->fetchArray($result)) {
$variable1 = $myrow['nom_champ1'];
$variable2 = $myrow['nom_champ2'];
}
```

- si une ligne ou plus

```
$i = 0;
while ( $myrow = $xoopsDB->fetchArray($result) ) {
$var_array[$i]['elt1'] = $myrow['nom_champ1'];
$var_array[$i]['elt2'] = $myrow['nom_champ2'];
$i++;
}
```

- ou encore

```
while ($myrow = $xoopsDB->fetchArray($result)) {
$var_array['element'][] = array('elt' => $myrow['nom_champ1'], 'elt2' => ($myrow['nom_champ2']));
}
```

---

fetchBoth

fetchBoth($result) Ã©quivaut Ã Â mysql_fetch_array($result, MYSQL_BOTH)

Retourne une ligne du rÃ©sultat de la requÃªte sous forme d'un tableau associatif et indexÃ© numÃ©riquement.

Retourne false si pas (ou plus) de ligne.

getInsertId

getInsertId($db) Ã©quivaut Ã Â mysql_insert_id($db).

Retourne l'identifiant gÃ©nÃ©rÃ© par la derniÃ¨re requÃªte INSERT Exemple d'utilisation dans une fonction d'enregistrement dans la Bdd

```
function store($id=null) {
        if (empty($id)) {
    // si $id est vide c'est un nouvel enregistrement
    $id = $xoopsDB->genId('matable_id_seq');
    $sql = 'INSERT INTO '.$xoopsDB->prefix('matable').'  (id, title, message) VALUES ('.$id.', '.$xoopsDB->quoteString($title).',  '.$xoopsDB->quoteString($message).')';
    } else {
    // sinon c'est une mise Ã Â  jour
    $sql = 'UPDATE '.$xoopsDB->prefix('matable').' SET title='.$xoopsDB->quoteString($title).',  message='.$xoopsDB->quoteString($message)..' WHERE id='.$id;
    }
    if (!$result = $xoopsDB->queryF($sql)) {
    return false;
    }
    if (empty($id)) {
    $id =$xoopsDB->getInsertId();
    }
    // on retourne false si la requÃªte Ã Â  Ã©chouÃ©,  sinon on retourne l'id
    return $id;
    }
```

---

getRowsNum

getRowsNum($result) Ã©quivaut Ã Â mysql_num_rows($result)

Retourne le nombre de lignes d'un rÃ©sultat provenant d'une requÃªte SELECT.

Permet de vÃ©rifier que l'on a au moins un rÃ©sultat, ou de compter le nombre de lignes.

Exemple de vÃ©rification d'au moins une ligne de rÃ©sultat

```
if ( !$row = $xoopsDB->getRowsNum($result) ) {
  redirect_header( "index.php", 1, _AM_NOEXIST ); // aucun rÃ©sultat Ã Â  afficher, on sort
  exit();
  }
```

---

getAffectedRows

getAffectedRows($db) Ã©quivaut Ã Â mysql_affected_rows($db).

Retourne le nombre de lignes affectÃ©es par une requÃªte INSERT, DELETE ou UPDATE.

---

quoteString

Retourne la chaine de caractÃ¨res passÃ©e en paramÃ¨tre avec des quotes simples. A utiliser systÃ©matiquement pour toute chaine de caractÃ¨res avant enregistrement dans la Bdd. Exemple dans une requÃªte de mise Ã Â jour

---

$sql = 'UPDATE '. $xoopsDB->prefix('xoopsheadline').' SET headline_name='. $xoopsDB->quoteString($headline_name).',

---

freeRecordSet

freeRecordSet() Ã©quivaut Ã Â mysql_free_result($result).

LibÃ©ration de la mÃ©moire utilisÃ©e par $result.

MentionnÃ©e pour mÃ©moire.

---

error

error() Ã©quivaut Ã Â mysql_error().

Retourne le message d'erreur gÃ©nÃ©rÃ© par la derniÃªre requÃªte, ou chaine vide si pas d'erreur. errno

errno() Ã©quivaut Ã Â mysql_errno()

Retourne le numÃ©ro de l'erreur gÃ©nÃ©rÃ© par la derniÃªre requÃªte.

---

queryFromFile

Execute la requÃªte contenue dans le fichier $file passÃ© en paramÃªtre (fichier dump SQL).

UtilisÃ© en particulier pour l'installation (ou l'upgrade) de Xoops. Voir fichier install/index.php. queryFromFile (string $file)

---

genId

GÃ©nÃ©ration d'un id avant insertion d'un enregistrement.

Pour la compatibilitÃ© avec d'autres bases (pas d'action avec mysql : auto-incrÃ©ment)

genId(string $sequence)

$sequence string composÃ©e du nom de la table, de l'id et de 'seq', sÃ©paration des mots par underscore Exemple

```
$headline_id = $xoopsDB->genId('xoopsheadline_headline_id_seq');
$sql = 'INSERT INTO '. $xoopsDB->prefix('xoopsheadline').' (headline_id, ....
```

Retour au sommaire

Version Xoops de rÃ©fÃ©rence: 2.06

C. Felix AKA theCat le 01/06/04

Classe xoopsObject

Fichier kernel/object.php

initVar isNew setErrors

getVar setNew getErrors

getVars unsetNew getHtmlErrors

assignVar

assignVars isDirty registerFilter

cleanVars setDirty clone

setVar unsetDirty

Classe de base dont sont dÃ©rivÃ©es toutes les classes du kernel (xoopsComment, xoopsGroup, xoopsUser,....)

Permet de crÃ©er des classes dÃ©rivÃ©es, hÃ©ritant des variables et mÃ©thodes.

Types de donnÃ©es

ParamÃªtre pour initVar Type de donnÃ©e ParamÃªtres format pour getVar TestÃ© *

XOBJ_DTYPE_TXTBOX DonnÃ©e de type texte s, show, e ,edit, f, formpreview, n, none OK

XOBJ_DTYPE_TXTAREA Zone de texte s, show, e ,edit, f, formpreview, n, none OK

XOBJ_DTYPE_INT Entier (integer) OK

XOBJ_DTYPE_URL Url OK

XOBJ_DTYPE_EMAIL Email OK

XOBJ_DTYPE_ARRAY Tableau ?

XOBJ_DTYPE_OTHER Autres (type mime, rÃ©pertoire, fichier,...) OK

XOBJ_DTYPE_SOURCE Source s, show, e ,edit, f, formpreview, n, none OK

XOBJ_DTYPE_STIME Date/time ?

XOBJ_DTYPE_MTIME Date/time ?

XOBJ_DTYPE_LTIME Date/time ? * TestÃ© : type de donnÃ©e testÃ© personnellement et/ou utilisÃ© par le core ou modules officiels.

initVar

Initialisation de la variable dont le nom est passÃ© en paramÃªtre

initVar (string $key, int $data_type, [mixed $value = null], [bool $required = false], [int $maxlength = null], [mixed $options = ''], string $option)

$key string nom

$data_type int type de donnÃ©e: dÃ©termine le 'text sanitizing'

$value 'mixed' Valeur par dÃ©faut

$required bool DonnÃ©e obligatoire ou non

$maxlength int Texte uniquement:longueur du champ

$option 'mixed' Option

Exemple classe XoopsUser

```
function XoopsUser($id = null)
{
$this->initVar('uid', XOBJ_DTYPE_INT, null, false);
$this->initVar('name', XOBJ_DTYPE_TXTBOX, null, false, 60);
$this->initVar('email', XOBJ_DTYPE_TXTBOX, null, true, 60);
$this->initVar('url', XOBJ_DTYPE_TXTBOX, null, false, 100);
$this->initVar('user_sig', XOBJ_DTYPE_TXTAREA, null, false, null);
$this->initVar('theme', XOBJ_DTYPE_OTHER, null, false);
$this->initVar('timezone_offset', XOBJ_DTYPE_OTHER, null, false);
```

setVar

Assigne une valeur Ã Â la variable setVar (string $key, mixed $value, [bool $not_gpc = false])

$key string nom de la variable
$value mixed valeur Ã Â assigner
$not_gpc bool

Exemple de register.php

```
$newuser->setVar('name', $name);
```

getVar

Retourne la valeur de la variable au format demandÃ©. getVar (string $key, [string $format = 's'])

Exemple avec $xoopsUser

```
$xoopsUser->getVar('name')
```

getVars

Retourne toutes les valeurs des variables de l'object sous forme d'un tableau associatif des paires clef => valeur. getVars()

assignVar

Assigne la valeur passÃ©e en paramÃªtre Ã Â la variable

assignVar (string $key, mixed $value)

Exemple (tirÃ© de la fonction insert de la classe xoopsUser)

```
$user->assignVar('uid', $uid);
```

assignVars

Assigne le tableau passÃ© en paramÃªtre Ã Â la variable

assignVars(array $var_arr)

Exemple classe xoopsUser: (assignation Ã Â $user des valeurs extraites de la base)

```
function &get($id)
{
$sql = 'SELECT * FROM '.$this->db->prefix('users').' WHERE uid='.$id;
$numrows = $this->db->getRowsNum($result);
$user = new XoopsUser();
$user->assignVars($this->db->fetchArray($result));
return $user;
}
```

cleanVars

PrÃ©paration de toutes les variables de l'object pour enregistrement dans la Bdd, en fonction du type de donnÃ©es (text sanitize).

Message d'erreur si valeur absent