Bisoños Usuarios de GNU/Linux de Mallorca y Alrededores | Bergantells Usuaris de GNU/Linux de Mallorca i Afegitons

Autotools: primers passos **(4608 lectures)**

Per **Joan Lledó**, ***illedo*** (http://gnetic.nongnu.org)

Creado el 21/05/2010 23:58 modificado el 23/05/2010 13:20

Aquest no és un manual que aprofundisca en les possibilitats de les autotools, ja que per això caldrien desenes de manuals i no acabaríem mai. El que es pretén amb aquesta petita guia és perdre-li la por a les autotools, ja que qualsevol persona que vulga programar en linux se les trobarà sí o sí, i si no s'hi tenen unes nocions bàsiques, un es pot fer un trencaclosques amb tots els scripts i fitxers de configuració, i no veure-li la lògica. L'objectiu d'aquesta guia és resoldre dubtes bàsics, tindre una visió amb perspectiva de la funció de les autotools en un programari, per així començar a profunditzar en el tema.

GNU Autotools: Primers passos

Introducció:

Què són les autotools:

Les autotools són unes eines l'objectiu de les quals és generar un paquet distribuïble per a tots els sistemes de tipus Unix, com per exemple linux o solaris.

A partir d'un codi font concret, les autotools creen una sèrie de scripts per a facilitar la instal·lació i la compilació del programari en diversos sistemes Unix, així com el paquet de distribució, el famós fitxer tar.gz que sempre trobarem quan vulguem descarregar el codi font d'un programa.

Si tot es fa correctament, un fitxer tar.gz generat amb autotools ha de poder-se instal·lar mitjançat les cèlebres comandes:

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

I açò ha de servir igualment per a Linux, Solaris, BSD, etc.

Compte: amb les autotools no obtindrem un paquet .deb o un .rpm, això s'obté amb altres utilitats específiques de cada distribució linux.

Què s'ensenya en aquest manual:

Aquest no és un manual que profunditze en les possibilitats de les autotools, ja que per això caldrien desenes de manuals i no acabaríem mai. El que es pretén amb aquesta petita guia és perdre-li la por a les autotools, ja que qualsevol persona que vulga programar en linux se les trobarà sí o sí, i si no s'hi tenen unes nocions bàsiques, un es pot fer un trencaclosques amb tots els scripts i fitxers de configuració, i no veure-li la lògica. L'objectiu d'aquesta guia és resoldre dubtes bàsics, tindre una visió amb perspectiva de la funció de les autotools en un programari, per així començar a profunditzar en el tema.

Concretament, quan vostè acabe de llegir aquest manual sabrà:

◊ Fer que el codi font d'un programa estiga gestionat amb autotools
◊ Crear una distribució per al programari en format tar.gz
◊ Internacionalitzar el paquet amb gettext i integrar-ho amb el sistema autotools

---

PART 1, Crear un paquet de distribució:

Començant:

Què és el primer que necessitem?

Òbviament, ens caldrà un codi font, i per aquest manual no caldrà que ens trenquem massa el cap, llavors, començarem pel més típic, un Hola, món! escrit en C.

holamon.c

```
#include <stdio.h>

int main()
{
        printf( Hello, world\n );

        return 0;
}
```

Observacions:

◊ L'exemple està escrit en C, però es poden fer servir altres llenguatges.
◊ Per norma general les fonts s'escriuen en anglès i després es creen les traduccions a altres llengües, no obstant això, no és obligatori, podem escriure el programa en la llengua que ens plaga.

Al nostre projecte l'he batejat amb el nom de holamon . Per tant, crearem una carpeta anomenada holamon i en seu interior un directori amb el nom src , on ficarem el codi font.

Ha de quedar així:

holamon

|_src

|_holamon.c

L'estructura de directoris del nostre projecte l'hem de mantindre de la manera més senzilla i clara possible. Quan li integrem les autotools, es crearan moltes noves carpetes i fitxers, llavors per claredat es convenient que el nostre codi tinga un directori propi.

Tot projecte d'autotools ha de tindre 3 fitxers bàsics, que són pels quals es comença, aquests són:

```
configure.ac
Makefile.am
src/Makefile.am
```

Observacions:

◊ En les versions antigues d'autotools, el fitxer configure.ac es deia configure.in.

Com hem vist abans, les autotools son un conjunt d'eines, i cadascuna d'elles llig i processa un fitxer o fitxers distints, els quals contenen macros específiques per a cada eina. Cada macro és un conjunt d'ordres, una cosa semblant a una funció. Llavors, editar els fitxers configure.ac o Makefile.am és semblant a fer servir un llenguatge de programació d'alt nivell.

Per a integrar-li les autotools al nostre programa només haurem de crear i editar aquestos 3 fitxers, tots els altres es generaran automàticament, nosaltres només ens hem de preocupar pel codi font i per aquestos 3 fitxers.

Però, per a què serveixen?

configure.ac:

Serà llegit per autoconf per a generar el ./configure. També serà llegit per l'automake per a generar els Makefile.in.

Makefile.am:

Serà llegit per l'automake per a generar el Makefile.in.

Al tall:

Terminal
`$ touch configure.ac Makefile.am src/Makefile.am`

L'arbre de directoris ha de quedar així:

holamon

|_src

|_holamon.c

|_Makefile.am

|_configure.ac

|_Makefile.am

Editem els fitxers:

configure.ac

```
AC_INIT([holamon],[0.1])
AM_INIT_AUTOMAKE([-Wall])
AC_PROG_CC
AC_CONFIG_HEADERS([config.h])
AC_CONFIG_FILES([Makefile src/Makefile])
AC_OUTPUT
```

Makefile.am

```
SUBDIRS=src
```

src/Makefile.am

bin_PROGRAMS=holamon

holamon_SOURCES= holamon.c

Línia a línia:

configure.ac:

- AC_INIT: Inicialitza l'autoconf. És una macro obligatòria i generalment ha de ser la primera del configure.ac. Té dos paràmetres obligatoris, el nom que del paquet, i la seua versió.

  + Informació[(1)]
- AM_INIT_AUTOMAKE: Inicialitza l'automake. Pot rebre com a paràmetres algunes opcions per a controlar l'execució de l'automake.

  + Informació[(2)]
- AC_PROG_CC: Estableix quin compilador es farà servir, en aquest cas, serà el gcc

+ Informació[3]

- AC_CONFIG_HEADERS: Aquesta macro és molt interessant, i és important conèixer la seua utilitat. Hem de recordar que les autotools serveixen per a poder portar un programari a diferents sistemes operatius o diferents màquines, llavors, a l'hora de compilar un programa, hi ha certes variables del seu codi que han de tindre valors distints segons el sistema on es vaja a instal·lar. Com es fa això? La solució que aporten les autotools a aquest problema és crear un fitxer, generalment anomenat config.h, que s'ha d'incloure amb un #include en el codi, on aquestes variables específiques s'omplin automàticament durant l'execució del ./configure. Aquesta macro rep com a paràmetre el nom de la capçalera de configuració que ha de generar.

  + Informació[4]
- AC_CONFIG_FILES: Amb aquesta macro indiquem quins són els fitxers que s'han de generar automàticament

  + Informació[5]
- AC_OUTPUT: És el conjunt d'instruccions que fa efectiva la creació dels fitxers especificats en AC_CONFIG_FILES. És una macro obligatòria i ha d'estar situada l'ultima del configure.ac.

  Compte: l'ús de paràmetres en aquesta macro és obsolet, s'ha de fer servir AC_CONFIG_FILES en el seu lloc.

  + Informació[6]

Makefile.am:

- SUBDIRS: Aquí se li ha d'afegir la llista de directoris que es vol construir, és a dir, carpetes on hi ha Makefiles i que es volen incloure al paquet.

  + Informació[7]

src/Makefile.am:

- bin_PROGRAMS: Amb aquesta macro s'indica quins seran els executables que s'hauran de compilar i d'instal·lar.

  +Informació[8]
- holamon_SOURCES: Per cada binari declarat a la macro anterior, s'ha d'especificar quins són els fitxers font que el componen.

  + Informació[9]

I ja està. Realment no és necessari fer cap més canvi .Ara hem d'executar un sèrie de comandes per a generar tot l'arbre fitxers i directoris típic dels tar.gz i generar el paquet de distribució en sí.

Com hem vist abans, les autotools són un conjunt d'eines, no n'és una sola, són vàries comandes que s'han d'executar sobre la carpeta on hem desat el configure.ac, i que ho han de fer en l'ordre correcte, aquestes eines són l'autoconf, l'autoheader, l'aclocal, l'automake, etc.

Per facilitar la tasca del programador, hi existeix una comanda que s'encarrega explícitament d'executar les comandes pertinents i en l'ordre adequat, aquesta eina és l'autoreconf. Nosaltres no haurem d'executar l'auotoconf, l'automake, etc, només l'autoreconf.

Però abans de fer-ho, cal tindre en compte que un paquet de distribució tar.gz, segons les convencions de distribució del programari per als sistemes UNIX, ha de contindre uns fitxers que informen d'alguns aspectes del programa tals com la llicència, els autors o la llista de canvis, aquestos fitxers són obligatoris i l'autoreconf fallarà si no els creem, llavors executem abans que res:

Terminal

```
$ touch NEWS README AUTHORS ChangeLog
```

No és obligatori emplenar-los, però és recomanable emplenar com a mínim l'AUTHORS amb el nostre nom i el README amb un petita descripció del programa. No hi ha cap norma o sintaxi sobre com s'han d'emplenar, és lliure.

Hi ha un cinquè fitxer, el COPYING que també és obligatori i que conté la llicència del programa. Si no el creem abans, l'autoreconf n'instal·larà un amb la llicència GPLv3

Ara ja podem executar

Terminal

```
$ autoreconf –i
```

El paràmetre -i o també install copia, en cas de que no existisquen, una sèrie de fitxers que necessita l'autoreconf per fer el seu treball, l'hem de fer servir sempre.

Quan l'autoreconf acabe, veurem que la carpeta del nostre projecte està prou poblada. En aquest punt ja podem executar les clàssiques comandes

Terminal

```
$ ./configure
$ make
```

I amb açò ja tindrem el nostre programa compilat i el podrem executar:

Terminal

```
$ src/holamon
```

Aquesta comanda mostrarà per pantalla el Hello, world! que amb tant de talent i tanta mestressa hem programat.

Com crear el paquet de distribució tar.gz?

Fàcil, executem:

Terminal
`$ make distcheck`

I ja tindrem el nostre paquet preparat per a publicar.

PART 2, Internacionalitzar el nostre programa:

Començant:

El primer que cal saber és que volen dir les paraules Internacionalitzar i Localitzar :

- ◊ Internacionalitzar (abreviat, I18N): És el treball de transformació d'un programa que no suporta idiomes en un que sí els suporte, és a dir, el treball del programador.
- ◊ Localitzar (abreviat, L10N): És el treball de realitzar la traducció de totes les cadenes de caràcters d'un programa, és a dir, el treball del traductor.

Per tant, primer haurem d'internacionalitzar i després, nosaltres o altra persona haurà de localitzar el programa a una llengua concreta.

Al tall:

Com internacionalitzar el nostre projecte:

La llibreria més estesa i utilitzada per aquest afer es la gettext de GNU.

Per poder utilitzar-la necessitarem fer dues coses:

1. Integrar-la amb les autotools
2. Modificar el nostre codi font

Com integrar el gettext amb les autotools:

El primer que haurem de fer és modificar el fitxer configure.ac, per afegir-li un parell de línies:

configure.ac

```
AC_INIT([holamon], [0.1])
AM_INIT_AUTOMAKE([-Wall])
AM_GNU_GETTEXT_VERSION([0.17])
AM_GNU_GETTEXT([external])
AC_PROG_CC
AC_CONFIG_HEADERS([config.h])
AC_CONFIG_FILES([Makefile src/Makefile])
AC_OUTPUT
```

Línia a línia:

- AM_GNU_GETTEXT_VERSION: Ací indiquem la versió *exacta* de la llibreria gettext que anem a fer servir.

  + Informació[10]
- AM_GNU_GETTEXT: Amb aquesta macro indiquem que volem usar gettext. El paràmetre *external* el que indica és que la llibreria libintl necessària per al funcionament del gettext no serà distribuïda amb el nostre programa.

  + Informació[11]

A continuació, des de la carpeta arrel del projecte executem gettextize:

Terminal

```
$ gettextize --copy --no-changelog
```

El gettextize modifica l'estructura de directoris i edita alguns fitxers del nostre projecte per adaptar-lo a la necessitat de internacionalitzar-lo. Aquesta eina crearà un directori po amb una sèrie de fitxers necessaris per a la localització.

Paràmetres:

- --copy: Copia al nostre arbre de directoris els fitxers necessaris per a internacionalitzar el projecte. Podem fer servir symlink si volem que cree enllaços simbòlics en comptes de copiar els fitxers.
- -- no-changelog: Si no s'especifica, s'escriurà automàticament una entrada al changelog on s'informa de que el projecte ha estat adaptat a gettext

Quan finalitze l'execució, haurem de fer el següent:

Copiar el fitxer gettext.h a la nostra carpeta src:

Terminal

```
$ cp /usr/share/gettext/gettext.h src/
```

Hem de saber que igual que l'automake i l'autoconf ens obliguen a editar els fitxers configure.ac, Makefile.am i src/Makefile.am; el gettext ens obliga a editar-ne un parell més, el po/POTFILES.in i el po/Makevars. I igual que abans, malgrat tota l'estructura de fitxers que s'ha creat a dins la carpeta po/, nosaltres només ens haurem de preocupar per aquestos dos fitxers.

Crear el fitxer po/Makevars:

Per a fer açò haurem d'obrir la seua plantilla, po/Makevars.template, omplir-la amb les dades del nostre projecte i desar-la amb el nom de po/Makevars

po/Makevars

```
{...}
COPYRIGHT_HOLDER = El-nostre-nom
{...}
MSGID_BUGS_ADDRESS = $(PACKAGE BUGREPORT)
{...}
```

Línia a línia:

Aquestes són les dues línies que ens caldrà canviar:

- ◊ COPYRIGHT_HOLDER: El nom de qui té el copyright de la nostra revolucionària obra d'enginyeria.
- ◊ MSGID_BUGS_ADDRESS: La direcció on informar d'errors en les traduccions. Si escrivim $(PACKAGE BUGREPORT), li estem donant el mateix valor que l'hàgem donat a la macro AC_INIT del configure.ac. En cas contrari haurem d'especificar una direcció de correu.

Editar el fitxer po/POTFILES.in:

En aquest fitxer hem d'escriure la llista de fitxers font que contenen cadenes de caràcters que s'han de traduir. En el nostre cas, només src/holamon.c:

po/POTFILES.in

```
src/holamon.c
```

Si fem una ullada als fitxers configure.ac i Makefile.am, veurem que el gettextize ha afegit algunes coses:

configure.ac:

- ◊ Un nou paràmetre al AC_CONFIG_FILES: S'ha afegit el fitxer po/Makefile.in.

Makefile.am:

- ◊ Un nou paràmetre al SUBDIRS: Com que hi ha un nou Makefile a la carpeta po, s'ha d'especificar ací.
- ◊ Noves macros ACLOCAL_AMFLAGS i EXTRA_DIST : Són necessàries per al funcionament del gettext. ACLOCAL_AMFLAGS indica que a l'interior de la carpeta m4/ hi ha noves macros que fa servir el gettext. D'altra banda, EXTRA_DIST llista una sèrie de fitxers addicionals que volem distribuir amb el nostre paquet, en concret, aquesta línia

inclourà al tar.gz el fitxer config.rpath necessari per al gettext.

El fitxer que resta, el src/Makefile.am, l'haurem d'editar nosaltres mateix:

src/Makefile.am

```
AM_CPPFLAGS = -DLOCALEDIR=\"$(localedir)\"
bin_PROGRAMS=holamon
holamon_SOURCES=holamon.c gettext.h
LDADD =$(LIBINTL)
```

Línia a línia:

- AM_CPPFLAGS: Amb aquesta macro li afegim paràmetres a la línia d'ordres amb la qual el Makefile compilarà les fonts que li especifiquem baix (bin_PROGRAMS).

  Amb el valor -DLOCALEDIR=\"$(localedir)\" estem definint la macro LOCALEDIR que contindrà la ruta dels fitxers de traduccions, per tant aquesta variable ha d'estar definida abans de compilar. Després veurem que accedirem a la macro LOCALEDIR des del codi font. Aquesta mena de variables es poden definir tant al Makefile.am con el config.h.

  + Informació[12]
- LDADD= $(LIBINTL): Aquí especifiquem les llibreries que volem enllaçar quan compilem els binaris de bin_PROGRAMS. Com que hem indicat al configure.ac que el nostre projecte no inclou la llibreria *libintl*, l'hem d'incloure ací.

  + Informació[13]

I ja haurem acabat d'integrar gettext a les autotools, ara hem d'editar el codi font del programa.

Com integrar-li gettext al nostre codi font:

Primer haurem d'afegir capçaleres necessàries:

holamon.c (fragment)

```
#include <config.h>
#include <locale.h>
#include "gettext.h"
```

En aquest ordre.

Després haurem d'inicialitzar el gettext, ho farem en les primeres línies de la funció main:

holamon.c (fragment)

```
int main()
{
        setlocale(LC_ALL, "");
        bindtextdomain(PACKAGE, LOCALEDIR);
```

```
        textdomain(PACKAGE);

        printf("Hello, world!\n");

        return 0;
}
```

Observacions:

◊ La macro PACKAGE es defineix al config.h i la LOCALEDIR al src/Makefile.am.

Ara haurem de marcar les cadenes que volem traduir, aquesta és la part més avorrida de fer, però en el nostre cas acabarem de seguida.

Començarem per definir una abreviació de la funció gettext perquè no ens embrute massa el codi.

holamon.c (fragment)

```
#define _(string) gettext(string)
```

I després, per a marcar les cadenes a traduir, ho farem amb l'abreviatura que hem definit _(string):

Per tant, la línia:

holamon.c (fragment)

```
printf("Hello, world!\n");
```

Es transformarà en:

holamon.c (fragment)

```
printf(_("Hello, world!\n"));
```

I si té paràmetres de printf?

En eixe cas, una cadena com aquesta:

```
printf("Hello, %s!\n", name);
```

S'ha de modificar d'aquesta manera:

```
printf(_("Hello, %s!\n"), name);
```

Quan acabem, el nostre codi ha de quedar així:

holamon.c

```
#include <stdio.h>

#include <config.h>
#include <locale.h>
#include "gettext.h"

#define _(string) gettext(string)

int main()
{
        setlocale(LC_ALL, "");
        bindtextdomain(PACKAGE, LOCALEDIR);
        textdomain(PACKAGE);

        printf(_("Hello, world!\n"));

        return 0;
}
```

I amb açò haurem acabat la internacionalització del nostre projecte. Podem executar de nou l'autoreconf per aplicar els canvis.

Terminal

```
$ autoreconf -i
```

Com localitzar el nostre programa:

Ens caldrà generar abans que res la plantilla de traduccions. Aquest és un fitxer que conté totes les cadenes del nostre programa i el qual farem servir com a base per començar a traduir.

Per generar-la, hem de construir el nostre projecte:

Terminal

```
$ ./configure
$ make
```

I si ens fiquem a la carpeta po/ veurem que s'ha creat a dins un nou fitxer anomenat holamon.pot amb aquest contingut.

holamon.pot

```
# SOME DESCRIPTIVE TITLE.
# Copyright (C) YEAR Free Software Foundation, Inc.
# This file is distributed under the same license as the PACKAGE package.
# FIRST AUTHOR <EMAIL@ADDRESS>, YEAR.
#
```

```
#, fuzzy
msgid ""
msgstr ""
"Project-Id-Version: holamon 0.1\n"
"Report-Msgid-Bugs-To: correu-electrònic\n"
"POT-Creation-Date: 2010-04-15 22:05+0200\n"
"PO-Revision-Date: YEAR-MO-DA HO:MI+ZONE\n"
"Last-Translator: FULL NAME <EMAIL@ADDRESS>\n"
"Language-Team: LANGUAGE <LL@li.org>\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=CHARSET\n"
"Content-Transfer-Encoding: 8bit\n"

#: src/holamon.c:34
#, c-format
msgid "Hello, world!\n"
msgstr ""
```

Aquesta serà la plantilla que necessitem. Per a fer-la servir, hem d'entrar a la carpeta po/ i executar la comanda msginit, la qual genera un fitxer de traducció en la llengua especificada basant-se en la plantilla.

Terminal

```
$ cd po/
$ msginit -l ca
$ msginit -l es
```

Per aquest exemple, farem dues traduccions, una al català i l'altra al castellà. Com que el nostre programa és molt senzill, no ens durà massa temps, els fitxers quedaran així:

ca.po

```
{...}
#: src/holamon.c:34
#, c-format
msgid "Hello, world!\n"
msgstr "Hola, món!\n"
```

es.po

```
{...}
#: src/holamon.c:34
#, c-format
msgid "Hello, world!\n"
msgstr "¡Hola, mundo!\n"
```

Si volem verificar que les traduccions no tenen cap error, ho podem fer amb les comandes:

Terminal

```
$ msgfmt --statistics --check ca.po
$ msgfmt --statistics --check es.po
```

Cada cop que agreguem una traducció al projecte, l'hem d'afegir al fitxer po/LINGUAS. Si no existeix, el crearem nosaltres:

Terminal

```
$ echo ca >> LINGUAS
$ echo es >> LINGUAS
```

La propera vegada que executem el make, llegirà aquest fitxer i compilarà els *.po corresponents.

I ja ho tindrem, amb açò haurem finalitzat la localització de l'holamon a dues llengües.

Per aplicar els canvis executem:

Terminal

```
$ cd ..
$ autoreconf -i
```

I per a crear el paquet distribuïble:

Terminal

```
$ ./configure
$ make
$ make distchek
```

Res més. Espere que aquest manual haja estat útil per a la comprensió i la manipulació de les autotools, i quede a disposició de qui em vulga contactar per enviar-me dubtes o amenaces.

Joan Lledó <joanlluislledo@gmail.com[14]>

Enllaços interessants:

[+] Referència de macros d'autoconf (Anglès)[15]

[+] Referència de macros d'automake (Anglès)[16]

[+] Manual clar i complet d'autotools (Anglès)[17]

[+] Descarrega aquest manual: PDF[18] ODT[19]

Lista de enlaces de este artículo:

1. http://www.gnu.org/software/hello/manual/autoconf/Initializing-configure.html
2. http://www.gnu.org/software/hello/manual/automake/Public-Macros.html
3. http://www.gnu.org/software/hello/manual/autoconf/C-Compiler.html
4. http://www.gnu.org/software/hello/manual/autoconf/Configuration-Headers.html
5. http://www.gnu.org/software/hello/manual/autoconf/Configuration-Files.html
6. http://www.gnu.org/software/automake/manual/autoconf/Output.html
7. http://www.gnu.org/software/hello/manual/automake/Subdirectories.html
8. http://ftp.gnu.org/old-gnu/Manuals/automake-1.6.1/html_chapter/automake_2.html#S
9. http://www.gnu.org/software/hello/manual/automake/Default-_005fSOURCES.html#Defa
10. http://www.gnu.org/software/hello/manual/gettext/AM_005fGNU_005fGETTEXT_005fVERS
11. http://www.gnu.org/software/hello/manual/gettext/AM_005fGNU_005fGETTEXT.html
12. http://www.gnu.org/software/hello/manual/automake/Flag-Variables-Ordering.html
13. http://www.gnu.org/software/hello/manual/automake/Linking.html
14. mailto:joanlluislledo@gmail.com
15. http://www.gnu.org/software/hello/manual/autoconf/Autoconf-Macro-Index.html
16. http://www.gnu.org/software/hello/manual/automake/Macros.html
17. http://www.lrde.epita.fr/~adl/autotools.html
18. http://bulma.net/~jlledo/articles/autotools/autotools-primers_passos.pdf
19. http://bulma.net/~jlledo/articles/autotools/autotools-primers_passos.odt
20. http://creativecommons.org/licenses/by-sa/3.0/es/deed.ca

E-mail del autor: joanlluislledo _ARROBA_ gmail.com
Podrás encontrar este artículo e información adicional en: http://bulma.net/body.phtml?nIdNoticia=2572