Supplemental Libraries (Supplibs)

From OpenGrads Wiki

Contents

Quick Instructions

Basically, you check out the sources and build with it a single command:

```
% cd workspace
% cvs ... co -P supplibs  (see below for check out instructions)
% cd supplibs/src
% gmake install
```

If you are very lucky everything will build out of the box (if often does.) After the install completes, enter

```
% make verify
```

for a summary of how things went,

```
-------+---------+--------------
Config | Install |   Package
-------+---------+--------------
  ok   |   ok    | pkg-config
  ok   |   ok    | udunits
  ...
  ok   |   ok    | libsx
  ok   |   --    | freetype
  ok   |   --    | fontconfig
  ok   |   --    | pixman
  ok   |   --    | cairo
  ok   |   ok    | shp
  ok   |   ok    | gd
-------+---------+--------------
```

In this particular example, everything appears to have configured ok (no **--** in the `Config` column.) However, freetype, fontconfig, pixman, and (not surprisingly) cairo did not build.

VERY IMPORTANT: Use `supplibs-2.0.0` for building GrADS v1.9-rc1; use `supplibs-2.1.0` and newer for building GrADS v1.10.r2 and GrADS v2.0.a5 and newer.

Read on if you need more information.

Introduction

As features continue to be added to GrADS with each new release, the number of external packages needed for preparing binary distributions has been steadly increasing. Recent efforts to package GrADS within the major Linux distributions have benefited from the package management inherent in these projects; see for example the GrADS packaged in the Extra Packages for Enterprise Linux (EPEL) (http://download.fedora.redhat.com/pub/epel/5/i386/repoview/grads.html) maintained by Patrice Dumas. However, GrADS is still primarily released in the

form of binary distrbutions. Creating these distributions require careful maintenance of the external packages, usually as a collection of static libraries, the so-called *supplibs*'.

Up to GrADS v1.9beta4, developers have maintained pre-compiled tarballs of the *supplibs* for the supported platforms, along with sources for those libraries which needed customization. This software is still available at COLA's anonymous ftp site (ftp://grads.iges.org/grads/Supplibs). Preparing *supplibs* for a new platform was an ad-hoc time consuming task, and different platforms had different versions of these external packages, potentially leading to non-uniform behavior among these builds.

Starting with GrADS v1.9 an effort was made to collect the source code for the *supplibs* in a central repository (http://opengrads.cvs.sourceforge.net/opengrads/supplibs/), including a mechanism for building all the external packages with a single command: `gmake install` . Having all the packages on a single repository allows for a more rigorous configuration management of the *supplibs*, leading to a uniform set of libraries across the different platforms.

As pre-compiled version of the *supplibs* become available it is our intention to make them available on the web to facilitate the task of building GrADS from sources by developers and end users alike. This document, however, focus on building the *supplibs* from sources.

Contents of the Supplibs

Currently the following packages are maintained in the supplibs:

- **Package management:**
  - Pkg-config
- **Data format related packages:**
  - NetCDF
  - Udunits Version 1
  - HDF Version 4
  - HDF Version 5
  - Grib2c: Grib 2 client library
  - shapelib: shapefile I/O
- **OPeNDAP suite:**
  - libdap: core client/server libraries, in C++
  - gadap: GrADS specific library for accessing station data
- **Basic graphics/images:**
  - jpeg
  - jasper
  - libpng
  - gd

    - tiff
    - geotiff
    - cairo
- **Graphical User Interface**
    - libsx: Simple X widgets, based on the Athena Widgets
- **Compression:**
    - zlib
    - szlib: for reading HDF files only (not used for output)
- **Miscellaneous:**
    - curl (needed by libdap, usually built *without* openssl)
    - xml2 (needed by libdap)
    - sunrpc (only needed for building libdap on cygwin)
- **Packages that have been dropped:**
    - NetCDF Version 4 as a separate package. Earlier on the NetCDF package had Version 3 of the software and an experimental package was created for NetCDF-4. Once NetCDF-4 became stable, the main NetCDF was upgraded to Version 4, and the old `netcdf4` package was abandoned.
    - neXtaw (a replacement for the ""Athena Widgets"); this package did not work well on some Linux distributions such as Ubuntu and was dropped. The stock Athena widgets is more robust and is available on all supported platforms.
    - nc-dap: this functionality is now included in NetCDF-4.

Tested Platforms

Although these are not necessarily supported platforms for GrADS, the *supplibs* are known to build in the following platforms:

- Linux
    - i686
    - x86_64
    - ia64 (Altix)
- Mac OS X (Darwin)
    - Intel
    - PowerPC
- Windows (under cygwin)
- SGI IRIX64
- Solaris
- FreeBSD 6.3 and 7
    - i386
    - amd64
- NetBSD 3.1

    - i386
- IBM
    - AIX 5.3

Getting the sources

Tarballs with stable releases of the *supplibs* sources will be posted on the OpenGrADS SourceForge download page (http://sourceforge.net/project/showfiles.php?group_id=161773). The latest and past releases can also be checked out from the OpenGRADS CVS repository (http://sourceforge.net/cvs/?group_id=161773); the module name is `opengrads`. For checking out the latest release using anonymous CVS access enter:

```
cvs -d:pserver:anonymous@opengrads.cvs.sourceforge.net:/cvsroot/opengrads login
cvs -z3 -d:pserver:anonymous@opengrads.cvs.sourceforge.net:/cvsroot/opengrads co -P supplibs
```

For developer access, consult the OpenGRADS CVS page at sf.net. (http://sourceforge.net/cvs/?group_id=161773)

Which Version do I Need?

Table 1. Matching Supplibs and GrADS versions.

You will need Supplibs version...	For building GrADS version...
2.0.0	1.9-rc1
2.1.0	1.10.r2, v2.0.a5 thru v2.0.a7
2.2.0	2.0.a8 and newer

Building the Supplibs

What you need

- C/C++ compilers; gcc v3 or later highly recommended; the C++ compiler is only needed if you want to build the OPeNDAP suite
- Fortran 77 compiler; g77 highly recommend. While Fortran is not used in GrADS, and Fortran supported is disabled in the supplibs, the `configure` script of some of the packages still requires a Fortran compiler around. Since `g77` is deprecated, `gfortran` may do the trick.
- GNU make, not sure which version as all that I tried worked. A generic POSIX make **will not** work.
- X11 libraries; these used to be built part of the *supplibs* and may be included again if portability becomes an issues

- You will very likely need recent versions GNU bison, flex and GNU m4 (mostly for dap, netcdf and hdf-4)
- It is not a bad idea to have the GNU autotools (autoconf, automake, libtool) installed. Sometimes doing a `autoreconf` for a given package is all you need for `configure` to work.,
- If building on Windows, you will need the Unix emulation layer provided by cygwin (http://cygwin.com). A native port is also being considered. Any volunteers?
- A bit of luck :-)

Doing the build

It couldn't be simpler. Check out or untar the downloaded source tarball and enter

```
% cd supplibs/src
% gmake install
```

and go make a cup of coffee, bake some muffins, and by the time you are back it should be built for you. Or not.

By default, the result of your build is installed in a subdirectory parallel to `src/` using the canonical name for your system; this can be determined with the included GNU `config.guess` script

```
% ./config.guess
```

If your canonical system name is **i686-pc-linux-gnu**, the result of your build will be installed under `../i686-pc-linux-gnu/`. Look around.

To save time, the build does to do a `make check`, although this optional feature should be added at some point. The primary way for you to check if all has been built is to enter

```
% ls *.install
```

and make sure you have a file for each of the packages in the subdirectories. If some of the packages are missing, you may ask yourself whether the absence of these packages will disable GrADS features that you care about. The simplest way to see the implication of what you are missing is by configuring the GrADS build and check what has been disabled; consult Building GrADS from Sources for more information.

If you decide that you need to fix what is broken, you may want to start by checking the *Platform Specific Notes* below for some ways of dealing with common problems. You may also want to customize the build to disable some particular feature in the individual packages; see next section. Ultimately, your best resource is to go through the package developers for support, if available.

Customizing the build

If your build does not work you will probably need to recode some of the video drivers in assembly language and rebuild the kernel. Just kidding. The top GNUmakefile controlling the build of all packages is relatively simple, there are just a few macros that you can modify from the command line. For example,

```
% make install ALLDIRS=netcdf
```

Here are your options:

- **ALLDIRS**. Use this to restrict which packages get built. If you only want NetCDF and udunits just enter

```
% make install ALLDIRS='readline netcdf undunits'
```

- **ROOTDIR** is the root directory for installing the binaries; the installation directory will in the directory `$(ROOTDIR)/$(SYSNAME)`

where `$(SYSNAME)` is the canonical system name. Example:

```
% make install ROOTDIR=/path/to/some/dir
```

- **prefix**. Use this to bypass the definition above and install the *supplibs* in this directory. It is recommended that you

specify ROOTDIR instead.

```
% make install prefix=/path/to/some/dir
```

- **INC_EXTRA** is an additional directory where packages can find header files, primarily used for X11 header. For example, if your `X11/Xpm.h` is in a non standard place, enter

```
% make install INC_EXTRA=/ford1/local/include
```

Partially Building the Supplibs

Sometimes your OS has most of the packages you need and you would like to use the supplibs for small subset of packages. For such cases there is a special module in our CVS repository called `supplibs_fixture`,

```
% cvs -z3 -d:pserver:anonymous@opengrads.cvs.sourceforge.net:/cvsroot/opengrads co -P supplibs_fixture
```

which allows you to checkout only a "shell" of the supplibs, with the top `GNUmakefile`, but with sources only for `pkg-config` (which is a must have for ensuring the self-consistency of the build.) Now, say, you want to build jasper, then you enter:

```
% make co_jasper
% make install
```

This will bring the jasper sources and build it. I wrote in some dependency tracking, so co_hdf4 would also check out zlib, szlib, etc, although I have not tested this dependency feature fully.

Maintaining the Supplibs: Some Guidelines for Developers

If you woud like to do make changes to the supplibs build you will need to understand how makefiles work, in particular GNU make, and have some familiarity with `configure` scripts generated by the GNU autotools (autoconf, automake, etc). The overall philosophy of the *supplibs* is to avoid patching the source code of the individual packages, this way making it easier to import new versions *as is*. If all possible, customizations are to be implemented in the top GNUmakefile, although sometimes this may not be possible. One example is the '*units* package which has been modified do search for the file `udunits/dat` in the GrADS data directory. If a bug fix or new functionality is added to one of the packages one should forward these to the maintainers of the package for inclusion in future releases. The best way to see which changes have been made is to use the `cvs diff` and compare the current release to one of the vendor branches.

The Top GNUmakefile

The top GNUmakefile simply drives the building mechanism of the the underlying packages, making sure it finds its dependencies in the *supplibs* (except for X11 and system libraries). After each package is configured an empty file named

packagename.config is created to indicate that this step has already been done. Similarly, an empty file named *packagename.install* is created after a packaged is successfully built. You may want to delete these files to force rebuild of the package.

The top GNUmakefile defines generic `%.config` and `%.install` rules which are used to build many of the packages. When packages need some special configuration, a specific target is hand coded for that package, e.g., `hdf4.config`. This is particularly important to ensure that each package finds its dependences within the *supplibs*.

The following targets are most useful: `install, clean, distclean`. When debugging specific packages you may want to invoke the config/install steps by hand, for example,

```
% rm netcdf.*
% make netcdf.config
% make netcdf.install
```

Notice that header files are all installed in a separate include directory for each package, e.g., `$(prefix)/include/netcdf`. This is essential to provide alternative implementation of the same functionality. For example, NetCDF-3, HDF-4 and NetCDF=4 all implement the NetCDF v2 API and install the `netcdf.h` header. This decision, however, lead to a non-standard layout of the installed packages, requiring custom version of the config targets to be introduced.

Adding New Packages

It is recommended that you download the original sources, preferably a *stable* version, and try to perform static builds on a number of platforms, and testing GrADS with this new library. Once you you determine that the library is portable, stable and works well with GrADS, go back to the original source and import them on a new vendor branch.

Example. Importing netcdf-3.6.2.tar.gz from `ftp://ftp.unidata.ucar.edu/pub/netcdf/`. Please do not repeat this as NetCDF is already in the repository. The purpose here is to show you how this should be done.

```
% tar xvfz netcdf-3.6.2.tar.gz
% cd netcdf-3.6.2
```

First issue a *dry run* CVS import to see if any conflicts are generated:

```
% cvs -z3 -d:ext:dasilva@opengrads.cvs.sourceforge.net:/cvsroot/opengrads \
      -n import -m 'xxx: importing NetCDF v3.6.2 from Unidata' supplibs/src UNIDATA rel-3_6_2
```

Please notice the vendor branch tag `UNIDATA` and the release tag `rel-3_6_2`. You should not have conflicts the first time around. If cvs reports no conflicts, then repeat the command above without the `-n` option.

```
% cvs -z3 -d:ext:dasilva@opengrads.cvs.sourceforge.net:/cvsroot/opengrads \
      import -m 'xxx: importing NetCDF v3.6.2 from Unidata' supplibs/src UNIDATA rel-3_6_2
```

Please consult the CVS manual (http://ximbiot.com/cvs/manual) or this freely available book (http://cvsbook.red-bean.com/) for additional information on CVS

Updating Packages

Upgrading a package to a new version should be done with caution, and done only when there is a good reason for it, such as important bug fixes or support for new platforms are introduced. The procedure is very similar to the one above to add new packages:

1. first do a dry run import and make sure you understand the conflicts, if any
2. do the real import
3. resolve the conflicts by merging the vendor branch into the trunk.

If you are not comfortable doing these CVS tasks please ask help from a developer with CVS experience.

A final note of caution

If you find something that looks kind of redundant, hold back your impulse to change it. Chances are this is needed on some crazy platform. You never know.

Platform Specific Notes

Here are some of the issues that we have experienced with individual platforms.

Linux

- On OpenSUSE 10.3, some users reported problems with building the udunits library for both `supplibs-2.0.0` and `supplibs-2.0.1`. A workaround is to make sure that calls to `$(MAKE)` in the top-level `GNUmakefile` do **not** contain a `-e` switch. That is, edit the `GNUmakefile` to only contain lines like

```
$(MAKE) install     # OK!
```

  instead of

```
  $(MAKE) -e install  # MIGHT NOT WORK!
```

Update: The `-e` has been removed at the CVS repository head and should appear on sf.net with versions newer than 2.0.1.

FreeBSD 6.3/DesktopBSD 1.6

- Updated `gcc` to 4.2, including `gfortran`
- Set the following environment variables:

```
export XAW7_CFLAGS=/usr/local/include
export XAW7_LIBS=/usr/local/lib/libXaw7.a
```

- Command line is:

```
gmake install CC=gcc42 CXX=g++42 FC=gfortran42 F77=gfortran42
```

At first, the build will likely fail for `szip`, `hdf4` and `udunits`. Here are simple work arounds: after the after configure step, manually modify the following

- Edit szip/Makefile and modify this line:

```
SUBDIRS = src # test examples
```

- Edit udunits/Makefile and modify this line:

```
MAKEWHATIS_CMD=# catman -w -M $(MANDIR)
```

Then finish the build:

```
gmake install CC=gcc42 CXX=g++42 ALLDIRS="szip hd4 udunits"
```

FreeBSD 8.0

Here are the customizations needed for building in FreeBSD 8.0:

- Invoke `make` as

```
   gmake GNUMAKE=gmake install
```

- After configure, in `hdf5/src/Makefile` add this to the `CFLAGS`:

```
    -D__BSD_VISIBLE
```

or `H5FDdirect.c` does not compile because it does not know about `O_DIRECT`.

- For `ncurses` to build you need to edit `tinfo/lib_baudrate.c` and delete the

```
#if defined __FreeBSD__
```

clause that undefines all the `BXX` parameters.

- In the top `GNUmakefile`, add this to the `ncurses.config` target:

```
    --disable-database --enable-termcap
```

IBM AIX

I have built the supplibs on AIX 5.3, PPC/64; in principle these instructions should be generally valid on 32-bit AIX (replacing "64" with "32" below), but I have not tried it. Your mileage may vary.

Make sure that you have recent versions GNU bison, flex and GNU m4 (mostly for dap, netcdf and hdf-4); the versions shipped with AIX 5.3 are old and do not work with recent software source code. Download the sources from ftp.gnu.org and build it on the side first, place these on your path.

On the platform I build this I did not have a fully functional g++ v4.1.1. However, from previous experience, one usually has a better chance of having the packages compiling with gcc rather than xlc. For this reason, I have customized the GNUmakefile to select the following compilers on AIX:

- CC = gcc -maix64
- CXX = xlc++ -q64

I have added "#include <algorithm>" to dap/GeoConstraint.cc for it to compile with xlc++; this has been committed to the repository (current supplibs-2.1.0 sources on sf.net should have this). However, the following hacks were necessary for dap to build:

- Commented out "#define malloc rpl_malloc" in config.h
- Edited Makefile and removed "-Wall -W ..." from CXX flags
- Edited das.y, adding a ";" at the end of line 396:

```
                add_alias(DAS_OBJ(arg), TOP_OF_STACK, *name, string($4));
```

SGI IRIX64

- Make sure you use GNU make (gmake) as the standard make will not cut it.
- The platform that I tried this on did not have Xpm instead along with the X libraries. I used the `INC_EXTRA` to specify this:

```
% gmake install INC_EXTRA=/ford1/local
```

Windows (under cygwin)

- By default cygwin (http://cygwin.com) does not come with the `rpc/xdr.h` headers needed by OPeNDAP. Since the OPeNDAP `configure` script does not flag this, one end up with an error during the compilation. The cygwin package `sunrpc` version 4.0.3 does provide the necessary functionality, but its headers are K&R style and do not compile with the current gcc 3.x, much less with g++. I have updated the `sunrpc` headers and added the whole package to the supplibs. These patches will forwarded to the maintainers. Since `sunrpc` installs to `/usr` I felt it was prudent not to install it by default on cygwin. Instead, you need to

```
% cd sunrpc
% make
% make install
```

- Another annoyance of this platform is that by default file names are not entirely case sensitive. The package libdap installs a header called `GetOpt.h` which cygwin confuses with its own `/usr/include/getopt.h` and nc-dap dies with a compilation error. The work around is to edit your `cygwin.bat` file that starts your bash shell and define the environment variable CYGWIN which tells CYGWIN to treat file names as case sensitive:

```
set CYGWIN=check_case:strict
```

After you are done compiling these modules comment out this option as it is error prone.

- For some reason nc-dap fails when linking `ncdump`; manually link it with g++ instatead of gcc

```
% cd nc-dap/ncdump
```

```
% make ncdump.exe CC=g++
```

This will be reported to the OPeNDAP developers.

Solaris 8 (and up, probably)

The support libs build fine if one uses the GNU version of make and the bash shell. One also has to tell GNU make to use bash as the shell, e.g.:

% gmake SHELL=/bin/bash install

FAQ

Why is the top makefile called *GNUmakefile* and not simply *Makefile*?

`GNUmakefile` is the first file GNU make uses for a default makefile. Any POSIX makefile would not know what to do with this file. Therefore, if mistakenly you enter `make install` where `make` is not GNU make it would report back that there is nothing to build here --- a clue that you need to get a hold of GNU make.

Why is *Makefile.in*, *configure* checked in with the packages sources?

It is common practice not to check these files in the repository, and use `autoreconf` to regenerate them after each check out. However, this requires that a compatible version of the GNU autotools is installed on the build machine. This is not always the case when getting a guest account on some platform for the sole purpose of building GrADS. There is already enough to build, and having to build the autotools is yet another time consuming task that one would like to avoid. For this reason, it is convenient to have a *turn key* build system ready to go, with all the files one would find in a distribution tarball. If adjustments to the `configure` script is necessary, you can perform this on a remote machine, check in the new `configure`, `Makefile.in`, etc files, and do a cvs update on the remote build machine. So, there is no deep reason, just convenience.

Do I need the bin/ directory in the *supplibs* for building GrADS?

The *supplibs* comes with its own copy of `pkg-config` which resides in the bin/ directory. If you have pkg-config already installed, you should be able to use it, but make sure you set the environment variable `PKG_CONFIG_LIBDIR` to find the *supplibs* version of the `*.pc` files. However, if you do not have pkg-config installed,

you must at least keep `bin/pkg-config`.

Why is *gradsdap* (formerly *gradsdods*) built without SSL support?

For portability and export control issues. The OPeNDAP core library `libdap` depends on the cURL package which in turn depends on OpenSSL. Building OpenSSL is technically a possibility, but we can easily get into all sorts of export control issues; read this (http://www.openssl.org/) and that (http://www.openssl.org/support/faq.html#LEGAL). Linking with a shared OpenSSL library could get around the legal issues, but it did not prove portable. It is not clear how many OPeNDAP sites are using the `https://` protocol; probably not too many. Therefore, SSL support is not included for the moment. For enabling it, build the *supplibs* from sources and comment out the `--without-ssl` in the `curl.config` target of the top GNUmakefile.
