The WideStudio is an integrated development environment for free and of open source independent of encodings, platforms and operating systems. The WideStudio is second to any other RAD tools in its facility and usability. It suits a novice of window programming and an expert working on a commercial application for developing from a game programming to a business tool.

WideStudio unofficial Tutorial — 1.

1 What is the WideStudio?

The WideStudio is an integrated development environment working on Linux, Windows, FreeBSD, Solaris and many other unix-like platforms. The project was started in 1999 by Mr. Hirabayashi, and the latest version 3.00 is released in June 2002. In December 2000, the WideStudio won the prize of the best software in personal section from the Turbo Linux Japan for its artistic and superb production.

Here are the features of the WideStudio.

- 50 and more original class libraries

  The WideStudio serves you a lot of GUI components; dialogs, buttons, checkboxes, pullup/down menus, lists, tables and more and also menu useful Non-GUI components including timers and balloon-help. Of course, 3D graphics, database and networking components are also available in ver.3.00.

  Using the application builder of the WideStudio (Fig.1), you can make a window application with the components by just drag-and-drop.

- C++ programming with auto code generation

  The WideStudio requires codes written in C++, one of the most famous languages. Since most of the codes for initializing and displaying a window are generated automatically by the application builder, you don't need to write any other codes but for an event procedure code; the most important thing.

- Source code compatibility between Linux and Windows

  You can compile your codes of applications for the WideStudio without any kind of modifications. All GUI/nonGUI components are available in every platform and you don't need to worry about the compatibility.

  Some resources, for example, fonts and external libraries, are depend on the platform on which the application works. However, in principle the code you wrote needs no change.

- Support for UNICODE, EUC, SJIS and other encodings

  The WideStudio supports more than 20 encodings including UNICODE, EUC and SJIS for a encoding of program files and read/write data, so that you can handle a text written in multiple language (Japanese, Chinese, Korean or Russian). The encoding support enable you to use an EUC text even on Windows.

- Call external editor, compiler and debugger

  The WideStudio is an IDE, however, its doesn't have its own text editor, compiler or debugger. Therefore, you can use an editor you use and the application builder calls it.

  The package for Windows bundles MinGW as a compiler.

- Same Look&Feel on different platforms

  The WideStudio has its own Look&Feel to absorb a difference of appearance between Windows and X Window System, and window managers on X.

- Comprehensive online manuals

  Tue user's guide, the programming guide and the reference manual are available in HTML (English and Japanese), and in PDF (Japanese).

- Free software and Open source

  The WideStudio is a free software based on the MIT/X license, and the source codes of the WideStudio are open for anyone to learn how to design and produce a software of large-scale GUI system.

2 Install

2.1 Requirement

The WideStudio works on the following systems.

- Linux kernel 2.0 or later
- FreeBSD 4.0 or later
- Solaris 2.5 or later
- Windows95/98/ME/NT/2000/XP

Note that the WideStudio may work on other platform based on unix and X Window System (X11R6).

You need a text editor, a C++ compiler, and (optionally) a debugger. The binary package for windows bundles MinGW32 (compiler and debugger).

And also you need the following libraries for unix: Xt, libxpm and libjpg (necessary), libpng, OpenGL (Mesa), ODBC and PostgreSQL (optionally and detected by configure script).

2.2 Get the software

The WideStudio is available at the web sites `www.widestudio.org` or `sourceforge.net`. In June 2002, the latest version is version 3.00. All you need is, for unix, the tarball `ws-v3.00.tar.gz` which includes all source codes and online manuals written in English and Japanese For Windows, the zip file `wswin300-ee.zip` with complete executable binaries, online manuals written in English, and also source codes along with makefiles that enable us to make the WideStudio from the source codes.

Note that packages in ports for FreeBSD, rpm for Linux and tarball for Solaris are also available.

2.3 Install on Windows

Uncompress the archive for windows, `wswin300-ee.zip`, and a folder named "wsinst" is created. Then execute `setup.exe`, the installer, in the folder.

Press the button in dialogs displayed for asking the following.

- install directory
- web browser
- text editor
- license agreement for WideStudio
- license agreement for GCC/MinGW32

The install begins when you press the "finish" button.

After the installation, reboot the computer in order to make changes in AUTOEXEC.BAT or other settings effective. The shortcuts "wsbuilder" on the desktop and the start menu are used to launch the WideStudio.

You can delete the folder "wsinst" because executable binaries, compilers, manuals are stored in the install directory (`c:/Program Files/WideStudio/` is default).

Note

The installer installs a set of GCC/MinGW32, `gcc` and `make`. Therefore, it may be a trouble if you have installed before other `gcc` (e.g., of Cygwin) or `make` (of BCC). If you find such a trouble, 1) change the order of PATH in AUTOEXEC.BAT or in your environment, or 2) specify full path of compiler and make in the project settings of the application builder.

2.4 Install on unix

Uncompress the tarball, `ws-v3.00.1.tar.gz`, and follow the instruction. Note that `ws/` means the directory made when the tarball is uncompressed, or the install directory.

```
% gzip -cd ws-v3.00.1.tar.gz | tar xvf -
% cd ws/src
% ./configure -LANG=EE
% make
% make install
```

By default, the install directory is `/usr/local/ws`. You can change it with what you want by modifying the following line

```
WS_DEFAULT_DIR = /usr/local/ws
```

in `ws/sys/config/mkflags`. See `ws/README`.

The following examples of the environment settings in csh and bash are that you have to do next. An appropriate setting is needed for other shell.

- Add the following to `~/.cshrc` for csh.

  ```
  setenv WSDIR /usr/local/ws
  setenv PATH $PATH:/usr/local/ws/bin
  setenv LD_LIBRARY_PATH $LD_LIBRARY_PATH:/usr/local/ws/lib
  ```

otherwise

```
setenv WSDIR /usr/local/ws
set path=($path /usr/local/ws/bin)
setenv LD_LIBRARY_PATH /usr/local/ws/lib
```

then update the change by

```
% source ~/.cshrc
```

- Add the following to `~/.bashrc` for bash.

```
WSDIR=/usr/local/ws
PATH=$PATH:/usr/local/ws/bin
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/ws/lib
export WSDIR PATH LD_LIBRARY_PATH
```

then update the change by

```
% source ~/.bashrc
```

Now it's time to begin the WideStudio with the command `wsbuilder`.

```
% wsbuilder &
```

Note

You need libxpm, libjpg or libpng because the WideStudio uses the libraries for reading and displaying icons or image files. Check them if you have a problem about display.

OpenGL, or Mesa, is required for 3D graphics component (J3W is included in the distributed source code of the WideStudio). ODBC and PostgreSQL are necessary when you use database components.

3 Quick start

Now we begin to use the WideStudio. Right after the start you see the title splash window, then the application builder (Fig.1) which is the main place of a development appears. The menu bar and the tool bar are in the top two lines of the builder. In the left half side, there is the inspector that shows the hierarchy of components placed on a window. The right half side of the builder changes its function according to the selection of the tab. The property editor may be most frequently used throughout development. It shows name, width, height, color and many other properties of a component, and the changes of values in the property editor affect immediately the appearance of the window that you are editing, and vice versa.

Many files needed to create an application are controlled and generated by the builder as a unit of a project. For example, a dialog and a button are placed on an application window, the main window you are editing, and the builder records the information in the file of the application window. Note that usually an application needs an application window, and in a special case you may need more than one window because all of other window including dialogs are child of the application window.

A project comprises of a component on the application window, and an event procedure that is usually called a call-back function or an event handler and implemented by user; the event procedure is described later. To build, to compile all of the files and to make them an executable file, is one of tasks the application builder has to do.

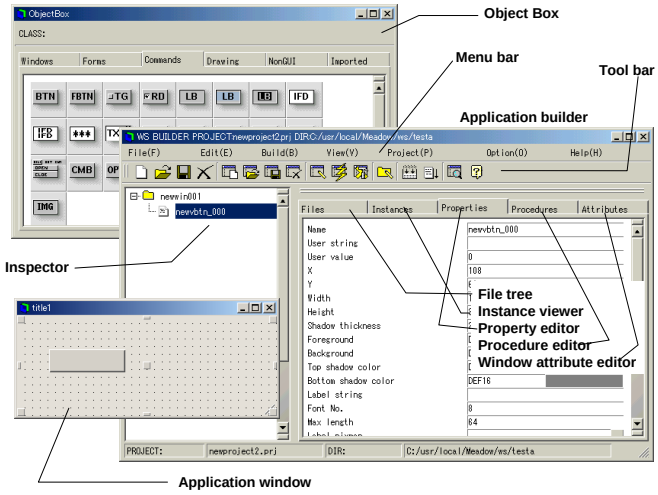

Figure 1: The application builder of the WideStudio

4 First Step toward to the WideStudio Programming

As the first step, an easy programming of a clock with an image on the background is described in this section. We don't give a full detail of the program, however, you can feel how you make an application with the WideStudio.

4.1 create a project

At first, create a "project" by selecting "Project" – "New project" on the menu bar or the "New project" icon on the tool bar. Then the project dialog will appear (Fig.3), and you enter the project name and the working directory, and press the "Next" button.

Next select the type of the application (Fig.4): usually we choose "Normal application". At last the locale type is specified (Fig.5). Select "English(ASCII)" and press the "Finish" button.

The project is created.

4.2 make a window

Second, make an application window, the main window of your application, by selection "File" – "New window" on the menu bar, or the "New application window" button on the tool bar. Then the window dialog (Fig.6) will appear. Choose "Normal window" for the "Type" and press "Next

Then check "Add to project" on "Project", and enter the name of the window; default name is "newwin000" as shown in Fig.7. Press "Next".

At last, select the "Empty" template (8) for this example, then press "Finish". The application window will appear as shown in Fig.9. You can change the size of the window by normal operation (Fig.10).

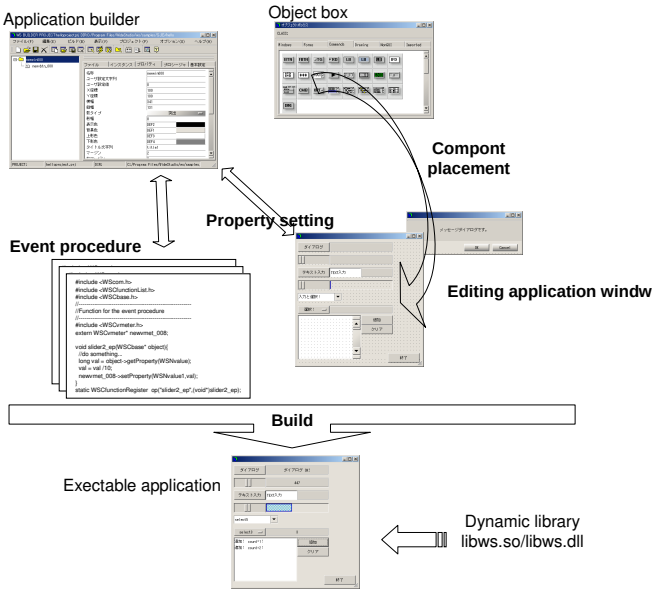

Figure 2: Overview of application development by the WideStudio

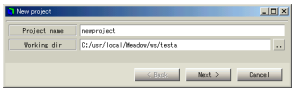

Figure 3: Project dialog

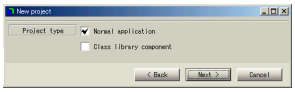

Figure 4: project type

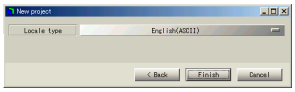

Figure 5: Locale type

4.3 attach a clock

Then, place a digital clock on the plane window.

See in the object box by selecting "View" – "ObjectBox" on the menu bar, or "Display the object box" icon on the tool bar (Fig.11). Select the "Commands" tab and choose the icon with green `12:00` on the black background: `WSCvclock`, the clock class.

Click the icon (the pointer changes to a cross), and drag onto the application window, then release the mouse. The clock is placed on the window where you release (Fig.12). Every component can be placed by drag & drop.

The size of the clock can be changed by pulling the squares on the edges and the corners (the pointer alters). By operating the square on the center, you can move the component.

4.4 modify the property

The clock works right after you put on the window, however, you can modify the properties of the clock component to change the appearance of the clock. The name of the instance of the WSCvclock component is "`newvclo_000`" by default. Select it on the inspector (left side of the builder) and see the property editor by press "Properties" tab. The following is an example for the clock `newvclo_000`.

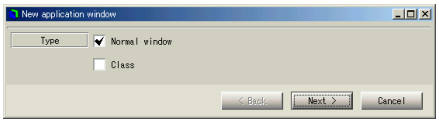

Figure 6: Window dialog

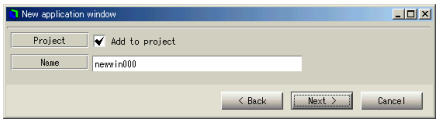

Figure 7: Add to project

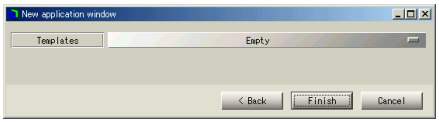

Figure 8: Template

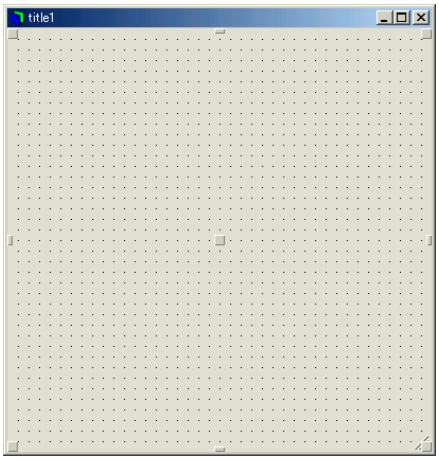

Figure 9: An application window

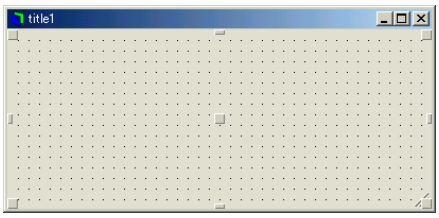

Figure 10: Change the window size

Property name	**Value**
Shadow type	None
Font No.	7
AnchorTop	On
AnchorBottom	On
AnchorLeft	On
AnchorRight	On
Date format	(None)
Week format	(None)
Clock format	23 00 00
Second	Visible

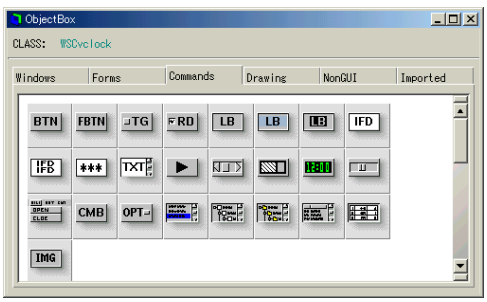

Figure 11: The object box

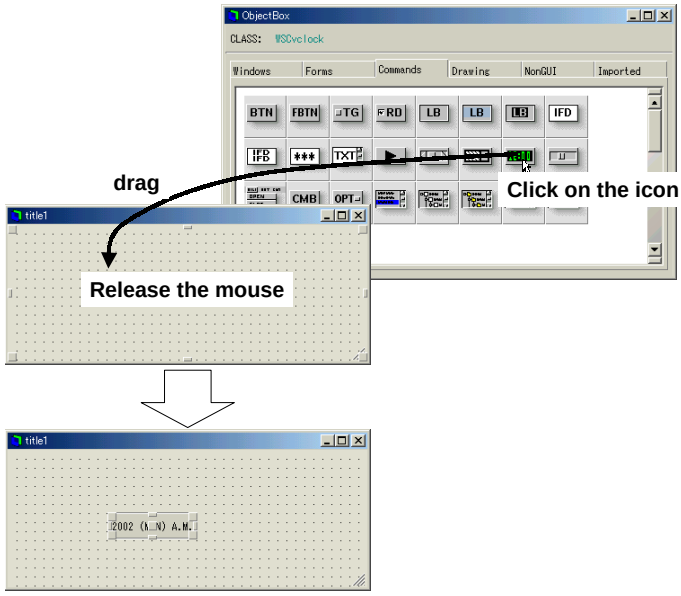

Figure 12: Component placement by drag & drop

Right after you set the properties, the application window has to be something like a clock. If you don't like the font displayed as the font no. 7, change the font by: "Project" – "project settings," then "Fonts" tab (Fig.13).

Next, specify the properties of `newwin000`, the window, as follows.

Property name	**Value**
Title string	Digital Clock
Pixmap style	Dynamic pixmap
Back pixmap	*see below*

To specify the image for the background, you can write the full path for the image in the "Back pixmap," or use the file select dialog (Fig.15) by pressing the [..] button in the right side (Fig.14). The image formats supported are xpm, jpg, bmp, and png.

After that, the image is drawn on the application window (Fig.16). If the size of the image is different from that of the window, adjust the window size. If you want to stretch the image and not to adjust the window size, use `WSCvimage`.

4.5 Build & Execution

What you have to do at the last is to build the application. Select "Build" – "Build all" on the menu bar, or "Build all" icon on the tool bar. Then the message box appears and the compilation messages are shown in it (Fig.17). After the compilation finished, select "Build" – "Execute" on the menu bar, or "Execute" icon on the tool bar, and the application will start (Fig.18).

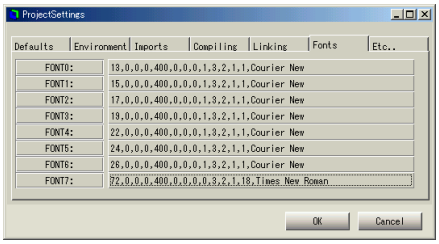

Figure 13: Font tab

Figure 14: Back pixmap property

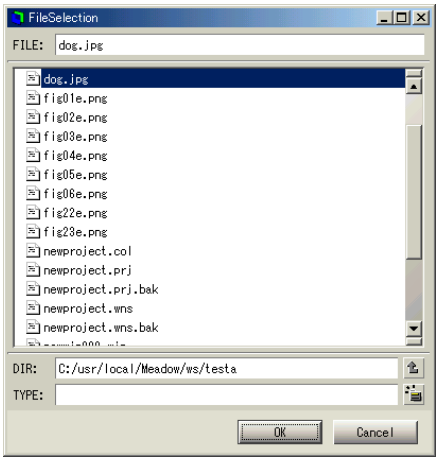

Figure 15: File selection dialog

Figure 16: Window with background image

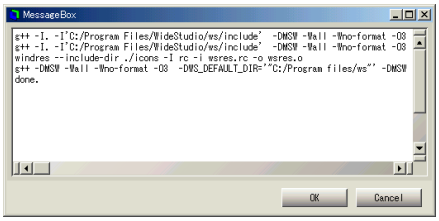

Figure 17: compilation message

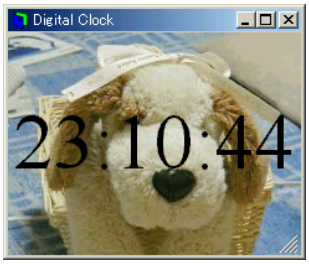

Figure 18: Execution

5 Programming of event procedure

As you see in the previous section, you can make an application without any programming. To make the application response events arising from mouse or keyboard, we need some short codes to write. For example, suppose that you want the application to show day and week when you press the mouse button. For this, you need to write only six lines of code to be added the template of the event procedure.

5.1 find a property

First, find an appropriate property of `WSCvclock` in the online manual by selecting "Help" – "Reference" on the menu bar. Then follow the links to "WSCvclock" – "Properties"

Now you can see the following properties.

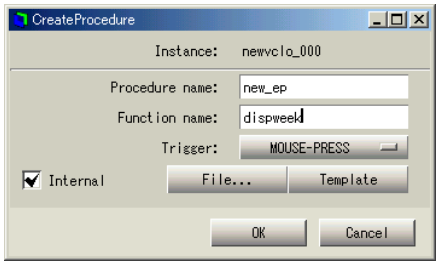

Figure 19: Create procedure window

name	setting	value
Date format	(None)	0
(WSNdateFormat)	2001/1/1	5
Week format	(None)	0
(WSNweekFormat)	(SUN)	2
Clock format	(none)	0
(WSNclockFormat)	23 00 00	5

What you do is the followings.

- When the mouse button is pressed, set the clock format to none(0), and date and week format to 5 and 2, respectively.
- When the mouse button is released, set the date and week format to none(0), and clock format to 5.

5.2 write a procedure function

At first, write a code to be called when the mouse button is pressed. In the WideStudio, a program that process events caused by a mouse or a keyboard is called an *event procedure*. Select "Procedures" tab on the right pane of the builder to launch the procedure editor.

Select `newvclo_000` in the inspector, then press the "Add procedure" icon on the procedure editor, and the create procedure window (Fig.19) popups. Leave "Procedure name" and fill in "Function name" with "dispweek", and select "MOUSE-PRESS" for the "Trigger". Finally, press "ok" button.

The procedure you have created is displayed on the procedure editor, and double-click the procedure or "Edit procedure" icon to edit the code with a text editor you specified at the installation. When the text editor is launched a template of the code is displayed on it. Add the following three lines in the code, then save it and quit the text editor.

```
#include <WScom.h>
#include <WSCfunctionList.h>
#include <WSCbase.h>
//-----------------------------------------------------------
//Function for the event procedure
//-----------------------------------------------------------
void dispweek(WSCbase* object){
  //do something...

  // the followings are to be added.
  object->setProperty(WSNdateFormat, 5);  // day format : 2001/1/1
```

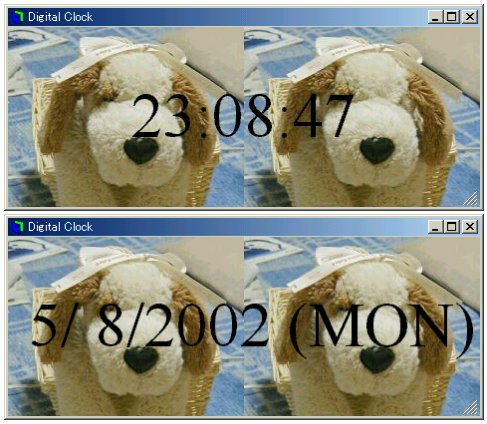

Figure 20: Click changes the display (top) before and (bottom) after

```
  object->setProperty(WSNweekFormat, 2);  // week format : (SUN)
  object->setProperty(WSNclockFormat, 0); // clock is not displayed.

}
static WSCfunctionRegister  op("dispweek",(void*)dispweek);
```

The first line added means to set the "WSNdateFormat" property of the object (`newvclo_000`) with 5, and the latter two lines are the same.

Then, write a code to be called when the mouse button is released. Go through the same step with the above: "disptime" for the function name, and "MOUSE-RELEASE" for the trigger, and the code is as follows.

```
#include <WScom.h>
#include <WSCfunctionList.h>
#include <WSCbase.h>
//-------------------------------------------------------------
//Function for the event procedure
//-------------------------------------------------------------
void disptime(WSCbase* object){
  //do something...

  // the followings are to be added.
  object->setProperty(WSNdateFormat, 0);  // date is not displayed.
  object->setProperty(WSNweekFormat, 0);  // week is not displayed.
  object->setProperty(WSNclockFormat, 5); // clock format : 23 00 00

}
static WSCfunctionRegister  op("disptime",(void*)disptime);
```

That's all for the programming. Build and execute the application. The digital clock changes its appearance when you click on the window (Fig.20).

6 Distribute

After you make an application, you want to distribute it. How do you package the files of the project?

6.1 source codes

The best way is to release the project itself.

Select "Project" – "Save project" or press "Save project" icon. If you are asked "Some application windows were modified. Save them?", press "ok" button. Then, select "Build" – "Clean" to remove files not necessary for the package.

You can use any archiver to compress the directory including the project as zip or tarball. Everyone that have the WideStudio can build it again.

6.2 binary

The package of the source codes is the smallest in file size, however, anyone who want to use it is required to install the WideStudio.

To distribute the executable binary files of the application, the dynamic/runtime libraries also have to be included.

The runtime DLLs needed for Windows are follows.

```
libws300.dll
libwsad300.dll
libwswin300.dll
libwsb300.dll
libwsj3w300.dll
libwsopengl300.dll
```

Note that glut32.dll and opengl32.dll are required for OpenGL.

These DLLs are installed in `C:\Windows\`, and you can find them in the directory `wsinst` when the WideStudio package is decompressed. You need to save the DLLs and the executable binary in the same directory.

The dynamic libraries (.so) needed for Linux and others are follows.

```
libws.so
libwsad.so
libwsx11.so
libwsb.so
libwsj3w.so
libwsopengl.so
jpg.so
xpm.so
png.so
```

Note that other related libraries are needed for OpenGL: e.g., Mesa3D.

These libraries are installed in `/usr/local/ws/lib` by default. When you execute the binary, they have to be saved in the directory included by LD_LIBRARY_PATH.

7 Know

7.1 Limitation

At the last but not least, it is better to know the limitation, or weakness, of the WideStudio.

First, to realize the multi-platform between X Window System and Windows, the WideStudio uses its own GUI components. Therefore, you may feel strange at the beginning if you are familiar with that of Windows, GNOME, and so on. The graphics drawing function is a little slow, but it is not a problem when you are using today's CPU.

Second, the number of the components of the WideStudio is relatively small as compared with other commercial IDEs. Especially, functions of sound, movie, and web are not supported, while the support of network, database, 3D graphics have began after ver.3.00-3. The development of the WideStudio is the cathedral of "the cathedral and the bazar," so the progress is not so fast.

Third, the most important and critical, is that the documents written in English are not well supplied. So, much support is welcome.

7.2 After

The latest version of the WideStudio is ver.3.00-3 2002/Aug/5, and many of feedbacks will be reflected on the future release. In Japan, the first book about the WideStudio is published 2002/Aug/27 by ASCII Publisher Co..

We hope you enjoy programming with the WideStudio.

Resources

[1] WideStudio web site

`www.widestudio.org`

[1] Sourceforge.net project

`sourceforge.net/projects/widestudio/`

[1] Mail list

[1] Mail list (in Japanese)

`www.egroup.co.jp/group/widestudio/`

[1] Independent JPEG Group

`www.ijg.org`

[1] libpng

`www.libng.org/pub/png/libpng.html`

[1] FreeBSD Ports

`www.freebsd.org/ja/ports/`

[1] Rpmdind.Net

`rpmfind.net`

[1] sunfreeware.com

`www.sunfreeware.com`

[1] The Mesa 3D Graphics Library

`www.mesa3d.org`

[1] 3D Polygon Animation Kit : J3W

`www.nk.rim.or.jp/ jun/j3w/`

[1] Debian Package

`www.debian.org/distrib/packages/`