README FOR CLIENT DISPLAY LIBRARY (CDL) V1.7
Updated May 1, 2000

---

21 Feb 1997	Initial V1.0 Release
04 Mar 1998	V1.6 Upgrade Release
01 May 2000	V1.7 Upgrade Release

See also the post-distribution notes at the end of this file. These are continually updated after the release as any problems are encountered.

---

1. INTRODUCTION

The Client Display Library (CDL) is a host interface for C or Fortran programs allowing them to display images or overlay graphics to display servers such as XImtool or SAOimage / SAOtng. High-level procedures allow IRAF or FITS images to be displayed simply, other routines permit access to all other server functions (e.g. cursor and image readback, frame selection, etc). The library also features a number of functions for doing image overlay graphics; supported graphics primitives include numerous point shapes, lines, circles, ellipses, polygons, annular shapes, and text.

1.1 WHAT'S NEW IN THIS RELEASE

The CDL V1.7 release is primarily a bug-fix distribution, there are no new features. User's should however consult the following section to learn more about interface changes.

The initial CDL release provided most of the functionality required for image display or graphic overlay clients, since then a number of new features have crept in that were needed. The V1.6 version number reflects the stop-n-start nature of this work, more work is planned but a release of the new version is long overdue. New features include:

- Support for the new V2.11 IRAF OIF image format
- A prototype SPP language binding
- Full ANSI C function prototype support
- Support for variable line widths
- A selection of "dashed" line styles for polyline/polygon markers
- A selection of text fonts including
  ```
  ROMAN           FUTURA          TIMES
  GREEK           BOLD
  ```
- Sub/Superscripting text
- In-line font changes
- Support for text line widths
- A "virtual" display server that may also be used as a 'proxy' server providing an image display "tee" functionality
- Numerous bug fixes

See the CDL Reference Guide in the 'doc' subdirectory for details on new features.

1.2 THINGS TO WATCH OUT FOR

The V1.7 release contains an interface change to the cdl_readCursor()

routine requiring all client programs to be recompiled. The routine now has an additional 'wcs' argument to return the WCS of the cursor read. This is to allow applications to get the frame number of the read from the wcs (i.e. wcs is coded as "frame * 100 + wcs". The new calling sequence is

```
        cdl_readCursor  (cdl, sample, &x, &y, &wcs, &key)
```

Users should consult the Revisions file for a list of additional changes and bug fixes. CDL is available standalone or as part of the X11IRAF V1.2 release.

The V1.6 release contains two small interface changes which may cause problems, otherwise the new library should work seemlessly with existing applications. The interface changes involve the routines

```
        cdl_readIRAF()      and        cdl_readFITS()
```

These routines now have an extra 'title' argument to return the title string from the image which allows the hi-level display routines to include this information in the WCS string. This argument is a character pointer and affects the Fortran and SPP bindings as well. See the Reference Guide for additional information.

New features such as line styles and text fonts have been well-tested but the use of ANSI prototypes and the SPP binding have only been minimally exercised. Anyone finding a problem with these features should send a bug report the iraf@noao.edu help address, we can probably provide a quick fix for most bugs.

2. Documentation and Example Applications

Documentation for the package is located in the 'doc' subdirectory as both TROFF source and Postscript. The reference manual covers each of the routines available in the library and contains a summary of the interface as well as code samples.

Working programs are also included in the 'examples' subdirectory. Users may find it easier to start by modifying these programs for a particular need (e.g. add a new command for marking, an option for overlaying points from a file, etc) before starting an application from scratch to become familiar with the library.

Fortran programmers should consult the reference manual for information on array passing in the CDL. In particular, while the CDL can be used from IMFORT programs special care needs to be taken when displaying arrays read using the IMFORT routines.

3. Building the Package

The CDL package relies on the Imake facility to generate the makefiles needed to compile the package. To build the CDL use the following command:

```
        % xmkmf                         # build the parent Makefile
        % make World                    # compile it all
```

There is no supported "make install" option, all files will be left in the CDL working directory following the World build. Users of the package must either build the library and install the lib and include files by hand in their final destination, or use compiler "-I" and "-L" flags to find the files when compiling clients.

As an alternative for sites that don't have Imake installed or who encounter problems you can try the generic makefiles provided. To use these simply do

```
        % configure                      # create the makefiles
        % make World                     # compile it all
```

Programs using the CDL will need one of three include files depending on the language interface being used: the "cdl.h" file for C programs, the "cdlftn.inc" file for Fortran programs, or the "cdlspp.h" file for SPP tasks. The C and SPP include files is required, the Fortran include file is optional and just defines symbolic names for various constants. Fortran programmers not using the cdlftn.inc file should see to Reference Manual for values of symbolic constants. The library file to be linked is "libcdl.a", all programs using the CDL must also link with the system math library.

```
***************************************************************************
*****  NOTE: Solaris users will also need to include "-lnsl -lsocket"  *****
*****  when linking programs to include the socket interfaces.         *****
***************************************************************************
```

4. Reporting Problems

Each of the CDL functions has been tested individually but there is no practical way to test how various combinations of routines may be used in any given application so bugs are inevitable. If you think you've found a bug, have a suggestion for future enhancements, or just have a question about how something works, contact

iraf@noao.edu

Messages may also be posted to the adass.iraf.programming newsgroup so that others may benefit from the exchange. If you're not familiar with the ADASS newsgroup hierarchy send a message

irafinfo@iraf.noao.edu

with the one-line message "get iraf newsgroups" in the body of the message for more information. A small code sample demonstrating the problem would help greatly in finding the solution. Debugging output for the package may be enabled by defining a unix CDL_DEBUG environment variable. For example

```
        % setenv CDL_DEBUG      0               # minimal debug output
        % setenv CDL_DEBUG      1               # CDL and IMD debug info
        % setenv CDL_DEBUG      2               # print it all
```

Send this output along with the code sample. An FAQ may be added later if needed.

Mike Fitzpatrick
NOAO/IRAF Project
May 2000

```
===========================
= Post-Distribution Notes =
```

===========================

None at this time.