

PHP-GTK

Ketika dilahirkan, PHP adalah sosok bahasa pemrograman khusus untuk lingkungan web. Sejak dilahirkan hingga sekarang, popularitas PHP sangat luar biasa. Rasanya tidak ada satu orang pun yang berkecimpung dalam dunia Internet tidak mengenal PHP, bahkan PHP sudah dianggap bahasa pemrograman yang wajib dikuasai oleh setiap programmer web.

Dari sekian banyak programmer Internet di dunia, ada seorang yang bernama **Andrei Zmievski** yang memiliki ketertarikan lain pada PHP. Dia melihat bahwa PHP bisa juga dijadikan sebagai bahasa pemrograman untuk lingkungan desktop. Ide awalnya adalah ketika dia melihat implementasi GTK + pada Python dan Perl. Kalau GTK + bisa diimplementasikan pada Python dan Perl, mengapa tidak jika GTK + diimplementasikan pada PHP.

Andrei Zmievski lalu meminta bantuan James Henstridge (pengembang PyGTK) untuk mengimplementasikan GTK + pada PHP. Di bulan Agustus 2000, lahirlah PHP-GTK. PHP-GTK adalah ekstension PHP pertama yang menjadikan PHP sebagai bahasa untuk lingkungan *desktop*. Awalnya PHP-GTK hanya bisa berjalan di lingkungan Linux saja, namun setelah Andrei Zmievski menyelesaikan sebagian proyek PHP-GTK-nya, dia menunjukkan kode PHP-GTK kepada **Frank Kromann**, dan dengan bantuan Frank Kromann, PHP-GTK bisa dijalankan di lingkungan Windows juga.

Instalasi PHP-GTK di Linux

PHP-GTK bisa di-download di <http://gtk.php.net/download.php/>. Untuk Linux pilih file source code PHP-GTK.

Karena PHP-GTK adalah ekstension PHP dan tidak bisa berdiri sendiri, maka sebelum menginstal dan menggunakan PHP-GTK, Anda harus memiliki PHP terlebih dahulu. Anda harus menginstal PHP sebagai binary/CGI bukan sebagai Apache Modul.

Jika Anda belum memiliki PHP, silakan download source code PHP di <http://www.php.net/downloads.php>. PHP yang harus Anda gunakan minimal versi 4.2.3.

Secara umum, instalasi PHP adalah sebagai berikut:

- Extract file source code PHP, misalkan file source code PHP adalah php-

4.3.3.tar.gz.

```
#tar -xvzf php-4.3.3.tar.gz
```

- Akan terbentuk sebuah direktori yang berisi source code PHP. Misalkan direktorinya adalah php-4.3.3. Masukkan ke dalam direktori tersebut.

```
#cd php-4.3.3
```

- Jalankan skrip configure untuk memeriksa sistem dan mempersiapkan file-file untuk proses kompilasi.

```
#./configure
```

- Anda bisa memberikan parameter `-with-mysql` agar PHP mendukung MySQL, tapi jangan gunakan opsi `-with-apache`, misalkan:

```
#./configure --with-mysql
```

- Jalankan skrip make untuk melakukan proses kompilasi source code PHP.

```
#make
```

- Jalankan skrip make sekali lagi dengan opsi `install` untuk menginstal file-file hasil kompilasi ke direktori yang bersesuaian.

```
#make install
```

Bisa terjadi proses instalasi PHP tidak berjalan dengan baik. Jika ada masalah, Anda dapat mencari jawabannya di <http://www.php.net>.

Setelah instalasi PHP selesai, Anda dapat melanjutkan dengan proses instalasi PHP-GTK. Secara umum instalasi PHP-GTK adalah sebagai berikut:

- Extract file source code PHP-GTK, misalkan file source code PHP-GTK adalah php-gtk-0.5.2.tar.gz.

```
#tar -xvzf php-gtk-0.5.2.tar.gz
```

- Akan terbentuk sebuah direktori yang berisi source code PHP-GTK. Misalkan direktorinya bernama php-gtk-0.5.2.



Masukkan ke dalam direktori tersebut.

```
#cd php-gtk-0.5.2
```

- Jalankan skrip buildconf untuk men-setup file-file yang diperlukan dan membuat skrip configure.

```
#./buildconf
```

- Jalankan skrip configure yang telah dibuat secara otomatis pada proses sebelumnya.

```
#./configure
```

- Jalankan skrip make untuk mengkompilasi source code PHP-GTK.

```
#make
```

- Jalankan skrip make lagi dengan opsi `install` untuk menginstal PHP-GTK pada direktori default extension PHP.

```
#make install
```

Pada beberapa kasus, instalasi tidak berjalan mulus. Misalkan pada RH 7.3 ke atas, proses kompilasi tidak berjalan dengan sempurna dan berhenti di tengah jalan. Jika hal tersebut terjadi, cobalah untuk mengubah isi file `php.ini`. Biasanya file `php.ini` ada di direktori `/etc/php.ini`. Tapi jika tidak coba cari dengan cara:

```
php -i | grep ini
```

Buka file `php.ini`, dan cari statement sebagai berikut.

```
memory_limit = 8MB
```

Ubahlah nilainya menjadi 64MB, lalu jalankan kembali skrip make. Jika masih ada kesalahan atau ada masalah lain selama proses instalasi, silakan untuk mencari penyelesaiannya di <http://gtk.php.net>.

Jika proses instalasi berjalan dengan baik, cobalah untuk masuk ke lingkungan XWindow, dan buka terminal emulator.

Masuk ke direktori test pada direktori php-gtk, dan jalankan file gtk.php sebagai berikut.

```
php -q gtk.php
```

Jika Anda melihat tampilan seperti Gambar 1, berarti proses instalasi PHP-GTK telah sukses dan siap untuk belajar PHP-GTK.

Hello World

Aplikasi PHP-GTK pertama yang akan Anda pelajari di sini adalah aplikasi 'Hello World'.

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

function shutdown()
{
    gtk::main_quit();
}

$mainwin = &new GtkWindow();
$mainwin->connect('destroy', 'shutdown');
$mainwin->set_title('PHP-GTK :');
$button = &new GtkButton('Hello World! Joy to the world!');
$button->connect('clicked', 'shutdown');
$mainwin->add($button);
$mainwin->show_all();

gtk::main();

?>
```

Simpanlah skrip tersebut dengan nama helloworld.php lalu ketik:

```
#php -q helloworld.php
```

Jika Anda klik tombol "Hello World! Joy to the world", maka skrip helloworld.php akan berakhir.

Sekarang mari kita bahas skrip di atas. Seperti halnya skrip PHP, skrip PHP-GTK diawali dengan tag <?php dan ditutup dengan tag ?>. Perhatikan pada bagian awal skrip yang diawali dengan pernyataan sebagai berikut:

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

Library PHP-GTK untuk Linux dan Windows berbeda, sehingga diperlukan pemeriksaan sistem operasi terlebih dahulu di awal skrip PHP-GTK. Namun jika Anda yakin bahwa skrip Anda hanya akan berjalan di Linux, maka pernyataan di atas bisa langsung memanggil library php_gtk.so tanpa harus memeriksa sistem operasi terlebih dahulu.

```
$mainwin = &new GtkWindow();
```

Pernyataan tersebut berguna untuk membuat window utama dengan membuat objek dari kelas GtkWindow(). Perhatikan bahwa pembuatan objek menggunakan kata kunci new yang diawali dengan operator referensi &.

```
$mainwin->connect('destroy', 'shutdown');
```

Kemudian, kita menggunakan method connect() untuk mendaftarkan fungsi shutdown() sebagai handler. Handler adalah fungsi yang bertugas untuk menangani signal destroy yang dipancarkan oleh window \$mainwin. Jadi ketika objek window \$mainwin memancarkan signal destroy, maka fungsi shutdown() akan dipanggil.

```
$mainwin->set_title('PHP-GTK :');
```

Pernyataan tersebut untuk menentukan judul window utama.

```
$button = &new GtkButton('Hello World! Joy to the world!');
```

Pernyataan di atas adalah membuat tombol. Untuk membuat tombol bisa dilakukan dengan membuat objek dari kelas GtkButton(). \$button->connect('clicked', 'shutdown');

Dengan menggunakan method connect(), signal clicked dihubungkan dengan fungsi shutdown().

```
$mainwin->add($button);
```

Pernyataan di atas akan menempatkan \$button pada \$mainwin.

```
$mainwin->show_all();
```

Kemudian semua objek ditampilkan pada layar.

```
gtk::main();
```

Fungsi statik gtk::main() adalah kunci dan mutlak ada di bagian akhir skrip PHP-GTK. Fungsi ini memberitahukan kepada PHP-GTK bahwa pembuatan interface sudah selesai, dan *main loop* dapat dimulai. Tanpa fungsi statis ini, maka skrip akan langsung berhenti sesaat setelah interface ditampilkan di layar.

Beberapa istilah seputar PHP-GTK

Widget

Dalam lingkungan PHP-GTK, *widget* adalah *user interface*. Jadi user interface apa pun, dalam PHP-GTK disebut widget. Widget mungkin berwujud text box, label, frame, window atau komponen GUI lainnya. Semua GTK widget berasal dari kelas abstrak GtkWidget. Semua widget juga mewarisi method, signal, dan properties yang diimplementasikan pada GtkWidget.

Dari sisi pandang programmer, suatu widget memiliki 5 bagian siklus hidup yaitu:

- **Creation.** Dalam lingkungan PHP-GTK, ini dilakukan dengan membuat suatu objek. Contoh:

```
$window = &new GtkWindow();
```

- **Placement.** Ini merupakan langkah menambahkan / menempatkan suatu widget pada suatu container. Secara umum, sintak penulisan langkah ini adalah.

```
$container->add($widget);
```

atau

```
$container->pack_start($widget);
```

atau

```
$container->pack_end($widget)
```

- **Signal Connection.** Ini merupakan langkah mengatur fungsi callback yang akan digunakan. Contoh:

```
$widget->connect("signal", "my_func");
```

Dengan "signal" merupakan keadaan yang terdefinisi. Yang dimaksud

terdefinisi adalah bahwa keadaan tersebut sudah terdaftar dalam PHP-GTK, misalkan seperti `clicked`. Sedangkan `"my_func"` adalah subroutine atau fungsi yang akan dipanggil saat signal dipancarkan.

- **Display.** Ini menggambarkan apakah suatu widget akan ditampilkan atau tidak. Caranya sederhana, yaitu:

```
$widget->show();
```

```
$widget->hide();
```

Container

Suatu *container* adalah sebuah widget yang bisa menampung widget lainnya. Yang umum termasuk dalam kategori container adalah: `GtkWindow`, `GtkBox`, dan sebagainya. Selebihnya mereka persis sama dengan widget lainnya. Container pun bisa ditambahkan ke dalam container lain. Container diturunkan dari kelas dasar `GtkContainer`. `GtkContainer` sendiri diturunkan dari kelas `GtkWidgetSignal` dan `Callback`.

Dalam pemrograman GUI, sangat diperlukan adanya mekanisme untuk merespon suatu aksi atau tindakan yang dilakukan oleh user ataupun oleh sesuatu dalam skrip itu sendiri. Mengklik suatu tombol adalah salah satu contoh aksi yang umum dilakukan oleh user. Ketika user mengklik suatu tombol, maka harus ada respon atau reaksi dari skrip sesuai dengan tujuan tombol tersebut. Misalkan, tombol EXIT berguna untuk menghentikan jalannya skrip. Untuk keperluan tersebut, harus ada mekanisme yang menghubungkan aksi klik dengan suatu fungsi yang akan memberikan respon atas aksi tersebut. Contohnya dapat dilihat pada skrip `helloworld.php` di atas:

```
$button->connect('clicked', 'shutdown');
```

Signal & Callback

Dalam PHP-GTK, mekanisme komunikasi demikian ditangani oleh signal. Signal berguna agar suatu skrip dapat mengetahui bahwa ada suatu aksi yang terjadi. Sumber suatu signal adalah widget. Signal adalah

suatu pemberitahuan yang dipancarkan oleh widget.

Dalam PHP-GTK, fungsi yang dibuat untuk menangani atau merespon signal disebut dengan nama callback. Oleh karena itu callback disebut juga signal handler function (fungsi yang menangani / merespon signal). Callback bisa berupa default handler, atau user defined handler. User defined handler adalah callback yang didefinisikan oleh programmer.

Membahas PHP-GTK tidaklah mungkin disajikan hanya dalam sebuah artikel. Contoh program lain juga perlu dipelajari. Oleh karena itu, bagi rekan-rekan yang tertarik untuk mempelajari PHP-GTK, silakan merujuk pada referensi berikut.

- Buku "Tuntunan Praktis PHP-GTK" yang ditulis oleh **Y.B. Mulyana** dan diterbitkan oleh ANDI.
- Manual PHP-GTK: <http://gtk.php.net/>

Semoga dari artikel kecil ini ada manfaat yang bisa dipetik oleh kita semua.🙏

Mulyana (mulyana@powerfulteam.biz)

IKLAN