# 报告内容

## 项目概述

完成一个由C语言，glade图形开发，gtk+库互相配合运作的计算器程序。实现简单的点击数字和运算符，文本框同步显示输入字符，最终运算出结果。

## **相关技术**

C语言：使用C语言作为程序的核心开发语言，负责实现计算器的基本逻辑与功能。

Glade：利用Glade设计计算器的GUI界面，快速构建图形化用户界面，减少手动编写GTK代码的工作量。

GTK+：使用GTK+库来实现计算器的图形界面，处理按钮点击事件和文本框显示等操作。

编译工具：利用GCC进行C语言程序的编译，并使用Makefile管理项目的构建过程。

## **需求分析**

该项目要求开发一个基本的计算器应用程序，具备以下主要功能：

完成基本的加、减、乘、除运算。

通过图形用户界面（GUI）与用户交互，用户点击相应按钮输入数字和运算符。

在文本框中动态显示用户输入的数字、运算符及结果，包括支持小数点输入和运算。

提供清除功能，允许用户清空输入的表达式。

## **设计过程**

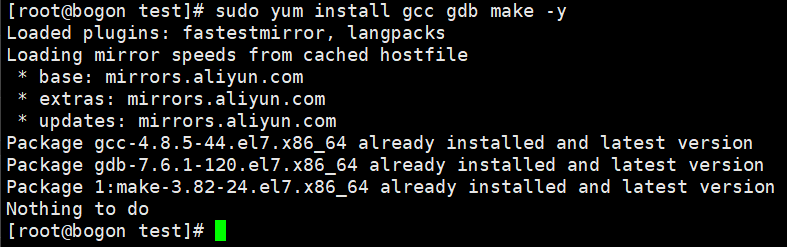
先使用glade创建一个计算器页面，页面布局如表格所示，创建一个main\_window窗口，加入一个表格，在表格中分别加入button和entry。

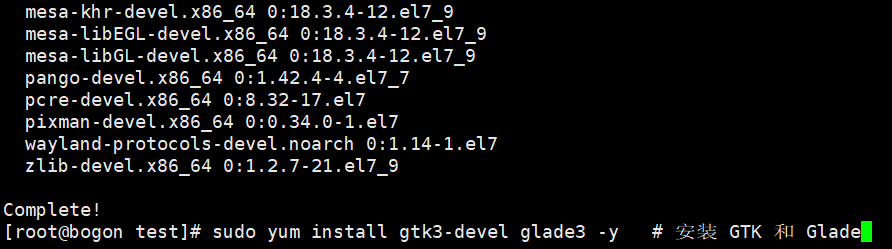
|  |  |  |  |
| --- | --- | --- | --- |
| Entry\_display:输出界面 | | | = |
| 1 | 2 | 3 | + |
| 4 | 5 | 6 | - |
| 7 | 8 | 9 | \* |
| c | 0 | . | / |

思路：鼠标点击相关按钮，输入正确的数字或者运算符号，最后点击等号完成运算，并且将运算结果输出到文本框中。支持小数点后两位的运算，点击C按钮清零界面。

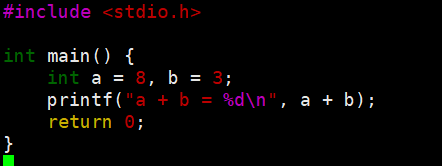
## **实现过程**

安装C语言和开发工具包，安装Glade+GTK+图形界面开发技术。



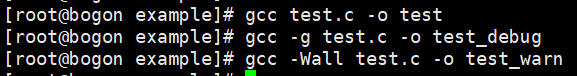


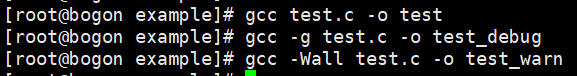
写一个简单C语言程序，输出两个变量相加之和。



使用gcc命令编译：



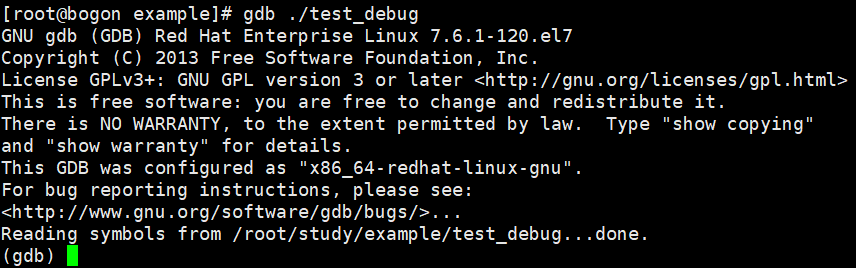


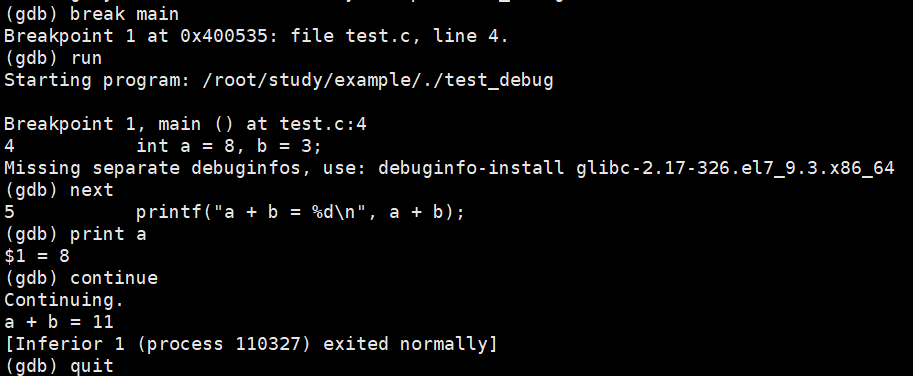


运行程序：



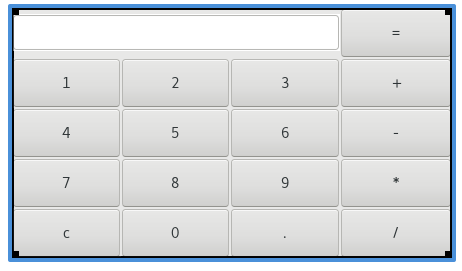
用gdb命令调试编译后的c语言程序：





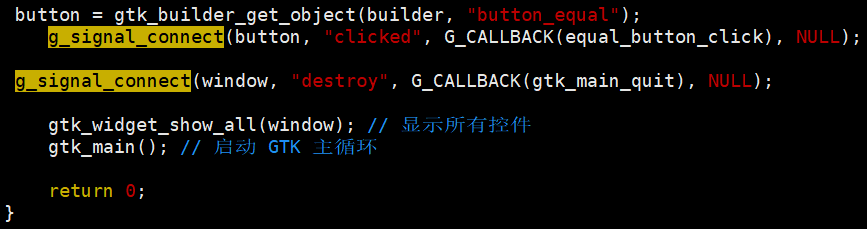
至此，正式进入编写一个计算器的C程序项目。

创建一个glade图形界面：



编写main.c文件:

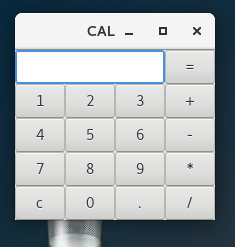




编写完成后编译代码，运行结果：

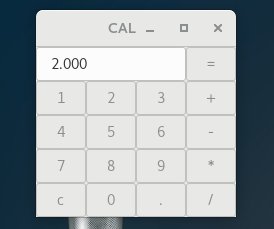






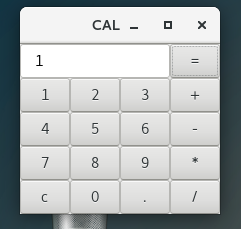
## **测试和总结**

测试功能1+1=2：



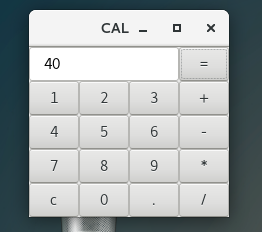
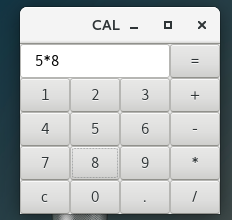
问题：不显示运算符，还有除了加和乘是正确的，其他运算符是错误的，修改代码。

计算5/5=1：

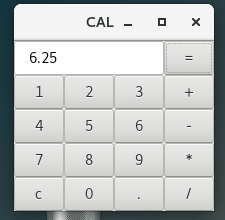
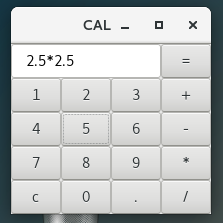


结果正确，但是不显示运算符，导致我无法确认我输入的原数字和运算符。

继续修改代码：显示运算符



测试小数点运算；



总结：本项目通过C语言、Glade、GTK+的组合，成功实现了一个功能简单、界面友好的计算器。通过图形界面与用户的交互，用户能够方便地进行数字输入和运算符选择，程序能够实时显示输入的数字和运算符，并执行准确的运算。通过小数点输入的支持，程序能够进行更为精确的计算。

项目代码：

#include <gtk/gtk.h>

#include <string.h>

#include <stdlib.h>

GtkWidget \*entry\_display;

void on\_button\_clicked(GtkButton \*button, gpointer user\_data) {

const gchar \*label = gtk\_button\_get\_label(button);

const gchar \*current = gtk\_entry\_get\_text(GTK\_ENTRY(entry\_display));

gchar \*new\_text = g\_strconcat(current, label, NULL);

gtk\_entry\_set\_text(GTK\_ENTRY(entry\_display), new\_text);

}

void on\_button\_clear\_clicked(GtkButton \*button, gpointer user\_data) {

gtk\_entry\_set\_text(GTK\_ENTRY(entry\_display), "");

}

void on\_button\_equal\_clicked(GtkButton \*button, gpointer user\_data) {

const gchar \*expr = gtk\_entry\_get\_text(GTK\_ENTRY(entry\_display));

double result = 0;

char op;

double a, b;

if (sscanf(expr, "%lf%c%lf", &a, &op, &b) == 3) {

switch (op) {

case '+': result = a + b; break;

case '-': result = a - b; break;

case '\*': result = a \* b; break;

case '/': result = b != 0 ? a / b : 0; break;

default: result = 0;

}

char buf[64];

snprintf(buf, sizeof(buf), "%g", result);

gtk\_entry\_set\_text(GTK\_ENTRY(entry\_display), buf);

}

}

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

GtkBuilder \*builder;

GtkWidget \*window;

GtkWidget \*button\_0, \*button\_1, \*button\_2, \*button\_3, \*button\_4, \*button\_5, \*button\_6, \*button\_7, \*button\_8, \*button\_9;

GtkWidget \*button\_plus, \*button\_minus, \*button\_multiply, \*button\_divide, \*button\_clear, \*button\_point, \*button\_equal;

gtk\_init(&argc, &argv);

builder = gtk\_builder\_new\_from\_file("/root/study/CAL/cal.glade");

gtk\_builder\_connect\_signals(builder, NULL);

window = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "main\_window"));

entry\_display = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "entry\_display"));

button\_0 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_0"));

button\_1 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_1"));

button\_2 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_2"));

button\_3 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_3"));

button\_4 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_4"));

button\_5 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_5"));

button\_6 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_6"));

button\_7 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_7"));

button\_8 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_8"));

button\_9 = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_9"));

button\_plus = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_plus"));

button\_minus = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_minus"));

button\_multiply = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_multiply"));

button\_divide = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_divide"));

button\_clear = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_clear"));

button\_point = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_point"));

button\_equal = GTK\_WIDGET(gtk\_builder\_get\_object(builder, "button\_equal"));

g\_signal\_connect(button\_0, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_1, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_2, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_3, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_4, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_5, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_6, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_7, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_8, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_9, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_plus, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_minus, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_multiply, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_divide, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_point, "clicked", G\_CALLBACK(on\_button\_clicked), NULL);

g\_signal\_connect(button\_clear, "clicked", G\_CALLBACK(on\_button\_clear\_clicked), NULL);

g\_signal\_connect(button\_equal, "clicked", G\_CALLBACK(on\_button\_equal\_clicked), NULL);

gtk\_widget\_show\_all(window);

gtk\_main();

return 0;

}