GNUstep HOWTO

Installing the GNUstep developement system

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

Last Update: 15 March 2024

Copyright © 1996 - 2007 Free Software Foundation, Inc.
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation.

Table of Contents

1 Introduction 1

2 Summary 2

3 Compiling and Installing the packages 5

3.1 Installing the Core Libraries 5

3.1.1 Alternate Library Setup 6

3.1.2 Building the Package 6

4 Additional Installation 7

4.1 Environment Setup 7

4.2 GNUstep Home 7

4.3 Time Zone 8

4.4 GNUstep deamons 8

5 Test Tools and Applications 9

6 Machine Specific Instructions 10

7 Getting Libraries via git 11

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 not required, but you have to explicitly disable use of XML when compiling GNUstep base if you do not have it.

'`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.

'`gif or ungif (OPTIONAL)`'
The GUI library uses either one of these libraries to load GIF images.

'`aspell (OPTIONAL)`'
The GUI library uses this to handle spell checking.

'`cups (OPTIONAL)`'
The GUI library uses this to handle interface to the CUPS print servers.

'`audiofile (OPTIONAL)`'
The GUI library uses this for playing sound files.

'`portaudio (OPTIONAL)`'
The GUI library uses this for the sound server. Use v19, which has several API changes since the previous version. v19 hasn't actually been formally released, but several distributions (SuSE, etc) use it anyway.

'`freetype2 (RECOMMENDED, REQUIRED for art backend)`'
This is used for font information. Freetype2 cache API is in flux. GNUstep tries to account for this, but if you get errors about undefined FTC_ symbols, you might be using an unsupported version of freetype.

'`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.info`.

‘`gnustep-objc package (REQUIRED BUT ONLY for gcc version < 3.0 or MINGW/Cygwin)`’

Note: Do not install this library unless you are sure you need it. You probably don’t need it except on MinGW and Cygwin (regardless of the gcc version you have). This is a special version of the Objective-C runtime that is compiled as a shared library. 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 capabilities which are used by all GNUstep software. The first package you will compile is gnustep-make. To configure gnustep-make just type:

```
./configure
```

The GNUstep makefile package can be configured to use different types of filesystem layouts. By default, GNUstep is installed with a Unix-style filesystem layout into /usr/local/. That is a good, recommended default if you don't have an opinion on which filesystem layout to use.

But you can also install it somewhere else by using the prefix parameter; the following command installs it in /opt/GNUstep:

```
./configure --prefix=/opt/GNUstep
```

You can also install GNUstep using a GNUstep layout (or some other filesystem layout of your choice) by using the with-layout parameter; the following command configures GNUstep to use the traditional GNUstep layout:

```
./configure --with-layout=gnustep
```

In this document we will always present examples that assume that you are using the default filesystem layout in /usr/local/. If you are using a different layout, you will need to make the obvious changes.

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 Package (gnustep-make). After installing gnustep-make you need to execute GNUstep's shell configuration script, as follows:

```
. /usr/local/share/GNUstep/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

You need to make sure your environment is properly setup in order to compile and run GNUstep software. The steps to setup your environment differ slightly depending on your filesystem layout.

There is a way of setting up your environment that always works: sourcing the `GNUstep.sh` shell script before using GNUstep. The shell script `GNUstep.sh` is located in the Makefile package; you may want to add it to your shell startup file (such as `.profile`). For instance, if you installed GNUstep with the default filesystem layout in `/usr/local`, then adding

```
. /usr/local/share/GNUstep/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 you installed GNUstep somewhere else, you need to replace `/usr/local/share/GNUstep/Makefiles/GNUstep.sh` with the path to your `GNUstep.sh` script. Another typical location is `/usr/GNUstep/System/Library/Makefiles`, which is the default location of your `GNUstep.sh` script when gnustep-make is configured with the GNUstep layout. The script 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/local/share/GNUstep/Makefiles/GNUstep.csh
```

Finally, in most filesystem configuration it's also possible to manually set up your environment by setting PATH, the linker library paths and the `GNUSTEP_MAKEFILES` variable (instead of using `GNUstep.sh`). For example, on GNU/Linux (with a default GNUstep installation), instead of sourcing `GNUstep.sh` you could manually add the Tools directories to your PATH:

```
PATH="/usr/local/bin:$PATH"
```

manually add `/usr/local/lib` to your `/etc/ld.so.conf` file (don't forget to run `ldconfig` every time you install a library), and set the environment variable `GNUSTEP_MAKEFILES` when you want to compile something:

```
GNUSTEP_MAKEFILES=/usr/local/share/GNUstep/Makefiles
```

4.2 GNUstep Home

Your home GNUstep directory should be created automatically the first time you use a GNUstep tool or application. 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). By default this is the directory `GNUstep` under your home directory, but you can change this (see the gnustep-make installation documentation).

4.3 Time Zone

In most cases, GNUstep should be able to determine your time zone, if you have already set it up correctly when setting up your computer. However, in some cases this might fail or the correct information may not be available. You can set it manually using the GNUstep defaults utility to set *`Local Time Zone`* to your local time zone. Type something like *`defaults write NSGlobalDomain "Local Time Zone" GB`*. Where *`GB`* is a time zone abbreviation.

See `/usr/local/lib/GNUstep/Libraries/gnustep-base/Versions/1.21/Resources/` (or equivalent on your system depending on your filesystem layout) for typical time zones.

4.4 GNUstep deamons

Set up your system to execute some GNUstep deamons. This is optional because 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)

```
if [ -f /usr/local/bin/gdomap ]; then
  /usr/local/bin/gdomap
fi
```

- gdnc - Start after sourcing `GNUstep.sh` (e.g. in .profile)
- gpbs - Same as with gdnc, make sure X-Windows is running.
- make_services - Not a deamon, but a tool that needs to be run everytime you install a new Application or service. This is NOT run automatically.

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

5 Test Tools and Applications

Example applications are located in the gstep-examples package. To build 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 `/usr/local/bin`). Usage is:

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

Good Luck!

6 Machine Specific Instructions

A list of machines that GNUstep works on can be found on the GNUstep Wiki `http://wiki.gnustep.org/index.php/Platform_compatibility`.

7 Getting Libraries via git

If you didn't get one of the snapshots, or if you want to be sure to stay on the bleeding edge, then you should get the libraries via git. Go to `http://www.gnustep.org/resources/sources.html` for information on how to get the sourcecode.

To fetch all the GNUstep core libraries, you can clone the covenient `core` repository:

```
git clone --recurse-submodules https://github.com/gnustep/core.git
```

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:

```
git pull
```

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