Technical Note

LUA 例程-常用的回调函数使用说明 V1.0

工程技术笔记

类别	内容
关键词	LUA API 回调函数
摘要	

修订历史

版本	日期	原因	编制	审查
V1.0	2019/02/19	创建文档	林青田	

销售与服务

广州大彩光电科技有限公司

电话：020-82186683

传真：020-82187676

Email：hmi@gz-dc.com（公共服务）

网站：www.gz-dc.com

地址：广州高新技术产业开发区玉树工业园富康西街 8 号 C 栋 303 房

官网零售淘宝店：www.gz-dc.taobao.com

目录

1. 适合范围.....1
2. 开发环境版本.....2
3. 概述.....3
4. 参考文档.....4
5. 实现教程.....5
5.1 工程准备.....5
5.1.1 硬件平台.....5
5.1.2 UI素材准备.....5
5.1.3 LUA编辑器.....6
5.2 API函数说明.....7
5.3 教程实现.....9
5.3.1 函数on_init().....9
5.3.2 函数on_systick（）.....11
5.3.3 函数on_timer(timer_id).....13
5.3.4 函数on_control_notify( ).....15
5.3.5 函数on_screen_change().....17
5.3.6 函数on_draw().....19
5.3.7 U盘和SD卡的回调函数.....21
5.4 下载工程.....22
5.4.1 操作过程.....23
6. LUA回调函数的完整程序清单.....24
7. 免责声明.....29

1. 适合范围

该文档适合所有大彩物联型系列。

2. 开发环境版本

1．VisualTFT 软件版本：V3.0.0.944 及以上的版本。

版本查看：

a) 打开 VisualTFT 软件启动页面如图 2-1 软件版本，右上角会显示的软件版本号；

图 2-1 软件版本

b) 打开 VisualTFT，在软件右下角可以查看软件版本图 2-2 软件版本，最新版本可登录http://www.gz-dc.com/进行下载。

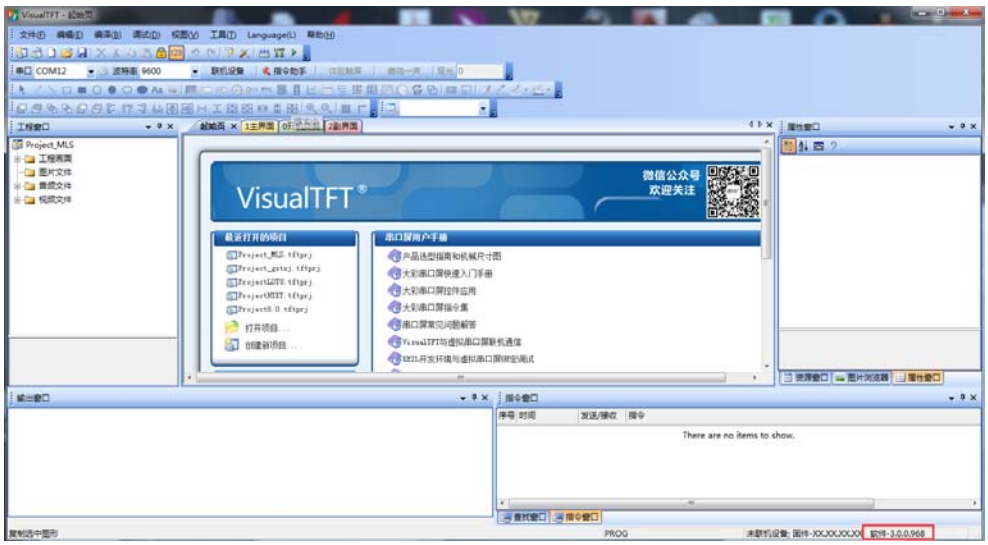


图 2-2 软件版本

2. 串口屏硬件版本：V3.0.287.0 及以上的版本。

版本查看：

a) 查看屏幕背面版本号贴纸；

b) VisualTFT 与屏幕联机成功后，右下角显示的版本号。

3. LUA 语言版本 V5.5

3. 概述

物联型串口屏可以通过 LUA 脚本配合工程完成丰富多样的操作。

本文将介绍大彩 LUA 脚本 API 函数中的回调函数使用方法，以及调用回调函数的注意事项。

4. 参考文档

1. 《大彩组态串口屏 LUA 脚本 API》 可通过以下链接下载物联型开发包获取:
   http:/www.gz-dc.com/index.php?s=/List/index/cid/19.html
2. 《LUA 基础学习》 可通过以下链接下载物联型开发包获取:
   http:/www.gz-dc.com/index.php?s=/List/index/cid/19.html
3. LUA脚本初学者可以通过下面链接进行学习。
   http://www.runoob.com/lua/lua-arrays.html

5. 实现教程

本章节主要通过 LUA 教程 demo 讲述如何实现回调函数的触发使用以及编写程序的注意事项。本文将分为以下是 3 个阶段讲述教程 DEMO 是如何实现的：

1. 准备工程素材，
2. 实现功能，
3. 下载工程。

5.1 工程准备

在实现教程 DEMO 前需要作以下三个准备：

1. 硬件平台，
2. UI 素材，
3. LUA 编辑器。

5.1.1 硬件平台

该例程使用大彩物联型 7 寸串口屏 DC80480W070 为验证开发平台。如图 5-1 所示

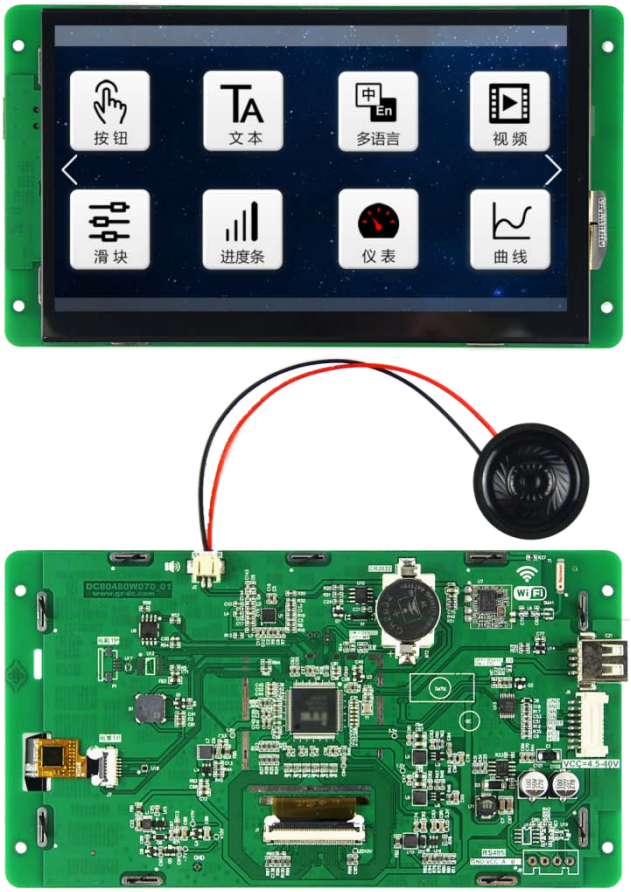


图 5-1 物联型 7 寸串口屏

其他尺寸的物联型串口屏均可借鉴此教程。

5.1.2 UI 素材准备

准备好相应的美工 UI。如图 5-2 所示

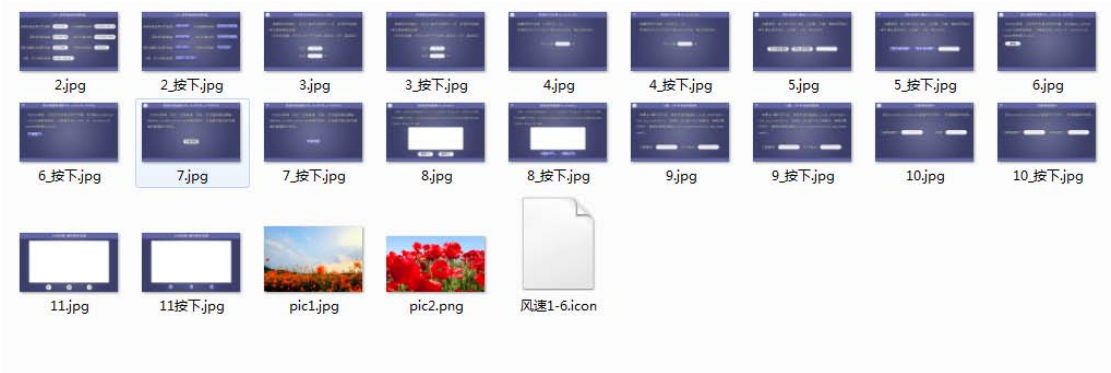


图 5-2 素材准备

5.1.3 LUA 编辑器

上位机 VisualTFT 内部已集成了 LUA 开发编译环境，点击菜单栏工具，选择 LUA 编辑器，如图 5-3 所示。

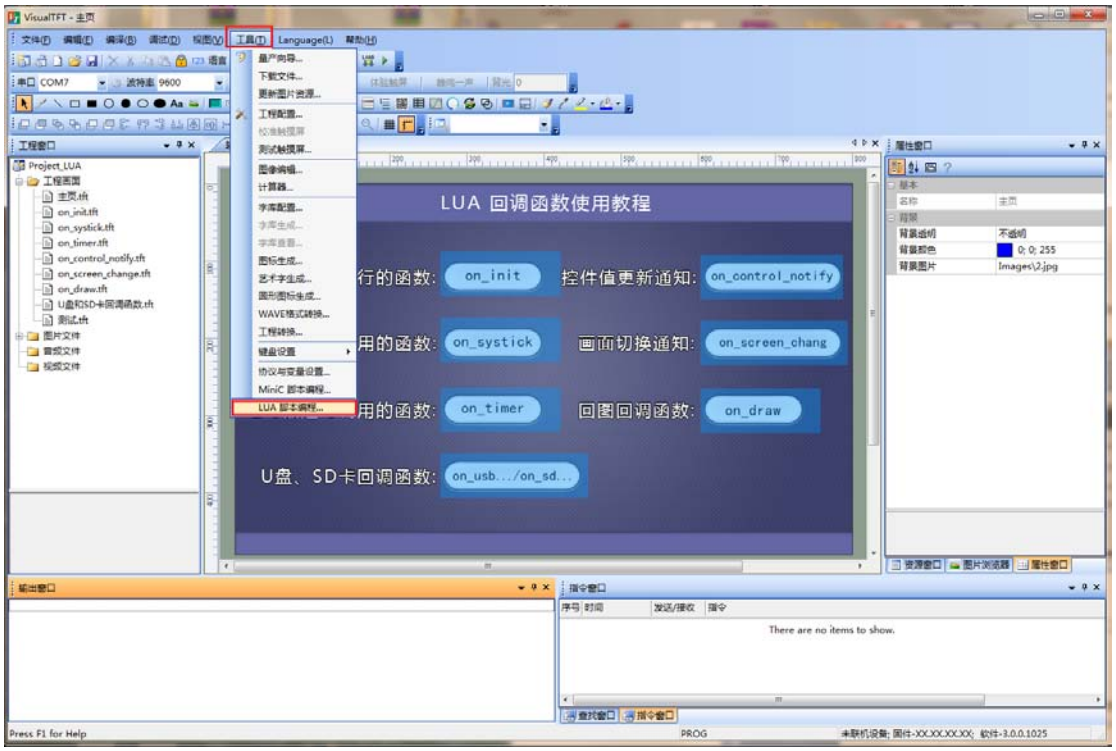


图 5-3 打开 LUA 编辑器

打开编辑器后，Visual TFT 画面如图 5-4 所示；

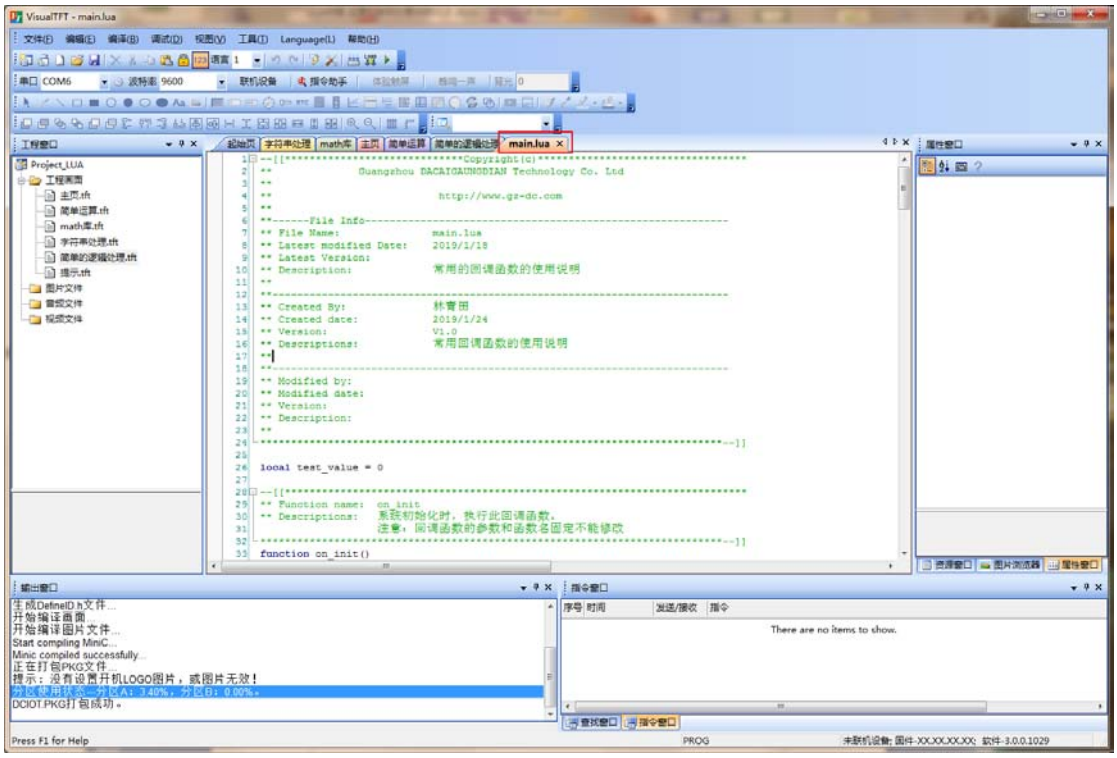


图 5-4 LUA 编辑器

5.2 API 函数说明

大彩科技针对 LUA 脚本提供了丰富的 API 接口函数，具体函数可以查阅文档《物联型 LUA 脚本 API 》，如图 5-5 所示。

3. API 接口函数 8
3.1 控件属性类 8
3.1.1 change_screen(screen) 8
3.1.2 set_value(screen,control,value) 8
3.1.3 get_value(screen,control) 8
3.1.4 set_visiable(screen,control,visiable) 8
3.1.5 set_enable(screen,control,enable) 8
3.1.6 set_fore_color(screen,control,color) 8
3.1.7 set_back_color(screen,control,color) 8
3.1.8 set_text(screen,control,text) 8
3.1.9 get_text(screen,control) 8
3.2 常用回调函数 8
3.2.1 on_init() 8
3.2.2 on_systick() 8
3.2.3 on_control_notify(screen,control,value) 8
3.2.4 on_screen_change(screen) 9
3.2.5 on_press(state,x,y) 9
3.2.6 on_usb_inserted(driver) 9
3.2.7 on_usb_removed() 9
3.3 绘图函数 9
3.3.1 on_draw(screen) 9
3.3.2 redraw() 9
3.3.3 set_pen_color(color) 9
3.3.4 draw_line(x0,y0,x1,y1,width) 9
3.3.5 draw_rect(x0,y0,x1,y1,fill) 9
3.3.6 draw_circle(x,y,r,fill) 9
3.3.7 draw_ellipse(x0,y0,x1,y1,fill) 10
3.3.8 draw_image(image_id,frame_id,dstx,dsty,width,height,srcx,srcy) 10
3.3.9 draw_image_file(filename,dstx,dsty,width,height,srcx,srcy) 10
3.3.10 load_surface (filename) 10
3.3.11 destroy_surface (surface) 10
3.3.12 draw_surface (surface,dstx,dsty,width,height,srcx,srcy) 10
3.3.13 draw_text(text,x,y,w,h,font,color,align) 11
3.4 寄存器访问 11

图 5-5 API 函数文档

本教程文档中所涉及到的部分的相关接口函数如下。

1. 函数 on_control_notify(screen,control,value)

注释：此函数可以在用户触摸控件后被回调。函数有三个参数：

screen 表示画面 ID

control 表示控件的编号

value 表示控件的值。

2. 函数 on_init()

注释：系统初始化时，执行此回调函数。

3. 函数 on_systick( )

注释：定时回调函数，系统每隔 1 秒钟自动调用。

4. 函数 on_timer(timer_id)

注释：定时器超时，执行此回调函数

参数： timer_id --定时器 ID

5. 函数 on_screen_change(screen)

注释：当画面切换至目标画面 ID 时，执行此回调函数

参数：screen 表示目标画面 ID

6. 函数 on_draw(screen)

注释：当调用 redraw（）或屏幕刷新时会触发此函数，在目标画面中进行绘图

参数：screen 表示目标画面 ID

7. 函数 on_usb_inserted (dir)

注释：当插入 U 盘后会触发调用此函数

参数：dir 表示系统分派给 U 盘的路径

8. 函数 on_sd_inserted(dir)

注释：当插入 SD 卡后会触发调用此函数

参数：dir 表示系统分派给 SD 卡的路径

9. 函数 on_usb_removed()

注释：当拔出 U 盘后会触发调用此函数

10. 函数 on_sd_removed()

注释：当拔出 SD 卡后会触发调用此函数

5.3 教程实现

本章节主要讲述回调函数的使用方法，教程中每个函数的功能实现步骤如下：

1. 配置工程属性，
2. 编写程序以及编译，
3. 运行程序。

5.3.1 函数 on_init()

1. 系统加载 LUA 脚本文件之后，立即调用此回调函数， 通常用于执行初始化操作，如图 5-6 所示。

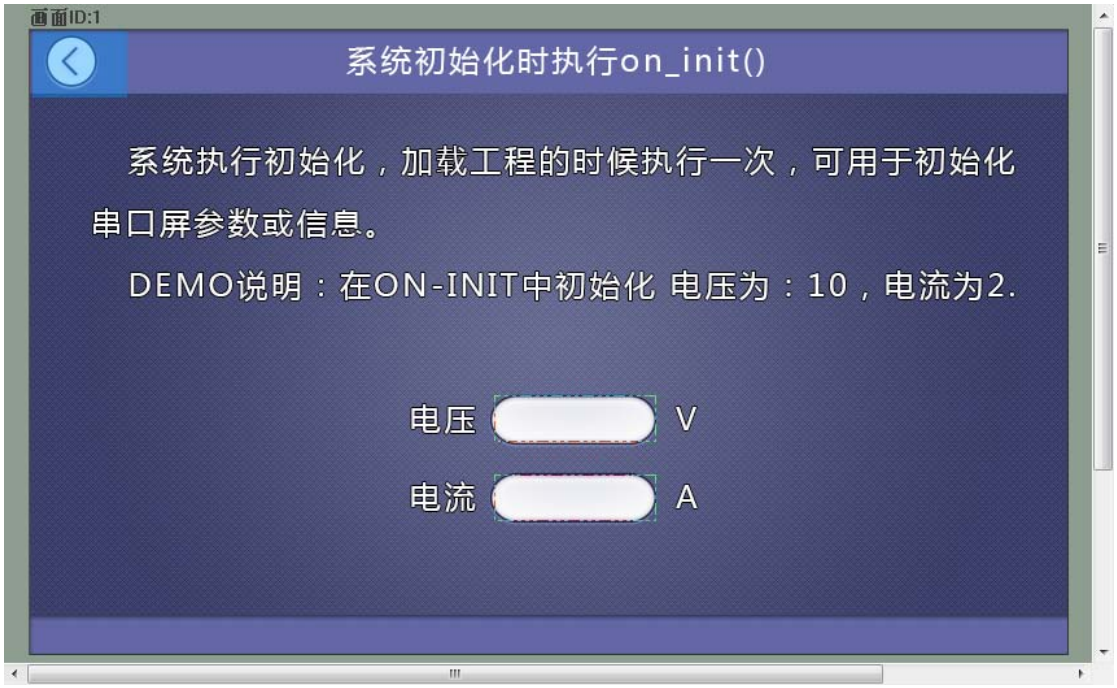


图 5-6 函数 on_init( )

2. 在 on_init( )函数中填写设置文本框数值的 API 函数，在系统初始化是会调用执行。具体代码如程序清单 1 中所示。

程序清单 1 系统初始化执行函数

```
--[[*********************************************************************************
** Function name:    on_init（）
** Descriptions:       系统初始化时，执行此回调函数。
                       注意：回调函数的参数和函数名固定不能修改
*********************************************************************************--]]
function    on_init( )
        set_value(1,1,10)                                    --设置文本控件的值
        set_value(1,2,2)                                     --设置文本控件的值
end
```

写完基础运算功能模块后将工程和脚本一起进行编译，编译成功后可以使用软件中的虚拟屏查看程序是否实现功能，点击工具栏中编译工程按钮，可将工程和脚本的一起编译，操作如图 5-7 所示

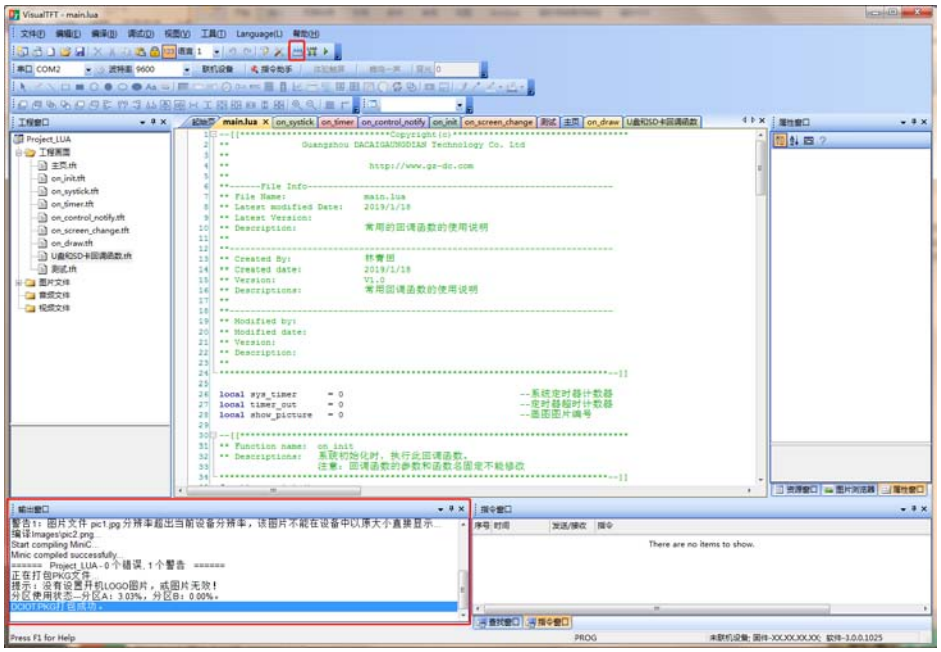


图 5-7 编译工程

注意：目前在软件 Visual TFT 集成的 LUA 脚本编译器无法进行语法检测，所以编写 LUA 程序是尽量分模块编写，已达到节省调试的时间。

3. 点击工具栏中的运行虚拟串口屏，软件先会将当前工程进行编译，编译成功后工程会在虚拟屏上运行起来，如图 5-8 和图 5-9 所示

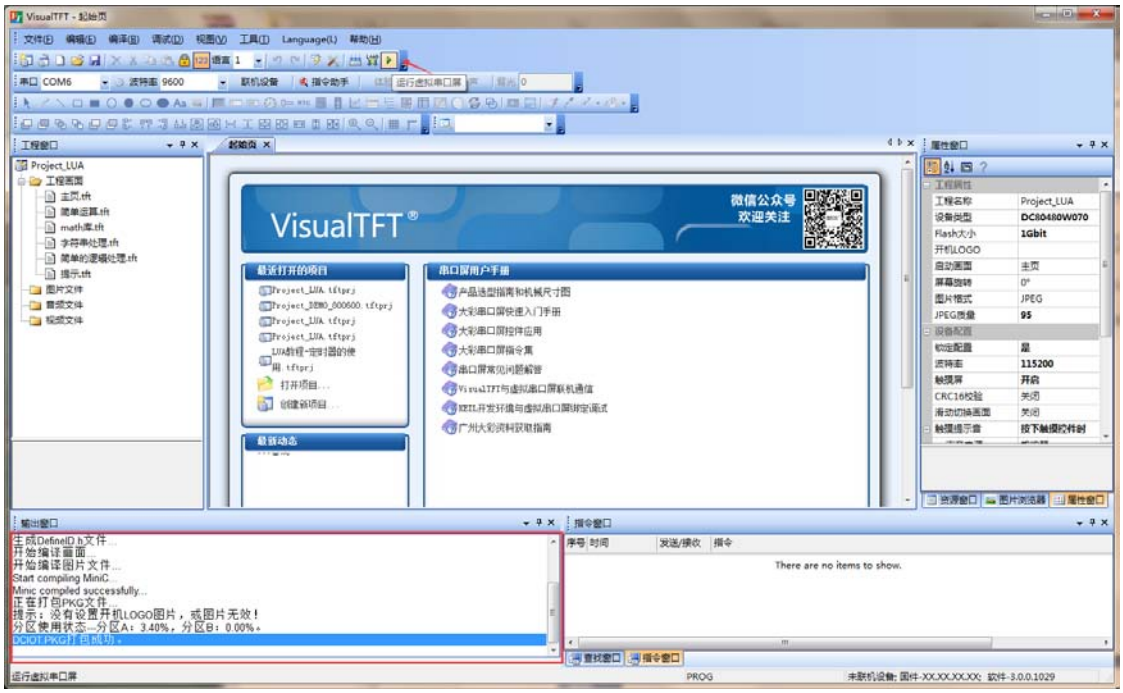


图 5-8 打开虚拟屏

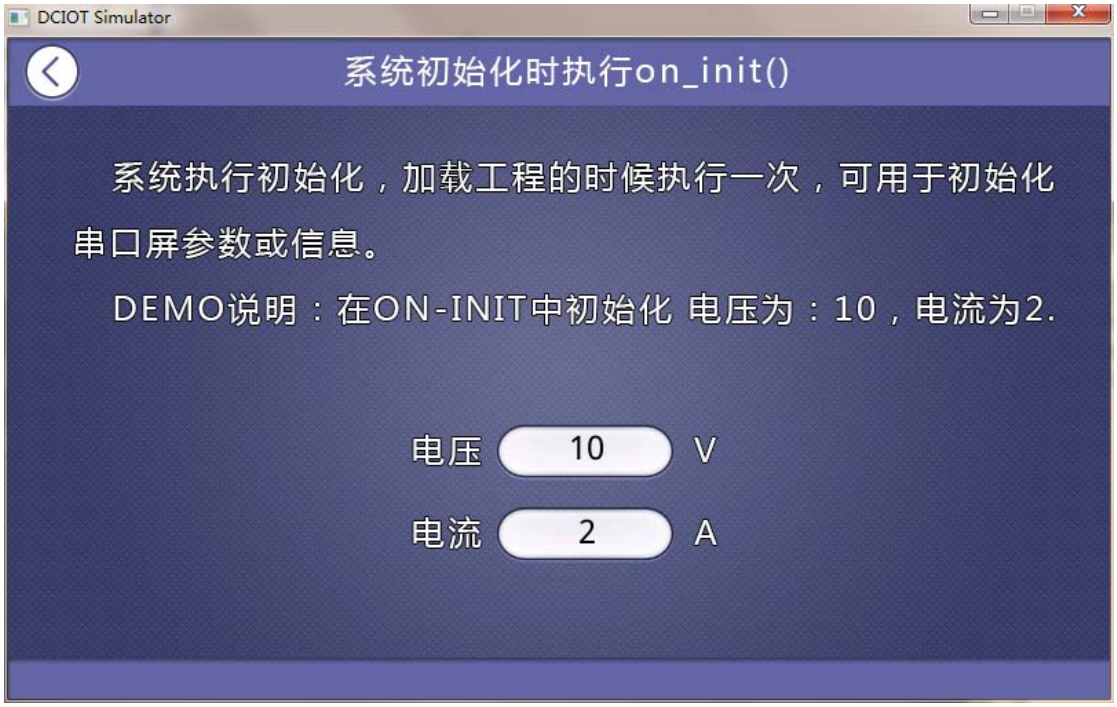


图 5-9 虚拟屏运行演示

5.3.2 函数 on_systick（）

1. 系统每隔 1 秒钟自动调用此回调函数，工程画面如图 5-10 所示

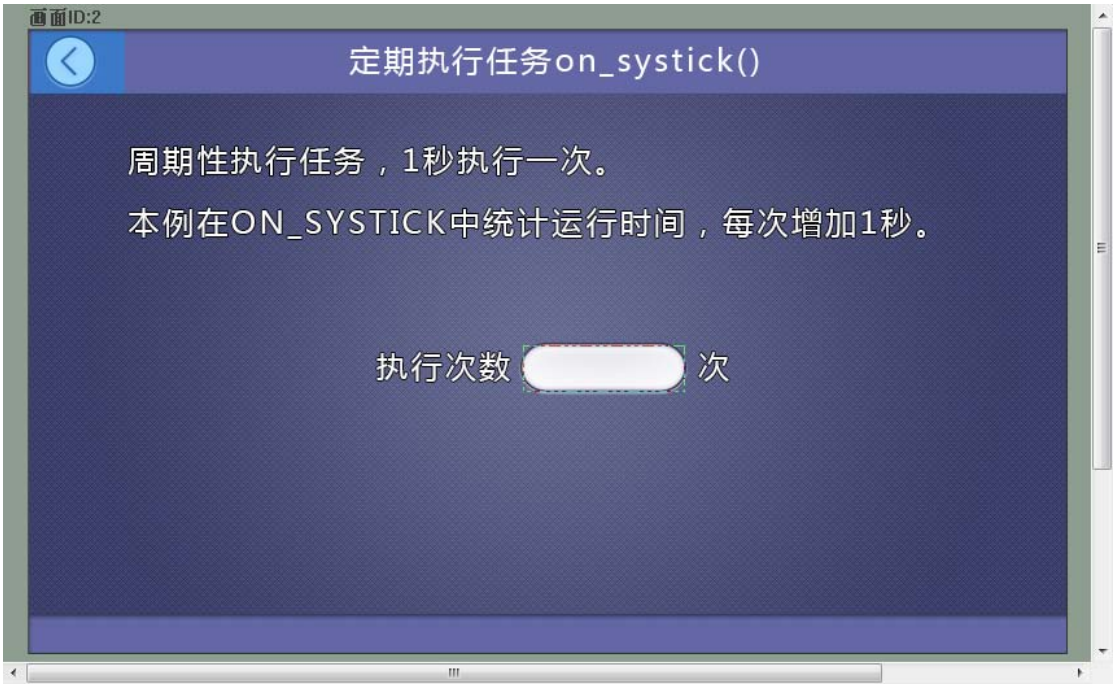


图 5-10 函数 on_systick

2. 教程中在 on_systick（）函数里添加了一个显示回调函数被执行次数的程序，具体代码如程序清单 2 中所示

程序清单 2 函数 on_systick( )

```
--[[********************************************************************************
** Function name:    on_systick
** Descriptions:     定时回调函数，系统每隔 1 秒钟自动调用。
                        注意：回调函数的参数和函数名固定不能修改
********************************************************************************--]]
function    on_systick( )
          sys_timer = sys_timer +1
          set_value(2,1,sys_timer)
end
```

工程编译请参考上一小节 5.3.1 的第 2 步；

3. 运行虚拟屏后切换到当前画面中，在对应的文本控件中的数值每隔一秒中会自行加 1，如图 5-11 虚拟屏所示；

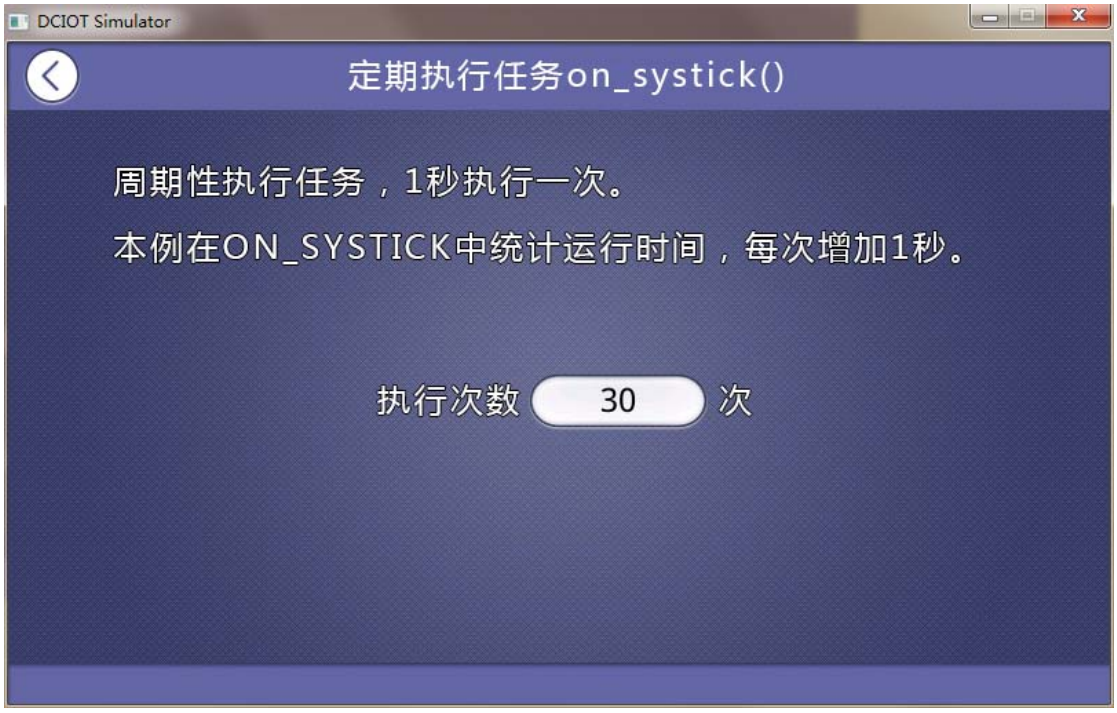


图 5-11 虚拟屏

5.3.3 函数 on_timer(timer_id)

1. on_timer(timer_id)定时器超时回调函数。教程画面中放置了两个按钮和一个文本框，通过按下按钮控件启动或停止定时器，然后在文本框中显示定时器超时后调用的的次数。教程演示画面如图 5-12 所示

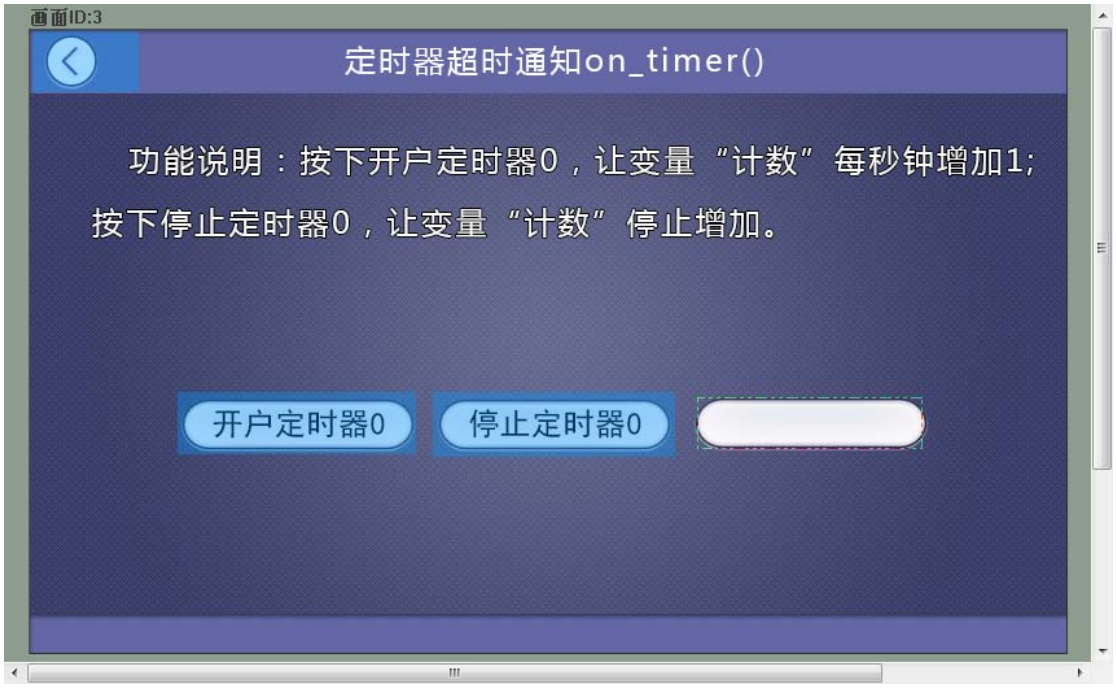


图 5-12 函数 on_timer( )

2. 教程中添加了按钮按下触发启用定时器以及按钮按下触发停止定时器的程序，具体代码如程序清单 3 中所示。

程序清单 3 函数 on_timer

```
--[[*********************************************************************************
** Function name:    on_timer
** Descriptions:     定时器超时，执行此回调函数
                     注意：回调函数的参数和函数名固定不能修改
** Input value :     timer_id  定时超时的定时器 ID 号，定时器编号 0~31
*********************************************************************************--]]
function    on_timer(timer_id)
        if    timer_id == 0 then                                    --定时器 0
            timer_out = timer_out +1
            set_value(3,3,timer_out)
        end
end
--[[*********************************************************************************
** Function name:    on_control_notify
** Descriptions:     用户通过触摸修改控件后，执行此回调函数。
                     点击按钮控件，修改文本控件、修改滑动条都会触发此事件。
                     注意：回调函数的参数和函数名固定不能修改
** Input value :     screen    画面 ID
                     control  控件 ID
                     value     控件值(包括文本控件输入的值)
*********************************************************************************--]]
function    on_control_notify(screen,control,value)
 --*************************************************************
 --功能：按下画面 4 的控件 1 启动定时器 0
 --调用函数：start_timer(timer_id, timeout, countdown, repeat)
 --函数功能：启动定时器
 --参数：        timer_id，  定时器 ID；
 --              timeout，    超时时间；
 --              countdown，1 顺计时，0 倒计时
 --              repeat        计时次数，0 无限循环
 --*************************************************************
      if    screen == 3    and    control == 1 and    value == 1 then
            start_timer(0,1000,1,0)                    --开启定时时器 0 工作，设置计时 1 秒触发一次 on_timer
      end
      if    screen == 3    and    control == 2 and    value == 1 then
            stop_timer(0)                             --停止定时时器 0 工作
            timer_out = 0
      end
……
End
```

工程编译请参考上一小节的第 2 步。

3. 点击运行串口屏后切换到当前画面，点击运行按钮后在当前画面中显示函数运行结果，如图 5-13 虚拟屏所示

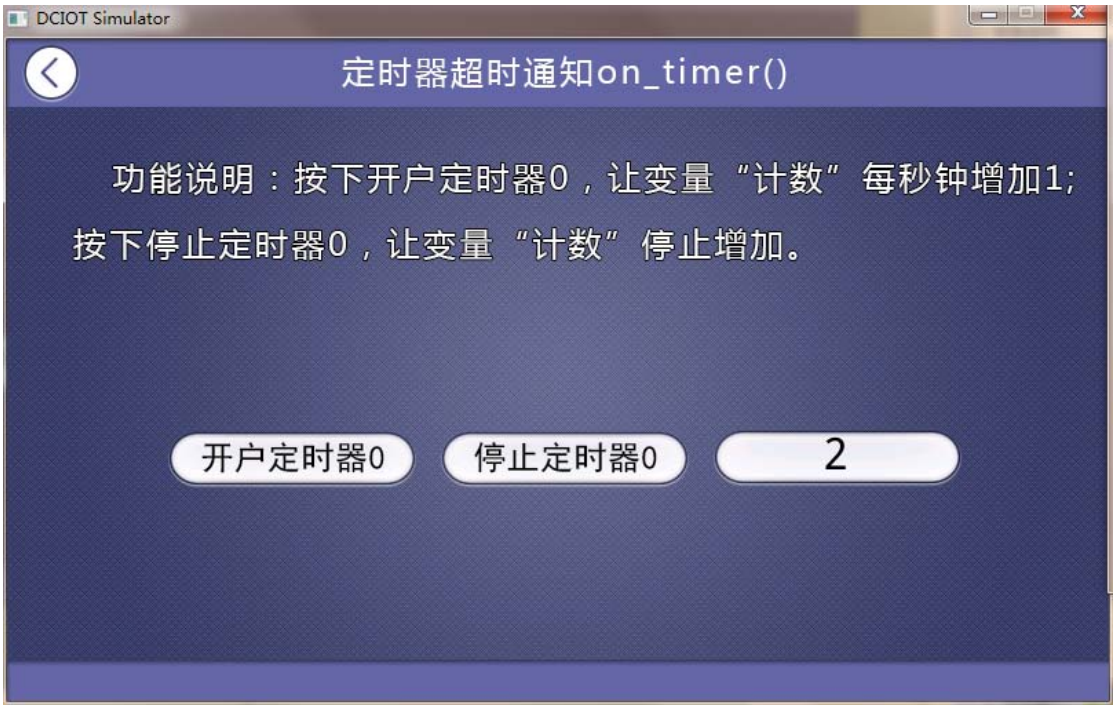


图 5-13 虚拟屏

5.3.4 函数 on_control_notify( )

1. 用户触摸修改控件后，执行此回调函数。点击按钮控件，修改文本控件、修改滑动条都会触发此函数。教程演示画面如图 5-14 所示

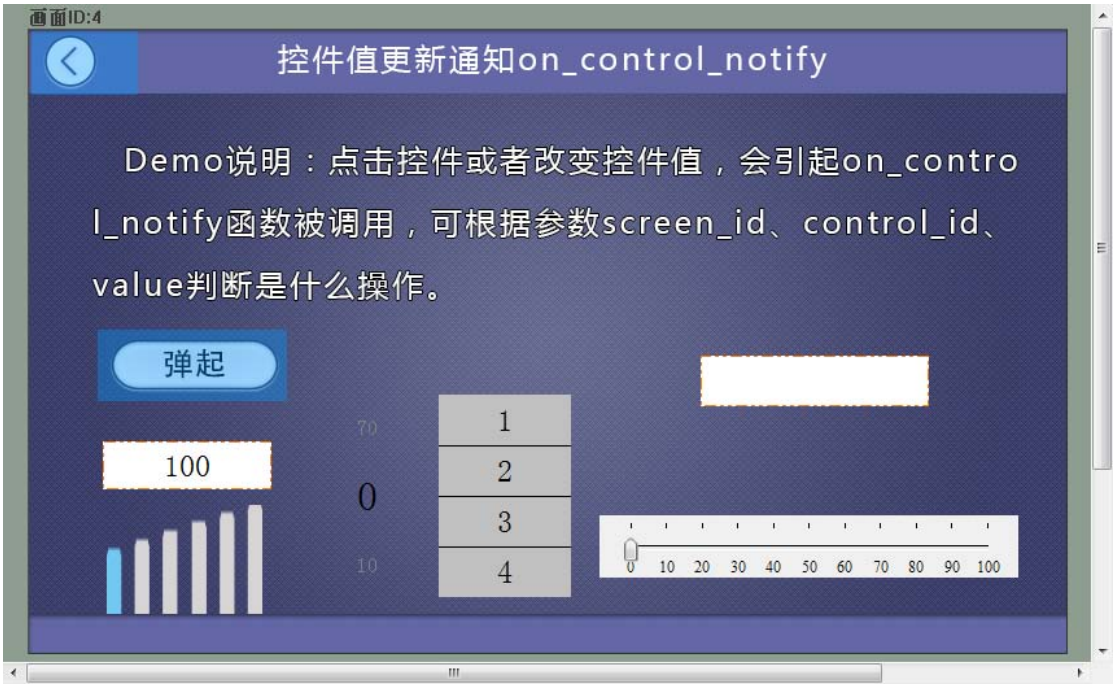


图 5-14 函数 on_control_notify( )

2. 教程中在回调函数函数中添加了对触发函数的控件编号判断，具体代码如程序清单 4 中。

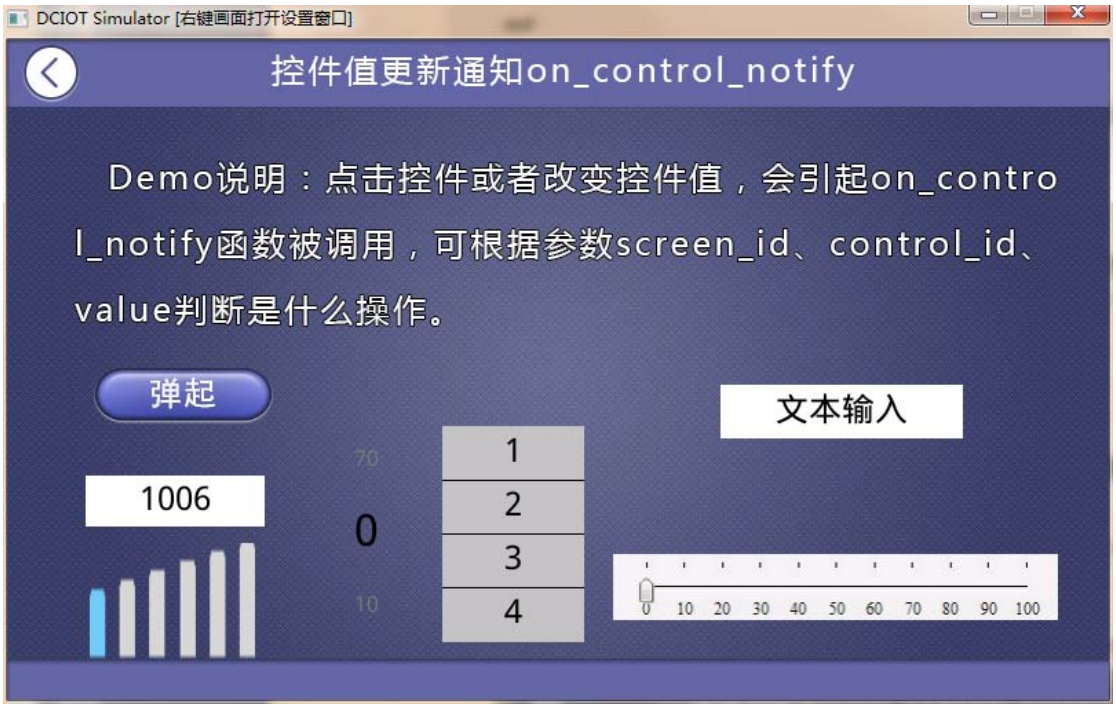
程序清单 4 函数 on_control_notify

```
--[[*********************************************************************************
** Function name:    on_control_notify
** Descriptions:     用户通过触摸修改控件后，执行此回调函数。
                     点击按钮控件，修改文本控件、修改滑动条都会触发此事件。
                     注意：回调函数的参数和函数名固定不能修改
** Input value :     screen    画面 ID
                     control  控件 ID
                     value     控件值(包括文本控件输入的值)
*********************************************************************************--]]
function    on_control_notify(screen,control,value)
 --*************************************************************
 --功能：按下画面 4 的控件 1 启动定时器 0
 --调用函数：start_timer(timer_id, timeout, countdown, repeat)
     --函数功能：启动定时器
 --参数：        timer_id，  定时器 ID；
 --              timeout，     超时时间；
 --                countdown，1 顺计时，0 倒计时
 --                repeat         计时次数，0 无限循环
 --*************************************************************
      ……
  if screen == 4    then
       if control == 1    and    value == 1 then                    --value == 1  按钮按下
              set_text(4,7,"按钮按下")
       elseif    control == 1    and    value == 0 then             --value == 0  按钮弹起
              set_text(4,7,"按钮弹起")
       elseif    control == 2      then                             --文本输入
              set_text(4,7,"文本输入")
       elseif    control == 3      then                             --图标控件
              set_text(4,7,"触摸图标")
       elseif    control == 4    then                               --滑动选择控件
              set_text(4,7,"滑动选择")
       elseif control == 5      then                               --菜单控件
              set_text(4,7,"菜单选择")
       elseif control == 6      then                               --滑块控件
              set_text(4,7,"滑块滑动")
      end
  end
……
```

工程编译请参考上一小节 5.3.1 的第 2 步；

3. 点击运行串口屏后切换到当前画面，点击运行按钮后在当前画面中显示函数运行结

果，如所示



5.3.5 函数 on_screen_change()

1. 当画面需要切换时，执行此回调函数。教程中将按钮控件的属性设置为画面切换，点击该按钮控件就会将画面切换到指定的画面中，演示画面如图 5-15、图 5-16 和图 5-17 所示

注意：内部调用 API 函数 change_screen 切换画面，不会执行函数 on_screen_change。

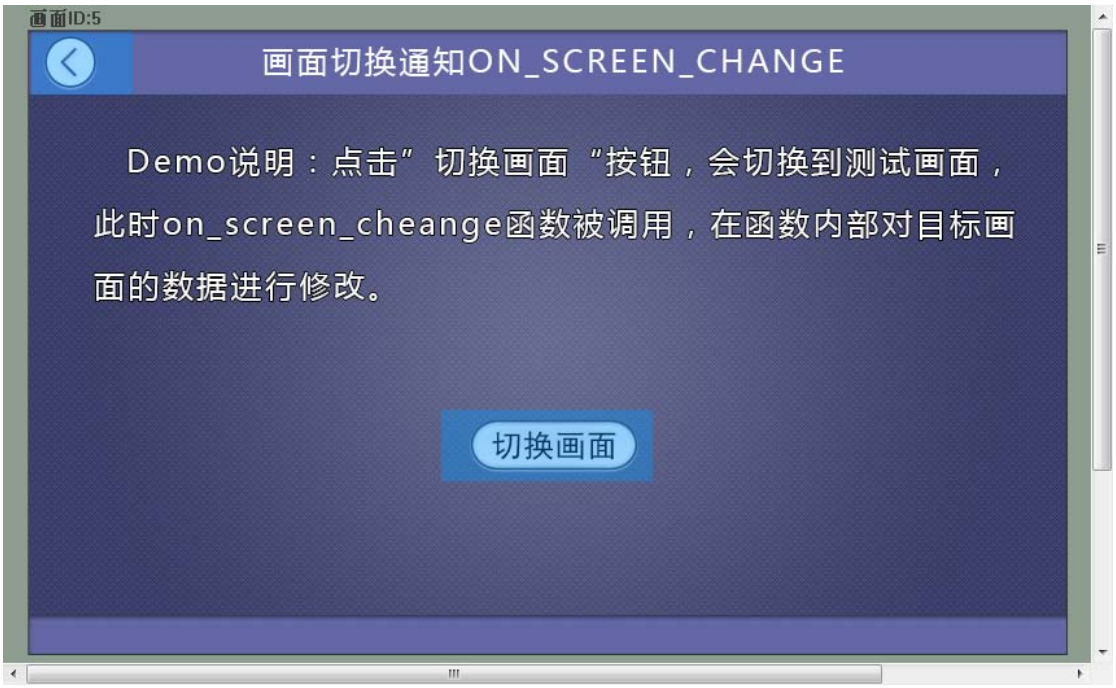


图 5-15 函数 on_screen_change

图 5-16 目标画面中的数据

2. 教程中在 on_screen_change 函数中添加了对目标画面数据修改的逻辑处理，屏在切换画面的同时会触发回调函数 on_screen_change，从而实现在切换画面是将目标画面的数据进行修改，具体代码如程序清单 5 中所示。

程序清单 5 函数 on_screen_change

```
--[[*********************************************************************************
** Function name:   on_screen_change
** Descriptions:    当画面切换至目标画面 ID 时，执行此回调函数
                    注意：回调函数的参数和函数名固定不能修改
** Input value :    screen   目标画面 ID
*********************************************************************************--]]
function on_screen_change(screen)
    if screen == 8 then                                  --切换到画面 9 时，修改控件 1，2 的数据
        set_value(8,1,9)                                 --修改控件 1 的数据
        set_value(8,2,30)                                --修改控件 2 的数据
    end
end
```

工程编译请参考上一小节 5.3.1 的第 2 步；

3. 点击运行串口屏后切换到当前画面，点击画面中切换画面的按钮后切换到目标画面中，如所示；

图 5-17 虚拟屏运行结果

5.3.6 函数 on_draw()

1. 屏重绘时执行此回调函数，通常所有绘图操作都在此函数中实现。显示工程中的图片也是在此回调函数中实现，教程实现点击按钮显示随工程编译下载的图片，教程演示画面如图 5-18 所示

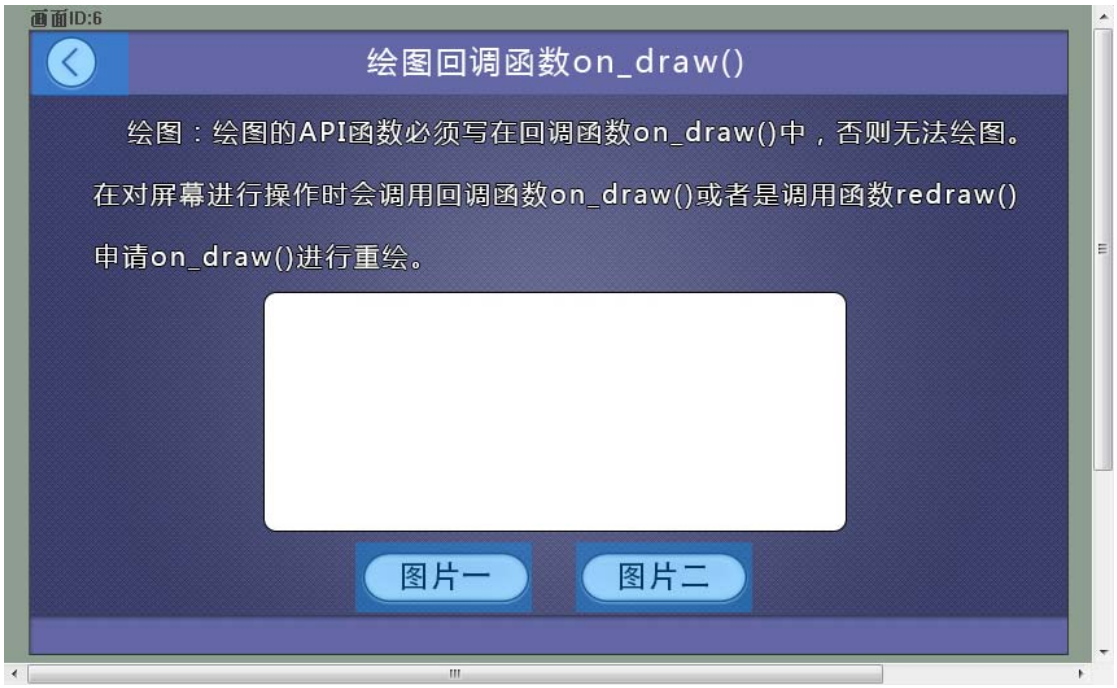


图 5-18 函数 on_draw

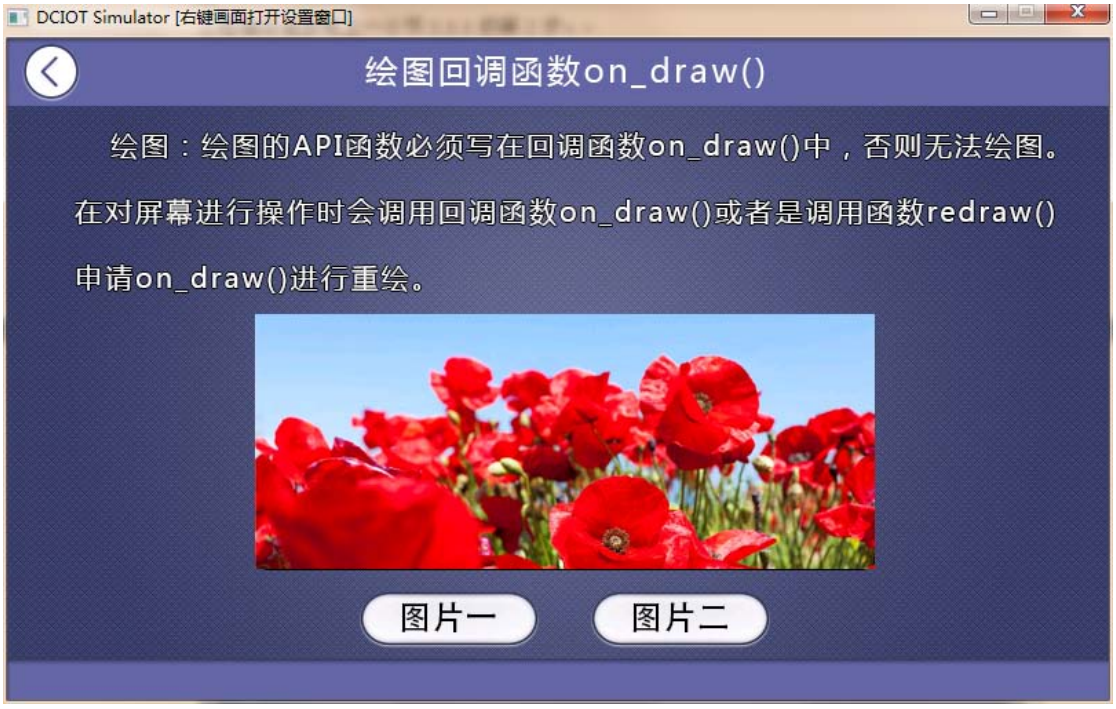
2. 教程中的显示图片的代码如程序清单 6 中所示

程序清单 6 函数 on_draw

```
--[[**********************************************************************
** Function name:   on_draw
** Descriptions:    画面刷新时，或者使用 API 函数 redraw 申请重绘，执行此回调函数
                    注意：回调函数的参数和函数名固定不能修改
** Input value :    screen   目标画面 ID
**********************************************************************--]]
function   on_draw(screen)
    --**************************************************************************
 --调用函数：draw_image(image_id,frame_id,dstx,dsty,width,height,srcx,srcy)
--函数功能：根据图片 ID 绘图
 --参数：     image_id 图片资源的 ID
 --frame_id 对应图标， 可以设置帧 ID，其他图片固定为 0
 --dstx      图片显示 X 坐标
 --dsty      图片显示 Y 坐标
 --width        图片显示宽度
 --height       图片显示高度
 --srcx      图片裁剪 X 坐标
 --srcy      图片裁剪 Y 坐标
 --**************************************************************************
    if   screen == 6 and   show_picture   == 1 then
        draw_image(23,0,190,160,430,230,0,0)
    end
    if screen == 6 and show_picture == 2 then
        draw_image(24,0,190,160,430,230,0,0)
    end
end
function   on_control_notify(screen,control,value)
    if   screen == 5   then
        if   control == 1 and value == 1   then                    --绘制图片 1
            show_picture = 1
            redraw ( )                                             --申请重绘
        end
        if control == 2 and value == 1   then                      --绘制图片 2
            show_picture = 2
            redraw(   )
        end
    end
end
```

工程编译请参考上一小节 5.3.1 的第 2 步；

3. 点击运行串口屏后切换到当前画面，点击画面中运行按钮后在当前画面中显示函数运行结果，如所示；



5.3.7 U 盘和 SD 卡的回调函数

1. 当插入 U 盘和 SD 卡或者拔出 U 盘或 SD 卡时，便会触发对应的回调函数。教程中编写了当 U 盘或 SD 卡插入时将系统分配的路径在文本框中显示出来，如果拔出则在文本框中提示已拔出的程序。教程演示画面如图 5-19 所示

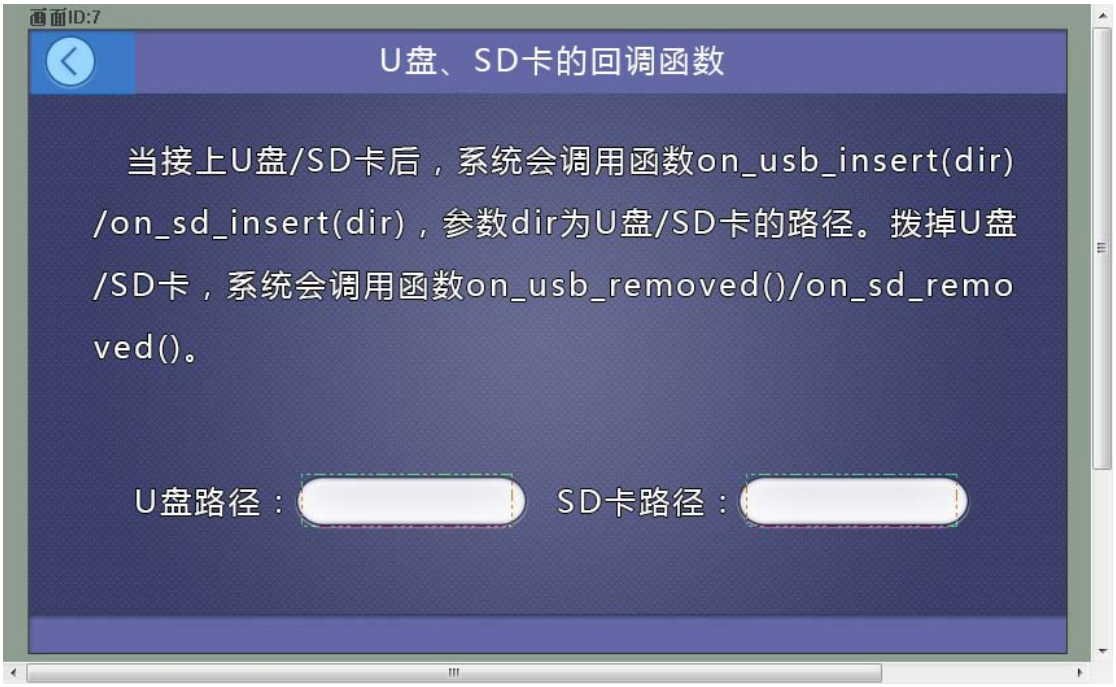


图 5-19 U 盘/SD 卡回调函数演示画面

2. 教程中实现显示 U 盘/SD 卡插入及拔出操作提示的 LUA 脚本代码如程序清单 7 中所示

程序清单 7 U 盘或 SD 卡回调函数

```
--[[*********************************************************************************
** Function name:    on_usb_inserted
** Descriptions:     插入 U 盘后，执行此回调函数
                     注意：回调函数的参数和函数名固定不能修改
** Input value :     dir    U 盘的路径
*********************************************************************************--]]
function    on_usb_inserted(dir)
          set_text(7,1,dir)                                    --显示 U 盘路径
end
--[[*********************************************************************************
** Function name:    on_sd_inserted
** Descriptions:     插入 SD 卡后，执行此回调函数
                     注意：回调函数的参数和函数名固定不能修改
** Input value :     dir    SD 卡的路径
*********************************************************************************--]]
function    on_sd_inserted(dir)
     set_text(7,2,dir)                                     --显示 SD 卡路径
end
--[[*********************************************************************************
** Function name:    on_sd_inserted
** Descriptions:     拔出 U 盘后，执行此回调函数
                     注意：回调函数的参数和函数名固定不能修改
*********************************************************************************--]]
function    on_usb_removed()
    set_text(7,1,"已拔出 U 盘")
end
--[[*********************************************************************************
** Function name:    on_sd_inserted
** Descriptions:     拔出 SD 卡后，执行此回调函数
                     注意：回调函数的参数和函数名固定不能修改
*********************************************************************************--]]
function    on_sd_removed()
     set_text(7,2,"已拔出 SD 卡")
end
```

工程编译请参考上一小节 5.3.1 的第 2 步

3. 该回调函数只能在实体屏中使用，虚拟屏中无法使用，下载 LUA 程序和工程请参考；

5.4 下载工程

在我司的上层软件 Visual TFT 中集成了 LUA 程序的编译器，可以实现在编译工程的同时将 LUA 脚本程序一起编译，并且将编译后的图片和程序集合在一个名为 DCIOT.PKG 的文件中。编译后只需要把 DCIOT.PKG 文件拷贝到 U 盘中，接上串口屏并重新上电即可将图

片和程序下载到屏中。

5.4.1 操作过程

工程编译成功后在输出窗口会提示编译成功，如所示。编译成功后打开工程目录，找到 output 文件夹，将文件夹中的 DCIOT.PKG 文件拷贝到 U 盘中，如图 5-20 和图 5-21 所示；接上串口屏重新上电，等到提示烧录工程成功后，拔掉 U 盘重新上电即可。

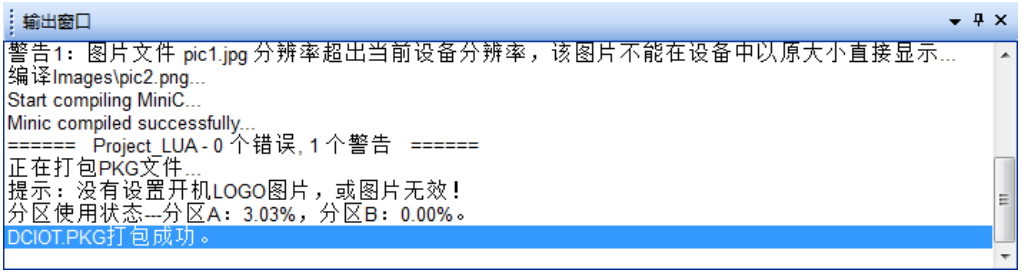


图 5-20 编译成功

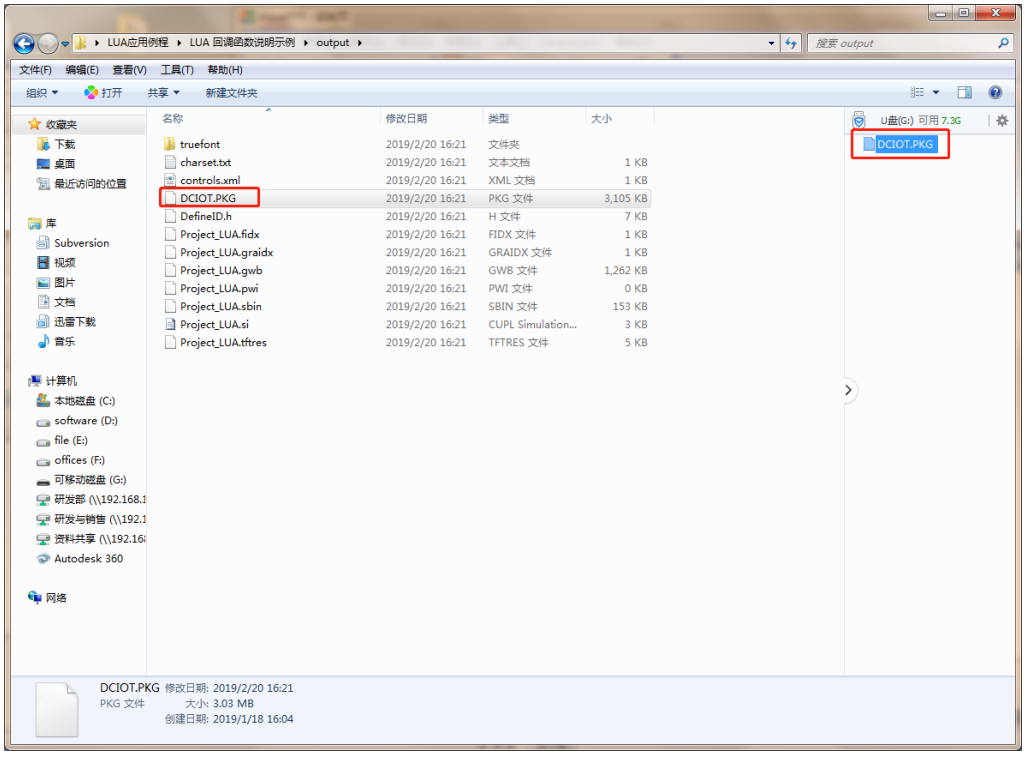


图 5-21 拷贝

6. LUA 回调函数的完整程序清单

LUA 教程-回调函数使用说明 demo 的完整程序，如程序清单 8 所示，查看程序时请结合教程 demo 理解：

程序清单 8 完整程序

```
--[[*****************************Copyright(c)*********************************
**                      Guangzhou DACAIGAUNGDIAN Technology Co. Ltd
**
**                                          http://www.gz-dc.com
**lua 教程网站;                             http://www.runoob.com/lua/lua-arrays.html
**------File Info-----------------------------------------------------------
** File Name:                main.lua
** Latest modified Date:     2019/1/18
** Description:              常用的回调函数的使用说明
**
**--------------------------------------------------------------------------
** Created By:                    林青田
** Created date:             2019/1/18
** Version:                       V1.0
** Descriptions:             常用回调函数的使用说明
******************************************************************************--]]
local sys_timer         = 0                                     --系统定时器计数器
local timer_out         = 0                                     --定时器超时计数器
local show_picture      = 0                                     --画图图片编号
--[[*****************************************************************************
** Function name:    on_init
** Descriptions:     系统初始化时，执行此回调函数。
                     注意：回调函数的参数和函数名固定不能修改
******************************************************************************--]]
function on_init()
        set_value(1,1,10)                                       --设置文本控件的值
        set_value(1,2,2)                                        --设置文本控件的值
end
--[[*****************************************************************************
** Function name:    on_systick
** Descriptions:     定时回调函数，系统每隔 1 秒钟自动调用。
                     注意：回调函数的参数和函数名固定不能修改
******************************************************************************--]]
function on_systick()
       sys_timer = sys_timer +1
      set_value(2,1,sys_timer)
end
--[[*****************************************************************************
```

```
** Function name:    on_timer
** Descriptions:     定时器超时，执行此回调函数
                      注意：回调函数的参数和函数名固定不能修改
** Input value :     timer_id  定时超时的定时器 ID 号，定时器编号 0~31
*****************************************************************************--]]
function on_timer(timer_id)
    if    timer_id == 0 then                                          --定时器 0
          timer_out = timer_out +1
          set_value(3,3,timer_out)
    end
end

--[[*****************************************************************************
** Function name:    on_control_notify
** Descriptions:     用户通过触摸修改控件后，执行此回调函数。
                      点击按钮控件，修改文本控件、修改滑动条都会触发此事件。
                      注意：回调函数的参数和函数名固定不能修改
** Input value :     screen    画面 ID
                      control  控件 ID
                      value     控件值(包括文本控件输入的值)
*****************************************************************************--]]
function on_control_notify(screen,control,value)
 --*************************************************************
 --功能：按下画面 4 的控件 1 启动定时器 0
 --调用函数：start_timer(timer_id, timeout, countdown, repeat)
    --函数功能：启动定时器
 --参数：       timer_id，  定时器 ID；
 --            timeout，    超时时间；
 --              countdown，1 顺计时，0 倒计时
 --              repeat       计时次数，0 无限循环
 --*************************************************************
    if screen == 3 and control == 1 and value == 1 then
          start_timer(0,1000,1,0)
    end
    if screen == 3 and control == 2 and value == 1 then
          stop_timer(0)                                          --停止定时时器 0 工作
          timer_out = 0
   end
  if screen == 4    then
       if control == 1 and value == 1 then                      --value == 1  按钮按下
            set_text(4,7,"按钮按下")
      elseif control == 1 and value == 0 then                   --value == 0  按钮弹起
            set_text(4,7,"按钮弹起")
      elseif control == 2     then                              --文本输入
```

```
            set_text(4,7,"文本输入")
        elseif control == 3     then                        --图标控件
            set_text(4,7,"触摸图标")
        elseif control == 4    then                         --滑动选择控件
            set_text(4,7,"滑动选择")
        elseif control == 5     then                        --菜单控件
            set_text(4,7,"菜单选择")
        elseif control == 6     then                        --滑块控件
            set_text(4,7,"滑块滑动")
      end
   end

   if screen == 5 then
       if control == 1 and value == 1    then               --绘制图片 1
            show_picture = 1
            redraw()                                        --申请重绘
       end
       if control == 2 and value == 1    then              --绘制图片 2
            show_picture = 2
            redraw()
     end
    end
end


--[[*********************************************************************
** Function name:    on_screen_change
** Descriptions:     当画面切换至目标画面 ID 时，执行此回调函数
                     注意：回调函数的参数和函数名固定不能修改
** Input value :     screen    目标画面 ID
*********************************************************************--]]
function on_screen_change(screen)
     if    screen == 8 then                                 --切换到画面 9 时，修改控件 1，2 的数据
           set_value(8,1,9)                                 --修改控件 1 的数据
           set_value(8,2,30)                                --修改控件 2 的数据
       end
end


--[[*********************************************************************
** Function name:    on_draw
** Descriptions:     画面刷新时，或者使用 API 函数  redraw  申请重绘，执行此回调函数
                     注意：回调函数的参数和函数名固定不能修改
** Input value :     screen    目标画面 ID
```

```
**********************************************************************--]]
function on_draw(screen)
 --**********************************************************************
 --调用函数：draw_image(image_id,frame_id,dstx,dsty,width,height,srcx,srcy)
 --函数功能：根据图片 ID 绘图
 --参数：      image_id  图片资源的  ID
 --frame_id  对应图标，  可以设置帧  ID，其他图片固定为  0
 --dstx       图片显示  X  坐标
 --dsty       图片显示  Y  坐标
 --width          图片显示宽度
 --height        图片显示高度
 --srcx       图片裁剪  X  坐标
 --srcy       图片裁剪  Y  坐标
 --**********************************************************************
      if screen == 6 and show_picture == 1 then
            draw_image(23,0,190,160,430,230,0,0)
      end
      if screen == 6 and show_picture == 2 then
           draw_image(24,0,190,160,430,230,0,0)
     end
end
--[[**********************************************************************
** Function name:   on_usb_inserted
** Descriptions:    插入 U 盘后，执行此回调函数
                      注意：回调函数的参数和函数名固定不能修改
** Input value :    dir   U 盘的路径
**********************************************************************--]]
function on_usb_inserted(dir)
          set_text(7,1,dir)                                    --显示 U 盘路径
end
--[[**********************************************************************
** Function name:   on_sd_inserted
** Descriptions:    插入 SD 卡后，执行此回调函数
                      注意：回调函数的参数和函数名固定不能修改
** Input value :    dir   SD 卡的路径
**********************************************************************--]]
function on_sd_inserted(dir)
     set_text(7,2,dir)                                     --显示 SD 卡路径
end
--[[**********************************************************************
** Function name:   on_sd_inserted
** Descriptions:    拔出 U 盘后，执行此回调函数
                      注意：回调函数的参数和函数名固定不能修改
**********************************************************************--]]
```

```
function on_usb_removed()
        set_text(7,1,"已拔出 U 盘")
end
--[[*********************************************************************************
** Function name:    on_sd_inserted
** Descriptions:     拔出 SD 卡后，执行此回调函数
                     注意：回调函数的参数和函数名固定不能修改
*********************************************************************************--]]
function on_sd_removed()
    set_text(7,2,"已拔出 SD 卡")
end
--[[*********************************************************************************
END FILE
*********************************************************************************--]]
```

7. 免责声明

广州大彩光电科技有限公司所提供的所有服务内容旨在协助客户加速产品的研发进度，在服务过程中或者其他渠道所提供的任何例程程序、技术文档、CAD 图等资料和信息，都仅供参考，客户有权不使用或自行参考修改，本公司不提供任何的完整性、可靠性等保证，若在客户使用过程中因任何原因造成的特别的、偶然的或间接的损失，本公司不承担任何责任。