Octave Gtk: LibGlade API Documentation

Muthiah A
gnumuthu@users.sf.net
National Institute of Technology,
Tiruchirapalli, India

March 2, 2005

Contents

1 Introduction

This is the API documentation of the Octave-LibGlade bindings, which are a part of the Octave-GTK project.

1.1 Purpose

This **Octave-Gtk** project API documentation will be useful for programmers to learn the API of LibGlade, and for hackers who want to extend **Octave-LibGlade** . These are explained, in details. Also various specifications, and standards the **Octave-Gtk** adheres to is also pointed out in this document.

1.2 Document conventions

1. References made, are indicated using [number]. eg: [1]
2. **Bold face** text is used to refer to projects . eg: **Octave-Gtk**
3. *Abbreviations* are italicized. eg: *GUI*

1.3 Intended audience and reading suggestions

1. Octave Users, Scientists, Engineers:
   Octave users will find out how **Octave-Gtk** and **Octave-LibGlade** can be useful to their work, and learn how to get, install and use **Octave-LibGlade** . As a user, reading [1.7] will suffice.
2. Developers:
   As a **Octave-LibGlade** hacker, developers are expected to understand the whole document, and to read from the resources & references used in making **Octave-LibGlade** .

1.4 Scope of the Development Project

Octave-Gtk aims to bring the full featured toolkit like **GTK**to GNU Octave, to make scientific computing tools with a GUI front end scripted from Octave itself.

This directly translates to end users, as a new and powerful paradigm for constructing GUI's with minimal knowledge of the constructs, and easy interface for scientific programs, in a really short time.**Octave-Gtk** and **Octave-LibGlade** will compete with other *Rapid Application Development* tools, in its ease of use, and short design-test-release cycle.

1.5 Definitions, Acronyms,and Abbreviations

1. **Octave-Gtk** : Octave-Gtk is the GTK binding for GNU Octave.
2. **Octave-LibGlade** : Octave-LibGlade is the LibGlade binding for GNU Octave.
3. **GTK**: Gimp Tool Kit
4. GNU Octave : GNU Octave is a high performance scientific computation tool

1.6 References

1. Octave-Gtk : `http://octave-gtk.sourceforge.org`
   Octave-Gtk binding, website Reference documentation for code, examples and tutorials.
2. GNU Octave :`www.octave.org`
   GNU Octave interpreter, Octave Language, Sources , Octave mailing lists
3. GTK : `www.gtk.org`
   GTK API documentation, GTK source, GTK mailing lists
4. Octave Wiki : `http://wiki.octave.org`
   Documentation on extending Octave, Coda docs on writing dynamic functions.
5. Octave-Forge : `http://octave.sourceforge.net`
   Extra packages, libraries and tools for GNU Octave.

1.7 Overview of Document

This document provides developers, **GTK**, and Octave users the power of **Octave-Gtk** , to integrate the GUI from **GTK**toolkit and add the scientific computation engine of GNU Octave to their programs, and enhance the developer time, and productivity in making scientific computing programs with GUI.

The mechanics of installing **Octave-Gtk** , learning and using the **GTK**API with GNU Octave are presented in detail, along with the requirements for using the **Octave-Gtk** software.

2 API Documentation

This section lists the functions present in the API, and a general description of most of the functions. The description follows the syntax like this

2.1 Documentation Convention

1. return type
2. function name
3. argument 1
4. argument 2
5. argument n

example:

1. void
2. -
3. glade_set_custom_handler
4. This function is used to set a custom widget drawing handler, which will not be used in the Octave-LibGlade bindings.
5. GladeXMLCustomWidgetHandler handler,
6. -
7. gpointer user_data,
8. -

2.2 API Functions

The functions available from LibGlade to Octave via the **Octave-LibGlade** are as follows.

1. glade_xml_get_type
2. glade_xml_new
3. glade_xml_new_from_buffer

4. glade_xml_construct
5. glade_xml_signal_connect
6. glade_xml_signal_connect_data
7. glade_xml_signal_autoconnect
8. glade_xml_signal_connect_full
9. glade_xml_signal_autoconnect_full
10. glade_xml_get_widget
11. glade_xml_get_widget_prefix
12. glade_xml_relative_file
13. glade_set_custom_handler

2.2.1 GType glade_xml_get_type(void)

- GType
-
- glade_xml_get_type
- This function returns the glade xml type id within the GTK Type system. Use this to check the type of widget if its a GType widget or not. eg: G_TYPE_CHECK_INSTANCE(obj,glade_xml_get_type()) will return 'TRUE' if obj is of the type GladeXML.
- void ,
- No arguments.

2.2.2 GladeXML * glade_xml_new(char * fname,char * root,char * domain)

- GladeXML *
- Returns a GladeXML structures, handle/id
- glade_xml_new

- The function loads the XML file description of the UI, and automatically builds the widget tree from the 'root' node specified. eg: xml=glade_xml_new("calculator.glade","window","");
- char * fname,
- The name of the .glade file containing the GUI description.
- char * root,
- Name of the widget from where the XML description will build the tree. Generally give the top level widget.
- char * domain,
- Set a empty string, or whatever.

2.2.3 GladeXML * glade_xml_new_from_buffer(char * buffer,int size,char * root,char * domain)

- GladeXML *
- Returns a GladeXML structures, handle/id
- glade_xml_new_from_buffer
- Takes the UI description from the string, and returns a widget tree. This is much like the glade_xml_new itself.
- char * buffer,
- Send a string which has the description of the UI.
- int size,
- set to -1
- char * root,
- Name of the root widget
- char * domain,
- Set to empty string.

2.2.4 gboolean glade_xml_construct(GladeXML * self,char * fname,char * root,char * domain)

- gboolean
- return true on success.
- glade_xml_construct
- It constructs the xml description into a filename from the given root
- GladeXML * self,
- the XML widget tree
- char * fname,
- filename to save the output
- char * root,
- name of widget from where you save the tree.
- char * domain,
- set to empty string.

2.2.5 void glade_xml_signal_connect(GladeXML * self,char * handlername,GCallback func)

- void
- -
- glade_xml_signal_connect
- The function is not implemented. Use glade_xml_signal_autoconnect(...) instead.
- GladeXML * self,
- -
- char * handlername,
- -
- GCallback func,
- -

2.2.6 void glade_xml_signal_connect_data(GladeXML * self,char * handlername,GCallback func,gpointer user_data)

- void
- -
- glade_xml_signal_connect_data
- Use glade_xml_signal_autoconnect(...) instead.
- GladeXML * self,
- -
- char * handlername,
- -
- GCallback func,
- -
- gpointer user_data,
- -

2.2.7 void glade_xml_signal_autoconnect(GladeXML * self)

- void
- -
- glade_xml_signal_autoconnect
- This function connects the whole widgets signal to the callbacks, present in the octave language, which you write. This process if fully automated.
- GladeXML * self,
- This the constructed widget tree.

2.2.8 void glade_xml_signal_connect_full(GladeXML * self,gchar * handler_name,GladeXMLConnectFunc func,gpointer user_data)

- void
- -
- glade_xml_signal_connect_full
- Use glade_xml_signal_autoconnect(...) instead
- GladeXML * self,
- -
- gchar * handler_name,
- -
- GladeXMLConnectFunc func,
- -
- gpointer user_data,
- -

2.2.9 void glade_xml_signal_autoconnect_full(GladeXML * self,GladeXMLConnectFun func,gpointer user_data)

- void
- -
- glade_xml_signal_autoconnect_full
- Use glade_xml_signal_autoconnect(...) instead.
- GladeXML * self,
- -
- GladeXMLConnectFunc func,
- -
- gpointer user_data,
- -

2.2.10 GtkWidget * glade_xml_get_widget(GladeXML * self,char * name)

- GtkWidget *
- Returns the widget of the given name.
- glade_xml_get_widget
- This function gets the widget, from the given name, by looking into the XML tree. In case it cannot find the widget, you will be returned 0.
- GladeXML * self,
- the XML widget tree.
- char * name,
- name of the widget you need.

2.2.11 GList* glade_xml_get_widget_prefix(GladeXML * self,char * name)

- GList*
- List of widgets with the same prefix.
- glade_xml_get_widget_prefix
- List of widgets with the same prefix.
- GladeXML * self,
- XML widget tree representation.
- char * name,
- Name of the widget prefix

2.2.12 gchar * glade_xml_relative_file(GladeXML * self,gchar * filename)

- gchar *
- filename as a string.
- glade_xml_relative_file
- This function takes the xml description and returns the relative filename
- GladeXML * self,
- XML widget tree description.
- gchar * filename,
- filename as string.

2.2.13 void glade_set_custom_handler(GladeXMLCustomWidgetHandler handler,gpointer user_data)

- void
- -
- glade_set_custom_handler
- This function is used to set a custom widget drawing handler, which will not be used in the Octave-LibGlade bindings.
- GladeXMLCustomWidgetHandler handler,
- -
- gpointer user_data,
- -

2.3 Product Functions

1. Create and use the GObject system and types
2. Make GUI programming easy with **GTK**
3. Allow creation of custom GUI components, which are interoperable with the C and Octave
4. Allow Octave scripts to interactively plot, perform actions with GUI
5. Create GTK/GNOME programs with Octave, so that numeric computation is not duplicated

2.4 User classes and characteristics

1. Octave Users, Scientists, Engineers:
   Users will generally learn from *octave-gtk-demo* command, on how to use **Octave-Gtk** for their octave work. Also GTK C API will guide them in learning the binding.
2. Developers:
   To hack **Octave-Gtk** , and enhance the code generator, please read though the source code in the source tree of octave-gtk-0.1.tgz. Also learn about GNU Octave, and GTK API tools. Developers can make test cases of octave scripts targeted at improving speed, detecting bugs.

2.5 Operating Environment

1. GTK+-2.0
2. Octave-2.1.50 compiled with dynamic loading enabled
3. Python
4. GNU/Linux or UNIX like system
5. any 32bit processor

2.6 Design and Implementation constraints

Implementation constraints are limited to the performance of the Octave interpreter. Consequently the callbacks and glue code, make **Octave-Gtk** programs **slower** than their corresponding GTK counterparts.

Octave GTK code has to be updated as and when GTK API changes. As of now, this cannot be overcome unless we design some kind of simpler API on top of what **Octave-Gtk** provides.

2.7 User Documentation

For programming convenience,all the function names in **Octave-LibGlade** are same as the standard C function names defined in LibGlade+ API. The names of C classes has been also preserved for sake of programming convenience.Thus for User Documentation the official GTK+/LibGlade can be referred except in few special cases.For all such cases our additional user documentation lists such peculiarities.

2.8 General Constraints, Assumptions, Dependencies, Guidelines

There is no memory limitations but because of bulky Octave interpretor the response time may be slightly more.

2.9 User View of Product Use

This library aims to make GUI programming(with Octave language) a cakewalk. Use of Octave language coupled with GTK+ makes GUI programming easy.Users will be relieved from the headache of handling pointers and can use powerful Octave language, and **Octave-LibGlade** makes it really easy.

2.10 Software Interfaces

GTK library version 2.0 or later, is the target of this **Octave-Gtk** binding, and we need *GTK+-2.0*.Glade UI designer, and LibGlade library version 2.0 or later, is the target of this **Octave-LibGlade** binding, and we need *LibGlade-2.0*. Also Octave extensions work if and only if Octave is compiled with *dynamic loading* enabled, in versions 2.1.50 or later. This is necessary for the working of **Octave-Gtk** binding binary installation.

For building **Octave-Gtk** though you need *Python* installed on your computer along with the above software, and their development packages.

i.e: development packages of GTK and Octave are also needed for building/hacking **Octave-Gtk** .

2.11 Communication Interface

Not Applicable.

3 System Features

This library aims to make GUI programming(with Octave language)a cakewalk.

3.1 API Export

This library aims to make GUI programming(with Octave language) possible first by exporting the GTK C API to Octave using the **Octave-Gtk** library.

4 Example

Simple LibGlade example. A sample Octave-LibGlade program is given here.

```
#! /usr/bin/octave -q
% Code AUtoGEnerated by Octave-Glade
xml=0;
% Beginning of Callbacks %

function changebtn_clicked(widget,data)
    disp('Change Button Clicked')
end

function changebtn_Clickeeed(widget,data)
    disp('Change Button Clicked Handler 2')
end

%function gtk_main_quit(widget,data) already in the library.
function main()
    global xml;

    gtk();   %load GTK
    glade(); %load GLADE library
    gtk_init();

    xml=glade_xml_new(``/welcome.glade","Welcome","");
    window = glade_xml_get_widget(xml,"Welcome")
    glade_xml_signal_autoconnect(xml);
    gtk_widget_show_all(window);
    gtk_main();
    return;
end
%starts the whole script
main();%free tolerance
```

4.1 Code Walkthrough: Using Octave-LibGlade

You may use the Octave-libGlade for using making GUI's as follows:

1. Install Octave-GTK & Octave-libGlade binding libraries first.

2. Generate the '.glade' file for your application using 'glade-2'. Mention all of the event handlers you want with this application in your Glade UI editor, so that it comes within the ".glade" file.

3. Now write a simple 'm' file in Octave to create the GUI & implement the same function calls as in the libGlade examples.

   (a) First you would load the 'gtk' & 'glade' libraries with the function calls 'gtk()' & 'glade()' in this order.

   (b) Then make sure you create the XML widget tree pointer with the filename as xml=glade_xml_new(¡filename¿,¡main toplevel widget-nam¿,¡domain¿) All the parameters are strings.

   (c) Find the main widget using glade_xml_get_widget() like this. calc=glade_xml_get_widget(x

   (d) To connect event handlers use the function call 'glade_xml_signal_autoconnect()' as follows.

   glade_xml_signal_autoconnect(xml);

   This function actually connects all the signals which you have specified in the '.glade' files to the 'Octave' callback handler functions you have created[You must write all these in Octave].

   (e) Code the rest like any other GTK Application, calling gtk_widget_show() & running the main loop with gtk_main().

[Example] See /examples/calc.m which is a complete simple calculator built on top of Octave-LibGlade & Octave-GTK.

Output : This is a simple GUI program with the Octave-GTK/LibGlade, which shows the power of Octave.

4.2 Data Migration

Migration from Matlab API is not supported currently.

4.3 User Training

Please read the examples from octave-gtk-demo code, that helps you start off easily.

1. GTK C API documentation & Tutorial
2. Octave-Gtk API FAQ
3. Octave-Gtk examples && '$ octave-gtk-demo'

4.4 Installation

Installation from source is like this.

1. ./configure –prefix=/you/usr/directory
2. make
3. make install

5 About

Muthiah Annamali is the project lead,Octave-GTK team.

Octave-GTK is a language bindings project to connect Octave, a scientific computational engine with GTK, a world class GUI toolkit.

Please find out more at **Main Page** `http://octave-gtk.sourceforge.net`

6 Licensing

The Library is under GPL and thus can be freely modified and redistributed. We have followed the cyclic software development method. **Octave-Gtk** is **GPL**'ed as it links with GNU Octave, which is **GPL**'ed itself. Business rules,

are not applicable otherwise. This obviously means, all future derivative works, must be GPL'ed.

This documentation itself is licensed under GNU FDL.

Figure 1: A Simple LibGlade example