AUTOTOOLS

Brief introduction

TREX BUILD-SYSTEM HACKATHON | 2021-11-08 I CHRISTIAN FELD

BACKGROUND

- Cross-sectional team Performance Analysis
- HPC-targeted performance tools
- Tools use autotools-based build system (since 2009), goal to run on every relevant HPC machine
- [advertisement] Get performance assessment of your code from POP CoE

Score-P
Scalable performance measurement infrastructure for parallel codes

scalasca

Performance Optimisation and Productivity

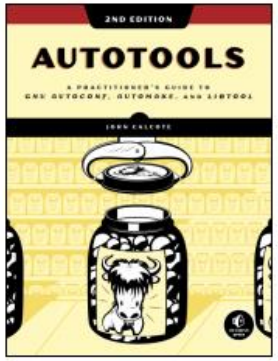
A Centre of Excellence in HPC

https://score-p.org
https://scalasca.org
https://pop-coe.eu/blog/pop-tool-descriptions-jsc-performance-tools

AUTOTOOLS?

- Autoconf, which is used to generate a configuration script for a project
- Automake, which is used to simplify the process of creating consistent and functional makefiles
- Libtool, which provides an abstraction for the portable creation of shared libraries

As stated in the preface to this book, the purpose of the GNU Autotools is to make life simpler for the end user, not the maintainer. Nevertheless, using the Autotools will make your job as a project maintainer easier in the long run, although maybe not for the reasons you suspect. The Autotools framework is as simple as it can be, given the functionality it provides. The real purpose of the Autotools is twofold: it serves the needs of your users, and it makes your project incredibly portable—even to systems on which you've never tested, installed, or built your code.



Autotools, 2nd Edition

A Practitioner's Guide to GNU Autoconf, Automake, and Libtool
by John Calcote

November 2019, 584 pp.
ISBN-13: 9781593279721

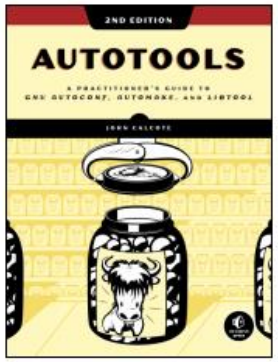
Print Book and FREE Ebook, $49.95
Ebook (PDF, Mobi, and ePub), $39.95

[1] https://nostarch.com/download/samples/Autotools2e_Sample_Ch2.pdf
https://www.gnu.org/software/autoconf/
https://www.gnu.org/software/automake/
https://www.gnu.org/software/libtool/

AUTOTOOLS

autoconf (1992)	automake (1996)	libtool (1998)
2.71 (2021)	1.65 (2021)	2.4.6 (**2015**)
2.70 (2020)		
2.69 (**2012**)		

Documentation:



Autotools, 2nd Edition

A Practitioner's Guide to GNU Autoconf, Automake, and Libtool
by John Calcote

November 2019, 584 pp.
ISBN-13: 9781593279721

- (●) Print Book and FREE Ebook, $49.95
- ( ) Ebook (PDF, Mobi, and ePub), $39.95

Autotools Mythbuster
by Diego Elio Pettenò
https://autotools.io/

https://www.gnu.org/software/autoconf/
https://www.gnu.org/software/automake/
https://www.gnu.org/software/libtool/
The Goat Book (outdated)

OUTLINE

- Build small autotools project, install binary and shared lib
- Add custom check for library and header
- Customize distribution tarball and installation
- Autotools in HPC, pros and cons

DISTINCT ROLES

Project Maintainer

- make life easy for users and packagers (and developers)
- develops configure checks that provide *conditionals* and *substitutions* to be used in source code and generated files
- creates release tarballs

- works in special environment (so do developers), needs (patched) autotools, m4, perl, shell, sed, awk

User/packager

```
wget <url><package>.tar.gz
tar xf <package>.tar.gz
mkdir <package>/_build
cd <package>/_build
../configure --help
../configure <options>
make all [check] install [installcheck]
```

Minimal requirements: compiler, shell, grep, ls, sed, awk
No (patched) autotools required here!

1ST PROJECT

```
.
├── bootstrap*
├── build-config/
│   └── m4/
├── configure.ac
├── doc/
│   └── documentation.md
├── Makefile.am
└── src/
    ├── foo.c
    ├── foo.h
    └── my_1st_prog.c
```

Maintainer needs to provide

- configure.ac
- Makefile.am
- boostrap (convenience only)
- Directories for buildsystem files

← This goes into your source code repository.

1ST PROJECT

configure.ac

```
.
├── bootstrap*
├── build-config/
│   └── m4/
├── configure.ac
├── doc/
│   └── documentation.md
├── Makefile.am
└── src/
    ├── foo.c
    ├── foo.h
    └── my_1st_prog.c
```

```
$ cat configure.ac
AC_INIT([my_project], [1.0])

# auxiliary build tools reside in build-config
AC_CONFIG_AUX_DIR([build-config])
AC_CONFIG_MACRO_DIRS([build-config/m4])

AC_PROG_CC
#AC_PROG_CXX
#AC_PROG_FC

AM_INIT_AUTOMAKE([foreign subdir-objects -Wall])
AM_SILENT_RULES([yes])
AM_PROG_AR
LT_INIT

AC_CONFIG_HEADERS([config.h])
AC_CONFIG_FILES([Makefile])
AC_OUTPUT
```

1ST PROJECT

Makefile.am

```
.
├── bootstrap*
├── build-config/
│   └── m4/
├── configure.ac
├── doc/
│   └── documentation.md
├── Makefile.am
└── src/
    ├── foo.c
    ├── foo.h
    └── my_1st_prog.c
```

```
$ cat Makefile.am
ACLOCAL_AMFLAGS = -I build-config/m4

bin_PROGRAMS = my_1st_prog
my_1st_prog_SOURCES = src/my_1st_prog.c
#my_1st_prog_CPPFLAGS =
#my_1st_prog_LDFLAGS =
my_1st_prog_LDADD = libfoo.la

lib_LTLIBRARIES = libfoo.la
libfoo_la_SOURCES = src/foo.c
#libfoo_la_CPPFLAGS =
#libfoo_la_LDFLAGS =
#libfoo_la_LIBADD =

include_HEADERS = src/foo.h

EXTRA_DIST = doc/documentation.md
```

1ST PROJECT

boostrap (convenience)

```
.
├── bootstrap*
├── build-config/
│   └── m4/
├── configure.ac
├── doc/
│   └── documentation.md
├── Makefile.am
└── src/
    ├── foo.c
    ├── foo.h
    └── my_1st_prog.c
```

```
$ cat bootstrap
#!/bin/sh
## Make ignores the next lines, but shell does not
## -*- mode: makefile -*- \
 argv0=$0; \
 exec make -f "$argv0" "$@"

all:
    autoreconf --install -Wall --no-recursive

clean:
    rm -rf Makefile.in \
        aclocal.m4 \
        config.h.in \
        ...
```

1ST PROJECT

Prepare for 1st configure

```
.
├── bootstrap*
├── build-config/
│   └── m4/
├── configure.ac
├── doc/
│   └── documentation.md
├── Makefile.am
└── src/
    ├── foo.c
    ├── foo.h
    └── my_1st_prog.c
```

```
$ ./bootstrap # aka autoreconf
$ tree
.
├── aclocal.m4
├── autom4te.cache/
│   ├── ...
├── bootstrap*
├── build-config/
│   ├── ar-lib
│   ├── ...
│   ├── m4/
│   │   ├── libtool.m4
│   │   ├── ...
├── config.h.in
├── configure*
├── configure.ac
├── doc/
│   └── documentation.md
├── Makefile.am
├── Makefile.in
└── src/
    ├── foo.c
    ├── foo.h
    └── my_1st_prog.c
```

autoreconf/boostrap to orchestrate calls to autotools programs

```
autotools/bin/
├── aclocal
├── aclocal-1.16
├── autoconf
├── autoheader
├── autom4te
├── automake
├── automake-1.16
├── autoreconf
├── autoscan
├── autoupdate
├── ifnames
├── libtool
└── libtoolize
```

1ST PROJECT

Create distribution tarball

```
.
├── bootstrap*
├── ...
├── config.h.in
├── configure*
├── configure.ac
├── doc/
│   └── documentation.md
├── Makefile.am
├── Makefile.in
└── src/
    ├── foo.c
    ├── foo.h
    └── my_1st_prog.c
```

Ready for first execution of `configure`.
Ready to create self-contained tarball (maintainer).
Ready to do (out-of-place) development builds.
Still using the special maintainer/developer environment.

See `<path>/configure --help` for options.

```
$ <path>/configure --prefix=`pwd`/_install ...
$ tree
.
├── config.h
├── config.log
├── config.status
├── libtool
├── Makefile
├── src
└── stamp-h1
$ make distcheck
...
==========================================
my_project-1.0 archives ready for distribution:
my_project-1.0.tar.gz
==========================================
```

Provide this tarball to your users or packagers. Make targets comply to GNU coding standards

https://www.gnu.org/prep/standards/standards.html#Managing-Releases

CUSTOM CHECK FOR LIBRARY

Macros to communicate test results

- **`AC_SUBST`**: substitutes `@var@` in `.in` files with value of `var` when `configure` is run
- **`AC_DEFINE`**: activate preprocessor defines by transforming `config.h.in` to `config.h` when `configure` is run.
- **`AM_CONDITIONAL`**: allow for conditionals in `Makefile.am` that provide two separate snippets in `Makefile.in` (when `bootstrap` is run) where one snippet gets activated when `configure` is run.

CUSTOM CHECK FOR LIBRARY

Before you write your own, check the **Autoconf Macro Archive** https://www.gnu.org/software/autoconf-archive/

AC_DEFUN to define own macro

```
.
├── bootstrap*
├── build-config/
│   └── m4/
│       └── papi.m4
├── configure.ac
├── doc
│   └── documentation.md
├── Makefile.am
└── src
    ├── foo.c
    ├── foo.h
    ├── my_1st_prog.c
    ├── my_papi.c
    └── my_papi.h
```

```
AC_DEFUN([LIBPAPI], [

AC_ARG_WITH([libpapi],
    [AS_HELP_STRING([--with-libpapi],
        [yes|no|<path> (default: yes)])],
    [with_libpapi=${withval}],
    [with_libpapi=yes])

AS_IF([test "x${with_libpapi}" != xno],
    [save_CPPFLAGS=${CPPFLAGS}
     save_LDFLAGS=${LDFLAGS}
     AS_IF([test "x${with_libpapi}" != xyes],
         [libpapi_dir="${with_libpapi}"
          CPPFLAGS="-I${libpapi_dir}/include $CPPFLAGS"
          LDFLAGS="-L${libpapi_dir}/lib $LDFLAGS"])

     AC_CHECK_LIB([papi], [PAPI_library_init], [papi_lib=yes], [papi_lib=no])
     AC_CHECK_HEADER([papi.h], [papi_header=yes], [papi_header=no])
     CPPFLAGS=${save_CPPFLAGS}
     LDFLAGS=${save_LDFLAGS}])

AM_CONDITIONAL([HAVE_LIBPAPI], [test "x${papi_lib}${papi_header}" = xyesyes])
AM_COND_IF([HAVE_LIBPAPI],
    [AC_DEFINE([HAVE_LIBPAPI], [1], [Define to 1 if PAPI is usable])
     AC_SUBST([LIBPAPI_LIB], [-lpapi])
     AS_IF([test "x${libpapi_dir}" != x],
         [AC_SUBST([LIBPAPI_CPPFLAGS], [-I${libpapi_dir}/include])
          AC_SUBST([LIBPAPI_LDFLAGS], ["-L${libpapi_dir}/lib -R ${libpapi_dir}/lib"])])])
])
```

CUSTOM CHECK FOR LIBRARY

```
.
├── bootstrap*
├── build-config/
│   └── m4/
│       └── papi.m4
├── configure.ac
├── doc
│   └── documentation.md
├── Makefile.am
└── src
    ├── foo.c
    ├── foo.h
    ├── my_1st_prog.c
    ├── my_papi.c
    └── my_papi.h
```

```
$ cat configure.ac
...
AM_PROG_AR
LT_INIT

LIBPAPI

AC_CONFIG_HEADERS([config.h])
...


$ cat Makefile.am
...

if HAVE_LIBPAPI
my_1st_prog_SOURCES += src/my_papi.c src/my_papi.h
my_1st_prog_CPPFLAGS += $(LIBPAPI_CPPFLAGS)
my_1st_prog_LDFLAGS += $(LIBPAPI_LDFLAGS)
my_1st_prog_LDADD += $(LIBPAPI_LIB)
else !HAVE_LIBPAPI
EXTRA_DIST += src/my_papi.c src/my_papi.h
endif !HAVE_LIBPAPI

...
```

CUSTOM CHECK FOR LIBRARY

```
.
├── bootstrap*
├── build-config/
│   └── m4/
│       └── papi.m4
├── configure.ac
├── doc
│   └── documentation.md
├── Makefile.am
└── src
    ├── foo.c
    ├── foo.h
    ├── my_1st_prog.c
    ├── my_papi.c
    └── my_papi.h
```

```
$ cat src/my_1st_prog.c

/* always first include for compilation units, provides cpp defines */
#include <config.h>

...

#ifdef HAVE_LIBPAPI
#include "my_papi.h"
#endif /* HAVE_LIBPAPI */

int main(int argc, char **argv)
{
    ...

#ifdef HAVE_LIBPAPI
    my_papi_init();
#endif /* HAVE_LIBPAPI */
    ...
}
```

CUSTOMIZE TARBALL AND INSTALLATION

AC_SUBST of predefined variables

```
.
├── bootstrap*
├── build-config/
│   └── m4/
│       └── papi.m4
├── configure.ac
├── doc
│   └── documentation.md
│   └── doxyfile.in
├── Makefile.am
└── src
    ├── foo.c
    ├── foo.h
    ├── my_1st_prog.c
    ├── my_papi.c
    └── my_papi.h
```

```
$ cat doc/doxfile.in
# This file describes the settings to be used by the documentation system
# doxygen (www.doxygen.org) for a project.
# ...

PROJECT_NAME           = "@PACKAGE@"
PROJECT_NUMBER         = @PACKAGE_VERSION@
INPUT                  = @abs_top_srcdir@
OUTPUT_DIRECTORY       = doxygen
QUIET                  = YES

# ...
```

CUSTOMIZE TARBALL AND INSTALLATION

Create doxyfile from doxyfile.in

```
.
├── bootstrap*
├── build-config/
│   └── m4/
│       └── papi.m4
├── configure.ac
├── doc
│   └── documentation.md
│   └── doxyfile.in
├── Makefile.am
└── src
    ├── foo.c
    ├── foo.h
    ├── my_1st_prog.c
    ├── my_papi.c
    └── my_papi.h
```

```
$ cat configure.ac

...
LIBPAPI

AC_CHECK_PROG([DOXYGEN], [doxygen], [`which doxygen`])
AM_CONDITIONAL([HAVE_DOXYGEN], [test -n "$DOXYGEN"])
AM_COND_IF([HAVE_DOXYGEN], [AC_CONFIG_FILES([doc/doxyfile])])

AC_CONFIG_HEADERS([config.h])
...
```

CUSTOMIZE TARBALL AND INSTALLATION

Use Makefile hooks (and -local targets)

```
.
├── bootstrap*
├── build-config/
│   └── m4/
│       └── papi.m4
├── configure.ac
├── doc
│   └── documentation.md
│   └── doxyfile.in
├── Makefile.am
└── src
    ├── foo.c
    ├── foo.h
    ├── my_1st_prog.c
    ├── my_papi.c
    └── my_papi.h
```

```
$ cat Makefile.am
...
dist-hook: doxygen-2-dist
if HAVE_DOXYGEN
doxygen-2-dist:
	$(DOXYGEN) doc/doxyfile
	tar czf doxygen.tar.gz doxygen/
	cp doxygen.tar.gz $(distdir)/doc
	rm -rf doxygen doxygen.tar.gz
else !HAVE_DOXYGEN
doxygen-2-dist:
endif !HAVE_DOXYGEN

install-data-hook: doxygen-install
doxygen-install:
	if test -e $(srcdir)/doc/doxygen.tar.gz; then \
		$(MKDIR_P) $(DESTDIR)$(docdir); \
		$(INSTALL_DATA) $(srcdir)/doc/doxygen.tar.gz $(DESTDIR)$(docdir)/; \
		cd $(DESTDIR)$(docdir); \
		tar xzf doxygen.tar.gz; \
		rm -f doxygen.tar.gz; \
	    fi

uninstall-hook: doxygen-uninstall
doxygen-uninstall:
	rm -rf $(DESTDIR)$(docdir)/doxygen
```

AUTOTOOLS IN HPC

Pros:

- Extremely flexible, in the end just (portable) shell code and Makefile(s)
- Customizable via own macros, Makefile hooks and -local targets, everything is possible
- make distcheck extremely helpful to get contents of (self-contained) tarball right
- configure --help to show possible customization options
- Parallel make and out-of-place builds
- Minimal requirements on user’s system

AUTOTOOLS IN HPC

Cons:

- Steep learning curve; online beginner guides, hm; time consuming; expect frustration
- Difficult to debug; compare generated files; expect m4-quoting surprises; time consuming
- Not continuously maintained; libtool/autoconf not aware of all HPC-relevant compilers; some vendors provide patches (that you have to apply to your project-specific autotools), others don’t.
- AC_CHECK_LIB doesn’t use libtool during configure, but make does, huh
- libtool generates and honors .la files; sometimes broken; prevents mixing compiler vendors; more harm than benefit
- libtool decides on compiler names; special handling needed for compiler wrappers (mpicc) otherwise no PIC; patching libtool is no fun (lots of code for ancient systems in the way)
- libtool issues on Cray systems (workaround available if you know a specific Cray employee)
- Usually everything OK for GNU compilers; doesn’t fulfill HPC requirements
- No built-in dependency management for **Fortan**, though OK for C/C++