GNOME 1.0 Library Roadmap

Michael Fulbright
Red Hat Software, Inc.

drmike@redhat.com

The GNOME desktop environment is built on top of numerous support libraries which handle the many tasks expected of a modern desktop. Using these libraries, the developer can build a fully integrated application that fits into the GNOME desktop seamlessly. This document provides a guide to the function and relationships among these libraries.

Basic GNOME Libraries

Table 1. GNOME Libraries

Library Name	Function
libaudiofile	Reads common audio file formats
libgdk_imlib	Loads and manipulates image formats
libgtk	GIMP Toolkit, widget used in GNOME
libgnome	GNOME utility functions
libgnomeui	GNOME UI utility functions
libglib	String functions, hashes, lists, etc
libgdk	Lower level drawing toolkit
libesd	Client interface to esound daemon

Modern desktop applications demand a wide variety of services from the underlying application framework. In addition to the expectations of a widget set, programmers need tools to access and display audio and image information as well.

libaudiofile

The `libaudiofile` library can read a wide variety of audio file formats (AIFF, AIFC, WAV, and NeXT/Sun au). Once samples have been loaded using `libaudiofile`, `libesd` routines are available to play these samples via the Esound sound daemon. The Esound sound daemon allows several processes to access the sound hardware simultaneously. Esound is required because the Linux kernel does not allow multiple process access. Esound also supports sound over a network. Sound samples can be preloaded into Esound for later playback, which signifcantly reduces latency on network connections since the sample does not have to be repeatedly sent over the wire.

libgdk_imlib

GUI applications require extensive use of images to create a friendly and comfortable user interface. Traditionally it has been difficult to load all the common graphic file formats into X11 applications. The `libbgdk_imlib` library addresses this issue by providing convenient and powerful functions to load multiple file formats (JPEG, GIF, TIFF, PNG, XPM, PPM, PGM, PBM, and BMP). These files are converted to an internal 24 bit RGB representation, and utility functions exist to scale as well as render from 24 bit RGB to a variety of other color depths (with dithering if desired). Input image files are cached internally by `libbgdk_imlib` to improve performance in applications which repeatedly use images.

libgtk

The `libgtk` library is the GIMP toolkit library. It is a professional quality widget set which, in many ways, is superior to other widget sets. GNOME applications are written entirely using `libgtk` for all GUI elements (buttons, menus, scrollbars, etc).

libgnome

The `libgnome` library provides many utility routines related to the GNOME desktop environment. Among the capabilities provided are config file support for applications to store persistent data, support for metadata (data attached to file objects, like the icon to display for a particular file type), and support for loading help documents into the GNOME help browser. An interface is also provided so GNOME applications can talk to the GNOME session manager. Finally, routines exist to configure how different mime-types are handled by GNOME and the GNOME file manager.

libgnomeui

GNOME applications use the `libgnomeui` library extensively. This library contains toolkit extensions to the GTK+ widget set. Users can easily create dialog boxes, and message boxes, as well as menubars, toolbars, and status lines. An extensive array of stock icons is provided for the programmer's use in dialogs, menu entries, and buttons.

Because all GNOME applications will use `libgnomeui` to create these common GUI elements, visual consistency is guaranteed. Another important capability provided is the GNOME canvas, which allows painless creation of complex interfaces. Similar in many ways to the Tk canvas, the GNOME canvas provides a framework to create address books, calendar applications, and spreadsheets.

CORBA—Enabled GNOME Libraries

CORBA is a mechanism that GNOME applications can use to access services from other GNOME components. The CORBA implementation used by GNOME is called ORBit.

Table 2. CORBA Libraries

Library Name	Function
`libIIOP`	Low level CORBA communications
`libORBitutil`	Convenience routines for ORBit
`libORBit`	CORBA API
`libgnorba`	GNOME CORBA framework

Applications will only access `libORBit` and `libgnorba` directly, as `libIIOP` and `libORBitutil` are used internally by ORBit.

libORBit

`libORBit` provides the implementation of the CORBA API. It is the ORB proper, and works in concert with `libIIOP` and the IDL compiler to provide the complete CORBA implementation to the application.

libgnorba

ORBit provides the CORBA framework to developers for problem solving. To expedite the use of CORBA in GNOME applications specifically, additional facilities are provided by `libgnorba`:

- CORBA intialization routine for all GNOME apps, to automatically take care of security and GTK+ main loop integration.
- Automatic name service bootstrapping and accessing.
- Standard object server directory, object server registration and activation.

Other GNOME Libraries

Several other useful libraries have been created for use by GNOME applications (although most will work with non-GNOME applications as well) which can help when solving different kinds of problems.

libxml

The `libxml` library assists in reading and parsing XML files. Several GNOME applications (glade and gnumeric among for example) use `libxml` when dealing with XML files.

libghttp

When accessing data via HTTP, the protocol used by web servers, the `libghttp` library can be helpful. It is designed with graphical applications in mind, so that requests can be made without block waiting on a response. This allows the graphical application to be responsive while a request is in progress.

Library Name to Package Name

The following table maps the name of the library (which resides in `/usr/lib`) to the name of the package(s) which contain them:

Table 3. CORBA Libraries

Library Name	Run-time Package	Development Package
`libaudiofile`	audiofile	audiofile-devel
`libesd`	esound	esound-devel
`libglib`	glib	glib-devel
`libgtk`	gtk+	gtk+-devel
`libgdk_imlib`	imlib	imlib-devel
`libgnome`	gnome-libs	gnome-libs-devel
`libgnomeui`	gnome-libs	gnome-libs-devel
`libgnorba`	gnome-libs	gnome-libs-devel
`libIIOP`	ORBit	ORBit-devel
`libORBitutil`	ORBit	ORBit-devel
`libORBit`	ORBit	ORBit-devel
`libxml`	libxml	libxml-devel
`libghttp`	libghttp	libghttp-devel

The "-devel" packages are only required for actual development using the library, and end-users will only require the runtime library (the non-"-devel" packages) to be installed to run a GNOME application.