

正点原子 littleVGL 开发指南

lv_font 字体

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_font 字体

1. 介绍

在 littleVGL 中的字体功能也是非常强大的,支持最高 8bpp(在老版本中存在)的抗锯齿,另外还有 1bpp,2bpp,4bpp 三个值可选,实现灵活性配置,选择越大的 bpp 值时,要求的 flash 存储资源也是成倍的增加的,比如 4bpp 的存储消耗是 1bpp 存储消耗的 4 倍,除了存储消耗变大之外,bpp 值越大,在界面上进行字体渲染时,绘制速度也会越慢,当然了,bpp 值越大,也是有好处的,那就是绘制出来的字体边缘越平滑,没有毛刺,使我们产品的 UI 界面看上去更高大上,所以在实际项目中,我们应该根据硬件平台的实际性能来选出一个比较合适的 bpp 值,一般的项目都是默认采用 4bpp 就可以了,littleVGL 的字体还支持 unicode 编码显示,注意这里所谓的 unicode 编码是一种泛指能显示所有字符的编码概念,并不是指实际的 unicode 编码,而是指 UTF-8 编码,所以大家不要搞混淆了,littleVGL 在 UTF-8 编码的支持下,可以实现显示全球所有的字符.最后要说的一个功能那就是它还支持“图标字体”,可能大家一听,有点懵,这是在“web 前端”中经常用到的一种技术,可以看出来 littleVGL 的作者涉猎范围很广呀,下面让我们开始更进一步的探究它的细节.

1.1 如何启用 UTF-8 编码

littleVGL 自身支持 2 种编码,一种是 ASCII 编码,此编码只支持英文字符的显示,另外一种也就是我们前面说到的 UTF-8 编码,此编码可以支持全球所有字符的显示,像我们要显示中文,显示图标字体等,那我们就必须得选择 UTF-8 编码了,至于选择哪一个编码是由 lv_conf.h 头文件中的 LV_TXT_ENC 宏来配置的,见如下:

```
//启用 UTF-8 编码,LV_TXT_ENC_ASCII 是启用 ASCII 编码
#define LV_TXT_ENC LV_TXT_ENC_UTF8
```


littleVGL 默认就是启用了 UTF-8 编码的,另外这里有一点需要注意的是当 littleVGL 启用 UTF-8 编码之后,为了我们开发的方便性,最好把我们的集成开发工具的编码也改成 UTF-8,弄成一致之后,我们就可以在编辑器中以字面量的方式直接写入我们想要显示的文本了,如下面代码所示:

```
//想显示啥,就直接写啥,当然了,前提是你的字库中有中国这两个字
lv_label_set_text(label1, "text:中国");
```

如果你不想把集成开发工具的编码弄成 UTF-8 也是可以的,只不过就是麻烦了点,那么你想要显示中国这两个字时,你只能以转义字符的形式来书写了,如下面代码所示:

```
#define ZHONG    "\xE4\xB8\xAD"    //中字的 UTF-8 编码
#define GUO      "\xE5\x9B\xBD"    //国字的 UTF-8 编码
lv_label_set_text(label1, "text:" ZHONG GUO);
```

这种方式的前提是你必须要知道字符的 UTF-8 编码,这个也简单,因为我们有专门的工具来转换,就在你们买的开发板资料盘中,大致路径为“软件资料\软件\中英文字符编码查询”,另外这种转义字符的形式也是很重要的,像我们后面要介绍的”图标字体”只能通过这种方式来实

现显示,下面我就以 Keil 集成开发环境为例,教大家怎么设置为 UTF-8 编码,先点击  图标,然后弹出如下对话框:

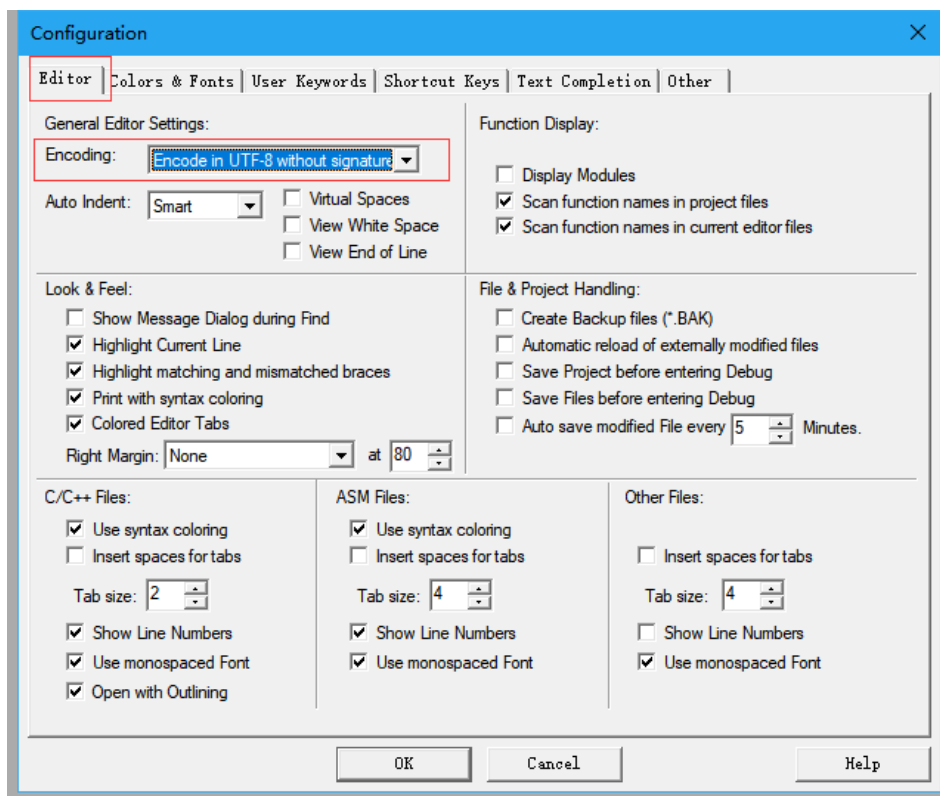


图 1.1.1 Keil 中设置 UTF-8 编码

1.2 啥叫图标字体

这首先是从 web 前端中流行起来的一种技术,见名知意,它是一个图标,按照我们以往的思维,图标一般都是指图片对不对?但是它在 littleVGL 中,即可以当成字体来显示,也可以当成图片来显示,两者都可以喔,不过这种图标有一个限制那就是它只能显示单色,果然是继承了字体的特性呀,littleVGL 系统给我们自带了许多个这样的常用小图标,如下图所示(左边是显示效果,右边是这个图标的名称):



















































	LV_SYMBOL_AUDIO
	LV_SYMBOL_VIDEO
	LV_SYMBOL_LIST
	LV_SYMBOL_OK
	LV_SYMBOL_CLOSE
	LV_SYMBOL_POWER
	LV_SYMBOL_SETTINGS
	LV_SYMBOL_TRASH
	LV_SYMBOL_HOME
	LV_SYMBOL_DOWNLOAD
	LV_SYMBOL_DRIVE
	LV_SYMBOL_REFRESH
	LV_SYMBOL_MUTE
	LV_SYMBOL_VOLUME_MID
	LV_SYMBOL_VOLUME_MAX
	LV_SYMBOL_IMAGE
	LV_SYMBOL_EDIT
	LV_SYMBOL_PREV
	LV_SYMBOL_PLAY
	LV_SYMBOL_PAUSE
	LV_SYMBOL_STOP
	LV_SYMBOL_NEXT
	LV_SYMBOL_EJECT
	LV_SYMBOL_LEFT
	LV_SYMBOL_RIGHT
	LV_SYMBOL_PLUS
	LV_SYMBOL_MINUS
	LV_SYMBOL_WARNING
	LV_SYMBOL_SHUFFLE
	LV_SYMBOL_UP
	LV_SYMBOL_DOWN
	LV_SYMBOL_LOOP
	LV_SYMBOL_DIRECTORY
	LV_SYMBOL_UPLOAD
	LV_SYMBOL_CALL
	LV_SYMBOL_CUT
	LV_SYMBOL_COPY
	LV_SYMBOL_SAVE
	LV_SYMBOL_CHARGE
	LV_SYMBOL_BELL
	LV_SYMBOL_KEYBOARD
	LV_SYMBOL_GPS
	LV_SYMBOL_FILE
	LV_SYMBOL_WIFI
	LV_SYMBOL_BATTERY_FULL
	LV_SYMBOL_BATTERY_3
	LV_SYMBOL_BATTERY_2
	LV_SYMBOL_BATTERY_1
	LV_SYMBOL_BATTERY_EMPTY
	LV_SYMBOL_BLUETOOTH

图 1.2.1 littleVGL 自带的图标字体效果

2. 如何使用字体

不论是使用 littleVGL 系统自带的字体,还是使用我们自己创建的字体,他们的使用方法都是一样的,为了大家更容易接受,我先给大家介绍怎么使用字体,后面我再给大家介绍怎么创建字体并使用它,我们就先以系统自带的字体为例来讲解怎么使用它,littleVGL 内置了 5 个字库,如下所示:

```
lv_font_roboto_12
lv_font_roboto_16
lv_font_roboto_22
lv_font_roboto_28
lv_font_unscii_8
```

其中的 lv_font_roboto_12, lv_font_roboto_16, lv_font_roboto_22,lv_font_roboto_28 都是 [Roboto](#) 字体,只不过是细分出了 4 个不同的字体大小,这四个字库全都是 4bpp 抗锯齿的,这四个字库中每一个字库除了包含所有的可见英文字符外,还都包含了一套相同的图标字符,也就是我们前面所说的图标字体,这套图标字符有一个自己的字体名为 [FontAwesome](#),然后 lv_font_unscii_8 是一种非常好的单色屏字体,是 1bpp 的,相当于无抗锯齿的功能,另外 lv_font_unscii_8 字库只包含英文字符,不包含上面所说的图标字符。

为了节省存储空间,这 5 个字库中,默认情况下只有 lv_font_roboto_16 字库是被使能了的,如果想使能其他的字体或者想设置默认的字体等,就必须得在 lv_conf.h 头文件中进行配置,相关的配置宏如下:

```
#define LV_FONT_ROBOTO_12    0    //是否使能 lv_font_roboto_12 字库
#define LV_FONT_ROBOTO_16    1    //是否使能 lv_font_roboto_16 字库,默认被使能了
#define LV_FONT_ROBOTO_22    0    //是否使能 lv_font_roboto_22 字库
#define LV_FONT_ROBOTO_28    0    //是否使能 lv_font_roboto_28 字库
#define LV_FONT_UNSCII_8     0    //是否使能 lv_font_unscii_8 字库

//这个是用来申明我们自己创建的字体的, 如果没有创建字体,那么可以留空
#define LV_FONT_CUSTOM_DECLARE

//把 lv_font_roboto_16 定义为默认字体
#define LV_FONT_DEFAULT      &lv_font_roboto_16

#define LV_FONT_FMT_TXT_LARGE 0    //当你的字库系统非常庞大时,比如字符数
                                   //超过 10000 个,或者遇到某些莫名的错误时,那么你可以使能它
```

这里还有一点是关于系统自带的图标字符,他们全部申明在 lv_symbol_def.h 头文件中,相对路径为 GUI\lvgl\src\lv_font,可以自行查看图标名,或者你直接查看图 1.2.1 也是没问题的,对于字体的使用,可以概括为如下三步:

1)申明字体

如果是自己创建的字体,那么在使用前,就必须得先申明字体,否则在其他的文件中就会报找不到字体的错误,这跟变量申明是一模一样的原理,如果是系统自带的字体,那么就可以不用申明了,因为 littleVGL 内部已经帮我们申明好了,申明字体是用 LV_FONT_DECLARE 宏来申明的,如下所示:

```
LV_FONT_DECLARE(my_font_1); //申明 my_font_1 字体
```

其实 LV_FONT_DECLARE 宏的定义是下面这样的:

```
#define LV_FONT_DECLARE(font_name) extern lv_font_t font_name;
```

为了避免在每个地方都要去申明一次字体,我们可以直接利用 lv_conf.h 文件中的 LV_FONT_CUSTOM_DECLARE 配置宏来进行全局申明,如下所示:

```
#define LV_FONT_CUSTOM_DECLARE LV_FONT_DECLARE(my_font_1) \
LV_FONT_DECLARE(my_font_2)
```

2)在样式中设置要使用的字体

直接给出示意代码,这样大家看的更明白

```
static lv_style_t label_style;
lv_style_copy(&label_style,&lv_style_plain_color); //样式拷贝
label_style.text.font = &lv_font_roboto_16; //设置字体
lv_label_set_style(label1, LV_LABEL_STYLE_MAIN, &label_style); //给标签对象设置样式
```

3)使用字体中的字符

常见的有下面这几种方式

```
lv_label_set_text(label1, "Hello"); //纯字符方式
lv_label_set_text(label1, "Hello" LV_SYMBOL_OK); //字符和图标组合方式
//多个图标组合方式,注意图标与图标之间至少得需要一个空格
lv_label_set_text(label1, LV_SYMBOL_OK LV_SYMBOL_CLOSE);
```



图 2.1 纯字符方式



图 2.2 字符和图标组合方式



图 2.3 多个图标组合方式

3. 如何创建自己的字体

如果想要创建自己的字体,那么就得借助官方提供的字体在线转换工具,它提供了图形操作界面,使用简单,此转换工具的在线网址为: <https://littlevgl.com/ttf-font-to-c-array> ,对于国内用户来说,此在线转换工具的缺点就是打开网页速度和转换速度都太慢了,真的急死人喔,有耐心的朋友可以慢慢等哈,打开网页之后,效果如下:

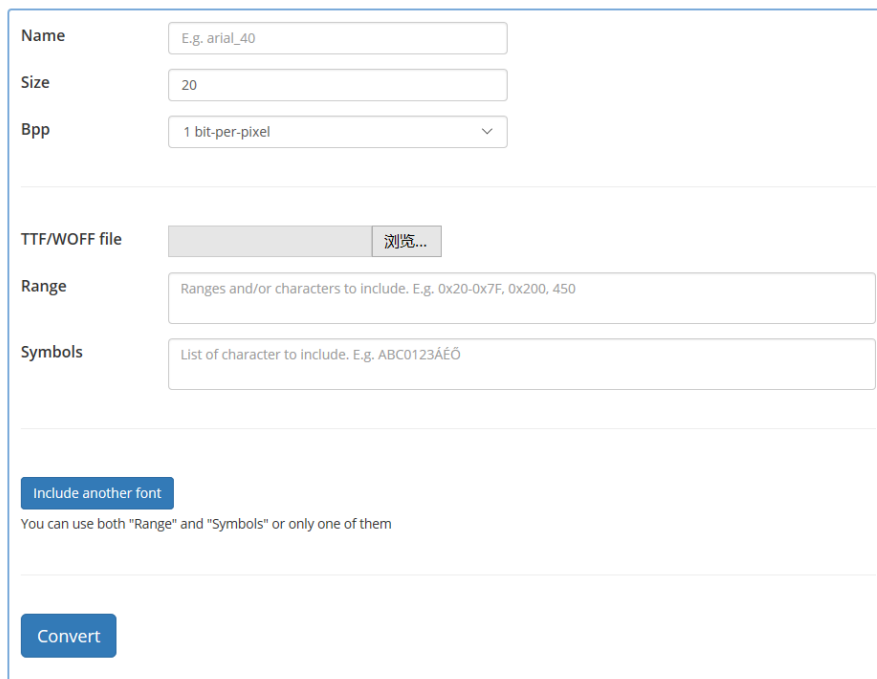


图 3.1 字体在线转换工具

littleVGL 的作者也是考虑到网速慢的原因,因此也推出了离线版本的转换工具,github 链接为: https://github.com/littlevgl/lv_font_conv ,这是用 node.js 写的,没有图形化界面,需要敲命令行,比上面的在线转换工具可能要复杂点,但是它的优点非常的明显,那就是转换速度快,因此我打算给大家介绍怎么使用离线版本的转换工具,学会了难的,再去搞定图形化界面的在线转换工具,那是分分钟的事哈

4. 离线字体转换工具的使用

离线工具是用 node.js 开发出来的,所以我们需要先安装 node.js 的运行环境,安装 node.js 的过程我就不细说了,网上有很多关于这方面的帖子,我这里给出一个网上的参考教程:

<https://www.runoob.com/nodejs/nodejs-install-setup.html> ,在安装完 node.js(会附带把 npm 包管理器给装好)之后,我们接下来要安装 lv_font_conv 离线转换工具了,操作很简单,先打开 cmd 命令窗口,然后直接输入 `npm i lv_font_conv -g` 命令,回车运行就会把 lv_font_conv 给安装好,装完之后,我们来讲一下 lv_font_conv 的主要命令行参数

--bpp: 抗锯齿大小,可选值为 1-4

--size: 字体的大小,实际就是指字符的高度

-o(或者--output): 输出路径,比如为

C:\Users\Administrator\Desktop\my_font_heiti_30.c

--format: 输出格式,可选值有 dump,bin,lvgl,我们只用 lvgl 就行了

--font: ttf/woff/woff2 字体文件的路径

-r(或--range): 所需字符的 unicode 编码范围,可选值为单个字符,字符范围,加可选的映射地址,首先你得必须保证你所需要的字符范围在--font 指定的字体文件中能找得到,-r 命令行参数可以在一条命令中多次出现,如下面例子所示:

```
-r 0xF450  单个字符,十六进制和十进制都行
-r 0xF450-0xF470  字符范围
-r '0xF450=>0x81'  单个字符加映射地址
-r '0xF450-0xF470=>0x81'  字符范围加映射地址
-r 0xF450 -r 0xF451-0xF470  一下使用 2 次-r 命令行参数
-r 0xF450,0xF451-0xF470  用单个-r 表示多种可选值
```

上面所说的映射地址只有在多个字体进行合并时才会用得到,比如前面所说的 lv_font_roboto_16 字体,那就是 [Roboto](#) 字体文件和 [FontAwesome](#) 图标字体文件合并而成的

--symbols: 和 -r 的作用差不多,都是用来指示所要用的字符,不过他们的表达形式不一样,对--symbols 而言,它是接受字符的字面量形式,而 -r 是接受字符的编码形式,--symbols 和 -r 可以同时使用,也可以只使用它们其中的一个,请看如下例子:
--symbols 0123456789 中 ABCD 国 EFG

--no-compress: 禁止进行 RLE 压缩,我们在生成字体时,请禁止进行压缩

上面介绍的都是重要的命令行参数,至于剩下的--autohint-off, --autohint-strong, --force-fast-kern-format, --no-prefilter, --no-kerning, --full-info 这些命令行参数都是不太重要的,自己可以进入 lv_font_conv 的 github 地址自行了解。

下面给出一条完整的创建命令示例(复制到 cmd 命令窗口运行的):

```
lv_font_conv --no-compress --format lvgl --font C:\Users\fish\Desktop\heiti.ttf -o C:\Users\fish\Desktop\my_font.c --bpp 4 --size 30 --symbols 我是熊家余 -r 0x20-0x7F
```


5. 创建普通字体

创建一个字体之前,我们首先要完成 2 点,第一点就是要获取到相应的 ttf/woff/woff2 字体文件,第二点就是要明确我们需要这个字体文件中的那些字符,对于 ttf/woff/woff2 字体文件我们可以从网上去下载,或者直接从系统自带的字体文件中拷贝过来,对于 Windows 系统而言,它的字体文件一般都存放在 C:\Windows\Fonts 目录下,下面我们以一个例子来进行讲解。

假如我们现在要创建一个 30 像素大小的普通字库(之所以称为普通,是因为它里面不含有图标字体),我准备用黑体来取字符,这个黑体的 ttf 文件我是从 Windows 系统自带的字体目录下拷贝过来的,然后直接放到桌面上,接着改名为 heiti.ttf,假如我们这个字库只需要 0-9 十个数字和正点原子四个汉字就可以了,也就是总共 14 个字符,采用 4bpp 抗锯齿,然后我们构建出用于创建此字库的命令,如下所示:

```
lv_font_conv --no-compress --format lvgl --font C:\Users\fish\Desktop\heiti.ttf -o
C:\Users\fish\Desktop\my_font_heiti_30.c --bpp 4 --size 30 --symbols 正点原 - r 0x30-0x39
- r 0x5B50
```

其中 0x5B50 是子字的 unicode 编码,见下图所示:

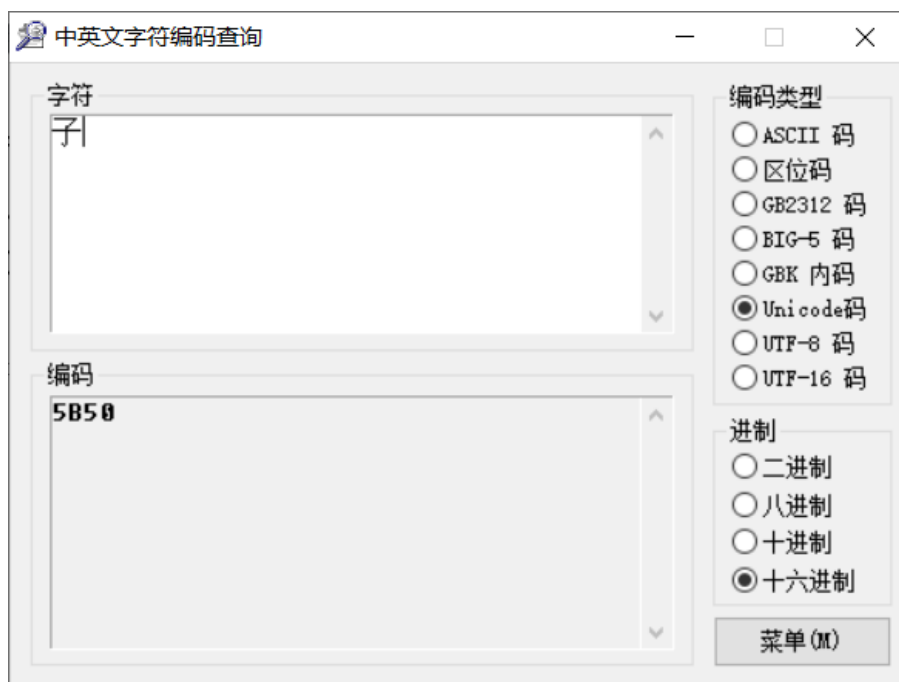


图 3.2.1 子字的 unicode 编码

此编码转换工具在我们提供的资源当中可以找到,我之所以不把子字和正点原三个汉字写到一起,是因为我想给你们多演示几种取字符范围的方式,然后复制上面的命令,粘贴到 cmd 命令窗口中去,如下图所示:

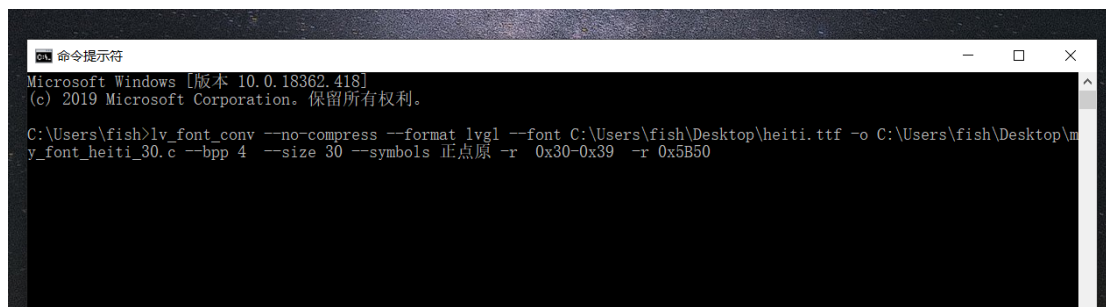


图 3.2.2 命令运行窗口

粘贴完成之后,直接回车运行,几秒钟后,就可以在桌面上看到刚创建出来的 my_font_heiti_30.c 这个字体文件,再接着我们把这个 my_font_heiti_30.c 文件拷贝到我们 Keil 项目的 GUI_APP 目录下,接着用 Keil 打开项目,把这个.c 文件导入到 GUI_APP 分组下,此时还需要注意 2 个细节,第一个就是把 Keil 的编码改成 UTF-8,第二个就是 littleVGL 得启用 UTF-8 编码,最后我们再来写一段简单的测试代码,如下所示:

```
LV_FONT_DECLARE(my_font_heiti_30);//申明字体
```

//测试例程入口

```
void lv_font_test_start()
{
    lv_obj_t* scr = lv_scr_act();

    lv_obj_t* label1 = lv_label_create(scr,NULL);
    static lv_style_t style1;
    lv_style_copy(&style1,&lv_style_plain_color);
    style1.text.font = &my_font_heiti_30;//在样式中使用这个字体
    lv_label_set_style(label1,LV_LABEL_STYLE_MAIN,&style1);
    lv_label_set_text(label1,"正点原子 123456");//设置文本
    lv_label_set_body_draw(label1,true);
    lv_obj_align(label1,NULL,LV_ALIGN_CENTER,0,0);
}
```

运行起来之后的效果如下图所示:



图 3.2.3 运行效果图

6. 创建图标字体

虽然 littleVGL 自带的字体中给我们提供了一些常用的图标,但是总有时候,我们需要的

图标在 littleVGL 提供的图标中是找不到的,那么这时候我们该怎么办呢?别慌,我们自己动手撸它,只要功夫深,铁棒也得被磨成针呀,现在我们来讲解如何创建一个自己的图标字库,其实这个最大的难点就是如何获取到一个 **ttf/woff/woff2** 格式的字体文件,只要解决了这个,那么图标字体的创建和上面普通字体的创建就没啥大区别了。

现在我给大家介绍一个免费的图标字体平台,叫“阿里巴巴矢量图标库”,简称 **iconfont**,没错他是我们国内开发的一个平台,官方网址为:<https://www.iconfont.cn/>,非常的好用,而且操作很简单,我们只需要在上面挑选出我们需要的图标,然后把其添加到我们创建的图标项目中,接着只要点击“下载至本地”按钮,就可以把相应的 **.ttf** 图标字体文件下载到我们的电脑上了,下面我们就开始讲一下这个阿里巴巴矢量图标库平台怎么使用。

在浏览器输入官网网址,打开网页后,如果是第一次使用的话,那么请先注册一下账号,如下图所示:

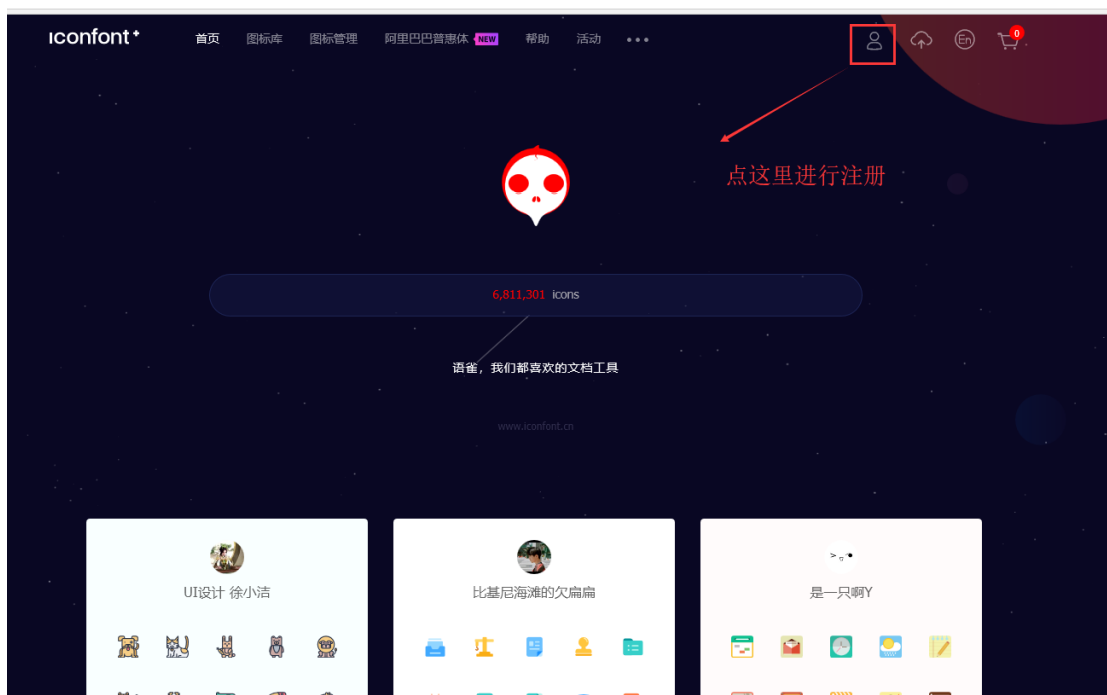


图 3.2.1 注册账号

注册完账号后,我们需要来创建一个图标项目了,先点击“图标管理”->“我的项目”进入项目管理面板,如下图所示:

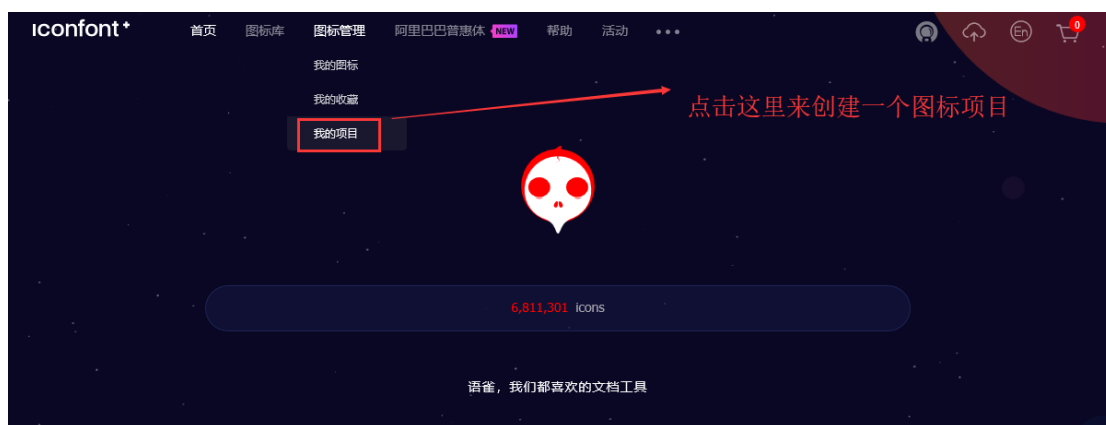


图 3.2.2 点击进入项目管理面板

进入项目管理面板之后,再接着点击  图标来创建项目了,如下图所示:



图 3.2.3 点击进行创建项目

点击之后,会弹出一个创建对话框,里面只需要输入完项目名称就可以了,其他的保持默认就行,如下图所示:



图 3.2.4 创建图标项目

创建完图标项目之后,现在我们要去挑选我们所需要的图标了,这里我以 wifi 和 usb 俩个图标为例进行演示,在搜索框中直接输入我们需要的图标名就可以了,如下图所示:

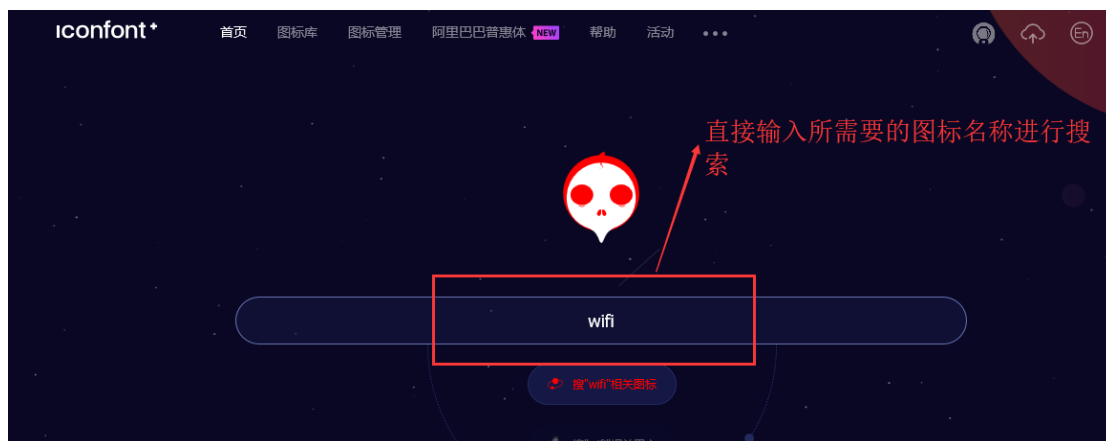


图 3.2.5 搜索图标

搜索之后,会出现一大堆的图标,我们需要把符合我们条件的图标先添加入库,如下图所示:



图 3.2.6 把符合条件的图标添加入库

wifi 图标添加入库之后,usb 图标也是同样的方式添加入库,接着我们需要把入库了的图标

保存到我们刚刚创建了图标项目中,这要先点击最右上角的购物车图标,如下图所示:



图 3.2.7 点击购物车图标

点击之后,会弹出库面板,然后点击"添加至项目"按钮,如下图所示:



图 3.2.8 把图标添加到刚创建的项目中

接下来的最后一步就是把此图标项目下载到我们的本地磁盘上,如下图所示:



图 3.2.9 下载图标项目

下载完后,我们会得到一个 download.zip 的压缩包,我们只需要其里面的 iconfont.ttf 这一个字体文件,我们把 iconfont.ttf 这个文件拷贝到桌面上,接下来我们需要构建一条创建图标字库的命令,命令如下所示:

```
lv_font_conv --no-compress --format lvgl --font C:\Users\fish\Desktop\iconfont.ttf -o C:\Users\fish\Desktop\my_font_icon_30.c --bpp 4 --size 30 -r 0xe7e0,0xed6a
```

其中 0xe7e0 是 wifi 图标的 unicode 编码,0xed6a 是 usb 图标的 unicode 编码,这个是在我们刚创建的图标项目中查看到的,如下图所示:



图 3.2.10 查看图标对应的 unicode 编码

最后我们复制上面的命令到 cmd 命令窗口中进行运行,然后把生成的 my_font_icon_30.c 文件拷贝到 GUI_APP 目录下,接着把此文件导入到 Keil 中,写一段简单的测试代码如下:

```
LV_FONT_DECLARE(my_font_icon_30);//申明字体

//定义图标,因为 littleVGL 只接受 utf-8 编码
//所以我们得通过工具把图标的 unicode 编码转化成 utf-8 编码
#define MY_ICON_WIFI "\xEE\x9F\xA0"
#define MY_ICON_USB "\xEE\xB5\xAA"

//测试例程入口
void lv_font_test_start()
{
    lv_obj_t* scr = lv_scr_act();

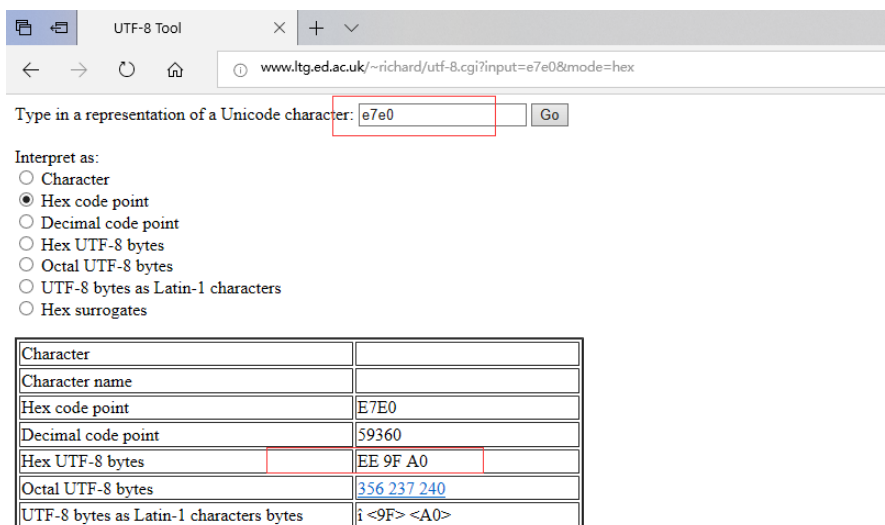
    lv_obj_t* label1 = lv_label_create(scr,NULL);
    static lv_style_t style1;
    lv_style_copy(&style1,&lv_style_plain_color);
    style1.text.font = &my_font_icon_30;//使用图标字体
    lv_label_set_style(label1,LV_LABEL_STYLE_MAIN,&style1);//设置样式
    lv_label_set_text(label1,MY_ICON_WIFI MY_ICON_USB);//设置文本
    lv_label_set_body_draw(label1,true);//绘制背景
    lv_obj_align(label1,NULL,LV_ALIGN_CENTER,0,0);//标签与屏幕居中对齐
}
```



图 3.2.11 运行效果

上面我们说到需要把图标的 unicode 编码转化为 utf-8 编码,这里需要用到转换工具,总共有 2 个工具,一个在线的,一个离线的,在线转换工具的网址为:

<http://www.ltg.ed.ac.uk/~richard/utf-8.cgi> ,打开之后界面效果如下:



Type in a representation of a Unicode character:

Interpret as:

- ☐ Character
- ☒ Hex code point
- ☐ Decimal code point
- ☐ Hex UTF-8 bytes
- ☐ Octal UTF-8 bytes
- ☐ UTF-8 bytes as Latin-1 characters
- ☐ Hex surrogates

Character	
Character name	
Hex code point	E7E0
Decimal code point	59360
Hex UTF-8 bytes	EE 9F A0
Octal UTF-8 bytes	356 237 240
UTF-8 bytes as Latin-1 characters bytes	i <9F> <A0>

图 3.2.12 unicode 转 utf-8 在线工具

然后离线版本的工具是笔者为了方便大家,简单写的一个c控制台应用,简陋之处,请大家多多包涵,如后面有机会,我会带领大家用一种全新的 PC 软件开发方式来开发一款 PC 软件,不是用传统的 MFC, QT, VB, C#等方式,而是用 electron 或 nw.js 前端技术,好处就是做出来的界面很容易美化,开发效率也高,而且一套代码跨 Windows, Linux, Mac os 等系统,废话不多说了,此离线版本的使用界面如下:

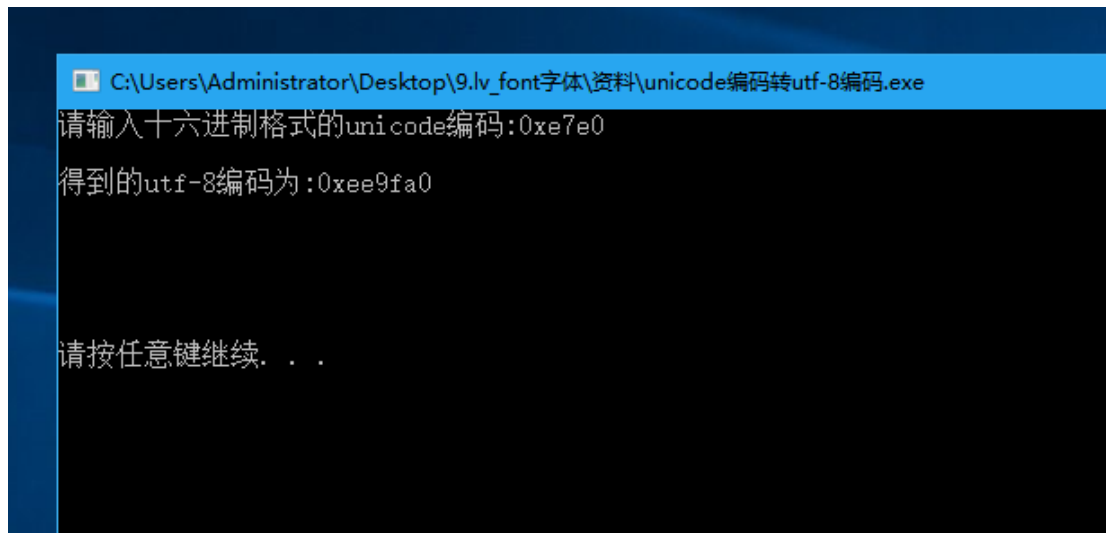


图 3.2.13 unicode 转 utf-8 离线工具

3.4 创建合并字体

有时候,我们可能会有这样的一个需求,那就是需要在一个 label 控件上同时显示普通字符和图标,如下面代码所示:

```
lv_label_set_text(label,MY_ICON_WIFI:"fish 已连接");
```

一般情况下是做不到的,因为图标和普通字符明显是俩个不同的.ttf 字体文件,但是在 littleVGL 中,它做到了,他可以让多个不同的.ttf(.wof 和.woff2 也是可以的)字体文件中的字符合并到一个字库上去,现在我就用 heiti.ttf 普通字体文件和 iconfont.ttf 图标字体文件以合并的方式来生成一个 my_font_30.c 的字库,操作过程跟前面的 3.2 和 3.3 章节差不多是一样的,下面我直接给出创建命令:

```
lv_font_conv --no-compress --format lvgl --font C:\Users\fish\Desktop\heiti.ttf -r 0x20-0x7F
--symbols 已连接 --font C:\Users\fish\Desktop\iconfont.ttf -r 0xe7e0
-o C:\Users\fish\Desktop\my_font_30.c --bpp 4 --size 30
```

复制上面的命令到 cmd 命令窗口进行运行,然后把创建出来的 my_font_30.c 文件拷贝到 GUI_APP 目录下,并添加到 Keil 中,最后写一段测试代码如下:

```
LV_FONT_DECLARE(my_font_icon_30);//申明字体
//定义 wifi 图标
#define MY_ICON_WIFI "\xEE\x9F\xA0"

//测试例程入口
void lv_font_test_start()
{
    lv_obj_t* scr = lv_scr_act();

    lv_obj_t* label1 = lv_label_create(scr,NULL);
    static lv_style_t style1;
    lv_style_copy(&style1,&lv_style_plain_color);
    style1.text.font = &my_font_30;//使用图标字体
    lv_label_set_style(label1,LV_LABEL_STYLE_MAIN,&style1);//设置样式
    lv_label_set_text(label,MY_ICON_WIFI":fish 已连接");//设置文本
    lv_label_set_body_draw(label1,true);//绘制背景
    lv_obj_align(label1,NULL,LV_ALIGN_CENTER,0,0);//标签与屏幕居中对齐
}
```

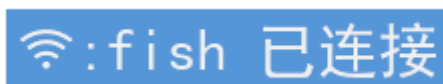


图 3.4.1 合并字库的运行效果

注：上面的 `lv_label_set_text(label,MY_ICON_WIFI":fish 已连接");`这句设置文本的代码在 Keil 中可能会报如下 2 条错误：

```
error: #8: missing closing quote
error: #18: expected a ")"
```

如果没有报此错误的朋友,可以直接忽略跳过此注意事项,因为笔者的电脑上出现了这个情况,这个应该是由 Keil 版本差异问题或者 Keil 自身存在这个小 bug 导致的,什么时候有可能触发这种问题呢?那就是当你的 Keil 设置为 UTF-8 编码,而且字符串又是以汉字结尾时,就有可能触发这种问题,解决思路就是让字符串不以汉字结尾,那么我们可以变相地在字符串末尾添加空格,或者其他不可见的字符,如下所示:

```
lv_label_set_text(label,MY_ICON_WIFI":fish 已连接 ");//添加空格方式
lv_label_set_text(label,MY_ICON_WIFI":fish 已连接\x00");//以转义字符的形式
//添加其他不可见字符
```

4. 例程设计

4.1 功能简介

我们用 digit.ttf, heiti.ttf, iconfont.ttf 三个字体文件来合成一个名为 my_font_30.c 的字库, 其中 digit.ttf 是数码管式的英文字体, 我们只取其中的 0x20-0x7F 字符范围, heiti.ttf 是黑体的中文字体, 我们只取其中的正点原子四个汉字, iconfont.ttf 是图标字体, 我们只取其中的 wifi 和 usb 图标

4.2 硬件设计

本例程所用到的硬件有:

- 1) 液晶屏

4.3 软件设计

先给出创建 my_font_30.c 字库的命令, 如下所示:

```
lv_font_conv --no-compress --format lvgl --font C:\Users\Administrator\Desktop\digit.ttf
-r 0x20-0x7F --font C:\Users\Administrator\Desktop\heiti.ttf --symbols 正点原子 --font C:\
Users\Administrator\Desktop\iconfont.ttf -r 0xe7e0,0xed6a -o C:\Users\Administrator\Desкто
p\my_font_30.c --bpp 4 --size 30
```

生成 my_font_30.c 文件后, 把此文件拷贝到 GUI_APP 目录下, 然后接着在 GUI_APP 目录下创建 lv_font_test.c 和 lv_font_test.h 俩个文件, 其中 lv_font_test.c 文件的内容如下:

```
#include "lv_font_test.h"
#include "lvgl.h"

LV_FONT_DECLARE(my_font_30); //申明字体

//定义图标
#define MY_ICON_WIFI "\xEE\x9F\xA0" //wifi 图标
#define MY_ICON_USB "\xEE\xB5\xAA" //usb 图标

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

```
static lv_style_t my_style;
lv_style_copy(&my_style,&lv_style_plain_color);//样式拷贝
my_style.text.font = &my_font_30;//在样式中使用字体

lv_obj_t* label = lv_label_create(src,NULL);//创建标签控件
lv_label_set_style(label,LV_LABEL_STYLE_MAIN,&my_style);//设置样式
//设置文本
lv_label_set_text(label,"正点原子 Nice\n"MY_ICON_WIFI MY_ICON_USB"\n0123456789");
lv_label_set_body_draw(label,true);//使能绘制背景
lv_obj_align(label,NULL,LV_ALIGN_CENTER,0,0);//标签与屏幕保持居中对齐
}
```

4.4 下载验证

把代码下载进去之后,正常的话,会看到如下所示的界面效果:



图 3.4.1 字体例程界面效果

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