GNUstep HOWTO

Installing the GNUstep developement system

This document explains how to build the different components of the GNUstep core libraries.

Last Update: 9 June 2004

Table of Contents

1 Introduction

This document explains how to build the GNUstep core libraries. The core libraries, along with associated tools and other files provide everything necessary for a working GNUstep system.

In order to easily compile and debug GNUstep projects, you will need the GNU Objective-C compiler 'GCC' as well as various other GNU packages.

You will need at least 80Mb of hard disk space (150Mb prefered) in order to compile the GNUstep core libraries.

2 Summary

In order to compile the libraries, you need to compile and install the following packages first (if you don't already have them):

- gcc (Version 2.95 or greater, 3.0.4 or greater recommended)
- GNU make (Version 3.75 or greater)
- gdb (Version 6.0 or greater recommended), if you plan to do any debugging

You may also need to install some of the following libraries and packages described below. Most of these packages are optional, but some are required.

'`ffcall libraries (HIGHLY RECOMMENDED)`'
: This is a library that provides stack-frame handling for NSInvocation and NSConnection. This library is highly recommended. The previous builtin method for stack frame handling is no longer supported and may be removed in the future. ffcall is under GNU GPL. As a special exception, if used in GNUstep or in derivate works of GNUstep, the included parts of ffcall are under GNU LGPL.

'`libffi library (ALTERNATIVE RECOMMENDATION)`'
: This is a library that provides stack frame handling for NSInvocation and NSConnection similar to ffcall. Use this instead of ffcall. You don't need both.

'`libxml2 (RECOMMENDED)`'
: The libxml library (Version 2) is used to translate some of the documentation for GNUstep and to provide suport for MacOS-X compatible XML-based property-lists. It is recommended but not currently required.

'`libxslt (OPTIONAL)`'
: Stylesheet support for use with XML.

'`openssl (OPTIONAL)`'
: The openssl library is used to provide support for https connections by the NSURL and HSURLHandle classes. This functionality is compiled as a separate bundle since the OpenSSL license is not compatible with GPL, and in the hopes that if someone writes an openssl replacement, it can quickly be used by creating another bundle.

'`libiconv (OPTIONAL)`'
: Note: Do not install this library unless you are sure you need it. You probably don't need it except perhaps on MinGW. Unicode support functions (iconv) come with glibc version 2.1 or greater. If you don't have glibc (try iconv –version), you can get the separate libiconv library from

`http://clisp.cons.org/~haible/packages-libiconv.html`. However, neither one is required to use GNUstep.

'The TIFF library (libtiff) (Version 3.4beta36 or greater) (REQUIRED)'
: The GUI library uses this to handle loading and saving TIFF images.

'The JPEG library (libjpeg) (RECOMMENDED)'
: The GUI library uses this to handle loading JPEG images.

'The PNG library (libpng) (RECOMMENDED)'
: The GUI library uses this to handle loading PNG images.

'libaudiofile (RECOMMENDED)'
: The GUI library uses this for the sound server

'freetype2 (RECOMMENDED, REQUIRED for art backend)'
: This is used for font information

'libart_lgpl2 (REQUIRED for art backend only)'
: Drawing library for the art backend.

'WindowMaker (Version >= 0.62) (OPTIONAL)'
: GNUstep and WindowMaker work together to provide a consistant interface. Although it is not required, GNUstep will work much better if you use it with the WindowMaker window manager. Get WindowMaker from `http://www.windowmaker.org`.

'gnustep-objc package (for gcc version < 3.0 ONLY) (RECOMMENDED)'
: Note: Do not install this library unless you are sure you need it. You probably don't need it except on MinGW. This is a special version of the Objective-C runtime that include several bug fixes and features that were not in gcc versions previous to 3.0. It is available at `ftp://ftp.gnustep.org/pub/gnustep/libs` which compiles using the GNUstep Makefile package (so you don't have to get the entire gcc dist). Make sure to set the THREADING variable in the GNUmakefile. It's possible to compile the library static (make shared=no) and just copy to the place where the gcc libobjc library is (type gcc -v to get this location). Note you have to install gnustep-make (below) before installing this library.

'GDB (OPTIONAL)'
: GDB can be obtained from `ftp://ftp.gnu.org/gnu/gdb`. As of release 6.0, gdb has special support for debugging Objective-C programs.

'TeX (OPTIONAL)'
: You need a TeX implementation, like tetex, to compile some of the documentation (although most of that is available on the web).

3 Compiling and Installing the packages

Get the following individual packages:

- gnustep-make
- gnustep-base
- gnustep-gui
- gnustep-back

See `http://www.gnustep.org` for information on where to get these packages.

Make sure you install (if necessary) all the previously mentioned libraries first before configuring and building GNUstep.

You should install these packages as root (read special note for the gnustep-base library, below, if you cannot do this).

For installation on specific systems, read the machine specific instructions at the end of this document or appropriate README files in the gnustep-make Documentation directory (such as README.MingW for Windows).

3.1 Installing the Core Libraries

The GNUstep packages uses the Autoconf mechanism for configuration; it checks some host capabilties which are used by all GNUstep software. To configure just type:

```
./configure
```

The GNUstep makefile package needs a root directory. If the GNUSTEP_SYSTEM_ROOT environment variable is set then configure will determine the root directory from its value (by removing the final /System path component from it). You can also specify the root directory when you run configure with the prefix paramater; the following command makes /usr/local/GNUstep the root directory:

```
./configure --prefix=/usr/local/GNUstep
```

If you do not have the GNUSTEP_SYSTEM_ROOT environment variable set and you do not specify a root directory when running configure, then configure will use /usr/GNUstep as the default root directory.

3.1.1 Alternate Library Setup

Read the installation instructions in the Makefile package (make) for more installation options. Make sure you use the same configuration options when configuring each GNUstep library.

3.1.2 Building the Package

To build the individual packages, use this familiar set of commands for each pacakge (add any additional options you decide upon):

```
./configure
make
make install
```

Start with the Makefile Pacakge (gnustep-make). After installing gnustep-make you need to execute GNUstep's shell configuration script, as follows:

```
 . /usr/GNUstep/System/Library/Makefiles/GNUstep.sh
```

before proceeding any further.

NOTE for gcc 2.X or MinGW users: Now install gnustep-objc. Before building gnustep-objc, edit the '`GNUmakefile`' and set the *THREADING* variable to the thread library used on your system (usually its posix, but you can leave it at single if you don't need threads). At this point you should probably re-configure, make and install gnustep-make, so it can pick up on any threading information that gnustep-objc provides.

Now install gnustep-base, gnustep-gui and finally gnustep-back.

NOTE: If you are trying to install the packages without root permission, you may need to change one thing in the base library. Edit the file gnustep-base/Tools/gdomap.h to uncomment the last line and modify the specified port number to a port which you *know* is not in use on your network. You should only do this if absolutely necessary since making this change will break communications with any systems where an identical change has not been made. Also, the standard gdomap port is the one officially registered with IANA and is reserved for use by gdomap - it should only be changed if you can't get your system administrator to start the gdomap server using it.

4 Additional Installation

4.1 Environment Setup

Add the shell script '`GNUstep.sh`' located in the Makefile package to your shell startup file (such as '`.profile`'). For instance, if your GNUSTEP_SYSTEM_ROOT is '`/usr/GNUstep/System`',

```
. /usr/GNUstep/System/Library/Makefiles/GNUstep.sh
```

in your '`.profile`' file will work (Note the period at the beginning of the line, and the space between the period and the following path; if your GNUSTEP_SYSTEM_ROOT is different, you need to replace '`/usr/GNUstep/System`' with your GNUSTEP_SYSTEM_ROOT). It defines environment variables that are needed to find GNUstep files and executables. Users of csh need to use the '`GNUstep.csh`' script. Read the make package '`README`' for more info. Some systems, like GNU/Linux have an '`/etc/profile.d`' directory where scripts can be executed automatically. If you want to set up GNUstep for every user on your system, you can try copying/linking the '`GNUstep.sh`' there. For csh or tcsh, try

```
source /usr/GNUstep/System/Library/Makefiles/GNUstep.csh
```

4.2 GNUstep Home

Set up your home GNUstep directory. This should be done automatically if you don't do it. This is where user defaults are kept as well as other user configuration files. User installed apps, libraries, etc are also here (if the default user directory is used).

```
cd
mkdir GNUstep
```

4.3 Time Zone

Next, set your local time zone. There are four ways to do this, pick one (see '`$GNUSTEP_SYSTEM_ROOT/Library/Libraries/Resources/gnustep-bsae/NSTimeZones/zones`' for typical time zones):

1. Use the defaults utility to set "Local Time Zone" to your local time zone (defaults is installed with GNUstep in the Tools directory). Type something like "defaults write NSGlobalDomain "Local Time Zone" GB".
2. Set the *GNUSTEP_TZ* environment variable.
3. Create the file '`$GNUSTEP_SYSTEM_ROOT/Library/Libraries/Resources/gnustep-base/NSTimeZones/localtime`' with the name of the local time zone in it.

4. Set the *TZ* environment variable (this may conflict with other software on your system though).

4.4 GNUstep deamons

Set up your system to execute some GNUstep deamons. If you don't do this, they will be started automatically when you run your first GNUstep app:

- gdomap - Put this in a system startup file, like '`/etc/rc.local`' or '`/etc/rc.d/rc.local`' (customize for your system)

  ```
  GNUSTEP_SYSTEM_ROOT=/usr/GNUstep/System
  if [ -f $GNUSTEP_SYSTEM_ROOT/Tools/gdomap ]; then
    $GNUSTEP_SYSTEM_ROOT/Tools/gdomap
  fi
  ```

- gdnc - Start after sourcing '`GNUstep.sh`' (e.g. in .profile)
- gpbs - Same as with gdnc, make sure X-Windows is running.

  ```
  if [ `gdomap -L GDNCServer | grep -c Unable` == 1 ]; then
    echo "Starting GNUstep services..."
    gdnc
    gpbs
  fi
  ```

5 Test Tools and Applications

Test programs for the base library are in '`base/Testing`'. Example applications are located in the gstep-examples package. To make these, just uncompress and untar this package, cd to the appropriate directory, and type make. You will need to install the GNUstep core libraries first before doing this.

To run the examples. Use the openapp utility that is part of the GNUstep makefile package (and stored in '`$GNUSTEP_SYSTEM_ROOT/Tools`'). Usage is:

```
openapp application [additional arguments to app]
```

Good Luck!

6 Machine Specific Instructions

\input texinfo

Below is a list of machines that people have attempted to compile GNUstep on. GNUstep compiles with little or no trouble on many of the more popular operating systems. Some machines marked with *Unstable* may have some trouble or may not work at all. Platforms marked *Needs Testing* are not actively tested by developers and need someone to help with reporting problems and fixes. Platforms marked *Obsolete* are very old distributions. No one really knows if GNUstep works on these although they may.

A recommended compiler is listed for each machine, if known. You should try to use the recommended compiler for compiling GNUstep, as GNUstep is quite complex and tends provoke a lot of errors in some compilers. Even versions newer than the listed compiler may not work, so don't just get the latest version of a compiler expecting it to be better than previous versions.

Compiler notes: If a recommended compiler is not listed, take note of the following information before choosing the compiler you use.

'`egcs or gcc < 2.95`'
: Might work, but few people use these now.

'`gcc 2.95.x`'
: Probably the oldest compiler that GNUstep is regularly tested with.

'`gcc 2.96`'
: Not an official gcc release. Some versions (Redhat, Mandrake) have problems that prevent GNUstep from being compiled correctly and cause mysterious errors.

'`gcc 3.0.x`'
: A fairly good compiler. Recommended.

'`gcc 3.1`'
: Several bugs where introduced in the version. It's probably better to avoid this one, although it might work fine.

'`gcc 3.2.x`'
: Pretty good. Recommended.

'`gcc 3.3.x`'
: Recommended. Fixes some bugs relating to protocols as well as other improvements.

'`gcc 3.4`'
: No info yet.

If you have compiled GNUstep on a specific machine, please send information about what you needed and any special instructions needed to GNUstep `bug-gnustep@gnu.org`.

If your having mysterious trouble with a machine, try compiling GNUstep without optimization. Particularly in the newer GCC compilers, optimization can break some code. The easiest way to do this is when configuring, '`CFLAGS="" ./configure`'. Or when building, '`make OPTFLAG=""`'.

Also if you manually upgraded gcc and/or make, we recommend reading the documentation at `http://www.LinuxFromScratch.org` for tips on compiling and installing gcc and make. If you had GNUstep previously installed, make sure you completely remove all of it, including installed init scripts.

6.1 Darwin/ix86

Currently tested on Darwin 7.x

'`Recommended compiler`'
: gcc 3.3.2. Older versions will not compile on Darwin.

  Default compiler has unknown problems. Use the GNU runtime. Download the gcc compiler and configure it with –enable-threads=posix. You don't need binutils or anything else.

'`Extra libs needed`'
'`Special Instructions`'
: Read the `README.Darwin` file in the gnustep-make/Documentation directory for complete instructions.

6.2 Darwin/PowerPC

Currently tested on Darwin 6.x, 7.x

'`Recommended compiler`'
: gcc 3.3.2. Older versions will not compile on Darwin.

  Default compiler has unknown problems. Use the GNU runtime. Download the gcc compiler and configure it with –enable-threads=posix. You don't need binutils or anything else.

'`Extra libs needed`'
: Use libffi (not ffcall). This should be enabled by default in gnustep-base so you don't have to type –enable-libffi. For 6.x, you need the dlcompat library (from `www.opendarwin.org`) to load bundles (not needed for 7.x).

'`Special Instructions`'
: Read the `README.Darwin` file in the gnustep-make/Documentation directory for complete instructions.

See also the MacOSX/PowerPC section

6.3 Debian/DEC-Alpha

'`Recommended compiler`'
: Unknown

'`Extra libs needed`'
: Unknown

'`Special Instructions`'
: None

6.4 Debian/i386

Tested on sid.

'`Recommended compiler`'
: Unknown

'`Extra libs needed`'
: Unknown

'`Special Instructions`'
: None

6.5 Debian/PowerPC

Tested on sid.

'`Recommended compiler`'
: Unknown

'`Extra libs needed`'
: Unknown

'`Special Instructions`'
: None

6.6 Debian/SPARC

Tested on sid.

'`Recommended compiler`'
: Unknown

'`Extra libs needed`'
: Unknown

'`Special Instructions`'
: None

6.7 FreeBSD 5.x

Tested on 5.0 and 5.1

'Recommended compiler'
'Extra libs needed'
None.

'Special Instructions'
Can install via /usr/ports/devel/gnustep, but not all required dependancies are installed. See the GNUstep-HOWTO for list of libraries.

6.8 FreeBSD 4.x

'Recommended compiler'
'Extra libs needed'
Unknown

'Special Instructions'
If you're using zsh, some shell variables may not be set correctly when executing GNUstep.sh. Try turing on SH_WORD_SPLIT (e.g. 'zsh –sh-word-split', 'zsh -o shwordsplit', 'zsh -y' or by setting 'setopt SH_WORD_SPLIT' in one of zsh's startup files.)

For gcc 3.0.4, make WANT_THREADS_SUPPORT=YES

For libxml2 2.4.24, make WITHOUT_PYTHON=YES

6.9 FreeBSD 3.x

Compiles "out of the box" on FreeBSD 3.4.

'Recommended compiler'
gcc 2.95.2

'Extra libs needed'
Unknown

'Special Instructions'
You need to use gmake not make to compile the GNUstep packages. A special port of gdb can be used with the Objective-C patches from `ftp://ftp.pcnet.com/users/eischen/FreeBSD/gdb-4.17-port.tar.gz`

The best compiler for GNUstep is the latest release of the GNU Compiler Collection (GCC). You can find it at `http://egcs.cygnus.com/`.

If you want to use the native POSIX threads support from 'libc_r' pass --enable-threads=posix to configure. This is the recommended option as this is the FreeBSD threads package that gives the best results –with others you may be unable to run some examples like 'diningPhilosophers'.

The whole compilation process can fail if you have another threads library installed so watch out for installed packages like 'pth' and such. Besides the support for libc_r, GNUstep will also look for 'pth' and 'pcthreads', so if you have installed them and they aren't detected prepare to write a nice bug report.

This can be done more much easily by using the port version. Just cd to '/usr/ports/lang/egcs' and do a "make WANT_THREADS=yes install". Easy.

If configure cannot find tiff.h or the tiff library and you have it installed in a non-standard place (even '/usr/local'), you may need to pass these flags to configure: CFLAGS="-I/usr/local/include" and LDFLAGS="-L/usr/local/lib".

6.10 FreeBSD 2.x (*Obsolete, Unstable*)

'Recommended compiler'
: gcc 2.8.x

'Extra libs needed'
: Unknown

'Special Instructions'
: Only static libraries work on this system. Use /stand/sysinstall to install these packages if you have not already done so:

gmake (GNU make)
gcc 2.8.x

Seems to compile ok, but some tests crash. Possibly due to a performace 'hack' in base. Might be a good idea to upgrade to FreeBSD 3.x. You need to use gmake not make to compile the GNUstep packages.

6.11 Irix 6.5/MIPS

'Recommended compiler'
: gcc 3.2.1

To use threads, it's necessary to bootstrap a compiler yourself: configure with –enable-threads=posix, that will work as long as you link EVERY objective C executable with -lpthread, no matter what warnings the irix linker produces!

'Extra libs needed'
: Unknown

'Special Instructions'
: If you cannot link the library because of the very low default limit (20480) for the command line length, then you should either use systune ncargs to increase the value (maximum is 262144) or link the library by hand. No libffi-support: Use ffcall

6.12 MacOSX/PowerPC

Currently tested on MacOSX 10.2, 10.3

'Recommended compiler'
: Default.

'Extra libs needed'
: None.

'Special Instructions'
: By default, on Mac OS X, only the GNUstep extensions are built. This is if you want to build gdl2, etc on Mac OS X. Xcode project files exist, but they may not be up-to-date. Try:

```
cd make
./configure --with-library-combo=apple-apple-apple
make install

cd ../base
./configure --with-xml-prefix=/usr --disable-xmltest
make debug=yes install
```

See also the Darwin/PowerPC section.

6.13 MkLinux/PowerPC

Tested with R2 RC2 (2004/03/04).

'Recommended compiler'
: gcc 3.x allthough earlier ones should work if you also install gnustep-objc

'Extra libs needed'
: None.

'Special Instructions'
: Unknown.

6.14 NetBSD

'Recommended compiler'
: Unknown

'Extra libs needed'
: libiconv

'Special Instructions'
: See the `README.NetBSD` file located in the gnustep-make package.

6.15 Netwinder (*Unstable*)

'Recommended compiler'
: Build #12 of the system.

'Extra libs needed'
: Unknown

'Special Instructions'
: See `http://www.netwinder.org/~patrix`

6.16 OpenBSD 3.x (*Needs Testing*)

'Recommended compiler'
: Unknown

'Extra libs needed'
: Unknown

'Special Instructions'
: Try reading the `README.NetBSD` which might work the same on OpenBSD.

6.17 OSF Alpha (*Unstable*)

Information is for Version 3.2C

'Recommended compiler'
: egcs-1.1.1/1.1.2, gcc-2.95

'Extra libs needed'
: Unknown

'Special Instructions'
: Can only compile with static libraries. Compiler may fail when linking executables (e.g. gdnc). Standard ranlib and ar programs are to feable to create libraries. Should use GNU binutils

versions. Linker sometimes fails to find symbols, in which case you may need to link with a library twice. For instance, add an extra -lgnustep-gui in ADDTIONAL_TOOL_LIBS in the GNUmakefile(.preamble).

6.18 RedHat/Intel

'`Recommended compiler`'
: Standard

'`Extra libs needed`'
: Standard (ffcall or libffi)

'`Special Instructions`'
: None

6.19 Slackware/Intel

'`Recommended compiler`'
: Unknown.

'`Extra libs needed`'
: Unknown.

'`Special Instructions`'
: Unknown.

6.20 Solaris 2.5.1/Sparc (*Obsolete*)

This configuration is no longer being tested, but it may still work.

'`Recommended compiler`'
: Unknown

'`Extra libs needed`'
: tiff, Don't use the one in /usr/openwin

'`Special Instructions`'
: See the Solaris 2.6 section for more instructions.

6.21 Solaris 2.[67]/Sparc

'`Recommended compiler`'
: gcc 3.2.1 gcc 2.95.3. Version 2.95.2 has several bugs that GNUstep tickles. gcc 3.04. Not 3.1 - does not compile parts of GNUstep.

'`Extra libs needed`'
: tiff, Don't use the one in /usr/openwin

'`Special Instructions`'
: No libffi-support unless you use the patch `http://gcc.gnu.org/ml/java-patches/2002-q3/msg00036.html` (not particularly recommended).

  Using a POSIX shell (zsh or bash, which should come with Solaris) is highly recommended. In fact, some functions, such as compiling frameworks, will not work without it.

Some people have reported problems when using binutils assembler and linker. Using the native Solaris assmebler and linker should work fine.

Older Instructions: If you are using threads, make sure the Objective-C runtime (libobjc that comes with gcc) is compiled with threads enabled (This is true by default) AND that it is compiled with the _REENTRANT flag defined (This does not seem to be true by default). Or use the gnustep-objc package. Also make sure THREADS is set to 'posix' not 'solaris'.

6.22 Solaris 2.7/Intel

'`Recommended compiler`'
: Unknown.

'`Extra libs needed`'
: Unknown

'`Special Instructions`'
: Make sure there are no -g compiler flags (i.e. compiling with debug=yes might be a problem). Unsure of correct bundle flags - You might need to use the alternate flags listed in target.make, line 989. Also, configuring gnustep-make with '`--disable-backend-bundle`' might be necessary if you can't get bundles to work. You will probable get a lot of text relocation warnings, which probably can be ignored. See the other Solaris instructions above for more information.

6.23 Suse 6.x/Intel

GNUstep has been tested on version 6.2-6.4 of Suse

'`Recommended compiler`'
: Standard

'`Extra libs needed`'
: None

'Special Instructions'
: It seems that there is a problem with the default kernel build distributed with Suse which means that the socket binding used by gdnc doesn't work. If you recompile the kernel then it starts working.

6.24 Suse/Intel

GNUstep has been tested on version 7.0, 8.0, 8.1, 8.2, 9.0, and 9.1 of Suse

'Recommended compiler'
: The default compiler that comes with Susu is fine. Also gcc2.95.x, gcc3.0.x, 3.1 and 3.2 work, but 2.95 is faster. Compile with –threads-enabled (non-standard).

'Extra libs needed'
: None

'Special Instructions'
: None.

6.25 Suse 7.x/PPC

GNUstep has been tested on version 7.0 of Suse/PPC

'Recommended compiler'
: Standard. gcc2.95.x, gcc3.0.x and gc3.1 work, but 2.95 is faster. Compile with –threads-enabled (non-standard).

'Extra libs needed'
: None

'Special Instructions'

6.26 Unixware-2.1.3/Intel

'Recommended compiler'
: Unknown

'Extra libs needed'
: Unknown

Special Instructions for GNUstep installation on Unixware 2.1 systems

1 Tune the kernel to increase the argument space so that we can pass long command-line argument strings to processes (which the makefiles do) (/etc/conf/bin/idtune ARG_MAX 102400)

2 Install raft of the latest GNU software

gzip (you need this to unpack other stuff)
make (to build everything)
m4 (for autoconf etc)
autoconf (if you need to change anything)
bison
flex
binutils (required by gcc if you want to debug)
gcc-2.8.1
(configure –with-gnu-as –with-gnu-ld –with-stabs)
NB. gcc-2.8.1 needs a fix to __do_global_dtors_aux()
in crtstuff.c on Unixware 2.1.3
(and possibly other unixware versions)
The fix is already in recent versions of egcs.

```
==================================
static void
__do_global_dtors_aux ()
{
  static func_ptr *p = __DTOR_LIST__ + 1;
  static int completed = 0;

  if (completed)
    return;

  while (*p)
    {
      p++;
      (*(p-1)) ();
    }

#ifdef EH_FRAME_SECTION_ASM_OP
  __deregister_frame_info (__EH_FRAME_BEGIN__);
#endif
  completed = 1;
}
======================================
```

3 Having got gcc working - it's probably a good idea to rebuild all your GNU software using it!

4 Build gstep as normal.

5 The SIOCGIFCONF ioctl sometimes doesn't work on unixware after applying some of the OS patches.

So I have added a '-a' flag to gdomap to give it the name of a file containing IP address and netmask information for the network interfaces on the system.

You need to set up a file (I suggest '/etc/gdomap_addresses') containing the information for your machine and modify your system startup files in /etc/rc?.d to run gdomap, telling it to use that file.

eg. If your machine has an IP address of '193.111.111.2' and is on a class-C network, your /etc/gdomap_addresses file would contain the line

```
193.111.111.2 255.255.255.0
```

and your startup file would contain the lines

```
. /usr/local/GNUstep/Library/Makefiles/GNUstep.sh
gdomap -a /etc/gdomap_addresses
```

If you don't set gdomap up correctly, Distributed Objects will not work.

6.27 Windows with CYGWIN (*Needs Testing*)

'`Recommended compiler`'
: gcc 3.3.1 or later (with libobjc and libjava (if using libffi))

'`Extra libs needed`'
: Objective-C library DLL (`ftp://ftp.gnustep.org/pub/gnustep/windows/cygwin`) for shared libs. It's a good idea to remove the libobjc.a that comes with gcc (gcc -v for location) so that it isn't accidentally found. For ffcall, you should get version 1.8b or above (the earlier ones don't compile). There are still some problems with structure passing, but that is generally not supported on any architecture. libffi also works.

'`Special Instructions`'
: Make sure you have good shared libraries for everthing. Sometimes a bad shared library (like libtiff) will cause odd and untraceable problems. See `README.Cygwin` for information on compiling.

6.28 Windows with MinGW

'`Recommended compiler`'
: See below.

'`Extra libs needed`'
: See below.

'`Special Instructions`'
: See the `README.MinGW` file located in the gnustep-make Documentation directory for instructions. Windows NT/2000/XP only. Win98 machines and earlier are very buggy and are not supported. Native GUI backend is alpha version.

6.29 Yellowdog/PowerPC

'`Recommended compiler`'
: Standard

'`Extra libs needed`'
: Standard (ffcall or libffi)

'`Special Instructions`'

7 Getting Libraries via Anonymous CVS

If you didn't get one of the snapshots, or if you want to be sure to stay on the bleading edge, then you should get the libraries via CVS. Go to `http://savannah.gnu.org/cvs/?group_id=99` for information on how to get anonymous CVS access.

If you haven't already done so, change to the directory, where you want the source to reside. To get a list of potential modules to check out, type

```
cvs -z3 checkout -c
```

For instance, to check our '`core`', which contains all the GNUstep code libraries:

```
cvs -z3 checkout core
```

After you have checked out the source you can compile it as usual. To update the source, go into the directory of the source tree you want to update, for example, go into 'base', and type:

```
cvs -z3 update -Pd
```

You don't have to re-checkout after you have the source, just update!