



- GnuCOBOL website:
<https://gnucobol.sourceforge.io/>
- GnuCOBOL Manual:
<https://gnucobol.sourceforge.io/doc/gnucobol.html>
- GnuCOBOL Language Quick Reference:
<https://gnucobol.sourceforge.io/HTML/gnucobqr.html>
- Sample Programs:
<https://gnucobol.sourceforge.io/HTML/gnucobsp.html>
- FAQ: <https://gnucobol.sourceforge.io/faq/index.html>

Installation

Source repositories:

REPO=<https://svn.code.sf.net/p/gnucobol/code>

svn checkout \$REPO/branches/gnucobol-3.x

or:

REPO=<https://svn.code.sf.net/p/gnucobol/code>

svn checkout \$REPO/code/trunk gnucobol-trunk

Build and Install:

COBPREFIX=/opt/gnucobol

./build_aux/bootstrap

mkdir _build

cd _build

../configure --enable-cobc-internal-checks --enable-debug \

--prefix \$COBPREFIX --exec-prefix \$COBPREFIX

make

sudo make install

Configuration to use:

COBPREFIX=/opt/gnucobol

PATH=\$COBPREFIX/bin:\$PATH

LD_LIBRARY_PATH=\$COBPREFIX/lib:\$LD_LIBRARY_PATH

export PATH LD_LIBRARY_PATH

Supported Dialects:

acu, acu-strict	ACUCOBOL-GT
bs2000, bs2000-strict	BS2000 COBOL (Siemens)
cobol2002	Cobol ISO-2002
cobol2014	Cobol ISO-2014
cobol85	Cobol ANSI-1985
default	Default config
ibm, ibm-strict	IBM COBOL
mf, mf-strict	Micro Focus COBOL
mvs, mvs-strict	MVS/VM COBOL
realia, realia-strict	CA Realia II COBOL
rm, rm-strict	RM-COBOL
xopen	OpenGroup Cobol 1991 C192

Environment Variables:

COB_PRE_LOAD=m1:m2	Lookup CALLs in m1,m2
COB_CURRENT_DATE=YYYYDDMMHH24MISS	Date returned by ACCEPT
COB_LOAD_CASE=UPPER/LOWER	Lookup filenames
COB_LIBRARY_PATH=[...]	Lookup dynamic modules
COB_FILE_PATH=[...]	Lookup data files
COB_DISABLE_WARNINGS=true/false	Turn off warnings
COB_SET_DEBUG=true/false	Turn on debug lines
COB_SET_TRACE=true/false	Turn on trace
COB_STACKTRACE=true/false	Print stacktrace on abort
COB_DUMP_FILE=filename	Set dump file

Compiler Options

General Options:

cobc --help Display help

cobc --version Display version

Debugging compilation:

cobc -v [...] Increase verbosity
 cobc -q [...] Suppress verbosity
 cobc -### [...] Do not execute sub-commands
 cobc --save-temps [...] Keep intermediary files
 cobc -E [...] Stop after pre-processing
 cobc -fsyntax-only [...] Stop after parsing
 cobc -C [...] Stop after C generation

Choosing Syntax:

cobc --free [...] Use Free Format

cobc --fixed [...] Use Fixed Format

Filenames conversions:

cobc --ffold-copy=[UPPER|LOWER] [...] During COPY
 cobc --ffold-call=[UPPER|LOWER] [...] During CALL
 cobc --ext CBL [...] Extension for COPY

Language configuration:

cobc --std=DIALECT [...] Use the specified dialect
 cobc --conf=FILE [...] Use the specified configuration

Output Configuration:

cobc -x -o FILE [...] Generate executable FILE
 cobc -c [...] Gen. static module (.o)
 cobc -m [...] Gen. dynamic module (.so)
 cobc -b [...] Gen. dyn. library from static modules

Compiler Configuration:

cobc -I DIR [...] Add DIR to include path
 cobc -L DIR [...] Add DIR to linking path
 cobc -lXXX [...] Link with libXXX.so
 cobc -l:XXX.so [...] Link with XXX.so
 cobc -A OPT [...] Pass option OPT to C compiler
 cobc -Q OPT [...] Pass option OPT to linker

Optimizations:

cobc -O2 [...] More optimizations
 cobc -O0 [...] Disable all optimizations
 cobc -fstatic-call [...] Disable dynamic lookup of CALL

Warnings and Errors:

cobc -Wall [...] Display all warnings
 cobc -Wextra [...] Display more warnings than -Wall
 cobc -fmax-errors=N [...] Display N errors max
 cobc -Werror [...] Handle warnings as errors

Display Language Help:

cobc --list-reserved List reserved keywords
 cobc --list-intrinsics List intrinsic functions
 cobc --list-mnemonics List mnemonics
 cobc --list-system List system routines
 cobc --list-registers List available registers

Debugging:

cobc -g [...] Generate debugging information for gdb
 cobc -d [...] Activate all error checks at execution
 cobc -ftrace [...] Generate limited execution trace
 cobc -ftraceall [...] Generate full execution trace

Execution:

cobcrun mod Execute module mod.so
 cobcrun -M f mod Execute entry f of module mod.so

Debugging with gdb and cbl-gdb

Installation:

git clone <https://gitlab.cobolworx.com/COBOLworx/cbl-gdb.git>

cd cbl-gdb

git checkout master

make

sudo make install

Create \$HOME/.gdbinit file with:

enable use of COBOL cbl-dbg extension

add-auto-load-safe-path /usr/local/bin

set directories /usr/local/bin

set auto-load python-scripts on

Build with cobcd:

export COBCD_COBC=/path/to/cobc

cobcd -x prog.cbl -o prog.exe

gdb prog.exe

COBOL-specific commands:

cstart [ARGS]	Start the program with COBOL args
cbreak FILE:LINE	Add a COBOL breakpoint
ctbreak FILE:LINE	Add a one-time breakpoint
cwatch VARIABLE	Breakpoint on VARIABLE value
cnext N	Advance without entering PERFORM
auto-step	Step automatically
until-cobol	Execute until next COBOL module
finish-module	Execute until end of current module
finish-out-of-line-perform	Execute until end of PERFORM
cprint VARIABLE	Print content of COBOL variable
cbacktrace	Print COBOL backtrace
local-backtrace	Print only local backtrace
cup	Select calling function
cdown	Select called function
add-symbol-file-cobol FILE	Add a FILE containing symbols

VSCode Extension

Install from VSIX: <https://cobolworx.com/pages/vsix.html>

Example of tasks.json:

```
{
  "version": "2.0.0",
  "tasks": [
    {
      "label": "make",
      "type": "shell",
      "command": "COBCD_COBC=/path/to/cobc cobcd -Q -WL, -rpath=/path/to/lib -L/path/to/lib -x ${fileBasename}"
    }
  ]
}
```

Example launch.json:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "cobc build and debug",
      "type": "cbl-gdb",
      "request": "launch",
      "preLaunchTask": "make",
      "program": "${workspaceFolder}/${fileBasenameNoExtension}",
      "cwd": "${workspaceFolder}",
      "arguments": ""
    },
    {
      "name": "Attach cbl-gdb to Cobol process",
      "type": "cbl-gdb",
      "cwd": "${workspaceFolder}",
      "solibs": "${env:PRIM_LIBRARY_PATH}",
      "request": "attach",
      "process_id": "${command:getAttachPID}"
    }
  ]
}
```