

Developing graphical user interfaces for embedded device – the open source way

In the past, and not too far, embedded devices used a very limited user interface capabilities, such as command line, dot matrix display and very limited LCD. Modern devices require rich user interfaces, ones that provide ease of use – nobody likes to take the time and learn how to use a new gadget, an interface which is nice to the eye – consumers may buy the device that has a shinier interface and not the one that has better features. In some cases, the embedded device needs to have a look and feel that resembles the desktop one as much as possible.

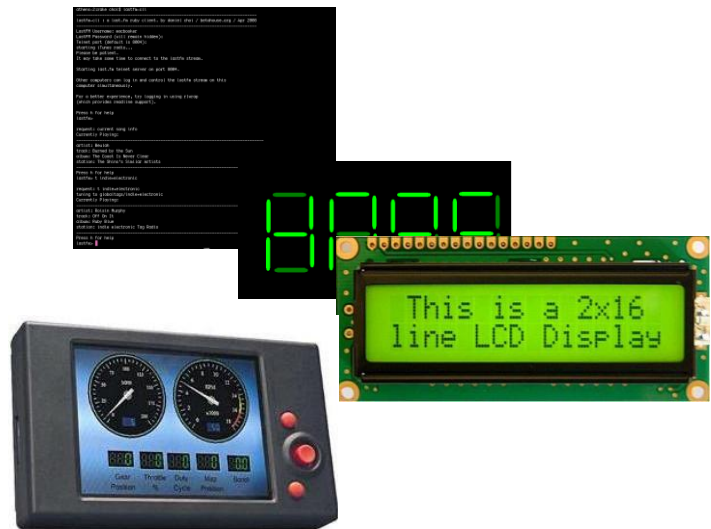
While the development of modern user interfaces on the desktop environment has evolved dramatically, that is not the case in the embedded devices one. A typical desktop computer is a well integrated environment, one that has an operating system, windows manager, graphics drivers and widget sets all designed to work seamlessly with each other, enabling the tools vendors to create development applications that provides RAD

(Rapid Application Development) capabilities which produce modern user interfaces. Unfortunately this is not the case in the embedded device user interface design, not all embedded operating systems include the needs for graphical user interfaces, and most of them do not have windows managers, graphics drivers, widget sets and most of all no development tools. Attempts to port graphical desktop environments to embedded devices proved to be unsuitable, desktop GUI requires too much resources and processing power to be a good fit for the embedded segment.

In the very basics a graphical user interface needs to provide two capabilities, user input and graphical data presentation. In order to achieve that, the following building blocks must be implemented:

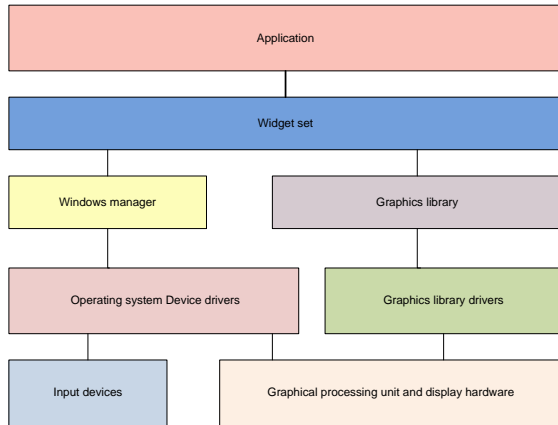
Drawing library, this library is responsible for basic shapes drawing, most user interface designs are two dimensional vectors based, and for that type of graphics the drawing library must support the following:

- Draw line, a line properties include start and end point, width, color and aliasing mode, a line may include pattern and end cups properties as well.
- Fill triangle, because every geometrical closed shape can be triangulated a triangle drawing properties include end points, fill color and aliasing mode, a triangle may include fill pattern and end cups.
- Image blitting, all interfaces must include some form of image drawing, the image is defined in form of bitmap and pixel format (the format is the relations between the red, green, blue and alpha channels, for example RGB32 is 8 bit for every channel).
- Alpha blending, to achieve fading and transparency effects two area of the interface screen needs to be blended using a given alpha value.



Developing graphical user interfaces for embedded device – the open source way

Most of the GPU (graphical processing units) today provides acceleration functionality that can be used for the drawing, the drawing library needs to implement drivers interface to utilize the acceleration functionality.

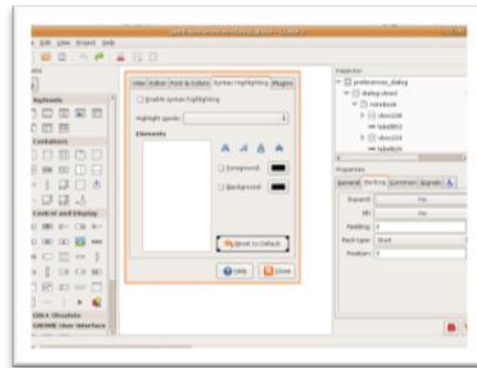


Typical System with User Interface application

example, click button is a widget, the graphical entities include drawing the button shape (filled rectangle and caption text), and the user interactions include changing the button from out to in display and passing indication to the application.

When it's come to design a GUI based application, the hardest task is envisioning the look of the applications.

This includes poisoning of the widgets, size of the GUI elements and combinations of colors and images. A Rapid Application Design tool provide a way to overcome those issues, a RAD tool is usually based on a canvas which is the application main window, into that canvas the designer can add widgets, set the widget properties and customize the look and feel of the application, because all of the design is done in a visual way the envisioning of the application look is intuitive. When the design is over the RAD tool will create a resource file that the application can link to or load at run time, all that the user have to do is add callback functions to handle user interaction with the application.

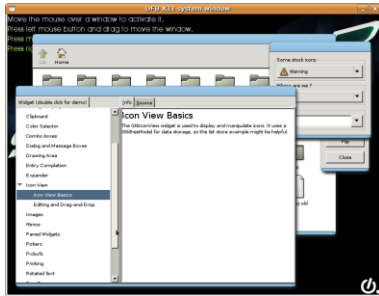


Glade user interface design tool

Developing graphical user interfaces for embedded device – the open source way

In the open source scene two projects are outstanding when it's come to graphical user interface development.

DirectFB, this project provides small and robust graphical library that is intended for embedded application development. This project provides all the facilities that are needed for rendering graphics and interaction with users while keep small footprint, low memory consumption and high performance, the features that DirectFB includes are:



UniQuE window manager for DirectFB

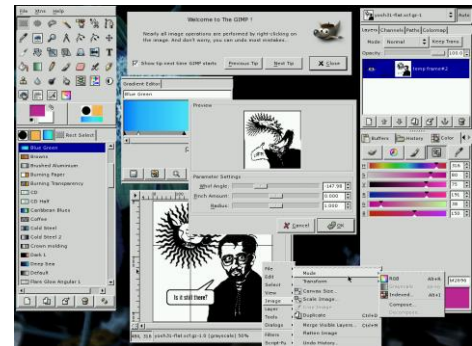
- Integrated windows managers, depend on the application needs this can go from undecorated fixed position windows to fully decorated and movable ones.
- Drawing library, this library provides all the rasterization functionality that may be required in order to draw user interfaces, more than that this library includes acceleration

drivers interface, which allows for high graphics performance while keeping host CPU utilization to a minimal.

- The project is designed with embedded application development in mind; as such it provides well documented methods for cross compiling the project to deferent target platforms.

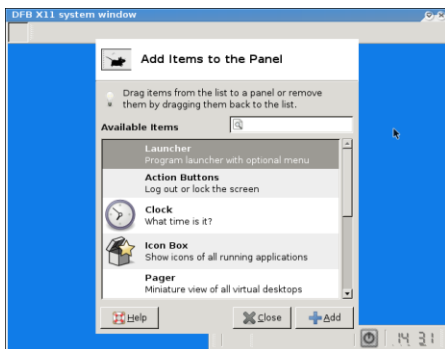
GTK+, this project is the base widgets set for the well known GNOME and XFCE desktop environment which have large number of applications that have been developed using this widgets set. GTK+ by itself relay on minimal set of dependencies and is highly portable between deferent backend graphical systems. One of the backend systems that GTK+ had been ported to is DirectFB, this combination (known as GTK-DFB) provide embedded application developer the advantages of having graphics backend that is intended for embedded need while using a popular widgets set.

GTK+ includes a RAD tool called **Glade**. Glade being based on GTK+ is a cross platform project, this provides the application developer to



GIMP over GTB-DFB

perform vast part of the development on his host PC, execute and test the designed GUI and application and when satisfied from the results cross compile the application code for the target platform and deploy the resource file created by Glade to the target device. This develop methodology enable embedded GUI designer to start and develop the application well before the target hardware is ready, using comfort ability of the host PC. When the target hardware is ready the application is seamlessly deployed on that hardware.



XFCE over GTK-DFB

Developing graphical user interfaces for embedded device – the open source way

Vindico Corporation offers custom application and device specific optimizations services. We have vast experience in embedded software development with emphasis on embedded GUI design, our service covers all part of embedded software projects:

- Board support packages and device drivers development
- Middleware and third party software integration
- Deploying and GTK-DFB on new platform and developing DirectFB drivers
- Provide graphics design services

Contact us at:

info@vindicocorp.com

www.vindicocorp.com

(613) 286-0339

