

# Initiation à PHP-GTK 1



par Xaviéra Autissier Jean-Marc Richon

Date de publication : 5 juin 2007

Dernière mise à jour :

Dans cet article nous allons voir ce qu'est PHP-GTK, son installation et expliquer succinctement son utilisation en créant une application simple.

- I - Présentation de PHP-GTK
  - I-A - PHP
  - I-B - GTK+
  - I-C - PHP-GTK
- II - Installation de PHP-GTK
  - II-A - Installation sous Linux
  - II-B - Installation sous Win32
  - II-C - Test de l'installation
- III - Description et réalisation d'un exemple
  - III-A - Chargement de la librairie GTK et notre premier widget : GtkWindow
  - III-B - Ajout d'une fonction : destroy()
  - III-C - Aspect de notre fenêtre GtkWindow
  - III-D - Ajout d'un bouton : GtkButton
  - III-E - Ajout de la fonction hello\_dvp()
  - III-F - Ajout d'un widget "tooltips" GtkTooltips
  - III-G - La version finale de notre script
- IV - Conclusion
- V - Annexes
  - V-A - Code objet
  - V-B - Quelques liens sur PHP-GTK
  - V-C - Remerciements

## I - Présentation de PHP-GTK

### I-A - PHP

PHP est principalement conçu pour servir de langage de script. Il est exécuté côté serveur, ce qui fait qu'il est capable de réaliser tout ce qu'un script CGI quelconque peut faire, mais PHP peut en faire bien plus.

#### Il y a trois façons différentes d'utiliser PHP

- **Langage de script côté serveur.** C'est l'utilisation la plus traditionnelle, et aussi le principal objet de PHP (pages ou applications Web). Vous aurez besoin de trois composants pour l'exploiter : un analyseur PHP (CGI ou module serveur), un serveur Web et un navigateur Web.
- **Langage de programmation en ligne de commande.** Vous pouvez écrire des scripts PHP et les exécuter en ligne de commandes, sans l'aide d'un serveur web et d'un navigateur (scripts de "cron" (Linux) ou task manager (Win32), opérations sur des fichiers *etc.*). Il vous suffit pour cela de disposer de l'exécutable PHP.
- **Écrire des applications clientes graphiques.** Cela nécessite de bien connaître PHP mais, si vous souhaitez exploiter des fonctionnalités avancées et graphiques dans vos applications clientes, vous pouvez utiliser PHP-GTK.

C'est à cette 3<sup>e</sup> partie que nous allons nous intéresser dans ce tutoriel.

PHP-GTK est une extension de PHP, qui n'est pas fournie dans la distribution de base.

### I-B - GTK+

GTK+ est un ensemble de bibliothèques écrites en C et développées pour le logiciel graphique GNU GIMP (GIMP Tool Kit). Les linuxiens connaissent bien GTK, il est le coeur de GNOME, un environnement graphique Linux. Il est aussi très connu dans le monde du développement en Langage C car GTK permet de créer des applications graphiques.

#### Dans cet ensemble de bibliothèques, nous trouvons :

- **Glib** : Permet d'accéder à divers outils nécessaires à la programmation en GTK+ ;
- **GDK** : Permet la programmation bas niveau de dessin sur les fenêtres (modification de la couleur, changement de police de caractères *etc.*) ;
- **GTK** : Donne accès aux objets graphiques (widgets) permettant la programmation de l'interface utilisateur (boutons, fenêtres *etc.*).

### I-C - PHP-GTK

L'alliance de PHP et GTK+ va permettre de créer des interfaces graphiques telles que nous en voyons dans des applications générées en Visual Basic, C++, Java ou Delphi. PHP-GTK possède encore d'autres avantages. Les bibliothèques de GTK+ ayant été portées de Linux vers Win32 ainsi que Mac OS, nous avons donc une portabilité optimale.




*PHP-GTK est multi plateformes.*

Un autre avantage est que nous pouvons faire tourner nos applications sans serveur Web (Apache, IIS, Websphere *etc.*). Celles-ci tourneront en mode "stand alone" sur le poste ou elles seront installées. Nous aurons alors, toute la puissance du langage PHP sans avoir la contrainte du serveur Web.

 *PHP-GTK tourne au niveau client et non plus au niveau serveur.*


Du fait que PHP-GTK se dissocie du monde Web, l'association des deux technologies donne ses lettres de noblesse au langage PHP. PHP entre dans le monde des langages de programmation au même titre que Visual basic, C, C++, Pascal avec DELPHI et Java. Il y a toutefois un bémol, vu la limite du langage PHP, nous ne pourrions pas réaliser de grosses applications comme c'est le cas en C, C++, Delphi et Java.

 *PHP-GTK peut être une bonne alternative à Visual Basic pour le développement de petites et moyennes applications.*

## II - Installation de PHP-GTK

Notez que nous allons utiliser la dernière version stable de PHP-GTK qui est la version 1.0.2. Cette version est prévue pour PHP 4. À l'heure où nous écrivons ce tutoriel, une version de GTK2 est en cours d'élaboration (version alpha) qui supportera le langage objet de PHP5. Pour l'instant nous ne pourrions utiliser que le "pseudo objet" de PHP4.

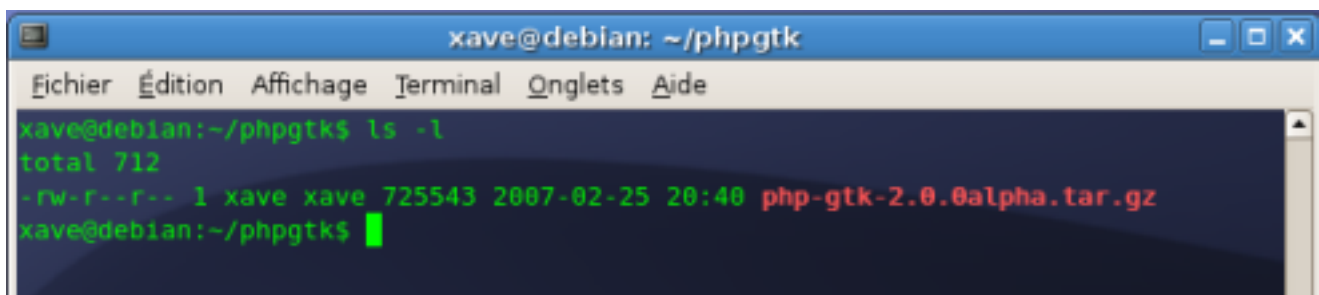
### II-A - Installation sous Linux

 *La version php-gtk 1.0.2 générant beaucoup d'erreurs à l'installation sous linux, nous avons choisi d'installer la version php-gtk 2.0. Avec cette version, vous pourrez suivre cet article même si certaines fonctions ne sont plus à utiliser (deprecated) dans la version php-gtk 2.0.*

**Nous avons testé l'installation de php-gtk sur linux DEBIAN ETCH avec le système standard suivant :**

- Le noyau 2.6.18 ;
- L'installation du kernel source + headers ;
- Le driver officiel NVIDIA ;
- Xorg ;
- L'outil gcc avec la libc6 ;
- Gnome ;
- Les outils de développement Gnome incluant GTK+2.X.

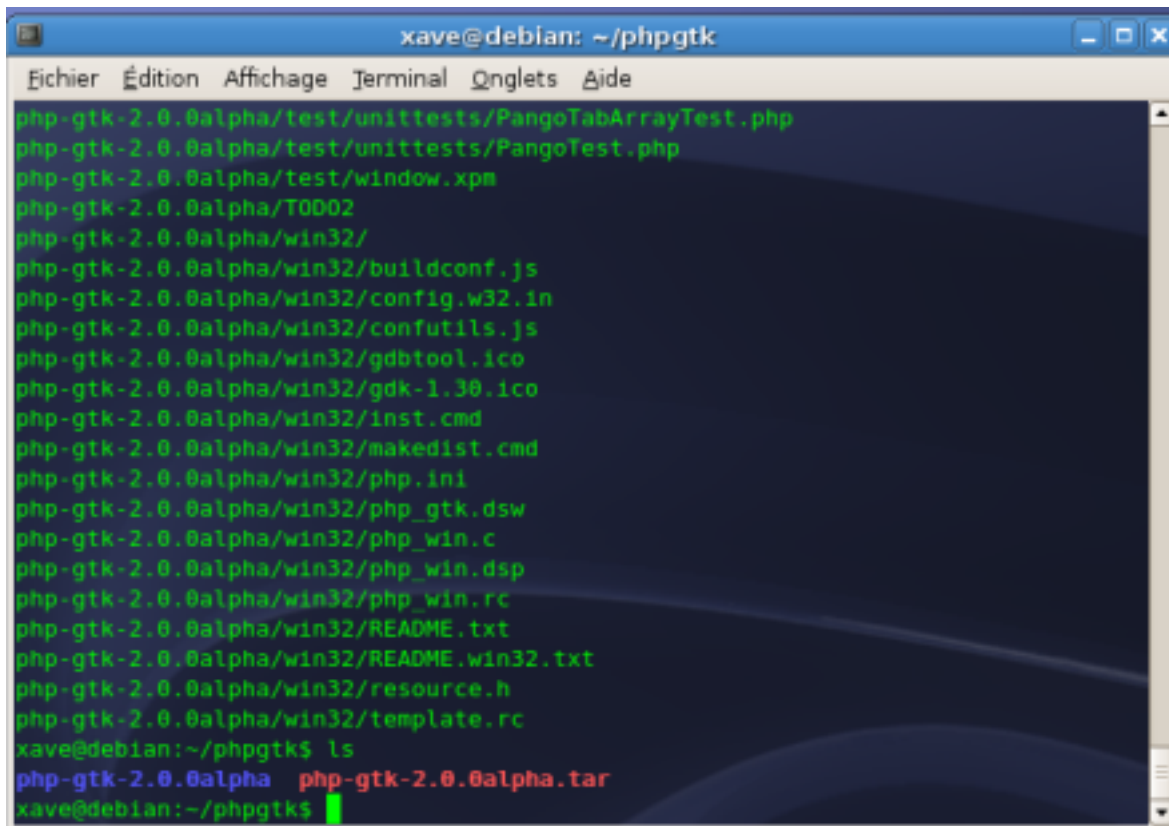
Téléchargez le paquetage **php-gtk-2.0alpha.tar.gz** et placez-le dans un dossier **phpgtk** que vous aurez créé dans le répertoire **home** de l'utilisateur.



```
xave@debian: ~/phpgtk
Fichier Édition Affichage Terminal Onglets Aide
xave@debian:~/phpgtk$ ls -l
total 712
-rw-r--r-- 1 xave xave 725543 2007-02-25 20:40 php-gtk-2.0.alpha.tar.gz
xave@debian:~/phpgtk$
```

Dézipper le dossier avec la commande :

```
gzip -d php-gtk-2.0.alpha.tar.gz
```



```
xave@debian: ~/phpgtk
Fichier  Édition  Affichage  Terminal  Onglets  Aide
php-gtk-2.0.0alpha/test/unittests/PangoTabArrayTest.php
php-gtk-2.0.0alpha/test/unittests/PangoTest.php
php-gtk-2.0.0alpha/test/window.xpm
php-gtk-2.0.0alpha/T0002
php-gtk-2.0.0alpha/win32/
php-gtk-2.0.0alpha/win32/buildconf.js
php-gtk-2.0.0alpha/win32/config.w32.in
php-gtk-2.0.0alpha/win32/confutils.js
php-gtk-2.0.0alpha/win32/gdbtool.ico
php-gtk-2.0.0alpha/win32/gdk-1.30.ico
php-gtk-2.0.0alpha/win32/inst.cmd
php-gtk-2.0.0alpha/win32/makedist.cmd
php-gtk-2.0.0alpha/win32/php.ini
php-gtk-2.0.0alpha/win32/php_gtk.dsw
php-gtk-2.0.0alpha/win32/php_win.c
php-gtk-2.0.0alpha/win32/php_win.dsp
php-gtk-2.0.0alpha/win32/php_win.rc
php-gtk-2.0.0alpha/win32/README.txt
php-gtk-2.0.0alpha/win32/README.win32.txt
php-gtk-2.0.0alpha/win32/resource.h
php-gtk-2.0.0alpha/win32/template.rc
xave@debian:~/phpgtk$ ls
php-gtk-2.0.0alpha  php-gtk-2.0.0alpha.tar
xave@debian:~/phpgtk$
```

Détarer ensuite le dossier en utilisant la commande :

```
tar -xvf php-gtk-2.0.0alpha.tar.gz
```

Pour entrer dans le dossier, utilisez la commande :

```
cd php-gtk-2.0.0alpha
```


En root, installez PHP5 avec la commande :

```
apt-get install php5
```

 Cette commande oblige à l'installation d'Apache2.

En root, installez le paquet php5-cli avec la ligne de commande :

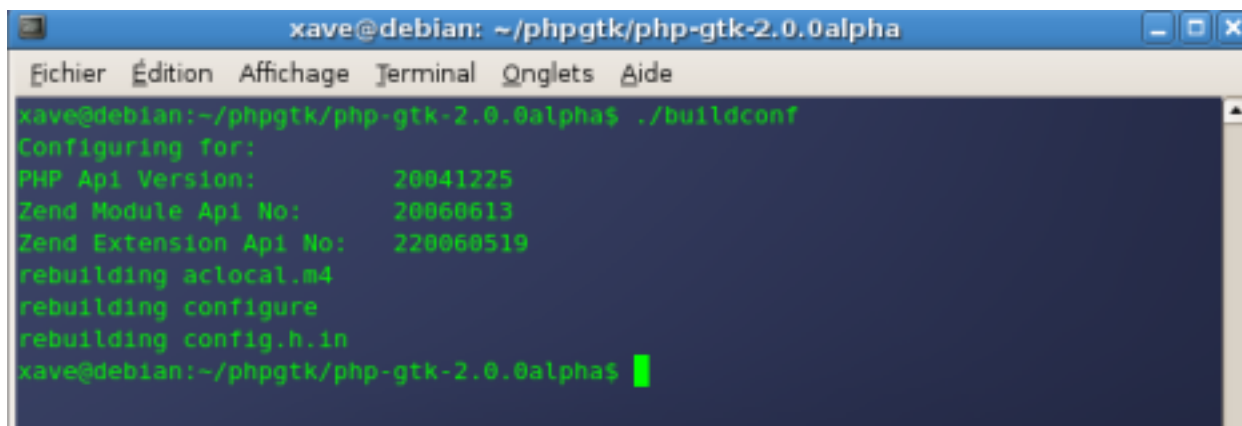
```
apt-get install php5-cli
```

 Pour ce tutoriel, nous n'installons que la base de PHP 5 mais vous pouvez installer la totalité des extensions.

Toujours en *root*, installez la paquetage **php5-dev** pour avoir la fonction **phpize()** nécessaire à la compilation de **php-gtk**.

Dans le répertoire de php-gtk, tapez la commande :

```
./buildconf
```



```

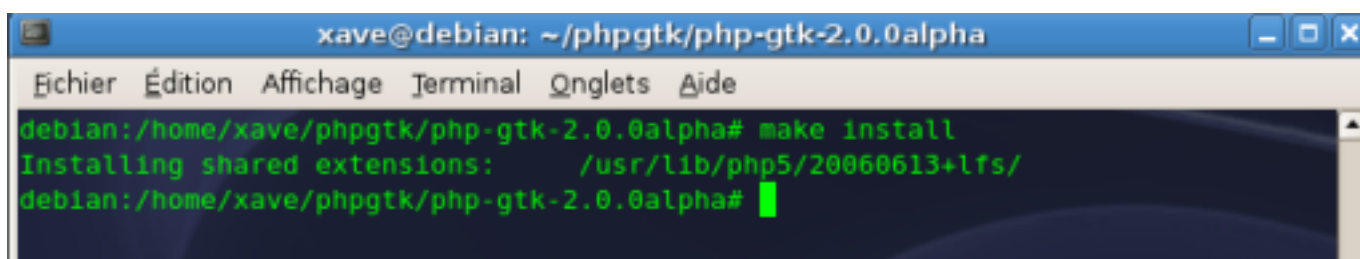
xave@debian: ~/phpgtk/php-gtk-2.0.0alpha
Fichier  Édition  Affichage  Terminal  Onglets  Aide
xave@debian:~/phpgtk/php-gtk-2.0.0alpha$ ./buildconf
Configuring for:
PHP Api Version:      20041225
Zend Module Api No:  20060613
Zend Extension Api No: 220060519
rebuilding aclocal.m4
rebuilding configure
rebuilding config.h.in
xave@debian:~/phpgtk/php-gtk-2.0.0alpha$ █
    
```

Puis :

```
./configure
make
```

En Root, procédez à l'installation en tapant la commande :

```
make install
```



```

xave@debian: ~/phpgtk/php-gtk-2.0.0alpha
Fichier  Édition  Affichage  Terminal  Onglets  Aide
debian:/home/xave/phpgtk/php-gtk-2.0.0alpha# make install
Installing shared extensions:  /usr/lib/php5/20060613+libs/
debian:/home/xave/phpgtk/php-gtk-2.0.0alpha# █
    
```


Enfin :
































```
cd /usr/lib/php5/20060613+libs/
mv php_gtk2.so php_gtk.so
```

## II-B - Installation sous Win32

L'installation sous Windows est relativement simple et rapide.

**Téléchargez la version php-gtk-1.0.2-win32.zip qui est la dernière version stable pour Windows :**

- Rendez-vous à l'adresse :  <http://gtk.php.net/download.php> ;
- Téléchargez l'archive **php-gtk-1.0.2-win32.zip** (php-gtk-1.0.2 Windows and PHP Binaries) ;
- Créez un répertoire "PHP4" à la racine de votre disque (ex : C:\PHP4) ;
- Décompressez l'archive dans ce répertoire ;
- **Vous devez retrouver les répertoires suivants dans votre C:\PHP4 :**
  - php4
  - text
  - winnt
- Copiez le contenu du répertoire C:\PHP4\php4 dans C:\PHP4

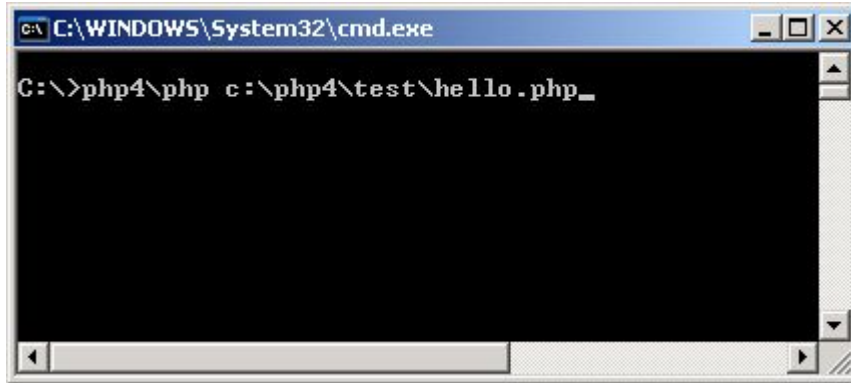
 dev		Dossier de fichiers
 php4		Dossier de fichiers
 test		Dossier de fichiers
 winnt		Dossier de fichiers
 gtkextra.dll	428 Ko	Extension de l'applic...
 iconv.dll	872 Ko	Extension de l'applic...
 intl.dll	36 Ko	Extension de l'applic...
 libgdk-0.dll	492 Ko	Extension de l'applic...
 libglade.dll	136 Ko	Extension de l'applic...
 libglib-2.0-0.dll	591 Ko	Extension de l'applic...
 libgmodule-2.0-0.dll	27 Ko	Extension de l'applic...
 libgobject-2.0-0.dll	254 Ko	Extension de l'applic...
 libgthread-2.0-0.dll	37 Ko	Extension de l'applic...
 libgtk-0.dll	1 630 Ko	Extension de l'applic...
 libxml2.dll	704 Ko	Extension de l'applic...
 NEWS	11 Ko	Fichier
 php4ts.dll	1 352 Ko	Extension de l'applic...
 php.exe	24 Ko	Application
 php.ini-gtk	1 Ko	Fichier INI-GTK
 php_gtk.dll	488 Ko	Extension de l'applic...
 php_gtk_combobutton.dll	68 Ko	Extension de l'applic...
 php_gtk_extra.dll	96 Ko	Extension de l'applic...
 php_gtk_libglade.dll	52 Ko	Extension de l'applic...
 php_gtk_scintilla.dll	268 Ko	Extension de l'applic...
 php_gtk_scrollpane.dll	60 Ko	Extension de l'applic...
 php_gtk_spaned.dll	60 Ko	Extension de l'applic...
 php_gtk_sqpane.dll	60 Ko	Extension de l'applic...
 php_win.exe	60 Ko	Application
 php-cgi.exe	44 Ko	Application
 README.txt	2 Ko	Document texte
 zlib.dll	72 Ko	Extension de l'applic...

## II-C - Test de l'installation

Dans ce tutorial, nous partons du principe que vous avez installé la version Win32 de PHP-GTK et qu'elle se trouve à la racine de votre disque.

Lancez le script **hello.php** pour vérifier que l'installation s'est bien passée. Pour cela, ouvrez une fenêtre de commande DOS, placez-vous à la racine du disque sur lequel PHP-GTK est installé et tapez la ligne de commande

suivante (à condition que votre installation soit sur C:, sinon Remplacez C: par le nom du disque dur où se trouve PHP-GTK) :




Si la fenêtre ci-dessous s'ouvre et que vous n'avez aucune erreur qui apparaît dans votre fenêtre d'invite de commandes DOS, c'est que votre installation s'est bien déroulée.



Bravo, vous pouvez maintenant utiliser PHP-GTK.

## III - Description et réalisation d'un exemple

### III-A - Chargement de la librairie GTK et notre premier widget : GtkWindow

Maintenant que PHP-GTK est installé et fonctionne sur votre poste, nous allons analyser pas à pas la structure du script d'exemple "hello.php" et créer le nôtre, qui aura pour nom "dvp.php". Pour ce faire, créez un répertoire "mes\_tests" sous le répertoire PHP4, ouvrez votre  **EDI** favori.


Le script "dvp.php" est très simple. Il crée une fenêtre, place un bouton dedans, affiche du texte et détruit la fenêtre lorsqu'on clique sur le bouton.

En premier lieu, il faut charger la librairie GTK de façon à disposer de ses différentes ressources, d'où les premières lignes suivantes dans le script hello.php :

```
<?php
if (!extension_loaded("gtk")) {
    dl( "php_gtk." . PHP_SHLIB_SUFFIX);
}
?>
```


On peut également l'écrire sous cette forme :

```
<?php
if (!class_exists("gtk")) {
    //si l'OS est windows, chargement de la php_gtk.dll
    if (strtoupper(substr(PHP_OS, 0,3)) == "WIN") { dl("php_gtk.dll"); }
    //sinon chargement du php_gtk.so (linux, mac os)
    else { dl("php_gtk.so"); }
}
?>
```

Cela nous rend indépendants de l'environnement sur lequel tourne notre script en testant l' **OS**, nous allons charger soit les extensions .dll (Win32) soit .so (Linux et MacOS).

Ensuite, nous créons un nouvel objet GTK. Nous appelons la classe **GtkWindow** grâce à l'opérateur **new**, ceci afin de produire un widget fenêtre qui sera notre container :

```
<?php
$window = &new GtkWindow();
$window->show_all();
Gtk::main();
?>
```

 Nous voyons que nous appelons l'objet *GtkWindow* par référence dans le constructeur (signe &).

**Petit rappel concernant les références** : une référence est un alias pointant directement une variable. Par exemple :


```
$a = 1;
$b = &$a; // $b a maintenant la valeur de $a
$a++;
Echo $b; // affiche 2, soit la valeur de $a incrémentée de 1
```

```
Unset ($a); // la variable $a n'existe plus
Echo $b; // affiche toujours 2, contient toujours la valeur de $a à l'origine.
```

On utilise également les références pour passer des paramètres aux fonctions (ce qui permet d'économiser le return dans la fonction et d'avoir plusieurs variables en retour de celle-ci) et pour faire référence aux objets.

Ici, on utilise le passage par référence pour l'objet **GtkWindow**, ainsi, quand on applique une fonction à **\$window**, c'est comme si on l'appliquait directement à **GtkWindow**.

**\$window->show\_all()** permet simplement d'afficher la fenêtre ainsi que son contenu et **Gtk::main()** exécute la boucle principale.

 *Ces deux fonctions et le chargement de la librairie GTK seront systématiquement présents dans chacun de vos scripts.*

Nous allons écrire ces trois lignes dans notre script `dvp.php` à la suite du chargement de la librairie GTK.

### III-B - Ajout d'une fonction : `destroy()`

Si vous lancez ce script tel qu'il est (avec la ligne de commande `C:\PHP4\php C:\PHP4\ votre_rep\ votre_script.php`), vous obtiendrez une fenêtre grise et vide avec pour titre '`votre_script.php`'.

Si vous essayez de fermer cette fenêtre avec la traditionnelle croix, vous remarquerez que la fenêtre disparaît mais le programme est toujours actif dans votre fenêtre de commande.

Vous n'avez alors pas d'autre choix que de fermer la fenêtre DOS ou de faire un `ctrl+C` pour arrêter le script.

Pour palier ce problème, nous allons ajouter une fonction qui s'exécutera lorsque nous cliquerons sur la croix de fermeture de la fenêtre :

```
<?php
function destroy() {
    Gtk::main_quit();
}
?>
```

Rajoutons cette fonction entre le chargement de la librairie et la création de notre fenêtre.

Nous allons appeler cette fonction avec la méthode **connect()**. Cette méthode prend deux paramètres. Le premier est le rappel (événement) "destroy" qui émet le signal "destroy". Ce rappel fait partie des événements du "delete-event" de PHP-GTK. Au signal reçu, nous appellerons la fonction **destroy()** (qui est le deuxième paramètre de la méthode `connect()`, sans les parenthèses) qui s'exécutera, arrêtera la boucle principale, détruira le widget **\$window** et fermera la fenêtre. Il est à noter qu'il n'est pas indispensable que la fonction appelée porte le même nom que le rappel. Nous aurions très bien pu l'appeler **toto()**, auquel cas nous l'aurions connectée de cette façon : **\$window->connect("destroy", "toto")**.

On appelle cette fonction, directement après avoir créé la fenêtre :

```
<?php
$window = &new GtkWindow();

// Appel à la fonction destroy()
```

```

$window->connect("destroy", "destroy");

$window->show_all();

Gtk::main();
?>
    
```

En exécutant le script, vous vous rendez compte que, cette fois, en cliquant sur la croix en haut à droite de la boîte de dialogue, le script est bien arrêté et vous récupérez la main dans la fenêtre de commande.

### III-C - Aspect de notre fenêtre GtkWidget

Occupons-nous maintenant de l'aspect de la fenêtre. Nous lui ajoutons une bordure de 10 pixels avec la méthode **set\_border\_width()**. Si nous ajoutons un widget à cette fenêtre, il en prendra la superficie moins 10 pixels en hauteur et largeur.

```

<?php
$window->set_border_width(10);
?>
    
```

Si nous mettons une bordure de 100 pixels et que nous lançons le script, nous voyons que la taille de la fenêtre a augmenté de façon à avoir une bordure de 100 pixels. La taille de la fenêtre est donc dynamique. Nous aurions très bien pu lui assigner une taille fixe grâce à la méthode **set\_default\_size(largeur,hauteur)** ainsi qu'un positionnement sur l'écran avec la méthode **set\_position()**. Cette dernière méthode possède quatre différentes valeurs de **GtkWindowPosition** :

Valeur	Nom symbolique	Description
0	GTK_WIN_POS_NONE	Laisse au système de fenêtrage de la machine définir la position automatiquement. Généralement, elle sera placée en haut à gauche de l'écran (Etat par défaut).
1	GTK_WIN_POS_CENTER	Place la fenêtre au centre de l'écran.
2	GTK_WIN_POS_MOUSE	Dessine la fenêtre à la pointe de la souris. Très utile pour le menus pop up ou autres widget similaires qui sont générés par un clic de souris.
3	GTK_WIN_POS_CENTER_ALWAYS	Place la fenêtre au centre de l'écran et garde en mémoire les coordonnées de sa position. Si vous l'agrandissez ou la déplacez, elle reprendra automatiquement sa place au centre.

La méthode **set\_position()** accepte la valeur ou le nom symbolique en paramètre. Prenez l'habitude de mettre le nom symbolique, qui est beaucoup plus parlant que la valeur. Nous venons de voir que le widget **GtkWindow** dispose de

deux autres méthodes : **set\_default\_size()** et **set\_position()**. En fait, il en possède beaucoup plus que cela. Nous le verrons dans un prochain tutoriel où nous nous amuserons à changer la couleur de fond de la fenêtre, changer la police de caractères, etc.

Nous n'incluons pas le **set\_default\_size()** et le **set\_position()** dans notre script, mais rien ne vous empêche de jouer avec.

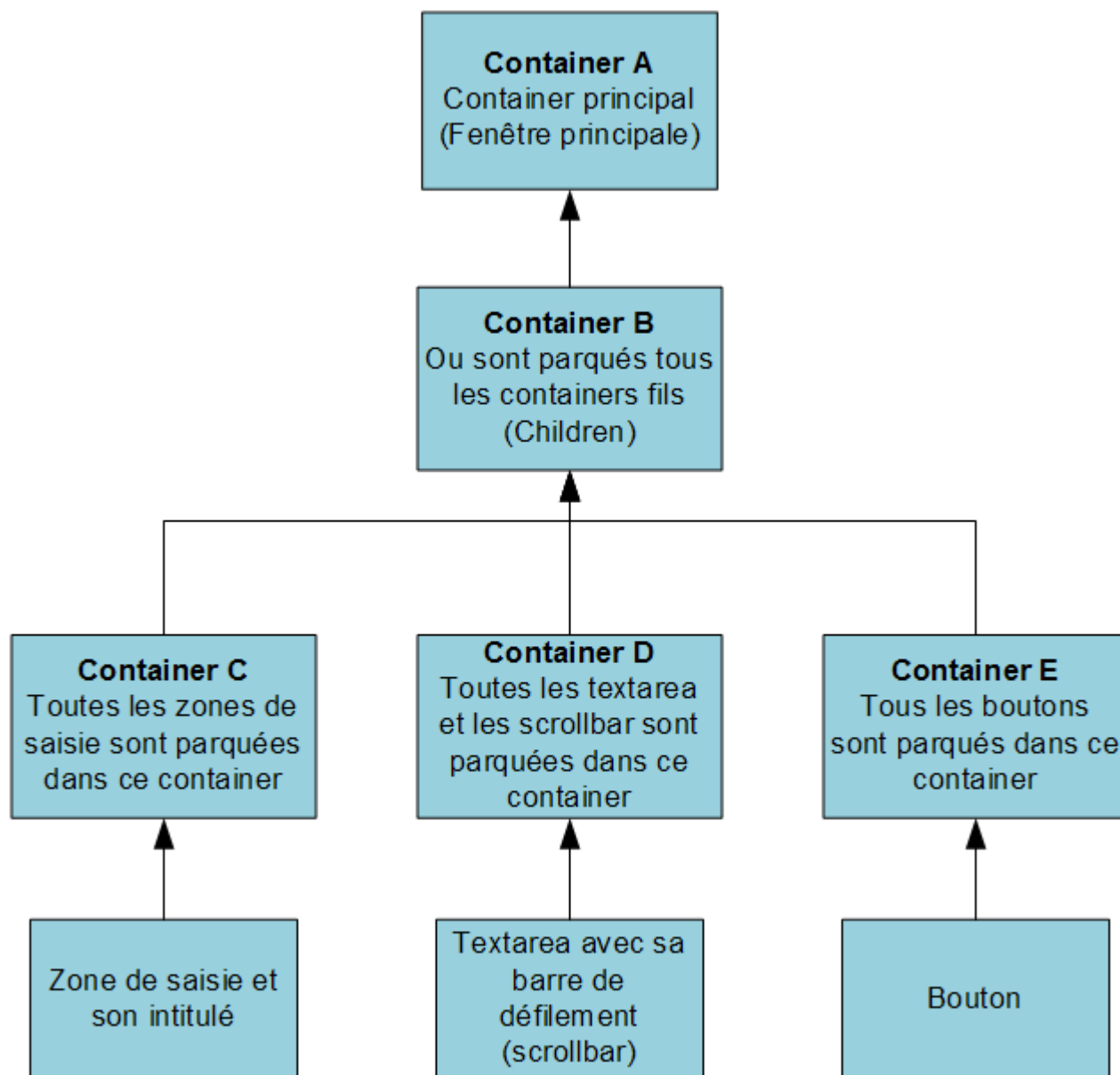
### III-D - Ajout d'un bouton : GtkButton

Ajoutons maintenant un bouton à notre fenêtre. Il sera associé à une fonction qui affiche sa valeur, ferme l'application et affiche un message dans l'invite de commandes. Nous allons, encore une fois, utiliser la méthode **connect()** et ses deux paramètres, le rappel et la fonction. Le widget **GtkButton** possède cinq rappels différents qui envoient cinq signaux homonymes :

Rappel	Description
pressed()	Emet le signal "pressed" quand le bouton est pressé
released()	Emet le signal "released" quand le bouton est libéré
clicked()	Emet le signal "clicked" quand le bouton est cliqué
enter()	Emet le signal "enter" quand le curseur est sur le bouton
leave()	Emet le signal "leave" quand le curseur a quitté la région du bouton

Nous allons utiliser le rappel **clicked()** et nous allons lier ce rappel à la fonction **hello\_dvp()** que nous écrirons ultérieurement. Pour l'instant, créons le nouveau widget **GtkButton**. Connectons-lui un rappel, qui exécutera une fonction **hello\_dvp()** grâce à la méthode **connect()**. Ajoutons notre bouton au container principal **\$window** avec la méthode **add()**.

Nous touchons là une partie très importante de PHP-GTK : la hiérarchie des containers. En effet, lorsque vous réaliserez des interfaces complexes (ex : zones de saisie avec *textarea*, *scrollbar* et différents boutons), il faudra parquer (avec la méthode **pack\_start()** que nous verrons dans un prochain tutorial) ces différents widgets dans des containers qui seront inclus dans des containers et ainsi de suite, afin de n'en avoir plus qu'un seul à ajouter à notre fenêtre principale avec la méthode **add()**.



Nous construirons les conteneurs dans l'ordre : A B C D E. Nous parquerons (méthode **pack\_start()**) dans l'ordre E D C dans B que nous ajouterons (méthode **add()**) dans le container principal A.

Puisque dans notre cas, nous n'avons qu'un seul widget (**GtkButton**) nous l'ajoutons, sans le parquer, dans notre fenêtre principale (**GtkWindow**).

```

<?php
if (!class_exists("gtk")) {
    if (strtoupper(substr(PHP_OS, 0,3)) == "WIN") { dl("php_gtk.dll"); }
    else { dl("php_gtk.so"); }
}

function destroy() {
    Gtk::main_quit();
}
    
```

```

$window = &new GtkWindow();
$window->connect("destroy", "destroy");
$window->set_border_width(10);

//Créé un nouvel objet bouton
$button = &new GtkButton("Dvp!");

//Associe la fonction hello_dvp() au bouton créé
$button->connect("clicked", "hello_dvp");


//Ajoute le bouton dans la fenêtre
$window->add($button);
$window->show_all();

Gtk::main();

?>
    
```

### III-E - Ajout de la fonction hello\_dvp()

Ajoutons la fonction **hello\_dvp()** que nous allons détailler maintenant. Nous devons déclarer notre variable **\$window** comme globale car nous faisons appel à notre **GtkWindow**. Nous faisons un **print()** d'un texte qui apparaîtra dans la fenêtre d'invite de commandes. Un **echo()** aurait aussi fait l'affaire. Enfin, nous faisons un appel de la fonction **destroy()** liée à notre **\$window** qui nous permettra de fermer la fenêtre principale et de terminer l'application.

 *Le meilleur moyen de débogage dans PHP-GTK est le **echo()** ou le **print()**.*

Ajoutons la fonction **hello\_dvp()** dans le code :

```

<?php
function hello_dvp() {
    global $window;
    //Texte affiché dans la fenêtre de commande
    print "Dvp, c'est vraiment bien\n";
    //Appel à la fonction destroy qui ferme l'application
    $window->destroy();
}

?>
    
```

### III-F - Ajout d'un widget "tooltips" GtkTooltips

Pour finir, nous allons ajouter un label au survol du bouton qui affichera un commentaire. Pour cela, on créé un objet de type Tooltips (**GtkTooltips**), on lui assigne un délai d'affichage avec la méthode **set\_delay()** (le temps est en millisecondes). Avec la méthode **set\_tip()**, nous connectons ce Tooltip au widget sur lequel nous voulons voir cette bulle d'information apparaître (**\$button**) et le texte qui s'y trouvera visible. Enfin, nous le rendons visible avec la méthode **enable()**.

```

<?php
//Création du "widget" Tooltips
$tt = &new GtkTooltips();
//Délai d'affichage du "widget"
$tt->set_delay(200);
//Contenu du "widget"
$tt->set_tip($button, "Donnez votre avis sur le site de dvp...", "");
//Affichage du "widget"
$tt->enable();

?>
    
```

## III-G - La version finale de notre script

Ce listing est donc la version finale de notre script :

```
<?php
/*****
 * Chargement de la librairie GTK *
 *****/
if (!extension_loaded("gtk")) {
    dl( "php_gtk." . PHP_SHLIB_SUFFIX);
}

/*****
 * Fonction appelée quand la fenêtre est détruite, permet de quitter le programme *
 *****/
function destroy() {
    Gtk::main_quit();
}

/*****
 * Fonction appelée quand le bouton est cliqué. Affiche le message dans la fenêtre de *
 * commande et détruit la boîte de dialogue *
 *****/
function hello_dvp() {
    global $window;
    print "Dvp, c'est vraiment bien\n";
    $window->destroy();
}

/*****
 * Créé une fenêtre et associe différentes fonctions de gestion de la fenêtre *
 *****/
$window = &new GtkWindow();
$window->connect("destroy", "destroy");
$window->set_border_width(10);

/*****
 * Créé un "widget" de type bouton et assigne la fonction hello *
 *****/
//Créé un nouveau "widget" bouton
$button = &new GtkButton("Dvp!");
//Associe la fonction hello au bouton créé
$button->connect("clicked", "hello_dvp");
//Ajoute le bouton dans la fenêtre ("container" $window)
$window->add($button);

/*****
 * Créé un "widget" de type Tooltips et assigne un contenu *
 *****/
//Création du "widget" Tooltips
$tt = &new GtkTooltips();
//Délai d'affichage du "widget"
$tt->set_delay(200);
//Contenu du "widget"
$tt->set_tip($button, "Donnez votre avis sur le site de dvp...", "");
//Affichage du "widget"
$tt->enable();

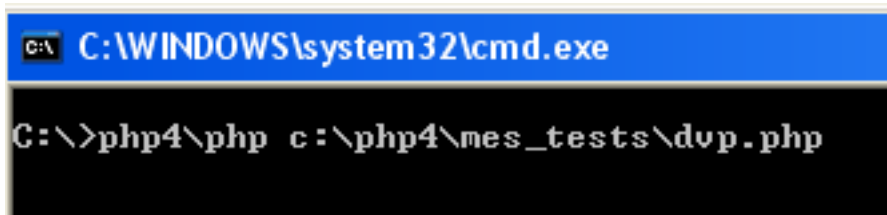
/*****
 * Montre la fenêtre et tous ses composants *
 *****/
$window->show_all();


/*****
 * Exécute la boucle principale *
 *****/
```

```
Gtk::main();  
?>
```

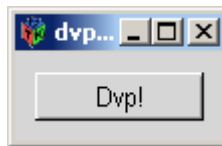
Comme nous l'avons dit précédemment, nous pouvons utiliser la version pseudo objet de PHP 4 dont le listing se trouve en annexe V-A.

Nous pouvons maintenant lancer l'application avec la ligne de commande suivante :



 Nous pouvons également faire un fichier dans lequel nous mettons cette ligne de commande, le sauvegarder sur le bureau avec l'extension `.bat` (win32). En double cliquant sur ce fichier, l'application se lancera.

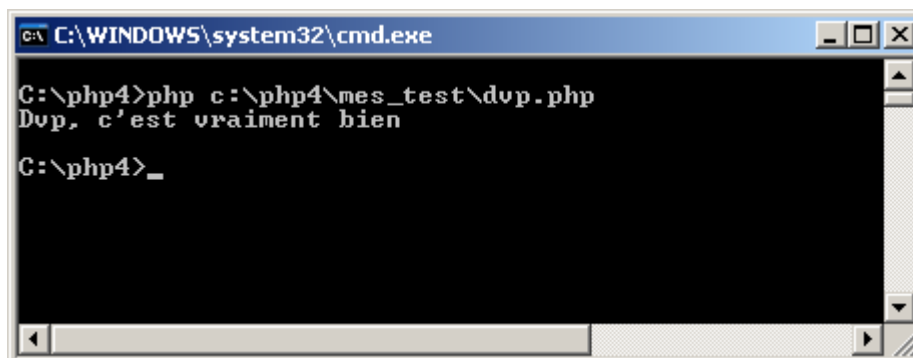
Lors de l'exécution de l'application, nous voyons s'ouvrir notre fenêtre contenant notre bouton :



Si nous mettons notre curseur sur le bouton, notre bulle d'information (Tooltip) apparaît.




Si nous cliquons sur notre bouton "Dvp", nous récupérons bien en sortie "Dvp, c'est vraiment bien" dans notre fenêtre d'invite de commandes grâce à `print()`. De même, notre `GtkWindow` se ferme grâce à la fonction `destroy()` se trouvant dans notre `hello_dvp()`.



## IV - Conclusion

Nous avons rapidement vu ce que peut produire PHP couplé avec GTK.

Nous avons vu que PHP-GTK nous permet de créer des fenêtres et de récupérer l'action de l'utilisateur par le biais d'événements (Click sur un bouton) et de déclencher une procédure (fonction) derrière. La librairie GTK est là pour nous permettre de faire une interface utilisateur, sans passer par du HTML et donc, sans avoir besoin d'installer un serveur web. Nous verrons dans de prochains articles que nous pourrons utiliser toute la puissance du langage PHP (connexion à des bases de données *etc.*) pour réaliser des applications avancées.

 *Dans cet article, nous avons utilisé PHP-GTK 1.0.2 car il y a encore beaucoup de développeurs qui travaillent avec PHP 4. Dans les prochains, nous utiliserons PHP-GTK 2 et PHP 5 car, même si c'est une version Alpha, elle est supportée par une grande communauté.*

## V - Annexes

### V-A - Code objet

Comme dit plus haut, voici le même script en pseudo objet PHP 4. Nous l'avons mis en annexe pour ne pas perturber la lecture.

Nous vous le proposons car il existe une différence essentielle entre le code procédural et l'objet.

Nous voyons que nous n'utilisons plus la méthode **connect()**, mais **connect\_object()** qui prend deux paramètres. Le premier est le rappel (comme dans **connect()**), mais le deuxième est un tableau contenant le widget (**&\$this == GtkWidget**) et la fonction associée.

```
<?php
if (!class_exists("gtk")) {
    if (strtoupper(substr(PHP_OS,0,3) == "WIN")) { dl("php_gtk.dll"); }
    else { dl("php_gtk.so"); }
}

class dvp_obj {
    var $window;
    var $button;
    var $tt;

    //Constructeur
    function dvp_obj() {
        //creation de la GtkWidget
        $this->window = &new GtkWidget();
        $this->window->connect_object("destroy",array(&$this,"destroy"));
        $this->window->set_border_width(10);

        //creation du GtkWidget
        $this->button = &new GtkWidget("Dvp");
        $this->button->connect_object("clicked",array(&$this,"hello_dvp"));
        $this->window->add($this->button);

        //creation du GtkWidget
        $this->tt = &new GtkWidget();
        $this->tt->set_delay(200);
        $this->tt->set_tip($this->button,"Donnez votre avis sur le site de dvp","");
        $this->tt->enable();

        //affichage de la fenetre
        $this->window->show_all();
    }






    //fonction destroy
    function destroy() {
        Gtk::main_quit();
    }

    //fonction hello_dvp
    function hello_dvp() {
        print "Dvp, c'est drolement bien!\n";
        $this->window->destroy();
    }
}

$win = new dvp_obj();
Gtk::main();

?>
```

## V-B - Quelques liens sur PHP-GTK

-  **Le site officiel de PHP-GTK ;**
-  **La page de telechargement de PHP-GTK ;**
-  **La page de la documentation ;**
-  **Site de la communaute des utilisateurs de PHP-GTK.**
-  **La page GTK+ de developpez.com (FAQ, tutoriels etc).**

## V-C - Remerciements

### **Nous voudrions remercier quelques membres de developpez.com :**

- BWP-Necromance pour son aide, sa patience et sa disponibilité ;
- Yogui pour sa patience, sa relecture ainsi que son aide à la mise en forme de cet article ;
- Titoumimi pour la filiation de Yiannis.

