xpminfo.c

Interesting part of xpminfo.c

```
/* INCLUDES */
#include “config.h”

#ifdef HAVE_STDIO_H
#   include <stdio.h>
#else
#   error “You need stdio.h”
#endif

#ifdef HAVE_STDLIB_H
#   include <stdlib.h>
#else
#   error “You need stdlib.h”
#endif

#ifdef HAVE_XPM_H
#   include <xpm.h>
#else
#   error “You need xpm.h”
#endif

/* CONSTANTS */
#define PRINT_INFORMATION_SPEC “%d”
```

config.h (config.h.in _after_ configure runs)

```
* config.h.  Generated automatically by configure.  */
/*
 * Config.h
 *
 * Defines all the possibilities.... :)
 */

/* Define if you have stdio.h */
#define HAVE_STDIO_H 1

/* Define if you have stdlib.h */
#define HAVE_STDLIB_H 1

/* Define if you have xpm.h */
/* #undef HAVE_XPM_H */

/* END OF LINE */
```

config.h.in

```
/*
 * Config.h
 *
 * Defines all the possibilities.... :)
 */

/* Define if you have stdio.h */
#undef HAVE_STDIO_H

/* Define if you have stdlib.h */
#undef HAVE_STDLIB_H

/* Define if you have xpm.h */
#undef HAVE_XPM_H

/* END OF LINE */
```

Makefile (after configure has run)

```
# Generated automatically from Makefile.in by configure.
##
## Makefile for xpminfo
##

#### Start of system configuration section. ####

srcdir = $(VPATH)

CC = gcc

INSTALL = /afs/eos.ncsu.edu/contrib/gnu/bin/install -c
INSTALL_PROGRAM = ${INSTALL}
INSTALL_DATA = ${INSTALL} -m 644

DEFS = -DHAVE_CONFIG_H
LIBS =   -lXt -lX11  -L/usr/openwin/lib -R/usr/openwin/lib -lsocket  -lnsl

CFLAGS = -g  -I/usr/openwin/include
LDFLAGS = -g

prefix = /usr/local
exec_prefix = $(prefix)

bindir = $(exec_prefix)/bin


# Prefix to be prepended to each installed program, normally empty or `g'.
binprefix =

#### End of system configuration section. ####
```

Makefile.in

```
##
## Makefile for xpminfo
##

#### Start of system configuration section. ####

VPATH = @srcdir@
srcdir = $(VPATH)

CC = @CC@

INSTALL = @INSTALL@
INSTALL_PROGRAM = @INSTALL_PROGRAM@
INSTALL_DATA = @INSTALL_DATA@

DEFS = @DEFS@
LIBS = @LIBS@ @X_PRE_LIBS@ -lXt -lX11 @X_LIBS@ @X_EXTRA_LIBS@

CFLAGS = -g @X_CFLAGS@
LDFLAGS = -g

prefix = /usr/local
exec_prefix = $(prefix)

bindir = $(exec_prefix)/bin


# Prefix to be prepended to each installed program, normally empty or `g'.
binprefix =

#### End of system configuration section. ####
```

configure.in

```
dnl Process this file with autoconf to produce a configure script.
AC_INIT(xpminfo.c)
AC_PROG_CC
AC_PROG_CPP
AC_PROG_INSTALL
dnl AC_STDC_HEADERS
AC_HAVE_HEADERS(stdio.h stdlib.h xpm.h)
AC_PATH_XTRA
dnl AC_CHECK_LIB(Xpm, XpmReadFileToPixmap)
AC_CONFIG_HEADER(config.h)
AC_OUTPUT(Makefile)

dnl END OF LINE
```

Needed Files

Before autoconf

```
c00753-400wi:xpminfo-0.2>ll
total 15
-rw-r--r--   1 tpmatthe ncsu       1189 Aug 18 17:47 Makefile
-rw-r--r--   1 tpmatthe ncsu        527 Jun 18 18:22 NOTES
drwxr-xr-x   2 tpmatthe ncsu       2048 Aug 19 15:58 samples/
-rw-r--r--   1 tpmatthe ncsu       7768 Aug 18 17:47 xpminfo.c
-rw-r--r--   1 tpmatthe ncsu       1218 Aug 19 15:58 xpminfo.prj
```

After autoconf

```
-rw-r--r--   1 tpmatthe ncsu       1770 Jul 11 17:05 Makefile.in
-rw-r--r--   1 tpmatthe ncsu        527 Jun 30 11:20 NOTES
-rw-r--r--   1 tpmatthe ncsu        243 Jul  9 14:24 config.h.in
-rw-r--r--   1 tpmatthe ncsu        306 Jul 11 17:00 configure.in
drwxr-xr-x   2 tpmatthe ncsu       2048 Aug 19 15:58 doc/
-rwx------   1 tpmatthe ncsu       5585 Jun 30 19:17 install-sh*
-rwxr-xr-x   1 tpmatthe ncsu        616 Jun 30 16:51 mkinstalldirs*
drwxr-xr-x   2 tpmatthe ncsu       2048 Jun 30 11:20 samples/
-rw-r--r--   1 tpmatthe ncsu       7983 Aug 18 20:38 xpminfo.c
-rw-r--r--   1 tpmatthe ncsu       1619 Aug 18 20:56 xpminfo.prj
```

Why AUTOCONF

- C code is fairly portable, but the API's and libraries used are not.
- Systems vary a great deal by vendor, release, and local modification.
- "Configuration" must be compile time: libraries are added and changed fairly frequently.
- IMake (its only major competitor) failed; most sites don't keep IMake updated after X gets built.
- Writing a custom configure script for each package is boring and repetitive.
- By just adding/using the parts that your application actually uses, the configure script can be "updated" anytime a new version of autoconf comes out.
- The shell (sh) and some basic and historical utilities (cut, uniq, echo) are the only remotely common parts of Unix systems, and even they vary way too much.

What is AUTOCONF

- AUTOCONF is a “build system” that creates a shell script.
- It uses M4 for the processing.
- Its language is a very generic sh syntax.
- The original developer creates or modifies certain files (at least Makefile.in and configure.in) to work with AUTOCONF.
- The original developer runs “autoconf” itself to generate the configure shell script. The person who “ports” or “installs” the new package on a particular system runs the configure shell script.
- The user doesn’t need anything but the generated configure script. (Not M4, not AUTOCONF, not a “database”).

CYGNUS AUTOCONF

- What is autoconf (why its not just “configure”)?
- Why autoconf?
- What files are needed?
- What changes (syntax) is needed?
- What tools does the developer need?
- What tools does the user need?
- What commands does the developer need to run?
- What commands does the user need to run?
- Where to get more information?