

Wx dan Perancangan User Interface

Bagian 1 dari 2 Artikel

Pada artikel sebelumnya, kita telah membahas dasar-dasar penggunaan wx untuk pemrograman GUI multiplatform. Artikel kali ini dan sambungannya akan membahas berbagai topik-topik lanjutan seperti penggunaan *internationalization*, pembuatan *tip of the day*, dan bekerja dengan *form-form* MDI. Selamat mengikuti!

Pustaka wx dapat Anda gunakan untuk membuat aplikasi yang serius. Banyak sekali aplikasi GUI yang dibangun menggunakan pustaka *multiplatform* ini. Di pembahasan sebelumnya, kita banyak membahas alasan penggunaan wx sampai contoh program wx yang sederhana. Di artikel sebelumnya, kita membahas pula separasi *class* dalam setiap file serta separasi deklarasi dan tubuh *class*. Dasar yang kita bahas sebelumnya akan kita gunakan dalam artikel kali ini.

Di pembahasan sebelumnya, secara sekilas kita telah membahas penggunaan beberapa *widget* seperti *button*, *statusbar*, menu, dan *text control*. Wx menyediakan banyak sekali *widget* dasar dan lanjutan yang bisa Anda gunakan.

Dalam merancang UI, aplikasi yang baik haruslah berpatokan pada panduan interaksi manusia dan komputer yang baik. Dalam penggunaan *widget* saja, ada banyak patokan yang bisa kita jadikan sebagai panduan dalam mengembangkan aplikasi. Salah satunya adalah dasar pemilihan *widget* untuk tugas tertentu. Sebagai contoh, apabila user harus memasukkan input dan nilai input tersebut bisa merupakan pilihan terbatas, maka sebaiknya kita mempergunakan *widget* seperti *combobox* daripada *inputbox*.

Di artikel ini, kita akan banyak fokuskan diri pada isu seputar pengembangan *user interface*. Termasuk di dalamnya adalah penggunaan *i18n*, *tip of the day* dan *mdi* (pada edisi berikut). Semua pembahasan tersebut akan kita kupas untuk menyajikan program terbaik bagi user.

Internationalization

Umumnya, program-program yang kita gunakan disajikan dalam bahasa Inggris, selalu bahasa Internasional. Oleh karena itu, pengguna komputer umumnya harus bisa berbahasa Inggris, minimal pasif.

Namun di lain sisi, tidak semua user bisa berbahasa Inggris. Dan, jumlahnya banyak. Oleh karena itu, kita mengenal penggunaan bahasa lain dalam penyajian *user interface*. Pengguna dapat memilih bahasa yang dikuasai untuk menggunakan suatu program.

Kita mengenal istilah penyajian berbagai bahasa ini sebagai *internationalization* (*i18n*) dan *localization* (*l10n*). Masing-masing disebut *i18n* dan *l10n* karena terdapat 18 karakter antara huruf pertama dan terakhir dalam *internationalization*, serta terdapat 10 karakter antara huruf pertama dan terakhir dalam *localization*.

Dahulu, ketika belum banyak pustaka yang datang bersama kemampuan *i18n*, banyak cara manual yang ditempuh ketika membangun suatu program yang mendukung banyak bahasa. Salah satunya adalah dengan menyajikan sebagian besar string yang ditampilkan kepada user sebagai konstanta, memasukkan ke dalam *header* dan selanjutnya, berbagai *header* untuk berbagai bahasa dibuat. Tergantung target, *header* yang bersesuaian akan digunakan.

Cara lain yang digunakan adalah dengan menggunakan teks file dan mengasosiasikan isinya sebagai variabel atau konstanta untuk string yang ditampilkan ke user. Hal ini untuk mengurangi efek rekompilasi seperti

yang ditemukan pada cara sebelumnya. Dengan penggunaan *text file* yang dibaca pada saat runtime, pengubahan tidak mengharuskan adanya rekompilasi.

Kedua cara ini dahulu banyak digunakan. Namun, kedua cara tersebut juga memiliki banyak kelemahan. Sebagai contoh, kedua cara tersebut sangatlah manual dan tidak efisien. Selain itu, cara pemrograman pun haruslah disesuaikan. Bukan cara yang baik.

Pustaka-pustaka modern seperti wx datang dengan kemampuan untuk memproses penggunaan bahasa. Cara yang digunakan juga sama seperti penggunaan pustaka modern lain. User cukup menyajikan file-file bahasa pendukung dan program secara otomatis akan mengganti teks yang ditampilkan kepada user sesuai bahasa yang dipilih. File-file tersebut dibuat khusus dan disajikan dalam format binari. Sementara, cara pemrograman pun tidak berbeda jauh. Kita hanya perlu mempergunakan *macro* tertentu. Urusan membaca file bahasa, penggantian teks dan lain sebagainya tidak perlu kita pikirkan lagi.

Berikut ini adalah langkah-langkah dalam mempergunakan pemilihan bahasa, dalam contoh program yang sangat sederhana.

Langkah 1: mempersiapkan program sederhana

Kita akan membuat program sederhana, yang terdiri dari satu *form*, dengan sebuah menu, *text control* dan *status bar*, yang masing-masing akan menampilkan teks di dalamnya. Teks tersebut akan berganti se-

suai bahasa yang dipergunakan. Begitu juga dengan menu dan sub menu di dalamnya. Sebutlah program ini sebagai hello world.

Berikut ini adalah file-file yang digunakan:

- helloworld.cpp sebagai tubuh class dari aplikasi.
- helloworld.h sebagai deklarasi class dari aplikasi.
- MainFrame.cpp sebagai tubuh class frame utama.
- MainFrame.h sebagai deklarasi class frame utama.
- makefile.unx sebagai makefile.

Setelah kompilasi dilakukan, akan terbentuk file-file sebagai berikut:

- helloworld.o sebagai objek hasil kompilasi class wxHelloWorld.
- MainFrame.o sebagai objek hasil kompilasi class wxMainFrame.
- helloworld sebagai binary aplikasi.

isi helloworld.h:

```
#ifndef HelloWorld_H
#define HelloWorld_H

class wxHelloWorld : public
wxApp
{
public:
    virtual bool OnInit ();
};
```

```
DECLARE_APP (wxHelloWorld)

#endif // HelloWorld_H
```

isi helloworld.cpp:

```
#include <wx/wx.h>
#include "helloworld.h"
#include "MainFrame.h"

IMPLEMENT_APP (wxHelloWorld)

bool wxHelloWorld :: OnInit ()
{
    wxMainFrame *frame = new
wxMainFrame (_("Hello World"),
wxPoint (50,50), wxSize
(200,200));

    frame -> Show (TRUE);
    SetTopWindow (frame);

    return TRUE;
};

isi MainFrame.h:
#ifndef MainFrame_H
#define MainFrame_H

enum
```

```
{
    MENU_QUIT,
    MENU_ABOUT,
};

class wxMainFrame : public
wxFrame
{
public:
    wxMainFrame (const
wxString &title, const wxPoint
&pos, const wxSize &size);
    void OnQuit
(wxCommandEvent &event);
    void OnAbout
(wxCommandEvent &event);

protected:
    DECLARE_EVENT_TABLE ();

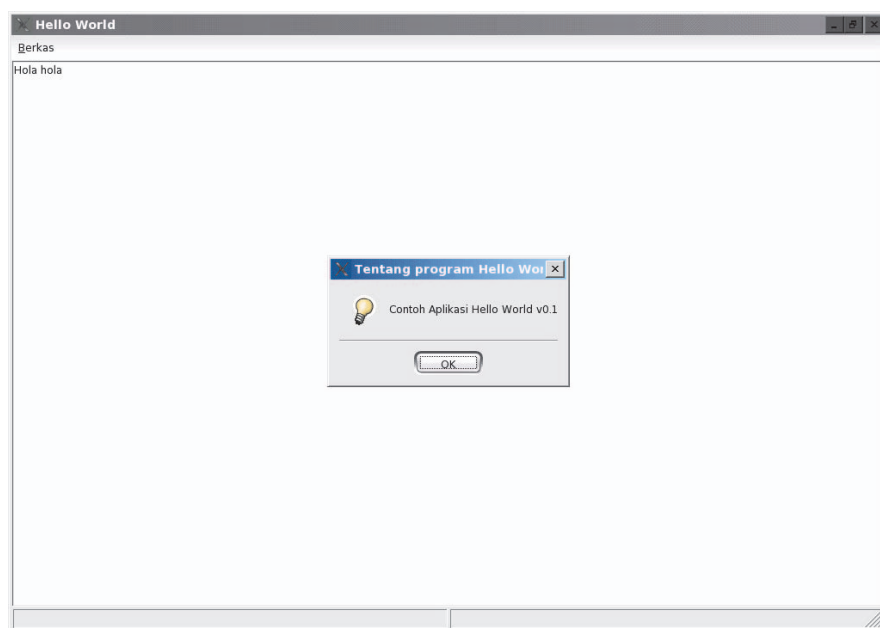
private:
    wxMenuBar *menubar;
    wxMenu *FileMenu;
    wxTextCtrl *textctrl;
};

#endif// MainFrame_H

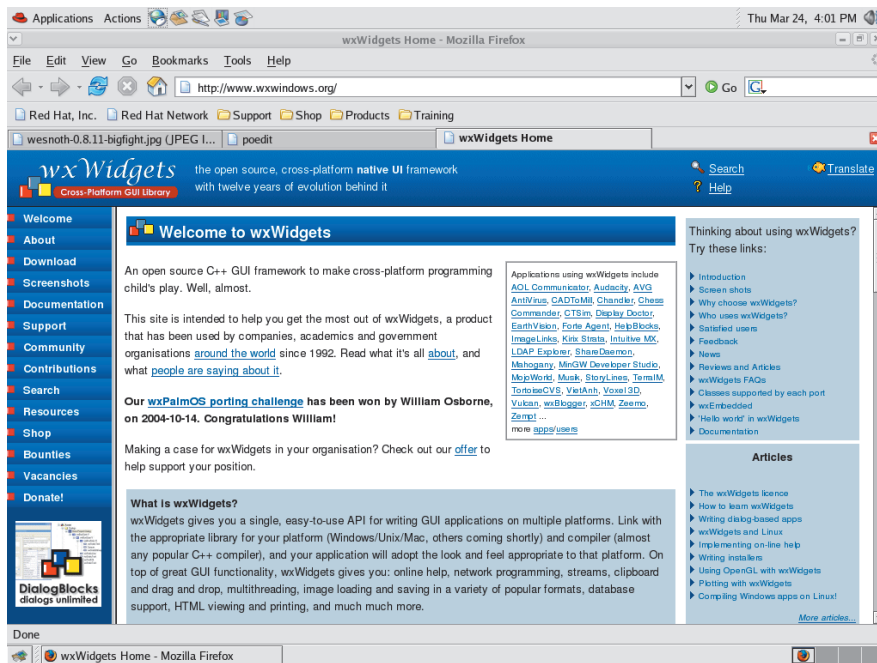
isi MainFrame.cpp:
#include <wx/wx.h>
#include "MainFrame.h"

BEGIN_EVENT_TABLE (wxMainFrame,
wxFrame)
    EVT_MENU (MENU_QUIT,
wxMainFrame :: OnQuit)
    EVT_MENU (MENU_ABOUT,
wxMainFrame :: OnAbout)
END_EVENT_TABLE ()

wxMainFrame :: wxMainFrame
(const wxString &title, const
wxPoint &pos, const wxSize
&size):
    wxFrame ( (wxFrame
*) NULL, -1, title, pos, size )
{
    CreateStatusBar (2);
    SetStatusText (_("Hello
World"));
```



Contoh i18n.



Situs web wx.

```

menubar = new wxMenuBar;

FileMenu = new wxMenu;
FileMenu -> Append (MENU_
ABOUT, _("&About"), _("Show
about info..."));
FileMenu ->
AppendSeparator ();
FileMenu -> Append (MENU_
QUIT, _("&Quit"), _("Quit from
application..."));

menubar -> Append
(FileMenu, _("&File"));

SetMenuBar (menubar);

textctrl = new wxTextCtrl
(this, -1, wxString (_
("Halo")), wxDefaultPosition,
wxDefaultSize, wxTE_MULTILINE);

Connect (MENU_QUIT,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxMainFrame :: OnQuit);
Connect (MENU_ABOUT,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)

```

```

&wxMainFrame :: OnAbout);
};

void wxMainFrame :: OnQuit
(wxCommandEvent &event)
{
    Close (TRUE);
};

void wxMainFrame :: OnAbout
(wxCommandEvent &event)
{
    wxMessageBox (_("Hello
World Example v0.1"), _("About
Hello World"), wxOK | wxICON_
INFORMATION, this);
};

```

isi makefile.unx:

```

CXX = $(shell wx-config --cxx)

PROGRAM = helloworld

OBJECTS = helloworld.o
MainFrame.o

# implementation

.SUFFIXES:      .o .cpp

```

```

.cpp.o :
    $(CXX) -c `wx-config --
cxxflags` -o $@ $<

all:      $(PROGRAM)

$(PROGRAM):      $(OBJECTS)
    $(CXX) -o $(PROGRAM)
$(OBJECTS) `wx-config --libs`

clean:
    rm -f *.o $(PROGRAM)

```

Penjelasan program:

- Kita telah mempersiapkan program untuk dukungan multibahasa.
- Tidak ada yang berbeda dengan program satu bahasa. Yang berbeda hanyalah penggunaan macro `_()` yang digunakan untuk mengapit teks yang ingin kita tampilkan.
- Dengan demikian, sebagai kesimpulan, untuk mempersiapkan program untuk dukungan multibahasa atau i18n, apa yang perlu dilakukan pada penggunaan macro `_()`, `_T()` atau `wxT()` dalam penulisan string yang ingin ditampilkan kepada user. Umumnya, apabila kita tidak menggunakan unicode, maka kita hanya perlu menggunakan `_()`. `wxT()` dan `_T()` umumnya dipergunakan untuk dukungan unicode. `wxT()` dan `_T()` sebenarnya sama saja dan dihadirkan untuk kompatibilitas dengan programmer Windows.

Langkah 2: mempersiapkan message catalog

Kita asumsikan bahwa kita akan membuat program untuk dua bahasa: Inggris dan Indonesia. Katalog untuk bahasa Inggris tidak perlu kita buat karena kita sudah menggunakan bahasa Inggris di dalam kode program. Kita sebut bahasa Inggris ini sebagai bahasa default. Katalog yang perlu kita buat hanyalah katalog untuk bahasa Indonesia.

Untuk itu, Anda akan membutuhkan program gettext. Program ini tersedia dalam hampir semua distribusi Linux. Berikut ini adalah langkah-langkah dalam pembuatan katalog.

- pembuatan direktori bahasa untuk menampung katalog. Gunakan kode negara sesuai ISO 3166. Sebagai contoh, karena

kita akan membuat direktori untuk bahasa Indonesia, maka kita akan membuat direktori id di bawah direktori program kita.

```
$ mkdir id
$ cd id
```

- Program gettext selanjutnya akan mengambil teks di dalam program kita

```
$ xgettext -C -n -k_ -o
helloworld.po ../*.cpp
```

Perhatikan bahwa -C digunakan untuk menandakan bahwa kita ingin membuat katalog untuk bahasa pemrograman C++. Opsi -k digunakan untuk menandakan nama macro yang digunakan (dalam hal ini kita mempergunakan _). Opsi -o digunakan untuk nama file keluaran. Setelah program ini dijalankan, kita akan memiliki file helloworld.po di dalam direktori id kita.

- Mengedit file helloworld.po
Kita selanjutnya akan mengedit file hello-

word.po kita dengan editor favorit. Berikut ini adalah contoh helloworld.po milik penulis setelah diedit.

```
# SOME DESCRIPTIVE TITLE.
# Copyright (C) YEAR THE
# PACKAGE'S COPYRIGHT HOLDER
# This file is distributed
# under the same license as the
# PACKAGE package.
# FIRST AUTHOR
# <EMAIL@ADDRESS>, YEAR.
#
#, fuzzy
msgid ""
msgstr ""
"Project-Id-Version: PACKAGE
VERSION\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2004-12-
11 12:48+0700\n"
"PO-Revision-Date: YEAR-MO-DA
HO:MI+ZONE\n"
"Last-Translator: FULL NAME
<EMAIL@ADDRESS>\n"
"Language-Team: LANGUAGE
```

```
<LL@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain;
charset=CHARSET\n"
"Content-Transfer-Encoding:
8bit\n"
#: ../helloworld.cpp:11 ../
MainFrame.cpp:14
msgid "Hello World"
msgstr "Hello World"
#: ../MainFrame.cpp:20
msgid "&"
msgstr "&Tentang program"
#: ../MainFrame.cpp:20
msgid "Show about info..."
msgstr "Tampilkan informasi
program"
#: ../MainFrame.cpp:22
msgid "&Quit"
msgstr "&Keluar"
```

Program Regular & In-House Training

BERKUALITAS & TERPERCAYA

Internet, Pemrograman & DataBASE

- Internet Standard
- Desain WEB Standard (HTML)
- Desain WEB Terpadu (HTML, CSS, Javascript, PHP, MySQL)
- Pemrograman WEB dgn PHP
- Java Script
- Pemrograman Java (Fundamental)
- Pemrograman WEB dgn Java
- SQL dgn PostgreSQL
- Pemrograman Python

Aplikasi Bisnis

- Aplikasi Bisnis 1 (Windows, Word, Excel)
- Aplikasi Bisnis 2 (PowerPoint, Access, LAN, Internet)

Linux

- Linux Basic User & X-Window
- Aplikasi Perkantoran with OpenOffice
- Database dlm OpenOffice ("Access" for LINUX)
- Linux SysAdmin & Networking

Visual Basic 6.0

- VB Standard
- VB Intermediate + SQL

AutoCAD

- AutoCAD Fundamental
- AutoCAD 3D Special

Teknisi Komputer

- Merakit PC & Membangun LAN (Intranet)

PRODUK LAIN LAIN

- Program Profesi 1 & 2 Tahun
- Program Intensif Linux Profesional
- Kerjasama Pend. Komputer Sekolah
- System Support (Migrasi & Linux Solution)

JADUAL

- Senin s.d. Kamis
- Sabtu
- Minggu
- Jam: 08.00 s.d 12.00 WIB.
- Jam: 12.30 s.d. 18.00 WIB.

FASILITAS

- Ruang kuliah full AC ☺
- Tiap peserta 1 PC ☺
- Min PIII, Ram 128
- Network + Multimedia
- Internet ☺
- Modul pelatihan ☺
- Block notes ☺
- Disket ☺
- Sertifikat ☺



LEMBAGA PENDIDIKAN KOMPUTER NURUL FIKRI

<http://www.nurulfikri.com> | info@nurulfikri.com

- Jl. Margonda Raya No. 522 - Depok ☎/Fax. (021) 7874223-24, 77206991
- Jl. Mampang Prapatan X/4 Jakarta Selatan ☎ (021) 7947115, 7975235
- Jl. A. Yani - Sentra Niaga B.I/12, Bekasi ☎/Fax. (021) 78853537

LPKNF

```
#: ../MainFrame.cpp:22
msgid "Quit from
application..."
msgstr "Keluar dari aplikasi"

#: ../MainFrame.cpp:24
msgid "&File"
msgstr "&Berkas"

#: ../MainFrame.cpp:29
msgid "Halo"
msgstr "Hola hola"

#: ../MainFrame.cpp:44
msgid "Hello World Example
v0.1"
msgstr "Contoh Aplikasi Hello
World v0.1"

#: ../MainFrame.cpp:44
msgid "About Hello World"
msgstr "Tentang program Hello
World"

Membuat message catalog

$ msgfmt -o helloworld.mo
helloworld.po
```

Setelah perintah ini dilakukan, Anda akan mendapatkan sebuah file helloworld.mo. Ini adalah message catalog bahasa Indonesia untuk program ini.

Sampai di sini, pembuatan message catalog telah selesai. Anda dapat melakukan cara serupa untuk bahasa lain. Hanya, untuk cara yang lebih mudah, barangkali Anda ingin mempergunakan program poedit (<http://poedit.sourceforge.net>) yang dibuat oleh Vaclav Slavik <vaclav.slavik@matfyz.cz>, salah satu developer Wx. Dengan menggunakan poedit, proses pembuatan *message catalog* akan menjadi jauh lebih mudah.

Satu catatan. Terkadang, Wx sendiri juga memiliki teks-teks yang ditampilkan kepada user. Contoh yang paling baik adalah teks yang dipergunakan dalam Open file dialog. Kata-kata seperti *show hidden file* adalah bahasa Inggris. Dan, akan terlihat jelek sekali apabila suatu program menggunakan bermacam ragam bahasa sekaligus. Solusinya, dalam membuat message catalog, Anda juga perlu membuat katalog bahasa yang ingin

digunakan untuk standard wx (file-file .po ini datang bersama wx).

Langkah 3: memodifikasi program untuk menggunakan message catalog

Kita akan memodifikasi file helloworld.h menjadi berikut ini (penebalan pada perubahan atau penambahan):

```
#ifndef HelloWorld_H
#define HelloWorld_H

class wxHelloWorld : public
wxApp
{
public:
    virtual bool OnInit ();

protected:
    wxLocale locale;
};

DECLARE_APP (wxHelloWorld)

#endif // HelloWorld_H
```

Kita akan memodifikasi file helloworld.cpp menjadi berikut ini (penebalan pada perubahan atau penambahan):

```
#include <wx/wx.h>
#include "helloworld.h"
#include "MainFrame.h"

IMPLEMENT_APP (wxHelloWorld)

bool wxHelloWorld :: OnInit ()
{
    const wxString langs[] =
    {
        _("(System default)"),
        _("Bahasa Indonesia"),
    };

    SetExitOnFrameDelete(FALSE);

    int lng =
wxGetSingleChoiceIndex(_
("Please choose language:"),
_("Language"), WXSZEOF(langs),
langs);

    SetExitOnFrameDelete(TRUE);
}
```

```
switch (lng)
{
    case 0: locale.
Init(wxLANGUAGE_DEFAULT); break;
    case 1: locale.
Init(wxLANGUAGE_INDONESIAN);
break;
};

locale.AddCatalog(_
("helloworld"));

wxMainFrame *frame =
new wxMainFrame (_("Hello
World"), wxPoint (50,50), wxSize
(200,200), locale);

frame -> Show (TRUE);
SetTopWindow (frame);

return TRUE;
};
```

Kita akan memodifikasi file MainFrame.h menjadi berikut ini (penebalan pada perubahan atau penambahan):

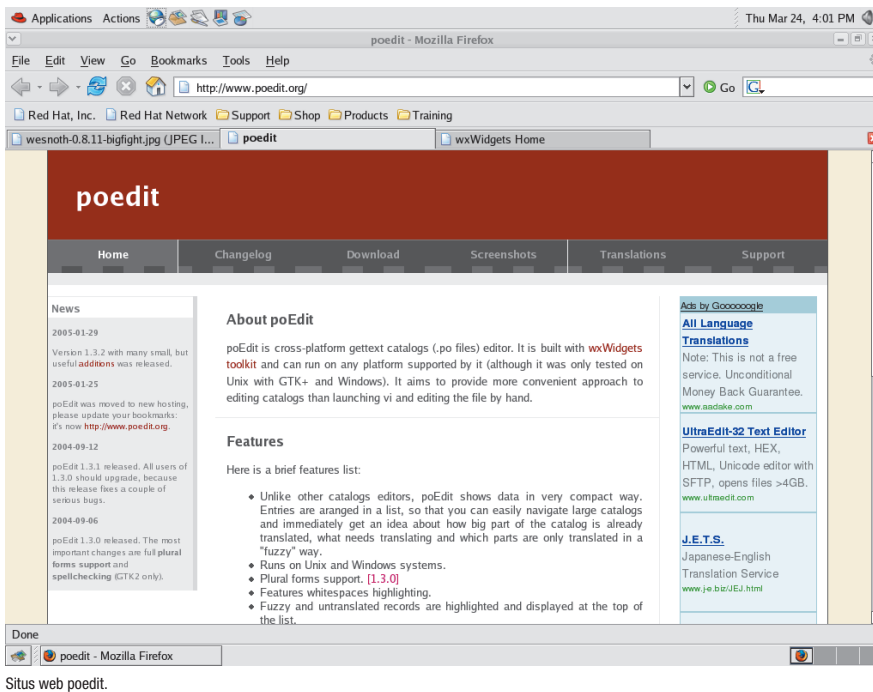
```
#ifndef MainFrame_H
#define MainFrame_H

enum
{
    MENU_QUIT,
    MENU_ABOUT,
};

class wxMainFrame : public
wxFrame
{
public:
    wxMainFrame (const
wxString &title, const wxPoint
&pos, const wxSize &size,
wxLocale &locale);

    void OnQuit
(wxCommandEvent &event);
    void OnAbout
(wxCommandEvent &event);
    wxLocale &locale;

protected:
```



```
DECLARE_EVENT_TABLE ();
```

```
private:
```

```
    wxMenuBar *menubar;
    wxMenu *FileMenu;
    wxTextCtrl *textctrl;
```

```
};
```

```
#endif// MainFrame_H
```

Kita akan memodifikasi file MainFrame.cpp menjadi berikut ini (penebalan pada perubahan atau penambahan):

```
#include <wx/wx.h>
```

```
#include "MainFrame.h"
```

```
BEGIN_EVENT_TABLE (wxMainFrame,
wxFrame)
```

```
    EVT_MENU (MENU_QUIT,
wxMainFrame :: OnQuit)
```

```
    EVT_MENU (MENU_ABOUT,
wxMainFrame :: OnAbout)
```

```
END_EVENT_TABLE ()
```

```
wxMainFrame :: wxMainFrame
(const wxString &title, const
wxPoint &pos, const wxSize
&size, wxLocale &l):
```

```
    wxFrame ( (wxFrame
*) NULL, -1, title, pos, size),
```

```
locale (l)
```

```
{
```

```
    CreateStatusBar (2);
```

```
    SetStatusText (_("Hello
World"));
```

```
    menubar = new wxMenuBar;
```

```
    FileMenu = new wxMenu;
    FileMenu -> Append (MENU_
ABOUT, _("&About"), _("Show
about info..."));
```

```
    FileMenu -> AppendSeparator
();
```

```
    FileMenu -> Append (MENU_
QUIT, _("&Quit"), _("Quit from
application..."));
```

```
    menubar -> Append
(FileMenu, _("&File"));
```

```
    SetMenuBar (menubar);
```

```
    textctrl = new wxTextCtrl
(this, -1, wxString (_
("Halo")), wxDefaultPosition,
wxDefaultSize, wxTE_MULTILINE);
```

```
    Connect (MENU_QUIT,
wxEVT_COMMAND_MENU_SELECTED,
```

```
(wxObjectEventFunction)
&wxMainFrame :: OnQuit);

    Connect (MENU_ABOUT,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxMainFrame :: OnAbout);
```

```
};
```

```
void wxMainFrame :: OnQuit
(wxCommandEvent &event)
```

```
{
```

```
    Close (TRUE);
```

```
};
```

```
void wxMainFrame :: OnAbout
(wxCommandEvent &event)
```

```
{
```

```
    wxMessageBox (_("Hello
World Example v0.1"), _("About
Hello World"), wxOK | wxICON_
INFORMATION, this);
```

```
};
```

Penjelasan:

- Secara umum, kita memang menambahkan atribut locale bertipe wxLocale ke dalam kedua class.
- Kita mengubah constructor untuk wxMainFrame dengan menambahkan atribut locale.
- Kita juga menambahkan pilihan bahasa pada saat program dijalankan.

Langkah 4: kompilasi, jalankan, dan selesai

Lakukanlah kompilasi seperti biasa. Pada saat program dijalankan, pemilihan bahasa akan disajikan kepada user. tergantung pada bahasa yang dipilih, program akan menggunakan katalog yang bersesuaian.

Program ini masih memiliki kekurangan. Hal ini karena beberapa bagian pada helloworld.cpp belum dimasukkan sebagai katalog.

Pada program sesungguhnya, bahasa yang dipilih oleh user akan disimpan di dalam file konfigurasi. Wx telah menyediakan pustaka untuk penanganan file konfigurasi.

Di edisi depan, kita akan melanjutkan dengan *tip of the day* dan *form MDI*.
Noprianto (noprianto@infolinux.co.id)

Wx dan Perancangan User Interface

Bagian 2 dari 2 Artikel

Pada bagian 1 yang lalu, kita telah membahas tentang i18n pada Wx. Di edisi lanjutan ini, kita akan membahas pembuatan *tip of the day* dan penggunaan *form* MDI.

Setelah membahas tentang bagaimana menyajikan berbagai bahasa dalam satu program, kita akan melanjutkan dengan bagaimana menyediakan *tip of the day* pada program Anda, serta bagaimana memanfaatkan *form* MDI. Sebagian contoh program yang dipergunakan akan merujuk kepada contoh program sebelumnya (yang mungkin telah Anda modifikasi).

Tip of the day

Banyak aplikasi ini hadir dengan tip penggunaan yang umum disebut sebagai tip of the day. Bahwa tip ini dibaca atau tidak oleh user, harus diakui bahwa penggunaan tip ini sudah merupakan kelengkapan dari sebuah aplikasi yang memperhatikan kemudahan penggunaan oleh user.

Bagi Anda yang memiliki waktu dan ingin melakukannya, tip of the day tentu saja dapat dengan mudah diimplementasikan menggunakan satu *form*, satu *text control* dan beberapa *button*. Dengan menggunakan *double linked list*, kita dapat mengimplementasikan penampilan tip-tipnya.

Namun, apabila Anda mempergunakan Wx, maka pengimplementasian sendiri sudah tidak diperlukan lagi karena tip of the day juga telah disediakan oleh Wx. Luar biasa sekali bukan? Cara penggunaannya juga sangat mudah.

Kita akan melengkapi aplikasi helloworld sebelumnya dengan implementasi tip of the day.

Langkah 1: mempersiapkan tip

Buatlah sebuah file text dan simpanlah sebagai tips.txt. Di dalam file ini, kita akan menuliskan masing-masing tip secara baris

per baris. Berikut ini adalah contoh tips.txt penulis:

```
ini adalah tip pertama
ini adalah tip kedua
ini adalah tip ketiga
ini adalah tip keempat
ini adalah tip kelima
```

Langkah 2: memodifikasi helloworld.cpp

Setelah file tip kita miliki, kita akan mengubah helloworld.cpp untuk mengimplementasikan tip of the day. Penambahan kode akan disajikan dalam huruf tebal.

```
#include <wx/wx.h>
#include <wx/tipdlg.h>
#include "helloworld.h"
#include "MainFrame.h"

IMPLEMENT_APP (wxHelloWorld)

bool wxHelloWorld :: OnInit ()
{
    const wxString langs[] =
    {
        _("(System default)"),
        _("(Bahasa Indonesia)"),
    };

    SetExitOnFrameDelete (FALSE);

    int lng =
    wxGetSingleChoiceIndex (
        ("Please choose language:"),
        _("(Language)"), WXSZEOF(langs),
        langs);

    SetExitOnFrameDelete(TRUE);
```

```
switch (lng)
{
    case 0: locale.
Init(wxLANGUAGE_DEFAULT); break;
    case 1: locale.
Init(wxLANGUAGE_INDONESIAN);
break;
};

locale.AddCatalog (
("helloworld"));

wxMainFrame *frame = new
wxMainFrame (_("Hello World"),
wxPoint (50,50), wxSize
(200,200), locale);

frame -> Show (TRUE);
SetTopWindow (frame);

wxTipProvider *tip =
wxCreateFileTipProvider (
("tips.txt"), 0);
bool showStart = wxShowTip
(frame, tip);
delete tip;

return TRUE;
};
```

Penjelasan:

dengan menambahkan penggunaan *header* tipdlg.h, kita sudah bisa membuat tip of the day dengan mudah (class wxTipProvider).

Dalam penggunaan tip of the day, kita juga dapat mempergunakan *internationalization*. Namun, hal ini membutuhkan banyak usaha dalam penyajian konten. Dari

sisi pemrograman, cara yang digunakan tidak banyak berbeda dengan contoh sebelumnya.

Form MDI

Pemilihan penampilan *window* juga merupakan hal yang penting dalam perancangan *user interface*. Apabila program kita memiliki kemampuan untuk menampilkan berbagai dokumen sekaligus, kita harus memperhatikan cara penampilannya. Apakah masing-masing dokumen dibuka dalam *window* terpisah, atau menggunakan pendekatan *Multiple Document Interface* (MDI).

Terkadang, apabila jumlah *window* yang dibuka terlalu banyak, user bisa saja menganggap suatu program kompleks dan membingungkan. Namun, penggunaan MDI juga terkadang memiliki isu soal mudahnya *switching* antardokumen (maka, sediakanlah *shortcut* yang memudahkan. *Shortcut default* seringkali menyebalkan).

Masalah ini bukanlah hal sepele. Pemikiran akan penyajian sistem penampilan dokumen ini juga merupakan salah satu alasan hadirnya tab *browsing*. Dulu, untuk membuka satu halaman web, kita akan membuka *window* baru. Namun, karena jumlah *window* yang dibuka bisa saja menjadi sangat banyak, maka desktop akan terlihat sangat tidak rapi. Terutama apabila Anda melakukan *browsing* sambil bekerja. *Switch* antar-aplikasi juga akan menjadi susah. Dengan adanya tab *browsing*, maka desktop akan terlihat lebih rapi.

Ketika Anda memutuskan akan menggunakan MDI, perhatikan juga konten yang akan ditampilkan. Apabila konten terlalu berat, usahakan untuk tidak menggunakan MDI. Bisa-bisa, aplikasi tidak mampu. Ketika diseparasi dalam aplikasi terpisah (bukan sekadar *group proses*), maka risiko *crash* karena tidak mampu akan lebih kecil.

Menggunakan *wx*, form MDI dapat diterapkan dengan cepat. Contoh aplikasi kita kali ini adalah aplikasi sederhana satu form, dengan sebuah menu, sebuah text control dan sebuah status bar. Ketika user memilih untuk membuat *window* baru dengan mengakses menu yang bersesuaian, maka *window* baru akan ditampilkan. Pada saat *window* baru dibuat, sebuah menu baru khusus untuk penanganan *child win-*

dow akan ditambahkan. Pengguna selanjutnya dapat pula mengubah *title window* baru ataupun menutup *window* baru tersebut. Program ini masih belum lengkap dan hanya menunjukkan kemampuan *wx* untuk bekerja dengan MDI.

Aplikasi kita, sebut saja *hello world MDI*, akan terdiri dari file-file sebagai berikut ini:

- *helloworld.cpp* sebagai tubuh class dari aplikasi.
- *helloworld.h* sebagai deklarasi class dari aplikasi.
- *MainFrame.cpp* sebagai tubuh class frame utama.
- *MainFrame.h* sebagai deklarasi class frame utama.
- *ChildFrame.cpp* sebagai tubuh class *window* anak.
- *ChildFrame.h* sebagai deklarasi class *window* anak.
- *makefile.unx* sebagai *makefile*.

Berikut ini adalah isi dari *helloworld.h*:

```
#ifndef HelloWorld_H
#define HelloWorld_H
```

```
class wxHelloWorld : public
wxApp
{
public:
    virtual bool OnInit ();

protected:
    wxLocale locale;
};

DECLARE_APP (wxHelloWorld)

#endif // HelloWorld_H
```

Berikut ini adalah isi dari *helloworld.cpp*:

```
#include <wx/wx.h>
#include "helloworld.h"
#include "MainFrame.h"

IMPLEMENT_APP (wxHelloWorld)

bool wxHelloWorld :: OnInit ()
```

MORE SPACE RELIABILITY & TIME & MONEY

LESS...

LINUX and FreeBSD

Features :

- Unlimited data transfer
- Complete control panels
- POP3 email, FTP access
- SSH, CGI, SQL.
- and much more...
- Start from Rp. 19.500,-/ month
- Free Setup *)
- 2 Months Free **)

Server Hosting

Features :

- Location NOC Jakarta - Indonesia (IIX)
- Size server : 1 U Rackmount
- Bandwidth : 128 kbps
- IP Address : 8 (max)
- Colocation : Rp. 1.000.000,-/ month

ALSO

- Colocation & Dedicated Server in USA
- Domain Name Register
- Benefit Reseller Program

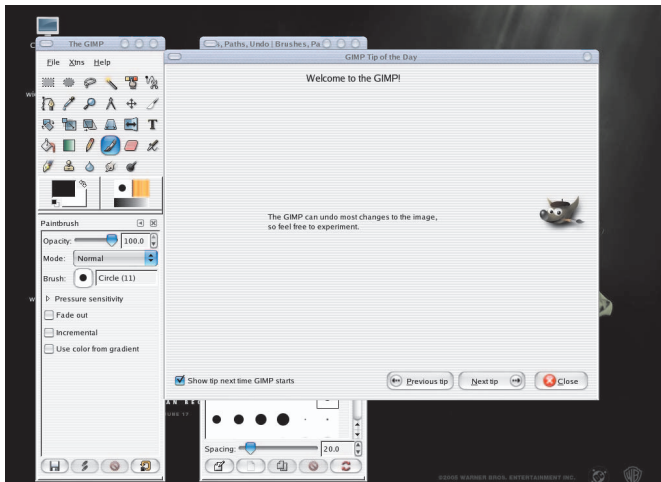
Limited Offer :
Dedicated Server
Rp. 1.250.000,-/ mo

**"IT'S NEVER BEEN EASIER
TO TAKE YOUR BUSINESS ONLINE"**

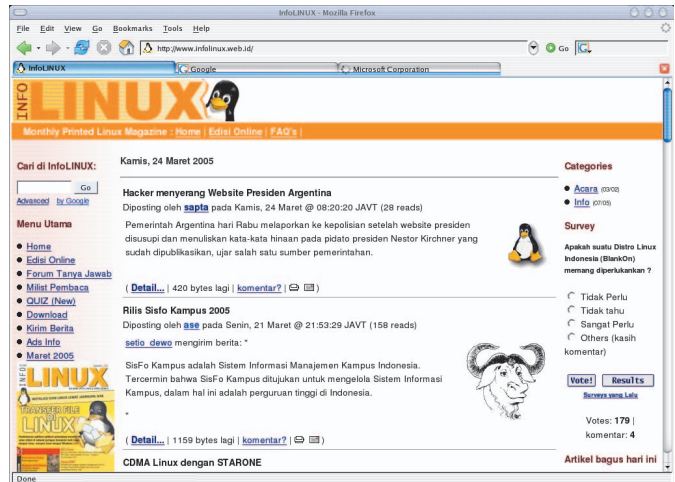
Note : *) Transfer (restriction apply)
**) 1 year payment

CAKRAWEB
Supporting You to a Web Success

Cyber Building (d/h Elektrindo) 10 th Floor
Jl. Kuningan Barat No. 8 Jakarta Selatan 12710
Phone. (021) 526 8000 Fax. (021) 52 66 444
<http://www.cakraweb.com> - info@cakraweb.com



Contoh tip of the day pada GIMP.



Contoh MDI pada Firefox.

```
{
    wxMainFrame * frame =
new wxMainFrame ( _("Hello World
MDI"), wxPoint (50,50), wxSize
(200,200), locale);

    frame -> Show (TRUE);
    SetTopWindow (frame);

    return TRUE;
};
```

Berikut ini adalah isi dari MainFrame.h:

```
#ifndef MainFrame_H
#define MainFrame_H

enum
{
    MENU_ABOUT = 100,
    MENU_NEW_WIN,
    MENU_QUIT,
};

class wxMainFrame : public
wxMDIParentFrame
{
public:
    wxMainFrame (const
wxString &title, const wxPoint
&pos, const wxSize &size,
wxLocale &locale);

    void OnQuit (wxCommandEvent
&event);
    void OnAbout
```

```
(wxCommandEvent &event);
    void OnNew (wxCommandEvent
&event);
    wxLocale &locale;

protected:
    DECLARE_EVENT_TABLE ();

private:
    wxMenuBar *menubar;
    wxMenu *FileMenu;
    wxMenu *ChildMenu;

};

#endif// MainFrame_H
```

Berikut ini adalah isi dari MainFrame.cpp:

```
#include <wx/wx.h>
#include <wx/mdi.h>
#include "MainFrame.h"
#include "ChildFrame.h"

BEGIN_EVENT_TABLE (wxMainFrame,
wxMDIParentFrame)
    EVT_MENU (MENU_QUIT,
wxMainFrame::OnQuit)
    EVT_MENU (MENU_ABOUT,
wxMainFrame::OnAbout)
    EVT_MENU (MENU_NEW_WIN,
wxMainFrame::OnNew)
END_EVENT_TABLE ()

wxMainFrame *frame =
(wxMainFrame *) NULL;
```

```
wxMainFrame :: wxMainFrame
(const wxString &title, const
wxPoint &pos, const wxSize
&size, wxLocale &l):
    wxMDIParentFrame
( (wxWindow *) NULL, -1, title,
pos, size), locale (l)
{
    CreateStatusBar (2);
    SetStatusText (_("Hello
World MDI"));

    menubar = new wxMenuBar;

    FileMenu = new wxMenu;
    FileMenu -> Append (MENU_
ABOUT, _("&About"), _("Show
about info..."));
    FileMenu -> AppendSeparator
();
    FileMenu -> Append (MENU_
NEW_WIN, _("&New"), _("New
window"));
    FileMenu -> AppendSeparator
();
    FileMenu -> Append (MENU_
QUIT, _("&Quit"), _("Quit from
application..."));

    menubar -> Append
(FileMenu, _("&File"));

    SetMenuBar (menubar);
```

```

        Connect (MENU_QUIT,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxMainFrame :: OnQuit);

        Connect (MENU_NEW_WIN,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxMainFrame :: OnNew);

        Connect (MENU_ABOUT,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxMainFrame :: OnAbout);

        frame = this;
};

void wxMainFrame :: OnQuit
(wxCommandEvent &event)
{
    Close (TRUE);
};

void wxMainFrame :: OnAbout
(wxCommandEvent &event)
{
    wxMessageBox (_("Hello
World MDI Example v0.1"),
_("About Hello World MDI"), wxOK
| wxICON_INFORMATION, this);
};

void wxMainFrame :: OnNew
(wxCommandEvent &event)
{
    wxChildFrame *child
= new wxChildFrame( frame ,
_("untitled"),wxDefaultPosition,
wxDefaultSize, wxDEFAULT_FRAME_
STY
LE, locale);
};

```

Berikut ini adalah isi dari ChildFrame.h:

```

#ifndef ChildFrame_H
#define ChildFrame_H

enum
{
    CHILD_QUIT = 200,
    CHILD_CHANGE_TITLE,
};

class wxChildFrame : public
wxMDIChildFrame

```

```

{
public:
    wxChildFrame
(wxMDIParentFrame *parent, const
wxString &title, const wxPoint
&pos, const wxSize &size, const
long s
tyle, wxLocale &locale);
    void OnQuit (wxCommandEvent
&event);
    void OnChangeTitle
(wxCommandEvent &event);
    wxLocale &locale;

protected:
    DECLARE_EVENT_TABLE ();

private:
    wxTextCtrl *textctrl;
    wxMenuBar *menubar;
    wxMenu *FileMenu;
    wxMenu *ChildMenu;
};

#endif// ChildFrame_H

```

Berikut ini adalah isi dari ChildFrame.cpp:

```

#include <wx/wx.h>
#include <wx/mdi.h>
#include "ChildFrame.h"
#include "MainFrame.h"

BEGIN_EVENT_TABLE (wxChildFrame,
wxMDIChildFrame)
    EVT_MENU (CHILD_QUIT,
wxChildFrame :: OnQuit)
    EVT_MENU (CHILD_CHANGE_
TITLE, wxChildFrame ::
OnChangeTitle)
END_EVENT_TABLE ()

wxChildFrame :: wxChildFrame
(wxMDIParentFrame *parent, const
wxString &title, const wxPoint
&pos, const wxSize &size, cons
t long style, wxLocale &l):
    wxMDIChildFrame ( parent,
-1, title, pos, size, style),
    locale (l)
{
    CreateStatusBar (2);

```

Linux bebas Pungli*



*Bebas Pungutan Lisensi

Linux™ menjamin kebebasan mempelajari source-codes, melakukan modifikasi, memanfaatkan dan meredistribusikan kembali, dengan atau tanpa imbalan!
[GNU General Public License]

Get The Software Freedom:

GudangLinux
Migration - Center
www.gudanglinux.net

```

        SetStatusText (_("Hello
World MDI"));

        textctrl = new wxTextCtrl
(this, -1, wxString (_
("Halo")), wxDefaultPosition,
wxDefaultSize, wxTE_MULTILINE);

        menubar = new wxMenuBar;

        FileMenu = new wxMenu;
        FileMenu -> Append (MENU_
ABOUT, _("&About"), _("Show
about info..."));
        FileMenu -> AppendSeparator
();
        FileMenu -> Append (MENU_
NEW_WIN, _("&New"), _("New
window"));
        FileMenu -> AppendSeparator
();
        FileMenu -> Append (MENU_
QUIT, _("&Quit"), _("Quit from
application..."));

        ChildMenu = new wxMenu;
        ChildMenu -> Append (CHILD_
QUIT, _("&Close"), _("Close
child"));
        ChildMenu -> Append
(CHILD_CHANGE_TITLE, _("C&hange
title"), _("Change title..."));

        menubar -> Append (FileMenu,
_("&File"));
        menubar -> Append
(ChildMenu, _("&Child"));

        SetMenuBar (menubar);

        Connect (MENU_QUIT,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxMainFrame :: OnQuit);
        Connect (MENU_NEW_WIN,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxMainFrame :: OnNew);
        Connect (MENU_ABOUT,
wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxMainFrame :: OnAbout);

        Connect (CHILD_QUIT,

```

```

wxEVT_COMMAND_MENU_SELECTED,
(wxObjectEventFunction)
&wxChildFrame :: OnQuit);
        Connect (CHILD_CHANGE_
TITLE, wxEVT_COMMAND_
MENU_SELECTED,
(wxObjectEventFunction)
&wxChildFrame :: OnChangeTitle);
    };

void wxChildFrame :: OnQuit
(wxCommandEvent &event)
{
    Close(TRUE);
};

void wxChildFrame ::
OnChangeTitle (wxCommandEvent
&event)
{
    static wxString s_title = _
("untitled");

    wxString title =
wxGetTextFromUser(_("Enter
new title"), _("Change
title"), s_title, GetParent()-
>GetParent());
    if ( !title ) return;
    s_title = title;
    SetTitle(s_title);
};

```

Berikut ini adalah isi dari makefile.unx:

```

CXX = $(shell wx-config --cxx)

PROGRAM = helloworld

OBJECTS = helloworld.o
MainFrame.o ChildFrame.o

# implementation

.SUFFIXES:      .o .cpp

.cpp.o :
    $(CXX) -c `wx-config --
cxxflags` -o $@ $<

all:    $(PROGRAM)

$(PROGRAM):    $(OBJECTS)
    $(CXX) -o $(PROGRAM)
$(OBJECTS) `wx-config --libs`

```

```
clean:
```

```
rm -f *.o $(PROGRAM)
```

Penjelasan:

- Class wxMainFrame kini diturunkan dari class wxMDIParentFrame, sementara class wxChildFrame diturunkan dari class wxMDIChildFrame. Hal ini telah disediakan oleh Wx.
- Class wxChildFrame memiliki atribut FileMenu dan ChildMenu bertipe wx-Menu. Pada saat child window dibuat (*constructor*), kita membuat ulang menu bar. Dalam pembuatan ulang tersebut, kita juga membuat menu ChildMenu. Dengan adanya proses berat di constructor wxChildFrame, maka pembuatan window baru dari class wxMainFrame akan terlihat sangat sederhana. Bagus untuk *information hiding*.
- Perhatikan konstanta yang digunakan untuk menu, dalam masing-masing header, enumerasi kini dimulai dari bilangan tertentu. Hal ini untuk mencegah tumpang tindih hasil enumerasi.
- Sampai saat ini, program ini masih memiliki kekurangan. Yang pertama adalah menu *Quit* yang hanya berfungsi untuk menutup *child window*. Harusnya, diimplementasikan untuk menutup aplikasi. Yang kedua, ketika kita klik pada tombol *close title bar*, aplikasi segera menutup walaupun masih ada child window yang terbuka. Yang seharusnya adalah mengonfirmasikan kepada user bahwa masih ada window yang terbuka. Yang ketiga, kita sebenarnya tidak mendaftar anak-anak window kita dengan benar. Pada aplikasi sesungguhnya, semua anak harus terdaftar dengan baik.

Demikianlah pembahasan kita tentang perancangan user interface menggunakan Wx. Tentu saja, artikel ini hanya membahas sedikit sekali dari aspek perancangan user interface dalam pembuatan aplikasi. Namun, mulai sekarang dan seterusnya, rasa-rasanya kita perlu banyak memperhatikan masalah user interface agar program yang kita hasilnya juga dapat dinikmati oleh banyak pihak. Canggih dan aman juga harus mudah dan enak digunakan. Ini penting sekali. ☺
Noprianto (noprianto@infolinux.co.id)