Introduction to Autotools

Mikael Djurfeldt
PDC

Prerequisites

- On your own linux computer, do:

  ```
  sudo apt-get install git autoconf automake libxml2-dev
  ```

  or something corresponding (if using a different package manager).

- On Dardel, do:

  ```
  module add libxml2
  ```

- To download this lesson, do:

  ```
  git clone https://github.com/PDC-support/introduction-to-autotools.git
  ```

Documentation

- `autoconf`: `https://www.gnu.org/software/autoconf/`
- `automake`: `https://www.gnu.org/software/automake/`

Overview

- autoconf
- automake
- Debugging packages configured using Autotools

What are autotools?

- GNU Autotools is a suite of tools used to automate the *configuration*, *building*, and *installation* of software packages across different systems.
- Typical installation of a package using GNU Autotools

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

Example 1: Using autoconf to adapt to the system

main.c:

```
#include <netdb.h>
#include <stdio.h>

/* Here and as follows code without error checking for brevity. */

int main() {
    struct hostent *host;
    host = gethostbyname("www.pdc.kth.se");
    ip = host->h_addr_list[0];
    printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]);
    return 0;
}
```

- `gethostbyname()` is provided by libnsl on Solaris

Example 1

configure.ac:

```
AC_INIT([example1], [1.0])

AC_SEARCH_LIBS([gethostbyname], [nsl])

AC_CONFIG_FILES([Makefile])
AC_OUTPUT
```

Makefile.in:

```
example1:
        gcc -o example1 main.c @LIBS@
```

`./configure` will now make a `Makefile` from `Makefile.in` replacing @LIBS@ with the proper value (-lnsl on those systems which provide gethostbyname through libnsl)

What is autoconf?

- Part of the GNU build system
- Tool for producing shell scripts that automatically configure software source code packages to adapt to many kinds of Posix-like systems
    - Running `autoconf` will take `configure.ac` as input and make `configure`
- The autoconf tool is not required when *building* the package:
    - `./configure` will run tests, take `Makefile.in` as input and make `Makefile`

Inner workings of autoconf

- `configure.ac` as well as the tests are written in the macro language M4
- The `configure` script is a Bourne shell script

Some useful existing tests

Test	Description
`AC_SEARCH_LIBS(FUNCTION, SEARCH_LIBS)`	Search for `FUNCTION` among `SEARCH_LIBS`
`AC_CHECK_LIB(LIBRARY, FUNCTION)`	Search for `LIBRARY` providing `FUNCTION`
`AC_CHECK_FUNCS(FUNCTION...)`	Define `HAVE_FUNCTION` if found
`AC_CHECK_HEADER(HEADER-FILE)`	Define `HAVE_HEADER-FILE` if found

Example 2: Further adaptation

- `gethostbyname()` is deprecated
- Use autoconf to test existence of alternative to gethostname:

  `AC_CHECK_FUNCS([getaddrinfo])`
- Need way to enable conditional compilation
  - `config.h` contains C preprocessor macro definitions which are the results of tests such as `AC_CHECK_FUNCS`
  - `./configure` takes template `config.h.in` and makes `config.h`
  - The *developer* needs to run `autoheader` to generate `config.h.in`

Example 2

configure.ac:

```
AC_INIT([example2], [1.0])
AC_CONFIG_HEADERS([config.h])

AC_SEARCH_LIBS([gethostbyname], [nsl])
AC_CHECK_FUNCS([getaddrinfo])

AC_CONFIG_FILES([Makefile])
AC_OUTPUT
```

- If found `AC_CHECK_FUNCS` defines `HAVE_GETADDRINFO` in `config.h`
- `AC_CONFIG_HEADERS` tells `configure` that `config.h` should be generated

Example 2

```
#include "config.h"

#include <netdb.h>
#include <stdio.h>

#ifdef HAVE_GETADDRINFO
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#endif /* HAVE_GETADDRINFO */

#define HOSTNAME "www.pdc.kth.se"

int main() {
    unsigned char *ip;
#ifdef HAVE_GETADDRINFO
    struct addrinfo hints, *res;
    memset(&hints, 0, sizeof hints);
    hints.ai_family = AF_INET;        // AF_INET for IPv4
    hints.ai_socktype = SOCK_STREAM;  // TCP stream sockets
    getaddrinfo(HOSTNAME, NULL, &hints, &res);
    ip = (unsigned char *) &((struct sockaddr_in *) res->ai_addr)->sin_addr;
#else /* HAVE_GETADDRINFO */
    struct hostent *host;
    host = gethostbyname(HOSTNAME);
    ip = host->h_addr_list[0];
#endif /* HAVE_GETADDRINFO */
    printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]);
    return 0;
}
```

Example 2

- Use make variables CC and CFLAGS in Makefile.in for more flexibility:

```
CC = @CC@
CFLAGS = @CFLAGS@

example2:
	$(CC) $(CFLAGS) -o pdcip main.c @LIBS@
```

- By default, `configure` will choose values for `CC` and `CFLAGS`
- You can also supply them to `configure`:

  `./configure CFLAGS="-g -O1"`

Exercise on computer

- Go to the directory example2: `cd example2`
- Run `autoconf` to make `configure` from `configure.in`
- Run `autoheader` to make `config.h.in` from `configure.ac`
- Run `./configure` to make `config.h` and `Makefile`
- Run `make`
- Now try out a different optimization level:

```
./configure CFLAGS="-g -O1"
make
```

What is automake?

- Writing `Makefile.in` for complex projects can get tedius and repetitive
- Automake simplifies and automates the making of Makefiles
- `automake` takes `Makefile.am` and makes `Makefile.in` for `./configure`
- A `Makefile.am` is essentially a set of variable definitions.

Automake introduces new M4 macros

- Just using `autoconf`: M4 macros "under the hood" (/usr/share/autoconf)
- With `automake`: M4 macros in a local file `aclocal.m4`
- Needed by `autoconf` and `automake`
- `aclocal` makes `aclocal.m4` from configure.ac

Example 3: Automake

Makefile.am:

```
bin_PROGRAMS = pdcip
pdcip_SOURCES = main.c
```

configure.ac:

```
AC_INIT([example1], [1.0])
AM_INIT_AUTOMAKE()
...
```

Invocation:

```
# Done by developer:
$ aclocal                # configure.ac -> aclocal.m4
$ autoconf               # configure.ac -> configure
$ automake --add-missing # Makefile.am  -> Makefile.in
# Done by "user":
$ ./configure            # Makefile.in  -> Makefile
```

Shorthand: autoreconf

- `aclocal`, `autoheader`, `autoconf` and `automake` (and more!) can all be done by `autoreconf`
- Recommended invocation is `autoreconf -ivf` (`autoreconf --install --verbose --force`)
- `-i` install missing files
- `-f` consider all generated and standard files obsolete
- `-v` be verbose and tell what you are doing

We automatically get compilation targets

Target	Description
make all	Build programs, libraries, documentation, etc. (= "make")
make install	Install what needs to be installed, copying the files from
	the package’s tree to system-wide directories.
make clean	Erase from the build tree the files built by ‘make all’.
make distclean	Additionally erase anything ‘./configure’ created.
make check	Run the test suite, if any.
make dist	Recreate ‘PACKAGE-VERSION.tar.gz’ from all the source files.

Invoking configure

- `./configure --help` shows how to invoke `configure`
- We have seen that we can define make variables on the `configure` line:

  `./configure CFLAGS="-g"` is useful when *debugging*
  (generate debug symbols and perform no optimization)
- In addition, we can give `configure` options
- The option `--prefix` controls where to install:
  - Default is `/usr/local` (`/usr/local/bin` etc)
  - To install in home directory, do:

```
./configure --prefix=$HOME
make
make install
```

Example 4: --with-XXX

- We may want to be able to optionally add functionality to a package
- In this example we add a new option to `configure`:

  ```
  configure --with-libxml2
  ```

  which lets `pdcip` use the library `libxml2` to output the ip address in XML

Example 4

Excerpt from configure.ac:

```
AC_ARG_WITH([libxml2],
    [AS_HELP_STRING([--with-libxml2], [Use libxml2 to generate output])],
    [with_libxml2=$withval], [with_libxml2=no])
```

- `AC_ARG_WITH([libxml2],...)` adds the option `--with-libxml2`
- `[...]` are quoting delimiters in M4; they protect against macro expansion
- 3rd argument of `AC_ARG_WITH` is what happens if option is given
- 4th argument of `AC_ARG_WITH` is what happens if option is not given

Example 4

Excerpt from configure.ac:

```
if test "$with_libxml2" != "no"; then
    PKG_CHECK_MODULES([LIBXML2], [libxml-2.0])
    AC_DEFINE([WITH_LIBXML2], [1], [Define if building with libxml2])
    CFLAGS="$CFLAGS $LIBXML2_CFLAGS"
    LIBS="$LIBS $LIBXML2_LIBS"
else
    AC_MSG_WARN([libxml2 not used])
fi
```

- `PKG_CHECK_MODULES([LIBXML2], ...)` sets `LIBXML2_CFLAGS` and `LIBXML2_LIBS`
- `AC_DEFINE` defines the C preprocessor macro `WITH_LIBXML2` in `config.h`

Example 4

- In addition, we have added a line `AC_PROG_CC` towards the top of `configure.ac`. This makes sure that tests for the compiler is done unconditionally and not automatically inside if/then/else:

```
AC_INIT([example4], [1.0])
AM_INIT_AUTOMAKE()
AC_CONFIG_HEADERS([config.h])

AC_PROG_CC
```

Example 4

Excerpt from main.c:

```
int main() {
    struct hostent *host;
    unsigned char *ip;
    host = gethostbyname("www.pdc.kth.se");
    ip = host->h_addr_list[0];
#ifdef WITH_LIBXML2
    {
      char buf[16];
      sprintf(buf, "%d.%d.%d.%d", ip[0], ip[1], ip[2], ip[3]);
      gen_xml(buf); /* uses libxml2 to print ip address in xml */
    }
#else
    printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]);
#endif
    return 0;
}
```

Debugging autotools

- Exercise 3 in the hands-on will give an opportunity to debug an autoconf problem
- Typical problems can be:
  - The project was made for outdated versions of Autotools
    - Action: Update `configure.ac` to state of the art
  - Conditional compilation in your code doesn't go as expected
    - Action: Check that macros in config.h are defined as expected
  - An error occurs when running the `configure` script
    - Action: Search for the location of the error in `config.log`

Hands-on

Exercise 1

- It is possible, even recommended, to use separate source and build directories
- First make sure that `configure` exists in the directory `example3` by doing `autoreconf -ivf` there
- Now enter the top-level directory `introduction-to-autotools` and make a new subdirectory `build` (mkdir `build`)
- Enter `build` (cd `build`)
- Invoke `example3/configure`: `../example3/configure`
- `make`

Exercise 2

- Extend the `example3` project to also build `hello` which prints `Hello world!`

Exercise 3: Debugging an autotools project

- Enter the subdirectory `exercise3`
- Here, `autoconf.ac` checks that `xmlBufferCreate()` is actually available
- Do `autoreconf -ivf` and try to:
  `./configure --with-libxml2`
- Carefully read the last lines of output from `configure`
- Can you spot the problem?
- Hint: Search for xmlBufferCreate in config.log and how the test is compiled (`gcc -o conftest ...`). What is missing? Why?