User Guide

正点原子 littleVGL 开发指南

lv_page 页面

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_page 页面

1. 介绍

lv_page 页面控件是由俩个 lv_cont 容器控件构成的,其中一个容器作为 lv_page 页面控件的背景层,另外一个容器作为 lv_page 页面控件的载体,此载体可以用来存放其他任何类型的子对象,算是发挥了它容器自身的本色,如果此载体的大小超过了 lv_page 控件自身的大小,那么此载体就可以在水平或者垂直方向上进行滚动了,通过滚动操作,就能实现在有限的可视区域内跟更多的子对象进行交互操作了,为了方便后面理解,我们把此容器或者此载体简称为 scrl ,其实就是 Scroll 单词的缩写.

当进行滚动操作时,lv_page 页面控件上可以出现可选的滚动条进行示意,根据不同的运用场景,此滚动条总共有 6 种模式,我们可以通过 lv_page_set_sb_mode(page, SB_MODE)接口来设置滚动条的模式,如果我们想要让 lv_page 页面控件中的某一个子对象处于可见状态时,除了用户滚动操作可以实现外,我们还可以调用 lv_page_focus(page, child, LV_ANIM_ON/OFF)接口来使此 child 子对象获得聚焦,接着 lv_page 页面控件会自动将此 child 子对象滚动到可见状态,在这个过程中,我们可以选择是否具有动画效果,动画时长可以通过 lv_page_set_anim_time(page, anim_time)接口来进行设置,除此之外,我们还可以通过 lv_page_scroll_hor(page, dist)和 lv_page_scroll_ver(page, dist)接口来更为具体的控制页面的滚动,一个是控制水平方向的,另外一个是控制垂直方向的.

最后我们来说一下 lv_page 页面控件的边缘半圆弧动画效果,当我们通过 lv_page_set_edge_flash(page, en)接口使能此效果之后,如果用户将页面滚动到了某边缘时,那么此边缘上就会出现一个半圆弧动画效果,主要是提醒用户你已经滑到边缘了,还算是一个比较人性化,炫酷的小功能.

图 1.1 lv_page 的构成

2. lv_page 的 API 接口

2.1 主要数据类型

2.1.1 滚动条模式数据类型

```
enum {
    LV_SB_MODE_OFF      = 0x0,
    LV_SB_MODE_ON       = 0x1,
    LV_SB_MODE_DRAG     = 0x2,
    LV_SB_MODE_AUTO     = 0x3,
    LV_SB_MODE_HIDE     = 0x4,
    LV_SB_MODE_UNHIDE = 0x5,
};
typedef uint8_t lv_sb_mode_t;
```

我们前面说过了 lv_page 的滚动条具有 6 种模式,但主要的就是前面 4 种,最后面的 2 种没啥用处.

LV_SB_MODE_OFF: 不显示任何滚动条
LV_SB_MODE_ON: 不论如何,都要把垂直和水平滚动条显示出来
LV_SB_MODE_DRAG: 当滚动页面时才显示出相应的滚动条,否则不显示
LV_SB_MODE_AUTO: 自动模式,当载体容器的大小超过 lv_page 页面的大小时,就会显示出相应的滚动条
LV_SB_MODE_HIDE: 把所有的滚动条都给隐藏了,跟 LV_SB_MODE_OFF 的作用差不多
LV_SB_MODE_UNHIDE: 把隐藏了的滚动条再次显示出来

2.1.2 页面边缘数据类型

```
enum {
    LV_PAGE_EDGE_LEFT = 0x1,
    LV_PAGE_EDGE_TOP = 0x2,
    LV_PAGE_EDGE_RIGHT = 0x4,
    LV_PAGE_EDGE_BOTTOM = 0x8
};
typedef uint8_t lv_calendar_style_t;
```

此数据类型主要是在 bool lv_page_on_edge(lv_obj_t * page, lv_page_edge_t edge);接口中被使用到了,用来判断页面是否到达了某边缘

2.1.3 页面样式数据类型

```
enum {
        LV_PAGE_STYLE_BG,
        LV_PAGE_STYLE_SCRL,
        LV_PAGE_STYLE_SB,
        LV_PAGE_STYLE_EDGE_FLASH,
};
typedef uint8_t lv_page_style_t;
```

LV_PAGE_STYLE_BG: 用来修饰页面背景的,使用样式中的 body 字段,默认值为 lv_style_pretty_color

LV_PAGE_STYLE_SCRL: 用来修饰载体容器的,使用样式中的 body 字段,默认值为 lv_style_pretty

LV_PAGE_STYLE_SB: 用来修饰水平和垂直滚动条的,使用样式中的 body 字段,默认值为 lv_style_pretty_color,其中 body.padding.right 是用来控制垂直滚动条与页面右边缘的距离,当此值为正数时,是在页面内部,当为负值时,是在页面外部,而 body.padding.bottom 是用来控制水平滚动条与页面底边缘的距离,当此值为正数时,是在页面内部,当为负值时,是在页面外部,而 body.padding.inner 是用来设置滚动条的宽度

LV_PAGE_STYLE_EDGE_FLASH: 用来修饰边缘半圆弧动画效果的,一般只用到里面的 body.main_color, body.grad_color, body.opa 等样式字段,对于上边缘只用 grad_color 来设置半圆弧的颜色,对于下边缘只用 main_color 来设置半圆弧的颜色,而对于左和右边缘,main_color 和 grad_color 都会用到

2.2 API 接口

2.2.1 创建对象

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

参数:
par: 父对象
copy: 拷贝的对象,如果无拷贝的话,传 NULL 值

返回值:
返回创建出来的对象,如果返回 NULL 的话,说明堆空间不够了

2.2.2 清空页面内的所有子对象

```
void lv_page_clean(lv_obj_t * obj);
```

参数:
obj: 页面对象

其实就是把页面载体容器内的所有子对象全部删除掉

2.2.3 获取页面的载体容器对象

```
lv_obj_t * lv_page_get_scrl(const lv_obj_t * page);
```

参数:
page: 页面对象

返回值:
返回 page 页面内部的载体容器对象

拿到此载体容器对象之后,我们就可以拿 lv_cont 容器专有的 API 接口来操作其特性了

2.2.4 设置滚动条的模式

```
void lv_page_set_sb_mode(lv_obj_t * page, lv_sb_mode_t sb_mode);
```

参数:
page: 页面对象
sb_mode: 滚动条的模式,有如下 6 个可选值

LV_SB_MODE_OFF: 不显示任何滚动条
LV_SB_MODE_ON: 不论如何,都要把垂直和水平滚动条显示出来
LV_SB_MODE_DRAG: 当滚动页面时才显示出相应的滚动条,否则不显示
LV_SB_MODE_AUTO: 自动模式,当载体容器的大小超过 lv_page 页面的大小时,就会显示出相应的滚动条
LV_SB_MODE_HIDE: 把所有的滚动条都给隐藏了,跟 LV_SB_MODE_OFF 的作用差不多
LV_SB_MODE_UNHIDE: 把隐藏了的滚动条再次显示出来如果不设置的话,则默认就是 LV_SB_MODE_AUTO 模式

2.2.5 设置动画时长

```
void lv_page_set_anim_time(lv_obj_t * page, uint16_t anim_time);
```

参数:
page: 页面对象
anim_time: 动画时长,单位 ms

2.2.6 是否使能页面的滚动特性传递

```
void lv_page_set_scroll_propagation(lv_obj_t * page, bool en);
```

参数:
page: 页面对象
en: 是否使能滚动特性传递

首先说一下,此 API 接口不是很重要,基本用不到,我给大家稍微讲解一下,了解即可,假如当 A 页面对象内部含有一个 B 页面子对象时,而且 B 页面子对象的大小超过了 A 页面的大小,那么当我们想滚动 A 页面时,通常情况下是没办法实现的,因为 B 页面子对象已经把 A 页面给填充满了,A 页面已经没有空间来提供滚动操作了,所有的滚动操作都会作用在 B 页面子对象上,但是如果我们给 B 页面使能了滚动特性传递的话,即如下代码:

```
lv_page_set_scroll_propagation(B, true);
```

那么 B 页面会将它接受到的滚动操作传递给它的父对象,从而使 A 页面能够接受滚动操作了

2.2.7 是否使能边缘半圆弧动画效果

```
void lv_page_set_edge_flash(lv_obj_t * page, bool en);
```

参数:
page: 页面对象
en: true 代表使能,false 代表不使能

当使能之后,用户把页面滚动到自身某边缘时,此边缘上就会出现一个半圆弧的动画效果,主要是提示用户已经划到底,配合 LV_PAGE_STYLE_EDGE_FLASH 样式,可以给用户带来友好的体验,给出一个简单例子(只给出关键代码):

```
lv_style_copy(&edge_flash_style,&lv_style_plain_color);
edge_flash_style.body.main_color = LV_COLOR_RED;//红色
edge_flash_style.body.grad_color = LV_COLOR_GREEN;//绿色
edge_flash_style.body.opa = 150;//透明度
lv_obj_t * page1 = lv_page_create(scr,NULL);//创建 page1 页面
//设置 page1 的样式
lv_page_set_style(page1,LV_PAGE_STYLE_EDGE_FLASH,&edge_flash_style);
lv_page_set_edge_flash(page1,true);//使能 page1 的边缘半圆弧动画特效
```

然后四个边缘的演示效果分别如下:

图 2.2.7.1 上边缘

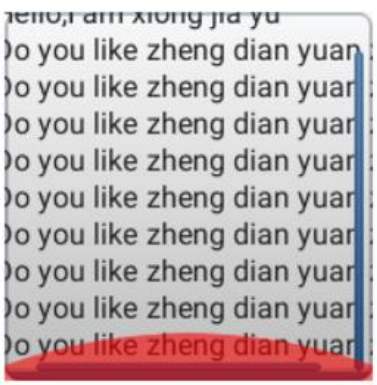

图 2.2.7.2 下边缘

图 2.2.7.3 左边缘

图 2.2.7.4 右边缘

从上面的效果我们可以得出上边缘只会用到 grad_color 颜色,而下边缘只会用到 main_color 颜色,而左和右边缘会把 main_color 和 grad_color 颜色都用到

2.2.8 设置载体容器的布局方式

```
static inline void lv_page_set_scrl_layout(lv_obj_t * page, lv_layout_t layout)
```

参数:

page: 页面对象

layout: 容器的布局方式,我们在 lv_cont 容器章节已经介绍过了

对于已经学习过 lv_cont 容器章节的朋友来说,,此 API 接口的实现机制其实特别简单,如下图所示:

```
229 static inline void lv_page_set_scrl_layout(lv_obj_t * page, lv_layout_t layout)
230 {
231     lv_cont_set_layout(lv_page_get_scrl(page), layout);
232 }
```

图 2.2.8.1 此接口的内部实现原理

2.2.9 设置样式

```
void lv_page_set_style(lv_obj_t * page, lv_page_style_t type, const lv_style_t * style);
```

参数:
page: 页面对象
type: 设置那部分的样式,目前有如下 4 个可选值:
LV_PAGE_STYLE_BG: 修饰背景的
LV_PAGE_STYLE_SCRL: 修饰载体容器的
LV_PAGE_STYLE_SB: 修饰滚动条的
LV_PAGE_STYLE_EDGE_FLASH: 修饰边缘半圆弧动画效果的
style: 样式

2.2.10 判断是否滚动到了某边缘

```
bool lv_page_on_edge(lv_obj_t * page, lv_page_edge_t edge);
```

参数:
page: 页面对象
edge: 哪些边缘,有如下 4 个可选值:
LV_PAGE_EDGE_LEFT,
LV_PAGE_EDGE_TOP,
LV_PAGE_EDGE_RIGHT,
LV_PAGE_EDGE_BOTTOM
这些值之间是可以进行位或操作的

返回值:
如果已经滚动到了 edge 参数所指定的边缘,则返回 true,否则返回 false

2.2.11 使某个子对象获得聚焦

```
void lv_page_focus(lv_obj_t * page, const lv_obj_t * obj, lv_anim_enable_t anim_en);
```

参数:
page: 页面对象
obj: page 页面内的子对象
anim_en: 在获得聚焦的过程中,是否开启动画效果,有如下 2 个可选值
LV_ANIM_ON:开启动画效果
LV_ANIM_OFF:不开启动画效果

所谓的获得聚焦,说得直白一点,就是让 obj 子对象在 page 页面内处于可见状态

2.2.12 让页面在水平方向上滚动指定距离

```
void lv_page_scroll_hor(lv_obj_t * page, lv_coord_t dist);
```

参数:
page: 页面对象
dist: 要滚动的距离,单位为像素,当此值小于 0 时,是往右边滚动,大于 0 时,是往左边滚动

2.2.13 让页面在垂直方向上滚动指定距离

```
void lv_page_scroll_ver(lv_obj_t * page, lv_coord_t dist);
```

参数:
page: 页面对象
dist: 要滚动的距离,单位为像素,当此值小于 0 时,是往下边滚动,大于 0 时,是往上边滚动

2.2.14 获取页面可填充区域的宽度

```
lv_coord_t lv_page_get_fit_width(lv_obj_t * page);
```

参数:
page: 页面对象

返回值:
返回页面可填充区域的宽度

所谓的可填充区域就是指页面中实际能够用来存放子对象的空间,它的宽度会比页面的宽度小一点,因为页面的宽度减去左右内边距之后就是等于它的宽度了,用公式表示如下:
fit_width = lv_obj_get_width(page) - bg_style->body.padding.left - bg_style->body.padding.right -scrl_style->body.padding.left - scrl_style->body.padding.right;其中 bg_style 是背景容器的样式,scrl_style 是载体容器的样式

图 2.2.14.1 填充区域示意图

2.2.15 获取页面可填充区域的高度

```
lv_coord_t lv_page_get_fit_height(lv_obj_t * page);
```

参数:
page: 页面对象

返回值:
返回页面可填充区域的高度

和 lv_page_get_fit_width 接口的用途是差不多的

2.2.16 备注

还有几个 get 获取类型的 API 接口我这里就不列举出来了,比较简单的

3.例程设计

3.1 功能简介

创建 2 种样式来分别修饰页面的滚动条和边缘半圆弧效果,然后创建一个 page1 页面,往其里面添加一个 label1 标签子对象和一个 btn1 按钮子对象,当点击 btn1 子对象时,会清空 page1 页面里的所有子对象,当按下 KEY0 按键时,会让 btn1 按钮获得聚焦而处于可见状态,当按下 KEY1 按键时,会让 page1 页面向下滚动,当按下 KEY2 按键时,会让 page1 页面向右滚动,当按下 WK_UP/KEY_UP 按键时,会来回切换页面的滚动条模式

3.2 硬件设计

本例程所用到的硬件有:

1) 液晶屏
2) KEY0,KEY1,KEY2,WK_UP/KEY_UP 按键

3.3 软件设计

在 GUI_APP 目录下创建 lv_page_test.c 和 lv_page_test.h 俩个文件,其中 lv_page_test.c 文件的内容如下:

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


lv_style_t sb_style;
lv_style_t edge_flash_style;
lv_obj_t * page1;
lv_obj_t * btn1;
lv_sb_mode_t sb_mode = LV_SB_MODE_AUTO;

//事件回调函数
void event_handler(lv_obj_t * obj,lv_event_t event)
{
  if(event==LV_EVENT_RELEASED)
  {
      lv_page_clean(page1);//清空内部的所有子对象
  }
```

```
}


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

    //1.创建 2 个样式
    //1.1  创建滚动条的样式
    lv_style_copy(&sb_style,&lv_style_plain);
    sb_style.body.main_color = LV_COLOR_BLACK;
    sb_style.body.grad_color = LV_COLOR_BLACK;
    sb_style.body.border.color = LV_COLOR_WHITE;
    sb_style.body.border.width = 1;
    sb_style.body.radius = LV_RADIUS_CIRCLE;
    sb_style.body.opa = LV_OPA_60;
    sb_style.body.padding.right = 3;//垂直滚动条的宽度
    sb_style.body.padding.bottom = 3;//水平滚动条的宽度
    sb_style.body.padding.inner = 8;//滚动条距页面边框的距离

    //1.2  创建边缘半圆弧样式
    lv_style_copy(&edge_flash_style,&lv_style_plain);
    edge_flash_style.body.main_color = LV_COLOR_GRAY;
    edge_flash_style.body.grad_color = LV_COLOR_GRAY;
    edge_flash_style.body.opa = 150;//透明度

    //2.创建页面
    //2.1  创建 page1 页面对象
    page1 = lv_page_create(scr,NULL);
    lv_obj_set_size(page1, 200, 200);//设置页面的大小
    lv_obj_align(page1, NULL, LV_ALIGN_CENTER,0,0);//与屏幕居中对齐
    //LV_SB_MODE_AUTO 是默认的滚动条模式
    lv_page_set_sb_mode(page1,sb_mode);
    lv_page_set_edge_flash(page1,true);//使能边缘半圆弧动画效果
    lv_page_set_style(page1,LV_PAGE_STYLE_SB,&sb_style);//设置滚动条的样式

    //设置边缘半圆弧样式
    lv_page_set_style(page1,LV_PAGE_STYLE_EDGE_FLASH,&edge_flash_style);

    //2.2  往 page1 页面中添加一个 label1 标签子对象
    lv_obj_t * label1 = lv_label_create(page1,NULL);//这里的父对象应该是 page1
    lv_label_set_long_mode(label1, LV_LABEL_LONG_BREAK);//设置长本文模式
```

```
    //设置与 page1 页面的填充区域宽度相等
    lv_obj_set_width(label1,lv_page_get_fit_width(page1));
    lv_label_set_text(label1, "Hello zheng dian yuan zi, I am xiong jia yu,\n"
    "sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.\n"
    "Ut enim ad minim veniam, quis nostrud exercitation ullamco\n"
"laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure\n"
"dolor in reprehenderit in voluptate velit esse cillum dolore\n"
"eu fugiat nulla pariatur.\n"
"Excepteur sint occaecat cupidatat non proident, sunt in culpa\n"
"qui officia deserunt mollit anim id est laborum.");

//2.3  往 page1 页面中添加一个 btn1 按钮子对象
btn1 = lv_btn_create(page1,NULL);//这里的父对象应该是 page1
lv_obj_set_size(btn1,80,50);

//设置与 label1 的对齐方式
lv_obj_align(btn1,label1,LV_ALIGN_OUT_RIGHT_MID,0,0);
lv_obj_t * btn_label = lv_label_create(btn1,NULL);//给 btn1 按钮添加文本标题
lv_label_set_text(btn_label,"Clean");

//设置事件回调函数,当点击时清空 page1 内部的所有子对象
lv_obj_set_event_cb(btn1,event_handler);

}

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

    if(key==KEY0_PRES)
    {
        lv_page_focus(page1,btn1,LV_ANIM_ON);//使 btn1 按钮处于可见状态
    }else if(key==KEY1_PRES)
    {
        lv_page_scroll_ver(page1,-10);//让 page1 页面往下滚动
    }else if(key==KEY2_PRES)
    {
        lv_page_scroll_hor(page1,-10);//让 page1 页面往右滚动
    }else if(key==WKUP_PRES)
    {
        sb_mode++;
        if(sb_mode>LV_SB_MODE_UNHIDE)
            sb_mode = LV_SB_MODE_OFF;
```

```
            lv_page_set_sb_mode(page1,sb_mode);//设置滚动条的模式
        }
    }
```

3.4 下载验证

把代码下载进去之后,就能看到如下所示的初始界面效果:

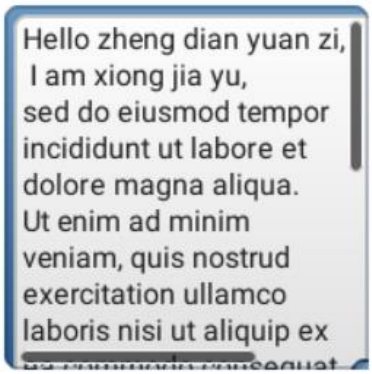

图 3.4.1 初始界面效果

然后我们可以按 WK_UP/KEY_UP 按键,来观看各种滚动条模式下的效果,接着我们来手动滚动页面或者按下 KEY0 键,让内部的按钮处于可见状态,如下所示:

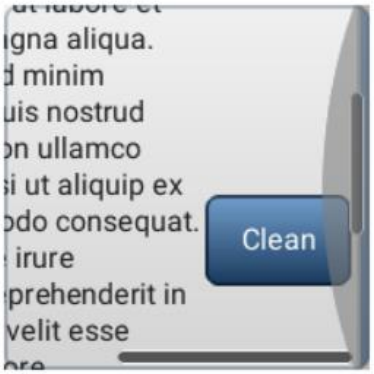

图 3.4.2 按钮处于可见状态

然后我们可以点击 Clean 按钮一下,它会把 page1 页面内的所有子对象全部给删除掉的,最后可以看到如下所示的空白页面效果:

图 3.4.3 空白页面

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