Integration of GNU Autotools with Python Development Environments

Franklin E. Diaz

June 03, 2022

Abstract

The goal of this document is to detail my use of GNU Autotools with Python development. My effort to reduce the complexity of delivering Python heavy projects to customers and end users has led to useful ways of working that I share in this paper.

Click here to download latest version.

1 Integration of GNU Autotools with Python Development Environments

My effort to reduce the complexity and ease configuration issues when delivering Python heavy projects to customers and end users has led to useful ways of working that I share in this paper. Autotools are a well-maintained set of Open Source tools with a gentle learning curve and are included in the distribution of many Open Source packages that we all rely on daily. Here I have assembled and battled-tested certain methods that I would like to encourage others to understand and adopt.

The full set of steps for using this autotools configuration is as follows:

Tool	Description
autoconf	Generates a configure script from configure.ac
automake	Generates a system-specific Makefile based on Makefile.am template
make	X
libtool	X

Table 1: Tools used in this project

The image shown in Figure 1 is from Gentoo Linux DevManual: The Basics of Autotools. The image illustrates the relationship between the components mentioned in Table 1.

2 The bootstrap.sh script

The purpose of the "bootstrap.sh" script is to allow project maintainers to prepare the local environment for use of autotools. The boostrap script will attempt to guess if the local host is running a certain Linux distribution, or MacOS. There are execution paths for each of these respective scenarios.

At first execution, autotools will run libtool, automake, and configure. Autotools creates a file named "config.log". If the config.log file is found on subsequent runs of the script, fewer configuration steps will be performed since a certain system state is assumed. To get a "clean start" the maintainer can simply delete the config.log file and rerun the bootstrap.sh script.

A full working example of the bootstrap.sh script can be found in the source repository for this paper.

2.1: Steps to use Autotools

```
./bootstrap.sh
./configure
make python
. _build/bin/activate
```

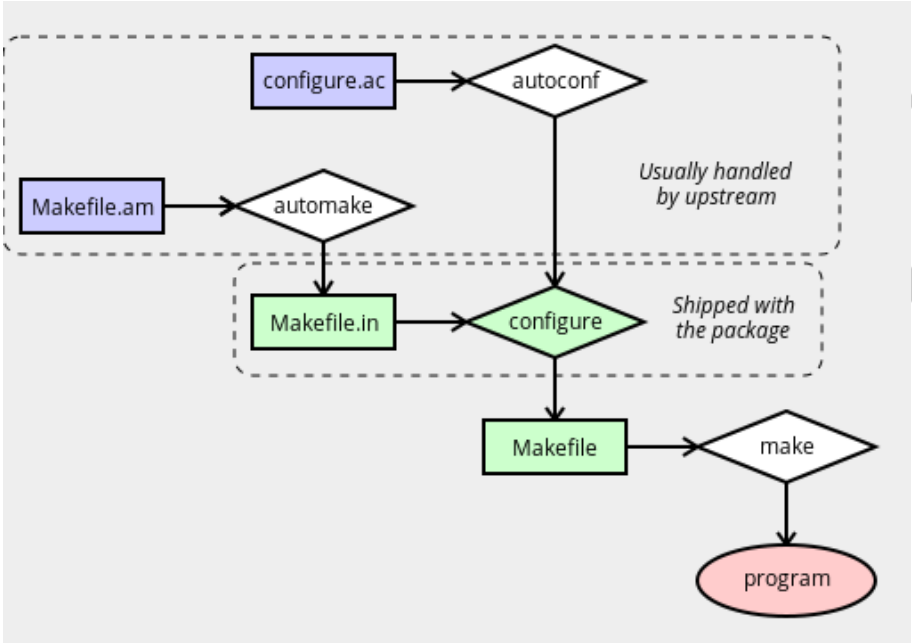

Figure 1: A basic overview of how the main autotools components fit together.

3 The configure.ac file

A "./configure" script can be generated from a template named "configure.ac". This template is comprised of macros that are used to identify local system software and program locations. In our case, we are interested in identifying the location of the Python 3.x installation on the local system.

Per the automake documentation we can define the minimum acceptable version of Python and a variable that automake can use to refer to the version of Python discovered on the system. These declarative statements and the resultant values can then be referenced by automake and added explicitly to the "Makefile.am" template.

3.1: Declaring Python Autoconf macros in configure.ac

```
dnl the Python configuration
AM_PATH_PYTHON(3.9) # minimum version of Python
PY39="python$PYTHON_VERSION" # define the python interpreter
dnl LDFLAGS="$LDFLAGS -l$PY39"
AC_SUBST(PY39, python$PYTHON_VERSION)
```

A full working example of the configure.ac file can be found in the source repository for this paper.

4 The Makefile.am file

The "Makefile.am" file is a template that autotools uses to generate the full Makefile.

5 Python requirements.txt files

My projects typically include three main requirements files, separated by functionality and named to foster a self-documenting paradigm.

Filename	Description
src/requirements.txt	Common Python modules required by the main application.
tests/requirements-test.txt	test, linting, syntax checking, formatting, etc.
tests/requirements-security.txt	scan and secure the main app

Table 2: Python requirements files

Revision History

Revision	Date	Author(s)	Description
v0.1	June 03, 2022	Franklin Diaz	Initial Draft

References

[1] John Calcote. *Autotools: A Practioner's Guide to GNU Autoconf, Automake, and Libtool*. No Starch Press, USA, 1st edition, 2010.