User Guide

正点原子 littleVGL 开发指南

lv_cont 容器

开发指南

正点原子

广州市星翼电子科技有限公司

修订历史

版本	日期	原因
V1.00	2020/05/01	第一次发布

lv_cont 容器

1. 介绍

所谓的容器就是一个载体,用来装东西的,在 littleVGL 中,可以用来存放各种各样的子对象,当子对象的数量越来越多时,子对象们在父容器中的排列方式就显得尤为重要,因此 lv_cont 容器就有一个专门的 Layout 布局属性来约束子对象们的摆放,layout 布局间隙是由样式来控制的,具体表现在 style.body.padding 样式属性上,lv_cont 容器除了 layout 这个重要特性外,还有一个 Auto fit 大小自动适应的特性.只要弄懂了 Layout 和 Auto fit 这俩个概念,lv_cont 容器的使用就特别简单了,所以请务必搞懂,因为后面还有许多复杂一点的控件是由 lv_cont 容器构成的.

2. lv_cont 的 API 接口

2.1 主要数据类型

2.1.1 布局数据类型

```
enum {
    LV_LAYOUT_OFF = 0,
    LV_LAYOUT_CENTER,
    LV_LAYOUT_COL_L,
    LV_LAYOUT_COL_M,
    LV_LAYOUT_COL_R,
    LV_LAYOUT_ROW_T,
    LV_LAYOUT_ROW_M,
    LV_LAYOUT_ROW_B,
    LV_LAYOUT_PRETTY,
    LV_LAYOUT_GRID,
    _LV_LAYOUT_NUM //这个值是无意义的,只是用来记录一下有多少种布局方式
};
typedef uint8_t lv_layout_t;
```

这个数据类型就是 lv_cont 容器的 Layout 布局特性,是重点内容,下面我们通过图示来一一介绍每种布局方式.

下面所有的图示中,假定 A,B,C 三个对象都属于 lv_cont 容器的子对象.

1) LV_LAYOUT_OFF

这个值是用来禁止布局功能的,当禁止之后,对于子对象在容器中的摆放位置,子对象必须得手动通过 lv_obj_set_pos(child,x,y);这样的接口来设置其具体的位置,如果没有禁止布局功能的话,就算子对象手动调用 lv_obj_set_pos(child,x,y);接口,也是无效的.

2) LV_LAYOUT_CENTER

把所有的子对象以从上到下的方式摆放到父容器的正中心,如下图所示:

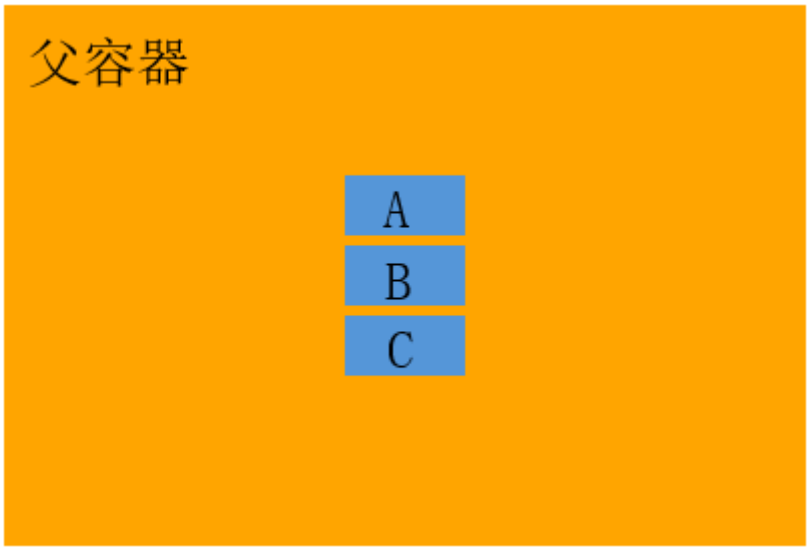

图 2.1.1.1 LV_LAYOUT_CENTER 布局效果

其中 A,B,C 子对象之间的垂直间距可以通过 lv_cont_style.body.padding.inner 样式值来修改

3) LV_LAYOUT_COL_L

从父容器的左上角开始,把所有的子对象以从上到下的方式进行摆放,如下图所示:

图 2.1.1.2 LV_LAYOUT_COL_L 布局效果

其中 A,B,C 子对象之间的垂直间距可以通过 lv_cont_style.body.padding.inner 样式值来修改,与父容器左边的距离可以通过 lv_cont_style.body.padding.left 样式值来修改,与父容器上边的距离可以通过 lv_cont_style.body.padding.top 样式值来修改

4) LV_LAYOUT_COL_M

从父容器的顶部中心点开始,把所有的子对象以从上到下的方式进行摆放,如下图所示:

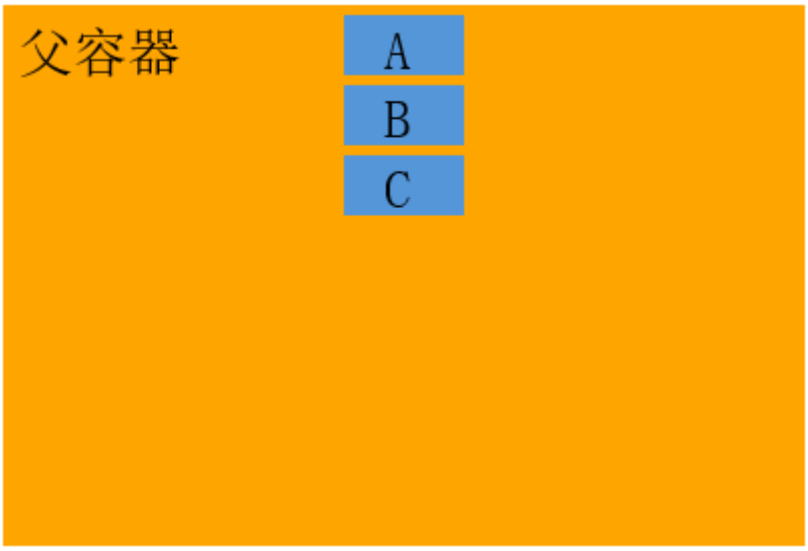

图 2.1.1.3 LV_LAYOUT_COL_M 布局效果

其中 A,B,C 子对象之间的垂直间距可以通过 lv_cont_style.body.padding.inner 样式值来修改,与父容器上边的距离可以通过 lv_cont_style.body.padding.top 样式值来修改

5) LV_LAYOUT_COL_R

从父容器的右上角开始,把所有的子对象以从上到下的方式进行摆放,如下图所示:

图 2.1.1.4 LV_LAYOUT_COL_R 布局效果

其中 A,B,C 子对象之间的垂直间距可以通过 lv_cont_style.body.padding.inner 样式值来修改,与父容器上边的距离可以通过 lv_cont_style.body.padding.top 样式值来修改

6) LV_LAYOUT_ROW_T

从父容器的左上角开始,把所有的子对象以从左到右的方式进行摆放,如下图所示:

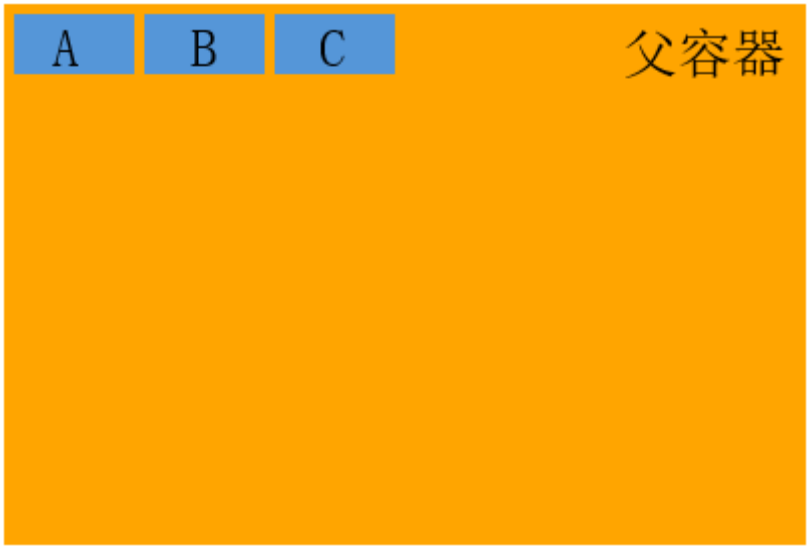

图 2.1.1.5 LV_LAYOUT_ROW_T 布局效果

其中 A,B,C 子对象之间的水平间距可以通过 lv_cont_style.body.padding.inner 样式值来修改,与父容器左边的距离可以通过 lv_cont_style.body.padding.left 样式值来修改,与父容器上边的距离可以通过 lv_cont_style.body.padding.top 样式值来修改

7) LV_LAYOUT_ROW_M

从父容器的左边中心点开始,把所有的子对象以从左到右的方式进行摆放,如下图所示:

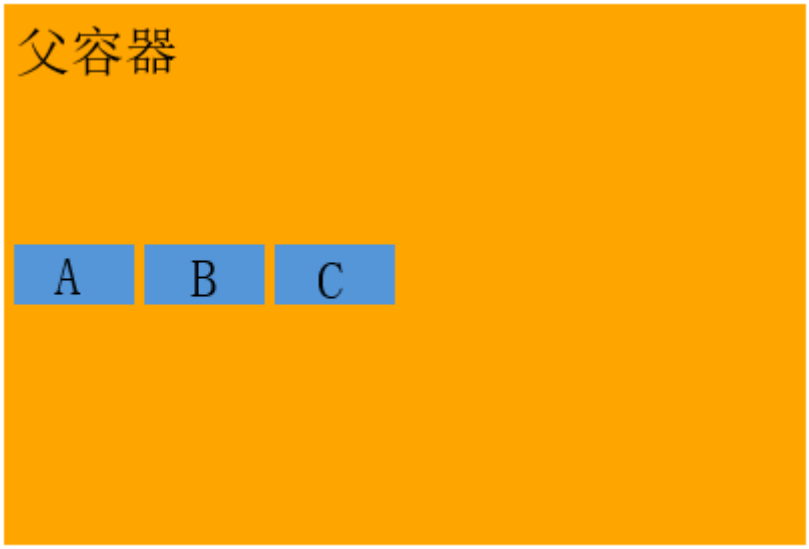

图 2.1.1.6 LV_LAYOUT_ROW_M 布局效果

其中 A,B,C 子对象之间的水平间距可以通过 lv_cont_style.body.padding.inner 样式值来修改,与父容器左边的距离可以通过 lv_cont_style.body.padding.left 样式值来修改

8) LV_LAYOUT_ROW_B

从父容器的左下角开始,把所有的子对象以从左到右的方式进行摆放,如下图所示:

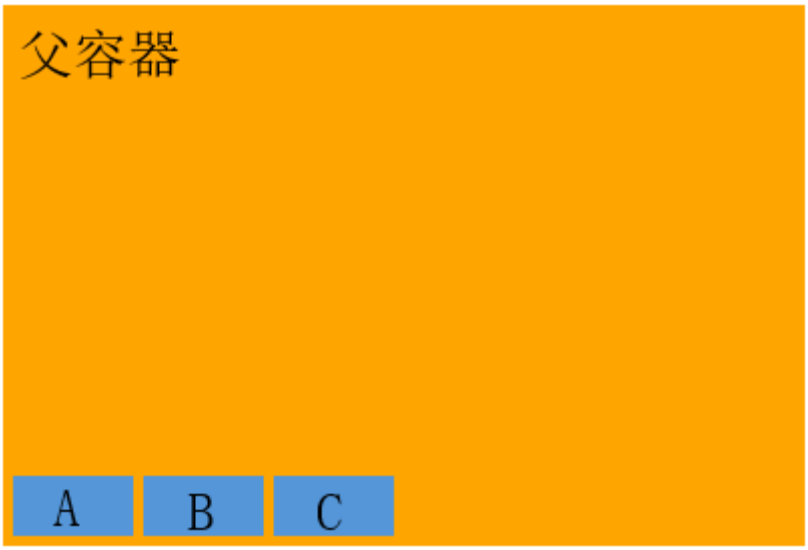

图 2.1.1.7 LV_LAYOUT_ROW_B 布局效果

其中 A,B,C 子对象之间的水平间距可以通过 lv_cont_style.body.padding.inner 样式值来修改,与父容器下边的距离可以通过 lv_cont_style.body.padding.bottom 样式值来修改

9) LV_LAYOUT_PRETTY

从父容器的左上角开始,把所有的子对象以从左到右,从上到下的方式进行摆放,并且保证每一行都均匀分布,如下图所示:

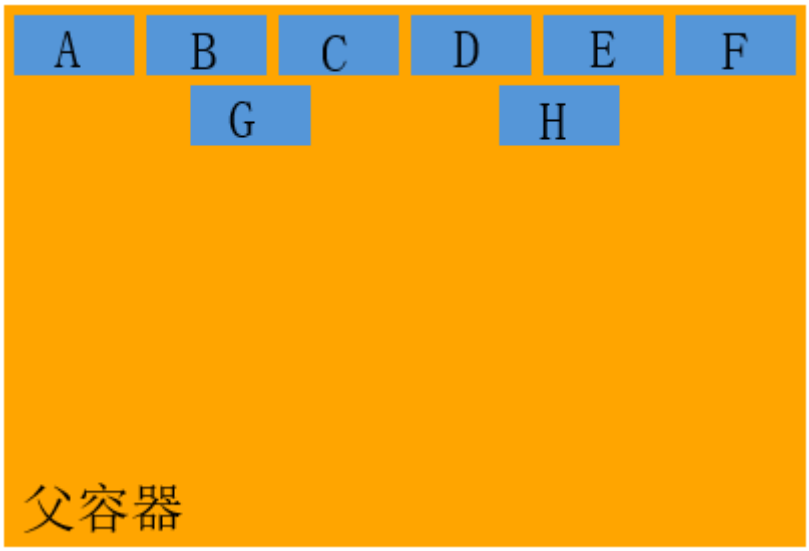

图 2.1.1.8 LV_LAYOUT_PRETTY 布局效果

其中子对象之间的水平和垂直距离都是通过 lv_cont_style.body.padding.inner 样式值来

修改的,另外通过 lv_cont_style.body.padding.left/top/right/bottom 等样式值来修改其与父容器相应边的距离

10) LV_LAYOUT_GRID

LV_LAYOUT_GRID 和 LV_LAYOUT_PRETTY 是很相似的,但是 LV_LAYOUT_GRID 不要求在每一行均匀分布,而是按照 lv_cont_style.body.padding.inner 指定的间隙依次摆放

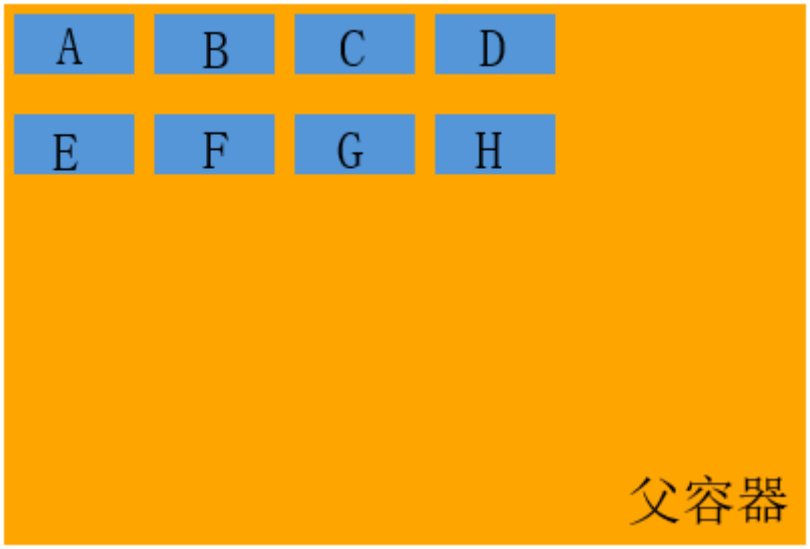

图 2.1.1.9 LV_LAYOUT_GRID 布局效果

2.1.2 大小自适应数据类型

```
enum {
    LV_FIT_NONE,
    LV_FIT_TIGHT,
    LV_FIT_FLOOD,
    LV_FIT_FILL,
    _LV_FIT_NUM //这个值是无意义的,只是用来记录一下有多少种自适应方式
};
typedef uint8_t lv_fit_t;
```

这个数据类型就是 lv_cont 容器的大小自适应特性,所谓的自适应就是指 lv_cont 容器的大小可以根据它的子对象们或父对象的大小来自动调整,下面详细说一下每一个值的具体作用

1) LV_FIT_NONE

禁止 lv_cont 容器使用自适应功能,当禁止之后,那么 lv_cont 容器只能通过手动调用 lv_obj_set_size 接口来设置容器的高和宽了,如果没有禁止的话,调用 lv_obj_set_size 接口是无效的

2) LV_FIT_TIGHT

包裹住所有的子对象,也就是说 lv_cont 容器的大小是随着子对象们的总大小变化而变化的,子对象与父容器四边的距离可以通过 lv_cont_style.body.padding.left/top/right/bottom 样式值来相应的设置

3) LV_FIT_FLOOD

lv_cont 容器平铺它父对象的整个空间,lv_cont 容器与其父对象四边的距离可以通过其父对象样式中的 body.padding.left/top/right/bottom 值来相应的修改

4) LV_FIT_FILL

当 lv_cont 容器比其父对象小时,采用 LV_FIT_FLOOD 方式,当比其父对象大时,采用 LV_FIT_TIGHT 方式

2.2 API 接口

2.2.1 创建容器

```
lv_obj_t * lv_cont_create(lv_obj_t * par, const lv_obj_t * copy);
```

参数:
par: 指向父对象
copy: 此参数可选,表示创建新对象时,把 copy 对象上的属性值复制过来

返回值:
返回新创建出来的容器对象,如果为 NULL 的话,说明堆空间不足了

2.2.2 设置容器的布局方式

```
void lv_cont_set_layout(lv_obj_t * cont, lv_layout_t layout);
```

参数:
cont: 容器对象
layout: 布局方式,请详看 2.1.1 布局数据类型那一小章节

2.2.3 设置容器 4 个方向上的自适应方式

```
void lv_cont_set_fit4(lv_obj_t * cont, lv_fit_t left, lv_fit_t right, lv_fit_t top, lv_fit_t bottom);
```

参数:
cont: 容器对象
left: 左边的自适应方式,请详看 2.1.2 大小自适应数据类型那一小章节
right: 右边的自适应方式
top: 上边的自适应方式
bottom: 下边的自适应方式

比如说,我设置容器 A 的左边为 LV_FIT_NONE 方式,设置容器 A 的右边为 LV_FIT_TIGHT 方式,假如现在子对象在宽度上增大了,那么容器 A 的宽度也会跟着变大,但是容器 A 只会朝着右边这个方向来把宽度增大的,因为容器 A 的左边是 LV_FIT_NONE 方式,即左边是无自适应的,通过这个例子,我相信大家能够举一反三

2.2.4 设置容器水平和垂直方向上的自适应方式

```
static inline void lv_cont_set_fit2(lv_obj_t * cont, lv_fit_t hor, lv_fit_t ver);
```

参数:
cont: 容器对象
hor: 水平方向上的自适应方式
ver: 垂直方向上的自适应方式

这个接口的内部实现原理很简单,没啥好解释的,直接看下图吧:

```
static inline void lv_cont_set_fit2(lv_obj_t * cont, lv_fit_t hor, lv_fit_t ver)
{
    lv_cont_set_fit4(cont, hor, hor, ver, ver);
}
```

图 2.2.4.1 内部实现原理

2.2.5 设置容器的自适应方式

```
static inline void lv_cont_set_fit(lv_obj_t * cont, lv_fit_t fit);
```

参数:
cont: 容器对象
fit: 四个方向上的共同自适应方式

这个接口的内部实现原理也很简单,没啥好解释的,直接看下图吧:

```
static inline void lv_cont_set_fit(lv_obj_t * cont, lv_fit_t fit)
{
    lv_cont_set_fit4(cont, fit, fit, fit, fit);
}
```

图 2.2.5.2 内部实现原理

2.2.6 设置容器的样式

```
    static inline void lv_cont_set_style(lv_obj_t * cont, lv_cont_style_t type, const lv_style_t *
style);
```

参数:

cont: 容器对象

type: 设置容器上的哪部分样式,目前只有 LV_CONT_STYLE_MAIN 一个可选值

style: 样式

3.例程设计

3.1 功能简介

创建一个 cont 容器和一个 tip 标签,tip 提示标签主要是用来显示 cont 容器当前的自适应方式和布局方式的,在创建容器时,默认先给容器一个 LV_LAYOUT_CENTER 的布局方式和一个 LV_FIT_TIGHT 的自适应方式,然后当用户每按下一次 KEY0 按键时,就会往 cont 容器中添加一个序号依次增加的标签子对象,当用户每按下一次 KEY1 按键时,就会把 cont 容器最后添加的一个子对象给删除掉,此时通过 KEY0 和 KEY1 键的相互操作,你可以发现容器的高度会相应的增加或者减少,这就演示了容器大小的自适应能力,然后当用户每按下一次 KEY2 键时,会切换一次 cont 容器的布局方式,此时再接着利用 KEY0 和 KEY1 键的配合,你可以看到 cont 容器的每一种布局效果

3.2 硬件设计

本例程所用到的硬件有:

1) 液晶屏
2) KEY0,KEY1,KEY2 按键

3.3 软件设计

在 GUI_APP 目录下创建 lv_cont_test.c 和 lv_ cont_test.h 俩个文件,其中 lv_ cont _test.c 文件的内容如下:

```
#include "lv_cont_test.h"
#include "lvgl.h"
#include "key.h"
#include <stdio.h>

lv_obj_t* cont;
lv_obj_t* tip;
lv_layout_t layout = LV_LAYOUT_CENTER;//起始的布局方式
lv_fit_t fit = LV_FIT_TIGHT;//起始的自适应方式
u8 child_no = 1;
//布局提示内容
const char* const LAYOUT_STR[] = {
    "LV_LAYOUT_OFF","LV_LAYOUT_CENTER","LV_LAYOUT_COL_L","LV_LAYOUT_COL_M","LV_LAYOUT_COL_R",
    "LV_LAYOUT_ROW_T","LV_LAYOUT_ROW_M","LV_LAYOUT_ROW_B","LV_LAYOUT_PRETTY","LV_LAYOUT_GRID"
```

```
    };
    //自适应提示内容
    const char* const FIT_STR[] = {"LV_FIT_NONE","LV_FIT_TIGHT","LV_FIT_FLOOD","LV_FIT_
FILL"};

    //将当前的布局方式和自适应方式通过 label 显示出来
    void info_tip()
    {
        char buff[80];
        sprintf(buff,"#ff0000 Fit:#%s\n#ff0000 Layout:#%s",FIT_STR[fit],LAYOUT_STR[layout]);
        lv_label_set_text(tip,buff);
        lv_obj_align(tip,NULL,LV_ALIGN_IN_BOTTOM_MID,0,-40);
    }


    //例程入口函数
    void lv_cont_test_start()
    {
        lv_obj_t* scr = lv_scr_act();//获取当前活跃的屏幕对象

        //1.创建容器并初始化
        cont = lv_cont_create(scr,NULL);//创建容器
        lv_obj_set_pos(cont,20,20);//设置坐标
        //设置 cont 容器四个方向上的自适应方式,我们让容器的 left 左边和 top 上边无自适
        //应,即保持不动,因为容器的坐标为(20,20),数值比较小,如果左边和上边不设置为
        //无自适应的话,当子对象数量增多时,容器因高或宽自动变大,就很容易碰到屏幕的
        //左边缘或上边缘
        lv_cont_set_fit4(cont,LV_FIT_NONE,fit,LV_FIT_NONE,fit);
        //先设置容器布局方式
        lv_cont_set_layout(cont,layout);

        static lv_style_t cont_style;
        lv_style_copy(&cont_style,&lv_style_plain_color);//样式拷贝
        //设置纯红色的背景
        cont_style.body.main_color = LV_COLOR_RED;
        cont_style.body.grad_color = LV_COLOR_RED;
        //设置容器的 4 个内边距
        cont_style.body.padding.top = 10;
        cont_style.body.padding.left = 10;
        cont_style.body.padding.right = 10;
        cont_style.body.padding.bottom = 10;
        cont_style.body.padding.inner = 10;//设置容器中子对象之间的间隙

        lv_cont_set_style(cont,LV_CONT_STYLE_MAIN,&cont_style);//给容器设置样式
```

```
    //2.创建一个提示用的 label
    tip = lv_label_create(scr,NULL);
    lv_label_set_recolor(tip,true);//使能颜色重绘
    info_tip();
}

//在容器中添加子对象
void cont_add_child()
{
    char no[4];//记录子对象的编号
    lv_obj_t* child;

    child = lv_label_create(cont,NULL);
    lv_label_set_long_mode(child,LV_LABEL_LONG_CROP);//设置长文本模式
    lv_label_set_body_draw(child,true);//使能背景绘制
    lv_obj_set_size(child,50,50);//设置固定的大小
    lv_label_set_style(child,LV_LABEL_STYLE_MAIN,&lv_style_plain_color);//设置样式
    lv_label_set_align(child,LV_LABEL_ALIGN_CENTER);//设置文本居中对齐
    sprintf(no,"%d",child_no++);
    lv_label_set_text(child,no);
}

//删除容器最后添加的一个子对象
void cont_del_child()
{
    lv_obj_t* child = lv_obj_get_child(cont,NULL);
    if(child)//如果为 NULL 的话,说明没有子对象了
    {
        lv_obj_del(child);//删除掉此子对象
        child_no--;
    }
}

//按键处理
void key_handler()
{
    u8 key = KEY_Scan(0);

    if(key==KEY0_PRES)
    {
        //在 cont 容器中添加子对象
        cont_add_child();
    }else if(key==KEY1_PRES)
```

```
        {
            //删除掉 cont 容器中最后被添加进来的子对象
            cont_del_child();
        }else if(key==KEY2_PRES)
        {
            //为了让布局切换看得更明显,我们先把容器的自适应功能给关掉,然后给容器
            //一个固定的大小
            if(fit!=LV_FIT_NONE)//只需要执行一次就可以了
            {
                fit = LV_FIT_NONE;
                lv_cont_set_fit(cont,fit);
                lv_obj_set_size(cont,200,200);
            }
            //循环切换容器的布局
            layout++;
            if(layout==_LV_LAYOUT_NUM)
                layout = LV_LAYOUT_CENTER;
            lv_cont_set_layout(cont,layout);
            info_tip();//信息提示
        }
    }
```

3.4 下载验证

把代码下载进去之后,初始的界面效果如下:

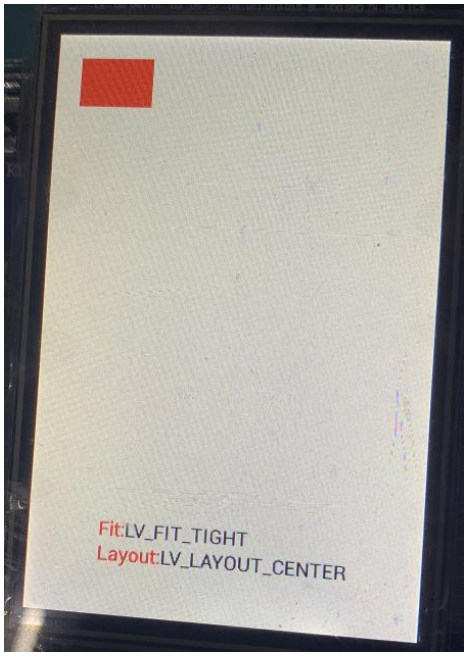

图 3.4.1 初始界面效果

然后按 KEY0 键三次,往 cont 容器里面添加了 3 个子对象之后的界面效果如下:

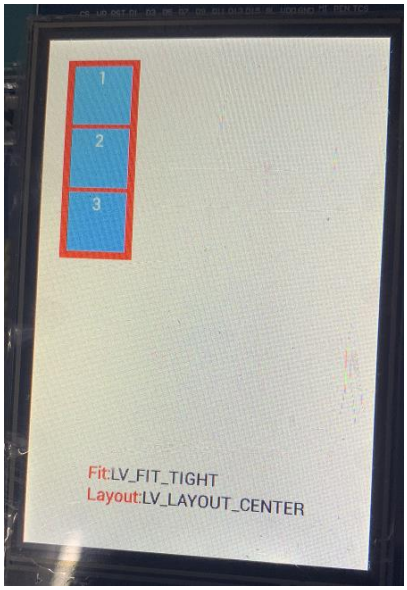

图 3.4.2 添加了 3 个子对象后的界面效果

往容器中添加子对象后,可以发现容器的高度明显增加了,然后我们再按一次 KEY1 键来删除掉 3 号子对象,界面效果如下:

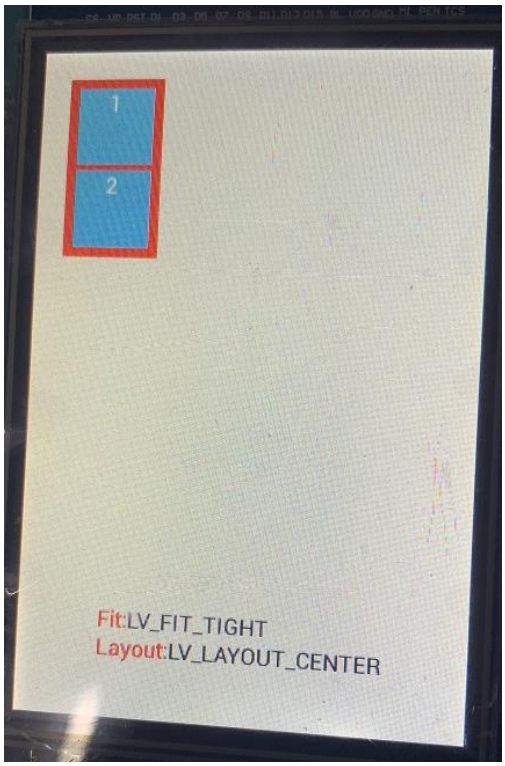

图 3.4.3 删除 3 号子对象后的界面效果

删除一个子对象后,可以发现容器的高度明显减少了,再接着我们按下 KEY2 键来切换容器的布局方式,界面效果如下:

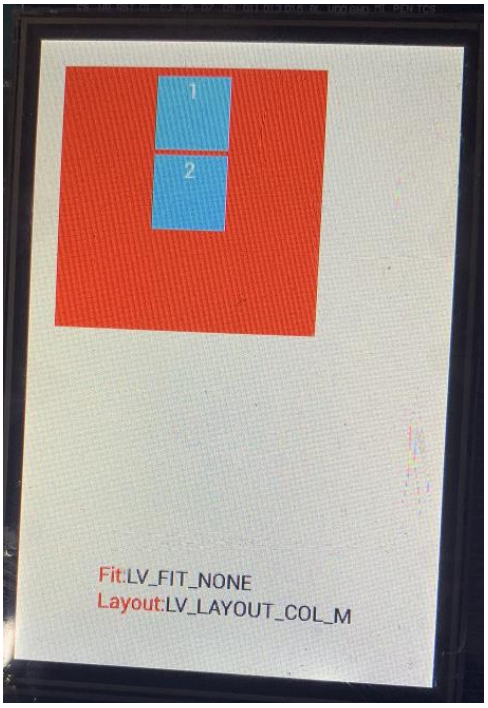

图 3.4.4 LV_LAYOUT_COL_M 布局方式

4. 资料下载

正点原子公司名称 ：广州市星翼电子科技有限公司

LittleVGL 资料连接 ：www.openedv.com/thread-309664-1-1.html

原子哥在线教学平台：www.yuanzige.com

正点原子淘宝店铺 ：https://openedv.taobao.com

正点原子官方网站 ：www.alientek.com

正点原子 B 站视频 ：https://space.bilibili.com/394620890

电话：020-38271790 传真：020-36773971

请下载原子哥 APP，数千讲视频免费学习，更快更流畅。

请关注正点原子公众号，资料发布更新我们会通知。

扫码下载“原子哥”APP

扫码关注正点原子公众号