

An example of using GTK and Gnuplot together

Below is a simple C program that combines **GTK** and **GNUplot** in a Linux environment. The program creates a GTK-based GUI with a button. When the button is clicked, it generates a sine wave dataset and uses GNUplot to plot it, displaying the plot in a GNUplot window. This example demonstrates how to integrate GTK for user interaction and GNUplot for data visualization.

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

// Callback function for button click: generate data and plot with GNUplot
static void on_button_clicked(GtkButton *button, gpointer user_data) {
    // Step 1: Generate sine wave data and write to a file
    FILE *data = fopen("data.txt", "w");
    if (!data) {
        g_print("Error opening data file\n");
        return;
    }

    for (double x = 0; x <= 10; x += 0.1) {
        fprintf(data, "%f %f\n", x, sin(x));
    }
    fclose(data);

    // Step 2: Open a pipe to GNUplot
    FILE *gp = popen("gnuplot -persist", "w");
    if (!gp) {
        g_print("Error opening GNUplot\n");
        return;
    }

    // Step 3: Send GNUplot commands
    fprintf(gp, "set title 'Sine Wave'\n");
    fprintf(gp, "set xlabel 'x'\n");
    fprintf(gp, "set ylabel 'sin(x)'\n");
    fprintf(gp, "plot 'data.txt' with lines title 'sin(x)'\n");
    fflush(gp);
    pclose(gp);

    g_print("Plot generated!\n");
}

// Callback function for window close
static void on_window_destroy(GtkWidget *widget, gpointer user_data) {
    gtk_main_quit();
}

int main(int argc, char *argv[]) {
    // Step 1: Initialize GTK
    gtk_init(&argc, &argv);

    // Step 2: Create the main window
    GtkWidget *window = gtk_window_new(GTK_WINDOW_TOPLEVEL);
    gtk_window_set_title(GTK_WINDOW(window), "GTK + GNUplot Example");
    gtk_window_set_default_size(GTK_WINDOW(window), 300, 200);

    // Step 3: Create a button
    GtkWidget *button = gtk_button_new_with_label("Plot Sine Wave");

    // Step 4: Add button to the window
    gtk_container_add(GTK_CONTAINER(window), button);
```

```
// Step 5: Connect signals
g_signal_connect(button, "clicked", G_CALLBACK(on_button_clicked), NULL);
g_signal_connect(window, "destroy", G_CALLBACK(on_window_destroy), NULL);

// Step 6: Show all widgets and start the main loop
gtk_widget_show_all(window);
gtk_main();

return 0;
}
```