API STRATEGY & PRACTICE

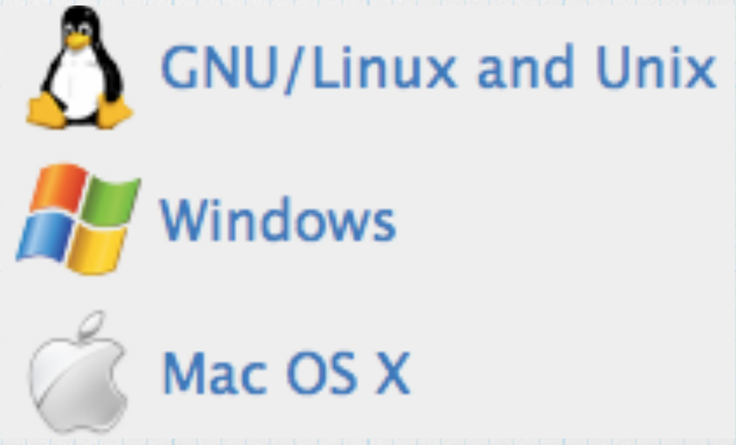
Starting with GTK

Julita Inca

THE LINUX FOUNDATION

What is GTK?

GTK+, or the GIMP Toolkit, is a multi-platform toolkit for creating graphical user interfaces.



GNU/Linux and Unix

Windows

Mac OS X

Source: https://www.gtk.org

Offering a complete set of widgets, GTK+ is suitable for projects ranging from small one-off tools to complete application suites.

Source: https://en.wikipedia.org/wiki/List_of_GTK%2B_applications#Puzzle_games

Stability

GTK+ is supported by a large community of developers and has core maintainers from companies such as

redhat

intel

Codethink

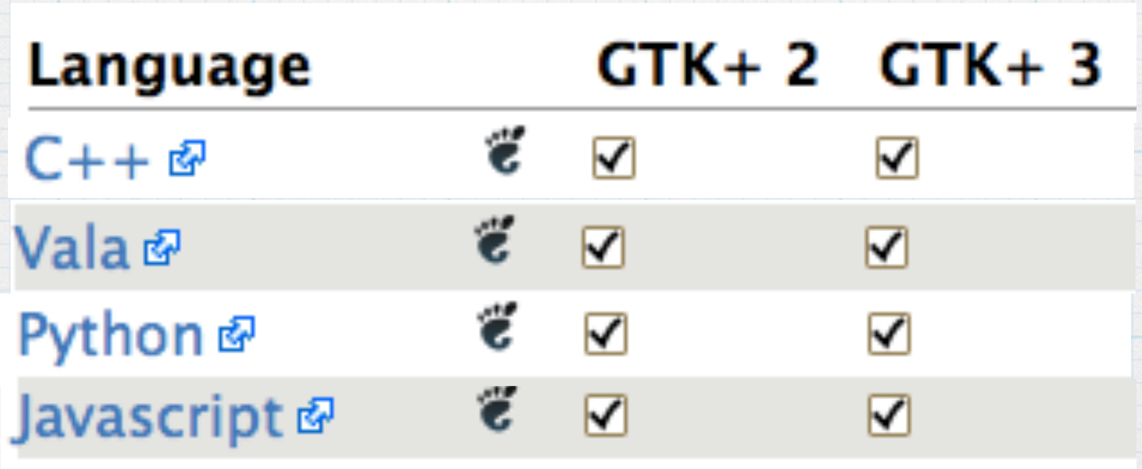
N

GNOME™

Source: https://www.gtk.org/features.php

Language Bindings

Language Bindings (or 'wrappers') allow GTK+ to be used from other programming languages, in the style of those languages.

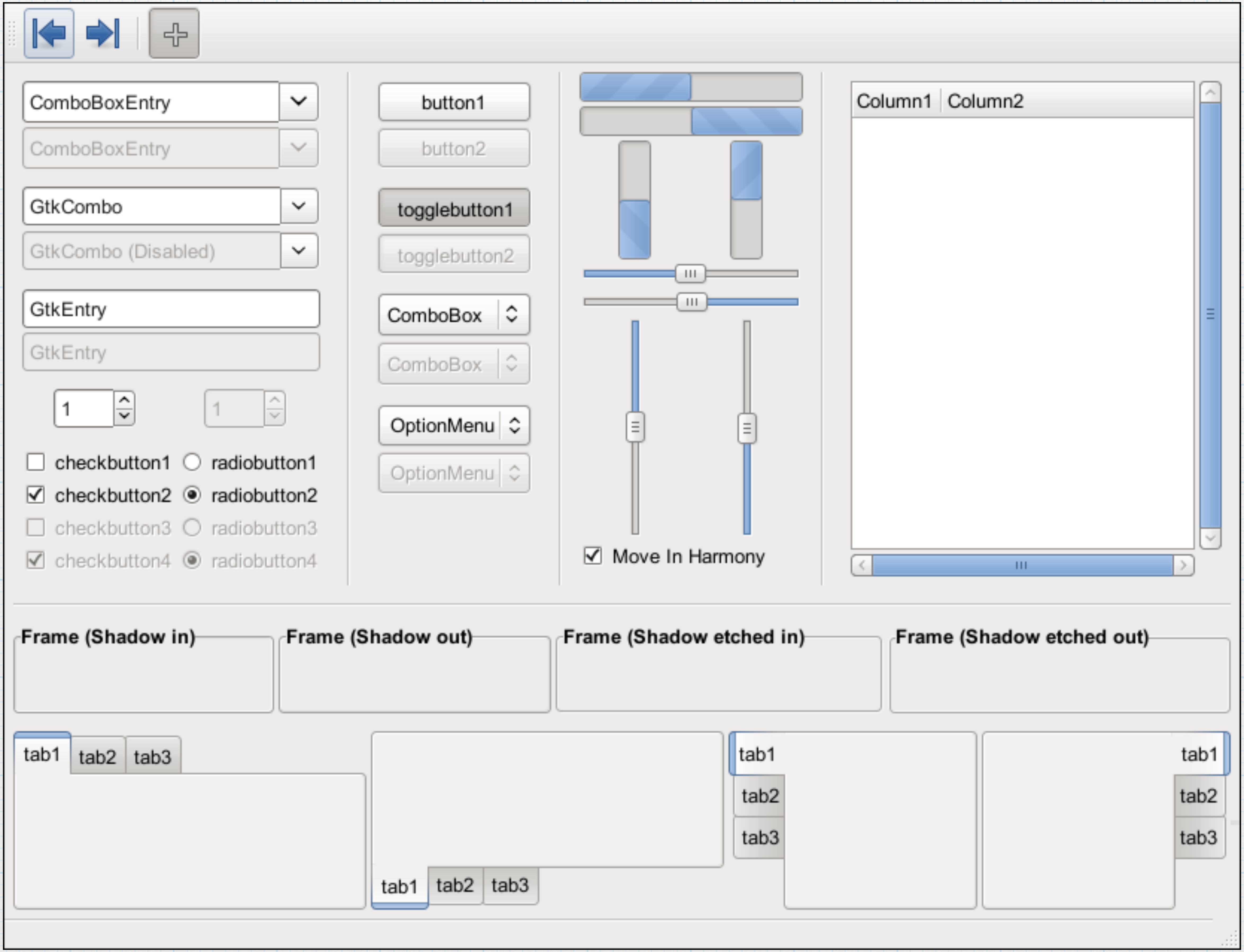


Language		GTK+ 2	GTK+ 3
C++	👣	☑	☑
Vala	👣	☑	☑
Python	👣	☑	☑
Javascript	👣	☑	☑

Source: https://www.gtk.org/language-bindings.php

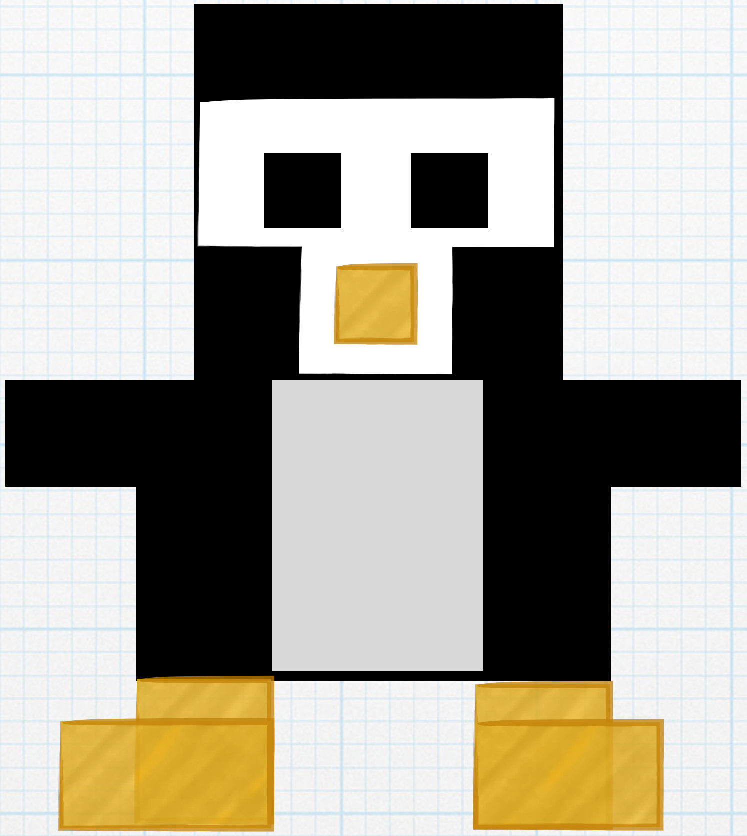
Interfaces

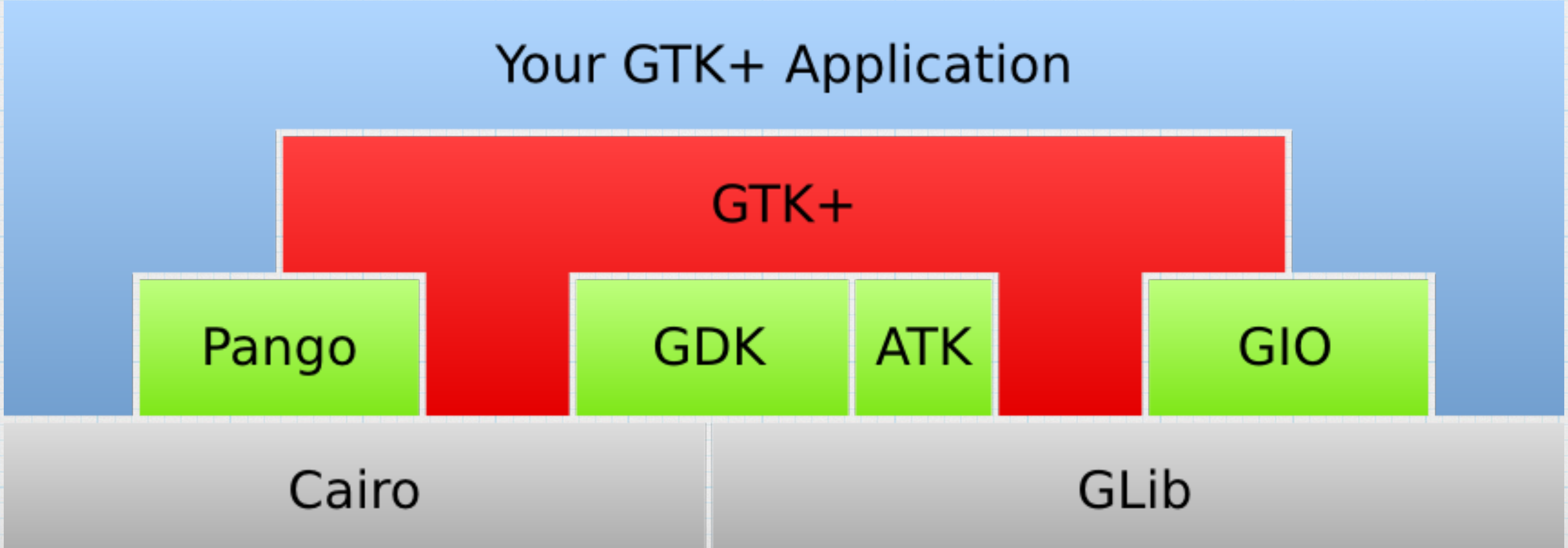
GTK+ has a comprehensive collection of core widgets and interfaces for use in your application. GTK+ has features pertaining to mobile and embedded platform requirements: Nokia 770 / N800 / N810 / N900, OLPC.



Source: https://www.gtk.org/features.php

Let’s Build our firsts widgets!!!





Your GTK+ Application

GTK+

Pango

GDK

ATK

GIO

Cairo

GLib

Commands to install, compile and run a GTK program:

To install on Fedora:

```
sudo dnf install libgtk-3-dev
```

To install on Ubuntu:

```
sudo apt-get install build-essential libgtk-3-dev
```

To compile:

```
gcc $(pkg-config --cflags gtk+-3.0) -o window window.c $(pkg-config --libs gtk+-3.0)
```

To run:

```
./window
```

Working with Window

```
#include <gtk/gtk.h>

int main (int argc, char *argv[])
{
    GtkWidget *window;


    git_init (&argc, &argv);

    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);


    g_signal_connect (window, "destroy", G_CALLBACK (gtk_main_quit), NULL);


    gtk_widget_show_all (window);

    gtk_main ();

    return 0;
}
```

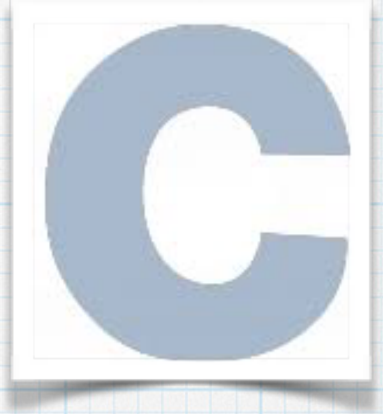
```
gcc $(pkg-config --cflags gtk+-3.0) -o window01 window01.c $(pkg-config --libs gtk+-3.0)
```

Adding features to a Window

```
#include <gtk/gtk.h>


int main (int argc, char *argv[])
{

    GtkWidget *window;


    gtk_init (&argc, &argv);


    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);

    gtk_window_set_default_size (GTK_WINDOW (window), 400, 600);

    gtk_window_set_title (GTK_WINDOW (window), "Linux Foundation");


    g_signal_connect (window, "destroy", G_CALLBACK (gtk_main_quit), NULL);


    gtk_widget_show_all (window);

    gtk_main ();

    return 0;
}
```

```
gcc $(pkg-config --cflags gtk+-3.0) -o window02 window02.c $(pkg-config --libs gtk+-3.0
```

Show a Label into the Window

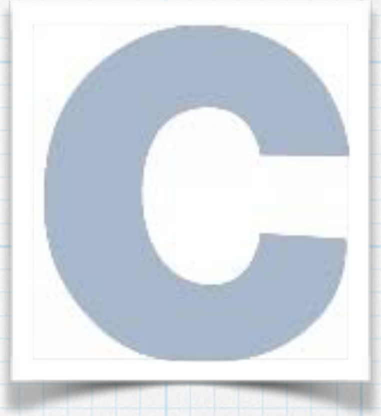


```
#include <gtk/gtk.h>

int main (int argc, char *argv[])
{

    GtkWidget *window, *label;


    gtk_init (&argc, &argv);


    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);

    gtk_window_set_default_size (GTK_WINDOW (window), 400, 600);

    gtk_window_set_title (GTK_WINDOW (window), "Linux Foundation");


    label = gtk_label_new ("Welcome to APISTRAT");

    gtk_container_add (GTK_CONTAINER (window), label);


    g_signal_connect (window, "destroy", G_CALLBACK (gtk_main_quit), NULL);


    gtk_widget_show_all (window);

    gtk_main ();

    return 0;
}
```

```
gcc $(pkg-config --cflags gtk+-3.0) -o window03 window03.c $(pkg-config --libs gtk+-3.0)
```

Adding a button to print a message

```
#include <gtk/gtk.h>

static void on_button_clicked (GtkWidget* widget, gpointer data)
{
    g_print ("Welcome to APISTRAT\n");
}

int main (int argc, char *argv[])
{

    GtkWidget *window, *button;


    gtk_init (&argc, &argv);


    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);

    gtk_window_set_default_size (GTK_WINDOW (window), 400, 600);

    gtk_window_set_title (GTK_WINDOW (window), "Linux Foundation");


    button = gtk_button_new_with_label ("Click me");

    gtk_container_add (GTK_CONTAINER (window), button);


    g_signal_connect (window, "detroy", G_CALLBACK (gtk_main_quit), NULL);

    g_signal_connect (button, "clicked", G_CALLBACK(on_button_clicked), NULL);


    gtk_widget_show_all (window);

    gtk_main ();

    return 0;
}
```



```
gcc $(pkg-config --cflags gtk+-3.0) -o window04 window04.c $(pkg-config --libs gtk+-3.0)
```

Button aligned to print a message

```
. . .
int main (int argc, char *argv[])
{

    GtkWidget *window, *button;

    gtk_init (&argc, &argv);

    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);

    gtk_window_set_default_size (GTK_WINDOW (window), 300, 300);

    gtk_window_set_title (GTK_WINDOW (window), "Linux Foundation");

    button = gtk_button_new_with_label ("Click me");

    gtk_widget_set_halign (button, GTK_ALIGN_CENTER);

    gtk_widget_set_valign (button, GTK_ALIGN_CENTER);

    gtk_container_add (GTK_CONTAINER (window), button);
. . .
```

```
gcc $(pkg-config --cflags gtk+-3.0) -o window05 window05.c $(pkg-config --libs gtk+-3.0)
```

Button to display a label (Box)

```
#include <gtk/gtk.h>


static void on_button_clicked (GtkWidget* widget, gpointer data)
{

    gtk_label_set_text (GTK_LABEL (data), "Welcome to APISTRAT");

}

int main (int argc, char* argv[])
{

    GtkWidget *window, *label, *button, *box;


    gtk_init (&argc, &argv);


    window = gtk_window_new (GTK_WINDOW_TOPLEVEL);

    gtk_window_set_default_size (GTK_WINDOW (window), 400, 600);

    gtk_window_set_title (GTK_WINDOW (window), "Linux Foundation");


    label = gtk_label_new (NULL);

    button = gtk_button_new_with_label ("Click me”);


    box = gtk_box_new (GTK_ORIENTATION_VERTICAL, 6);

    gtk_box_pack_start (GTK_BOX (box), button, TRUE, TRUE, 0);

    gtk_box_pack_start (GTK_BOX (box), label, TRUE, TRUE, 0);

    gtk_container_add (GTK_CONTAINER (window), box);
```

```
gcc $(pkg-config --cflags gtk+-3.0) -o window06 window06.c $(pkg-config --libs gtk+-3.0)
```

Button aligned to display a aligned label

```
#include <gtk/gtk.h>

static void on_button_clicked (GtkWidget* widget, gpointer data)
{

    gtk_label_set_text (GTK_LABEL (data), "Welcome to APISTRAT");

}

int main (int argc, char* argv[])
{ . . .

    button = gtk_button_new_with_label ("Click me");

    gtk_widget_set_halign (button, GTK_ALIGN_START);

    gtk_widget_set_valign (button, GTK_ALIGN_START);


    label = gtk_label_new (NULL);

    gtk_widget_set_halign (label, GTK_ALIGN_END);

    gtk_widget_set_valign (label, GTK_ALIGN_END);


    box = gtk_box_new (GTK_ORIENTATION_VERTICAL, 6);

    gtk_box_pack_start (GTK_BOX (box), button, TRUE, TRUE, 0);

    gtk_box_pack_start (GTK_BOX (box), label, TRUE, TRUE, 0);

    gtk_container_add (GTK_CONTAINER (window), box);


    g_signal_connect (window, "destroy", G_CALLBACK (gtk_main_quit), NULL);

    g_signal_connect (button, "clicked", G_CALLBACK (on_button_clicked), label);
```

```
gcc $(pkg-config --cflags gtk+-3.0) -o window07 window07.c $(pkg-config --libs gtk+-3.0)
```

Using an entry to print into a terminal

```
static void on_button_clicked (GtkWidget *widget, gpointer data)
{

    g_print ("%s\n", gtk_entry_get_text (GTK_ENTRY (data)));

}

int main (int argc, char *argv[])

{. . .
    gtk_container_set_border_width (GTK_CONTAINER (window), 50);


    entry = gtk_entry_new ();

    button = gtk_button_new_with_label ("Click me");


    box = gtk_box_new (GTK_ORIENTATION_VERTICAL, 6);

    gtk_box_pack_start (GTK_BOX (box), button, TRUE, TRUE, 0);

    gtk_box_pack_start (GTK_BOX (box), entry, TRUE, TRUE, 0);

    gtk_container_add (GTK_CONTAINER (window), box);


    g_signal_connect (window, "destroy", G_CALLBACK (gtk_main_quit), NULL);

    g_signal_connect (button, "clicked", G_CALLBACK (on_button_clicked), entry);
```

```
gcc $(pkg-config --cflags gtk+-3.0) -o window08 window08.c $(pkg-config --libs gtk+-3.0)
```

An entry that print a label into a window

```
...
static void on_entry_activate (GtkEntry *entry, GtkLabel *label)
{

    gtk_label_set_text (GTK_LABEL (label), gtk_entry_get_text (GTK_ENTRY (entry)));

}

int main (int argc, char *argv[])

{
. . .

    entry = gtk_entry_new ();

    gtk_entry_set_activates_default (GTK_ENTRY(entry), TRUE);

    label = gtk_label_new ("Enter a message and press <ENTER> ...");


    box = gtk_box_new (GTK_ORIENTATION_VERTICAL, 6);

    gtk_box_pack_start (GTK_BOX (box), label, TRUE, TRUE, 0);

    gtk_box_pack_start (GTK_BOX (box), entry, TRUE, TRUE, 0);

    gtk_container_add (GTK_CONTAINER (window), box);


    g_signal_connect (window, "destroy", G_CALLBACK (gtk_main_quit), NULL);

    g_signal_connect (G_OBJECT (entry), "activate", G_CALLBACK (on_entry_activate), label);
...
```

```
gcc $(pkg-config --cflags gtk+-3.0) -o window09 window09.c $(pkg-config --libs gtk+-3.0)
```

Recommendations

Please consider using the formatting rules:

- A single space after a comma.
- A single space after a keyword
- A single space after a function name, before the opening parenthesis.
- Indentation of four spaces.

@yulwitter

jinca@gnome.org

LINUX
AT|UNI

https://github.com/jinca/GTK/tree/master/StartingGTK_C